Learning Probabilistic and Causal Models with(out) Imperfect Advice

Choo XianJun, Davin

B.Sc. (First Class Hons), Computer Science, National University of Singapore
B.Sc. (First Class Hons), Applied Mathematics, National University of Singapore
M.Sc., Computer Science, ETH Zürich

2025

A thesis submitted for the degree of
Doctor of Philosophy in Computer Science

School of Computing
National University of Singapore

Research Advisor:
Associate Professor Arnab Bhattacharyya

Thesis Advisor:
Professor Seth Lewis Gilbert

Examiners:
Associate Professor Jonathan Mark Scarlett
Assistant Professor Warut Suksompong

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis. This thesis has also not been submitted for any degree in any university previously.

Name: Choo XianJun, Davin

Date: 14 January 2025

This research/project is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG-PhD/2021-08-013). The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the Singapore government, or any other entity.

Abstract

From early education, we've learned a general problem-solving approach: abstract real-world problems into models, solve them, and map the solutions back to reality. While effective, this "Abstract, Solve, Map back" framework can sometimes overlook crucial contextual details in real-world instances. This thesis addresses key questions in learning probabilistic and causal models, and introduces algorithmic innovations to incorporate contextual information as imperfect advice. The contributions are organized into three themes: (I) Algorithms for learning probabilistic models, (II) Algorithms for learning causal models, and (III) Algorithms with imperfect advice.

Theme (I) explores finite-sample distribution learning under the PAC framework. We design algorithms for degree-bounded Bayesian networks, extending existing results for linear Gaussian models and offering new insights for discrete polytrees. A key insight for this theme is that structural sparsity enables more sample-efficient learning.

Theme (II) focuses on causal inference, specifically on the problems of causal graph discovery via adaptive interventions and causal effect estimation. For discovery, we provide the first full characterization for identifying causal graphs with interventions and propose adaptive algorithms for various settings. For estimation, we derive PAC bounds for covariate adjustments and develop algorithms to find small adjustment sets. The key idea is that reframing certain causal problems as graph learning or distribution learning tasks allows us to leverage tools from graph theory and statistics respectively.

Theme (III) studies algorithms that utilize instance-specific side-information, explored through the framework of learning-augmented algorithms. Inspired by the property testing insight that "testing can be cheaper than learning", we introduce the TestAndAct framework for using imperfect advice. This approach incorporates a test to assess the quality of the advice and adapts the algorithm's behavior accordingly. We instantiate variants of this idea to improve competitive ratios in online bipartite matching under random arrivals, reduce sample complexity in learning multivariate Gaussians, and minimize interventions for learning causal graphs. Our methods provide performance guarantees that scales with the quality of the advice, without requiring prior knowledge of its accuracy.

Acknowledgements

This thesis would not be possible without the assistance and support of many people in my life. Thank you everyone!

First and foremost, I would like to express my deepest gratitude to my (co-)advisors, Arnab Bhattacharyya and Seth Gilbert, for their invaluable guidance, unwavering support, and endless patience throughout my research journey. I have learned so much from both of you — not only about how to approach research, but also how to think critically and creatively about complex problems.

> **To Arnab:** Thank you for always believing in me and for being there to help me grow as a researcher. I am deeply grateful for the opportunities to visit the Simons Institute for the Theory of Computing and the Empirical Inference group of Bernhard Schölkopf at the Max Planck Institute for Intelligent Systems. These experiences broadened my research horizons and introduced me to exciting new areas of inquiry. In fact, everything I know about causality began with my visit to the Simons Institute in my first year of the Ph.D. program. Despite my initial unfamiliarity with the field, you consistently showed enthusiasm for every small progress I made, making research an enjoyable experience. You are one of the nicest and most creative people I have worked with, and I feel fortunate to have you as my advisor.

> **To Seth:** Thank you for being my co-advisor. Without a doubt, my journey into theory and algorithms began in my freshman year when I took the CS2020 module (Data Structures and Algorithms Accelerated) that you taught. Till this day, I still remember the quote you shared with us on the first day of class (paraphrased, and I might have the exact numbers wrong, but the takeaway is clear): "If you want a 10x speedup, hire better programmers. If you want a 100x speedup, design a better algorithm." I came away from the course excited to learn more and contribute, in my own small way, to the design of algorithms with provable guarantees. I was also fortunate to have you as my undergraduate thesis advisor. Through the years, your kindness, patience, and valuable insights about life have had a significant impact on me.

I would also like to extend my sincere thanks to the members of my thesis committee and chair (Jonathan Scarlett, Warut Suksompong, and Prashant Vasudevan) for their valuable feedback and insightful suggestions, which have greatly enhanced the quality of this thesis.

All the work I accomplished during my Ph.D. would not have been possible without my incredible collaborators. In addition to some of the names mentioned above, I've had the privilege of working with and learning from some of the most humble and brilliant minds in the world, including esteemed faculty members like Caroline Uhler, Clément Canonne, Constantinos Daskalakis, David Sontag, Hady Lauw, and Simina Brânzei; newly minted faculty members such as Billy Jin, Chun Kai Ling, Sutanu Gayen, Themistoklis Gouleakis, and Yuval Dagan; as well as talented colleagues like Anthimos-Vardis Kandiros, Chandler Squires, Dimitrios Myrisiotis, Jian Zhang, Jia Peng Lim, Joy Qiping Yang, Kirankumar Shiraguar, Nicholas Recker, Nicholas Teh, Philips George John, Raghavendra Addanki, Rishikesh Gajjala, Yan Hao Ling, Yongho Shin, and Yuhao Wang. It has been a wonderful experience collaborating with and learning from each of you. In particular, I would like to thank Kiran for offering valuable advice on research, presentation, and life in general.

Thank you to all my friends for sharing the journey with me so far and for being in my life: Chris, Chun Kai, Desmond, and Shawn for the nonsensical musings about graduate school; Chun Mun, Kah Hou, Keng Kiat, Yanxian, and Yik Jiun for being there for me when I need another opinion; Benson, Justin, Juan-Ting, and Teng Foong for reminiscing about our shared experiences in ETH Zürich. Meanwhile, most of my social interactions at NUS SoC have revolved around the members of the Bryan Low's GLOW.AI group and the (unofficial) AlgoTheory group, which includes amazing people like Aditya, Ayaz, Dimitrios, Eldon, Esty, Jiaqi, Joy, Kushagra, Mathews, Naganand, Philips, Pranjal, Sanjana, Sayantan, Sutanu, Themis, Vijeth, Vinh, Vipul, Yan Hao, Yuhao, and Zeyong. A special shout-out to my frequent lunch buddies, Ayaz, Joy, Kushagra, and Yuhao! I'll miss our lunches where we'd chat about everything under the sun — from new research ideas and the latest games we played over the weekend to retirement plans and random trivia facts. I'm truly glad to have been part of such a diverse and active group during my Ph.D. Thank you all for the camaraderie, the insightful discussions, social hangouts at conferences, and the entertaining WhatsApp, Telegram, and Slack chats!

I would also like to acknowledge the financial support from AI Singapore for funding my Ph.D. The support was indispensable and enabled me to focus my attention on research.

Lastly, and most importantly, I want to thank my family for their unconditional love, patience, and support. I feel incredibly fortunate to have all of you by my side — your presence has made this Ph.D. journey not only possible but also more enjoyable and less stressful than I ever could have imagined. To my dearest wife, you make me a better person every day, and I can't imagine anyone more wonderful to share my life with. Thank you for always believing in me, encouraging me to follow my dreams, and keeping me grounded when things felt overwhelming. I'm so lucky to have you in my life!

Contents

Chapter 1

Overview of thesis

> "The important thing is not to stop questioning. Curiosity has its own reason for existence. One cannot help but be in awe when he contemplates the mysteries of eternity, of life, of the marvelous structure of reality. It is enough if one tries merely to comprehend a little of this mystery each day."
>
> - Albert Einstein in *Death of a Genius* [Mil55]

Since our early education, we have been taught a very general problem solving framework that helps us tackle complex real-world issues: model real world problems into their clean abstract counterparts which we can solve, then map back the solutions to the real world instances; see Fig. 1.1. For example, let us consider the toy problem of finding a word in a dictionary. This can be modelled as searching for an element in a sorted array of length n. Then, we see that we can our problem by simulating binary search by querying $\Theta(\log n)$ pages in the dictionary. This paradigm of "Abstract, Solve, and Map back" is highly effective because it allows us to simplify and address complex, nuanced problems by reducing them to well-understood problems with established solutions. This is why educational systems emphasize teaching methods for solving general problems, equipping us with tools to tackle a wide range of real-world challenges through effective modeling.

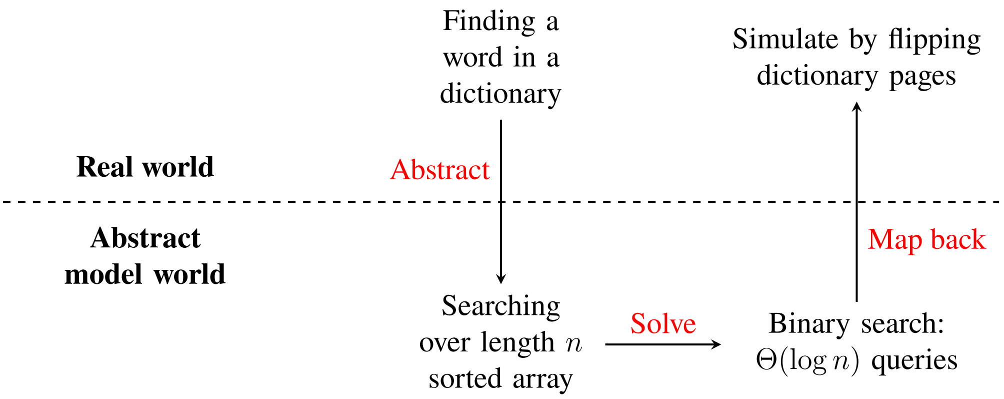

Finding a word in a dictionary
Simulate by flipping dictionary pages
Real world
Abstract
Abstract model world
Map back
Searching over length n sorted array
Solve
Binary search: $\Theta(\log n)$ queries

Figure 1.1: The general "Abstract, Solve, Map back" problem solving framework

To address the complexities of real-world phenomena, two widely studied abstract models have proven effective: probabilistic models for prediction tasks and causal models for understanding the effects of interventions on systems.

Probabilistic models are built on the foundation of probability theory, which provides a mathematical framework to handle uncertainty. These models are particularly useful when we aim to capture the variability in observed data or make predictions in the presence of incomplete information. By representing the world through random variables and their distributions, probabilistic models allow us to quantify uncertainty, compute the likelihood of various outcomes, and infer hidden structures from data. Common examples include Bayesian networks [Pea88], Markov chains [Nor97], Hidden Markov Models (HMMs) [RJ86], and Gaussian Mixture Models (GMMs) [Rey15]. Based on the characteristics of the problem of interest, certain models may be more useful than others.

In contrast, causal models are based on the theory of causal inference [Rub74, SN90, Sek09, Pea09a] and seek to move beyond simple correlations, aiming to capture the underlying mechanisms that drive relationships between variables of interest. It is well-known that causal thinking requires one to go beyond statistical inference as there are fundamentally unanswerable questions given only observational data such as "how does the distribution change if I intervene on a particular variable in a particular way?" (interventional question) or "given that the patient died, would he/she have lived if we administered a different treatment?" (counterfactual question). [PM18, BCII22] provide excellent exposition and examples on this separation phenomenon through the "causal hierarchy".

Setting aside modeling concerns, the generality of the "Abstract, Solve, Map back" paradigm (Fig. 1.1) often means we overlook instance-specific details that could potentially improve our problem-solving approach. In the dictionary example, we could have utilized additional information such as letter frequency tables, sought predictions from a machine learning model such as ChatGPT, or consulted a friend who recently looked up a nearby word. For instance, if we are searching for the word "Heuristic", we might disregard a prefix of the dictionary proportional to the words starting with letters A to G. Alternatively, if a friend found the word "Happy" on page 400, then they might suggest us to search in that vicinity. In fact, we can principally account for such an advice and design an algorithm[1] which provably uses $\mathcal{O}(\log |x - x^*|)$ queries, where x^* is the true page which the element we are interested in lies in and $|x - x^*|$ measures the advice error; see Fig. 1.2. Observe that this reverts to $\mathcal{O}(\log n)$ in the worst case but we could conceivably use much less queries when the advice x is of high quality, i.e. when x and x^* are close.

This thesis tackles some of the fundamental and basic questions regarding the learning of probabilistic and causal models, as well as exploring the idea of principally incorporating imperfect advice in different problems. Throughout my Ph.D., I have made contributions to various settings in this problem space, some of which are highlighted via the Venn diagram

[1] See https://en.wikipedia.org/wiki/Learning_augmented_algorithm#Binary_search

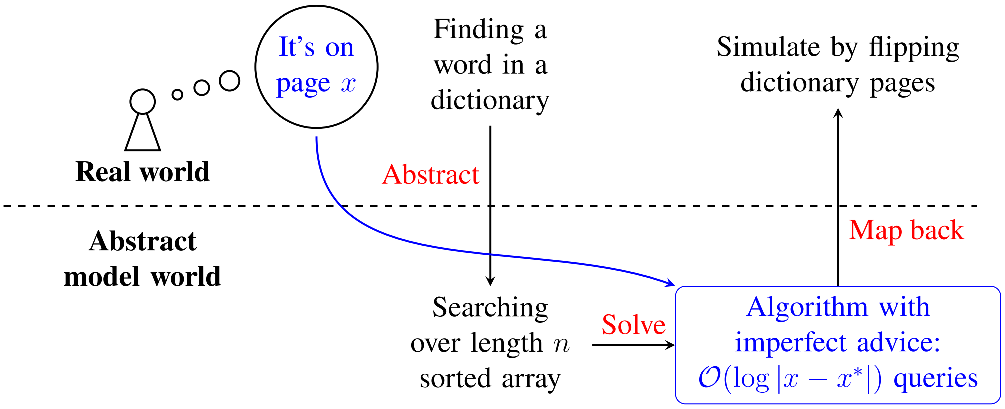
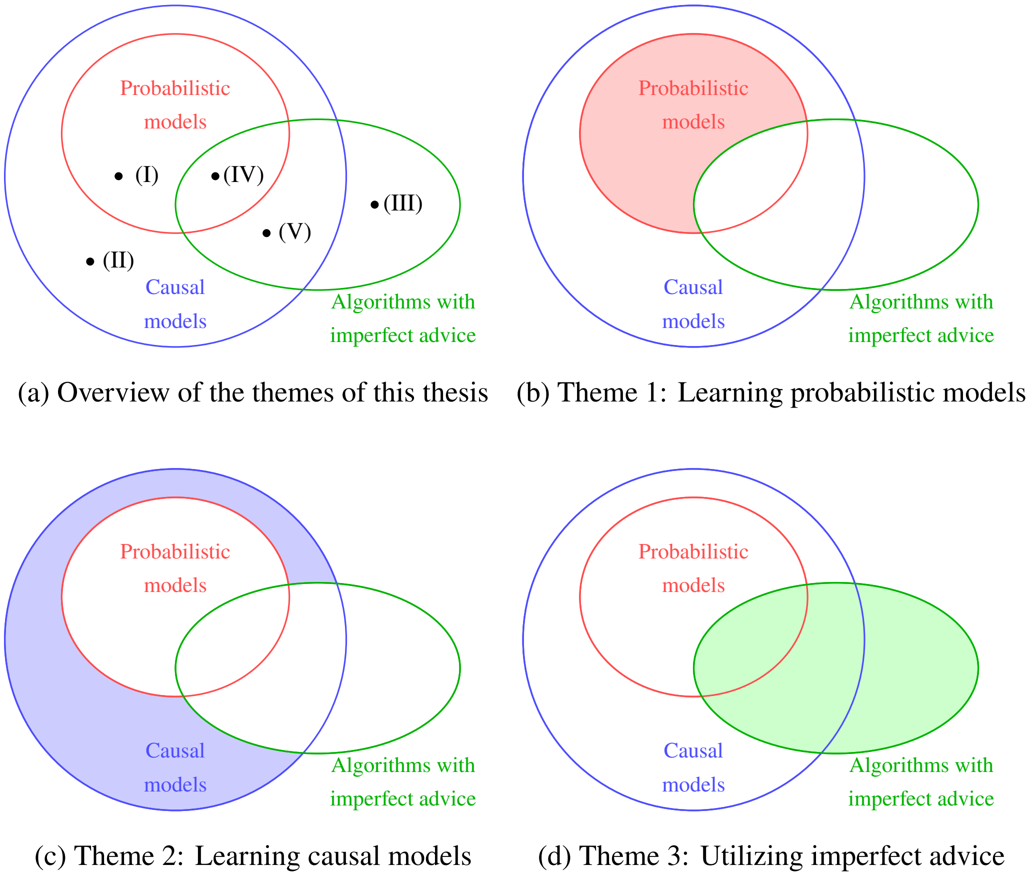

It's on page x
Real world
Finding a word in a dictionary
Simulate by flipping dictionary pages
Abstract
Map back
Abstract model world
Searching over length n sorted array
Solve
Algorithm with imperfect advice: $\mathcal{O}(\log |x - x^*|)$ queries

Figure 1.2: Extending the general "Abstract, Solve, Map back" framework by designing a new algorithm that can principally exploit an advice in the dictionary example

in Fig. 1.3: (I) [BCG+22, DDKC23, CYBC24, BCGM25]; (II) [CSB22, CS23c, CS23b, CS23a, CSU24, CSBS25]; (III) [CGLB24, CL24]; (IV) [BCGJG24]; (V) [CGB23].



(a) Overview of the themes of this thesis

(b) Theme 1: Learning probabilistic models

(c) Theme 2: Learning causal models

(d) Theme 3: Utilizing imperfect advice

Figure 1.3: The three themes covered in this thesis

In the rest of this chapter, we provide high level introductions to each theme explored in this thesis and discuss some of the contributions made within the respective themes.

1.1 Theme 1: Learning probabilistic models

Classic results in statistics show asymptotic convergence of estimators in the limit of large data. A natural question is how well do such methods work under finite sample scenarios in the real-world settings. We consider this from the viewpoint of distribution learning [KMR+94] under the Probably Approximately Correct (PAC) learning model [Val84]. Given sample access to an unknown underlying distribution $\mathcal{P}$, the goal here is to learn a distribution $\widehat{\mathcal{P}}$ that is close (in total variational distance) to the ground-truth distribution $\mathcal{P}$, using an efficient algorithm, i.e. $\mathrm{d}_{\mathrm{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ with success probability at least $1 - \delta$. Here, pointwise convergence of the distributional parameters is no longer a requirement and the aim is rather to approximately learn the induced distribution. This latter relaxed objective may be achievable when the former may not be (e.g. for ill-conditioned systems) and can be the more relevant requirement for downstream inference tasks.

This part of the thesis focuses on learning Bayesian networks, a type of probabilistic graphical model used to model beliefs in a wide variety of domains, e.g. see [JN07, KF09, Pea09b] and references therein. A key insight underlying my results is that one can design more sample efficient procedures when the variables have structural relations that admit a sparse Bayesian network representation. Sparsity is a common and very useful assumption for statistical learning problems; see [HTW15] for an overview of the role of sparsity in regression. In particular, we study sparsity from the popular lens of bounded in-degree [Das97, BCD20] as it naturally reflects the belief that each variable is affected by a small number of variables in a correct model. In Chapter 3, we study this problem in the setting of linear Gaussian Bayesian networks over n variables [BCG+22]. While $\Theta(n^2/\varepsilon^2)$ samples are known to be necessary and sufficient (e.g. see [ABDH+20, Appendix C]), we generalized this sample complexity bound to $\widetilde{\Theta}(nd/\varepsilon^2)$ samples when we are guaranteed that $\mathcal{P}$ is described by a known Bayesian network with maximum in-degree d. Meanwhile, in Chapter 4, we establish finite-sample guarantees for efficient proper learning of discrete distributions described by bounded-degree polytrees, a subclass of Bayesian networks where the graph is a tree if we ignore the edge directions [CYBC24]. Prior to our work, the only known results for learning polytree with finite sample guarantees is for 1-polytrees [BGP+23, DP21] where the Bayesian network has maximum in-degree of 1. We generalized this by designing a PAC-learner algorithm that recovers d-polytrees under certain assumptions.

1.2 Theme 2: Learning causal models

Understanding the world and the impact of algorithms through a causal lens is becoming increasingly important as automated techniques are being operationalized more widely. A central problem in causality is to distinguish causes from effects in large environments and learning causal relationships from data is an important problem with applications in many areas such as medicine, biology, genetics, econometrics, and philosophy [Hoo90, Rei91, KWJ$^+$04, Woo05, RW06, ES07, SC17, RHT$^+$17, POS$^+$18]. There has also been a growing interest in the machine learning community to use causal inference techniques to improve generalizability to novel testing environments, e.g. see [LKC17, Sch22]. For example, causal inference techniques have been used to design methods that generalize to out-of-distribution samples [GUA$^+$16, ABGLP19, Arj20]. Under the assumption that there are no latent confounders, a common representation for causal models is via directed acyclic graphs (DAGs) subject to causal operations such as do-calculus [Pea09b]; more complicated graphical notations exist when there are hidden variables in the causal system.

In this part of the thesis, we focus on acyclic causal models primarily because of their simplicity and interpretability as DAGs provide a natural and unambiguous framework for modeling cause-and-effect relationships. More complicated causal models such as cyclic causal models [BFPM21] and mixture of causal DAGs [SPU20] have been explored though research in these areas is still relatively nascent. Under the acyclic causal model setting, we study two fundamental problems in causal inference: structure learning and causal effect estimation. The former aims to recover a graph $\mathcal{G}^*$ which is causally representative of the underlying distribution while the latter aims to estimate the effect of a variable given an intervention on another.

Causal structure learning. In general, observational data can only recover the causal DAG $\mathcal{G}^*$ up to an equivalence class [Pea09b, SGS00]. Hence, if one wants to avoid making parametric assumptions about the causal mechanisms, the only recourse is to obtain experimental data from interventions [EGS05, EGS06, Ebe10]. In practice, interventions may correspond to randomized controlled trials or performing certain gene knockout operations. Under some standard causal assumptions, we can abstract this as a graph learning problem with specialized causal graph manipulation operations. In Chapter 6, we gave a complete characterization for a set of adaptive interventions that can correctly identify $\mathcal{G}^*$ amongst its observational equivalence class, and also improved the (then) state-of-the-art search algorithm running as much as 10x faster in certain settings whilst having stronger theoretical guarantees [CSB22]. Our characterization also recovers several existing results in causal graph discovery in a clean unified perspective. Prior to this work, only approximation bounds on the size of the interventional set were known, which impeded the development of algorithms with provable guarantees. We have also extended line of work to other settings to address practical concerns and natural questions such as "what if we only care about the relations between a subset of variables?" [CS23c], "can we optimize for total cost incurred if interventions have varying node-specific costs?" [CS23b], "what is the optimal trade-off between adaptivity and total number of interventions required?" [CS23a], and "what if interventions have off-target effects?" [CSU24].

Causal effect estimation. Here, one wishes to estimate the interventional distribution of Y when X is set to x, which can be written as $\mathcal{P}_x(y)$ notationally. Accurate estimates of such causal effects play a key role in decision-making across applications such as healthcare, economics, and operations. The solution to this problem is traditionally conceptualized as a two-step process: first estimate a graph $\widehat{\mathcal{G}}$, then apply closed-form graphical criteria to $\widehat{\mathcal{G}}$. The first step is known as causal discovery (a.k.a. structure learning, what we addressed in the paragraph above) and the second step is known as causal identifiability where one has to output an expression of an interventional query $\mathcal{P}_x(y)$ given a causal graph $\mathcal{G}^*$, or correctly determine that there exists no such expression for some distribution represented by $\mathcal{G}^*$. Each of these two steps are independent topics of intense research and the second step typically treats the first as a black-box tool and fully trusts its output. Unfortunately, such a two-step approach is suboptimal for estimating $\mathcal{P}_x(y)$. Firstly, to correctly learn the causal graph $\mathcal{G}^*$, one may need strong assumptions on the underlying distribution $\mathcal{P}$. Secondly, the graphical characterization results for causal identifiability do not apply to an erroneous graph, but a huge number of samples may be needed to correctly learn the causal graph $\mathcal{G}^*$ since it is difficult to tell whether an edge is actually missing or is just very "weak". In the absence of randomized experiments, a common approach to estimating causal effects uses *covariate adjustment*. In Chapter 7, we study covariate adjustment for discrete distributions from the PAC learning perspective, assuming knowledge of a valid adjustment set $\boldsymbol{Z}$, which might be high-dimensional [CSBS25]. Our first main result PAC-bounds the estimation error of covariate adjustment by a term that is exponential in the size of the adjustment set; it is known that such a dependency is unavoidable even if one only aims to minimize the mean squared error. Motivated by this result, we introduce the notion of an *ε-Markov blanket*, give bounds on the misspecification error of using such a set for covariate adjustment, and provide an algorithm for ε-Markov blanket discovery; our second main result upper bounds the sample complexity of this algorithm. Furthermore, we provide a misspecification error bound and a constraint-based algorithm that allow us to go beyond ε-Markov blankets to even smaller adjustment sets. Our third main result upper bounds the sample complexity of this algorithm, and our final result combines the first three into an overall PAC bound. Altogether, our results highlight that one does not need to perfectly recover causal structure in order to ensure accurate estimates of causal effects.

1.3 Theme 3: Utilizing imperfect advice

Here, we study algorithms that extend the "Abstract, Solve, and Map back" approach, as portrayed in Fig. 1.2, by incorporating useful contextual information about the actual real-world instances that we are solving. Instance-specific advice can come from machine learning predictions or domain experts, and there is no general restriction on the type and on quality assurance of the provided advice. The design and analysis of such methods are often explored under the framework of learning-augmented algorithms[2] [MV22] where the typical performance measures are consistency and robustness which quantify the two extremes of perfect advice and arbitrarily bad advice. Ideally, one would want to design methods with high consistency while having robustness no worse than advice-free baselines. A closely related subfield is that of data-driven algorithms [Bal20, GR20] where one aims to design parameterized algorithms whose parameters are pre-tuned based on some training data or information about the data distribution.

Motivated by the insight from the property testing literature that "testing can be cheaper than learning", this thesis introduces the framework of TestAndAct for using imperfect advice in learning-augmented algorithms. On a high level, we incorporate a suitable test to determine the quality of the given advice and "act suitably" in our algorithmic design. We instantiate variants of this idea to obtain better competitive ratios in online bipartite matching under random arrivals [CGLB24] in Chapter 9, use less samples in learning Gaussians [BGGJ$^+$24] in Chapter 10, and use less interventions to learn causal graphs [CGB23] in Chapter 11. Crucially, our methods provide guarantees that interpolate between the extremes of perfect advice and arbitrarily bad advice depending on the quality of the given advice, *without* knowing its quality as input a priori.

[2] The website https://algorithms-with-predictions.github.io/ tracks recent progress in this research area.

1.4 Roadmap of thesis

The remainder of the thesis is organized around the three major themes discussed above. Given the diversity of these themes, we will provide a contextualized conclusion chapter for each part.

- We begin with a generalized preliminaries in Chapter 2 to introduce some notation, definitions, and basic results that are relevant across the themes.
- Part I consists of two chapters, covering results related to learning probabilistic models. In particular, we focus on designing sample-efficient algorithms that utilize i.i.d. observational samples from an unknown underlying distribution $\mathcal{P}$ and produce an estimated distribution $\widehat{\mathcal{P}}$ which is "close" to $\mathcal{P}$. Chapter 3 studies the setting where $\mathcal{P}$ is described by a Gaussian Bayesian network with unknown parameters and the goal is to compute good estimates for these parameters. Meanwhile, Chapter 4 explores the learning of bounded-degree polytrees, a rich class of high-dimensional probability distributions and a subclass of Bayesian networks.
- Part II consists of two chapters, covering results related to learning causal models. As discussed earlier, there are two fundamental problems in causal inference: structure learning and causal effect estimation. Chapter 6 investigates the former problem using adaptive interventions and characterizes the conditions for a set of interventions to recover an underlying causal structure. On the other hand, Chapter 7 provides PAC-style estimation bounds for the latter problem while relying on a minimal set of causal assumptions.
- Part III consists of three chapters, covering results related to learning using imperfect advice. Chapter 9 explores the use of imperfect advice in the context of the classic online bipartite matching problem and introduces the TestAndAct framework. We then show how to apply this framework to tackle problems related to learning probabilistic (Chapter 10) and causal (Chapter 11) models.

Remark 1.1 (Reading notes). We recommend readers to skim the notations in Section 2.1 and refer back to Chapter 2 as and when needed. Readers are encouraged to read the first 3 sections (i.e. "Introduction", "Our main results", and "Technical overview") to get a high-level appreciation of the contributions of each content chapter. Details and proofs will follow in subsequent sections and the appendices.

To facilitate a more coherent presentation, this thesis only presents a selection of the work done during my Ph.D.; please see Appendix D for a full list.

Chapter 2

Preliminaries

"Mathematics is written for mathematicians."
- Nicolaus Copernicus in *De revolutionibus orbium coelestium* [Cop95].

"Often, the most important step is making the right notion, defining the right notion. Once you have the right notion, you know, the rest of the theory, theorems, proofs, [and] constructions follow."
- Avi Wigderson, 2024, in *Alan Turing: A TCS Role Model*[3]

Readers are recommended to skim Section 2.1 and refer back to this chapter as and when needed for relevant known definitions and results when reading subsequent chapters. The results presented in this section are mostly textbook level material. For completeness, we also present some proofs as a suitable citation could not be found at the time when we were using them for our own work.

2.1 Typography, abbreviations, and notation

Throughout this thesis, we use typography in the following way:

Typography	Representations
Lowercase letters	Scalar, set element, random variable instantiation
Uppercase letters	Random variable
Bolded lowercase letters	Vector, set
Bolded uppercase letters	Set/Vector of random variables, matrix
Calligraphic letters	Probability distribution, graph, set of sets
Bolded calligraphic letters	Set of probability distributions
Small caps	Algorithm name

[3] Turing Award Lecture: "Alan Turing: A TCS Role Model".
Available at https://www.youtube.com/live/f2NiGO8zC1c?t=3211s; see the 53min 31sec mark

Intuitively, we use non-bolded versions for singletons, bolded versions for collections of items, and calligraphic for more complicated objects. The context should be clear enough to distinguish between various representations of the same typography. We also employ the following abbreviations:

Abbreviations	Full form
w.l.o.g.	without loss of generality
w.p.	with probability
w.h.p.	with high probability
a.k.a.	also known as
i.i.d.	independent and identically distributed
w.r.t.	with respect to

We use $\mathbb{N}$ and $\mathbb{R}$ to represent the set of natural and real numbers respectively. To denote natural numbers without 0, we use $\mathbb{N}^+ = \mathbb{N} \setminus \{0\}$. For any natural number $n \in \mathbb{N}^+$, we write $[n]$ as shorthand for the set $\{1, 2, \ldots, n\}$. We also write $\mathrm{poly}(n)$ to mean "some polynomial in n", $\exp(n)$ to mean "some exponential in n", and employ the standard asymptotic notations [Knu76, GK90] such as $\mathcal{O}(\cdot)$, $\Omega(\cdot)$, and $\Theta(\cdot)$. We also write $\widetilde{\mathcal{O}}(\cdot)$, $\widetilde{\Omega}(\cdot)$, and $\widetilde{\Theta}(\cdot)$ to hide logarithmic factors. We generally use superscript * to denote ground truth and hats $\widehat{\cdot}$ to denote estimated quantities.

The indicator function $\mathbb{1}_{\text{predicate}}$ is 1 if the predicate is true and 0 otherwise. The notation $\wedge$ and $\vee$ refer to logical-AND and logical-OR respectively. We denote the domain of the variable V as Σ_V, and extend this to sets by letting $\Sigma_{\boldsymbol{A}} = \Sigma_{V_1} \times \ldots \times \Sigma_{V_k}$, where $\boldsymbol{A} = \{V_1, \ldots, V_k\}$ and $\times$ denotes the Cartesian product. To lighten notation, we write $\mathcal{P}(\boldsymbol{A} = \boldsymbol{a})$ as $\mathcal{P}(\boldsymbol{a})$ as shorthand and summations are always taken over the entire alphabet of the index, i.e. $\sum_{\boldsymbol{a}} f(\boldsymbol{a})$ denotes $\sum_{\boldsymbol{a} \in \Sigma_{\boldsymbol{A}}} f(\boldsymbol{a})$ for some function f over possible values $\boldsymbol{a}$ of variables $\boldsymbol{A} = \{V_1, \ldots, V_k\}$.

Consider two arbitrary sets $\boldsymbol{A}$ and $\boldsymbol{B}$. We denote the powerset of $\boldsymbol{A}$ by $2^{\boldsymbol{A}}$ and denote the set of all subsets of $\boldsymbol{A}$ of size $k \in \mathbb{N}^+$ by $\binom{\boldsymbol{A}}{k}$. $\boldsymbol{A} \subseteq \boldsymbol{B}$ means that $\boldsymbol{A}$ is a (possibly improper) subset of $\boldsymbol{B}$ while $\boldsymbol{A} \subset \boldsymbol{B}$ (or $\boldsymbol{A} \subsetneq \boldsymbol{B}$) means that $\boldsymbol{A}$ is a proper subset of $\boldsymbol{B}$. The notation $\boldsymbol{A} \sqcup \boldsymbol{B}$ refers to the disjoint union of sets $\boldsymbol{A}$ and $\boldsymbol{B}$, i.e. $\boldsymbol{A} \sqcup \boldsymbol{B} = \boldsymbol{A} \cup \boldsymbol{B}$ and $\boldsymbol{A} \cap \boldsymbol{B} = \emptyset$.

Finally, as a remark, note that this thesis mostly consider graphs $\mathcal{G}$ where vertices $\boldsymbol{V}$ correspond to variables $\boldsymbol{X}$ of a multivariate distribution $\mathcal{P}$ (except graphs in Chapter 9). As such, we may switch between using $\boldsymbol{X}$ and $\boldsymbol{V}$ to refer to variables/nodes/vertices interchangeably, but it should be clear from context.

2.2 Linear algebraic and combinatorial notions

2.2.1 Vectors

Definition 2.1 (L1 norm of a vector)**.** Let $\boldsymbol{x} \in \mathbb{R}^d$ be an arbitrary d-dimensional real vector with i-th entry $\boldsymbol{x}_i = x_i$. Then, the ℓ_1 norm of $\boldsymbol{x}$ is defined as $\ell_1(\boldsymbol{x}) = \|\boldsymbol{x}\|_1 = \sum_{i=1}^{d} |x_i|$.

Definition 2.2 (L2 norm of a vector)**.** Let $\boldsymbol{x} \in \mathbb{R}^d$ be an arbitrary d-dimensional real vector with i-th entry $\boldsymbol{x}_i = x_i$. Then, the ℓ_2 norm of $\boldsymbol{x}$ is defined as $\ell_2(\boldsymbol{x}) = \|\boldsymbol{x}\|_2 = \sqrt{\sum_{i=1}^{d} x_i^2}$.

Lemma 2.3 (Relation between L1 and L2 vector norms; e.g. see Exercise 5.4.P3 of [HJ12])**.** *Let $\boldsymbol{x} \in \mathbb{R}^d$ be an arbitrary d-dimensional real vector. Then, $\|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}\|_1 \leq \sqrt{d} \cdot \|\boldsymbol{x}\|_2$.*

Definition 2.4 (Projected vector)**.** Let $\boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_d) \in \mathbb{R}^d$ be a d-dimensional vector and $\boldsymbol{I} = \{i_1, \ldots, i_w\} \subseteq [d]$ be a subset of $1 \leq w \leq d$ indices, where $i_1 < \ldots < i_w$. Then, we define $\boldsymbol{x}_{\boldsymbol{I}} = (\boldsymbol{x}_{i_1}, \ldots, \boldsymbol{x}_{i_w}) \in \mathbb{R}^w$ as the projection of the vector $\boldsymbol{x}$ to the coordinates indicated by $\boldsymbol{I}$.

2.2.2 Matrices

We write the $d \times d$ dimensional identity matrix by $\boldsymbol{I}_d$. For an arbitrary matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, we denote the smallest and largest eigenvalues of $\boldsymbol{A}$ by $\sigma_{\min}(\boldsymbol{A})$ and $\sigma_{\max}(\boldsymbol{A})$ respectively. The rank of a matrix is the maximum number of linearly independent rows or columns in the matrix. When $m = n$, $\boldsymbol{A}$ is a square matrix and has trace $\operatorname{Tr}(\boldsymbol{A}) = \sum_{i=1}^{n} a_{i,i}$. If $\boldsymbol{A}$ is invertible, we write $\boldsymbol{A}^{-1}$ to denote its inverse. To convert between matrices and vectors, we use the notations $\operatorname{vec}(\cdot)$ and $\operatorname{mat}(\cdot)$. For example, we vectorize $\boldsymbol{A}$ via

$$\operatorname{vec}(\boldsymbol{A}) = (\boldsymbol{A}_{1,1}, \ldots, \boldsymbol{A}_{1,n}, \boldsymbol{A}_{2,1}, \ldots, \boldsymbol{A}_{2,n}, \ldots, \boldsymbol{A}_{m,1}, \ldots, \boldsymbol{A}_{m,n}) \in \mathbb{R}^{mn} ,$$

and unvectorize it via $\operatorname{mat}(\operatorname{vec}(\boldsymbol{A})) = \boldsymbol{A}$.

The following are some facts about matrix norms and the relations between them; e.g. see [HJ12]. Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be an arbitrary $m \times n$ real matrix of rank $r \leq \min\{m, n\}$ with (i, j)-th entries $\boldsymbol{A}_{i,j} = a_{i,j}$ and singular values $\sigma_1(\boldsymbol{A}), \ldots, \sigma_{\min\{m,n\}}(\boldsymbol{A})$. It is common to use $\sigma_{\min}(\boldsymbol{A})$ and $\sigma_{\max}(\boldsymbol{A})$ to denote the smallest and largest singular values of $\boldsymbol{A}$.

The Frobenius norm of $\boldsymbol{A}$ is defined as

$$\|\boldsymbol{A}\|_F = \sqrt{\sum_{i=1}^{m} \sum_{j=1}^{n} a_{i,j}^2} = \sqrt{\sum_{i=1}^{\min\{m,n\}} (\sigma_i(\boldsymbol{A}))^2} = \|\operatorname{vec}(\boldsymbol{A})\|_2$$

The spectral norm of $\boldsymbol{A}$ is defined as

$$\|\boldsymbol{A}\| = \|\boldsymbol{A}\|_2 = \sigma_{\max}(\boldsymbol{A}) = \max_{\boldsymbol{x} \neq \boldsymbol{0}} \frac{\|\boldsymbol{A}\boldsymbol{x}\|_2}{\|\boldsymbol{x}\|_2}$$

It is known that $\|\boldsymbol{A}\|_2 \leq \|\boldsymbol{A}\|_F \leq \sqrt{r} \cdot \|\boldsymbol{A}\|_2 \leq \sqrt{\min\{m,n\}} \cdot \|\boldsymbol{A}\|_2$.

Suppose $m = n$, so $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is a square matrix with eigenvalues $\lambda_1(\boldsymbol{A}), \ldots, \lambda_n(\boldsymbol{A})$. If $\boldsymbol{A}$ is symmetric, i.e. $\boldsymbol{A}^\top = \boldsymbol{A}$, then its singular values are the absolute values of its eigenvalues. Furthemore, if $\boldsymbol{A}$ is an invertible, then $\frac{1}{\|\boldsymbol{A}\|} = \frac{1}{\sigma_{\max}(\boldsymbol{A})} \leq \frac{1}{\sigma_{\min}(\boldsymbol{A})} = \|\boldsymbol{A}^{-1}\|$.

Lemma 2.5 (Upper bound on Frobenius norm of matrix product)**.** *Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ and $\boldsymbol{B} \in \mathbb{R}^{n \times k}$ be two arbitrary real matrices. Then, $\|\boldsymbol{AB}\|_F \leq \min\{\|\boldsymbol{A}\|_2 \|\boldsymbol{B}\|_F, \|\boldsymbol{A}\|_F \|\boldsymbol{B}\|_2\}$.*

Lemma 2.6 ([RV09]; Theorem 6.1 and Equation 6.10 in [Wai19])**.** *Let $\ell \geq d$ and $\boldsymbol{G} \in \mathbb{R}^{\ell \times d}$ be a matrix with i.i.d. $N(0,1)$ entries. Denote $\sigma_{\min}(\boldsymbol{G})$ as the smallest singular value of $\boldsymbol{G}$. Then, for any $0 < t < 1$, we have $\Pr\left(\sigma_{\min}(\boldsymbol{G}) \geq \sqrt{\ell}(1-t) - \sqrt{d}\right) \leq \exp\left(-\ell t^2/2\right)$.*

Lemma 2.7 (Trace inequality)**.** *For any three matrices $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \in \mathbb{R}^{d \times d}$, we have $\operatorname{Tr}(\boldsymbol{ABC}) \leq \|\operatorname{vec}(\boldsymbol{BA})\|_1 \cdot \|\boldsymbol{C}\|_2$.*

Proof. Let $\lambda_1(\boldsymbol{M}), \ldots, \lambda_d(\boldsymbol{M})$ denote the eigenvalues of a matrix $\boldsymbol{M} \in \mathbb{R}^{d \times d}$.

$$\begin{aligned}
\operatorname{Tr}(\boldsymbol{ABC}) &\leq \sum_i \lambda_i(\boldsymbol{AB}) \cdot \lambda_i(\boldsymbol{C}) && \text{(by von Neumann trace inequality)} \\
&= \sum_i \lambda_i(\boldsymbol{BA}) \cdot \lambda_i(\boldsymbol{C}) && \text{(e.g. see Theorem 1.3.22 of [HJ12])} \\
&\leq \sum_i |\lambda_i(\boldsymbol{BA}) \cdot \lambda_i(\boldsymbol{C})| \\
&\leq \left\| \begin{pmatrix} \lambda_1(\boldsymbol{BA}) \\ \vdots \\ \lambda_d(\boldsymbol{BA}) \end{pmatrix} \right\|_1 \cdot \left\| \begin{pmatrix} \lambda_1(\boldsymbol{C}) \\ \vdots \\ \lambda_d(\boldsymbol{C}) \end{pmatrix} \right\|_\infty && \text{(Hölder's inequality)} \\
&= \sum_i |\lambda_i(\boldsymbol{BA})| \cdot \max_i \lambda_i(\boldsymbol{C}) && \text{(Definitions of vector } \ell_1 \text{ and } \ell_\infty \text{ norms)} \\
&\leq \sum_i |\lambda_i(\boldsymbol{BA})| \cdot \|\boldsymbol{C}\|_2 && \text{(Definition of matrix spectral norm)}
\end{aligned}$$

It remains to argue that $\sum_i |\lambda_i(\boldsymbol{BA})| \leq \|\operatorname{vec}(\boldsymbol{BA})\|_1$. To this end, consider the singular value decomposition (SVD) of $\boldsymbol{BA} = \boldsymbol{U\Sigma V}^\top$ with unitary matrices $\boldsymbol{U}, \boldsymbol{V}$ and diagonal matrix $\boldsymbol{\Sigma} = \operatorname{diag}(\sigma_1, \ldots, \sigma_d)$. Let us denote the eigenvalues of $\boldsymbol{BA}$ by $\sigma_1, \ldots, \sigma_d$ and the columns of $\boldsymbol{BA}$ by $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_d \in \mathbb{R}^d$. Then,

$$\begin{aligned}
\sum_i |\lambda_i(\boldsymbol{BA})| &\leq \sum_i \sigma_i && \text{(e.g. see Equation (7.3.17) in [HJ12])} \\
&= \operatorname{Tr}(\boldsymbol{\Sigma}) && \text{(By definition of } \boldsymbol{\Sigma}\text{)} \\
&= \operatorname{Tr}(\boldsymbol{V}^\top \boldsymbol{V} \boldsymbol{U}^\top \boldsymbol{U} \boldsymbol{\Sigma}) && \text{(Since } \boldsymbol{U} \text{ and } \boldsymbol{V} \text{ are unitary matrices)} \\
&= \operatorname{Tr}(\boldsymbol{V} \boldsymbol{U}^\top \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{V}^\top) && \text{(By cyclic property of trace)} \\
&= \operatorname{Tr}(\boldsymbol{V} \boldsymbol{U}^\top \boldsymbol{BA}) && \text{(By SVD of } \boldsymbol{BA}\text{)}
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{i=1}^{d} (\boldsymbol{V}\boldsymbol{U}^\top \boldsymbol{z}_i)_i && \text{(By definition of trace)} \\
&\leq \sum_{i=1}^{d} \|\boldsymbol{V}\boldsymbol{U}^\top \boldsymbol{z}_i\|_2 \\
& \qquad \text{(Since } (\boldsymbol{V}\boldsymbol{U}^\top \boldsymbol{z}_i)_i^2 \text{ is just one term in summation of } \|\boldsymbol{V}\boldsymbol{U}^\top \boldsymbol{z}_i\|_2^2) \\
&= \sum_{i=1}^{d} \|\boldsymbol{z}_i\|_2 && \text{(Since } \boldsymbol{U} \text{ and } \boldsymbol{V} \text{ are unitary matrices)} \\
&\leq \sum_{i=1}^{d} \|\boldsymbol{z}_i\|_1 && \text{(Since } \ell_2 \leq \ell_1) \\
&= \sum_{i=1}^{d} \sum_{j=1}^{d} |(\boldsymbol{B}\boldsymbol{A})_{i,j}| && \text{(By definition of vector } \ell_1 \text{ norm)} \\
&= \|\mathrm{vec}(\boldsymbol{B}\boldsymbol{A})\|_1 && \text{(By definition of } \|\mathrm{vec}(\boldsymbol{B}\boldsymbol{A})\|_1)
\end{aligned}
$$

Putting together, we get $\mathrm{Tr}(\boldsymbol{A}\boldsymbol{B}\boldsymbol{C}) \leq \sum_i |\lambda_i(\boldsymbol{B}\boldsymbol{A})| \cdot \|\boldsymbol{C}\|_2 \leq \|\mathrm{vec}(\boldsymbol{B}\boldsymbol{A})\|_1 \cdot \|\boldsymbol{C}\|_2$ as desired. □

Lemma 2.8. *For any two matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{d \times d}$, we have*

- $\|\mathrm{vec}(\boldsymbol{A}+\boldsymbol{B})\|_1 \leq \|\mathrm{vec}(\boldsymbol{A})\|_1 + \|\mathrm{vec}(\boldsymbol{B})\|_1$, *and*
- $\|\mathrm{vec}(\boldsymbol{A}\boldsymbol{B})\|_1 \leq \|\mathrm{vec}(\boldsymbol{A})\|_1 \cdot \|\mathrm{vec}(\boldsymbol{B})\|_1$

Proof. To see $\|\mathrm{vec}(\boldsymbol{A}+\boldsymbol{B})\|_1 \leq \|\mathrm{vec}(\boldsymbol{A})\|_1 + \|\mathrm{vec}(\boldsymbol{B})\|_1$, observe that

$$
\begin{aligned}
\|\mathrm{vec}(\boldsymbol{A}+\boldsymbol{B})\|_1 &= \sum_{i=1}^{d}\sum_{j=1}^{d} |\boldsymbol{A}_{ij} + \boldsymbol{B}_{ij}| \\
&\leq \sum_{i=1}^{d}\sum_{j=1}^{d} |\boldsymbol{A}_{ij}| + \sum_{i=1}^{d}\sum_{j=1}^{d} |\boldsymbol{B}_{ij}| = \|\mathrm{vec}(\boldsymbol{A})\|_1 + \|\mathrm{vec}(\boldsymbol{B})\|_1
\end{aligned}
$$

To see $\|\mathrm{vec}(\boldsymbol{A}\boldsymbol{B})\|_1 \leq \|\mathrm{vec}(\boldsymbol{A})\|_1 \cdot \|\mathrm{vec}(\boldsymbol{B})\|_1$, observe that

$$
\begin{aligned}
\|\mathrm{vec}(\boldsymbol{A}\boldsymbol{B})\|_1 &= \sum_{i=1}^{d}\sum_{j=1}^{d}\sum_{k=1}^{d} |\boldsymbol{A}_{ij}\boldsymbol{B}_{jk}| \\
&\leq \left(\sum_{i=1}^{d}\sum_{j=1}^{d} |\boldsymbol{A}_{ij}|\right) \cdot \left(\sum_{j=1}^{d}\sum_{k=1}^{d} |\boldsymbol{B}_{jk}|\right) = \|\mathrm{vec}(\boldsymbol{A})\|_1 \cdot \|\mathrm{vec}(\boldsymbol{B})\|_1
\end{aligned}
$$

□

Lemma 2.9 (Chapter 5.6 of [HJ12])**.** *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be two square real matrices where $\boldsymbol{A}$ is an invertible matrix. Then, $\|\boldsymbol{A}\boldsymbol{B}\| = \|\boldsymbol{B}\boldsymbol{A}\|$.*

Proof. Exercise 5.6.P58(b) of [HJ12] tells us that $\|\boldsymbol{AB}\| = \|\boldsymbol{BA}\|$ when $\boldsymbol{A}$ normal and $\boldsymbol{B}$ is Hermitian. Since normal matrices are invertible and every real matrix is Hermitian, the claim follows. □

2.3 Probabilistic notions

2.3.1 Concentration bounds

Let us recap some basic concentration bounds and techniques used for bounding unlikely bad events. We say an event $\mathcal{E}$ holds with high probability[4] (w.h.p.) in n if $\Pr[\mathcal{E}] \geq 1 - \frac{1}{\text{poly}(n)}$. Markov's inequality is one of the simplest concentration bounds that makes almost no assumptions on the random variable. By using Markov's inequality and additional assumptions, one can show stronger concentration bounds such as Chebyshev's inequality, Chernoff bounds and Hoeffding bounds. It is also known that classes of random variables such as sub-Gaussian and sub-exponential random variables yield strong tail bounds. Interested readers may check out resources such as [Ver18, Chapter 2] to learn more.

Theorem 2.10 (Markov's inequality). *If X is a non-negative random variable and $t > 0$, then $\Pr(X \geq t) \leq \frac{\mathbb{E}(X)}{t}$.*

Theorem 2.11 (Chebyshev's inequality). *If X is a random variable with finite variance and $t > 0$, then $\Pr(|X - \mathbb{E}(X)| \geq t) \leq \frac{\mathsf{Var}(X)}{t^2}$.*

Theorem 2.12 (Chernoff bound). *For independent Bernoulli variables $X_1, \ldots, X_n$, let $X = \sum_{i=1}^{n} X_i$. Then*

$$\Pr(X \geq (1+\varepsilon) \cdot \mathbb{E}(X)) \leq \exp\left(-\frac{\varepsilon^2 \cdot \mathbb{E}(X)}{3}\right) \quad \textit{for } 0 < \varepsilon$$

$$\Pr(X \leq (1-\varepsilon) \cdot \mathbb{E}(X)) \leq \exp\left(-\frac{\varepsilon^2 \cdot \mathbb{E}(X)}{2}\right) \quad \textit{for } 0 < \varepsilon < 1$$

By union bound, for $0 < \varepsilon < 1$, we have

$$\Pr(|X - \mathbb{E}(X)| \geq \varepsilon \cdot \mathbb{E}(X)) \leq 2\exp\left(-\frac{\varepsilon^2 \cdot \mathbb{E}(X)}{3}\right)$$

One can convert expectation results to high probability ones by paying an extra $\mathcal{O}(\log n)$ factor via standard applications of Markov and Chernoff bounds: each event succeeds with constant probability via Markov inequality, so Chernoff bounds ensure that at least one out of $\mathcal{O}(\log n)$ independent runs succeeds with high probability.

When dealing with multiple bad events $\mathcal{E}_1, \ldots, \mathcal{E}_n$, we wish to upper bound the event $\mathcal{E}_1 \cup \ldots \cup \mathcal{E}_n$ that *at least one* of them occurs. If this probability is small, then we can be

[4] See https://en.wikipedia.org/wiki/With_high_probability

sure that *no* bad event occurred. When $n = 2$, the inclusion–exclusion principle[5] tells us that $\Pr[\mathcal{E}_1 \cup \mathcal{E}_2] = \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2] - \Pr[\mathcal{E}_1 \cap \mathcal{E}_2]$. Since probabilities are non-negative, one can conclude that $\Pr[\mathcal{E}_1 \cup \mathcal{E}_2] \leq \Pr[\mathcal{E}_1] + \Pr[\mathcal{E}_2]$ even without knowing how the events are correlated. Generalizing this for $n > 2$ yields the union bound.

Theorem 2.13 (Union bound)**.** *For any countable set of events $\mathcal{E}_1, \mathcal{E}_2, \ldots$, we have $\Pr(\cup_{i=1}^{\infty} \mathcal{E}_i) \leq \sum_{i=1}^{\infty} \Pr(\mathcal{E}_i)$.*

Union bound is by no means tight and one will get tighter bounds if information about the event intersections is known. However, union bound is very easy to apply as individual event probabilities are typically known. If each event occurs with exponentially small probability (as in the case of many concentration bounds), then the union bound still gives an exponentially small probability as long as the number of events is not "too many".

Hoeffding's lemma provides a bound on the moment generating function of a bounded random variable, serving as a key tool in concentration inequalities. In particular, it plays an important role in deriving tail bounds for sums of independent random variables.

Lemma 2.14 (Hoeffding's lemma; [Hoe94])**.** *Let X be any real-valued random variable in the range $[a, b]$. Then, for any $\lambda \in \mathbb{R}$, we have $\mathbb{E}\left(e^{\lambda(X - \mathbb{E}(X))}\right) \leq \exp\left(\frac{\lambda^2 (b-a)^2}{8}\right)$.*

2.3.2 Distributional distances

Definition 2.15 (Total variational (TV) distance)**.**
For two continuous distributions $\mathcal{P}$ and $\mathcal{Q}$ over domain $\boldsymbol{X}$,

$$\mathrm{d}_{\mathrm{TV}}(\mathcal{P}, \mathcal{Q}) = \frac{1}{2} \int_{\boldsymbol{x} \in \boldsymbol{X}} |\mathcal{P}(\boldsymbol{x}) - \mathcal{Q}(\boldsymbol{x})| \, dx$$

For two discrete distributions $\mathcal{P}$ and $\mathcal{Q}$ over $[n]$ for some $n \in \mathbb{N}^+$,

$$\mathrm{d}_{\mathrm{TV}}(\mathcal{P}, \mathcal{Q}) = \frac{1}{2} \sum_{x=1}^{n} |\mathcal{P}(x) - \mathcal{Q}(x)|$$

Definition 2.16 (Kullback–Leibler (KL) divergence)**.**
For two continuous distributions $\mathcal{P}$ and $\mathcal{Q}$ over $\boldsymbol{X}$,

$$\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) = \int_{\boldsymbol{x} \in \boldsymbol{X}} \mathcal{P}(\boldsymbol{x}) \log\left(\frac{\mathcal{P}(\boldsymbol{x})}{\mathcal{Q}(\boldsymbol{x})}\right) d\boldsymbol{x}$$

For two discrete distributions $\mathcal{P}$ and $\mathcal{Q}$ over $[n]$ for some $n \in \mathbb{N}^+$,

$$\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) = \sum_{x=1}^{n} \mathcal{P}(x) \log\left(\frac{\mathcal{P}(x)}{\mathcal{Q}(x)}\right)$$

[5] See https://en.wikipedia.org/wiki/Inclusion%E2%80%93exclusion_principle

Note that KL divergence is not symmetric in general.

Definition 2.17 (Squared Hellinger distance)**.**
For two continuous distributions $\mathcal{P}$ and $\mathcal{Q}$ over domain $\boldsymbol{X}$,

$$\mathrm{d}_{\mathrm{H}}^2(\mathcal{P}, \mathcal{Q}) = 1 - \int_{\boldsymbol{x} \in \boldsymbol{X}} \sqrt{\mathcal{P}(\boldsymbol{x}) \mathcal{Q}(\boldsymbol{x})} \, d\boldsymbol{x}$$

For two discrete distributions $\mathcal{P}$ and $\mathcal{Q}$ over $[n]$ for some $n \in \mathbb{N}^+$,

$$\mathrm{d}_{\mathrm{H}}^2(\mathcal{P}, \mathcal{Q}) = \frac{1}{2} \sum_{x=1}^{n} \sqrt{\mathcal{P}(x) \mathcal{Q}(x)}$$

Instead of directly dealing with total variational distance, a common analytic technique is to bound the KL divergence and then appeal to the Pinsker's inequality [Tsy09, Lemma 2.5, page 88] to upper bound d_{TV} via d_{KL}. This is because KL divergence tensorizes (while TV does not) which enables node-wise analyses.

Theorem 2.18 (Pinsker's inequality)**.** *For any two distributions $\mathcal{P}$ and $\mathcal{Q}$ on the same measurable space,*

$$\mathrm{d}_{\mathrm{TV}}(\mathcal{P}, \mathcal{Q}) \leq \sqrt{\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q})/2}$$

Thus, if $s(\varepsilon)$ samples are needed to learn a distribution $\mathcal{Q}$ such that $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) \leq \varepsilon$, $s(\varepsilon^2)$ samples are needed to ensure $\mathrm{d}_{\mathrm{TV}}(\mathcal{P}, \mathcal{Q}) \leq \varepsilon$.

Definition 2.19 ((Conditional) Mutual Information)**.** Fix a distribution $\mathcal{P}$. For random variables X and Y with domains $\boldsymbol{\Sigma}_X$ and $\boldsymbol{\Sigma}_Y$ respectively, their mutual information is defined as

$$I(X;Y) = \sum_{x \in \boldsymbol{\Sigma}_X, y \in \boldsymbol{\Sigma}_Y} \mathcal{P}(x, y) \cdot \log \left(\frac{\mathcal{P}(x, y)}{\mathcal{P}(x) \cdot \mathcal{P}(y)} \right) .$$

Conditioning on a third random variable Z with domain $\boldsymbol{\Sigma}_Z$, the conditional mutual information is defined as

$$I(X;Y \mid Z) = \sum_{x \in \boldsymbol{\Sigma}_X, y \in \boldsymbol{\Sigma}_Y, z \in \boldsymbol{\Sigma}_Z} \mathcal{P}(x, y, z) \cdot \log \left(\frac{\mathcal{P}(x, y, z) \cdot \mathcal{P}(z)}{\mathcal{P}(x, z) \cdot \mathcal{P}(y, z)} \right) .$$

2.3.3 Learning concepts and terminology

The Probably Approximately Correct (PAC) learning model [Val84] is a framework in computational learning theory that formalizes the concept of learning from examples. It defines a learning algorithm's ability to learn a target concept from a class of functions within specified bounds of accuracy and confidence. In the context of learning distributions, the PAC learning framework measures how many samples from an unknown underlying distribution $\mathcal{P}$ it takes for an algorithm to recover a close distribution $\widehat{\mathcal{P}}$, with good success probability, e.g. $\Pr(\mathrm{d}_{\mathrm{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon) \geq 1 - \delta$.

Proper versus improper learning. When the underlying $\mathcal{P}$ is assumed to belong to a certain class of distributions $\boldsymbol{\mathcal{C}}$, proper learning restricts the learner to produce an estimate $\widehat{\mathcal{P}} \in \boldsymbol{\mathcal{C}}$ while improper learning allows estimates to belong outside of $\boldsymbol{\mathcal{C}}$.

Realizable versus agnostic (non-realizable) setting. Given a class of distributions $\boldsymbol{\mathcal{C}}$ which the learner has to produce an estimate $\widehat{\mathcal{P}} \in \boldsymbol{\mathcal{C}}$ from, the realizable setting refers to the case where $\mathcal{P} \in \boldsymbol{\mathcal{C}}$ while the agnostic setting does not have this restriction.

2.3.4 Scheffé tournament

The classic method to select an approximate distribution amongst a set of candidate distributions is via the Scheffé tournament of [DL01], which provides a logarithmic dependency on the number of candidates.

Given sample access to an input distribution and explicit access to some candidate distributions, the Scheffé-based algorithm of [DK14] outputs with high probability a candidate distribution that is sufficiently close to the input distribution.

Theorem 2.20 ([DK14])**.** *Fix any accuracy parameter $\varepsilon > 0$ and confidence parameter $\delta > 0$. Suppose there is a distribution $\mathcal{P}$ over variables $\boldsymbol{X}$ and a collection of explicit distributions $\boldsymbol{\mathcal{Q}} = \{\mathcal{Q}_1, \ldots, \mathcal{Q}_m\}$, where each distribution $\mathcal{Q}_i$ is defined over the same set $\boldsymbol{X}$ and there exists some $\mathcal{Q}^* \in \boldsymbol{\mathcal{Q}}$ such that $\mathrm{d_{TV}}(\mathcal{P}, \mathcal{Q}^*) \leq \varepsilon$. Then, there is an algorithm that uses $\mathcal{O}\left(\frac{\log 1/\delta}{\varepsilon^2} \cdot \log m\right)$ samples from $\mathcal{P}$ and returns some $\mathcal{Q} \in \boldsymbol{\mathcal{Q}}$ such that $\mathrm{d_{TV}}(\mathcal{P}, \mathcal{Q}) \leq 10\varepsilon$ with success probability at least $1 - \delta$ and running time* $\mathrm{poly}(m, 1/\delta, 1/\varepsilon^2)$.

The result of [DK14] is actually more general than what we stated here. For instance, they only require sample access to the distributions in $\boldsymbol{\mathcal{Q}} = \{\boldsymbol{Q}_1, \ldots, \boldsymbol{Q}_m\}$ while our setting is simpler as we will actually have explicit descriptions of each of these distributions.

2.3.5 Nets and covering

In approximate learning, we are often interested in recovering a point in some $\mathbb{R}^d$ space which is close to the some true unknown point. As there are infinitely many points in $\mathbb{R}^d$, a useful trick is to discretize the space into sufficiently few points via ε-nets so that one can apply techniques such as Scheffé tournament (Section 2.3.4) to search for a "sufficiently good" point or union bound (Theorem 2.13) to argue that the total failure probability is "sufficiently small". In this thesis, we are mainly interested in the metric space on the set of d-dimensional reals $\mathbb{R}^d$ with metric function being the ℓ_2 distance $\mathrm{dist}(\boldsymbol{x}, \boldsymbol{y}) = \ell_2(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{x} - \boldsymbol{y}\|_2$ for any $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^d$.

Definition 2.21 (ε-net and covering number)**.** Let $(\boldsymbol{X}, \mathrm{dist})$ be a metric space with set $\boldsymbol{X}$ and metric function dist. For any $\varepsilon > 0$, a subset $\boldsymbol{S} \subseteq \boldsymbol{X}$ is called an ε-net of $\boldsymbol{X}$ if every point in $\boldsymbol{X}$ is within distance ε from some point in $\boldsymbol{S}$: $\forall x \in \boldsymbol{X}$, $\exists x_0 \in \boldsymbol{S}$ such that $\operatorname{dist}(x, x_0) \leq \varepsilon$. The covering number $\mathcal{N}(\boldsymbol{X}, \varepsilon)$ is defined as the smallest ε-nets of $\boldsymbol{X}$.

Theorem 2.22 (Bounds on covering number)**.** *Consider the metric space* $(\mathbb{R}^d, \ell_2)$*. Let* $\boldsymbol{B}(d,1,r) = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_1 \leq r\}$ *and* $\boldsymbol{B}(d,2,r) = \{\boldsymbol{x} \in \mathbb{R}^d : \|\boldsymbol{x}\|_2 \leq r\}$ *be the* ℓ_1 *and* ℓ_2 *Euclidean balls in* $\mathbb{R}^d$ *with radius* r *respectively. For any* $\varepsilon > 0$*,*

- $\left(\frac{1}{\varepsilon}\right)^d \leq \mathcal{N}(\boldsymbol{B}(d,2,1), \varepsilon) \leq \left(1 + \frac{2}{\varepsilon}\right)^d \leq \left(\frac{3}{\varepsilon}\right)^d$
- $\mathcal{N}(\boldsymbol{B}(d,1,r), \varepsilon) \leq d^{\frac{cr^2}{\varepsilon^2}}$*, for some absolute constant* $c > 0$

Proof. See [Ver18, Proposition 4.2.13] and [Ver12, Chapter 4, Example 2.8] respectively. □

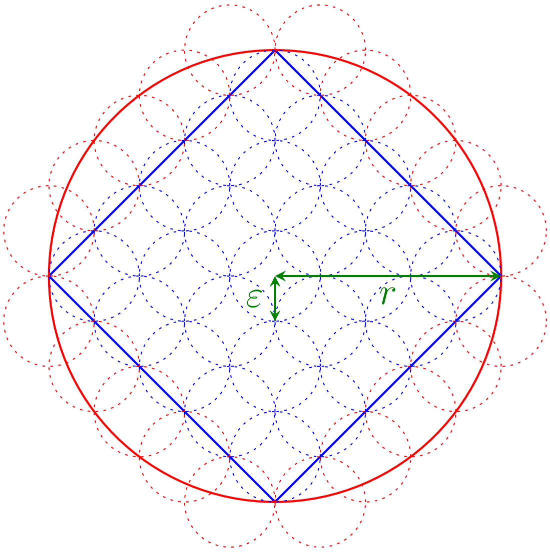


Figure 2.1: A sample illustration of covering a 2-dimensional ℓ_1 (in blue) and ℓ_2 (in red) balls of radius r with smaller ℓ_2 balls of radius ε. Observe that the blue ε-balls suffice to cover in the ℓ_1 ball while the ℓ_2 ball also requires the red ε-balls. The fact that the ℓ_2 ball incurs a larger covering number than the ℓ_1 ball is exacerbated in higher dimensions.

2.3.6 Fano's inequality

Fano's inequality is a commonly used information-theoretic tool used to provide lower bounds on the probability of error of *any* algorithm in estimating a discrete random variable X given observations $\boldsymbol{Y}$. Specifically, it relates the probability of producing an erroneous estimate $\widehat{X}$ given $\boldsymbol{Y}$ with the conditional entropy $H(X \mid \widehat{X})$. [SC21] provides an excellent introductory exposition on Fano's inequality and its various applications. Here we state the variant of Fano's inequality where the unknown X is drawn uniformly from a set.

Theorem 2.23 (Fano's inequality)**.** *Fix a finite alphabet $\boldsymbol{X}$ and an arbitrary estimator. Let $X \in \boldsymbol{V}$ be a discrete random variable in a hypothesis test and $\widehat{X} \in \boldsymbol{X}$ be an estimate of X by the estimator given observations $\boldsymbol{Y}$. If X is uniformly distributed over $\boldsymbol{X}$, then*

$$\Pr(X \neq \widehat{X}) \geq 1 - \frac{I(X; \widehat{X}) + 1}{\log |\boldsymbol{X}|}$$

where $I(X; \widehat{X}) = H(X) - H(X \mid \widehat{X})$ is the mutual information function.

The following inequality makes the Fano's inequality more user-friendly since it replaces the $I(X; \widehat{X})$ term with $I(X; \boldsymbol{Y})$. This is useful since it is typically easier to bound $I(X; \boldsymbol{Y})$ as we know how $\boldsymbol{Y}$ is generated given X.

Lemma 2.24 (Data processing inequality)**.** *Suppose X, $\boldsymbol{Y}$ and $\widehat{X}$ form a Markov chain $X \to \boldsymbol{Y} \to \widehat{X}$, where X and $\widehat{X}$ are independent given $\boldsymbol{Y}$. Then, $I(X; \boldsymbol{Y}) \geq I(X; \widehat{X})$.*

2.4 Some common distributions and random variables

2.4.1 Gaussian distribution

It is known that the Gaussian $\mathcal{Q} \sim N(\mathbf{0}, \widehat{\boldsymbol{\Sigma}})$ defined by the empirical covariance matrix $\widehat{\boldsymbol{\Sigma}}$, computed with $\mathcal{O}(n^2/\varepsilon^2)$ samples from $\mathcal{P}$, is ε-close in TV distance to $\mathcal{P}$ with constant probability. This sample complexity is also necessary for learning general n-dimensional Gaussians and hence general Gaussian Bayesian networks on n variables.

Lemma 2.25 (Folklore; e.g. see Appendix C of [ABDH+20])**.** *Fix $\varepsilon, \delta \in (0, 1)$. Given $2n$ i.i.d. samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{2n} \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for some unknown mean $\boldsymbol{\mu}$ and unknown covariance $\boldsymbol{\Sigma}$, define empirical mean and covariance as*

$$\widehat{\boldsymbol{\mu}} = \frac{1}{2n} \sum_{i=1}^{2n} \boldsymbol{x}_i \quad \textit{and} \quad \widehat{\boldsymbol{\Sigma}} = \frac{1}{2n} \sum_{i=1}^{n} (\boldsymbol{x}_{2i} - \boldsymbol{x}_{2i-1})(\boldsymbol{x}_{2i} - \boldsymbol{x}_{2i-1})^\top$$

Then,

- *When $n \in \mathcal{O}\left(\frac{d^2 + d \log(1/\delta)}{\varepsilon^2}\right)$, we have* $\Pr\left(\mathrm{d}_{\mathrm{TV}}(N(\boldsymbol{\mu}, \boldsymbol{\Sigma}), N(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}})) \leq \varepsilon\right) \geq 1 - \delta$
- *When $n \in \mathcal{O}\left(\frac{d + \sqrt{d \log(1/\delta)}}{\varepsilon^2}\right)$, we have* $\Pr\left((\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1} (\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}) \leq \varepsilon^2\right) \geq 1 - \delta$

There are also some additional known properties about the empirical covariance pertaining to eigenspaces.

Lemma 2.26 (Properties of empirical covariance; e.g. see Fact 3.4 of [KLSU19])**.** *Let $\widehat{\boldsymbol{\Sigma}} \in \mathbb{R}^{d \times d}$ be the empirical covariance constructed from n i.i.d. samples from $N(\mathbf{0}, \boldsymbol{\Sigma})$ for some unknown covariance $\boldsymbol{\Sigma}$. Then,*

- *When $n = d$, with probability 1, we have that $\widehat{\mathbf{\Sigma}}$ and $\mathbf{\Sigma}$ share the same eigenspace.*
- *Let $\lambda_1 \leq \ldots \leq \lambda_d$ and $\widehat{\lambda}_1 \leq \ldots \leq \widehat{\lambda}_d$ be the eigenvalues of $\mathbf{\Sigma}$ and $\widehat{\mathbf{\Sigma}}$ respectively. With probability at least $1 - \delta$, we have $\frac{\widehat{\lambda}_1}{\lambda_1} \leq 1 + \mathcal{O}\left(\sqrt{\frac{d+\log 1/\delta}{n}}\right)$.*

The following lemmas are known concentration results about Gaussian samples drawn from $N(0, \boldsymbol{I}_d)$.

Lemma 2.27. *Suppose $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n \sim N(0, \boldsymbol{I}_d)$. Then,*

$$\Pr\left(\left\|\sum_{i=1}^{n} \boldsymbol{g}_i\right\|_\infty \geq \sqrt{2n \log\left(\frac{2d}{\delta}\right)}\right) \leq \delta$$

Proof. Since $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n \sim N(0, \boldsymbol{I}_d)$, we see that $\boldsymbol{y} = \boldsymbol{g}_1 + \ldots + \boldsymbol{g}_n \sim N(0, n\boldsymbol{I}_d)$. Furthermore, each coordinate $i \in [d]$ of $\boldsymbol{y}_i = (y_1, \ldots, y_d)$ is distributed according to $N(0, n)$. By standard Gaussian tail bounds, we know that $\Pr(|y_i| \geq t) \leq 2\exp\left(-\frac{t^2}{2n}\right)$ for any $i \in [d]$ and $t > 0$. So,

$$\begin{aligned}
\Pr\left(\left\|\sum_{i=1}^{n} \boldsymbol{g}_i\right\|_\infty \geq \sqrt{2n \log\left(\frac{2d}{\delta}\right)}\right) &= \Pr\left(\|\boldsymbol{y}\|_\infty \geq \sqrt{2n \log\left(\frac{2d}{\delta}\right)}\right) \\
&= \Pr\left(\max_{i \in [d]} \|y_i\| \geq \sqrt{2n \log\left(\frac{2d}{\delta}\right)}\right) \\
&\leq \sum_{i=1}^{d} \Pr\left(\|y_i\| \geq \sqrt{2n \log\left(\frac{2d}{\delta}\right)}\right) \\
&\qquad \text{(Union bound over all } d \text{ coordinates)} \\
&\leq 2d \exp\left(-\frac{2n \log\left(\frac{2d}{\delta}\right)}{2n}\right) \\
&\qquad \text{(Setting } t = 2n \log\left(\tfrac{2d}{\delta}\right)\text{)} \\
&= \delta
\end{aligned}$$

□

Lemma 2.28 (Lemma C.4 in [ABDH+20]; Corollary 5.50 in [Ver10])**.** *Let $\boldsymbol{g}_1, \ldots, \boldsymbol{g}_n \sim N(\mathbf{0}, \boldsymbol{I}_d)$ and let $0 < \varepsilon < 1 < t$. If $n \geq c_0 \cdot \frac{t^2 d}{\varepsilon^2}$, for some absolute constant c_0, then*

$$\Pr\left(\left\|\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{g}_i \boldsymbol{g}_i^\top - \boldsymbol{I}_d\right\|_2 > \varepsilon\right) \leq 2\exp(-t^2 d)$$

The next lemma simplifies the KL divergence between two Gaussians in the special cases of identity covariance and equal means.

Lemma 2.29 (KL divergence of two Gaussians)**.** *Given two d-dimensional multivariate Gaussian distributions $\mathcal{P} \sim N(\boldsymbol{\mu}_{\mathcal{P}}, \boldsymbol{\Sigma}_{\mathcal{P}})$ and $\mathcal{Q} \sim N(\boldsymbol{\mu}_{\mathcal{Q}}, \boldsymbol{\Sigma}_{\mathcal{Q}})$ where $\boldsymbol{\Sigma}_{\mathcal{P}}$ and $\boldsymbol{\Sigma}_{\mathcal{Q}}$ are invertible, we have*

$$\begin{aligned} \mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) &= \frac{1}{2} \cdot \left(\mathrm{Tr}(\boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} \boldsymbol{\Sigma}_{\mathcal{P}}) - d + (\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}})^\top \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} (\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}}) + \ln\left(\frac{\det \boldsymbol{\Sigma}_{\mathcal{Q}}}{\det \boldsymbol{\Sigma}_{\mathcal{P}}}\right)\right) \\ &\leq \frac{1}{2} \cdot \left((\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}})^\top \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} (\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}}) + \|\boldsymbol{X}\|_F^2 \right) \end{aligned}$$

where $\boldsymbol{X} = \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1/2} \boldsymbol{\Sigma}_{\mathcal{P}} \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1/2} - \boldsymbol{I}_d$ with eigenvalues $\lambda_1, \ldots, \lambda_d$. In particular, $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) = \frac{1}{2}\|\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}}\|_2^2$ when $\boldsymbol{\Sigma}_{\mathcal{P}} = \boldsymbol{\Sigma}_{\mathcal{Q}} = \boldsymbol{I}_d$ and $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) \leq \frac{1}{2}\|\boldsymbol{X}\|_F^2$ when $\boldsymbol{\mu}_{\mathcal{P}} = \boldsymbol{\mu}_{\mathcal{Q}}$.

Proof. Let $\mathcal{P} \sim N(\boldsymbol{\mu}_{\mathcal{P}}, \boldsymbol{\Sigma}_{\mathcal{P}})$ and $\mathcal{Q} \sim N(\boldsymbol{\mu}_{\mathcal{Q}}, \boldsymbol{\Sigma}_{\mathcal{Q}})$ be two d-dimensional multivariate Gaussian distributions where $\boldsymbol{\Sigma}_{\mathcal{P}}$ and $\boldsymbol{\Sigma}_{\mathcal{Q}}$ are full rank invertible covariance matrices.

By definition, the KL divergence between $\mathcal{P}$ and $\mathcal{Q}$ is

$$\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) = \frac{1}{2} \cdot \left(\mathrm{Tr}(\boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} \boldsymbol{\Sigma}_{\mathcal{P}}) - d + (\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}})^\top \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} (\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}}) + \ln\left(\frac{\det \boldsymbol{\Sigma}_{\mathcal{Q}}}{\det \boldsymbol{\Sigma}_{\mathcal{P}}}\right)\right) \tag{2.1}$$

Let us define the matrix $\boldsymbol{X} = \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1/2} \boldsymbol{\Sigma}_{\mathcal{P}} \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1/2} - \boldsymbol{I}_d$ with eigenvalues $\lambda_1, \ldots, \lambda_d$. Note that $\boldsymbol{X}$ is invertible because $\boldsymbol{\Sigma}_{\mathcal{P}}$ and $\boldsymbol{\Sigma}_{\mathcal{Q}}$ are invertible, so $\lambda_1, \ldots, \lambda_d > 0$. Then, Eq. (2.1) can be upper bounded as

$$\begin{aligned} \mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) &= \frac{1}{2} \cdot \left(\mathrm{Tr}(\boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} \boldsymbol{\Sigma}_{\mathcal{P}}) - d + (\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}})^\top \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} (\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}}) + \ln\left(\frac{\det \boldsymbol{\Sigma}_{\mathcal{Q}}}{\det \boldsymbol{\Sigma}_{\mathcal{P}}}\right)\right) \\ &\leq \frac{1}{2} \left((\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}})^\top \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} (\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}}) + \|\boldsymbol{X}\|_F^2 \right) \end{aligned} \tag{2.2}$$

This is because $\mathrm{Tr}(\boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} \boldsymbol{\Sigma}_{\mathcal{P}}) = \mathrm{Tr}(\boldsymbol{\Sigma}_{\mathcal{Q}}^{-1/2} \boldsymbol{\Sigma}_{\mathcal{P}} \boldsymbol{\Sigma}_{\mathcal{Q}}^{-1/2}) = \mathrm{Tr}(\boldsymbol{X} + \boldsymbol{I}_d) = \mathrm{Tr}(\boldsymbol{X}) + d$ and

$$\begin{aligned} -\ln\left(\frac{\det \boldsymbol{\Sigma}_{\mathcal{Q}}}{\det \boldsymbol{\Sigma}_{\mathcal{P}}}\right) &= \ln\det\left(\boldsymbol{\Sigma}_{\mathcal{Q}}^{-1} \boldsymbol{\Sigma}_{\mathcal{P}}\right) = \ln\det(\boldsymbol{X} + \boldsymbol{I}_d) = \ln \prod_{i=1}^{d} (1 + \lambda_i) \\ &= \sum_{i=1}^{d} \ln(1 + \lambda_i) \geq \sum_{i=1}^{d} (\lambda_i - \lambda_i^2) = \mathrm{Tr}(\boldsymbol{X}) - \sum_{i=1}^{d} \lambda_i^2 = \mathrm{Tr}(\boldsymbol{X}) - \|\boldsymbol{X}\|_F^2 \end{aligned}$$

where the inequality holds due to $\lambda_1, \ldots, \lambda_d > 0$.

When $\boldsymbol{\Sigma}_{\mathcal{P}} = \boldsymbol{\Sigma}_{\mathcal{Q}} = \boldsymbol{I}_d$, Eq. (2.1) reduces to $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) = \frac{1}{2}\|\boldsymbol{\mu}_{\mathcal{Q}} - \boldsymbol{\mu}_{\mathcal{P}}\|_2^2$. Meanwhile, when $\boldsymbol{\mu}_{\mathcal{P}} = \boldsymbol{\mu}_{\mathcal{Q}}$, Eq. (2.2) reduces to $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) \leq \frac{1}{2}\left(\|\boldsymbol{X}\|_F^2\right)$. □

Finally, one can relate arbitrary Gaussians with the standard Gaussian through the following lemma.

Lemma 2.30 (Theorem 2.2 in [Gut09])**.** *Let $\boldsymbol{X}_1, \ldots, \boldsymbol{X}_p \sim N(0, \boldsymbol{L}\boldsymbol{L}^\top)$ be p i.i.d. n-dimensional multivariate Gaussians with covariance matrix $\boldsymbol{L}\boldsymbol{L}^\top \in \mathbb{R}^{n \times n}$, i.e. $\boldsymbol{L} \in \mathbb{R}^{n \times p}$.*

If $\boldsymbol{X} \in \mathbb{R}^{p\times n}$ is the matrix formed by stacking $\boldsymbol{X}_1, \dots, \boldsymbol{X}_p$ as rows of $\boldsymbol{X}$, then $\boldsymbol{X} = \boldsymbol{G}\boldsymbol{L}^\top$ where $\boldsymbol{G} \in \mathbb{R}^{p\times p}$ is a random matrix with i.i.d. $N(0,1)$ entries.

The transformation stated in [Gut09, Theorem 2.2, page 120] is for a single multivariate Gaussian vector, thus we need to take the transpose when we stack them in rows in Lemma 2.30. Note that $\boldsymbol{G}$ and $\boldsymbol{G}^\top$ are identically distributed.

2.4.2 Chi-square distribution

A closely related distribution to the Gaussian distribution is the chi-square distribution.

Lemma 2.31 (Equation 2.19 in [Wai19])**.** *Let $y = \sum_{k=1}^n z_k^2$, where each $z_k \sim N(0,1)$. Then, $y \sim \chi_n^2$ and for any $0 < t < 1$, we have $\Pr\left(|\frac{y}{n} - 1| \geq t\right) \leq 2\exp\left(-nt^2/8\right)$.*

Lemma 2.32. *Fix $n \geq 1$ and $d \geq 1$. Suppose we draw n samples $\boldsymbol{X}_1, \dots, \boldsymbol{X}_n \sim N(\boldsymbol{\mu}, \boldsymbol{I}_d)$, for some unknown mean $\boldsymbol{\mu} \in \mathbb{R}^d$. Define $\boldsymbol{Z}_n = \frac{1}{\sqrt{n}}\sum_{i=1}^n \boldsymbol{X}_i$ and $Y_n = \|\boldsymbol{Z}_n\|_2^2$. Then,*

1. *Y_n follows the* non-central chi-squared distribution *$\chi_d'^2(\lambda)$ for $\lambda = n\|\boldsymbol{\mu}\|_2^2$. This also implies that $\mathbb{E}[Y_n] = d + n\|\boldsymbol{\mu}\|_2^2$ and $\mathsf{Var}(Y_n) = 2d + 4n\|\boldsymbol{\mu}\|_2^2$.*

2. *For any $t > 0$,*

$$\begin{aligned}\Pr(Y_n > d + \lambda + t) &\leq \exp\left(-\frac{d}{2}\left(\frac{t}{d+2\lambda} - \log\left(1 + \frac{t}{d+2\lambda}\right)\right)\right)\\ &\leq \exp\left(-\frac{dt^2}{4(d+2\lambda)(d+2\lambda+t)}\right)\end{aligned}$$

3. *For any $t \in (0, d+\lambda)$,*

$$\begin{aligned}\Pr(Y_n < d + \lambda - t) &\leq \exp\left(\frac{d}{2}\left(\frac{t}{d+2\lambda} + \log\left(1 - \frac{t}{d+2\lambda}\right)\right)\right)\\ &\leq \exp\left(-\frac{dt^2}{4(d+2\lambda)^2}\right)\end{aligned}$$

Proof. The first item follows from the definition of the non-central chi-squared distribution, noting that the random vector $\boldsymbol{Z}_n$ is distributed as $N(\sqrt{n}\cdot\boldsymbol{\mu}, \boldsymbol{I}_d)$. The second and third items follow from Theorems 3 and 4 of [Gho21] respectively. □

2.4.3 Sub-Gaussian random variables

Sub-Gaussian random variables are a class of random variables that exhibit tail behavior similar to a Gaussian distribution. They are characterized by their tight concentration around the mean, and their moment generating function is bounded in a way that allows for stronger tail bounds than those provided by Markov's inequality.

Definition 2.33 (Sub-Gaussian distribution; e.g. see Section 1.2 of [RH23])**.** A random variable X is said to be sub-Gaussian with parameter σ^2 if we have $\mathbb{E}(X) = 0$ and $\mathbb{E}\left(e^{\lambda X}\right) \leq \exp\left(\frac{\lambda^2\sigma^2}{2}\right)$ for all $\lambda \in \mathbb{R}$. If $X \sim \mathrm{subG}(\sigma^2)$, it is known that we have $\Pr(|X| \geq t) \leq 2\exp\left(-\frac{t^2}{2\sigma^2}\right)$ for any $t \geq 0$.

Lemma 2.34 (Sub-Gaussian additivity; e.g. see Corollary 1.7 of [RH23])**.** *For $i \in [k]$, let $X_i \sim \mathrm{subG}(\sigma_i^2)$ be an independent sub-Gaussian random variable with parameter σ_i^2. Then, for any set of real coefficients $a_1, \ldots, a_k \in \mathbb{R}$, we have $\left(\sum_{i=1}^k a_i X_i\right) \sim \mathrm{subG}(\sum_{i=1}^k a_i^2\sigma_i^2)$.*

Lemma 2.35. *Let X and Y be discrete random variables. If $(X \mid Y = y) \sim \mathrm{subG}(\sigma_y^2)$ for every $y \in \mathbf{\Sigma}_Y$, then $X \sim \mathrm{subG}(\max_{y\in\mathbf{\Sigma}_Y} \sigma_y^2)$.*

Proof. By iterated expectation,

$$\begin{aligned}
\mathbb{E}\left(e^{\lambda X}\right) &= \mathbb{E}\left(\mathbb{E}\left(e^{\lambda X} \mid Y\right)\right) \\
&\leq \mathbb{E}\left(\exp\left(\frac{\lambda^2\sigma_Y^2}{2}\right)\right) \\
&\leq \mathbb{E}\left(\exp\left(\frac{\lambda^2 \max_{y\in\Sigma_Y}\sigma_y^2}{2}\right)\right) \\
&\leq \exp\left(\frac{\lambda^2 \max_{y\in\Sigma_Y}\sigma_y^2}{2}\right),
\end{aligned}$$

i.e., $X \in \mathrm{subG}(\max_{y\in\Sigma_Y}\sigma_y^2)$, as desired. □

2.4.4 Poisson random variables

The following known result regarding the concentration of the Poisson random variables is also helpful in bounding the overall algorithmic success probability when using the Poissonization technique; see Section 2.5.1.

Lemma 2.36 (Poisson concentration; e.g. see [Can19] and Theorem A.8 in [Can22])**.** *Let $N \sim \mathrm{Poi}(n)$ be a Poisson random variable with parameter n. Then, for any $t > 0$, we have $\Pr(N \geq n + t) \leq \exp\left(-\frac{t^2}{2(n+t)}\right)$, and for any $0 < t < n$, we have $\Pr(N \leq n - t) \leq \exp\left(-\frac{t^2}{2(n+t)}\right)$. In particular, setting $t = n/2$, we have $\Pr(N \leq n/2) \leq \exp\left(-\frac{n}{12}\right)$. Furthermore, by union bound, we have $\Pr(|N - n| \geq t) \leq 2\exp\left(-\frac{t^2}{2(n+t)}\right)$.*

2.5 Distribution testing and distance estimation

We will later use results from [JHW18] for the problem of ℓ_1 distance estimation. This is closely related to *tolerant identity testing*, where the tester's task is to distinguish whether a distribution $\mathcal{P}$ is ε_1-close to some known distribution $\mathcal{Q}$ from the case where $\mathcal{P}$ is ε_2-far from $\mathcal{Q}$, according to some natural distance measure.

The following theorem states the number of samples from an unknown distribution $\mathcal{P}$ that needed by the algorithm in [JHW18] to get an estimate of $\ell_1(\mathcal{P}, \mathcal{Q})$ for some reference distribution $\mathcal{Q}$ with additive error ε and error probability δ.[6]

Theorem 2.37 (adapted from [JHW18])**.** *Fix a reference distribution $\mathcal{Q}$ over a domain T of size $|T| = r$ and let $s \in \mathcal{O}\left(\frac{r \cdot \log(1/\delta)}{\varepsilon^2 \cdot \log r}\right)$ be an even integer. There exists an algorithm that draws $s_1 + s_2$ i.i.d. samples from an unknown distribution $\mathcal{P}$ over T, where $s_1, s_2 \sim \mathrm{Poisson}(s/2)$, and outputs an estimate $\widehat{\ell_1}$ such that $|\widehat{\ell_1} - \ell_1(\mathcal{P}, \mathcal{Q})| \leq \varepsilon$ with success probability at least $1 - \delta$.*

Proof. By [JHW18, Theorem 2], their estimator has ε additive error in expectation when $s = \Theta(\frac{r}{\varepsilon^2 \log r})$. So, with $100s$ samples, we can achieve $\varepsilon/10$ additive error in expectation, i.e. $\mathbb{E}[|\widehat{\ell_1} - \ell_1(\mathcal{P}, \mathcal{Q})|] = \varepsilon/10$. By Markov's inequality, we get $\Pr[|\widehat{\ell_1} - \ell_1(\mathcal{P}, \mathcal{Q})| > \varepsilon] \leq 1/10$. Thus, by repeating the entire algorithm $O(\log(1/\delta))$ times and choosing the median $\widetilde{\ell_1}$ of the resulting estimates, we get $\Pr[|\widetilde{\ell_1} - \ell_1(\mathcal{P}, \mathcal{Q})| > \varepsilon] \leq \delta$. □

2.5.1 Poissonization

The algorithm of Theorem 2.37, and our sample complexity bounds in Chapter 7, rely on a standard technique in distribution testing known as *Poissonization* which aims to eliminate correlations between samples at the expense of not having a fixed sample size; e.g. see [Val08, Section 4.3], [Can20b, Appendix D.3], and [Can22, Appendix C].

When drawing n i.i.d. samples from an underlying distribution $\mathcal{P}(X)$ over a domain $\Sigma_X = \{1, \ldots, k\}$, the vector of counts $(N_1, \ldots, N_k)$ follows a multinomial distribution with parameters n and $(\mathcal{P}(X = 1), \ldots, \mathcal{P}(X = k))$, where each random variable N_i is the number of times we observe $i \in [k]$ amongst the $n = N_1 + \ldots + N_k$ drawn samples. Oftentimes, in analysis, we would like that the random variables $N_1, \ldots, N_k$ are independent but this is unfortunately false in this setting since they are dependent and in fact negatively correlated.

Instead of directly drawing n i.i.d. samples, the idea behind Poissonization is to modify the sampling process by first sampling a Poisson number $N_{\mathrm{Poi}} \sim \mathrm{Poi}(n)$ with mean n and then drawing N_{Poi} i.i.d samples. Under this Poissonization sampling process, the resulting count vector has a few desirable properties.

Lemma 2.38 (Appendix C of [Can22])**.** *Let $(N_1, \ldots, N_k)$ be the sample counts in the Poissonized sampling process such that $N_1 + \ldots + N_k = N_{\mathrm{Poi}} \sim \mathrm{Poi}(n)$. Then, the following statements hold:*

[6] It is our understanding that the tester proposed by [JHW18] requires a significant amount of hyperparameter tuning and no off-the-shelf implementation is available [Han24].

(a) The random count variables $N_1, \ldots, N_k$ are mutually independent

(b) For each $i \in [k]$, we have $N_i \sim \mathrm{Poi}(n \cdot \mathcal{P}(X = i))$

(c) For each $i \in [k]$ and $n' \in \mathbb{N}$, we have $(N_i \mid N_{\mathrm{Poi}} = n') \sim \mathrm{Bin}(n', \mathcal{P}(X = i))$.

2.5.2 Known sample complexity results for discrete distributions

In Chapter 7, part of our analysis involves estimating $\mathcal{P}(\boldsymbol{A})$ well in TV distance, for some subset of variables $\boldsymbol{A} \subseteq \boldsymbol{V}$. In the distribution testing literature, this task is well-known to require $\widetilde{\Theta}\left(\frac{|\boldsymbol{\Sigma_A}|}{\varepsilon^2}\right)$ i.i.d. samples, where $\boldsymbol{\Sigma_A}$ is the alphabet size of the variables $\boldsymbol{A}$.

Lemma 2.39 (Estimating well in TV; e.g. see [Can20a])**.** *Given tolerance parameters $\varepsilon, \delta > 0$ and sample access to a distribution $\mathcal{P}(\boldsymbol{V})$, the empirical distribution $\widehat{\mathcal{P}}(\boldsymbol{V})$ constructed from $\mathcal{O}\left(\frac{|\boldsymbol{\Sigma_V}| + \log \frac{1}{\delta}}{\varepsilon^2}\right)$ i.i.d. samples has the property that*

$$\Pr\left(\sum_{\boldsymbol{v} \in \boldsymbol{\Sigma_V}} |\mathcal{P}(\boldsymbol{v}) - \widehat{\mathcal{P}}(\boldsymbol{v})| \leq \varepsilon\right) \geq 1 - \delta$$

Meanwhile, in many practical settings, exact conditional independence is rarely satisfied due to noise and complex interactions between variables. Instead, we often rely on approximate conditional independence, which relaxes this assumption while still preserving the essence of conditional independence. This approach allows us to model more realistic scenarios and still capture important structural dependencies within the data.

Definition 2.40 (Approximate conditional independence)**.** For disjoint sets $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \subseteq \boldsymbol{V}$, we define $\Delta_{\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{C}} = \sum_{\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}} \mathcal{P}(\boldsymbol{c}) \cdot |\mathcal{P}(\boldsymbol{a}, \boldsymbol{b} \mid \boldsymbol{c}) - \mathcal{P}(\boldsymbol{a} \mid \boldsymbol{c}) \cdot \mathcal{P}(\boldsymbol{b} \mid \boldsymbol{c})|$. If $\Delta_{\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{C}} \leq \varepsilon$ for $\varepsilon \geq 0$, we write $\boldsymbol{A} \perp\!\!\!\perp_\varepsilon \boldsymbol{B} \mid \boldsymbol{C}$.

When $\varepsilon = 0$ in Definition 2.40, we recover the usual notion of conditional independence. Several methods have been developed which satisfy the requirements of the ε-approximate conditional independence tester for Definition 2.40. In this thesis, we call our ε-approximate conditional independence tester ApproxCondInd. Assuming that ε^{-1} is sufficiently large[7] compared to $|\boldsymbol{\Sigma_A}|$, $|\boldsymbol{\Sigma_B}|$, and $|\boldsymbol{\Sigma_C}|$, [CDKS18] proposes a test based on total variation distance that uses $\widetilde{\mathcal{O}}\left(\frac{1}{\varepsilon^2} \cdot \sqrt{|\boldsymbol{\Sigma_A}| \cdot |\boldsymbol{\Sigma_B}| \cdot |\boldsymbol{\Sigma_C}|}\right)$ samples from $\mathcal{P}$; see their Theorem 1.3 and Lemma 2.2. There is also a simpler test based on the empirical mutual information, proposed by [BGP$^+$23], that uses $\widetilde{\mathcal{O}}\left(\frac{1}{\varepsilon^2} \cdot |\boldsymbol{\Sigma_A}| \cdot |\boldsymbol{\Sigma_B}| \cdot |\boldsymbol{\Sigma_C}|\right)$ samples from $\mathcal{P}$, though we use the former to obtain optimal dependence on the alphabet sizes.

Lemma 2.41 (Using [CDKS18] as a blackbox in ApproxCondInd)**.** *Given tolerance parameters $\varepsilon, \delta > 0$ and sample access to a distribution $\mathcal{P}(\boldsymbol{V})$, the* ApproxCondInd

[7] For instance, $\frac{1}{\varepsilon} > |\boldsymbol{\Sigma_C}|^{\frac{1}{4}} \cdot (\max\{|\boldsymbol{\Sigma_A}|, |\boldsymbol{\Sigma_B}|, |\boldsymbol{\Sigma_C}|\})^{\frac{1}{4}}$ would suffice.

algorithm uses $\widetilde{\mathcal{O}}\left(\frac{1}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{A}}| \cdot |\Sigma_{\boldsymbol{B}}| \cdot |\Sigma_{\boldsymbol{C}}|} \cdot \log \frac{1}{\delta}\right)$ *samples and correctly determines whether* $\Delta_{\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{C}} = 0$ *(output YES) or* $\Delta_{\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{C}} > \varepsilon$ *(output NO) with probability at least* $1 - \delta$*, for any disjoint sets* $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \subseteq \boldsymbol{V}$.

Note that when $0 < \Delta_{\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{C}} \leq \varepsilon$, ApproxCondInd is allowed to output arbitrarily. In particular, when ApproxCondInd outputs YES on inputs $(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \varepsilon, \delta)$, then we have $\Pr(\Delta_{\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{C}} \leq \varepsilon) \geq 1 - \delta$.

2.6 Graphical notions

Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a graph on $|\boldsymbol{V}| = n$ vertices and $|\boldsymbol{E}|$ edges. Formally speaking, $\boldsymbol{E}$ is a set that contains both unordered and ordered pairs of vertices $U, V \in \boldsymbol{V}$ subject to the following two constraints:

- $\{U, V\} \in \boldsymbol{E} \implies (U, V) \not\in \boldsymbol{E} \wedge (V, U) \not\in \boldsymbol{E}$.
  Or equivalently, $(U, V) \in \boldsymbol{E} \vee (V, U) \in \boldsymbol{E} \implies \{U, V\} \not\in \boldsymbol{E}$
- $(U, V) \in \boldsymbol{E} \iff (V, U) \not\in \boldsymbol{E}$

For $U, V \in \boldsymbol{V}$, unordered pairs represent unoriented adjacencies (e.g. $U - V$) while ordered pairs represent directionality / arcs / oriented adjacencies (e.g. $U \to V$).

We use “oriented” and “directed” interchangeably and often omit the subscript $\mathcal{G}$ in the above definitions when the graph in discussion is clear from context.

2.6.1 General graph notions

A graph $\mathcal{G}$ may be partially oriented in general and the skeleton $\mathrm{skel}(\mathcal{G}) = (\boldsymbol{V}, \boldsymbol{E}')$ of $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ refers to the resulting fully undirected graph when all arc directions are made unoriented, i.e. $\boldsymbol{E}' = \{\{A, B\} : \{A, B\} \in \boldsymbol{E} \vee (A, B) \in \boldsymbol{E} \vee (B, A) \in \boldsymbol{E}\}$. When orientation is unspecified, $\mathcal{G}$ is assumed to be fully undirected, i.e. $\mathrm{skel}(\mathcal{G}) = \mathcal{G}$. Graph $\mathcal{G}$ is said to be *complete* if every pair of nodes are adjacent, i.e. the underlying undirected graph forms a clique. The clique number $\omega(\mathcal{G})$ refers to the largest size of any clique in $\mathcal{G}$. A *maximal clique* is an vertex-induced subgraph of a graph that is a clique and ceases to be one if we add any other vertex to the subgraph. If all edges in the clique are oriented in an acyclic manner, then there is a unique valid permutation π that respects this orientation and we denote $V = \mathrm{argmax}_{U \in \boldsymbol{V}} \pi(U)$ as the *sink* of the clique. A directed cycle is a sequence of edges forming an undirected cycle with at least one oriented arc, and all oriented arcs are in the same direction along this cycle. Partially oriented graphs without directed cycles are also known as *chain graphs*. A chain component is a maximally connected subgraph after removing arcs from $\mathcal{G}$; we denote the set of chain components by $CC(\mathcal{G})$. For any subset $\boldsymbol{V}' \subseteq \boldsymbol{V}$ and $\boldsymbol{E}' \subseteq \boldsymbol{E}$, we use $\mathcal{G}[\boldsymbol{V}']$ and $\mathcal{G}[\boldsymbol{E}']$ to denote the node-induced and edge-induced subgraphs of $\mathcal{G}$ respectively. Meanwhile, $\boldsymbol{V}(\mathcal{G})$, $\boldsymbol{E}(\mathcal{G})$, and $\boldsymbol{A}(\mathcal{G})$ refer to the set of vertices, edges, and oriented arcs of any given graph $\mathcal{G}$ respectively.

2.6.2 Undirected graph notions

Suppose graph $\mathcal{G}$ is fully unoriented. For vertices $U, V \in \boldsymbol{V}$, subset of vertices $\boldsymbol{V}' \subseteq \boldsymbol{V}$ and integer $r \geq 0$, define $\mathrm{dist}_{\mathcal{G}}(U, V)$ as the shortest path length between U and V, $\mathrm{dist}_{\mathcal{G}}(V, \boldsymbol{V}') = \min_{U \in \boldsymbol{V}'} \mathrm{dist}_{\mathcal{G}}(U, V)$, and $N^r_{\mathcal{G}}(\boldsymbol{V}') = \{V \in \boldsymbol{V} : \mathrm{dist}_{\mathcal{G}}(V, \boldsymbol{V}') \leq r\} \subseteq \boldsymbol{V}$ as the set of vertices that are r-hops away from $\boldsymbol{V}'$, i.e. r-hop neighbors of $\boldsymbol{V}'$. We omit r when referring to 1-hop neighbors, e.g. $N_{\mathcal{G}}(V)$. Graph $\mathcal{G}$ is said to be connected if there is a path between every pair of vertices. A chordal graph is a graph where every cycle of length at least 4 has a chord, which is an edge that is not part of the cycle but connects two vertices of the cycle. See [BP93] for more properties. A minimum vertex cover in a graph is a smallest possible set of vertices such that every edge in the graph is incident to at least one vertex in this set. In other words, each edge in the graph must be "covered" by at least one vertex from this set. If all vertices of a graph $\mathcal{G}$ can be colored with k colors (but not by $k-1$ colors) such that each vertex is assigned a color distinct from its neighbors, then the coloring number of $\mathcal{G}$ is defined to be $\chi(\mathcal{G}) = k$.

2.6.3 Directed graph notions

Suppose graph $\mathcal{G}$ is fully oriented. For any node $V \in \boldsymbol{V}$, we write $\mathrm{Pa}_{\mathcal{G}}(V)$, $\mathrm{An}_{\mathcal{G}}(V)$, $\mathrm{De}_{\mathcal{G}}(V) \subseteq \boldsymbol{V}$ to denote its parents, ancestors and descendants respectively. By convention, we have $V \not\in \mathrm{Pa}_{\mathcal{G}}(V)$, $V \in \mathrm{An}_{\mathcal{G}}(V)$, and $V \in \mathrm{De}_{\mathcal{G}}(V)$ and $\mathrm{pa}_{\mathcal{G}}(V)$ denotes the values taken by V's parents. We further define $\mathrm{Ch}_{\mathcal{G}}(V) \subseteq \mathrm{De}_{\mathcal{G}}(V)$ as the set of *direct children* of V whereby for any $W \in \mathrm{Ch}_{\mathcal{G}}(V)$ there does *not* exists $Z \in \boldsymbol{V} \setminus \{V, W\}$ such that $Z \in \mathrm{De}_{\mathcal{G}}(V) \cap \mathrm{An}_{\mathcal{G}}(W)$. Note that $\mathrm{Ch}_{\mathcal{G}}(V) \subseteq \{W \in \boldsymbol{V} : V \to W\} \subseteq \mathrm{De}_{\mathcal{G}}(V)$. An edge $U \to V$ is called a covered edge [Chi95] if $\mathrm{Pa}(U) = \mathrm{Pa}(V) \setminus \{U\}$.

2.6.4 Directed acyclic graph (DAG)

A directed acyclic graph (DAG) is a fully oriented graph $\mathcal{G}$ that does not contain any directed cycles. A vertex V_i on any simple path $V_1 - \ldots - V_k$ is called a *collider* if the arcs are such that $V_{i-1} \to V_i \leftarrow V_{i+1}$. If we further have $V_{i-1} \not\sim V_{i+1}$, then $V_{i-1} \to V_i \leftarrow V_{i+1}$ is also called a v-structure with center V_i in $\mathcal{G}$. One can associate a (not necessarily unique) *valid permutation / topological ordering* $\pi_{\mathcal{G}} : \boldsymbol{V} \to [n]$ to any (partially directed) DAG $\mathcal{G}$ such that oriented arcs (U, V) satisfy $\pi_{\mathcal{G}}(U) < \pi_{\mathcal{G}}(V)$ and unoriented arcs $\{U, V\}$ can be oriented as $U \to V$ without forming directed cycles when $\pi_{\mathcal{G}}(U) < \pi_{\mathcal{G}}(V)$.

2.6.5 Graph separators

Existence and efficient computation of graph separators are well studied [LT79, GHT84, GRE84, AST90, KR10, WN11] and are commonly used in divide-and-conquer graph algorithms and as analysis tools.

Definition 2.42 (α-separator and α-clique separator)**.** Let $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ be a partition of the vertices $\boldsymbol{V}$ of a graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, i.e. $\boldsymbol{A} \sqcup \boldsymbol{B} \sqcup \boldsymbol{C} = \boldsymbol{V}$. We say that $\boldsymbol{C}$ is an *α-separator* if no edge joins a vertex in $\boldsymbol{A}$ with a vertex in $\boldsymbol{B}$ and $|\boldsymbol{A}|, |\boldsymbol{B}| \leq \alpha \cdot |\boldsymbol{V}|$. We call $\boldsymbol{C}$ an *α-clique separator* if it is an *α-separator* and a clique.

Theorem 2.43 ([GRE84], instantiated for unweighted graphs)**.** *Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a chordal graph with $|\boldsymbol{V}| \geq 2$ and p vertices in its largest clique. There exists a $1/2$-clique-separator $\boldsymbol{C}$ of size $|\boldsymbol{C}| \leq p - 1$. The clique $\boldsymbol{C}$ can be computed in $\mathcal{O}(|\boldsymbol{E}|)$ time.*

Lemma 2.44 ([GRE84])**.** *Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a chordal graph with $|\boldsymbol{V}| \geq 2$ and p vertices in its largest clique. Suppose each vertex v is assigned a non-negative weight $c(v) \geq 0$ such that $\sum_v c(v) = n$. Then, there exists a $1/2$-clique-separator $\mathcal{S}$ of size $|\boldsymbol{V}(\mathcal{S})| \leq p-1$ such that any connected component in $\mathcal{G}$ after the removal has total weight of no more than $\sum_{v \in \boldsymbol{V}} c(v)/2$. The clique $\mathcal{S}$ can be computed in $\mathcal{O}(|\boldsymbol{E}|)$ time.*

2.6.6 Meek rules

Meek rules are a set of 4 edge orientation rules that are sound and complete with respect to any given set of arcs that has a consistent DAG extension [Mee95]. Given any edge orientation information, one can always repeatedly apply Meek rules till a unique fixed point (where no further rules trigger) to maximize the number of oriented arcs.

Definition 2.45 (The four Meek rules [Mee95], see Fig. 2.2 for an illustration)**.**

R1 Edge $\{A, B\} \in \boldsymbol{E}(\mathcal{G})$ is oriented as $A \to B$ if $\exists\, C \in \boldsymbol{V}$ such that $C \to A$ and $C \not\sim B$.

R2 Edge $\{A, B\} \in \boldsymbol{E}(\mathcal{G})$ is oriented as $A \to B$ if $\exists\, C \in \boldsymbol{V}$ such that $A \to C \to B$.

R3 Edge $\{A, B\} \in \boldsymbol{E}(\mathcal{G})$ is oriented as $A \to B$ if $\exists\, D, D \in \boldsymbol{V}$ such that $D - A - C$, $D \to B \leftarrow C$, and $C \not\sim D$.

R4 Edge $\{A, B\} \in \boldsymbol{E}(\mathcal{G})$ is oriented as $A \to B$ if $\exists\, C, D \in \boldsymbol{V}$ such that $D - A - C$, $D \to C \to B$, and $B \not\sim D$.

Note that Meek R3 will trigger before any interventions are performed because $C \to B \leftarrow D$ is a v-structure that would have been oriented just from observational data.

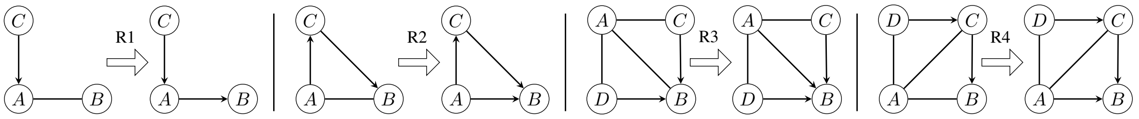


Figure 2.2: An illustration of the four Meek rules

There exists an algorithm (Algorithm 2 of [WBL21]) that runs in $\mathcal{O}(d \cdot |\boldsymbol{E}(\mathcal{G})|)$ time and computes the closure under Meek rules, where d is the degeneracy of the graph skeleton: a d-degenerate graph is an undirected graph in which every subgraph has a vertex of degree at most d. Note that the degeneracy of a graph is typically smaller than the maximum degree of the graph.

2.7 Bayesian networks

A Bayesian network $\mathcal{G}$ for a set of n variables $X_1, \ldots, X_n$ is described by a DAG $(\boldsymbol{X}, \boldsymbol{E})$ and n corresponding conditional probability tables (CPTs), e.g., the CPT for $X_i \in \boldsymbol{X}$ describes $\mathcal{P}(x_i \mid \mathrm{pa}_{\mathcal{G}}(X_i))$ for all possible values of x_i and $\mathrm{pa}_{\mathcal{G}}(X_i)$. In a Bayesian network $(\mathcal{G}, \mathcal{P})$, the joint distribution for $\mathcal{P}$ factorizes as $\mathcal{P}(\boldsymbol{x}) = \prod_{i=1}^{n} \mathcal{P}(x_i \mid \mathrm{pa}_{\mathcal{G}}(X_i))$, according to $\mathcal{G}$. As we can see, Bayesian networks allow us to establish a connection between a probability distribution $\mathcal{P}$ over n variables and a graph $\mathcal{G}$ with n nodes.

All independence constraints that hold in the joint distribution of a Bayesian network that has underlying DAG $\mathcal{G}$ are exactly captured by the *d-separation* criterion [Pea88, Section 3.3.1]. Two nodes $X, Y \in \boldsymbol{X}$ are said to be *d-separated* in a DAG $\mathcal{G} = (\boldsymbol{X}, \boldsymbol{E})$ given a set $\boldsymbol{Z} \in \boldsymbol{X} \setminus \{X, Y\}$ if and only if there is no $\boldsymbol{Z}$-active path in $\mathcal{G}$ between X and Y; a $\boldsymbol{Z}$-active path is a simple path Q such that any vertex from $\boldsymbol{Z}$ on Q occurs as a collider and any vertex from $\boldsymbol{X} \setminus \boldsymbol{Z}$ appears as a non-collider. Two nodes are *d-connected* if they are not d-separated. It is known that X is d-separated from its non-descendants given its parents [Pea88, Section 3.3.1, Corollary 4]. A Markov blanket of $X \in \boldsymbol{X}$ is a subset of variables $S \subseteq \boldsymbol{X}$ such that all other variables are independent of X, conditioned on $\boldsymbol{S}$. We use $X \perp_d Y \mid \boldsymbol{Z}$ to denote d-separation and $X \perp\!\!\!\perp_{\mathcal{P}} Y \mid \boldsymbol{Z}$ to denote conditional independence with respect to a distribution $\mathcal{P}$.

Definition 2.46 (Markov)**.** A probability distribution $\mathcal{P}$ is said to be Markov with respect to a DAG $\mathcal{G}$ if d-separation in $\mathcal{G}$ implies conditional independence in $\mathcal{P}$.

Note that any distribution is Markov with respect to the complete DAG, since there are no d-separations implied by this kind of DAG. Two DAGs are said to be Markov equivalent if they encode the same set of conditional independence relations. It is known that two graphs are Markov equivalent if and only if they have the same skeleton and v-structures [VP90, AMP97]. In fact, the Markov equivalence class (MEC) of any DAG

$\mathcal{G}$ can be represented by a partially oriented version of $\mathcal{G}$ called the essential graph $\mathcal{E}(\mathcal{G})$, which can be computed from $\mathcal{G}$ by orienting v-structures in $\mathrm{skel}(\mathcal{G})$ and applying Meek rules. Essential graphs are also known as completely partially directed acyclic graphs (CPDAGs). A special and important class of DAGs is that of moral DAGs.

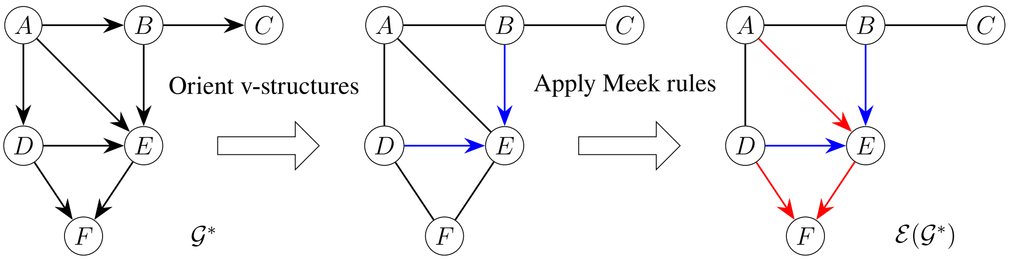


Figure 2.3: Example on how to compute an essential graph $\mathcal{E}(\mathcal{G}^*)$ of a given DAG $\mathcal{G}^*$

Definition 2.47 (Moral DAG)**.** A graph $\mathcal{G}$ is a *moral DAG* if its essential graph only has a single chain component. That is, after removing directed edges from $\mathcal{E}(\mathcal{G})$, there is only one single undirected connected component remaining.

Recall that $U \to V$ is called a covered edge [Chi95] if $\mathrm{Pa}(U) = \mathrm{Pa}(V) \setminus \{U\}$. Covered edges are special arcs in a causal graph discovery because their orientation can be reversed and they still yield the same conditional independencies. See Fig. 2.4 for an illustration. Note that one can compute all covered edges of a given DAG $\mathcal{G}$ in polynomial time.

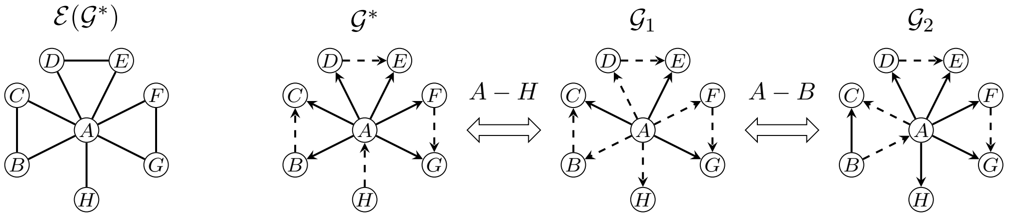


Figure 2.4: A DAG $\mathcal{G}^*$ with its essential graph $\mathcal{E}(\mathcal{G}^*)$ on the left. $\mathcal{G}_1$ and $\mathcal{G}_2$ are two other DAGs that belong to the same Markov equivalence class $[\mathcal{G}^*]$. Dashed arcs are covered edges in each DAG. One can perform a sequence of covered edge reversals to transform between the DAGs (see Lemma 2.49). Note that the sizes of the minimum vertex cover of the covered edges may differ across DAGs.

The following is a well-known result relating covered edges and MECs.

Definition 2.48 (Covered edge reversal)**.** A covered edge reversal means that we replace $U \to V$ with $V \to U$, for some covered edge $U \to V$, while keeping all other arcs unchanged.

Lemma 2.49 ([Chi95])**.** *If $\mathcal{G}$ and $\mathcal{G}'$ belong in the same MEC if and only if there exists a sequence of covered edge reversals to transform between them.*

The converse of the Markov property (Definition 2.46) is that of the faithfulness property. While the Markov property enables one to draw conclusions about $\mathcal{P}$ from $\mathcal{G}$, the faithfulness property allows one to make inferences in the reverse direction. This is particular useful when one wishes to construct a graphical representation of a distribution and variants of faithfulness has been used to recover causal graphs; see [Lam23].

Definition 2.50 (Faithfulness)**.** A probability distribution $\mathcal{P}$ is said to be faithful with respect to a DAG $\mathcal{G}$ if conditional independence in $\mathcal{P}$ implies d-separation in $\mathcal{G}$.

2.8 Causal DAGs

While we use DAGs for both causal and probabilistic models models, the former focuses on capturing associations or correlations among variables while the latter aims to uncover the underlying mechanisms that drive these associations. In particular, for a distribution $\mathcal{P}$ that is Markov with respect to DAG $\mathcal{G}^*$, any graph in the Markov equivalence class $[\mathcal{G}^*]$ of $\mathcal{G}^*$ is an equally good representation for $\mathcal{P}$. However, from a causal perspective, only one of them would be a good representation since the direction of the arcs in the DAG explicitly captures the causal relationship between variables. For example, if $\mathcal{P}$ is a joint distribution over the variables temperature (X) and altitude (Y), then the correct causal relation should be $Y \to X$ but the graph $X \to Y$ belongs in the same Markov equivalence class and is an equally good probabilistic representation. This distinction enables causal DAGs to answer questions not just about what happens, but about why and how it happens, making them particularly valuable when one wishes to understand the impact of performing interventions to the system.

2.8.1 Interventions

An intervention in a causal DAG refers to an external action that forcibly sets the values of a particular set of variables, thereby breaking their natural causal relationships with their parents in the graph. This process modifies the underlying causal structure, allowing for the analysis of the effects of this change on the other variables in the system.

Here, we study ideal interventions. Graphically speaking, an ideal intervention $\boldsymbol{S}$ on $\mathcal{G}$ induces an interventional graph $\mathcal{G}_{\overline{\boldsymbol{S}}}$ (also called a mutilated graph) where all incoming arcs to vertices $V \in \boldsymbol{S}$ are removed [EGS05]. An intervention $\boldsymbol{I} \subseteq \boldsymbol{V}$ is said to be k-bounded if $|\boldsymbol{I}| \leq k$; it is called an atomic intervention if $k = 1$ and non-atomic if $k \geq 2$. One can view observational data as a special case where $\boldsymbol{I} = \emptyset$. The reason for considering non-atomic interventions is to reduce the number of adaptive rounds required to recover $\mathcal{G}^*$ since potentially more edges may be separated by a single non-atomic intervention. Also, as real-world interventions may be costly, it is of practical importance to minimize the number of interventions required. Finally, an edge $U \to V$ is said to be cut by an intervention $\boldsymbol{S}$ if exactly one endpoint of the edge lies in $\boldsymbol{S}$, i.e. $|\boldsymbol{S} \cap \{U, V\}| = 1$. Fig. 2.5 provides an illustration of the above concepts.

Definition 2.51 (Separation of edges by interventions)**.** We say that an intervention $\boldsymbol{S} \subseteq \boldsymbol{V}$ *separates* a covered edge $U - V$ if $|\{U, V\} \cap \boldsymbol{S}| = 1$. That is, *exactly* one of the endpoints is intervened by $\boldsymbol{S}$. We say that an intervention set $\mathcal{I}$ separates a covered edge $U - V$ if there exists $\boldsymbol{S} \in \mathcal{I}$ that separates $U - V$.

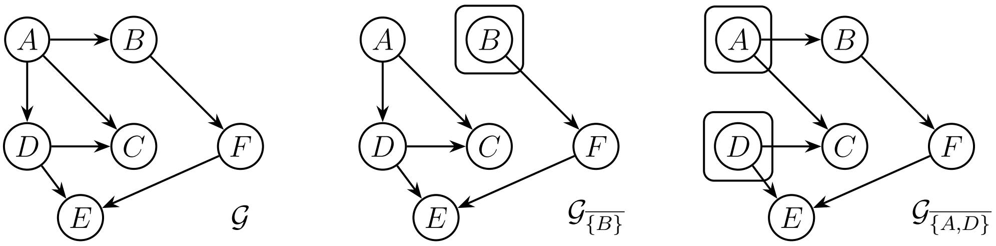

A
B
D
C
F
E
$\mathcal{G}$
A
B
D
C
F
E
$\mathcal{G}_{\overline{\{B\}}}$
A
B
D
C
F
E
$\mathcal{G}_{\overline{\{A,D\}}}$

Figure 2.5: A causal graph $\mathcal{G}$ over variables $\{A, B, C, D, E, F\}$ along with two interventional graphs under an atomic intervention set $\{B\}$ and non-atomic intervention set $\{A, D\}$. Observe that the covered edge $A \to D$ is *not* cut by the intervention $\{A, D\}$.

It is known that intervening on a set $\boldsymbol{S} \subseteq \boldsymbol{V}$ allows us to infer the edge orientation of any edge separated by $\boldsymbol{S}$ and $\boldsymbol{V} \setminus \boldsymbol{S}$ [Ebe07, HEH13, HLV14, SKDV15, KDV17]. An intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ is a collection of interventions and an $\mathcal{I}$-essential graph $\mathcal{E}_{\mathcal{I}}(\mathcal{G})$ of $\mathcal{G}$ is the essential graph representing the Markov equivalence class of graphs whose interventional graphs for each intervention is Markov equivalent to $\mathcal{G}_{\boldsymbol{S}}$ for any intervention $\boldsymbol{S} \in \mathcal{I}$. Interventions affect the joint distribution of the variables and are formally captured by Pearl's do-calulus [Pea09b]:

1. R1 (add/remove obs): If $(\boldsymbol{Y} \perp_d \boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{W})_{\mathcal{G}_{\overline{\boldsymbol{X}}}}$, then for all $\boldsymbol{x}'$, we have
$$\mathcal{P}(\boldsymbol{Y} \mid \mathrm{do}(\boldsymbol{x}'), \boldsymbol{Z}, \boldsymbol{W}) = \mathcal{P}(\boldsymbol{Y} \mid \mathrm{do}(\boldsymbol{x}'), \boldsymbol{W})$$

2. R2 (swap obs with do): If $(\boldsymbol{Y} \perp_d \boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{W})_{\mathcal{G}_{\overline{\boldsymbol{X}}\underline{\boldsymbol{Z}}}}$, then for all $\boldsymbol{x}', \boldsymbol{z}'$, we have
$$\mathcal{P}(\boldsymbol{Y} \mid \mathrm{do}(\boldsymbol{x}'), \mathrm{do}(\boldsymbol{z}'), \boldsymbol{W}) = \mathcal{P}(\boldsymbol{Y} \mid \mathrm{do}(\boldsymbol{x}'), \boldsymbol{z}', \boldsymbol{W})$$

3. R3 (add/remove do): If $(\boldsymbol{Y} \perp_d \boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{W})_{\mathcal{G}_{\overline{\boldsymbol{X}\boldsymbol{Z}(\boldsymbol{W})}}}$, then for all $\boldsymbol{x}', \boldsymbol{z}'$, we have
$$\mathcal{P}(\boldsymbol{Y} \mid \mathrm{do}(\boldsymbol{x}'), \mathrm{do}(\boldsymbol{z}'), \boldsymbol{W}) = \mathcal{P}(\boldsymbol{Y} \mid \mathrm{do}(\boldsymbol{x}'), \boldsymbol{W})$$
where $\boldsymbol{Z}(\boldsymbol{W}) = \boldsymbol{Z} \setminus \mathrm{An}(\boldsymbol{W})$ are $\boldsymbol{Z}$ nodes that are not ancestors of any $\boldsymbol{W}$ nodes in $\mathcal{G}_{\overline{\boldsymbol{X}}}$.

These rules were later generalized to allow for latent variables [Zha07, JRZB22].

There are several known properties about $\mathcal{I}$-essential graph properties [HB12, HB14, SMG+20]: Every $\mathcal{I}$-essential graph is a chain graph with chordal chain components. This includes the case of $\boldsymbol{S} = \emptyset$. Orientations in one chain component do not affect orientations in other components. In other words, to fully orient any essential graph $\mathcal{E}(\mathcal{G}^*)$, it is necessary and sufficient to orient every chain component in $\mathcal{E}(\mathcal{G}^*)$ independently. More formally, we have the following properties.

Lemma 2.52 (Proposition 15 of [HB12])**.** *Consider the $\mathcal{I}$-essential graph $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)$ of some DAG $\mathcal{G}^*$ and let $\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))$ be one of its chain components. Then, $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)$ is a chain graph and $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)[V(\mathcal{H})]$ is chordal.*

Lemma 2.53 (Modified lemma 1 of [HB14])**.** *Let $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ be an intervention set. Consider the $\mathcal{I}$-essential graph $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)$ of some DAG $\mathcal{G}^*$ and let $\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))$ be one of its chain components. Then, for any additional interventional set $\mathcal{I}' \subseteq 2^{\boldsymbol{V}}$ such that $\mathcal{I} \cap \mathcal{I}' = \emptyset$, we have*

$$\mathcal{E}_{\mathcal{I} \cup \mathcal{I}'}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})] = \mathcal{E}_{\{\boldsymbol{S} \cap \boldsymbol{V}(\mathcal{H})\,:\, \boldsymbol{S} \in \mathcal{I}'\}}(\mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]).$$

Lemma 1 of [HB14] actually considers a *single* additional intervention, but a closer look at their proof shows that the statement can be strengthened to allow for *multiple* additional interventions; see [CSB22, Appendix B]. Note that we can drop the $\emptyset$ intervention in the statement since essential graphs are defined with the observational data provided.

As a consequence of Lemma 2.53, one may assume without loss of generality that $CC(\mathcal{E}(\mathcal{G}^*))$ is a single connected component and then generalize results by summing across all connected components.

For any intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$, we write $\boldsymbol{R}(\mathcal{G}, \mathcal{I}) = \boldsymbol{A}(\mathcal{E}_{\mathcal{I}}(\mathcal{G}))$ to mean the set of oriented arcs in the $\mathcal{I}$-essential graph of a DAG $\mathcal{G}$. Under this notation, we see that the directed arcs in the partially directed graph $\mathcal{E}_{\mathcal{I}}(\mathcal{G})$ can be expressed as $\boldsymbol{A}(\mathcal{E}_{\mathcal{I}}(\mathcal{G})) = \boldsymbol{R}(\mathcal{G}, \mathcal{I})$. For cleaner notation, we write $\boldsymbol{R}(\mathcal{G}, \mathcal{I})$ for single interventions $\mathcal{I} = \{\boldsymbol{I}\}$ for some $\boldsymbol{I} \subseteq \boldsymbol{V}$, and $\boldsymbol{R}(\mathcal{G}, V)$ for single atomic interventions $\mathcal{I} = \{\{V\}\}$ for some $V \in \boldsymbol{V}$.

For any subset $\boldsymbol{S} \subseteq \boldsymbol{E}(\mathcal{G})$, we denote $\boldsymbol{R}(\mathcal{G}, \boldsymbol{S}) \subseteq \boldsymbol{E}(\mathcal{G})$ as the set of oriented arcs in the essential graph of $\mathcal{G}$ if we orient $\boldsymbol{S}$, along with the v-structure arcs in $\mathcal{G}$, then apply Meek rules till convergence. In particular, when $\boldsymbol{S} = \{(U, V) : U \in \boldsymbol{I} \text{ or } V \in \boldsymbol{I}\} \subseteq \boldsymbol{E}$ is the set of incident edges to some vertex set $\boldsymbol{I} \subseteq \boldsymbol{V}$, then $\boldsymbol{R}(\mathcal{G}, \boldsymbol{S}) = \boldsymbol{R}(\mathcal{G}, \boldsymbol{I})$ are precisely the oriented arcs in the interventional essential graph $\mathcal{E}_{\mathcal{I}}(\mathcal{G})$. Furthermore, if $\boldsymbol{S}$ is a *superset* of the set of incident edges to some vertex set $\boldsymbol{I} \subseteq \boldsymbol{V}$, then $\boldsymbol{R}(\mathcal{G}, \boldsymbol{I}) \subseteq \boldsymbol{R}(\mathcal{G}, \boldsymbol{S})$. When we use the $\boldsymbol{R}(\mathcal{G}, \cdot)$ notation, we will be explicit about its type – whether $\cdot$ is a subset of vertices $\boldsymbol{V}$, a subset of a subset of vertices $2^{\boldsymbol{V}}$, or a subset of edges $\boldsymbol{E}$.

The following lemma implies that the combined knowledge of two intervention sets do not further trigger any Meek rules. While [GSKB18] studies atomic interventions, their proof extends to non-atomic intervention sets, and even the observational case where the intervention set could be $\emptyset$.

Lemma 2.54 (Modified lemma 2 of [GSKB18])**.** *For any DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ and any two intervention sets $\mathcal{I}_1, \mathcal{I}_2 \subseteq 2^{\boldsymbol{V}}$, we have $\boldsymbol{R}(\mathcal{G}, \mathcal{I}_1 \cup \mathcal{I}_2) = \boldsymbol{R}(\mathcal{G}, \mathcal{I}_1) \cup \boldsymbol{R}(\mathcal{G}, \mathcal{I}_2)$.*

We define $\boldsymbol{R}_1^{-1}(\mathcal{G}, A \to B) \subseteq \boldsymbol{V}$ and $\boldsymbol{R}_k^{-1}(\mathcal{G}, A \to B) \subseteq 2^{\boldsymbol{V}}$ to refer to interventions orienting an arc $A \to B \in \boldsymbol{A}(\mathcal{G})$:

$$\boldsymbol{R}_1^{-1}(\mathcal{G}, A \to B) = \{V \in \boldsymbol{V} : A \to B \in \boldsymbol{R}(\mathcal{G}, V)\}$$
$$\boldsymbol{R}_k^{-1}(\mathcal{G}, A \to B) = \{\boldsymbol{I} \subseteq \boldsymbol{V} : |\boldsymbol{I}| \leq k, A \to B \in \boldsymbol{R}(\mathcal{G}, \boldsymbol{I})\}$$

For any oriented arc $A \to B \in \boldsymbol{A}(\mathcal{G})$, we define $\boldsymbol{R}_1^{-1}(\mathcal{G}, A \to B) = \boldsymbol{V}$ and $\boldsymbol{R}_k^{-1}(\mathcal{G}, A \to B) = \{\boldsymbol{I} \subseteq \boldsymbol{V} : |\boldsymbol{I}| \leq k\}$ as the set of interventions that would have oriented $A \to B$. For notational simplicity, we write $\boldsymbol{R}^{-1}$ to mean $\boldsymbol{R}_1^{-1}$.

Definition 2.55 (Oriented subgraphs and recovered parents)**.** For any interventional set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ and $U \in \boldsymbol{V}$, define $\mathcal{G}^{\mathcal{I}} = \mathcal{G}[\boldsymbol{E} \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{I})]$ as the *fully directed* subgraph DAG induced by the *unoriented arcs* in $\mathcal{G}$ and $\mathrm{Pa}_{\mathcal{G},\mathcal{I}}(U) = \{X \in \boldsymbol{V} : X \to U \in \boldsymbol{R}(\mathcal{G}, \mathcal{I})\}$ as the recovered parents of U by $\mathcal{I}$.

2.8.2 Common causal assumptions

Causal inference involves drawing conclusions about cause-and-effect relationships from data, often using statistical methods. Several key assumptions are commonly made in causal inference to ensure that these conclusions are valid. Below are the most common assumptions:

- Causal Markov Condition (CMC); see Definition 2.46

  The Causal Markov Condition states that a variable is conditionally independent of its non-effects, given its direct causes. In the causal DAG representation, this means that variables are conditionally independent of non-descendants given its parents. This assumption allows us to relate DAG graphical structure to probabilistic dependencies.

- Faithfulness; see Definition 2.50

  The faithfulness assumption asserts that the only conditional independencies that exist in the data are those implied by the causal DAG. In other words, if two variables are independent in the data, then there should be no direct or indirect causal relationship between them in the DAG. Faithfulness prevents coincidental cancellations of dependencies that might arise in specific parameter configurations which ensures that the statistical relationships observed in the data reflect the true causal structure.

- Causal sufficiency

  Causal sufficiency assumes that all common causes (confounders) of the variables being studied are measured and included in the analysis. This means there are no hidden variables that simultaneously affect multiple observed variables. This assumption simplifies the setting for causal inference problems. While some causal inference problems are completely resolved in the causal sufficiency setting, the problem of computing an optimal adaptive intervention policy for causal graph discovery (which we address in Chapter 6) was not.

- Positivity (Overlap)

  The positivity assumption (also known as overlap or common support) requires that for every combination of covariates, there is a positive probability of receiving each treatment. This means that all groups defined by covariates have a chance of being exposed to each level of the treatment. Without positivity, it would be impossible to make valid comparisons across treatment groups because some groups would have no representation in certain treatment categories. Positivity ensures that the causal effect is estimable for all subgroups.

- Stable Unit Treatment Value Assumption (SUTVA)

  SUTVA has two parts: (1) the treatment of one individual does not affect the outcome of another (no interference), and (2) there is only one version of each treatment (no hidden variations of the treatment). SUTVA is necessary to ensure that the treatment effect is well-defined and that the causal effect can be consistently estimated. Violations of SUTVA can lead to biased estimates and incorrect causal conclusions.

In Chapter 6, we additionally make the following two assumptions.

Assumption 2.56. We are given access to the essential graph (or equivalently we know the Markov equivalence class) of the true causal graph.

Assumption 2.57. We are able to determine orientations of edges that are separated by intervened vertices.

Assumption 2.56 is reasonable since there are a plethora of algorithms that recover the essential graph from observational data (which is abundant in many applications) under some standard causal assumptions such as those listed above, such as the PC [SGS00], FCI [SGS00] and RFCI algorithms [CMKR12]; see [GZS19, VCB22] for a survey. Meanwhile, Assumption 2.57 is always valid when we use hard or ideal interventions (which is the form of interventions we study in Chapter 6). Note that Assumption 2.57 may still hold with weaker forms of interventions (soft, imperfect, shift, etc) under additional conditions.

2.9 Learning-augmented algorithms

Given the widespread success and prevalence of machine-learned systems across various application domains, it is natural to ask whether the predictions generated by these systems can be used to improve the performance of classic algorithmic problems. For example, [KBC+18] examined the conditions under which machine-learned indexes outperform traditional index structures, demonstrating their advantages empirically. Similarly, [MNS12] proposed a black-box meta-algorithm that, assuming the availability of a highly competitive optimistic algorithm (potentially trained on historical data and problem instances), achieves a competitive ratio interpolating between that of the worst-case algorithm and the optimistic algorithm based on a given input interpolation parameter.

The field of learning-augmented algorithms formally studies how additional instance-specific predictions can used to enhance or guide traditional algorithmic processes. This approach combines the strengths of machine learning with classical algorithms to improve performance, adapt to new data, and solve complex problems more effectively. Learning-augmented algorithms as a whole have received significant attention since the seminal work of [LV21], which investigated the online caching problem with predictions; their result was further improved by [Roh20, Wei20, ACE+23]. Algorithms with advice were also studied for the ski-rental problem [GP19, WLW20, ADJ+20], non-clairvoyant scheduling [KPS18], scheduling [LLMV20, BMRS20, AJS22], augmenting classical data structures with predictions (e.g. indexing [KBC+18] and Bloom filters [Mit18]), online selection and matching problems [DLPLV21, AGKK23], online TSP [BLMS+22, GLS23], a more general framework of online primal-dual algorithms [BMS20], graph algorithms [CSVZ22, DIL+21], and mechanism design [GKST22, ABG+22].

The two main ways to evaluate a learning-augmented algorithm are the metrics of consistency and robustness. As this field mostly evolved from online algorithms, these metrics are first defined in terms of achievable competitive ratios[8]:

○ An algorithm is a-*consistent* if it is a-*competitive* with perfect advice

○ An algorithm is b-*robust* if it is b-*competitive* with arbitrary advice quality

Adapting the language of consistency and robustness to other settings such as query complexity, we see that the learning-augmented binary search described in Chapter 1 is 1-consistent and $\mathcal{O}(\log n)$-robust: 1 query suffices when the predicted page is correct while $\mathcal{O}(\log n)$ queries suffice in the worst case regardless of any given predicted page. We will later adapt this language of consistency and robustness to measure the number of adaptive interventions (in Chapter 11) and number of samples (in Chapter 10).

[8] Competitive analysis evaluates the performance of an online algorithm by comparing its objective to that of an optimal offline algorithm, which has complete knowledge of the entire input beforehand. While the online algorithm processes the input sequentially and must make irrevocable decisions without foresight, the offline algorithm can optimize its decisions with full knowledge of future inputs. The competitive ratio is defined as the ratio between the objective achieved by the online algorithm and the optimal objective achieved by the offline algorithm.

Part I

Learning probabilistic models

Chapter 3

Learning parameters of sparse linear Gaussian Bayesian networks

> "Frustra fit per plura quod potest fieri per pauciora."
> *("It is futile to do by more what can be done by fewer.")*
> - William of Ockham

3.1 Introduction

Linear structural equation models (SEMs) with additive Gaussian noise are widely used to model uncertainty in AI systems [Pea88]. A Gaussian Bayesian network can be described by linear structural models with additive Gaussian noise, a special case of structural equation models (SEMs) where variables have a linear relation with their parents' values in the presence of an additive Gaussian noise. For each $i \in [n]$, the variable X_i relates to its parents $\mathrm{Pa}(X_i)$ as follows:

$$X_i = \begin{cases} \eta_i + \sum_{X_j \in \mathrm{Pa}(X_i)} a_{i,j} X_j & \text{if } \mathrm{Pa}(X_i) \neq \emptyset \\ \eta_i & \text{if } \mathrm{Pa}(X_i) = \emptyset \end{cases} \tag{3.1}$$

where $\eta_i \sim N(0, \sigma_i^2)$ is a variable-specific independent Gaussian random variable. The scalars σ_i and $\{a_{i,j}\}_{j \in \mathrm{Pa}(X_i)}$ are the parameters associated to variable X_i in this model. Stacking the parameters into matrix form, we see that

$$\begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{pmatrix} = \begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \dots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n} \end{pmatrix} \begin{pmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{pmatrix} + \begin{pmatrix} \eta_1 \\ \eta_2 \\ \vdots \\ \eta_n \end{pmatrix}$$

where $a_{i,j} = 0$ whenever $j \notin \mathrm{Pa}(X_i)$. By expressing the above relation as $\boldsymbol{X} = \boldsymbol{A}\boldsymbol{X} + \boldsymbol{\eta}$, we see that $\boldsymbol{X} = (\boldsymbol{I}_n - \boldsymbol{A})^{-1}\boldsymbol{\eta}$ where $\boldsymbol{I}_n$ is the $n \times n$ identity matrix, and so $\boldsymbol{X}$ follows a multivariate Gaussian; see Section 2.4.1.

Recall from Chapter 2 that a Bayesian network corresponds to a pair $(\mathcal{P}, \mathcal{G})$ of distribution $\mathcal{P}$ and DAG structure $\mathcal{G}$. The question of structure learning (recovering $\mathcal{G}$) for Gaussian Bayesian networks has been extensively studied. A number of works have proposed increasingly general conditions for ensuring identifiability of the network structure from samples [PB14, GH17, CDW19, PK20, Par20, GDA20] and structure learning algorithms that work for high-dimensional Gaussian Bayesian networks have also been proposed [AZ15, AGZ19, GZ20].

In this chapter, we consider the task of learning a sparse linear Gaussian Bayesian network on n variables, *given its DAG structure* $\mathcal{G}$. The usual formulation of this problem is in terms of parameter estimation, where one wants a consistent estimator that exactly recovers the parameters of the Bayesian network in the limit, as the the number of samples approaches infinity. Parameter estimation has been well-studied in practice and maximum likelihood estimators are known for various simple settings such as when the conditional distribution is Gaussian or the variables are discrete-valued. For example, see the implementation of FIT in the R package `bnlearn` [Scu10].

In contrast to asymptotic parameter estimation, we consider the problem from the viewpoint of distribution learning [KMR+94] under the Probably Approximately Correct (PAC) learning model [Val84]. The goal here is to learn, with high probability, a distribution $\widehat{\mathcal{P}}$ that is close to the ground-truth distribution $\mathcal{P}$, using an efficient algorithm, i.e. $\mathrm{d}_{\mathrm{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$. In this setting, pointwise convergence of the parameters is no longer a requirement; the aim is rather to approximately learn the induced distribution. Indeed, the latter relaxed objective may be achievable when the former may not be (e.g. for ill-conditioned systems) and can be the more relevant requirement for downstream inference tasks. For a survey on the current state-of-the-art in distribution learning from an algorithmic perspective, see [Dia16].

While the definition of linear Gaussian SEMs from Eq. (3.1) may suggest an approach of viewing $\mathcal{P}$ as an n-dimensional multivariate Gaussian and estimating it from samples, we know from Section 2.4.1 that such an approach necessarily require $\Omega(n^2/\varepsilon^2)$ samples in general. As such, we focus on the setting where the structure of the network is *sparse*, whereby each variable has at most d parents.

3.2 Our main results

Theorem 3.1. *Let $\varepsilon, \delta \in (0,1)$ be the error and failure parameters respectively. Suppose $\mathcal{G}$ is a DAG on n variables $\{X_1, \ldots, X_n\}$, each with in-degree at most d, and the total degree is $d_{total} = \sum_{i=1}^{n} |\mathrm{Pa}(X_i)|$. Given the DAG $\mathcal{G}$ and sample access to distribution $\mathcal{P}$*

that is Markov with respect to $\mathcal{G}$, there is an algorithm that uses $\mathcal{O}\left(\frac{d_{total}}{\varepsilon^2} \log\left(\frac{n}{\delta}\right)\right)$ samples from $\mathcal{P}$ and produces a distribution $\widehat{\mathcal{P}}$ such that $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$. This algorithm runs in $\mathcal{O}\left(\frac{d_{total}^2 \cdot d}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right)$ time and succeeds with probability at least $1 - \delta$.

To complement our upper bound of $\widetilde{\mathcal{O}}(d_{total}/\varepsilon^2) \subseteq \widetilde{\mathcal{O}}(nd/\varepsilon^2)$, we also show that our sample complexity is nearly optimal in terms of the dependence on the parameters n, d, and ε as $\Omega(nd/\varepsilon^2)$ samples is unavoidable in general.

Theorem 3.2. *Let $\varepsilon \in (0, 1)$ be the error parameter. There exists a distribution $\mathcal{P}$ that is Markov with respect to a DAG $\mathcal{G}$ over n variables, each with in-degree at most $d \leq n/2$, such that producing a distribution $\widehat{\mathcal{P}}$ achieving $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ with success probability $2/3$ given $\mathcal{G}$ and sample access to $\mathcal{P}$ requires $\Omega(nd/\varepsilon^2)$ samples from $\mathcal{P}$.*

Observe that as $d \to n$, our sample complexity results recover the known bound of $\Theta(n^2/\varepsilon^2)$ for learning general n-dimensional multivariate Gaussians in Section 2.4.1.

3.3 Technical overview

Given the linear Gaussian Bayesian network with DAG $\mathcal{G}$ over variables $\{X_1, \ldots, X_n\}$, we know that X_i has parameters σ_i and $\{a_{i,j}\}_{j \in \mathrm{Pa}(X_i)}$, so producing an estimated distribution $\widehat{\mathcal{P}}$ can be done by providing estimates $\widehat{\sigma}_i$ and $\{\widehat{a}_{i,j}\}_{j \in \mathrm{Pa}(X_i)}$. We can represent $\{a_{i,j}\}_{j \in \mathrm{Pa}(X_i)}$ as a vector $\boldsymbol{a}_i \in \mathbb{R}^n$ where the j^{th} entry is $a_{i,j}$ for $j \in \mathrm{Pa}(X_i)$ and zero for $j \notin \mathrm{Pa}(X_i)$. Then, by grouping the parameters into $\boldsymbol{\alpha}_i = (\boldsymbol{a}_i, \sigma_i)$, we use $\boldsymbol{\alpha}_i^*$ to denote the ground truth parameters corresponding to $\mathcal{P}$ and $\widehat{\boldsymbol{\alpha}}_i$ as the estimated parameters for X_i.

3.3.1 Building blocks

We begin by stating two Gaussian concentration bounds. The derivation and proofs of these statements are deferred to Appendix A.1.2.

Lemma 3.3. *Let $\boldsymbol{G} \in \mathbb{R}^{k \times d}$ be a matrix with i.i.d. $N(0, 1)$ entries. Then, for any constant $0 < c_1 < 1/2$ and $k \geq d/c_1^2$,*

$$\Pr\left(\|(\boldsymbol{G}^\top \boldsymbol{G})^{-1}\| \leq \frac{1}{(1 - 2c_1)^2\, k}\right) \geq 1 - \exp\left(-\frac{kc_1^2}{2}\right)$$

Lemma 3.4. *Let $\boldsymbol{G} \in \mathbb{R}^{k \times p}$ be a matrix with i.i.d. $N(0, 1)$ entries and $\boldsymbol{\eta} \in \mathbb{R}^k$ be a vector with i.i.d. $N(0, \sigma^2)$ entries, where $\boldsymbol{G}$ and $\boldsymbol{\eta}$ are independent. Then, for any constant $c_2 > 0$,*

$$\Pr\left(\|\boldsymbol{G}^\top \boldsymbol{\eta}\| < 2\sigma c_2 \sqrt{kp}\right) \geq 1 - 2p \exp\left(-2k\right) - p \exp\left(-\frac{c_2^2}{2}\right)$$

Cauchy random variables arise naturally when studying Gaussians because $Z = X/Y$ is a Cauchy random variable when X and Y are two independent Gaussians. The next two lemmas provide results relating to Cauchy random variables and may be of independent interest beyond our analysis. Lemma 3.5 gives the non-asymptotic convergence of medians of Cauchy random variables and Lemma 3.6 gives a condition where a vector is term-wise Cauchy random variable.

Lemma 3.5 (Non-asymptotic convergence of Cauchy median)**.** *Consider a collection of m i.i.d.* $\mathrm{Cauchy}(0,1)$ *random variables $X_1, \ldots, X_m$. Given a threshold $0 < \tau < 1$, we have*

$$\Pr\left(\mathrm{median}\{X_1, \ldots, X_m\} \not\in [-\tau, \tau]\right) \leq 2\exp\left(-\frac{m\tau^2}{8}\right)$$

Lemma 3.6. *Consider the matrix equation $\boldsymbol{A}\boldsymbol{B} = \boldsymbol{E}$ where $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, $\boldsymbol{B} \in \mathbb{R}^{n \times 1}$, and $\boldsymbol{E} \in R^{n \times 1}$ such that entries of $\boldsymbol{A}$ and $\boldsymbol{E}$ are independent Gaussians, elements in each* column *of $\boldsymbol{A}$ have the same variance, and all entries in $\boldsymbol{E}$ have the same variance. That is, $\boldsymbol{A}_{\cdot,j} \sim N(0, \sigma_i^2)$ and $\boldsymbol{E}_i \sim N(0, \sigma_{n+1}^2)$. Then, for all $i \in [n]$, we have that $\boldsymbol{B}_i \sim \frac{\sigma_{n+1}}{\sigma_i} \cdot$* $\mathrm{Cauchy}(0,1)$.

3.3.2 Upper bound

For our upper bound (Theorem 3.1), we analyze the KL divergence between two distributions defined by parameters $\boldsymbol{\alpha}^*$ and $\widehat{\boldsymbol{\alpha}}$, and then apply Pinsker's inequality (Theorem 2.18) to obtain a corresponding bound on d_{TV}. The motivation behind such an approach is because one can decompose the KL divergence in terms of node-wise estimation error (Section 3.4). This enables one to design algorithms for recovering parameters on a per-node basis and then apply simple union bound arguments to conclude that the overall approach succeeds with the required success probability.

Now, consider an arbitrary variable $Y \in \{X_1, \ldots, X_n\}$ with p parents and associated parameters $\boldsymbol{a}^*$ and σ^*. If $p = 0$, then $\boldsymbol{a}^* = \boldsymbol{0}$ (the all-zero vector) and we can simply set the coefficients $\widehat{\boldsymbol{a}} = \boldsymbol{0}$. Meanwhile, if $p \geq 1$, we may assume w.l.o.g. that $X_1, \ldots, X_p$ are the parents of Y by relabeling. Let matrix $\boldsymbol{M} \in \mathbb{R}^{p \times p}$ denote the covariance matrix defined by the parents of Y, where the (i,j)-th entry of $\boldsymbol{M}$ is $\mathbb{E}[X_i X_j]$. Under this notation, we see the vector $(X_1, \ldots, X_p) \sim N(0, \boldsymbol{M})$ is distributed as a multivariate Gaussian. Let us further define $\boldsymbol{\Delta} = \widehat{\boldsymbol{a}} - \boldsymbol{a}^*$ as the entry-wise difference vector between the estimated coefficients and true coefficients. We later show that the set of parameters $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n, \widehat{\sigma}_1, \ldots, \widehat{\sigma}_n$ implies that $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ when the following two conditions hold for all $i \in [n]$.

$$\left|\boldsymbol{\Delta}_i^\top \boldsymbol{M}_i \boldsymbol{\Delta}_i\right| \leq (\sigma_i^*)^2 \cdot \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}} \qquad , \forall i \in [n] \qquad \text{(Condition 1)}$$

$$1 - \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}} \leq \left(\frac{\widehat{\sigma}_i}{\sigma_i^*}\right)^2 \leq 1 + \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}} \quad , \forall i \in [n] \qquad \text{(Condition 2)}$$

Roughly speaking, the first condition requires that the estimation error in the cofficients $|(\widehat{\boldsymbol{a}}_i - \boldsymbol{a}_i)^\top \boldsymbol{M}_i (\widehat{\boldsymbol{a}}_i - \boldsymbol{a}_i)| = |\boldsymbol{\Delta}_i^\top \boldsymbol{M}_i \boldsymbol{\Delta}_i|$ is "bounded relative to σ_i^*" while the second condition requires that $\widehat{\sigma}_i$ is a "good multiplicative approximation" of σ_i^*. Note that Condition 1 is trivially satisfied when $\mathrm{Pa}(X_i) = \emptyset$.

Motivated by the above insights, we will estimate parameters for each variable in an independent fashion. That is, given the samples, the parameters related to each variable can be estimated in parallel. Our algorithmic approach is a two-phased one; see Algorithm 1. In the first phase, we aim to produce estimates $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n$ such that Condition 1 is satisfied. In the second phase, we use the estimates $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n$ to produce estimates $\widehat{\sigma}_1^2, \ldots, \widehat{\sigma}_n^2$ such that Condition 2 is satisfied.

Algorithm 1 Two-phased recovery algorithm

```
1: **Input**: Sample access to $\mathcal{P}$, DAG $\mathcal{G}$, and sample parameters $m_1$ and $m_2$
2: Draw $m = m_1 + m_2$ independent samples of $(X_1, \ldots, X_n)$ from $\mathcal{P}$.
3: $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n \leftarrow$ Run a coefficient recovery algorithm using first $m_1$ samples.
4: $\widehat{\sigma}_1^2, \ldots, \widehat{\sigma}_n^2 \leftarrow$ Run a variance recovery algorithm using last $m_2$ samples and $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n$
5: **return** $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n, \widehat{\sigma}_1^2, \ldots, \widehat{\sigma}_n^2$
```

In Section 3.5, we show that empirical variance given the produced $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n$ estimates suffice, and then provide two different classes of methods for coefficient recovery. In Section 3.6, we analyze algorithms based on the maximum likelihood estimator linear least squares regression and provide explicit sample complexity bounds. Meanwhile, in Section 3.7, we develop and analyze an alternative algorithm based on Cauchy random variables, which is a uncommon in the context of regression and statistical learning.

3.3.3 Lower bound

Consider the graph construction in shown in Fig. 3.1. Here, the DAG $\mathcal{G}$ is bipartite with maximum in-degree d. For $j \in \{1, \ldots, d\}$, each variable $X_j = \eta_j$ is distributed according to a standard Gaussian random variable $\eta_j \sim N(0, 1)$. For $i \in \{d+1, \ldots, n\}$, each variable $X_i = \eta_i + \sum_{j=1}^d a_{i,j} X_j$ has all $\{X_1, \ldots, X_d\}$ as parents and $\eta_i \sim N(0, 1)$. We also associate a d-bit binary string to each variable X_i such that the coefficient

$$a_{i,j} = \begin{cases} \frac{1}{\sqrt{d(n-d)}} & \text{if the } j^{th} \text{ bit of the binary string is 0} \\ \frac{1+\varepsilon}{\sqrt{d(n-d)}} & \text{if the } j^{th} \text{ bit of the binary string is 1} \end{cases} \tag{3.2}$$

Now, let $\boldsymbol{s} \in \{0, 1\}^{d(n-d)}$ be a collection of $(n-d)$ binary strings of length d. In the above setup, with $a_{j,1}^{(\boldsymbol{s})}, \ldots, a_{j,d}^{(\boldsymbol{s})}$ determined by the j^{th} consecutive d bits of $\boldsymbol{s}$ via Eq. (3.2),

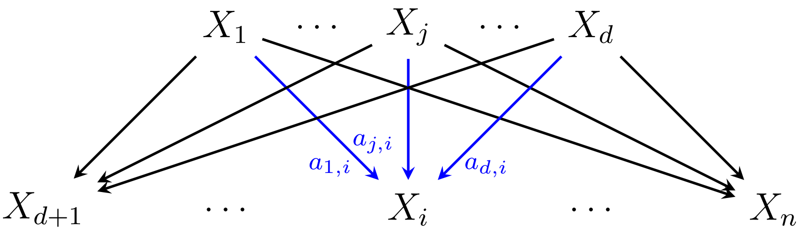


Figure 3.1: Hardness construction: complete bipartite DAG with maximum in-degree d.

we can define an induced conditional distribution $\mathcal{Q}_{\boldsymbol{s}}$ on $\{X_1, \ldots, X_n\}$. Through the same decomposition that we derived in Section 3.4, one can show that

$$\begin{aligned} \mathrm{d}_{\mathrm{KL}}(\mathcal{Q}_{\boldsymbol{s}}, \mathcal{Q}_{\boldsymbol{s}'}) &= \frac{1}{2} \sum_{i=d+1}^{n} \sum_{j=1}^{d} (a_{j,i}^{(\boldsymbol{s})} - a_{j,i}^{(\boldsymbol{s}')})^\top (a_{j,i}^{(\boldsymbol{s})} - a_{j,i}^{(\boldsymbol{s}')}) \\ &= \frac{1}{2} \sum_{i=d+1}^{n} \sum_{j=1}^{d} \left(\frac{\varepsilon}{\sqrt{d(n-d)}} \cdot \mathbb{1}_{\boldsymbol{s}_{i,j} \neq \boldsymbol{s}'_{i,j}} \right)^2 \\ &= \frac{\varepsilon^2}{2d(n-d)} \mathrm{d}_{\mathrm{hamm}}(\boldsymbol{s}, \boldsymbol{s}') \end{aligned}$$

for any two distributions $\mathcal{Q}_{\boldsymbol{a}}$ and $\mathcal{Q}_{\boldsymbol{b}}$ induced by binary strings $\boldsymbol{a}$ and $\boldsymbol{a}'$ respectively, where $\mathrm{d}_{\mathrm{hamm}}$ refers to the Hamming distance between two binary strings.

To obtain our lower bound (Theorem 3.2), we consider the subset $\mathcal{C}$ of length $d(n-d)$ binary strings such that $\mathrm{d}_{\mathrm{hamm}}(\boldsymbol{s}, \boldsymbol{s}') = d(n-d)/2$ for distinct strings $\boldsymbol{s}, \boldsymbol{s}' \in \mathcal{C}$. One can show that $|\mathcal{C}| \geq 2^{\Omega(d(n-d))}$, $\mathrm{d}_{\mathrm{KL}}(\mathcal{Q}_{\boldsymbol{a}}, \mathcal{Q}_{\boldsymbol{b}}) \in \mathcal{O}(\varepsilon^2)$, and $\mathrm{d}_{\mathrm{TV}}(\mathcal{Q}_{\boldsymbol{a}}, \mathcal{Q}_{\boldsymbol{b}}) \in \Omega(\varepsilon)$. So, if we were to uniformly choose a binary string $\boldsymbol{s} \in \mathcal{C}$ and use its induced distribution as $\mathcal{P} = \mathcal{Q}_{\boldsymbol{s}}$, requiring $\mathrm{d}_{\mathrm{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ implies that one has to correctly identify $\boldsymbol{s}$ amongst $\mathcal{C}$. When $d \leq n/2$, one can then conclude that $\Omega(d(n-d)/\varepsilon^2) \subseteq \Omega(nd/\varepsilon^2)$ samples by using a packing argument based on Fano's inequality (Theorem 2.23).

For a detailed proof of Theorem 3.2, we refer readers to [BCG$^+$22, Section 5].

3.4 Decomposing the KL divergence

Our analysis relies on decomposing and bounding the KL divergence between $\mathcal{P}$ and $\widehat{\mathcal{P}}$, and then applying Pinsker's inequality (see Theorem 2.18) to obtain a bound on the TV distance between $\mathcal{P}$ and $\widehat{\mathcal{P}}$.

Following the approach of [Das97][9], we decompose $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \widehat{\mathcal{P}})$ into n terms that can be computed by analyzing the quality of recovered parameters for each variable X_i. To relate the overall KL distance between two sets of parameters $\boldsymbol{\alpha}_i^*$ and $\widehat{\boldsymbol{\alpha}}_i$, we first define a

[9] [Das97] analyzes the *non-realizable* setting where the distribution $\mathcal{P}$ may not correspond to the causal structure of the given Bayesian network. As we study the *realizable* setting, we have a much simpler derivation.

distance measure $\mathrm{d_{CP}}$ between the conditional probabilities on a per-node basis:

$$\mathrm{d_{CP}}(\boldsymbol{\alpha}_i^*, \widehat{\boldsymbol{\alpha}}_i) = \int_{\mathrm{pa}(X_i)} \int_{x_i} \mathcal{P}(x_i, \mathrm{pa}(X_i)) \log\left(\frac{\mathcal{P}(x_i \mid \mathrm{pa}(X_i))}{\mathcal{Q}(x_i \mid \mathrm{pa}(X_i))}\right) dx_i \, d\mathrm{pa}(X_i)$$

Then, if $\widehat{\mathcal{P}}$ is the distribution defined by parameters $\widehat{\boldsymbol{\alpha}}$ on the same Bayesian network structure $\mathcal{G}$, one can check that the Bayesian network decomposition of joint probabilities and marginalization yields

$$\mathrm{d_{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) = \sum_{i=1}^{n} \mathrm{d_{CP}}(\boldsymbol{\alpha}_i^*, \widehat{\boldsymbol{\alpha}}_i) \ . \tag{3.3}$$

Now, consider an arbitrary variable $Y \in \{X_1, \ldots, X_n\}$ with p parents and associated parameters $\boldsymbol{a}^*$ and σ^*. If $p = 0$, then $\boldsymbol{a}^* = \mathbf{0}$ (the all-zero vector) and we can simply set the coefficients $\widehat{\boldsymbol{a}} = \mathbf{0}$. Meanwhile, if $p \geq 1$, we may assume w.l.o.g. that $X_1, \ldots, X_p$ are the parents of Y by relabeling. Let matrix $\boldsymbol{M} \in \mathbb{R}^{p \times p}$ denote the covariance matrix defined by the parents of Y, where the (i,j)-th entry of $\boldsymbol{M}$ is $\mathbb{E}[X_i X_j]$. Under this notation, we see the vector $(X_1, \ldots, X_p) \sim N(0, \boldsymbol{M})$ is distributed as a multivariate Gaussian. Let us further define $\boldsymbol{\Delta} = \widehat{\boldsymbol{a}} - \boldsymbol{a}^*$ as the entry-wise difference vector. Then, one can show that

$$\mathrm{d_{CP}}(\boldsymbol{\alpha}_i^*, \widehat{\boldsymbol{\alpha}}_i) = \ln\left(\frac{\widehat{\sigma}_i}{\sigma_i^*}\right) + \frac{(\sigma_i^*)^2 - \widehat{\sigma}_i^2}{2\widehat{\sigma}_i^2} + \frac{\boldsymbol{\Delta}_i^\top \boldsymbol{M}_i \boldsymbol{\Delta}_i}{2\widehat{\sigma}_i^2} \qquad , \forall i \in [n] \tag{3.4}$$

For full derivation details of Eq. (3.3) and Eq. (3.4), see Appendix A.1.1.

As foreshadowed in Section 3.3, the following lemma uses the above decomposition to conclude that $\mathrm{d_{CP}}(\boldsymbol{\alpha}_i^*, \widehat{\boldsymbol{\alpha}}_i)$ is small enough to imply that $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ when Condition 1 and Condition 2 hold on $\boldsymbol{\alpha}_i^*$ and $\widehat{\boldsymbol{\alpha}}_i$, for all $i \in [n]$.

Lemma 3.7. *Let $\varepsilon \leq 0.17$ be a constant. Suppose distributions $\mathcal{P}$ and $\widehat{\mathcal{P}}$ are defined on a Bayesian network $\mathcal{G}$ with parameters $\boldsymbol{\alpha}^*$ and $\widehat{\boldsymbol{\alpha}}$. If Condition 1 and Condition 2 hold on $\boldsymbol{\alpha}_i^*$ and $\widehat{\boldsymbol{\alpha}}_i$ for all $i \in [n]$, then $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \sqrt{3}\varepsilon$.*

Proof. Let us denote the total in-degree of all variables by $d_{total} = \sum_{i=1}^{n} |\mathrm{Pa}(X_i)|$. We begin the proof by observing the following inequality, which is also used in [ABDH+20, Lemma 2.9].

$$\gamma - 1 - \ln(\gamma) \leq (\gamma - 1)^2 \qquad \text{for } \gamma \geq 0.316\ldots \tag{3.5}$$

Consider an arbitrary fixed $i \in [n]$. Since $|\mathrm{Pa}(X_i)| \leq d_{total}$, Condition 2 implies that

$$\left(\frac{\sigma_i^*}{\widehat{\sigma}_i}\right)^2 \geq \frac{1}{1 + \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}}} \geq \frac{1}{1 + \sqrt{\varepsilon}}$$

For $\varepsilon \leq 0.17$, one can check that $\frac{1}{1+\sqrt{\varepsilon}} \geq 0.5$, and so Eq. (3.5) applies to $\left(\frac{\sigma_i^*}{\widehat{\sigma}_i}\right)^2$.

Furthermore, we will have $0 \leq \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}} \leq \varepsilon \leq \frac{1}{4}$.

So,

$$\begin{aligned}
&\ln\left(\frac{\widehat{\sigma}_i}{\sigma_i^*}\right) + \frac{(\sigma_i^*)^2 - \widehat{\sigma}_i^2}{2\widehat{\sigma}_i^2} \\
&= \frac{1}{2} \cdot \left(\left(\frac{\sigma_i^*}{\widehat{\sigma}_i}\right)^2 - 1 - \ln\left(\left(\frac{\sigma_i^*}{\widehat{\sigma}_i}\right)^2\right)\right) \\
&\leq \frac{1}{2} \cdot \left(\left(\frac{\sigma_i^*}{\widehat{\sigma}_i}\right)^2 - 1\right)^2 && \text{By Eq. (3.5)} \\
&\leq \frac{1}{2} \cdot \left(\frac{1}{1 - \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}}} - 1\right)^2 && \text{By Condition 2} \\
&\leq \frac{2\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}} && \text{Holds when } 0 \leq \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}} \leq \frac{1}{4}
\end{aligned}$$

Meanwhile,

$$\begin{aligned}
\frac{\boldsymbol{\Delta}_i^\top \boldsymbol{M}_i \boldsymbol{\Delta}_i}{2\widehat{\sigma}_i^2} &\leq \frac{|\boldsymbol{\Delta}_i^\top \boldsymbol{M}_i \boldsymbol{\Delta}_i|}{2\widehat{\sigma}_i^2} \\
&\leq \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{2d_{total}} \cdot \left(\frac{\sigma_i^*}{\widehat{\sigma}_i}\right)^2 && \text{By Condition 1} \\
&\leq \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{2d_{total}} \cdot \frac{1}{1 - \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}}} && \text{By Condition 2} \\
&\leq \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}} && \text{Holds when } 0 \leq \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}} \leq \frac{1}{4}
\end{aligned}$$

Putting together, we see that $d_{CP}(\alpha_i^*, \widehat{\alpha}_i) \leq \frac{3\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}$. The claim then holds by invoking the decompositions of Eq. (3.3) and Eq. (3.4):

$$\mathrm{d_{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) = \sum_{i=1}^{n} \mathrm{d_{CP}}(\boldsymbol{\alpha}_i^*, \widehat{\boldsymbol{\alpha}}_i) \leq \sum_{i=1}^{n} \frac{3\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}} = 3\varepsilon$$

Finally, we can apply Theorem 2.18 to conclude that $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \sqrt{3\varepsilon^2} = \sqrt{3}\varepsilon$. □

Remark 3.8. Showing $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \sqrt{3}\varepsilon$ in Lemma 3.7 is qualitatively the same as showing $\mathrm{d_{TV}}(\mathcal{P}, \mathcal{Q}) \leq \varepsilon$ since one can repeat the entire analysis above with a smaller error $\varepsilon' = \varepsilon/\sqrt{3}$.

3.5 Variance recovery

As discussed in Section 3.3, the computing empirical variance suffices to obtain estimates $\widehat{\sigma}_1^2, \ldots, \widehat{\sigma}_n^2$ satisfying Condition 2 when given coefficient estimates satisfying Condition 1.

We formalize this in VarianceRecovery (Algorithm 2). Note that the algorithm only uses *one* batch of samples for all the nodes. This is possible as we can obtain high-probability bounds on the error events at each node.

Algorithm 2 VarianceRecovery: Variance recovery algorithm

```
Input: DAG $\mathcal{G}$, coefficient estimates, and $m_2 \in \mathcal{O}\left(\frac{d_{total}}{\varepsilon^2} \log\left(\frac{n}{\delta}\right)\right)$ samples
for variable $Y$ with coefficient estimate $\widehat{\boldsymbol{a}}$ do        ▷ If $|\mathrm{Pa}(Y)| = 0$, then $\widehat{\boldsymbol{a}} = \mathbf{0}$.
    W.l.o.g., by renaming variables, let $X_1, \ldots, X_{|\mathrm{Pa}(Y)|}$ be the parents of $Y$.
    for $s = 1, \ldots, m_2$ do
        Define $y^{(s)}$ and $x_i^{(s)}$ as the $s^{th}$ sample of $Y$ and $X_i$, for $i \in \{1, \ldots, |\mathrm{Pa}(Y)|\}$.
        Define $z^{(s)} = \left(y^{(s)} - \langle x^{(s)}, \widehat{\boldsymbol{a}} \rangle\right)^2$, where $x^{(s)} = \left(x_1^{(s)}, \ldots, x_{|\mathrm{Pa}(Y)|}^{(s)}\right)$.
    Estimate $\widehat{\sigma}^2 = \frac{1}{m_2} \sum_{s=1}^{m_2} z^{(s)}$
return $\widehat{\sigma}_1^2, \ldots, \widehat{\sigma}_n^2$
```

To analyze VarianceRecovery, we first prove guarantees for an arbitrary variable $Y \in \{X_1, \ldots, X_n\}$ and then take union bound over all n variables. When $\mathrm{Pa}(Y) = \emptyset$, $\widehat{\boldsymbol{a}} = \mathbf{0}$ and $\widehat{\sigma}^2 = \frac{1}{m_2} \sum_{s=1}^{m_2} (y^{(s)})^2$ is distributed according to $(\sigma^*)^2 \cdot \chi^2_{m_2}$. Meanwhile, when $\mathrm{Pa}(Y) \neq \emptyset$, one can show that $\widehat{\sigma}^2 \sim \left((\sigma^*)^2 + \boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}\right) \cdot \chi^2_{m_2}$. In either case, we can apply standard concentration bounds for χ^2 random variables (see Lemma 2.31) to argue that VarianceRecovery produces estimates $\widehat{\sigma}_1^2, \ldots, \widehat{\sigma}_n^2$ that satisfy Condition 2. The next lemma formalizes this. Note that the proof for the $\mathrm{Pa}(Y) \neq \emptyset$ case relies on the given coefficients $\boldsymbol{a}$ satisfying Condition 1, i.e. $|\boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}|$ is bounded relative to $(\sigma^*)^2$.

Theorem 3.9 (Guarantees of VarianceRecovery)**.** *Suppose $0 \leq \varepsilon \leq 3 - 2\sqrt{2} \leq 0.17$ and coefficient estimates $\widehat{\boldsymbol{a}}_i$ satisfies Condition 1 for all $i \in [n]$. With $\mathcal{O}\left(\frac{d_{total}}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$ samples, the VarianceRecovery algorithm recovers $\widehat{\sigma}_1^2, \ldots, \widehat{\sigma}_n^2$ such that*

$$\Pr\left(\forall i \in [n], \quad 1 - \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}} \leq \left(\frac{\widehat{\sigma}_i}{\sigma_i^*}\right)^2 \leq 1 + \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}}\right) \geq 1 - \delta$$

The total running time is $\mathcal{O}\left(\frac{d_{total}^2}{\varepsilon} \log\left(\frac{1}{\delta}\right)\right)$.

Proof. Fix any arbitrary variable $Y \in \{X_1, \ldots, X_n\}$ with parameters $(\boldsymbol{a}^*, \sigma^*)$, associated noise variable η, and associated covariance matrix $\boldsymbol{M}$. It suffices to show that with $\mathcal{O}\left(\frac{d_{total}}{\varepsilon \cdot |\mathrm{Pa}(Y)|} \log\left(\frac{1}{\delta}\right)\right)$ samples, the VarianceRecovery algorithm recovers $\widehat{\sigma}$ such that

$$\Pr\left(1 - \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(Y)|}{d_{total}}} \leq \left(\frac{\widehat{\sigma}}{\sigma^*}\right)^2 \leq 1 + \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(Y)|}{d_{total}}}\right) \geq 1 - \delta \qquad (3.6)$$

Then, for each $i \in [n]$, we apply Eq. (3.6) with $\delta' = \delta / n$ and $m \in \mathcal{O}\left(\frac{d_{total}}{\varepsilon} \log\left(\frac{1}{\delta'}\right)\right)$, then take the union bound over all n variables. The computational complexity for a variable with p parents is $\mathcal{O}(mp)$ and so the total runtime is $\mathcal{O}(md_{total})$ since $\sum_{i=1}^{n} p_i = d_{total}$. In the rest of this proof, we will establish Eq. (3.6) as discussed.

Suppose we drew $k \in \mathcal{O}\left(\frac{d_{total}}{\varepsilon \cdot |\mathrm{Pa}(Y)|} \log\left(\frac{1}{\delta}\right)\right) \subseteq \mathcal{O}\left(\frac{d_{total}}{\varepsilon} \log\left(\frac{1}{\delta}\right)\right)$ samples and estimated $\widehat{\sigma} = \frac{1}{k}\sum_{s=1}^{k} z^{(s)} = \frac{1}{k}\sum_{s=1}^{k}(y^{(s)} - \boldsymbol{x}^{(s)}\widehat{\boldsymbol{a}})^2$ as per VarianceRecovery. We will now argue that $\widehat{\sigma}^2 \sim \frac{(\sigma^*)^2 + \boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta}}{k} \cdot \chi_k^2$, then apply standard concentration bounds for χ^2 random variables; see Lemma 2.31. For any sample $s \in [k]$, we see that

$$y^{(s)} - \boldsymbol{x}^{(s)}\widehat{\boldsymbol{a}} = \boldsymbol{a}^{(s)}\boldsymbol{a} + \eta^{(s)} - \boldsymbol{x}^{(s)}\widehat{\boldsymbol{a}} = \eta^{(s)} - \boldsymbol{a}^{(s)}\boldsymbol{\Delta}\ ,$$

where $\boldsymbol{\Delta} = \widehat{\boldsymbol{a}} - \boldsymbol{a} \in \mathbb{R}^p$ is an unknown constant vector (because we do not actually know $\boldsymbol{a}$). For fixed $\boldsymbol{\Delta}$, we see that $\boldsymbol{x}^{(s)}\boldsymbol{\Delta} \sim N(0, \boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta})$. Since $\eta^{(s)} \sim N(0, (\sigma^*)^2)$ and $\boldsymbol{x}^{(s)}$ are independent, we have that $y^{(s)} - \boldsymbol{x}^{(s)}\widehat{\boldsymbol{a}} \sim N(0, (\sigma^*)^2 + \boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta})$. So, for any sample $s \in [k]$, $z^{(s)} = (y^{(s)} - \boldsymbol{x}^{(s)}\widehat{\boldsymbol{a}})^2 \sim \left((\sigma^*)^2 + \boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta}\right) \cdot \chi_1^2$. Therefore, $\widehat{\sigma} = \frac{1}{k}\sum_{s=1}^{k} z^{(s)} \sim \frac{(\sigma^*)^2 + \boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta}}{k} \cdot \chi_k^2$ as desired. Now, let us define

$$\gamma = \left(\frac{\widehat{\sigma}}{\sigma^*}\right)^2 \cdot \left(\frac{1}{1 + \frac{\boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta}}{(\sigma^*)^2}}\right) \sim \frac{\chi_k^2}{k}$$

Since $p \le d_{total}$, if $\varepsilon \le 3 - 2\sqrt{2}$, then $\frac{\varepsilon p}{d_{total}} \le 3 - 2\sqrt{2} \le 3 + 2\sqrt{2}$. We first make two observations:

1. For $0 \le \frac{\varepsilon p}{d_{total}} \le 3 - 2\sqrt{2}$, $\left(1 + \sqrt{\frac{\varepsilon p}{d_{total}}}\right) \cdot \left(\frac{1}{1 + \frac{\boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta}}{(\sigma^*)^2}}\right) \ge 1 + \sqrt{\frac{\varepsilon p}{4d_{total}}}$
2. For $0 \le \frac{\varepsilon p}{d_{total}} \le 3 + 2\sqrt{2}$, $\left(1 - \sqrt{\frac{\varepsilon p}{d_{total}}}\right) \cdot \left(\frac{1}{1 + \frac{\boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta}}{(\sigma^*)^2}}\right) \le 1 - \sqrt{\frac{\varepsilon p}{4d_{total}}}$

Using Lemma 2.31 with the above discussion, we have

$$\begin{aligned}
&\Pr\left(\left(\frac{\widehat{\sigma}}{\sigma^*}\right)^2 \ge 1 + \sqrt{\frac{\varepsilon p}{d_{total}}} \vee \left(\frac{\widehat{\sigma}}{\sigma^*}\right)^2 \le 1 - \sqrt{\frac{\varepsilon p}{d_{total}}}\right) \\
&= \Pr\left(\gamma \ge \left(\frac{1 + \sqrt{\frac{\varepsilon p}{d_{total}}}}{1 + \frac{\boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta}}{(\sigma^*)^2}}\right) \vee \gamma \le \left(\frac{1 - \sqrt{\frac{\varepsilon p}{d_{total}}}}{1 + \frac{\boldsymbol{\Delta}^\top \boldsymbol{M}\boldsymbol{\Delta}}{(\sigma^*)^2}}\right)\right) \\
&\le \Pr\left(\gamma \ge 1 + \sqrt{\frac{\varepsilon p}{4d_{total}}} \vee \gamma \le 1 - \sqrt{\frac{\varepsilon p}{4d_{total}}}\right) \\
&= \Pr\left(|\gamma - 1| \ge \sqrt{\frac{\varepsilon p}{4d_{total}}}\right) \\
&\le 2\exp\left(-\frac{k\varepsilon p}{32 d_{total}}\right) \\
&\le \delta && \text{(By definition of } k\text{)}
\end{aligned}$$

This establishes Eq. (3.6) as desired. □

3.6 Coefficient recovery based on linear least squares

In this section, we provide an algorithm LeastSquares for recovering the coefficients in a Bayesian network using linear least squares. As discussed in Section 3.3, we will recover the coefficients for each variable such that Condition 1 is satisfied. To this end, let us consider an arbitrary variable $Y \in \{X_1, \ldots, X_n\}$ with p parents. If $p = 0$, we simply set the coefficients $\widehat{\boldsymbol{a}} = \mathbf{0}$ and observe that Condition 1 is trivially satisfied since $\boldsymbol{\Delta} = \widehat{\boldsymbol{a}} - \boldsymbol{a}^* = \mathbf{0} - \mathbf{0} = \mathbf{0}$. Meanwhile, if $p \geq 1$, we may assume w.l.o.g. (by relabeling) that $X_1, \ldots, X_p$ are the parents of Y and proceed to estimate the coefficients $\widehat{\boldsymbol{a}}$ using independent samples from $\mathcal{P}$.

Using m_1 independent samples, we form matrix $\boldsymbol{X} \in \mathbb{R}^{m_1 \times p}$ using m_1 independent samples, where the r^{th} row consists of sample values $x_1^{(r)}, \ldots, x_p^{(r)}$, and the column vector $\boldsymbol{B} = (y^{(1)}, \ldots, y^{(m_1)})^\top \in \mathbb{R}^{m_1}$. Then, we define $\widehat{\boldsymbol{a}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{B}$ as the solution to the least squares problem $\boldsymbol{X}\widehat{\boldsymbol{a}} = \boldsymbol{B}$.

Algorithm 3 LeastSquares: Coefficient recovery algorithm

```
1: Input: DAG $\mathcal{G}$ and $m_1 \in \mathcal{O}\left(\frac{d_{total}}{\varepsilon} \cdot \ln\left(\frac{n}{\delta}\right)\right)$ samples
2: for variable $Y$ with $p \geq 1$ parents $X_1, \ldots, X_p$ do
3:     Form matrix $\boldsymbol{X} \in \mathbb{R}^{m_1 \times p}$, where the $r^{th}$ row consists of samples $(x_1^{(r)}, \ldots, x_p^{(r)})$
4:     Form column vector $\boldsymbol{B} = (y^{(1)}, \ldots, y^{(m_1)})^\top \in \mathbb{R}^{m_1}$
5:     Define $\widehat{\boldsymbol{a}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{B}$ as the solution to the least squares problem $\boldsymbol{X}\widehat{\boldsymbol{a}} = \boldsymbol{B}$
6: return $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n$
```

Theorem 3.10 (Distribution learning using LeastSquares). *Let $\varepsilon, \delta \in (0, 1)$. Suppose $\mathcal{G}$ is a known DAG on n variables with in-degree at most d and we have sample access to distribution $\mathcal{P}$ that is Markov with respect to $\mathcal{G}$. Given $\mathcal{O}\left(\frac{d_{total}}{\varepsilon^2} \log\left(\frac{n}{\delta}\right)\right)$ samples from $\mathcal{P}$, by using LeastSquares for coefficient recovery in Algorithm 1, one can produce a distribution $\widehat{\mathcal{P}}$ such that $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$. The time complexity is $\mathcal{O}\left(\frac{d_{total}^2 \cdot d}{\varepsilon^2} \log\left(\frac{1}{\delta}\right)\right)$ and the success probaility is at least $1 - \delta$.*

Observe that as $d \to n$, the sample complexity bound of Theorem 3.10 recovers the known bound of $\Theta(n^2/\varepsilon^2)$ for learning general n-dimensional multivariate Gaussians discussed in Section 2.4.1.

Our analysis begins by proving guarantees for an arbitrary variable.

Lemma 3.11. *Fix an arbitrary variable Y with p parents, parameters $(\boldsymbol{a}^*, \sigma^*)$, and associated covariance matrix $\boldsymbol{M}$. With $k \geq \frac{4c_2^2}{(1-c_1)^4} \cdot \frac{d_{total}}{\varepsilon}$ samples, for any constants $0 < c_1 < 1/2$ and $c_2 > 0$, LeastSquares recovers the coefficients $\widehat{\boldsymbol{a}}$ such that*

$$\Pr\left(|\boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}| \geq (\sigma^*)^2 \cdot \frac{\varepsilon \cdot p}{d_{total}}\right) \leq \exp\left(-\frac{kc_1^2}{2}\right) + 2p \exp\left(-2k\right) + p \exp\left(-\frac{c_2^2}{2}\right)$$

Proof. Let $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^\top$ be the Cholesky decomposition of $\boldsymbol{M}$ via the lower triangular matrix $\boldsymbol{L}$. Since $|\boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}| = |\boldsymbol{\Delta}^\top \boldsymbol{L}\boldsymbol{L}^\top \boldsymbol{\Delta}| = \|\boldsymbol{L}^\top \boldsymbol{\Delta}\|^2$, it suffices to bound $\|\boldsymbol{L}^\top \boldsymbol{\Delta}\|$.

W.l.o.g., Y has additive Gaussian noise variable η and the parents $X_1, \ldots, X_p$ by relabeling. Define $\boldsymbol{X} \in \mathbb{R}^{k \times p}$, $\boldsymbol{B} \in \mathbb{R}^k$, and $\widehat{\boldsymbol{a}} \in \mathbb{R}^p$ as in LeastSquares. Let $\boldsymbol{\eta} = (\eta^{(1)}, \ldots, \eta^{(k)}) \in \mathbb{R}^k$ be the instantiations of Gaussian η in the k samples. By the structural equations, we know that $\boldsymbol{B} = \boldsymbol{X}\boldsymbol{a} + \boldsymbol{\eta}$. So,

$$\widetilde{\boldsymbol{a}} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{B} = (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top (\boldsymbol{X}\boldsymbol{a}^* + \boldsymbol{\eta}) = \boldsymbol{a} + (\boldsymbol{X}^\top \boldsymbol{X})^{-1} \boldsymbol{X}^\top \boldsymbol{\eta}$$

By Lemma 2.30, we can express $\boldsymbol{X} = \boldsymbol{G}\boldsymbol{L}^\top$ where matrix $\boldsymbol{G} \in \mathbb{R}^{k \times p}$ is a random matrix with i.i.d. $N(0,1)$ entries. Since $\boldsymbol{\Delta} = \widehat{\boldsymbol{a}} - \boldsymbol{a}^*$, we see that $\boldsymbol{\Delta} = (\boldsymbol{L}^\top)^{-1} (\boldsymbol{G}^\top \boldsymbol{G})^{-1} \boldsymbol{G}^\top \boldsymbol{\eta}$. Rearranging, we have $\boldsymbol{L}^\top \boldsymbol{\Delta} = (\boldsymbol{G}^\top \boldsymbol{G})^{-1} \boldsymbol{G}^\top \boldsymbol{\eta}$ and so $\|\boldsymbol{L}^\top \boldsymbol{\Delta}\| \leq \|(\boldsymbol{G}^\top \boldsymbol{G})^{-1}\| \cdot \|\boldsymbol{G}^\top \boldsymbol{\eta}\|$. Combining Lemma 3.3 and Lemma 3.4, which bound $\|(\boldsymbol{G}^\top \boldsymbol{G})^{-1}\|$ and $\|\boldsymbol{G}^\top \boldsymbol{\eta}\|$ respectively, we get

$$\Pr\left(\|\boldsymbol{L}^\top \boldsymbol{\Delta}\| > \frac{2\sigma^* c_2 \sqrt{p}}{(1-2c_1)^2 \sqrt{k}} \right) \leq \exp\left(-\frac{kc_1^2}{2}\right) + 2p \exp(-2k) + p \exp\left(-\frac{c_2^2}{2}\right) \tag{3.7}$$

for any constants $0 < c_1 < 1/2$ and $c_2 > 0$. The claim follows by setting $k = \frac{4c_2^2}{(1-c_1)^4} \cdot \frac{d_{total}}{\varepsilon}$. □

We can now establish Condition 1 of Lemma 3.7 for LeastSquares.

Lemma 3.12. *With $m_1 \in \mathcal{O}\left(\frac{d_{total}}{\varepsilon} \cdot \ln\left(\frac{n}{\delta}\right)\right)$ samples, LeastSquares recovers the coefficients $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n$ such that*

$$\Pr\left(\forall i \in [n], \quad |\boldsymbol{\Delta}_i^\top \boldsymbol{M}_i \boldsymbol{\Delta}_i| \geq (\sigma_i^*)^2 \cdot \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}} \right) \leq \delta$$

The total running time is $\mathcal{O}\left(\frac{d_{total}^2 \cdot d}{\varepsilon} \ln\left(\frac{1}{\delta}\right)\right)$.

Proof. By setting $c_1 = 1/4$, $c_2 = \sqrt{2 \ln(3n/\delta)}$, and $k = \frac{32 d_{total}}{\varepsilon} \ln\left(\frac{3n}{\delta}\right) \geq \frac{4c_2^2}{(1-c_1)^4} \cdot \frac{d_{total}}{\varepsilon}$ in Lemma 3.11, we have

$$\begin{aligned} &\Pr\left(|\boldsymbol{\Delta}_i^\top \boldsymbol{M}_i \boldsymbol{\Delta}_i| \geq (\sigma_i^*)^2 \cdot \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}} \right) \\ \leq\ & \exp\left(-\frac{kc_1^2}{2}\right) + |\mathrm{Pa}(X_i)| \cdot \exp(-2k) + |\mathrm{Pa}(X_i)| \cdot \exp\left(-\frac{c_2^2}{2}\right) \\ \leq\ & \frac{\delta}{3n} + \frac{\delta}{3n} + \frac{\delta}{3n} \\ =\ & \frac{\delta}{n} \end{aligned}$$

for any $i \in [n]$. The claim holds by a union bound over all n variables.

As $\max_{i \in [n]} |\mathrm{Pa}(X_i)| \leq d$, $\sum_{i=1}^{n} |\mathrm{Pa}(X_i)| = d_{total}$, and the computational complexity for a variable with p parents is $\mathcal{O}(m_1 \cdot p^2)$, the total runtime is $\mathcal{O}(m_1 \cdot d_{total} \cdot d) \subseteq \mathcal{O}\left(\frac{d_{total}^2 \cdot d}{\varepsilon} \ln\left(\frac{1}{\delta}\right)\right)$. □

Theorem 3.10 follows from combining the guarantees of LeastSquares (Lemma 3.12) and VarianceRecovery (Theorem 3.9) via Lemma 3.7.

Proof of Theorem 3.10. We will show sample and time complexities before giving the proof for the d_{TV} distance.

Let $m_1 \in \mathcal{O}\left(\frac{d_{total}}{\varepsilon} \cdot \ln\left(\frac{n}{\delta}\right)\right)$ and $m_2 \in \mathcal{O}\left(\frac{d_{total}}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$. Then, the total number of samples needed is $m = m_1 + m_2 \in \mathcal{O}\left(\frac{d_{total}}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$. LeastSquares runs in $\mathcal{O}\left(\frac{d_{total}^2 \cdot d}{\varepsilon} \ln\left(\frac{1}{\delta}\right)\right)$ time while VarianceRecovery runs in $\mathcal{O}\left(\frac{d_{total}^2}{\varepsilon} \log\left(\frac{1}{\delta}\right)\right)$ time. Therefore, the overall running time is $\mathcal{O}\left(\frac{d_{total}^2 \cdot d}{\varepsilon} \log\left(\frac{1}{\delta}\right)\right)$.

By Lemma 3.12, LeastSquares recovers coefficients $\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n$ such that

$$\Pr\left(\forall i \in [n], \quad |\boldsymbol{\Delta}_i^\top \boldsymbol{M}_i \boldsymbol{\Delta}_i| \geq (\sigma_i^*)^2 \cdot \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}\right) \leq \delta$$

By Theorem 3.9 and Using the recovered coefficients from LeastSquares, the guarantees of Theorem 3.9 tells us that VarianceRecovery recovers variance estimates $\widehat{\sigma}_i^2$ such that

$$\Pr\left(\forall i \in [n], \quad 1 - \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}} \leq \left(\frac{\widehat{\sigma}_i}{\sigma_i^*}\right)^2 \leq 1 + \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}}\right) \geq 1 - \delta$$

As our estimated parameters satisfy Condition 1 and Condition 2, Lemma 3.7 tells us that $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) \leq 3\varepsilon$. Thus, $\mathrm{d}_{\mathrm{TV}}(\mathcal{P}, \mathcal{Q}) \leq \sqrt{\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q})/2} \leq \sqrt{3\varepsilon/2}$.

The claim follows by repeating the above analysis with $\varepsilon' = \sqrt{3\varepsilon/2}$. □

3.7 Coefficient recovery based on Cauchy variables

In this section, we provide novel algorithms CauchyEst and CauchyEstTree for recovering the coefficients in polytree Bayesian networks. We will show that CauchyEstTree has near-optimal sample complexity. Of technical interest, our analysis involves Cauchy random variables, which are somewhat of a rarity in statistical learning. As in LeastSquares, CauchyEst and CauchyEstTree use independent samples to recover the coefficients associated to each individual variable in an independent fashion.

Let us consider an arbitrary variable $Y \in \{X_1, \ldots, X_n\}$ with p parents with additive noise η, corresponding coefficients $\boldsymbol{a}^*$, and covariance matrix $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^\top$. As before, w.l.o.g., we may assume that Y has parents $X_1, \ldots, X_p$ by relabelling. The intuition is as follows: if $\eta = 0$, then one can form a linear system of equations using p samples to *exactly* solve for the coefficients $\boldsymbol{a}^*$. Unfortunately, η is non-zero in general. Instead of exactly recovering $\boldsymbol{a}$, we partition the m_1 independent samples into $k = \lfloor m_1/p \rfloor$ batches involving p samples and form intermediate estimates $\widetilde{\boldsymbol{a}}^{(1)}, \dots, \widetilde{\boldsymbol{a}}^{(k)}$ by solving a system of linear equations for each batch (see Batch, Algorithm 4). Then, we "combine" these intermediate estimates to obtain our estimate $\widehat{\boldsymbol{a}}$.

Algorithm 4 Batch: Batch coefficient recovery algorithm for variable with p parents

```
Input: DAG $\mathcal{G}$, a variable $Y$ with $p$ parents $X_1, \dots, X_p$, and $p$ samples
Form matrix $\boldsymbol{X} \in \mathbb{R}^{p \times p}$, where the $r^{th}$ row consists of samples $(x_1^{(r)}, \dots, x_p^{(r)})$
Form column vector $\boldsymbol{B} = (y^{(1)}, \dots, y^{(p)})^\top \in \mathbb{R}^p$
Define $\widetilde{\boldsymbol{a}}$ as any solution to $\boldsymbol{X}\widetilde{\boldsymbol{a}} = \boldsymbol{B}$.
return $\widetilde{\boldsymbol{a}}$
```

An astute reader will notice that Batch is very similar to LeastSquares in that both attempt to estimate $\widetilde{\boldsymbol{a}}$ via $\boldsymbol{X}\widetilde{\boldsymbol{a}} = \boldsymbol{B}$. While LeastSquares uses the least squares estimate, Batch works with *any* solution to the linear system of equations.

For an arbitrary copy of recovered coefficients $\widetilde{\boldsymbol{a}}$, let $\boldsymbol{\Delta} = \widetilde{\boldsymbol{a}} - \boldsymbol{a}^*$ be a vector measuring the ccoordinate-wise gap between these recovered coefficients and the ground truth as before. The following lemma shows that each entry of the vector $\boldsymbol{L}^\top \boldsymbol{\Delta}$ is distributed according to $\sigma^* \cdot \mathrm{Cauchy}(0, 1)$, although the entries may be correlated with each other in general.

Lemma 3.13. *Consider a batch estimate $\widetilde{\boldsymbol{a}}$ from Batch. Then, $\boldsymbol{L}^\top \boldsymbol{\Delta}$ is entry-wise distributed as $\sigma^* \cdot \mathrm{Cauchy}(0, 1)$, where $\boldsymbol{\Delta} = \widetilde{\boldsymbol{a}} - \boldsymbol{a}^*$. Note that the entries of $\boldsymbol{L}^\top \boldsymbol{\Delta}$ may be correlated in general.*

Proof. Observe that each row of $\boldsymbol{X}$ is an independent sample drawn from a multivariate Gaussian $N(0, \boldsymbol{M})$. By denoting $\boldsymbol{\eta} = (\eta^{(1)}, \dots, \eta^{(p)})^\top$ as the p samples of η, we can write $\boldsymbol{X}\widetilde{\boldsymbol{a}} = \boldsymbol{X}\boldsymbol{a} + \boldsymbol{\eta}$ and thus $\boldsymbol{X}\boldsymbol{\Delta} = \boldsymbol{\eta}$ by rearranging terms. By Lemma 2.30, we can express $\boldsymbol{X} = \boldsymbol{G}\boldsymbol{L}^\top$ where matrix $\boldsymbol{G} \in \mathbb{R}^{p \times p}$ is a random matrix with i.i.d. $N(0, 1)$ entries. By substituting $\boldsymbol{X} = \boldsymbol{G}\boldsymbol{L}^\top$ into $\boldsymbol{X}\boldsymbol{\Delta} = \boldsymbol{\eta}$, we have $\boldsymbol{L}^\top \boldsymbol{\Delta} = \boldsymbol{G}^{-1}\boldsymbol{\eta}$.[10]

By applying Lemma 3.6 with the following parameters: $\boldsymbol{A} = \boldsymbol{G}, \boldsymbol{B} = \boldsymbol{L}^\top \boldsymbol{\Delta}, \boldsymbol{E} = \eta$, we conclude that each entry of $\boldsymbol{L}^\top \boldsymbol{\Delta}$ is distributed as $\sigma^* \cdot \mathrm{Cauchy}(0, 1)$. However, note that *these entries are generally correlated.* □

If we have direct access to the matrix $\boldsymbol{L}$, then one can do the following (see CauchyEst, Algorithm 5): take coordinate-wise *medians* of $\boldsymbol{L}^\top \widetilde{a}$ to form $\texttt{MED}_i$ and then estimate $\widehat{\boldsymbol{a}} = (\boldsymbol{L}^\top)^{-1}(\texttt{MED}_1, \dots, \texttt{MED}_n)^\top$. The reason why we use medians is because the typical strategy of averaging independent estimates does not work here as the variance of a Cauchy variable is unbounded. By the convergence of Cauchy random variables to their median, one can show that each coordinate of $\widehat{\boldsymbol{a}}$ converges to the true coefficient $\boldsymbol{a}^*$ as before.

[10] Note that event that $\boldsymbol{G}$ is singular has measure 0.

Unfortunately, we do not have $\boldsymbol{L}$ and can only hope to estimate it with some matrix $\widehat{\boldsymbol{L}}$ using the *empirical* covariance matrix $\widehat{\boldsymbol{M}}$.

Algorithm 5 CauchyEst: Coefficient recovery algorithm for general Bayesian networks

```
Input: DAG $\mathcal{G}$ and $m$ samples
for variable $Y$ with $p \geq 1$ parents $X_1, \ldots, X_p$ do
    Let $\widehat{\boldsymbol{M}}$ be the empirical covariance matrix with respect to $X_1, \ldots, X_p$.
    Compute the Cholesky decomposition $\widehat{\boldsymbol{M}} = \widehat{\boldsymbol{L}}\widehat{\boldsymbol{L}}^\top$ of $\widehat{\boldsymbol{M}}$.
    for $s = 1, \ldots, \lfloor m/p \rfloor$ do
        Using $p$ samples and Batch, compute a batch estimate $\widetilde{\boldsymbol{a}}^{(s)}$.
    For each $i \in [n]$, define $\texttt{MED}_i = \text{median}\{(\widehat{\boldsymbol{L}}^\top \widetilde{\boldsymbol{a}}^{(1)})_i, \ldots, (\widehat{\boldsymbol{L}}^\top \widetilde{\boldsymbol{A}}^{(\lfloor m/p \rfloor)})_i\}$.
    return $\widehat{\boldsymbol{a}} = (\widehat{\boldsymbol{L}}^\top)^{-1}(\texttt{MED}_1, \ldots, \texttt{MED}_n)^\top$.
```

3.7.1 Special case of polytree Bayesian networks

If the Bayesian network is a polytree, then $\boldsymbol{L}$ is diagonal. In this case, we specialize CauchyEst to CauchyEstTree and are able to give theoretical guarantees. We begin with simple corollary which tells us that the i^{th} entry of $\boldsymbol{\Delta}$ is distributed according to $\frac{\sigma^*}{\sigma_i} \cdot \text{Cauchy}(0, 1)$.

Corollary 3.14. *Consider a batch estimate $\widetilde{\boldsymbol{a}}$ from* Batch*. If the Bayesian network is a polytree, then $\boldsymbol{\Delta}_i = (\widetilde{\boldsymbol{a}} - \boldsymbol{a}^*)_i \sim \frac{\sigma^*}{\sigma_i} \cdot \text{Cauchy}(0, 1)$.*

Proof. Observe that each row of $\boldsymbol{X}$ is an independent sample drawn from a multivariate Gaussian $N(0, \boldsymbol{M})$. By denoting $\boldsymbol{\eta} = (\eta^{(1)}, \ldots, \eta^{(p)})^\top$ as the p samples of η, we can write $\boldsymbol{X}\widetilde{\boldsymbol{a}} = \boldsymbol{X}\boldsymbol{a} + \boldsymbol{\eta}$ and thus $\boldsymbol{X}\boldsymbol{\Delta} = \boldsymbol{\eta}$ by rearranging terms. Since the parents of any variable in a polytree are not correlated, each element in the i^{th} column of $\boldsymbol{X}$ is a $N(0, \sigma_i^2)$ Gaussian random variable.

By applying Lemma 3.6 with the following parameters: $\boldsymbol{A} = \boldsymbol{X}, \boldsymbol{B} = \boldsymbol{\Delta}\boldsymbol{E} = \boldsymbol{\eta}$, we conclude that $\boldsymbol{\Delta}_i = (\widetilde{\boldsymbol{a}} - \boldsymbol{a})_i \sim \frac{\sigma^*}{\sigma_i} \cdot \text{Cauchy}(0, 1)$. □

For each $i \in \text{Pa}(Y)$, we combine the k independently copies of $\widetilde{\boldsymbol{a}}^{(1)}, \ldots, \widetilde{\boldsymbol{a}}^{(k)}$ using the coordinate-wise median: $(\widehat{\boldsymbol{a}})_i = \text{median}_{s \in [k]}(\widetilde{\boldsymbol{a}}^{(s)})_i$ for each coordinate i. For arbitrary sample $s \in [k]$ and parent index $i \in \text{Pa}(Y)$, observe that the i^{th} coordinate error is $(\boldsymbol{\Delta}^{(s)})_i = (\widetilde{\boldsymbol{a}}^{(s)})_i - (\boldsymbol{a}^*)_i$. Since $(\boldsymbol{a}^*)_i$ is just an unknown *constant*,

$$(\widehat{\boldsymbol{a}})_i = \text{median}_{s \in [k]} \left(\widetilde{\boldsymbol{a}}^{(s)}\right)_i = (\boldsymbol{a}^*)_i + \text{median}_{s \in [k]} \left(\boldsymbol{\Delta}^{(s)}\right)_i$$

Since each $(\boldsymbol{\Delta}^{(s)})_i$ term is i.i.d. distributed as $\sigma^* \cdot \text{Cauchy}(0, 1)$, the median term $\text{median}_{s \in [k]}(\boldsymbol{\Delta}^{(s)})_i$ converges to 0 with sufficiently large k, and thus $(\widehat{\boldsymbol{a}})_i$ converges to the true coefficient i^{th} coordinate $(\boldsymbol{a}^*)_i$ of $\boldsymbol{a}^*$.

The goal of this section is to prove Theorem 3.15 given CauchyEstTree (Algorithm 6).

Algorithm 6 CauchyEstTree: Coefficient recovery for polytrees

```
1: Input: A polytree $\mathcal{G}$ and $m_1 \in \mathcal{O}\left(\frac{d_{total} \cdot d}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$ samples
2: for variable $Y$ with $p \geq 1$ parents $X_1, \ldots, X_p$ do
3:     for $s = 1, \ldots, \lfloor m_1/p \rfloor$ do
4:         Using $p$ samples and Batch, compute a batch estimate $\widetilde{\boldsymbol{a}}^{(s)}$.
5:     return column vector $\widehat{\boldsymbol{a}}$, where $(\widehat{\boldsymbol{a}})_i = \begin{cases} \text{median}_{s \in [k]} (\widetilde{\boldsymbol{a}}^{(s)})_i & \text{if } i \in \text{Pa}(Y) \\ 0 & \text{if } i \not\in \text{Pa}(Y) \end{cases}$
```

Theorem 3.15 (Distribution learning using CauchyEstTree)**.** *Let $\varepsilon, \delta \in (0,1)$. Suppose $\mathcal{G}$ is a fixed directed acyclic graph on n variables with degree at most d. Given $\mathcal{O}\left(\frac{d_{total} \cdot d}{\varepsilon} \log\left(\frac{n}{\varepsilon\delta}\right)\right)$ samples from an unknown Bayesian network $\mathcal{P}$ over $\mathcal{G}$, if we use CauchyEstTree for coefficient recovery in Algorithm 1, then with probability at least $1-\delta$, we recover a Bayesian network $\widehat{\mathcal{P}}$ over $\mathcal{G}$ such that $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ in $\mathcal{O}\left(\frac{d_{total}^2 \cdot d^{\omega-1}}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$ time.*

Note that for polytrees, d_{total}/n is just a constant. As before, we will first prove guarantees for an arbitrary variable and then take union bound over n variables.

Lemma 3.16. *Consider the CauchyEstTree algorithm. Fix an arbitrary variable of interest Y with p parents, parameters $(\boldsymbol{a}^*, \sigma^*)$, and associated covariance matrix $\boldsymbol{M}$. With $k = \frac{8 d_{total}}{\varepsilon} \log\left(\frac{2}{\delta}\right)$ samples, we recover coefficient estimates $\widehat{\boldsymbol{a}}$ such that*

$$\Pr\left(|\boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}| \leq (\sigma^*)^2 \cdot \frac{\varepsilon \cdot p}{d_{total}}\right) \geq 1 - \delta$$

Proof. Since $\boldsymbol{M} = \boldsymbol{L}\boldsymbol{L}^\top$, it suffices to bound $\|\boldsymbol{L}^\top \boldsymbol{\Delta}\|$. Lemma 3.13 tells us that each entry of the vector $\boldsymbol{L}^\top \boldsymbol{\Delta}$ is the median of k copies of $\text{Cauchy}(0,1)$ random variables multiplied by σ_y. Setting $k = \frac{8 d_{total}}{\varepsilon} \log\left(\frac{2}{\delta}\right)$ and $0 < \tau = \sqrt{\frac{\varepsilon}{d_{total}}} < 1$ in Lemma 3.5, we see that

$$\Pr\left(\text{median of } k \text{ i.i.d. } \text{Cauchy}(0,1) \text{ random variables} \not\in \left[-\sqrt{\frac{\varepsilon}{d_{total}}}, \sqrt{\frac{\varepsilon}{d_{total}}}\right]\right) \leq \delta$$

That is, each entry of $\boldsymbol{L}^\top \boldsymbol{\Delta}$ has absolute value at most $\sigma^* \cdot \sqrt{\frac{\varepsilon}{d_{total}}}$. By summing across all p entries of $L^\top \boldsymbol{\Delta}$, we see that

$$|\boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}| = |\boldsymbol{\Delta}^\top \boldsymbol{L}\boldsymbol{L}^\top \boldsymbol{\Delta}| = \|\boldsymbol{L}^\top \boldsymbol{\Delta}\|^2 \leq p \cdot (\sigma^*)^2 \cdot \frac{\varepsilon}{d_{total}} = (\sigma^*)^2 \cdot \frac{\varepsilon \cdot p}{d_{total}}$$

□

We can now establish Condition 1 of Lemma 3.7 for CauchyEstTree.

Lemma 3.17. *Consider the CauchyEstTree algorithm. Suppose the Bayesian network is a polytree. With $m_1 \in \mathcal{O}\left(\frac{d_{total} \cdot d}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$ samples, we recover coefficient estimates*

$\widehat{\boldsymbol{a}}_1, \ldots, \widehat{\boldsymbol{a}}_n$ such that

$$\Pr\left(\forall i \in [n], \quad |\boldsymbol{\Delta}_i^\top \boldsymbol{M}_i \boldsymbol{\Delta}_i| \geq (\sigma_i^*)^2 \cdot \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}\right) \leq \delta$$

The total running time is $\mathcal{O}\left(\frac{d_{total}^2 \cdot d^{\omega-1}}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$ where ω is the matrix multiplication exponent.

Proof. For each $i \in [n]$, apply Lemma 3.16 with $\delta' = \delta/n$ and $m_1 = \frac{8d_{total}}{\varepsilon} \log\left(\frac{2n}{\delta}\right)$, then take the union bound over all n variables.

The runtime of Batch is the time to find the inverse of a $p \times p$ matrix, which is $\mathcal{O}(p^\omega)$ for some matrix multiplication constant $\omega \in (2,3)$. Therefore, the computational complexity for a variable with p parents is $\mathcal{O}(p^{\omega-1} \cdot m_1)$. Since $\max_{i\in[n]} |\mathrm{Pa}(X_i)| \leq d$ and $\sum_{i=1}^n |\mathrm{Pa}(X_i)| = d_{total}$, the total runtime is $\mathcal{O}(m_1 \cdot d_{total} \cdot d^{\omega-2})$. □

We are now ready to prove Theorem 3.15.

Theorem 3.15 follows from combining Lemma 3.17 and Theorem 3.9 (the guarantees of CauchyEstTree and VarianceRecovery respectively) via Lemma 3.7.

Proof of Theorem 3.15. We will show sample and time complexities before giving the proof for the $\mathrm{d_{TV}}$ distance.

Let $m_1 \in \mathcal{O}\left(\frac{d_{total} \cdot d}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$ and $m_2 \in \mathcal{O}\left(\frac{d_{total}}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$. Then, the total number of samples needed is $m = m_1 + m_2 \in \mathcal{O}\left(\frac{d_{total} \cdot d}{\varepsilon} \log\left(\frac{n}{\varepsilon\delta}\right)\right)$. CauchyEstTree runs in $\mathcal{O}\left(\frac{d_{total}^2 \cdot d^{\omega-1}}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$ time while VarianceRecovery runs in $\mathcal{O}\left(\frac{d_{total}^2}{\varepsilon} \log\left(\frac{1}{\delta}\right)\right)$ time, where ω is the matrix multiplication exponent. Therefore, the overall running time is $\mathcal{O}\left(\frac{d_{total}^2 \cdot d^{\omega-1}}{\varepsilon} \log\left(\frac{n}{\delta}\right)\right)$.

By Lemma 3.17, CauchyEstTree recovers coefficients $\widehat{A}_1, \ldots, \widehat{A}_n$ such that

$$\Pr\left(\forall i \in [n], |\boldsymbol{\Delta}_i^\top M_i \boldsymbol{\Delta}_i| \geq (\sigma_i^*)^2 \cdot \frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}\right) \leq \delta$$

Using the recovered coefficients from CauchyEstTree, the guarantees of Theorem 3.9 tells us that VarianceRecovery recovers variance estimates $\widehat{\sigma}_i^2$ such that

$$\Pr\left(\forall i \in [n], \quad 1 - \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}} \leq \left(\frac{\widehat{\sigma}_i}{\sigma_i^*}\right)^2 \leq 1 + \sqrt{\frac{\varepsilon \cdot |\mathrm{Pa}(X_i)|}{d_{total}}}\right) \geq 1 - \delta$$

As our estimated parameters satisfy Condition 1 and Condition 2, Lemma 3.7 tells us that $\mathrm{d_{KL}}(\mathcal{P}, \mathcal{Q}) \leq 3\varepsilon$. Thus, $\mathrm{d_{TV}}(\mathcal{P}, \mathcal{Q}) \leq \sqrt{\mathrm{d_{KL}}(\mathcal{P}, \mathcal{Q})/2} \leq \sqrt{3\varepsilon/2}$. The claim follows by setting $\varepsilon' = \sqrt{3\varepsilon/2}$ throughout. □

Chapter 4

Learning bounded-degree polytrees with known skeleton

"All models are wrong but some are useful."

- George Box [Box79].

4.1 Introduction

Polytrees are a subclass of Bayesian networks (Section 2.7) where the undirected graph underlying the DAG is a forest, i.e. there are no cycles in the undirected graph obtained by ignoring edge directions. Since there is a unique path in $\mathrm{skel}(\mathcal{G})$ between any two vertices in the graph in polytrees, ancestors of any vertex V are mutually independent and typically become mutually dependent when V (or any of V's descendants) are being conditioned over. A polytree with maximum in-degree d is also known as a d-polytree. Polytrees are of particular interest because inference on polytree-structured Bayesian networks can be performed efficiently [PK83, Pea86]. Another motivation to study polytrees is due to [GA21] showing that polytrees are easier to learn than general Bayesian networks due to the underlying graph being a tree, allowing typical assumptions such as faithfulness (Definition 2.50) to be dropped when designing efficient learning algorithms.

With an infinite number of samples, one can recover the DAG of a non-degenerate polytree in the equivalence class with the Chow-Liu algorithm [CL68] and some additional conditional independence tests [RP88]. However, this does *not* work in the finite sample regime and the only known finite sample result for learning polytrees is for 1-polytrees [BGP$^+$23, DP21]. Furthermore, in the agnostic setting, the learning problem of finding the closest polytree distribution to an arbitrary distribution $\mathcal{P}$ is NP-hard [Das99].

In this chapter, we consider the task of PAC-learning from samples of a discrete distribution that is described by a degree-bounded polytrees. While d^* denotes the true maximum in-degree of the underlying polytree, our algorithm and results are with respect to a given upper bound d of d^*. Specifically, we focus on the realizable setting where the discrete distribution $\mathcal{P}$ on n variables (each with domain Σ) which we draw samples from is Markov with respect to an unknown d^*-polytree $\mathcal{G}^*$. Prior to our work, the only known result known is for the $d^* = 1$: [BGP+23, DP21] tell us that $\widetilde{\Theta}(\frac{n \cdot |\mathbf{\Sigma}|^2}{\varepsilon})$ samples are sufficient to produce $\widehat{\mathcal{P}}$ such that $\mathrm{d_{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ by analyzing the Chow-Liu algorithm. This sample complexity is also necessary in the worst case. Note that we can focus on bounding $\mathrm{d_{KL}}$ instead of $\mathrm{d_{TV}}$ because $\mathrm{d_{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ implies $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \sqrt{\varepsilon/2}$ via Pinsker's inequality (Theorem 2.18) so one can obtain corresponding bounds for $\mathrm{d_{TV}}$ by replacing ε with ε^2 throughout.

4.2 Our main results

We give a sample-efficient algorithm for proper Bayesian network learning in the realizable setting, when provided with the ground truth skeleton (i.e., the underlying forest). Crucially, our result does not require any distributional assumptions such as strong faithfulness (Definition 2.50), etc. We also give information-theoretic sample complexity lower bounds that hold even when the ground truth skeleton is known and given to us.

Theorem 4.1. *Let $\varepsilon, \delta \in (0, 1)$ be the error and failure parameters respectively. Consider a discrete distribution $\mathcal{P}$ on n variables, each with alphabet Σ, defined on a polytree $\mathcal{G}^*$ with an unknown maximum in-degree d^*. Given $m = \widetilde{\Omega}\left(\frac{n \cdot |\mathbf{\Sigma}|^{d+1}}{\varepsilon} \log \frac{1}{\delta}\right)$ samples from $\mathcal{P}$, the skeleton of $\mathcal{G}^*$, and an in-degree upper bound $d \geq d^*$, there exists an algorithm that outputs a d-polytree distribution $\widehat{\mathcal{P}}$ such that $\mathrm{d_{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$. This algorithm runs in time polynomial in m, $|\Sigma|^d$, and n^d and succeeds with probability at least $1 - \delta$.*

We remark that Theorem 4.1 only requires an upper bound d on the true in-degree d^*. In particular, our result yields a sample complexity upper bound of $\widetilde{\mathcal{O}}(n/\varepsilon)$ for learning $\mathcal{O}(1)$-polytrees with constant $|\Sigma|$ and d. In Section 4.5, we state sufficient distributional conditions that enable recovery of the ground truth skeleton. Informally, we require that the data processing inequality hold in a strong sense with respect to the edges in the skeleton. Applying Theorem 4.1 under these conditions would then imply a polynomial-time PAC algorithm to learn bounded-degree polytrees from samples.

Our next result shows that this dependence on the dimension n and the accuracy parameter ε is optimal, up to logarithmic factors, even for $d = |\Sigma| = 2$.

Theorem 4.2. *Let $\varepsilon \in (0, 1)$ be the error parameter. There exists a choice of distribution $\mathcal{P}$ over $\{0, 1\}^n$ that is Markov with respect to some 2-polytree $\mathcal{G}^*$ such that producing $\widehat{\mathcal{P}}$ such that $\mathrm{d_{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ with success probability at least $2/3$ requires $\Omega(n/\varepsilon)$ samples from $\mathcal{P}$, even given we are given $\mathrm{skel}(\mathcal{G}^*)$ as input.*

In some sense, Theorem 4.2 generalizes the $\widetilde{\Omega}\left(\frac{n}{\varepsilon}\right)$ sample complexity lower bound of [BGP+23, Theorem 7.6] in the case where $d = 1$ and $\mathrm{skel}(\mathcal{G}^*)$ is not given as input.

4.3 Technical overview

4.3.1 Some setup

Let us begin by introducing some notation and preliminary concepts for this chapter.

Let $\boldsymbol{X}$ be the set of n variables which the distribution $\mathcal{P}$ is defined over. For any subset of variables $\boldsymbol{S} \subseteq \boldsymbol{X}$ and graph $\mathcal{G}$, $\mathcal{P}_{\mathcal{S}}$ denotes the projection of $\mathcal{P}$ while $\mathcal{P}_{\mathcal{G}}$ dnotes the projection of $\mathcal{P}$ onto $\mathcal{G}$. More specifically, we have $\mathcal{P}_{\mathcal{G}}(x_1, \ldots, x_n) = \prod_{x \in \boldsymbol{X}} \mathcal{P}(x \mid \mathrm{pa}_{\mathcal{G}}(X))$. Note that $\mathcal{P}_{\mathcal{G}}$ is the closest distribution on $\mathcal{G}$ to $\mathcal{P}$ in d_{KL}, i.e. $\mathcal{P}_{\mathcal{G}} = \mathrm{argmin}_{\mathcal{Q} \in \mathcal{G}} \mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q})$. One can verify this using [BGP+23, Lemma 3.3]: for any distribution $\mathcal{Q}$ defined on $\mathcal{G}$,

$$\begin{aligned} &\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{Q}) - \mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{P}_{\mathcal{G}}) \\ &\qquad = \sum_{V \in \boldsymbol{V}} \mathcal{P}(\mathrm{pa}_{\mathcal{G}}(V)) \cdot \mathrm{d}_{\mathrm{KL}}(\mathcal{P}(V \mid \mathrm{Pa}_{\mathcal{G}}(V)), \mathcal{Q}(V \mid \mathrm{Pa}_{\mathcal{G}}(V))) \geq 0 \end{aligned}$$

By [CL68], we also know that

$$\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{P}_{\mathcal{G}}) = - \sum_{i=1}^{n} I(X_i; \mathrm{Pa}_{\mathcal{G}}(X_i)) - H(\mathcal{P}_{\boldsymbol{X}}) + \sum_{i=1}^{n} H(\mathcal{P}_{X_i}) \;, \tag{4.1}$$

where H is the entropy function. Since only the first term depends on the graph structure of $\mathcal{G}$, this motivate the Chow-Liu algorithm [CL68]: given mutual information between each pairs of variables as edge weights, compute the maximum weight spanning tree.

Our goal in this chapter is to obtain approximately good graph $\widehat{\mathcal{G}}$ for $\mathcal{P}$ in the sense of $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \mathcal{P}_{\widehat{\mathcal{G}}}) \leq \varepsilon$. With $\widehat{\mathcal{G}}$, one can employ sample and computational-efficient learning algorithms to output the final hypothesis $\widehat{\mathcal{P}}$.

Note that for some distributions there could be more than one ground truth graph, e.g. when the Markov equivalence class has multiple graphs. In such situations, for analysis purposes, we are free to choose any graph that $\mathcal{P}$ is Markov with respect to. As the mutual information (MI) scores, i.e. the sum of MI terms in Eq. (4.1), are the same for any graphs that $\mathcal{P}$ is Markov with respect to, the choice of $\mathcal{G}^*$ does not matter here.

We also study a generalized version of v-structures (*deg-ℓ v-structure*) where the center has $\ell \geq 2$ parents $u_1, u_2, \ldots, u_\ell$. We say that a deg-ℓ v-structure is said to be ε-strong if we can reliably identify them in the finite sample regime.

Definition 4.3 (ε-strong deg-ℓ v-structure)**.** Let $0 < c_0 < 1$ be the universal constant appearing in Corollary 4.4. A deg-ℓ v-structure is a subgraph on $\ell + 1$ nodes $v, u_1, \ldots, u_\ell$ such that:

1. **deg-ℓ v-structure**: $v \leftarrow u_k$ for all $k \in [\ell]$, and $u_k \not\sim u_{k'}$ for all $k, k' \in [\ell]$ and $k \neq k'$
2. **ε-strong**: $I(u_k; \{u_1, u_2, \ldots, u_\ell\} \setminus u_k \mid v) \geq c_0 \cdot \varepsilon$ for all $k \in [\ell]$

Our algorithmic correctness relies on the following result (Corollary 4.4) about conditional mutual information (CMI) testers, which is adapted from Theorem 1.3 of [BGP$^+$23]; see Appendix A.2.1 for derivation details and [CYBC24, Appendix B] for a derivation of a constant c_0 that works.

Corollary 4.4 (CMI tester). *Fix any $\varepsilon > 0$. Let (X, Y, Z) be three random variables over $\mathbf{\Sigma}_X, \mathbf{\Sigma}_Y, \mathbf{\Sigma}_Z$ respectively. Given the empirical distribution $(\widehat{X}, \widehat{Y}, \widehat{Z})$ over a size N sample of (X, Y, Z), there exists a universal constant $0 < c_0 < 1$ so that for any N at least*

$$\Theta\left(\frac{|\mathbf{\Sigma}_X| \cdot |\mathbf{\Sigma}_Y| \cdot |\mathbf{\Sigma}_Z|}{\varepsilon} \cdot \log \frac{|\mathbf{\Sigma}_X| \cdot |\mathbf{\Sigma}_Y| \cdot |\mathbf{\Sigma}_Z|}{\delta} \cdot \log \frac{|\mathbf{\Sigma}_X| \cdot |\mathbf{\Sigma}_Y| \cdot |\mathbf{\Sigma}_Z| \cdot \log\left(\frac{1}{\delta}\right)}{\varepsilon}\right),$$

the following statements hold with probability $1 - \delta$:
(1) If $I(X; Y \mid Z) = 0$, then $\widehat{I}(X; Y \mid Z) < c_0 \cdot \varepsilon$.
(2) If $\widehat{I}(X; Y \mid Z) \leq c_0 \cdot \varepsilon$, then $I(X; Y \mid Z) < \varepsilon$.
Unconditional statements for $I(X; Y)$ and $\widehat{I}(X; Y)$ hold similarly by setting $|\mathbf{\Sigma}_Z| = 1$.

Using the contrapositive of the first statement of Corollary 4.4 and non-negativity of CMI, one can also see that if $\widehat{I}(X; Y \mid Z) \geq c_0 \cdot \varepsilon$, then $I(X; Y \mid Z) > 0$.

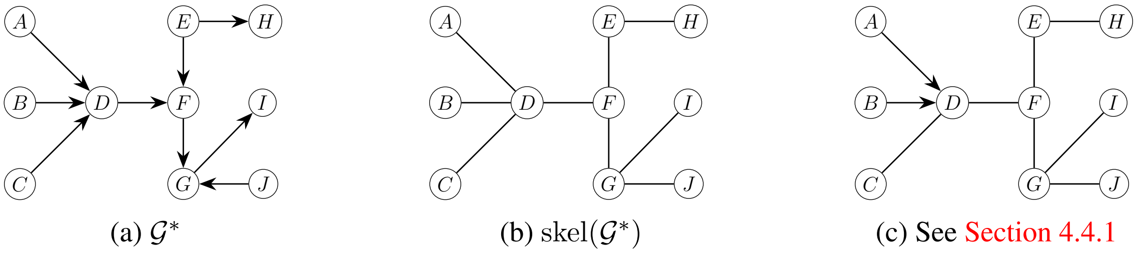
4.3.2 Overview of algorithm

Our algorithm is designed with Eq. (4.1) in mind. Since there are efficient algorithms for estimating the parameters of a Bayesian network with in-degree d once a close-enough graph $\widehat{\mathcal{G}}$ is recovered [Das97, BGMV20], it suffices to find a good approximation of the underlying DAG $\mathcal{G}^*$. For a distribution $\mathcal{P}$ that is Markov with respect to a DAG $\mathcal{G}^*$, the quality (in terms of KL divergence) of approximating $\mathcal{G}^*$ with $\mathcal{G}$ is $\mathrm{d_{KL}}(\mathcal{P}, \mathcal{P}_{\mathcal{G}}) = \mathrm{d_{KL}}(\mathcal{P}_{\mathcal{G}^*}, \mathcal{P}_{\mathcal{G}}) = \sum_{x \in \boldsymbol{X}} I(X; \mathrm{Pa}_{\mathcal{G}^*}(X)) - I(X; \mathrm{Pa}_{\mathcal{G}}(X))$, where $I(\cdot; \cdot)$ refers to mutual information between the terms. When the true skeleton $\mathrm{skel}(\mathcal{G}^*)$ is given to us in advance, what remains is to orient each edge. As such, given error parameter $\varepsilon > 0$ and upper bound on in-degree d, the goal of our algorithm is to judiciously orient the edges of $\mathrm{skel}(\mathcal{G}^*)$ such that $\mathrm{d_{KL}}(\mathcal{P}_{\mathcal{G}^*}, \mathcal{P}_{\mathcal{G}})$ is at most ε while ensuring that every vertex has at most d incoming edges.

Our algorithm relies on estimating MI and CMI terms involving subsets of variables. A naïve approach of estimating these terms additively would incur unnecessary sample complexity overhead. One of our technical contributions is to show that it suffices to have access to a *tester* that can distinguish between a CMI term being 0 or at least some threshold $\eta > 0$. As shown in [BGP$^+$23], the sample complexity for testing (see Corollary 4.4) is an $O(\eta)$ factor smaller than that for estimating the CMI up to additive error of $\pm\, \eta/2$. Note that the tester is probabilistic in nature and we will upper bound the overall failure rate using union bound later.

Our algorithm works in three phases. In the first phase, we orient "strong v-structures". In the second phase, we locally check if an edge is "forced" to orient in a specific direction. In the third phase, we orient the remaining unoriented edges as a 1-polytree. Throughout the algorithm, we do *not* unorient edges as we will be able to argue that any orientations performed by the first two phases are guaranteed to respect the orientations of the underlying causal graph from which we draw samples from.

To explain the intuition behind the first two phases, consider the example of a path on 3 vertices $U - V - W$ within a possibly larger graph; see Fig. 4.1 for a slightly more sophisticated example. We will orient edges by using the finite-sample CMI tester to determine whether certain CMI values are "large" or "small". If $U \to V \leftarrow W$, then U and W are *dependent* given V. Otherwise, U and W are *independent* given V since $\mathcal{G}$ is a polytree. That is, one would expect $I(U; W \mid V)$ to be large if and only if $U \to V \leftarrow W$ was a v-structure. If it is indeed the case that $I(U; W \mid V)$ is "large", then this would be detected by the tester (i.e. $U \to V \leftarrow W$ was "strong") and so we orient $U \to V$ and $W \to V$ in Phase 1. Now, after Phase 1, the graph would be partially oriented; say, we have $U \to V - W$ after Phase 1. If $U \to V \to W$ was the ground truth, then $I(U; W \mid V) = 0$ and the tester will detect this term as "small". If $U \to V \leftarrow W$ was the ground truth, then $I(U; W) = 0$ and the tester will detect this term as "small". Via the contrapositive of the previous two statements, if $I(U; W \mid V)$ or $I(U; W)$ is "large", then we are "forced" to orient a specific orientation of the edge $V - W$. We may also leave $V - W$ unoriented if neither term was "large". Another form of "forced orientation" is due to the given upper bound d on the number of parents any vertex can have: we should point all remaining incident unoriented edges *away* from a vertex V whenever V already has d incoming arcs. For example, if $d = 1$, then we have to orient $V \to W$ if we observe $U \to V - W$ after Phase 1. Given the above intuition, any edge that remains unoriented till the Phase 3 must have been "flexible" in the sense that it could be oriented either way. In fact, we later show that "not too much error" will be incurred if the edge orientations from the final phase only increases the incoming degrees of any vertex by at most one.



(a) $\mathcal{G}^*$ (b) $\text{skel}(\mathcal{G}^*)$ (c) See Section 4.4.1

Figure 4.1: 3-polytree example where $I(A; B, C) = I(B; A, C) = I(C; A, B) = 0$ due to deg-3 v-structure centered at D. By Corollary 4.4, $I(A; F \mid D) = 0$ implies $\widehat{I}(A; F \mid D) \leq c_0 \cdot \varepsilon$, and so we will *not* detect $A \to D \to F$ erroneously as a strong deg-2 v-structure $A \to D \leftarrow F$.

4.3.3 Overview of information-theoretic lower bound

Our lower bound shows that $\Omega(n/\varepsilon)$ samples are necessary, *even when a known skeleton is provided*. To show this, we first show that $\Omega(1/\varepsilon)$ samples are required for the case where $n = 3$ by reducing the problem finding an ε-close graph orientation to the problem of *testing* whether samples are drawn from two given distributions. To accomplish this, we designed a pair of distributions $\mathcal{P}_1$ and $\mathcal{P}_2$ and a pair of graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ such that

1. $\mathcal{P}_1$ and $\mathcal{P}_2$ have "small" squared Hellinger distance,

2. $\mathcal{P}_i$ has zero KL divergence if projected onto $\mathcal{G}_i$, and

3. $\mathcal{P}_i$ has "large" KL divergence if projected onto $\mathcal{G}_j$ (for $j \neq i$).

Since the distributions have small squared Hellinger distance, say less than ε, one needs $\Omega(1/\varepsilon)$ samples to distinguish them, thus showing that $\Omega(1/\varepsilon)$ samples are required for the case where $n = 3$. To obtain a dependency on n, we construct $n/3$ independent copies of the above gadget, à la proof strategy of [BGP$^+$23, Theorem 7.6].

4.4 Recovering given a skeleton and degree bound

Here, we describe and analyze an algorithm for estimating a probability distribution $\mathcal{P}$ that is defined on a d^*-polytree $\mathcal{G}^*$. We assume that we are given $\text{skel}(\mathcal{G}^*) = (\boldsymbol{V}, \boldsymbol{E})$ and d as input, where $d^* \leq d$.

4.4.1 Algorithm RecoverOrientation

At any point in the algorithm, let us define the following sets. Let $N(V)$ be the set of all neighbors of V in $\text{skel}(\mathcal{G}^*) = (\boldsymbol{V}, \boldsymbol{E})$ over $|\boldsymbol{V}| = n$ variables. Let $N^{\text{in}}(V) \subseteq N(V)$ be the current set of incoming neighbors of V. Let $N^{\text{out}}(V) \subseteq N(V)$ be the current set of outgoing neighbors of V. Let $N^{\text{un}}(V) \subseteq N(V)$ be the current set of unoriented neighbors of V. That is, $N(V) = N^{\text{in}}(V) \sqcup N^{\text{out}}(V) \sqcup N^{\text{un}}(V)$.

We define an algorithmic subroutine "Meek $R1(d)$" to orient all incident unoriented edges away from V whenever V already has d parents in a partially oriented graph. The reason for this naming is because it generalizes the idea behind the first of the four Meek rules [Mee95]; see Section 2.6.6 for details.

Our algorithm has three phases. In Phase 1, we orient strong v-structures. In Phase 2, we locally check if an edge is forced to orient one way or another to avoid incurring too much error. In Phase 3, we orient the remaining unoriented edges as a 1-polytree. Since the remaining edges were not forced, we may orient the remaining edges in an arbitrary direction (while not incurring "too much error") as long as the final incoming degrees of

Algorithm 7 RecoverOrientation: Algorithm for known skeleton and max in-degree.

```
Input: $c_0, \varepsilon > 0$, skeleton $\mathrm{skel}(\mathcal{G}^*)$, and max in-degree $d$
Output: A complete orientation of $\mathrm{skel}(\mathcal{G}^*)$
Run Phase1: Orient strong v-structures                               ▷ $\mathcal{O}(n^{d+1})$ time
Run Phase2: Local search and Meek $R1(d)$                             ▷ $\mathcal{O}(n^3)$ time
Run Phase3: Freely orient remaining unoriented edges                ▷ $\mathcal{O}(n)$ time via DFS
return $\widehat{\mathcal{G}}$
```

any vertex does not increase by more than 1. Subroutine Orient (Algorithm 8) performs the necessary updates when we orient $U - V$ to $U \to V$.

Algorithm 8 Orient: Subroutine to orient edges

```
Input: Vertices $U$ and $V$ where $U - V$ is currently unoriented
Orient $U - V$ as $U \to V$.
Update $N^{\mathrm{in}}(V)$ to $N^{\mathrm{in}}(V) \cup \{U\}$ and $N^{\mathrm{un}}(V)$ to $N^{\mathrm{un}}(V) \setminus \{U\}$.
Update $N^{\mathrm{out}}(U)$ to $N^{\mathrm{out}}(U) \cup \{V\}$ and $N^{\mathrm{un}}(U)$ to $N^{\mathrm{un}}(U) \setminus \{V\}$.
```

Algorithm 9 Phase1: Orient strong v-structures

```
Input: $c_0, \varepsilon > 0$, skeleton $\mathrm{skel}(\mathcal{G}^*)$, and max in-degree $d$
$\gamma \leftarrow d$
while $\gamma \geq 2$ do
    for $V \in \boldsymbol{V}$ do                                               ▷ Arbitrary order
        for $\boldsymbol{T} \in \mathcal{N}_\gamma$ do      ▷ $\mathcal{N}_\gamma \subseteq 2^{N(V)}$ are the $\gamma$ neighbors of $V$; $|\mathcal{N}_\gamma| = \binom{|N(V)|}{\gamma}$
            if $|\boldsymbol{T} \cup N^{\mathrm{in}}(V)| \leq d$ and $\widehat{I}(U; \boldsymbol{T} \setminus \{U\} \mid V) \geq c_0 \cdot \varepsilon, \forall U \in \boldsymbol{T}$ then
                for $U \in \boldsymbol{T}$ do                              ▷ Strong deg-$\gamma$ v-structure
                    Orient$(U, V)$
$\gamma \leftarrow \gamma - 1$                                              ▷ Decrement degree bound
```

Example Suppose we have the partially oriented graph Fig. 4.1(c) after Phase 1. Since $N^{\mathrm{in}}(D) = \{A, B\}$, we will check the edge orientations of $C - D$ and $F - D$. Since $I(F; \{A, B\} \mid D) = 0$, we will have $\widehat{I}(F; \{A, B\} \mid D) \leq \varepsilon$, so we will *not* erroneously orient $F \to D$. Meanwhile, $I(C; \{A, B\}) = 0$, we will have $\widehat{I}(C; \{A, B\}) \leq \varepsilon$, so we will *not* erroneously orient $D \to C$.

Remark 4.5. Note that within the for-loop from Line 7 of Phase2 (Algorithm 10), neither condition may hold, in which case we do not orient anything, hence the "missing" else.

When we freely orient a forest, we pick arbitrary root nodes in the connected components and orient to form a 1-polytree.

Algorithm 10 PHASE2: Local search and Meek $R1(d)$

```
Input: $c_0, \varepsilon > 0$, partially oriented graph, and max in-degree $d$
while True do                                          ▷ $\mathcal{O}(n)$ iterations, $\mathcal{O}(n^2)$ time per iteration
    if $\exists V \in \boldsymbol{V}$ such that $|N^{\text{in}}(V)| = d$ and $N^{\text{un}}(V) \neq \emptyset$ then        ▷ Meek $R1(d)$
        Orient all unoriented arcs away from $V$
        Update $N^{\text{out}}(V) \leftarrow N^{\text{out}}(V) \cup N^{\text{un}}(V)$; $N^{\text{un}}(V) \leftarrow \emptyset$
    for every node $V \in \boldsymbol{V}$ do
        if $1 \leq |N^{\text{in}}(V)| < d$ then
            for every $U \in N^{\text{un}}(V)$ do                            ▷ See Remark 4.5
                if $\widehat{I}(U; N^{\text{in}}(V) \mid V) > c_0 \cdot \varepsilon$ then
                    ORIENT($U$, $V$)
                else if $\widehat{I}(U; N^{\text{in}}(V)) > c_0 \cdot \varepsilon$ then
                    ORIENT($V$, $U$)
    if No new edges are being oriented then
        break
```

Algorithm 11 PHASE3: Freely orient remaining unoriented edges

```
Input: $c_0, \varepsilon > 0$, partially oriented graph, and max in-degree $d$
Let $\mathcal{H}$ be the forest induced by the remaining unoriented edges.
Freely orient $\mathcal{H}$ as a 1-polytree, i.e. maximum in-degree in $\mathcal{H}$ is 1.
Let $\widehat{\mathcal{G}}$ be the combination of the oriented $\mathcal{H}$ and the previously oriented arcs.
return $\widehat{\mathcal{G}}$
```

4.4.2 Analysis

We rely on the conclusions of Corollary 4.4 with error tolerance $\varepsilon' = \frac{\varepsilon}{2n \cdot (d+1)}$. Via a union bound over $\mathcal{O}(n^{d+1})$ events, Lemma 4.6 ensures that *all* our (conditional) MI tests in RECOVERORIENTATION (Algorithm 7) will behave as expected with probability at least $1 - \delta$, with sufficient samples. Full proofs are deferred to Appendix A.2.2.

Lemma 4.6. *Suppose all variables in the Bayesian network have alphabet Σ, for $|\Sigma| \geq 2$. For $\varepsilon' > 0$, $\mathcal{O}(n^{d+1})$ statements of the following forms all simultaneously succeed with probability at least $1 - \delta$:*
(1) If $I(\boldsymbol{X}; \boldsymbol{Y} \mid Z) = 0$, then $\widehat{I}(\boldsymbol{X}; \boldsymbol{Y} \mid Z) < c_0 \cdot \varepsilon'$,
(2) If $\widehat{I}(\boldsymbol{X}; \boldsymbol{Y} \mid Z) \leq c_0 \cdot \varepsilon'$, then $I(\boldsymbol{X}; \boldsymbol{Y} \mid Z) < \varepsilon'$.
with m empirical samples, where $Z \in V \cup \{\emptyset\}$, $\boldsymbol{X}, \boldsymbol{Y} \subseteq V \setminus \{Z\}$, $|\boldsymbol{X} \sqcup \boldsymbol{Y}| \leq d$, and

$$m \in \mathcal{O}\left(\frac{|\boldsymbol{\Sigma}|^{d+1}}{\varepsilon'} \cdot \log \frac{|\boldsymbol{\Sigma}|^{d+1} \cdot n^d}{\delta} \cdot \log \frac{|\boldsymbol{\Sigma}|^{d+1} \cdot \log(n^d/\delta)}{\varepsilon'}\right)$$

Proof. Set $\varepsilon' = \frac{\varepsilon}{2n \cdot (d+1)}$, use Corollary 4.4, and apply union bound over $\mathcal{O}(n^{d+1})$ tests. □

In the remaining of our analysis, we will analyze under the assumption that all our $\mathcal{O}(n^{d+1})$ tests are correct with the required tolerance level.

Recall that $\mathrm{Pa}(V)$ is the set of true parents of V in $\mathcal{G}^*$. Let $\mathcal{H}$ be the forest induced by the remaining unoriented edges after Phase 2 and $\widehat{\mathcal{G}}$ be returned graph of RecoverOrientation. Let us denote the final $N^{\text{in}}(V)$ as $\mathrm{Pa}^{\text{in}}(V)$ at the end of Phase 2, just before freely orienting, i.e. the vertices pointing into V in $\widehat{\mathcal{G}} \setminus \mathcal{H}$. Then, $\mathrm{Pa}^{\text{in}}(V) = \mathrm{Pa}(V) \setminus \mathrm{Pa}^{\text{in}}(V)$ is the set of ground truth parents that are not identified in both Phase 1 and Phase 2. Lemma 4.7 argues that the algorithm does not make mistakes for orientations in $\widehat{\mathcal{G}} \setminus \mathcal{H}$, so all edges in $\mathrm{Pa}^{\text{in}}(V)$ will be unoriented at the end of Phase 2.

Lemma 4.7. *Any oriented arc in $\widehat{\mathcal{G}} \setminus \mathcal{H}$ is a ground truth orientation. That is, any vertex parent set in $\widehat{\mathcal{G}} \setminus \mathcal{H}$ is a subset of* $\mathrm{Pa}(V)$*, i.e.* $\mathrm{Pa}^{\text{in}}(V) \subseteq \mathrm{Pa}(V)$*, and* $N^{\text{in}}(V)$ *at any time during the algorithm will have* $N^{\text{in}}(V) \subseteq \mathrm{Pa}^{\text{in}}(V)$*.*

Let $\widehat{\mathrm{Pa}}(V)$ be the proposed parents of V output by RecoverOrientation. The KL divergence between the true distribution and our output distribution is

$$\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}(V)) - \sum_{V \in \boldsymbol{V}} I(V; \widehat{\mathrm{Pa}}(V))$$

as the structure independent terms will cancel out. To get a bound on the KL divergence, we will upper bound $\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}(V))$ and lower bound $\sum_{V \in \boldsymbol{V}} I(V; \widehat{\mathrm{Pa}}(V))$.

To upper bound $\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}(V))$, we bounding each $I(V; \mathrm{Pa}(V))$ in terms of $\mathrm{Pa}^{\text{in}}(V) \subseteq \mathrm{Pa}(V)$ and $I(V; U)$ for $U \in \mathrm{Pa}^{\text{in}}(V)$ using Lemma 4.9, which relies on repeated applications of Lemma 4.8.

Lemma 4.8. *Fix any vertex V, any $\boldsymbol{S} \subseteq \mathrm{Pa}^{\text{in}}(V)$, and any $\boldsymbol{S}' \subseteq \mathrm{Pa}^{\text{in}}(V)$. If $\boldsymbol{S} \neq \emptyset$, then there exists a vertex $U \in \boldsymbol{S} \cup \boldsymbol{S}'$ with*

$$I(V; \boldsymbol{S} \cup \boldsymbol{S}') \leq I(V; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\}) + I(V; U) + \varepsilon \; . \tag{4.2}$$

Lemma 4.9. *For any vertex V with $\mathrm{Pa}^{\text{in}}(V)$, we can show that*

$$I(V; \mathrm{Pa}(V)) \leq \varepsilon \cdot |\mathrm{Pa}(V)| + I(V; \mathrm{Pa}^{\text{in}}(V)) + \sum_{U \in \mathrm{Pa}^{\text{in}}(V)} I(V; U) \; .$$

To lower bound $\sum_{V \in \boldsymbol{V}} I(V; \widehat{\mathrm{Pa}}(V))$, we rely on Lemma 4.10, which tells us that we lose at most an additive ε error per vertex in Phase 3, where we increase the incoming edges to any vertex by at most one. Note that orienting "freely" in Phase 3 could also increase the mutual information score and this is considering the worst case.

Lemma 4.10. *Consider an arbitrary vertex V with $\mathrm{Pa}^{\text{in}}(V)$ at the start of Phase 3. If Phase 3 orients $U \to V$ for some $U - V \in \mathcal{H}$, then*

$$I(V; \mathrm{Pa}^{\text{in}}(V) \cup \{U\}) \geq I(V; \mathrm{Pa}^{\text{in}}(V)) + I(V; U) - \varepsilon$$

Lemma 4.11. *Let* $\mathrm{Pa}(V)$ *be the true parents of* v*. Let* $\widehat{\mathrm{Pa}}(V)$ *be the proposed parents of* v *output by our algorithm. Then,*

$$\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}(V)) - \sum_{V \in \boldsymbol{V}} I(V; \widehat{\mathrm{Pa}}(V)) \leq n \cdot (d^* + 1) \cdot \varepsilon \ .$$

Note that Lemma 4.11 is a bound with respect to the true max-degree d^* despite only given an upper bound d as input. With these results in hand, we are ready to establish our main theorem.

Proof of Theorem 4.1. We first combine Lemma 4.11 and Lemma 4.6 with $\varepsilon' = \frac{\varepsilon}{2n \cdot (d+1)} \leq \frac{\varepsilon}{2n \cdot (d^*+1)}$ in order to obtain an orientation $\widehat{\mathcal{G}}$ which is close to $\mathcal{G}^*$. Now, much similar to the proof of [BGP+23, Theorem 1.4], we recall that there exist efficient algorithms for estimating the parameters of a Bayesian network with in-degree-d (note that this includes d-polytrees) $\mathcal{P}$ once a close-enough graph $\widehat{\mathcal{G}}$ is recovered [Das97, BGMV20], with sample complexity $\widetilde{\mathcal{O}}(n \cdot |\Sigma|^d / \varepsilon)$. Denote the final output $\widehat{\mathcal{P}}_{\widehat{\mathcal{G}}}$, a distribution that is estimated using the conditional probabilities implied by $\widehat{\mathcal{G}}$. One can bound the KL divergences as follows:

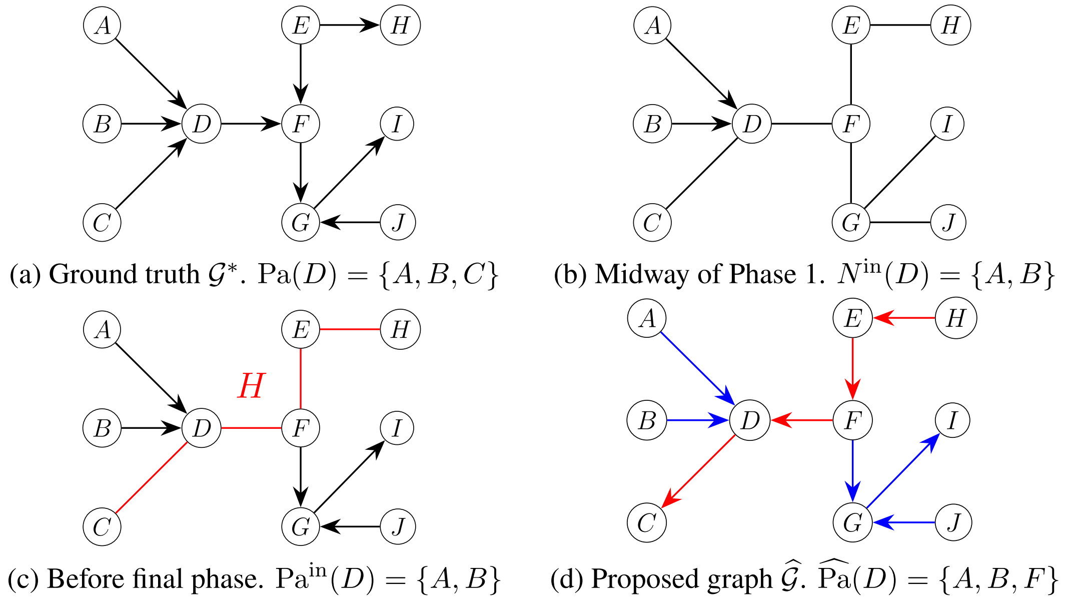
$$\mathrm{d_{KL}}(\mathcal{P}, \mathcal{P}_{\widehat{\mathcal{G}}}) - \mathrm{d_{KL}}(\mathcal{P}, \mathcal{P}_{\mathcal{G}^*}) \leq \varepsilon/2 \quad \text{and} \quad \mathrm{d_{KL}}(\mathcal{P}, \widehat{\mathcal{P}}_{\widehat{\mathcal{G}}}) - \mathrm{d_{KL}}(\mathcal{P}, \mathcal{P}_{\widehat{\mathcal{G}}}) \leq \varepsilon/2 \ .$$

The first inequality follows from our graph learning guarantees on $\widehat{\mathcal{G}}$ while the second is due to performing parameter learning algorithms on $\widehat{\mathcal{G}}$. Thus, $\mathrm{d_{KL}}(\mathcal{P}, \widehat{\mathcal{P}}_{\widehat{\mathcal{G}}}) \leq \varepsilon + \mathrm{d_{KL}}(\mathcal{P}, \mathcal{P}_{\mathcal{G}^*}) = \varepsilon$. □

4.5 Skeleton assumption

Here, we present a set of *sufficient* assumptions (Assumption 4.12) under which the Chow-Liu algorithm will recover the true skeleton even with finite samples. We note that the conditions listed here are in spirit very similar to the assumptions made to recover exact graphical structures in other works [GA21, GH17, GTA22], i.e., assuming a sufficiently detectable gap on an edge or from an alternate graph. Otherwise, it is not hard to find counter examples to thwart learners from recovering the correct network structure with finite sample access. For example, on a distribution on $X \rightarrow Y$ with infinitely small $I(X;Y)$, no algorithm can distinguish the actual graph from the empty graph given finite sample access. As such, it is often necessary to make these assumptions for exact structure recovery. Aside from the ones presented here, [BH20, CV22] study other sufficient conditions for recovering the skeleton of polytrees and Bayesian networks.

Nevertheless, we would like to highlight that results in this chapter has made progress in polytree PAC-learning in the following statisical sense: it suffices to have exact first



(a) Ground truth $\mathcal{G}^*$. $\mathrm{Pa}(D) = \{A, B, C\}$

(b) Midway of Phase 1. $N^{\mathrm{in}}(D) = \{A, B\}$

(c) Before final phase. $\mathrm{Pa}^{\mathrm{in}}(D) = \{A, B\}$

(d) Proposed graph $\widehat{\mathcal{G}}$. $\widehat{\mathrm{Pa}}(D) = \{A, B, F\}$

Figure 4.2: An example run to illustrate notations. In the ground truth graph $\mathcal{G}^*$, vertex D has parents $\mathrm{Pa}(D) = \{A, B, C\}$. While the algorithm executes, we track a tentative parent set $N^{\mathrm{in}}(D)$ of D and fix it to $\mathrm{Pa}^{\mathrm{in}}(D)$ right before the final phase. Since $d = 3$, observe that $G \to I$ must have been oriented due to a local search step and *not* due to Meek $R1(3)$ in Phase 2. At the end, in the proposed graph $\widehat{\mathcal{G}}$, the proposed parent set of D is $\widehat{\mathrm{Pa}}(D) = \{A, B, F\}$. Note that $\widehat{\mathcal{G}}$ only shows one possible orientation of the red unoriented subgraph $\mathcal{H}$ before the final phase; see Fig. 4.3 for others.

order mutual information and approximate higher order mutual information to learn (most) bounded in-degree polytrees in polynomial time. For prior works, it is only known that one can recover polytrees efficiently with exact first and second order mutual information [RP88] or exponential time algorithm for approximating bounded in-degree Bayesian networks [KCG+23].

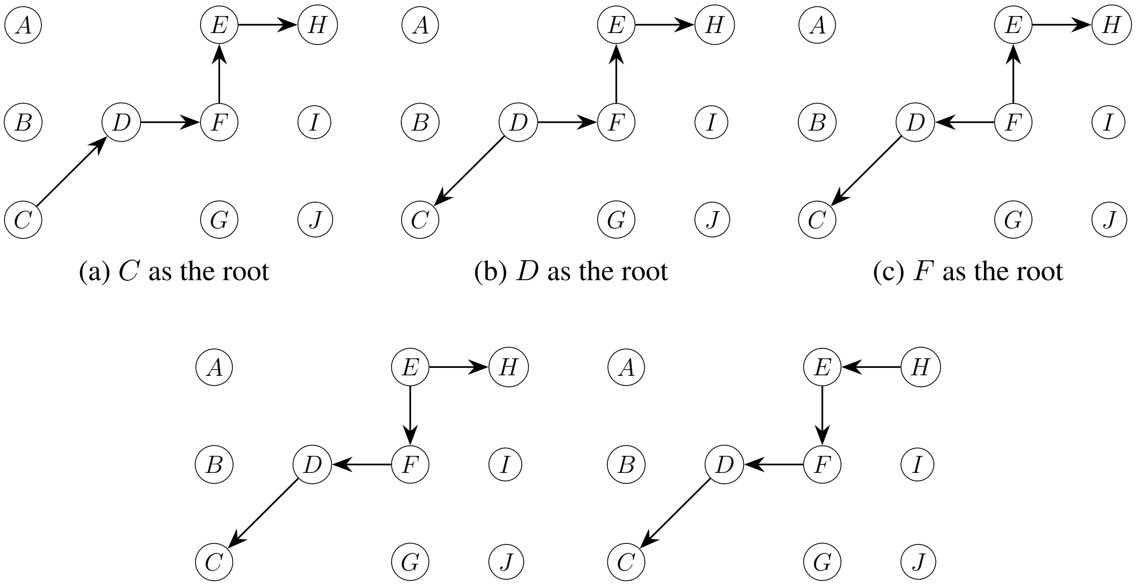
Assumption 4.12. For any given distribution $\mathcal{P}$, there exists a constant $\varepsilon_{\mathcal{P}} > 0$ such that:

1. For every pair of nodes U and V, if there exists a path $U - \cdots - V$ of length greater than 2 in $\mathcal{G}^*$, then then $I(U; V) + \varepsilon_{\mathcal{P}} \leq I(A; B)$ for every pair of adjacent vertices $A - B$ in the path.

2. For every pair of directly connected nodes $A - B$ in $\mathcal{G}^*$, $I(A; B) \geq \varepsilon_{\mathcal{P}}$.

Suppose there is a large enough gap of $\varepsilon_{\mathcal{P}}$ between edges in $\mathcal{G}^*$ and edges outside of $\mathcal{G}^*$. Then, with $\mathcal{O}(1/\varepsilon_{\mathcal{P}}^2)$ samples, each estimated mutual information $\widehat{I}(A; B)$ will be sufficiently close to the true mutual information $I(A; B)$. Thus, running the Chow-Liu algorithm (which is maximum spanning tree on the estimated mutual information on each pair of vertices) recovers $\mathrm{skel}(\mathcal{G}^*)$. See Appendix A.2.3 for the full proof.

Lemma 4.13. *Under Assumption 4.12, running the Chow-Liu algorithm on the m-sample empirical estimates $\{\widehat{I}(U; V)\}_{U,V \in \mathbf{V}}$ recovers a ground truth skeleton with high probability when $m \geq \Omega(\frac{\log n}{\varepsilon_{\mathcal{P}}^2})$.*



(a) C as the root (b) D as the root (c) F as the root

(d) E as the root (e) H as the root

Figure 4.3: The five different possible orientations of $\mathcal{H}$. Observe that the ground truth orientation of these edges is inconsistent with all five orientations shown here.

Combining Lemma 4.13 with RecoverOrientation (Algorithm 7), one can learn a polytree that is ε-close in KL with $\widetilde{\mathcal{O}}\left(\max\left\{\frac{\log(n)}{\varepsilon_{\mathcal{P}}^2}, \frac{2^d \cdot n}{\varepsilon}\right\}\right)$ samples, where $\varepsilon_{\mathcal{P}}$ depends on the distribution $\mathcal{P}$.

4.6 Lower bound

In this section, we show that $\Omega(n/\varepsilon)$ samples are necessary *even when a known skeleton is provided*. For constant in-degree d, this shows that our proposed algorithm in Section 4.4 is sample-optimal up to logarithmic factors.

We first begin by showing a lower bound of $\Omega(1/\varepsilon)$ on a graph with three vertices, even when the skeleton is given. Let $\mathcal{G}_1$ be $X \to Z \to Y$ and $\mathcal{G}_2$ be $X \to Z \leftarrow Y$, such that $\operatorname{skel}(\mathcal{G}_1) = \operatorname{skel}(\mathcal{G}_2)$ is $X - Z - Y$. Letting $\operatorname{Bern}(1/2)$ denote the Bernoulli distribution with parameter $1/2$, i.e. a fair coin flip, we define $\mathcal{P}_1$ and $\mathcal{P}_2$ as follows:

$$\mathcal{P}_1 : \begin{cases} X \sim \operatorname{Bern}\left(\frac{1}{2}\right) \\ Z = \begin{cases} X & \text{w.p. } \frac{1}{2} \\ \operatorname{Bern}\left(\frac{1}{2}\right) & \text{w.p. } \frac{1}{2} \end{cases} \\ Y = \begin{cases} Z & \text{w.p. } \sqrt{\varepsilon} \\ \operatorname{Bern}\left(\frac{1}{2}\right) & \text{w.p. } 1 - \sqrt{\varepsilon} \end{cases} \end{cases} \qquad \mathcal{P}_2 : \begin{cases} X \sim \operatorname{Bern}\left(\frac{1}{2}\right) \\ Y \sim \operatorname{Bern}\left(\frac{1}{2}\right) \\ Z = \begin{cases} X & \text{w.p. } \frac{1}{2} \\ Y & \text{w.p. } \sqrt{\varepsilon} \\ \operatorname{Bern}\left(\frac{1}{2}\right) & \text{w.p. } \frac{1}{2} - \sqrt{\varepsilon} \end{cases} \end{cases} \tag{4.3}$$

The intuition is that we keep the edge $X \to Z$ "roughly the same" and tweak the edge

$Y - Z$ between the distributions. By defining $\mathcal{P}_{i,\mathcal{G}}$ as projecting P_i onto $\mathcal{G}$, one can show Lemma 4.14; see Appendix A.2.4 for its proof.

Lemma 4.14 (Key lower bound lemma). *Let $\mathcal{G}_1$ be $X \to Z \to Y$ and $\mathcal{G}_2$ be $X \to Z \leftarrow Y$, such that $\operatorname{skel}(\mathcal{G}_1) = \operatorname{skel}(\mathcal{G}_2)$ is $X - Z - Y$. With respect to Eq. (4.3), we have the following:*

1. $\mathrm{d}_{\mathrm{H}}^2(\mathcal{P}_1, \mathcal{P}_2) \in \mathcal{O}(\varepsilon)$
2. $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_1, \mathcal{P}_{1,\mathcal{G}_1}) = 0$ *and* $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_1, \mathcal{P}_{1,\mathcal{G}_2}) \in \Omega(\varepsilon)$
3. $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_2, \mathcal{P}_{2,\mathcal{G}_2}) = 0$ *and* $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_2, \mathcal{P}_{2,\mathcal{G}_1}) \in \Omega(\varepsilon)$

Our hardness result (Lemma 4.15) is obtained by reducing the problem of finding an ε-close graph orientation of $X - Z - Y$ to the problem of *testing* whether the samples are drawn from $\mathcal{P}_1$ or $\mathcal{P}_2$. To ensure ε-closeness in the graph orientation, one has to correctly determine whether the samples come from $\mathcal{P}_1$ or $\mathcal{P}_2$ and then pick $\mathcal{G}_1$ or $\mathcal{G}_2$ respectively. Put differently, if one can solve the problem in Lemma 4.15, then one can use that algorithm to solve the problem in Lemma 4.14. However, it is well-known that distinguishing two distributions whose squared Hellinger distance is ε requires $\Omega(1/\varepsilon)$ samples (e.g. see [BY02, Theorem 4.7]).

Lemma 4.15. *Even when given $\operatorname{skel}(\mathcal{G}^*)$, it takes $\Omega(1/\varepsilon)$ samples to learn an ε-close graph orientation of $\mathcal{G}^*$ for distributions on $\{0,1\}^3$.*

Proof. Consider the construction in Lemma 4.14. To ensure ε-closeness in the graph orientation, one has to correctly determine whether the samples come from $\mathcal{P}_1$ or $\mathcal{P}_2$ and then pick $\mathcal{G}_1$ or $\mathcal{G}_2$ respectively. This requires $\Omega(1/\varepsilon)$ samples. □

Using the above construction as a gadget, we can obtain a dependency on n in our lower bound by constructing $n/3$ independent copies of the above gadget, à la proof strategy of [BGP+23, Theorem 7.6]. For some constant $c > 0$, we know that a constant $1/c$ fraction of the gadgets will incur an error or more than ε/n if less than cn/ε samples are used. The desired result then follows from the tensorization of KL divergence, i.e., $\mathrm{d}_{\mathrm{KL}}\left(\prod_i \mathcal{P}_i, \prod_i \mathcal{Q}_i\right) = \sum_i \mathrm{d}_{\mathrm{KL}}(\mathcal{P}_i, \mathcal{Q}_i)$.

Theorem 4.2. *Let $\varepsilon \in (0,1)$ be the error parameter. There exists a choice of distribution $\mathcal{P}$ over $\{0,1\}^n$ that is Markov with respect to some 2-polytree $\mathcal{G}^*$ such that producing $\widehat{\mathcal{P}}$ such that $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ with success probability at least $2/3$ requires $\Omega(n/\varepsilon)$ samples from $\mathcal{P}$, even given we are given $\operatorname{skel}(\mathcal{G}^*)$ as input.*

Proof. Consider a distribution $\mathcal{P}$ on $n/3$ independent copies of the lower bound construction from Lemma 4.15, where each copy is indexed by $\mathcal{P}_i$ for $i \in \{1, \ldots, n/3\}$. Suppose, for a contradiction, that the algorithm draws cn/ε samples for sufficiently small

$c > 0$, and manages to output $\mathcal{Q}$ that is ε-close to $\mathcal{P}$ with probability at least $2/3$. From Lemma 4.15 with error tolerance $\Omega(\varepsilon/n)$, we know that each copy is *not* $\Omega(\varepsilon/n)$-close with probability at least $1/5$. By Chernoff bound, at least $\Omega(n)$ copies are *not* $\Omega(\varepsilon/n)$-close with probability at least $2/3$. Then, by the tensorization of KL divergence, we see that $\mathrm{d}_{\mathrm{KL}}\left(\prod_{i=1}^{n/3} \mathcal{P}_i \parallel \prod_{i=1}^{n/3} \mathcal{Q}_i\right) = \sum_{i=1}^{n/3} \mathrm{d}_{\mathrm{KL}}(\mathcal{P}_i, \mathcal{Q}_i) > \Omega(\varepsilon)$. This contradicts the assumption that $\mathcal{Q}$ is ε-close to $\mathcal{P}$ with probability at least $2/3$. □

Chapter 5

Conclusion for Part I

The results presented in Chapter 3 and Chapter 4 are from the works of [BCG+22] and [CYBC24] respectively.

In Chapter 3, we presented a coefficient recovery algorithm LeastSquares based on node-wise linear least squares regression. Formal details of our lower bound is presented in [BCG+22, Section 5]. We actually provide and analyze a generalization dubbed BatchAvgLeastSquares in [BCG+22] by allowing any interpolation between "batch size" and "number of batches" — LeastSquares is a special case of a single batch. In a nutshell, for each variable with $p \geq 1$ parents, BatchAvgLeastSquares solves $b \geq 1$ batches of linear systems made up of $k > p$ samples and then uses the *mean* of the recovered solutions as an estimate for the coefficients. Since each solution to batch can be computed independently before their results are combined, BatchAvgLeastSquares facilitates further parallelism. In view of CauchyEst, a natural question is whether one can use a coordinate-wise *median* of these recovered batch coefficients in order to be robust towards sample contamination. While we did not provide formal analysis, our empirical evaluation suggests that such taking the median does indeed improve robustness against certain forms of data contamination; see [BCG+22, Section 6].

In Chapter 4, we studied the problem of estimating a distribution defined on a d^*-polytree $\mathcal{P}$ with graph structure $\mathcal{G}^*$ using finite observational samples. We designed and analyzed an efficient algorithm that produces an estimate $\widehat{\mathcal{P}}$ such that $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ assuming access to $\mathrm{skel}(\mathcal{G}^*)$ and an upper bound d of d^*. The skeleton $\mathrm{skel}(\mathcal{G}^*)$ is recoverable under Assumption 4.12 and we show that there is an inherent hardness in the learning problem even under the assumption that $\mathrm{skel}(\mathcal{G}^*)$ is given. For constant d, our hardness result shows that our proposed algorithm is sample-optimal up to logarithmic factors. Our algorithm in Chapter 4 heavily relies on the realizability assumption that $\mathcal{P}$ is indeed Markov to some d-polytree $\mathcal{G}^*$ on n nodes. This assumption was subsequently removed in [BGGJ+24] with a dynamic programming approach that uses roughly a multiplicatve factor of $\widetilde{\mathcal{O}}(n^2)$ additional samples.

It is natural to ask whether what we can do with access to a false skeleton that is approximately correct (i.e. has some orientation close in KL to the ground truth) produced by running the Chow-Liu algorithm on the sample statistics. However, it is unclear to us why we can hope to design efficient algorithms with provable guarantees in this case for two reasons:

- The Chow-Liu algorithm only uses order-1 mutual information while the KL divergence of Eq. (4.1) requires information from order-d mutual information. It is unclear why one can hope that this false skeleton would yield provable guarantees with respect to Eq. (4.1).
- An "approximately correct" skeleton may have potentially unknown number of edges in the skeleton being wrong and we do not see how to design efficient global orientation algorithms using only statistics from the ground truth samples.

Without the true skeleton, a "local algorithm" (such as ours) can be tricked into some "local optima" and it is hard to argue why the output would obtain "global guarantees" with respect to the parent sets of Eq. (4.1).

Another interesting open question is whether one can extend the hardness result to arbitrary $d \geq 1$, or design more efficient learning algorithms for d-polytrees. In particular, we are unaware of any obstruction a lower bound for $|\Sigma| > 2$ and $d > 2$. While we do not know an optimal construction, the following construction (emulating Appendix A.2 of [CDKS17]) yields $\Omega(\frac{n2^d}{(d+1)\varepsilon^2})$, showing that the exponential dependence on d is unavoidable when learning the parameters of a given d-polytree. Consider $\frac{n}{d+1}$ stars with binary alphabets, where each star center has d incoming parents. Each parental node is set to be an independent uniform coin flip over the binary alphabet and so it takes $\Omega(2^d/\varepsilon^2)$ to learn each star to accuracy ε. As KL is additive, one would require any constant fraction of the stars to incur less than $\frac{\varepsilon(d+1)}{n}$ error. To do so, one would need $\Omega(\frac{n2^d}{(d+1)\varepsilon^2})$ samples.

5.1 Some additional related work

[Das97] first looked at the problem of parameter learning for fixed structure Bayesian networks in the discrete and continuous settings and gave finite sample complexity bounds for these problems based on the VC-dimensions of the hypothesis classes. In particular, he gave an algorithm for learning the parameters of a Bayesian network on n binary variables of bounded in-degree in d_{KL} distance using a quadratic in n samples. Subsequently, tight (linear) sample complexity upper and lower bounds were shown for this problem [BGMV20, BGP$^+$23, CDKS17]. To the best of our knowledge, a finite PAC-style bound for fixed-structure Gaussian Bayesian networks was not known previously.

Structure learning of Bayesian networks is an old problem in machine learning and statistics that has been intensively studied, e.g. see [KF09, Chapter 18]. Many early approaches required faithfulness, a condition which permits learning of the Markov equivalence class, e.g. [SG91, FNP99, Chi03]. Finite sample complexity of such algorithms assuming faithfulness-like conditions has also been studied, e.g. [FY96]. An alternate line of more modern work has considered various other distributional assumptions that permits for efficient learning, e.g. [CM02, HJM$^+$08, SHHK06, PB14, GH17, PR18, AAZ19], with the final three also showing finite sample complexities. Specifically for polytrees, [RP88] and [GPP90] studied recovery of the DAG for polytrees under the infinite sample regime.

[AKN06] studied the problem of efficiently learning a bounded degree factor graph. Using their method and conversion scheme between factor graphs and Bayesian networks, one could efficiently learn polytrees (Bayesian networks) with bounded in- *and* out-degrees. However, as we only consider an upper bound on the in-degrees in Chapter 4, directly applying their method scales badly in sample complexity (exponential in the number of variables) for even the simple star-like polytree: a center node V with undirected edges to the rest of the $n-1$ nodes such that V's in-degree is d and out-degree is $n-d-1$.

More recently, [GA21] studied the more general problem of learning Bayesian networks, and their sufficient conditions simplified in the setting of polytrees. Their approach emphasizes exact recovery, and thus the sample complexity has to depend on the minimum gap of some key mutual information terms. In contrast, we allow the algorithm to make mistakes when certain mutual information terms are too small to detect for the given sample complexity budget and achieve a PAC-type guarantee. As such, once the underlying skeleton is discovered, our sample complexity only depends on the d, n, ε and not on any distributional parameters.

There are also existing works on Bayesian network learning with tight bounds in total variation distance with a focus on sample complexity (and not necessarily computational efficiency), e.g. [CDKS17]. Meanwhile, [ABDK18] consider the problem of learning (in TV distance) a bounded-degree causal Bayesian network from interventions, assuming the underlying DAG is known.

5.2 Other unpresented works in Part I

In [DDKC23], we provide time and sample efficient algorithms for learning and testing latent-tree Ising models, i.e. Ising models that may only be observed at their leaf nodes. On the learning side, we obtain efficient algorithms for learning a tree-structured Ising model whose leaf node distribution is close in TV distance, improving on the results of [CGG01]. On the testing side, we provide an efficient algorithm with fewer samples for testing whether two latent-tree Ising models have leaf-node distributions that are close or far in TV distance. We obtain our algorithms by showing novel localization results for the total variation distance between the leaf-node distributions of tree-structured Ising models, in terms of their marginals on pairs of leaves.

Given data, computing a "score maximizing" DAG is known to be NP-hard [Chi96]. Furthermore, [CHM04] showed that deciding whether a given distribution $\mathcal{P}$ is Markov with respect to some Bayesian network of at most $p \in \mathbb{N}$ parameters or not is NP-hard. In [BCGM25], we extend the hardness result of [CHM04] to the setting where we are guaranteed that the Bayesian network in question is promised to have a small number of parameters. In computational complexity theory, this is also known as a *promise problem*, which generalizes a decision problem in that the input is promised to belong to a certain subset of possible inputs. Our new hardness result confirms the common intuition that it is hard to search for a Bayesian network $\mathcal{G}$ that is Markov with respect to a given probability distribution, even if it is known that the distribution in question is Markov with respect to a Bayesian network that has a small number of parameters. In [BCGM25], we also generalized the finite sample result of [BCD20] for producing a TV-close estimate $\widehat{\mathcal{P}}$ of $\mathcal{P}$, from the *degree-bounded* setting to the *parameter-bounded* setting.

Part II

Learning causal models

Chapter 6

Causal graph discovery with adaptive interventions

> "No causation without manipulation."
>
> - Paul Holland and Donald Rubin [Hol86]

6.1 Introduction

In this chapter, we study the problem of recovering the true underlying causal graph using adaptive interventions under some standard causal assumptions in the causal graph discovery literature. To be precise, suppose the true underlying DAG generating the data is $\mathcal{G}^* = (\boldsymbol{V}, \boldsymbol{E})$ belonging to the Markov equivalence class $[\mathcal{G}^*]$ with corresponding essential graph $\mathcal{E}(\mathcal{G}^*)$. Under the following causal assumptions, we aim to fully orient $\mathcal{E}(\mathcal{G}^*)$ into $\mathcal{G}^*$ by performing adaptive interventions:

1. Causal sufficiency, i.e. no unobserved variables or hidden confounders.
2. We are given access to the essential graph of the true causal graph, or equivalently we know its Markov equivalence class.
3. When we perform interventions on a subset of variables, we recover the orientations of edges with exactly one endpoint amongst the intervened variables.

The third assumption enable us to abstract the above problem of causal graph learning into a graph problem with specialized graph operations: first orient edges separated by interventions, then apply Meek rules (Section 2.6.6) till convergence. This assumption holds in the setting where we perform ideal/hard interventions. See Section 2.8.2 for further discussion of these assumptions.

Problem 6.1 (The search problem)**.** Given the essential graph $\mathcal{E}(\mathcal{G}^*)$ of an unknown underlying causal graph $\mathcal{G}^*$, use the minimal number of interventions to fully recover the ground truth causal graph $\mathcal{G}^*$.

Example 6.2. Fig. 6.1 gives an example DAG $\mathcal{G}^*$ on 6 variables, along with its observational essential graph $\mathcal{E}(\mathcal{G}^*)$ and Markov equivalence class $[\mathcal{G}^*]$. See Fig. 2.3 for an example on how to compute an essential graph given a DAG. In this example, a single intervention on $\{A\}$ will suffice to orient $\mathcal{E}(\mathcal{G}^*)$ into $\mathcal{G}^*$ with the help of Meek rules (Section 2.6.6).

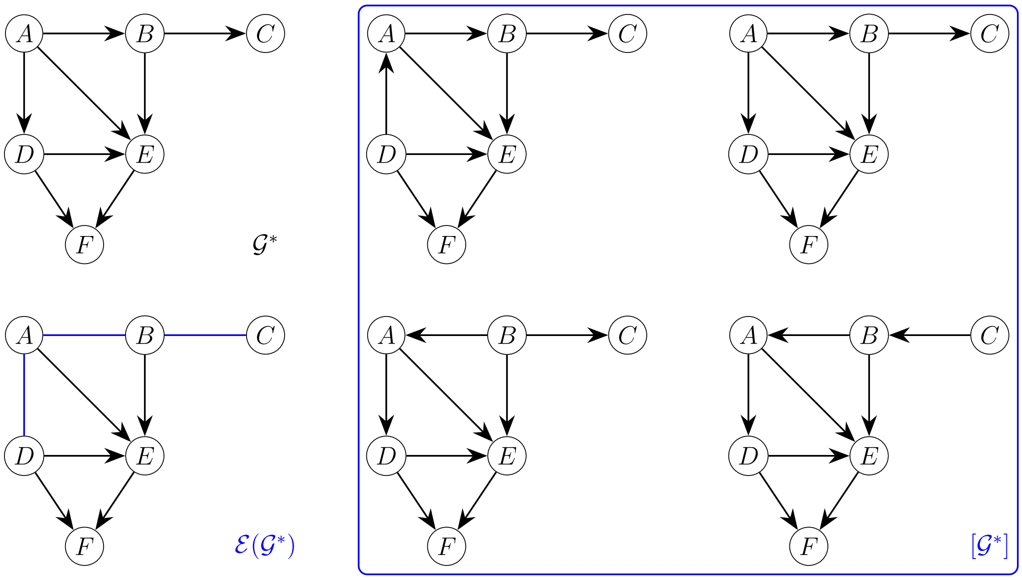


Figure 6.1: A causal graph $\mathcal{G}^*$ and its partially oriented essential graph $\mathcal{E}(\mathcal{G}^*)$ obtained from observational data, where there is uncertainty in the 3 unoriented blue edges. Here, $\mathcal{E}(\mathcal{G}^*)$ represents 4 possible DAGs in $[\mathcal{G}^*]$, the Markov equivalence class of $\mathcal{G}^*$.

Besides minimizing the number of interventions performed, many applications care about recovering only a *subset* of the causal relationships. For instance, in *local* causal graph discovery, efficient learning of localized causal relationships play a central role in feature selection via Markov blankets [ATS03, TA03, MC04, AST$^+$10a] while scalability is of significant concern when one only wishes to learn localized causal effects (e.g. the direct causes and effects of a target variable of interest) [SMH$^+$15, FMT$^+$21] within a potentially large causal graph (e.g. gene regulatory networks [LD05]). Meanwhile, in the context of designing algorithms that generalize to novel distributions [ABGLP19, LWHLS22], it suffices to just learn the causal relationship between the target variable and feature/latent variables while ignoring all other causal relationships. Furthermore, in practice, there may be constraints on the interventions that one can perform and it is natural to prioritize the recovery of important causal relationships. As such, in many practical situations, one is interested in learning the causal relationship only for a subset of the edges of the causal graph while minimizing the number of interventions.

Now, given two algorithms that solve Problem 6.1, how should we compare and decide which algorithm is better? A natural comparison metric to use in this case is to compare how well an algorithm performs against an all-knowing oracle that *knows* the true underlying DAG — how many interventions would such an oracle require us to perform in order to convince us of the ground truth?

Problem 6.3 (The verification problem)**.** Given the essential graph $\mathcal{E}(\mathcal{G}^*)$ of an unknown underlying causal graph $\mathcal{G}^*$ and a proposed graph $\mathcal{G} \in [\mathcal{G}^*]$, use the minimal number of interventions to verify whether $\mathcal{G} \overset{?}{=} \mathcal{G}^*$.

To tackle both problems, one needs to compute a collection of interventions that will completely orient a given essential graph. For any DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, we call such a set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ a *verifying set* for $\mathcal{G}$. Each element $\boldsymbol{I} \in \mathcal{I}$ represents a subset of vertices on which an ideal intervention will be conducted, recovering the orientations of separated edges and any implied edge orientations due to Meek rules. In other words, for any graph $\mathcal{G}$ and any verifying set $\mathcal{I}$ of $\mathcal{G}$, we have $\mathcal{E}_{\mathcal{I}}(\mathcal{G})[\boldsymbol{V}'] = \mathcal{G}[\boldsymbol{V}']$ for *any* subset of vertices $\boldsymbol{V}' \subseteq \boldsymbol{V}$. Furthermore, if $\mathcal{I}$ is a verifying set for $\mathcal{G}$, then $\mathcal{I} \cup \{\boldsymbol{S}\}$ is also a verifying set for $\mathcal{G}$ for any additional intervention $\boldsymbol{S} \subseteq \boldsymbol{V}$. This is because interventions do not remove information and oriented arcs always remain oriented.

Definition 6.4 (Minimum size verifying set)**.** An intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ is called a verifying set for a DAG $\mathcal{G}^* = (\boldsymbol{V}, \boldsymbol{E})$ if $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*) = \mathcal{G}^*$ and is said to have minimum size if $\mathcal{E}_{\mathcal{I}'}(\mathcal{G}^*) \neq \mathcal{G}^*$ for any $\mathcal{I}' \subseteq 2^{\boldsymbol{V}}$ such that $|\mathcal{I}'| < |\mathcal{I}|$.

Note that there may be multiple minimum sized verifying sets of minimum size. This motivates the definition of a verification number $\nu_k(\mathcal{G})$ to denote the minimum sized verifying set; $\nu_1(\mathcal{G})$ refers the special case where only atomic interventions are allowed.

Definition 6.5 (Verification number)**.** Given $k \in \mathbb{N}^+$, the verification $\nu_k(\mathcal{G})$ of a DAG $\mathcal{G}$ is defined as $\nu_k(\mathcal{G}) = |\mathcal{I}|$, where $\mathcal{I}$ is a minimum size verifying set for $\mathcal{G}$ such that any intervention in $\boldsymbol{I} \in \mathcal{I}$ involves at most $|\boldsymbol{I}| \leq k$ vertices, i.e. k-bounded interventions.

The verification number is a useful analytical tool for the search problem as $\nu_k(\mathcal{G}^*)$ is a lower bound on the number of interventions used by an optimal search algorithm. Furthermore, since the search problem needs to fully orient $\mathcal{E}(\mathcal{G}^*)$ regardless of which DAG is the ground truth, any search algorithm given $\mathcal{E}(\mathcal{G}^*)$ requires *at least* $\min_{\mathcal{G} \in [\mathcal{G}^*]} \nu_k(\mathcal{G})$ interventions, even if the algorithm is adaptive and randomized. In fact, the *strongest possible universal lower bound* guarantee one can prove must be *at most* $\min_{\mathcal{G} \in [\mathcal{G}^*]} \nu_k(\mathcal{G})$ and the *strongest possible universal upper bound* guarantee one can prove must be *at least* $\max_{\mathcal{G} \in [\mathcal{G}^*]} \nu_k(\mathcal{G})$. Note that if the search algorithm is *non-adaptive*, then it trivially needs *at least* $\max_{\mathcal{G} \in [\mathcal{G}^*]} \nu_k(\mathcal{G})$ interventions.

Remark 6.6. In this chapter, we mainly focus on our results for atomic interventions and only briefly provide some tools and results to lift these results to the non-atomic setting of k-bounded interventions. We will also assume that all interventions have unit cost and focus on the goal of obtaining intervention sets of minimum size. In Chapter 8, we discuss several other results and extensions that we have also studied in this problem space.

6.2 Our main results

The main technical contributions of this chapter are the characterization of verifying sets, and efficient algorithms for solving the verification and search problems that are competitive with the verification number.

6.2.1 Characterization of verifying sets

Theorem 6.7. *Fix a DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$. An intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ is a minimum sized verifying set for $\mathcal{G}$ if and only if every covered edge of $\mathcal{G}$ is separated by some intervention in $\mathcal{I}$.*

Theorem 6.7 provides a formal justification to the observation of [SMG+20] that "In general, the size of an [atomic verifying set] cannot be calculated from just its essential graph". This is because essential graphs could imply minimum vertex covers of different sizes (see Fig. 6.4). The following examples illustrate applications of Theorem 6.7 to some classes of special graphs.

Example 6.8 (Directed cliques). Consider the directed clique $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ on n vertices given in Fig. 6.2 where the direct child arcs (indicated by dashed arrows) are precisely the covered edges in $\mathcal{G}$. Theorem 6.7 tells us that $\nu_1(\mathcal{G}) = \lfloor n/2 \rfloor$ where one can intervene atomically on a minimum vertex cover of the path induced by these covered edges.

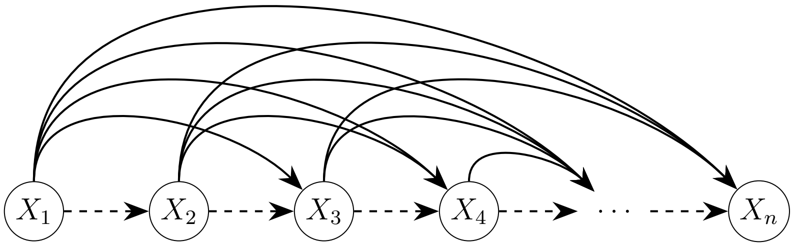


Figure 6.2: A directed clique $\mathcal{G}$ on n vertices where the direct child arcs (indicated by dashed arrows) are precisely the covered edges in $\mathcal{G}$; see Lemma 6.15. As there are no v-structures, the essential graph is completely unoriented. Here, $\nu_1(\mathcal{G}) = \lfloor n/2 \rfloor$.

Example 6.9 (Directed trees). Consider the directed tree $\mathcal{G}$ given in Fig. 6.3 where $R \in \boldsymbol{V}$ is the root vertex. One can easily verify that the only covered edges (indicated by dashed arrows) in $\mathcal{G}$ are arcs leaving R. Theorem 6.7 tells us that $\nu_1(\mathcal{G}) = 1$ since intervening on $\{R\}$ would orient these covered edges and Meek R1 would orient the remaining edges.

Example 6.10 (Standing windmill). Consider the graph $\mathcal{G}^*$ in Fig. 6.4 (a replication of Fig. 2.4) where the essential graph $\mathcal{E}(\mathcal{G}^*)$ representing the MEC $[\mathcal{G}^*]$ is the standing windmill[11]. One can check that $\nu_1(\mathcal{G}^*) = \nu_1(\mathcal{G}_1) = 4$ while $\nu_1(\mathcal{G}_2) = 3$. In fact, we

[11] To be precise, it is the Wd(3,3) windmill graph with an additional edge from the center.

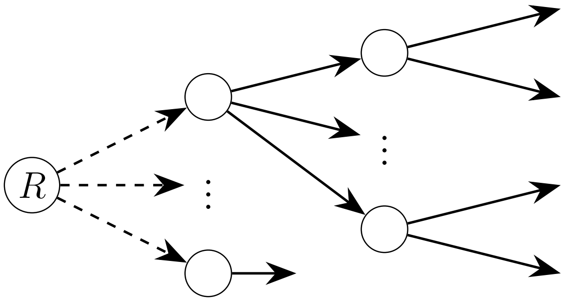


Figure 6.3: A directed tree $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ with root vertex $R \in \boldsymbol{V}$. One can easily verify that the only covered edges (indicated by dashed arrows) in $\mathcal{G}$ are arcs leaving R. As there are no v-structures, the essential graph is completely unoriented. Here, $\nu_1(\mathcal{G}) = 1$.

actually show that $\min_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G}) = 3$ and $\max_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G}) = 4$ in Appendix B.1.1. Thus, any search algorithm using only atomic interventions on $\mathcal{E}(\mathcal{G}^*)$ needs at least 3 atomic interventions.

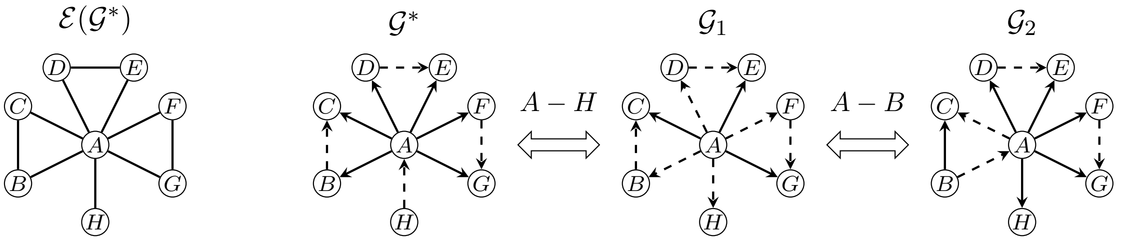


Figure 6.4: A DAG $\mathcal{G}^*$ with its essential graph $\mathcal{E}(\mathcal{G}^*)$ on the left. $\mathcal{G}_1$ and $\mathcal{G}_2$ are two other DAGs that belong to the same Markov equivalence class $[\mathcal{G}^*]$. Note that the sizes of the minimum vertex cover of the covered edges (dashed arcs) may differ across DAGs.

Recall from Lemma 2.49 that any undirected edge in $\mathcal{E}(\mathcal{G}^*)$ is a covered edge for *some* $\mathcal{G} \in [\mathcal{G}^*]$. So, Theorem 6.7 implies a simple alternative proof for an earlier known result that characterizes non-adaptive search algorithms via separating systems [HEH13, SKDV15]: any non-adaptive search algorithm, which has *no* knowledge of $\mathcal{G}^*$, should separate *every* undirected edge in $\mathcal{E}(\mathcal{G}^*)$.

Through the lens of covered edges, we can also see that existing universal bounds of [SMG+20, PSS22] are *not* tight. Consider the case where the essential graph $\mathcal{E}(\mathcal{G}^*)$ is the standing windmill graph given in Fig. 6.4. The graph $\mathcal{E}(\mathcal{G}^*)$ has $n = 8$ nodes, $r = 4$ maximal cliques and the largest maximal clique is size 3. The lower bound of [SMG+20] yields $\sum_{\mathcal{H} \in CC(\mathcal{E}(\mathcal{G}^*))} \lfloor \frac{\omega(H)}{2} \rfloor = \lfloor \frac{3}{2} \rfloor = 1$ while lower bound of [PSS22] yields $\lceil \frac{n-r}{2} \rceil = \lceil \frac{8-4}{2} \rceil = 2$. Meanwhile, we show $\min_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G}) = 3$ in Appendix B.1.1.

Another immediate application of Theorem 6.7 is to resolve the verification problem. Specifically, for the hardness analysis, observe that if one performs strictly less interventions than the intervention number, then there will be a covered edge which is not separated and thus the graph will not be fully oriented.

Corollary 6.11. *Given an essential graph $\mathcal{E}(\mathcal{G}^*)$ of an unknown ground truth DAG $\mathcal{G}^*$ and a causal DAG $\mathcal{G} \in [\mathcal{G}^*]$, we can test if $\mathcal{G} \overset{?}{=} \mathcal{G}^*$ by intervening on any verifying set of $\mathcal{G}$. Furthermore, in the worst case,* any *algorithm that correctly resolves $\mathcal{G} \overset{?}{=} \mathcal{G}^*$ using k-bounded interventions needs at least $\nu_k(\mathcal{G})$ interventions.*

6.2.2 Verification

Theorem 6.12. *A minimum sized atomic verifying set for $\mathcal{G}$ can be computed in polynomial time in the size of $\mathcal{G}$.*

Theorem 6.12 provides the first efficient algorithm for computing minimum sized atomic verifying set for general graphs. Prior to this result, efficient algorithms for computing minimum sized atomic verifying sets were only known for simple graphs such as cliques and trees. For general graphs, only a brute force algorithm is known [SMG$^+$20, Appendix F] which takes exponential time in the worst case. Meanwhile, in contrast to computing a minimum sized verifying set, [PSS22] provides an efficient algorithm that returns a verifying set of size at most 2 times that of the optimum.

6.2.3 Search

Theorem 6.13. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ with an unknown underlying ground truth DAG $\mathcal{G}^*$. There is an algorithm that runs in polynomial time and computes an atomic intervention set $\mathcal{I} \subseteq 2^{\mathbf{V}}$ in a deterministic and adaptive manner such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*) = \mathcal{G}^*$ and $|\mathcal{I}| \in \mathcal{O}(\log(n) \cdot \nu_1(\mathcal{G}^*))$.*

Since *any* search algorithm will incur at least $\nu_1(\mathcal{G}^*)$ interventions, Theorem 6.13 implies that search is (almost, up to $\log n$ multiplicative factor) as easy as the verification. This result is the first competitive results that holds for using atomic interventions on *general graphs*. The only previously known result of $\mathcal{O}(\log_2(\max_{\mathcal{H} \in CC(\mathcal{E}(\mathcal{G}^*))} \omega(\mathcal{H})) \cdot \nu_1(\mathcal{G}^*))$ by [SMG$^+$20] was an algorithm based on directed clique trees with provable guarantees only for atomic interventions on intersection-incomparable chordal graphs.

The approximation of $\mathcal{O}(\log n)$ to $\nu_1(\mathcal{G}^*)$ is the tightest one can hope for atomic interventions in general. For instance, consider the case where $\mathcal{E}(\mathcal{G}^*)$ is an undirected line graph on n vertices. Then, using a similar reasoning as the lower bound for binary search, one can show that *any* adaptive algorithm needs $\Omega(\log n)$ atomic interventions in the worst case while $\nu_1(\mathcal{G}^*) = 1$. The line graph also provides a clear distinction between adaptive and non-adaptive search algorithms since *any* non-adaptive algorithm needs $\Omega(n)$ atomic interventions to separate all the edges in $\mathcal{E}(\mathcal{G}^*)$.

6.2.4 Extensions to subset versions and k-bounded interventions

We also generalize our verification and search results to the setting where one is only concerned about recovering orientations of a subgraph of interest, and to the setting with k-bounded interventions. To the best of our knowledge, our work provides the first known efficient algorithms for computing near-optimal verifying sets and performing subset search with provable guarantees on general graphs.

6.3 Technical overview

Our first technical tool is to invoke the following property about interventional essential graphs so that we can focus on instances where $\mathcal{G}^*$ is a moral DAG when studying the verification and search problems under interventions.

Theorem 6.14 (Properties of interventional essential graphs)**.** *Fix a DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$. For any intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ and any vertex $U \in \boldsymbol{V}$, let $\boldsymbol{R}(\mathcal{G}, \mathcal{I}) \subseteq \boldsymbol{E}$ denote the set of oriented arcs in the $\mathcal{I}$-essential graph of $\mathcal{G}$, $\mathcal{G}^{\mathcal{I}}$ be the fully directed subgraph DAG of $\mathcal{G}$ obtained by arcs in $\boldsymbol{R}(\mathcal{G}, \mathcal{I})$, and $\mathrm{Pa}_{\mathcal{G},\mathcal{I}}(U) = \{X \in V : X \to U \in \boldsymbol{R}(\mathcal{G}, \mathcal{I})\}$ be the parents of U recovered by $\mathcal{I}$. The following statements are true with respect to any two arbitrary intervention sets $\mathcal{A} \subseteq 2^{\boldsymbol{V}}$ and $\mathcal{B} \subseteq 2^{\boldsymbol{V}}$:*

1. *Any v-structures in $\mathcal{G}^{\mathcal{A}}$ are also present in $\mathcal{G}$.*
2. *Any acyclic completion of $\mathcal{E}(\mathcal{G}^{\mathcal{A}})$ that does not form new v-structures can be combined with $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$ to obtain a valid DAG belonging to both $\mathcal{E}(\mathcal{G})$ and $\mathcal{E}_{\mathcal{A}}(\mathcal{G})$.*
3. *$\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) = \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{A})$.*
4. *$\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}, \mathcal{A})$.*
5. *$\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) \sqcup (R(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B}))$.*
6. *$\boldsymbol{R}(\mathcal{G}, \emptyset)$ does not contain any covered edge of $\mathcal{G}$.*

An important implication of Theorem 6.14 for verification and search problems is that it suffices to solve these problems only on moral DAGs without v-structures. This is because any oriented arcs in the observational graph can be removed *before performing any interventions* as the optimality of the solution is unaffected: $\boldsymbol{R}(\mathcal{G}, \mathcal{I}) = \boldsymbol{R}(\mathcal{G}^{\emptyset}, \mathcal{I}) \sqcup \boldsymbol{R}(\mathcal{G}, \emptyset)$, where $\mathcal{G}^{\emptyset}$ is the graph obtained after removing all the oriented arcs in the observational essential graph due to v-structures. In other words, w.l.o.g., we can focus on instances where $\mathcal{G}^*$ is a moral DAGs when studying the verification and search problems under interventions. Fig. 6.5 gives an illustration example. Note that Theorem 6.14 holds even when the intervention sets are non-atomic.

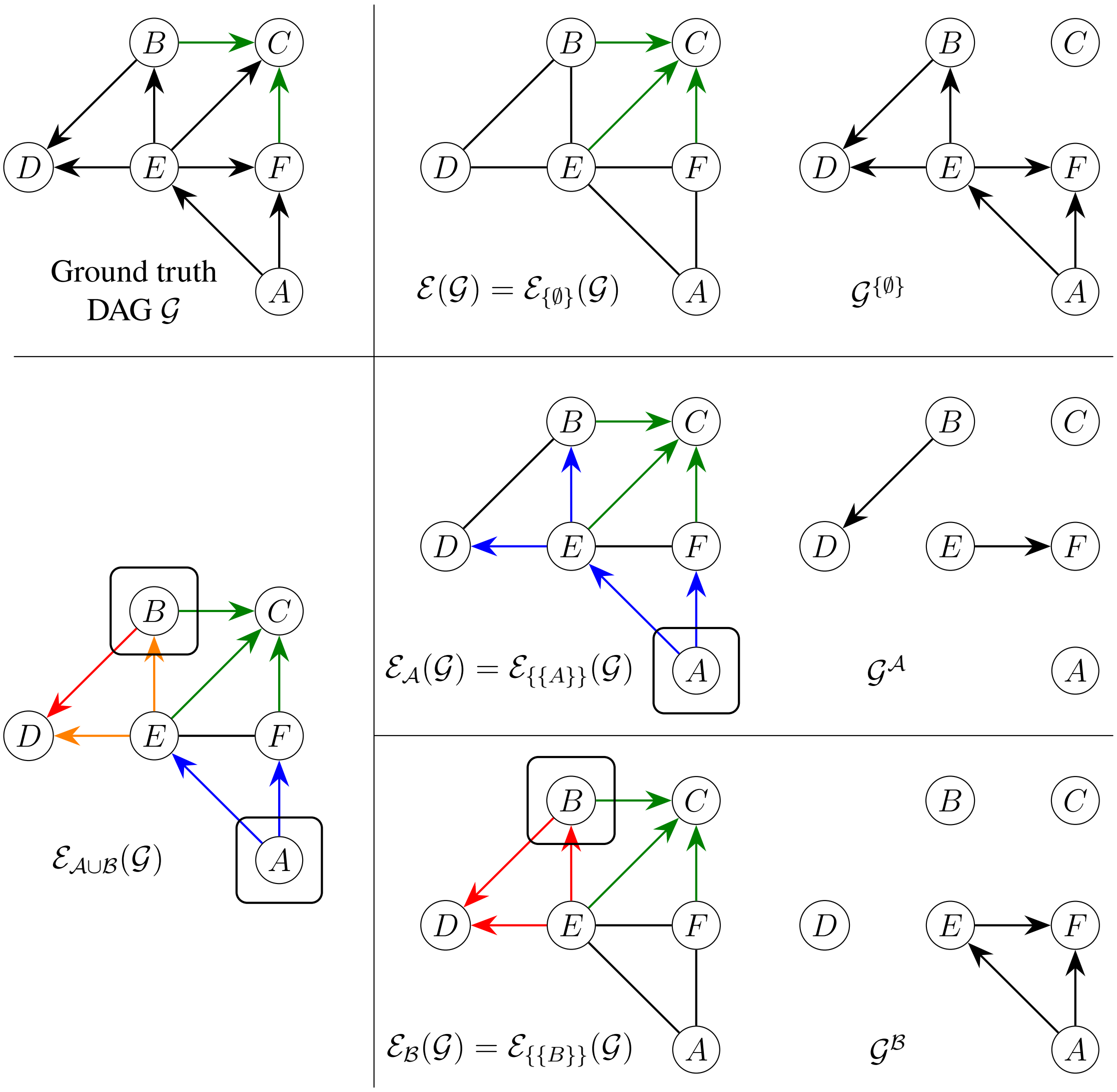

B
C
D
E
F
A
Ground truth DAG $\mathcal{G}$
$\mathcal{E}(\mathcal{G}) = \mathcal{E}_{\{\emptyset\}}(\mathcal{G})$
$\mathcal{G}^{\{\emptyset\}}$
$\mathcal{E}_{\mathcal{A}}(\mathcal{G}) = \mathcal{E}_{\{\{A\}\}}(\mathcal{G})$
$\mathcal{G}^{\mathcal{A}}$
$\mathcal{E}_{\mathcal{A}\cup\mathcal{B}}(\mathcal{G})$
$\mathcal{E}_{\mathcal{B}}(\mathcal{G}) = \mathcal{E}_{\{\{B\}\}}(\mathcal{G})$
$\mathcal{G}^{\mathcal{B}}$

Figure 6.5: Example for Theorem 6.14. Here, recovered edges $\boldsymbol{R}(\mathcal{G}, \cdot)$ are colored while the black edges are the hidden arc directions. Since $B \to C \leftarrow F$ is a v-structure in $\mathcal{G}$, these edges are oriented in the observational essential graph $\mathcal{E}(\mathcal{G})$ and so Meek R3 orients $E \to C$ in $\mathcal{E}(\mathcal{G})$. Intervening on $\mathcal{A} = \{\boldsymbol{A}\} = \{\{A\}\}$ orients the edges $\{A \to E, A \to F\}$ and Meek R1 further orients the edges $\{E \to B, E \to D\}$. Intervening on $\mathcal{B} = \{\boldsymbol{B}\} = \{\{B\}\}$ orients the edges $\{E \to B, B \to D\}$ and Meek R2 further orients the edge $\{E \to D\}$. Observe that $\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) = \{B \to D\}$, $\boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) = \{A \to E, A \to F\}$, and $\boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \emptyset) = \{E \to B, E \to D\}$. Finally, note that $E \to F$ is a covered edge so Theorem 6.7 tells us it will remain unoriented under interventions $\mathcal{A} \cup \mathcal{B}$.

While classic results [AMP97, HB12] tell us that chain components of interventional essential graphs are chordal, it is not immediately obvious why such edge-induced subgraphs cannot have v-structures in any of the DAGs compatible with $\mathcal{E}(\mathcal{G})$. The first statement of Theorem 6.14 formalizes this fact. Meanwhile, recall that from [GSKB18], we have $\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cup \boldsymbol{R}(\mathcal{G}, \mathcal{B})$ for any two interventions $\mathcal{A}$ and $\mathcal{B}$ (see Lemma 2.54). Informally, this means that combining prior *orientations* will not trigger Meek rules. On the other hand, Theorem 6.14 states that the *adjacencies* will also not, thus we can simplify the causal graphs by removing any oriented edges before performing further interventions.

6.3.1 Characterization of verifying sets

The characterization is proven in two directions separately. For necessity, we show that all four Meek rules (which are known to be consistent and complete) will *not* orient any unoriented covered edge of $\mathcal{G}$ that is *not* separated by any intervention. Our proof is simple due to the usage of covered edges. For sufficiency, we show that *every* unoriented non-covered edge of $\mathcal{G}$ will be oriented by Meek rules if all covered edges are separated. We prove this using a subtle induction over a valid topological ordering of the vertices π of $\mathcal{G}^*$: Let V_i be the first i smallest vertices in π, for $i = 1, 2, \ldots, n$. Consider subgraph $\mathcal{E}(\mathcal{G}^*)[V_i]$ induced by V_i with V_i being the last vertex in the ordering of V_i. By induction, it then suffices to show that all non-covered $U \to V_i$ edges are oriented for $U \in \boldsymbol{V}_{i-1}$.

6.3.2 Verification

For efficient computation of optimal verifying sets, we first prove several additional properties of covered edges, which may be of independent interest. For instance, the second property is used to easily identify covered edges in Fig. 6.2.

Lemma 6.15 (Properties of covered edges)**.**

1. *Let $\mathcal{H}$ be the edge-induced subgraph by covered edges of a DAG $\mathcal{G}$. Then, every vertex in $\mathcal{H}$ has at most one incoming edge and thus $\mathcal{H}$ is a forest of directed trees.*

2. *If a DAG $\mathcal{G}$ is a clique on $n \geq 3$ vertices $V_1, V_2, \ldots, V_n$ with $\pi(V_1) < \pi(V_2) < \ldots < \pi(V_n)$ with topological ordering π, then $V_1 \to V_2, \ldots, V_{n-1} \to V_n$ are the covered edges of $\mathcal{G}$.*

3. *If $U \to V$ is a covered edge in a DAG $\mathcal{G}$, then U* cannot *be a sink of any maximal clique of $\mathcal{G}$.*

The forest property enables us to utilize standard dynamic programming techniques to compute minimum vertex covers for the unoriented covered edges of $\mathcal{G}$ in an efficient manner. In contrast, it is known that minimum vertex covers are NP-complete to compute in general [Kar72].

6.3.3 Search

To obtain our results, we are *not* simply improving the analysis of [SMG+20]. Algorithmically, we developed a new approach that is based on graph separators [GRE84] which is a much simpler concept than directed clique trees. This ensures that our proposed algorithm terminates in $\mathcal{O}(\log n)$ iterations. To argue that each iteration uses at most $\mathcal{O}(\nu_1(\mathcal{G}))$ atomic interventions, we prove the following stronger universal lower bound that is built upon the lower bound of [SMG+20] stated in Lemma 6.16.

Lemma 6.16 (Lemma 6 of [SMG+20])**.** *Let $\mathcal{G}$ be a moral DAG. Then, $\nu_1(\mathcal{G}) \geq \lfloor \frac{\omega(\mathrm{skel}(\mathcal{G}))}{2} \rfloor$.*

Lemma 6.17. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ with an underlying ground truth DAG $\mathcal{G}^*$.*

$$\nu_1(\mathcal{G}^*) \geq \max_{\text{atomic intervention set } \mathcal{I} \subseteq 2^{\boldsymbol{V}}} \sum_{\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))} \left\lfloor \frac{\omega(\mathcal{H})}{2} \right\rfloor$$

Observe that the lower bound of Lemma 6.17 *not* computable because it involves a maximization over all possible atomic interventions *and* we do not know the interventional essential graphs $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)$. Nevertheless, it is a very powerful lower bound for analysis: In Fig. 6.6, we give an example where $\nu_1(\mathcal{G}^*) \approx n$ while the lower bound of [SMG+20] on $CC(\mathcal{E}(\mathcal{G}^*))$ is a constant. Meanwhile, there exists a set of atomic interventions $\mathcal{I}$ such that applying [SMG+20] on $CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))$ yields a much stronger $\Omega(n)$ bound.

6.3.4 Extension: Subset verification and search

For subset verification and search, we are interested in oriented the edges of a subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$. Despite a simple generalization, our earlier approaches fail to directly extend to this setting. We generalize the notions of minimum size verifying set (Definition 6.4) and verfication number (Definition 6.5) accordingly.

Definition 6.18 (Minimum size subset verifying set)**.** An intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ is called a verifying set for a DAG $\mathcal{G}^* = (\boldsymbol{V}, \boldsymbol{E})$ and subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$ if $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)[\boldsymbol{T}] = \mathcal{G}^*[\boldsymbol{T}]$ and is said to have minimum size if $\mathcal{E}_{\mathcal{I}'}(\mathcal{G}^*)[\boldsymbol{T}] \neq \mathcal{G}^*[\boldsymbol{T}]$ for any $\mathcal{I}' \subseteq 2^{\boldsymbol{V}}$ such that $|\mathcal{I}'| < |\mathcal{I}|$.

Definition 6.19 (Subset verification number)**.** Given $k \in \mathbb{N}^+$, the verification $\nu_k(\mathcal{G})$ of a DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ and subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$ is defined as $\nu_k(\mathcal{G}, \boldsymbol{T}) = |\mathcal{I}|$, where $\mathcal{I}$ is a minimum size verifying set for $\mathcal{G}$ and $\boldsymbol{T}$ such that any intervention in $\boldsymbol{I} \in \mathcal{I}$ involves at most $|\boldsymbol{I}| \leq k$ vertices, i.e. k-bounded interventions.

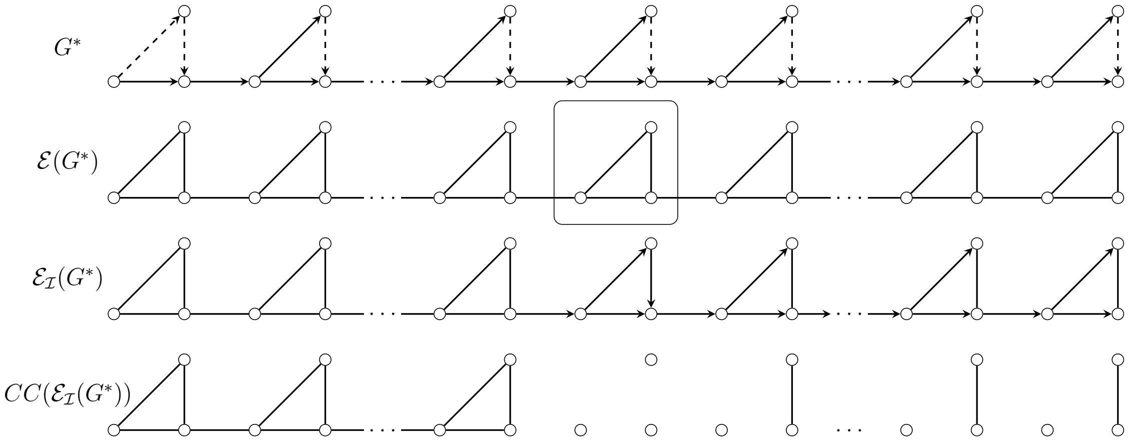

G^*
$\mathcal{E}(G^*)$
$\mathcal{E}_{\mathcal{I}}(G^*)$
$CC(\mathcal{E}_{\mathcal{I}}(G^*))$

Figure 6.6: A DAG $\mathcal{G}^*$ where minimum vertex cover of the unoriented covered edges (dashed arcs) is much larger than the size of the maximal clique (triangle): $\nu_1(\mathcal{G}^*) \approx n$ while the lower bound of [SMG+20] on $\mathcal{E}(\mathcal{G}^*)$ is a constant. Let $\mathcal{I}$ be an atomic intervention set on the middle triangle, i.e. the three vertices boxed up in $\mathcal{E}(\mathcal{G}^*)$. The partially directed graph $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)$ shows the learnt arc directions after intervening on $\mathcal{I}$ and applying Meek rules. Applying the lower bound of [SMG+20] on $CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))$ now gives a much stronger lower bound of $\approx n$ due to the single edge components.

Subset verification

For subset verification, we first establish interesting properties of the Hasse diagrams for moral DAGs which enables us to obtain structural properties regarding the arc directions that are recovered by an atomic intervention and then show that the atomic subset verification problem is equivalent to the problem of interval stabbing on a rooted tree, which we define later in Definition 6.47. The interval stabbing problem on a rooted tree can be viewed both as a special case of the set cover problem, and as a generalization of the interval stabbing problem on a line. The former is NP-hard [Kar72] while the latter can be solved using a polynomial time greedy algorithm (e.g. see [Eri19, Chapter 4, Exercise 4]). Our subset verification result follows from our polynomial time algorithm to solve the problem of interval stabbing on a rooted tree.

Theorem 6.20. *For any DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ and subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$, there exists a polynomial time algorithm to compute the minimum sized atomic subset verifying set.*

Subset search

Since we obtained a bound of $\mathcal{O}(\log n \cdot \nu_1(\mathcal{G}^*, \boldsymbol{E}))$ in Theorem 6.13, it may be natural to wonder if we can obtain a bound of $\mathcal{O}(\log n \cdot \nu_1(\mathcal{G}^*, \boldsymbol{T}))$ for any subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$. Unfortunately, this is not possible in general.

While a vertex cover of the target edges is a trivial upper bound for atomic subset search, we show that one needs to perform that many number of atomic interventions asymptotically in the worst case when facing an adaptive adversary which gets to see the interventions made by the adaptive algorithm and then gets to choose the ground truth DAG among the set of all DAGs that are consistent with the already revealed information.

Lemma 6.21. *Given a subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$, intervening on the vertices in a vertex cover of $\boldsymbol{T}$ atomically will fully orient all edges in $\boldsymbol{T}$.*

Lemma 6.22. *Fix any integer $n \geq 1$. There exists a fully unoriented essential graph on $2n$ vertices and a subset $\boldsymbol{T} \subseteq \boldsymbol{E}$ on n edges such that the size of the minimum vertex cover of $\boldsymbol{T}$ is $\mathrm{vc}(\boldsymbol{T})$ and any algorithm needs at least $\mathrm{vc}(\boldsymbol{T}) - 1$ number atomic interventions to orient all the edges in $\boldsymbol{T}$ against an adaptive adversary that reveals arc directions consistent with a DAG $\mathcal{G}^* \in [\mathcal{G}]$ with $\nu_1(\mathcal{G}^*, \boldsymbol{T}) = 1$.*

The above results tell us that we cannot hope for non-trivial subset search results in general for subset of target edges. On the other hand, if we restrict the class of target edges to be edges within a node-induced subgraph $\mathcal{H}$, then we can actually obtain the following non-trivial subset search result based on the definition of relevant nodes, which we define next. The special case of $\boldsymbol{T} = \boldsymbol{E}(\mathcal{H})$ is interesting because instances of local causal graph discovery often involve node-induced subgraphs.

Definition 6.23 (Relevant nodes)**.** Fix a DAG $\mathcal{G}^* = (\boldsymbol{V}, \boldsymbol{E})$ and arbitrary subset $\boldsymbol{V}' \subseteq \boldsymbol{V}$. For any intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ and resulting interventional essential graph $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)$, we define the *relevant nodes* $\rho(\mathcal{I}, \boldsymbol{V}') \subseteq \boldsymbol{V}'$ as the set of nodes within $\boldsymbol{V}'$ that is adjacent to some unoriented arc within the node-induced subgraph $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)[\boldsymbol{V}']$.

Theorem 6.24. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ of an unknown underlying DAG $\mathcal{G}^*$ and let $\mathcal{H}$ be an node-induced subgraph of $\mathcal{G}^*$. There exists an algorithm that runs in polynomial time and computes an atomic intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ in a deterministic and adaptive manner such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})] = \mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]$ and $|\mathcal{I}| \in \mathcal{O}(\log(|\rho(\mathcal{I}, \boldsymbol{V}(\mathcal{H}))|) \cdot \nu_1(\mathcal{G}^*, \boldsymbol{E}))$.*

Note that Theorem 6.24 compares against $\nu_1(\mathcal{G}^*, \boldsymbol{E})$ and not $\nu_1(\mathcal{G}^*, \boldsymbol{E}(\mathcal{H}))$. Since node-induced subgraphs of a chordal graph are also chordal, the chain components in $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})]$ are chordal. Our subset search algorithm generalizes the algorithm of Theorem 6.13, where we employ the *weighted* chordal graph separator guarantees from [GRE84]; see Lemma 2.44.

6.3.5 Extension: k-bounded interventions

We begin with a structural result relating $\nu_1(\mathcal{G})$ and $\nu_k(\mathcal{G})$ which our guarantees for k-bounded interventions heavily rely upon.

Theorem 6.25. *For any DAG $\mathcal{G}$, we have $\nu_k(\mathcal{G}) \geq \lceil \frac{\nu_1(\mathcal{G})}{k} \rceil$.*

For bounded size verifying sets, we exploit the fact that trees are bipartite and so we can divide the minimum vertex covers into two partitions. Since vertices within each partite are non-adjacent, we can group them into larger interventions without affecting the overall number of separated edges, giving us the desired guarantees.

Theorem 6.26. *If $\nu_1(\mathcal{G}) = \ell$, then $\nu_k(\mathcal{G}) \geq \lceil \ell/k \rceil$ and there exists a polynomial time algorithm to compute a bounded size intervention set $\mathcal{I}$ of size $|\mathcal{I}| \leq \lceil \frac{\ell}{k} \rceil + 1$.*

To the best of our knowledge, our work provides the first known efficient algorithm for computing near-optimal bounded sized verifying sets for general graphs.

A similar proof strategy also works for subset verification but we will need a slightly different argument for why there is a tree with respect to the target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$ of interest. We defer the details for the subset case to Appendix B.1.4.

The (subset) search algorithms in both Theorem 6.13 and Theorem 6.24 can be also be generalized to perform bounded size interventions on the computed clique separators to yield a multiplicative optimality gap of $\mathcal{O}(\log(n) \cdot \log(k))$ with respect to $\nu_k(\mathcal{G}^*)$. That is, we pay an additional factor of $\log(k)$ when competing against $\nu_k(\mathcal{G}^*)$. Again, this is the first competitive result of its kind that holds on *general graphs*. We achieve this via black-box applications of the labelling scheme due to [SKDV15]; see Algorithm 12.

Lemma 6.27 (Lemma 1 of [SKDV15])**.** *Let (n, k, a) be parameters where $k \leq n/2$. There is a polynomial time labeling scheme that produces distinct ℓ length labels for all elements in $[n]$ using letters from the integer alphabet $\{0\} \cup [a]$ where $\ell = \lceil \log_a n \rceil$. Further, in every digit (or position), any integer letter is used at most $\lceil n/a \rceil$ times. This labelling scheme is a separating system: for any $i, j \in [n]$, there exists some digit $d \in [\ell]$ where the labels of i and j differ.*

Algorithm 12 Labeling scheme subroutine for producing non-atomic interventions.

```
Input: Set of vertices $\boldsymbol{A}$, size upper bound $k \geq 1$.
Output: A $k$-separating system $\boldsymbol{B} \subseteq 2^{\boldsymbol{A}}$.
if $k = 1$ then
    Set $\boldsymbol{B} = \boldsymbol{A}$.
else
    Define $k' = \min\{k, |\boldsymbol{A}|/2\}$, $a = \lceil |\boldsymbol{A}|/k' \rceil \geq 2$, and $\ell = \lceil \log_a |\boldsymbol{A}| \rceil$.
    Compute labelling scheme of [SKDV15, Lemma 1] on $\boldsymbol{A}$ with $(|\boldsymbol{A}|, k', a)$.
    Set $\boldsymbol{B} = \{S_{x,y}\}_{x \in [\ell], y \in [a]}$, where $\mathcal{S}_{x,y} \subseteq \boldsymbol{A}$ is the subset of vertices whose $x^{th}$
    letter in the label is $y$.
return $\boldsymbol{B}$
```

Theorem 6.28. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ with an unknown underlying ground truth DAG $\mathcal{G}^*$. For any integer $k > 1$, there is an algorithm that runs in polynomial time and computes a k-bounded intervention set $\mathcal{I} \subseteq 2^{\mathbf{V}}$ in a deterministic and adaptive manner such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*) = \mathcal{G}^*$ and $|\mathcal{I}| \in \mathcal{O}(\log(n) \cdot \log(k) \cdot \nu_k(\mathcal{G}^*))$.*

Theorem 6.29. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ of an unknown underlying DAG $\mathcal{G}^*$ and let $\mathcal{H}$ be an node-induced subgraph of $\mathcal{G}^*$. For any integer $k > 1$, there is an algorithm that runs in polynomial time and computes a k-bounded intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ in a deterministic and adaptive manner such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})] = \mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]$ and $|\mathcal{I}| \in \mathcal{O}(\log(|\rho(\mathcal{I}, \boldsymbol{V}(\mathcal{H}))|) \cdot \log(k) \cdot \nu_k(\mathcal{G}^*, \boldsymbol{E}))$.*

Theorem 6.25 enables us to easily relate $\nu_1(G)$ with $\nu_k(G)$ while our approach for k-bounded intervention guarantees have an additional multiplicative $\log k$ factor compared to their atomic counterparts due to applications of Lemma 6.27 to efficiently compute k-bounded intervention sets of a given set of nodes.

6.4 Properties about interventional essential graph

In this section, we prove Theorem 6.14 which provide some structural properties of interventional essential graphs $\mathcal{E}_{\mathcal{I}}(\mathcal{G})$; note that the observational essential graph $\mathcal{E}(\mathcal{G}) = \mathcal{E}_{\emptyset}(\mathcal{G})$ is a special case. These properties enable us to ignore v-structures and justify the study of the (subset) verification and search problems solely on moral DAGs without v-structures.

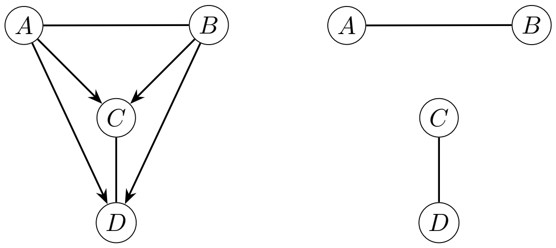
We prove each statement in Theorem 6.14 in separate lemmas for ease of reading. Our proofs are greatly simplified by Lemma 6.30, an observation that triangles in interventional essential graphs *cannot* have exactly one oriented arc, whose proof relies on Lemma 2.52. A similar argument to Lemma 6.30 was made in [GSKB18, Appendix B, Figure 4, Structure S_0] for their case analysis proof of Lemma 2.54.

Lemma 6.30 (Triangle lemma)**.** *Fix a DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ and an intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$. For any triangle induced by vertices $U, V, W \in \boldsymbol{V}$ with edges $U - V, V - W, U - W \in E$, it cannot be the case that exactly one of $\{U - V, V - W, U - W\}$ is oriented in $\boldsymbol{R}(\mathcal{G}, \mathcal{I})$.*

Proof. We know by Lemma 2.52 that $\mathcal{E}_{\mathcal{I}}(\mathcal{G})$ is a chain graph, so it is does not contain directed cycles. Suppose, for a contradiction, that there is a triangle induced by the vertices $\{U, V, W\}$ and exactly one of $\{U - V, V - W, U - W\}$ is oriented in $\boldsymbol{R}(\mathcal{G}, \mathcal{I})$. W.l.o.g., by relabeling, suppose $U \to V \in \boldsymbol{R}(\mathcal{G}, \mathcal{I})$. Then, $U \to V - W - U$ is a directed cycle in $\mathcal{E}_{\mathcal{I}}(\mathcal{G})$, contradicting the fact that $\mathcal{E}_{\mathcal{I}}(\mathcal{G})$ is a chain graph. □

Note that there exists partially oriented chain graphs that are *not* interventional essential graphs where every triangle does not have exactly one oriented arc, and the edge-induced subgraph on the unoriented edges do not form v-structures for any acyclic completion. Fig. 6.7 provides such an example.

Lemma 6.31. *Consider the setting of Theorem 6.14. Any v-structures in $\mathcal{G}^{\mathcal{A}}$ are also present in $\mathcal{G}$.*



$\mathcal{G}$ $\quad$ $\mathcal{G}[\boldsymbol{E}(\mathcal{G}) \setminus \boldsymbol{A}(\mathcal{G})]$

Figure 6.7: In the partially oriented chain graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, all triangles have exactly two oriented arcs. Since $A - B$ and $C - D$ could be independently oriented in either directions, there are four possible acyclic completions of $\mathcal{G}$. The edge-induced subgraph of $\mathcal{G}$ on the unoriented edges does not have any v-structures for any of these possible acyclic completions. However, $\mathcal{G}$ *cannot* be an interventional essential graph as there are no v-structures and every vertex is incident to some unoriented edge.

Proof. To be false, there must exist a triangle in $\mathcal{G}$ on 3 vertices U, V, W such that $U \to V \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ and $U \to W, V \to W \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$. This is impossible by Lemma 6.30. □

Lemma 6.32. *Consider the setting of Theorem 6.14. Any acyclic completion of $\mathcal{E}(\mathcal{G}^{\mathcal{A}})$ that does not form new v-structures can be combined with $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$ to obtain a valid DAG belonging to both $\mathcal{E}(\mathcal{G})$ and $\mathcal{E}_{\mathcal{A}}(\mathcal{G})$.*

Proof. Fix an acyclic completion $\mathcal{G}'$ of $\mathcal{E}(\mathcal{G}^{\mathcal{A}})$. Suppose, for a contradiction, that there is a cycle in $\boldsymbol{E}(\mathcal{G}') \cup \boldsymbol{R}(\mathcal{G}, \mathcal{A})$. Let $\mathcal{C} = V_0 \to V_1 \to \ldots \to V_k \to V_0$ be the *smallest* such cycle. Since $\mathcal{G}'$ is an acyclic completion, we know that at least one arc of $\mathcal{C}$ was from $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$. W.l.o.g., suppose that $V_0 \to V_1 \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$. Since $\mathcal{G}^*$ is acyclic to begin with, we also know that at least one arc of $\mathcal{C}$ is *not* from $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$.

If $k = 2$, then $\mathcal{C} = V_0 \to V_1 \to V_2 \to V_0$. From above, we know $V_0 \to V_1 \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ and at least one arc of $\mathcal{C}$ is *not* in $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$. Meanwhile, Lemma 6.30 implies that either $V_1 \to V_2 \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ or $V_2 \to V_0 \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$. In either case, we get a contradiction:

- If $V_0 \to V_1, V_1 \to V_2 \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ and $V_2 \to V_0 \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$, then Meek R2 will orient $V_0 \to V_2$ via $V_0 \to V_1 \to V_2 - V_0$. So, $V_0 - V_2 \not\in \boldsymbol{E}[\mathcal{E}(\mathcal{G}^{\mathcal{A}})]$ but $V_2 \to V_0 \in \boldsymbol{E}(\mathcal{G}')$.
- If $V_2 \to V_0, V_0 \to V_1 \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ and $V_1 \to V_2 \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$, then Meek R2 will orient $V_2 \to V_1$ via $V_2 \to V_0 \to V_1 - V_2$. So, $V_1 - V_2 \not\in \boldsymbol{E}[\mathcal{E}(\mathcal{G}^{\mathcal{A}})]$ but $V_1 \to V_2 \in \boldsymbol{E}(\mathcal{G}')$.

Now, consider the case where $k > 2$. From above, we know $V_0 \to V_1 \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ and at least one arc of $\mathcal{C}$ is *not* in $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$. Let $V_i \to V_j \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ be the arc of $\mathcal{C}$ with the smallest source index $i \geq 1$, where we write $j = (i+1) \mod k$ for notational convenience. By minimality of i, we know that $V_{i-1} \to V_i \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$. Now, since $V_i \to V_j \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$, it must be the case that the arc $V_{i-1} - V_j$ exists in $\mathcal{G}$, otherwise Meek R1 will orient $V_i \to V_j$ via $V_{i-1} \to V_i - V_j$. By Lemma 6.30 and the assumptions that $V_{i-1} \to V_i \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ and $V_i \to V_j \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$, it must be the case that $V_{i-1} - V_j$ is oriented in $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$. In either of the two cases below, we get a contradiction.

- If $V_{i-1} \to V_j \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$, then $V_0 \to \ldots \to V_{i-1} \to V_j \to \ldots \to V_k \to V_0$ is a smaller cycle than $\mathcal{C}$ in $\mathcal{G}' \cup \boldsymbol{R}(\mathcal{G}, \mathcal{A})$.

- If $V_j \to V_{i-1} \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$, then Meek R2 orients $V_j \to V_i$ via $V_j \to V_{i-1} \to V_i - V_j$. So, $V_i - V_j \not\in \boldsymbol{E}[\mathcal{E}(\mathcal{G}^{\mathcal{A}})]$ but $V_i \to V_j \in \boldsymbol{E}(\mathcal{G}')$.

Therefore, the claim follows since there are no cycles in $\boldsymbol{E}(\mathcal{G}') \cup \boldsymbol{R}(\mathcal{G}, \mathcal{A})$. □

Lemma 6.33. *Consider the setting of Theorem 6.14. We have*

$$\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) = \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{A})$$

Proof. We show containment in both directions.

Direction 1: $\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \subseteq \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{A})$

Suppose, for a contradiction, that there exists an arc $A \to B \in \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$ but $A \to B \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{A})$. Note that $A \to B \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ otherwise $A \to B \not\in \boldsymbol{E}(\mathcal{G}^{\mathcal{A}})$ and thus $A \to B \not\in \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$. So, to show a contradiction, it suffices to argue that $A \to B \in \boldsymbol{R}(\mathcal{G}, \mathcal{B})$.

There are two possible situation explaining $A \to B \in \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$: either (i) there is some intervention $\boldsymbol{I} \in \mathcal{B}$ such that $|\boldsymbol{I} \cap \{A, B\}| = 1$, or (ii) Meek rules oriented $A - B$.

(i) In the first situation where there is some intervention $\boldsymbol{I} \in \mathcal{B}$ such that $|\boldsymbol{I} \cap \{A, B\}| = 1$, then the edge $A - B$ is separated by $\boldsymbol{I}$ and so $A \to B \in \boldsymbol{R}(\mathcal{G}, \mathcal{B})$ as well. Contradiction.

(ii) In the second situation, let us consider the sequence of Meek rule configurations that oriented $A \to B$ in $\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$. By definition of $\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$, all the edges (oriented or not) involved in these configurations do *not* belong to $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$. If these configurations also appear in $\boldsymbol{R}(\mathcal{G}, \mathcal{B})$, then $A \to B \in \boldsymbol{R}(\mathcal{G}, \mathcal{B})$ as well. The only reason why any of these configurations may not appear in $\boldsymbol{R}(\mathcal{G}, \mathcal{B})$ is because there was some other edge in the node-induced subgraph that was removed due to being in $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$. So, it suffices to consider missing edges within the node-induced subgraph of each Meek rule configuration:

- Suppose the R1 configuration involving three vertices $U \to V - W$ and $U \not\sim W$ was one of the configurations used by $\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$ to orient $A \to B$, but this configuration *did not* appear for $\boldsymbol{R}(\mathcal{G}, \mathcal{B})$. Then, it was because $U - W$ appears in $\mathcal{G}$ and was removed from $\mathcal{G}^{\mathcal{A}}$ due to it being oriented in $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$. However, this contradicts Lemma 6.30 since $U \to V \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ and $\{U - W, V - W\}$ are not in $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$.

- All possible edges are present in the node-induced subgraph of the R2 configuration.

- There is only one possible edge removed by $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$ in configurations R3 and R4. By the same argument to the R1 configuration above, one can check that this implies that there is some triangle on three vertices contradicting Lemma 6.30.

In other words, $A \to B \in \boldsymbol{R}(\mathcal{G}, \mathcal{B})$ whenever $A \to B \in \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$ due to Meek rules.

Direction 2: $\boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{A}) \subseteq \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$

For any arc $A \to B \in \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{A})$, we have that $A \to B \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$ and so the edge $A - B$ appears unoriented in $\mathcal{E}(\mathcal{G}^{\mathcal{A}})$, which implies that $A - B$ appears in $\mathcal{E}(\mathcal{G})$ as an unoriented edge. So, we may ignore v-structure arcs in $\boldsymbol{R}(\mathcal{G}, \mathcal{B})$. There are two possible situation explaining why an arc $A \to B$ belongs in $\boldsymbol{R}(\mathcal{G}, \mathcal{B})$: either (i) there is some intervention $I \in \mathcal{B}$ such that $|\boldsymbol{I} \cap \{A, B\}| = 1$, or (ii) Meek rules oriented $A - B$.

(i) In the first situation, we have $A \to B \in \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$ as well.

(ii) We prove the second situation by contradiction. For notation simplicity, let us write $\boldsymbol{S} = (R(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{A})) \setminus \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) = \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus (R(\mathcal{G}, \mathcal{A}) \cup \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}))$. Suppose, for a contradiction, that $\boldsymbol{S} \neq \emptyset$. Let $A \to B \in \boldsymbol{S}$ be oriented via a sequence of Meek rule configurations such that only the last configuration does not appear in $\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$. By calling such a Meek rule configuration a *bad* configuration, we can see why such an arc $A \to B$ exists: for any arc in $\boldsymbol{S}$ that uses more than one bad configuration, one of the oriented arcs in the bad configuration is an arc in $\boldsymbol{S}$ that is oriented with strictly fewer bad orientations. Now, consider the last Meek rule configuration used to orient $A \to B$ in $\boldsymbol{R}(\mathcal{G}, \mathcal{B})$. We make two observations:

O1 If *none* of the oriented arcs of this Meek rule configuration belongs to $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$, then these arcs appear in $\mathcal{G}^{\mathcal{A}}$ and will be oriented due to $\mathcal{B}$, thus $A \to B \in \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$. This contradicts to $A \to B \in \boldsymbol{S}$.

O2 If *all* of the oriented arcs of this Meek rule configuration belong to $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$, then $A \to B \in \boldsymbol{R}(\mathcal{G}, \mathcal{A})$. This contradicts to $A \to B \in \boldsymbol{S}$.

There is only one arc in the R1 configuration, so either O1 or O2 applies. Meanwhile, the arcs in the R3 configuration form a v-structure and so *both* of them belong to $\boldsymbol{S}$, so O2 applies. In R2 or R4 configurations, there are two arcs. If none or both arcs are in $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$, then we can apply O1 or O2 to reach a contradiction. If exactly one of the arcs are in $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$, then there will be a triangle on three vertices contradicting Lemma 6.30. □

Lemma 6.34. *Consider the setting of Theorem 6.14. We have*

$$\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}, \mathcal{A})$$

Proof. By Lemma 2.54, we have that $\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cup \boldsymbol{R}(\mathcal{G}, \mathcal{B})$. The claim follows using Lemma 6.33. □

Lemma 6.35. *Consider the setting of Theorem 6.14. We have*

$$\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) \sqcup (R(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B}))$$

Proof. The disjointness follows from definitions of $\mathcal{G}^{\mathcal{A}}$ and $\mathcal{G}^{\mathcal{B}}$. We now argue containment in both directions.

Direction 1: $\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) \subseteq \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) \sqcup (R(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B}))$

By Lemma 2.54, we know that $\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cup \boldsymbol{R}(\mathcal{G}, \mathcal{B})$. Consider an arbitrary arc $e \in \boldsymbol{E}$ such that $e \in \boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cup \boldsymbol{R}(\mathcal{G}, \mathcal{B})$. Suppose $e \not\in \boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B})$. If $e \in \boldsymbol{R}(\mathcal{G}, \mathcal{A}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{B})$, then e appears in $\mathcal{G}^{\mathcal{B}}$ and so $e \in \boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A})$. If $e \in \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{A})$, then e appears in $\mathcal{G}^{\mathcal{A}}$ and so $e \in \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$. In either case, we see that $e \in \boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) \cup \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) = \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) \subseteq \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) \sqcup (R(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B}))$.

Direction 2: $\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) \sqcup (R(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B})) \subseteq \boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B})$

We argue that each of $\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B})$, $\boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A})$, and $\boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B})$ is a subset of $\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B})$. By Lemma 6.34, $\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \subseteq \boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B})$ and $\boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) \subseteq \boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B})$. By Lemma 2.54, we know that $\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cup \boldsymbol{R}(\mathcal{G}, \mathcal{B})$ and so $\boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \subseteq \boldsymbol{R}(\mathcal{G}, \mathcal{A}) \cup \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \subseteq \boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B})$. □

Lemma 6.36. *Consider the setting of Theorem 6.14. $\boldsymbol{R}(\mathcal{G}, \emptyset)$ does not contain any covered edge of $\mathcal{G}$.*

Proof. By definition, covered edges are not v-structure edges. By Theorem 6.7, covered edges will not be oriented by Meek rules and we need to intervene on either of the endpoints to orient it. Therefore, $\boldsymbol{R}(\mathcal{G}, \emptyset)$ does not contain any covered edges. □

Theorem 6.14 follows from the combination of the above lemmas.

Theorem 6.14 (Properties of interventional essential graphs)**.** *Fix a DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$. For any intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ and any vertex $U \in \boldsymbol{V}$, let $\boldsymbol{R}(\mathcal{G}, \mathcal{I}) \subseteq \boldsymbol{E}$ denote the set of oriented arcs in the $\mathcal{I}$-essential graph of $\mathcal{G}$, $\mathcal{G}^{\mathcal{I}}$ be the fully directed subgraph DAG of $\mathcal{G}$ obtained by arcs in $\boldsymbol{R}(\mathcal{G}, \mathcal{I})$, and $\mathrm{Pa}_{\mathcal{G},\mathcal{I}}(U) = \{X \in V : X \to U \in \boldsymbol{R}(\mathcal{G}, \mathcal{I})\}$ be the parents of U recovered by $\mathcal{I}$. The following statements are true with respect to any two arbitrary intervention sets $\mathcal{A} \subseteq 2^{\boldsymbol{V}}$ and $\mathcal{B} \subseteq 2^{\boldsymbol{V}}$:*

1. *Any v-structures in $\mathcal{G}^{\mathcal{A}}$ are also present in $\mathcal{G}$.*
2. *Any acyclic completion of $\mathcal{E}(\mathcal{G}^{\mathcal{A}})$ that does not form new v-structures can be combined with $\boldsymbol{R}(\mathcal{G}, \mathcal{A})$ to obtain a valid DAG belonging to both $\mathcal{E}(\mathcal{G})$ and $\mathcal{E}_{\mathcal{A}}(\mathcal{G})$.*
3. *$\boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) = \boldsymbol{R}(\mathcal{G}, \mathcal{B}) \setminus \boldsymbol{R}(\mathcal{G}, \mathcal{A})$.*
4. *$\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}, \mathcal{A})$.*
5. *$\boldsymbol{R}(\mathcal{G}, \mathcal{A} \cup \mathcal{B}) = \boldsymbol{R}(\mathcal{G}^{\mathcal{A}}, \mathcal{B}) \sqcup \boldsymbol{R}(\mathcal{G}^{\mathcal{B}}, \mathcal{A}) \sqcup (R(\mathcal{G}, \mathcal{A}) \cap \boldsymbol{R}(\mathcal{G}, \mathcal{B}))$.*
6. *$\boldsymbol{R}(\mathcal{G}, \emptyset)$ does not contain any covered edge of $\mathcal{G}$.*

Proof. Combine Lemma 6.31, Lemma 6.32, Lemma 6.33, Lemma 6.34, Lemma 6.35, and Lemma 6.36. □

6.5 Characterization of verifying sets

In this section, we formally prove Theorem 6.7 by proving both directions separately.

Lemma 6.37 (Necessary)**.** *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ and $\mathcal{G} \in [\mathcal{G}^*]$. If $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ is a verifying set, then $\mathcal{I}$ separates all unoriented covered edge $U - V$ of $\mathcal{G}$.*

Proof. Let $U \to V$ be an arbitrary unoriented covered edge in $\mathcal{E}(\mathcal{G}^*)$ and $\mathcal{I}$ be an intervention set where U and V are *never* separated by any $\boldsymbol{S} \in \mathcal{I}$. Then, interventions will not orient $U \to V$ and we can only possibly orient it via Meek rules. We check that all four Meek rules will *not* orient $U \to V$:

(R1) For R1 to trigger, we need to have $W \to U \to V$ and $W \not\sim V$ for some vertex $W \in \boldsymbol{V} \setminus \{U, V\}$. However, such a vertex W will imply that $U \to V$ is *not* a covered edge.

(R2) For R2 to trigger, we need to have $U \to W \to V$ for some $W \in \boldsymbol{V} \setminus \{U, V\}$. However, such a vertex W will imply that $U \to V$ is *not* a covered edge.

(R3) For R3 to trigger, we must have $W - U - X$, $W \to V \leftarrow X$, and $W \not\sim X$ for some $W, X \in \boldsymbol{V} \setminus \{U, V\}$. Since $U \to V$ is a covered edge, we must have $W \to U \leftarrow X$. This implies that $W \to U \leftarrow X$ appears as a v-structure in $\mathcal{E}(\mathcal{G}^*)$ and thus R3 will not trigger due to interventions from $\mathcal{I}$.

(R4) For R4 to trigger, we must have $W - U - X$, $W \to X \to V$, and $W \not\sim V$ for some $W, X \in \boldsymbol{V} \setminus \{U, V\}$. Since $U \to V$ is covered, we must have $X \to U$. To avoid directed cycles, it must be the case that $W \to U$. However, this implies that $U \to V$ is *not* covered since $W \to U$ while $W \not\sim V$.

Therefore, $\mathcal{I}$ *cannot* be a verifying set if U and V are *never* separated by any $S \in \mathcal{I}$. □

Lemma 6.38 (Sufficient)**.** *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ and $\mathcal{G} \in [\mathcal{G}^*]$. If $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ is an intervention set that separates every unoriented covered edge $U - V$ of $\mathcal{G}$, then $\mathcal{I}$ is a verifying set.*

Proof. Let $\mathcal{I}$ be an arbitrary intervention set such that every unoriented covered edge $U - V$ of $\mathcal{G}$ has an set $\boldsymbol{S} \in \mathcal{I}$ that separates U and V. Fix an arbitrary valid vertex permutation $\pi : \boldsymbol{V} \to [n]$ of $\mathcal{G}$. For any $i \in [n]$, define $\boldsymbol{V}_i = \{\pi^{-1}(1), \ldots, \pi^{-1}(i)\} \subseteq \boldsymbol{V}$ as the i smallest vertices according to π's ordering. We argue that any unoriented edges in $\mathcal{E}(\mathcal{G}^*)[\boldsymbol{V}_i]$ will be oriented by $\mathcal{I}$ by performing induction on i.

Base case ($i = 1$): There are no edges in $\mathcal{G}[\boldsymbol{V}_1]$ so $\mathcal{E}(\mathcal{G}^*)[\boldsymbol{V}_1]$ is trivially fully oriented.

Inductive case ($i > 1$): Suppose $V = \pi^{-1}(i)$. By induction hypothesis, $\mathcal{E}(\mathcal{G}^*)[\boldsymbol{V}_{i-1}]$ is fully oriented so any unoriented edge in $\mathcal{E}(\mathcal{G}^*)[\boldsymbol{V}_i]$ must have the form $U \to V$, where $\pi(U) < \pi(V)$. For any $U \to V$ is an unoriented covered edge in $\mathcal{E}(\mathcal{G}^*)[\boldsymbol{V}_i]$, there will be an intervention $\boldsymbol{S} \in \mathcal{I}$ that separates U and V, and hence covered edges will all be oriented. Now suppose, for a contradiction, that there exists unoriented edges in $\mathcal{E}(\mathcal{G}^*)[\boldsymbol{V}_i]$ that are *not* covered edges. Let $U \to V$ be the unoriented edge where $\pi(U)$ is *maximized*. Since $U \to V$ is not a covered edge, one of the two cases must occur:

Case 1 $\exists W \in \boldsymbol{V} \setminus \{U, V\}$ such that $W \to U$ and $W \not\to V$ in $\mathcal{G}^*$

Since $W \to U$ and $U \to V$ in $\mathcal{G}^*$, we see that $\pi(V) > \pi(W)$. So, $W \not\to V$ implies $W \not\sim V$ in $\mathcal{G}^*$. By induction, $W \to U$ will be oriented. So, Meek R1 will orient $U \to V$.

Case 2 $\exists W \in \boldsymbol{V} \setminus \{U, V\}$ such that $W \to V$ and $W \not\to U$ in $\mathcal{G}^*$

If $W \not\sim U$ in $\mathcal{G}^*$, then $U \to V \leftarrow W$ is a v-structure in $\mathcal{G}^*$ and $U \to V$ would have been oriented. If $W - U$ in $\mathcal{G}^*$, then we must have $U \to W$ and $\pi(U) < \pi(W)$. By induction, $U \to W$ will be oriented. Since $\pi(U) < \pi(W)$ and $\pi(U)$ is maximized out of all possible unoriented edges in $\mathcal{E}(\mathcal{G}^*)[\boldsymbol{V}_i]$ involving V, $W \to V$ must be an oriented edge and will be oriented by $\mathcal{I}$. So, Meek R2 will orient $U \to V$.

In either case, $U \to V$ will be oriented. Contradiction. □

Combining Lemma 6.37 and Lemma 6.38 yields our characterization of verifying sets.

Theorem 6.7. *Fix a DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$. An intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ is a minimum sized verifying set for $\mathcal{G}$ if and only if every covered edge of $\mathcal{G}$ is separated by some intervention in $\mathcal{I}$.*

Proof. Combine Lemma 6.37 and Lemma 6.38. □

As mentioned in Section 6.3, we can use Theorem 6.7 to prove Corollary 6.11

Corollary 6.11. *Given an essential graph $\mathcal{E}(\mathcal{G}^*)$ of an unknown ground truth DAG $\mathcal{G}^*$ and a causal DAG $\mathcal{G} \in [\mathcal{G}^*]$, we can test if $\mathcal{G} \overset{?}{=} \mathcal{G}^*$ by intervening on any verifying set of $\mathcal{G}$. Furthermore, in the worst case,* any *algorithm that correctly resolves $\mathcal{G} \overset{?}{=} \mathcal{G}^*$ using k-bounded interventions needs at least $\nu_k(\mathcal{G})$ interventions.*

Proof. Using Theorem 6.7, we know that the minimal verifying set for $\mathcal{G}$ is the smallest possible set of interventions $\mathcal{I}$ such that *all* covered edges of $\mathcal{G}$ is separated by some intervention $\boldsymbol{S} \in \mathcal{I}$. If the graph is fully oriented after intervening on all $\boldsymbol{S} \in \mathcal{I}$, then it must be the case that $\mathcal{G} = \mathcal{G}^*$. Otherwise, we will either detect that some edge orientation disagrees with $\mathcal{G}$ or there remains some unoriented edge at the end of all our interventions. In the first case, we trivially conclude that $\mathcal{G} \neq \mathcal{G}^*$. In the second case, Theorem 6.7 tells us that any such unoriented edge must be an unoriented covered edge of $\mathcal{G}^*$ (but $\mathcal{I}$ separated all covered edges of $\mathcal{G}$) and so we can also conclude that $\mathcal{G} \neq \mathcal{G}^*$.

Suppose, for a contradiction, that some algorithm managed to use strictly less than $\nu_k(\mathcal{G})$ interventions to verify a graph $\mathcal{G}$. Then, there exists at least one covered edge

$U \to V$ in $\mathcal{G}$ that is not separated by the interventions used. Define $\mathcal{G}'_1 = \mathcal{G}$ and $\mathcal{G}'_2$ as $\mathcal{G}$ with this covered edge reversed (i.e. $V \to U$ instead). Note that $\mathcal{G}'_2$ is also a DAG in the same MEC due to Lemma 2.49. We see that this algorithm *cannot* distinguish between $\mathcal{G}'_1$ and $\mathcal{G}'_2$ and thus cannot correctly output $\mathcal{G} = \mathcal{G}'$ or $\mathcal{G} \neq \mathcal{G}'$ respectively. This is a contradiction, i.e. at least $\nu_k(\mathcal{G})$ interventions are needed in the worst case. □

6.6 Verification using atomic interventions

In this section, we show how to obtain an efficient computation of minimum atomic verifying set. We begin by showing some properties of covered edges.

Lemma 6.15 (Properties of covered edges)**.**

1. *Let $\mathcal{H}$ be the edge-induced subgraph by covered edges of a DAG $\mathcal{G}$. Then, every vertex in $\mathcal{H}$ has at most one incoming edge and thus $\mathcal{H}$ is a forest of directed trees.*

2. *If a DAG $\mathcal{G}$ is a clique on $n \geq 3$ vertices $V_1, V_2, \ldots, V_n$ with $\pi(V_1) < \pi(V_2) < \ldots < \pi(V_n)$ with topological ordering π, then $V_1 \to V_2, \ldots, V_{n-1} \to V_n$ are the covered edges of $\mathcal{G}$.*

3. *If $U \to V$ is a covered edge in a DAG $\mathcal{G}$, then U* cannot *be a sink of any maximal clique of $\mathcal{G}$.*

Proof.

1. Suppose, for a contradiction, that there exists some vertex W with two incoming covered edges $U \to W \leftarrow V$. For $U \to W$ to be covered, we must have $V \to U$. Similarly, for $V \to W$ to be covered, we must have $U \to V$. However, we cannot simultaneously have both $U \to V$ and $V \to U$, as it would lead to a contradiction as $\mathcal{G}$ is a DAG. Furthermore, since $\mathcal{G}$ itself is acyclic, it implies that the edge-induced subgraph $\mathcal{H}$ must also be acyclic. Therefore, $\mathcal{H}$ is a forest of directed trees.

2. Note that π is the only valid topological ordering since $\mathcal{G}$ is a complete graph. Let $\boldsymbol{A} = \{V_1 \to V_2, V_2 \to V_3, \ldots, V_{n-1} \to V_n\}$ be the set of arcs of interest. For any arc $V_i \to V_{i+1} \in \boldsymbol{A}$, one can check that they share the same parents by the topological ordering π. Consider an arbitrary arc $V_i \to V_j \not\in \boldsymbol{A}$. Since $V_i \to V_j \not\in \boldsymbol{A}$, there exists $V_k \in \boldsymbol{V}$ such that $\pi(V_i) < \pi(V_k) < \pi(V_j)$. Then, since $\mathcal{G}$ is a clique, we must have $V_i \to V_k \to V_j$ and so $V_i \to V_j$ *cannot* be covered since $V_k \in \mathrm{Pa}(V_j) \setminus \{V_i\}$ but $V_k \not\in \mathrm{Pa}(V_i) \setminus \{V_j\}$.

3. Suppose, for a contradiction, that U is a sink of some maximal clique $\mathcal{K}$ of size h and $U \to V$ is a covered edge. Then, we must have $\mathrm{Pa}(V) \setminus \{U\} = \mathrm{Pa}(U)$. However, that means that $\boldsymbol{V}(\mathcal{K}) \cup \{V\}$ forms a clique of size $h+1$. This contradicts the assumption that $\mathcal{K}$ was a maximal clique. □

Using Lemma 6.15, we can design an efficient algorithm to compute a minimum atomic verifying set for any given DAG $\mathcal{G}$.

Theorem 6.12. *A minimum sized atomic verifying set for $\mathcal{G}$ can be computed in polynomial time in the size of $\mathcal{G}$.*

Proof. An atomic intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ separates every unoriented covered edge in $\mathcal{E}(\mathcal{G})$ if and only if the vertex set $\bigcup_{\boldsymbol{S} \in \mathcal{I}} \boldsymbol{S}$ is a vertex cover of the unoriented covered edges in $\mathcal{E}(\mathcal{G})$. Since Lemma 6.15 tells us that the edge-induced subgraph on covered edges of $\mathcal{G}$ is a forest, one can perform the standard dynamic programming algorithm to efficiently compute the minimum vertex cover on each tree. □

6.7 Adaptive search algorithm using atomic interventions

Here, we give our search algorithm based on graph separators and prove Theorem 6.13.

Algorithm 13 Search algorithm via graph separators and atomic interventions.

```
Input: Essential graph $\mathcal{E}(\mathcal{G}^*)$
Output: A fully oriented graph $\mathcal{G} \in [\mathcal{G}^*]$
Initialize $i = 0$ and $\boldsymbol{I}_0 = \emptyset$.
while the essential graph $\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*)$ still has undirected edges do
    For each chain component $\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*))$ of size $|\mathcal{H}| \geq 2$, find a $1/2$-clique
    separator $\boldsymbol{K}_{\mathcal{H}} \subseteq \boldsymbol{V}$ using Theorem 2.43.
    Define $\boldsymbol{Q} = \bigcup_{\substack{\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*)) \\ |\boldsymbol{V}(\mathcal{H})| \geq 2}} \boldsymbol{K}_{\mathcal{H}}$ as the union of clique separator nodes.
    Increment $i \leftarrow i + 1$ and form a new intervention set $\mathcal{S}_i = \{\{V\} : V \in \boldsymbol{Q}\}$ by
    interpreting $\boldsymbol{Q}$ as a collection of $|\boldsymbol{Q}|$ atomic interventions.
    Intervene on $\mathcal{S}_i$ to obtain $\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*)$ and update $\boldsymbol{I}_i \leftarrow \mathcal{I}_{i-1} \cup \mathcal{S}_i$.
```

To analyze Algorithm 13, we first prove that it terminates in $\mathcal{O}(\log n)$ iterations.

Lemma 6.39. *Algorithm 13 terminates after at most $\mathcal{O}(\log n)$ iterations.*

Proof. Fix an iteration i and chain component $\mathcal{H}$ with $1/2$-clique separator $\boldsymbol{K}_{\mathcal{H}}$. By construction, edges *within* each clique separator $\boldsymbol{K}_{\mathcal{H}}$ will be fully oriented when we perform atomic interventions on $\boldsymbol{Q}$ atomically. Note that by doing so, any edge with exactly one endpoint in $\boldsymbol{K}_{\mathcal{H}}$ will also be oriented. Thus, after each iteration, the only remaining unoriented edges lie completely within the separated components that are of half the size.

Since the algorithm always recurse on graphs of size at least half the previous iteration, we see that $|\boldsymbol{V}(\mathcal{H})| \leq n/2^i$ for any $\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*))$. Thus, all chain components will become singletons after $\mathcal{O}(\log n)$ iterations and the algorithm terminates with a fully oriented graph. □

To bound the number of interventions used in each iteration, we prove a stronger universal lower bound that is built upon the lower bound of [SMG+20].

Lemma 6.17. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ with an underlying ground truth DAG $\mathcal{G}^*$.*

$$\nu_1(\mathcal{G}^*) \geq \max_{\text{atomic intervention set } \mathcal{I} \subseteq 2^{\boldsymbol{V}}} \sum_{\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))} \left\lfloor \frac{\omega(\mathcal{H})}{2} \right\rfloor$$

Proof. Consider an arbitrary set of atomic interventions $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ and the resulting $\mathcal{I}$-essential graph $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)$. Let $\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))$ be an arbitrary chain component and define $\mathcal{S}_{\mathcal{H}} = \{\{V\} :\in \boldsymbol{V}(\mathcal{H})\}$ as the set of vertices of $\boldsymbol{V}(\mathcal{H})$. Now, let $\mathcal{I}' \subseteq 2^{\boldsymbol{V}}$ be an arbitrary atomic verifying set of $\mathcal{G}^*$. Since $\mathcal{I}'$ is a verifying set of $\mathcal{G}^*$, we have $\mathcal{E}_{\mathcal{I}'}(\mathcal{G}^*) = \mathcal{G}^*$ and $\mathcal{E}_{\mathcal{I}'}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})] = \mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]$. Then, we see that

$$\begin{aligned}
\mathcal{E}_{(\mathcal{I}' \setminus \mathcal{I}) \cap \mathcal{S}_{\mathcal{H}}}(\mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]) &= \mathcal{E}_{\mathcal{I} \cup (\mathcal{I}' \setminus \mathcal{I})}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})] && \text{By Lemma 2.53} \\
&= \mathcal{E}_{\mathcal{I}'}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})] && \text{Since } \mathcal{I} \cup (\mathcal{I}' \setminus \mathcal{I}) = \mathcal{I}' \\
&= \mathcal{G}^*[\boldsymbol{V}(\mathcal{H})] && \text{Since } \mathcal{I}' \text{ is a verifying set of } \mathcal{G}^*
\end{aligned}$$

So, $(\mathcal{I}' \setminus \mathcal{I}) \cap \mathcal{S}_{\mathcal{H}}$ is a verifying set for $\mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]$, and so is $\mathcal{I}' \cap \mathcal{S}_{\mathcal{H}}$. Thus, by minimality of ν_1, we have

$$\nu_1(\mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]) \leq |\mathcal{I}' \cap \mathcal{S}_{\mathcal{H}}| \tag{6.1}$$

for *any* atomic verifying set $\mathcal{I}' \subseteq 2^{\boldsymbol{V}}$ of $\mathcal{G}^*$. Since $\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))$, the graph $\mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]$ is a moral DAG. Since $\mathcal{H}$ is a subgraph of $\mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]$, $\omega(\mathcal{H}) \leq \omega(\mathcal{G}^*[\boldsymbol{V}(\mathcal{H})])$. Thus, by Lemma 6.16, we have

$$\nu_1(\mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]) \geq \left\lfloor \frac{\omega(\mathcal{G}^*[\boldsymbol{V}(\mathcal{H})])}{2} \right\rfloor \geq \left\lfloor \frac{\omega(\mathcal{H})}{2} \right\rfloor \tag{6.2}$$

Now, suppose $\mathcal{I}^*$ is a minimum size verifying set of $\mathcal{G}^*$. Then,

$$\begin{aligned}
\nu_1(\mathcal{G}^*) &= |\mathcal{I}^*| && \text{By definition of } \mathcal{I}^* \\
&\geq \sum_{\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))} |\mathcal{I}^* \cap \mathcal{S}_{\mathcal{H}}| && \text{Since the chain components } \mathcal{H} \text{ are disjoint} \\
&\geq \sum_{\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))} \nu_1(\mathcal{G}[\boldsymbol{V}(\mathcal{H})]) && \text{By Eq. (6.1)} \\
&\geq \sum_{\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*))} \left\lfloor \frac{\omega(\mathcal{H})}{2} \right\rfloor && \text{By Eq. (6.2)}
\end{aligned}$$

The claim follows maximizing over all possible atomic interventions $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$. □

Theorem 6.13. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ with an unknown underlying ground truth DAG $\mathcal{G}^*$. There is an algorithm that runs in polynomial time and computes an atomic* *intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ in a deterministic and adaptive manner such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*) = \mathcal{G}^*$ and $|\mathcal{I}| \in \mathcal{O}(\log(n) \cdot \nu_1(\mathcal{G}^*))$.*

Proof. Algorithm 13 runs in polynomial time because $1/2$-clique separators can be computed efficiently (see Theorem 2.43).

Fix an arbitrary iteration i of Algorithm 13 and let $\mathcal{G}_i$ be the partially oriented graph obtained after intervening on $\boldsymbol{I}_i$. By Lemma 6.17, $\sum_{\mathcal{H} \in CC(\mathcal{E}_{\boldsymbol{I}_i}(\mathcal{G}^*))} \lfloor \frac{\omega(\mathcal{H})}{2} \rfloor \leq \nu_1(\mathcal{G}^*)$. By definition of ω, we always have $|\boldsymbol{K}_{\mathcal{H}}| \leq \omega(\mathcal{H})$. Thus, Algorithm 13 uses at most $2 \cdot \nu_1(\mathcal{G}^*)$ interventions in each iteration.

By Lemma 6.39, there are $\mathcal{O}(\log n)$ iterations and so $\mathcal{O}(\log(n) \cdot \nu_1(\mathcal{G}^*))$ atomic interventions are used by Algorithm 13. □

6.8 Extension: Subset verification and search

6.8.1 Subset verification

Our subset verification results are based on the Hasse diagrams for moral DAGs. As such, we begin by defining and stating some properties about them. After which, we show that the atomic subset verification problem is equivalent to the problem of interval stabbing on a rooted tree. Full proofs are deferred to the Appendix B.1.2.

Definition 6.40 (Partial order)**.** The tuple $(\boldsymbol{X}, \leq)$ is a partially ordered set (a.k.a. poset) whenever the partial order $\leq$ on a set $\boldsymbol{X}$ satisfies three properties:

1. Reflexivity: For all $X \in \boldsymbol{X}$, $X \leq X$;

2. Anti-symmetric: For all $X, Y \in \boldsymbol{X}$, if $X \leq Y$ and $Y \leq X$, then $X = Y$;

3. Transitivity: For all $X, Y, Z \in \boldsymbol{X}$, if $X \leq Y$ and $Y \leq Z$, then $X \leq Z$.

There may be incomparable pairs of elements in $\boldsymbol{X}$. For any two elements $X, Y \in \boldsymbol{X}$, we say that Y *covers* X if $X \leq Y$ and there is no $Z \in \boldsymbol{X} \setminus \{X, Y\}$ such that $X \leq Z \leq Y$.

Definition 6.41 (Transitive reduction)**.** A transitive reduction of a directed graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ is another directed graph $\boldsymbol{G}^t = (\boldsymbol{V}, \boldsymbol{E}')$ with minimum sized $|\boldsymbol{E}'|$ such that there is a directed path from U to V in $\mathcal{G}$ if and only if there is a directed path from U to V in $\mathcal{G}^t$ for any $U, V \in \boldsymbol{V}$.

Definition 6.42 (Directed Hasse diagram)**.** Any poset $(\boldsymbol{X}, \leq)$ can be *uniquely* represented by a *directed Hasse diagram* $\mathcal{H}_{(\boldsymbol{X}, \leq)}$, a directed graph where each element in $\boldsymbol{X}$ is a vertex and there is an arc $Y \to X$ whenever Y covers X for any two elements $X, Y \in \boldsymbol{X}$. We call these arcs as *Hasse arcs*.

Any DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ induces a poset on the vertices $\boldsymbol{V}$ with respect to the ancestral relationships in the graph: $X \leq_{\mathrm{An}} Y$ whenever $X \in \mathrm{An}[Y]$. Furthermore, it is known (e.g. see [AGU72]) that the transitive reduction $\mathcal{G}^t$ of a DAG $\mathcal{G}$ is *unique*, is defined on a subset of edges (i.e. $\mathcal{E}' \subseteq \mathcal{E}$), is polynomial time computable, and is exactly the Hasse diagram $\mathcal{H}_{(\boldsymbol{V}, \leq_{\mathrm{An}})}$ defined with respect to $(\boldsymbol{V}, \leq_{\mathrm{An}})$. Since "covers" correspond to "direct children" for DAGs, we will say "Y is a direct child of X" instead of "X covers Y" to avoid confusion with the notion of covered edges. In the following, we will use $\mathcal{H}_{\mathcal{G}} = \mathcal{H}_{(\boldsymbol{V}, \leq_{\mathrm{An}})}$ to denote the Hasse diagram corresponding to a DAG $\mathcal{G}$. Note that $\mathcal{H}_{\mathcal{G}}$ can be computed in polynomial time and may have multiple roots (vertices without incoming arcs) in general.

Lemma 6.43. *If moral DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ is a single connected component, then the Hasse diagram $\mathcal{H}_{\mathcal{G}}$ is a directed tree with a unique root vertex.*

As it is known [HB12, Lemma 23] that any moral DAG whose skeleton is a connected chordal graph has exactly one source vertex, Lemma 6.43 is not entirely surprising. However, it enables us to properly define the notion of rooted subtrees in a Hasse diagram.

Definition 6.44 (Rooted subtree)**.** Let $\mathcal{H}_{\mathcal{G}}$ be a Hasse diagram of a single component moral DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$. By Lemma 6.43, $\mathcal{H}_{\mathcal{G}}$ is a rooted tree. For any vertex $Y \in \boldsymbol{V}$, the rooted subtree $\mathcal{T}_Y$ has vertices $\boldsymbol{V}(\mathcal{T}_Y) = \{U \in \boldsymbol{V} : Y \in \mathrm{An}[U]\}$ and edges $\boldsymbol{E}(\mathcal{T}_Y) = \{A \to B : A, B \in \boldsymbol{V}(\mathcal{T}_Y)\}$. See Fig. 6.8 for an illustration.

Using rooted subtrees, we prove several structural properties regarding the arc directions that are recovered by an atomic intervention, cumulating into Theorem 6.45 which states that the set $\boldsymbol{R}^{-1}(\mathcal{G}, U \to V)$ of vertices whose intervention recovers $U \to V$ forms a consecutive sequence of vertices in some branch in the Hasse diagram $\mathcal{H}_{\mathcal{G}}$.

Theorem 6.45. *Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a moral DAG and $U \to V$ be an unoriented arc in $\mathcal{E}(\mathcal{G})$. Then, $\boldsymbol{R}^{-1}(\mathcal{G}, U \to V) = \mathrm{De}[W] \cap \mathrm{An}[V]$ for some $W \in \mathrm{An}[U]$.*

Meanwhile, the following lemma tells us that covered edges correspond directly to an interval involving only the endpoints. We will later see that our subset verification result (Theorem 6.20) is a non-trivial generalization of our verification result (Theorem 6.12).

Lemma 6.46. *If $\mathcal{G}$ be a moral DAG, then the covered edges of $\mathcal{G}$ are a subset of the Hasse edges in $\mathcal{H}_{\mathcal{G}}$.*

Now, for any rooted tree $\widehat{\mathcal{G}} = (\boldsymbol{V}, \boldsymbol{E})$, an ordered pair $[U, V]_{\widehat{\mathcal{G}}} \in \boldsymbol{V} \times \boldsymbol{V}$ is called an *interval* if $U \in \mathrm{An}(V)$. If the graph is clear from context, we will drop the subscript $\widehat{\mathcal{G}}$. We say that a vertex $Z \in \boldsymbol{V}$ *stabs* an interval $[U, V]$ if and only if $Z \in \mathrm{De}[U] \cap \mathrm{An}[V]$, and that a subset $\boldsymbol{S} \subseteq \boldsymbol{V}$ stabs $[A, B]$ if $\boldsymbol{S}$ has a vertex that stabs it. Interpreting Theorem 6.45 with respect to the definition of an interval, we see that every edge $U \to V$ can be associated with some interval $[W, V]_{\mathcal{H}_{\mathcal{G}}}$, for some $W \in \mathrm{An}[U]$, such that $U \to V \in \boldsymbol{R}(\mathcal{G}, \mathcal{I})$ if and only if $\mathcal{I}$ stabs $[W, V]_{\mathcal{H}_{\widehat{\mathcal{G}}}}$. As such, we can reduce the subset verification problem on moral DAGs to the following problem.

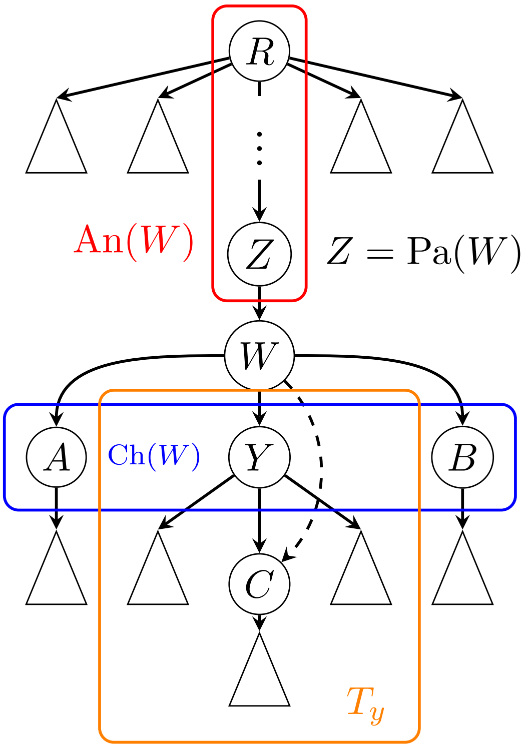


Figure 6.8: A Hasse diagram $\mathcal{H}_{\mathcal{G}}$ of some DAG $\mathcal{G}$ with root R where triangles represent unexpanded subtrees. For a vertex W, $\mathrm{An}(W)$ is the set of vertices (in red) along the unique path from R to W and $Z = \mathrm{Pa}(W)$ is the vertex directly before W. The direct children (in blue) of W are $\mathrm{Ch}(W) = \{A, B, Y\}$. If the arc $W \to C$ exists in $\mathcal{G}$, it will *not* appear in $\mathcal{H}_{\mathcal{G}}$ because $W \to Y \to C$ exists, i.e. $C \not\in \mathrm{Ch}(W)$. The rooted subtree (in orange) $\mathcal{T}_Y$ at Y includes *all* the nodes that have Y as an ancestor.

Definition 6.47 (Interval stabbing problem on a rooted tree)**.** Given a rooted tree $\widehat{\mathcal{G}} = (\boldsymbol{V}, \boldsymbol{E})$ with root $R \in \boldsymbol{V}$ and a set $\boldsymbol{J} \subseteq 2^{\boldsymbol{V} \times \boldsymbol{V}}$ of intervals of the form $[U, V]$, find a set $\boldsymbol{I} \subseteq \boldsymbol{V}$ of minimum size such that $\boldsymbol{I}$ stabs $[U, V]$ for all $[U, V] \in \boldsymbol{J}$.

The interval stabbing problem on a rooted tree can be viewed both as a special case of the set cover problem, and as a generalization of the interval stabbing problem on a line. The former is NP-hard [Kar72], while the latter can be solved using a polynomial time greedy algorithm (e.g. see [Eri19, Chapter 4, Exercise 4]). The next two results formally shows that one can reduce the subset verification problem on moral DAGs to the interval stabbing problem in polynomial time, and that the latter can be solved efficiently (see Section 6.8.1), and thus we obtain an efficient algorithm for the subset verification problem.

Lemma 6.48. *Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a connected moral DAG, $\mathcal{H}$ be the Hasse tree of $\mathcal{G}$, and $\boldsymbol{T} \subseteq \boldsymbol{E}$ be a subset of target edges. Then, there exists a set of intervals $\boldsymbol{J} \subseteq 2^{\boldsymbol{V} \times \boldsymbol{V}}$ on $\mathcal{H}$ such that any solution to minimum interval stabbing problem on $(\mathcal{H}, \boldsymbol{J})$ is a solution to the minimum sized atomic subset verification set $(\mathcal{G}, \boldsymbol{T})$.*

Lemma 6.49. *There exists a polynomial time algorithm for solving the interval stabbing problem on a rooted tree.*

Theorem 6.20. *For any DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ and subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$, there exists a polynomial time algorithm to compute the minimum sized atomic subset verifying set.*

Proof. Since closure under Meek rules can be computed in polynomial time (e.g. via [WBL21, Algorithm 2]), we can compute all $\boldsymbol{R}(\mathcal{G}, \{\{V\}\})$ for each $V \in \boldsymbol{V}$, and thus $\boldsymbol{R}^{-1}(U \to V)$ in polynomial time. The reduction given in Lemma 6.48 runs in polynomial time and we can apply the polynomial time algorithm of Lemma 6.49 to solve the resulting interval stabbing instance. □

Interestingly, *any* instance of interval stabbing on a rooted tree can also be reduced in polynomial time to an instance of subset verification on moral DAGs.

Lemma 6.50. *Let $\mathcal{H}$ be a rooted tree and $\boldsymbol{J} \subseteq 2^{\boldsymbol{V} \times \boldsymbol{V}}$ be a set of intervals on $\mathcal{H}$, for some set $\boldsymbol{V}$. Then, there exists a connected moral DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ and a subset $\boldsymbol{T} \subseteq \boldsymbol{E}$ of edges such that any solution to the minimum sized atomic subset verification set $(\mathcal{G}, \boldsymbol{T})$ is a solution to minimum interval stabbing problem on $(\mathcal{H}, \boldsymbol{J})$.*

Interval stabbing on a rooted tree

Here, we formulate a recurrence relation for the interval stabbing problem on a rooted tree and give an efficient dynamic programming implementation in Appendix B.1.3.

To formally describe the recurrence relation, we will use the following definitions to partition the given set of intervals. Given a set of intervals $\boldsymbol{J}$, we define the following subsets of intervals with respect to an arbitrary vertex $V \in \boldsymbol{V}$:

$$\begin{aligned}
\boldsymbol{E}_V &= \{[A, B] \in \boldsymbol{J} : B = V\} && \text{(End with } V\text{)}\\
\boldsymbol{M}_V &= \{[A, B] \in \boldsymbol{J} : V \in (A, B)\} && \text{(Middle with } V\text{)}\\
\boldsymbol{S}_V &= \{[A, B] \in \boldsymbol{J} : A = V\} && \text{(Start with } V\text{)}\\
\boldsymbol{W}_V &= \{[A, B] \in \boldsymbol{J} : A, B \in \boldsymbol{V}(\mathcal{T}_V) \setminus \{V\}\} && \text{(Without } V\text{)}\\
\boldsymbol{I}_V &= \boldsymbol{E}_V \cup \boldsymbol{M}_V \cup \boldsymbol{S}_V \cup \boldsymbol{W}_V && \text{(Intersect } \mathcal{T}_V\text{)}\\
\boldsymbol{B}_V &= \boldsymbol{S}_V \cup \boldsymbol{W}_V && \text{(Back of } \boldsymbol{I}_V\text{)}\\
\boldsymbol{C}_V &= \boldsymbol{E}_V \cup \boldsymbol{M}_V \cup \boldsymbol{S}_V && \text{(Covered by } V\text{)}
\end{aligned}$$

Note that $\boldsymbol{I}_V$ includes all the intervals in $\boldsymbol{J}$ that intersect with the subtree $\mathcal{T}_V$, i.e. has some vertex in $\boldsymbol{V}(\mathcal{T}_V)$. Meanwhile, $\boldsymbol{C}_V$ includes all the intervals that will be covered whenever V is chosen in the output set, i.e. V will stab intervals in $\boldsymbol{C}_V$. Observe that $\boldsymbol{I}_Y \subseteq \boldsymbol{I}_V$ for any $Y \in \mathrm{De}(V)$. See Fig. 6.9 for an example illustrating these definitions.

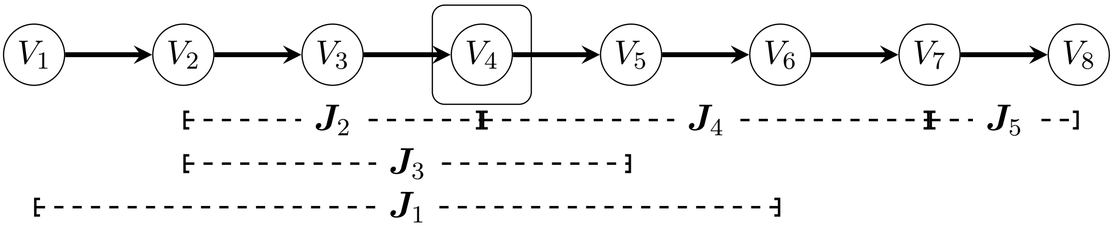


Figure 6.9: Consider the rooted tree $\widehat{\mathcal{G}}$ with $V_1 \to \ldots \to V_8$ and $\boldsymbol{J} = \{\boldsymbol{J}_1, \ldots, \boldsymbol{J}_5\}$, where $\boldsymbol{J}_1 = [V_1, V_6]$, $\boldsymbol{J}_2 = [V_2, V_4]$, $\boldsymbol{J}_3 = [V_2, V_5]$, $\boldsymbol{J}_4 = [V_4, V_7]$, and $\boldsymbol{J}_5 = [V_7, V_8]$. Then, $\boldsymbol{E}_{V_4} = \{\boldsymbol{J}_2\}$, $\boldsymbol{M}_{V_4} = \{\boldsymbol{J}_1, \boldsymbol{J}_3\}$, $\boldsymbol{S}_{V_4} = \{\boldsymbol{J}_4\}$, $\boldsymbol{W}_{V_4} = \{\boldsymbol{J}_5\}$.

To solve the interval stabbing problem, we perform recursion from the root towards the leaves, solving subproblems defined on subsets of the intervals that are still “relevant” at each subtree. More formally, for any set of intervals $\boldsymbol{U} \subseteq \boldsymbol{J}$, let $\mathrm{opt}(\boldsymbol{U}, V)$ denote the *size* of the optimum solution to stab all the intervals in $\boldsymbol{U}$ using only vertices $\boldsymbol{V}(\mathcal{T}_V)$ in the subtree $\mathcal{T}_V$ rooted at V. There are three possible cases while recursing from the root towards the leaves:

Case 1. If $\boldsymbol{U} \cap \boldsymbol{E}_V \neq \emptyset$, then V *must* be in any valid solution output and we recurse on the set $(\boldsymbol{U} \setminus \boldsymbol{C}_V) \cap \boldsymbol{I}_Y$ for subtree $\mathcal{T}_Y$ rooted at each child $Y \in \mathrm{Ch}(V)$.

Case 2. If $\boldsymbol{U} \cap \boldsymbol{E}_V = \emptyset$ and V is in the output, then we *can* recurse on the set $(\boldsymbol{U} \setminus \boldsymbol{C}_V) \cap \boldsymbol{I}_Y$ for subtree $\mathcal{T}_Y$ rooted at each child $Y \in \mathrm{Ch}(V)$.

Case 3. If $\boldsymbol{U} \cap \boldsymbol{E}_V = \emptyset$ and V is *not* in the output, then we *need to* recurse on the set $\boldsymbol{U} \cap \boldsymbol{I}_Y$ subtree $\mathcal{T}_Y$ rooted at each child $Y \in \mathrm{Ch}(V)$.

For any $V \in \boldsymbol{V}$ and $Y \in \mathrm{Ch}(V)$, we have $\boldsymbol{C}_V \cap \boldsymbol{I}_Y \subseteq \boldsymbol{E}_Y \cup \boldsymbol{M}_Y$ by definition. So, $(\boldsymbol{U} \setminus \boldsymbol{C}_V) \cap \boldsymbol{I}_Y = \boldsymbol{U} \cap \boldsymbol{B}_Y$. The correctness of the first case is trivial while Lemma 6.51 formalizes the correctness of the second and third cases.

Lemma 6.51. *At least one of the following must hold for any optimal solution* opt *with size* $\mathrm{opt}(\boldsymbol{U}, R)$ *to the interval stabbing problem with respect to ordering* π *and any vertex* $V \in \boldsymbol{V}$ *with* $\boldsymbol{E}_V = \emptyset$*:*

1. *Either* $V \in \mathrm{opt}$ *or* opt *includes some ancestor of* V*.*

2. *For* $Y \in \mathrm{Ch}(V)$ *such that* $\boldsymbol{C}_V \cap \boldsymbol{I}_Y \neq \emptyset$*, we must have* $W_{V,Y} \in \mathrm{opt}$ *for some* $W_{V,Y} \in \mathrm{De}(V) \cap \mathrm{An}[B_{V,Y}]$*, where* $[A_{V,Y}, B_{V,Y}] = \underset{[A,B] \in \boldsymbol{U} \cap \boldsymbol{C}_V \cap \boldsymbol{I}_Y}{\mathrm{argmin}} \{\pi(B)\}$.

Therefore, we have the following recurrence relation:

$$\mathrm{opt}(\boldsymbol{U}, V) = \begin{cases} \infty & \text{if } \boldsymbol{U} \not\subseteq \boldsymbol{I}_V \\ \alpha_V & \text{if } \boldsymbol{U} \subseteq \boldsymbol{I}_V, \boldsymbol{U} \cap \boldsymbol{E}_V \neq \emptyset \\ \min\{\alpha_V, \beta_V\} & \text{if } \boldsymbol{U} \subseteq \boldsymbol{I}_V, \boldsymbol{U} \cap \boldsymbol{E}_V = \emptyset \end{cases} \tag{6.3}$$

$$\text{where} \qquad \alpha_V = 1 + \sum_{Y \in \mathrm{Ch}(V)} \mathrm{opt}(\boldsymbol{U} \cap \boldsymbol{B}_Y, Y)$$
$$\beta_V = \sum_{Y \in \mathrm{Ch}(V)} \mathrm{opt}(\boldsymbol{U} \cap \boldsymbol{I}_Y, Y)$$

That is, we must pick V to be in the output set whenever $\boldsymbol{E}_V \neq \emptyset$, while α_V and β_V correspond to the decisions of picking V into the output and ignoring V from the output respectively. Then, $\mathrm{opt}(\boldsymbol{J}, R)$ is the optimum solution size to the interval stabbing problem, where R as the root of the given rooted tree.

In Appendix B.1.3, we explain how to implement Eq. (6.3) efficiently using dynamic programming. To do so, we first compute the Euler tour data structure on $\mathcal{G}$ and use it to define an ordering on $\boldsymbol{J}$ so that our state space ranges over the indices of a sorted array instead of a subset of intervals.

6.8.2 Subset search

We first give the proofs of Lemma 6.21 and Lemma 6.22 to justify why a bound of $\mathcal{O}(\log n \cdot \nu_1(\mathcal{G}^*, \boldsymbol{T}))$ for any subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$ is unattainable in general. Then, we propose a subset search algorithm SubsetSearch for a given node-induced subgraph $\mathcal{H}$ and prove Theorem 6.24.

Lemma 6.21. *Given a subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$, intervening on the vertices in a vertex cover of $\boldsymbol{T}$ atomically will fully orient all edges in $\boldsymbol{T}$.*

Proof. Each intervention will orient all the incident edges. □

Lemma 6.22. *Fix any integer $n \geq 1$. There exists a fully unoriented essential graph on $2n$ vertices and a subset $\boldsymbol{T} \subseteq \boldsymbol{E}$ on n edges such that the size of the minimum vertex cover of $\boldsymbol{T}$ is $\mathrm{vc}(\boldsymbol{T})$ and any algorithm needs at least $\mathrm{vc}(\boldsymbol{T}) - 1$ number atomic interventions to orient all the edges in $\boldsymbol{T}$ against an adaptive adversary that reveals arc directions consistent with a DAG $\mathcal{G}^* \in [\mathcal{G}]$ with $\nu_1(\mathcal{G}^*, \boldsymbol{T}) = 1$.*

Proof. Let the vertex set be $\{V_1, \ldots, V_{2n}\}$. Fig. 6.10 illustrates our lower bound construction: form a clique $\{V_1, \ldots, V_n\}$ and add an edge between vertex $V_i - V_{n+i}$ for $i \in [n]$, then let $\boldsymbol{T} = \{V_i \to V_{n+i} : i \in [n]\}$ be the set of target edges. We will restrict the vertices outside the clique come after the clique nodes in any topological ordering and allow the adversary to adaptively decide on the relative orderings of vertices within the clique based on the performed interventions.

By construction, the essential graph has no v-structures and the minimum vertex cover of $\boldsymbol{T}$ has size $\omega(n)$. To orient all the edges in set $\boldsymbol{T}$, we just need to orient on the source vertex of the clique and then apply Meek rules. Therefore, $\nu_1(\mathcal{G}^*, \boldsymbol{T}) = 1$ for any graph $\mathcal{G}^*$ in this equivalence class. Meanwhile, an adaptive adversary can always decide that vertices outside the clique have come after the the clique nodes in the ordering, and always decide that the i^{th} vertex V in the clique that we intervene on within the clique has ordering $\pi(V) = n - i + 1$, and thus we only learn the orientations of arcs incident to V. Since intervening on vertices outside the clique only learns the incident arc itself while intervening on the other endpoint in the clique recovers more arc orientations, we may assume w.l.o.g. that search algorithms will only intervene on vertices within the clique. So, to orient all the edges in the set $\boldsymbol{T}$, we need to figure out the source vertex and this is as hard to finding an item in unsorted array of length n, which requires at least $n - 1$ queries. □

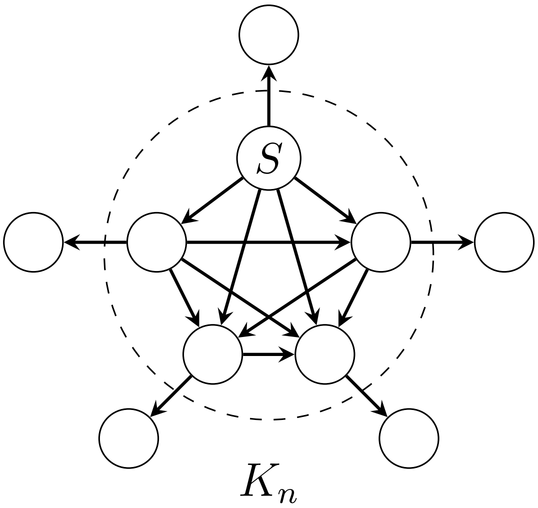


Figure 6.10: Adaptive lower bound construction of $\mathcal{G}$ with $n = 5$: Given an integer $n \geq 1$, construct a directed clique $\mathcal{K}_n$ and have each clique node point to a fresh node outside of the clique. The dashed n dashed arcs are chosen to be the target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$. The essential graph $\mathcal{E}(\mathcal{G})$ is completely undirected and any permutation ordering on the clique nodes are valid. Intervening on the source S of the clique is sufficient to fully orient $\boldsymbol{T}$ with the aid of Meek rules. However, an adaptive adversary can always decide that vertices outside the clique have come after the the clique nodes in the ordering, and always decide that the i^{th} vertex V in the clique that we intervene on within the clique has ordering $\pi(V) = n - i + 1$, and thus we only learn the orientations of arcs incident to V.

We design SubsetSearch by modifying Algorithm 13 to only assign non-zero weights on vertices from the given node-induced subgraph $\mathcal{H}$.

For analysis, we rely on the following known results Lemma 2.44 and Lemma 6.17 and mirrors the approach of Theorem 6.13: we first argue that Algorithm 14 terminates after $\mathcal{O}(\log |\boldsymbol{V}(\mathcal{H})|)$ iterations, and then argue that each iteration uses at most $\mathcal{O}(\nu_1(\mathcal{G}^*))$ atomic interventions.

Lemma 6.52. *SubsetSearch terminates after at most $\mathcal{O}(\log |\rho(\mathcal{I}, \boldsymbol{V}(\mathcal{H}))|)$ iterations.*

Proof. For any iteration i, the chain components in $\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})]$ are chordal since node-induced subgraphs of a chordal graph are also chordal. By choice of $c(V)$ and

Algorithm 14 SubsetSearch

```
Input: Essential graph $\mathcal{E}(\mathcal{G}^*)$ and node-induced subgraph $\mathcal{H}$
Output: A partially oriented graph $\mathcal{G}$ such that $\mathcal{G}[\boldsymbol{V}(\mathcal{H})] = \mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]$.
Initialize $i = 0$ and $\boldsymbol{I}_0 = \emptyset$.
while the essential graph $\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})]$ still has undirected edges do
    For each chain component $\mathcal{H}_{CC} \in CC(\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*))$ with $|\rho(\mathcal{I}_i \cup \mathcal{I}, \boldsymbol{V}(\mathcal{H}_{CC}))| \geq 2$,
    find a $1/2$-clique separator $\mathcal{K}_{\mathcal{H}_{CC}}$ using Lemma 2.44, with respect to
        $$c(V) = \begin{cases} \frac{n}{\rho(\boldsymbol{I}_i \cup \mathcal{I}, \boldsymbol{V}(\mathcal{H}_{CC}))} & \text{for } V \in \boldsymbol{V}(\mathcal{H}) \\ 0 & \text{for } V \in \boldsymbol{V} \setminus \boldsymbol{V}(\mathcal{H}) \end{cases}$$
    Define $\boldsymbol{Q} = \bigcup_{\substack{\mathcal{H} \in CC(\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*)) \\ |\boldsymbol{V}(\mathcal{H})| \geq 2}} \boldsymbol{K}_{\mathcal{H}_{CC}}$ as the union of clique separator nodes.
    Increment $i \leftarrow i + 1$ and form a new intervention set $\mathcal{S}_i = \{\{V\} : V \in \boldsymbol{Q}\}$ by
    interpreting $\boldsymbol{Q}$ as a collection of $|\boldsymbol{Q}|$ atomic interventions.
    Intervene on $\mathcal{S}_i$ to obtain $\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*)$ and update $\boldsymbol{I}_i \leftarrow \mathcal{I}_{i-1} \cup \mathcal{S}_i$.
```

Lemma 2.44, all connected components will have the same total weight. Since each vertex in $\boldsymbol{V}(\mathcal{H}_{CC}) \cap \boldsymbol{V}(\mathcal{H})$ is assigned the same weight via $c(V)$, the number of vertices from $\boldsymbol{V}(\mathcal{H})$ within any connected component $\mathcal{H}_{CC}$ is at least halved per iteration. Thus, after $\mathcal{O}(\log |\boldsymbol{V}(\mathcal{H})|)$ iterations, all connected components have at most one vertex from $\boldsymbol{V}(\mathcal{H})$, which in turn means that all edges within the node-induced graph $\mathcal{H}$ has been oriented. □

Theorem 6.24. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ of an unknown underlying DAG $\mathcal{G}^*$ and let $\mathcal{H}$ be an node-induced subgraph of $\mathcal{G}^*$. There exists an algorithm that runs in polynomial time and computes an atomic intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ in a deterministic and adaptive manner such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})] = \mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]$ and $|\mathcal{I}| \in \mathcal{O}(\log(|\rho(\mathcal{I}, \boldsymbol{V}(\mathcal{H}))|) \cdot \nu_1(\mathcal{G}^*, \boldsymbol{E}))$.*

Proof. Fix an arbitrary iteration i of SubsetSearch and let $\mathcal{G}_i$ be the partially oriented graph obtained after intervening on $\boldsymbol{I}_i$. By Lemma 6.17, $\sum_{H \in CC(\mathcal{E}_{\boldsymbol{I}_i}(G^*))} \lfloor \frac{\omega(H)}{2} \rfloor \leq \nu_1(\mathcal{G}^*, \boldsymbol{E})$. By definition of ω, we always have $|\mathcal{K}_{\mathcal{H}}| \leq \omega(\mathcal{H})$. Thus, SubsetSearch uses at most $2 \cdot \nu_1(\mathcal{G}^*, \boldsymbol{E})$ interventions in each iteration. Meanwhile, Lemma 6.52 states that SubsetSearch terminates after $\mathcal{O}(\log |\rho(\mathcal{I}, \boldsymbol{V}(\mathcal{H}))|)$ iterations and so the algorithm uses at most $\mathcal{O}(\log(|\rho(\mathcal{I}, \boldsymbol{V}(\mathcal{H}))|) \cdot \nu_1(\mathcal{G}^*, \boldsymbol{E}))$ atomic interventions in total. □

6.9 Extension: k-bounded interventions

The proof of Theorem 6.25 relies on an intermediate result that performing bounded sized interventions atomically can only increase the number of oriented edges; this result is intuitive because doing so only increases the total number of separated edges.

Lemma 6.53. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ and $\mathcal{G} \in [\mathcal{G}^*]$. Suppose $\mathcal{I}$ is an arbitrary bounded size intervention set. Intervening on vertices in $\cup_{\boldsymbol{S} \in \mathcal{I}} \boldsymbol{S}$ one at a time, in an* *atomic fashion, can only increase the number of separated covered edges of $\mathcal{G}$.*

Proof. Consider an arbitrary covered edge $U - V$ that was separated by some intervention $\boldsymbol{S} \in \mathcal{I}$. This means that $|\{U, V\} \cap \boldsymbol{S}| = 1$. W.l.o.g., suppose $U \in \boldsymbol{S}$. Then, when we intervene on U in an atomic fashion, we would also separate the edge $U - V$. □

Theorem 6.25. *For any DAG $\mathcal{G}$, we have $\nu_k(\mathcal{G}) \geq \lceil \frac{\nu_1(\mathcal{G})}{k} \rceil$.*

Proof. A bounded size intervention set of size strictly less than $\lceil \frac{\ell}{k} \rceil$ involves strictly less than ℓ vertices. By Theorem 6.12 and Lemma 6.53, such an intervention set cannot be a verifying set. □

Theorem 6.26. *If $\nu_1(\mathcal{G}) = \ell$, then $\nu_k(\mathcal{G}) \geq \lceil \ell/k \rceil$ and there exists a polynomial time algorithm to compute a bounded size intervention set $\mathcal{I}$ of size $|\mathcal{I}| \leq \lceil \frac{\ell}{k} \rceil + 1$.*

Proof. By Lemma 6.15, the edge-induced subgraph on covered edges of $\mathcal{G}$ is a forest and is thus 2-colorable.

Let $\mathcal{A}$ be any minimum sized atomic verifying set of $\mathcal{G}$ involving ℓ vertices. Split the vertices in $\mathcal{A}$ into partitions according to the 2-coloring. By construction, vertices belonging in the same partite will *not* be adjacent and thus choosing them together to be in an intervention $\boldsymbol{S}$ will *not* reduce the number of separated covered edges. Now, form interventions of size k by greedily picking vertices in $\mathcal{A}$ within the same partite. For the remaining unpicked vertices (strictly less than k of them), we form a new intervention with them. Repeat the same process for the other partite.

This greedy process forms groups of size k and at most 2 groups of sizes strictly less than k, one from each partite. Suppose that we formed z groups of size k in total and two "leftover groups" of sizes x and y, where $0 \leq x, y < k$. Then, $\ell = z \cdot k + x + y$, $\frac{\ell}{k} = z + \frac{x+y}{k}$, and we formed at most $z + 2$ groups. If $0 \leq x + y < k$, then $\lceil \frac{\ell}{k} \rceil = z + 1$. Otherwise, if $k \leq x + y < 2k$, then $\lceil \frac{\ell}{k} \rceil = z + 2$. In either case, we use at most $\lceil \frac{\ell}{k} \rceil + 1$ interventions, each of size $\leq k$.

One can compute a bounded size intervention set efficiently because the following procedures can all be run in polynomial time: (i) checking if each edge is a covered edge; (ii) computing a minimum vertex cover on a tree; (iii) 2-coloring a tree; (iv) greedily grouping vertices into sizes $\leq k$.

The claim follows by invoking Theorem 6.25 and recalling that $\mathcal{A}$ is a minimum sized atomic verifying set of $\mathcal{G}$, i.e. $\nu_1(\mathcal{G}) = \ell$. □

Observe that there exists graphs and values k such that the optimal bounded size verifying set requires at least $\lceil \frac{\ell}{k} \rceil + 1$, and thus our upper bound is tight in the worst case: Fig. 6.11 shows there exists a family of graphs (and values k) such that the optimal bounded size verifying set requires $\lceil \frac{\ell}{k} \rceil + 1$. However, we do not have a proof that Theorem 6.26 is optimal in general, or counter example that it is not.

Conjecture 6.54. The construction of bounded size verifying set given in Theorem 6.26 is optimal for all causal graphs.

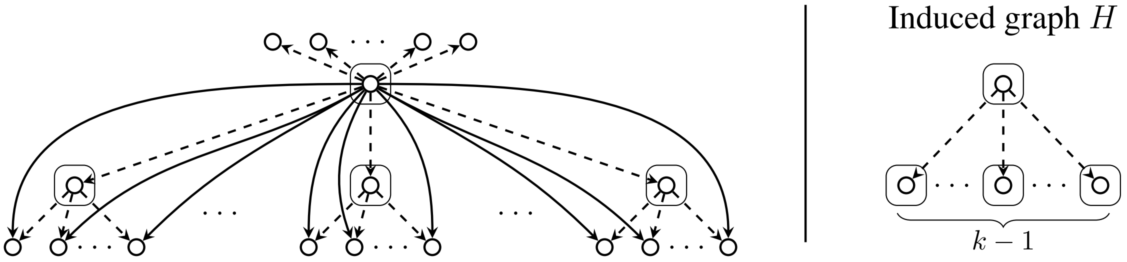


Figure 6.11: A DAG with its covered edges given in dashed arcs. The edge-induced subgraph of the covered edges is a tree and the minimum vertex cover is all the non-leaf vertices (the boxed vertices) of size ℓ. Denote the graph induced by the boxed vertices by $\mathcal{H}$ (right figure). Now consider the star graph $\mathcal{H}$ on $\ell = k$ nodes with $k-1$ leaves. All the leaf nodes can be put in the same intervention without affecting the separation of any covered edges. However, including the root with any of the leaf nodes in a same intervention will cause covered edges to be unseparated. Thus, using bounded size interventions of size at most k, verifying such a DAG requires at least $\lceil \frac{\ell}{k} \rceil + 1 = 2$ interventions.

We now show how to apply the label computation of Algorithm 12 as a black-box to generalize our (subset) search results to the setting with k-bounded interventions.

Theorem 6.28. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ with an unknown underlying ground truth DAG $\mathcal{G}^*$. For any integer $k > 1$, there is an algorithm that runs in polynomial time and computes a k-bounded intervention set $\mathcal{I} \subseteq 2^{\mathbf{V}}$ in a deterministic and adaptive manner such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*) = \mathcal{G}^*$ and $|\mathcal{I}| \in \mathcal{O}(\log(n) \cdot \log(k) \cdot \nu_k(\mathcal{G}^*))$.*

Proof. Recall from Algorithm 13 that, in each of the $\mathcal{O}(\log n)$ iterations, we repeatedly compute $1/2$-clique separators and aggregate the nodes into $\boldsymbol{Q}$, which we interpret as a collection of atomic intervention set. We modify the algorithm by invoking Algorithm 12 on $\boldsymbol{Q}$ to k-bounded intervention sets. The produced k-bounded intervention set $\boldsymbol{S}_i$ will separate all edges separated by $\boldsymbol{Q}$ whilst having size $|\mathcal{S}_i| \leq \left\lceil \frac{|\boldsymbol{Q}|}{k'} \right\rceil \cdot \left\lceil \log_{\lceil \frac{|\boldsymbol{Q}|}{k'} \rceil} |\boldsymbol{Q}| \right\rceil$, where $k' = \min\{k, |\boldsymbol{Q}|/2\} > 1$. Note that our modified algorithm still runs in polynomial time because the additional step of label computation runs in polynomial time.

For analysis, let us fix an arbitrary iteration i of our modified algorithm and let $\mathcal{G}_i$ be the partially oriented graph obtained after intervening on $\boldsymbol{I}_i$. Then,

$$\begin{aligned}
\nu_k(\mathcal{G}^*) &\geq \left\lceil \frac{\nu_1(\mathcal{G}^*)}{k} \right\rceil && \text{By Theorem 6.25} \\
&\geq \left\lceil \frac{1}{k} \cdot \sum_{\mathcal{H} \in CC(\mathcal{E}(\mathcal{G}^*))} \left\lfloor \frac{\omega(\mathcal{H})}{2} \right\rfloor \right\rceil && \text{By Lemma 6.17} \\
&\geq \Omega(|\boldsymbol{Q}|/k) && \text{Since } \boldsymbol{Q} \text{ is the union of clique separator nodes}
\end{aligned}$$

Note that $\nu_k(\mathcal{G}^*) \geq 1$ always. We now consider the two cases of k'.

Case 1: $k \leq |\boldsymbol{Q}|/2$. Then, $k' = k$ and

$$|\mathcal{S}_i| \leq \left\lceil \frac{|\boldsymbol{Q}|}{k} \right\rceil \cdot \left\lceil \log_{\lceil \frac{|\boldsymbol{Q}|}{k} \rceil} |\boldsymbol{Q}| \right\rceil \leq \left\lceil \frac{|\boldsymbol{Q}|}{k} \right\rceil \cdot \left\lceil \frac{\log |\boldsymbol{Q}|}{\log \frac{|\boldsymbol{Q}|}{k}} \right\rceil$$
$$\leq \left\lceil \frac{|\boldsymbol{Q}|}{k} \right\rceil \cdot \lceil \log(k) + 1 \rceil \in \mathcal{O}\left(\frac{|\boldsymbol{Q}|}{k} \cdot \log(k) \right)$$

Case 2: $k \geq |\boldsymbol{Q}|/2$. Then, $k' = |\boldsymbol{Q}|/2$ and

$$|\mathcal{S}_i| \leq 2 \cdot \lceil \log_2 |\boldsymbol{Q}| \rceil \in \mathcal{O}(\log(k))$$

In either case, we see that $|\boldsymbol{S}_i| \in \mathcal{O}\left(\nu_k(\mathcal{G}^*) \cdot \log k\right)$. By Lemma 6.39, there are $\mathcal{O}(\log n)$ iterations and so the total number of k-bounded interventions used by our modified algorithm is $\mathcal{O}(\log(n) \cdot \log(k) \cdot \nu_k(\mathcal{G}^*))$. □

Theorem 6.29. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ of an unknown underlying DAG $\mathcal{G}^*$ and let $\mathcal{H}$ be an node-induced subgraph of $\mathcal{G}^*$. For any integer $k > 1$, there is an algorithm that runs in polynomial time and computes a k-bounded intervention set $\mathcal{I} \subsetneq 2^{\boldsymbol{V}}$ in a deterministic and adaptive manner such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*)[\boldsymbol{V}(\mathcal{H})] = \mathcal{G}^*[\boldsymbol{V}(\mathcal{H})]$ and $|\mathcal{I}| \in \mathcal{O}(\log(|\rho(\mathcal{I}, \boldsymbol{V}(\mathcal{H}))|) \cdot \log(k) \cdot \nu_k(\mathcal{G}^*, \boldsymbol{E}))$.*

Proof. Fix an arbitrary iteration i of SubsetSearch and let $\mathcal{G}_i$ be the partially oriented graph obtained after intervening on $\boldsymbol{I}_i$. Applying exactly the same proof as Theorem 6.28, we see that $|\boldsymbol{S}_i| \in \mathcal{O}\left(\nu_k(\mathcal{G}^*, \boldsymbol{E}) \cdot \log k\right)$. By Lemma 6.52, there are $\mathcal{O}(\log |\rho(\mathcal{I}, \boldsymbol{V}(\mathcal{H}))|)$ iterations and so $\mathcal{O}(\log(|\rho(\mathcal{I}, V(H))|) \cdot \log(k) \cdot \nu_k(\mathcal{G}^*, \boldsymbol{E}))$ bounded size interventions are used by SubsetSearch. □

Chapter 7

Probably approximately correct high-dimensional causal effect estimation given a valid adjustment set

> “Causa latet: vis est notissima.”
> *(“The cause is hidden: the effect is well known.”)*
>
> \- Ovid in *Metamorphoses*

7.1 Introduction

Suppose that we can draw i.i.d. samples from an *unknown* probability distribution $\mathcal{P}(\boldsymbol{V})$ over discrete random variables $\boldsymbol{V}$, and that we wish to estimate the interventional distribution of $\boldsymbol{Y} \subset \boldsymbol{V}$ when $\boldsymbol{X} \subset \boldsymbol{V}$ is set to $\boldsymbol{x}$. This causal problem has been studied under different assumptions in the two major causal frameworks commonly referred to as Neyman-Rubin's potential outcomes (PO) framework [Rub74, SN90, Sek09] and Pearl's graphical causal modeling framework [Pea09a], where the desired estimand is written as $\mathcal{P}(\boldsymbol{Y}(\boldsymbol{x}) = \boldsymbol{y})$ and $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) = \mathcal{P}(\boldsymbol{Y} = \boldsymbol{y} \mid \mathrm{do}(\boldsymbol{X} = \boldsymbol{x}))$ respectively. In both frameworks, this problem is known as *causal effect estimation* and has important downstream implications such as estimating treatment effects $\mathbb{E}(\boldsymbol{Y} \mid \mathrm{do}(\boldsymbol{X} = \boldsymbol{x})) - \mathbb{E}(\boldsymbol{Y} \mid \mathrm{do}(\boldsymbol{X} = \boldsymbol{x}'))$, or $\mathbb{E}(\boldsymbol{Y}(\boldsymbol{x})) - \mathbb{E}(\boldsymbol{Y}(\boldsymbol{x}'))$ in the potential outcome notation, for $\boldsymbol{x} \neq \boldsymbol{x}'$ and where the expectations are taken over the values of $\boldsymbol{Y}$. In this chapter, we consider the following problem from the viewpoint of distribution learning [KMR^{+}94] under the Probably Approximately Correct (PAC) learning model [Val84].

The PAC causal effect estimation problem. Given (1) estimation tolerance $\lambda > 0$, (2) failure tolerance $\delta > 0$, (3) sample access to a distribution $\mathcal{P}(\boldsymbol{V})$, and (4) an interventional query $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$, output an estimate $\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y})$ such that

$$\Pr\left(\left|\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y}) - \mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})\right| \leq \lambda\right) \geq 1 - \delta$$

For this problem to be well-posed, one must be able to relate the observational distribution $\mathcal{P}(\boldsymbol{V})$ to the interventional distribution $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ via some identification formula, i.e. $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ must be uniquely determined by $\mathcal{P}(\boldsymbol{V})$. While identifiability of $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ requires additional assumptions in general, here we focus on a commonly studied identification formula that involve a set of variables $\boldsymbol{Z} \subset \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ such that

$$\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) = T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}, \text{ where } T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} := \sum\nolimits_{\boldsymbol{z} \in \Sigma_{\boldsymbol{Z}}} \mathcal{P}(\boldsymbol{Y} = \boldsymbol{y} \mid \boldsymbol{Z} = \boldsymbol{z}, \boldsymbol{X} = \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{Z} = \boldsymbol{z}), \tag{7.1}$$

where $\Sigma_{\boldsymbol{Z}}$ denotes the alphabet of the variables $\boldsymbol{Z}$. For instance, in the PO framework, Eq. (7.1) holds under the assumptions of consistency and conditional ignorability of $\boldsymbol{X}$ with respect to $\boldsymbol{Z}$; see Lemma B.11 for a simple derivation. Meanwhile, the graphical framework models the observational and interventional distributions via a causal graph $\mathcal{G}^*$ over $\boldsymbol{V}$, where $\mathcal{G}^*$ is possibly known or unknown, and may contain directed, bidirected, and other kinds of edges depending on the context. Then, Eq. (7.1) can be shown to hold if $\boldsymbol{Z}$ satisfies certain graphical criterion with respect to $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$, such as the (generalized) backdoor criterion or the (generalized) adjustment criteria [Pea95, SVR10, MC15, PTKM18]. Following the latter viewpoint, we call $\boldsymbol{Z}$ a valid adjustment set for $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ if $\boldsymbol{Z}$ satisfies $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) = T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$ in Eq. (7.1), but we emphasize that our results are framework-agnostic, i.e., they do not depend on how Eq. (7.1) is derived.

In particular, we establish our PAC guarantees by directly analyzing the sample complexity required to produce an estimate $\widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$ of $T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$. A recent work [ZBHK24] shows that $\Omega\left(\frac{1}{\lambda^2 \alpha_{\boldsymbol{Z}}} + \frac{|\Sigma_{\boldsymbol{Z}}|}{\lambda \alpha_{\boldsymbol{Z}}}\right)$ samples are sufficient to ensure an expectation bound of $\mathbb{E}\left(|T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}|\right) \leq \lambda$, where $\alpha_{\boldsymbol{Z}}$ is a positivity (a.k.a. overlap) parameter that is common in causal effect estimation; see Appendix B.2.2 for derivation translating their stated bound into this form. [ZBHK24] also presented a minimax lower bound showing that linear dependency on $|\Sigma_{\boldsymbol{Z}}|$ is unavoidable. Since $|\Sigma_{\boldsymbol{Z}}|$ grows exponentially with the size of $\boldsymbol{Z}$ (e.g. when all variables are binary, we have $|\Sigma_{\boldsymbol{Z}}| = 2^{|\boldsymbol{Z}|}$), it is critical to use *small* adjustment sets whenever possible.

Given a valid adjustment set $\boldsymbol{Z} \subseteq \boldsymbol{V}$ as an initial input, we explore the possibility of searching for smaller adjustment sets with the objective of using less total samples than directly producing a λ-good estimate $\widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$. We are able to obtain lower sample complexities because of the adage from the property testing literature that "testing can be cheaper than learning". In particular, we develop testing-based algorithms to find a candidate adjustment set $\boldsymbol{S} \subseteq \boldsymbol{Z}$, then estimate $\widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$ and bound its error from $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) = T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$ via triangle inequality:

$$\left|\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) - \widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}\right| = \left|T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}\right| \leq \left|T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}\right| + \left|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}\right| \\ \leq \varepsilon_1 + \varepsilon_2 = \lambda$$

The overall error bound (λ) will then be a function of the misspecification bias error term (ε_1) and the estimation error term (ε_2). There is a natural tradeoff between these two sources of error: using $\boldsymbol{S} \subseteq \boldsymbol{Z}$ for adjustment introduces misspecification bias if $\boldsymbol{S}$ is not a valid adjustment set, but this bias may dominated by a corresponding reduction in estimation error if $\boldsymbol{S}$ is smaller than $\boldsymbol{Z}$. While our approach is best appreciated through the lens of the graphical causality framework, it also applies in the PO setting since our method only relies on conditional independence tests using i.i.d. samples from $\mathcal{P}(\boldsymbol{V})$.

7.1.1 Valid adjustment sets in the context of different frameworks

In this chapter, we take as our starting point knowledge of some $\boldsymbol{Z} \subset \boldsymbol{V}$ that is a valid adjustment set for $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$, i.e., we assume that Eq. (7.1) holds for some known set $\boldsymbol{Z} \subset \boldsymbol{V}$. Instead of starting directly from this point, one may prefer to derive Eq. (7.1) from more foundational assumptions. We emphasize that our results hold in *any* framework from which Eq. (7.1) can be derived, and briefly describe two such examples here.

In the potential outcomes (PO) framework, $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ is usually written as $\mathcal{P}(\boldsymbol{Y}(\boldsymbol{x}) = \boldsymbol{y})$, where $\boldsymbol{Y}(\boldsymbol{x})$ is a random variable denoting the potential outcome under an intervention that sets $\boldsymbol{X}$ to $\boldsymbol{x}$. Then, Eq. (7.1) is implied under the standard consistency assumption and conditional ignorability of $\boldsymbol{X}$ with respect to $\boldsymbol{Z}$; see Lemma B.11.

Alternatively, Eq. (7.1) can be derived in the graphical causality framework, which relates the distributions $\mathcal{P}(\boldsymbol{V})$ and $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ to a (possibly unknown) causal graph $\mathcal{G}$ over the random variables $\boldsymbol{V}$. In the graphical causality framework, one typically assumes that the distributions $\mathcal{P}(\boldsymbol{V})$ and $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ are related via d-separation in $\mathcal{G}$ and related graphs. Thus, Eq. (7.1) can be derived from these assumptions and graphical conditions on $\boldsymbol{Z}$, see Section 8.1 for examples of such conditions.

For sake of clarity, we have positioned this work from the perspective of causal effect estimation, and emphasized how our primary assumption (knowledge of a valid adjustment set $\boldsymbol{Z}$) is compatible with both the potential outcomes (PO) and graphical frameworks for causality; see Section 8.1 for related work and connections between this work and existing work from both of these perspectives.

7.2 Our main results

We denote $\boldsymbol{X}$ as the intervened treatment variables and $\boldsymbol{Y}$ as the outcome variables of interest. For some $\boldsymbol{x} \in \Sigma_{\boldsymbol{X}}$ and $\boldsymbol{y} \in \Sigma$, our goal is to estimate $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$, which denotes the probability that $\boldsymbol{Y}$ takes on the value $\boldsymbol{y}$ if we intervene to set $\boldsymbol{X}$ equal to $\boldsymbol{x}$. Throughout this chapter, for any $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ arbitrary, we define

$$\alpha_{\boldsymbol{A}} = \min_{\boldsymbol{a} \in \Sigma_{\boldsymbol{A}}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{a}) \tag{7.2}$$

Our first main result extends the result of [ZBHK24] to the PAC setting by bounding the estimation error $|T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}|$ for arbitrary subsets $\boldsymbol{A} \subseteq \boldsymbol{V}$, where $\widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ is the estimate of $T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ obtained using empirical sample estimates of $\mathcal{P}(\boldsymbol{Y} = \boldsymbol{y} \mid \boldsymbol{A} = \boldsymbol{a}, \boldsymbol{X} = \boldsymbol{x})$ and $\mathcal{P}(\boldsymbol{A} = \boldsymbol{a})$ for all $\boldsymbol{a} \in \Sigma_{\boldsymbol{A}}$.

Theorem 7.1 (Estimation error)**.** *Suppose we are given (1) estimation tolerance $\varepsilon > 0$, (2) failure tolerance $\delta > 0$, (3) sample access to $\mathcal{P}(\boldsymbol{V})$, and (4) a subset $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$. Then, there is an algorithm that uses $\widetilde{\mathcal{O}}\left(\left(\frac{|\Sigma_{\boldsymbol{A}}|}{\varepsilon \alpha_{\boldsymbol{A}}} + \frac{1}{\varepsilon^2 \alpha_{\boldsymbol{A}}} + \frac{|\Sigma_{\boldsymbol{A}}|}{\varepsilon^2}\right) \cdot \log \frac{1}{\delta}\right)$ samples and produces an estimate $\widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ such that $\Pr(|\widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \varepsilon) \geq 1 - \delta$.*

Note that, up to logarithmic factors and the additional $\frac{|\Sigma_{\boldsymbol{A}}|}{\varepsilon^2}$ factor, the sample complexity of the PAC bound matches the sample complexity of the expectation bound. Here, we switched from λ to ε and from $\boldsymbol{Z}$ to $\boldsymbol{A}$ to emphasize that the estimation error is only one part of our overall bound. Surprisingly, although covariate adjustment is one of the simplest and most widely-used estimation techniques in causality, this result is (to the best of our knowledge) the first PAC bound on causal effect estimation for discrete variables. In particular, previous works either focus on different estimands (under additional assumptions such as knowing a causal graph) or consider continuous variables and primarily provide only asymptotic results; we discuss related works in Section 8.1.

Importantly, the sample complexity depends exponentially on $|\boldsymbol{A}|$, and so when $\boldsymbol{A}$ is large, it is of great practical interest to use another adjustment set $\boldsymbol{S}$ of smaller size, i.e. $|\boldsymbol{S}| < |\boldsymbol{A}|$. As a paradigmatic example, consider the causal graph given in Fig. 7.1: instead of directly using $\boldsymbol{Z} = \{A_1, \ldots, A_k, B\}$, there are two possible smaller subsets within $\boldsymbol{Z}$ itself that satisfy the (generalized) backdoor adjustment criterion [Pea95, SVR10, MC15, PTKM18], and that therefore also serve as valid adjustment sets.

As a first approach for obtaining smaller adjustment sets, we consider Markov blankets of the treatments $\boldsymbol{X}$. In particular, we say that $\boldsymbol{S} \subseteq \boldsymbol{Z}$ is a Markov blanket of $\boldsymbol{X}$ with respect to $\boldsymbol{Z}$ when $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Z} \setminus \boldsymbol{S} \mid \boldsymbol{S}$, and one can show that $T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} = T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$ when this conditional independence holds. In the example given in Fig. 7.1, the parental set $\mathrm{Pa}(X) = \{A_1, \ldots, A_k\}$ satisfies this condition: $X \perp\!\!\!\perp \{B\} \mid \mathrm{Pa}(X)$. To adapt this notion in the finite sample setting, we consider an approximate version of conditional independence and define a parameter $\Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Z} \setminus \boldsymbol{S} \mid \boldsymbol{S}}$ that measures how much the conditional independence

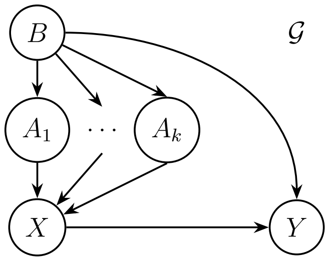


Figure 7.1: Consider the graphical causality framework and suppose we are given $\boldsymbol{Z} = \{A_1, \ldots, A_k, B\}$ as a valid adjustment set for $\mathcal{P}_x(y)$ in the above causal graph $\mathcal{G}$. Both the parental set $\mathrm{Pa}(X) = \{A_1, \ldots, A_k\}$ and the singleton set $\{B\}$ satisfy the backdoor adjustment criterion [Pea95] and are also valid adjustment sets.

condition is violated in $\mathcal{P}(\boldsymbol{V})$; see Definition 2.40. When $\Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{Z} \setminus \boldsymbol{S} \mid \boldsymbol{S}} \leq \varepsilon$, we say that $\boldsymbol{S}$ is an ε-Markov blanket of $\boldsymbol{X}$ with respect to $\boldsymbol{Z}$.

Definition 7.2 ((Approximate) Markov blanket)**.** Consider an arbitrary subset $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$. A subset $\boldsymbol{S} \subseteq \boldsymbol{A}$ is called a *Markov blanket* of $\boldsymbol{X}$ with respect to $\boldsymbol{A}$ if $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{A} \setminus \boldsymbol{S} \mid \boldsymbol{S}$ and an *ε-Markov blanket* if $\boldsymbol{X} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{A} \setminus \boldsymbol{S} \mid \boldsymbol{S}$.

We show that $|T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}}$ whenever $\boldsymbol{S}$ is an ε-Markov blanket of $\boldsymbol{A}$. In particular, if $\boldsymbol{A} = \boldsymbol{Z}$ is a valid adjustment set, then this bound applies to the misspecification bias mentioned above. Our next result bounds the sample complexity for discovering an ε-Markov blanket.

Theorem 7.3 (Approximate Markov blanket discovery)**.** *Suppose we are given (1)* $\varepsilon > 0$, *(2)* $\delta > 0$, *(3) sample access to a distribution* $\mathcal{P}(\boldsymbol{V})$*, and (4) an arbitrary subset* $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$. *Suppose that there is a Markov blanket of* $\boldsymbol{X}$ *with respect to* $\boldsymbol{A}$ *with* k *variables. Then, there is an algorithm that uses* $\widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{S}|}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{A}}|} \cdot \log \frac{1}{\delta}\right)$ *samples and produces a subset* $\boldsymbol{S} \subseteq \boldsymbol{A}$ *such that* $|\boldsymbol{S}| \leq k$, $\Pr\left(\Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{A} \setminus \boldsymbol{S} \mid \boldsymbol{S}} > \varepsilon\right) \geq 1 - \delta$, *and* $\Pr\left(|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}}\right) \geq 1 - \delta$.

Now, suppose that Theorem 7.3 outputs $\boldsymbol{S} \subseteq \boldsymbol{Z}$ when given a valid adjustment set $\boldsymbol{Z}$. While $|\boldsymbol{S}|$ may be smaller than $|\boldsymbol{Z}|$, it may still be much larger than the smallest valid adjustment set for $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$. For example, we see that $|\boldsymbol{Z}| = k + 1 > k = |\mathrm{Pa}(X)| \gg |\{B\}| = 1$ in Fig. 7.1 where $\boldsymbol{Z}$, $\mathrm{Pa}(X)$, and $\{B\}$ are all valid adjustment sets. Our next result aims to find an adjustment set $\boldsymbol{S}' \subseteq \boldsymbol{Z}$ of *minimal size* given a valid adjustment set $\boldsymbol{Z}$ and an ε-Markov blanket $\boldsymbol{S} \subseteq \boldsymbol{Z}$ of it. To this end, we introduce the more general concept of a screening set of an arbitrary subset $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$.

Definition 7.4 ((Approximate) Screening set)**.** Let $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ and $\boldsymbol{S} \subseteq \boldsymbol{A}$. A subset $\boldsymbol{B} \subseteq \boldsymbol{A}$ is called a screening set for $(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{X}, \boldsymbol{Y})$ if $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{B} \mid \boldsymbol{X} \cup \boldsymbol{B}$ and $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{B} \setminus \boldsymbol{S} \mid \boldsymbol{S}$. Meanwhile, the subset $\boldsymbol{B}$ is called an ε-screening set for $(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{X}, \boldsymbol{Y})$ if $\boldsymbol{Y} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{S} \setminus \boldsymbol{B} \mid \boldsymbol{X} \cup \boldsymbol{B}$ and $\boldsymbol{X} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{B} \setminus \boldsymbol{S} \mid \boldsymbol{S}$.

As a technical side note (see the exposition after Lemma 7.9), given an adjustment set $\boldsymbol{S}$, the screening set condition for $(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{X}, \boldsymbol{Y})$ is sound for $\boldsymbol{B}$ to be a valid adjustment set, but it is incomplete in general, in that sense that there may exist valid adjustment sets that do not satisfy the screening set condition. In the worst case, our algorithm in Theorem 7.5 will output $\boldsymbol{S}' = \boldsymbol{S}$.

Theorem 7.5 (Beyond approximate Markov blankets)**.** *Suppose we are given (1) $\varepsilon > 0$, (2) $\delta > 0$, (3) sample access to $\mathcal{P}(\boldsymbol{V})$, (4) an arbitrary subset $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$, and (5) an ε-Markov blanket $\boldsymbol{S} \subseteq \boldsymbol{A}$. Suppose there is a screening set $\boldsymbol{B}$ for $(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{X}, \boldsymbol{Y})$ such that $|\boldsymbol{B}| = k'$ and $|\Sigma_{\boldsymbol{B}}| \leq |\Sigma_{\boldsymbol{S}}|$. There is an algorithm that uses $\widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{S}'|}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{A}}|} \cdot \log \frac{1}{\delta}\right)$ samples and produces a subset $\boldsymbol{S}' \subseteq \boldsymbol{A}$ such that $|\boldsymbol{S}'| \leq k'$, $|\Sigma_{\boldsymbol{S}'}| \leq |\Sigma_{\boldsymbol{S}}|$ and $\Pr\left(|T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{2\varepsilon}{\alpha_{\boldsymbol{S}}}\right) \geq 1 - \delta$.*

As we shall see, unlike many existing causal discovery methods, e.g. the PC algorithm [SGS00], which perform a sequence of dependent conditional independence checks, our algorithms for Theorem 7.3 and Theorem 7.5 use a *non-dependent* collection of conditional independence tests, allowing us to avoid error propagation and control the sample complexity of our procedures.

Finally, one can combine the PAC bound results above to yield an overall PAC bound guarantee for solving the causal effect estimation problem as follows. Since Lemma 7.7 tells us that $T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$ and $T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$ are close whenever $\boldsymbol{S}$ is an ε-Markov blanket of $\boldsymbol{X}$ with respect to $\boldsymbol{Z}$, we can employ the algorithm in Theorem 7.3 to find a subset $\boldsymbol{S} \subseteq \boldsymbol{Z}$ such that $T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} \approx T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$. Using the ε-Markov blanket $\boldsymbol{S}$, we can further use the algorithm in Theorem 7.5 to find a subset $\boldsymbol{S}' \subseteq \boldsymbol{Z}$ such that $T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} \approx T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$. Depending on whether $|\Sigma_{\boldsymbol{S}}|$ or $|\Sigma_{\boldsymbol{S}'}|$ is smaller, we can employ Theorem 7.1 to obtain an estimate $\widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$ or $\widehat{T}_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}$, and use that as an estimate for $T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} = \mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$.

In practical situations where one is given a fixed number of samples, we can re-express the results of Theorem 7.1, Theorem 7.3 and Theorem 7.5 in terms of an error upper bound. Then, one can derive a condition under which a combined approach based on above results estimates $\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y})$ via $\widehat{T}_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}$, for some $\boldsymbol{S}' \subseteq \boldsymbol{Z}$, and provably achieves a smaller asymptotic error than directly estimating $\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y})$ via $\widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$. The condition relies on the positivity of $\alpha_{\boldsymbol{S}}$ for subsets $\boldsymbol{S}, \boldsymbol{S}' \subseteq \boldsymbol{Z}$ which are unknown a priori. However, if one is willing to make lower bound assumptions on these α values, possibly due to background knowledge, then one can obtain a result in the same vein as Theorem 7.6.

Theorem 7.6 (PAC causal effect estimation with positivity)**.** *Suppose we are given (1) $\varepsilon > 0$, (2) $\delta > 0$, (3) n i.i.d. samples from $\mathcal{P}(\boldsymbol{V})$, (4) an interventional query $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$, (5) a valid adjustment set $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$, and (6) guaranteed that $\alpha_{\boldsymbol{S}} \geq \alpha \in (0, 1)$ for any $\boldsymbol{S} \subseteq \boldsymbol{Z}$. Then, there is an algorithm that outputs a subset $\boldsymbol{S}^* \subseteq \boldsymbol{Z}$ and an estimate*

$\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y}) = \widehat{T}_{\boldsymbol{S}^*,\boldsymbol{x},\boldsymbol{y}}$ *such that* $\Pr\left(\left|\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y}) - \mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})\right| \leq \varepsilon\right) \geq 1 - \delta$ *for some error term*

$$\varepsilon \in \widetilde{\mathcal{O}}\left(\frac{1}{n} \cdot \frac{|\boldsymbol{\Sigma}_{\boldsymbol{S}^*}|}{\alpha} + \frac{1}{\sqrt{n}} \cdot \left(\frac{\sqrt{|\boldsymbol{Z}|} \cdot (|\boldsymbol{\Sigma}_{\boldsymbol{X}}| \cdot |\boldsymbol{\Sigma}_{\boldsymbol{Y}}| \cdot |\boldsymbol{\Sigma}_{\boldsymbol{Z}}|)^{\frac{1}{4}}}{\alpha} + \frac{1}{\sqrt{\alpha}} + \sqrt{|\boldsymbol{\Sigma}_{\boldsymbol{S}^*}|}\right)\right).$$

Moreover, if there exists a Markov blanket $\boldsymbol{S}$ of $\boldsymbol{X}$ such that $|\boldsymbol{S}| \cdot \sqrt{\frac{|\boldsymbol{\Sigma}_{\boldsymbol{X}}|}{|\boldsymbol{\Sigma}_{\boldsymbol{Z}}|}} < \max\left\{\frac{|\boldsymbol{\Sigma}_{\boldsymbol{Z}}|}{n}, \frac{\alpha_{\boldsymbol{S}}}{|\boldsymbol{\Sigma}_{\boldsymbol{Z}}|}, \alpha_{\boldsymbol{S}}^2\right\}$*, then* $|\boldsymbol{S}^*| \leq k$*.*

7.3 Technical overview

While our notation and language is closer to Pearl's graphical causal modeling framework [Pea09a], all of our results are compatible with both the PO and graphical frameworks as long as Eq. (7.1) holds for the given $\boldsymbol{Z}$. This is because our analysis is purely probabilistic in nature, with the causal interpretation always going back to assuming that $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) = T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$ as a starting point. As such, the technical results presented here may be of independent interest for future work.

The sample complexity bound of Theorem 7.1 heavily relies on a common technique in the property testing literature known as Poissonization; see Section 2.5.1. The high level idea is that instead of drawing n i.i.d. samples, we will draw $N_{\mathrm{Poi}} \sim \mathrm{Poi}(n)$ i.i.d samples, where N_{Poi} is a random Poisson variable, so that the random count for each realized value will be independent. Meanwhile, in our error analyses in Theorem 7.3 and Theorem 7.5, we manipulate approximate conditional independence terms $\Delta_{\boldsymbol{A} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{B} \mid \boldsymbol{C}}$ (from Definition 2.40) and adjustment terms $T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ (from Eq. (7.1)) for various subsets $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \subseteq \boldsymbol{V}$. We begin by formally defining $\alpha_{\boldsymbol{A}}$ with respect to any $\boldsymbol{x} \in \Sigma_{\boldsymbol{X}}$ and any arbitrary subset $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$:

$$\alpha_{\boldsymbol{A}} = \min_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A}}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{a}) \tag{7.3}$$

By standard probability manipulations, one can easily obtain the following alternative representation of $\Delta_{\boldsymbol{A} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{B} \mid \boldsymbol{C}}$ for arbitrary disjoint subsets $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \subseteq \boldsymbol{V}$.

$$\begin{aligned}\Delta_{\boldsymbol{A} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{B} \mid \boldsymbol{C}} &= \sum_{\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}} \mathcal{P}(\boldsymbol{c}) \cdot |\mathcal{P}(\boldsymbol{a}, \boldsymbol{b} \mid \boldsymbol{c}) - \mathcal{P}(\boldsymbol{a} \mid \boldsymbol{c}) \cdot \mathcal{P}(\boldsymbol{b} \mid \boldsymbol{c})| \\ &= \sum_{\boldsymbol{a},\boldsymbol{b},\boldsymbol{c}} \mathcal{P}(\boldsymbol{a}, \boldsymbol{c}) \cdot |\mathcal{P}(\boldsymbol{b} \mid \boldsymbol{a}, \boldsymbol{c}) - \mathcal{P}(\boldsymbol{b} \mid \boldsymbol{c})| \leq \varepsilon \end{aligned} \tag{7.4}$$

Meanwhile, for any arbitrary disjoint subsets $\boldsymbol{A}, \boldsymbol{B} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$, $T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ can be re-expressed in multiple ways (depending on the desired analytical use case) using law of total probability as

$$T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} = \sum_{\boldsymbol{a}} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{a}) = \sum_{\boldsymbol{a},\boldsymbol{b}} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{b}, \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{b} \mid \boldsymbol{a}, \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{a})$$
$$= \sum_{\boldsymbol{a},\boldsymbol{b}} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{a}) \cdot \mathcal{P}(\boldsymbol{b} \mid \boldsymbol{a}) \quad (7.5)$$

The correctness of Theorem 7.3 follows from the following result that $T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$ and $T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ are close whenever $\boldsymbol{S}$ is an ε-Markov blanket of $\boldsymbol{S}$ with respect to $\boldsymbol{A}$, and that there is a sample efficient way to obtain such an ε-Markov blanket.

Lemma 7.7 (Misspecification error)**.** *If $\boldsymbol{S} \subseteq \boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ such that $\boldsymbol{X} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{A} \setminus \boldsymbol{S} \mid \boldsymbol{S}$, then $|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}}$.*

We additionally complement Lemma 7.7 with a hardness result of Lemma 7.8.

Lemma 7.8 (Misspecification error lower bound)**.** *Let $0 \leq \sqrt{\varepsilon} \leq \alpha \leq 1/2$. There exists $\mathcal{P}(\boldsymbol{V})$ such that (i) $\boldsymbol{Z}$ is a valid adjustment set, (ii) $\boldsymbol{S} \subset \boldsymbol{Z}$ satisfies $\boldsymbol{X} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{Z} \setminus \boldsymbol{S} \mid \boldsymbol{S}$, (iii) $\alpha_{\boldsymbol{S}} \geq \alpha$, and (iv) $|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}| \geq \frac{\varepsilon}{16\alpha}$.*

Similar in spirit to Theorem 7.3, the correctness of Theorem 7.5 relies on the relating $T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$ and $T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ via some conditional independence relations. Given an ε-Markov blanket $\boldsymbol{S} \subseteq \boldsymbol{A}$, we will search for a minimal sized $\boldsymbol{S}' \subseteq \boldsymbol{A}$ satisfying $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'$ and $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}$. This is a sound approach because of Lemma 7.9.

Lemma 7.9 (Adjustment soundness)**.** *Let $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ be an arbitrary subset. If $\boldsymbol{S} \subseteq \boldsymbol{A}$ and $\boldsymbol{S}' \subseteq \boldsymbol{A}$ such that $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'$ and $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}$, then $T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} = T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$.*

While this approach is always sound, it may not discover the smallest possible subset satisfying $T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} = T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$ for any choice of $\boldsymbol{S} \subseteq \boldsymbol{A}$. However, there exists special scenarios in which this approach is also complete; see Appendix B.2.3. In the context of valid backdoor adjustments, the intuition behind the additional $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'$ condition can be appreciated by setting $\boldsymbol{A} = \{A_1, \ldots, A_k, B\}$, $\boldsymbol{S} = \{A_1, \ldots, A_k\}$, and $\boldsymbol{S}' = \{B\}$ in Fig. 7.1. In this setup, we see that $\boldsymbol{S}$ is a valid backdoor adjustment set because it blocks all non-causal backdoor paths from X to Y, $Y \perp\!\!\!\perp \{A_1, \ldots, A_k\} \mid \{X, B\}$, and $X \perp\!\!\!\perp B \mid \{A_1, \ldots, A_k\}$. Observe that conditioning on X blocks any paths from $\boldsymbol{S}$ to Y that has a causal path from X to Y as a subpath. So, $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'$ would imply that $\boldsymbol{S}'$ also blocks non-causal X to Y paths since any such paths passing through $\boldsymbol{S} \setminus \boldsymbol{S}'$ has to pass through $\boldsymbol{S}'$ to reach Y.

Finally, there is nothing technically special about Theorem 7.6 besides simply combining the results Theorem 7.1, Theorem 7.3, and Theorem 7.5 in a straightforward fashion.

7.4 Sample complexity for empirical estimation

In this section, we prove Theorem 7.1, our upper bound on the sample complexity of estimating $T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ given any $\boldsymbol{A} \subseteq \boldsymbol{V}$. For analysis purposes, we will use the Poissonization sampling process (Section 2.5.1) so that we invoke Lemma 2.38 to obtain PAC style bounds.

Lemma 7.10. *Suppose we have i.i.d. sample access to $\mathcal{P}(\boldsymbol{V})$. Given integer $n > 0$ as a sampling parameter, we take $N_{\mathrm{Poi}} \sim \mathrm{Poi}(n)$ samples. For any $\boldsymbol{U} \subseteq \boldsymbol{V}$, let the random variable $N_{\boldsymbol{u}}$ denote the number of times $\boldsymbol{u} \in \boldsymbol{\Sigma_U}$ was realized within the n_{Poi} samples. Then, the following statements hold:*

1. *Let $\boldsymbol{A}, \boldsymbol{B} \subseteq \boldsymbol{V}$ be disjoint sets of variables. For any $\boldsymbol{a}, \boldsymbol{a}' \in \boldsymbol{\Sigma_A}$ and $\boldsymbol{b}, \boldsymbol{b}' \in \boldsymbol{\Sigma_B}$ with $\boldsymbol{b} \neq \boldsymbol{b}'$, the ratios of random variables $\frac{N_{\boldsymbol{a},\boldsymbol{b}}}{N_{\boldsymbol{b}}}$ and $\frac{N_{\boldsymbol{a}',\boldsymbol{b}'}}{N_{\boldsymbol{b}'}}$ are independent.*

2. *Let $\boldsymbol{A}, \boldsymbol{B} \subseteq \boldsymbol{V}$ be disjoint sets of variables. For any $\boldsymbol{a} \in \boldsymbol{\Sigma_A}$, $\boldsymbol{b} \in \boldsymbol{\Sigma_B}$, and integer $k \geq 1$, we have $\left(\frac{N_{\boldsymbol{a},\boldsymbol{b}}}{N_{\boldsymbol{b}}} - \mathcal{P}(\boldsymbol{a} \mid \boldsymbol{b}) \mid N_{\boldsymbol{b}} \geq k\right) \sim \mathrm{subG}\left(\frac{1}{4k}\right)$.*

Proof. We prove each item one at a time.

1. By Lemma 2.38, the random variables $N_{\boldsymbol{b}}$ and $N_{\boldsymbol{b}'}$ are independent since $\boldsymbol{b} \neq \boldsymbol{b}'$. Then since $N_{\boldsymbol{a},\boldsymbol{b}}$ and $N_{\boldsymbol{a}',\boldsymbol{b}'}$ are subcounts of $N_{\boldsymbol{b}}$ and $N_{\boldsymbol{b}'}$ respectively, so the corresponding ratios are also independent.

2. By Lemma 2.38, we have $(N_{\boldsymbol{a},\boldsymbol{b}} \mid N_{\boldsymbol{b}} = k) \sim \mathrm{Bin}(k, \mathcal{P}(\boldsymbol{a} \mid \boldsymbol{b}))$. Conditioned on $N_{\boldsymbol{b}} = k$, Lemma 2.14 implies that $(N_{\boldsymbol{a},\boldsymbol{b}} - \mathbb{E}(N_{\boldsymbol{a},\boldsymbol{b}})) = (N_{\boldsymbol{a},\boldsymbol{b}} - k \cdot \mathcal{P}(\boldsymbol{a} \mid \boldsymbol{b})) \sim \mathrm{subG}(\frac{k}{4})$. Thus, $\left(\frac{N_{\boldsymbol{a},\boldsymbol{b}}}{N_{\boldsymbol{b}}} - \mathcal{P}(\boldsymbol{a} \mid \boldsymbol{b}) \mid N_{\boldsymbol{b}} = k\right) \sim \mathrm{subG}(\frac{1}{4k})$. The claim follows via Lemma 2.35. □

Theorem 7.1 (Estimation error)**.** *Suppose we are given (1) estimation tolerance $\varepsilon > 0$, (2) failure tolerance $\delta > 0$, (3) sample access to $\mathcal{P}(\boldsymbol{V})$, and (4) a subset $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$. Then, there is an algorithm that uses $\widetilde{\mathcal{O}}\left(\left(\frac{|\boldsymbol{\Sigma_A}|}{\varepsilon \alpha_{\boldsymbol{A}}} + \frac{1}{\varepsilon^2 \alpha_{\boldsymbol{A}}} + \frac{|\boldsymbol{\Sigma_A}|}{\varepsilon^2}\right) \cdot \log \frac{1}{\delta}\right)$ samples and produces an estimate $\widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ such that $\Pr(|\widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \varepsilon) \geq 1 - \delta$.*

Proof. By definition, we have

$$\begin{aligned} T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} &= \sum_{\boldsymbol{a}} \left(\mathcal{P}(\boldsymbol{a}) \cdot \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{a}) - \frac{N_{\boldsymbol{a}}}{N_{\mathrm{Poi}}} \cdot \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}}\right) \\ &= \sum_{\boldsymbol{a}} \mathcal{P}(\boldsymbol{a}) \cdot \left(\mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{a}) - \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}}\right) + \sum_{\boldsymbol{a}} \left(\mathcal{P}(\boldsymbol{a}) - \frac{N_{\boldsymbol{a}}}{N_{\mathrm{Poi}}}\right) \cdot \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} \end{aligned}$$

where $N_{\boldsymbol{a}}$, N_{Poi}, $N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}$, and $N_{\boldsymbol{x},\boldsymbol{a}}$ are random Poisson variables from the Poissonization process with $N_{\mathrm{Poi}} \sim \mathrm{Poi}(n)$ for some parameter n; see Section 2.5.1. Since $N_{\mathrm{Poi}} = \sum_{\boldsymbol{a}} N_{\boldsymbol{a}} = \sum_{\boldsymbol{a},\boldsymbol{x}} N_{\boldsymbol{x},\boldsymbol{a}} = \sum_{\boldsymbol{a},\boldsymbol{x},\boldsymbol{y}} N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}$, we see that $0 \leq \frac{N_{\boldsymbol{a}}}{N_{\mathrm{Poi}}} \leq 1$ and $0 \leq \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} \leq 1$ for each of these fractional terms.

Let us define a threshold $\tau > 0$ and partition the values of $\boldsymbol{A}$ accordingly:

$$\boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau} = \{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A}} : \mathcal{P}(\boldsymbol{x}, \boldsymbol{a}) \geq \tau\}$$

Since $\alpha_{\boldsymbol{A}} = \min_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A}}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{a})$, we see that $\mathcal{P}(\boldsymbol{a}) \leq \frac{\tau}{\alpha_{\boldsymbol{A}}}$ for $\boldsymbol{a} \not\in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}$. Now, let us define three summations $J_{<\tau}$, $J_{\geq \tau}$, and K so that $T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} = J_{<\tau} + J_{\geq \tau} + K$:

$$J_{<\tau} = \sum_{\boldsymbol{a} \not\in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \mathcal{P}(\boldsymbol{a}) \cdot \left(\mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{a}) - \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} \right) \tag{7.6}$$

$$J_{\geq \tau} = \sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \mathcal{P}(\boldsymbol{a}) \cdot \left(\mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{a}) - \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} \right) \tag{7.7}$$

$$K = \sum_{\boldsymbol{a}} \left(\mathcal{P}(\boldsymbol{a}) - \frac{N_{\boldsymbol{a}}}{N_{\text{Poi}}} \right) \cdot \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} \tag{7.8}$$

We will proceed to bound each of $|J_{<\tau}|$, $|J_{\geq \tau}|$, and $|K|$.

The easiest is $|J_{<\tau}|$, which follows from the definition of $\boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}$:

$$\begin{aligned}
|J_{<\tau}| &= \left| \sum_{\boldsymbol{a} \not\in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \mathcal{P}(\boldsymbol{a}) \cdot \left(\frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} - \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{a}) \right) \right| && \text{(Definition of } |J_{<\tau}|\text{)} \\
&\leq \sum_{\boldsymbol{a} \not\in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \mathcal{P}(\boldsymbol{a}) \cdot \left| \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} - \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{a}) \right| && \text{(By triangle inequality and } \mathcal{P}(\boldsymbol{a}) \geq 0\text{)} \\
&\leq \sum_{\boldsymbol{a} \not\in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \mathcal{P}(\boldsymbol{a}) && \left(\text{Since } \left| \tfrac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} - \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{a}) \right| \leq 1\right) \\
&\leq \frac{\tau \cdot |\boldsymbol{\Sigma}_{\boldsymbol{A}}|}{\alpha_{\boldsymbol{A}}} && \left(\text{Since } \mathcal{P}(\boldsymbol{a}) \leq \tfrac{\tau}{\alpha_{\boldsymbol{A}}} \text{ for } \boldsymbol{a} \not\in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau} \text{ and } |\boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}| \leq |\boldsymbol{\Sigma}_{\boldsymbol{A}}|\right)
\end{aligned}$$

To bound $|J_{\geq \tau}|$, consider the concentration event $\mathcal{E}^J_{\geq \tau}$ defined as follows:

$$\mathcal{E}^J_{\geq \tau} = \bigcap_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \left\{ N_{\boldsymbol{x},\boldsymbol{a}} > \frac{n \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{a})}{2} \right\} \tag{7.9}$$

We first observe that the event $\mathcal{E}^J_{\geq \tau}$ holds with good probability.

$$\begin{aligned}
1 - \Pr(\mathcal{E}^J_{\geq \tau}) &\leq \sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \Pr\left(N_{\boldsymbol{x},\boldsymbol{a}} \leq \frac{n \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{a})}{2} \right) && \text{(Union bound)} \\
&\leq \sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \exp\left(-\frac{n \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{a})}{12} \right) \\
& && \text{(Using that } N_{\boldsymbol{x},\boldsymbol{a}} \sim \text{Poi}(n \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{a})) \text{ and applying Lemma 2.36)} \\
&\leq |\boldsymbol{\Sigma}_{\boldsymbol{A}}| \cdot \exp\left(-\frac{n\tau}{12} \right) && \text{(Since } \mathcal{P}(\boldsymbol{x}, \boldsymbol{a}) \geq \tau \text{ for } \boldsymbol{z} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau} \subseteq \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}\text{)}
\end{aligned}$$

Under the event $\mathcal{E}^J_{\geq \tau}$, we have $N_{\boldsymbol{x},\boldsymbol{a}} > \frac{n \cdot \mathcal{P}(\boldsymbol{x},\boldsymbol{a})}{2}$ for any $\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}$, so item 2 of Lemma 7.10 implies that

$$\left(\frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} - \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{a}) \;\middle|\; N_{\boldsymbol{x},\boldsymbol{a}} \geq \frac{n \cdot \mathcal{P}(\boldsymbol{x},\boldsymbol{a})}{2} \right) \\ \sim \mathrm{subG}\left(\frac{1}{4} \cdot \frac{2}{n \cdot \mathcal{P}(\boldsymbol{x},\boldsymbol{a})} \right) = \mathrm{subG}\left(\frac{1}{2n \cdot \mathcal{P}(\boldsymbol{x},\boldsymbol{a})} \right) ,$$

for any $\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}$. For any two disjoint $\boldsymbol{a}, \boldsymbol{a}' \in \boldsymbol{\Sigma}_{\boldsymbol{A}}$, we see that $(\boldsymbol{x}, \boldsymbol{a})$ and $(\boldsymbol{x}, \boldsymbol{a}')$ are distinct values in the domain $\boldsymbol{\Sigma}_{\boldsymbol{X}} \times \boldsymbol{\Sigma}_{\boldsymbol{A}}$, so item 1 of Lemma 7.10 tells us that the terms $\frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}}$ and $\frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}'}}{N_{\boldsymbol{x},\boldsymbol{a}'}}$ are independent. Lemma 2.34 further tells us that $J_{\geq \tau} = \sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \mathcal{P}(\boldsymbol{a}) \cdot \left(\frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} - \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{a}) \right) \sim \mathrm{subG}\left(\sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \frac{\mathcal{P}(\boldsymbol{a})^2}{2n \cdot \mathcal{P}(\boldsymbol{x},\boldsymbol{a})} \right)$ since coefficients $\{\mathcal{P}(\boldsymbol{a})\}_{\boldsymbol{a} \in \boldsymbol{A}}$ are just (unknown) real numbers. Then, for any $t > 0$, Definition 2.33 states that

$$\Pr\left(|J_{\geq \tau}| > t \mid \mathcal{E}^J_{\geq \tau} \right) \leq 2 \exp\left(- \frac{t^2}{2 \sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \frac{\mathcal{P}(\boldsymbol{a})^2}{2n \cdot \mathcal{P}(\boldsymbol{x},\boldsymbol{a})}} \right) \leq 2 \exp\left(-n \alpha_{\boldsymbol{A}} t^2 \right)$$

where the last inequality is because $\alpha_{\boldsymbol{A}} = \min_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A}}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{a})$ and $\boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau} \subseteq \boldsymbol{\Sigma}_{\boldsymbol{A}}$:

$$\sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \frac{\mathcal{P}(\boldsymbol{a})^2}{2n \cdot \mathcal{P}(\boldsymbol{x},\boldsymbol{a})} = \sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \frac{\mathcal{P}(\boldsymbol{a})}{2n \cdot \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{a})} \leq \sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A} \geq \tau}} \frac{\mathcal{P}(\boldsymbol{a})}{2n \cdot \alpha_{\boldsymbol{A}}} \leq \frac{1}{2n \cdot \alpha_{\boldsymbol{A}}}$$

To bound $|K|$, we reduce to the analysis to the problem of producing an ε-close estimate of $\mathcal{P}(\boldsymbol{A})$ by observing that $0 \leq \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} \leq 1$ and $\frac{N_{\boldsymbol{a}}}{N_{\mathrm{Poi}}}$ is the empirical estimate of $\mathcal{P}(\boldsymbol{a})$ for each $\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A}}$. That is,

$$\begin{aligned} |K| &= \left| \sum_{\boldsymbol{a}} \left(\mathcal{P}(\boldsymbol{a}) - \frac{N_{\boldsymbol{a}}}{N_{\mathrm{Poi}}} \right) \cdot \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} \right| && \text{(By Eq. (7.8))} \\ &= \sum_{\boldsymbol{a}} \left| \mathcal{P}(\boldsymbol{a}) - \frac{N_{\boldsymbol{a}}}{N_{\mathrm{Poi}}} \right| \cdot \left| \frac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} \right| && \text{(By triangle inequality)} \\ &\leq \sum_{\boldsymbol{a}} \left| \mathcal{P}(\boldsymbol{a}) - \frac{N_{\boldsymbol{a}}}{N_{\mathrm{Poi}}} \right| && \text{(Since } 0 \leq \tfrac{N_{\boldsymbol{y},\boldsymbol{x},\boldsymbol{a}}}{N_{\boldsymbol{x},\boldsymbol{a}}} \leq 1\text{)} \\ &\leq \sum_{\boldsymbol{a}} \left| \mathcal{P}(\boldsymbol{a}) - \widehat{\mathcal{P}}(\boldsymbol{a}) \right| && \text{(By defining empirical distribution } \widehat{\mathcal{P}}(\boldsymbol{a}) = \tfrac{N_{\boldsymbol{a}}}{N_{\mathrm{Poi}}}\text{)} \end{aligned}$$

By Lemma 2.39, when $N_{\mathrm{Poi}} \geq c_0 \cdot \left(\frac{|\boldsymbol{\Sigma}_{\boldsymbol{A}}| + \log \frac{1}{\delta'}}{(\varepsilon')^2} \right)$ for some tolerance parameters $\varepsilon', \delta' > 0$ and absolute constant $c_0 > 0$, we will have

$$\Pr\left(|K| \leq \varepsilon' \right) \leq \Pr\left(\sum_{\boldsymbol{a} \in \boldsymbol{\Sigma}_{\boldsymbol{A}}} |\mathcal{P}(\boldsymbol{a}) - \widehat{\mathcal{P}}(\boldsymbol{a})| \leq \varepsilon' \right) \geq 1 - \delta'$$

Before we proceed to wrap up the proof, let us collect the proven bounds below:

$$|J_{<\tau}| \leq \frac{\tau \cdot |\boldsymbol{\Sigma}_{\boldsymbol{A}}|}{\alpha_{\boldsymbol{A}}} \qquad \text{deterministically} \tag{7.10}$$

$$\Pr(\neg \mathcal{E}^{J}_{\geq \tau}) \leq |\boldsymbol{\Sigma}_{\boldsymbol{A}}| \cdot \exp\left(-\frac{n\tau}{12}\right) \tag{7.11}$$

$$\Pr\left(|J_{\geq \tau}| > t \mid \mathcal{E}^{J}_{\geq \tau}\right) \leq 2 \exp\left(-n \alpha_{\boldsymbol{A}} t^{2}\right) \qquad \text{for any } t > 0 \tag{7.12}$$

$$\Pr\left(|K| \leq \varepsilon'\right) \leq 1 - \delta' \qquad \text{for any } \varepsilon', \delta' > 0 \tag{7.13}$$

$$\text{when } N_{\mathrm{Poi}} \in \mathcal{O}\left(\frac{|\boldsymbol{\Sigma}| + \log \frac{1}{\delta'}}{(\varepsilon')^{2}}\right) \tag{7.14}$$

Now, observe that $|J_{<\tau}| \leq \frac{\varepsilon}{3}$, $|J_{\geq \tau}| \leq \frac{\varepsilon}{3}$ and $|K| \leq \frac{\varepsilon}{3}$ jointly implies $|J_{<\tau} + J_{\geq \tau} + K| \leq |J_{<\tau}| + |J_{\geq \tau}| + |K| \leq \varepsilon$ by triangle inequality. So,

$$\begin{aligned}
&\Pr\left(\left|T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}\right| > \varepsilon\right) \\
&= \Pr\left(|J_{<\tau} + J_{\geq \tau} + K| > \varepsilon\right) && \text{(By definition)} \\
&\leq \Pr\left(|J_{<\tau}| > \frac{\varepsilon}{3}\right) + \Pr\left(|J_{\geq \tau}| > \frac{\varepsilon}{3}\right) + \Pr\left(|K| > \frac{\varepsilon}{3}\right) && \text{(Triangle inequality)} \\
&\leq 0 + \Pr\left(|J_{\geq \tau}| > \frac{\varepsilon}{3}\right) + \Pr\left(|K| > \frac{\varepsilon}{3}\right) \\
& && \text{(If we set } \tfrac{\varepsilon}{3} = \tfrac{\tau \cdot |\boldsymbol{\Sigma}_{\boldsymbol{A}}|}{\alpha_{\boldsymbol{A}}} \text{ in the deterministic bound of Eq. (7.10))} \\
&\leq \Pr\left(|J_{\geq \tau}| > \frac{\varepsilon}{3}\right) + \frac{\delta}{3} \\
& && \text{(If we set } \varepsilon' = \tfrac{\varepsilon}{3} \text{ and } \delta' = \tfrac{\delta}{3} \text{ in Eq. (7.14) with } N_{\mathrm{Poi}} \in \mathcal{O}\left(\tfrac{|\boldsymbol{\Sigma}_{\boldsymbol{A}}| + \log \frac{1}{\delta'}}{(\varepsilon')^{2}}\right)\text{)} \\
&\leq \Pr(\neg \mathcal{E}^{J}_{\geq \tau}) + \Pr\left(|J_{\geq \tau}| > \frac{\varepsilon}{3} \mid \mathcal{E}^{J}_{\geq \tau}\right) + \frac{\delta}{3} && \text{(Conditioning on event } \mathcal{E}^{J}_{\geq \tau}\text{)} \\
&\leq |\boldsymbol{\Sigma}_{\boldsymbol{A}}| \cdot \exp\left(-\frac{n\tau}{12}\right) + 2 \exp\left(-n \alpha_{\boldsymbol{A}} t^{2}\right) + \frac{\delta}{3} \\
& && \text{(If we set } t = \tfrac{\varepsilon}{3} \text{ then apply Eq. (7.11) and Eq. (7.12))}
\end{aligned}$$

Recall that we set $\frac{\varepsilon}{3} = \frac{\tau \cdot |\boldsymbol{\Sigma}_{\boldsymbol{A}}|}{\alpha_{\boldsymbol{A}}} \iff \tau = \frac{\varepsilon \alpha_{\boldsymbol{A}}}{|3 \boldsymbol{\Sigma}_{\boldsymbol{A}}|}$ and $t = \frac{\varepsilon}{3}$ above. So, if we set

$$\begin{aligned}
n &= \frac{36 |\boldsymbol{\Sigma}_{\boldsymbol{A}}|}{\varepsilon \alpha_{\boldsymbol{A}}} \log\left(\frac{3 |\boldsymbol{\Sigma}_{\boldsymbol{A}}|}{\delta}\right) + \frac{9}{\varepsilon^{2} \alpha_{\boldsymbol{A}}} \log\left(\frac{6}{\delta}\right) + \mathcal{O}\left(\frac{|\boldsymbol{\Sigma}_{\boldsymbol{A}}| + \log \frac{1}{\delta}}{\varepsilon^{2}}\right) \\
&\in \widetilde{\mathcal{O}}\left(\left(\frac{|\boldsymbol{\Sigma}_{\boldsymbol{A}}|}{\varepsilon \alpha_{\boldsymbol{A}}} + \frac{1}{\varepsilon^{2} \alpha_{\boldsymbol{A}}} + \frac{|\boldsymbol{\Sigma}_{\boldsymbol{A}}|}{\varepsilon^{2}}\right) \cdot \log\left(\frac{1}{\delta}\right)\right)
\end{aligned}$$

then $\Pr\left(\left|T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}\right| > \varepsilon\right) \leq \frac{\delta}{3} + \frac{\delta}{3} + \frac{\delta}{3} = \delta$. □

7.5 Approximate Markov blankets

As discussed in Section 7.2, the bound in Theorem 7.1 motivates the use of small adjustment sets whenever possible. Since $T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} = T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ if $\boldsymbol{S}$ is a Markov blanket of $\boldsymbol{X}$ with respect to $\boldsymbol{A}$, a straightforward approach to finding a smaller adjustment set is to search for an approximate Markov blanket of $\boldsymbol{X}$. In this section, we study ε-Markov blankets of $\boldsymbol{X}$ with respect to $\boldsymbol{A}$.

In this section, we prove Lemma 7.7, which extends the equality $T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} = T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ for exact Markov blankets to a bound on the misspecification error $|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}|$ for approximate Markov blankets. To accompany this result, we give a matching worst case lower bound on the misspecification error in Lemma 7.8. Finally, we show Theorem 7.3, our PAC style sample complexity upper bound for finding an ε-Markov blanket of $\boldsymbol{X}$ with respect to an arbitrary set $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$, using the ApproximateMarkovBlanketAdjustment algorithm (Algorithm 15).

Lemma 7.7 (Misspecification error)**.** *If* $\boldsymbol{S} \subseteq \boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ *such that* $\boldsymbol{X} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{A} \setminus \boldsymbol{S} \mid \boldsymbol{S}$, *then* $|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}}$.

Proof. Since $\boldsymbol{S} \subseteq \boldsymbol{A}$, we see that

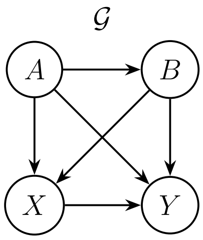
$$
\begin{aligned}
|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| &= \left| \sum_{\boldsymbol{a}} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{a} \setminus \boldsymbol{s} \mid \boldsymbol{s}, \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{s}) - \sum_{\boldsymbol{a}} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{a}) \right| \\
&\qquad \text{(By Eq. (7.5) and } \boldsymbol{S} \subseteq \boldsymbol{A}\text{)} \\
&= \left| \sum_{\boldsymbol{a}} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{s}) \cdot (\mathcal{P}(\boldsymbol{a} \setminus \boldsymbol{s} \mid \boldsymbol{s}, \boldsymbol{x}) - \mathcal{P}(\boldsymbol{a} \setminus \boldsymbol{s} \mid \boldsymbol{s})) \right| \\
&\qquad \text{(Pull out common terms)} \\
&= \left| \sum_{\boldsymbol{a}} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{x}) \cdot \frac{\mathcal{P}(\boldsymbol{s}, \boldsymbol{x})}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot (\mathcal{P}(\boldsymbol{a} \setminus \boldsymbol{s} \mid \boldsymbol{s}, \boldsymbol{x}) - \mathcal{P}(\boldsymbol{a} \setminus \boldsymbol{s} \mid \boldsymbol{s})) \right| \\
&\leq \sum_{\boldsymbol{a}} \frac{\mathcal{P}(\boldsymbol{s}, \boldsymbol{x})}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot |\mathcal{P}(\boldsymbol{a} \setminus \boldsymbol{s} \mid \boldsymbol{s}, \boldsymbol{x}) - \mathcal{P}(\boldsymbol{a} \setminus \boldsymbol{s} \mid \boldsymbol{s})| \\
&\qquad \text{(Triangle inequality, non-negativity of probabilities, and since } \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{a}, \boldsymbol{x}) \leq 1\text{)} \\
&\leq \frac{1}{\alpha_{\boldsymbol{S}}} \cdot \sum_{\boldsymbol{a}} \mathcal{P}(\boldsymbol{s}, \boldsymbol{x}) \cdot |\mathcal{P}(\boldsymbol{a} \setminus \boldsymbol{s} \mid \boldsymbol{s}, \boldsymbol{x}) - \mathcal{P}(\boldsymbol{a} \setminus \boldsymbol{s} \mid \boldsymbol{s})| \quad \text{(By Eq. (7.3))} \\
&\leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}} \qquad \text{(Since } \boldsymbol{X} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{A} \setminus \boldsymbol{S} \mid \boldsymbol{S} \text{ and using Eq. (7.4))}
\end{aligned}
$$

□

Lemma 7.8 (Misspecification error lower bound)**.** *Let* $0 \leq \sqrt{\varepsilon} \leq \alpha \leq 1/2$. *There exists* $\mathcal{P}(\boldsymbol{V})$ *such that (i)* $\boldsymbol{Z}$ *is a valid adjustment set, (ii)* $\boldsymbol{S} \subset \boldsymbol{Z}$ *satisfies* $\boldsymbol{X} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{Z} \setminus \boldsymbol{S} \mid \boldsymbol{S}$, *(iii)* $\alpha_{\boldsymbol{S}} \geq \alpha$, *and (iv)* $|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}| \geq \frac{\varepsilon}{16\alpha}$.

Proof. Consider the following probability distribution $\mathcal{P}$ defined over 4 binary variables $\{A, B, X, Y\}$ in a topological ordering of $A \prec B \prec X \prec Y$: see Fig. 7.2.

We show in Appendix B.2.4 that all the (conditional) probabilities of $\mathcal{P}$ are well-defined, and that we have the following conditional probabilities for $\mathcal{P}$:



$$A = \begin{cases} 1 & \text{w.p. } \frac{\varepsilon}{4\alpha} \cdot \frac{\alpha - \varepsilon/4}{1-\sqrt{\varepsilon}/2} \\ 0 & \text{else} \end{cases} \qquad X = \begin{cases} A & \text{w.p. } 1-\alpha \\ 1-A & \text{w.p. } \alpha - \sqrt{\varepsilon}/2 \\ B & \text{w.p. } \sqrt{\varepsilon}/2 \end{cases}$$

$$B = \begin{cases} 1-A & \text{w.p. } 1-\sqrt{\varepsilon} \\ 0 & \text{w.p. } \sqrt{\varepsilon}/2 \\ 1 & \text{w.p. } \sqrt{\varepsilon}/2 \end{cases} \qquad Y = \begin{cases} 1 & \text{if } X=0, A=1, B=0 \\ 0 & \text{else} \end{cases}$$

Figure 7.2: Probability distribution $\mathcal{P}$ defined over 4 binary variables $\{A, B, X, Y\}$ in a topological ordering of $A \prec B \prec X \prec Y$ with parameters ε and α, where $0 < \sqrt{\varepsilon} \leq \alpha \leq 1/2$.

a	b	$\mathcal{P}(b \mid a)$	$\mathcal{P}(X=0 \mid a,b)$	$\mathcal{P}(X=0 \mid a)$	$\sum_x \lvert\mathcal{P}(x \mid a,b) - \mathcal{P}(x \mid a)\rvert$
0	0	$\sqrt{\varepsilon}/2$	$1-\alpha+\sqrt{\varepsilon}/2$	$1-\alpha+\varepsilon/4$	$\sqrt{\varepsilon}-\varepsilon/2$
0	1	$1-\sqrt{\varepsilon}/2$	$1-\alpha$	$1-\alpha+\varepsilon/4$	$\varepsilon/2$
1	0	$1-\sqrt{\varepsilon}/2$	α	$\alpha-\varepsilon/4$	$\varepsilon/2$
1	1	$\sqrt{\varepsilon}/2$	$\alpha-\sqrt{\varepsilon}/2$	$\alpha-\varepsilon/4$	$\sqrt{\varepsilon}-\varepsilon/2$

Let us identify $\boldsymbol{Z}$ with $\{A, B\}$ and $\boldsymbol{S}$ with $\{A\}$, so $\boldsymbol{Z} \setminus \boldsymbol{S} = \{B\}$. We now show the four properties.

1. $\boldsymbol{Z}$ is a valid adjustment set

   This is true since $\{A, B\}$ satifies the backdoor adjustment criterion [Pea95].

2. $\boldsymbol{S} \subset \boldsymbol{Z}$ satisfies $\boldsymbol{X} \perp\!\!\!\perp_\varepsilon \boldsymbol{Z} \setminus \boldsymbol{S} \mid \boldsymbol{S}$

   Recall that $\boldsymbol{Z} = \{A, B\}$ and $\boldsymbol{S} = \{A\}$. To see that $X \perp\!\!\!\perp_\varepsilon B \mid A$, observe the following:

$$\begin{aligned}
&\sum_{x,a,b} \mathcal{P}(a) \cdot \lvert \mathcal{P}(x, b \mid a) - \mathcal{P}(x \mid a) \cdot \mathcal{P}(b \mid a) \rvert \\
&= \sum_{a,b} \mathcal{P}(a) \cdot \mathcal{P}(b \mid a) \cdot \sum_{x} \lvert \mathcal{P}(x \mid a, b) - \mathcal{P}(x \mid a) \rvert \\
&= \mathcal{P}(A=0) \cdot \mathcal{P}(B=0 \mid A=0) \cdot (\sqrt{\varepsilon} - \varepsilon/2) \\
&\quad + \mathcal{P}(A=0) \cdot \mathcal{P}(B=1 \mid A=0) \cdot (\varepsilon/2) \\
&\quad + \mathcal{P}(A=1) \cdot \mathcal{P}(B=0 \mid A=1) \cdot (\varepsilon/2) \\
&\quad + \mathcal{P}(A=1) \cdot \mathcal{P}(B=1 \mid A=1) \cdot (\sqrt{\varepsilon} - \varepsilon/2) \\
&= \mathcal{P}(A=0) \cdot (\sqrt{\varepsilon}/2) \cdot (\sqrt{\varepsilon} - \varepsilon/2) + \mathcal{P}(A=0) \cdot (1 - \sqrt{\varepsilon}/2) \cdot (\varepsilon/2) \\
&\quad + \mathcal{P}(A=1) \cdot (1 - \sqrt{\varepsilon}/2) \cdot (\varepsilon/2) + \mathcal{P}(A=1) \cdot (\sqrt{\varepsilon}/2) \cdot (\sqrt{\varepsilon} - \varepsilon/2) \\
&= (\sqrt{\varepsilon}/2) \cdot (\sqrt{\varepsilon} - \varepsilon/2) + (1 - \sqrt{\varepsilon}/2) \cdot \varepsilon/2 \\
&= \varepsilon
\end{aligned}$$

3. $\alpha_{\boldsymbol{S}} \geq \alpha$

   Since $\boldsymbol{S} = \{A\}$ and $\varepsilon \leq \alpha/2$, we have $\min_a \mathcal{P}(x \mid a) = \alpha - \varepsilon/4 \geq \alpha/2$.

4. $|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}| \geq \frac{\varepsilon}{16\alpha}$

$$
\begin{aligned}
&|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}| \\
&= \left| \sum_{a} \mathcal{P}(a) \cdot \mathcal{P}(y \mid x, a) - \sum_{a,b} \mathcal{P}(a,b) \cdot \mathcal{P}(y \mid x, a, b) \right| \\
&\qquad \text{(Since } \boldsymbol{S} = \{A\}, \boldsymbol{Z} = \{A, B\}, \text{ and by definition of } T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} \text{ and } T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}\text{)} \\
&= \left| \sum_{a,b} \mathcal{P}(a) \cdot \mathcal{P}(y \mid x, a, b) \cdot (\mathcal{P}(b \mid a) - \mathcal{P}(b \mid x, a)) \right| \\
&\qquad \text{(Since } \mathcal{P}(y \mid x, a) = \textstyle\sum_b \mathcal{P}(y \mid x, a, b) \cdot \mathcal{P}(b \mid x, a) \text{ and } \mathcal{P}(a, b) = \mathcal{P}(a) \cdot \mathcal{P}(b \mid a)\text{)} \\
&= \left| \sum_{a,b} \mathcal{P}(a) \cdot \mathcal{P}(y \mid x, a, b) \cdot \frac{\mathcal{P}(b \mid a)}{\mathcal{P}(x \mid a)} \cdot (\mathcal{P}(x \mid a) - \mathcal{P}(x \mid a, b)) \right| \\
&\qquad \text{(Since } \mathcal{P}(b \mid x, a) = \tfrac{\mathcal{P}(b|a) \cdot \mathcal{P}(x|a,b)}{\mathcal{P}(x|a)}\text{)} \\
&= \mathcal{P}(A = 1) \cdot \frac{\mathcal{P}(B = 0 \mid A = 1)}{\mathcal{P}(X = 0 \mid A = 1)} \\
&\qquad \cdot \left| \mathcal{P}(X = 0 \mid A = 1) - \mathcal{P}(X = 0 \mid A = 1, B = 0) \right| \\
&\qquad \text{(Since } Y \text{ is an indicator variable for whether } (A, B, X) = (1, 0, 0)\text{)} \\
&= \frac{\varepsilon}{4\alpha} \cdot \frac{\alpha - \varepsilon/4}{1 - \sqrt{\varepsilon}/2} \cdot \frac{1 - \sqrt{\varepsilon}/2}{\alpha - \varepsilon/4} \cdot \frac{\varepsilon}{4} \qquad \text{(From construction in Fig. 7.2)} \\
&= \frac{\varepsilon}{16\alpha}
\end{aligned}
$$

□

Algorithm 15 ApproximateMarkovBlanketAdjustment (AMBA)

```
Input: $\varepsilon, \delta > 0$, dataset $\boldsymbol{D}$ of $n$ i.i.d. samples from $\mathcal{P}(\boldsymbol{V})$, and subset $\boldsymbol{A} \subseteq \boldsymbol{V}$
Output: $\boldsymbol{S} \subseteq \boldsymbol{A}$
for $k = 0, 1, 2, \ldots, |\boldsymbol{A}|$ do
    Let $w_k = \left(|\boldsymbol{A}| \cdot \binom{|\boldsymbol{A}|}{k}\right)^{-1}$
    Let $\boldsymbol{C}_k = \Big\{\boldsymbol{S} \subseteq \boldsymbol{A} : |\boldsymbol{S}| = k$, where
            ApproxCondInd$(\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{A} \setminus \boldsymbol{S} \mid \boldsymbol{S}, \varepsilon, \delta w_k, \boldsymbol{D})$ outputs YES $\Big\}$
    if $|\boldsymbol{C}_k| > 0$ then
        return any $\boldsymbol{S} \in \boldsymbol{C}_k$
return $\boldsymbol{A}$
```

Theorem 7.3 (Approximate Markov blanket discovery)**.** *Suppose we are given (1)* $\varepsilon > 0$*, (2)* $\delta > 0$*, (3) sample access to a distribution* $\mathcal{P}(\boldsymbol{V})$*, and (4) an arbitrary subset* $\boldsymbol{A} \subseteq$

$\boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$. *Suppose that there is a Markov blanket of* $\boldsymbol{X}$ *with respect to* $\boldsymbol{A}$ *with* k *variables. Then, there is an algorithm that uses* $\widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{S}|}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{A}}|} \cdot \log \frac{1}{\delta}\right)$ *samples and produces a subset* $\boldsymbol{S} \subseteq \boldsymbol{A}$ *such that* $|\boldsymbol{S}| \leq k$*,* $\Pr\left(\Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{A} \setminus \boldsymbol{S} | \boldsymbol{S}} > \varepsilon\right) \geq 1 - \delta$*, and* $\Pr\left(|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}}\right) \geq 1 - \delta$.

Proof. Suppose the Amba algorithm (Algorithm 15) terminates at some iteration $|\boldsymbol{S}| \in \{0, 1, \ldots, |\boldsymbol{A}|\}$.

Correctness. Suppose all calls to ApproxCondInd succeed, then Lemma 2.41 tells us that any produced $\boldsymbol{S} \subseteq \boldsymbol{A}$ satisfies the property that $\Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{A} \setminus \boldsymbol{S} | \boldsymbol{S}} \leq \varepsilon$. Lemma 7.7 then further tells us that $\Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{A} \setminus \boldsymbol{S} | \boldsymbol{S}} \leq \varepsilon$ implies $|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}}$.

Failure rate. Note that there are at most $\binom{|\boldsymbol{A}|}{k}$ possible candidate sets in $\boldsymbol{C}_k$ for each $k \in \{0, 1, \ldots, |\boldsymbol{A}|\}$. Since we invoked each call to ApproxCondInd with δw_k in iteration k, union bound tells us that the probability of *any* call failing across all calls is at most

$$\sum_{k=0}^{|\boldsymbol{S}|} \delta w_k \cdot \binom{|\boldsymbol{A}|}{k} = \sum_{k=0}^{|\boldsymbol{S}|} \delta \cdot \frac{1}{|\boldsymbol{A}| \cdot \binom{|\boldsymbol{A}|}{k}} \cdot \binom{|\boldsymbol{A}|}{k} = \sum_{k=0}^{|\boldsymbol{S}|} \frac{\delta}{|\boldsymbol{A}|} \leq \frac{\delta \cdot |\boldsymbol{S}|}{|\boldsymbol{A}|} \leq \delta$$

Sample complexity. Since we are using union bound to bound our overall failure probability, we can reuse samples in all our calls to ApproxCondInd. Thus, the total sample complexity is attributed to the final call when $k = |\boldsymbol{S}|$. Such an invocation of ApproxCondInd uses $\widetilde{\mathcal{O}}\left(\frac{1}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{A} \setminus \boldsymbol{S}}| \cdot |\Sigma_{\boldsymbol{S}}|} \cdot \log \frac{1}{\delta w_k}\right)$ samples according to Lemma 2.41 and $w_k = \left(|\boldsymbol{A}| \cdot \binom{|\boldsymbol{A}|}{k}\right)^{-1}$, so the total number of samples used is at most

$$\widetilde{\mathcal{O}}\left(\frac{1}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{A} \setminus \boldsymbol{S}}| \cdot |\Sigma_{\boldsymbol{S}}|} \cdot \log \frac{1}{\delta w_k}\right) \subseteq \widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{S}|}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{A}}|} \cdot \log \frac{1}{\delta}\right)$$

We omit $\log |\boldsymbol{A}|$ within $\widetilde{\mathcal{O}}(\cdot)$ because $|\boldsymbol{A}| \leq |\Sigma_{\boldsymbol{A}}|$. □

We can combine Amba (Algorithm 15) with the estimation procedure of Theorem 7.1 to yield an overall PAC bound guarantee for searching for a small adjustment set $\boldsymbol{S}$ when given a valid adjustment set $\boldsymbol{Z} \subseteq \boldsymbol{V}$ to begin with; see Section 7.7.

7.6 Beyond approximate Markov blankets

Motivated by Fig. 7.1, which shows that the Markov blanket of $\boldsymbol{X}$ with respect to $\boldsymbol{Z}$ may still be large compared to the smallest adjustment set, we study in this section an approach for finding smaller adjustment sets than the Markov blanket.

We begin by proving Lemma 7.9, which establishes conditions on sets $\boldsymbol{S}' \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ and $\boldsymbol{S} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ such that $T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} = T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$; this result suggest an approach for going beyond adjustment by Markov blankets. Then, we prove Theorem 7.5, our upper bound on the sample complexity of finding a set $\boldsymbol{S}'$ that approximately satisfies the conditions of Lemma 7.9 with respect to an ε-Markov blanket $\boldsymbol{S}$.

Lemma 7.9 (Adjustment soundness)**.** *Let $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ be an arbitrary subset. If $\boldsymbol{S} \subseteq \boldsymbol{A}$ and $\boldsymbol{S}' \subseteq \boldsymbol{A}$ such that $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'$ and $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}$, then $T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} = T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$.*

Proof. Consider arbitrary subsets $\boldsymbol{S} \subseteq \boldsymbol{A} \subseteq \boldsymbol{V}$ and $\boldsymbol{S}' \subseteq \boldsymbol{A} \subseteq \boldsymbol{V}$. Observe that

$$\begin{aligned}
T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} &= \sum_{\boldsymbol{s},\boldsymbol{s}'\setminus\boldsymbol{s}} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}, \boldsymbol{s}' \setminus \boldsymbol{s}) \cdot \mathcal{P}(\boldsymbol{s}' \setminus \boldsymbol{s} \mid \boldsymbol{x}, \boldsymbol{s}) \cdot \mathcal{P}(\boldsymbol{s}) && \text{(By Eq. (7.5))}\\
&= \sum_{\boldsymbol{s},\boldsymbol{s}'\setminus\boldsymbol{s}} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}, \boldsymbol{s}' \setminus \boldsymbol{s}) \cdot \mathcal{P}(\boldsymbol{s}' \setminus \boldsymbol{s} \mid \boldsymbol{s}) \cdot \mathcal{P}(\boldsymbol{s}) && \text{(Since } \boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}\text{)}\\
&= \sum_{\boldsymbol{s}',\boldsymbol{s}\setminus\boldsymbol{s}'} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}', \boldsymbol{s} \setminus \boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{s}' \setminus \boldsymbol{s} \mid \boldsymbol{s}) \cdot \mathcal{P}(\boldsymbol{s}) && \text{(Regrouping)}\\
&= \sum_{\boldsymbol{s}',\boldsymbol{s}\setminus\boldsymbol{s}'} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{s}' \setminus \boldsymbol{s} \mid \boldsymbol{s}) \cdot \mathcal{P}(\boldsymbol{s}) && \text{(Since } \boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'\text{)}\\
&= T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} && \text{(By Eq. (7.5))}
\end{aligned}$$

□

Algorithm 16 BeyondApproximateMarkovBlanketAdjustment (Bamba)

```
Input: $\varepsilon, \delta > 0$, dataset $\boldsymbol{D}$ of $n$ i.i.d. samples from $\mathcal{P}(\boldsymbol{V})$, subset $\boldsymbol{A} \subseteq \boldsymbol{V}$, and $\varepsilon$-Markov blanket $\boldsymbol{S} \subseteq \boldsymbol{A}$
Output: $\boldsymbol{S}' \subseteq \boldsymbol{A}$ such that $|\Sigma_{\boldsymbol{S}'}| \leq |\Sigma_{\boldsymbol{S}}|$
for $k = 0, 1, 2, \ldots, |\boldsymbol{A}|$ do
    Let $w_k = \left(|\boldsymbol{A}| \cdot \binom{|\boldsymbol{A}|}{k}\right)^{-1}$
    Let $\boldsymbol{C}_k = \Big\{ \boldsymbol{S}' \subseteq \boldsymbol{A} : |\boldsymbol{S}'| = k$, where
            ApproxCondInd($\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}', \varepsilon, \frac{\delta w_k}{2}, \boldsymbol{D}$) outputs YES,
            ApproxCondInd($\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}, \varepsilon, \frac{\delta w_k}{2}, \boldsymbol{D}$) outputs YES,
        and $|\Sigma_{\boldsymbol{S}'}| \leq |\Sigma_{\boldsymbol{S}}| \Big\}$
    if $|\boldsymbol{C}_k| > 0$ then
        return any $\boldsymbol{S}' \in \boldsymbol{C}_k$
return $\boldsymbol{S}$
```

Theorem 7.5 (Beyond approximate Markov blankets)**.** *Suppose we are given (1) $\varepsilon > 0$, (2) $\delta > 0$, (3) sample access to $\mathcal{P}(\boldsymbol{V})$, (4) an arbitrary subset $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$, and (5) an ε-Markov blanket $\boldsymbol{S} \subseteq \boldsymbol{A}$. Suppose there is a screening set $\boldsymbol{B}$ for*

$(\boldsymbol{S}, \boldsymbol{A}, \boldsymbol{X}, \boldsymbol{Y})$ such that $|\boldsymbol{B}| = k'$ and $|\Sigma_{\boldsymbol{B}}| \leq |\Sigma_{\boldsymbol{S}}|$. There is an algorithm that uses $\widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{S}'|}{\varepsilon^2} \cdot \sqrt{|\boldsymbol{\Sigma_X}| \cdot |\boldsymbol{\Sigma_Y}| \cdot |\boldsymbol{\Sigma_A}|} \cdot \log \frac{1}{\delta}\right)$ samples and produces a subset $\boldsymbol{S}' \subseteq \boldsymbol{A}$ such that $|\boldsymbol{S}'| \leq k'$, $|\boldsymbol{\Sigma_{S'}}| \leq |\boldsymbol{\Sigma_S}|$ and $\Pr\left(|T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{2\varepsilon}{\alpha_{\boldsymbol{S}}}\right) \geq 1 - \delta$.

Proof. Suppose the Bamba algorithm (Algorithm 16) terminates at some iteration $|\boldsymbol{S}'| \in \{0, 1, \ldots, |\boldsymbol{A}|\}$.

Correctness. If Bamba returns the ε-Markov blanket $\boldsymbol{S} \subseteq \boldsymbol{A}$ (e.g. in Line 8), then $|T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| = |T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}} \leq \frac{2\varepsilon}{\alpha_{\boldsymbol{S}}}$ by Definition 7.2 and Lemma 7.7. Suppose all calls to ApproxCondInd succeed across all iterations. Then, Lemma 2.41 tells us that $\Delta_{\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' | \boldsymbol{X} \cup \boldsymbol{S}'} \leq \varepsilon$, $\Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} | \boldsymbol{S}} \leq \varepsilon$, and $|\boldsymbol{\Sigma_{S'}}| \leq |\boldsymbol{\Sigma_S}|$ whenever $\boldsymbol{C}_k \neq \emptyset$.

For subsequent analytical purposes, let us define an intermediate term $Z_{\boldsymbol{x},\boldsymbol{y}}$ as follows:

$$Z_{\boldsymbol{x},\boldsymbol{y}} = \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \frac{1}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}') \tag{7.15}$$

By triangle inequality, we have

$$|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}| = |T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - Z_{\boldsymbol{x},\boldsymbol{y}} + Z_{\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}| \leq |T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - Z_{\boldsymbol{x},\boldsymbol{y}}| + |Z_{\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}|$$

We will bound each of these terms separately.

1. Bounding $|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - Z_{\boldsymbol{x},\boldsymbol{y}}|$.

$$\begin{aligned}
|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - Z_{\boldsymbol{x},\boldsymbol{y}}| &= \Bigg| \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{s}' \setminus \boldsymbol{s} \mid \boldsymbol{x}, \boldsymbol{s}) \cdot \mathcal{P}(\boldsymbol{s}) \\
&\qquad - \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \frac{1}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}') \Bigg| \\
&\qquad\qquad \text{(By Eq. (7.5) and Eq. (7.15))} \\
&= \Bigg| \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot \frac{\mathcal{P}(\boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}')}{\mathcal{P}(\boldsymbol{x}, \boldsymbol{s})} \cdot \mathcal{P}(\boldsymbol{s}) \\
&\qquad - \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \frac{1}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}') \Bigg| \\
&= \left| \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \frac{1}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot (\mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') - \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}')) \right| \\
&\qquad\qquad \text{(Pull out common terms)} \\
&\leq \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \frac{1}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot |\mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') - \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}')| \\
&\qquad\qquad \text{(Triangle inequality and non-negative of probabilities)}
\end{aligned}$$

$$\leq \frac{1}{\alpha_{\boldsymbol{S}}} \cdot \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \mathcal{P}(\boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot |\mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') - \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}')|$$

(By definition of $\alpha_{\boldsymbol{S}}$ in Eq. (7.3))

$$\leq \frac{1}{\alpha_{\boldsymbol{S}}} \cdot \sum_{\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}'} \mathcal{P}(\boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot |\mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') - \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}')|$$

(Summing over more terms)

$$\leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}}$$

(Since $\Delta_{\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'} \leq \varepsilon$ and using Eq. (7.4))

2. Bounding $|Z_{\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}|$.

$$\begin{aligned}
&|T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} - Z_{\boldsymbol{x},\boldsymbol{y}}| \\
&= \Bigg| \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{s} \setminus \boldsymbol{s}' \mid \boldsymbol{s}') \\
&\qquad - \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \frac{1}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot \mathcal{P}(\boldsymbol{x}, \boldsymbol{s} \cup \boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}') \Bigg| && \text{(By Eq. (7.5) and Eq. (7.15))} \\
&= \left| \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \frac{1}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}') \cdot \mathcal{P}(\boldsymbol{s} \cup \boldsymbol{s}') \cdot (\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s}) - \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s} \cup \boldsymbol{s}')) \right| \\
& && \text{(Pull out common terms)} \\
&\leq \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \frac{1}{\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s})} \cdot \mathcal{P}(\boldsymbol{s} \cup \boldsymbol{s}') \cdot |\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s}) - \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s} \cup \boldsymbol{s}')| \\
& && \text{(Triangle inequality, non-negativity of probabilities, and since } \mathcal{P}(\boldsymbol{y} \mid \boldsymbol{x}, \boldsymbol{s}') \leq 1\text{)} \\
&\leq \frac{1}{\alpha_{\boldsymbol{S}}} \cdot \sum_{\boldsymbol{s} \cup \boldsymbol{s}'} \mathcal{P}(\boldsymbol{s} \cup \boldsymbol{s}') \cdot |\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s}) - \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{s} \cup \boldsymbol{s}')| && \text{(By definition of } \alpha_{\boldsymbol{S}} \text{ in Eq. (7.3))} \\
&\leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}} && \text{(Since } \Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}} \leq \varepsilon \text{ and using Eq. (7.4))}
\end{aligned}$$

Putting together.

We see that

$$|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}| \leq |T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - Z_{\boldsymbol{x},\boldsymbol{y}}| + |Z_{\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}} + \frac{\varepsilon}{\alpha_{\boldsymbol{S}}} = \frac{2\varepsilon}{\alpha_{\boldsymbol{S}}}$$

Failure rate. Note that there are at most $\binom{|\boldsymbol{A}|}{k}$ possible candidate sets in $\boldsymbol{C}_k$ for each $k \in \{0, 1, \ldots, |\boldsymbol{A}|\}$. Since we invoked two calls to ApproxCondInd in iteration k, each with failure parameter $\delta w_k / 2$, union bound tells us that the probability of *any* call failing across all calls is at most

$$\sum_{k=0}^{|\boldsymbol{S}'|} 2 \cdot \frac{\delta w_k}{2} \cdot \binom{|\boldsymbol{A}|}{k} = \sum_{k=0}^{|\boldsymbol{S}'|} \delta \cdot \frac{1}{|\boldsymbol{A}| \cdot \binom{|\boldsymbol{A}|}{k}} \cdot \binom{|\boldsymbol{A}|}{k} = \sum_{k=0}^{|\boldsymbol{S}'|} \frac{\delta}{|\boldsymbol{A}|} \leq \frac{\delta \cdot |\boldsymbol{S}|}{|\boldsymbol{A}|} \leq \delta$$

Sample complexity. Since we are using union bound to bound our overall failure probability, we can reuse samples in all our calls to ApproxCondInd. Thus, the total sample complexity is attributed to the final call when $k = |\boldsymbol{S}'|$. Such an invocation of ApproxCondInd uses $\widetilde{\mathcal{O}}\left(\frac{1}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{A} \setminus \boldsymbol{S}'}| \cdot |\Sigma_{\boldsymbol{S}'}|} \cdot \log \frac{1}{\delta w_k}\right)$ samples according to Lemma 2.41 and $w_k = \left(|\boldsymbol{A}| \cdot \binom{|\boldsymbol{A}|}{k}\right)^{-1}$, so the total number of samples used is at most

$$\begin{aligned}
&\widetilde{\mathcal{O}}\left(\frac{1}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{A} \setminus \boldsymbol{S}'}| \cdot |\Sigma_{\boldsymbol{S}'}|} \cdot \log \frac{1}{\delta w_k}\right) \\
&\qquad \subseteq \widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{S}'|}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{A}}|} \cdot \log \frac{1}{\delta}\right)
\end{aligned}$$

We omit $\log |\boldsymbol{A}|$ within $\widetilde{\mathcal{O}}(\cdot)$ because $|\boldsymbol{A}| \leq |\Sigma_{\boldsymbol{A}}|$. □

7.7 Estimating causal effects using AMBA and BAMBA

By re-expressing Theorem 7.1, Theorem 7.3 and Theorem 7.5 in terms of an upper bound on error for a fixed number of samples n, we get the following three corollaries.

Corollary 7.11 (Estimation corollary). *Suppose we are given (1) failure tolerance $\delta > 0$, (2) n i.i.d. samples from distribution $\mathcal{P}(\boldsymbol{V})$, (3) a subset $\boldsymbol{A} \subseteq \boldsymbol{V}$ with $\alpha_{\boldsymbol{A}} = \max_{\boldsymbol{a} \in \Sigma_{\boldsymbol{A}}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{a})$. Then, there is an algorithm that produces an estimate $\widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ such that $\Pr(|\widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \varepsilon) \geq 1 - \delta$ for some error term*

$$\varepsilon \in \widetilde{\mathcal{O}}\left(\frac{|\Sigma_{\boldsymbol{A}}|}{n\alpha_{\boldsymbol{A}}} + \frac{1}{\sqrt{n\alpha_{\boldsymbol{A}}}} + \sqrt{\frac{|\Sigma_{\boldsymbol{A}}|}{n}}\right)$$

Proof. From Theorem 7.1, we know that $\widetilde{\mathcal{O}}\left(\left(\frac{|\Sigma_{\boldsymbol{A}}|}{\varepsilon\alpha_{\boldsymbol{A}}} + \frac{1}{\varepsilon^2\alpha_{\boldsymbol{A}}} + \frac{|\Sigma_{\boldsymbol{A}}|}{\varepsilon^2}\right) \cdot \log\left(\frac{1}{\delta}\right)\right)$ samples suffice to produce an estimate $\widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}$ such that $\Pr(|\widehat{T}_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \varepsilon) \geq 1 - \delta$. Ignoring the logarithmic terms and constant factors, the result follows by re-expressing $n \leq \frac{|\Sigma_{\boldsymbol{A}}|}{\varepsilon\alpha_{\boldsymbol{A}}} + \frac{1}{\varepsilon^2\alpha_{\boldsymbol{A}}} + \frac{|\Sigma_{\boldsymbol{A}}|}{\varepsilon^2}$ in terms of ε. □

Corollary 7.12 (Amba corollary). *Suppose we are given (1) failure tolerance $\delta > 0$, (2) n i.i.d. samples from distribution $\mathcal{P}(\boldsymbol{V})$, and (3) an arbitrary subset $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$. Then, there is an algorithm that produces a subset $\boldsymbol{S} \subseteq \boldsymbol{A}$ such that $\Pr\left(\Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{A} \setminus \boldsymbol{S} | \boldsymbol{S}} > \varepsilon\right) \geq 1 - \delta$ and $\Pr\left(|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \varepsilon\right) \geq 1 - \delta$ for some error term*

$$\varepsilon \in \widetilde{\mathcal{O}}\left(\frac{1}{\alpha_{\boldsymbol{S}}} \cdot \sqrt{\frac{|\boldsymbol{S}|}{n}} \cdot (|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{A}}|)^{\frac{1}{4}}\right)$$

Proof. From Theorem 7.3, we know that $\widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{S}|}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{A}}|} \cdot \log \frac{1}{\delta}\right)$ samples suffice to produce a subset $\boldsymbol{S} \subseteq \boldsymbol{A}$ such that

- $\Pr\left(\Delta_{\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{A} \setminus \boldsymbol{S} \mid \boldsymbol{S}} > \varepsilon\right) \geq 1 - \delta$
- $\Pr\left(|T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}}\right) \geq 1 - \delta$

Ignoring the logarithmic terms and constant factors, the result follows by re-expressing $n = \frac{|\boldsymbol{S}|}{(\varepsilon')^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{A}}|}$ in terms of $\varepsilon' = \varepsilon \alpha_{\boldsymbol{S}} \leq \varepsilon$. □

Corollary 7.13 (BAMBA corollary). *Suppose we are given (1) failure tolerance $\delta > 0$, (2) n i.i.d. samples from distribution $\mathcal{P}(\boldsymbol{V})$, (3) an arbitrary subset $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$, and (4) an ε-Markov blanket $\boldsymbol{S} \subseteq \boldsymbol{A}$. Then, there is an algorithm that produces a subset $\boldsymbol{S}' \subseteq \boldsymbol{A}$ such that $|\Sigma_{\boldsymbol{S}'}| \leq |\Sigma_{\boldsymbol{S}}|$ and $\Pr\left(|T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \varepsilon\right) \geq 1 - \delta$ for some error term*

$$\varepsilon \in \widetilde{\mathcal{O}}\left(\frac{1}{\alpha_{\boldsymbol{S}}} \cdot \sqrt{\frac{|\boldsymbol{S}'|}{n}} \cdot \left(|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{A}}|\right)^{\frac{1}{4}}\right)$$

Proof. From Theorem 7.5, we know that $\widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{S}'|}{\varepsilon^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{A}}|} \cdot \log \frac{1}{\delta}\right)$ samples suffice to produce a subset $\boldsymbol{S}' \subseteq \boldsymbol{A}$ such that

- $|\Sigma_{\boldsymbol{S}'}| \leq |\Sigma_{\boldsymbol{S}}|$
- $\Pr\left(|T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{A},\boldsymbol{x},\boldsymbol{y}}| \leq \frac{\varepsilon}{\alpha_{\boldsymbol{S}}}\right) \geq 1 - \delta$

Ignoring the logarithmic terms and constant factors, the result follows by re-expressing $n = \frac{|\boldsymbol{S}'|}{(\varepsilon')^2} \cdot \sqrt{|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{A}}|}$ in terms of $\varepsilon' = \varepsilon \alpha_{\boldsymbol{S}} \leq \varepsilon$. □

In light of Corollary 7.11, Corollary 7.12, and Corollary 7.13, there are a couple of ways one could attempt to estimate $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ when given a valid adjustment set $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$:

1. Directly estimate using $\boldsymbol{Z}$. By Corollary 7.11, this yields an error of

$$|\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) - \widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y})| = |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}| \in \widetilde{\mathcal{O}}\left(\frac{|\Sigma_{\boldsymbol{Z}}|}{n\alpha_{\boldsymbol{Z}}} + \frac{1}{\sqrt{n\alpha_{\boldsymbol{Z}}}} + \sqrt{\frac{|\Sigma_{\boldsymbol{Z}}|}{n}}\right)$$

2. Use AMBA on $\boldsymbol{Z}$ to produce a subset $\boldsymbol{S} \subseteq \boldsymbol{Z}$ and estimate using $\boldsymbol{S}$. By Corollary 7.11 and Corollary 7.12, this yields an error of

$$\begin{aligned} |\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) - \widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y})| &= |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}| \leq |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}| + |T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}| \\ &\in \widetilde{\mathcal{O}}\left(\frac{1}{\alpha_{\boldsymbol{S}}} \cdot \sqrt{\frac{|\boldsymbol{S}|}{n}} \cdot \left(|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Z}}|\right)^{\frac{1}{4}} + \frac{|\Sigma_{\boldsymbol{S}}|}{n\alpha_{\boldsymbol{S}}} + \frac{1}{\sqrt{n\alpha_{\boldsymbol{S}}}} + \sqrt{\frac{|\Sigma_{\boldsymbol{S}}|}{n}}\right) \end{aligned}$$

3. Use AMBA on $\boldsymbol{Z}$ to produce a subset $\boldsymbol{S} \subseteq \boldsymbol{Z}$, then use BAMBA to further produce subset $\boldsymbol{S}'$, and then estimate using $\boldsymbol{S}'$. By Corollary 7.11, Corollary 7.12, and Corollary 7.13, this yields an error of

$$\begin{aligned}|\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})-\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y})| &= |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}-\widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}| \leq |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}-T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}| + |T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}-\widehat{T}_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}| \\ &\in \widetilde{\mathcal{O}}\Bigg(\frac{1}{\alpha_{\boldsymbol{S}}}\cdot\sqrt{\frac{|\boldsymbol{S}|}{n}}\cdot(|\Sigma_{\boldsymbol{X}}|\cdot|\Sigma_{\boldsymbol{Z}}|)^{\frac{1}{4}} + \frac{1}{\alpha_{\boldsymbol{S}}}\cdot\sqrt{\frac{|\boldsymbol{S}'|}{n}}\cdot(|\Sigma_{\boldsymbol{X}}|\cdot|\Sigma_{\boldsymbol{Y}}|\cdot|\Sigma_{\boldsymbol{Z}}|)^{\frac{1}{4}} \\ &\qquad + \frac{|\Sigma_{\boldsymbol{S}'}|}{n\alpha_{\boldsymbol{S}'}} + \frac{1}{\sqrt{n\alpha_{\boldsymbol{S}'}}} + \sqrt{\frac{|\Sigma_{\boldsymbol{S}'}|}{n}}\Bigg)\end{aligned}$$

In both cases 2 and 3, with appropriate constant factors, we see that

$$|\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})-\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y})| = |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}-\widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}| \leq |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}-T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}| + |T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}-\widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}| \leq \varepsilon+\varepsilon = 2\varepsilon$$

The following lemma tells us that $\alpha_{\boldsymbol{Z}} \leq \alpha_{\boldsymbol{S}}$ and $\alpha_{\boldsymbol{Z}} \leq \alpha_{\boldsymbol{S}'}$, i.e. $\frac{1}{\alpha_{\boldsymbol{S}}} \leq \frac{1}{\alpha_{\boldsymbol{Z}}}$ and $\frac{1}{\alpha_{\boldsymbol{S}'}} \leq \frac{1}{\alpha_{\boldsymbol{Z}}}$, so it is always beneficial to use a smaller subset with respect to the error incurred by estimation in Corollary 7.11.

Lemma 7.14. *For any value $\boldsymbol{x}$ for $\boldsymbol{X}$ and subsets $\boldsymbol{A} \subseteq \boldsymbol{B} \subseteq \boldsymbol{V} \setminus \boldsymbol{X}$, we have*

$$\alpha_{\boldsymbol{A}} = \min_{\boldsymbol{a}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{a}) \geq \min_{\boldsymbol{b}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{b}) = \alpha_{\boldsymbol{B}}$$

Proof. Fix an arbitrary values of $\boldsymbol{x}$ for $\boldsymbol{X}$ and $\boldsymbol{a}$ for $\boldsymbol{A}$, we see that

$$\mathcal{P}(\boldsymbol{x} \mid \boldsymbol{a}) = \sum_{\boldsymbol{b}\setminus\boldsymbol{a}} \mathcal{P}(\boldsymbol{x}, \boldsymbol{b}\setminus\boldsymbol{a} \mid \boldsymbol{a}) \geq \min_{\boldsymbol{b}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{b}) \cdot \sum_{\boldsymbol{b}\setminus\boldsymbol{a}} \mathcal{P}(\boldsymbol{b}\setminus\boldsymbol{a} \mid \boldsymbol{a}) = \min_{\boldsymbol{b}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{b})$$

Therefore, $\min_{\boldsymbol{a}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{a}) \geq \min_{\boldsymbol{b}} \mathcal{P}(\boldsymbol{x} \mid \boldsymbol{b})$. □

Observe that $\frac{1}{\alpha_{\boldsymbol{S}}} \leq \frac{1}{\alpha_{\boldsymbol{Z}}}$ from Lemma 7.14 and $|\Sigma_{\boldsymbol{S}}| \leq |\Sigma_{\boldsymbol{Z}}|$ since $\boldsymbol{S} \subseteq \boldsymbol{Z}$. So, the second approach of estimating $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ using the subset $\boldsymbol{S} \subseteq \boldsymbol{Z}$ produced by Amba would yield an asymptotically smaller error than directly using $\boldsymbol{Z}$ whenever $\frac{1}{\alpha_{\boldsymbol{S}}} \cdot \sqrt{\frac{|\boldsymbol{S}|}{n}} \cdot (|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Z}}|)^{\frac{1}{4}} \leq \frac{|\Sigma_{\boldsymbol{Z}}|}{n\alpha_{\boldsymbol{S}}} + \frac{1}{\sqrt{n\alpha_{\boldsymbol{S}}}} + \sqrt{\frac{|\Sigma_{\boldsymbol{Z}}|}{n}}$. This happens when

$$|\boldsymbol{S}| \cdot \sqrt{\frac{|\Sigma_{\boldsymbol{X}}|}{|\Sigma_{\boldsymbol{Z}}|}} < \max\left\{\frac{|\Sigma_{\boldsymbol{Z}}|}{n}, \frac{\alpha_{\boldsymbol{S}}}{|\Sigma_{\boldsymbol{Z}}|}, \alpha_{\boldsymbol{S}}^2\right\} \tag{7.16}$$

Observe that we know all terms in Eq. (7.16) except for $\alpha_{\boldsymbol{S}}$. For small n, say when $n \ll |\Sigma_{\boldsymbol{Z}}|$, the first term justifies estimating using the subset $\boldsymbol{S}$ produced by Amba instead of directly estimating using $\boldsymbol{Z}$. However, for large n, one would need to make the decision based on $\alpha_{\boldsymbol{S}}$. A similar kind of decision has to be made whether the third approach, of running Amba to produce $\boldsymbol{S} \subseteq \boldsymbol{Z}$ then Bamba to produce $\boldsymbol{S}' \subseteq \boldsymbol{Z}$, would yield a smaller estimation error. Note that $|\Sigma_{\boldsymbol{S}'}| \leq |\Sigma_{\boldsymbol{S}}|$ would imply $|\boldsymbol{S}'| \leq |\boldsymbol{S}|$ when all variables have the same domain size.

Theorem 7.6 (PAC causal effect estimation with positivity)**.** *Suppose we are given (1) $\varepsilon > 0$, (2) $\delta > 0$, (3) n i.i.d. samples from $\mathcal{P}(\boldsymbol{V})$, (4) an interventional query $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$, (5) a valid adjustment set $\boldsymbol{Z} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$, and (6) guaranteed that $\alpha_{\boldsymbol{S}} \geq \alpha \in (0,1)$ for any $\boldsymbol{S} \subseteq \boldsymbol{Z}$. Then, there is an algorithm that outputs a subset $\boldsymbol{S}^* \subseteq \boldsymbol{Z}$ and an estimate $\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y}) = \widehat{T}_{\boldsymbol{S}^*,\boldsymbol{x},\boldsymbol{y}}$ such that $\Pr\left(\left|\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y}) - \mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})\right| \leq \varepsilon\right) \geq 1 - \delta$ for some error term*

$$\varepsilon \in \widetilde{\mathcal{O}}\left(\frac{1}{n} \cdot \frac{|\Sigma_{\boldsymbol{S}^*}|}{\alpha} + \frac{1}{\sqrt{n}} \cdot \left(\frac{\sqrt{|\boldsymbol{Z}|} \cdot (|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{Z}}|)^{\frac{1}{4}}}{\alpha} + \frac{1}{\sqrt{\alpha}} + \sqrt{|\Sigma_{\boldsymbol{S}^*}|}\right)\right).$$

Moreover, if there exists a Markov blanket $\boldsymbol{S}$ of $\boldsymbol{X}$ such that
$|\boldsymbol{S}| \cdot \sqrt{\frac{|\Sigma_{\boldsymbol{X}}|}{|\Sigma_{\boldsymbol{Z}}|}} < \max\left\{\frac{|\Sigma_{\boldsymbol{Z}}|}{n}, \frac{\alpha_{\boldsymbol{S}}}{|\Sigma_{\boldsymbol{Z}}|}, \alpha_{\boldsymbol{S}}^2\right\}$, then $|\boldsymbol{S}^*| \leq k$.

Proof. Consider the following algorithm:

1. Run Amba to obtain $\boldsymbol{S} \subseteq \boldsymbol{Z}$
2. Check if $|\boldsymbol{S}| \cdot \sqrt{\frac{|\Sigma_{\boldsymbol{X}}|}{|\Sigma_{\boldsymbol{Z}}|}} < \max\left\{\frac{|\Sigma_{\boldsymbol{Z}}|}{n}, \frac{\alpha_{\boldsymbol{S}}}{|\Sigma_{\boldsymbol{Z}}|}, \alpha_{\boldsymbol{S}}^2\right\}$ according to Eq. (7.16)
3. If so, run Bamba to obtain $\boldsymbol{S}' \subseteq \boldsymbol{Z}$ and produce estimate $\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y}) = \widehat{T}_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}$
4. Otherwise, produce estimate $\widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y}) = \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$

That is, depending on Eq. (7.16), we decide to perform estimation based on $\boldsymbol{S}^* = \boldsymbol{S}'$ or $\boldsymbol{S}^* = \boldsymbol{Z}$. It remains to show that the bound holds for each case separately while noting that $\alpha_{\boldsymbol{S}}, \alpha_{\boldsymbol{S}'}, \alpha_{\boldsymbol{Z}} \geq \alpha$.

Case 1: $|\boldsymbol{S}| \cdot \sqrt{\frac{|\Sigma_{\boldsymbol{X}}|}{|\Sigma_{\boldsymbol{Z}}|}} < \max\left\{\frac{|\Sigma_{\boldsymbol{Z}}|}{n}, \frac{\alpha_{\boldsymbol{S}}}{|\Sigma_{\boldsymbol{Z}}|}, \alpha_{\boldsymbol{S}}^2\right\}$
So, we estimate using $\boldsymbol{S}^* = \boldsymbol{S}'$ produced from Bamba. This incurs an error of

$$\begin{aligned}
&|\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) - \widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y})| = |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}| \leq |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}| + |T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}| \\
&\in \widetilde{\mathcal{O}}\Bigg(\frac{1}{\alpha} \cdot \sqrt{\frac{|\boldsymbol{S}|}{n}} \cdot (|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Z}}|)^{\frac{1}{4}} + \frac{1}{\alpha} \cdot \sqrt{\frac{|\boldsymbol{S}'|}{n}} \cdot (|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{Z}}|)^{\frac{1}{4}} \\
&\qquad + \frac{|\Sigma_{\boldsymbol{S}'}|}{n\alpha} + \frac{1}{\sqrt{n\alpha}} + \sqrt{\frac{|\Sigma_{\boldsymbol{S}'}|}{n}}\Bigg) \\
&\hfill \text{(From Corollary 7.11, Corollary 7.12, and Corollary 7.13)} \\
&\subseteq \widetilde{\mathcal{O}}\left(\frac{1}{\alpha} \cdot \sqrt{\frac{|\boldsymbol{Z}|}{n}} \cdot (|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{Z}}|)^{\frac{1}{4}} + \frac{|\Sigma_{\boldsymbol{S}'}|}{n\alpha} + \frac{1}{\sqrt{n\alpha}} + \sqrt{\frac{|\Sigma_{\boldsymbol{S}'}|}{n}}\right) \\
&\hfill \text{(Since } \max\{|\boldsymbol{S}|, |\boldsymbol{S}'|\} \leq |\boldsymbol{Z}|\text{)} \\
&\subseteq \widetilde{\mathcal{O}}\left(\frac{1}{n} \cdot \frac{|\Sigma_{\boldsymbol{S}^*}|}{\alpha} + \frac{1}{\sqrt{n}} \cdot \left(\frac{\sqrt{|\boldsymbol{Z}|} \cdot (|\Sigma_{\boldsymbol{X}}| \cdot |\Sigma_{\boldsymbol{Y}}| \cdot |\Sigma_{\boldsymbol{Z}}|)^{\frac{1}{4}}}{\alpha} + \frac{1}{\sqrt{\alpha}} + \sqrt{|\Sigma_{\boldsymbol{S}^*}|}\right)\right) \\
&\hfill \text{(Since } \boldsymbol{S}^* = \boldsymbol{S}'\text{)}
\end{aligned}$$

Case 2: $|\boldsymbol{S}| \cdot \sqrt{\frac{|\Sigma_{\boldsymbol{X}}|}{|\Sigma_{\boldsymbol{Z}}|}} \geq \max\left\{\frac{|\Sigma_{\boldsymbol{Z}}|}{n}, \frac{\alpha_{\boldsymbol{S}}}{|\Sigma_{\boldsymbol{Z}}|}, \alpha_{\boldsymbol{S}}^2\right\}$

So, we estimate using $\boldsymbol{S}^* = \boldsymbol{Z}$. This incurs an error of

$$\begin{aligned}
&|\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) - \widehat{\mathcal{P}}_{\boldsymbol{x}}(\boldsymbol{y})| = |T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}| \\
&\in \widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{\Sigma_Z}|}{n\alpha} + \frac{1}{\sqrt{n\alpha}} + \sqrt{\frac{|\boldsymbol{\Sigma_Z}|}{n}}\right) && \text{(From Corollary 7.11)} \\
&\subseteq \widetilde{\mathcal{O}}\left(\frac{|\boldsymbol{\Sigma}_{\boldsymbol{S}^*}|}{n\alpha} + \frac{1}{\sqrt{n\alpha}} + \sqrt{\frac{|\boldsymbol{\Sigma}_{\boldsymbol{S}^*}|}{n}}\right) && \text{(Since } \boldsymbol{S}^* = \boldsymbol{S}'\text{)} \\
&\subseteq \widetilde{\mathcal{O}}\left(\frac{1}{n} \cdot \frac{|\boldsymbol{\Sigma}_{\boldsymbol{S}^*}|}{\alpha} + \frac{1}{\sqrt{n}} \cdot \left(\frac{\sqrt{|\boldsymbol{Z}|} \cdot (|\boldsymbol{\Sigma_X}| \cdot |\boldsymbol{\Sigma_Y}| \cdot |\boldsymbol{\Sigma_Z}|)^{\frac{1}{4}}}{\alpha} + \frac{1}{\sqrt{\alpha}} + \sqrt{|\boldsymbol{\Sigma}_{\boldsymbol{S}^*}|}\right)\right) \\
& && \text{(Adding more terms)}
\end{aligned}$$

Therefore, we see that the error upper bound holds for either case. □

Chapter 8

Conclusion for Part II

The results presented in Chapter 6 and Chapter 7 are from the works of [CSB22, CS23c] and [CSBS25] respectively.

In Chapter 6, we gave a complete understanding of the verification problem and an improved search algorithm under some standard causal inference assumptions, and solved the verification and search problems on a variety of settings. However, if our assumptions are violated by the data, then wrong causal conclusions may be drawn and possibly lead to unintended downstream consequences. A crucial limitation of this work is that we study an idealized setting with hard interventions and infinite samples while soft interventions may be more realistic in certain real-life scenarios (e.g. effects from parental vertices are not completely removed but only altered) and sample complexities play a crucial role when one has limited experimental budget (e.g. see [KJSB19] and [ABDK18] respectively). As such, we view our work as initiating the study of a flexible off-target model and establishing the theoretical foundations for the problem of causal discovery under off-target inteventions. There are several interesting extensions and open problems that remain. For instance, it would be of great practical interest to extend our results to more general causal models that include latents or even cycles. Furthermore, we did not consider sample complexity concerns nor study possible effects of non-compliance of interventions.

In Chapter 7, we focused on the problem of estimating the causal effect $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ in the PAC setting, given access to a valid adjustment set $\boldsymbol{Z}$, i.e. $\boldsymbol{Z}$ such that $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) = T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$, defined in Eq. (7.1). Our sample complexity and algorithmic results of Amba and Bamba hold for any arbitrary subset $\boldsymbol{A} \subseteq \boldsymbol{V}$ and thus also apply to the setting when $\boldsymbol{A} = \boldsymbol{Z}$ is a valid adjustment set, allowing us to relate the estimated quantities $T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$ and $T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}}$ to $T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} = \mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$. These results pave the way for future connections between causal discovery and causal effect estimation, while each standing alone as results of independent interest for fields such as local causal discovery. In [CSBS25], we also discuss how our methods relate to and are applicable to settings with latents. Three immediate open problems follow from the results of Chapter 7, which we expect to be of immediate future interest. Firstly, in comparison with the expectation bound of [ZBHK24], our PAC bounds contains an additional $\widetilde{\mathcal{O}}\left(\frac{|\Sigma_A|}{\varepsilon^2}\right)$ term. Can this term be eliminated, or can a matching lower bound show that it is necessary? Secondly, our AMBA algorithm for ε-Markov blanket discovery performs an exhaustive search over subsets of increasing size. Fortunately, this search is embarrassingly parallel, but is computationally prohibitive without access to parallel computing. Is there a more computationally efficient algorithm for this problem with (nearly) the same sample complexity? Thirdly, our BAMBA algorithm introduces an unclear tradeoff between using $\boldsymbol{S}'$ and $\boldsymbol{S}$ for adjustment, due to the potential of having $\alpha_{\boldsymbol{S}'} < \alpha_{\boldsymbol{S}}$ when the conditions relating $\boldsymbol{S}'$ and $\boldsymbol{S}$ hold only approximately. Is there an algorithm which optimally combines BAMBA and AMBA to achieve the better of their two sample complexities?

8.1 Some additional related work

We begin by reviewing graphical characterizations of valid adjustment sets given a causal graph (or an equivalence class of graphs) as input in Section 8.1.1. In some domains, these causal graphs may be constructed from expert knowledge, but when $\boldsymbol{V}$ is large or the system under consideration is not well-studied, practitioners may be unable to specify an accurate causal graph. Thus, we also review conditions for causal effect estimation which require minimal graphical knowledge. In Section 8.1.2, we review a different approach to the unspecified graph setting and discuss methods for learning all or part of a causal graph from data and interventions. Finally, we pivot to the potential outcomes (PO) perspective in Section 8.1.3, focusing on existing results on the statistical aspects of causal effect estimation.

8.1.1 Causal effect identification in the graphical setting

In the graphical framework, several classes of graphs have been used to formally define causal assumptions about a system, with the nodes of these graphs corresponding to the observed variables $\boldsymbol{V}$. Here, we focus our discussion on the causally sufficient setting where there are no unobserved variables in the causal DAGs.

Causal effect identification given a graph

For an intervention set $\boldsymbol{X}$, the graph $\mathcal{G}$ and $\mathcal{G}_{\overline{\boldsymbol{X}}}$ can be used to model the behavior of the system, and to derive relationships between the observational distribution $\mathcal{P}(\boldsymbol{V})$ and the interventional distribution $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{V})$. The details of these definitions are not necessary for our discussion; instead, we describe some of the major results which have been shown when taking these definitions as a starting point. Most importantly for our discussion, these definitions can be used to derive identification formulas, which express interventional queries $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ in terms of equations which only involve $\mathcal{P}(\boldsymbol{V})$, and thus permit causal effects to be estimated from only observational data. These identification formulas can be derived algorithmically, for example using the ID Algorithm [TP02], which is both sound and complete [SP06, HV06]. PAC bounds have also been established for the ID algorithm in [BGK^{+}22].

Importantly, the ID Algorithm may be able to construct an identification formula even if the adjustment formula (Eq. (7.1)) does not hold for any set $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$. However, in practice, the adjustment formula remains one of the most widely-used and well-studied identification approaches, due in part to its simplicity and its familiarity in the potential outcomes literature (see Appendix B.2.1). Particular attention has been given to developing graphical criteria for determining whether a set $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ is a valid adjustment set for $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$, and algorithmically finding such a set if one exists. A simple and intuitive condition for adjustment validity is the backdoor criterion [Pea95] in DAGs, which is sound, but not complete. This criterion has been refined by long line of work on sound and complete conditions [SVR10, vdZLT14, MC15, PTKM18, Per20] for different classes (and equivalence classes) of causal graphs. Our results further contribute to this line of work: as we discuss in Section 7.1, Lemma 7.9 directly implies a graphical condition that is sound for determining whether a subset is an adjustment set given a valid adjustment set; Appendix B.2.3 shows that under additional assumptions, this condition is also complete.

Causal effect identification without a graph

While the criteria above are stated in terms of a known causal graph $\mathcal{G}$, they can also be used in our setting to derive conditions under which Eq. (7.1) holds, even when the graph is an unknown. Indeed, using Eq. (7.1) requires quite minimal background knowledge of $\mathcal{G}$, as we now discuss. For simplicity, we limit our discussion to a single treatment variable X. In the case of DAGs, the backdoor criterion implies that $\boldsymbol{Z} = \mathrm{ND}(X)$ is a valid adjustment set, where $\mathrm{ND}(X)$ denotes the set of non-descendants of X in $\mathcal{G}$. Thus, assuming causal sufficiency, our method can be employed given only knowledge of $\mathrm{ND}(X)$, a quite common setting in applications such as healthcare, where a doctor's treatment assignment X can only depend on pre-treatment patient covariates. Under causal sufficiency and $\boldsymbol{Z} = \mathrm{ND}(X)$, the Markov blanket of X with respect to $\boldsymbol{Z}$ is the set $\boldsymbol{S} = \mathrm{Pa}(X)$, and our AMBA algorithm can be interpreted as searching for the parents of X.

In light of these connections, our results fit into a recent line of work establishing identifiability of causal effects with minimal graphical background knowledge. [EHS13] consider a setting that matches ours in the DAG setting with $\boldsymbol{Z} = \mathrm{ND}(X)$, and establish a condition similar to Lemma 7.9 to determine whether $\boldsymbol{A} \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ is a valid adjustment set. While our condition is sound, their condition is both sound *and* complete, but relies on conditional dependence checks instead of only conditional independence checks. Furthermore, in contrast with our work, where statistical guarantees are a primary focus, their work does not provide any guarantees outside of the oracle setting, though it would be interesting to study their approach in the finite-sample setting.

Follow-up works in this space have extended this problem to the causally insufficient setting by incorporating additional background knowledge on $\mathcal{G}$; all of the works discussed assume knowledge of $\boldsymbol{Z} = \mathrm{ND}(X)$. For example, [CLL+22] assumes knowledge of some variable A that is a "cause or spouse of treatment only (COSO)" variable, i.e. that A is adjacent to X but not to Y in $\mathcal{G}$, and establishes a sound condition for determining whether $\boldsymbol{S} \subseteq \boldsymbol{Z}$ is an adjustment set. Relatedly, [SSA22] assumes knowledge of some variable A that is a parent of X and establishes a similar condition. Both conditions are sound, but not complete; in contrast, we show in Appendix B.2.3 that the Bamba approach is both sound and complete in the causally sufficient setting when $\boldsymbol{Z} = \mathrm{ND}(X)$. Finally, [SSK23] goes beyond using the adjustment formula for identification, in particular studying when background knowledge is sufficient to identify the causal effect using frontdoor adjustment.

8.1.2 Causal graph discovery

Chapter 7 is strongly motivated by our recognition of the pressing need for better connections between the areas of causal effect estimation and causal structure learning. In a typical causal discovery (a.k.a. causal structure learning) task, one takes data on the observed variables $\boldsymbol{V}$ as input, and seeks to return a causal graph $\mathcal{G}$ (or an equivalence class of graphs) that provides an accurate causal model of the system. Traditionally, this goal is (implicitly or explicitly) motivated by the utility of such a model for generating causal predictions, e.g. predicting $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$ as discussed in Chapter 7.

Causal discovery and faithfulness

The field of causal discovery is quite well-developed, and has been the subject of several surveys, e.g. [HDMM18, GZS19, VCB22, SU23]. Various approaches address settings such as learning from observational data in the causally sufficient setting [SGS00, Chi03, ZARX18, SWU21] and in the causally insufficient setting [SGS00, CMKR12], as well as learning from interventional data, possibly involving actively chosen interventions [EGS05, EGS06, Ebe07, HB12, HLV14, SKDV15, WSYU17, KDV17, LKDV18, GKS+19, JKSB20, SMG+20, CSB22, CS23c, CGB23, CS23b, CS23a].

Table 8.1 and Table 8.2 summarize some existing upper (sufficient) and lower (worst case necessary) bounds on the size ($|\mathcal{I}|$, or $\mathbb{E}(|\mathcal{I}|)$ for randomized algorithms) of intervention sets that fully orient a given essential graph using ideal interventions. These lower bounds are "worst case" in the sense that there exists a graph, typically a clique, which requires the stated number of interventions. Observe that there are settings where adaptivity and randomization strictly improves the number of required interventions.

Size	Adaptive	Randomized	Graph	Upper bound	Reference
1	✗	✗	General	$n-1$	[EGS06]
1	✗	✓	General	$\frac{2}{3}n - \frac{1}{3}$ for $n > 3$	[Ebe10]
1	✓	✗	Tree	$\mathcal{O}(\log n)$	[SKDV15]
1	✓	✗	Tree	$\lceil \log n \rceil$	[GKS+19]
$\leq k$	✗	✗	General	$(\frac{n}{k} - 1) + \frac{n}{2k} \log_2 k$	[EGS05]
$\leq k$	✓	✗	Tree	$\lceil \log_{k+1} n \rceil$	[GKS+19]
$\leq k$	✓	✓	Clique	$\mathcal{O}(\frac{n}{k} \log \log k)$	[SKDV15]
∞	✗	✗	General	$\log_2 n$	[EGS05]
∞	✗	✗	General	$\lceil \log_2(\omega(\mathcal{E}(\mathcal{G}))) \rceil$	[HB14]
∞	✗	✓	General	$\mathcal{O}(\log \log n)$	[HLV14]

Table 8.1: Some known upper bounds on the size ($|\mathcal{I}|$, or $\mathbb{E}(|\mathcal{I}|)$ for randomized algorithms) of the intervention set sufficient to fully orient a given essential graph $\mathcal{E}(\mathcal{G})$. The first three columns indicate the setting which the algorithm operates in terms of intervention size, adaptivity, and randomness. The fourth column indicate whether the algorithm is for special graph classes. Roughly speaking, the algorithm has more power as we move down the rows since it can use larger intervention sets, be adaptive, utilize randomization, and possibly only work on special graph classes.

Size	Adaptive	Randomized	Lower bound	Reference
1	✗	✓	$\frac{2}{3}n - \frac{1}{3}$ for $n > 3$	[Ebe10]
1	✓	✗	$n-1$	[EGS06]
$\leq k$	✗	✗	$(\frac{n}{k} - 1) + \frac{n}{2k} \log_2 k$	[EGS05]
$\leq k$	✓	✓	$\frac{n}{2k}$	[SKDV15]
∞	✗	✗	$\log_2 n$	[EGS05]
∞	✗	✓	$\Omega(\log \log n)$	[HLV14]
∞	✓	✓	$\lceil \log_2(\omega(\mathcal{E}(\mathcal{G}))) \rceil$	[HB14]

Table 8.2: Some known lower bounds on the size ($|\mathcal{I}|$, or $\mathbb{E}(|\mathcal{I}|)$ for randomized algorithms) of the intervention set necessary to fully orient a given essential graph $\mathcal{E}(\mathcal{G})$. The first three columns indicate the setting which the algorithm operates in terms of intervention size, adaptivity, and randomness. Roughly speaking, the setting becomes easier as we move down the rows so the lower bounds are stronger as we move down the rows. On cliques, [SKDV15] also showed that $\geq n/2$ vertices must be intervened.

A few comments are in order:

Intervention size Since interventions are expensive, natural restrictions on the size of any intervention $\boldsymbol{S} \in \mathcal{I}$ has been studied. Bounded size interventions enforce that an upper bound of $|\boldsymbol{S}| \leq k$ always while unbounded size interventions allow k to be as large as $n/2$. Note that it does not make sense to intervene on a set $\boldsymbol{S}$ with $|\boldsymbol{S}| > n/2$ since intervening on $\overline{\boldsymbol{S}}$ yields the same information while being a strictly smaller interventional set. Atomic interventions are a special case where $k = 1$.

Adaptivity A passive/non-adaptive/simultaneous algorithm is one which, given an essential graph $\mathcal{E}(\mathcal{G}^*)$, decides a *set* of interventions without looking at the outcomes of the interventions. Meanwhile, active/adaptive algorithms can provide a *sequence* of interventions one-at-a-time, possibly using any information gained from the outcomes of earlier chosen interventions.

Determinism An algorithm is deterministic if it always produces the same output given the same input. Meanwhile, randomized algorithms produces an output from a distribution. Analyses of randomized algorithms typically involve probabilistic arguments and their performance is measured in expectation with probabilistic success. Typically, they will be shown to succeed with high probability in n: as the size of the graph n increases, the failure probability decays quickly in the form of n^{-c} for some constant $c > 1$. The ability to use random bits (e.g. outcome of coin flips) is very powerful and may allow one to circumvent known deterministic lower bounds.

Special graph classes Two graph classes of particular interest are cliques and trees. If $CC(\mathcal{E}(\mathcal{G}^*))$ is a clique, then all $\binom{n}{2}$ edges are present and fully orienting the clique is equivalent to finding the unique valid permutation on the vertices. As such, cliques are often used to prove worst case lower bounds. Meanwhile, if $CC(\mathcal{E}(\mathcal{G}^*))$ is a tree, then there must be a unique root (else there will be v-structures) and it suffices to intervene on the root node to fully orient the tree. This will later be obvious through the lenses of covered edges: all covered edges are incident to the root.

Separating systems [HEH13] drew connections between causal discovery via interventions and the concept of separating systems from the combinatorics literature. This was extended by [SKDV15] to the bounded size and adaptive settings. An (n, k)-separating system is a Boolean matrix with n columns where each row has at most k ones, indicating which vertex is to be intervened upon. Using their proposed separating system construction based on "label indexing", [SKDV15] showed that roughly $\frac{n}{k} \log_{\frac{n}{k}} n$ interventions is sufficient to fully an essential graph $\mathcal{G}$ with bounded size interventions. On cliques (i.e. worst case lower bound), [SKDV15] showed that the bound is tight while only roughly $\frac{\chi(\mathcal{E}(\mathcal{G}))}{k} \log_{\frac{\chi(\mathcal{E}(\mathcal{G}))}{k}} \chi(\mathcal{E}(\mathcal{G}))$ interventions are necessary for general graphs, even if the interventions are chosen adaptively or in a randomized fashion. Note that there is a slight gap between $\frac{\chi(\mathcal{E}(\mathcal{G}))}{k} \log_{\frac{\chi(\mathcal{E}(\mathcal{G}))}{k}} \chi(\mathcal{E}(\mathcal{G}))$ and $\frac{n}{k} \log_{\frac{n}{k}} n$ on general graphs.

Universal bounds for minimum sized atomic interventions Beyond worst case lower bounds, recent works have studied universal bounds for orienting essential graphs $\mathcal{E}(\mathcal{G}^*)$ using atomic interventions [SMG+20, PSS22]. These universal bounds depend on graph parameters of $\mathcal{E}(\mathcal{G})$ beyond the number of nodes n. [SMG+20] showed that search algorithms must use at least $\sum_{H \in CC(\mathcal{E}(\mathcal{G}^*))} \lfloor \frac{\omega(H)}{2} \rfloor$ interventions, where $\mathcal{H}$ is a chain component of $\mathcal{E}(\mathcal{G}^*)$ and the summation across chain components is a consequence of Lemma 2.53. They also introduced a graph concept called directed clique trees and designed an adaptive, deterministic algorithm. On intersection-incomparable chordal graphs, their algorithm outputs an intervention set of size $\mathcal{O}(\log_2(\max_{\mathcal{H} \in CC(\mathcal{E}(\mathcal{G}^*))} \omega(H)) \cdot \nu_1(\mathcal{G}^*))$. More recently, [PSS22] introduced the notion of clique-block shared-parents orderings and showed that any search algorithm for an essential graph $\mathcal{E}(\mathcal{G}^*)$ with r maximal cliques requires at least $\lceil \frac{n-r}{2} \rceil$ interventions and $\nu_1(\mathcal{G}) \leq n - r$ for any $\mathcal{G} \in [\mathcal{G}^*]$.

Non-atomic interventions The randomized algorithm of [HLV14] fully orients an essential graph using $\mathcal{O}(\log(\log(n)))$ unbounded interventions in expectation. Building upon this, [SKDV15] shows that $\mathcal{O}(\frac{n}{k} \log(\log(k)))$ bounded sized interventions (each involving at most k nodes) suffice.

Additive vertex costs [KDV17, GSKB18, LKDV18] studied the *non-adaptive* search setting where vertices may have different intervention costs and intervention costs accumulate additively. [GSKB18] studied the problem of maximizing number of oriented edges given a budget of atomic interventions while [KDV17, LKDV18] studied the problem of finding a minimum cost (bounded size) intervention set that fully orients the essential graph. [LKDV18] showed that computing the minimum cost intervention set is NP-hard and gave search algorithms with constant approximation factors.

Random graphs [HLV14, KSSU19] showed that Erdős-Rényi graphs can be easily oriented.

Many of the above described algorithms enjoy theoretical guarantees in the well-specified setting, i.e. under the assumption that the system is correctly described by some (unknown) causal graph $\mathcal{G}^*$. In this setting, an algorithm is said to be consistent if it recovers $\mathcal{G}^*$, or an appropriate equivalence class, with probability one in the limit of infinite data. Significant attention has been devoted to finding conditions under which various causal discovery algorithms are consistent. For example, the well-known faithfulness assumption requires that if $\boldsymbol{A}$ and $\boldsymbol{B}$ and not d-separated by $\boldsymbol{C}$ in $\mathcal{G}^*$, then $\boldsymbol{A} \not\perp\!\!\!\perp \boldsymbol{B} \mid \boldsymbol{C}$ in $\mathcal{P}(\boldsymbol{V})$. Although faithfulness is a sufficient condition for the consistency of many causal discovery algorithms, it is often a stronger condition than necessary, and many weaker conditions have been established, see [Lam23] for a recent review and comparison of such conditions. The search for weaker consistency conditions is motivated by a practical issue: although the consistency of an algorithm may depend only on there being no violations of faithfulness, near violations of faithfulness (where the conditional independence $\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} \mid \boldsymbol{C}$ nearly holds, e.g. $\Delta_{\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} | \boldsymbol{C}} \leq \varepsilon$ for some small ε) can significantly affect its finite sample properties. Therefore, finite sample guarantees for graph recovery [KB07, MKB09, GDA20, WD21, GTA22] often depend on assumptions such as strong faithfulness, which may be significantly more restrictive in practice [URBY13].

In Chapter 7, we avoid making any such assumptions. Indeed, since our goal is causal effect estimation, rather than graph recovery, faithfulness conditions are unnecessary, and existing sample complexity guarantees for causal discovery are pessimistic for our purposes. Within the graphical framework, a main message of our work is that accurate causal effect estimation does not require learning the correct causal graph $\mathcal{G}^*$. For example, if $\mathcal{G}^*$ has "weak" edges, these may be hard to distinguish from missing edges, but those edges are also exactly those that do not significantly impact causal effects; in pragmatic terms, whether an edge is weak or missing is "a difference that doesn't make a difference". We provide a concrete example of this phenomenon in Appendix B.2.5. Nonetheless, such conditions may be useful in improving the sample complexity and/or the computational complexity of our approach, as we discuss in Appendix B.2.6.

Cautious approaches and local causal discovery

To better align theory and practice, a few recent works have focused on new kinds of theoretical guarantees. Two contemporaneous works [Mal24, CGM24] explicitly consider the interplay between causal discovery and causal effect estimation. As in our work, [Mal24] advocates the use of conditional dependence tests (as opposed to conditional independence tests) to control model misspecification, an approach they call "cautious" causal discovery, where [CGM24] advocate a bootstrap-style approach. However, their guarantees are for the asymptotic setting, rather than the PAC setting considered in this chapter, and their approaches aim to recover an entire causal graph, unlike our approach.

More closely related to our approach are methods for local causal discovery, which aim to recover only part of a causal graph. Indeed, one of the canonical problems in local discovery is Markov blanket recovery [KS96, FFT$^+$03, TAS03, Ram06, PNBT07, FD08, AST$^+$10a, AST$^+$10b, GJ17, LYW$^+$20, DW22], potentially combined with partial edge orientation [YZW$^+$08, WZZG14, GJ15, GCL23] and often used in the context of full causal discovery algorithms [MC04, TBA06, SWU21, GA21]. A number of these algorithms employ greedy search, adding variables to the Markov blanket one at a time (e.g. [TAS03, FD08, GJ17]). However, greedy search is not guaranteed to return a correct Markov blanket without additional assumptions, such as those in [GA21], in which the authors also provide finite sample guarantees. In contrast, many non-greedy algorithms do enjoy consistency guarantees (i.e. recovery of a correct Markov blanket in the infinite data limit), but thus far lack finite sample guarantees.

Thus, our finite sample guarantees for the (non-greedy) Amba algorithm contribute to this important line of work, and may be of independent interest beyond the context of causal effect estimation. Furthermore, our Bamba highlights that using only local structure may be suboptimal for some estimation problems. This fact suggests that we extend from local causal discovery to the more general problem of targeted causal discovery, i.e., causal discovery tailored to specific estimation problems, analogous to techniques such as targeted maximum likelihood estimation [vdLR06, SR17].

8.1.3 Causal effect estimation via covariate adjustment

Now, we relate our results to existing statistical results on causal effect estimation, focusing on estimation using the adjustment formula. Existing results are largely written in terms of potential outcomes but, as with our result, are usually applicable as long as Eq. (7.1) holds and are thus independent of framework choice.[12] In many domains such as healthcare and econometrics, Eq. (7.1) can be justified by domain knowledge. For example, in healthcare, where $\boldsymbol{X}$ and $\boldsymbol{Y}$ may represent medical treatments and patient outcomes, respectively, it is sufficient for $\boldsymbol{Z}$ to contain all information that doctors may be using to assign treatment, e.g. patient demographic information and past medical history. In such domains, $\boldsymbol{Z}$ are often referred to as a set of covariates; we adopt this terminology here.

As datasets become larger and richer, causal effect estimation is increasingly being applied to problems with high-dimensional covariates. These problems present novel challenges, including violations of the overlap assumption [DDF$^+$21] and the breakdown of traditional asymptotic results. Dimensionality reduction techniques such as feature selection are often crucial to addressing the challenges. However, in the context of treatment effect estimation, naïve usage of feature selection methods such as the Lasso can introduce substantial misspecification bias. Several works aim to address this issue; here, we focus on methods based on feature selection, pointing readers to [YNB$^+$22] as a starting point for methods using other forms of dimensionality reduction, and to [WD19, YGL$^+$20] for a more complete review and comparison of methods based on feature selection.

Whereas our work focuses on discrete covariates, with no additional assumptions on $\mathcal{P}(\boldsymbol{Z})$, $\mathcal{P}(\boldsymbol{X} \mid \boldsymbol{Z})$ and $\mathcal{P}(\boldsymbol{Y} \mid \boldsymbol{X}, \boldsymbol{Z})$, the majority of prior works consider *continuous*

[12] When the random variables are continuous or mixed, Eq. (7.1) is written as $T_{\boldsymbol{s},\boldsymbol{x},\boldsymbol{y}} = \mathbb{E}_{\boldsymbol{S}}[\mathcal{P}(\boldsymbol{Y} = \boldsymbol{y} \mid \boldsymbol{X} = \boldsymbol{x}, \boldsymbol{S})]$.

covariates $\boldsymbol{Z}$, and thus require additional assumptions, such as parametric or smoothness assumptions. When X is a binary treatment, a common assumption is that $\mathcal{P}(X \mid \boldsymbol{Z})$ follows a logit model, so that $\mathcal{P}(X \mid \boldsymbol{Z})$ is parameterized by a vector $\boldsymbol{\beta} \in \mathbb{R}^{|\boldsymbol{Z}|}$. Similarly, when Y is a scalar outcome, a common assumption is that $\mathcal{P}(Y \mid X, \boldsymbol{Z})$ follows a linear model, i.e. it is parameterized by a vector $\boldsymbol{\gamma} \in \mathbb{R}^{|\boldsymbol{Z}|}$. Sparsity assumptions may be imposed on one or both of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$; for example, [SE17] and [WS20] assume sparsity on $\boldsymbol{\beta}$, [BWZ19] and [AIW18] assume sparsity on $\boldsymbol{\gamma}$, and [GSK21] assumes sparsity on both. Other common assumptions include semiparametric restrictions, e.g. partially linear models [BCH14, CCD$^+$18], and smoothness assumptions [FLM21].

In these works, sparse regression methods (e.g. Lasso and its variants) play a role similar to our search for a smaller adjustment set $\boldsymbol{S} \subseteq \boldsymbol{Z}$, and the choice of regularization parameter plays a role similar to our choice of ε in balancing between misspecification bias and estimation error. In comparison to these methods, our focus on discrete variables obviates the need for additional assumptions, and allows us to establish deeper connections between causal effect estimation and fields such as distribution testing [Can20b] and property estimation [CSS19]; connections which make the problem accessible to a wider audience and provide access to a broader range of tools.

8.2 Other unpresented works in Part II

In [CS23a], we define r-adaptivity that interpolates between non-adaptivity (for $r = 1$) and full adaptivity (for $r = n$). We provide a r-adaptive algorithm that achieves $\mathcal{O}(\min\{r, \log n\} \cdot n^{1/\min\{r, \log n\}})$ approximation with respect to the verification number. We further extended this to the k-bounded intervention setting and also showed that our approximation factor is tight for any r.

In [CS23b], we show that the previously considered benchmark of the verification number is no longer meaningful in the context of weighted causal graphs. More formally, we prove that no algorithm (even with infinite computational power) can achieve an asymptotically better approximation than $\mathcal{O}(n)$ with respect to the verification cost $\overline{\nu}(\mathcal{G}^*)$ for all ground truth causal graphs on n nodes. Therefore, $\overline{\nu}(\mathcal{G}^*)$ is too strong and an unreasonable benchmark to compare against in the weighted setting. This is similar in spirit to the negative result of Lemma 6.22 to justify why a bound of $\mathcal{O}(\log n \cdot \nu_1(\mathcal{G}^*, \boldsymbol{T}))$ for any subset of target edges $\boldsymbol{T} \subseteq \boldsymbol{E}$ is unattainable in general. Both these negative results are pointing out that comparing against an algorithm that *knows* $\mathcal{G}^*$ can be overly pessimistic in certain settings and suggests that one should "compare against the "best" algorithm that does *not* know $\mathcal{G}^*$". Given the abovementioned negative result, we propose the following new benchmark $\overline{\nu}^{\max}(\mathcal{G}^*) = \max_{\mathcal{G} \in [\mathcal{G}^*]} \overline{\nu}(\mathcal{G})$ in [CS23b] which captures the intuition that any algorithm has to grapple with the worst-case causal graph in the given MEC, and then provide adaptive search algorithms that are competitive against the

$\overline{\nu}^{\max}(\mathcal{G}^*)$.

In [CSU24], we propose and study a stochastic interventional model that aims to model off-target interventions; the ideal intervention setting that we have studied so far in Chapter 6 is a special case. After proposing and formalizing the off-target intervention model, we establish a two-way reduction between the off-target verification problem and the well-studied stochastic set covering problem. This equivalence allows us to leverage existing results and techniques in the literature to design our algorithms. Then, we prove that no algorithm can achieve non-trivial competitive approximation guarantees against the off-target verification number, even when all actions have unit weight. This shows the difficulty of the off-target search problem and motivates the need for new benchmarks. With respect to the benchmark $\overline{\nu}^{\max}(\mathcal{G}^*) = \max_{\mathcal{G} \in [\mathcal{G}^*]} \overline{\nu}(\mathcal{G})$ proposed in [CS23b], we propose algorithms that are competitive against a quantity that captures the performance of any algorithm against the worst-case causal graph within the same Markov equivalence class. Our algorithm runs in polynomial time and is guaranteed to use at most a polylogarithmic number of expected interventions more than the worst-case optimal solution.

Part III

Utilizing imperfect advice

Chapter 9

Online bipartite matching with imperfect advice

> "As far as the laws of mathematics refer to reality, they are not certain; and as far as they are certain, they do not refer to reality."
>
> \- Albert Einstein [Ein22]

> "If you aren't in the moment, you are either looking forward to uncertainty, or back to pain and regret."
>
> \- Jim Carrey, 60 Minutes [Leu04]

9.1 Introduction

Finding matchings in bipartite graphs is a mainstay of algorithms research. The area's mathematical richness is complemented by a vast array of applications — any two-sided market (e.g., kidney exchange, ridesharing) yields a matching problem. In particular, the *online* variant enjoys much attention due to its application in internet advertising. Consider a website with a number of pages and ad slots (videos, images, etc.). Advertisers specify ahead of time the pages and slots they like their ads to appear in, as well as the target user. The website is paid based on the number of ads appropriately fulfilled. Crucially, ads slots are available only when traffic occurs on the website and are not known in advance. Thus, the website is faced with the *online* decision problem of matching advertisements to open ad slots.

The classic online unweighted bipartite matching problem by [KVV90] features n offline vertices $\boldsymbol{U}$ and n online vertices $\boldsymbol{V}$. Each $V \in \boldsymbol{V}$ reveals its incident edges sequentially upon arrival. With each arrival, one makes an irrevocable decision whether (and how) to match V with a neighboring vertex in $\boldsymbol{U}$. The final offline graph $\mathcal{G} = (\boldsymbol{U} \cup \boldsymbol{V}, \boldsymbol{E})$ is assumed to have a largest possible matching of size $n^* \leq n$, and we seek online algorithms producing matchings of size as close to n^* as possible. More formally, a matching in the graph $\mathcal{G}$ is a set of edges $\boldsymbol{M} \subseteq \boldsymbol{E}$ such that for every vertex $W \in \boldsymbol{U} \cup \boldsymbol{V}$, there is at most one edge in $\boldsymbol{M}$ incident to W.

The performance of a (randomized) algorithm $\mathcal{A}$ is measured by its *competitive ratio*:

$$\min_{\mathcal{G}=(U \cup V, E)} \min_{V\text{'s arrival sequence}} \frac{\mathbb{E}[\text{number of matches by } \mathcal{A}]}{n^*}, \tag{9.1}$$

where the randomness is over any random decisions made by $\mathcal{A}$. Traditionally, one assumes the *adversarial arrival model*, i.e., an adversary controls both the final graph $\mathcal{G}$ and the arrival sequence of online vertices.

Since any maximal matching has size at least $n^*/2$, a greedy algorithm trivially attains a competitive ratio of $1/2$. Indeed, [KVV90] show that no deterministic algorithm can guarantee better than $1/2 - o(1)$. Meanwhile, the randomized RANKING algorithm of [KVV90] attains an asymptotic competitive ratio of $1 - 1/e$ which is also known to be optimal [KVV90, GM08, BM08, Vaz22].

In practice, *advice* (also called predictions or side information) is often available for these online instances. For example, online advertisers often aggregate past traffic data to estimate the future traffic and corresponding user demographic. While such advice may be imperfect, it may nonetheless be useful in increasing revenue and improving upon aforementioned worst-case guarantees. Designing algorithms that utilize such advice in a principled manner falls under the research paradigm of learning-augmented algorithms. In the context of online bipartite matching, a natural design goal for learning-augmented algorithms is as follows.

Goal 9.1. Let β be the best-known competitive ratio attainable by any classical advice-free online algorithm. Can we design a learning-augmented algorithm for the online bipartite matching problem that is 1-consistent and β-robust?

Clearly, Goal 9.1 depends on the form of advice as well as a suitable measure of its quality. Setting these technicalities aside for now, we remark that Goal 9.1 strikes the best of all worlds: it requires that a perfect matching be obtained when the advice is perfect, while not sacrificing performance with respect to advice-free algorithms when faced with low-quality advice. In other words, there is potential to benefit, but no possible harm when employing such an algorithm.

Remark 9.2. Throughout this chapter, we will use star $(\cdot)^*$ and hat $\widehat{(\cdot)}$ to denote ground truth and advice quantities respectively. In particular, we use $n^* \leq n$ and $\widehat{n} \leq n$ to denote the maximum matching size in the final offline graph $\mathcal{G}^*$ and advice graph $\widehat{\mathcal{G}}$ respectively. Note that star $(\cdot)^*$ quantities are not known and exist purely for the purpose of analysis.

9.2 Our main results

9.2.1 Impossibility in adversarial arrival model

While there has been some learning-augmented results in the space of online bipartite matching [ACI22, JM22, AGKK23, LYR23], none of them are able to achieve Goal 9.1.

With this in mind, our first main result states that any learning augmented algorithm that is 1-consistent cannot be strictly more than $1/2$-robust under adversarial arrivals. As this robustness factor is worse than the competitive ratio of $1 - 1/e$ guaranteed by known advice-free algorithms, Goal 9.1 is unattainable under the adversarial arrival model.

Theorem 9.3. *For even n, there exists input graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ such that no advice can distinguish between the two within $n/2$ online arrivals. Consequently, an algorithm* cannot *be both 1-consistent and strictly more than $1/2$-robust.*

While Theorem 9.3 appears simple, we stress that hardness results for learning-augmented algorithms are rare since the form of advice and its utilization is arbitary. For instance, [ACI22] only showed that when advice is the true degrees of the offline vertices, there exist inputs such that any learning-augmented algorithm can only achieve a competitive ratio of at most $1 - 1/e + o(1)$. In fact, Theorem 9.3 can be strengthened: for any $\alpha \in [0, 1/2]$, no algorithm can be simultaneously $(1 - \alpha)$-consistent and strictly more than $(1/2 + \alpha)$-robust. The proof is essentially identical and deferred to Appendix C.1.1.

9.2.2 Achievability in random order arrival model

Following the TestAndAct framework for designing learning-augmented algorithms, we propose an algorithm TestAndMatch achieving Goal 9.1 under the weaker random arrival model. In this arrival model, the offline graph is still worst case adversarial but the online vertices arrive in random order; see Section 9.3.1.

Theorem 9.4 (Informal; see Theorem 9.5)**.** *Given an advice for the offline graph that implies a perfect matching, under the random order arrival model, TestAndMatch produces a matching with competitive ratio at least* $\max\{1 - \gamma, \beta \cdot (1 - o_n(1))\}$*, where $\gamma \in (0, 1)$ is a measure of the advice quality, and succeeds with probability $1 - \delta$.*

Noting that $\lim_{n \to \infty} \beta \cdot (1 - o_n(1)) = \beta$, we see that Theorem 9.4 not only achieves Goal 9.1 asymptotically but also provide a competitive ratio that interpolates smoothly between 1 and β depending on the quality of the advice. In fact, TestAndMatch is a meta-algorithm that uses any advice-free baseline algorithm as a black-box and so our robustness guarantee improves as β improves.

The advice considered here is a *histogram over types* of online vertices. In the context of online advertising, this corresponds to a forecast of the user demographic and which ads they can be matched to. More formally, a *type* of an online vertex $V \in \boldsymbol{V}$ refers to the subset of offline vertices $\{U \in \boldsymbol{U} : \{U, V\} \in \boldsymbol{E}\}$ that V is neighbors with [BKP20], and is only revealed when V itself arrives. Note that the offline graph is fully defined whenever given the types of the online vertices and that at most n types are realized through $\boldsymbol{V}$ even though there are 2^n possible types.

TestAndMatch assumes perfect advice while simultaneously testing for its accuracy via the initial arrivals. If the advice is deemed useful, we mimic the matching suggested by it; else, we revert to an advice-free method. The testing phase is kept short (sublinear in n) by utilizing state-of-the-art ℓ_1 estimators from distribution testing. We analyze our algorithm's performance as a function of the quality of advice, showing that its competitive ratio gracefully degrades to β as quality of advice decays. To the best of our knowledge, our work is the first that shows how one can leverage techniques from the property testing literature to designing learning-augmented algorithms.

While our contributions are mostly theoretical, we discuss various practical extensions of TestAndMatch in Section 9.6 and show preliminary experiments in Appendix C.1.5.

9.3 Technical overview

9.3.1 Some background

Before we give the technical overview of our results, we first give a brief introduction to the relevant background on the fertile landscape of online bipartite matching to provide some context for our results.

Arrival models

The degree of control an adversary has over $\boldsymbol{V}$ affects analysis and algorithms. The *adversarial arrival model* is the most challenging, with both the final graph $\mathcal{G}$ and the order in which online vertices arrive chosen by the adversary. Here, an algorithm's competitive ratio is given by Eq. (9.1). In *random arrival models*, $\mathcal{G}$ remains adversarial but the arrival order is random. For Theorem 9.4, we assume the *Random Order* setting, where an adversary chooses a $\mathcal{G}$, but the arrival order of $\boldsymbol{V}$ is a uniformly random permutation. In this setting, the competitive ratio is defined as

$$\min_{\mathcal{G}=(\boldsymbol{U}\cup\boldsymbol{V},\boldsymbol{E})} \mathbb{E}_{\boldsymbol{V}\text{'s arrival sequence}} \frac{\mathbb{E}[\text{number of matches by } \mathcal{A}]}{n^*} . \tag{9.2}$$

Two even easier random arrival models exist: (i) *known-i.i.d. model* [FMMM09], where the adversary chooses a distribution over types (which is known to us), and the arrivals of V are chosen by sampling i.i.d. from this distribution, and (ii) *unknown-i.i.d. model*, which is the same as known-i.i.d. but with the distributions are not revealed to us. The competitive ratios between these arrival models are known to exhibit a hierarchy of difficulty [Meh13]:

$$\text{Adversarial} \leq \text{Random Order} \leq \text{Unknown-i.i.d.} \leq \text{Known-i.i.d.}$$

As our Random Order setting is the most challenging amongst these random arrival models, our methods also apply to the unknown-i.i.d. and known-i.i.d. settings.

Advice-free online bipartite matching

Table 9.1 summarizes known results about attainable competitive ratios and impossibility results in the adversarial and Random Order arrival models. In particular, observe that there is a gap between the upper and lower bounds in the Random Order arrival model which remains unresolved.

	Adversarial	Random Order
Deterministic algorithm	$1/2$	$1 - 1/e$
Deterministic hardness	$1/2$	$3/4$
Randomized algorithm	$1 - 1/e$	0.696
Randomized hardness	$1 - 1/e + o(1)$	0.823

Table 9.1: Known competitive ratios for the classic unweighted online bipartite matching problem for deterministic and randomized algorithms under the adversarial and Random Order arrival models. Note that $1 - 1/e \approx 0.63$.

On the positive side of things, the deterministic GREEDY algorithm which matches newly arrived vertex with any unmatched offline neighbor attains a competitive ratio of at least $1/2$ in the adversarial arrival model and at least $1 - 1/e$ in the random arrival model [GM08]. Meanwhile, the randomized RANKING algorithm of [KVV90] achieves a competitive ratio of $1 - 1/e$ in the adversarial arrival model. In the Random Order arrival model, RANKING achieves a strictly larger competitive ratio, shown to be at least 0.653 in [KMT11] and 0.696 in [MY11]. However, [KMT11] showed that RANKING cannot beat 0.727 in general; so, new ideas will be required if one believes that the tight competitive ratio bound is 0.823 [MGS12].

On the negative side, the following example highlights the key difficulty faced by online algorithms. Consider the gadget for $n = 2$ in Fig. 9.1, where the first online vertex V_1 neighbors with both U_1 and U_2 and the second online vertex V_2 neighbors with only one of U_1 or U_2. Even when promised that the true graph is either $\mathcal{G}_1$ or $\mathcal{G}_2$, any online algorithm needs to correctly guess whether to match V_1 with U_1 or U_2 to achieve perfect matching when V_2 arrives.

By repeating the gadget of Fig. 9.1 multiple times sequentially, any deterministic algorithm can only attain competitive ratios of $1/2$ and $3/4$ in the adversarial and random arrival models respectively. For randomized algorithms, [KVV90] showed that RANKING

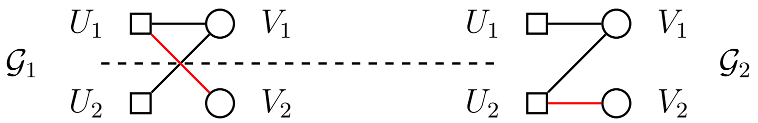


Figure 9.1: Gadget for $n = 2$. Red edges observed when V_2 arrives.

is essentially optimal for the adversarial arrival model since no algorithm can achieve a competitive ratio better than $1 - 1/e + o(1)$. In the Random Order arrival model, [GM08, Appendix E] showed that a ratio better than $5/6 \approx 0.83$ cannot be attained by brute force analysis of a 3×3 gadget bipartite graph. Subsequently, [MGS12] showed that no algorithm (deterministic or randomized) can achieve a competitive ratio better than 0.823.

Technically speaking, the hardness result of [MGS12] is for the known i.i.d. model introduced by [FMMM09], but this extends to the Random Order arrival model since the former is an easier setting; e.g. see [Meh13, Theorem 2.1] for an explanation. Under the easier known i.i.d. model, the current state of the art algorithms achieve a competitive ratio of 0.7299 using linear programming approches [JL14, BSSX20].

9.3.2 Impossibility under adversarial arrivals

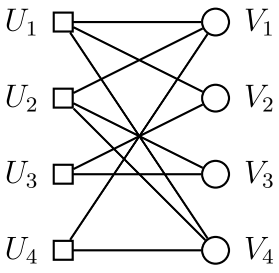
Our construction is based on generalizing the gadget in Fig. 9.1 such that the two graphs are indistinguishable from the first $n/2$ arrivals. Then since any advice is no stronger than the 1-bit advice of whether the online graph is $\mathcal{G}_1$ or $\mathcal{G}_2$, any 1-consistent algorithm has to "blindly trust" the advice and matching according to predicted graph $\mathcal{G}_i$ for $i \in \{1, 2\}$ in the first $n/2$ arrivals. However, if the graph was $\mathcal{G}_{(i+1) \mod 2}$ instead, then this algorithm will not be able to match the remaining $n/2$ arrivals and thus suffer a robustness of $1/2$.

9.3.3 TestAndMatch: Achievability under random arrivals

Using realized type counts as advice

Given the final offline graph $\mathcal{G}^* = (\boldsymbol{U} \cup \boldsymbol{V}, \boldsymbol{E})$ with maximum matching size $n^* \leq n$, we define the vector $\boldsymbol{c}^* \in \mathbb{N}^{2^n}$ indexed by the possible types $2^{\boldsymbol{U}}$ such that $\boldsymbol{c}^*(\boldsymbol{T})$ is the number of times type $\boldsymbol{T} \in 2^{\boldsymbol{U}}$ occurs in $\mathcal{G}^*$. Even though there are 2^n possible types, the number of *realized* types is at most n. Let $\mathcal{T}^* \subseteq 2^{\boldsymbol{U}}$ be the set of types with non-zero counts in $\boldsymbol{c}^*$. Since $|\boldsymbol{U}| = |\boldsymbol{V}| = n$, $\boldsymbol{c}^*$ is sparse and contains $r^* = |\mathcal{T}^*| \leq n \ll 2^n$ non-zero elements; see Fig. 9.2. Note that $\boldsymbol{c}^*$ fully determines $\mathcal{G}^*$ for our purposes, as vertices may be permuted but n^* remains identical.

As mentioned earlier, we consider advice to be an estimate of the *realized type counts* $\widehat{\boldsymbol{c}} \in \mathbb{N}^{2^n}$ with non-zero entries in $\widehat{\mathcal{T}} \subseteq 2^{\boldsymbol{U}}$. As before, we assume that $\widehat{\boldsymbol{c}}$ sums to n and contains $\widehat{r} = |\widehat{\mathcal{T}}| \leq n \ll 2^n$ non-zero entries. Just like $\boldsymbol{c}^*$, the vector $\widehat{\boldsymbol{c}}$ fully defines some "advice graph" $\widehat{\mathcal{G}} = (\boldsymbol{U} \cup \boldsymbol{V}, \widehat{\boldsymbol{E}})$ that we can find a maximum matching for in polynomial



Type counts $\boldsymbol{c}^*$ in $\mathcal{G}^*$

	Type	Count
$\mathcal{T}^*$	$\{U_1, U_3\}$	1
	$\{U_2, U_3\}$	1
	$\{U_1, U_2, U_4\}$	2
$2^{\boldsymbol{U}} \backslash \mathcal{T}^*$	$\vdots$	0

Figure 9.2: For $n = 4$, there may be $2^4 = 16$ possible types but at most $n = 4$ of them can ever be non-zero. Here, $\boldsymbol{c}^*(\{U_1, U_3\}) = 1$, $\boldsymbol{c}^*(\{U_2, U_3\}) = 1$ and $\boldsymbol{c}^*(\{U_1, U_2, U_4\}) = 2$. We see that type $\{U_1, U_2, U_4\}$ appears twice in $\boldsymbol{c}^*$ and $|\mathcal{T}^*| = 3$.

time. We discuss the practicality of obtaining such advice in Appendix C.1.2.

The intuition behind TestAndMatch is as follows. If $\widehat{\boldsymbol{c}} = \boldsymbol{c}^*$, one trivially obtains a 1-consistency by solving for a maximum matching $\widehat{\boldsymbol{M}}$ on the advice graph $\widehat{\mathcal{G}}$ and then mimicking matches based on $\widehat{\boldsymbol{M}}$ as vertices arrive. While $\widehat{\boldsymbol{c}} \neq \boldsymbol{c}^*$ in general, we may consider distributions $\mathcal{P}^* = \boldsymbol{c}^*/n$ and $\mathcal{Q} = \widehat{\boldsymbol{c}}/n$ and test if $\mathcal{P}^*$ is close to $\mathcal{Q}$ in ℓ_1 distance via Theorem 2.37. This is can done sample efficiently using just the first $o(n)$ online vertices; see Section 9.5.2. If $\ell_1(\mathcal{P}^*, \mathcal{Q})$ is less than some threshold τ, we conclude $\widehat{\boldsymbol{c}} \approx \boldsymbol{c}^*$ and continue mimicking $\widehat{\boldsymbol{M}}$, enjoying a competitive ratio close to 1. If not, we revert to Baseline. Crucially, each wrong match made during the testing phase hurts our final matching size by at most a constant, yielding a competitive ratio of $\beta \cdot (1 - o(1))$.

9.4 Impossibility for adversarial arrival model

Here, we give the proof of our impossibility result Theorem 9.3.

Theorem 9.3. *For even n, there exists input graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ such that no advice can distinguish between the two within $n/2$ online arrivals. Consequently, an algorithm* cannot *be both 1-consistent and strictly more than $1/2$-robust.*

Proof. Consider the restricted case where there are only two possible final offline graphs $\mathcal{G}^{(1)} = (\boldsymbol{U} \cup \boldsymbol{V}^{(1)}, \boldsymbol{E}^{(1)})$ and $\mathcal{G}^{(2)} = (\boldsymbol{U} \cup \boldsymbol{V}^{(2)}, \boldsymbol{E}^{(2)})$ where

$$\boldsymbol{E}^{(1)} = \left\{ \{U_j^{(1)}, V_j^{(1)}\}, \{U_{j+n/2}^{(1)}, V_j^{(1)}\} : 1 \leq j \leq n/2 \right\} \\ \cup \left\{ \{U_{j-n/2}^{(1)}, V_j^{(1)}\} : n/2 + 1 \leq j \leq n \right\}$$

$$\boldsymbol{E}^{(2)} = \left\{ \{U_j^{(2)}, V_j^{(2)}\}, \{U_{j+n/2}^{(2)}, V_j^{(2)}\} : 1 \leq j \leq n/2 \right\} \\ \cup \left\{ \{U_j^{(2)}, V_j^{(2)}\} : n/2 + 1 \leq j \leq n \right\}$$

We will even restrict the first $n/2$ to be exactly $V_1^{(i)}, \ldots, V_{n/2}^{(i)}$, where $i \in \{1,2\}$ is the chosen input graph by the adversary. See Fig. 9.3 for an illustration.

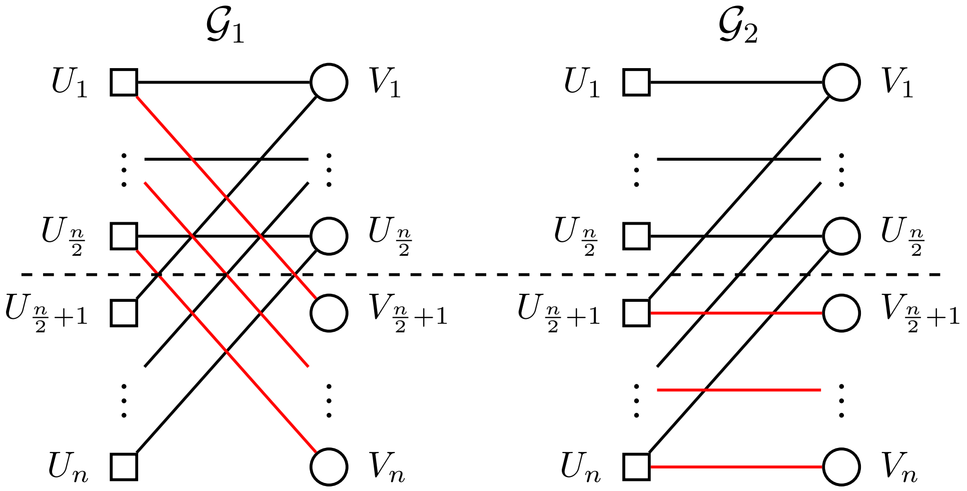


Figure 9.3: Illustration of $\mathcal{G}_1$ and $\mathcal{G}_2$ for Theorem 9.3

Suppose $\mathcal{G}_i$ was the chosen graph, for $i \in \{1,2\}$. In this restricted problem input setting, the strongest possible advice is knowing the bit i since all other viable advice can be derived from this bit. Thus, for the sake of a hardness result, it suffices to only consider the advice of $\widehat{i} \in \{1,2\}$.

Within the first $n/2$ arrivals, any algorithm cannot distinguish between $\mathcal{G}_1$ and $\mathcal{G}_2$, and will behave in the same manner. Suppose there is a 1-consistent algorithm $\mathcal{A}$ given bit $\widehat{i}$. In the first $n/2$ steps, since $\mathcal{A}$ is 1-consistent, $\mathcal{A}$ needs to match V_j to $U_{j+n/2}$ if $\widehat{i} = 1$ and V_j to U_j for $\widehat{i} = 2$. However, if $i \neq \widehat{i}$, then $\mathcal{A}$ will not be able to match any remaining arrivals and hence be at most $1/2$-robust. □

9.5 TestAndMatch for random arrival model

In this section, we present our learning-augmented algorithm TestAndMatch which is 1-consistent, $(\beta - o(1))$-robust, and achieves a smooth interpolation on an appropriate notion of advice quality, where β is any achieveable competitive ratio by some advice-free baseline algorithm. As discussed in Section 9.3.1, the best known competitive ratio of $\beta = 0.696$ is achieveable using Ranking [KVV90] but it is unknown if it can be improved. As TestAndMatch is a meta-algorithm that uses any advice-free baseline algorithm as a black-box, one may choose to treat Baseline as Ranking for concreteness.

TestAndMatch is described in Algorithm 17, which takes as input a number of additional parameters (δ, ϵ, etc) and subroutines that we will explain in a bit. For now, we formalize Theorem 9.4 in the context of Algorithm 17.

Algorithm 17 TestAndMatch

```
    Input: Advice $\widehat{\boldsymbol{c}}$ with $\widehat{r} = |\widehat{\mathcal{T}}|$, Baseline advice-free algorithm with competitive ratio
    $\beta < 1$, error threshold $\varepsilon > 0$, failure rate $\delta = \delta' + \delta_{poi}$ for $\delta_{poi} \in \mathcal{O}\left(\frac{1}{\text{poly}(\widehat{r})}\right)$
 1: Compute advice matching $\widehat{\boldsymbol{M}}$ from $\widehat{\boldsymbol{c}}$
 2: if $\frac{\widehat{n}}{n} \leq \beta$ then
 3:     Run Baseline on all arrivals
 4: Define $s_{\widehat{r},\varepsilon,\delta} \in \mathcal{O}\left(\frac{(\widehat{r}+1)\cdot\log(1/\delta')}{\varepsilon^2\cdot\log(\widehat{r}+1)}\right)$
 5: Define testing threshold $\tau = 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon$
 6: Run Mimic on $s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)}$ arrivals while tracking online arrivals in a set $\mathcal{A}$
 7: if MinimaxTest$(s_{\widehat{r},\varepsilon,\delta}, \frac{\widehat{c}}{n}, \mathcal{A}, \tau, \delta')$ outputs OK then
 8:     Run Mimic on the remaining arrivals
 9: else
10:     Run Baseline on the remaining arrivals
```

Theorem 9.5. *For any advice $\widehat{\boldsymbol{c}}$ with $|\widehat{\mathcal{T}}| = \widehat{r}$, $\varepsilon > 0$ and $\delta > \frac{1}{\text{poly}(\widehat{r})}$, let $\widehat{\ell_1}$ be the estimate of $\ell_1(\mathcal{P}^*, \mathcal{Q} = \frac{\widehat{c}}{n})$ obtained from $k = s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)}$ i.i.d. samples of $\mathcal{P}^*$. With success probability $\geq 1 - \delta$, TestAndMatch produces a matching with competitive ratio at least $\frac{\widehat{n}}{n} - \frac{\ell_1(\mathcal{P}^*,\mathcal{Q})}{2} \geq \beta$ when $\widehat{\ell_1} \leq 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon$, and at least $\beta \cdot (1 - \frac{k}{n})$ otherwise.*

Let m be the size of the produced matching. For sufficiently large n and constants (ε, δ), we have $s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)} \in o(1)$, so Theorem 9.5 implies a lower bound on the achieved competitive ratio of $\frac{m}{n^*}$ (see Fig. 9.4) where

$$\frac{m}{n^*} \geq \frac{m}{n} \geq \begin{cases} \frac{\widehat{n}}{n} - \frac{\ell_1(\mathcal{P}^*,\mathcal{Q})}{2} & \text{when } \widehat{\ell_1} \leq 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon \\ \beta \cdot (1 - o(1)) & \text{otherwise} \end{cases}$$

Under random order arrivals, the competitive ratio is measured in expectation over all possible arrival sequences; see Eq. (9.2). One can easily convert the guarantees of Theorem 9.5 to one in expectation by assuming the extreme worst case scenario of obtaining 0 matches whenever the tester fails. So, the expected competitive ratio is simply $(1 - \delta)$ factor of the bounds given in Theorem 9.5. Setting $\delta = 0.001$, we get a robustness guarantee of $\beta \cdot (1 - o(1)) \cdot 0.999$ in expectation. Note that our guarantees hold regardless of what value of ε is used. In the event that a very small ε is chosen and the test always fails, we are still guaranteed the robustness guarantees of $\approx \beta$. One possible default for ε could be to assume that the optimal offline matching has size n and just set it to half the threshold value, i.e. set $\varepsilon = \frac{\widehat{n}}{n} - \beta$.

Remark 9.6 (Lines 4 and 6 in TestAndMatch). As we subsequently require $\text{Poi}(s_{\widehat{r},\varepsilon,\delta})$ i.i.d. samples from $\mathcal{P}^*$ for testing, we collect $s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)}$ online arrivals into the set $\mathcal{A}$. Note that $\mathbb{E}[\text{Poi}(s_{\widehat{r},\varepsilon,\delta})] = s_{\widehat{r},\varepsilon,\delta}$ and $\text{Poi}(s_{\widehat{r},\varepsilon,\delta}) \leq s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)}$ with high probability. This additional slack of $\sqrt{\log(\widehat{r}+1)}$ allows for Theorem 9.5 to hold with high probability (as opposed to constant) while ensuring that the competitive ratio remains

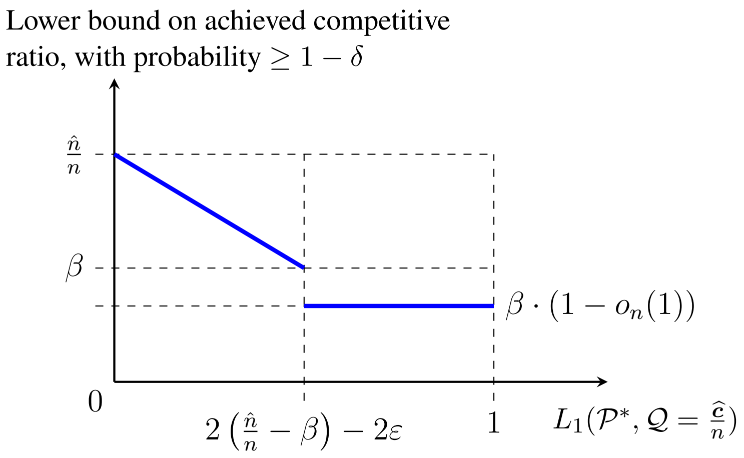


Figure 9.4: A (conservative) competitive ratio plot for $\frac{\widehat{n}}{n} > \beta$. If MinimaxTest (Algorithm 20) succeeds, we have $\ell_1(\mathcal{P}^*, \mathcal{Q}) < 2\left(\frac{\widehat{n}}{n} - \beta\right) - 2\varepsilon$ whenever $\widehat{\ell_1} < 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon$. Observe that there is a smooth interpolation between the achieveable competitive ratio as $\ell_1(\mathcal{P}^*, \mathcal{Q})$ degrades whilst paying only $o(1)$ for robustness.

in the $\beta \cdot (1 - o(1))$ regime. Finally, when $r \in \Omega(n)$, we remark that $s_{\widehat{r},\varepsilon,\delta}$ is sublinear in n only for sufficiently large n; see Section 9.6 for some practical modifications.

The rest of this section is devoted to describing TestAndMatch in greater detail and formally proving Theorem 9.5. We study in Section 9.5.1 how mimicking poor advice quality impacts matching sizes, yielding conditions where mimicking is desirable, which we test for via Theorem 2.37. Section 9.5.2 describes transformations to massage our problem into the form required by Theorem 2.37. Lastly, we tie up our analysis of Theorem 9.5 in Section 9.5.3.

9.5.1 Effect of advice quality on matching sizes

Algorithm 18 Mimic

```
Input: Matching $\widehat{\boldsymbol{M}}$, advice counts $\widehat{\boldsymbol{c}}$, arrival type label $\boldsymbol{T}$
if $\boldsymbol{c}(\boldsymbol{T}) > 0$ then
    Mimic an arbitrary unused type $\boldsymbol{T}$ match in $\widehat{\boldsymbol{M}}$
    Decrement $\boldsymbol{c}(\boldsymbol{T})$ by 1
return $\boldsymbol{c}$                                        ▷ Updated counts
```

Given an advice $\widehat{\boldsymbol{c}}$ of type counts, we first solve optimally for a maximum matching $\widehat{\boldsymbol{M}}$ on the advice graph $\widehat{\mathcal{G}}$ and then mimic the matches for online arrivals whenever possible; see Algorithm 18. That is, whenever new vertices arrive, we match according to some unused vertex of the same type if possible and leave it unmatched otherwise.

Let us normalize counts into proper distributions $\mathcal{P}^* = \boldsymbol{c}^*/n$ and $\mathcal{Q} = \widehat{\boldsymbol{c}}/n$. These are distributions on the realized and predicted (by advice) counts, and have sparse support

$\mathcal{T}^*$ and $\widehat{\mathcal{T}}$. Now, suppose $\widehat{\boldsymbol{M}}$ has matching size $\widehat{n} \leq n$. By definition of $\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}})$ and Mimic, one would obtain a matching of size at least $\widehat{n} - \ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}})/2$ by blindly following advice. This yields a competitive ratio of $\frac{\widehat{n} - \ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}})/2}{n^*}$. Rearranging, we see that Mimic outperforms the advice-free baseline (in terms of worst case guarantees) if and only if

$$\frac{\widehat{n} - \ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}})/2}{n^*} > \beta \iff \ell_1(\mathcal{P}^*, \mathcal{Q}) < \frac{2}{n}\left(\widehat{n} - \beta n^*\right)$$

The above analysis suggests a natural way to use advice type counts: use Mimic if $\ell_1(\mathcal{P}^*, \mathcal{Q}) \leq \frac{2}{n}\left(\widehat{n} - \beta n^*\right)$, and Baseline otherwise. Note that one should always just use Baseline whenever $\frac{\widehat{n}}{n^*} < \beta$, matching the intuition of ignoring advice of poor quality. Unfortunately, as we only know n but not n^*, our algorithm can only check whether $\ell_1(\mathcal{P}^*, \mathcal{Q}) < 2\left(\frac{\widehat{n}}{n} - \beta\right)$, and so the resulting guarantee is *conservative* since $n^* \leq n$.

9.5.2 Estimating advice quality via property testing

As $\boldsymbol{c}^*$ is unknown, we cannot obtain $\ell_1(\mathcal{P}^* = \frac{\boldsymbol{c}^*}{n}, \mathcal{Q} = \frac{\widehat{\boldsymbol{c}}}{n})$. However, as $\mathcal{P}^*$ and $\mathcal{Q}$ are proper distributions, we can apply the property testing method of Theorem 2.37 to estimate $\ell_1(\mathcal{P}^*, \mathcal{Q})$ to some $\varepsilon > 0$ accuracy. Applying Theorem 2.37 raises two difficulties.

Simulating i.i.d. arrivals

Under the Random Order arrival model, online vertices arrive "without replacement", which is incompatible with Theorem 2.37. Thankfully, we can apply a standard trick to simulate i.i.d. "sampling with replacement" from $\mathcal{P}^*$ by "re-observing arrivals".

Algorithm 19 SimulateP

```
Input: Collection $\mathcal{A}$ of arrivals and number of desired i.i.d. samples $s$, where $s \leq |\mathcal{A}|$
Output: Collection $\mathcal{T}^s_{\mathcal{P}^*}$ of types                ▷ $s$ i.i.d. samples from $\mathcal{P}^*$
$\mathcal{T}^s_{\mathcal{P}^*} \leftarrow \emptyset$                       ▷ Collect simulated i.i.d. arrivals from $\mathcal{P}^*$
$i \leftarrow 0$
while $|\mathcal{T}^s_{\mathcal{P}^*}| < s$ do
    if $\mathrm{Bern}(i/n) == 1$ then                                       ▷ Biased coin flip with probability $i/n$
        $\boldsymbol{X} \leftarrow$ Pick uniformly at random from the set $\{\mathcal{A}[0], \ldots, \mathcal{A}[i-1]\}$
    else
        $\boldsymbol{X} \leftarrow \mathcal{A}[i]$                  ▷ i.i.d. sample from $\mathcal{P}^*$ under the random arrival model
        $i \leftarrow i + 1$
    Add $\boldsymbol{X}$ to $\mathcal{T}^s_{\mathcal{P}^*}$
return $\mathcal{T}^s_{\mathcal{P}^*}$
```

Lemma 9.7. *In the output of* SimulateP *(Algorithm 19), $\mathcal{T}^s_{\mathcal{P}^*}$ contains s i.i.d. samples from the realized type count distribution $\mathcal{P}^*$ while using at most s actual online arrivals.*

Proof. With probability i/n, we choose a uniform at random item from $\{\mathcal{A}[0], \ldots, \mathcal{A}[i-1]\}$. With probability $1 - i/n$, we pick the next item $\mathcal{A}[i]$ from the existing arrivals which was uniform at random under the random arrival model assumption. Since we could possibly reuse arrivals, $\mathcal{T}^{s}_{\mathcal{P}^*}$ is formed by using at most s fresh arrivals. □

Operating in reduced domains

Strictly speaking, the domain of $\mathcal{P}^*$ and $\mathcal{Q}$ could be as large as 2^n, since any one of these types may occur. If all of these types occur with non-zero probability, then applying Theorem 2.37 for testing could take a near-exponential (in n) number of online vertex arrivals, which is clearly impossible. However, as established earlier, $\mathcal{P}^*$ and $\mathcal{Q}$ enjoy sparsity; in particular, $\widehat{\boldsymbol{c}}$ and thus $\mathcal{Q}$ contain 0 in all but at most $\widehat{r} = |\widehat{\mathcal{T}}| \leq n \ll 2^n$ entries. The key insight is to express ℓ_1 distances by operating on $\widehat{\mathcal{T}}$, plus an additional dummy type $\boldsymbol{T}_0$ which has 0 counts in $\widehat{\boldsymbol{c}}$. We classify any online vertex with type $\boldsymbol{T} \in \mathcal{T}^* \setminus \widehat{\mathcal{T}}$ as type $\boldsymbol{T}_0$. Specifically,

$$\begin{aligned}\ell_1(\mathcal{P}^*, \mathcal{Q}) = \sum_{\boldsymbol{T} \in 2^U} |\mathcal{P}^*(\boldsymbol{T}) - \mathcal{Q}(\boldsymbol{T})| &= \sum_{\boldsymbol{T} \in \widehat{\mathcal{T}} \cup T^*} |\mathcal{P}^*(\boldsymbol{T}) - \mathcal{Q}(\boldsymbol{T})| \\ &= \sum_{\boldsymbol{T} \in \widehat{\mathcal{T}}} |\mathcal{P}^*(\boldsymbol{T}) - \mathcal{Q}(\boldsymbol{T})| + \sum_{\boldsymbol{T} \in \mathcal{T}^* \setminus \widehat{\mathcal{T}}} \mathcal{P}^*(\boldsymbol{T})\end{aligned}$$

That is, we can view $\ell_1(\mathcal{P}^*, \mathcal{Q})$ as an ℓ_1 distance on distributions with support $\widehat{\mathcal{T}} \cup \{\boldsymbol{T}_0\}$. Thus, the domain size when applying Theorem 2.37 is $\widehat{r} + 1 \leq n + 1$. For any constants $\epsilon > 0$ and $\delta > 0$, the required samples is then $s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)} \in o(n)$.

Property testing

Now that these difficulties are overcome, the estimation of $\ell_1(\mathcal{P}^*, \mathcal{Q}) = \ell_1(\frac{\boldsymbol{c}^*}{n}, \frac{\widehat{\boldsymbol{c}}}{n})$ is done via MinimaxTest (Algorithm 20), whose correctness follows from Theorem 2.37.

Lemma 9.8. *Given $s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)}$ online arrivals in a set $\mathcal{A}$ and threshold $\tau = 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon$, we have $\ell_1(\mathcal{P}^*, \mathcal{Q}) < 2\left(\frac{\widehat{n}}{n} - \beta\right)$ whenever MinimaxTest (Algorithm 20) outputs* OK*. The success probability of MinimaxTest is at least $1 - \delta$.*

Proof. The algorithm of Theorem 2.37 guarantees tells us that $|\widehat{\ell}_1 - \ell_1(\mathcal{P}^*, \mathcal{Q})| \leq \varepsilon$ with probability at least $1 - \delta'$. Therefore, when MinimaxTest outputs OK, we are guaranteed that $\widehat{\ell}_1 < \tau$. That is, $\ell_1(\mathcal{P}^*, \mathcal{Q}) \leq \widehat{\ell}_1 + \varepsilon < \tau + \varepsilon = 2\left(\frac{\widehat{n}}{n} - \beta\right)$.

Meanwhile, in the analysis of Theorem 2.37, one actually needs to use $s_1 + s_2$ i.i.d. samples from $\mathcal{P}^*$, where $s_1, s_2 \sim \mathrm{Poi}(s_{\widehat{r},\varepsilon,\delta'})$, which can be simulated from the arrival set $\mathcal{A}$; see SimulateP (Algorithm 19). By Lemma 2.36, we may assume that $s_1 + s_2 \leq s$ with probability at least $1 - \delta_{poi}(s)$. Taking a union bound over the failure probability of

Algorithm 20 MinimaxTest

```
Input: Sample size $s$, distribution $\mathcal{Q} = \widehat{\mathbf{c}}/n$, $s\sqrt{\log(\widehat{r}+1)}$ online arrivals $\mathcal{A}$, testing threshold $\tau$, failure rate $\delta'$
Compute $s_1, s_2 \sim \mathrm{Poi}(s/2)$
if $s_1 + s_2 > s\sqrt{\log(\widehat{r}+1)}$ then        ▷ Occurs with probability $\delta_{poi} \leq 1/\mathrm{poly}(\widehat{r})$
    return Fail
Collect $s_1 + s_2$ i.i.d. samples from $\mathcal{P}^* = \frac{c^*}{n}$ by running SimulateP with $\mathcal{A}$.
Run algorithm of Theorem 2.37 to obtain estimate $\widehat{\ell}_1$ such that $|\widehat{\ell}_1 - \ell_1(\mathcal{P}^*, \mathcal{Q})| \leq \varepsilon$ with probability $1 - \delta'$
if $\widehat{\ell}_1 < \tau$ then
    return OK
else
    return Fail
```

the "Poissonization" event and the estimator, we see that the overall success probability is at least $1 - (\delta' + \delta_{poi}) = 1 - \delta$; note that $\delta = \delta' + \delta_{poi}$ is defined in TestAndMatch. □

9.5.3 Tying up our analysis of TestAndMatch

If we run Baseline from the beginning due to $\frac{\widehat{n}}{n} \leq \beta$, then we trivially recover a β-competitive ratio. The following lemma gives a lower bound on the obtained matching size if we performed MinimaxTest but decided switch to Baseline due to the estimated $\widehat{\ell}_1$ being too large.

Lemma 9.9. *Suppose we run an arbitrary algorithm for the first $k \in [n]$ online arrivals and then switch to Baseline for the remaining $n - k$ online arrivals. If j matches made in the first k arrivals, where $0 \leq j \leq k$, then the overall produced matching size is at least $\beta \cdot (n - k - j) + j$.*

Proof. Any match made in the first k arrivals decreases the maximum attainable matching size by at most two, *excluding the match made*. As the maximum attainable matching size was originally n, the maximum attainable matching size on the postfix sequence after the k is at least $n - k - j$. Since Baseline has competitive ratio β, running Baseline on the remaining $n - k$ steps will produce a matching of size at least $\beta \cdot (n - k - j)$. Thus, the overall produced matching size is at least $\beta \cdot (n - k - j) + j$. □

The proof of Theorem 9.5 requires the following lemma.

Lemma 9.10. *For any advice $\widehat{\mathbf{c}}$ with $|\widehat{\mathcal{T}}| = \widehat{r}$, $\varepsilon > 0$ and $\delta > \frac{1}{\mathrm{poly}(\widehat{r})}$, let $\widehat{\ell}_1$ be the estimate of $\ell_1(\mathcal{P}^*, \mathcal{Q} = \frac{\widehat{\mathbf{c}}}{n})$ in MinimaxTest. If MinimaxTest succeeds, then TestAndMatch produces a matching with competitive ratio at least $\frac{\widehat{n}}{n} - \frac{\ell_1(\mathcal{P}^*, \mathcal{Q})}{2} \geq \beta$ when $\widehat{\ell}_1 \leq 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon$, and at least $\beta \cdot (1 - \frac{s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)}}{n})$ otherwise.*

Proof. Let m be the size of the produced matching with competitive ratio $\frac{m}{n^*}$. Since MinimaxTest succeeds, $|\widehat{\ell}_1 - \ell_1(\mathcal{P}^*, \mathcal{Q})| \leq \varepsilon$ and TestAndMatch executed Mimic for all online arrivals if $\widehat{\ell}_1 < \tau = 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon$, and switches to Baseline after an initial batch of $k = s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)}$ otherwise. We consider each case separately.

Case 1: $\widehat{\ell}_1 \leq 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon$

TestAndMatch executed Mimic for all online arrivals, yielding a matching of size $m \geq \widehat{n} - \frac{\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}})}{2}$. Since $|\widehat{\ell}_1 - \ell_1(\mathcal{P}^*, \mathcal{Q})| \leq \varepsilon$, we see that $\ell_1(\mathcal{P}^*, \mathcal{Q}) \leq \widehat{\ell}_1 + \varepsilon \leq 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon + \varepsilon = 2\left(\frac{\widehat{n}}{n} - \beta\right)$. Therefore,

$$\frac{m}{n^*} \geq \frac{m}{n} \geq \frac{\widehat{n}}{n} - \frac{\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}})}{2n} = \frac{\widehat{n}}{n} - \frac{\ell_1(\mathcal{P}^*, \mathcal{Q})}{2} \geq \beta$$

Case 2: $\widehat{\ell}_1 > 2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon$

TestAndMatch executes Baseline after an initial batch of $k = s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)}$ arrivals that follow Mimic. Suppose we made j matches via Mimic before MinimaxTest. Then, Lemma 9.9 tells us that the overall produced matching size is at least $m \geq \beta \cdot (n - k - j) + j$. Since $\beta < 1$, we have $\beta \cdot (n - k - j) + j \geq \beta \cdot (n - k)$. Therefore,

$$\frac{m}{n^*} \geq \frac{m}{n} \geq \frac{\beta \cdot (n - s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)})}{n} = \beta \cdot \left(1 - \frac{s_{\widehat{r},\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r}+1)}}{n}\right) \qquad \square$$

Theorem 9.5 follows from bounding the failure probability.

Proof of Theorem 9.5. Whenever MinimaxTest succeeds, the competitive ratio guarantees follow directly from Lemma 9.10. Therefore, it only remains to bound the failure probability, i.e. the probability that MinimaxTest (Algorithm 20) fails. This can happen if either line 3 is executed (event $\mathcal{E}_1$) or the algorithm in line 5 fails (event $\mathcal{E}_2$).

The event $\mathcal{E}_1$ occurs when the one of the Poisson random variables in line 1 exceed the expectation by a $\sqrt{\log \widehat{r}}$ factor. Since $s_1, s_2 \sim \text{Poi}(s/2)$, we have that $(s_1 + s_2) \sim \text{Poi}(s)$. Thus, by Lemma 2.36 we have that:

$$\begin{aligned} \delta_{poi} &= \Pr\left[|(s_1 + s_2) - s| > s\sqrt{\log \widehat{r}}\right] \\ &\leq 2\exp\left(-\frac{s^2 \log \widehat{r}}{2(s + s\sqrt{\log \widehat{r}})}\right) \in \mathcal{O}\left(\widehat{r}^{-\frac{s}{2(1+\sqrt{\log \widehat{r}})}}\right) \subseteq \mathcal{O}\left(\frac{1}{\text{poly}(\widehat{r})}\right) \end{aligned}$$

for the value of $s \in \mathcal{O}\left(\frac{(\widehat{r}+1)\cdot\log(1/\delta')}{\varepsilon^2 \cdot \log(\widehat{r}+1)}\right)$ chosen. Meanwhile, $\mathcal{E}_2$ occurs with probability δ' by choice of parameters while invoking Theorem 2.37. Combining the above with Lemma 9.8 via union bound yields a total failure rate of at most $\Pr(\mathcal{E}_1) + \Pr(\mathcal{E}_2) \leq \delta_{poi} + \delta' = \delta$. $\square$

9.6 Practical considerations

While our contributions are mostly theoretical, we discuss some practical considerations here. In particular, we would like to highlight that there is no existing practical implementation of the algorithm of Theorem 2.37 by [JHW18]. As is the case for most state-of-the-art distribution testing algorithms, this implementation is highly non-trivial and requires the use of optimal polynomial approximations over functions, amongst other complicated constructions. The tester proposed by [JHW18] requires a significant amount of hyperparameter tuning and no off-the-shelf implementation is available [Han24]. For completeness, we implemented a proof-of-concept based on the empirical ℓ_1 estimation in Appendix C.1.5. While it is known that the estimation error scales with the sample size in the form $\Omega(r/\varepsilon^2)$, we observe good empirical performance when r is sublinear in n or when combined with some of the practical extensions that we discussed below.

Section 9.6.1 and Section 9.6.2 can be viewed as ways to extend the usefulness of a given advice. Section 9.6.3 provides a way to "patch" an advice with $\widehat{n} < n$ to one with perfect matching, without hurting the provable guarantees. Section 9.6.4 gives a pre-processing step that can be prepended to any procedure: by losing $o(1)$, one can test whether $|\mathcal{T}^*|$ is small and if so learn $\mathcal{P}^*$ up to ε error to fully exploit it.

9.6.1 Remapping online arrival types

Consider the graph example in Fig. 9.2 with type counts $\boldsymbol{c}^*$ and we are given some advice count $\widehat{\boldsymbol{c}}$ as follows:

Types $\mathcal{T}^*$	Type counts $\boldsymbol{c}^*$	Types $\widehat{\mathcal{T}}$	Type counts $\widehat{\boldsymbol{c}}$
$\{U_1, U_3\}$	1	$\{U_1\}$	1
$\{U_2, U_3\}$	1	$\{U_3\}$	1
$\{U_1, U_2, U_4\}$	2	$\{U_4\}$	1
		$\{U_2, U_4\}$	1

While one can verify that both the true graph $\mathcal{G}^*$ and the advice graph $\widehat{\mathcal{G}}$ have perfect matching, $\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}) = 4$ since as $\mathcal{T}^*$ and $\widehat{\mathcal{T}}$ have disjoint types. Using our earlier analysis, $\widehat{\boldsymbol{c}}$ would be deemed as a poor quality advice and one should default to BASELINE.

However, a closer look reveals there exists a mapping σ from $\mathcal{T}^*$ to $\widehat{\mathcal{T}}$ such that one can credibly "mimic" the proposed matching of $\widehat{\mathcal{G}}$ as online vertices arrive. For example, when an online vertex V with neighborhood type $\{U_1, U_3\}$ arrive, one can "ignore" the edge $U_3 - V$ and treat it as if V had the type $\{U_1\}$. Similarly, $\{U_2, U_3\}$ could be treated as $\{U_3\}$, the first instance of $\{U_1, U_2, U_4\}$ could be treated as $\{U_2, U_4\}$, and the second instance of $\{U_1, U_2, U_4\}$ could be treated as $\{U_4\}$. Running MIMIC under such a remapping of online types would then produce a perfect matching! Note that the proposed remappings always maps an online type to a *subset* so that any subsequent proposed matching can be credibly performed.

In an offline setting, given $\boldsymbol{c}^*$ and $\widehat{\boldsymbol{c}}$, one can efficiently compute a mapping σ that maximizes overlap using a max-flow formulation (see Appendix C.1.3) and then redefine the quality of $\widehat{\boldsymbol{c}}$ in terms of $\ell_1(\sigma(\boldsymbol{c}^*), \widehat{\boldsymbol{c}})$. As this is impossible in an online setting, we propose a following simple mapping heuristic: when type $\boldsymbol{L}$ arrives, map it to the largest subset type $\boldsymbol{A} \subseteq \boldsymbol{L}$ with the highest remaining possible match count. Note that it may be the case that all subset types of $\boldsymbol{L}$ no longer have a matching available to mimic from $\widehat{\boldsymbol{M}}$. In the example above, we first mapped $\{U_1, U_2, U_4\}$ to $\{U_2, U_4\}$ and then to $\{U_4\}$ as $\widehat{\boldsymbol{c}}$ only had one count for $\{U_2, U_4\}$.

9.6.2 Coarsening of advice

While Theorem 9.5 has good asymptotic guarantees as $n \to \infty$, the actual number of vertices n is finite in practice. In particular, when n is "not large enough", TestAndMatch will never utilize the advice and always default to Baseline for all problem instances where $n \ll s_{\widehat{r},\varepsilon,\delta}$.

In practice, while the given advice types may be diverse, there could be many "overlapping subtypes" and a natural idea is to "coarsen" the advice by grouping similar types together in an effort to reduce the resultant support size of the advice (and hence $s_{\widehat{r},\varepsilon,\delta}$). Fig. 9.5 illustrates an extreme example where we could decrease the support size from n to 2 while still maintaining a perfect matching.

While one could treat this coarsening subproblem as an optimization pre-processing task. For completeness, we show in Appendix C.1.4 how one may potentially model the coarsening optimization as an integer linear program (ILP) but remark that it does not scale well in practice. That said, there are many natural scenarios where a coarsening is readily available to us. For instance, in the online advertising, market studies typically classify users into "types" (with the number of types significantly less than n) where each type of user typically have a "core set" of suitable ads though the actual realized type of each arrival may be perturbed due to individual differences.

Another way to reduce the required samples for testing is to "bucket" the counts which are below a certain threshold to reduce the number of distinct types within the advice. The newly created bucket type will then be a union of the types that are being grouped together.

9.6.3 Advice does not have perfect matching

As the given advice $\widehat{\boldsymbol{c}}$ is arbitrary, it could be the case that any maximum matching of size $\widehat{n}$ in the graph $\widehat{\mathcal{G}}$ implied by $\widehat{\boldsymbol{c}}$ is not perfect, i.e. $\widehat{n} < n$. A natural idea would be to "patch" $\widehat{\boldsymbol{c}}$ into some other type count $\widehat{\boldsymbol{c}}'$ which has a maximum matching size of $\widehat{n}' = n$ in

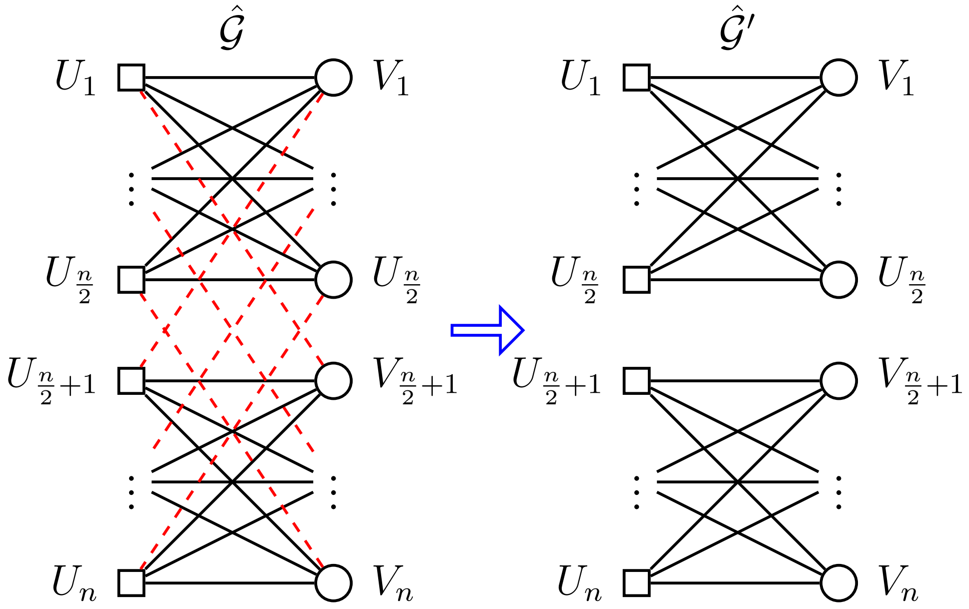


Figure 9.5: Consider $\widehat{\mathcal{G}}$ made by taking the union of two complete bipartite graphs ($\widehat{\mathcal{G}}'$) and adding the red dashed edges. By connecting V_i to $U_{(i+n/2) \bmod n}$, $|\widehat{\mathcal{T}}| = r = n$. Meanwhile, if we coarsen $\widehat{\boldsymbol{c}}$ into $\widehat{\boldsymbol{c}}'$ by ignoring the red dashed edges, $\widehat{\mathcal{G}}'$ still has a maximum matching of size $\widehat{n}' = n$ while $|\widehat{\mathcal{T}}'| = r' = 2$, thus requiring significantly less samples to test since $s_{\widehat{r}',\varepsilon,\delta} \ll s_{\widehat{r},\varepsilon,\delta}$. Furthermore, if $\mathcal{G}^* = \widehat{\mathcal{G}}'$, then $\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}) = 2n$ and we will reject the advice $\widehat{\boldsymbol{c}}$ if we do not coarsen it first.

the tweaked graph $\widehat{\mathcal{G}}'$. This can be done by augmenting $\widehat{\boldsymbol{c}}$ with additional edges between the unmatched vertices in the advice graph to obtain $\widehat{\boldsymbol{c}}'$.

The following lemma tells us that there is an explicit way of augmenting $\widehat{\boldsymbol{c}}$ to form a new advice $\widehat{\boldsymbol{c}}'$ such that using $\widehat{\boldsymbol{c}}'$ in TestAndMatch does not hurt the provable theoretical guarantees as compared to directly using $\widehat{\boldsymbol{c}}$.

Lemma 9.11. *Let $\widehat{\boldsymbol{c}}$ be an arbitrary type count with labels $\widehat{\mathcal{T}}$ implying a graph $\widehat{\mathcal{G}}$ with maximum matching size $\widehat{n}$. There is an explicit way to augment $\widehat{\boldsymbol{c}}$ to obtain $\widehat{\boldsymbol{c}}'$ with labels $\widehat{\mathcal{T}}'$ such that the implied graph $\widehat{\mathcal{G}}'$ has maximum matching size $\widehat{n}' = n$. Furthermore, running TestAndMatch with a slight modification of Mimic on $(\widehat{\boldsymbol{c}}', \widehat{\mathcal{T}}')$ produces a matching of size m where*

$$\frac{m}{n^*} \geq \frac{m}{n} \geq \begin{cases} \frac{\widehat{n}}{n} - \frac{\ell_1(\mathcal{P}^*, \mathcal{Q})}{2} & \text{when } \widehat{\ell}_1 \leq 2(1-\beta) - \varepsilon \\ \beta \cdot (1 - o(1)) & \text{otherwise} \end{cases}$$

Proof. Suppose we are given an arbitrary pattern count $\widehat{\boldsymbol{c}}$ and corresponding labels $\widehat{\boldsymbol{L}}$ such that the corresponding graph $\widehat{\mathcal{G}}$ has maximum matching $\widehat{\boldsymbol{M}}$ of size $\widehat{n} < n$. Let us fix any arbitrary maximum matching $\widehat{\boldsymbol{M}}$. Denote $\boldsymbol{A_U} \subseteq \boldsymbol{U}$ as the set of $k = n - \widehat{n}$ offline vertices and $\boldsymbol{A_V} \subseteq \boldsymbol{V}$ as the set of k online vertices that are unmatched in $\widehat{\boldsymbol{M}}$. We construct a new graph $\widehat{\mathcal{G}}'$ by adding a complete bipartite graph of size k on $\boldsymbol{A_U} \cup \boldsymbol{A_V}$ to $\widehat{\mathcal{G}}$. By construction, the resulting graph $\widehat{\mathcal{G}}'$ has a maximum matching of size $\widehat{n}' = n$ due to the modified adjacency patterns of the online vertices $\boldsymbol{A_V}$.

We now explain how to modify the pattern counts and labels accordingly. Define the new set of labels $\widehat{\boldsymbol{L}}'$ as $\widehat{\boldsymbol{L}}$ with a new pattern called "New". Then, we subtract away the counts of $\boldsymbol{A_V}$ from $\widehat{\boldsymbol{c}}$ and add a count of k to the label "New" to obtain a new pattern count $\widehat{\boldsymbol{c}}'$. By construction, we see that $|\widehat{\boldsymbol{L}}'| = |\widehat{\boldsymbol{L}}| + 1$ and

$$\ell_1(\widehat{\boldsymbol{c}}, \widehat{\boldsymbol{c}}') = |\widehat{\boldsymbol{c}}(\text{“New”}) - \widehat{\boldsymbol{c}}'(\text{“New”})| + \sum_{\ell \in \widehat{\boldsymbol{L}}} |\widehat{\boldsymbol{c}}(\ell) - \widehat{\boldsymbol{c}}'(\ell)| = k + k = 2k$$

Note that $\boldsymbol{c}^*(\text{“New”}) = 0$. By triangle inequality, we also see that

$$\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}') \leq \ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}) + \ell_1(\widehat{\boldsymbol{c}}, \widehat{\boldsymbol{c}}') \leq \ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}) + 2k$$

Slight modification of Mimic. Mimic will now be informed of the sets $\boldsymbol{A_U}$ and $\boldsymbol{A_V}$ along with the proposed matching $\widehat{\boldsymbol{M}}$ for the online vertices $\boldsymbol{V} \setminus \boldsymbol{A_V}$. Then, whenever an online vertex V arrives whose pattern does not match any in $\widehat{\boldsymbol{L}}$, we first try to match V to an unmatched neighbor in $\boldsymbol{A_U}$ if possible before leaving it unmatched. Observe that this modified procedure can only increase the number of resultant matches since we do not disrupt any possible matchings under $(\widehat{\boldsymbol{c}}, \widehat{\boldsymbol{L}})$ while only possibly increasing the matching size via the complete bipartite graph between $\boldsymbol{A_U}$ and $\boldsymbol{A_V}$.

To complete the analysis, we again consider whether Mimic was executed throughout the online arrivals or we switched to Baseline, as in the analysis of Theorem 9.5. Note that now $\widehat{\ell_1}$ is an estimate of $\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}')$ instead of $\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}})$ and the threshold is $2\left(\frac{\widehat{n}'}{n} - \beta\right) - \varepsilon = 2(1 - \beta) - \varepsilon$ instead of $2\left(\frac{\widehat{n}}{n} - \beta\right) - \varepsilon$ since $\widehat{n}' = n$. Also, recall that $k = n - \widehat{n}$.

Case 1: $\widehat{\ell_1} < 2(1 - \beta) - \varepsilon$

Then, TestAndMatch executed Mimic throughout for all online arrivals, yielding a matching of size $m \geq n - \frac{\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}')}{2}$. Therefore,

$$\begin{aligned}\frac{m}{n^*} \geq \frac{m}{n} \geq 1 - \frac{\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}')}{2n} &\geq 1 - \frac{\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}) + 2k}{2n} \\ &= 1 - \frac{\ell_1(\mathcal{P}^*, \mathcal{Q})}{2} - \frac{n - \widehat{n}}{n} = \frac{\widehat{n}}{n} - \frac{\ell_1(\mathcal{P}^*, \mathcal{Q})}{2}\end{aligned}$$

Case 2: $\widehat{\ell_1} \geq 2(1 - \beta) - \varepsilon$

Repeat the exact same analysis as in Theorem 9.5 but with $\widehat{r}$ replaced by $\widehat{r}' = |\widehat{\mathcal{T}}'| = |\widehat{\mathcal{T}}| + 1 = \widehat{r} + 1$ yields a matching size of at least $\beta \cdot n - s_{\widehat{r}+1,\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r} + 1)}$, where

$$s_{\widehat{r},\varepsilon,\delta} \in \mathcal{O}\left(\frac{(\widehat{r} + 1) \cdot \log 1/\delta}{\varepsilon^2 \cdot \log(\widehat{r} + 1)}\right)$$

and $s_{\widehat{r}+1,\varepsilon,\delta} \cdot \sqrt{\log(\widehat{r} + 1)} \in o(1)$. □

9.6.4 True distribution has small support size

If the support size of the true types is $o(n)$, a natural thing to do is to *learn* $\boldsymbol{c}^*$ up to some ε accuracy while forgoing some $o(n)$ initial matches, and then obtain $\approx 1-\varepsilon$ competitive ratio on the remaining arrivals. Though this is wholly possible in the random arrival model, it crucially depends on $\boldsymbol{c}^*$ having at most $o(n)$ types. Although we do not know the support size of $\boldsymbol{c}^*$ a priori, we can again employ techniques from property testing. For any desired support size k and constant ε, [VV17, WY19] tell us that $O(\frac{k}{\log k})$ samples are sufficient for us to estimate the support size of a discrete distribution up to additive error of εk. Therefore, for any $k \in o(n)$ and constant ε, given any algorithm ALG under the random arrival model achieving competitive ratio α, we can first spend $o(1)$ arrivals to test whether $\boldsymbol{c}^*$ is supported on $(1+\varepsilon) \cdot k$ types:

- If "Yes", then we can spend another $O(k/\varepsilon^2) \subseteq o(1)$ arrivals to estimate $\boldsymbol{c}^*$ up to ε accuracy, i.e. we can form $\widehat{\boldsymbol{c}}$ with $\ell_1(\boldsymbol{c}^*, \widehat{\boldsymbol{c}}) \leq \varepsilon$, then exploit $\widehat{\boldsymbol{c}}$ via Mimic.
- If "No", use ALG and achieve a competitive ratio of $\alpha - o(1)$.

The choice of k is flexible in practice, depending on how much one is willing to lose in the $o(1)$ in the "No" case.

Chapter 10

Learning multivariate Gaussians with imperfect advice

> "Events may appear to us to be random, but this could be attributed to human ignorance about the details of the processes involved."
>
> - Brian Everitt [Eve99]

> "While in theory randomness is an intrinsic property, in practice, randomness is incomplete information."
>
> - Nassim Nicholas Taleb [Tal07]

10.1 Introduction

The problem of approximating an underlying distribution from its observed samples is a fundamental scientific problem. The distribution learning problem has been studied for more than a century in statistics, and it is the underlying engine for much of applied machine learning. The emphasis in modern applications is on high-dimensional distributions, with the goal being to understand when one can escape the curse of dimensionality. The survey [Dia16] gives an excellent overview of classical and modern techniques for distribution learning, especially when there is some underlying structure to be exploited.

In this chapter, we investigate how to go beyond worst case sample complexities for learning distributions by leveraging imperfect advice about the underlying distribution. More specifically, we study the classical problem of learning high-dimensional Gaussian distributions. It is known that it takes $\widetilde{\Theta}(d/\varepsilon^2)$ samples to learn a Gaussian mean $\widehat{\boldsymbol{\mu}} \in \mathbb{R}^d$ such that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) \leq \varepsilon$ when $\boldsymbol{\Sigma} = \boldsymbol{I}_d$ and $\widetilde{\Theta}(d^2/\varepsilon^2)$ samples for general covariance matrices, e.g. see Lemma 2.25. The algorithm for both cases is the most obvious one: compute the empirical mean and empirical covariance. Meanwhile, note that if one is given as advice the correct mean $\widetilde{\boldsymbol{\mu}} = \boldsymbol{\mu}$, then using distribution testing, one can certify that $\|\widetilde{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2 \leq \varepsilon$ using only $\widetilde{\Theta}(\sqrt{d}/\varepsilon^2)$ samples [DKS17, Appendix

C], quadratically better than without advice. Observing the gap in sample complexities between testing and learning, we design algorithms under `TestAndAct` framework for the problem of multivariate Gaussian learning with imperfect advice, yielding provably lower sample complexities when given high quality advice.

> **Multivariate Gaussian Learning with Advice**: Given samples from a Gaussian $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, as well as advice $\widetilde{\boldsymbol{\mu}}$ and $\widetilde{\boldsymbol{\Sigma}}$, how many samples are required to recover $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\Sigma}}$ such that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{\Sigma}), N(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}}) \leq \varepsilon$ with probability at least $1 - \delta$? The sample complexity should a function of the dimension, ε, δ, as well as a measure of how close $\widetilde{\boldsymbol{\mu}}$ and $\widetilde{\boldsymbol{\Sigma}}$ are to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ respectively.

10.2 Our main results

We give the first known results in distribution learning with imperfect advice. Our techniques are piecewise elementary and easy to follow. Following the TestAndAct framework for designing learning-augmented algorithms, we present two polynomial time algorithms TestAndOptimizeMean and TestAndOptimizeCovariance for producing the estimates $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\Sigma}}$ based on LASSO and SDP formulations. These algorithms provably improve upon the sample complexities of $\widetilde{\Theta}(d/\varepsilon^2)$ and $\widetilde{\Theta}(d^2/\varepsilon^2)$ for identity and general covariances respectively when given high quality advice of a mean $\widetilde{\boldsymbol{\mu}} \in \mathbb{R}^d$ or covariance matrix $\widetilde{\boldsymbol{\Sigma}} \in \mathbb{R}^{d \times d}$.

Theorem 10.1. *For any given $\varepsilon, \delta \in (0,1)$, $\eta \in [0, \frac{1}{4}]$, and $\widetilde{\boldsymbol{\mu}} \in \mathbb{R}^d$, the TestAndOptimizeMean algorithm uses $n \in \widetilde{\mathcal{O}}\left(\frac{d}{\varepsilon^2} \cdot (d^{-\eta} + \min\{1, f(\boldsymbol{\mu}, \widetilde{\boldsymbol{\mu}}, d, \eta, \varepsilon)\})\right)$, where*

$$f(\boldsymbol{\mu}, \widetilde{\boldsymbol{\mu}}, d, \eta, \varepsilon) = \frac{\|\boldsymbol{\mu} - \widetilde{\boldsymbol{\mu}}\|_1^2}{d^{1-4\eta}\varepsilon^2} ,$$

i.i.d. samples from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ for some unknown mean $\boldsymbol{\mu}$ and identity covariance $\boldsymbol{I}_d$, and can produce $\widehat{\boldsymbol{\mu}}$ in $\mathrm{poly}(n, d)$ time such that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) \leq \varepsilon$ with success probability at least $1 - \delta$.

Theorem 10.2. *For any given $\varepsilon, \delta \in (0,1)$, $\eta \in [0,1]$ and $\widetilde{\boldsymbol{\Sigma}} \in \mathbb{R}^{d \times d}$, TestAndOptimizeCovariance uses $n \in \widetilde{\mathcal{O}}\left(\frac{d^2}{\varepsilon^2} \cdot \left(d^{-\eta} + \min\left\{1, f(\boldsymbol{\Sigma}, \widetilde{\boldsymbol{\Sigma}}, d, \eta, \varepsilon)\right\}\right)\right)$, where*

$$f(\boldsymbol{\Sigma}, \widetilde{\boldsymbol{\Sigma}}, d, \eta, \varepsilon) = \frac{\|\mathrm{vec}(\widetilde{\boldsymbol{\Sigma}}^{-1/2}\boldsymbol{\Sigma}\widetilde{\boldsymbol{\Sigma}}^{-1/2} - \boldsymbol{I}_d)\|_1^2}{d^{2-\eta}\varepsilon^2} ,$$

i.i.d. samples from $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for some unknown mean $\boldsymbol{\mu}$ and unknown covariance $\boldsymbol{\Sigma}$, and can produce $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\Sigma}}$ in $\mathrm{poly}(n, d, \log(1/\varepsilon))$ time such that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{\Sigma}), N(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}})) \leq \varepsilon$ with success probability at least $1 - \delta$.

In particular, the TestAndOptimizeMean algorithm uses only $\widetilde{\mathcal{O}}(\frac{d^{1-\beta}}{\varepsilon^2})$ samples when $\|\boldsymbol{\mu} - \widetilde{\boldsymbol{\mu}}\|_1 < \varepsilon d^{\frac{1-3\beta}{2}} = \varepsilon\sqrt{d} \cdot d^{-\frac{3\beta}{2}}$ and the TestAndOptimizeCovariance algorithm uses only $\widetilde{\mathcal{O}}(\frac{d^{2-\beta}}{\varepsilon^2})$ samples when $\|\mathrm{vec}(\widetilde{\boldsymbol{\Sigma}}^{-1/2}\boldsymbol{\Sigma}\widetilde{\boldsymbol{\Sigma}}^{-1/2} - \boldsymbol{I}_d)\|_1 < \varepsilon d^{1-\beta} = \varepsilon d \cdot d^{-\beta}$. Moreover, both our algorithms have runtime which is polynomial in d.

The choice of representing the quality of the advice in terms of the ℓ_1-norm is well-motivated. It is known, e.g. see [FR13, Theorem 2.5], that if a vector $\boldsymbol{x}$ satisfies $\|\boldsymbol{x}\|_1 \leq \tau$, then for any positive integer s, $\sigma_s(\boldsymbol{x}) \leq \tau/(2\sqrt{s})$, where $\sigma_s(\boldsymbol{x})$ is the ℓ_2-error of the best s-sparse approximation to $\boldsymbol{x}$. Thus, if $\|\widetilde{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_1 \leq 2\varepsilon d^{(1-\eta)/2}$, then $\sigma_{d^{1-\eta}}(\widetilde{\boldsymbol{\mu}} - \boldsymbol{\mu}) \leq \varepsilon$. The latter may be very reasonable, as one may have good predictions for most of the coordinates of the mean with the error in the advice concentrated on a sublinear $d^{1-\eta}$ coordinates. The same consideration also applies to the entrywise-ℓ_1 norm for error in the covariance matrix advice. Algorithmically, we employ sublinear property testing algorithms to evaluate the quality of the given advice before deciding how to produce a final estimate, similar in spirit to the TestAndMatch approach in Chapter 9. The idea of incorporating property testing as a way to verify whether certain distributional assumptions are satisfied that enable efficient subsequent learning has also been explored in recent works on testable learning [RV23, KSV24, Vas24].

We supplement our algorithmic upper bounds with information-theoretic lower bounds. Here, we say that an algorithm $(\varepsilon, 1-\delta)$-PAC learns a distribution $\mathcal{P}$ if it can produce another distribution $\widehat{\mathcal{P}}$ such that $\mathrm{d_{TV}}(\mathcal{P}, \widehat{\mathcal{P}}) \leq \varepsilon$ with success probability at least $1-\delta$. Our lower bounds tell us that $\widetilde{\Omega}(d/\varepsilon^2)$ and $\widetilde{\Omega}(d^2/\varepsilon^2)$ samples are unavoidable for PAC-learning $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ and $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ respectively when given low quality advice.

Theorem 10.3. *Suppose we are given $\widetilde{\boldsymbol{\mu}} \in \mathbb{R}^d$ as advice with only the guarantee that $\|\boldsymbol{\mu} - \widetilde{\boldsymbol{\mu}}\|_1 \leq \Delta$. Then, any algorithm that $(\varepsilon, \frac{2}{3})$-PAC learns $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ requires $\Omega\left(\frac{\min\{d, \Delta^2/\varepsilon^2\}}{\varepsilon^2 \log(1/\varepsilon)}\right)$ samples in the worst case.*

Theorem 10.4. *Suppose we are given a symmetric and positive-definite $\widetilde{\boldsymbol{\Sigma}} \in \mathbb{R}^{d \times d}$ as advice with only the guarantee that $\|\mathrm{vec}\left(\widetilde{\boldsymbol{\Sigma}}^{-\frac{1}{2}}\boldsymbol{\Sigma}\widetilde{\boldsymbol{\Sigma}}^{-\frac{1}{2}} - \boldsymbol{I}_d\right)\|_1 \leq \Delta$. Then, any algorithm that $(\varepsilon, \frac{2}{3})$-PAC learns $N(\boldsymbol{0}, \boldsymbol{\Sigma})$ requires $\Omega\left(\frac{\min\{d^2, \Delta^2/\varepsilon^2\}}{\varepsilon^2 \log(1/\varepsilon)}\right)$ samples in the worst case.*

Both of our lower bounds are tight in the following sense. TestAndOptimizeMean gives a polynomially-smaller sample complexity compared to $\widetilde{\mathcal{O}}(d/\varepsilon^2)$ when the advice quality (measured in terms of the ℓ_1-norm) is polynomially smaller compared to $\varepsilon\sqrt{d}$. Theorem 10.3 shows that this is the best we can do; there is a hard instance where the advice quality is $\leq \varepsilon\sqrt{d}$ and we need $\widetilde{\Omega}(d/\varepsilon^2)$ samples. A similar situation happens between TestAndOptimizeCovariance and Theorem 10.4, when the guarantee on the advice quality is at most εd.

The lower bounds in Theorem 10.3 and Theorem 10.4 apply when the parameter Δ is known to the algorithm. Our algorithms are stronger since they do not need to know

Δ beforehand. In case Δ is known, the sample complexity of the distribution learning component of our algorithms match the above lower bounds up to log factors.

10.3 Technical overview

To obtain our upper bounds, we first show that the existing test statistics for non-tolerant testing can actually be used for tolerant testing with the same asymptotic sample complexity bounds and then use these new tolerant testers to test the advice quality. The tolerance is with respect to the ℓ_2-norm for mean testing and with respect to the Frobenius norm for covariance testing. These results are folklore, but we did not manage to find formal proofs for them. As these may be of independent interest, we present their proofs in Appendix C.2.1 for completeness.

Lemma 10.5 (Tolerant mean tester)**.** *Given $\varepsilon_2 > \varepsilon_1 > 0$, $\delta \in (0,1)$, and $d \geq \left(\frac{16\varepsilon_2^2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2$, there is a tolerant tester that uses $\mathcal{O}\left(\frac{\sqrt{d}}{\varepsilon_2^2-\varepsilon_1^2}\log\left(\frac{1}{\delta}\right)\right)$ i.i.d. samples from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ and satisfies the following two conditions:*

1. *If $\|\boldsymbol{\mu}\|_2 \leq \varepsilon_1$, then the tester outputs* Accept *with probability at least $1-\delta$.*
2. *If $\|\boldsymbol{\mu}\|_2 \geq \varepsilon_2$, then the tester outputs* Reject *with probability at least $1-\delta$.*

The tester is allowed to output Accept *or* Reject *arbitrarily when $\varepsilon_1 < \|\boldsymbol{\mu}\|_2 < \varepsilon_2$.*

Lemma 10.6 (Tolerant covariance tester)**.** *Given $\varepsilon_2 > \varepsilon_1 > 0$, $\delta \in (0,1)$, and $d \geq \varepsilon_2^2$, there is a tolerant tester that uses $\mathcal{O}\left(d \cdot \max\left\{\frac{1}{\varepsilon_1^2}, \left(\frac{\varepsilon_2^2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2, \left(\frac{\varepsilon_2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2\right\}\log\left(\frac{1}{\delta}\right)\right)$ i.i.d. samples from $N(\mathbf{0}, \boldsymbol{\Sigma})$ and satisfies the following two conditions:*

1. *If $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F \leq \varepsilon_1$, then the tester outputs* Accept *with probability at least $1-\delta$.*
2. *If $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F \geq \varepsilon_2$, then the tester outputs* Reject *with probability at least $1-\delta$.*

The tester is allowed to output Accept *or* Reject *arbitrarily when $\varepsilon_1 < \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_2 < \varepsilon_2$.*

In the remaining, we will first explain how to obtain our result for TestAndOptimizeMean before explaining how a similar approach works for TestAndOptimizeCovariance. For a detailed proof of our lower bounds, we refer readers to [BCGJG24, Section 5].

10.3.1 Approach for TestAndOptimizeMean

Without loss of generality, we may assume henceforth that $\widetilde{\boldsymbol{\mu}} = \mathbf{0}$ since one can always pre-process samples by subtracting $\widetilde{\boldsymbol{\mu}}$ and then add $\widetilde{\boldsymbol{\mu}}$ back to the estimated $\widehat{\boldsymbol{\mu}}$. Our overall approach is quite natural: (i) use the tolerant testing algorithm in Lemma 10.5 to get an upper bound on the "advice quality", and (ii) enforce the constraint on the "advice quality" when learning $\widehat{\boldsymbol{\mu}}$.

The most immediate notion of advice quality one may posit is $\|\boldsymbol{\mu} - \mathbf{0}\|_2 = \|\boldsymbol{\mu}\|_2$. Let us see what issues arise. Using an exponential search process, we can invoke Lemma 10.5 directly to find some $r > 0$, such that $r/2 \leq \|\boldsymbol{\mu} - \widetilde{\boldsymbol{\mu}}\|_2 = \|\boldsymbol{\mu}\|_2 \leq r$. To argue about the sample complexity for learning $\widehat{\boldsymbol{\mu}}$, and ignoring computational efficiency, one can invoke the Scheffé tournament approach for density estimation; see Section 2.3.4. Let $\mathcal{N}$ be an ε-cover in ℓ_2 of the the ℓ_2-ball of radius r around $\mathbf{0}$; see Section 2.3.5. Clearly, $\boldsymbol{\mu}$ is ε-close in ℓ_2 to one of the points in $\mathcal{N}$. It is known (e.g. see [DL01, Chapter 4] and Theorem 2.20) that the sample complexity of the Scheffé tournament algorithm scales as $\log |\mathcal{N}|$. However, point 1 of Theorem 2.22 tells us that $\log |\mathcal{N}| = \Omega(d)$. Indeed, one can get a formal lower bound showing that the sample complexity cannot be made sublinear in d for non-trivial values of r. To get around this barrier, we will instead take the notion of advice quality to be $\|\boldsymbol{\mu}\|_1$ instead of $\|\boldsymbol{\mu}\|_2$. Point 2 of Theorem 2.22 tells us that that $d^{\frac{cr^2}{\varepsilon^2}}$ ℓ_2 balls of radius ε suffice to cover an ℓ_1-ball of radius r, for some absolute constant $c > 0$. Using this modified approach, the Scheffé tournament only requires $\mathcal{O}(\frac{r^2}{\varepsilon^4} \log d)$ samples which could be $o(d/\varepsilon^2)$ for a wide range of values of r.

There are still two issues to address: (i) how to obtain an ℓ_1 estimate r of $\boldsymbol{\mu}$, i.e. $r/2 \leq \|\boldsymbol{\mu}\|_1 \leq r$, and (ii) how to get a computationally efficient learning algorithm.

(i) We can apply the standard inequality $\|\boldsymbol{\mu}\|_2 \leq \|\boldsymbol{\mu}\|_1 \leq \sqrt{d}\|\boldsymbol{\mu}\|_2$ bound to transform our ℓ_2 estimate from Lemma 10.5 into an ℓ_1 one. However, since the number of samples has a quadratic relation with r, we need a better approximation than $\sqrt{d}$ to achieve sample complexity that is sublinear in d. To achieve this, we partition the $\boldsymbol{\mu}$ vector into blocks of size at most $k \leq d$ and approximate the ℓ_1 norm of each smaller dimension vector separately and then add them up to obtain an ℓ_1 estimate of the overall $\boldsymbol{\mu}$. Doing so improves the resulting multiplicative error to $\approx \sqrt{d/k}$ instead of $\sqrt{d}$.

(ii) The Scheffé tournament approach requires time at least linear in the size of the ε-cover. In order to do better, we observe that we can formulate our task as an optimization problem with an ℓ_1-constraint. Specifically, given samples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$, we solve the following program:

$$\widehat{\boldsymbol{\mu}} = \operatorname*{argmin}_{\|\boldsymbol{\beta}\|_1 \leq r} \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{y}_i - \boldsymbol{\beta}\|_2^2$$

The error $\|\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}}\|_2$ can be analyzed by similar techniques as those used for analyzing ℓ_1-regularization in the context of LASSO or compressive sensing; e.g. see [Tib96, Tib97, HTW15].

10.3.2 Approach for TestAndOptimizeCovariance

As before, we may assume without loss of generality that $\widetilde{\boldsymbol{\Sigma}} = \boldsymbol{I}_d$ by pre-processing the samples appropriately. Furthermore, we can invest $\Omega(d/\varepsilon^2)$ samples up-front to ensure that the empirical mean $\widehat{\boldsymbol{\mu}}$ will be an ε-good estimate of $\boldsymbol{\mu}$. Then, it will suffice to obtain an estimate $\widehat{\boldsymbol{\Sigma}}$ of $\boldsymbol{\Sigma}$ such that $\|\boldsymbol{\Sigma}^{-1}\widehat{\boldsymbol{\Sigma}} - \boldsymbol{I}_d\|_F \leq \mathcal{O}(\varepsilon)$ suffices. At a high level, the approach for TestAndOptimizeCovariance is the same as TestAndOptimizeMean after three key adjustments to adapt the approach from vectors to matrices.

The first adjustment is that we perform a suitable preconditioning process using an additional $\mathcal{O}(d)$ samples so that we can subsequently argue that $\|\boldsymbol{\Sigma}^{-1}\|_2 \leq 1$. This will then allow us to argue that $\|\boldsymbol{\Sigma}^{-1}\widehat{\boldsymbol{\Sigma}} - \boldsymbol{I}_d\|_F \leq \|\boldsymbol{\Sigma}^{-1}\|_2 \|\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}\|_F \in \mathcal{O}(\varepsilon)$. Our preconditioning technique is inspired by [KLSU19]; while they use $\mathcal{O}(d)$ samples to construct a preconditioner to control the maximum eigenvalue, we use a similar approach to control the minimum eigenvalue.

The second adjustment pertains to the partitioning idea used for multiplicatively approximating $\|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1$. Observe that the covariance matrix of a marginal of a multivariate Gaussian is precisely the principal submatrix of the original covariance $\boldsymbol{\Sigma}$ on the corresponding projected coordinates. For example, if one focuses on coordinates $\{i, j\} \subseteq [d]$ of each sample, then the corresponding covariance matrix is $\begin{bmatrix} \boldsymbol{\Sigma}_{i,i} & \boldsymbol{\Sigma}_{i,j} \\ \boldsymbol{\Sigma}_{j,i} & \boldsymbol{\Sigma}_{j,j} \end{bmatrix}$, for $i < j$. To this end, we generalize the partitioning scheme described for TestAndOptimizeMean to higher ordered objects.

Definition 10.7 (Partitioning scheme)**.** Fix $q \geq 1$, $d \geq 1$, and a q-ordered d-dimensional tensor $\mathcal{T} \in \mathbb{R}^{d^{\otimes q}}$. Let $\boldsymbol{B} \subseteq [d]$ be a subset of indices and define $\mathcal{T}_{\boldsymbol{B}}$ as the principal subtensor of $\mathcal{T}$ indexed by $\boldsymbol{B}$. A collection of subsets $\boldsymbol{B}_1, \ldots, \boldsymbol{B}_w \subseteq [d]$ is called an (q, d, k, a, b)-partitioning of the tensor $\mathcal{T}$ if the following three properties hold:

- $|\boldsymbol{B}_1| \leq k, \ldots, |\boldsymbol{B}_w| \leq k$
- For every cell of $\mathcal{T}$ appears in *at least* a of the w principal subtensors $\mathcal{T}_{\boldsymbol{B}_1}, \ldots, \mathcal{T}_{\boldsymbol{B}_w}$.
- For every cell of $\mathcal{T}$ appears in *at most* b of the w principal subtensors $\mathcal{T}_{\boldsymbol{B}_1}, \ldots, \mathcal{T}_{\boldsymbol{B}_w}$.

For example, when $q = 2$, $\boldsymbol{T} \in \mathbb{R}^{d \times d}$ is just a $d \times d$ matrix. Observe one can always obtain a partitioning with $k \leq d^q$ by letting the index sets $\boldsymbol{B}_1, \ldots, \boldsymbol{B}_w$ encode every possible index, but this results in a large $w = \binom{d}{q}$ which can be undesirable for downstream analysis. The partitioning used in TestAndOptimizeMean is a special case of Definition 10.7 with $q = a = b = 1$, $k = \lceil d/w \rceil$. For TestAndOptimizeCovariance, we are interested in the case where $q = 2$ and $a = 1$. Ideally, we want to minimize k and b as well. Fig. 10.1 illustrates an example of a $(q = 2, d = 5, k = 3, a = 1, b = 3)$-partitioning.

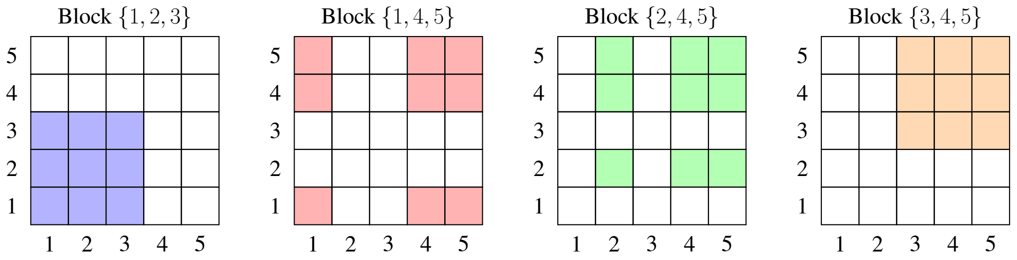


Figure 10.1: Consider partitioning a $d \times d$ matrix (i.e. $d = 5$, $q = 2$) with $w = 4$ blocks $\{(1, 2, 3), (1, 4, 5), (2, 4, 5), (3, 4, 5)\}$, each of size $k = 3$. Every cell in the original 5×5 matrix appears in at least $a = 1$ and at most $b = 3$ times across all the induced submatrices.

The last change is to the optimization program for learning $\widehat{\Sigma}$. Given samples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n$ from $N(\boldsymbol{\mu}, \Sigma)$, we define:

$$\widehat{\Sigma} = \operatorname*{argmin}_{\substack{\boldsymbol{A} \in \mathbb{R}^{d \times d} \text{ is p.s.d.} \\ \|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq r \\ \|\boldsymbol{A}^{-1}\|_2 \leq 1}} \sum_{i=1}^{n} \|\boldsymbol{A} - \boldsymbol{y}_i \boldsymbol{y}_i^\top\|_F^2$$

Observe that Σ is a feasible solution to the above program. The optimization problem can be solved efficiently since it can be written as an SDP with convex constraints; see Appendix C.2.3. We finally bound $\|\Sigma - \widehat{\Sigma}\|_F$ using an analysis that mirrors that for TestAndOptimizeMean but is in terms of matrix algebra.

10.4 TestAndOptimizeMean for the identity covariance setting

We begin by defining a parameterized sample count $m(d, \varepsilon, \delta)$. Then, we will state our ApproxL1 algorithm and show how to use it according to the strategy outlined in Section 10.3.1.

Definition 10.8. Fix any $d \geq 1$, $\varepsilon > 0$, and $\delta \in (0, 1)$. We define $m(d, \varepsilon, \delta) = n_{d,\varepsilon} \cdot r_\delta$, where

$$n_{d,\varepsilon} = \left\lceil \frac{16\sqrt{d}}{3\varepsilon^2} \right\rceil \qquad \text{and} \qquad r_\delta = 1 + \left\lceil \log\left(\frac{12}{\delta}\right) \right\rceil$$

Given samples from a d-dimensional isotropic Gaussian $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ with unknown mean $\boldsymbol{\mu}$ and identity covariance, the ApproxL1 algorithm partitions the d coordinates into $w = \lceil d/k \rceil$ buckets each of length at most $k \in [d]$ and separately perform an exponential search to find the 2-approximation of the ℓ_2 norm of each bucket by repeatedly invoking the tolerant tester from Lemma 10.5. In the terminology of Definition 10.7, this is a partitioning scheme with $q = 1$, $a = 1$, and $b = 1$. Crucially, projecting the samples in $\mathbb{R}^d$ of $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ into the subcoordinates of $\boldsymbol{B} \subseteq [d]$ yields samples in $\mathbb{R}^{|\boldsymbol{B}|}$ from $N(\boldsymbol{\mu}_{\boldsymbol{B}}, \boldsymbol{I}_{|\boldsymbol{B}|})$

so we can obtain valid estimates using each of these marginals. After obtaining the ℓ_2 estimate of each bucket, we use Lemma 2.3 to obtain bounds on the ℓ_1 and then combine them by summing up these estimates: if we have an ε-multiplicative approximation of each bucket's ℓ_1, then their sum will be an $\mathcal{O}(\varepsilon)$-multiplicative approximation of the entire $\boldsymbol{\mu}$ vector whenever the partition overlap parameters a and b of Definition 10.7 are constants.

Algorithm 21 The ApproxL1 algorithm.

```
Input: Error rate $\varepsilon > 0$, failure rate $\delta \in (0,1)$, block size $k \in [d]$, lower bound $\alpha > 0$,
upper bound $\zeta > 2\alpha$, and i.i.d. samples $\mathcal{S}$ from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$
Output: Fail, OK, or $\lambda \in \mathbb{R}$
Define $w = \lceil d/k \rceil$ and $\delta' = \frac{\delta}{w \cdot \lceil \log_2 \zeta/\alpha \rceil}$
Partition the index set $[d]$ into $w$ blocks:
    $$\boldsymbol{B}_1 = \{1, \ldots, k\}, \boldsymbol{B}_2 = \{k+1, \ldots, 2k\}, \ldots, \boldsymbol{B}_w = \{k(w-1)+1, \ldots, d\}$$
for $j \in \{1, \ldots, w\}$ do
    Define $\mathcal{S}_j = \{\boldsymbol{x}_{\boldsymbol{B}_j} \in \mathbb{R}^{|\boldsymbol{B}_j|} : \boldsymbol{x} \in \mathcal{S}\}$ as the samples projected to $\boldsymbol{B}_j$
                                                                        ▷ See Definition 2.4
    Initialize $o_j$ = Fail
    for $i = 1, 2, \ldots, \lceil \log_2 \zeta/\alpha \rceil$ do
        Define $l_i = 2^{i-1} \cdot \alpha$
        Let Outcome be the output of the tolerant tester of Lemma 10.5 using sample
        set $\mathcal{S}_j$ with parameters $\varepsilon_1 = l_i$, $\varepsilon_2 = 2l_i$, and $\delta = \delta'$
        if Outcome is Accept then
            Set $o_j = l_i$ and break                           ▷ Escape inner loop for block $j$
if there exists a Fail amongst $\{o_1, \ldots, o_w\}$ then
    return Fail
else if $4\sum_{j=1}^{w} o_j^2 \leq \alpha^2$ then
    return OK                                   ▷ Note: $o_j$ is an estimate for $\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2$
elsereturn $\lambda = 2\sum_{j=1}^{w} \sqrt{|\boldsymbol{B}_j|} \cdot o_j$          ▷ $\lambda$ is an estimate for $\|\boldsymbol{\mu}\|_1$
```

One can show that the ApproxL1 algorithm has the following guarantees.

Lemma 10.9. *Let ε, δ, k, α, and ζ be the input parameters to the ApproxL1 algorithm (Algorithm 21). Given $m(k, \alpha, \delta')$ i.i.d. samples from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$, the ApproxL1 algorithm succeeds with probability at least $1 - \delta$ and has the following properties:*

- *If ApproxL1 outputs* Fail, *then $\|\boldsymbol{\mu}\|_2 > \zeta/2$.*
- *If ApproxL1 outputs* OK, *then $\|\boldsymbol{\mu}\|_2 \leq \alpha$.*
- *If ApproxL1 outputs $\lambda \in \mathbb{R}$, then $\|\boldsymbol{\mu}\|_1 \leq \lambda \leq 2\sqrt{k} \cdot (\lceil d/k \rceil \cdot \alpha + 2\|\boldsymbol{\mu}\|_1)$.*

Proof. We begin by stating some properties of $o_1, \ldots, o_w$. Fix an arbitrary index $j \in \{1, \ldots, w\}$ and suppose o_j is *not* a Fail, i.e. the tolerant tester of Lemma 10.5 outputs Accept for some $i^* \in \{1, 2, \ldots, \lceil \log_2 \zeta/\alpha \rceil\}$. Note that ApproxL1 sets $o_j = \ell_{i^*}$ and the tester outputs Reject for all smaller indices $i \in \{1, \ldots, i^* - 1\}$. Since the tester outputs Accept for i^*, we have that $\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2\ell_{i^*} = 2o_j$. Meanwhile, if $i^* > 1$, then $\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 > \ell_{i^*-1} = \ell_{i^*}/2 = o_j/2$ since the tester outputs Reject for $i^* - 1$. Thus, we see that

- When o_j is not Fail, we have $\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2o_j$.
- When $\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2\alpha$, we have $i^* = 1$ and $o_j = \ell_1 = \alpha$.
- When $\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 > 2\alpha = 2\ell_1$, we have $i^* > 1$ and so $o_j < 2\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2$.

Success probability. Fix an arbitrary index $i \in \{1, 2, \ldots, \lceil \log_2 \zeta/\alpha \rceil\}$ with $\ell_i = 2^{i-1}\alpha$, where $\ell_i \leq \ell_1 = \alpha$ for any i. We invoke the tolerant tester with $\varepsilon_2 = 2\ell_i = 2\varepsilon_1$, so the i^{th} invocation uses at most $n_{k,\varepsilon} \cdot r_\delta$ i.i.d. samples to succeed with probability at least $1 - \delta$; see Definition 10.8 and Algorithm 30. So, with $m(k, \alpha, \delta')$ samples, *any* call to the tolerant tester succeeds with probability at least $1 - \delta'$, where $\delta' = \frac{\delta}{w \cdot \lceil \log_2 \zeta/\alpha \rceil}$. By construction, there will be at most $w \cdot \lceil \log_2 \zeta/\alpha \rceil$ calls to the tolerant tester. Therefore, by union bound, *all* calls to the tolerant tester jointly succeed with probability at least $1 - \delta$.

Property 1. When ApproxL1 outputs Fail, there exists a Fail amongst $\{o_1, \ldots, o_w\}$. For any fixed index $j \in \{1, \ldots, w\}$, this can only happen when all calls to the tolerant tester outputs Reject. This means that $\|\boldsymbol{x}_{\boldsymbol{B}_j}\|_2 > \varepsilon_1 = \ell_i = 2^{i-1} \cdot \alpha$ for all $i \in \{1, 2, \ldots, \lceil \log_2 \zeta/\alpha \rceil\}$. In particular, this means that $\|\boldsymbol{x}_{\boldsymbol{B}_j}\|_2 > \zeta/2$.

Property 2. When ApproxL1 outputs OK, we have $4\sum_{j=1}^{w} o_j^2 \leq \alpha^2$. Then, since $\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2o_j$ for each index $j \in \{1, \ldots, w\}$, we see that

$$\|\boldsymbol{\mu}\|_2^2 = \sum_{j=1}^{w} \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2^2 \leq \sum_{j=1}^{w} (2o_j)^2 = 4\sum_{j=1}^{w} o_j^2 \leq \alpha^2$$

That is, $\|\boldsymbol{\mu}\|_2 \leq \alpha$ as desired.

Property 3. When ApproxL1 outputs $\lambda = 2\sum_{j=1}^{w} \sqrt{|\boldsymbol{B}_j|} \cdot o_j \in \mathbb{R}$, we have $4\sum_{j=1}^{w} o_j^2 > \varepsilon^2$. We can lower bound λ as follows:

$$\begin{aligned}
\lambda &= 2\sum_{j=1}^{w} \sqrt{|\boldsymbol{B}_j|} \cdot o_j \\
&\geq 2\sum_{j=1}^{w} \sqrt{|\boldsymbol{B}_j|} \cdot \frac{\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2}{2} && \text{(since } \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2o_j\text{)} \\
&\geq \sum_{j=1}^{w} \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1 && \text{(since } \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1 \leq \sqrt{|\boldsymbol{B}_j|} \cdot \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2\text{)}
\end{aligned}$$

$$= \|\boldsymbol{\mu}\|_1 \qquad \text{(since } \textstyle\sum_{j=1}^{w} \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1 = \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1\text{)}$$

That is, $\lambda \geq \|\boldsymbol{\mu}\|_1$. Meanwhile, we can also upper bound λ as follows:

$$\begin{aligned}
\lambda &= 2\sum_{j=1}^{w} \sqrt{|\boldsymbol{B}_j|} \cdot o_j \\
&\leq 2\sqrt{k} \sum_{j=1}^{w} o_j && \text{(since } |\boldsymbol{B}_j| \leq k\text{)} \\
&= 2\sqrt{k} \cdot \left(\sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2\alpha}}^{w} o_j + \sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 > 2\alpha}}^{w} o_j \right) \\
& && \text{(partitioning the blocks based on } \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \text{ versus } 2\alpha\text{)} \\
&= 2\sqrt{k} \cdot \left(\sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2\alpha}}^{w} \alpha + \sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 > 2\alpha}}^{w} o_j \right) && \text{(since } \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2\alpha \text{ implies } o_j = \alpha\text{)} \\
&\leq 2\sqrt{k} \cdot \left(\sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2\alpha}}^{w} \alpha + \sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 > 2\alpha}}^{w} 2\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \right) \\
& && \text{(since } \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 > 2\alpha \text{ implies } o_j \leq 2\|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2\text{)} \\
&\leq 2\sqrt{k} \cdot \left(\sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2\alpha}}^{w} \alpha + 2\sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 > 2\alpha}}^{w} \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1 \right) && \text{(since } \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1\text{)} \\
&\leq 2\sqrt{k} \cdot \left(\lceil d/k \rceil \cdot \alpha + 2\sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 > 2\alpha}}^{w} \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1 \right) \\
& && \text{(since } |\{j \in [w] : \boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 \leq 2\alpha\}| \leq w\text{)} \\
&\leq 2\sqrt{k} \cdot (\lceil d/k \rceil \cdot \alpha + 2\|\boldsymbol{\mu}\|_1) \\
& && \text{(since } \textstyle\sum_{\substack{j=1 \\ \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_2 > 2\alpha}}^{w} \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1 \leq \sum_{j=1}^{w} \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1 = \|\boldsymbol{\mu}_{\boldsymbol{B}_j}\|_1\text{)}
\end{aligned}$$

That is, $\lambda \leq 2\sqrt{k} \cdot (\lceil d/k \rceil \cdot \alpha + 2\|\boldsymbol{\mu}\|_1)$. The property follows by putting together both bounds. □

Now, suppose ApproxL1 tells us that $\|\boldsymbol{\mu}\|_1 \leq r$. We can then perform a constrained version of LASSO to search for a candidate $\widehat{\boldsymbol{\mu}} \in \mathbb{R}^d$ using $\mathcal{O}\left(\frac{r^2}{\varepsilon^4} \log \frac{d}{\delta}\right)$ samples from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$.

Lemma 10.10. *Fix $d \geq 1$, $r \geq 0$, and $\varepsilon, \delta > 0$. Given $\mathcal{O}\left(\frac{r^2}{\varepsilon^4} \log \frac{d}{\delta}\right)$ samples from*

$N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ for some unknown $\boldsymbol{\mu} \in \mathbb{R}^d$ with $\|\boldsymbol{\mu}\|_1 \leq r$, one can produce an estimate $\widehat{\boldsymbol{\mu}} \in \mathbb{R}^d$ in $\operatorname{poly}(n, d)$ time such that $\mathrm{d}_{\mathrm{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) \leq \varepsilon$ with success probability at least $1 - \delta$.

Proof. Suppose we get n samples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \sim N(\boldsymbol{\mu}, \boldsymbol{I}_d)$. For $i \in [n]$, we can re-express each $\boldsymbol{y}_i$ as $\boldsymbol{y}_i = \boldsymbol{\mu} + \boldsymbol{g}_i$ for some $\boldsymbol{g}_i \sim N(\boldsymbol{0}, \boldsymbol{I}_d)$. Let us define $\widehat{\boldsymbol{\mu}} \in \mathbb{R}^d$ as follows:

$$\widehat{\boldsymbol{\mu}} = \operatorname*{argmin}_{\|\boldsymbol{\beta}\|_1 \leq r} \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{y}_i - \boldsymbol{\beta}\|_2^2 \tag{10.1}$$

By optimality of $\widehat{\boldsymbol{\mu}}$ in Eq. (10.1), we have

$$\frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{y}_i - \widehat{\boldsymbol{\mu}}\|_2^2 \leq \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{y}_i - \boldsymbol{\mu}\|_2^2 \tag{10.2}$$

By expanding and rearranging Eq. (10.2), one can show (see Appendix C.2.2)

$$\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2^2 \leq \frac{2}{n} \langle \sum_{i=1}^{n} \boldsymbol{g}_i, \widehat{\boldsymbol{\mu}} - \boldsymbol{\mu} \rangle \tag{10.3}$$

Therefore, with probability at least $1 - \delta$,

$$\begin{aligned}
\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2^2 &\leq \frac{2}{n} \langle \sum_{i=1}^{n} \boldsymbol{g}_i, \widehat{\boldsymbol{\mu}} - \boldsymbol{\mu} \rangle && \text{(From Eq. (10.3))} \\
&\leq \frac{2}{n} \cdot \left\| \sum_{i=1}^{n} \boldsymbol{g}_i \right\|_\infty \cdot \|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_1 && \text{(Hölder's inequality)} \\
&\leq \frac{2}{n} \cdot \left\| \sum_{i=1}^{n} \boldsymbol{g}_i \right\|_\infty \cdot (\|\widehat{\boldsymbol{\mu}}\|_1 + \|\boldsymbol{\mu}\|_1) && \text{(Triangle inequality)} \\
&\leq 4r \cdot \sqrt{\frac{2 \log\left(\frac{2d}{\delta}\right)}{n}} && \text{(From Lemma 2.27, } \|\widehat{\boldsymbol{\mu}}\|_1 \leq r \text{, and } \|\boldsymbol{\mu}\|_1 \leq r)
\end{aligned}$$

When $n = \frac{2r^2 \log \frac{2d}{\delta}}{\varepsilon^4} \in \mathcal{O}\left(\frac{r^2}{\varepsilon^4} \log \frac{d}{\delta}\right)$, we have $\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2^2 \leq 4r \cdot \sqrt{\frac{2\log\left(\frac{2d}{\delta}\right)}{n}} = 4\varepsilon^2$. So, by Theorem 2.18 and Lemma 2.29, we see that

$$\begin{aligned}
\mathrm{d}_{\mathrm{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) &\leq \sqrt{\frac{1}{2} \mathrm{d}_{\mathrm{KL}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d))} \\
&\leq \sqrt{\frac{1}{4} \|\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}}\|_2^2} \leq \sqrt{\frac{4\varepsilon^2}{4}} = \varepsilon
\end{aligned}$$

Finally, it is well-known that LASSO runs in $\operatorname{poly}(n, d)$ time. □

Algorithm 22 The TestAndOptimizeMean algorithm.

```
Input: Error rate $\varepsilon > 0$, failure rate $\delta \in (0,1)$, parameter $\eta \in [0, \frac{1}{4}]$, and sample access to $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$
Output: $\widehat{\boldsymbol{\mu}} \in \mathbb{R}^d$
Define $k = \lceil d^{4\eta} \rceil$, $\alpha = \varepsilon \cdot d^{-(1-3\eta)/2}$, $\zeta = 4\varepsilon \cdot \sqrt{d}$, and $\delta' = \frac{\delta}{\lceil d/k \rceil \cdot \lceil \log_2 \zeta/\alpha \rceil}$
                                                                    ▷ Note: $\zeta > 2\alpha$
Draw $m(k, \alpha, \delta')$ i.i.d. samples from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ and store it into a set $\mathcal{S}$
                                                                    ▷ See Definition 10.8
Let Outcome be the output of the ApproxL1 algorithm given $k$, $\alpha$, $\zeta$, and $\boldsymbol{S}$ as inputs
if Outcome is $\lambda \in \mathbb{R}$ and $\lambda < \varepsilon\sqrt{d}$ then
    Draw $n \in \widetilde{\mathcal{O}}(\lambda^2/\varepsilon^4)$ i.i.d. samples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \in \mathbb{R}^d$ from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$
    return $\widehat{\boldsymbol{\mu}} = \mathrm{argmin}_{\|\boldsymbol{\beta}\|_1 \leq \lambda} \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{y}_i - \boldsymbol{\beta}\|_2^2$      ▷ See Eq. (10.1)
else
    Draw $n \in \widetilde{\mathcal{O}}(d/\varepsilon^2)$ i.i.d. samples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \in \mathbb{R}^d$ from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$
    return $\widehat{\boldsymbol{\mu}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{y}_i$      ▷ Empirical mean
```

Theorem 10.1. *For any given $\varepsilon, \delta \in (0,1)$, $\eta \in [0, \frac{1}{4}]$, and $\widetilde{\boldsymbol{\mu}} \in \mathbb{R}^d$, the TestAndOptimizeMean algorithm uses $n \in \widetilde{\mathcal{O}}\left(\frac{d}{\varepsilon^2} \cdot (d^{-\eta} + \min\{1, f(\boldsymbol{\mu}, \widetilde{\boldsymbol{\mu}}, d, \eta, \varepsilon)\})\right)$, where*

$$f(\boldsymbol{\mu}, \widetilde{\boldsymbol{\mu}}, d, \eta, \varepsilon) = \frac{\|\boldsymbol{\mu} - \widetilde{\boldsymbol{\mu}}\|_1^2}{d^{1-4\eta}\varepsilon^2} \ ,$$

i.i.d. samples from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ for some unknown mean $\boldsymbol{\mu}$ and identity covariance $\boldsymbol{I}_d$, and can produce $\widehat{\boldsymbol{\mu}}$ in $\mathrm{poly}(n, d)$ time such that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) \leq \varepsilon$ with success probability at least $1 - \delta$.

Proof. Without loss of generality, we may assume that $\widetilde{\boldsymbol{\mu}} = \mathbf{0}$. This is because we can pre-process all samples by subtracting $\widetilde{\boldsymbol{\mu}}$ to yield i.i.d. samples from $N(\boldsymbol{\mu}', \boldsymbol{I}_d)$ where $\boldsymbol{\mu}' = \boldsymbol{\mu} - \widetilde{\boldsymbol{\mu}}$. Suppose we solved this problem to produce $\widehat{\boldsymbol{\mu}}'$ where $\mathrm{d_{TV}}(N(\boldsymbol{\mu}', \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}', \boldsymbol{I}_d)) \leq 10\varepsilon$, we can then output $\widehat{\boldsymbol{\mu}} = \widehat{\boldsymbol{\mu}}' + \widetilde{\boldsymbol{\mu}}$ and see from data processing inequality that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) = \mathrm{d_{TV}}(N(\boldsymbol{\mu}', \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}', \boldsymbol{I}_d)) \leq 10\varepsilon$; see the coupling characterization of TV in [DMR18].

Correctness of $\widehat{\boldsymbol{\mu}}$ output. Consider the TestAndOptimizeMean algorithm given in Algorithm 22. There are three possible outputs for $\widehat{\boldsymbol{\mu}}$:

1. $\widehat{\boldsymbol{\mu}} = \mathbf{0}$, which can only happen when `Outcome` is OK
2. $\widehat{\boldsymbol{\mu}} = \mathrm{argmin}_{\|\boldsymbol{\beta}\|_1 \leq \lambda} \frac{1}{n} \sum_{i=1}^{n} \|\boldsymbol{y}_i - \boldsymbol{\beta}\|_2^2$, which can only happen when `Outcome` is $\lambda \in \mathbb{R}$
3. $\widehat{\boldsymbol{\mu}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{y}_i$

Conditioned on ApproxL1 succeeding, with probability at least $1 - \delta$, we will show that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) \leq \varepsilon$ and failure probability at most δ in each of these cases, which implies the theorem statement.

1: When `Outcome` is OK, Lemma 10.9 tells us that $\|\boldsymbol{\mu}\|_2 \leq \alpha \leq \varepsilon$, with failure probability at most δ. So, by Theorem 2.18 and Lemma 2.29, we see that

$$\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) \leq \sqrt{\frac{1}{2} \cdot \mathrm{d_{KL}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d))} \\ = \sqrt{\frac{1}{4} \cdot \|\boldsymbol{\mu} - \mathbf{0}\|_2^2} \leq \sqrt{\frac{\varepsilon^2}{4}} \leq \varepsilon$$

2: Using $r = \lambda$ as the upper bound, Lemma 10.10 tells us that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) \leq \varepsilon$ with failure probability at most δ when $\widetilde{\mathcal{O}}(\lambda^2/\varepsilon^4)$ i.i.d. samples are used.

3: With $\widetilde{\mathcal{O}}(d/\varepsilon^2)$ samples, Lemma 2.25 tells us that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) \leq \varepsilon$ with failure probability at most δ.

Sample complexity used. By Definition 10.8, ApproxL1 uses $|\boldsymbol{S}| = m(k, \alpha, \delta') \in \widetilde{\mathcal{O}}(\sqrt{k}/\alpha^2)$ samples to produce `Outcome`. Then, ApproxL1 further uses $\widetilde{\mathcal{O}}(\lambda^2/\varepsilon^4)$ samples or $\widetilde{\mathcal{O}}(d/\varepsilon^2)$ samples depending on whether $\lambda < \varepsilon\sqrt{d}$. So, TestAndOptimizeMean has a total sample complexity of

$$\widetilde{\mathcal{O}}\left(\frac{\sqrt{k}}{\alpha^2} + \min\left\{\frac{\lambda^2}{\varepsilon^4}, \frac{d}{\varepsilon^2}\right\}\right) \tag{10.4}$$

Meanwhile, Lemma 10.9 states that $\|\boldsymbol{\mu}\|_1 \leq \lambda \leq 2\sqrt{k} \cdot (\lceil d/k \rceil \cdot \alpha + 2\|\boldsymbol{\mu}\|_1)$ whenever `Outcome` is $\lambda \in \mathbb{R}$. Since $(a+b)^2 \leq 2a^2 + 2b^2$ for any two real numbers $a, b \in \mathbb{R}$, we see that

$$\frac{\lambda^2}{\varepsilon^4} \in \mathcal{O}\left(\frac{k}{\varepsilon^4} \cdot \left(\frac{d^2\alpha^2}{k^2} + \|\boldsymbol{\mu}\|_1^2\right)\right) \subseteq \mathcal{O}\left(\frac{d}{\varepsilon^2} \cdot \left(\frac{d\alpha^2}{\varepsilon^2 k} + \frac{k \cdot \|\boldsymbol{\mu}\|_1^2}{d\varepsilon^2}\right)\right) \tag{10.5}$$

Putting together Eq. (10.4) and Eq. (10.5), we see that the total sample complexity is

$$\widetilde{\mathcal{O}}\left(\frac{\sqrt{k}}{\alpha^2} + \frac{d}{\varepsilon^2} \cdot \min\left\{1, \frac{d\alpha^2}{\varepsilon^2 k} + \frac{k \cdot \|\boldsymbol{\mu}\|_1^2}{d\varepsilon^2}\right\}\right)$$

Recalling that $\boldsymbol{\mu}$ in the analysis above actually refers to the pre-processed $\boldsymbol{\mu} - \widetilde{\boldsymbol{\mu}}$, and that TestAndOptimizeMean sets $k = \lceil d^{4\eta} \rceil$ and $\alpha = \varepsilon d^{-(1-3\eta)/2}$, with $0 \leq \eta \leq \frac{1}{4}$, the above expression simplifies to

$$\widetilde{\mathcal{O}}\left(\frac{d}{\varepsilon^2} \cdot \left(d^{-\eta} + \min\{1, f(\boldsymbol{\mu}, \widetilde{\boldsymbol{\mu}}, d, \eta, \varepsilon)\}\right)\right)$$

where $f(\boldsymbol{\mu}, \widetilde{\boldsymbol{\mu}}, d, \eta, \varepsilon) = \frac{\|\boldsymbol{\mu} - \widetilde{\boldsymbol{\mu}}\|_1^2}{d^{1-4\eta}\varepsilon^2}$. □

Remark on setting upper bound ζ. As ζ only affects the sample complexity logarithmically, one may be tempted to use a larger value than $\zeta = 4\varepsilon\sqrt{d}$. However, observe that running ApproxL1 with a larger upper bound than $\zeta = 4\varepsilon\sqrt{d}$ would not be helpful since $\|\boldsymbol{\mu}\|_2 > \zeta/4$ whenever ApproxL1 currently returns Fail and we have $\|\boldsymbol{\mu}\|_1 \leq \lambda$ whenever ApproxL1 returns $\lambda \in \mathbb{R}$. So, $\varepsilon\sqrt{d} = \zeta/4 < \|\boldsymbol{\mu}\|_2 \leq \|\boldsymbol{\mu}\|_1 \leq \lambda$ and TestAndOptimizeMean would have resorted to using the empirical mean anyway.

10.5 TestAndOptimizeCovariance for the general covariance setting

We will later define analogs of $m(d, \alpha, \delta)$ and ApproxL1 from Section 10.4 to the unknown covariance setting: $m'(d, \alpha, \delta)$ and VectorizedApproxL1 respectively. Then, after stating the guarantees of VectorizedApproxL1, we show how to use them according to the strategy outlined in Section 10.3.2.

For the rest of this section, we assume that we get i.i.d. samples from $N(\mathbf{0}, \Sigma)$ and also that Σ is full rank. These are without loss of generality for the following reasons:

- Instead of a single sample from $N(\boldsymbol{\mu}, \Sigma)$, we will draw two samples $\boldsymbol{x}_1, \boldsymbol{x}_2 \sim N(\boldsymbol{\mu}, \Sigma)$ and consider $\boldsymbol{x}' = \frac{\boldsymbol{x}_1 + \boldsymbol{x}_2}{\sqrt{2}}$. One can check that $\boldsymbol{x}'$ is distributed according to $N(\mathbf{0}, \Sigma)$ and we only use a multiplicative factor of 2 additional samples, which is subsumed in the big-O.

- By Lemma 2.26, the empirical covariance constructed from d i.i.d. samples of $N(\mathbf{0}, \Sigma)$ will have the same rank as Σ itself, with probability at least $1 - \delta$. So, we can simply project and solve the problem on the full rank subspace of the empirical covariance matrix.

10.5.1 The adjustments

To begin, we elaborate on the adjustments mentioned in Section 10.3.2 to adapt the approach from the identity covariance setting to the unknown covariance setting. The formal proofs of the following two adjustment lemmas are deferred to Appendix C.2.3.

The first adjustment relates to performing a suitable preconditioning process using an additional d samples so that we can subsequently argue that $\lambda_{\text{min}}(\Sigma) \geq 1$. The idea is as follows: we will compute a preconditioning matrix $\boldsymbol{A}$ using d i.i.d. samples such that $\boldsymbol{A}\Sigma\boldsymbol{A}$ has eigenvalues at least 1, i.e. $\lambda_{\text{min}}(\boldsymbol{A}\Sigma\boldsymbol{A}) \geq 1$. That is, $\|(\boldsymbol{A}\Sigma\boldsymbol{A})^{-1}\|_2 = \frac{1}{\lambda_{\text{min}}(\boldsymbol{A}\Sigma\boldsymbol{A})} \leq 1$. Then, we solve the problem treating $\boldsymbol{A}\Sigma\boldsymbol{A}$ as our new Σ. This adjustment succeeds with probability at least $1 - \delta$ for any given $\delta \in (0, 1)$ and is possible because, with probability 1, the empirical covariance $\widehat{\Sigma}$ formed by using d i.i.d. samples would have the same eigenspace as $\mathbf{\Sigma}$, and so we would have a bound on the ratios between the minimum eigenvalues between $\widehat{\mathbf{\Sigma}}$ and $\mathbf{\Sigma}$; see Lemma 2.26.

Lemma 10.11. *For any $\delta \in (0,1)$, there is an explicit preconditioning process that uses d i.i.d. samples from $N(\mathbf{0}, \mathbf{\Sigma})$ and succeeds with probability at least $1-\delta$ in constructing a matrix $\boldsymbol{A} \in \mathbb{R}^{d \times d}$ such that $\lambda_{\min}(\boldsymbol{A}\mathbf{\Sigma}\boldsymbol{A}) \geq 1$. Furthermore, for any full rank PSD matrix $\widetilde{\mathbf{\Sigma}} \in \mathbb{R}^{d \times d}$, we have $\|(\boldsymbol{A}\widetilde{\mathbf{\Sigma}}\boldsymbol{A})^{-1/2}\boldsymbol{A}\mathbf{\Sigma}\boldsymbol{A}(\boldsymbol{A}\widetilde{\mathbf{\Sigma}}\boldsymbol{A})^{-1/2} - \boldsymbol{I}_d\| = \|\widetilde{\mathbf{\Sigma}}^{-1/2}\mathbf{\Sigma}\widetilde{\mathbf{\Sigma}}^{-1/2} - \boldsymbol{I}_d\|$.*

The matrix $\boldsymbol{A}$ in Lemma 10.11 is essentially constructed by combining the eigenspace corresponding to "large eigenvalues" with a suitably upscaled eigenspace corresponding to "small eigenvalues" in the empirical covariance matrix obtained by d i.i.d. samples and relying on Lemma 2.26 for correctness arguments.

The second adjustment relates to showing that the partitioning idea also works for obtaining sample efficient ℓ_1 estimates of $\mathrm{vec}(\mathbf{\Sigma} - \boldsymbol{I}_d)$. While an existence result suffices, we show that a simple probabilistic construction will in fact succeed with high probability.

Lemma 10.12. *Fix dimension $d \geq 2$ and group size $k \leq d$. Consider the $q = 2$ setting where $\boldsymbol{T} \in \mathbb{R}^{d \times d}$ is a matrix. Define $w = \frac{10d(d-1)\log d}{k(k-1)}$. Pick sets $\boldsymbol{B}_1, \ldots, \boldsymbol{B}_w$ each of size k uniformly at random (with replacement) from all the possible $\binom{d}{k}$ sets. With high probability in d, this is a $(q = 2, d, k, a = 1, b = \frac{30(d-1)\log d}{(k-1)})$-partitioning scheme.*

We can obtain a $(q = 2, d, k, a = 1, b = \mathcal{O}(\frac{d \log d}{k}))$-partitioning scheme by repeating the construction of Lemma 10.12 until it satisfies required conditions. Since it succeeds with high probability in d, we should not need many tries. The key idea behind utilizing partitioning schemes is that the marginal over a subset of indices $\boldsymbol{B} \subseteq [d]$ of a d-dimensional Gaussian with covariance matrix $\mathbf{\Sigma}$ has covariance matrix that is the principal submatrix $\mathbf{\Sigma}_{\boldsymbol{B}}$ of $\mathbf{\Sigma}$. So, if we can obtain a multiplicative α-approximation of a collection of principal submatrices $\mathbf{\Sigma}_{\boldsymbol{B}_1}, \ldots \mathbf{\Sigma}_{\boldsymbol{B}_w}$ such that all cells of $\mathbf{\Sigma}$ are present, then we can obtain a multiplicative α-approximation of $\mathbf{\Sigma}$ just like in Section 10.4. Meanwhile, the b parameter allows us to upper bound the overestimation factor due to repeated occurrences of any cell of $\mathbf{\Sigma}$.

10.5.2 Following the approach from the identity covariance setting

We begin by defining a parameterized sample count $m'(d, \varepsilon, \delta)$, similar to Definition 10.8.

Definition 10.13. Fix any $d \geq 1$, $\varepsilon > 0$, and $\delta \in (0,1)$. We define $m'(d, \varepsilon, \delta) = n'_{d,\varepsilon} \cdot r_\delta$, where

$$n'_{d,\varepsilon} = \left\lceil 3200d \cdot \max\left\{\frac{1}{\varepsilon^2}, \frac{1}{\varepsilon}, 1\right\}\right\rceil \qquad \text{and} \qquad r_\delta = 1 + \left\lceil \log\left(\frac{12}{\delta}\right)\right\rceil$$

The VectorizedApproxL1 algorithm corresponds to ApproxL1 in Section 10.4: it performs an exponential search to find the 2-approximation of the $\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2$ by repeatedly invoking the tolerant tester from Lemma 10.6 and then utilize a suitable partitioning scheme to bound $\|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1$; see Lemma 10.12 and the discussions below it.

Algorithm 23 The VectorizedApproxL1 algorithm.

```
Input: Error rate $\varepsilon > 0$, failure rate $\delta \in (0,1)$, block size $k \in [d]$, lower bound $\alpha > 0$,
upper bound $\zeta > 2\alpha$, and i.i.d. samples $\mathcal{S}$ from $N(\mathbf{0}, \boldsymbol{\Sigma})$
Output: Fail, OK, or $\lambda \in \mathbb{R}$
Define $w = \frac{10d(d-1)\log d}{k(k-1)}$, $\delta' = \frac{\delta}{w \cdot \lceil \log_2 \zeta/\alpha \rceil}$, and let $\boldsymbol{B}_1, \ldots, \boldsymbol{B}_w \subseteq [d]^2$ be a ($q = 2, d, k, a = 1, b = \mathcal{O}(\frac{d \log d}{k})$)-partitioning scheme as per Lemma 10.12
for $j \in \{1, \ldots, w\}$ do
    Define $\boldsymbol{S}_{\boldsymbol{B}_j} = \{\boldsymbol{x}_{\boldsymbol{B}_j} \in \mathbb{R}^{|\boldsymbol{B}_j|} : \boldsymbol{x} \in \boldsymbol{S}\}$ as the projected samples
                                                                ▷ See Definition 2.4
    Initialize $o_j = $ Fail
    for $i = 1, 2, \ldots, \lceil \log_2 \zeta/\alpha \rceil$ do
        Define $l_i = 2^{i-1} \cdot \alpha$
        Let Outcome be the output of the tolerant tester of Lemma 10.6 using sample set $\mathcal{S}_{\boldsymbol{B}_j}$ with $\varepsilon_1 = l_i$, $\varepsilon_2 = 2l_i$, and $\delta = \delta'$
        if Outcome is Accept then
            Set $o_j = l_i$ and break                        ▷ Escape inner loop for block $j$
if there exists a Fail amongst $\{o_1, \ldots, o_w\}$ then
    return Fail
else if $4b \sum_{j=1}^{w} o_j^2 \leq \alpha^2$ then
    return OK
else
    return $\lambda = 2 \sum_{j=1}^{w} \sqrt{|\boldsymbol{B}_j|} \cdot o_j$          ▷ $\lambda$ is an estimate for $\|vec(\Sigma - \boldsymbol{B}_d)\|_1$
```

One can show that the VectorizedApproxL1 algorithm has the following guarantees.

Lemma 10.14. *Let ε, δ, k, α, and ζ be the input parameters to the VectorizedApproxL1 algorithm (Algorithm 23). Given $m(k, \alpha, \delta')$ i.i.d. samples from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$, the VectorizedApproxL1 algorithm succeeds with probability at least $1 - \delta$ and has the following properties:*

- *If VectorizedApproxL1 outputs* Fail, *then $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 > \zeta/2$.*
- *If VectorizedApproxL1 outputs* OK, *then $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \leq \alpha^2$.*
- *If VectorizedApproxL1 outputs $\lambda \in \mathbb{R}$, then*

$$\|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1 \leq \lambda \leq 2\sqrt{k} \cdot \left(\frac{10d(d-1)\log d}{k(k-1)} \cdot \alpha + 2\|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1 \right)$$

Proof. We begin by stating some properties of $o_1, \ldots, o_w$. Fix an arbitrary index $j \in \{1, \ldots, w\}$ and suppose o_j is *not* a Fail, i.e. the tolerant tester of Lemma 10.6 outputs Accept for some $i^* \in \{1, 2, \ldots, \lceil \log_2 \zeta/\alpha \rceil\}$. Note that VectorizedApproxL1 sets $o_j = \ell_{i^*}$ and the tester outputs Reject for all smaller indices $i \in \{1, \ldots, i^* - 1\}$. Since the tester outputs Accept for i^*, we have that $\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F \leq 2\ell_{i^*} = 2o_j$. Meanwhile, if $i^* > 1$, then $\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F > \ell_{i^*-1} = \ell_{i^*}/2 = o_j/2$ since the tester outputs Reject for $i^* - 1$. Thus, we see that

- When o_j is not Fail, we have $\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F \leq 2o_j$.
- When $\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F \leq 2\alpha$, we have $i^* = 1$ and $o_j = \ell_1 = \alpha$.
- When $\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F > 2\alpha = 2\ell_1$, we have $i^* > 1$ and so $o_j < 2\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F$.

Success probability. Fix an arbitrary index $i \in \{1, 2, \ldots, \lceil \log_2 \zeta/\alpha \rceil\}$ with $\ell_i = 2^{i-1}\alpha$, where $\ell_i \leq \ell_1 = \alpha$ for any i. We invoke the tolerant tester with $\varepsilon_2 = 2\ell_i = 2\varepsilon_1$, so the i^{th} invocation uses at most $n'_{k,\varepsilon} \cdot r_\delta$ i.i.d. samples to succeed with probability at least $1 - \delta$; see Definition 10.13 and Algorithm 31. So, with $m(k, \alpha, \delta')$ samples, *any* call to the tolerant tester succeeds with probability at least $1 - \delta'$, where $\delta' = \frac{\delta}{w \cdot \lceil \log_2 \zeta/\alpha \rceil}$. By construction, there will be at most $w \cdot \lceil \log_2 \zeta/\alpha \rceil$ calls to the tolerant tester. Therefore, by union bound, *all* calls to the tolerant tester jointly succeed with probability at least $1 - \delta$.

Property 1. When VectorizedApproxL1 outputs Fail, there exists a Fail amongst $\{o_1, \ldots, o_w\}$. For any fixed index $j \in \{1, \ldots, w\}$, this can only happen when all calls to the tolerant tester outputs Reject. This means that $\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F > \varepsilon_1 = \ell_i = 2^{i-1} \cdot \alpha$ for all $i \in \{1, 2, \ldots, \lceil \log_2 \zeta/\alpha \rceil\}$. In particular, this means that $\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F > \zeta/2$.

Property 2. When VectorizedApproxL1 outputs OK, we have $4b \sum_{j=1}^{w} o_j^2 \leq \alpha^2$. Then, since $\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F \leq 2o_j$ for each index $j \in \{1, \ldots, w\}$ and since each cell in $\mathbf{\Sigma}$ appears at most b times across all submatrices $\mathbf{\Sigma}_{\mathbf{B}_1}, \ldots, \mathbf{\Sigma}_{\mathbf{B}_w}$, we see that

$$\|\mathbf{\Sigma} - \mathbf{I}_d\|_F^2 \leq b \cdot \sum_{j=1}^{w} \|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F^2 \leq b \cdot \sum_{j=1}^{w} (2o_j)^2 \leq \alpha^2$$

That is, $\|\mathbf{\Sigma} - \mathbf{I}_d\|_F^2 \leq \alpha^2$ as desired.

Property 3. When VectorizedApproxL1 outputs $\lambda = 2\sum_{j=1}^{w} \sqrt{|\mathbf{B}_j|} \cdot o_j \in \mathbb{R}$, we have $4b \sum_{j=1}^{w} o_j^2 > \alpha^2$. We can lower bound λ as follows:

$$\begin{aligned}
\lambda &= 2\sum_{j=1}^{w} \sqrt{|\mathbf{B}_j|} \cdot o_j \\
&\geq 2\sum_{j=1}^{w} \sqrt{|\mathbf{B}_j|} \cdot \frac{\|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F}{2} && \text{(since } \|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F \leq 2o_j\text{)} \\
&= \sum_{j=1}^{w} \sqrt{|\mathbf{B}_j| \cdot \|\mathrm{vec}(\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d)\|_2^2} && \text{(since } \|\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d\|_F^2 = \|\mathrm{vec}(\mathbf{\Sigma}_{\mathbf{B}_j} - \mathbf{I}_d)\|_2^2\text{)}
\end{aligned}$$

$$\geq \sum_{j=1}^{w} \|\mathrm{vec}(\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_1 \qquad \text{(since } \|\mathrm{vec}(\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_1^2 \leq |\boldsymbol{B}_j| \cdot \|\mathrm{vec}(\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_2^2\text{)}$$

$$\geq \|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1$$

(Since each cell in $\boldsymbol{\Sigma}$ appears at least $a = 1$ times across all submatrices $\boldsymbol{\Sigma}_{\boldsymbol{B}_1}, \ldots, \boldsymbol{\Sigma}_{\boldsymbol{B}_w}$)

That is, $\lambda \geq \|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1$. Meanwhile, we can also upper bound λ as follows:

$$\lambda = 2 \sum_{j=1}^{w} \sqrt{|\boldsymbol{B}_j|} \cdot o_j$$

$$\leq 2\sqrt{k} \cdot \sum_{j=1}^{w} o_j \qquad \text{(since } |\boldsymbol{B}_j| \leq k\text{)}$$

$$= 2\sqrt{k} \cdot \left(\sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha}}^{w} o_j + \sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F > 2\alpha}}^{w} o_j \right)$$

(partitioning based on $\|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F$ versus 2α)

$$= 2\sqrt{k} \cdot \left(\sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha}}^{w} \alpha + \sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F > 2\alpha}}^{w} o_j \right)$$

(since $\|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha$ implies $o_j = \alpha$)

$$\leq 2\sqrt{k} \cdot \left(\sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha}}^{w} \alpha + 2 \sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F^2 \leq 2\alpha}}^{w} \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \right)$$

(since $\|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F > 2\alpha$ implies $o_j \leq 2\|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F$)

$$= 2\sqrt{k} \cdot \left(\sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha}}^{w} \alpha + 2 \sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha}}^{w} \|\mathrm{vec}(\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_2 \right)$$

(since $\|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F^2 = \|\mathrm{vec}(\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_2^2$)

$$\leq 2\sqrt{k} \cdot \left(\sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha}}^{w} \alpha + 2 \sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha}}^{w} \|\mathrm{vec}(\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_1 \right)$$

(since $\|\mathrm{vec}(\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_2 \leq \|\mathrm{vec}(\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_1$)

$$\leq 2\sqrt{k} \cdot \left(w\alpha + 2 \sum_{\substack{j=1 \\ \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F^2 \leq 2\alpha}}^{w} \|\mathrm{vec}(\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_1 \right)$$

(since $|\{j \in [w] : \|\boldsymbol{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha\}| \leq w$)

$$\leq 2\sqrt{k} \cdot (w\alpha + 2\|\mathrm{vec}(\mathbf{\Sigma} - \boldsymbol{I}_d)\|_1)$$

$$\text{(since } \sum_{\substack{j=1 \\ \|\mathbf{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d\|_F \leq 2\alpha}}^{w} \|\mathrm{vec}(\mathbf{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_1 \leq \sum_{j=1}^{w} \|\mathrm{vec}(\mathbf{\Sigma}_{\boldsymbol{B}_j} - \boldsymbol{I}_d)\|_1 = \|\mathrm{vec}(\mathbf{\Sigma} - \boldsymbol{I}_d)\|_1)$$

That is, $\lambda \leq 2\sqrt{k} \cdot (w\alpha + 2\|\mathrm{vec}(\mathbf{\Sigma} - \boldsymbol{I}_d)\|_1)$, where $w = \frac{10d(d-1)\log d}{k(k-1)}$. The property follows by putting together both bounds. □

Now, suppose VectorizedApproxL1 tells us that $\|\mathrm{vec}(\mathbf{\Sigma} - \boldsymbol{I}_d)\|_1 \leq r$. We can then construct a SDP to search for a candidate $\widehat{\mathbf{\Sigma}} \in \mathbb{R}^{d \times d}$ using i.i.d. samples from $N(\mathbf{0}, \mathbf{\Sigma})$.

Lemma 10.15. *Fix $d \geq 1$, $r \geq 0$, and $\varepsilon, \delta > 0$. Given $\mathcal{O}\left(\frac{r^2}{\varepsilon^4} \log \frac{1}{\delta} + \frac{d + \sqrt{d \log(1/\delta)}}{\varepsilon^2}\right)$ i.i.d. samples from $N(\mathbf{0}, \mathbf{\Sigma})$ for some unknown $\mathbf{\Sigma} \in \mathbb{R}^{d \times d}$ with $\|\mathrm{vec}(\mathbf{\Sigma} - \boldsymbol{I}_d)\|_1 \leq r$, one can produce estimates $\widehat{\boldsymbol{\mu}} \in \mathbb{R}^d$ and $\widehat{\mathbf{\Sigma}} \in \mathbb{R}^{d \times d}$ in $\mathrm{poly}(n, d, \log(1/\varepsilon))$ time such that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \mathbf{\Sigma}), N(\widehat{\boldsymbol{\mu}}, \widehat{\mathbf{\Sigma}})) \leq \varepsilon$ with success probability at least $1 - \delta$.*

Proof. Suppose we get n samples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \sim N(\mathbf{0}, \mathbf{\Sigma})$. For $i \in [n]$, we can re-express each $\boldsymbol{y}_i$ as $\boldsymbol{y}_i = \mathbf{\Sigma}^{1/2} \boldsymbol{g}_i$, for some $\boldsymbol{g}_i \sim N(\mathbf{0}, \boldsymbol{I}_d)$. Let us define $\boldsymbol{T} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{g}_i \boldsymbol{g}_i^\top$ and $\boldsymbol{S} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{y}_i \boldsymbol{y}_i^\top = \mathbf{\Sigma}^{1/2} \left(\frac{1}{n} \sum_{i=1}^{n} \boldsymbol{g}_i \boldsymbol{g}_i^\top\right) \mathbf{\Sigma}^{1/2} = \mathbf{\Sigma}^{1/2} \boldsymbol{T} \mathbf{\Sigma}^{1/2}$.

Let us define $\widehat{\mathbf{\Sigma}} \in \mathbb{R}^{d \times d}$ as follows:

$$\widehat{\mathbf{\Sigma}} = \operatorname*{argmin}_{\substack{\boldsymbol{A} \in \mathbb{R}^{d \times d} \text{ is p.s.d.} \\ \|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq r \\ \lambda_{\min}(\boldsymbol{A}) \geq 1}} \sum_{i=1}^{n} \|\boldsymbol{A} - \boldsymbol{y}_i \boldsymbol{y}_i^\top\|_F^2 \tag{10.6}$$

Observe that $\mathbf{\Sigma}$ is a feasible solution to Eq. (10.6). We show in Appendix C.2.3 that Eq. (10.6) is a semidefinite program (SDP) that is polynomial time solvable.

Since $\mathbf{\Sigma}$ and $\widehat{\mathbf{\Sigma}}$ are symmetric p.s.d. matrices, observe that

$$\begin{aligned}
\sum_{i=1}^{n} \|\widehat{\mathbf{\Sigma}} - \boldsymbol{y}_i \boldsymbol{y}_i^\top\|_F^2 &= \sum_{i=1}^{n} \|\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}^{1/2} \boldsymbol{g}_i \boldsymbol{g}_i^\top \mathbf{\Sigma}^{1/2}\|_F^2 && \text{(Since } \boldsymbol{y}_i = \mathbf{\Sigma}^{1/2} \boldsymbol{g}_i) \\
&= \sum_{i=1}^{n} \mathrm{Tr}\left(\left(\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}^{1/2} \boldsymbol{g}_i \boldsymbol{g}_i^\top \mathbf{\Sigma}^{1/2}\right)^\top \left(\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}^{1/2} \boldsymbol{g}_i \boldsymbol{g}_i^\top \mathbf{\Sigma}^{1/2}\right)\right) \\
&&& \text{(Since } \|\boldsymbol{A}\|_F^2 = \mathrm{Tr}(\boldsymbol{A}^\top \boldsymbol{A}) \text{ for any matrix } \boldsymbol{A}) \\
&= \sum_{i=1}^{n} \mathrm{Tr}\left(\widehat{\mathbf{\Sigma}}^2 - 2 \boldsymbol{g}_i \boldsymbol{g}_i^\top \mathbf{\Sigma}^{1/2} \widehat{\mathbf{\Sigma}} \mathbf{\Sigma}^{1/2} + \boldsymbol{g}_i \boldsymbol{g}_i^\top \mathbf{\Sigma} \boldsymbol{g}_i \boldsymbol{g}_i^\top \mathbf{\Sigma}\right) \\
&&& \text{(Expanding and applying cyclic property of trace)}
\end{aligned}$$

Similarly, by replacing $\widehat{\mathbf{\Sigma}}$ with $\mathbf{\Sigma}$, we see that

$$\sum_{i=1}^{n} \|\mathbf{\Sigma} - \boldsymbol{y}_i \boldsymbol{y}_i^\top\|_F^2 = \sum_{i=1}^{n} \mathrm{Tr}\left(\mathbf{\Sigma}^2 - 2 \boldsymbol{g}_i \boldsymbol{g}_i^\top \mathbf{\Sigma}^2 + \boldsymbol{g}_i \boldsymbol{g}_i^\top \mathbf{\Sigma} \boldsymbol{g}_i \boldsymbol{g}_i^\top \mathbf{\Sigma}\right)$$

By standard SDP results (e.g. see [VB96, Fre04, GM12]), Eq. (10.6) can be solved optimally up to additive ε in the objective function. We show explicitly in Appendix C.2.3 that our problem can be transformed into a SDP and be solved in $\mathrm{poly}(n, d, \log(1/\varepsilon))$ time. Since we solve up to additive ε in the objective function, we have

$$\sum_{i=1}^{n} \|\widehat{\mathbf{\Sigma}} - \boldsymbol{y}_i \boldsymbol{y}_i^\top\|_F^2 \leq \varepsilon + \sum_{i=1}^{n} \|\mathbf{\Sigma} - \boldsymbol{y}_i \boldsymbol{y}_i^\top\|_F^2 \tag{10.7}$$

which implies that

$$\sum_{i=1}^{n} \mathrm{Tr}\left(\widehat{\mathbf{\Sigma}}^2 - 2\boldsymbol{g}_i\boldsymbol{g}_i^\top \mathbf{\Sigma}^{1/2}\widehat{\mathbf{\Sigma}}\mathbf{\Sigma}^{1/2} + \boldsymbol{g}_i\boldsymbol{g}_i^\top \mathbf{\Sigma}\boldsymbol{g}_i\boldsymbol{g}_i^\top \mathbf{\Sigma}\right) \leq \varepsilon + \sum_{i=1}^{n} \mathrm{Tr}\left(\mathbf{\Sigma}^2 - 2\boldsymbol{g}_i\boldsymbol{g}_i^\top \mathbf{\Sigma}^2 + \boldsymbol{g}_i\boldsymbol{g}_i^\top \mathbf{\Sigma}\boldsymbol{g}_i\boldsymbol{g}_i^\top \mathbf{\Sigma}\right)$$

Cancelling the common $\boldsymbol{g}_i\boldsymbol{g}_i^\top \mathbf{\Sigma}\boldsymbol{g}_i\boldsymbol{g}_i^\top \mathbf{\Sigma}$ term and rearranging, we get

$$\mathrm{Tr}\left(\widehat{\mathbf{\Sigma}}^2 - \mathbf{\Sigma}^2\right) \leq \frac{\varepsilon}{n} + \frac{2}{n}\sum_{i=1}^{n} \mathrm{Tr}\left(\boldsymbol{g}_i\boldsymbol{g}_i^\top \left(\mathbf{\Sigma}^{1/2}\widehat{\mathbf{\Sigma}}\mathbf{\Sigma}^{1/2} - \mathbf{\Sigma}^2\right)\right) \tag{10.8}$$

Therefore,

$$\begin{aligned}
\|\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}\|_F^2 &= \mathrm{Tr}\left(\left(\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}\right)^\top \left(\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}\right)\right) \\
&= \mathrm{Tr}\left(\widehat{\mathbf{\Sigma}}^2 - 2\widehat{\mathbf{\Sigma}}\mathbf{\Sigma} + \mathbf{\Sigma}^2\right) \\
&\leq \frac{\varepsilon}{n} + \frac{2}{n}\sum_{i=1}^{n} \mathrm{Tr}\left(\boldsymbol{g}_i\boldsymbol{g}_i^\top \left(\mathbf{\Sigma}^{1/2}\widehat{\mathbf{\Sigma}}\mathbf{\Sigma}^{1/2} - \mathbf{\Sigma}^2\right) - \widehat{\mathbf{\Sigma}}\mathbf{\Sigma} + \mathbf{\Sigma}^2\right) \\
&\qquad\qquad \text{(Add } 2\mathbf{\Sigma}^2 - 2\widehat{\mathbf{\Sigma}}\mathbf{\Sigma} \text{ to both sides of Eq. (10.8))} \\
&= \frac{\varepsilon}{n} + \frac{2}{n}\sum_{i=1}^{n} \mathrm{Tr}\left(\left(\boldsymbol{g}_i\boldsymbol{g}_i^\top - \boldsymbol{I}_d\right) \cdot \left(\mathbf{\Sigma}^{1/2}\widehat{\mathbf{\Sigma}}\mathbf{\Sigma}^{1/2} - \mathbf{\Sigma}^2\right)\right) \\
&\qquad\qquad \text{(Since } \mathrm{Tr}(\widehat{\mathbf{\Sigma}}\mathbf{\Sigma}) = \mathrm{Tr}(\mathbf{\Sigma}^{1/2}\widehat{\mathbf{\Sigma}}\mathbf{\Sigma}^{1/2})\text{)} \\
&= \frac{\varepsilon}{n} + 2 \cdot \mathrm{Tr}\left(\left(\mathbf{\Sigma}^{1/2}\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}^{1/2}\mathbf{\Sigma}\right) \cdot \mathbf{\Sigma}^{1/2} \cdot \left(\left(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{g}_i\boldsymbol{g}_i^\top\right) - \boldsymbol{I}_d\right)\right) \\
&\qquad\qquad \text{(Rearranging with cyclic property of trace)} \\
&= \frac{\varepsilon}{n} + 2 \cdot \mathrm{Tr}\left(\left(\mathbf{\Sigma}^{1/2}\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}^{1/2}\mathbf{\Sigma}\right) \cdot \mathbf{\Sigma}^{1/2} \cdot \left(\boldsymbol{T} - \boldsymbol{I}_d\right)\right) \\
&\qquad\qquad \text{(Since } \boldsymbol{T} = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \boldsymbol{g}_i\boldsymbol{g}_i^\top\text{)} \\
&\leq \frac{\varepsilon}{n} + 2 \cdot \left\|\mathrm{vec}\left(\mathbf{\Sigma}\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}^2\right)\right\|_1 \cdot \|\boldsymbol{T} - \boldsymbol{I}_d\|_2 \\
&\qquad \text{(By Lemma 2.7 with } \boldsymbol{A} = \mathbf{\Sigma}^{1/2}\widehat{\mathbf{\Sigma}} - \mathbf{\Sigma}^{1/2}\mathbf{\Sigma}\text{, } \boldsymbol{B} = \mathbf{\Sigma}^{1/2}\text{, and } \boldsymbol{C} = \boldsymbol{T} - \boldsymbol{I}_d\text{)}
\end{aligned}$$

Lemma 2.28 tells us that $\Pr\left(\|\boldsymbol{T} - \boldsymbol{I}_d\|_2 > \varepsilon\right) \leq 2\exp(-t^2 d)$ when the number of samples $n = \frac{c_0}{\varepsilon^2}\log\frac{2}{\delta}$, for some absolute constant c_0. So, to complete the proof, it suffices to upper bound $\left\|\operatorname{vec}\left(\boldsymbol{\Sigma}\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}^2\right)\right\|_1$. Consider the following:

$$\begin{aligned}
\left\|\operatorname{vec}\left(\boldsymbol{\Sigma}\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}^2\right)\right\|_1 &= \left\|\operatorname{vec}\left((\boldsymbol{I}_d - \boldsymbol{\Sigma})(\boldsymbol{\Sigma} - \widehat{\boldsymbol{\Sigma}}) - \boldsymbol{\Sigma} + \widehat{\boldsymbol{\Sigma}}\right)\right\|_1 \\
&\leq \|\operatorname{vec}(\boldsymbol{I}_d - \boldsymbol{\Sigma})\|_1 \cdot \left\|\operatorname{vec}(\boldsymbol{\Sigma} - \widehat{\boldsymbol{\Sigma}})\right\|_1 + \left\|\operatorname{vec}(\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma})\right\|_1 \\
&\qquad \text{(By Lemma 2.8)} \\
&= (\|\operatorname{vec}(\boldsymbol{I}_d - \boldsymbol{\Sigma})\|_1 + 1) \cdot \left\|\operatorname{vec}(\widehat{\boldsymbol{\Sigma}} - \boldsymbol{I}_d + \boldsymbol{I}_d - \boldsymbol{\Sigma})\right\|_1 \\
&\qquad \text{(Rearranging and adding 0)} \\
&\leq (\|\operatorname{vec}(\boldsymbol{I}_d - \boldsymbol{\Sigma})\|_1 + 1) \cdot \left(\|\operatorname{vec}(\widehat{\boldsymbol{\Sigma}} - \boldsymbol{I}_d)\|_1 + \|\operatorname{vec}(\boldsymbol{I}_d - \boldsymbol{\Sigma})\|_1\right) \\
&\qquad \text{(By Lemma 2.8)} \\
&\leq (r+1) \cdot 2r \\
&\qquad \text{(Since } \|\operatorname{vec}(\boldsymbol{I}_d - \boldsymbol{\Sigma})\|_1 \leq r \text{ and } \left\|\operatorname{vec}(\widehat{\boldsymbol{\Sigma}} - \boldsymbol{I}_d)\right\|_1 \leq r\text{)}
\end{aligned}$$

When $\frac{2}{\varepsilon} \leq n$ and $n \in \mathcal{O}\left(\frac{r^2}{\varepsilon^4}\log\frac{1}{\delta}\right)$, the following holds with probability at least $1 - \delta$:

$$\begin{aligned}
\|\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}\|_F^2 &\leq \frac{\varepsilon}{n} + 2 \cdot \left\|\operatorname{vec}\left(\boldsymbol{\Sigma}\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}^2\right)\right\|_1 \cdot \|\boldsymbol{T} - \boldsymbol{I}_d\|_2 \\
&\leq \frac{\varepsilon}{n} + 4r(r+1) \cdot \|\boldsymbol{T} - \boldsymbol{I}_d\|_2 \leq \frac{\varepsilon}{n} + \frac{\varepsilon^2}{2} \leq \varepsilon^2
\end{aligned}$$

Now, Lemma 2.25 tells us that the empirical mean $\widehat{\boldsymbol{\mu}}$ formed using $\mathcal{O}\left(\frac{d + \sqrt{d\log(1/\delta)}}{\varepsilon^2}\right)$ samples satisfies $(\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}) \leq \varepsilon^2$, with failure probability at most δ. So,

$$\begin{aligned}
&\mathrm{d}_{\mathrm{KL}}(N(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}}), N(\boldsymbol{\mu}, \boldsymbol{\Sigma})) \\
&= \frac{1}{2} \cdot \left(\operatorname{Tr}(\boldsymbol{\Sigma}^{-1}\widehat{\boldsymbol{\Sigma}}) - d + (\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}}) + \ln\left(\frac{\det \boldsymbol{\Sigma}}{\det \widehat{\boldsymbol{\Sigma}}}\right)\right) \\
&\leq \frac{1}{2} \cdot \left((\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}}) + \|\boldsymbol{\Sigma}^{-1/2}\widehat{\boldsymbol{\Sigma}}\boldsymbol{\Sigma}^{-1/2} - \boldsymbol{I}_d\|_F^2\right) \qquad \text{(By Lemma 2.29)} \\
&= \frac{1}{2} \cdot \left((\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}})^\top \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}}) + \|\widehat{\boldsymbol{\Sigma}}\boldsymbol{\Sigma}^{-1} - \boldsymbol{I}_d\|_F^2\right) \qquad \text{(By Lemma 2.9)} \\
&\leq \frac{1}{2} \cdot \left(\varepsilon^2 + \|\widehat{\boldsymbol{\Sigma}}\boldsymbol{\Sigma}^{-1} - \boldsymbol{I}_d\|_F^2\right) \\
&\qquad \text{(Since } (\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}) \leq \varepsilon\text{, with probability at least } 1 - \delta\text{)} \\
&\leq \frac{1}{2} \cdot \left(\varepsilon^2 + \|\boldsymbol{\Sigma}^{-1}\|_2^2 \cdot \|\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}\|_F^2\right) \qquad \text{(Submultiplicativity of Frobenius norm)} \\
&\leq \frac{1}{2} \cdot \left(\varepsilon^2 + \|\widehat{\boldsymbol{\Sigma}} - \boldsymbol{\Sigma}\|_F^2\right) \qquad \text{(Since } \|\boldsymbol{\Sigma}^{-1}\|_2 = \tfrac{1}{\lambda_{\min}(\boldsymbol{\Sigma})} \leq 1\text{)} \\
&\leq \frac{1}{2} \cdot \left(\varepsilon^2 + \varepsilon^2\right) \qquad \text{(From above, with probability at least } 1 - \delta\text{)}
\end{aligned}$$

$$= \varepsilon^2$$

By union bound, the above events jointly hold with probability at least $1 - 2\delta$. Thus, by symmetry of TV distance and Theorem 2.18, we see that

$$\begin{aligned} \mathrm{d}_{\mathrm{TV}}(N(\boldsymbol{\mu}, \boldsymbol{I}_d), N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d)) &= \mathrm{d}_{\mathrm{TV}}(N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d), N(\boldsymbol{\mu}, \boldsymbol{I}_d)) \\ &\leq \sqrt{\frac{1}{2} \mathrm{d}_{\mathrm{KL}}(N(\widehat{\boldsymbol{\mu}}, \boldsymbol{I}_d), N(\boldsymbol{\mu}, \boldsymbol{I}_d))} \leq \sqrt{\varepsilon^2} = \varepsilon \end{aligned}$$

The claim holds by repeating the same argument after scaling δ by an appropriate constant. □

Algorithm 24 The TestAndOptimizeCovariance algorithm.

```
Input: Error rate $\varepsilon > 0$, failure rate $\delta \in (0,1)$, parameter $\eta \in [0,1]$, and sample access to $N(\mathbf{0}, \boldsymbol{\Sigma})$
Output: $\widehat{\boldsymbol{\Sigma}} \in \mathbb{R}^{d \times d}$
Define $k = \lceil d^{\eta} \rceil$, $\alpha = \varepsilon d^{-(2-\eta)/2}$, $\zeta = 4\varepsilon d$, and $\delta' = \frac{\delta}{w \cdot \lceil \log_2 \zeta / \alpha \rceil}$    ▷ Note: $\zeta > 2\alpha$
Draw $m'(k, \alpha, \delta')$ i.i.d. samples from $N(\mathbf{0}, \boldsymbol{\Sigma})$ and store it into a set $\mathcal{S}$    ▷ See Definition 10.13
Let Outcome be the output of the VectorizedApproxL1 algorithm given $\varepsilon$, $\delta$, $k$, $\alpha$, $\zeta$, and $\boldsymbol{S}$ as inputs
if Outcome is $\lambda \in \mathbb{R}$ and $\lambda < \varepsilon d$ then
    Draw $n \in \widetilde{\mathcal{O}}(\lambda^2 / \varepsilon^4)$ i.i.d. samples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \in \mathbb{R}^d$ from $N(\mathbf{0}, \boldsymbol{I}_d)$
    return $\widehat{\boldsymbol{\Sigma}} = \operatorname{argmin}_{\substack{\boldsymbol{A} \in \mathbb{R}^{d \times d} \text{ is p.s.d.} \\ \|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq \lambda \\ \lambda_{\min}(\boldsymbol{A}) \geq 1}} \sum_{i=1}^{n} \|\boldsymbol{A} - \boldsymbol{y}_i \boldsymbol{y}_i^\top\|_F^2$    ▷ See Eq. (10.6)
else
    Draw $2n \in \widetilde{\mathcal{O}}(d^2 / \varepsilon^2)$ i.i.d. samples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{2n} \in \mathbb{R}^d$ from $N(\mathbf{0}, \boldsymbol{I}_d)$
    return $\widehat{\boldsymbol{\Sigma}} = \frac{1}{2n} \sum_{i=1}^{2n} (\boldsymbol{y}_{2i} - \boldsymbol{y}_{2i-1})(\boldsymbol{y}_{2i} - \boldsymbol{y}_{2i-1})^\top$    ▷ Empirical covariance
```

Theorem 10.2. *For any given $\varepsilon, \delta \in (0,1)$, $\eta \in [0,1]$ and $\widetilde{\boldsymbol{\Sigma}} \in \mathbb{R}^{d \times d}$, TestAndOptimizeCovariance uses $n \in \widetilde{\mathcal{O}}\left(\frac{d^2}{\varepsilon^2} \cdot \left(d^{-\eta} + \min \left\{ 1, f(\boldsymbol{\Sigma}, \widetilde{\boldsymbol{\Sigma}}, d, \eta, \varepsilon) \right\} \right) \right)$, where*

$$f(\boldsymbol{\Sigma}, \widetilde{\boldsymbol{\Sigma}}, d, \eta, \varepsilon) = \frac{\|\mathrm{vec}(\widetilde{\boldsymbol{\Sigma}}^{-1/2} \boldsymbol{\Sigma} \widetilde{\boldsymbol{\Sigma}}^{-1/2} - \boldsymbol{I}_d)\|_1^2}{d^{2-\eta} \varepsilon^2},$$

i.i.d. samples from $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for some unknown mean $\boldsymbol{\mu}$ and unknown covariance $\boldsymbol{\Sigma}$, and can produce $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\Sigma}}$ in $\mathrm{poly}(n, d, \log(1/\varepsilon))$ time such that $\mathrm{d}_{\mathrm{TV}}(N(\boldsymbol{\mu}, \boldsymbol{\Sigma}), N(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}})) \leq \varepsilon$ with success probability at least $1 - \delta$.

Proof. Without loss of generality, we may assume that $\widetilde{\boldsymbol{\Sigma}} = \boldsymbol{I}_d$. This is because we can pre-process all samples by pre-multiplying $\widetilde{\boldsymbol{\Sigma}}^{-1/2}$ each of them to yield i.i.d. samples from $N(\boldsymbol{\mu}, \widetilde{\boldsymbol{\Sigma}}^{-1/2} \boldsymbol{\Sigma} \widetilde{\boldsymbol{\Sigma}}^{-1/2})$ and then post-process the estimated $\widehat{\boldsymbol{\Sigma}}$ by outputting $\widetilde{\boldsymbol{\Sigma}}^{1/2} \widehat{\boldsymbol{\Sigma}} \widetilde{\boldsymbol{\Sigma}}^{1/2}$ instead.

Correctness of $\widehat{\mathbf{\Sigma}}$ output. Consider the TestAndOptimizeCovariance algorithm given in Algorithm 24. Using the empirical mean $\widehat{\boldsymbol{\mu}} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{y}_i$ formed by $\mathcal{O}\left(\frac{d+\sqrt{d \log(1/\delta)}}{\varepsilon^2}\right) \subseteq \widetilde{\mathcal{O}}(d/\varepsilon^2)$ samples, Lemma 2.25 tells us that $(\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu})^\top \mathbf{\Sigma}^{-1} (\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}) \leq \varepsilon$ with probability at least $1 - \delta$. There are three possible outputs for $\widehat{\mathbf{\Sigma}}$:

1. $\widehat{\mathbf{\Sigma}} = \boldsymbol{I}_d$, which can only happen when `Outcome` is OK
2. $\widehat{\mathbf{\Sigma}} = \operatorname{argmin}_{\substack{\boldsymbol{A} \in \mathbb{R}^{d \times d} \text{ is p.s.d.} \\ \|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq r \\ \lambda_{\min}(\boldsymbol{A}) \geq 1}} \sum_{i=1}^{n} \|\boldsymbol{A} - \boldsymbol{y}_i \boldsymbol{y}_i^\top\|_F^2$, which can only happen when `Outcome` is $\lambda \in \mathbb{R}$
3. $\widehat{\mathbf{\Sigma}} = \frac{1}{2n} \sum_{i=1}^{2n} (\boldsymbol{y}_{2i} - \boldsymbol{y}_{2i-1})(\boldsymbol{y}_{2i} - \boldsymbol{y}_{2i-1})^\top$

Conditioned on VectorizedApproxL1 succeeding, with probability at least $1-\delta$, we will now show that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \mathbf{\Sigma}), N(\widehat{\boldsymbol{\mu}}, \widehat{\mathbf{\Sigma}})) \leq \varepsilon$ and failure probability at most 2δ in each of these cases, which implies the theorem statement as we can repeat the argument by scaling ε and δ by appropriate constants.

1: When `Outcome` is OK, Lemma 10.14 tells us that $\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \leq \alpha^2$, with failure probability at most δ. Meanwhile, Lemma 2.25 tells us that $(\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu})^\top \mathbf{\Sigma}^{-1} (\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}) \leq \varepsilon^2$, with failure probability at most δ. By union bound, both of these jointly hold with probability at least $1 - 2\delta$. Now, by setting $\widehat{\boldsymbol{\mu}} = \boldsymbol{I}_d$, we see that

$$\begin{aligned}
&\mathrm{d_{KL}}(N(\widehat{\boldsymbol{\mu}}, \widehat{\mathbf{\Sigma}}), N(\boldsymbol{\mu}, \mathbf{\Sigma})) \\
&= \frac{1}{2} \cdot \left(\mathrm{Tr}(\mathbf{\Sigma}^{-1} \widehat{\mathbf{\Sigma}}) - d + (\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}})^\top \mathbf{\Sigma}^{-1} (\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}}) + \ln \left(\frac{\det \mathbf{\Sigma}}{\det \widehat{\mathbf{\Sigma}}} \right) \right) \\
&\leq \frac{1}{2} \cdot \left((\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}})^\top \mathbf{\Sigma}^{-1} (\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}}) + \|\mathbf{\Sigma}^{-1/2} \widehat{\mathbf{\Sigma}} \mathbf{\Sigma}^{-1/2} - \boldsymbol{I}_d\|_F^2 \right) && \text{(By Lemma 2.29)} \\
&= \frac{1}{2} \cdot \left((\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}})^\top \mathbf{\Sigma}^{-1} (\boldsymbol{\mu} - \widehat{\boldsymbol{\mu}}) + \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \right) && \text{(Since } \widehat{\mathbf{\Sigma}} = \boldsymbol{I}_d\text{)} \\
&\leq \frac{1}{2} \cdot \left(\varepsilon^2 + \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \right) \\
& && \text{(Since } (\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu})^\top \mathbf{\Sigma}^{-1} (\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}) \leq \varepsilon \text{, with probability at least } 1 - \delta\text{)} \\
&\leq \frac{1}{2} \cdot \left(\varepsilon^2 + \alpha^2 \right) && \text{(Since } \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \leq \alpha^2 \text{, with probability at least } 1 - \delta\text{)} \\
&\leq \frac{1}{2} \cdot \left(\varepsilon^2 + \varepsilon^2 \right) && \text{(since } \alpha = \tfrac{\varepsilon k}{d} \leq \varepsilon \text{ as } k \leq d\text{)} \\
&= \varepsilon^2
\end{aligned}$$

Thus, by symmetry of TV distance and Theorem 2.18, we see that

$$\begin{aligned}
\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \mathbf{\Sigma}), N(\widehat{\boldsymbol{\mu}}, \widehat{\mathbf{\Sigma}})) &= \mathrm{d_{TV}}(N(\widehat{\boldsymbol{\mu}}, \widehat{\mathbf{\Sigma}}), N(\boldsymbol{\mu}, \mathbf{\Sigma})) \\
&\leq \sqrt{\frac{1}{2} \mathrm{d_{KL}}(N(\widehat{\boldsymbol{\mu}}, \widehat{\mathbf{\Sigma}}), N(\boldsymbol{\mu}, \mathbf{\Sigma}))} \leq \sqrt{\varepsilon^2} = \varepsilon
\end{aligned}$$

2: Using $r = \lambda$ as the upper bound, Lemma 10.15 tells us that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{\Sigma}), N(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}})) \leq \varepsilon$ with failure probability at most δ when $\widetilde{\mathcal{O}}(\frac{\lambda^2}{\varepsilon^4} + \frac{d}{\varepsilon^2})$ i.i.d. samples are used.
3: With $\widetilde{\mathcal{O}}(d^2/\varepsilon^2)$ samples, Lemma 2.25 tells us that $\mathrm{d_{TV}}(N(\boldsymbol{\mu}, \boldsymbol{\Sigma}), N(\widehat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}})) \leq \varepsilon$ with failure probability at most δ.

Sample complexity used. By Definition 10.13, VectorizedApproxL1 uses $|\boldsymbol{S}| = m'(k, \alpha, \delta') \in \widetilde{\mathcal{O}}(k/\alpha^2)$ samples to produce `Outcome`. Then, VectorizedApproxL1 further uses $\widetilde{\mathcal{O}}(\frac{\lambda^2}{\varepsilon^4} + \frac{d}{\varepsilon^2})$ samples or $\widetilde{\mathcal{O}}(d^2/\varepsilon^2)$ samples depending on whether $\lambda < \varepsilon d$. So, TestAndOptimizeCovariance has a total sample complexity of

$$\widetilde{\mathcal{O}}\left(\frac{k}{\alpha^2} + \min\left\{\frac{\lambda^2}{\varepsilon^4} + \frac{d}{\varepsilon^2}, \frac{d^2}{\varepsilon^2}\right\}\right) \subseteq \widetilde{\mathcal{O}}\left(\frac{k}{\alpha^2} + \frac{d}{\varepsilon^2} + \min\left\{\frac{\lambda^2}{\varepsilon^4}, \frac{d^2}{\varepsilon^2}\right\}\right) \tag{10.9}$$

Meanwhile, Lemma 10.14 states that

$$\|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1 \leq \lambda \leq 2\sqrt{k} \cdot \left(\frac{10d(d-1)\log d}{k(k-1)} \cdot \alpha + 2\|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1\right)$$

whenever `Outcome` is $\lambda \in \mathbb{R}$. Since $(a+b)^2 \leq 2a^2 + 2b^2$ for any two real numbers $a, b \in \mathbb{R}$, we see that

$$\begin{aligned}\frac{\lambda^2}{\varepsilon^4} &\in \mathcal{O}\left(\frac{k}{\varepsilon^4} \cdot \left(\frac{d^4\alpha^2}{k^4} + \|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1^2\right)\right) \\ &\subseteq \mathcal{O}\left(\frac{d^2}{\varepsilon^2} \cdot \left(\frac{d^2\alpha^2}{\varepsilon^2 k^3} + \frac{k \cdot \|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1^2}{d^2\varepsilon^2}\right)\right)\end{aligned} \tag{10.10}$$

Putting together Eq. (10.9) and Eq. (10.10), we see that the total sample complexity is

$$\widetilde{\mathcal{O}}\left(\frac{k}{\alpha^2} + \frac{d}{\varepsilon^2} + \frac{d^2}{\varepsilon^2} \cdot \min\left\{1, \frac{d^2\alpha^2}{\varepsilon^2 k^3} + \frac{k \cdot \|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1^2}{d^2\varepsilon^2}\right\}\right)$$

Recalling that $\boldsymbol{\Sigma}$ in the analysis above actually refers to the pre-processed $\widetilde{\boldsymbol{\Sigma}}^{-1/2}\boldsymbol{\Sigma}\widetilde{\boldsymbol{\Sigma}}^{-1/2}$, and that TestAndOptimizeCovariance sets $k = \lceil d^\eta \rceil$, $\alpha = \varepsilon d^{-(2-\eta)/2}$, with $0 \leq \eta \leq 1$, the above expression simplifies to

$$\widetilde{\mathcal{O}}\left(\frac{d^2}{\varepsilon^2} \cdot \left(d^{-\eta} + \min\left\{1, f(\boldsymbol{\Sigma}, \widetilde{\boldsymbol{\Sigma}}, d, \eta, \varepsilon)\right\}\right)\right)$$

where $f(\boldsymbol{\Sigma}, \widetilde{\boldsymbol{\Sigma}}, d, \eta, \varepsilon) = \frac{\|\mathrm{vec}(\widetilde{\boldsymbol{\Sigma}}^{-1/2}\boldsymbol{\Sigma}\widetilde{\boldsymbol{\Sigma}}^{-1/2} - \boldsymbol{I}_d)\|_1^2}{d^{2-\eta}\varepsilon^2}$. □

Remark on setting upper bound ζ. As ζ only affects the sample complexity logarithmically, one may be tempted to use a larger value than $\zeta = 4\varepsilon d$. However, observe that running VectorizedApproxL1 with a larger upper bound than $\zeta = 4\varepsilon\sqrt{d}$ would not be helpful since $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 > \zeta/2$ whenever VectorizedApproxL1 currently returns Fail and we have $\|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1 \leq \lambda$ whenever VectorizedApproxL1 returns $\lambda \in \mathbb{R}$. So, $\varepsilon d = \zeta/4 < \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 = \|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_2 \leq \|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1 \leq \lambda$ and TestAndOptimizeMean would have resorted to using the empirical mean anyway.

Chapter 11

Causal graph discovery with adaptive interventions and imperfect advice

> “Felix qui potuit rerum cognoscere causas.”
> *(“Happy is he who has been able to learn the causes of things.”)*
> - Virgil in *Georgics*

11.1 Introduction

In Chapter 6, we studied the problem of recovering the true underlying causal graph using adaptive interventions, providing a characterization of verification sets and search algorithms that use at most a logarithmic factor more interventions that worst case necessary. Typically though, in most applications of causal structure learning, there are domain experts and practitioners who can provide additional “advice” about the causal relations. Indeed, there has been a long line of work studying how to incorporate expert advice into the causal graph discovery process; e.g. see [Mee95, SSG$^+$98, dCJ11, FNB$^+$11, LB18, ASC20, FH20, POE21]. In this chapter, we study in a principled way how using purported expert advice can lead to improved algorithms for interventional design.

Before discussing our specific contributions, let us ground the above discussion with a concrete problem of practical importance. In modern virtualized infrastructure, it is increasingly common for applications to be modularized into a large number of interdependent microservices. These microservices communicate with each other in ways that depend on the application code and on the triggering userflow. Crucially, the communication graph between microservices is often unknown to the platform provider as the application code may be private and belong to different entities. However, knowing the graph is useful for various critical platform-level tasks, such as fault localization [ZPX$^+$19], active probing [TJG$^+$19], testing [JBT$^+$19], and taint analysis [CLO07]. Recently, [WRJ$^+$23] and [ICM$^+$22] suggested viewing the microservices communication graph as a sparse causal DAG. In particular, [WRJ+23] show that arbitrary interventions can be implemented as fault injections in a staging environment, so that a causal structure learning algorithm can be deployed to generate a sequence of interventions sufficient to learn the underlying communication graph. In such a setting, it is natural to assume that the platform provider already has an approximate guess about the graph, e.g. the graph discovered in a previous run of the algorithm or the graph suggested by public metadata tagging microservice code. The research program we put forth is to design causal structure learning algorithms that can take advantage of such potentially imperfect advice. Note however that the system in [WRJ+23] is not causally sufficient due to confounding user behavior and [ICM+22] does not actively perform interventions. So, the algorithm proposed in this work cannot be used directly for the microservices graph learning problem.

11.2 Our main results

Following the TestAndAct framework for designing learning-augmented algorithms, we consider the setting where the advice is a DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ purported to be the orientations of all the edges of the input essential graph $\mathcal{E}(\mathcal{G}^*)$.

While verification numbers of DAGs in the same Markov equivalence class may differ in general, we show that minimum and maximum atomic verification numbers of DAGs from the same Markov equivalence class cannot differ by a multiplicative factor of two.

Theorem 11.1. *We have* $\max_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G}) \leq 2 \cdot \min_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G})$. *Furthermore, there exist two DAGs* $\mathcal{G}_1$ *and* $\mathcal{G}_2$ *such that* $[\mathcal{G}_1] = [\mathcal{G}_2]$ *and* $\nu_1(\mathcal{G}_1) = 2 \cdot \nu_1(\mathcal{G}_2)$.

Using Theorem 11.1, we can test whether $\widetilde{\mathcal{G}} \overset{?}{=} \mathcal{G}^*$ while incurring at most twice the number of necessary interventions. In some sense, Theorem 11.1 enables us to "blindly trust" the information provided by imperfect advice to some extent. Meanwhile, we can define a distance measure ψ (see Definition 11.4) which is always bounded by n, the number of variables, and equals 0 when $\widetilde{\mathcal{G}} = \mathcal{G}^*$. Using ψ, we propose an adaptive algorithm TestAndSubsetSearch that can exploit a given advice DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$.

Theorem 11.2. *Fix an essential graph* $\mathcal{E}(\mathcal{G}^*)$ *of an unknown underlying DAG* $\mathcal{G}^*$. *Given an advice graph* $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$, *TestAndSubsetSearch runs in polynomial time and computes an atomic intervention set* $\mathcal{I} \subseteq 2^{\mathbf{V}}$ *in a deterministic and adaptive manner such that* $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*) = \mathcal{G}^*$ *and* $|\mathcal{I}| \in \mathcal{O}(\max\{1, \log \psi(\mathcal{G}^*, \widetilde{\mathcal{G}})\} \cdot \nu_1(\mathcal{G}^*))$.

Observe that when the advice is perfect (i.e. $\widetilde{\mathcal{G}} = \mathcal{G}^*$), we use $\mathcal{O}(\nu_1(\mathcal{G}^*))$ interventions, i.e. a constant multiplicative factor of the minimum number of interventions necessary. Meanwhile, even with low quality advice, we still use $\mathcal{O}(\log n \cdot \nu(\mathcal{G}^*))$ interventions, asymptotically matching the best known guarantees for adaptive search without advice in Chapter 6. To the best of our knowledge, Theorem 11.2 is the first known result that principally employs imperfect expert advice with provable guarantees in the context of causal graph discovery via interventions.

In Appendix C.3.1, we explain why TestAndSubsetSearch is simply the classic learning-augmented binary search given in Chapter 1 when the given essential graph $\mathcal{E}(\mathcal{G}^*)$ is an undirected path. So, another way to view our result is as a *generalization* that works on essential graphs of arbitrary moral DAGs.

For $k > 1$, Theorem 11.2 also extends to the k-bounded intervention setting where the algorithm uses $\mathcal{O}(\max\{1, \log \psi(\mathcal{G}^*, \widetilde{\mathcal{G}})\} \cdot \log k \cdot \nu_k(\mathcal{G}^*))$ interventions. This is achieved using techniques from Section 6.3.5. We omit this extension in this chapter and refer interested readers to [CGB23].

11.3 Technical overview

In this chapter, we use the notation of $\boldsymbol{C}(\mathcal{G}) \subseteq \boldsymbol{E}(\mathcal{G})$ to denote the set of covered edges of a DAG $\mathcal{G}$. That is, any atomic intervention set of $\mathcal{G}$ is a minimum vertex cover of $\boldsymbol{C}(\mathcal{G})$. Meanwhile, for any verifying set $\widetilde{\mathcal{V}} \subseteq 2^{\boldsymbol{V}}$, we also define $\widetilde{\boldsymbol{V}} = \{V \in \boldsymbol{V} : \exists \boldsymbol{I} \in \mathcal{V} \text{ such that } V \in \boldsymbol{I}\} \subseteq \boldsymbol{V}$ to refer to the set of nodes involved in the verifying set $\widetilde{\mathcal{V}}$.

11.3.1 Defining a suitable quality metric

To define the quality of the advice DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$, we first define the notion of min-hop-coverage which measures how "far" a given verifying set of $\widetilde{\mathcal{G}}$ is from the set of covered edges of $\mathcal{G}^*$. Recalling the definition of relevant nodes (Definition 6.23), we then define a quality measure $\psi(\mathcal{G}^*, \widetilde{\mathcal{G}})$ for DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ as an advice for DAG $\mathcal{G}^*$.

Definition 11.3 (Min-hop-coverage)**.** Fix a DAG $\mathcal{G}^*$ with MEC $[\mathcal{G}^*]$. For any DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ and any verifying set $\widetilde{\mathcal{V}} \subseteq 2^{\boldsymbol{V}}$ of $\widetilde{\mathcal{G}}$, we define the *min-hop-coverage* $r = h(\mathcal{G}^*, \widetilde{\mathcal{V}}) \in \{0, 1, 2, \ldots, n\}$ as the minimum number of hops such that *both* endpoints of covered edges $\boldsymbol{C}(\mathcal{G}^*)$ of $\mathcal{G}^*$ belong in the r-hop neighborhood $N^r_{\text{skel}(\mathcal{E}(\mathcal{G}^*))}(\widetilde{\boldsymbol{V}})$.

Definition 11.4 (Quality measure)**.** Fix a DAG $\mathcal{G}^*$ with MEC $[\mathcal{G}^*]$. For any DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$, we define $\psi(\mathcal{G}^*, \widetilde{\mathcal{G}})$ as follows:

$$\psi(\mathcal{G}^*, \widetilde{\mathcal{G}}) = \max_{\substack{\text{minimum sized atomic} \\ \text{verifying set } \widetilde{\mathcal{V}} \subseteq 2^{\boldsymbol{V}} \text{ of } \widetilde{\mathcal{G}}}} \left| \rho\left(\widetilde{\mathcal{V}}, N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}_{\text{skel}(\mathcal{E}(\mathcal{G}^*))}(\widetilde{\boldsymbol{V}})\right) \right|$$

In words, within the maximization term, the quality metric measures the number of nodes within $N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}_{\text{skel}(\mathcal{E}(\mathcal{G}^*))}(\widetilde{\boldsymbol{V}})$ that are adjacent to some unoriented arc within the node-induced subgraph $\mathcal{E}_{\widetilde{\mathcal{V}}}(\mathcal{G}^*)[N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}_{\text{skel}(\mathcal{E}(\mathcal{G}^*))}(\widetilde{\boldsymbol{V}})]$, i.e. intervene on $\widetilde{\mathcal{V}}$ then measure how many nodes within the subgraph is do *not* belong to singleton chain components.

By definition, $\psi(\mathcal{G}^*, \mathcal{G}^*) = 0$ and $\max_{\mathcal{G} \in [\mathcal{G}^*]} \psi(\mathcal{G}^*, \mathcal{G}) \leq n$. In words, $\psi(\mathcal{G}^*, \widetilde{\mathcal{G}})$ only counts the relevant nodes within the min-hop-coverage neighborhood after intervening on the *worst* possible verifying set of $\widetilde{\mathcal{G}}$. We define ψ in terms of the worst set because any search algorithm *cannot* evaluate $h(\mathcal{G}^*, \widetilde{\mathcal{V}})$, since $\mathcal{G}^*$ is unknown, and can only consider an *arbitrary* minimum sized atomic verifying set of $\widetilde{\mathcal{G}}$. The following example illustrates the concepts introduced above while showing that ψ is *not* symmetric in general.

Example 11.5. Consider the moral DAGs $\mathcal{G}^*$ and $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ on $n + 5$ nodes in Fig. 11.1, where dashed arcs represent the covered edges in each DAG. The covered edges of $\mathcal{G}^*$ are $A \to B$, $E \to D$, and $D \to C$. A minimum sized verifying set of $\widetilde{\mathcal{G}}$ is $\widetilde{\mathcal{V}} = \{\{A\}, \{E\}, \{Z_2\}\}$ with $\widetilde{\boldsymbol{V}} = \{A, E, Z_2\}$, given by the boxed nodes. As $N^1_{\text{skel}(\mathcal{E}(\mathcal{G}^*))}(\widetilde{\boldsymbol{V}}) = \{A, B, C, D, E, Z_1, Z_2, Z_3\}$ includes both endpoints of all covered edges of $\mathcal{G}^*$, we see that $h(\mathcal{G}^*, \widetilde{\mathcal{V}}) = 1$. Intervening on $\widetilde{\mathcal{V}}$ in $\mathcal{G}^*$ orients the arcs $B \to A \leftarrow C$, $C \leftarrow E \to D$, and $Z_3 \to Z_2 \to Z_1$ respectively which then triggers Meek R1 to orient $C \to B$ via $E \to C - B$ and to orient $Z_4 \to Z_3$ via $E \to C \to \ldots \to Z_4 - Z_3$ (after a few invocations of Meek R1), so $\{A, B, E, Z_1, Z_2, Z_3\}$ will *not* be relevant nodes in $\mathcal{E}_{\widetilde{\mathcal{V}}}(\mathcal{G}^*)[N^1_{\text{skel}(\mathcal{E}(\mathcal{G}^*))}(\widetilde{\boldsymbol{V}})]$. Meanwhile, the edge $C - D$ remains unoriented in $\mathcal{E}_{\widetilde{\mathcal{V}}}(\mathcal{G}^*)[N^1_{\text{skel}(\mathcal{E}(\mathcal{G}^*))}(\widetilde{\boldsymbol{V}})]$, so $\rho(\widetilde{\mathcal{V}}, N^1(\widetilde{\boldsymbol{V}})) = |\{C, D\}| = 2$. One can check that $\psi(\mathcal{G}^*, \widetilde{\mathcal{G}}) = 2$ while n could be arbitrarily large. On the other hand, observe that ψ is *not* symmetric: in the hypothetical situation where we use $\mathcal{G}^*$ as an advice for $\widetilde{\mathcal{G}}$, the min-hop-coverage has to extend along the chain $Z_1 - \ldots - Z_n$ to reach $\{Z_1, Z_2\}$, so $h(\mathcal{G}^*, V^*) \approx n$ and $\psi(\widetilde{\mathcal{G}}, \mathcal{G}^*) \approx n$ since the entire chain remains unoriented with respect to any minimum sized atomic verifying set of $\mathcal{G}^*$.

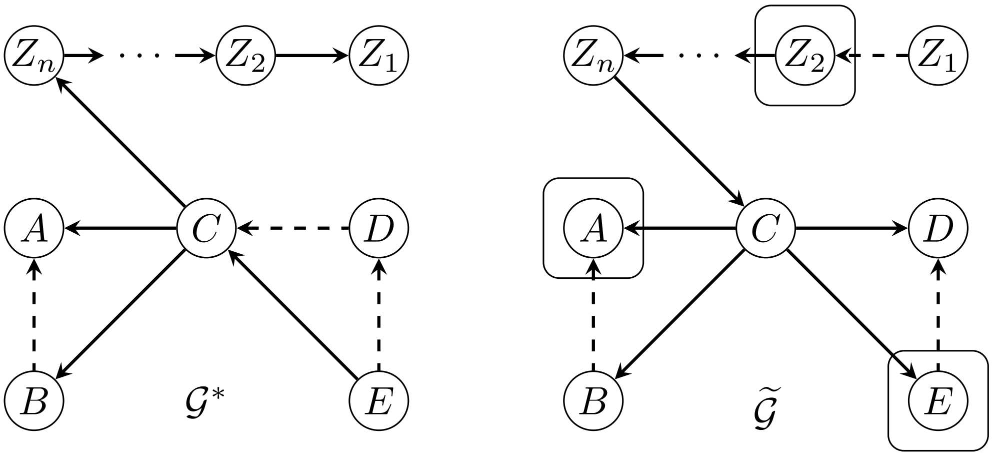


Figure 11.1: Two moral DAGs $\mathcal{G}^*$ and $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ on $n + 5$ nodes, where dashed arcs represent the covered edges in each DAG. The covered edges of $\mathcal{G}^*$ are $A \to B$, $E \to D$, and $D \to C$. A minimum sized verifying set of $\widetilde{\mathcal{G}}$ is $\widetilde{\mathcal{V}} = \{\{A\}, \{E\}, \{Z_2\}\}$ with $\widetilde{\boldsymbol{V}} = \{A, E, Z_2\}$, given by the boxed nodes.

Our main algorithmic result is that it is possible to design an algorithm that leverages an advice DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ and performs interventions to fully recover an unknown underlying DAG $\mathcal{G}^*$, whose performance depends on the advice quality $\psi(\mathcal{G}^*, \widetilde{\mathcal{G}})$. Our search algorithm only knows $\mathcal{E}(\mathcal{G}^*)$ and $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ but knows neither $\psi(\mathcal{G}^*, \widetilde{\mathcal{G}})$ nor $\nu_1(\mathcal{G}^*)$.

Remark 11.6 (Readability). In the rest of this chapter, we only consider neighborhoods of $\mathrm{skel}(\mathcal{E}(\mathcal{G}^*))$ and we always refer to the quality measure $\psi(\mathcal{G}^*, \widetilde{\mathcal{G}})$ with $\mathcal{G}^*$ and $\widetilde{\mathcal{G}}$ in the first and second parameters respectively. Going forward, we will drop the subscript $\mathrm{skel}(\mathcal{E}(\mathcal{G}^*))$ when referring to r-hop neighbors $N^r_{\mathrm{skel}(\mathcal{E}(\mathcal{G}^*))}(\cdot)$, and write ψ to mean $\psi(\mathcal{G}^*, \widetilde{\mathcal{G}})$.

11.3.2 Ratio of verification numbers

Our strategy for proving Theorem 11.1 is to consider two arbitrary DAGs $\mathcal{G}_s$ (source) and $\mathcal{G}_t$ (target) in the same equivalence class and transform a verifying set for $\mathcal{G}_s$ into a verifying set for $\mathcal{G}_t$ using Lemma 2.49 due to [Chi95]; we present the explicit algorithm in Algorithm 25. The correctness of Algorithm 25 is given in [Chi95] where the key idea is to show that $X \to Y$ in Line 9 is a covered edge; see [Chi95, Lemma 2].

Algorithm 25 Transform between DAGs within the same MEC via covered edge reversals

```
1: Input: Two DAGs $\mathcal{G}_s = (\boldsymbol{V}, \boldsymbol{E}_s)$ and $\mathcal{G}_t = (\boldsymbol{V}, \boldsymbol{E}_t)$ from the same MEC
2: Output: A sequence seq of covered edge reversals that transforms $\mathcal{G}_s$ to $\mathcal{G}_t$
3: seq $\leftarrow \emptyset$
4: while $\mathcal{G}_s \neq \mathcal{G}_t$ do
5:     Fix an arbitrary valid ordering $\pi$ for $\mathcal{G}_s$
6:     Let $\boldsymbol{A} \leftarrow \boldsymbol{A}(\mathcal{G}_s) \setminus \boldsymbol{A}(\mathcal{G}_t)$ be the set of differing arcs
7:     Let $Y = \underset{\substack{Z \in \boldsymbol{V} \,:\, \exists\, U \in \boldsymbol{V} \\ \text{such that } U \to Z \,\in\, \boldsymbol{A}}}{\mathrm{argmin}} \{\pi(Z)\}$
8:     Let $X = \underset{Z \in \boldsymbol{V} \,:\, Z \to Y \,\in\, \boldsymbol{A}}{\mathrm{argmax}} \{\pi(Z)\}$
9:     Add $X \to Y$ to seq                      ▷ [Chi95, Lemma 2]: $X \to Y \in \boldsymbol{C}(\mathcal{G}_s)$
10:    Update $\mathcal{G}_s$ by replacing $X \to Y$ with $Y \to X$
11: return seq
```

Instead of proving Theorem 11.1 by analyzing the exact sequence `seq` of covered edges produced by Algorithm 25 when transforming between the $\mathcal{G}_{\min} = \mathrm{argmin}_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G})$ and $\mathcal{G}_{\max} = \mathrm{argmax}_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G})$, we will prove something more general.

Observe that taking both endpoints of any maximal matching of covered edges is a valid verifying set that is at most *twice* the size of the minimum verifying set. This is because maximal matching is a 2-approximation to the minimum vertex cover. Motivated by this observation, our proof for Theorem 11.1 uses the following transformation argument (see Lemma 11.7 below): for two DAGs $\mathcal{G}$ and $\mathcal{G}'$ that differ only on the arc direction of a single covered edge $X - Y$, we show that given a special type of maximal matching called CRG maximal matching (which we formally define later in Definition 11.10) on the covered edges of $\mathcal{G}$, we can obtain another CRG maximal matching *of the same size* on the covered edges of $\mathcal{G}'$, after reversing $X - Y$ and transforming $\mathcal{G}$ to $\mathcal{G}'$. So, starting from $\mathcal{G}_s$, we compute a CRG maximal matching, then we apply the transformation argument above on the sequence of covered edges given by Algorithm 25 until we get a CRG maximal matching of $\mathcal{G}_t$ *of the same size*. Thus, we can conclude that the minimum vertex cover sizes of $\mathcal{G}_s$ and $\mathcal{G}_t$ differ by a factor of at most two. This argument holds for *any* pair of DAGs $(\mathcal{G}_s, \mathcal{G}_t)$ from the same MEC which implies Theorem 11.1.

Lemma 11.7 (Informal)**.** *For any two moral DAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ from the same MEC differing only on the direction of a covered edge $X - Y$, i.e. $X \to Y \in \boldsymbol{E}(\mathcal{G}_1)$ and $Y \to X \in \boldsymbol{E}(\mathcal{G}_2)$, there exists an explicit modification to transform a CRG maximal matching of $\mathcal{G}_1$ to a CRG maximal matching of $\mathcal{G}_2$ such that both maximal matchings have the same size.*

11.3.3 TestAndSubsetSearch

Our adaptive search algorithm TestAndSubsetSearch uses SubsetSearch (see Algorithm 14 from Chapter 6) as a subroutine. We begin by observing that running SubsetSearch on any subset $\boldsymbol{A} \subseteq \boldsymbol{V}$ fully orients $\mathcal{E}(\mathcal{G}^*)$ into $\mathcal{G}^*$ whenever the covered edges of $\mathcal{G}^*$ lies in the node-induced subgraph $\mathcal{G}^*[\boldsymbol{A}]$.

Lemma 11.8. *Fix a DAG $\mathcal{G}^* = (\boldsymbol{V}, \boldsymbol{E})$. Let $\boldsymbol{V}' \subseteq \boldsymbol{V}$ be any subset of nodes and $\mathcal{I}_{\boldsymbol{V}'} \subseteq \boldsymbol{V}$ be the intervention set intervened upon by SubsetSearch when run on subset $\boldsymbol{V}'$. If $\boldsymbol{C}(\mathcal{G}^*) \subseteq \boldsymbol{E}(\mathcal{G}^*[\boldsymbol{V}'])$, then $\mathcal{E}_{\mathcal{I}_{\boldsymbol{V}'}}(\mathcal{G}^*) = \mathcal{G}^*$.*

Motivated by Lemma 11.8, we design TestAndSubsetSearch to repeatedly invoke SubsetSearch on node-induced subgraphs $N^r(\widetilde{\boldsymbol{V}})$, starting from an *arbitrary* minimum sized atomic verifying set $\widetilde{\mathcal{V}} \subseteq 2^{\boldsymbol{V}}$ of $\widetilde{\mathcal{G}}$ and for *increasing* values of r. While the high-level subroutine invocation idea seems simple, one needs to invoke SubsetSearch at *suitably chosen intervals* in order to achieve our theoretical guarantees of Theorem 11.2. Below, we explain how to do so in three successive attempts while explaining the algorithmic decisions behind each modification introduced. As a reminder, we *do not* know $\mathcal{G}^*$ and thus *do not* know $h(\mathcal{G}^*, \widetilde{\mathcal{V}})$ for any minimum sized atomic verifying set $\widetilde{\mathcal{V}}$ of $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$.

For $i \in \mathbb{N}$, let us denote n_i as the number of relevant nodes and $r(i) \in \mathbb{N}$ as the value of r in the i-th invocation of SubsetSearch, where we insist that $r(0) = 0$ and $r(j) > r(j-1)$ for any $j \in \mathbb{N}$. Note that $n_i \leq |\rho(\widetilde{\mathcal{V}}, N^{r(i)}(\widetilde{\boldsymbol{V}}))|$ since the number of relevant nodes in the i-th invocation is at most the number of relevant nodes in the neighborhood, but it may have decreased due to interventions from earlier invocations. When $r = 0$, we are simply intervening on the verifying set $\widetilde{\mathcal{V}}$, which only incurs $\mathcal{O}(\nu_1(\mathcal{G}^*))$ interventions due to Theorem 11.1. Meanwhile, we can appeal to Lemma 11.8 to conclude that $\mathcal{E}(\mathcal{G}^*)$ is completely oriented into $\mathcal{G}^*$ in the t-th invocation if $r(t) \geq h(\mathcal{G}^*, \widetilde{\mathcal{V}})$.

Naive attempt: Invoke for $r = 0, 1, 2, 3, \ldots$

The most straightforward attempt would be to invoke SubsetSearch repeatedly each time we increase r by 1 until the graph is fully oriented – in the worst case, $t = h(\mathcal{G}^*, \widetilde{\mathcal{V}})$. However, this may cause us to incur way too many interventions. Using Theorem 6.24, one can only argue that the overall number interventions incurred is $\mathcal{O}(\sum_{i=0}^{t} \log n_i \cdot \nu_1(\mathcal{G}^*))$. However, $\sum_i \log n_i$ could be significantly larger than $\log(\sum_i n_i)$ in general, e.g. $\log 2 + \ldots + \log 2 = (n/2) \cdot \log 2 \gg \log n$. In fact, if $\mathcal{G}^*$ was a path on n nodes $v_1 \to v_2 \to \ldots \to v_n$ and $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ misleads us with $v_1 \leftarrow v_2 \leftarrow \ldots \leftarrow v_n$, then this approach incurs $\Omega(n)$ interventions in total.

Tweak 1: Only invoke periodically

Since Theorem 6.24 provides us a logarithmic factor in the analysis, we could instead consider only invoking SubsetSearch after the number of nodes in the subgraph *increases by a polynomial factor*. For example, if we invoked SubsetSearch with n_i previously, then we will wait until the number of relevant nodes surpasses n_i^2 before invoking SubsetSearch again, where we define $n_0 \geq 2$ for simplicity. Since $\log n_i \geq 2 \log n_{i-1}$, we can see via an inductive argument that the number of interventions used in the final invocation will dominate the total number of interventions used so far: $n_t \geq 2 \log n_{t-1} \geq \log n_{t-1} + 2 \log n_{t-2} \geq \ldots \geq \sum_{i=0}^{t-1} \log n_i$. Since $n_i \leq n$ for any i, we can already prove that $\mathcal{O}(\log n \cdot \nu_1(\mathcal{G}^*))$ interventions suffice, matching the advice-free bound of Theorem 6.13. However, this approach does *not* take into account the quality of $\widetilde{\mathcal{G}}$ and is *insufficient* to relate n_t with the advice measure ψ.

Tweak 2: Also invoke one round before

Suppose the final invocation of SubsetSearch is on $r(t)$-hop neighborhood while incurring $\mathcal{O}(\log n_t \cdot \nu_1(\mathcal{G}^*))$ interventions. This means that $\boldsymbol{C}(\mathcal{G}^*)$ lies within $N^{r(t)}(\widetilde{\boldsymbol{V}})$ but *not* within $N^{r(t-1)}(\widetilde{\boldsymbol{V}})$. That is, $N^{r(t-1)}(\widetilde{\boldsymbol{V}}) \subsetneq N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}}) \subseteq N^{r(t)}(\widetilde{\boldsymbol{V}})$. While this tells us that $n_{t-1} \leq |\rho(\widetilde{\mathcal{V}}, N^{r(t-1)}(\widetilde{\boldsymbol{V}}))| < |\rho(\widetilde{\mathcal{V}}, N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}}))| \leq \psi$, what we want is to conclude that $n_t \in \mathcal{O}(\psi)$. Unfortunately, even when ψ can be attained with a neighbood radius of $r(t-1)+1$, it could be the case that $\psi \ll |N^{r(t)}(\widetilde{\boldsymbol{V}})|$ as the number of relevant nodes could blow up within a single hop, but before the next invocation occurs (see Fig. 11.2). To control this potential blow up, we use the following technical fix: whenever we want to invoke SubsetSearch on neighbood radius of $r(i)$, first invoke SubsetSearch on neighbood radius of $r(i) - 1$ and terminate earlier if the graph is already fully oriented into $\mathcal{G}^*$. Doing so enables a case analysis to show that at most $\mathcal{O}\left(\max\{1, \log \psi\} \cdot \nu_1(\mathcal{G}^*)\right)$ interventions are always performed.

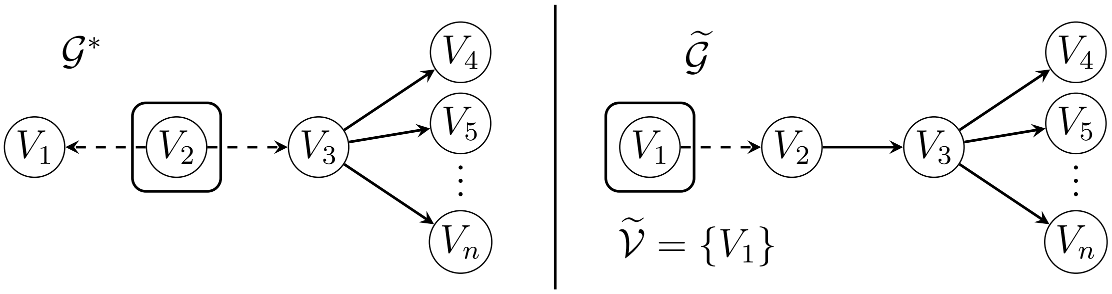

$\mathcal{G}^*$
V_1
V_2
V_3
V_4
V_5
V_n
$\widetilde{\mathcal{G}}$
$\widetilde{\mathcal{V}} = \{V_1\}$

Figure 11.2: Consider the ground truth DAG $\mathcal{G}^*$ with unique minimum verifying set $\{V_2\}$ and an advice DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ with chosen minimum verifying set $\widetilde{\mathcal{V}} = \{V_1\}$. So, $h(\mathcal{G}^*, \widetilde{\mathcal{V}}) = 1$ and ideally we want to argue that our algorithm uses a constant number of interventions. Without tweak 2 and starting with $n_0 = 2$, an algorithm that increases hop radius until the number of relevant nodes is squared will *not* invoke SubsetSearch until $r = 3$ because $\rho(\widetilde{\mathcal{V}}, N^1(\{V_1\})) = 1 < n_0^2$ and $\rho(\widetilde{\mathcal{V}}, N^2(\{V_1\})) = 2 < n_0^2$. However, $\rho(\widetilde{\mathcal{V}}, N^3(\{V_1\})) = n - 1$ and we can only conclude that the algorithm uses $\mathcal{O}(\log n)$ interventions by invoking SubsetSearch on a subgraph on $n-1$ nodes.

11.4 Ratio of verification numbers

In this section, we set out to prove Theorem 11.1 by repeated applications of the transformation argument of Lemma 11.11; the formal version of Lemma 11.7. To do so, we seek to first understand how the status of covered edges evolve when we perform a single edge reversal in Lemma 11.9 and properly defining CRG maximal matchings in Definition 11.10. The proofs of Lemma 11.9 and Lemma 11.11 are given in Appendix C.3.2.

Lemma 11.9 (Covered edge status changes due to covered edge reversal)**.** *Let $\mathcal{G}^*$ be a moral DAG with MEC $[\mathcal{G}^*]$ and consider any DAG $\mathcal{G} \in [\mathcal{G}^*]$. Suppose $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ has a covered edge $X \to Y \in \boldsymbol{C}(\mathcal{G})$ and we reverse $X \to Y$ to $Y \to X$ to obtain a new DAG $\mathcal{G}' \in [\mathcal{G}^*]$. Then, all of the following statements hold:*

1. *$Y \to X \in \boldsymbol{C}(\mathcal{G}')$. Note that this is the covered edge that was reversed.*

2. *If an edge E does not involve X or Y, then $E \in \boldsymbol{C}(\mathcal{G})$ if and only if $E \in \boldsymbol{C}(\mathcal{G}')$.*

3. *If $X \in \mathrm{Ch}_{\mathcal{G}}(A)$ for some $A \in \boldsymbol{V} \setminus \{X, Y\}$, then $A \to X \in \boldsymbol{C}(\mathcal{G})$ if and only if $A \to Y \in \boldsymbol{C}(\mathcal{G}')$.*

4. *If $B \in \mathrm{Ch}_{\mathcal{G}}(Y)$ and $X \to B \in \boldsymbol{E}(\mathcal{G})$ for some $B \in \boldsymbol{V} \setminus \{X, Y\}$, then $Y \to B \in \boldsymbol{C}(\mathcal{G})$ if and only if $X \to B \in \boldsymbol{C}(\mathcal{G}')$.*

We now define what is a conditional-root-greedy (CRG) maximal matching. As the set of covered edges $\boldsymbol{C}(\mathcal{G})$ of any DAG $\mathcal{G}$ induces a forest (see Lemma 6.15), we define the CRG maximal matching using a particular greedy process on the tree structure of $\boldsymbol{C}(\mathcal{G})$.

Definition 11.10 (Conditional-root-greedy (CRG) maximal matching)**.** Given a DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ with a valid ordering $\pi_{\mathcal{G}}$ and a subset of edges $\boldsymbol{S} \subseteq \boldsymbol{E}$, we define the conditional-root-greedy (CRG) maximal matching $\boldsymbol{M}_{\mathcal{G},\pi_{\mathcal{G}},\boldsymbol{S}}$ as the *unique* maximal matching on $\boldsymbol{C}(\mathcal{G})$ computed via Algorithm 26: greedily choose arcs $X \to Y$ where the X has no incoming arcs by minimizing $\pi_{\mathcal{G}}(Y)$, conditioned on *favoring arcs outside of* $\boldsymbol{S}$.

Algorithm 26 Conditional-root-greedy (CRG) maximal matching

```
Input: A DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, a valid ordering $\pi_{\mathcal{G}}$ of $\mathcal{G}$, a subset of edges $\boldsymbol{S} \subseteq \boldsymbol{E}$
Output: A CRG maximal matching $\boldsymbol{M}_{\mathcal{G},\pi_{\mathcal{G}},\boldsymbol{S}}$
Initialize $\boldsymbol{M}_{\mathcal{G},\pi_{\mathcal{G}},\boldsymbol{S}} \leftarrow \emptyset$ and $\boldsymbol{C} \leftarrow \boldsymbol{C}(\mathcal{G})$
while $\boldsymbol{C} \neq \emptyset$ do
    Let $X = \operatorname{argmin}_{Z \in \boldsymbol{V} : Z \to V \in \boldsymbol{C}} \{\pi_{\mathcal{G}}(Z)\}$    ▷ $X$ is a root with no incoming arcs
    Let $Y = \operatorname{argmin}_{Z \in \boldsymbol{V} : X \to Z \in \boldsymbol{C}} \{\pi_{\mathcal{G}}(Z) + n^2 \cdot \mathbb{1}_{X \to Z \in \boldsymbol{S}}\}$
    Add the arc $X \to Y$ to $\boldsymbol{M}_{\mathcal{G},\pi_{\mathcal{G}},\boldsymbol{S}}$
    Remove all arcs with $X$ or $Y$ as endpoints from $\boldsymbol{C}$
return $\boldsymbol{M}_{\mathcal{G},\pi_{\mathcal{G}},\boldsymbol{S}}$
```

The CRG maximal matching is unique with respect to a fixed valid ordering π of $\mathcal{G}$ and subset $\boldsymbol{S}$. We will later consider CRG maximal matchings with $\boldsymbol{S} = \boldsymbol{A}(\mathcal{G}_s) \cap \boldsymbol{A}(\mathcal{G}_t)$, where the arc set $\boldsymbol{S}$ *remains unchanged throughout the entire transformation process*.

Lemma 11.11 (Formal version of Lemma 11.7)**.** *Consider two moral DAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ from the same MEC such that they differ only in one covered edge direction: $X \to Y \in \boldsymbol{E}(\mathcal{G}_1)$ and $Y \to X \in \boldsymbol{E}(\mathcal{G}_2)$. Let $\boldsymbol{S} \subseteq \boldsymbol{E}$ be a subset such that $X \to Y, Y \to X \not\in \boldsymbol{S}$. If X has a direct parent $A \in \boldsymbol{V}$ in $\mathcal{G}_1$, we further require $A \to X \in \boldsymbol{S}$. When $\pi_{\mathcal{G}_1}$ is an ordering for $\mathcal{G}_1$ such that $Y = \operatorname{argmin}_{Z \in \boldsymbol{V} : X \to Z \in \boldsymbol{C}(\mathcal{G}_1)} \{\pi_{\mathcal{G}_1}(Z) + n^2 \cdot \mathbb{1}_{X \to Z \in \boldsymbol{S}}\}$ with CRG maximal matching $\boldsymbol{M}_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$, one can transform $\pi_{\mathcal{G}_1}$ to $\pi_{\mathcal{G}_2}$ and $\boldsymbol{M}_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$ to another CRG maximal matching $\boldsymbol{M}_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ for $\boldsymbol{C}(\mathcal{G}_2)$ such that $|\boldsymbol{M}_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}| = |\boldsymbol{M}_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}|$.*

To be precise, given $\pi_{\mathcal{G}_1}$, we will define $\pi_{\mathcal{G}_2}$ in Lemma 11.11 as follows:

$$\pi_{\mathcal{G}_2}(V) = \begin{cases} \pi_{\mathcal{G}_1}(X) & \text{if } V = Y \\ \pi_{\mathcal{G}_1}(U) & \text{if } V = X \\ \pi_{\mathcal{G}_1}(Y) & \text{if } V = U \\ \pi_{\mathcal{G}_1}(V) & \text{else} \end{cases} \tag{11.1}$$

For illustrated examples of conditional-root-greedy (CRG) maximal matchings and how we update the permutation ordering, see Fig. 11.3 and Fig. 11.4. The former gives an example where X has directed parent A while X has no directed parents in the latter.

As discussed in Section 11.3, the proof of Theorem 11.1 follows by picking $\mathcal{G}_s = \operatorname{argmax}_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G})$ and $\mathcal{G}_t = \operatorname{argmin}_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G})$, applying Algorithm 25 to find a trans-

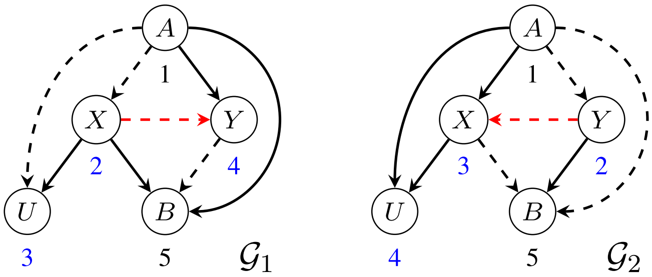


Figure 11.3: DAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ agree on all arc directions except for $X \to Y$ in $\mathcal{G}_1$ and $Y \to X$ in $\mathcal{G}_2$. Dashed arcs represent the covered edges in each DAG. The numbers below each vertex indicate the $\pi_{\mathcal{G}_1}$ and $\pi_{\mathcal{G}_2}$ orderings respectively. In $\mathcal{G}_1$, $U = \text{argmin}_{Z \in \text{Ch}_{\mathcal{G}_1}(X)} \{\pi_{\mathcal{G}_1}(Z)\}$. Observe that Eq. (11.1) modifies the ordering only for $\{X, Y, U\}$ (in blue) while keeping the ordering of all other nodes fixed. Suppose $\boldsymbol{S} = \boldsymbol{A}(\mathcal{G}_1) \cap \boldsymbol{A}(\mathcal{G}_2) = \{A \to B, A \to X, A \to Y, A \to U, X \to B, X \to U, Y \to B\}$. With respect to $\pi_{\mathcal{G}_1}$ and $\boldsymbol{S}$, The conditional-root-greedy maximal matchings (see Algorithm 26) are $\boldsymbol{M}_{\mathcal{G}_1, \pi_{\mathcal{G}_1}, \boldsymbol{S}} = \{A \to X, Y \to B\}$ and $\boldsymbol{M}_{\mathcal{G}_2, \pi_{\mathcal{G}_2}, \boldsymbol{S}} = \{A \to Y, X \to B\}$.

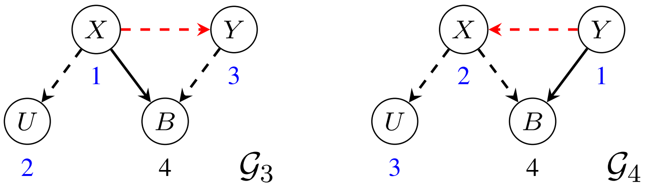


Figure 11.4: DAGs $\mathcal{G}_3$ and $\mathcal{G}_4$ agree on all arc directions except for $X \to Y$ in $\mathcal{G}_3$ and $Y \to X$ in $\mathcal{G}_4$. Dashed arcs represent the covered edges in each DAG. The numbers below each vertex indicate the $\pi_{\mathcal{G}_3}$ and $\pi_{\mathcal{G}_4}$ orderings respectively. Observe that $\boldsymbol{C}(\mathcal{G}_3) = \{X \to U, X \to Y, Y \to B\}$. If we define $\boldsymbol{S} = \boldsymbol{A}(\mathcal{G}_3) \cap \boldsymbol{A}(\mathcal{G}_4) = \{X \to B, X \to U, Y \to B\}$, we see that the conditional-root-greedy maximal matchings (see Algorithm 26) are $\boldsymbol{M}_{\mathcal{G}_3, \pi_{\mathcal{G}_3}, \boldsymbol{S}} = \{X \to Y\}$ and $\boldsymbol{M}_{\mathcal{G}_4, \pi_{\mathcal{G}_4}, \boldsymbol{S}} = \{Y \to X\}$. Note that Algorithm 26 does *not* choose $X \to U \in \boldsymbol{C}(\mathcal{G}_1)$ despite $\pi(U) < \pi(Y)$ because $X \to U \in \boldsymbol{S}$, so $\pi(Y) < \pi(U) + n^2$.

formation sequence of covered edge reversals between them, and repeatedly applying Lemma 11.11 with the conditioning set $\boldsymbol{S} = \boldsymbol{A}(\mathcal{G}_s) \cap \boldsymbol{A}(\mathcal{G}_t)$ to conclude that $\mathcal{G}_s$ and $\mathcal{G}_t$ have the same sized CRG maximal matchings, and thus implying that $\min_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G}) = \nu_1(\mathcal{G}_s) \leq 2 \cdot \nu_1(\mathcal{G}_t) = 2 \cdot \text{argmax}_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G})$. Note that we keep the conditioning set $\boldsymbol{S}$ *unchanged throughout the entire transformation process* from $\mathcal{G}_s$ to $\mathcal{G}_t$.

Theorem 11.1. *We have* $\max_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G}) \leq 2 \cdot \min_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G})$. *Furthermore, there exist two DAGs* $\mathcal{G}_1$ *and* $\mathcal{G}_2$ *such that* $[\mathcal{G}_1] = [\mathcal{G}_2]$ *and* $\nu_1(\mathcal{G}_1) = 2 \cdot \nu_1(\mathcal{G}_2)$.

Proof. Consider any two DAGs $\mathcal{G}_s, \mathcal{G}_t \in [\mathcal{G}^*]$. To transform $\mathcal{G}_s = (\boldsymbol{V}, \boldsymbol{E}_s)$ to $\mathcal{G}_t = (\boldsymbol{V}, \boldsymbol{E}_t)$, Algorithm 25 flips covered edges one by one such that $|\boldsymbol{E}_s \setminus \boldsymbol{E}_t|$ decreases in a monotonic manner. We will repeatedly apply Lemma 11.11 with $\boldsymbol{S} = \boldsymbol{A}(\mathcal{G}_s) \cap \boldsymbol{A}(\mathcal{G}_t)$ on the sequence of covered edge reversals produced by Algorithm 25.

Let $\text{Pa}_{\mathcal{G}_s}$ be an arbitrary ordering for $\mathcal{G}_s$ and we compute an initial conditional-root-greedy maximal matching for $\boldsymbol{C}(\mathcal{G}_s)$ with respect to some ordering $\text{Pa}_{\mathcal{G}_s}$ and conditioning set $\boldsymbol{S}$. To see why Lemma 11.11 applies at each step for reversing a covered edge from $X \to Y$ to $Y \to X$, we need to ensure the following:

1. If X has a parent vertex A (i.e. $X \in \mathrm{Ch}_{\mathcal{G}_1}(A)$), then $A \to X \in \boldsymbol{S}$.

   If $A \to X \not\in \boldsymbol{S}$, then then $A \to X$ is a covered edge that should be flipped to transform from $\mathcal{G}_s$ to $\mathcal{G}_t$. However, this means that Algorithm 25 would pick $A \to X$ to reverse instead of picking $X \to Y$ to reverse. Contradiction.

2. $X \to Y, Y \to X \not\in \boldsymbol{S}$.

   This is satisfied by the definition of $\boldsymbol{S} = \boldsymbol{E}_s \cap \boldsymbol{E}_t$ since reversing $X \to Y$ to $Y \to X$ implies that neither of them are in $\boldsymbol{S}$.

3. $Y = \mathrm{argmin}_{Z \,:\, X \to Z \in \boldsymbol{C}(\mathcal{G}_1)} \{\mathrm{Pa}_{\mathcal{G}_1}(Z) + n^2 \cdot \mathbb{1}_{X \to Z \in \boldsymbol{S}}\}$.

   Since $X \to Y \not\in \boldsymbol{S}$, this holds when $Y = \mathrm{argmin}_{Z \,:\, X \to Z \in \boldsymbol{C}(\mathcal{G}_1)} \{\mathrm{Pa}_{\mathcal{G}_1}(Z)\}$. This is satisfied by line 7 of Algorithm 25.

4. $\boldsymbol{M}_{G_1, \mathrm{Pa}_{\mathcal{G}_1}, \boldsymbol{S}}$ is a conditional-root-greedy maximal matching for $\boldsymbol{C}(\mathcal{G}_1)$ with respect to some ordering $\mathrm{Pa}_{\mathcal{G}_1}$ and conditioning set $\boldsymbol{S}$.

   This is satisfied since we always maintain a conditional-root-greedy maximal matching and $\boldsymbol{S}$ is unchanged throughout.

By applying Lemma 11.7 with $\boldsymbol{S} = \boldsymbol{A}(\mathcal{G}_s) \cap \boldsymbol{A}(\mathcal{G}_t)$ repeatedly on the sequence of covered edge reversals produced by Algorithm 25, we see that there exists a conditional-root-greedy maximal matching $\boldsymbol{M}_{\mathcal{G}_s, \pi_{\mathcal{G}_s}}$ for $\boldsymbol{C}(\mathcal{G}_s)$ and a conditional-root-greedy maximal matching $\boldsymbol{M}_{\mathcal{G}_t, \pi_{\mathcal{G}_t}}$ for $\boldsymbol{C}(\mathcal{G}_t)$ such that $|\boldsymbol{M}_{\mathcal{G}_s, \pi_{\mathcal{G}_s}}| = |\boldsymbol{M}_{\mathcal{G}_t, \pi_{\mathcal{G}_t}}|$.

The claim follows since maximal matching is a 2-approximation to minimum vertex cover, and the verification number $\nu(\mathcal{G})$ of any DAG $\mathcal{G}$ is the size of the minimum vertex cover of its covered edges $\boldsymbol{C}(\mathcal{G})$.

To see that the ratio of 2 is tight, refer to Fig. 11.5 for two explicit DAGs. □

11.5 TestAndSubsetSearch

Lemma 11.8. *Fix a DAG $\mathcal{G}^* = (\boldsymbol{V}, \boldsymbol{E})$. Let $\boldsymbol{V}' \subseteq \boldsymbol{V}$ be any subset of nodes and $\mathcal{I}_{\boldsymbol{V}'} \subseteq \boldsymbol{V}$ be the intervention set intervened upon by SubsetSearch when run on subset $\boldsymbol{V}'$. If $\boldsymbol{C}(\mathcal{G}^*) \subseteq \boldsymbol{E}(\mathcal{G}^*[\boldsymbol{V}'])$, then $\mathcal{E}_{\mathcal{I}_{\boldsymbol{V}'}}(\mathcal{G}^*) = \mathcal{G}^*$.*

Proof. By Theorem 6.24, SubsetSearch fully orients edges within the node-induced subgraph induced by $\boldsymbol{V}'$, i.e. SubsetSearch will perform atomic interventions $\mathcal{I}_{\boldsymbol{V}'} \subseteq 2^{\boldsymbol{V}}$ resulting in $\mathcal{E}_{\mathcal{I}_{\boldsymbol{V}'}}(\mathcal{G}^*)[\boldsymbol{V}'] = \mathcal{G}^*[\boldsymbol{V}']$. As $\boldsymbol{C}(\mathcal{G}^*) \subseteq \boldsymbol{E}(\mathcal{G}^*[\boldsymbol{V}'])$, all covered edges $\boldsymbol{C}(\mathcal{G}^*)$ will be oriented. Then, since Theorem 6.7 tells us that any intervention set that fully orients $\boldsymbol{C}(\mathcal{G}^*)$ is a verifying set for $\mathcal{G}^*$, we see that $\mathcal{E}_{\mathcal{I}_{\boldsymbol{V}'}}(\mathcal{G}^*) = \mathcal{G}^*$. □

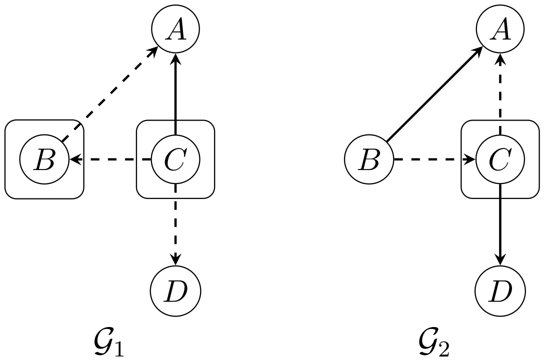


Figure 11.5: The ratio of 2 in Theorem 11.1 is tight: $\mathcal{G}_1$ and $\mathcal{G}_2$ belong in the same MEC with $\nu(\mathcal{G}_1) = 2$ and $\nu(\mathcal{G}_2) = 1$. The dashed arcs represent the covered edges and the boxed nodes represent a minimum vertex cover of the covered edges.

TestAndSubsetSearch is presented in Algorithm 27. The first tweak mentioned in Section 11.3.3 is captured in the inequality $\rho(\mathcal{I}_i, N^r(\widetilde{\boldsymbol{V}})) \geq n_i^2$ while the second tweak correspond to the terms $\mathcal{C}_i$ and $\mathcal{C}_i'$. As a side note, observe that $n_0 = 2$ ensures that $n_0^2 > n_0$ and that the intervention sets will be disjoint since a vertex will no longer be relevant after intervention so SubsetSearch will never intervene on intervened nodes.

Algorithm 27 TestAndSubsetSearch: Adaptive search algorithm with advice.

```
Input: Essential graph $\mathcal{E}(\mathcal{G}^*)$ and advice DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$
Output: An atomic intervention set $\mathcal{I}$ such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*) = \mathcal{G}^*$
Let $\widetilde{\mathcal{V}}$ be any minimum sized atomic verifying set of $\widetilde{\mathcal{G}}$
Intervene on $\mathcal{I}_0 = \widetilde{\mathcal{V}}$                    ▷ Test quality of $\widetilde{\mathcal{G}}$
Initialize $r \leftarrow 0$, $i \leftarrow 0$, and $n_0 \leftarrow 2$
while $\mathcal{E}_{\mathcal{I}_i}(\mathcal{G}^*)$ still has undirected edges do
    if $\rho(\mathcal{I}_i, N^r(\widetilde{\boldsymbol{V}})) \geq n_i^2$ then
        Increment $i \leftarrow i + 1$ and record $r(i) \leftarrow r$
        Update $n_i \leftarrow \rho(\mathcal{I}_i, N^r(\widetilde{\boldsymbol{V}}))$
        Let $\mathcal{C}_i$ be the intervention set intervened upon by SubsetSearch when run on
        subset $N^{r-1}(\widetilde{\boldsymbol{V}})$          ▷ Essential graph is now $\mathcal{E}_{\mathcal{I}_{i-1} \cup \mathcal{C}_i}(\mathcal{G}^*)$
        if $\mathcal{E}_{\mathcal{I}_{i-1} \cup \mathcal{C}_i}(\mathcal{G}^*)$ still has undirected edges then
            Let $\mathcal{C}_i'$ be the intervention set intervened upon by SubsetSearch when run
            on subset $N^r(\widetilde{\boldsymbol{V}})$          ▷ Essential graph is now $\mathcal{E}_{\mathcal{I}_{i-1} \cup \mathcal{C}_i \cup \mathcal{C}_i'}(\mathcal{G}^*)$
            Update $\mathcal{I}_i \leftarrow \mathcal{I}_{i-1} \cup \mathcal{C}_i \cup \mathcal{C}_i'$
        else
            Update $\mathcal{I}_i \leftarrow \mathcal{I}_{i-1} \cup \mathcal{C}_i$
    Increment $r \leftarrow r + 1$
return $\mathcal{I}_i$
```

We will now prove our main result (Theorem 11.2) which shows that the number of interventions needed is a function of the quality of the given advice DAG.

Theorem 11.2. *Fix an essential graph $\mathcal{E}(\mathcal{G}^*)$ of an unknown underlying DAG $\mathcal{G}^*$. Given an advice graph $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$, TestAndSubsetSearch runs in polynomial time and computes* *an atomic intervention set $\mathcal{I} \subseteq 2^{\boldsymbol{V}}$ in a deterministic and adaptive manner such that $\mathcal{E}_{\mathcal{I}}(\mathcal{G}^*) = \mathcal{G}^*$ and $|\mathcal{I}| \in \mathcal{O}(\max\{1, \log \psi(\mathcal{G}^*, \widetilde{\mathcal{G}})\} \cdot \nu_1(\mathcal{G}^*))$.*

Proof. If Algorithm 27 terminates when $i = 0$, then $\mathcal{I} = \mathcal{I}_0 = \widetilde{\mathcal{V}}$ and Theorem 11.1 tells us that $|\mathcal{I}| \in \mathcal{O}(\nu_1(\mathcal{G}^*))$. In the remaining of this proof, let us suppose Algorithm 27 terminates with $i = t$, for some final round $t > 0$.

As TestAndSubsetSearch uses an arbitrary verifying set of $\widetilde{\mathcal{G}}$ in step 3, we will argue that $\mathcal{O}(\max\{1, \log |N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}})|\} \cdot \nu_1(\mathcal{G}^*))$ interventions are used in the while-loop, for any arbitrary chosen verifying set $\widetilde{\mathcal{V}} \subseteq 2^{\boldsymbol{V}}$ of $\widetilde{\mathcal{G}}$. The theorem then follows by taking a maximization over all possible verifying sets.

In Line 6, $r(i)$ records the hop value such that $\rho(\mathcal{I}_i, N^{r(i)}(\widetilde{\boldsymbol{V}})) \geq n_i^2$, for any $0 \leq i < t$. By construction of the algorithm, we know the following:

1. For any $0 < i \leq t$,

$$\rho(\mathcal{I}_i, N^{r(i)}(\widetilde{\boldsymbol{V}})) = n_i \geq n_{i-1}^2 > \rho(\mathcal{I}_i, N^{r(i)-1}(\widetilde{\boldsymbol{V}})) \tag{11.2}$$

   because $r(i) - 1$ did *not* trigger TestAndSubsetSearch to record $r(i)$ on Line 6.

2. For any $1 \leq i \leq t$, we have

$$\begin{aligned} |\mathcal{C}_i| &\in \mathcal{O}(\log \rho(\mathcal{I}_i, N^{r(i)-1}(\widetilde{\boldsymbol{V}})) \cdot \nu_1(\mathcal{G}^*)) && \subseteq \mathcal{O}(\log(n_{i-1}) \cdot \nu_1(\mathcal{G}^*)) \\ |\mathcal{C}_i'| &\in \mathcal{O}(\log \rho(\mathcal{I}_i, N^{r(i)}(\widetilde{\boldsymbol{V}})) \cdot \nu_1(\mathcal{G}^*)) && \subseteq \mathcal{O}(\log(n_i) \cdot \nu_1(\mathcal{G}^*)) \end{aligned} \tag{11.3}$$

   by applying Theorem 6.24 and then Eq. (11.2) on $|\mathcal{C}_i|$ and $|\mathcal{C}_i'|$ separately. Note that the bound for $|\mathcal{C}_i'|$ is an *over-estimation* (but this is okay for our analytical purposes) since some nodes previously counted for $\rho(\mathcal{I}_i, N^{r(i)}(\widetilde{\boldsymbol{V}}))$ may no longer be relevant in $\mathcal{E}_{\mathcal{I}_i \cup C_i}(\mathcal{G}^*)$ after intervening on $\mathcal{C}_i$.

3. Since $n_{i-1} \leq \sqrt{n_i}$ for any $0 < i \leq t$, we know that $n_j \leq n_t^{\frac{1}{2^{t-j}}}$ for any $0 \leq j \leq t$. So, for any $0 \leq t' \leq t$, we have

$$\sum_{i=0}^{t'} \log(n_i) \leq \sum_{i=0}^{t'} \log\left(n_{t'}^{\frac{1}{2^{t'-i}}}\right) = \sum_{i=0}^{t'} \frac{\log(n_{t'})}{2^{t'-i}} \leq 2 \cdot \log(n_{t'}) \tag{11.4}$$

4. By definitions of r, t, and $h(\mathcal{G}^*, \widetilde{\mathcal{V}})$, and Lemma 11.8, we have

$$\begin{aligned} r(t-1) < h(\mathcal{G}^*, \widetilde{\mathcal{V}}) &\leq r(t) \\ N^{r(t-1)}(\widetilde{\boldsymbol{V}}) \subsetneq N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}}) &\subseteq N^{r(t)}(\widetilde{\boldsymbol{V}}) \\ |N^{r(t-1)}(\widetilde{\boldsymbol{V}})| < |N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}})| &\leq |N^{r(t)}(\widetilde{\boldsymbol{V}})| \end{aligned} \tag{11.5}$$

5. Combining the above, we get

$$\begin{aligned}
\sum_{i=1}^{t-1} (|\mathcal{C}_i| + |\mathcal{C}'_i|) &\in \mathcal{O}\left(\left(\sum_{i=1}^{t-1} \log(n_{i-1}) + \log(n_i)\right) \cdot \nu_1(\mathcal{G}^*)\right) && \text{By Eq. (11.3)}\\
&\subseteq \mathcal{O}\left(\sum_{i=1}^{t-1} \log(n_i) \cdot \nu_1(\mathcal{G}^*)\right) && \text{By Eq. (11.2)}\\
&\subseteq \mathcal{O}\left(\log(n_{t-1}) \cdot \nu_1(\mathcal{G}^*)\right) && \text{By Eq. (11.4)}
\end{aligned}$$

That is,

$$\sum_{i=1}^{t-1} (|\mathcal{C}_i| + |\mathcal{C}'_i|) \subseteq \mathcal{O}\left(\log n_{t-1} \cdot \nu_1(\mathcal{G}^*)\right) \tag{11.6}$$

To relate $|\mathcal{I}_t|$ with $|N^{h(\mathcal{G}^*,\widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}})|$, we consider two scenarios depending on whether the essential graph was fully oriented after intervening on $\mathcal{C}_t$ or $\mathcal{C}'_t$. In either case, remember that $|\widetilde{\mathcal{V}}| \in \mathcal{O}(\nu_1(\mathcal{G}^*))$ via Theorem 11.1.

Scenario 1: Fully oriented after intervening on $\mathcal{C}_t$, i.e. $\mathcal{E}_{\mathcal{I}_{t-1} \cup \mathcal{C}_t}(\mathcal{G}^*) = \mathcal{G}^*$
(In this case, $h(\mathcal{G}^*, \widetilde{\mathcal{V}}) \leq r(t) - 1$, but this information is not useful for the analysis.)
Since $n_{t-1} \leq |N^{r(t-1)}(\widetilde{\boldsymbol{V}})|$ by definition, Eq. (11.5) tells us that

$$n_{t-1} < |N^{h(\mathcal{G}^*,\widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}})| \tag{11.7}$$

Meanwhile,

$$\mathcal{I}_t = \mathcal{C}_t \cup \mathcal{I}_{t-1} = \mathcal{C}_t \cup (\mathcal{C}_{t-1} \cup \mathcal{C}'_{t-1}) \cup \mathcal{I}_{t-2} = \ldots = \mathcal{C}_t \cup \bigcup_{i=1}^{t-1} (\mathcal{C}_i \cup \mathcal{C}'_i) \cup \widetilde{\mathcal{V}}$$

Recalling that $2 = n_0 \leq n_{t-1}$, with $n_0 \leq n_{t-1}$ in case $t = 1$, we see that

$$\begin{aligned}
|\mathcal{I}_t| &\leq |\mathcal{C}_t| + \sum_{i=1}^{t-1} (|\mathcal{C}_i| + |\mathcal{C}'_i|) + |\widetilde{\mathcal{V}}|\\
&\in \mathcal{O}\left(\log(n_{t-1}) \cdot \nu_1(\mathcal{G}^*)\right) + \sum_{i=1}^{t-1} (|\mathcal{C}_i| + |\mathcal{C}'_i|) + |\widetilde{\mathcal{V}}| && \text{By Eq. (11.3)}\\
&\in \mathcal{O}\left(\log(n_{t-1}) \cdot \nu_1(\mathcal{G}^*)\right) + \mathcal{O}\left(\log(n_{t-1}) \cdot \nu_1(\mathcal{G}^*)\right) + |\widetilde{\mathcal{V}}| && \text{By Eq. (11.6)}\\
&\in \mathcal{O}\left(\log(n_{t-1}) \cdot \nu_1(\mathcal{G}^*)\right) + \mathcal{O}\left(\log(n_{t-1}) \cdot \nu_1(\mathcal{G}^*)\right) + \mathcal{O}(\nu_1(\mathcal{G}^*)) && \text{By Theorem 11.1}\\
&\subseteq \mathcal{O}\left(\log |N^{h(\mathcal{G}^*,\widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}})| \cdot \nu_1(\mathcal{G}^*)\right) && \text{By Eq. (11.7)}
\end{aligned}$$

Scenario 2: Fully oriented after intervening on $\mathcal{C}'_t$, i.e. $\mathcal{E}_{\mathcal{I}_{t-1} \cup \mathcal{C}_t \cup \mathcal{C}'_t}(\mathcal{G}^*) = \mathcal{G}^*$
In this case, $h(\mathcal{G}^*, \widetilde{\mathcal{V}}) = r(t)$ and $N^{h(\mathcal{G}^*,\widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}}) = N^{r(t)}(\widetilde{\boldsymbol{V}})$. So,

$$n_t \leq |N^{r(t)}(\widetilde{\boldsymbol{V}})| = |N^{h(\mathcal{G}^*,\widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}})| \tag{11.8}$$

Meanwhile,

$$\mathcal{I}_t = \mathcal{C}_t \cup \mathcal{C}'_t \cup \mathcal{I}_{t-1} = \ldots = \mathcal{C}_t \cup \mathcal{C}'_t \cup \bigcup_{i=1}^{t-1} (\mathcal{C}_i \cup \mathcal{C}'_i) \cup \widetilde{\mathcal{V}}$$

Recalling that $2 = n_0 \leq n_{t-1} < n_t$, with $n_0 \leq n_{t-1}$ in case $t = 1$, we see that

$$\begin{aligned}
|\mathcal{I}_t| &= |C_t| + |C'_t| + \sum_{i=1}^{t-1} (|\mathcal{C}_i| + |\mathcal{C}'_i|) + |\widetilde{\mathcal{V}}| \\
&\in \mathcal{O}\left((\log(n_{t-1}) + \log(n_t)) \cdot \nu_1(\mathcal{G}^*)\right) + \sum_{i=1}^{t-1} (|\mathcal{C}_i| + |\mathcal{C}'_i|) + |\widetilde{\mathcal{V}}| && \text{By Eq. (11.3)} \\
&\in \mathcal{O}\left(\log(n_t) \cdot \nu_1(\mathcal{G}^*)\right) + \sum_{i=1}^{t-1} (|\mathcal{C}_i| + |\mathcal{C}'_i|) + |\widetilde{\mathcal{V}}| && \text{Since } n_{t-1} \leq n_t \\
&\in \mathcal{O}\left(\log(n_t) \cdot \nu_1(\mathcal{G}^*)\right) + \mathcal{O}\left(\log(n_{t-1}) \cdot \nu_1(\mathcal{G}^*)\right) + |\widetilde{\mathcal{V}}| && \text{By Eq. (11.6)} \\
&\in \mathcal{O}\left(\log(n_t) \cdot \nu_1(\mathcal{G}^*)\right) + |\widetilde{\mathcal{V}}| && \text{Since } n_{t-1} \leq n_t \\
&\in \mathcal{O}\left(\log(n_t) \cdot \nu_1(\mathcal{G}^*)\right) + \mathcal{O}(\nu_1(\mathcal{G}^*)) && \text{By Theorem 11.1} \\
&\subseteq \mathcal{O}\left(\log |N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}})| \cdot \nu_1(\mathcal{G}^*)\right) && \text{By Eq. (11.8)}
\end{aligned}$$

In either scenarios, we see that $|\mathcal{I}_t| \in \mathcal{O}\left(\log |N^{h(\mathcal{G}^*, \widetilde{\mathcal{V}})}(\widetilde{\boldsymbol{V}})| \cdot \nu_1(\mathcal{G}^*)\right)$ as desired. The theorem then follows by taking a maximization over all $\widetilde{\mathcal{V}} \in \mathcal{V}(\widetilde{G})$.

Running time. By construction, TestAndSubsetSearch is deterministic. It runs in polynomial time because: (1) Hop information and relevant nodes can be computed in polynomial time via breadth first search and maintaining suitable neighborhood information; (2) It is known that performing Meek rules to obtain essential graphs takes polynomial time [WBL21]; (3) TestAndSubsetSearch makes at most two calls to SubsetSearch whenever the number of relevant nodes is squared (for a total of at most $\mathcal{O}(\log n)$ times) and each SubsetSearch call runs in polynomial time (Theorem 6.24). □

Chapter 12

Conclusion for Part III

The results presented in Chapter 9, Chapter 10, and Chapter 11 are from the works of [CGLB24], [BGGJ$^+$24], and [CGB23] respectively.

In Chapter 9, we studied the online bipartite matching problem with respect to a very natural design goal of 1-consistency and β-robustness; see Goal 9.1. We showed that this goal is impossible under the adversarial arrival model and designed a meta algorithm TestAndMatch for the random arrival model that is 1-consistent and $\beta \cdot (1 - o(1))$-robust while using histograms over arrival types as advice. The guarantees TestAndMatch degrades gracefully as the quality of the advice worsens, and improves whenever the state-of-the-art β improves. The obvious follow-up question is whether our approach extends to other variants of online matching, or even other online problems with random arrivals. For instance, consider the relatively new *adversarial-order model with a sample* (AOS and AOS$_p$) for relaxing the adversarial arrival model [KNR20, KNR22, CCF$^+$24]: a worst-case adversarial input is chosen, a random subset of the online input is revealed upfront to the algorithm, and then performance is measured on the subsequent worst-case adversarial arrivals. In this setup, the random sample of the input allows the algorithm to learn something about the actual instance, but it is *not* allowed to pick anything from this prefix. Our TestAndMatch algorithm would translate to the AOS model when the prefix is sufficiently large for the testing phase, and we can actually obtain β (instead of $\beta \cdot (1 - o(1))$) when the advice is of low quality because we will not incur any loss on the competitive ratio as the prefix is not part of the performance measurement.

In Chapter 10, we revisited the problem of distribution learning within the framework of learning-augmented algorithms, specifically in the context of learning a multivariate Gaussian distribution in a sample efficient manner. Formal details of our lower bound is presented in [BCG$^+$22, Section 5]. An immediate and natural follow up question is whether we can extend the TestAndAct approach to discrete distributions, beyond multivariate Gaussians.

In Chapter 11, we gave the first result that utilizes imperfect advice in the context of causal discovery via TestAndSubsetSearch. We do so in a way that the performance (i.e. the number of interventions in our case) does not degrade significantly even when the advice is inaccurate, which is consistent with the objectives of learning-augmented algorithms. Specifically, we show a smooth bound that matches the number of interventions needed for verification of the causal relationships in a graph when the advice is completely accurate and also depends logarithmically on the distance of the advice to the ground truth. This ensures robustness to "bad" advice, the number of interventions needed is asymptotically the same as in the case where no advice is available.

Apart from the chapter-specific comments above, there is an argument to be made about moving beyond the metrics of consistency and robustness when evaluating learning-augmented algorithms. To be more concrete, observe that TestAndMatch's performance guarantee is based on the L_1 distance over type histograms. This is very sensitive to certain types of noise, e.g. adding or removing edges at random (Erdos-Renyi). However, Section 9.6 suggests there are practical extensions that hold even when L_1 is large, implying it is a non-ideal metric despite satisfying consistency and robustness. Is there another criterion that could fill this gap? Could we formalize how practical/reasonable/brittle an advice is? It would also be interesting to explore the human interaction/interface portion of how to elicit useful advice from human domain experts. For instance, the notion of "confidence level" and "correctness" of an advice are orthogonal issues – an expert can be confidently wrong. In Chapter 11, we focused on the case where the expert is fully confident but may be providing imperfect advice. It is an interesting problem to investigate how to principally handle both issues simultaneously; for example, what if the advice is not a DAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ in the essential graph but a distribution over all DAGs in $[\mathcal{G}^*]$? Bayesian ideas may apply here.

12.1 Some additional related work

12.1.1 Learning-augmented algorithms for matching

[ACI22] studied the adversarial arrival models with offline vertex degrees as advice. While their algorithm is optimal under the Chung-Lu-Vu random graph model [CLV03], the class of offline degree advice is unable to attain 1-consistency. [FNS21] propose a two-stage vertex-weighted variant, where advice is a proposed matching for the online vertices arriving in the first stage. [JM22] showed in this setting a tight robustness-consistency tradeoff and derive a continuum of algorithms tracking this Pareto frontier. [AGKK23] studied settings with random vertex arrival and weighted edges. Their advice is a prediction on edge weights adjacent to V under an optimal offline matching. Furthermore, their algorithm and analysis uses a hyper-pamareter quatifying confidence in the advice, leading to different consistency and robustness tradeoffs. Another relevant work is the LOMAR method proposed by [LYR23]. Using a pre-trained reinforcement learning (RL) model along with a switching mechanism based on regret to guarantee robustness with respect to any provided expert algorithm, they claim "for some tuning parameter $\rho \in [0, 1]$, LOMAR is ρ-competitive against our choice of expert online algorithm". We differ from LOMAR in two key ways:

1. Our method does not require any pre-training phase and directly operate on the sequence of online vertices themselves. This means that whatever mistakes made during our "testing" phase contributes to our competitive ratio; a key technical contribution is the use of distribution testing to ensure that the number of such mistakes incurred is sublinear.

2. The robustness guarantee of [LYR23] is substantially weaker than what we provide. Suppose the expert used by LOMAR is β-competitive, just like how we use the state-of-the-art algorithm as the baseline. Although [LYR23] does not analyze the consistency guarantee of their method, one can see that LOMAR is $(1 - \rho)$-consistent and $\rho \cdot \beta$-robust (ignoring the $B \geq 0$ hyperparameter). LOMAR can only be 1-consistent when $\rho = 0$, i.e. it blindly follows the RL-based method; but then it will have no robustness guarantees. In other words, LOMAR cannot simultaneously achieve 1-consistency and $\rho \cdot \beta$-robustness without knowing the RL quality. In contrast, our method is *simultaneously* 1-consistent and $\approx \beta$-robust *without* knowing the quality of our given advice; we evaluate its quality as vertices arrive.

Table 12.1 compares the consistency-robustness tradeoffs.

	[JM22]	LOMAR	Ours
Robustness	R	$\rho \cdot \beta$	$\approx \beta$
Consistency	$1 - (1 - \sqrt{1 - R})^2$	$1 - \rho$	1

Table 12.1: Consistency-robustness guarantees of methods that can achieve 1-consistency. Here, $R \in [0, 3/4]$ and $\rho \in [0, 1]$. Note that [JM22] is for the 2-staged setting.

More broadly, [LMRX21a, LMRX21b] learn and exploit parameters of the online matching problem and provide PAC-style guarantees. [DIL$^+$22] studied the use of multiple advice and seek to compete with the best on a per-instance basis. Finally, others suggest using advice to speedup offline matching via "warm-start" heuristics [DIL$^+$21, CSVZ22, SO22].

12.1.2 Expert advice in causal graph discovery

There are three main types of information that a domain expert may provide (e.g. see the references given in Chapter 11):

(I) Required parental arcs: $X \to Y$

(II) Forbidden parental arcs: $X \not\to Y$

(III) Partial order or tiered knowledge: A partition of the n variables into $1 \leq t \leq n$ sets $S_1, \ldots, S_t$ such that variables in S_i *cannot come after* S_j, for all $i < j$.

In the context of orienting unoriented $X - Y$ edges in an essential graph, it suffices to consider only information of type (I): $X \not\to Y$ implies $Y \to X$, and a partial order can be converted to a collection of required parental arcs. For every edge $X - Y$ with $X \in S_i$ and $Y \in S_j$, enforce the required parental arc $X \to Y$ if and only if $i < j$.

Maximally oriented partially directed acyclic graphs (MPDAGs), a refinement of essential graphs under additional causal information, are often used to model such expert advice and there has been a recent growing interest in understanding them better [PKM17, Per20, GP21]. MPDAGs are obtained by orienting additional arc directions in the essential graph due to background knowledge, and then applying Meek rules. See Fig. 12.1 for an example.

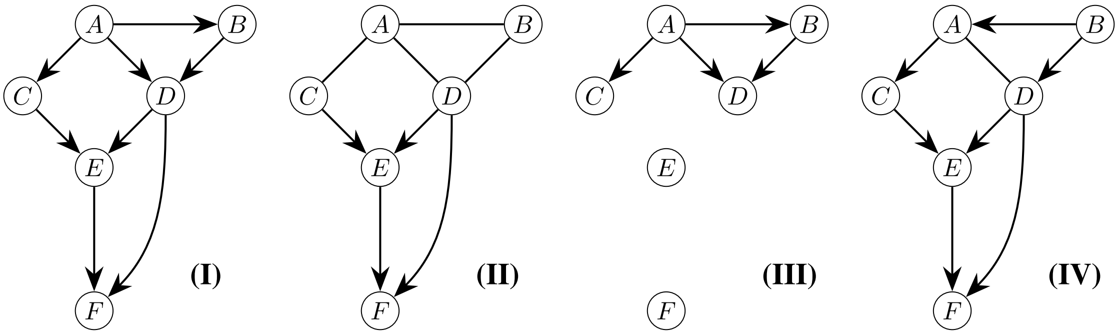


Figure 12.1: **(I)** Ground truth DAG $\mathcal{G}^*$; **(II)** Observational essential graph $\mathcal{E}(\mathcal{G}^*)$ where $C \to E \leftarrow D$ is a v-structure and Meek rules orient arcs $D \to F$ and $E \to F$; **(III)** $\mathcal{G}^{\emptyset} = \mathcal{G}[\boldsymbol{E} \setminus \boldsymbol{R}(\mathcal{G}, \emptyset)]$ where oriented arcs in $\mathcal{E}(\mathcal{G}^*)$ are removed from G^*; **(IV)** MPDAG $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$ incorporating the following partial order advice $(\boldsymbol{S}_1 = \{B\}, \boldsymbol{S}_2 = \{A, D\}, \boldsymbol{S}_3 = \{C, E, F\})$, which can be converted to required arcs $B \to A$ and $B \to D$. Observe that $A \to C$ is oriented by Meek R1 via $B \to A - C$, the arc $A - D$ is still unoriented, the arc $B \to A$ disagrees with G^*, and there are two possible DAGs consistent with the resulting MPDAG.

12.2 Other unpresented works in Part III

In [CL24], we studied the secretary problem through the lens of learning-augmented algorithms and showed an impossibility result that is similar in style to Goal 9.1 in [CGLB24]. To be precise, we gave a simple construction showing that no learning-augmented algorithm for the secretary problem that is 1-consistent can have robustness guarantee better than $1/3 + o(1)$, even when the candidates' true values are constants that do not scale with the number of candidates.

Appendix A

Addendum for Part I

A.1 Addendum for Chapter 3

A.1.1 Derivation for KL decomposition

In this section, we provide the full derivations of Eq. (3.3) and Eq. (3.4).

We first establish Eq. (3.3). Observe that

$$\begin{aligned}
&\mathrm{d}_{\mathrm{KL}}(\mathcal{P}, \widehat{\mathcal{P}}) \\
&= \int_{\boldsymbol{x}} \mathcal{P}(\boldsymbol{x}) \log \left(\frac{\mathcal{P}(\boldsymbol{x})}{\widehat{\mathcal{P}}(\boldsymbol{x})} \right) d\boldsymbol{x} \\
&= \int_{\boldsymbol{x}} \mathcal{P}(\boldsymbol{x}) \log \left(\frac{\Pi_{i=1}^{n} \mathcal{P}(x_i \mid \mathrm{pa}(X_i))}{\Pi_{i=1}^{n} \widehat{\mathcal{P}}(x_i \mid \mathrm{pa}(X_i))} \right) d\boldsymbol{x} \\
&\qquad\qquad\qquad\qquad \text{(Bayesian network decomposition of joint probabilities)} \\
&= \sum_{i=1}^{n} \int_{\boldsymbol{x}} \mathcal{P}(\boldsymbol{x}) \log \left(\frac{\mathcal{P}(x_i \mid \mathrm{pa}(X_i))}{\widehat{\mathcal{P}}(x_i \mid \mathrm{pa}(X_i))} \right) d\boldsymbol{x} \\
&= \sum_{i=1}^{n} \int_{\mathrm{pa}(X_i)} \int_{x_i} \mathcal{P}(x_i, \mathrm{pa}(X_i)) \log \left(\frac{\mathcal{P}(x_i \mid \mathrm{pa}(X_i))}{\widehat{\mathcal{P}}(x_i \mid \mathrm{pa}(X_i))} \right) dx_i \, d\mathrm{pa}(X_i) \quad \text{(Marginalization)} \\
&= \sum_{i=1}^{n} \mathrm{d}_{\mathrm{CP}}(\boldsymbol{\alpha}_i^*, \widehat{\boldsymbol{\alpha}}_i)
\end{aligned}$$

For Eq. (3.4), consider an arbitrary variable Y with p parents and associated parameters $\boldsymbol{a}^*$ and σ^*. If $p = 0$, then $\boldsymbol{a}^* = \mathbf{0}$ (the all-zero vector) and we can simply set the coefficients $\widehat{\boldsymbol{a}} = \mathbf{0}$. Meanwhile, if $p \geq 1$, we may assume w.l.o.g. that $X_1, \ldots, X_p$ are the parents of Y by relabeling. Let matrix $\boldsymbol{M} \in \mathbb{R}^{p \times p}$ denote the covariance matrix defined by the parents of Y, where the (i,j)-th entry of $\boldsymbol{M}$ is $\mathbb{E}[X_i X_j]$. Under this notation, we see the vector $(X_1, \ldots, X_p) \sim N(0, \boldsymbol{M})$ is distributed as a multivariate Gaussian. Let us further define $\boldsymbol{\Delta} = \widehat{\boldsymbol{a}} - \boldsymbol{a}^*$ as the entry-wise difference vector.

We can write the conditional distribution density of Y as

$$\Pr\left(y \mid \boldsymbol{x}, \boldsymbol{a}^*, \sigma^*\right) = \frac{1}{\sigma^* \sqrt{2\pi}} \exp\left(-\frac{1}{2(\sigma^*)^2} \cdot \left(y - \sum_{i=1}^{p} a_i^* x_i \right)^2 \right)$$

We now analyze $\mathrm{d_{CP}}(\boldsymbol{\alpha}^*, \widehat{\boldsymbol{\alpha}}^*)$ with respect to the our estimates $\widehat{\boldsymbol{\alpha}} = (\widehat{\boldsymbol{a}}, \widehat{\sigma})$ and the hidden true parameters $\boldsymbol{\alpha}^* = (\boldsymbol{a}^*, \sigma^*)$, where $\widehat{\boldsymbol{a}} = (\widehat{\boldsymbol{a}}_{y,1}, \ldots, \widehat{\boldsymbol{a}}_{y,p})$ and $\boldsymbol{a}^* = (\boldsymbol{a}^*_{y,1}, \ldots, \boldsymbol{a}^*_{y,p})$. With respect to variable Y with parents $\boldsymbol{X} = (X_1, \ldots, X_p)$, we see that

$$\begin{aligned}
&\mathrm{d_{CP}}(\boldsymbol{\alpha}^*, \widehat{\boldsymbol{\alpha}}) \\
&= \int_{\boldsymbol{x}} \int_{y} \mathcal{P}(\boldsymbol{x}, y) \log \left(\frac{\frac{1}{\sigma^* \sqrt{2\pi}} \exp\left(-\frac{1}{2(\sigma^*)^2} \cdot \left(y - \sum_{i=1}^{p} \boldsymbol{a}_{y,i} x_i\right)^2 \right)}{\frac{1}{\widehat{\sigma} \sqrt{2\pi}} \exp\left(-\frac{1}{2\widehat{\sigma}^2} \cdot \left(y - \sum_{i=1}^{p} \widehat{\boldsymbol{a}}_{y,i} x_i\right)^2 \right)} \right) dy\, d\boldsymbol{x} \\
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) - \frac{1}{2(\sigma^*)^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left(y - \sum_{i=1}^{p} \boldsymbol{a}_{y,i} x_i \right)^2 + \frac{1}{2\widehat{\sigma}^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left(y - \sum_{i=1}^{p} \widehat{\boldsymbol{a}}_{y,i} x_i \right)^2 \\
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) - \frac{1}{2(\sigma^*)^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left(y - (\boldsymbol{a}^*)^\top \boldsymbol{x} \right)^2 + \frac{1}{2\widehat{\sigma}^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left(y - \widehat{\boldsymbol{a}}^\top \boldsymbol{x} \right)^2
\end{aligned}$$

Since $\boldsymbol{\Delta} = \widehat{\boldsymbol{a}} - \boldsymbol{a}^*$ is the entry-wise difference vector, we can see that for any instantiation of y and $\boldsymbol{x}$,

$$\begin{aligned}
\left(y - \widehat{\boldsymbol{a}}^\top \boldsymbol{x}\right)^2 &= \left(y - (\boldsymbol{\Delta} + \boldsymbol{a}^*)^\top \boldsymbol{x}\right)^2 && \text{(By definition of } \boldsymbol{\Delta}\text{)} \\
&= \left((y - (\boldsymbol{a}^*)^\top \boldsymbol{x}) - \boldsymbol{\Delta}^\top \boldsymbol{x}\right)^2 \\
&= (y - (\boldsymbol{a}^*)^\top \boldsymbol{x})^2 - 2(y - (\boldsymbol{a}^*)^\top \boldsymbol{x})(\boldsymbol{\Delta}^\top \boldsymbol{x}) + \left(\boldsymbol{\Delta}^\top \boldsymbol{x}\right)^2 \\
&= (y - (\boldsymbol{a}^*)^\top \boldsymbol{x})^2 - 2\left(y \boldsymbol{\Delta}^\top \boldsymbol{x} - (\boldsymbol{a}^*)^\top \boldsymbol{x} \boldsymbol{\Delta}^\top \boldsymbol{x}\right) + \left(\boldsymbol{\Delta}^\top \boldsymbol{x}\right)^2 \\
&= (y - (\boldsymbol{a}^*)^\top \boldsymbol{x})^2 - 2\left(y \boldsymbol{x}^\top \boldsymbol{\Delta} - (\boldsymbol{a}^*)^\top \boldsymbol{x} \boldsymbol{x}^\top \boldsymbol{\Delta}\right) + \boldsymbol{\Delta}^\top \boldsymbol{x} \boldsymbol{x}^\top \boldsymbol{\Delta} \\
& && \text{(Since } \boldsymbol{\Delta}^\top \boldsymbol{x} \text{ is just a number)}
\end{aligned}$$

Recall that the matrix $\boldsymbol{M} \in \mathbb{R}^{p \times p}$ denotes the covariance matrix defined by the parents of Y, where the (i, j)-th entry of $\boldsymbol{M}$ is $\mathbb{E}[X_i X_j]$. Then, we can further simplify $\mathrm{d_{CP}}(\boldsymbol{\alpha}^*, \widehat{\boldsymbol{\alpha}})$ as follows:

$$\begin{aligned}
&\mathrm{d_{CP}}(\boldsymbol{\alpha}^*, \widehat{\boldsymbol{\alpha}}) \\
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) - \frac{1}{2(\sigma^*)^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left(y - (\boldsymbol{a}^*)^\top \boldsymbol{x} \right)^2 + \frac{1}{2\widehat{\sigma}^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left(y - \widehat{\boldsymbol{a}}^\top \boldsymbol{x} \right)^2 \quad \text{(From above)} \\
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) - \frac{1}{2(\sigma^*)^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left(y - (\boldsymbol{a}^*)^\top \boldsymbol{x} \right)^2 \\
&\quad + \frac{1}{2\widehat{\sigma}^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left[(y - (\boldsymbol{a}^*)^\top \boldsymbol{x})^2 - 2\left(y \boldsymbol{x}^\top \boldsymbol{\Delta} - (\boldsymbol{a}^*)^\top \boldsymbol{x} \boldsymbol{x}^\top \boldsymbol{\Delta}\right) + \boldsymbol{\Delta}^\top \boldsymbol{x} \boldsymbol{x}^\top \boldsymbol{\Delta} \right] \\
&\qquad \text{(From above)}
\end{aligned}$$

$$
\begin{aligned}
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) - \frac{1}{2(\sigma^*)^2} \cdot \mathbb{E}_{\boldsymbol{x},y}\, \eta^2 + \frac{1}{2\widehat{\sigma}^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left[\eta^2 - 2\left(\eta \boldsymbol{x}^\top \boldsymbol{\Delta}\right) + \boldsymbol{\Delta}^\top \boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{\Delta}\right] \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{(Since } y = \eta + (\boldsymbol{a}^*)^\top \boldsymbol{x}\text{)} \\
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) - \frac{1}{2} + \frac{1}{2\widehat{\sigma}^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left[(\sigma^*)^2 - 2\left(\eta \boldsymbol{x}^\top \boldsymbol{\Delta}\right) + \boldsymbol{\Delta}^\top \boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{\Delta}\right] \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\text{(Since } \eta \sim N(0, (\sigma^*)^2)\text{)} \\
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) - \frac{1}{2} + \frac{1}{2\widehat{\sigma}^2} \cdot \mathbb{E}_{\boldsymbol{x},y} \left[(\sigma^*)^2 - 0 + \boldsymbol{\Delta}^\top \boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{\Delta}\right] \\
&\qquad\qquad\qquad\qquad\qquad\text{(Since } \mathbb{E}_{\boldsymbol{x},y}\left(\eta \boldsymbol{x}^\top \boldsymbol{\Delta}\right) = (\mathbb{E}_{\boldsymbol{x},y}\, \eta) \cdot (\mathbb{E}_{\boldsymbol{x},y}\, \boldsymbol{x}^\top \boldsymbol{\Delta}) = 0\text{)} \\
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) - \frac{1}{2} + \frac{1}{2\widehat{\sigma}^2} \cdot \left((\sigma^*)^2 - 0 + \boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}\right) \\
&\qquad\qquad\qquad\qquad\text{(Since } \mathbb{E}_{\boldsymbol{x},y}\, \boldsymbol{\Delta}^\top \boldsymbol{x}\boldsymbol{x}^\top \boldsymbol{\Delta} = \boldsymbol{\Delta}^\top (\mathbb{E}_{\boldsymbol{x},y}\, \boldsymbol{x}\boldsymbol{x}^\top) \boldsymbol{\Delta} = \boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}\text{)} \\
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) - \frac{1}{2} + \frac{(\sigma^*)^2}{2\widehat{\sigma}^2} + \frac{\boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}}{2\widehat{\sigma}^2} \\
&= \log\left(\frac{\widehat{\sigma}}{\sigma^*}\right) + \frac{(\sigma^*)^2 - \widehat{\sigma}^2}{2\widehat{\sigma}^2} + \frac{\boldsymbol{\Delta}^\top \boldsymbol{M} \boldsymbol{\Delta}}{2\widehat{\sigma}^2}
\end{aligned}
$$

A.1.2 Deferred proofs

Lemma 3.3. *Let $\boldsymbol{G} \in \mathbb{R}^{k \times d}$ be a matrix with i.i.d. $N(0,1)$ entries. Then, for any constant $0 < c_1 < 1/2$ and $k \geq d/c_1^2$,*

$$
\Pr\left(\|(\boldsymbol{G}^\top \boldsymbol{G})^{-1}\| \leq \frac{1}{(1-2c_1)^2 k}\right) \geq 1 - \exp\left(-\frac{kc_1^2}{2}\right)
$$

Proof. Observe that $\boldsymbol{G}^\top \boldsymbol{G}$ is symmetric, thus $(\boldsymbol{G}^\top \boldsymbol{G})^{-1}$ is also symmetric and the eigenvalues of $\boldsymbol{G}^\top \boldsymbol{G}$ equal the singular values of $\boldsymbol{G}^\top \boldsymbol{G}$. Also, note that event that $\boldsymbol{G}^\top \boldsymbol{G}$ is singular has measure 0. To see this, consider fixing all but one arbitrary entry of $\boldsymbol{G}$. The event of this independent $N(0,1)$ entry making $\det(\boldsymbol{G}^\top \boldsymbol{G}) = 0$ has measure 0.

By definition of operation norm, $\|(\boldsymbol{G}^\top \boldsymbol{G})^{-1}\|$ equals the square root of *maximum* eigenvalue of

$$
((\boldsymbol{G}^\top \boldsymbol{G})^{-1})^\top ((\boldsymbol{G}^\top \boldsymbol{G})^{-1}) = ((\boldsymbol{G}^\top \boldsymbol{G})^{-1})^2,
$$

where the equality is because $(\boldsymbol{G}^\top \boldsymbol{G})^{-1}$ is symmetric. Since $\boldsymbol{G}^\top \boldsymbol{G}$ is invertible, we have $\|(\boldsymbol{G}^\top \boldsymbol{G})^{-1}\| = 1/\|\boldsymbol{G}^\top \boldsymbol{G}\|$, which is equal to the inverse of *minimum* eigenvalue $\lambda_{\min}(\boldsymbol{G}^\top \boldsymbol{G})$ of $\boldsymbol{G}^\top \boldsymbol{G}$, which is in turn equal to the square of minimum singular value $\sigma_{\min}(\boldsymbol{G})$ of $\boldsymbol{G}$. Therefore, the following holds with probability at least $1 - \exp\left(-kc_1^2/2\right)$:

$$
\begin{aligned}
\|\left(\boldsymbol{G}^\top \boldsymbol{G}\right)^{-1}\| = \frac{1}{\|\boldsymbol{G}^\top \boldsymbol{G}\|} = \frac{1}{\lambda_{\min}(\boldsymbol{G}^\top \boldsymbol{G})} &= \frac{1}{\sigma_{\min}^2(\boldsymbol{G})} \\
&\leq \frac{1}{\left(\sqrt{k}(1-c_1) - \sqrt{d}\right)^2} \leq \frac{1}{(1-2c_1)^2 k}
\end{aligned}
$$

where the second last inequality is due to Lemma 2.6 and the last inequality holds when $k \geq d/c_1^2$. □

Lemma 3.4. *Let $\boldsymbol{G} \in \mathbb{R}^{k\times p}$ be a matrix with i.i.d. $N(0,1)$ entries and $\boldsymbol{\eta} \in \mathbb{R}^k$ be a vector with i.i.d. $N(0,\sigma^2)$ entries, where $\boldsymbol{G}$ and $\boldsymbol{\eta}$ are independent. Then, for any constant $c_2 > 0$,*

$$\Pr\left(\|\boldsymbol{G}^\top\boldsymbol{\eta}\| < 2\sigma c_2\sqrt{kp}\right) \geq 1 - 2p\exp\left(-2k\right) - p\exp\left(-\frac{c_2^2}{2}\right)$$

Proof. Let us denote $\boldsymbol{g}_r \in \mathbb{R}^k$ as the r^{th} row of $\boldsymbol{G}^\top$. Then, we see that $\|\boldsymbol{G}^\top\boldsymbol{\eta}\|_2^2 = \sum_{r=1}^p\langle\boldsymbol{g}_r,\boldsymbol{\eta}\rangle^2$. For any row r, we see that $\langle\boldsymbol{g}_r,\boldsymbol{\eta}\rangle = \|\boldsymbol{\eta}\|_2\cdot\langle\boldsymbol{g}_r,\boldsymbol{\eta}/\|\boldsymbol{\eta}\|_2\rangle$. We will bound values of $\|\boldsymbol{\eta}\|_2$ and $|\langle\boldsymbol{g}_r,\boldsymbol{\eta}/\|\boldsymbol{\eta}\|_2\rangle|$ separately.

It is well-known (e.g. see [JNG+19, Lemma 2]) that the norm of a Gaussian vector concentrates around its mean. So, $\Pr\left(\|\boldsymbol{\eta}\|_2 \leq 2\sigma\sqrt{k}\right) \leq 2\exp\left(-2k\right)$. Since $\boldsymbol{g}_r \sim N(0, I_k)$ and $\boldsymbol{\eta}$ are independent, we see that $\langle\boldsymbol{g}_r,\boldsymbol{\eta}/\|\boldsymbol{\eta}\|_2\rangle \sim N(0,1)$. By standard Gaussian bounds, we have that $\Pr\left(|\langle\boldsymbol{g}_r,\boldsymbol{\eta}/\|\boldsymbol{\eta}\|_2\rangle| \geq c_2\right) \leq \exp\left(-c_2^2/2\right)$.

By applying a union bound over these two events, we see that $\|\langle\boldsymbol{g}_r,\boldsymbol{\eta}\rangle\| < 2\sigma c_2\sqrt{k}$ for any row with probability $2\exp\left(-2k\right) + \exp\left(-c_2^2/2\right)$. The claim follows from applying a union bound over all p rows. □

Lemma 3.5 (Non-asymptotic convergence of Cauchy median)**.** *Consider a collection of m i.i.d.* $\mathrm{Cauchy}(0,1)$ *random variables $X_1,\ldots,X_m$. Given a threshold $0 < \tau < 1$, we have*

$$\Pr\left(\mathrm{median}\{X_1,\ldots,X_m\} \notin [-\tau,\tau]\right) \leq 2\exp\left(-\frac{m\tau^2}{8}\right)$$

Proof. Let $s_{>\tau} = \sum_{i=1}^m \mathbb{1}_{X_i>\tau}$ be the number of values that are larger than τ, where $\mathbb{E}[\mathbb{1}_{X_i>\tau}] = \Pr(X \geq \tau)$. Similarly, let $s_{<-\tau}$ be the number of values that are smaller than $-\tau$. If $s_{>\tau} < m/2$ *and* $s_{<-\tau} < m/2$, then we see that $\mathrm{median}\{X_1,\ldots,X_m\} \in [-\tau,\tau]$.

For a random variable $X \sim \mathrm{Cauchy}(0,1)$, we know that $\Pr(X \leq x) = 1/2 + \arctan(x)/\pi$. For $0 < \tau < 1$, we see that $\Pr(X \geq \tau) = 1/2 - \arctan(\tau)/\pi \leq 1/2 - \tau/4$. By additive Chernoff bounds, we see that

$$\Pr\left(s_{>\tau} \geq \frac{m}{2}\right) \leq \exp\left(-\frac{2m^2\tau^2}{16m}\right) = \exp\left(-\frac{m\tau^2}{8}\right)$$

Similarly, we have $\Pr\left(s_{<-\tau} \geq m/2\right) \leq \exp\left(-m\tau^2/8\right)$. The claim follows from a union bound over the events $s_{>\tau} \geq m/2$ and $s_{<-\tau} \geq m/2$. □

Lemma 3.6. *Consider the matrix equation $\boldsymbol{AB} = \boldsymbol{E}$ where $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B} \in \mathbb{R}^{n\times 1}$, and $\boldsymbol{E} \in R^{n\times 1}$ such that entries of $\boldsymbol{A}$ and $\boldsymbol{E}$ are independent Gaussians, elements in each* column *of $\boldsymbol{A}$ have the same variance, and all entries in $\boldsymbol{E}$ have the same variance. That is, $\boldsymbol{A}_{\cdot,j} \sim N(0,\sigma_i^2)$ and $\boldsymbol{E}_i \sim N(0,\sigma_{n+1}^2)$. Then, for all $i \in [n]$, we have that $\boldsymbol{B}_i \sim \frac{\sigma_{n+1}}{\sigma_i}\cdot\mathrm{Cauchy}(0,1)$.*

Proof. As the event that $\boldsymbol{A}$ is singular has measure zero, we can write $\boldsymbol{B} = \boldsymbol{A}^{-1}\boldsymbol{E}$. By Cramer's rule,

$$\boldsymbol{A}^{-1} = \frac{1}{\det(\boldsymbol{A})} \cdot \operatorname{adj}(\boldsymbol{A}) = \frac{1}{\det(\boldsymbol{A})} \cdot \boldsymbol{C}^\top$$

where $\det(\boldsymbol{A})$ is the determinant of $\boldsymbol{A}$, $\operatorname{adj}(\boldsymbol{A})$ is the adjugate/adjoint matrix of $\boldsymbol{A}$, and $\boldsymbol{C}$ is the cofactor matrix of $\boldsymbol{A}$. Recall that the $\det(\boldsymbol{A})$ can defined with respect to elements in $\boldsymbol{C}$. So, for any *column* $i \in [n]$,

$$\det(\boldsymbol{A}) = \boldsymbol{A}_{1,i} \cdot \boldsymbol{C}_{1,i} + \boldsymbol{A}_{2,i} \cdot \boldsymbol{C}_{2,i} + \ldots + \boldsymbol{A}_{n,i} \cdot \boldsymbol{C}_{n,i}$$

So, $\det(\boldsymbol{A}) \sim N\left(0, \sigma_i^2 \left(\boldsymbol{C}_{1,i} + \ldots + \boldsymbol{C}_{n,i}\right)\right)$. Thus, for any $i \in [n]$,

$$\boldsymbol{B}_i = \left(\frac{1}{\det(\boldsymbol{A})}\boldsymbol{C}^\top \boldsymbol{E}\right)_i \sim \frac{N\left(0, \sigma_{n+1}^2 \left(\boldsymbol{C}_{1,i} + \ldots + \boldsymbol{C}_{n,i}\right)\right)}{N\left(0, \sigma_i^2 \left(\boldsymbol{C}_{1,i} + \ldots + \boldsymbol{C}_{n,i}\right)\right)} = \frac{\sigma_{n+1}}{\sigma_i} \cdot \operatorname{Cauchy}(0,1) \quad \square$$

A.2 Addendum for Chapter 4

A.2.1 Adapting the known tester result of [BGP+23]

Corollary 4.4 is adapted from Theorem 1.3 of [BGP+23].

Theorem A.1 (Conditional MI Tester, [BGP+23, Theorem 1.3])**.** *Fix any $\varepsilon > 0$. Let (X, Y, Z) be three random variables over $\Sigma_X, \Sigma_Y, \Sigma_Z$ respectively. Given the empirical distribution $(\hat{X}, \hat{Y}, \hat{Z})$ over a size N sample of (X, Y, Z), there exists a universal constant $0 < c < 1$ so that for any N at least*

$$\Theta\left(\frac{|\Sigma_X| \cdot |\Sigma_Y| \cdot |\Sigma_Z|}{\varepsilon} \cdot \log \frac{|\Sigma_X| \cdot |\Sigma_Y| \cdot |\Sigma_Z|}{\delta} \cdot \log \frac{|\Sigma_X| \cdot |\Sigma_Y| \cdot |\Sigma_Z| \cdot \log\left(\frac{1}{\delta}\right)}{\varepsilon}\right),$$

the following results hold with probability $1 - \delta$:

1. *If $I(X; Y \mid Z) = 0$, then $I(\hat{X}; \hat{Y} \mid \hat{Z}) < \varepsilon$.*
2. *If $I(X; Y \mid Z) \geq \varepsilon$, then $I(\hat{X}; \hat{Y} \mid \hat{Z}) > c \cdot I(X; Y \mid Z)$.*

In our notation, we use $\widehat{I}(X; Y \mid Z)$ to mean the mutual information of the empirical distribution $I(\hat{X}; \hat{Y} \mid \hat{Z})$.

Corollary 4.4 (CMI tester)**.** *Fix any $\varepsilon > 0$. Let (X, Y, Z) be three random variables over $\Sigma_X, \Sigma_Y, \Sigma_Z$ respectively. Given the empirical distribution $(\widehat{X}, \widehat{Y}, \widehat{Z})$ over a size N sample of (X, Y, Z), there exists a universal constant $0 < c_0 < 1$ so that for any N at least*

$$\Theta\left(\frac{|\Sigma_X| \cdot |\Sigma_Y| \cdot |\Sigma_Z|}{\varepsilon} \cdot \log \frac{|\Sigma_X| \cdot |\Sigma_Y| \cdot |\Sigma_Z|}{\delta} \cdot \log \frac{|\Sigma_X| \cdot |\Sigma_Y| \cdot |\Sigma_Z| \cdot \log\left(\frac{1}{\delta}\right)}{\varepsilon}\right),$$

the following statements hold with probability $1 - \delta$*:*
(1) If $I(X; Y \mid Z) = 0$*, then* $\widehat{I}(X; Y \mid Z) < c_0 \cdot \varepsilon$*.*
(2) If $\widehat{I}(X; Y \mid Z) \leq c_0 \cdot \varepsilon$*, then* $I(X; Y \mid Z) < \varepsilon$*.*
Unconditional statements for $I(X; Y)$ *and* $\widehat{I}(X; Y)$ *hold similarly by setting* $|\mathbf{\Sigma}_Z| = 1$*.*

Proof. In the original proof of (1) in [BGP+23, Theorem 1.3], it is possible to change ε to $c_0 \cdot \varepsilon$ by paying a factor $1/c_0$ more in sample complexity, yielding our first statement.

Now, suppose $\widehat{I}(X; Y \mid Z) \leq c_0 \cdot \varepsilon$. Assume, for a contradiction, that $I(X; Y \mid Z) \geq \varepsilon$. Then, statement 2 of Theorem A.1 tells us that $\widehat{I}(X; Y \mid Z) > C \cdot I(X; Y \mid Z) \geq c_0 \cdot \varepsilon$. This contradicts the assumption that $\widehat{I}(X; Y \mid Z) \leq c_0 \cdot \varepsilon$. Therefore, we must have $I(X; Y \mid Z) < \varepsilon$. □

A.2.2 Algorithm analysis

The following identity (Lemma A.2) of mutual information and two properties about (conditional) mutual information on a polytree (Lemma A.3) which will be helpful in our proofs later.

Lemma A.2 (A useful identity)**.** *For any variable* $V \in \boldsymbol{V}$ *and sets* $\boldsymbol{A}, \boldsymbol{B} \subseteq \boldsymbol{V} \setminus \{V\}$*, we have*

$$I(V; \boldsymbol{A} \cup \boldsymbol{B}) = I(V; \boldsymbol{A}) + I(V; \boldsymbol{B}) + I(\boldsymbol{A}; \boldsymbol{B} \mid V) - I(\boldsymbol{A}; \boldsymbol{B}).$$

Proof. By the chain rule for mutual information, we can express $I(V, \boldsymbol{A}; \boldsymbol{B})$ in the following two ways:

1. $I(V, \boldsymbol{A}; \boldsymbol{B}) = I(V; \boldsymbol{B}) + I(\boldsymbol{A}; \boldsymbol{B} \mid V)$
2. $I(V, \boldsymbol{A}; \boldsymbol{B}) = I(\boldsymbol{A}; \boldsymbol{B}) + I(V; \boldsymbol{B} \mid \boldsymbol{A})$

So,

$$\begin{aligned} I(V; \boldsymbol{A} \cup \boldsymbol{B}) &= I(V; \boldsymbol{A}) + I(V; \boldsymbol{B} \mid \boldsymbol{A}) \\ &= I(V; \boldsymbol{A}) + I(V, \boldsymbol{A}; \boldsymbol{B}) - I(\boldsymbol{A}; \boldsymbol{B}) \\ &= I(V; \boldsymbol{A}) + I(V; \boldsymbol{B}) + I(\boldsymbol{A}; \boldsymbol{B} \mid V) - I(\boldsymbol{A}; \boldsymbol{B}) \end{aligned}$$ □

Lemma A.3. *Let* V *be an arbitrary vertex in a Bayesian polytree with parents* $\mathrm{Pa}(V)$*. Then, we have*

1. *For any disjoint subsets* $\boldsymbol{A}, \boldsymbol{B} \subseteq \mathrm{Pa}(V)$*,*

$$I(V; \boldsymbol{A} \cup \boldsymbol{B}) = I(V; \boldsymbol{A}) + I(V; \boldsymbol{B}) + I(\boldsymbol{A}; \boldsymbol{B} \mid V)$$

2. *For any subset* $\boldsymbol{A} \subseteq \mathrm{Pa}(V)$*,*

$$I(V; \boldsymbol{A}) \geq \sum_{U \in \boldsymbol{A}} I(V; U)$$

Proof. For the first equality, apply Lemma A.2 by observing that $I(\boldsymbol{A}; \boldsymbol{B}) = 0$ since $\boldsymbol{A}, \boldsymbol{B} \subseteq \mathrm{Pa}(V)$.

For the second inequality, apply the first equality $|\boldsymbol{A}|$ times with the observation that conditional mutual information is non-negative. Suppose $\boldsymbol{A} = \{A_1, \ldots, A_k\}$. Then,

$$\begin{aligned} I(V; \boldsymbol{A}) &= I(V; \{A_1\}) + I(V; \boldsymbol{A} \setminus \{A_1\}) + I(\{A_1\}; \boldsymbol{A} \setminus \{A_1\} \mid V) \\ &\geq I(V; \{A_1\}) + I(V; \boldsymbol{A} \setminus \{A_1\}) \\ &\ldots \\ &\geq \sum_{U \in \boldsymbol{A}} I(V; U) \end{aligned}$$ □

Lemma 4.7. *Any oriented arc in* $\widehat{\mathcal{G}} \setminus \mathcal{H}$ *is a ground truth orientation. That is, any vertex parent set in* $\widehat{\mathcal{G}} \setminus \mathcal{H}$ *is a subset of* $\mathrm{Pa}(V)$*, i.e.* $\mathrm{Pa}^{\mathrm{in}}(V) \subseteq \mathrm{Pa}(V)$*, and* $N^{\mathrm{in}}(V)$ *at any time during the algorithm will have* $N^{\mathrm{in}}(V) \subseteq \mathrm{Pa}^{\mathrm{in}}(V)$*.*

Proof. We consider the three cases in which we orient edges within the while loop:

1. Strong v-structures (in Phase 1)

2. Forced orientation due to local checks (in Phase 2)

3. Forced orientation due to Meek $R1(d)$ (in Phase 2)

Case 1: Strong v-structures Consider an arbitrary strong deg-d v-structure with center V. That is, there is a set $\boldsymbol{S}$ (all neighbors of V) with size $|\boldsymbol{S}| = d$, such that $\widehat{I}(U; \boldsymbol{S} \setminus \{U\} \mid V) \geq c_0 \cdot \varepsilon$ for any $U \in \boldsymbol{S}$. So, by Corollary 4.4, we know that $I(U; \boldsymbol{S} \setminus \{U\} \mid V) > 0$ for all $U \in \boldsymbol{S}$.

Consider an arbitrary vertex $U_0 \in \boldsymbol{S}$. Suppose, for a contradiction, that the ground truth orients *some* edge outwards from V, say $V \to U_0$ for some $U_0 \in \boldsymbol{S}$. This would imply that $I(U_0; \boldsymbol{S} \setminus \{U_0\} \mid V) = 0$. This contradicts the fact that we had $I(U_0; \boldsymbol{S} \setminus \{U_0\} \mid V) > 0$ for any $U \in \boldsymbol{S}$. Therefore, for all $U \in \boldsymbol{S}$, orienting $U \to V$ is a ground truth orientation.

Case 2: Forced orientation due to local checks Consider an arbitrary vertex V. Suppose it currently has incoming oriented arcs $N^{\mathrm{in}}(V)$ and we are checking for the orientation for an unoriented neighbor U of V. By induction, the existing incoming arcs to v are ground truth orientations.

If the ground truth orients $U \to V$, then $I(U; N^{\text{in}}(V)) = 0$ and we should have $\widehat{I}(U; N^{\text{in}}(V)) < c_0 \cdot \varepsilon \leq \varepsilon$ via Corollary 4.4. Hence, if we detect $\widehat{I}(N^{\text{in}}(V); U) > \varepsilon$, it must be the case that the ground truth orientation is $U \leftarrow V$, which is what we also orient.

Meanwhile, if the ground truth orients $U \leftarrow V$, then $I(U; N^{\text{in}}(V) \mid V) = 0$ and we should have $\widehat{I}(U; N^{\text{in}}(V) \mid V) \leq c_0 \cdot \varepsilon \leq \varepsilon$ via Corollary 4.4. Hence, if we detect $\widehat{I}(U; N^{\text{in}}(V) \mid V) > \varepsilon$, it must be the case that the ground truth orientation is $U \to V$, which is what we also orient.

Note that we may possibly detect both $\widehat{I}(U; N^{\text{in}}(V)) \leq \varepsilon$ and $\widehat{I}(U; N^{\text{in}}(V) \mid V) \leq \varepsilon$. In that case, we leave the edge $U - V$ unoriented.

Case 3: Forced orientation due to Meek $R1(d)$ Meek $R1(d)$ only triggers when there are d incoming arcs to a particular vertex. Since oriented arcs are inductively ground truth orientations and there are at most $d^* \leq d$ incoming arcs to any vertex, the forced orientations due to Meek $R1(d)$ will always be correct. □

Lemma 4.8. *Fix any vertex V, any $\boldsymbol{S} \subseteq \text{Pa}^{\text{in}}(V)$, and any $\boldsymbol{S}' \subseteq \text{Pa}^{\text{in}}(V)$. If $\boldsymbol{S} \neq \emptyset$, then there exists a vertex $U \in \boldsymbol{S} \cup \boldsymbol{S}'$ with*

$$I(V; \boldsymbol{S} \cup \boldsymbol{S}') \leq I(V; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\}) + I(V; U) + \varepsilon \,. \tag{4.2}$$

Proof. Since $\boldsymbol{S} \cup \boldsymbol{S}' \subseteq \text{Pa}(V)$, we see that $I(U; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\}) = 0$. Furthermore, since $\boldsymbol{S} \neq \emptyset$, Phase 1 guarantees that there exists a vertex $U \in \boldsymbol{S} \cup \boldsymbol{S}'$ such that $\widehat{I}(U; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\} \mid V) \leq c_0 \cdot \varepsilon$. To see why, we need to look at Line 4 of Algorithm 9 where we check all subsets $\boldsymbol{T}$ of $\text{Pa}(V)$ (as well as some other sets) to see if *every* $U \in \boldsymbol{T}$ satisfies $\widehat{I}(U; \boldsymbol{T} \setminus \{U\} \mid V) \geq c_0 \cdot \varepsilon$. From here, we can see that if a subset $\boldsymbol{T}$ of $\text{Pa}(V)$ is not *all* oriented into V, then we know that from Line 4 of Algorithm 9 that there exists some $U \in \boldsymbol{T}$ such that $\widehat{I}(U; \boldsymbol{T} \setminus \{U\} \mid V) < c_0 \cdot \varepsilon$. Applying this to $\boldsymbol{T} = \boldsymbol{S} \cup \boldsymbol{S}'$, where the set of unoriented neighboring nodes $\boldsymbol{S}$ is non-empty, we have our claim. As $\widehat{I}(U; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\} \mid V) \leq c_0 \cdot \varepsilon$, Corollary 4.4 tells us that $I(U; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\}) < \varepsilon$, we get

$$\begin{aligned}
&I(V; \boldsymbol{S} \cup \boldsymbol{S}') \\
&= I(V; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\}) + I(V; U) + I(U; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\} \mid V) - I(U; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\}) \\
&= I(V; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\}) + I(V; U) + I(U; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\} \mid V) \\
&\leq I(V; \boldsymbol{S} \cup \boldsymbol{S}' \setminus \{U\}) + I(V; U) + \varepsilon
\end{aligned}$$ □

Lemma 4.9. *For any vertex V with $\text{Pa}^{\text{in}}(V)$, we can show that*

$$I(V; \text{Pa}(V)) \leq \varepsilon \cdot |\text{Pa}(V)| + I(V; \text{Pa}^{\text{in}}(V)) + \sum_{U \in \text{Pa}^{\text{in}}(V)} I(V; U) \,.$$

Proof. Initializing $\boldsymbol{S}' = \mathrm{Pa}^{\mathrm{in}}(V)$ and $\boldsymbol{S} = \mathrm{Pa}(V) \setminus \mathrm{Pa}^{\mathrm{in}}(V) = \mathrm{Pa}^{\mathrm{un}}(V)$, we can repeatedly apply Lemma 4.8 to remove vertices one by one, until $\boldsymbol{S} = \emptyset$. Without loss of generality, by relabelling the vertices, we may assume that Lemma 4.8 removes U_1, then U_2, and so on. Let us denote the set of all removed vertices by $\boldsymbol{U}$ and note that some of the removed vertices may come from $\boldsymbol{S}' = \mathrm{Pa}^{\mathrm{in}}(V)$.

$$\begin{aligned}
I(V;\mathrm{Pa}(V)) &\leq I(V;\mathrm{Pa}(V) \setminus \{U_1\}) + I(V;U_1) + \varepsilon && \text{(By Lemma 4.8)}\\
&\leq I(V;\mathrm{Pa}(V) \setminus \{U_1,U_2\}) + I(V;U_1) + I(V;U_2) + 2\varepsilon && \text{(By Lemma 4.8)}\\
&\leq \dots\\
&\leq I(V;\mathrm{Pa}(V) \setminus \boldsymbol{U}) + \sum_{U \in \boldsymbol{U}} I(V;U) + \varepsilon \cdot |\boldsymbol{U}| && \text{(By Lemma 4.8)}
\end{aligned}$$

Since $I(\boldsymbol{A};\boldsymbol{B}) = 0$ for any $\boldsymbol{A} \sqcup \boldsymbol{B} \subseteq \mathrm{Pa}^{\mathrm{in}}(V)$, we have

$$\begin{aligned}
&I(V;\mathrm{Pa}^{\mathrm{in}}(V))\\
&= I(V;\mathrm{Pa}^{\mathrm{in}}(V) \setminus \boldsymbol{U}) + I(V;\mathrm{Pa}^{\mathrm{in}}(V) \cap \boldsymbol{U}) + I(\mathrm{Pa}^{\mathrm{in}}(V) \cap \boldsymbol{U};\mathrm{Pa}^{\mathrm{in}}(V) \setminus \boldsymbol{U} \mid V)\\
&&& \text{(By Lemma A.3)}\\
&\geq I(V;\mathrm{Pa}^{\mathrm{in}}(V) \setminus \boldsymbol{U}) + I(V;\mathrm{Pa}^{\mathrm{in}}(V) \cap \boldsymbol{U})\\
&\geq I(V;\mathrm{Pa}^{\mathrm{in}}(V) \setminus \boldsymbol{U}) + \sum_{u \in \mathrm{Pa}^{\mathrm{in}}(V) \cap \boldsymbol{U}} I(V;U) && \text{(By Lemma A.3)}
\end{aligned}$$

where the second last inequality is because $I(\mathrm{Pa}^{\mathrm{in}}(V);\mathrm{Pa}^{\mathrm{in}}(V) \cap \boldsymbol{U} \mid V) \geq 0$.

$$\begin{aligned}
I(V;\mathrm{Pa}(V)) &\leq I(V;\mathrm{Pa}^{\mathrm{in}}(V) \setminus \boldsymbol{U}) + \sum_{U \in \boldsymbol{U}} I(V;U) + \varepsilon \cdot |\boldsymbol{U}| && \text{(From above)}\\
&= I(V;\mathrm{Pa}^{\mathrm{in}}(V) \setminus \boldsymbol{U}) + \sum_{U \in \mathrm{Pa}^{\mathrm{in}}(V) \cap U} I(V;U) + \sum_{U \in \mathrm{Pa}^{\mathrm{un}}(V)} I(V;U) + \varepsilon \cdot |\boldsymbol{U}|\\
&&& \text{(Since } \mathrm{Pa}^{un}(V) \subseteq \boldsymbol{U}\text{)}\\
&\leq I(V;\mathrm{Pa}^{\mathrm{in}}(V)) + \sum_{U \in \mathrm{Pa}^{\mathrm{un}}(V)} I(V;U) + \varepsilon \cdot |\boldsymbol{U}| && \text{(From above)}\\
&\leq I(V;\mathrm{Pa}^{\mathrm{in}}(V)) + \sum_{U \in \mathrm{Pa}^{\mathrm{un}}(V)} I(V;U) + \varepsilon \cdot |\mathrm{Pa}(V)| && \text{(Since } \boldsymbol{U} \subseteq \mathrm{Pa}(V)\text{)}
\end{aligned}$$

□

Lemma 4.10. *Consider an arbitrary vertex V with $\mathrm{Pa}^{\mathrm{in}}(V)$ at the start of Phase 3. If Phase 3 orients $U \to V$ for some $U - V \in \mathcal{H}$, then*

$$I(V;\mathrm{Pa}^{\mathrm{in}}(V) \cup \{U\}) \geq I(V;\mathrm{Pa}^{\mathrm{in}}(V)) + I(V;U) - \varepsilon$$

Proof. Since $U - V \in E(\mathcal{H})$ remained unoriented, Phase 2 guarantees that $\widehat{I}(U; \mathrm{Pa}^{\mathrm{in}}(V) \mid V) \leq \varepsilon$ and $\widehat{I}(U; \mathrm{Pa}^{\mathrm{in}}(V)) \leq \varepsilon$. Since $0 < c_0 < 1$, we also see that $\widehat{I}(U; \mathrm{Pa}^{\mathrm{in}}(V) \mid V) \leq c_0 \cdot \varepsilon$ and $\widehat{I}(U; \mathrm{Pa}^{\mathrm{in}}(V)) \leq c_0 \cdot \varepsilon$ and so Corollary 4.4 tells us that $I(U; \mathrm{Pa}^{\mathrm{in}}(V) \mid V) \leq \varepsilon$ and $I(U; \mathrm{Pa}^{\mathrm{in}}(V)) \leq \varepsilon$. So,

$$\begin{aligned}
&|I(V; \mathrm{Pa}^{\mathrm{in}}(V) \cup U) - I(V; \mathrm{Pa}^{\mathrm{in}}(V)) - I(V; U)| \\
&= |I(U; \mathrm{Pa}^{\mathrm{in}}(V) \mid V) - I(U; \mathrm{Pa}^{\mathrm{in}}(V))| && \text{(By Lemma A.2)} \\
&= \max\{I(U; \mathrm{Pa}^{\mathrm{in}}(V) \mid V), I(U; \mathrm{Pa}^{\mathrm{in}}(V))\} \\
& && \text{(At most one of these term can be non-zero)} \\
&\leq \varepsilon
\end{aligned}$$

□

Lemma 4.11. *Let* $\mathrm{Pa}(V)$ *be the true parents of* v. *Let* $\widehat{\mathrm{Pa}}(V)$ *be the proposed parents of* v *output by our algorithm. Then,*

$$\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}(V)) - \sum_{V \in \boldsymbol{V}} I(V; \widehat{\mathrm{Pa}}(V)) \leq n \cdot (d^* + 1) \cdot \varepsilon \ .$$

Proof. We will argue that this summation is bounded by individually bounding each term in the summation. The main argument of the proof is that once we identified all the strong v-structures (and thus cancel out the scores of every strong v-structures), the rest should be roughly the score of a tree (up to additive ε error). Then, since we are guaranteed to be given $\mathrm{skel}(\mathcal{G}^*)$, the tree scores will match.

Let $\boldsymbol{A} \subseteq \boldsymbol{V}$ be the set of vertices which receive an additional incoming neighbor in the final phase, which we denote by $A_V \in \boldsymbol{V}$, i.e. $\widehat{\mathrm{Pa}}(V) = \mathrm{Pa}^{\mathrm{in}}(V) \cup \{A_V\}$. Note that the set of edges $\{A_V \to V\}_{V \in \boldsymbol{A}}$ is exactly the edges of the undirected graph $\mathcal{H}$ in the final phase. See Fig. A.1 for an illustration.

To lower bound $\sum_{V \in \boldsymbol{V}} I(V; \widehat{\mathrm{Pa}}(V))$, we can show

$$\begin{aligned}
&\sum_{V \in \boldsymbol{V}} I(V; \widehat{\mathrm{Pa}}(V)) \\
&= \sum_{V \in \boldsymbol{A}} I(V; \widehat{\mathrm{Pa}}(V)) + \sum_{V \in \boldsymbol{V} \setminus \boldsymbol{A}} I(V; \widehat{\mathrm{Pa}}(V)) \\
&\geq \sum_{V \in \boldsymbol{A}} \Big(I(V; \widehat{\mathrm{Pa}}(V) \setminus \{A_V\}) + I(V; A_V) - \varepsilon \Big) + \sum_{V \in \boldsymbol{V} \setminus \boldsymbol{A}} I(V; \widehat{\mathrm{Pa}}(V)) \\
& \qquad \text{(By Lemma 4.10)} \\
&= \sum_{V \in \boldsymbol{A}} I(V; \mathrm{Pa}^{\mathrm{in}}(V)) + \sum_{V \in \boldsymbol{A}} I(V; A_V) + \sum_{V \in \boldsymbol{V} \setminus \boldsymbol{A}} I(V; \mathrm{Pa}^{\mathrm{in}}(V)) - |\boldsymbol{A}| \cdot \varepsilon \\
&= \sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}^{\mathrm{in}}(V)) + \sum_{V \in \boldsymbol{A}} I(V; A_V) - |\boldsymbol{A}| \cdot \varepsilon \\
&\geq \sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}^{\mathrm{in}}(V)) + \sum_{V \in \boldsymbol{A}} I(V; A_V) - n\varepsilon && \text{(Since } \boldsymbol{A} \subseteq \boldsymbol{V} \text{ and } |\boldsymbol{V}| = n)
\end{aligned}$$

Meanwhile, to upper bound $\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}(V))$, we can show

$$\begin{aligned}
&\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}(V)) \\
&= \sum_{\substack{V \in \boldsymbol{V} \\ \mathrm{Pa}^{\mathrm{un}}(V) \neq \emptyset}} I(V; \mathrm{Pa}(V)) + \sum_{\substack{V \in \boldsymbol{V} \\ \mathrm{Pa}^{\mathrm{un}}(V) = \emptyset}} I(V; \mathrm{Pa}(V)) \\
&\leq \sum_{\substack{V \in \boldsymbol{V} \\ \mathrm{Pa}^{\mathrm{un}}(V) \neq \emptyset}} \left(\varepsilon \cdot |\mathrm{Pa}(V)| + I(V; \mathrm{Pa}^{\mathrm{in}}(V)) + \sum_{U \in \mathrm{Pa}^{\mathrm{un}}(V)} I(V; U) \right) \\
&\quad + \sum_{\substack{V \in \boldsymbol{V} \\ \mathrm{Pa}^{\mathrm{un}}(V) = \emptyset}} I(V; \mathrm{Pa}(V)) \qquad \text{(By Lemma 4.9)} \\
&= \sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}^{\mathrm{in}}(V)) + \sum_{\substack{V \in \boldsymbol{V} \\ \mathrm{Pa}^{\mathrm{un}}(V) \neq \emptyset}} \left(\varepsilon \cdot |\mathrm{Pa}(V)| + \sum_{U \in \mathrm{Pa}^{\mathrm{un}}(V)} I(V; U) \right)
\end{aligned}$$

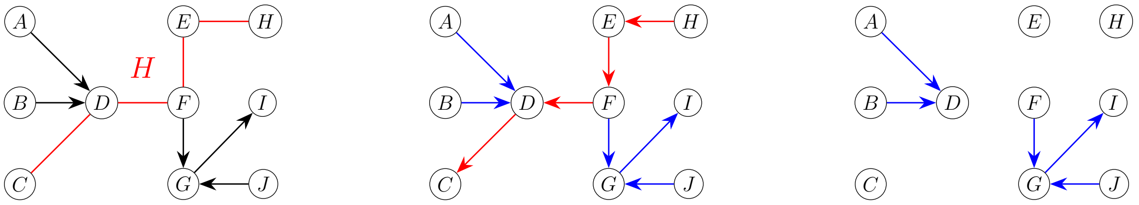
where the final equality is because $\mathrm{Pa}^{\mathrm{in}}(V) = \mathrm{Pa}(V)$ when $\mathrm{Pa}^{\mathrm{un}}(V) = \emptyset$. Since $|\mathrm{Pa}(V)| \leq d^*$ and $|\boldsymbol{V}| = n$, we get

$$\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}(V)) \leq nd^* \varepsilon + \sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}^{\mathrm{in}}(V)) + \sum_{\substack{V \in \boldsymbol{V} \\ \mathrm{Pa}^{\mathrm{in}}(V) \neq \emptyset}} \sum_{U \in \mathrm{Pa}^{\mathrm{un}}(V)} I(V; U)$$

Putting together, we get

$$\begin{aligned}
&\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}(v)) - \sum_{V \in \boldsymbol{V}} I(V; \widehat{\mathrm{Pa}}(V)) \\
&\leq \left(nd^* \varepsilon + \sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}^{\mathrm{in}}(V)) + \sum_{\substack{V \in \boldsymbol{V} \\ \mathrm{Pa}^{\mathrm{in}}(V) \neq \emptyset}} \sum_{U \in \mathrm{Pa}^{\mathrm{un}}(V)} I(V; U) \right) \qquad \text{(From above)} \\
&\quad - \left(\sum_{V \in \boldsymbol{V}} I(V; \mathrm{Pa}^{\mathrm{in}}(v)) + \sum_{V \in \boldsymbol{A}} I(V; A_V) - n\varepsilon \right) \\
&= n \cdot (d^* + 1) \cdot \varepsilon + \sum_{V \in \boldsymbol{V}} \sum_{U \in \mathrm{Pa}^{\mathrm{un}}(V)} I(V; U) - \sum_{V \in \boldsymbol{A}} I(V; A_V) \\
&= n \cdot (d^* + 1) \cdot \varepsilon
\end{aligned}$$

where the last equality is because the last two terms are two different ways to enumerate the edges of $\mathcal{H}$, e.g. see Fig. A.1. □



(a) Before final phase

(b) $\hat{G}$ under an arbitrary orientation of H

(c) $\hat{G} \setminus H$

Figure A.1: Illustration of notation used in proof of Lemma 4.11. Suppose (a) is the partial orientation of Fig. 4.1 after Phase 2, with H as the edge-induced subgraph on the unoriented edges in red. Before the final phase, we have $\mathrm{Pa}^{\mathrm{in}}(D) = \{A, B\}$, $\mathrm{Pa}^{\mathrm{in}}(G) = \{F, J\}$, $\mathrm{Pa}^{\mathrm{in}}(I) = \{G\}$, $\mathrm{Pa}^{\mathrm{un}}(C) = \{D\}$, $\mathrm{Pa}^{\mathrm{un}}(D) = \{C, F\}$, $\mathrm{Pa}^{\mathrm{un}}(F) = \{D, E\}$, $\mathrm{Pa}^{\mathrm{un}}(E) = \{H, F\}$, and $\mathrm{Pa}^{\mathrm{un}}(H) = \{E\}$. With respect to $\mathcal{H}$'s orientation in (b), we have $\boldsymbol{A} = \{C, D, F, E, H\}$, $A_C = D$, $A_D = F$, $A_F = E$, and $A_E = H$. Observe that the $\mathrm{Pa}^{\mathrm{un}}$s and $A_{\square}$s are two different ways to refer to the red edges and (b) only shows one possible orientation of $\mathcal{H}$ (see Fig. 4.3 for others).

A.2.3 Skeleton assumption

Lemma 4.13. *Under Assumption 4.12, running the Chow-Liu algorithm on the m-sample empirical estimates $\{\widehat{I}(U; V)\}_{U,V \in \boldsymbol{V}}$ recovers a ground truth skeleton with high probability when $m \geq \Omega(\frac{\log n}{\varepsilon_{\mathcal{P}}^2})$.*

Proof. Fix a graph $\mathcal{G}^*$. Recall that the Chow-Liu algorithm can be thought of as running maximum spanning tree with the edge weights as the estimated mutual information between any pair of vertices. With $m \geq \Omega(\log(n)/\varepsilon_{\mathcal{P}}^2)$ samples and Assumption 4.12, one can estimate $\widehat{I}(U; V)$ up to $(\varepsilon_{\mathcal{P}})/3$-closeness with high probability in n, i.e. $|I(U; V) - \widehat{I}(U; V)| \leq \varepsilon_{\mathcal{P}}/3$ for any pair of vertices $U, V \in \boldsymbol{V}$.

Now, consider two arbitrary vertices U and V that are *not* neighbors in $\mathcal{G}^*$.

Case 1 (**U and V belong in the same connected component in $\mathcal{G}^*$**): Let $\boldsymbol{P}_{U,V} = Z_0 - Z_1 - \ldots - Z_k - Z_{k+1}$ be the unique path between $U = Z_0$ and $V = Z_{k+1}$, where $k \geq 1$. Then,

$$\widehat{I}(U; V) - \varepsilon_{\mathcal{P}}/3 \leq I(U; V) \leq I(Z_i, Z_{i+1}) - \varepsilon_{\mathcal{P}} \leq \widehat{I}(Z_i, Z_{i+1}) - 2 \cdot \varepsilon_{\mathcal{P}}/3$$

for any $i \in \{1, \ldots, k\}$. Since $\widehat{I}(U; V) \leq \widehat{I}(Z_i, Z_{i+1}) - \varepsilon_{\mathcal{P}}/3$ for each $i \in \{1, \ldots, k\}$, the Chow-Liu algorithm will *not* add the edge $U - V$ in the output tree.

Case 2 (**U and V belong in the different connected components in $\mathcal{G}^*$**): Since U and V belong in the different connected components in $\mathcal{G}^*$, we have $I(U; V) = 0$. With m samples, for any two edge $A - B$ in $\mathcal{G}^*$, we have

$$\widehat{I}(U; V) \leq I(U; V) + \varepsilon_{\mathcal{P}}/3 = \varepsilon_{\mathcal{P}}/3 < 2 \cdot \varepsilon_{\mathcal{P}}/3 \leq I(A; B) - \varepsilon_{\mathcal{P}}/3 \leq \widehat{I}(A; B)$$

That is, the Chow-Liu algorithm will always consider edges crossing different components *after* all true edges have been considered. □

A.2.4 Proof of key lower bound lemma

We will use the following inequality in our proofs.

Lemma A.4. *For* $x > 0$*, we have* $\log_2(1+x) \geq \log_2(e) \cdot \left(x - \frac{x^2}{2}\right) = \log_2(e) \cdot x \cdot \left(1 - \frac{x}{2}\right)$.

Recall the lower bound distributions from Section 4.6, but we replace $\sqrt{\varepsilon}$ with α for notational convenience:

$$\mathcal{P}_1 : \begin{cases} X \sim \mathrm{Bern}(1/2) \\ Z = \begin{cases} X & \text{w.p. } 1/2 \\ \mathrm{Bern}(1/2) & \text{w.p. } 1/2 \end{cases} \\ Y = \begin{cases} Z & \text{w.p. } \alpha \\ \mathrm{Bern}(1/2) & \text{w.p. } 1-\alpha \end{cases} \end{cases} \qquad \mathcal{P}_2 : \begin{cases} X \sim \mathrm{Bern}(1/2) \\ Y \sim \mathrm{Bern}(1/2) \\ Z = \begin{cases} X & \text{w.p. } 1/2 \\ Y & \text{w.p. } \alpha \\ \mathrm{Bern}(1/2) & \text{w.p. } 1/2-\alpha \end{cases} \end{cases}$$

By construction, we have

$$\mathcal{P}_1(x,y,z) = \frac{1}{2} \cdot \left(\frac{1}{2} \cdot \frac{1}{2} + \frac{1}{2} \cdot \mathbb{1}_{x=z}\right) \cdot \left(\alpha \cdot \mathbb{1}_{y=z} + (1-\alpha) \cdot \frac{1}{2}\right)$$

and

$$\mathcal{P}_2(x,y,z) = \frac{1}{2} \cdot \frac{1}{2} \cdot \left(\frac{1}{2} \cdot \mathbb{1}_{x=z} + \alpha \cdot \mathbb{1}_{y=z} + \left(\frac{1}{2} - \alpha\right) \cdot \frac{1}{2}\right)$$

which corresponds to the probability tables given in Table A.1.

Lemma A.5. $d_H^2(\mathcal{P}_1, \mathcal{P}_2) \leq \alpha^2$

Proof. From Table A.1, we see that

$$\begin{aligned} &\sum_{(x,y,z)\in\{0,1\}^3} \sqrt{\mathcal{P}_1(x,y,z) \cdot \mathcal{P}_2(x,y,z)} \\ &= \frac{1}{8} \cdot \Big(\sqrt{3 \cdot (1+\alpha) \cdot (3+2\alpha)} + \sqrt{(1-\alpha) \cdot (1-2\alpha)} \\ &\qquad + \sqrt{3 \cdot (1-\alpha) \cdot (3-2\alpha)} + \sqrt{(1+\alpha) \cdot (1+2\alpha)}\Big) \end{aligned}$$

Considering the Taylor expansion of each of the four terms at $\alpha = 0$:

$$\sum_{(x,y,z)\in\{0,1\}^3} \sqrt{\mathcal{P}_1(x,y,z) \cdot \mathcal{P}_2(x,y,z)} \geq \frac{1}{8} \cdot \left(8 - \frac{\alpha^2}{3} - \mathcal{O}(\alpha^4)\right) \geq 1 - \frac{\alpha^2}{24} - \mathcal{O}(\alpha^4)$$

x	y	z	$\mathcal{P}_1(x,y,z)$	$\mathcal{P}_2(x,y,z)$
0	0	0	$\frac{3}{16}\cdot(1+\alpha)$	$\frac{1}{16}\cdot(3+2\alpha)$
0	0	1	$\frac{1}{16}\cdot(1-\alpha)$	$\frac{1}{16}\cdot(1-2\alpha)$
0	1	0	$\frac{3}{16}\cdot(1-\alpha)$	$\frac{1}{16}\cdot(3-2\alpha)$
0	1	1	$\frac{1}{16}\cdot(1+\alpha)$	$\frac{1}{16}\cdot(1+2\alpha)$
1	0	0	$\frac{1}{16}\cdot(1+\alpha)$	$\frac{1}{16}\cdot(1+2\alpha)$
1	0	1	$\frac{3}{16}\cdot(1-\alpha)$	$\frac{1}{16}\cdot(3-2\alpha)$
1	1	0	$\frac{1}{16}\cdot(1-\alpha)$	$\frac{1}{16}\cdot(1-2\alpha)$
1	1	1	$\frac{3}{16}\cdot(1+\alpha)$	$\frac{1}{16}\cdot(3+2\alpha)$

x	y	$\mathcal{P}_1(x,y)$
0	0	$\frac{1}{8}\cdot(2+\alpha)$
0	1	$\frac{1}{8}\cdot(2-\alpha)$
1	0	$\frac{1}{8}\cdot(2-\alpha)$
1	1	$\frac{1}{8}\cdot(2+\alpha)$

x	y	$\mathcal{P}_2(x,y \mid z=0)$	$\mathcal{P}_2(x,y \mid z=1)$
0	0	$\frac{1}{8}\cdot(3+2\alpha)$	$\frac{1}{8}\cdot(1-2\alpha)$
0	1	$\frac{1}{8}\cdot(3-2\alpha)$	$\frac{1}{8}\cdot(1+2\alpha)$
1	0	$\frac{1}{8}\cdot(1+2\alpha)$	$\frac{1}{8}\cdot(3-2\alpha)$
1	1	$\frac{1}{8}\cdot(1-2\alpha)$	$\frac{1}{8}\cdot(3+2\alpha)$

x	$\mathcal{P}_2(x \mid z=0)$	$\mathcal{P}_2(x \mid z=1)$
0	$\frac{3}{4}$	$\frac{1}{4}$
1	$\frac{1}{4}$	$\frac{3}{4}$

y	$\mathcal{P}_2(y \mid z=0)$	$\mathcal{P}_2(y \mid z=1)$
0	$\frac{1+\alpha}{2}$	$\frac{1-\alpha}{2}$
1	$\frac{1-\alpha}{2}$	$\frac{1+\alpha}{2}$

Table A.1: Explicit (conditional) probability tables for our lower bound construction.

Hence,

$$d_H^2(\mathcal{P}_1,\mathcal{P}_2) = 1 - \sum_{(x,y,z)\in\{0,1\}^3} \sqrt{\mathcal{P}_1(x,y,z)\cdot\mathcal{P}_2(x,y,z)} \le \frac{\alpha^2}{24} + \mathcal{O}(\alpha^4) \in \mathcal{O}(\alpha^2)$$

□

Lemma A.6. $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_1,\mathcal{P}_{1,\mathcal{G}_1}) = 0$ *and* $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_1,\mathcal{P}_{1,\mathcal{G}_2}) \in \Omega(\alpha^2)$

Proof. We have $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_1,\mathcal{P}_{1,\mathcal{G}_1}) = 0$ by definition of $\mathcal{P}_1$: Z depends on X and Y depends on Z. Observe that

$$\begin{aligned}
&\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_1,\mathcal{P}_{1,\mathcal{G}_2})\\
&= I(X;Z) + I(Z;Y) - I(Z;X,Y)\\
&= I(X;Z) + I(Z;Y) - \Big(I(Z;X) + I(Z;Y) + I(X;Y \mid Z) - I(X;Y)\Big)\\
&\qquad\qquad\qquad\qquad\text{(By applying Lemma A.2 with } v = Z,\ A = \{X\},\ B = \{Y\}\text{)}\\
&= I(X;Y) - I(X;Y \mid Z)\\
&= I(X;Y) \qquad\qquad\qquad\qquad\text{(Since } I(X;Y \mid Z) = 0 \text{ in } \mathcal{P}_1\text{)}
\end{aligned}$$

We will now show that $I(X;Y) \in \Omega(\alpha^2)$. From Table A.1, one can verify that

$\mathcal{P}_1(x=0) = \mathcal{P}_1(x=1) = \mathcal{P}_1(y=0) = \mathcal{P}_1(y=1) = 1/2$. So,

$$\begin{aligned}
I(X;Y) &= \sum_{(x,y)\in\{0,1\}^2} \mathcal{P}_1(x,y) \cdot \log \frac{\mathcal{P}_1(x,y)}{\mathcal{P}_1(x)\cdot\mathcal{P}_1(y)} \\
&= \frac{1}{4} \cdot \left((2+\alpha)\cdot \log\left(1+\frac{\alpha}{2}\right) + (2-\alpha)\cdot\log\left(1-\frac{\alpha}{2}\right)\right) && \text{(From Table A.1)} \\
&\geq \frac{1}{4}\cdot\log_2(e)\cdot\frac{\alpha}{2}\cdot\left((2+\alpha)\cdot\left(1-\frac{\alpha}{4}\right) - (2-\alpha)\cdot\left(1+\frac{\alpha}{4}\right)\right) \\
& && \text{(By Lemma A.4)} \\
&\in \Omega(\alpha^2)
\end{aligned}$$

□

Lemma A.7. $\mathrm{d_{KL}}(\mathcal{P}_2, P_{2,G_2}) = 0$ *and* $\mathrm{d_{KL}}(\mathcal{P}_2, P_{2,G_1}) \in \Omega(\alpha^2)$

Proof. We have $\mathrm{d_{KL}}(\mathcal{P}_2, P_{2,G_2}) = 0$ by definition of $\mathcal{P}_2$: Z depends on both X and Y.

Observe that

$$\begin{aligned}
&\mathrm{d_{KL}}(\mathcal{P}_2, P_{2,G_1}) \\
&= I(Z;X,Y) - I(X;Z) - I(Z;Y) \\
&= (I(Z;X) + I(Z;Y) + I(X;Y \mid Z) - I(X;Y)) - I(X;Z) - I(Z;Y) \\
& \qquad \text{(By applying Lemma A.2 with } v = Z,\ A = \{X\},\ B = \{Y\}) \\
&= I(X;Y \mid Z) - I(X;Y) \\
&= I(X;Y \mid Z) && \text{(Since } I(X;Y) = 0 \text{ in } \mathcal{P}_2)
\end{aligned}$$

We will now show that $I(X;Y \mid Z) \in \Omega(\alpha^2)$. By definition,

$$\begin{aligned}
I(X;Y \mid Z) &= \sum_{(x,y,z)\in\{0,1\}^3} \mathcal{P}_2(x,y \mid z)\cdot\log\left(\frac{\mathcal{P}_2(x,y\mid z)}{\mathcal{P}_2(x\mid z)\cdot\mathcal{P}_2(y\mid z)}\right) \\
&= I(X;Y \mid Z=0) + I(X;Y\mid Z=1)
\end{aligned}$$

From Table A.1, one can verify that $\mathcal{P}_2(z=0) = \mathcal{P}_2(z=1) = 1/2$ and $I(X;Y \mid Z=0) = I(X;Y \mid Z=1)$. So, it suffices to show that $I(X;Y\mid Z=0)\in\Omega(\alpha^2)$.

$$\begin{aligned}
&I(X;Y \mid Z=0) \\
&= \sum_{(x,y)\in\{0,1\}^2} \mathcal{P}_2(x,y \mid z=0)\cdot\log\left(\frac{\mathcal{P}_2(x,y\mid z=0)}{\mathcal{P}_2(x\mid z=0)\cdot\mathcal{P}_2(y\mid z=0)}\right) \\
&= \frac{3}{8}\cdot\log\left(\frac{3+2\alpha}{3+3\alpha}\cdot\frac{3-2\alpha}{3-3\alpha}\right) + \frac{1}{8}\cdot\log\left(\frac{1+2\alpha}{1+\alpha}\cdot\frac{1-2\alpha}{1-\alpha}\right) \\
&\qquad + \frac{\alpha}{4}\cdot\log\left(\frac{3+2\alpha}{3+3\alpha}\cdot\frac{3-3\alpha}{3-2\alpha}\cdot\frac{1+2\alpha}{1+\alpha}\cdot\frac{1-\alpha}{1-2\alpha}\right)
\end{aligned}$$

Using Taylor series expansion, one can verify that for $0 \leq \alpha \leq 1/2$,

- $\frac{3+2\alpha}{3+3\alpha} \cdot \frac{3-2\alpha}{3-3\alpha} \geq 1 + \frac{5}{9}\alpha^2$
- $\frac{1+2\alpha}{1+\alpha} \cdot \frac{1-2\alpha}{1-\alpha} \geq 1 - 3\alpha^2 - 4\alpha^4$
- $\frac{3+2\alpha}{3+3\alpha} \cdot \frac{3-3\alpha}{3-2\alpha} \cdot \frac{1+2\alpha}{1+\alpha} \cdot \frac{1-\alpha}{1-2\alpha} \geq 1 + \frac{4}{3}\alpha$

Thus, using Lemma A.4, we get

$$\begin{aligned}
&I(X; Y \mid Z = 0) \\
&= \frac{3}{8} \cdot \log\left(\frac{3+2\alpha}{3+3\alpha} \cdot \frac{3-2\alpha}{3-3\alpha}\right) + \frac{1}{8} \cdot \log\left(\frac{1+2\alpha}{1+\alpha} \cdot \frac{1-2\alpha}{1-\alpha}\right) \\
&\qquad + \frac{\alpha}{4} \cdot \log\left(\frac{3+2\alpha}{3+3\alpha} \cdot \frac{3-3\alpha}{3-2\alpha} \cdot \frac{1+2\alpha}{1+\alpha} \cdot \frac{1-\alpha}{1-2\alpha}\right) \\
&\geq \frac{3}{8} \cdot \log\left(1 + \frac{5}{9}\alpha^2\right) + \frac{1}{8} \cdot \log\left(1 - 3\alpha^2 - 4\alpha^4\right) + \frac{\alpha}{4} \cdot \log\left(1 + \frac{4}{3}\alpha\right) \\
&\geq \log_2(e) \cdot \left(\frac{3}{8} \cdot \left(\frac{5}{9}\alpha^2 - \left(\frac{5}{9}\alpha^2\right)^2\right)\right. \\
&\qquad \left. - \frac{1}{8} \cdot \left(3\alpha^2 + 4\alpha^4 + \left(3\alpha^2 + 4\alpha^4\right)^2\right) + \frac{\alpha}{4} \cdot \left(\frac{4}{3}\alpha - (\frac{4}{3}\alpha)^2\right)\right) \\
&\in \mathcal{O}(\alpha^2)
\end{aligned}$$

□

Lemma 4.14 (Key lower bound lemma). *Let $\mathcal{G}_1$ be $X \to Z \to Y$ and $\mathcal{G}_2$ be $X \to Z \leftarrow Y$, such that $\mathrm{skel}(\mathcal{G}_1) = \mathrm{skel}(\mathcal{G}_2)$ is $X - Z - Y$. With respect to Eq. (4.3), we have the following:*

1. $\mathrm{d}_{\mathrm{H}}^2(\mathcal{P}_1, \mathcal{P}_2) \in \mathcal{O}(\varepsilon)$
2. $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_1, \mathcal{P}_{1,\mathcal{G}_1}) = 0$ *and* $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_1, \mathcal{P}_{1,\mathcal{G}_2}) \in \Omega(\varepsilon)$
3. $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_2, \mathcal{P}_{2,\mathcal{G}_2}) = 0$ *and* $\mathrm{d}_{\mathrm{KL}}(\mathcal{P}_2, \mathcal{P}_{2,\mathcal{G}_1}) \in \Omega(\varepsilon)$

Proof. Combine Lemma A.5, Lemma A.6, and Lemma A.7 with α as $\sqrt{\varepsilon}$. □

Appendix B

Addendum for Part II

B.1 Addendum for Chapter 6

B.1.1 Further analysis of the standing windmill essential graph

In this section, we show that *all* DAGs in the standing windmill essential graph requires at least 3 and at most 4 atomic interventions.

By Theorem 6.12, we know that the optimal number of atomic interventions needed to verify any graph is the size of the minimum vertex cover of its oriented edges. To explore the space of DAGs in the essential graph, we will perform covered edge reversals (as justified by Lemma 2.49).

Consider the DAG $\mathcal{G}^*$ with MEC $[\mathcal{G}^*]$ and the standing windmill essential graph $\mathcal{E}(\mathcal{G}^*)$ in Fig. B.1. Starting from $\mathcal{G}^*$, if we fix the arc direction $H \to A$, then reversing any arc (possibly multiple times) from the set $\{B - C, D - E, F - G\}$ does *not* change the covered edge status of any edge (i.e. the covered edges remain exactly the same 4 edges) and thus the size of the minimum vertex cover remains unchanged. Meanwhile, reversing $A - H$ in $\mathcal{G}^*$ yields the graph $\mathcal{G}_1$. Fixing the arc direction $A \to H$, we observe that the three sets of edges $\{A - B, A - C, B - C\}$, $\{A - D, A - E, D - E\}$, and $\{A - F, A - G, F - G\}$ are symmetric. Furthermore, if we flip one of the edges from $\{A - B, A - D, A - F\}$ from $\mathcal{G}_1$ (or $\{A - C, A - D, A - F\}$ from $\mathcal{G}_4$), then all other two $A \to \cdot$ arcs are no longer covered edges. So, it suffices to study what happens when we only reverse arc directions in one of these sets: $\{A - B, A - C, B - C\}$, $\{A - D, A - E, D - E\}$, and $\{A - F, A - G, F - G\}$. The graphs $\mathcal{G}_1$ to $\mathcal{G}_6$ illustrate all possible cases when we fix $A \to H$ and only reverse edges in the set $\{A - B, A - C, B - C\}$. We see that $\nu_1(\mathcal{G}^*) = \nu_1(\mathcal{G}_1) = \nu_1(\mathcal{G}_4) = 4$ and $\nu_1(\mathcal{G}_2) = \nu_1(\mathcal{G}_3) = \nu_1(\mathcal{G}_5) = \nu_1(\mathcal{G}_6) = 3$. Thus, we can conclude that $\min_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G}) = 3$ and $\max_{\mathcal{G} \in [\mathcal{G}^*]} \nu_1(\mathcal{G}) = 4$.

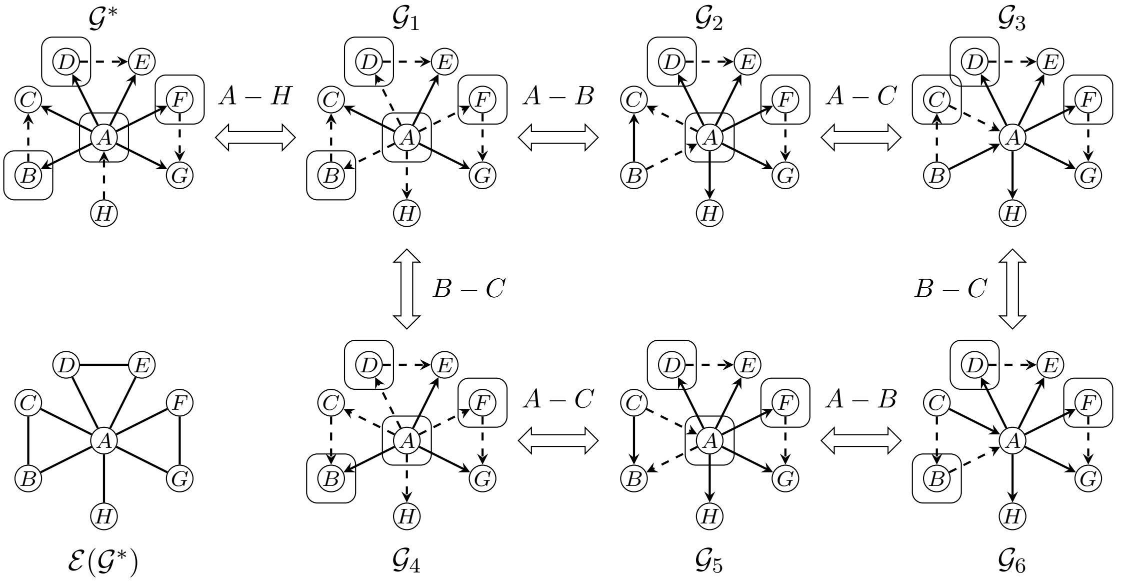


Figure B.1: A DAG $\mathcal{G}^*$ with its essential graph $\mathcal{E}(\mathcal{G}^*)$ and some of the graphs $\mathcal{G} \in [\mathcal{G}^*]$. In each DAG, dashed arcs are covered edges and the boxed vertices represent a minimum vertex cover.

B.1.2 Subset verification with atomic interventions

The following results tell us that Meek rules can only "propagate downstream", and will be useful for proving subsequent properties about the set of vertices $\boldsymbol{R}^{-1}$.

Lemma B.1. *Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a DAG. If $V \in \boldsymbol{R}^{-1}(\mathcal{G}, A \to B)$, then there exists a directed path from V to B in $\mathcal{G}$. That is, $V \in \mathrm{An}[B]$.*

Proof. If $V \in \{A, B\}$, then we trivially have $V \in \mathrm{An}[B]$. Otherwise, since $V \in \boldsymbol{R}^{-1}(\mathcal{G}, A \to B)$, there must be at least one new arc in the Meek rule (see Fig. 2.2) that fired to orient $A \to B$ due to V. Now suppose $V \not\in \{A, B\}$. Let us perform induction on the number of triggered Meek rules to orient $A \to B$.

Base case: Vertex V appears in all of Meek rules R1 to R4 that orients $A \to B$, and we see that $V \in \mathrm{An}[B]$ in all cases.

1. Meek R1: V can only be C and we have $C \to A \to B$
2. Meek R2: V can only be C and we have $C \to B$
3. Meek R3: This rule is not applicable as Meek R3 will trigger before any intervention is done, so it will not the reason why $V \in \boldsymbol{R}^{-1}(\mathcal{G}, A \to B)$.
4. Meek R4: V can either be C or D. In either case, we have $D \to C \to B$.

Inductive case: By induction, vertex $V \in \boldsymbol{R}^{-1}(X \to Y)$ for some $X \to Y$ at the start of the Meek rule configuration with $V \in \mathrm{An}[Y]$. Observe that for any oriented arc $X \to Y$ at the start of any Meek rule configuration that orients $A \to B$, we have $Y \in \mathrm{An}[B]$, and so we have $V \in \mathrm{An}[B]$ as well. □

Lemma 6.43. *If moral DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ is a single connected component, then the Hasse diagram $\mathcal{H}_{\mathcal{G}}$ is a directed tree with a unique root vertex.*

Proof. Since $\mathcal{G}$ is a single connected component, so is $\mathcal{H}_{\mathcal{G}}$. This is because reachability is preserved in Hasse diagrams. Suppose, for a contradiction, that $\mathcal{H}_{\mathcal{G}}$ does not have a unique root. Then, there exists two distinct directed paths $\boldsymbol{P}_1 = U \to \ldots \to U' \to X$ and $\boldsymbol{P}_2 = V \to \ldots \to V' \to X$ in $\mathcal{H}_{\mathcal{G}}$ with $U' \neq V'$ that end at some common vertex $X \in \boldsymbol{V}$. Notice that we must have $U' - V'$ in $\mathcal{G}$, otherwise $U' \to X \leftarrow V'$ is a v-structure in $\mathcal{G}$. W.l.o.g., suppose $U' \to V'$. But this means that $X \notin \mathrm{Ch}(U')$ and so we should not have an arc $U' \to X$ in the Hasse diagram $\mathcal{H}_{\mathcal{G}}$ in the first place. Contradiction. □

The proof of Theorem 6.45 relies on Lemma B.2, Lemma B.3, and Lemma B.4, which we prove first.

Lemma B.2. *Consider a moral DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$. For any two vertices $U, V \in \boldsymbol{V}$ such that $U \to V \in \boldsymbol{E}(\mathcal{G})$, we have $U \to W \in \boldsymbol{E}(\mathcal{G})$ for all $W \in \mathrm{De}(U) \cap \mathrm{An}(V)$.*

Proof. If $U \to V \in \boldsymbol{E}(\mathcal{G})$, then there exists a path $\boldsymbol{P}_{U \to V}$ in $\mathcal{H}_{\mathcal{G}}$. If $\boldsymbol{P}_{U \to V} = U \to V$ is a direct arc, then the claim is vacuously true since $\mathrm{De}(U) \cap \mathrm{An}(V) = \emptyset$. Now, suppose $\boldsymbol{P}_{U \to V} = U \to W_1 \to \ldots \to W_k \to V$ where $\mathrm{De}(U) \cap \mathrm{An}(V) = \{W_1, \ldots, W_k\}$. This implies that the arcs $U \to W_1 \to \ldots \to W_k \to V$ are all present in $\mathcal{G}$. Since $\mathcal{G}$ has no v-structures, it must be the case that the arc $U \to W_k$ exists (otherwise $U \to V \leftarrow W_k$ is a v-structure). Thus, by recursive argument from W_{k-1} up to W_1, there must be arcs $U \to W \in \boldsymbol{E}(\mathcal{G})$ for *any* $W \in \{W_1, \ldots, W_k\}$. □

Lemma B.3. *Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a moral DAG and $U \to V$ be an unoriented arc in $\mathcal{E}(\mathcal{G})$. Then, we have $W \in \boldsymbol{R}^{-1}(\mathcal{G}, U \to V)$ for any $W \in \mathrm{De}(U) \cap \mathrm{An}(V)$ in the Hasse diagram $\mathcal{H}_{\mathcal{G}}$.*

Proof. If $V \in \mathrm{Ch}(U)$, then the result is vacuously true since $\mathrm{De}(U) \cap \mathrm{An}(V) = \emptyset$. Suppose $\boldsymbol{P}_{U \to V} = U \to W_1 \to \ldots \to W_k \to V$ is the unique path from U to V in $\mathcal{H}_{\mathcal{G}}$, where $\mathrm{De}(U) \cap \mathrm{An}(V) = \{W_1, \ldots, W_k\}$. By Lemma B.2, we know that the arc $U \to W$ exists in $\mathcal{G}$ for any $W \in \mathrm{An}(V) \cap \mathrm{De}(U)$.

Suppose we intervened on an arbitrary $W_i \in \{W_1, \ldots W_k\}$, where $\mathrm{De}(W_i) \cap \mathrm{An}(V) = \{W_{i+1}, \ldots, W_k\}$. For any fixed arbitrary valid permutation π, define

$$\texttt{last}(\pi, W_i) = \underset{\substack{Z \in \mathrm{De}(W_i) \cap \mathrm{An}(V) \\ W_i \to Z \in \boldsymbol{E}}}{\operatorname{argmax}} \{\pi(Z)\}$$

as the "last" vertex in $\mathrm{De}(W_i) \cap \mathrm{An}(V)$ that W_i has a direct arc within $\mathcal{G}$.

If $\texttt{last}(\pi, W_i) = V$, then intervening on W_i yields $U \to W_i \to V - U$. So, Meek rule R2 will trigger to orient $U \to V$. Otherwise, if $\texttt{last}(\pi, W_i) \neq V$, then intervening on W_i will cause two sets of Meek rules to fire:

1. Meek rule R2 will orient the arcs $U \to Z$, for all $Z \in \mathrm{De}(W_i) \cap \mathrm{An}[\mathtt{last}(\pi, W_i)]$ since $U \to W_i \to Z - U$

2. Meek rule R1 will orient all outgoing arcs from $\mathtt{last}(\pi, W_i)$ towards vertices in $\mathrm{De}(W_i) \cap \mathrm{An}(V)$, since $W_i \not\to Z$ for all $Z \in \mathrm{De}(\mathtt{last}(\pi, W_i)) \cap \mathrm{De}(W_i) \cap \mathrm{An}(V)$, by maximality of $\mathtt{last}(\pi, W_i)$

Repeating the above argument by replacing the role of W_i by $\mathtt{last}(\pi, W_i)$, we see that the arc $W_{final} \to V$ will eventually be oriented by some $W_{final} \in \mathrm{De}(W_i) \cap \mathrm{An}(V)$, and so Meek rule R2 will orient $U \to V$. Intuitively, the direction $U \to V$ is forced in order to avoid a directed cycle since we will have $U \to W_i \to \mathtt{last}(\pi, W_i) \to \mathtt{last}(\mathtt{last}(\pi, W_i)) \ldots \to \mathtt{last}(\mathtt{last}(\ldots(\mathtt{last}(\pi, W_i)))) = W_{final} \to V - U$. □

Lemma B.4. *Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a moral DAG and $U \to V$ be an unoriented arc in $\mathcal{E}(\mathcal{G})$. For any vertex $W \in \boldsymbol{R}^{-1}(\mathcal{G}, U \to V)$, we have $Y \in \boldsymbol{R}^{-1}(\mathcal{G}, U \to V)$ for all $Y \in \mathrm{De}(W) \cap \mathrm{An}(U)$.*

Proof. Without loss of generality, we may assume $W \not\in \{U, V\}$ and $Y \in \mathrm{Ch}(W)$ is a direct child of W due to the following two observations:

1. If $W \in \{U, V\}$, then $\mathrm{De}(W) \cap \mathrm{An}(U) = \emptyset$ and the result is trivially true.

2. Suppose the chain of direct children from W to U is $W \to Y_1 \to Y_2 \to \ldots \to Y_k \to U$. To prove the result, it suffices to argue that $Y_1 \in \boldsymbol{R}^{-1}(\mathcal{G}, U \to V)$ and then apply induction to conclude that $Y_2 \in \boldsymbol{R}^{-1}(\mathcal{G}, U \to V)$, and so on.

Since $W \not\in \{U, V\}$ and $Y \in \mathrm{Ch}(W)$, we see that the arc $U \to V$ belongs in the set $\boldsymbol{R}(\mathcal{G}, W) \cap \boldsymbol{E}(\mathcal{T}_Y)$, where $\mathcal{T}_Y$ is the subtree rooted at Y in the Hasse diagram $\mathcal{H}_{\mathcal{G}}$ with $\boldsymbol{E}(\mathcal{T}_Y) = \{A \to B : A, B \in \mathrm{De}[Y]\}$. So, it suffices to show that $\boldsymbol{R}(\mathcal{G}, W) \cap \boldsymbol{E}(\mathcal{T}_Y) \subseteq \boldsymbol{R}(\mathcal{G}, Y)$. By Lemma B.1, intervening on W will only cause Meek rules to orient arcs of the form $A \to B$ where $W \in \mathrm{An}[B]$. So, we can partition the *newly recovered arcs* in $\boldsymbol{R}(\mathcal{G}, W)$ into three disjoint sets $\boldsymbol{R}^1(W)$, $\boldsymbol{R}^2(W)$, and $\boldsymbol{R}^3(W)$ as follows:

$$\boldsymbol{R}^1(W) = \{A \to B \in \boldsymbol{R}(\mathcal{G}, W) : W = A \vee W = B\} \qquad (W \text{ is an endpoint})$$

$$\boldsymbol{R}^2(W) = \{A \to B \in \boldsymbol{R}(\mathcal{G}, W) : A \in \mathrm{An}(W), B \in \mathrm{De}(W)\}$$
$$(W \text{ lies between endpoints})$$

$$\boldsymbol{R}^3(W) = \{A \to B \in \boldsymbol{R}(\mathcal{G}, W) : A, B \in \mathrm{De}(W)\} \qquad (W \text{ is ancestral to endpoints})$$

Clearly, $\boldsymbol{R}^1(W) \cap \boldsymbol{E}(\mathcal{T}_Y) = \emptyset$ since $W \not\in \boldsymbol{V}(\mathcal{T}_Y)$ and so $\boldsymbol{R}^1(W) \cap \boldsymbol{E}(\mathcal{T}_Y) \subseteq \boldsymbol{R}(\mathcal{G}, Y)$ trivially. Meanwhile, since $Y \in \mathrm{De}(W) \cap \mathrm{An}(U)$ implies that $Y \in \mathrm{De}(A) \cap \mathrm{An}(B)$ for any arc $A \to B \in \boldsymbol{R}^2(W) \cap \boldsymbol{E}(\mathcal{T}_Y)$, Lemma B.3 tells us that $A \to B \in \boldsymbol{R}(\mathcal{G}, Y)$ and so $\boldsymbol{R}^2(W) \cap \boldsymbol{E}(\mathcal{T}_Y) \subseteq \boldsymbol{R}(\mathcal{G}, Y)$. It remains to argue that $\boldsymbol{R}^3(W) \cap \boldsymbol{E}(\mathcal{T}_Y) \subseteq \boldsymbol{R}(\mathcal{G}, Y)$.

For any $A \to B \in \boldsymbol{R}^3(W)$, we see that $A, B \in \mathrm{De}[Y]$ since $A, B \in \mathrm{De}(W)$ and $Y \in \mathrm{Ch}(W)$. Furthermore, since $A = Y$ implies that $A \to B \in \boldsymbol{R}(\mathcal{G}, Y)$ trivially, we may further assume that $A, B \in \mathrm{De}(Y)$. Let $A \to B$ be the *first* arc within $\boldsymbol{R}^3(W) \cap \boldsymbol{E}(\mathcal{T}_Y)$ to be oriented when W is intervened upon. Since $A, B \in \mathrm{De}(W)$, it must be the case that $A \to B$ was oriented due to some Meek rule orientation (see Section 2.6.6). Observe that any oriented arc $A' \to B'$ appearing at the start of the Meek rule configuration to orient $A \to B$ has the property that $B' \in \mathrm{An}[B]$, so arcs in $\boldsymbol{R}^3(W) \setminus \boldsymbol{E}(\mathcal{T}_Y)$ *cannot* be the reason why $A \to B$ is oriented via Meek rules. Since $A \to B$ be the *first* arc within $\boldsymbol{R}^3(W) \cap \boldsymbol{E}(\mathcal{T}_Y)$ to be oriented, this means that arcs outside of where the oriented arcs appearing at the start of the orientation belong to either $\boldsymbol{R}^1(W)$ or $\boldsymbol{R}^2(W)$, i.e. the arc must begin with some vertex from $\mathrm{An}[W]$. We now check the four Meek rule configurations that could have oriented $A \to B$, where any oriented arc at the start of the Meek rule configuration begins with some vertex from $\mathrm{An}[W]$:

R1 The only oriented arc is $C \to A$. Since $C \in \mathrm{An}[W]$ and $A \in \mathrm{De}(Y)$, Lemma B.3 tells us that $C \to A \in \boldsymbol{R}(\mathcal{G}, Y)$ and so Meek R1 would trigger and orient $A \to B$ via $C \to A - B$ upon intervening on Y, i.e. $A \to B \in \boldsymbol{R}(\mathcal{G}, Y)$.

R2 There is an oriented arc $A \to C$ which means that $A \in [W]$, but this is not possible since we also have that $A \in \mathrm{De}(Y)$. So, this rule cannot be the reason why $A \to B \in \boldsymbol{R}^3(W)$.

R3 This rule is not applicable because Meek R3 will trigger before any intervention is done, so it will not be the reason why $A \to B \in \boldsymbol{R}^3(W)$.

R4 There is an oriented arc $D \to C$. Since $D \in \mathrm{An}[W]$ and $A \in \mathrm{De}(Y)$, Lemma B.3 tells us that $D \to A \in \boldsymbol{R}(\mathcal{G}, Y)$ and so Meek R1 would trigger and orient $A \to B$ via $D \to A - B$ upon intervening on Y, i.e. $A \to B \in \boldsymbol{R}(\mathcal{G}, Y)$.

In any case, we always see that $\boldsymbol{R}^3(W) \cap \boldsymbol{E}(\mathcal{T}_Y) \subseteq \boldsymbol{R}(\mathcal{G}, Y)$. □

We are now ready to prove Theorem 6.45.

Theorem 6.45. *Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a moral DAG and $U \to V$ be an unoriented arc in $\mathcal{E}(\mathcal{G})$. Then, $\boldsymbol{R}^{-1}(\mathcal{G}, U \to V) = \mathrm{De}[W] \cap \mathrm{An}[V]$ for some $W \in \mathrm{An}[U]$.*

Proof. We have $U, V \in \boldsymbol{R}^{-1}(\mathcal{G}, U \to V)$ trivially. By Lemma B.1, we also have $\mathrm{De}(V) \cap R^{-1}(\mathcal{G}, U \to V) = \emptyset$ and $\boldsymbol{R}^{-1}(\mathcal{G}, U \to V) \subseteq \mathrm{An}[V]$. For an arbitrary consistent topological ordering π, let

$$W = \operatorname*{argmin}_{Z \in \boldsymbol{R}^{-1}(\mathcal{G}, U \to V) \cap \mathrm{An}[U]} \{\pi(Z)\}$$

be the "furthest" ancestor vertex of U that orients $U \to V$. By minimality of W, we see that $\mathrm{An}(W) \cap \boldsymbol{R}^{-1}(\mathcal{G}, U \to V) = \emptyset$. Meanwhile, Lemma B.4 tells us that $\mathrm{De}(W) \cap \mathrm{An}(U) \subseteq \boldsymbol{R}^{-1}(\mathcal{G}, U \to V)$. Putting together, we get $\boldsymbol{R}^{-1}(\mathcal{G}, U \to V) = \mathrm{De}[W] \cap \mathrm{An}[V]$. □

Lemma 6.46. *If $\mathcal{G}$ be a moral DAG, then the covered edges of $\mathcal{G}$ are a subset of the Hasse edges in $\mathcal{H}_{\mathcal{G}}$.*

Proof. To prove this, we argue that any edge $A \to B \not\in \boldsymbol{E}(\mathcal{H}_{\mathcal{G}})$ *cannot* be a covered edge. Since all direct children arcs belong to $\boldsymbol{E}(\mathcal{H}_{\mathcal{G}})$ and $A \to B \not\in \boldsymbol{E}(\mathcal{H}_{\mathcal{G}})$, it must be the case that $B \not\in \mathrm{Ch}(A)$. So, there exists some $Z \in \mathrm{De}(A) \cap \mathrm{An}(B)$ such that $Z \to B$ but $Z \not\to A$. Thus, $A \to B$ *cannot* be a covered edge. □

Lemma 6.48. *Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be a connected moral DAG, $\mathcal{H}$ be the Hasse tree of $\mathcal{G}$, and $\boldsymbol{T} \subseteq \boldsymbol{E}$ be a subset of target edges. Then, there exists a set of intervals $\boldsymbol{J} \subseteq 2^{\boldsymbol{V} \times \boldsymbol{V}}$ on $\mathcal{H}$ such that any solution to minimum interval stabbing problem on $(\mathcal{H}, \boldsymbol{J})$ is a solution to the minimum sized atomic subset verification set $(\mathcal{G}, \boldsymbol{T})$.*

Proof. By Theorem 6.45, we know that each target edge $E \in \boldsymbol{T}$ has a corresponding interval $[A_E, B_E]_{\mathcal{H}}$ will be oriented if and only if some vertex in $[A_E, B_E]_{\mathcal{H}}$ is selected into the intervention set. Define $\boldsymbol{J} = \{[A_E, B_E]_{\mathcal{H}} : E \in \boldsymbol{T}\}$ as the collection of intervals corresponding to each edge $E \in \boldsymbol{T}$ of the target edges. Then, any solution to the interval stabbing problem on $(\mathcal{H}, \boldsymbol{J})$ ensures that every interval is stabbed, which translates to every edge in $\boldsymbol{T}$ being oriented via Theorem 6.45. Finally, to conclude, observe that the minimality of the interval stabbing solution corresponds to the minimality of the atomic verification set size. □

Lemma 6.49. *There exists a polynomial time algorithm for solving the interval stabbing problem on a rooted tree.*

Proof. See Theorem B.8. □

Lemma 6.50. *Let $\mathcal{H}$ be a rooted tree and $\boldsymbol{J} \subseteq 2^{\boldsymbol{V} \times \boldsymbol{V}}$ be a set of intervals on $\mathcal{H}$, for some set $\boldsymbol{V}$. Then, there exists a connected moral DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ and a subset $\boldsymbol{T} \subseteq \boldsymbol{E}$ of edges such that any solution to the minimum sized atomic subset verification set $(\mathcal{G}, \boldsymbol{T})$ is a solution to minimum interval stabbing problem on $(\mathcal{H}, \boldsymbol{J})$.*

Proof. Fix a consistent topological ordering π and consider the following construction:

1. Treat each interval $[U, V]_{\mathcal{H}}$ in $\boldsymbol{J}$ as an edge (U, V) where $\pi(U) < \pi(V)$
2. Define the set of arcs as $\boldsymbol{E} = \boldsymbol{A} \cup \boldsymbol{B}$, where

$$\begin{aligned} \boldsymbol{A} &= \{U \to V \in \boldsymbol{E}(\mathcal{H}) : \pi(U) < \pi(V)\} \\ \boldsymbol{B} &= \bigcup_{(U,V) \in \boldsymbol{J}} \{Z \to W : Z \in \mathrm{An}[V], W \in \mathrm{De}[U] \cap \mathrm{An}[V]\} \end{aligned}$$

3. Let $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ be the resulting DAG and let $\boldsymbol{T} = \{U \to V : (U, V) \in \boldsymbol{J}\}$.

Note that $\boldsymbol{J} \subseteq \boldsymbol{B}$ and $\mathcal{G}$ is a moral DAG and so its Hasse diagram $\mathcal{H}_{\mathcal{G}} = \mathcal{H}$.

To argue that the solution to the subset verification problem instance $(\mathcal{G}, \boldsymbol{T})$ is a solution to the interval stabbing on a tree instance $(\mathcal{H}, \boldsymbol{J})$, it suffices to show that $\boldsymbol{R}^{-1}(\mathcal{G}, U \to V) = \mathrm{De}[U] \cap \mathrm{An}[V]$ for each arc $U \to V \in \boldsymbol{T}$.

Consider an arbitrary $A \to B \in \boldsymbol{T}$. By Theorem 6.45, we know that $\boldsymbol{R}^{-1}(\mathcal{G}, A \to B) = \mathrm{De}[W] \cap \mathrm{An}[B]$ for some $W \in \mathrm{An}[A]$. It remains to argue that $A \to B \not\in \boldsymbol{R}(\mathcal{G}, W)$ for $W \in \mathrm{An}(A)$. Suppose, for contradiction, that $A \to B \in \boldsymbol{R}(\mathcal{G}, W)$ for some $W \in \mathrm{An}(A)$. Since $W \in \mathrm{An}(A)$, it must be the case that $A \to B$ was oriented due to some Meek rule orientation (see Section 2.6.6). We first argue that the configurations for Meek R1, R3, and R4 *cannot* be the reason why $A \to B \in \boldsymbol{R}(\mathcal{G}, W)$.

R1 This configuration will not occur by construction of $\boldsymbol{B}$ in $\mathcal{G}$ (since $A \to B \in \boldsymbol{J}$, $C \in \mathrm{An}[B]$, and $B \in \mathrm{An}[B]$ implies $C \to B \in \boldsymbol{B} \subseteq \boldsymbol{E}$), so this rule cannot be the reason why $A \to B \in \boldsymbol{R}(\mathcal{G}, W)$.

R3 This configuration will not occur since $\mathcal{G}$ is a moral DAG, so this rule cannot be the reason why $A \to B \in \boldsymbol{R}(\mathcal{G}, W)$.

R4 This configuration will not occur by construction of $\boldsymbol{B}$ in $\mathcal{G}$ (since $A \to B \in \boldsymbol{J}$, $D \in \mathrm{An}[B]$, and $B \in \mathrm{An}[B]$ implies $D \to B \in \boldsymbol{B} \subseteq \boldsymbol{E}$), so this rule cannot be the reason why $A \to B \in \boldsymbol{R}(\mathcal{G}, W)$.

Now, consider the Meek R2 configuration where $A \to C \to B - A$ was used to trigger the orientation of $A \to B$. Observe that $C \in \mathrm{De}(A) \cap \mathrm{An}(B)$. If $\mathrm{De}(A) \cap \mathrm{An}(B) = \emptyset$, this configuration cannot occur. Suppose $\mathrm{De}(A) \cap \mathrm{An}(B) \neq \emptyset$ and let Z be the *earliest* such vertex, i.e. $Z \in \mathrm{An}[Z']$ for any $Z' \in \mathrm{De}(A) \cap \mathrm{An}(B)$. Since $W \in \mathrm{An}(A)$, we know that $A \to Z$ must be oriented by some Meek rule. By choice of Z, Meek rule R2 cannot be the reason why $A \to Z \in R(\mathcal{G}, W)$. Meanwhile, $A \to B \in \boldsymbol{J}$ and $Z \in \mathrm{De}(A) \cap \mathrm{An}(B)$ imply that the other rule configurations do not apply by construction of $\boldsymbol{B}$ in $\mathcal{G}$. Therefore, $A \to Z$ will *not* be oriented, which contradicts the assumption that Meek R2 could be used to orient $A \to B \in \boldsymbol{R}(\mathcal{G}, W)$. □

B.1.3 Efficient dynamic programming implementation of recurrence

Recall the definitions and recurrence equations established in Section 6.8.1, we now explain how to solve Definition 6.47 in polynomial time via dynamic programming.

Definition 6.47 (Interval stabbing problem on a rooted tree)**.** Given a rooted tree $\widehat{\mathcal{G}} = (\boldsymbol{V}, \boldsymbol{E})$ with root $R \in \boldsymbol{V}$ and a set $\boldsymbol{J} \subseteq 2^{\boldsymbol{V} \times \boldsymbol{V}}$ of intervals of the form $[U, V]$, find a set $\boldsymbol{I} \subseteq \boldsymbol{V}$ of minimum size such that $\boldsymbol{I}$ stabs $[U, V]$ for all $[U, V] \in \boldsymbol{J}$.

Further, recall that we say that a vertex $Z \in \boldsymbol{V}$ *stabs* an interval $[U, V]$ if and only if $Z \in \mathrm{De}[U] \cap \mathrm{An}[V]$, and that a subset $\boldsymbol{S} \subseteq \boldsymbol{V}$ stabs $[A, B]$ if $\boldsymbol{S}$ has a vertex that stabs it.

Naively computing the recurrence relation of Eq. (6.3) will incur an exponential blow-up in state space. Instead, we will define an ordering $\prec$ on $\boldsymbol{J}$ so that our state space is over the indices of a sorted array instead of a subset of intervals (see Eq. (B.1)), so that we can implement the recurrence as a polynomial time dynamic programming (DP) problem.

Our $\prec$ ordering relies on the Euler tour data structure for rooted trees [TV84, HK99], which computes a sequence τ of vertices visited in a depth-first search (DFS) from the root. Using this sequence τ, we can obtain the first (f) and last (ℓ) times that a vertex is visited. More formally, we can define the mappings $\tau : \{1, \ldots, 2n-1\} \to \boldsymbol{V}$, $f : \boldsymbol{V} \to \{1, \ldots, 2n-1\}$, and $\ell : \boldsymbol{V} \to \{1, \ldots, 2n-1\}$. These mappings can be computed in linear time (via DFS) and $f(V) \leq \ell(V)$ with equality only if V is a leaf of the tree. See Fig. B.2 for an illustration of τ, f, and ℓ.

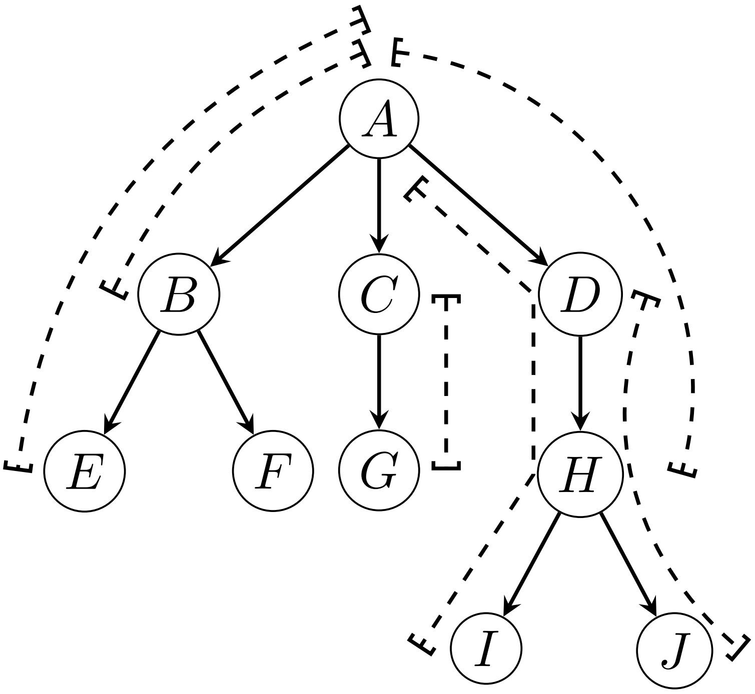


Figure B.2: Consider the rooted tree $\mathcal{G}$ on $n = 10$ vertices with intervals $\boldsymbol{J} = \{[A, B], [A, E], [A, H], [A, I], [C, G], [D, J]\}$. Recalling Eq. (6.3), we see that $\mathrm{opt}(\boldsymbol{J}, A) = 3$, where $\{A, C, D\}$ and $\{B, G, H\}$ are possible optimal sized interval covers. One possible Euler tour sequence τ is $(A, B, E, B, F, B, A, C, G, C, A, D, H, I, H, J, H, D, A)$ of length $|\tau| = 2n - 1 = 19$. Table B.1 shows the first (f) and last (ℓ) indices within τ. Observe that the leaves E, F, G, I, J have the same first and last indices, and vertices $D, H, I, J \in \boldsymbol{V}(\mathcal{T}_D)$ have indices between $f(D) = 12$ and $\ell(D) = 18$.

Using the Euler tour data structure, we can efficiently remove a subset of “unnecessary intervals” from $\boldsymbol{J}$, whose removal will not affect the optimality of the recurrence while granting us some additional structural properties which we will exploit. We call these “unnecessary intervals” *superset intervals*.

	A	B	C	D	E	F	G	H	I	J
f	1	2	8	12	3	5	9	13	14	16
ℓ	19	6	10	18	3	5	9	17	14	16

Table B.1: First (f) and last (ℓ) indices for Fig. B.2

Definition B.5 (Superset interval)**.** We say that an interval $[C, D] \in \boldsymbol{J}$ is a *superset interval* if there exists another interval $[A, B] \in \boldsymbol{J}$ such that $C \in \mathrm{An}[A]$ and $B \in \mathrm{An}[D]$.

In Fig. B.2, $[A, E]$ is a superset interval of $[A, B]$. Note that $A \in \mathrm{An}[B]$ in the definition of superset interval since $[A, B]$ is an interval. Observe that the removal of superset intervals will not affect the optimality of the solution because stabbing $[A, B]$ will stab $[C, D]$. For an interval $[A, B]$, we call A the *starting vertex* and B the *ending vertex* of the interval $[A, B]$ respectively. Using the Euler tour data structure, superset intervals can be removed in $\mathcal{O}(|\boldsymbol{J}| \log |\boldsymbol{J}|)$ time by first sorting the intervals according to the ending vertex, then only keep the intervals with the latest starting vertex amongst any pair of intervals that share the same ending vertex. After removing superset intervals, we are guaranteed that the ending vertices in $\boldsymbol{J}$ are unique.

We now define and sort according an ordering $\prec$ on $\boldsymbol{J}$ using the Euler tour mapping f so that $\boldsymbol{J}[i] \prec \boldsymbol{J}[j]$ for any $i < j$ in the following sense:

$$[A, B] \prec [C, D] \iff f(A) < f(C) \text{ or } (A = C \text{ and } \ell(B) > \ell(D)) \tag{B.1}$$

In Fig. B.2, we have $[A, H] \prec [A, I] \prec [A, B] \prec [A, E] \prec [C, G] \prec [D, J]$.

We write $\boldsymbol{J}^{-1}([A, B])$ to refer to the index of $[A, B]$ in $\boldsymbol{J}$. Since $\boldsymbol{I}_Y \subseteq \boldsymbol{I}_V$ for any $Y \in \mathrm{Ch}(V)$, we are guaranteed that $\min_{[A,B] \in \boldsymbol{I}_V} \boldsymbol{J}^{-1}([A, B]) \leq \min_{[A,B] \in \boldsymbol{I}_Y} \boldsymbol{J}^{-1}([A, B])$ for any $Y \in \mathrm{Ch}(V)$. However, note that there may be intervals *outside* of $\boldsymbol{I}_V$ with indices between $\min_{[A,B] \in \boldsymbol{I}_V} \boldsymbol{J}^{-1}([A, B])$ and $\max_{[A,B] \in \boldsymbol{I}_V} \boldsymbol{J}^{-1}([A, B])$. For any vertex $V \in \boldsymbol{V}$, we define

$$\boldsymbol{J}_V = \textit{sorted}\left(\{\boldsymbol{J}^{-1}([A, B]) \in \{1, \ldots, |\boldsymbol{J}|\} : [A, B] \in \boldsymbol{J} \cap \boldsymbol{I}_V\}\right)$$

as the array of indices of intervals in $\boldsymbol{I}_V$ such that $\boldsymbol{J}_V[i] \prec \boldsymbol{J}_V[j]$ for all $1 \leq i < j \leq |\boldsymbol{I}_V| = |\boldsymbol{J}_V|$. See Fig. B.3.

We begin with a simple lemma relating the first time a depth-first search visits a vertex and the ancestry of vertices.

Lemma B.6. *Consider arbitrary vertices $A, B, V \in \boldsymbol{V}$ in a rooted tree $\mathcal{G}$ with root R. If $A, B \in \mathrm{An}(V)$, then either $A \in \mathrm{An}[B]$ or $B \in \mathrm{An}[A]$. Furthermore, if $f(A) \leq f(B)$, then $A \in \mathrm{An}[B]$.*

Proof. Since $\mathcal{G}$ is a rooted tree, there is a unique path $\boldsymbol{P}$ from R to any vertex $V \in \boldsymbol{V}$

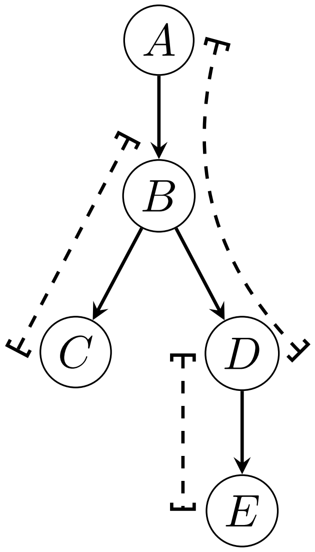


Figure B.3: Consider the rooted tree $\mathcal{G}$ on 5 vertices with intervals $\boldsymbol{J} = \{[A, D], [B, C], [D, E]\}$ and the Euler tour visits A, B, C, D, E in sequence. Then, $[A, D] \prec [B, C] \prec [D, E]$ and $\boldsymbol{J}^{-1}([A, D]) = 1$, $\boldsymbol{J}^{-1}([B, C]) = 2$, and $\boldsymbol{J}^{-1}([D, E]) = 3$. In this example, $\boldsymbol{I}_D = \{[A, D], [D, E]\}$ and $\boldsymbol{J}_D = [1, 3]$. Observe that $\min_i \boldsymbol{J}_D[i] < \boldsymbol{J}^{-1}([B, C]) < \max_i \boldsymbol{J}_D[i]$ despite $[B, C] \not\in \boldsymbol{I}_D$.

that involves all ancestors of V. Since $A, B \in \mathrm{An}(V)$, then either A appears before B in $\boldsymbol{P}$ (i.e. $A \in \mathrm{An}[B]$) or B appears before A in $\boldsymbol{P}$ (i.e. $B \in \mathrm{An}[A]$). By definition of depth-first search from R, if we visit A before B (i.e. $f(A) \leq f(B)$), then it must be the case that $A \in \mathrm{An}[B]$. □

The next lemma tells us that unstabbed intervals form a contiguous interval in $\boldsymbol{J}_V$ and that $\boldsymbol{E}_V$ appears first within $\boldsymbol{J}_V$.

Lemma B.7 (Properties of $\boldsymbol{J}$ w.r.t. $\prec$)**.** *Consider an arbitrary $V \in \boldsymbol{V}$ where $|\boldsymbol{I}_V| \geq 2$.*

- *For any $1 \leq i < j \leq |\boldsymbol{I}_V|$, if $\boldsymbol{J}_V[j] = [C, D]$ is stabbed by some $Z \in \mathrm{An}(V)$, then $\boldsymbol{J}_V[i] = [A, B]$ is also stabbed by Z.*
- *If $\boldsymbol{E}_V \neq \emptyset$ and $\boldsymbol{I}_V \setminus \boldsymbol{E}_V \neq \emptyset$, then*

$$\max_{[A,B] \in \boldsymbol{E}_V} \boldsymbol{J}^{-1}([A, B]) \leq \min_{[A,B] \in \boldsymbol{I}_V \setminus \boldsymbol{E}_V} \boldsymbol{J}^{-1}([A, B])$$

Proof. We prove each property one by one.

First property: Since $\boldsymbol{J}^{-1}([A, B]) = i < j = \boldsymbol{J}^{-1}([C, D])$, we have $[A, B] \prec [C, D]$. By Eq. (B.1), this means that either $f(A) < f(C)$ or ($A = C$ and $\ell(B) > \ell(D)$). Since $f(A) \leq f(C)$ alawys, we see that $A \in \mathrm{An}[C]$ by Lemma B.6. We also have $[A, B], [C, D] \in \boldsymbol{I}_V$ by definition of $\boldsymbol{J}_V$, so we have $B, D \in \boldsymbol{V}(\mathcal{T}_V)$. That is, $V \in \mathrm{An}[B]$ and $V \in \mathrm{An}[D]$, which implies that $Z \in \mathrm{An}[B] \cap \mathrm{An}[D]$ since $Z \in \mathrm{An}(V)$. So,

$$Z \in \mathrm{De}[C] \cap (\mathrm{An}[B] \cap \mathrm{An}[D]) \qquad \text{(Since } Z \text{ stabs } [C, D] \text{ and } Z \in \mathrm{An}[B] \cap \mathrm{An}[D])$$

$$
\begin{aligned}
&\subseteq \mathrm{De}[C] \cap \mathrm{An}[B] \\
&\subseteq \mathrm{De}[A] \cap \mathrm{An}[B] && \text{(Since } A \in \mathrm{An}[C]\text{)}
\end{aligned}
$$

In other words, Z stabs $[A, B]$ as well.

Second property: It suffices to argue that $[A, B] \prec [C, D]$ for any $[A, B] \in \boldsymbol{E}_V \subseteq \boldsymbol{I}_V$ and $[C, D] \in \boldsymbol{I}_V \setminus \boldsymbol{E}_V$. Since $[A, B], [C, D] \in \boldsymbol{I}_V$, we know that $B, D \in \boldsymbol{V}(\mathcal{T}_V)$, i.e. $V \in \mathrm{An}[B]$ and $V \in \mathrm{An}[D]$. Since $[A, B] \in \boldsymbol{E}_V$, we further have $B = V$, so $A \in \mathrm{An}(B) = \mathrm{An}(V)$ and $B = V \in \mathrm{An}[D]$.

We will now argue that $A \in \mathrm{An}(C)$. This is true for any $[C, D] \in \boldsymbol{S}_V \cup \boldsymbol{W}_V \subseteq \boldsymbol{I}_V \setminus \boldsymbol{E}_V$ because $V \in \mathrm{An}[C]$ in these cases, so it remains to consider $[C, D] \in \boldsymbol{M}_V \subseteq \boldsymbol{I}_V \setminus \boldsymbol{E}_V$. When $[C, D] \in \boldsymbol{M}_V$, we see that $A \in \mathrm{An}(V)$ and $C \in \mathrm{An}(V)$, so Lemma B.6 tells us that either $A \in \mathrm{An}[C]$ or $C \in \mathrm{An}[A]$. However, we cannot have $C \in \mathrm{An}[A]$ or $A = C$ because this will imply that $[C, D]$ is a superset interval with respect to $[A, B]$ since $C \in \mathrm{An}[A]$ and $B \in \mathrm{An}[D]$, but we have already removed all superset intervals. So, it must be the case that $A \in \mathrm{An}(C)$.

To conclude, observe that $A \in \mathrm{An}(C)$ implies $f(A) < f(C)$, therefore $[A, B] \prec [C, D]$ by Eq. (B.1). □

Algorithm 28 and Algorithm 29 describe our DP approach where we *always* recurse on subsets within $\boldsymbol{I}_V$, starting with $V = R$. For any vertex $V \in \boldsymbol{V}$, our DP state will recurse on the smallest index of the remaining unstabbed intervals within $\boldsymbol{I}_V$. If all intervals within $\boldsymbol{I}_V$ are stabbed, then the recursed index will be ∞ and the recursion terminates.

Algorithm 28 Minimum interval stab size on a rooted tree.

```
Input: Rooted tree $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ with root $R$ and a set of intervals $\boldsymbol{J}$.
Output: $\mathrm{opt}(\boldsymbol{J}, R) = \mathtt{DP}(R, 0)$
Compute Euler tour mappings $f$ and $\ell$, sort $\boldsymbol{J}$ according to $\prec$ ordering
Remove superset intervals from $\boldsymbol{J}$
Pre-compute indices $e_V$, $a_Y$ and $b_{V,Y}$ for all $V \in \boldsymbol{V}$ and $Y \in \mathrm{Ch}(V)$
                                        ▷ See Eq. (B.3), Eq. (B.4), and Eq. (B.5)
return $\mathtt{DP}(R, 0)$
```

Algorithm 29 Dynamic programming subroutine DP.

```
Input: Vertex $V \in \boldsymbol{V}$ and an index $i \in \{0, 1, \ldots, |\boldsymbol{J}| - 1\}$.
Output: $\mathtt{DP}(V, i)$
if $i = \infty$ then return 0                                   ▷ Done processing $\boldsymbol{J}$
$\alpha_V = 1 + \sum_{Y \in \mathrm{Ch}(V)} \mathtt{DP}(Y, \max\{a_Y, i\})$
$\beta_V = \sum_{Y \in \mathrm{Ch}(V)} \mathtt{DP}(Y, \max\{b_{V,Y}, i\})$
if $e_V \geq i$ then $\mathtt{memo}(V, i) \leftarrow \alpha_V$   ▷ $\boldsymbol{U} \cap \boldsymbol{E}_V \neq \emptyset$; see Eq. (B.2) for definition of $\boldsymbol{U}$
else $\mathtt{memo}(V, i) \leftarrow \min\{\alpha_V, \beta_V\}$
return $\mathtt{memo}(V, i)$
```

Recall the DP recurrence relation given in Eq. (6.3). When recursing on vertex $V \in \boldsymbol{V}$ and index i, we define the set $\boldsymbol{U}$ as follows:

$$\boldsymbol{U} = \{[A, B] \in \boldsymbol{I}_V : \boldsymbol{J}^{-1}([A, B]) \geq i\} \tag{B.2}$$

Note that $\boldsymbol{U} = \boldsymbol{J}$ initially when we start off the DP at the root $R \in \boldsymbol{V}$.

To determine whether $\boldsymbol{U} \cap \boldsymbol{E}_V$ is empty, we can define

$$e_V = \begin{cases} \max_{[A,B] \in \boldsymbol{E}_V} \boldsymbol{J}^{-1}([A, B]) & \text{if } \boldsymbol{E}_V \neq \emptyset \\ -\infty & \text{if } \boldsymbol{E}_V = \emptyset \end{cases} \tag{B.3}$$

and check whether $e_V \geq i$. This works because Lemma B.7 guarantees that $\boldsymbol{E}_V$ appears in the front of $\boldsymbol{J}_V$, so $e_V \geq i \iff \boldsymbol{U} \cap \boldsymbol{E}_V \neq \emptyset$. Meanwhile, the appropriate index update for $\boldsymbol{U} \cap \boldsymbol{B}_Y$ in the α_V case is $\max\{a_Y, i\}$ where

$$a_Y = \begin{cases} \min_{[A,B] \in \boldsymbol{B}_Y} \boldsymbol{J}^{-1}([A, B]) & \text{if } \boldsymbol{B}_Y \neq \emptyset \\ \infty & \text{if } \boldsymbol{B}_Y = \emptyset \end{cases} \tag{B.4}$$

Similarly, the index update $\boldsymbol{U} \cap \boldsymbol{I}_Y$ in the β_V case is $\max\{b_{V,Y}, i\}$ where

$$b_{V,Y} = \begin{cases} \min_{[A,B] \in \boldsymbol{I}_Y} \boldsymbol{J}^{-1}([A, B]) & \text{if } \boldsymbol{I}_Y \neq \emptyset \\ \infty & \text{if } \boldsymbol{I}_Y = \emptyset \end{cases} \tag{B.5}$$

One can verify that all the $e_V, a_Y, b_{V,Y}$ indices can be pre-computed in polynomial time before executing the DP. To extract a minimum sized stabbing set for $\boldsymbol{J}$ of size $\operatorname{opt}(\boldsymbol{J}, R)$, one can perform a standard backtracing of the memoization table.

Theorem B.8. *Together, Algorithm 28 and Algorithm 29 correctly output* $\operatorname{opt}(\boldsymbol{J}, R)$ *in* $\mathcal{O}(n^2 \cdot |\boldsymbol{J}|)$ *time.*

Proof. **Correctness** The indices $e_V, a_Y, b_{V,Y}$ are defined to match Eq. (6.3) and the correctness follows from Lemma 6.51 (see the next proof). The invariant we maintain throughout the recursion is as follows: $\boldsymbol{J}[i]$ has *not* been stabbed by $\operatorname{An}(V)$ whenever we are in a recursive step at some vertex $V \in \boldsymbol{V}$ and index i. We know from Lemma B.7 that any interval $[A, B]$ with $\boldsymbol{J}^{-1}([A, B]) < i$ would have been stabbed. So, recursing on $\max\{a_Y, i\}$ is equivalent to recursing on $\boldsymbol{U} \cap \boldsymbol{I}_Y$ and $\max\{b_{V,Y}, i\}$ is equivalent to recursing on $\boldsymbol{U} \cap \boldsymbol{B}_Y$ in Eq. (6.3), for any $Y \in \operatorname{Ch}(V)$. Since we immediately recurse on the α_V case whenever $\boldsymbol{U} \cap \boldsymbol{E}_V \neq \emptyset$, we avoid the ∞ case in Eq. (6.3).

Runtime The computation time of Euler tour data structure can be done in $\mathcal{O}(n)$ time via depth-first-search on the rooted tree. The removal of superset intervals can be done in $\mathcal{O}(|\boldsymbol{J}| \log |\boldsymbol{J}|)$ time. Sorting of $\boldsymbol{J}$ according to the $\prec$ ordering can be done in $\mathcal{O}(|\boldsymbol{J}| \log |\boldsymbol{J}|)$

time. For any $V \in \boldsymbol{V}$, the sets $\boldsymbol{E}_V, \boldsymbol{M}_V, \boldsymbol{S}_V, \boldsymbol{W}_V, \boldsymbol{I}_V, \boldsymbol{B}_V, \boldsymbol{C}_V$ can be computed in $\mathcal{O}(|\boldsymbol{J}|)$ time, then the indices E_V, A_Y and $B_{V,Y}$ can be computed in $\mathcal{O}(|\boldsymbol{J}| \log |\boldsymbol{J}|)$ time (we may need to sort to compute the minimum and maximum values). The DP has at most $\mathcal{O}(n \cdot |\boldsymbol{J}|)$ states and an execution of Algorithm 29 at vertex V takes $\mathcal{O}(|\mathrm{Ch}(V)|)$ time (accounting for memoization), so the Algorithm 29 takes $\mathcal{O}(n \cdot |\boldsymbol{J}| \cdot \sum_{V \in \boldsymbol{V}} |\mathrm{Ch}(V)|) \subseteq \mathcal{O}(n^2 \cdot |\boldsymbol{J}|)$ time. Putting everything together, we see that the overall runtime is $\mathcal{O}(|\boldsymbol{J}| \log |\boldsymbol{J}| + n^2 \cdot |\boldsymbol{J}|) \subseteq \mathcal{O}(n^2 \cdot |\boldsymbol{J}|)$ since $|\boldsymbol{J}| \leq \binom{n}{2} \leq n^2$. □

Lemma 6.51. *At least one of the following must hold for any optimal solution* opt *with size* $\mathrm{opt}(\boldsymbol{U}, R)$ *to the interval stabbing problem with respect to ordering* π *and any vertex* $V \in \boldsymbol{V}$ *with* $\boldsymbol{E}_V = \emptyset$*:*

1. *Either* $V \in \mathrm{opt}$ *or* opt *includes some ancestor of* V.

2. *For* $Y \in \mathrm{Ch}(V)$ *such that* $\boldsymbol{C}_V \cap \boldsymbol{I}_Y \neq \emptyset$*, we must have* $W_{V,Y} \in \mathrm{opt}$ *for some* $W_{V,Y} \in \mathrm{De}(V) \cap \mathrm{An}[B_{V,Y}]$*, where* $[A_{V,Y}, B_{V,Y}] = \operatorname*{argmin}_{[A,B] \in \boldsymbol{U} \cap \boldsymbol{C}_V \cap \boldsymbol{I}_Y} \{\pi(B)\}$.

Proof. Consider any arbitrary vertex $V \in \boldsymbol{V}$ and any child $Y \in \mathrm{Ch}(V)$ such that $\boldsymbol{C}_V \cap \boldsymbol{I}_Y \neq \emptyset$. For any $\boldsymbol{U} \subseteq \boldsymbol{J}$, define

$$L_{\boldsymbol{U},V,Y} = \operatorname*{argmin}_{[A,B] \in \boldsymbol{U} \cap \boldsymbol{C}_V \cap \boldsymbol{I}_Y} \{\pi(B)\} = [A_{V,Y}, B_{V,Y}]$$

as the earliest ending interval within $\boldsymbol{U}$ that is covered by V in subtree $\mathcal{T}_Y$.

Suppose $V \not\in \mathrm{opt}$ and opt does not include any ancestor of V. To stab any interval in $[A, B] \in \boldsymbol{C}_V$, we must have $W \in \mathrm{opt}$ for some $W \in \mathrm{De}(V) \cap \mathrm{An}[B]$. Since subtrees $\mathcal{T}_Y$ are disjoint, we can partition $\boldsymbol{C}_V$ into $\bigsqcup_{Y \in \mathrm{Ch}(V)} \boldsymbol{C}_{V,Y} = \bigsqcup_{Y \in \mathrm{Ch}(V)} \boldsymbol{C}_V \cap \boldsymbol{I}_Y$, where $\boldsymbol{C}_{V,Y}$ is associated to subtree $\mathcal{T}_Y$. So, for each interval $[A, B] \in \boldsymbol{C}_{V,Y}$, we need to ensure that $W \in \mathrm{opt}$ for some $W \in \mathrm{De}(V) \cap \mathrm{An}[B]$. By minimality of $B_{V,Y}$, stabbing $L_{\boldsymbol{U},V,Y}$ ensures that all intervals in $\boldsymbol{C}_{V,Y}$ are stabbed and any stabbing for $\boldsymbol{C}_{V,Y}$ must also stab $L_{\boldsymbol{U},V,Y}$. □

B.1.4 Subset verification with k-bounded interventions

In this section, we extend the result of Theorem 6.26 to the subset setting by following a similar proof strategy. One crucial difference in is that previously we rely on the bipartiteness of the covered edges of $\mathcal{G}^*$ while now we rely on Lemma B.9 to argue that there is a way to 2-color the atomic minimum subset verifying set.

Lemma B.9. *Let* $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ *be a moral DAG and* $\boldsymbol{S} \subseteq \boldsymbol{E}$*. Then, there exists a subset* $\boldsymbol{S}' \subseteq \boldsymbol{E}$ *computable in polynomial time such that* $\mathcal{G}[\boldsymbol{S}']$ *is a forest,* $\boldsymbol{R}(\mathcal{G}, \boldsymbol{S}) \subseteq \boldsymbol{R}(\mathcal{G}, \boldsymbol{S}')$*, and* $\bigcup_{(U,V) \in \boldsymbol{S}'} \{U, V\} \subseteq \bigcup_{(U,V) \in \boldsymbol{S}} \{U, V\}$.

Proof. If $\mathcal{G}[\boldsymbol{S}]$ is a forest, the claim trivially holds. Otherwise, we apply the following recursive argument to tranform $\boldsymbol{S}$: as long as $\mathcal{G}$ still contains an undirected cycle, we can update $\boldsymbol{S}$ to $\boldsymbol{S}'$ such that $\mathcal{G}[\boldsymbol{S}']$ has fewer cycles than $\mathcal{G}[\boldsymbol{S}]$ while still ensuring that $\boldsymbol{R}(\mathcal{G}, \boldsymbol{S}) \subseteq \boldsymbol{R}(\mathcal{G}, \boldsymbol{S}')$.

Let π be an arbitrary valid ordering for $\mathcal{G}$. Suppose $\mathcal{G}[\boldsymbol{S}]$ contains an undirected cycle $\boldsymbol{C} = R \to U_1 \to \ldots \to U_k \to S \leftarrow V_\ell \leftarrow \ldots \leftarrow V_1 \leftarrow R$ of length $|\boldsymbol{C}| = k+\ell+2 \geq 3$, where $R = U_0 = V_0 = \text{argmin}_{Z \in \boldsymbol{V}(\boldsymbol{C})}\{\pi(Z)\}$ and $S = \text{argmax}_{Z \in \boldsymbol{V}(\boldsymbol{C})}\{\pi(Z)\}$. We write $\boldsymbol{C} = R \to S \leftarrow V_\ell \leftarrow \ldots \leftarrow V_1 \leftarrow R$ and $\boldsymbol{C} = R \to U_1 \to \ldots \to U_k \to S \leftarrow R$ if $k = 0$ or $\ell = 0$ respectively.

Since $\mathcal{G}$ has no v-structures, we must have $V_\ell - U_k$ in $\mathcal{G}$. Without loss of generality, suppose $V_\ell \to U_k$. Then, we update $\boldsymbol{S}$ to $\boldsymbol{S}' = \boldsymbol{S} \cup \{V_\ell \to U_k\} \setminus \{V_\ell \to S\}$. Note that $V_\ell, S \in \boldsymbol{S}$, so the vertices of the endpoints in $\boldsymbol{S}'$ are a subset of $\boldsymbol{S}$. Observe that $\boldsymbol{R}(\mathcal{G}, \boldsymbol{S}) \subseteq \boldsymbol{R}(\mathcal{G}, \boldsymbol{S}')$ because Meek rule R2 will orient $V_\ell \to S$ via $V_\ell \to U_k \to S - V_\ell$. Furthermore, the cycle $\boldsymbol{C}$ is either destroyed (if $|\boldsymbol{C}| = 3$) or is shortened by one (if $|\boldsymbol{C}| > 3$). We can repeat this edge replacement argument until $\mathcal{G}[\boldsymbol{S}']$ has strictly one less undirected cycle than $\mathcal{G}[\boldsymbol{S}]$, and eventually until $\mathcal{G}[\boldsymbol{S}']$ has no undirected cycles, i.e. $\mathcal{G}[\boldsymbol{S}']$ is a forest.

It remains to argue that the recursive procedure described above runs in polynomial time. We first note that cycle finding can be done in polynomial time using depth-first search (DFS). Now, consider the potential function $\phi(\boldsymbol{S}) = \sum_{e=(U,V) \in \boldsymbol{S}} \pi(U) + \pi(V)$. In each round, $\phi(\boldsymbol{S})$ decreases since we replace $V_\ell \to U_k$ by $V_\ell \to S$ and $\pi(U_k) < \pi(S)$. Since the initial potential function value is polynomial in n, and we decrease it by at least 1 in each step, the entire procedure runs in polynomial time. □

By invoking Lemma B.9 with $\boldsymbol{S}$ as the incident arcs of the minimum size subset verification set $\mathcal{I}$, we can obtain a 2-coloring of $\mathcal{I}$ with respect to $\boldsymbol{S}'$. Thus, we can apply the "greedy grouping" generalization strategy as before to achieve the similar guarantees as Theorem 6.26, generalizing the results beyond $\boldsymbol{T} = \boldsymbol{E}$.

Theorem B.10. *If $\nu_1(\mathcal{G}, \boldsymbol{T}) = \ell$, then $\nu_k(\mathcal{G}, \boldsymbol{T}) \geq \lceil \ell/k \rceil$ and there exists a polynomial time algorithm to compute a bounded size intervention set $\mathcal{I}$ of size $|\mathcal{I}| \leq \lceil \frac{\ell}{k} \rceil + 1$.*

Proof. Consider any atomic subset verifying set $\mathcal{I}$ of $\mathcal{G}$ of size ℓ. Let $\boldsymbol{S}$ be the set of edges incident to vertices in $\mathcal{I}$. By Lemma B.9, there is a subset $\boldsymbol{S}' \subseteq \boldsymbol{E}$ such that $\mathcal{G}[\boldsymbol{S}']$ is a forest, $\boldsymbol{R}(\mathcal{G}, \mathcal{I}) = \boldsymbol{R}(\mathcal{G}, \boldsymbol{S}) \subseteq \boldsymbol{R}(\mathcal{G}, \boldsymbol{S}')$ and $\bigcup_{(U,V) \in \boldsymbol{S}'}\{U, V\} \subseteq \bigcup_{(U,V) \in \boldsymbol{S}}\{U, V\}$. Since $\mathcal{G}[\boldsymbol{S}']$ is a forest and $\boldsymbol{V}(\mathcal{G}[\boldsymbol{S}']) \subseteq \mathcal{I}$, there is a 2-coloring of the vertices in $\mathcal{I}$.

Split the vertices in $\mathcal{I}$ into partitions according to the 2-coloring. By construction, vertices belonging in the same partite will *not* be adjacent and thus choosing them together to be in an intervention $\boldsymbol{S}$ will *not* reduce the number of separated covered edges. Now, form interventions of size k by greedily picking vertices in $\mathcal{I}$ within the same partite. For the remaining unpicked vertices (strictly less than k of them), we form a new intervention with them. Repeat the same process for the other partite.

This greedy process forms groups of size k and at most 2 groups of sizes, one from each partite. Suppose that we formed z groups of size k in total and two "leftover groups" of sizes x and y, where $0 \leq x, y < k$. Then, $\ell = z \cdot k + x + y$, $\frac{\ell}{k} = z + \frac{x+y}{k}$, and we formed at most $z+2$ groups. If $0 \leq x + y < k$, then $\lceil \frac{\ell}{k} \rceil = z + 1$. Otherwise, if $k \leq x + y < 2k$, then $\lceil \frac{\ell}{k} \rceil = z + 2$. In either case, we use at most $\lceil \frac{\ell}{k} \rceil + 1$ interventions, each of size $\leq k$.

One can compute a bounded size intervention set efficiently because the following procedures can all be run in polynomial time: (i) Lemma B.9 runs in polynomial time; (ii) 2-coloring a tree; (iii) greedily grouping vertices into sizes $\leq k$. □

B.2 Addendum for Chapter 7

B.2.1 Covariate adjustment in the potential outcomes framework

For simplicity, we describe the potential outcomes framework in the i.i.d. setting, i.e., we assume that n samples are drawn independently from a distribution $\mathcal{P}(\boldsymbol{V})$, though we note that the following result can be extended to weaker settings (e.g. when samples are exchangeable but not necessarily independent).

In the PO framework, the treatment variables $\boldsymbol{X}$ are considered to be given, along with a set $\Sigma_{\boldsymbol{X}}$ of possible values for $\boldsymbol{X}$. Given $\boldsymbol{X}$ and $\Sigma_{\boldsymbol{X}}$, one takes as their starting point an indexed set of random variables $\{\boldsymbol{Y}(\boldsymbol{x})\}_{\boldsymbol{x} \in \Sigma_{\boldsymbol{X}}}$, with $\boldsymbol{Y}(\boldsymbol{x})$ denoting the potential outcome associated with intervening to set $\boldsymbol{X}$ equal to $\boldsymbol{x}$. Then, the *factual* outcome $\boldsymbol{Y}$ is generated according to $\boldsymbol{X}$ and the potential outcomes; typically, one assumes *consistency*, i.e., that if $\boldsymbol{X} = \boldsymbol{x}$, then $\boldsymbol{Y} = \boldsymbol{Y}(\boldsymbol{x})$. Hence, under the PO framework, we have $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y}) = \mathcal{P}(\boldsymbol{Y}(\boldsymbol{x}) = \boldsymbol{y})$ is the probability that $\boldsymbol{Y}$ takes on value $\boldsymbol{y}$ if $\boldsymbol{X}$ is set to $\boldsymbol{x}$.

Now, Eq. (7.1) can be derived as a consequence of consistency and an additional assumption about conditional independences. In particular, $\boldsymbol{X}$ is called *conditionally ignorable* with respect to $\boldsymbol{Z}$ if $\boldsymbol{Y}(\boldsymbol{x}) \perp\!\!\!\perp \boldsymbol{X} \mid \boldsymbol{Z}$ for all $\boldsymbol{x} \in \Sigma_{\boldsymbol{X}}$.

Lemma B.11. *Under consistency and conditional ignorability of $\boldsymbol{X}$ with respect to $\boldsymbol{Z}$, we have*

$$\mathcal{P}(\boldsymbol{Y}(\boldsymbol{x}) = \boldsymbol{y}) = \sum_{\boldsymbol{z} \in \boldsymbol{Z}} \mathcal{P}(\boldsymbol{Y} = \boldsymbol{y} \mid \boldsymbol{Z} = \boldsymbol{z}, \boldsymbol{X} = \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{Z} = \boldsymbol{z})$$

Proof.

$$\begin{aligned}
\mathcal{P}(\boldsymbol{Y}(\boldsymbol{x}) = y) &= \sum_{\boldsymbol{z} \in \boldsymbol{Z}} \mathcal{P}(\boldsymbol{Y}(\boldsymbol{x}) = \boldsymbol{y} \mid \boldsymbol{Z} = \boldsymbol{z}) \cdot \mathcal{P}(\boldsymbol{Z} = \boldsymbol{z}) && \text{(Law of total probability)} \\
&= \sum_{\boldsymbol{z} \in \boldsymbol{Z}} \mathcal{P}(\boldsymbol{Y}(\boldsymbol{x}) = \boldsymbol{y} \mid \boldsymbol{Z} = \boldsymbol{z}, \boldsymbol{X} = \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{Z} = \boldsymbol{z}) \\
& && \text{(Since } \boldsymbol{Y}(\boldsymbol{x}) \perp\!\!\!\perp \boldsymbol{X} \mid \boldsymbol{Z}\text{)}
\end{aligned}$$

$$= \sum_{\boldsymbol{z} \in \boldsymbol{Z}} \mathcal{P}(\boldsymbol{Y} = \boldsymbol{y} \mid \boldsymbol{Z} = \boldsymbol{z}, \boldsymbol{X} = \boldsymbol{x}) \cdot \mathcal{P}(\boldsymbol{Z} = \boldsymbol{z}) \qquad \text{(By consistency)}$$

□

B.2.2 Derivation of expectation bound

Here, we translate the result of [ZBHK24] into our language, showing that $\mathcal{O}(\frac{|\boldsymbol{\Sigma_Z}|}{\lambda \alpha_{\boldsymbol{Z}}} + \frac{1}{\lambda^2 \alpha_{\boldsymbol{Z}}})$ samples suffice to obtain an expectation bound of $\mathbb{E}(|T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}|) \leq \lambda$, for $T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$ defined as in Eq. (7.1).

[ZBHK24] studies the setting where one is given n i.i.d. copies of (Y, X, A) where $Y \in \{0, 1\}$ is the binary outcome, $X \in \{0, 1\}$ is the binary treatment, and $A \in [d] = \{1, \ldots, d\}$ is a multivariate covariate. Under their positivity assumption [ZBHK24, Assumption 2], $\mathcal{P}(X = 1 \mid A = k) \in [\varepsilon, 1 - \varepsilon]$ holds for some constant $\varepsilon \in (0, 1/2)$ and any $k \in [d]$. Then, for $\psi_1 = \sum_{k=1}^{d} \mathcal{P}(A = k) \cdot \mathcal{P}(Y = 1 \mid X = 1, A = k)$ and plug-in estimator $\widehat{\psi_1}$, Theorem 1 of [ZBHK24] states that $\mathbb{E}[\psi_1 - \widehat{\psi_1}] \leq \frac{|\boldsymbol{\Sigma_Z}|^2}{\alpha_{\boldsymbol{Z}}^2 n^2} + \frac{C}{\alpha_{\boldsymbol{Z}} n}$ when $\widehat{\psi_1}$ is computed using n i.i.d. samples from $\mathcal{P}(Y, X, A)$, for the worst case distribution $\mathcal{P}(Y, X, A)$ satisfying their positivity assumption.

To adapt their result to our setting, let us define $Y' = \mathbb{1}_{\boldsymbol{Y}=\boldsymbol{y}}$, $X' = \mathbb{1}_{\boldsymbol{X}=\boldsymbol{x}}$, and A' as a flattened version of $\boldsymbol{Z}$. Relating (Y', X', A') to their (Y, X, A) setup, we see that $\psi_1 = T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}$, $d = |\boldsymbol{\Sigma_Z}|$, and $\alpha_{\boldsymbol{Z}} = \varepsilon$. So,

$$\mathbb{E}\left[\left(T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}\right)^2\right] \leq \frac{|\boldsymbol{\Sigma_Z}|^2}{\alpha_{\boldsymbol{Z}}^2 n^2} + \frac{C}{\alpha_{\boldsymbol{Z}} n}, \tag{B.6}$$

for some absolute constant $C > 0$, where we have replaced ε by $\alpha_{\boldsymbol{Z}}$, d by $|\boldsymbol{\Sigma_Z}|$, and used that $(1 - \alpha_{\boldsymbol{Z}})^2 \leq 1$.

To translate this bound into our desired form, we first apply Jensen's inequality [Jen06]:

$$\begin{aligned}
\left(\mathbb{E}\left[\left|T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}\right|\right]\right)^2 &\leq \mathbb{E}\left[\left(\left|T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}\right|\right)^2\right] && \text{(Jensen's inequality)} \\
&= \mathbb{E}\left[\left(T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}\right)^2\right] \\
&\leq 2 \max\left(\frac{|\boldsymbol{\Sigma_Z}|^2}{\alpha_{\boldsymbol{Z}}^2 n^2}, \frac{C}{\alpha_{\boldsymbol{Z}} n}\right) && \text{(By Eq. (B.6))}
\end{aligned}$$

Thus, to obtain that $\mathbb{E}\left(\left|T_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}} - \widehat{T}_{\boldsymbol{Z},\boldsymbol{x},\boldsymbol{y}}\right|\right) \leq \lambda$, it suffices to have $2 \max\left(\frac{|\boldsymbol{\Sigma_Z}|^2}{\alpha_{\boldsymbol{Z}}^2 n^2}, \frac{C}{\alpha_{\boldsymbol{Z}} n}\right) \leq \lambda^2$. Then, solving for n yields $n \in \mathcal{O}\left(\frac{|\boldsymbol{\Sigma_Z}|}{\lambda \alpha_{\boldsymbol{Z}}} + \frac{1}{\lambda^2 \alpha_{\boldsymbol{Z}}}\right)$ as stated.

B.2.3 Completeness of Bamba for a special case

In Section 8.1, we described a special case of our setting in the graphical framework. In particular, assuming that $\mathcal{G}$ is a DAG and considering only a single treatment variable X, it is easy to see that $\boldsymbol{Z} = \mathrm{ND}(X)$ is a valid adjustment set, and that $\boldsymbol{S} = \mathrm{Pa}(X)$ is a Markov blanket of X with respect to $\boldsymbol{Z}$.

Here, we show that Bamba is not just sound (Lemma 7.9) but also complete in a special setting, i.e. the two conditional independences in Lemma 7.9 are not just sufficient to ensure that $\boldsymbol{S}'$ is an adjustment set, they are also *necessary*. In particular, Lemma B.12 implies that searching for a minimal sized $\boldsymbol{S}' \subseteq \boldsymbol{Z}$ that satisfies both $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'$ and $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}$ necessarily produces a minimal sized adjustment set.

Lemma B.12. *Consider the graphical causal framework in the causally sufficient setting, where variables in $\boldsymbol{X}$ are non-ancestors of each other. Let $\boldsymbol{Z} = \mathrm{ND}(\boldsymbol{X}) \subseteq \boldsymbol{V} \setminus (\boldsymbol{X} \cup \boldsymbol{Y})$ be the set of non-descendants of $\boldsymbol{X}$ and $\boldsymbol{S} = \mathrm{Pa}(\boldsymbol{X}) = \bigcup_{X \in \boldsymbol{X}} \mathrm{Pa}(X) \subseteq \mathrm{ND}(\boldsymbol{X}) = \boldsymbol{Z}$ are the parents of $\boldsymbol{X}$. Then, any subset $\boldsymbol{S}' \subseteq \mathrm{ND}(\boldsymbol{X}) = \boldsymbol{Z}$ such that $T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} = T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$ must satisfy both (i) $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'$ and (ii) $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}$.*

Proof. We know that $\boldsymbol{S} = \mathrm{Pa}(\boldsymbol{X})$ is a valid adjustment set and so it must block any non-causal paths between $\boldsymbol{X}$ and $\boldsymbol{Y}$ [PTKM18]. Then, since $T_{\boldsymbol{S}',\boldsymbol{x},\boldsymbol{y}} = T_{\boldsymbol{S},\boldsymbol{x},\boldsymbol{y}}$, it must be the case that $\boldsymbol{S}'$ is also a valid adjustment set.

Condition (i) : $\boldsymbol{Y} \perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'$

Suppose, for a contradiction, that $\boldsymbol{Y} \not\perp\!\!\!\perp \boldsymbol{S} \setminus \boldsymbol{S}' \mid \boldsymbol{X} \cup \boldsymbol{S}'$. By contrapositive of the Markov property (Definition 2.46), there is an active d-connected path in $\mathcal{G}$ from some $Y \in \boldsymbol{Y}$ to some $A \in \boldsymbol{S} \setminus \boldsymbol{S}'$, when $\boldsymbol{X} \cup \boldsymbol{S}'$ is conditioned upon. Let $\boldsymbol{P}_{Y,A}$ denote such an active path of minimal length. By minimality of $\boldsymbol{P}_{Y,A}$, there are no internal vertices from $\boldsymbol{S} \setminus \boldsymbol{S}'$ within the path $\boldsymbol{P}_{Y,A}$. We will argue that such a path $\boldsymbol{P}_{Y,A}$ *cannot* exist by considering the two cases of whether the path $\boldsymbol{P}_{Y,A}$ contains some vertex from $\boldsymbol{X}$ internally.

Case 1: Suppose $\boldsymbol{P}_{Y,A}$ contains some vertex from $\boldsymbol{X}$, i.e. $\boldsymbol{V}(\boldsymbol{P}_{Y,A}) \cap \boldsymbol{X} \neq \emptyset$. Let $X \in \boldsymbol{X}$ be the vertex in $\boldsymbol{V}(\boldsymbol{P}_{Y,A}) \cap \boldsymbol{X}$ that is closest to Y, i.e. there are no other vertices between X and Y along the path $\boldsymbol{P}_{Y,A}$. Let $\boldsymbol{Q}_{Y,X}$ denote this subpath of $\boldsymbol{P}_{Y,A}$. Since $\boldsymbol{P}_{Y,A}$ is active with respect to $\boldsymbol{X} \cup \boldsymbol{S}'$, X must appear as a collider on $\boldsymbol{P}_{Y,X}$. That is, $\boldsymbol{Q}_{Y,X}$ is a non-causal path from X to Y that does not contain any internal $\boldsymbol{X}$ vertices.

Case 2: Suppose $\boldsymbol{P}_{Y,A}$ does *not* contain any vertex from $\boldsymbol{X}$, i.e. $\boldsymbol{V}(\boldsymbol{P}_{Y,A}) \cap \boldsymbol{X} = \emptyset$. Since $A \in \boldsymbol{S} \setminus \boldsymbol{S}' \subseteq \boldsymbol{S} = \mathrm{Pa}(\boldsymbol{X})$, there must be an edge $A \to X$ for some $X \in \boldsymbol{X}$. Therefore, the extended path $\boldsymbol{Q}_{Y,X} = \boldsymbol{P}_{Y,A} \cup \{A \to X\}$ is a non-causal path from X to Y that does not contain any internal $\boldsymbol{X}$ vertices.

In either case, we have some non-causal path from X to Y that does not contain any internal $\boldsymbol{X}$ vertices denoted by $\boldsymbol{Q}_{Y,X}$. Since $\boldsymbol{S}'$ is a valid adjustment set, $\boldsymbol{S}'$ must block

$\boldsymbol{Q}_{Y,X}$, which implies that $\boldsymbol{P}_{Y,A}$ will be blocked by $\boldsymbol{X} \cup \boldsymbol{S}'$. This is a contradiction to the existence of such an active path $\boldsymbol{P}_{Y,A}$ in the first place.

Condition (ii) : $\boldsymbol{X} \perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}$

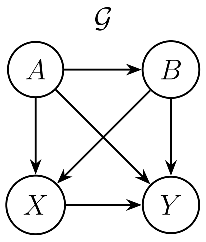
Suppose, for a contradiction, that $\boldsymbol{X} \not\!\perp\!\!\!\perp \boldsymbol{S}' \setminus \boldsymbol{S} \mid \boldsymbol{S}$. By contrapositive of the Markov property (Definition 2.46), there is an active d-connected path from some $X \in \boldsymbol{X}$ to some $B \in \boldsymbol{S}' \setminus \boldsymbol{S}$, when $\boldsymbol{S}$ is being conditioned upon. Let $\boldsymbol{P}_{X,B}$ denote such an active path. Note that $\boldsymbol{P}_{X,B}$ *cannot* begin with an incoming edge into X. This is because otherwise $\boldsymbol{P}_{X,B}$ has the form $X \leftarrow C - \ldots$ for some $C \in \mathrm{Pa}(\boldsymbol{X}) = \boldsymbol{S}$ and so would be not be active when $\boldsymbol{S}$ is being conditioned upon. So, it must be the case that $\boldsymbol{P}_{X,B}$ begins with an outgoing edge from X. Then, there must be a collider on $\boldsymbol{P}_{X,B}$ involving a descendant of X because $B \in \boldsymbol{S}' \setminus \boldsymbol{S} \subseteq \mathrm{ND}(\boldsymbol{X})$. However, the conditioning set $\boldsymbol{S} \subseteq \mathrm{ND}(\boldsymbol{X})$ would not include this descendant, so $\boldsymbol{P}_{X,B}$ would not be active. This contradicts the existence of such an active path $\boldsymbol{P}_{X,B}$ in the first place. □

B.2.4 Derivations for hardness proof

In the proof of Lemma 7.8, we argued that the distribution $\mathcal{P}$ described in Fig. 7.2 has the following well-defined conditional probabilities:

a	b	$\mathcal{P}(b \mid a)$	$\mathcal{P}(X = 0 \mid a, b)$	$\mathcal{P}(X = 0 \mid a)$	$\sum_x \lvert\mathcal{P}(x \mid a, b) - \mathcal{P}(x \mid a)\rvert$
0	0	$\sqrt{\varepsilon}/2$	$1 - \alpha + \sqrt{\varepsilon}/2$	$1 - \alpha + \varepsilon/4$	$\sqrt{\varepsilon} - \varepsilon/2$
0	1	$1 - \sqrt{\varepsilon}/2$	$1 - \alpha$	$1 - \alpha + \varepsilon/4$	$\varepsilon/2$
1	0	$1 - \sqrt{\varepsilon}/2$	α	$\alpha - \varepsilon/4$	$\varepsilon/2$
1	1	$\sqrt{\varepsilon}/2$	$\alpha - \sqrt{\varepsilon}/2$	$\alpha - \varepsilon/4$	$\sqrt{\varepsilon} - \varepsilon/2$

For convenience, we produce Fig. 7.2 below.



$$A = \begin{cases} 1 & \text{w.p. } \frac{\varepsilon}{4\alpha} \cdot \frac{\alpha - \varepsilon/4}{1 - \sqrt{\varepsilon}/2} \\ 0 & \text{else} \end{cases} \qquad X = \begin{cases} A & \text{w.p. } 1 - \alpha \\ 1 - A & \text{w.p. } \alpha - \sqrt{\varepsilon}/2 \\ B & \text{w.p. } \sqrt{\varepsilon}/2 \end{cases}$$

$$B = \begin{cases} 1 - A & \text{w.p. } 1 - \sqrt{\varepsilon} \\ 0 & \text{w.p. } \sqrt{\varepsilon}/2 \\ 1 & \text{w.p. } \sqrt{\varepsilon}/2 \end{cases} \qquad Y = \begin{cases} 1 & \text{if } X = 0, A = 1, B = 0 \\ 0 & \text{else} \end{cases}$$

Figure B.4: Reproduced: Probability distribution $\mathcal{P}$ defined over 4 binary variables $\{A, B, X, Y\}$ in a topological ordering of $A \prec B \prec X \prec Y$ with parameters ε and α, where $0 < \sqrt{\varepsilon} \leq \alpha \leq 1/2$.

We first check that all the (conditional) probabilities of $\mathcal{P}$ are well-defined. Since $0 < \sqrt{\varepsilon} \leq \alpha \leq 1/2$, the only non-straightforward term to verify is $\mathcal{P}(A = 1)$. Observe that

$$\frac{\varepsilon}{4\alpha}\cdot\frac{\alpha-\varepsilon/4}{1-\sqrt{\varepsilon}/2}\leq 1 \iff \varepsilon\cdot(\alpha-\varepsilon/4)\leq 4\alpha\cdot(1-\sqrt{\varepsilon}/2) \iff 2\alpha\sqrt{\varepsilon}+\alpha\varepsilon-\varepsilon^2/4\leq 4\alpha$$

which is true as $0<\varepsilon<\sqrt{\varepsilon}<\alpha\leq 1$ implies $2\alpha\sqrt{\varepsilon}+\alpha\varepsilon-\varepsilon^2/4\leq 3\alpha\sqrt{\varepsilon}\leq 3\alpha\leq 4\alpha$. Therefore, $0\leq\mathcal{P}(A=1)\leq 1$.

We now proceed to verify the conditional probabilities shown in the table above.

$$\begin{aligned}
&\mathcal{P}(X=0\mid A=0)\\
&=\mathcal{P}(B=0\mid A=0)\cdot\mathcal{P}(X=0\mid A=0,B=0)\\
&\quad+\mathcal{P}(B=1\mid A=0)\cdot\mathcal{P}(X=0\mid A=0,B=1)\\
&=(\sqrt{\varepsilon}/2)\cdot(1-\alpha+\sqrt{\varepsilon}/2)+(1-\sqrt{\varepsilon}/2)\cdot(1-\alpha)\\
&=1-\alpha+\varepsilon/4
\end{aligned}$$

and

$$\begin{aligned}
&\mathcal{P}(X=0\mid A=1)\\
&=\mathcal{P}(B=0\mid A=1)\cdot\mathcal{P}(X=0\mid A=1,B=0)\\
&\quad+\mathcal{P}(B=1\mid A=1)\cdot\mathcal{P}(X=0\mid A=1,B=1)\\
&=(1-\sqrt{\varepsilon}/2)\cdot\alpha+(\sqrt{\varepsilon}/2)\cdot(\alpha-\sqrt{\varepsilon}/2)\\
&=\alpha-\varepsilon/4\\
&=1-\mathcal{P}(X=0\mid A=0)
\end{aligned}$$

The detailed workings for $\sum_x|\mathcal{P}(x\mid a,b)-\mathcal{P}(x\mid a)|$ for different values of $a,b\in\{0,1\}$ are given below.

When $A=0$ and $B=0$:

$$\begin{aligned}
&\sum_x|\mathcal{P}(x\mid a,b)-\mathcal{P}(x\mid a)|\\
=\;&|\mathcal{P}(X=0\mid A=0,B=0)-\mathcal{P}(X=0\mid A=0)|\\
&+|\mathcal{P}(X=1\mid A=0,B=0)-\mathcal{P}(X=1\mid A=0)|\\
=\;&\left|(1-\alpha+\sqrt{\varepsilon}/2)-(1-\alpha+\varepsilon/4)\right|+\left|(\alpha-\sqrt{\varepsilon}/2)-(\alpha-\varepsilon/4)\right|\\
=\;&2\left(\sqrt{\varepsilon}/2-\varepsilon/4\right)\\
=\;&\sqrt{\varepsilon}-\varepsilon/2
\end{aligned}$$

When $A=0$ and $B=1$:

$$\sum_x|\mathcal{P}(x\mid a,b)-\mathcal{P}(x\mid a)|$$

$$
\begin{aligned}
&= |\mathcal{P}(X = 0 \mid A = 0, B = 1) - \mathcal{P}(X = 0 \mid A = 0)| \\
&\quad + |\mathcal{P}(X = 1 \mid A = 0, B = 1) - \mathcal{P}(X = 1 \mid A = 0)| \\
&= |(1 - \alpha) - (1 - \alpha + \varepsilon/4)| + |(\alpha) - (\alpha - \varepsilon/4)| \\
&= 2\,(\varepsilon/4) \\
&= \varepsilon/2
\end{aligned}
$$

When $A = 1$ and $B = 0$:

$$
\begin{aligned}
&\sum_x |\mathcal{P}(x \mid a, b) - \mathcal{P}(x \mid a)| \\
&= |\mathcal{P}(X = 0 \mid A = 1, B = 0) - \mathcal{P}(X = 0 \mid A = 1)| \\
&\quad + |\mathcal{P}(X = 1 \mid A = 1, B = 0) - \mathcal{P}(X = 1 \mid A = 1)| \\
&= |(\alpha) - (\alpha - \varepsilon/4)| + |(1 - \alpha) - (1 - \alpha + \varepsilon/4)| \\
&= 2\,(\varepsilon/4) \\
&= \varepsilon/2
\end{aligned}
$$

When $A = 1$ and $B = 1$:

$$
\begin{aligned}
&\sum_x |\mathcal{P}(x \mid a, b) - \mathcal{P}(x \mid a)| \\
&= |\mathcal{P}(X = 0 \mid A = 1, B = 1) - \mathcal{P}(X = 0 \mid A = 1)| \\
&\quad + |\mathcal{P}(X = 1 \mid A = 1, B = 1) - \mathcal{P}(X = 1 \mid A = 1)| \\
&= \left|(\alpha - \sqrt{\varepsilon}/2) - (\alpha - \varepsilon/4)\right| + \left|(1 - \alpha + \sqrt{\varepsilon}/2) - (1 - \alpha + \varepsilon/4)\right| \\
&= 2\left(\sqrt{\varepsilon}/2 - \varepsilon/4\right) \\
&= \sqrt{\varepsilon} - \varepsilon/2
\end{aligned}
$$

B.2.5 Weak edges

In this section, we describe a simple concrete example whereby it is suboptimal to first learn a correct causal graph and then apply identifiability formulas to estimate $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$. In particular, correctly learning the causal graph $\mathcal{G}^*$ may require taking a large number of samples, especially in the presence of "weak edges". However, one would expect such edges to contribute little to $\mathcal{P}_{\boldsymbol{x}}(\boldsymbol{y})$.

Suppose a probability distribution $\mathcal{P}$ on variables $\{X, Y, Z\}$ is generated as follows:

$$
\begin{aligned}
Z &\leftarrow \mathrm{Bern}(1/2) \\
X &\leftarrow \begin{cases} Z & \text{with probability } \varepsilon > 0 \\ \mathrm{Bern}(1/2) & \text{with probability } 1 - \varepsilon \end{cases}
\end{aligned}
$$

$$Y \leftarrow X \oplus Z$$

The causal graph that exactly captures $\mathcal{P}$ is a complete DAG with edges $Z \to X \to Y$ and $Z \to Y$; see $\mathcal{G}_1$ in Fig. B.5. However, for extremely small ε, one would require $\Omega(1/\varepsilon)$ samples to detect a dependency between X and Z. So, with small ε and insufficient samples, one may erroneously recover a subgraph without the $Z \to X$ arc; see $\mathcal{G}_2$ in Fig. B.5.

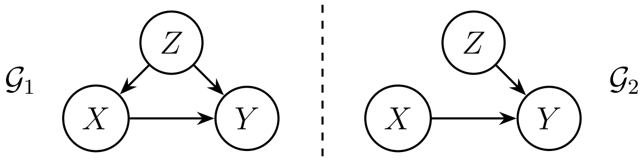


Figure B.5: While it is hard to distinguish $\mathcal{G}_1$ from $\mathcal{G}_2$ for small ε with few samples from $\mathcal{P}$, estimating $\mathcal{P}_x(y)$ using $\mathcal{G}_2$ only incurs an additive error of $O(\varepsilon)$.

Now, suppose we are interested in estimating $\mathcal{P}_0(1) = \mathcal{P}(Y = 1 \mid \mathrm{do}(X = 0))$ from observational data. One can check that the correct answer is $\mathcal{P}(Y = 1 \mid \mathrm{do}(X = 0)) = 1/2$. Applying standard adjustment formulas under $\mathcal{G}_1$ yield $\mathcal{P}(Y = 1 \mid \mathrm{do}(X = 0)) = \sum_{z \in \{0,1\}} \mathcal{P}(Z = z) \cdot \mathcal{P}(Y = 1 \mid X = 0, Z = z) = 1/2$ as expected. Meanwhile, under $\mathcal{G}_2$, the estimation would simply by $\mathcal{P}(Y = 1 \mid X = 0) = (1 - \varepsilon)/2 = 1/2 - \varepsilon/2$. Thus, see that the estimation error is only an additive $O(\varepsilon)$ factor away from the ground truth.

B.2.6 Stronger results under causal faithfulness

Recall from Amba (Algorithm 15) that we need to perform conditional independence checks of the form $\boldsymbol{X} \perp\!\!\!\perp_\varepsilon \mathrm{ND}(\boldsymbol{X}) \setminus \boldsymbol{S} \mid \boldsymbol{S}$, which could potentially involve up to $|\boldsymbol{V}|$ variables. Furthermore, we also know from Theorem 7.3 that the required sample complexity of conditional independence testing typically increases as the total number of variables involved increases. Thus, it would be preferable if we just check whether $\boldsymbol{X} \perp\!\!\!\perp_\varepsilon V \mid \boldsymbol{S}$ for each $V \in \mathrm{ND}(\boldsymbol{X}) \setminus \boldsymbol{S}$, and derive that $\boldsymbol{X} \perp\!\!\!\perp \mathrm{ND}(\boldsymbol{X}) \mid \boldsymbol{S}$. When $\varepsilon = 0$, this implication is a form *compositionality*, and is well-known to hold under the faithfulness assumption (since the set of d-separation statements in a graph is a *graphoid*, see e.g. [MDLW18, Chapter 1]), we provide an elementary proof below.

Lemma B.13. *Let $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}$ be disjoint subsets of variables. Under the causal faithfulness assumption, if $\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} \mid \boldsymbol{C}$ and $\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{D} \mid \boldsymbol{C}$, then $\boldsymbol{A} \perp\!\!\!\perp (\boldsymbol{B} \cup \boldsymbol{D}) \mid \boldsymbol{C}$.*

Proof. Suppose, for a contradiction, that $\boldsymbol{A} \not\perp\!\!\!\perp (\boldsymbol{B} \cup \boldsymbol{D}) \mid \boldsymbol{C}$. Under the causal faithfulness assumption, this means that there is a d-connected path P from some $A \in \boldsymbol{A}$ to some $V \in \boldsymbol{B} \cup \boldsymbol{D}$ that is active with respect to $\boldsymbol{C}$. Without loss of generality, due to symmetry of the statement, suppose that $V \in \boldsymbol{B}$. That is, P is a path from $A \in \boldsymbol{A}$ to some $V \in \boldsymbol{V}$

that is active with respect to $\boldsymbol{C}$. But such an active path P contradicts the assumption that $\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} \mid \boldsymbol{C}$. Contradiction. □

Note that Lemma B.13 is *false* in general with respect to unfaithful distributions.

Example B.14. The simple 3-variable distribution $X = Z_1 \oplus Z_2$, where Z_1 and Z_2 are independent fair coin flips is unfaithful to any DAG on 3 nodes. To see why, observe that any two variables are unconditionally independent but completely dependent upon conditioning on the third. So, one would minimally have to use a v-structure, say $Z_1 \to X \leftarrow X_2$ to represent this. However, $Z_1 \to X$ is an active path which implies $Z_1 \not\perp\!\!\!\perp X$ under the causal faithfulness assumption, which is not true in $\mathcal{P}(Z_1, Z_2, X)$.

Unfortunately, faithfulness alone is not sufficient to ensure the desired implication. As we demonstrate in the following example (a minor adaptation of the above example), a distribution $\mathcal{P}(\boldsymbol{V})$ may be faithful to a DAG, but fail to satisfy the desired compositionality-style property.

Lemma B.15. *Let* $0 < \varepsilon \leq 1/2$. *Consider a probability distribution* $\mathcal{P}$ *over three binary variables* (A, B, X) *where* $A \sim \mathrm{Bern}(1/2)$ *and* $B \sim \mathrm{Bern}(1/2)$ *are two independent Bernoulli random variables, each with success probability* $1/2$ *and* X *is defined as follows:*

$$X = \begin{cases} A \oplus B & \textit{with probability } 1 - 2\varepsilon \\ A & \textit{with probability } \varepsilon \\ B & \textit{with probability } \varepsilon \end{cases}$$

We have $\Delta_{X \perp\!\!\!\perp A \mid \varnothing} = \Delta_{X \perp\!\!\!\perp B \mid \varnothing} = \varepsilon$ *and* $\Delta_{X \perp\!\!\!\perp (A,B) \mid \varnothing} = \frac{1}{2} - \varepsilon$.

Proof. By construction, $\mathcal{P}(A = 0) = \mathcal{P}(B = 0) = \mathcal{P}(X = 0) = 1/2$. Meanwhile, one can check that $\mathcal{P}(X = 0, A = 0) = \mathcal{P}(X = 1, A = 1) = \frac{1}{4} + \frac{\varepsilon}{2}$ and $\mathcal{P}(X = 0, A = 1) = \mathcal{P}(X = 1, A = 0) = \frac{1}{4}$. For instance, $\mathcal{P}(X = 0, A = 0) = \mathcal{P}(X = 0 \mid A = 0) \cdot \mathcal{P}(A = 0) = \left((1 - \varepsilon) \cdot \frac{1}{2} + \varepsilon + \varepsilon \cdot \frac{1}{2}\right) \cdot \frac{1}{2} = \frac{1}{4} + \frac{\varepsilon}{2}$. So, $\sum_{x,a \in \{0,1\}} |\mathcal{P}(x, a) - \mathcal{P}(x) \cdot \mathcal{P}(a)| = \varepsilon$. By Definition 2.40, this establishes $\Delta_{X \perp\!\!\!\perp A \mid \varnothing} = \varepsilon$.

The analysis of $\Delta_{X \perp\!\!\!\perp B \mid \varnothing} = \varepsilon$ is symmetric by replacing the role of A by B in the above analysis.

Since A and B are independent Bernoulli random variables, we see that $\mathcal{P}(A = a, B = b) = \mathcal{P}(A = a) \cdot \mathcal{P}(B = b) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$ for any $a, b \in \{0, 1\}$. Meanwhile,

$$\begin{aligned} \mathcal{P}(X = 0 \mid A = 0, B = 0) &= 1 \\ \mathcal{P}(X = 0 \mid A = 0, B = 1) &= \varepsilon \\ \mathcal{P}(X = 0 \mid A = 1, B = 0) &= \varepsilon \\ \mathcal{P}(X = 0 \mid A = 1, B = 1) &= 1 - 2\varepsilon \end{aligned}$$

So,

$$\begin{aligned}
&\sum_{x,a,b\in\{0,1\}} |\mathcal{P}(x,a,b) - \mathcal{P}(x)\cdot\mathcal{P}(a,b)| \\
&= \sum_{x,a,b\in\{0,1\}} \mathcal{P}(a,b)\cdot|\mathcal{P}(x \mid a,b) - \mathcal{P}(x)| \\
&= \frac{1}{4}\cdot\sum_{x,a,b\in\{0,1\}} \left|\mathcal{P}(x \mid a,b) - \frac{1}{2}\right| && \text{(Since } \mathcal{P}(a,b) = \tfrac{1}{4} \text{ and } \mathcal{P}(x) = \tfrac{1}{2} \text{ always)} \\
&= \frac{1}{4}\cdot\left(\left|1-\frac{1}{2}\right| + \left|\varepsilon - \frac{1}{2}\right| + \left|\varepsilon - \frac{1}{2}\right| + \left|1 - 2\varepsilon - \frac{1}{2}\right|\right) && \text{(From above)} \\
&= \frac{1}{4}\cdot(2-4\varepsilon) && \text{(Since } \varepsilon \le \tfrac{1}{2}\text{)} \\
&= \frac{1}{2} - \varepsilon
\end{aligned}$$

By Definition 2.40, this establishes $\Delta_{X\,\perp\!\!\!\perp (A,B)|\varnothing} = \frac{1}{2} - \varepsilon$. □

The above example demonstrates that the faithfulness assumption is insufficient for our purposes. Instead, we need an assumption of the following form; as we will show, this assumption is implied by a type of *strong faithfulness* assumption.

Definition B.16. We say that $\mathcal{P}(\boldsymbol{V})$ obeys (ε,γ)-*strong compositionality* if, for any disjoint sets $\boldsymbol{A},\boldsymbol{B},\boldsymbol{C},\boldsymbol{D} \subseteq \boldsymbol{V}$, the following is true:

$$(\boldsymbol{A} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{B} \mid \boldsymbol{C}) \wedge (\boldsymbol{A} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{D} \mid \boldsymbol{C}) \implies (\boldsymbol{A} \perp\!\!\!\perp_{\gamma\varepsilon} \boldsymbol{B} \cup \boldsymbol{D} \mid \boldsymbol{C})$$

Under (ε,γ)-strong compositionality, we can derive that $\boldsymbol{X} \perp\!\!\!\perp_{\varepsilon} \boldsymbol{Z} \setminus \boldsymbol{S} \mid \boldsymbol{S}$ from smaller conditional independence tests; in particular, using a bisection arguments, if $\boldsymbol{X} \perp\!\!\!\perp_{\varepsilon} V \mid \boldsymbol{S}$ for all $V \in \boldsymbol{Z} \setminus \boldsymbol{S}$, then $\boldsymbol{X} \perp\!\!\!\perp_{\gamma^k\varepsilon} \boldsymbol{Z} \setminus \boldsymbol{S} \mid \boldsymbol{S}$ for $k = \lceil \log_2(|\boldsymbol{Z} \setminus \boldsymbol{S}|) \rceil$. Finally, we relate can strong compositionality to the faithfulness assumption: strong faithfulness implies strong compositionality with $\gamma = 0$, as follows.

Assumption B.17 (TV Strong faithfulness). If $\boldsymbol{A}$ is d-connected to $\boldsymbol{B}$ given $\boldsymbol{C}$ in $\mathcal{G}^*$, then

$$\Delta_{\boldsymbol{A}\,\perp\!\!\!\perp\,\boldsymbol{B}|\boldsymbol{C}} > \beta$$

Equivalently, $\Delta_{\boldsymbol{A}\,\perp\!\!\!\perp\,\boldsymbol{B}|\boldsymbol{C}} \le \beta \implies \boldsymbol{A}$ is d-separated from $\boldsymbol{B}$ given $\boldsymbol{C}$.

Lemma B.18 (TV strong faithfulness implies strong compositionality)**.** *Suppose $\mathcal{P}(\boldsymbol{V})$ is β-TV strong faithful to $\mathcal{G}^*$. Then $\mathcal{P}(\boldsymbol{V})$ is $(\beta,0)$-compositional.*

Proof. Suppose $\boldsymbol{A} \perp\!\!\!\perp_{\beta} \boldsymbol{B} \mid \boldsymbol{C}$ and $\boldsymbol{A} \perp\!\!\!\perp_{\beta} \boldsymbol{D} \mid \boldsymbol{C}$. Then, by β-TV strong faithfulness, $\boldsymbol{A}$ is is d-separated from $\boldsymbol{B}$ given $\boldsymbol{C}$, and $\boldsymbol{A}$ is d-separated from $\boldsymbol{D}$ given $\boldsymbol{C}$. Thus, $\boldsymbol{A}$ is d-separated from $\boldsymbol{B} \cup \boldsymbol{D} \mid \boldsymbol{C}$, so $\boldsymbol{A} \perp\!\!\!\perp \boldsymbol{B} \cup \boldsymbol{D} \mid \boldsymbol{C}$. □

Appendix C

Addendum for Part III

C.1 Addendum for Chapter 9

C.1.1 Extended variant of Theorem 9.3

Let us first prove the case when the algorithm $\mathcal{A}$ is deterministic, but $\alpha \in [0, 1/2]$. We will again use $\mathcal{G}_1$ and $\mathcal{G}_2$ of Fig. 9.3 (replicated below for convenience as Fig. C.1) as a counterexample. Our argument follows that of the case where $\alpha = 0$.

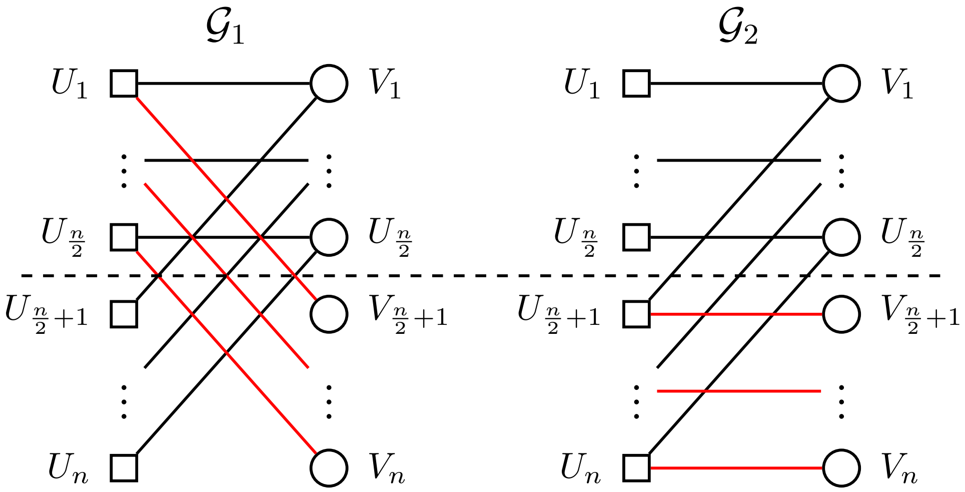


Figure C.1: (Restated) Illustration of $\mathcal{G}_1$ and $\mathcal{G}_2$ for Theorem 9.3

Special case: $\mathcal{A}$ is deterministic. As before, we observe that any algorithm cannot distinguish between the $\mathcal{G}_1$ and $\mathcal{G}_2$ after the first $n/2$ arrivals. Suppose $\mathcal{A}$ is $(1-\alpha)$-consistent. Without loss of generality, by symmetry of the argument, suppose $\mathcal{G}^* = \mathcal{G}_2$ and $\mathcal{A}$ is given advice bit $\hat{\imath} = 2$.

Since $\mathcal{A}$ is $(1-\alpha)$-consistent, it has to make at least $\frac{n}{2} - (1-\alpha) \cdot n$ matches in the first $\frac{n}{2}$ arrivals[13], leaving at most $\alpha \cdot n$ unmatched offline vertices amongst $\{u_1, \ldots u_{\frac{n}{2}}\}$.

[13] Otherwise, even if the remaining $\frac{n}{2}$ vertices are matched, $\mathcal{A}$ cannot achieve $(1-\alpha) \cdot n$ total matches, violating $(1-\alpha)$-consistency

Meanwhile, if $\mathcal{G}^* = \mathcal{G}_1$ instead, there can only be at most $\alpha \cdot n$ matches amongst the remaining $\frac{n}{2}$ arrivals $\{v_{\frac{n}{2}+1}, \ldots, v_n\}$, resulting in a total matching size of at most $\frac{n}{2}+\alpha \cdot n = \left(\frac{1}{2} + \alpha\right) \cdot n$. That is, any deterministic $\mathcal{A}$ that is $(1-\alpha)$-consistent cannot be strictly more than $\left(\frac{1}{2} + \alpha\right)$-robust.

General case where $\mathcal{A}$ could be randomized. Unfortunately, randomization does not appear to help much, as we can repeat all of the above arguments in expectation. That is, if $\hat{i} = 2$, it follows from consistency that in expectation, at least $(1 - \alpha) \cdot n$ of all vertices must be eventually matched, meaning that in expectation there must be $\frac{n}{2} - \alpha \cdot n$ matches in the first half. Now, if $\mathcal{G}_1$ was the true graph, then in expectation we only have $\alpha \cdot n$ possible matches to make in the second half, thus we have a maximum of $\left(\frac{1}{2} + \alpha\right) \cdot n$ matches in expectation when $\hat{i}$ is wrong.

C.1.2 Examples of realized type counts as advice

Example 1: Online Ads. The canonical example of online bipartite matching is that of online ads [Meh13]. Recall that the online vertices are advertisement slots (also called impressions) and the offline vertices are advertisers. We can see that the distribution over types can be possibly forecasted by machine learning models (and in fact, indirectly used [AMK21] for bipartite matching) and used as advice. This directly gives us $\mathcal{Q}$, possibly bypassing $\hat{c}$. Regardless, the more accurate the forecasting, the lower $\ell_1(\mathcal{P}, \mathcal{Q})$ will be.

Example 2: Food allocation. Consider a conference organizer catering lunch. As a cost-cutting measure, they cater *exactly* one food item per attendee, based on their self-reported initial dietry preferences reported during registration (each attendee may report more than one item). During the conference, attendees will queue up in random order, *sequentially* reporting their preferences once again and being assigned their food. Organizers have the flexibility to assign food items based on this new reporting of preferences (or, in a somewhat morally questionable fashion refuse to serve the attendee—though in the unweighted setting, reasonable algorithms should not have to do this!). Alas, a fraction of attendees claim a different preference from their initial preference, e.g. because they were fickle, or did not take initial dietry preference questionnaire seriously. Given that food is already catered, how should the conference organizers sequentially distribute meals to minimize hungry attendees?

The attendees are represented by n online vertices, while each of the n offline vertices represent one of k types of food item[14]. The attendees' initial preference gives our advice $\mathcal{Q}$ (the distribution over types of food prefernces), which also describes a perfect matching. This preference may differ from the distribution over true preferences reported on the day of the conference $\mathcal{P}$. However, one can reasonable assume that only a small fraction of

[14] For practical settings, the types of food items is generally much smaller thatn n.

attendees exhibit such a mismatch, meaning that the $\ell_1(\mathcal{P}, \mathcal{Q})$ is fairly small and advice should be accepted most of the time.

Example 3: Centralized labor Allocation. Suppose there are n employees and m jobs. There are η different qualifications. This is represented by a binary matrix $\{0,1\}^{n,\eta}$, where $X_{i,k} = 1$, if employee i posseses qualification k. Therefore, the i-th row of X, X_i is a length η boolean string containing all of i's skills.

For employee i to perform a job, X_i needs to satisfy a boolean formula (say, given in conjunctive form). This is quite reasonable, e.g. to be an AI researcher, it needs to have knowledge of some programming language (Python, Matlab, etc.), some statistics (classical or modern), and some optimization (whether discrete or continuous). In the bipartite graph, employee i has an edge to job j if and only if X_i satisfies this formula.

In this case, the qualifications of each employee are known by the company, who has access to their employees. Given the qualifications, the set of jobs that may be performed can be computed offline and used as advice. This advice may not be entirely correct: for example, employees may have picked up new skills (hence there may be more edges than we thought, but no less). Of course, there could also be some employees with phoney qualifications; this fraction is not too high.

One interesting property about this application is that advice may only be imperfect in the sense that edges could be added. This means that if we just mimicked, we are guaranteed to get at least $\hat{n}$. Also, the coarsening method is more easily applied.

C.1.3 Computing the optimal remapping σ via a maximum flow formulation for offline setting

Consider the offline setting where we are given the true counts $\boldsymbol{c}^*$ and the advice counts $\widehat{\boldsymbol{c}}$. Suppose $\boldsymbol{c}^*$ has r non-zero counts, represented by: $\langle L_1^*, \boldsymbol{c}_1^* \rangle, \langle L_2^*, \boldsymbol{c}_2^* \rangle, \ldots \langle L_r^*, \boldsymbol{c}_r^* \rangle$, where $\sum_{i=1}^r \boldsymbol{c}_i^* = n$. Meanwhile, suppose $\widehat{\boldsymbol{c}}$ has s non-zero counts, represented by: $\langle \widehat{L}_1, \widehat{\boldsymbol{c}}_1 \rangle, \langle \widehat{L}_2, \widehat{\boldsymbol{c}}_2 \rangle, \ldots \langle \widehat{L}_s, \widehat{\boldsymbol{c}}_s \rangle$, where $\sum_{i=1}^s \widehat{\boldsymbol{c}}_i = n$. To compute a remapping from $\boldsymbol{c}^*$ to $\widehat{\boldsymbol{c}}$ to maximize the number of resulting overlaps, consider the following max flow formulation on a directed graph $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ with $|\boldsymbol{V}| = r + s + 2$ nodes:

- Create a node for each of $L_1^*, \ldots, L_r^*, \widehat{L}_1, \ldots, \widehat{L}_s$.
- Create a "source" and a "destination" node.
- Add an edge with a capacity $\boldsymbol{c}_i^*$ from the "source" node to each of the L_i^* nodes, for $i \in \{1, \ldots, r\}$
- $(*)$ Add an edge from L_i^* to $\widehat{L}_j$ with capacity $\boldsymbol{c}_i^*$ if $\widehat{L}_j \subseteq L_i^*$, for $i \in \{1, \ldots, r\}$ and $j \in \{1, \ldots, s\}$.

- Add an edge with a capacity $\widehat{\boldsymbol{c}}_j$ from each of the $\widehat{L}_j$ nodes to the "destination" node, for $j \in \{1, \ldots, s\}$.
- Compute the maximum flow from "source" to "destination".

Since the graph has integral edge weights, the maximum flow is integral and the flow across each edge is integral. The resultant maximum flow is the maximum attainable overlap between a remapped $\boldsymbol{c}^*$ and $\widehat{\boldsymbol{c}}$, and we can obtain the remapping σ by reading off the flows between on the edges from $(*)$.

C.1.4 ILP for advice coarsening

Here, we give an integer linear program (ILP) that takes in any number $|\hat{T}| = \hat{r}$ of desired groupings as input and produces a grouping proposed advice count $\hat{c}_{\hat{r}}$ on $\hat{r}$ labels that implies the maximum possible matching. Recall that the a smaller number of resulting groups $\hat{r}$ directly translates to fewer samples $s_{\hat{r},\varepsilon,\delta}$ required in TestAndMatch. So, to utilize this ILP, one can solve for decreasing values of $r = |\hat{L}|, |\hat{L}| - 1, \ldots, 1$ and evaluate the resulting maximum matching size $\hat{n}_r$ for each proposed advice count $\hat{c}_r$. Then, one can either use the smallest possible r which still preserves the size of maximum matching or even combine this with the idea from Section 9.6.3 if one needs to further decrease r.

We propose to update the labels by taking *intersections* of the patterns, i.e. for any resulting group $\boldsymbol{G}_i$, we define its label pattern as $\bigcap_{V \in \boldsymbol{G}_i} N(V)$. Since taking intersections only restricts the edges which can be used in forming a maximum matching, this ensures that Mimic will always be able to mimic any proposed matching implied by the grouped patterns.

Explanation of constants and variables

- Given the n online input patterns, $b_{i,j}$ is a Boolean constant indicating whether online vertex $i \in [n]$ does *not* have $j \in [n]$ as a neighbor in its pattern.
- Main decision variable: $x_{i,j}$ whether edge from online vertex i to offline vertex j is part of the matching.
- Auxiliary variable: $z_{i,\ell}$ is an indicator whether online vertex $i \in [n]$ is assigned to group $\ell \in [k]$.
- Product variable: $w_{i,j,\ell} = z_{i,\ell} \cdot z_{j,\ell}$ is an indicator whether *both* online vertices i and j are in group ℓ

The ILP

$$\max \quad \sum_{(i,j) \in \boldsymbol{E}} x_{i,j}$$

$$
\begin{aligned}
s.t. \quad & \sum_{\substack{j \in [n] \\ (i,j) \in \boldsymbol{E}}} x_{i,j} \leq 1 && \forall i \in [n] && (C1) \\
& \sum_{\substack{i \in [n] \\ (i,j) \in \boldsymbol{E}}} x_{i,j} \leq 1 && \forall j \in [n] && (C2) \\
& x_{i,j} \leq 1 - w_{i,q,\ell} \cdot b_{q,j} && \forall (i,j) \in \boldsymbol{E}, q \in [n], \ell \in [k] && (C3) \\
& w_{i,j,\ell} \leq z_{i,\ell} && \forall i \in [n], \ell \in [k] && (C4) \\
& w_{i,j,\ell} \leq z_{j,\ell} && \forall j \in [n], \ell \in [k] && (C5) \\
& w_{i,j,\ell} \geq z_{i,\ell} + z_{j,\ell} - 1 && \forall i, j \in [n], \ell \in [k] && (C6) \\
& \sum_{\ell=1}^{k} z_{i,\ell} = 1 && \forall i \in [n] && (C7) \\
& x_{i,j} \in \{0,1\} && \forall (i,j) \in \boldsymbol{E} \\
& z_{i,\ell} \in \{0,1\} && \forall i \in [n], \ell \in [k] \\
& w_{i,j,\ell} \in \{0,1\} && \forall i, j \in [n], \ell \in [k]
\end{aligned}
$$

Explanation of constraints

- (C1, C2) Standard matching constraints.
- (C3) Can only use edge (i, j) if it is not "disabled" due to intersections. As long as *some* other vertex in the same group as i does *not* have j, the edge (i, j) will be disabled.
- (C4, C5, C6) Encoding $w_{i,j,\ell} = z_{i,\ell} \cdot z_{j,\ell}$.
- (C7) Every vertex assigned exactly one group.

C.1.5 Proof of concept

It is our understanding that the tester proposed by [JHW18] requires a significant amount of hyperparameter tuning and no off-the-shelf implementation is available [Han24]. One may consider using an older method by [VV11] which is also sublinear in the number of samples but their proposed algorithm is for non-tolerant testing and requires a non-trivial code adaptation before it is applicable to ℓ_1 estimation.

As a proof-of-concept, we implemented the TestAndMatch algorithm with the empirical ℓ_1 estimator and study the resultant competitive ratio under degrading advice quality. See https://github.com/cxjdavin/online-bipartite-matching-with-imperfect-advice for the source code.

Implementation details

From Section 9.3.1, we know that the state-of-the-art advice-less algorithm for random order arrival is the RANKING algorithm of [KVV90] which achieve a competitive ratio of $\beta = 0.696$ [MY11].

For our testing threshold, we set $\varepsilon = \hat{n}/n - \beta$ so that $\tau = 2(\hat{n}/n - \beta) - \varepsilon = \hat{n}/n - \beta$. We also implemented the following practical extensions to TESTANDMATCH which we discussed in Section 9.6:

1. Sigma remapping (Section 9.6.1)
2. Bucketing so that $\hat{r}/\varepsilon^2 < n$ (Section 9.6.2)
3. Patching so that $\hat{n}' = n$ (Section 9.6.3)

We tested 4 variants of TESTANDMATCH, one with all extensions enabled and three others that disables one extension at a time (for ablation testing).

Instances

Our problem instances are generated from the synthetic hard known i.i.d. instance of [MGS12] where any online algorithm achieves a competitive ratio of at most 0.823 in expectation:

- Let Y_k denote the set of online vertices with k random offline neighbors (out of $\binom{n}{k}$)
- Let $m = \frac{c^*_{2.5}}{2} \cdot n$, where $c^*_{2.5} = 0.81034$ is some constant defined in [MGS12] (not to be confused with our type counts c^*)
- Sample m random online vertices from Y_2, i.e. each online vertex is adjacent to a random subset of 2 offline vertex.
- Sample m random online vertices from Y_3, i.e. each online vertex is adjacent to a random subset of 3 offline vertex.
- Sample $n - 2m$ random online vertices from Y_n, i.e. each online vertex is adjacent to every offline vertex.
- Permute the online vertices for a random order arrival

Here, the support size of any generated type count c^* is roughly $0.8n$ due to the samples from Y_2 and Y_3.

Corrupting advice

Starting with perfect advice $\hat{c} = c^*$, we corrupt the advice by an α parameter using two types of corruption.

1. Pick a random $\alpha \in [0, 1]$ fraction of online vertices
2. Generate a random type for each of them by independently connecting to each offline vertex with probability $\frac{\ln n}{10n}$.
3. Type 1 corruption (add extra connections): Define the new type as the union of the old vertex type and the new random type.
4. Type 2 corruption (replace connections): Define the new type as the new random type.

As a remark, our random type generation biases towards a relatively sparse corrupted graph.

Preliminary results

We generated 10 random graph instances with $n = 2000$ offline and n online vertices. Fig. C.2 illustrates the resulting plots with error bars.

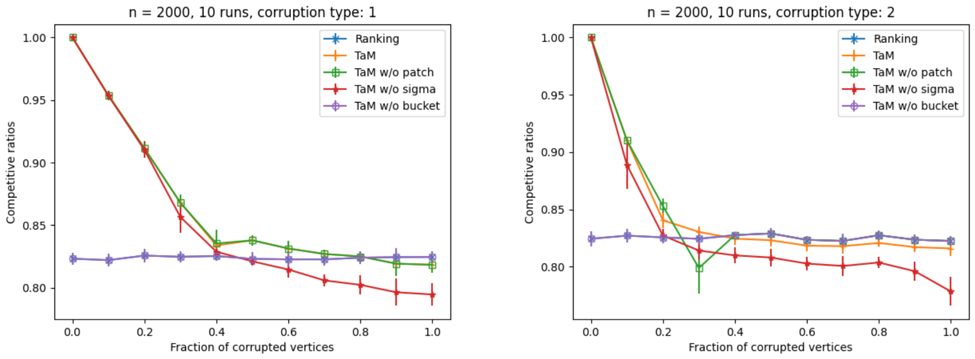


Figure C.2: $n = 2000$, averaged over 10 runs. TaM refers to our implementation of TestAndMatch.

In all cases, we see that the attained competitive ratio is highest when all extensions are enabled. We also see that the degradation below the baseline is not very severe (< 0.1 for all cases, even when not all extensions are enabled).

Unsurprisingly, the competitive ratios of Ranking and "TaM without bucket" coincide because because $r/\varepsilon^2 > n$ and we always default to baseline without performing any tests (to maintain robustness).

For corruption type 1, the "sigma remapping" extension makes our algorithm robust against additive edge corruption, and so the "patching" extension has no further impact.

C.2 Addendum for Chapter 10

C.2.1 Tolerant testing

In this section, we present an algorithm for testing whether an unknown distribution is close to a standard normal distribution. More specifically, we first describe a tolerant tester for the property that the mean of an isotropic Gaussian distribution equals zero. Subsequently, we present a tolerant tester for the property that the covariance matrix equals the identity matrix.

Tolerant testing for mean

The definition of a tolerant tester for the mean of an isotropic Gaussian distribution is given below.

Definition C.1 (Tolerant testing of isotropic Gaussian mean)**.** Fix $m \geq 1$, $d \geq 1$, $\varepsilon_2 > \varepsilon_1 > 0$, and $\delta > 0$. Suppose $\boldsymbol{\mu} \in \mathbb{R}^d$ is a hidden mean vector and we draw m samples $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m \sim N(\boldsymbol{\mu}, \boldsymbol{I}_d)$. An algorithm ALG is said to be a $(\varepsilon_1, \varepsilon_2, \delta)$-tolerant isotropic Gaussian mean tester if it satisfies the following two conditions:

1. If $\|\boldsymbol{\mu}\|_2 \leq \varepsilon_1$, then ALG should *Accept* with probability at least $1 - \delta$
2. If $\|\boldsymbol{\mu}\|_2 \geq \varepsilon_2$, then ALG should *Reject* with probability at least $1 - \delta$.

ALG is allowed to decide arbitrarily when $\varepsilon_1 < \|\boldsymbol{\mu}\|_2 < \varepsilon_2$.

It is known that the test statistic $y_n = \left\| \frac{1}{\sqrt{n}} \sum_{i=1}^n \boldsymbol{x}_i \right\|_2^2$ can be used for *non-tolerant* isotropic Gaussian mean testing with an appropriate threshold; see [DKS17, Appendix C]. With the following lemma we show that y_n can also be used for *tolerant* isotropic Gaussian mean testing.

Lemma C.2. *Fix $m \geq 1$, $d \geq 1$, $\varepsilon_2 > \varepsilon_1 > 0$, and $\delta > 0$. Suppose $\boldsymbol{\mu} \in \mathbb{R}^d$ is a hidden mean vector and we draw m i.i.d. samples $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m \sim N(\boldsymbol{\mu}, \boldsymbol{I}_d)$. When $d \geq \left(\frac{16\varepsilon_2^2}{\varepsilon_2^2 - \varepsilon_1^2}\right)^2$ and $m \in \mathcal{O}\left(\frac{\sqrt{d}}{\varepsilon_2^2 - \varepsilon_1^2} \log\left(\frac{1}{\delta}\right)\right)$, ToleRantIGMT (Algorithm 30) is a $(\varepsilon_1, \varepsilon_2, \delta)$-tolerant isotropic Gaussian mean tester.*

Proof. The total number of samples m required is $nr \in \mathcal{O}\left(\frac{\sqrt{d}}{\varepsilon_2^2 - \varepsilon_1^2} \log\left(\frac{1}{\delta}\right)\right)$ since ToleRantIGMT uses $n = \frac{16\sqrt{d}}{\varepsilon_2^2 - \varepsilon_1^2}$ i.i.d. samples in each of the $r \in \mathcal{O}(\log(\frac{1}{\delta}))$ rounds.

For correctness, we will prove that each round $i \in \{1, \dots, r\}$ succeeds with probability at least $2/3$. Then, by Chernoff bound, the majority outcome out of $r \geq \log(\frac{12}{\delta})$ independent tests will be correct with probability at least $1 - \delta$.

Now, fix an arbitrary round $i \in \{1, \dots, r\}$. ToleRantIGMT uses $n = \frac{16\sqrt{d}}{\varepsilon_2^2 - \varepsilon_1^2} \geq 1$ i.i.d. samples to form a statistic $y_n^{(i)}$ and tests against the threshold $\tau = d + \frac{n(\varepsilon_1^2 + \varepsilon_2^2)}{2}$. From

Algorithm 30 The TOLERANTIGMT algorithm.

```
Input: $\varepsilon_2 > \varepsilon_1 > 0$, $\delta \in (0,1)$, $m$ i.i.d. samples of $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$, where $\boldsymbol{\mu} \in \mathbb{R}^d$
Output: Fail (too little samples), Accept ($\|\boldsymbol{\mu}\|_2 \leq \varepsilon_1$), or Reject ($\|\boldsymbol{\mu}\|_2 \geq \varepsilon_2$).
Define testing threshold $\tau = d + \frac{n(\varepsilon_1^2+\varepsilon_2^2)}{2}$
Define sample batch size $n = \frac{16\sqrt{d}}{\varepsilon_2^2-\varepsilon_1^2}$
Define number of rounds $r = \lceil \log(\frac{12}{\delta}) \rceil$ if $\lceil \log(\frac{12}{\delta}) \rceil$ is odd, otherwise define $r = 1 + \lceil \log(\frac{12}{\delta}) \rceil$
if $m < nr$ then
    return Fail
else
    for $i \in \{1, \ldots, r\}$ do
        Use an unused batch of $n$ i.i.d. samples $\boldsymbol{x}_1^{(i)}, \ldots, \boldsymbol{x}_n^{(i)} \sim N(\boldsymbol{\mu}, \boldsymbol{I}_d)$
        Compute test statistic $y_n^{(i)} = \left\| \frac{1}{\sqrt{n}} \sum_{i=1}^n \boldsymbol{x}_i^{(i)} \right\|_2^2$ for the $i^{th}$ test
        Define $i^{th}$ outcome $\mathsf{R}^{(i)}$ as Accept if $y_n^{(i)} \leq \tau$ and Reject otherwise
    return majority($\mathsf{R}^{(1)}, \ldots, \mathsf{R}^{(r)}$)
```

Lemma 2.32 (first item), we know that $y_n^{(i)} \sim \chi_d'^2(\lambda)$ is a non-central chi-square random variable with $\lambda = n\|\boldsymbol{\mu}\|_2^2$. Let us define $t = \frac{n(\varepsilon_2^2-\varepsilon_1^2)}{2} > 0$. Observe that we can rewrite the testing threshold τ in two different ways: $\tau = d + \frac{n(\varepsilon_1^2+\varepsilon_2^2)}{2} = d + n\varepsilon_1^2 + t = d + n\varepsilon_2^2 - t$.

Case 1: $\|\boldsymbol{\mu}\|_2 \leq \varepsilon_1$

In this case, we have $\lambda = n\|\boldsymbol{\mu}\|_2^2 \leq n\varepsilon_1^2$ and $\tau = d + n\varepsilon_1^2 + t$. So,

$$\begin{aligned}
\Pr(y_n^{(i)} > \tau) &= \Pr(y_n^{(i)} > d + n\varepsilon_1^2 + t) && \text{(since } \tau = d + n\varepsilon_1^2 + t\text{)}\\
&\leq \Pr(y_n^{(i)} > d + \lambda + t) && \text{(since } \lambda \leq n\varepsilon_1^2\text{)}\\
&\leq \exp\left(-\frac{dt^2}{4(d+2\lambda)(d+2\lambda+t)}\right)\\
& && \text{(apply Lemma 2.32 (second item) with } t > 0\text{)}\\
&\leq \exp\left(-\frac{dt^2}{4(d+2n\varepsilon_1^2)(d+2n\varepsilon_1^2+t)}\right) && \text{(since } \lambda \leq n\varepsilon_1^2\text{)}\\
&\leq \exp\left(-\frac{dn^2(\varepsilon_2^2-\varepsilon_1^2)^2}{16(d+2n\varepsilon_1^2)(d+2n\varepsilon_2^2)}\right) && \text{(since } t = \tfrac{n(\varepsilon_2^2-\varepsilon_1^2)}{2} \leq 2n(\varepsilon_2^2-\varepsilon_1^2)\text{)}\\
&= \exp\left(-\frac{16^2d^2}{16(d+2n\varepsilon_1^2)(d+2n\varepsilon_2^2)}\right) && \text{(since } n = \tfrac{16\sqrt{d}}{\varepsilon_2^2-\varepsilon_1^2}\text{)}\\
&= \exp\left(-\frac{16}{\left(1+\frac{2n\varepsilon_1^2}{d}\right)\left(1+\frac{2n\varepsilon_2^2}{d}\right)}\right)\\
& && \text{(dividing both numerator and denominator by } 16d^2\text{)}\\
&= \exp\left(-\frac{16}{\left(1+\frac{32\varepsilon_1^2}{\sqrt{d}(\varepsilon_2^2-\varepsilon_1^2)}\right)\left(1+\frac{32\varepsilon_2^2}{\sqrt{d}(\varepsilon_2^2-\varepsilon_1^2)}\right)}\right) && \text{(since } n = \tfrac{16\sqrt{d}}{\varepsilon_2^2-\varepsilon_1^2}\text{)}
\end{aligned}$$

$$\begin{aligned} &= \exp\left(-\frac{16}{(1+2)(1+2)}\right) && \left(\text{since } d \geq \left(\frac{16\varepsilon_2^2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2 \geq \left(\frac{16\varepsilon_1^2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2\right) \\ &= \exp\left(-\frac{16}{9}\right) < \frac{1}{3} \end{aligned}$$

Thus, when $\|\boldsymbol{\mu}\|_2 \leq \varepsilon_1$, we have $\Pr(y_n^{(i)} \leq \tau) \geq 2/3$ and the i^{th} test outcome will be correctly an Accept with probability at least $2/3$.

Case 2: $\|\boldsymbol{\mu}\|_2 \geq \varepsilon_2$

In this case, we have $\lambda = n\|\boldsymbol{\mu}\|_2^2 \geq n\varepsilon_2^2 > n\varepsilon_1^2$ and $\tau = d + n\varepsilon_2^2 - t$. We first observe the following inequalities:

- Since $n \geq 1$, $d \geq 1$, $\lambda \geq n\varepsilon_2^2$, and $\varepsilon_2 > \varepsilon_1 > 0$, we see that

$$\left(2 - \frac{n\varepsilon_1^2}{\lambda} - \frac{n\varepsilon_2^2}{\lambda}\right)^2 \geq \left(1 - \frac{\varepsilon_1^2}{\varepsilon_2^2}\right)^2 \quad \text{and} \quad \left(\frac{d}{\lambda} + 2\right)^2 \leq \left(\frac{d}{n\varepsilon_2^2} + 2\right)^2 \tag{C.1}$$

- Since $n = \frac{16\sqrt{d}}{\varepsilon_2^2-\varepsilon_1^2} \geq 1$ and $d \geq \left(\frac{16\varepsilon_2^2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2 \geq 1$, we see that

$$\left(1 + \frac{2n\varepsilon_2^2}{d}\right)^2 \leq 3^2 \tag{C.2}$$

So,

$$\begin{aligned} \Pr(y_n^{(i)} < \tau) &= \Pr(y_n^{(i)} < d + n\varepsilon_2^2 - t) && (\text{since } \tau = d + n\varepsilon_2^2 - t) \\ &= \Pr(y_n^{(i)} < d + \lambda - (\lambda + t - n\varepsilon_2^2)) && (\text{Rewriting}) \\ &\leq \exp\left(-\frac{d(\lambda + t - n\varepsilon_2^2)^2}{4(d+2\lambda)^2}\right) \\ & && (\text{apply Lemma 2.32 (third item) with } 0 < \lambda + t - n\varepsilon_2^2 < d + \lambda) \\ &= \exp\left(-\frac{d\left(\lambda - \frac{n}{2}\varepsilon_1^2 - \frac{n}{2}\varepsilon_2^2\right)^2}{4(d+2\lambda)^2}\right) && \left(\text{since } t = \frac{n(\varepsilon_2^2-\varepsilon_1^2)}{2}\right) \\ &= \exp\left(-\frac{d\left(2 - \frac{n\varepsilon_1^2}{\lambda} - \frac{n\varepsilon_2^2}{\lambda}\right)^2}{16\left(\frac{d}{\lambda}+2\right)^2}\right) \\ & && \left(\text{Pulling out the factor of } \tfrac{\lambda}{2} \text{ from numerator}\right) \\ &\leq \exp\left(-\frac{d\left(1 - \frac{\varepsilon_1^2}{\varepsilon_2^2}\right)^2}{16\left(\frac{d}{n\varepsilon_2^2}+2\right)^2}\right) && (\text{by Eq. (C.1)}) \\ &\leq \exp\left(-\frac{n^2\left(\varepsilon_2^2 - \varepsilon_1^2\right)^2}{16d\left(1 + \frac{n\varepsilon_2^2}{d}\right)^2}\right) && \left(\text{Pulling out factors of } n,\ d, \text{ and } \varepsilon_2^2\right) \end{aligned}$$

$$= \exp\left(-\frac{16}{\left(1+\frac{n\varepsilon_2^2}{d}\right)^2}\right) \qquad \text{(since } n = \tfrac{16\sqrt{d}}{\varepsilon_2^2-\varepsilon_1^2}\text{)}$$

$$= \exp\left(-\frac{16}{3^2}\right) = \exp\left(-\frac{16}{9}\right) < \frac{1}{3} \qquad \text{(by Eq. (C.2))}$$

Thus, when $\|\boldsymbol{\mu}\|_2 \geq \varepsilon_2$, we have $\Pr(y_n^{(i)} \geq \tau) \geq 2/3$ and the i^{th} test outcome will be correctly a Reject with probability at least $2/3$. □

We are now ready to state the main theorem below.

Lemma 10.5 (Tolerant mean tester). *Given $\varepsilon_2 > \varepsilon_1 > 0$, $\delta \in (0,1)$, and $d \geq \left(\frac{16\varepsilon_2^2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2$, there is a tolerant tester that uses $\mathcal{O}\left(\frac{\sqrt{d}}{\varepsilon_2^2-\varepsilon_1^2}\log\left(\frac{1}{\delta}\right)\right)$ i.i.d. samples from $N(\boldsymbol{\mu}, \boldsymbol{I}_d)$ and satisfies the following two conditions:*

1. *If $\|\boldsymbol{\mu}\|_2 \leq \varepsilon_1$, then the tester outputs* Accept *with probability at least $1-\delta$.*

2. *If $\|\boldsymbol{\mu}\|_2 \geq \varepsilon_2$, then the tester outputs* Reject *with probability at least $1-\delta$.*

The tester is allowed to output Accept *or* Reject *arbitrarily when $\varepsilon_1 < \|\boldsymbol{\mu}\|_2 < \varepsilon_2$.*

Proof. Use the guarantee of Lemma C.2 on TOLERANTIGMT (Algorithm 30) with parameters $\varepsilon_1 = \varepsilon$ and $\varepsilon_2 = 2\varepsilon$. □

Tolerant testing for covariance matrix

We now give the definition of a tolerant tester for the unknown covariance matrix being equal to identity.

Definition C.3 (Tolerant testing of zero-mean Gaussian covariance matrix)**.** Fix $m \geq 1$, $d \geq 1$, $\varepsilon_2 > \varepsilon_1 > 0$, and $\delta > 0$. Suppose $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ is a hidden full rank covariance matrix and we draw m samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \sim N(\boldsymbol{0}, \boldsymbol{\Sigma})$. An algorithm ALG is said to be a $(\varepsilon_1, \varepsilon_2, \delta)$-tolerant zero-mean Gaussian covariance tester if it satisfies the following two conditions:

1. If $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F \leq \varepsilon_1$, then ALG should *Accept* with probability at least $1-\delta$

2. If $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F \geq \varepsilon_2$, then ALG should *Reject* with probability at least $1-\delta$.

ALG is allowed to decide arbitrarily when $\varepsilon_1 < \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_2 < \varepsilon_2$.

Definition C.4 (Test statistic $\mathtt{T}_n$)**.** Let $x_1, \ldots, x_n$ be n i.i.d. samples from $\sim N(\boldsymbol{0}, \boldsymbol{\Sigma})$ for an unknown $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$. For $i \neq j$, we define $h(x_i, x_j) = (x_i^\top x_j)^2 - (x_i^\top x_i + x_j^\top x_j) + d$. Then, we define $\mathtt{T}_n$ as

$$\mathtt{T}_n = \frac{2}{k(k-1)} \sum_{1\leq i<j\leq k} h(x_i, x_j)$$

It is known that the test statistic $\mathtt{T}_n$ (Definition C.4) can be used for *non-tolerant* zero-mean Gaussian covariance testing with an appropriate threshold; see [CM13]. With the following lemma, we show that $\mathtt{T}_n$ can also be used for *tolerant* zero-mean Gaussian covariance testing.

Algorithm 31 TolerantZMGCT.

```
Input: $\varepsilon_2 > \varepsilon_1 > 0$, $\delta \in (0,1)$, $m$ i.i.d. samples of $N(\mathbf{0}, \mathbf{\Sigma})$, where $\mathbf{\Sigma} \in \mathbb{R}^{d \times d}$
Output: Fail (too little samples), Accept ($\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \leq \varepsilon_1^2$), or Reject ($\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \geq \varepsilon_2^2$)
Define testing threshold $\tau = \frac{\varepsilon_2^2 + \varepsilon_1^2}{2}$
Define sample batch size $n = 3200 \cdot d \cdot \max\left\{\frac{1}{\varepsilon_1^2}, \left(\frac{\varepsilon_1^2}{\varepsilon_2^2 - \varepsilon_1^2}\right)^2, 2\left(\frac{\varepsilon_2}{\varepsilon_2^2 - \varepsilon_1^2}\right)^2\right\}$
Define number of rounds $r = \lceil \log(\frac{12}{\delta}) \rceil$ if $\lceil \log(\frac{12}{\delta}) \rceil$ is odd, otherwise define $r = 1 + \lceil \log(\frac{12}{\delta}) \rceil$
if $m < nr$ then
    return Fail
else
    for $i \in \{1, \ldots, r\}$ do
        Use an unused batch of $n$ i.i.d. samples $\boldsymbol{x}_1^{(i)}, \ldots, \boldsymbol{x}_n^{(i)} \sim N(\mathbf{0}, \mathbf{\Sigma})$
        Compute test statistic $T_n^{(i)}$ according to Definition C.4 for the $i^{th}$ test
        Define $i^{th}$ outcome $R^{(i)}$ as Accept if $T_n^{(i)} \leq \tau$ and Reject otherwise
    return majority$(R^{(1)}, \ldots, R^{(r)})$
```

Lemma C.5. *Fix $m \geq 1$, $d \geq 1$, $\varepsilon_2 > \varepsilon_1 > 0$, and $\delta > 0$. Suppose $\mathbf{\Sigma} \in \mathbb{R}^{d \times d}$ is a hidden full rank covariance matrix and we draw m i.i.d. samples $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \sim N(\mathbf{0}, \mathbf{\Sigma})$. When $d \geq \varepsilon_2^2$ and $m \geq \mathcal{O}\left(d \cdot \max\left\{\frac{1}{\varepsilon_1^2}, \left(\frac{\varepsilon_1^2}{\varepsilon_2^2 - \varepsilon_1^2}\right)^2, \left(\frac{\varepsilon_2}{\varepsilon_2^2 - \varepsilon_1^2}\right)^2\right\} \cdot \log\left(\frac{1}{\delta}\right)\right)$, TolerantZMGCT (Algorithm 31) is a $(\varepsilon_1, \varepsilon_2, \delta)$-tolerant zero-mean Gaussian covariance tester.*

To prove Lemma C.5, we first state the expectation and variance of $\mathtt{T}_n$ known from [CM13], and give an upper bound on the variance that will be useful for subsequent analysis.

Lemma C.6 ([CM13])**.** *For the test statistic T_n defined in Definition C.4, we have $\mathbb{E}(T_n) = \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2$ and $\sigma^2(T_n) = \frac{4}{n(n-1)}\left[\mathrm{Tr}^2(\mathbf{\Sigma}^2) + \mathrm{Tr}(\mathbf{\Sigma}^4)\right] + \frac{8}{n}\,\mathrm{Tr}(\mathbf{\Sigma}^2(\mathbf{\Sigma} - \boldsymbol{I}_d)^2)$.*

Lemma C.7. *Fix $d, n \geq 1$, $\mathbf{\Sigma} \in \mathbb{R}^{d \times d}$, and $b \geq 0$. If $\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 = \frac{b^2 d}{n}$, then $\|\mathbf{\Sigma}\|_F^2 \leq d \cdot \left(1 + \frac{b}{\sqrt{n}}\right)^2$.*

Proof. Since the matrices can be treated as vectors in $\mathbb{R}^{d^2}$ and then the Frobenius norm corresponds to the ℓ_2 norm, we see that

$$\|\mathbf{\Sigma}\|_F \leq \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F + \|\boldsymbol{I}_d\|_F \qquad \text{(Triangle inequality)}$$

$$\begin{aligned} &= b \cdot \sqrt{\frac{d}{n}} + \sqrt{d} && \text{(Since } \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 = \tfrac{b^2 d}{n} \text{ and } \|\boldsymbol{I}_d\|_F^2 = d\text{)} \\ &= \sqrt{d}\left(1 + \frac{b}{\sqrt{n}}\right) \end{aligned}$$

Therefore, $\|\boldsymbol{\Sigma}\|_F^2 \leq d \cdot \left(1 + \frac{b}{\sqrt{n}}\right)^2$ as desired. □

Lemma C.8. *Fix $d \geq 1$, $n \geq 2$, $\boldsymbol{\Sigma} \in \mathbb{R}^{d \times d}$, and $b \geq 0$. If $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 = \frac{b^2 d}{n}$, then for the test statistic $\mathtt{T}_n$ defined in Definition C.4, we have*

$$\sigma^2(\mathtt{T}_n) \leq \frac{64 d^2}{n^2} \cdot \left(1 + \frac{b^2}{n}\right) \cdot \left(1 + \frac{b^2}{n} + b^2\right)$$

Proof. We begin by observing two simple upper bounds for $\mathrm{Tr}(\boldsymbol{\Sigma}^4)$ and $\mathrm{Tr}(\boldsymbol{\Sigma}^2(\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2)$.

$$\mathrm{Tr}(\boldsymbol{\Sigma}^4) = \|\boldsymbol{\Sigma}^2\|_F^2 \leq \|\boldsymbol{\Sigma}\|_F^2 \cdot \|\boldsymbol{\Sigma}\|_F^2 = \|\boldsymbol{\Sigma}\|_F^4 = \mathrm{Tr}^2(\boldsymbol{\Sigma}^2) \tag{C.3}$$

Since $\boldsymbol{\Sigma}(\boldsymbol{\Sigma} - \boldsymbol{I}_d) = \boldsymbol{\Sigma}^2 - \boldsymbol{\Sigma} = (\boldsymbol{\Sigma} - \boldsymbol{I}_d)\boldsymbol{\Sigma}$, i.e. $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma} - \boldsymbol{I}_d$ commute, we have

$$\begin{aligned} \mathrm{Tr}(\boldsymbol{\Sigma}^2(\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2) &= \mathrm{Tr}((\boldsymbol{\Sigma}(\boldsymbol{\Sigma} - \boldsymbol{I}_d))^2) = \|\boldsymbol{\Sigma}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_F^2 \\ &\leq \|\boldsymbol{\Sigma}\|_F^2 \cdot \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 = \mathrm{Tr}(\boldsymbol{\Sigma}^2) \cdot \mathrm{Tr}((\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2) \end{aligned} \tag{C.4}$$

$$\begin{aligned} &\boldsymbol{\Sigma}^2(\mathtt{T}_n) \\ &= \frac{4}{n(n-1)}\left[\mathrm{Tr}^2(\boldsymbol{\Sigma}^2) + \mathrm{Tr}(\boldsymbol{\Sigma}^4)\right] + \frac{8}{n}\,\mathrm{Tr}(\boldsymbol{\Sigma}^2(\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2) && \text{(By Lemma C.6)} \\ &\leq \frac{8}{n(n-1)}\left[\mathrm{Tr}^2(\boldsymbol{\Sigma}^2) + (n-1) \cdot \mathrm{Tr}(\boldsymbol{\Sigma}^2(\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2)\right] && \text{(By Eq. (C.3))} \\ &\leq \frac{8}{n(n-1)}\left[\mathrm{Tr}^2(\boldsymbol{\Sigma}^2) + (n-1) \cdot \mathrm{Tr}(\boldsymbol{\Sigma}^2) \cdot \mathrm{Tr}((\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2)\right] && \text{(By Eq. (C.4))} \\ &= \frac{8}{n(n-1)} \cdot \mathrm{Tr}(\boldsymbol{\Sigma}^2) \cdot \left[\mathrm{Tr}(\boldsymbol{\Sigma}^2) + (n-1) \cdot \mathrm{Tr}((\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2)\right] \\ &\leq \frac{8}{n(n-1)} \cdot \mathrm{Tr}(\boldsymbol{\Sigma}^2) \cdot \left[\mathrm{Tr}(\boldsymbol{\Sigma}^2) + n \cdot \mathrm{Tr}((\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2)\right] && \text{(Since } \mathrm{Tr}((\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2) \geq 0\text{)} \\ &\leq \frac{8}{n(n-1)} \cdot d \cdot \left(1 + \frac{b}{\sqrt{n}}\right)^2 \cdot \left(d \cdot \left(1 + \frac{b}{\sqrt{n}}\right)^2 + n \cdot \mathrm{Tr}((\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2)\right) \\ & && \text{(Since } \mathrm{Tr}(\boldsymbol{\Sigma}^2) = \|\boldsymbol{\Sigma}\|_F^2 \text{ and by Lemma C.7)} \\ &= \frac{8}{n(n-1)} \cdot d \cdot \left(1 + \frac{b}{\sqrt{n}}\right)^2 \cdot \left(d \cdot \left(1 + \frac{b}{\sqrt{n}}\right)^2 + b^2 \cdot d\right) \\ & && \text{(Since } \mathrm{Tr}((\boldsymbol{\Sigma} - \boldsymbol{I}_d)^2) = \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 = \tfrac{b^2 d}{n}\text{)} \end{aligned}$$

$$
\begin{aligned}
&= \frac{8d^2}{n(n-1)} \cdot \left(1 + \frac{b}{\sqrt{n}}\right)^2 \cdot \left(\left(1 + \frac{b}{\sqrt{n}}\right)^2 + b^2\right) \\
&\leq \frac{16d^2}{n^2} \cdot \left(1 + \frac{b}{\sqrt{n}}\right)^2 \cdot \left(\left(1 + \frac{b}{\sqrt{n}}\right)^2 + b^2\right) && \text{(Since } n \geq 2\text{)} \\
&\leq \frac{64d^2}{n^2} \cdot \left(1 + \frac{b^2}{n}\right) \cdot \left(1 + \frac{b^2}{n} + b^2\right) && \text{(Since } (a+b)^2 \leq 2a^2 + 2b^2\text{)}
\end{aligned}
$$

□

Proof of Lemma C.5. Let us define $\Delta_{\varepsilon_1,\varepsilon_2} = \max\left\{\frac{1}{\varepsilon_1^2}, \left(\frac{\varepsilon_1^2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2, 2\left(\frac{\varepsilon_2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2\right\} > 0$ and suppose $\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 = \frac{b^2 d}{n}$ for some $b \geq 0$.

The total number of samples m required is $nr \in \mathcal{O}\left(d \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \log\left(\frac{1}{\delta}\right)\right)$ since TolerantZMGCT uses $n = 3200 \cdot d \cdot \Delta_{\varepsilon_1,\varepsilon_2}$ i.i.d. samples in each of the $r \in \mathcal{O}(\log(\frac{1}{\delta}))$ rounds.

For correctness, we will prove that each round $i \in \{1, \ldots, r\}$ succeeds with probability at least $2/3$. Then, by Chernoff bound, the majority outcome out of $r \geq \log(\frac{12}{\delta})$ independent tests will be correct with probability at least $1 - \delta$.

Now, fix an arbitrary round $i \in \{1, \ldots, r\}$. TolerantZMGCT uses $n = 3200 \cdot d \cdot \Delta_{\varepsilon_1,\varepsilon_2}$ i.i.d. samples to form a statistic $T_n^{(i)}$ (Definition C.4) and tests against the threshold $\tau = \frac{\varepsilon_2^2+\varepsilon_1^2}{4}$.

Case 1: $\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \leq \varepsilon_1^2$

We see that

$$
\begin{aligned}
b^2 &= \frac{n}{d} \cdot \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 && \text{(Since } \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 = \tfrac{b^2 d}{n}\text{)} \\
&= 3200 \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 && \text{(Since } n = 3200 \cdot d \cdot \Delta_{\varepsilon_1,\varepsilon_2}\text{)} \\
&\leq 3200 \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \varepsilon_1^2 && \text{(Since } \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \leq \varepsilon_1^2\text{)}
\end{aligned}
$$

and

$$
\begin{aligned}
1 + \frac{b^2}{n} &= 1 + \frac{\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2}{d} && \text{(Since } \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 = \tfrac{b^2 d}{n}\text{)} \\
&\leq 1 + \frac{\varepsilon_1^2}{d} && \text{(Since } \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \leq \varepsilon_1^2\text{)} \\
&\leq 2 && \text{(Since } d \geq \varepsilon_2^2 > \varepsilon_1^2\text{)}
\end{aligned}
$$

So,

$$
\begin{aligned}
\sigma^2(\mathtt{T}_n) &\leq \frac{64d^2}{n^2} \cdot \left(1 + \frac{b^2}{n}\right) \cdot \left(1 + \frac{b^2}{n} + b^2\right) && \text{(By Lemma C.8)} \\
&\leq \frac{64d^2}{n^2} \cdot 2 \cdot \left(2 + 3200 \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \varepsilon_1^2\right) && \text{(From above)}
\end{aligned}
$$

$$
\begin{aligned}
&= \frac{64 \cdot 2}{3200^2} \cdot \frac{1}{\Delta^2_{\varepsilon_1,\varepsilon_2}} \cdot \left(2 + 3200 \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \varepsilon_1^2\right) && \text{(Since } n = 3200 \cdot d \cdot \Delta_{\varepsilon_1,\varepsilon_2}\text{)}\\
&\leq \frac{64 \cdot 2}{3200^2} \cdot \frac{1}{\Delta^2_{\varepsilon_1,\varepsilon_2}} \cdot 3202 \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \varepsilon_1^2 && \text{(Since } \Delta_{\varepsilon_1,\varepsilon_2}\varepsilon_1^2 \geq 1\text{)}\\
&\leq \frac{64 \cdot 2 \cdot 3202}{3200^2} \cdot (\varepsilon_2^2 - \varepsilon_1^2)^2 && \text{(Since } \left(\frac{\varepsilon_1^2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2 \leq \Delta_{\varepsilon_1,\varepsilon_2}\text{)}
\end{aligned}
$$

Chebyshev's inequality then tells us that

$$
\begin{aligned}
\Pr\left(\mathtt{T}_n > \tau\right) &= \Pr\left(\mathtt{T}_n > \varepsilon_1^2 + \frac{\varepsilon_2^2 - \varepsilon_1^2}{2}\right) && \text{(Since } \tau = \tfrac{\varepsilon_2^2+\varepsilon_1^2}{2} = \varepsilon_1^2 + \tfrac{\varepsilon_2^2-\varepsilon_1^2}{2}\text{)}\\
&\leq \Pr\left(\mathtt{T}_n > \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 + \frac{\varepsilon_2^2 - \varepsilon_1^2}{2}\right) && \text{(Since } \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \leq \varepsilon_1^2\text{)}\\
&= \Pr\left(\mathtt{T}_n > \mathbb{E}[\mathtt{T}_n] + \frac{\varepsilon_2^2 - \varepsilon_1^2}{2}\right) && \text{(By Lemma C.6)}\\
&\leq \Pr\left(|\mathtt{T}_n - \mathbb{E}[\mathtt{T}_n]| > \frac{\varepsilon_2^2 - \varepsilon_1^2}{2}\right) && \text{(Adding absolute sign)}\\
&\leq \sigma^2(\mathtt{T}_n) \cdot \left(\frac{2}{\varepsilon_2^2 - \varepsilon_1^2}\right)^2 && \text{(Chebyshev's inequality)}\\
&\leq \frac{64 \cdot 2 \cdot 3202}{3200^2} \cdot (\varepsilon_2^2 - \varepsilon_1^2)^2 \cdot \frac{4}{(\varepsilon_2^2 - \varepsilon_1^2)^2} && \text{(From above)}\\
&< \frac{1}{3}
\end{aligned}
$$

Thus, when $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \leq \varepsilon_1^2$, we have $\Pr\left(\mathtt{T}_n < \tau\right) \geq 2/3$ and the i^{th} test outcome will be correctly an Accept with probability at least $2/3$.

Case 2: $\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \geq \varepsilon_2^2$

We can lower bound b^2 as follows:

$$
\begin{aligned}
b^2 &= \frac{n}{d} \cdot \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 && \text{(Since } \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 = \tfrac{b^2 d}{n}\text{)}\\
&= 3200 \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 && \text{(Since } n = 3200 \cdot d \cdot \Delta_{\varepsilon_1,\varepsilon_2}\text{)}\\
&\geq 3200 \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \varepsilon_2^2 && \text{(Since } \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \geq \varepsilon_2^2\text{)}
\end{aligned}
$$

Meanwhile, we can lower bound n as follows:

$$
\begin{aligned}
n &= 3200 \cdot d \cdot \Delta_{\varepsilon_1,\varepsilon_2} && \text{(Since } n = 3200 \cdot d \cdot \Delta_{\varepsilon_1,\varepsilon_2}\text{)}\\
&\geq 3200 \cdot \varepsilon_2^2 \cdot \Delta_{\varepsilon_1,\varepsilon_2} && \text{(Since } d \geq \varepsilon_2^2\text{)}\\
&\geq \frac{3200 \cdot \varepsilon_2^2 \cdot \Delta_{\varepsilon_1,\varepsilon_2}}{\Delta_{\varepsilon_1,\varepsilon_2} \cdot \left(\frac{\varepsilon_2^2-\varepsilon_1^2}{\varepsilon_2}\right)^2 - 1} && \text{(Since } \Delta_{\varepsilon_1,\varepsilon_2} \geq 2\left(\tfrac{\varepsilon_2}{\varepsilon_2^2-\varepsilon_1^2}\right)^2\text{)}
\end{aligned}
$$

Using these lower bounds on b^2 and n (which we color for convenience), we can conclude that $1 + \frac{b^2}{n} \leq \frac{b^2}{3200} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2$ via the following two equivalences:

$$1 + \frac{b^2}{n} \leq \frac{b^2}{3200} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 \iff b^2 \geq \frac{n}{\frac{n}{3200} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 - 1}$$

and

$$\begin{aligned} 3200 \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \varepsilon_2^2 &\geq \frac{n}{\frac{n}{3200} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 - 1} \\ &\iff n \geq \frac{3200 \cdot \Delta_{\varepsilon_1,\varepsilon_2} \cdot \varepsilon_2^2}{\Delta_{\varepsilon_1,\varepsilon_2} \cdot \varepsilon_2^2 \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 - 1} = \frac{3200 \cdot \varepsilon_2^2 \cdot \Delta_{\varepsilon_1,\varepsilon_2}}{\Delta_{\varepsilon_1,\varepsilon_2} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2} \right)^2 - 1} \end{aligned}$$

So,

$$\begin{aligned} \sigma^2(\mathtt{T}_n) &\leq \frac{64d^2}{n^2} \cdot \left(1 + \frac{b^2}{n} \right) \cdot \left(1 + \frac{b^2}{n} + b^2 \right) && \text{(By Lemma C.8)} \\ &\leq 64 \cdot 2 \cdot \frac{d^2}{n^2} \cdot \left(\frac{b^2}{3200} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 \right) \cdot \left(\frac{b^2}{3200} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 + b^2 \right) \\ & && \text{(Since } 1 + \tfrac{b^2}{n} \leq \tfrac{b^2}{3200} \cdot \left(\tfrac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 \text{)} \\ &= \frac{64 \cdot 2 \cdot 2}{3200} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 \cdot \frac{d^2}{n^2} \cdot b^4 && \text{(Since } \tfrac{1}{3200} \left(\tfrac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 \leq 1 \text{)} \\ &= \frac{64 \cdot 2 \cdot 2}{3200} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2} \right)^2 \cdot \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^4 && \text{(Since } \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 = \tfrac{b^2 d}{n} \text{)} \end{aligned}$$

Chebyshev's inequality then tells us that

$$\begin{aligned} \Pr\left(\mathtt{T}_n < \tau\right) &= \Pr\left(\mathtt{T}_n < \varepsilon_2^2 \cdot \left(1 - \frac{\varepsilon_2^2 - \varepsilon_1^2}{2\varepsilon_2^2} \right) \right) \\ & && \text{(Since } \tau = \tfrac{\varepsilon_2^2 + \varepsilon_1^2}{2} = \varepsilon_2^2 - \tfrac{\varepsilon_2^2 - \varepsilon_1^2}{2} = \varepsilon_2^2 \cdot \left(1 - \tfrac{\varepsilon_2^2 - \varepsilon_1^2}{2\varepsilon_2^2} \right) \text{)} \\ &\leq \Pr\left(\mathtt{T}_n < \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \cdot \left(1 - \frac{\varepsilon_2^2 - \varepsilon_1^2}{2\varepsilon_2^2} \right) \right) && \text{(Since } \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \geq \varepsilon_2^2 \text{)} \\ &= \Pr\left(\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 - \mathtt{T}_n > \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \cdot \frac{\varepsilon_2^2 - \varepsilon_1^2}{2\varepsilon_2^2} \right) && \text{(Rearranging)} \\ &= \Pr\left(\mathbb{E}[\mathtt{T}_n] - \mathtt{T}_n > \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \cdot \frac{\varepsilon_2^2 - \varepsilon_1^2}{2\varepsilon_2^2} \right) && \text{(By Lemma C.6)} \\ &\leq \Pr\left(|\mathbb{E}[\mathtt{T}_n] - \mathtt{T}_n| > \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2 \cdot \frac{\varepsilon_2^2 - \varepsilon_1^2}{2\varepsilon_2^2} \right) && \text{(Adding absolute sign)} \\ &\leq \sigma^2(\mathtt{T}_n) \cdot \left(\frac{1}{\|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F^2} \cdot \frac{2\varepsilon_2^2}{\varepsilon_2^2 - \varepsilon_1^2} \right)^2 && \text{(Chebyshev's inequality)} \end{aligned}$$

$$\begin{aligned}
&\leq \frac{64 \cdot 2 \cdot 2}{3200} \cdot \left(\frac{\varepsilon_2^2 - \varepsilon_1^2}{\varepsilon_2^2}\right)^2 \cdot \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^4 \cdot \left(\frac{1}{\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2} \cdot \frac{2\varepsilon_2^2}{\varepsilon_2^2 - \varepsilon_1^2}\right)^2 \\
&\hspace{22em} \text{(From above)} \\
&= \frac{64 \cdot 2 \cdot 2 \cdot 4}{3200} \\
&< \frac{1}{3}
\end{aligned}$$

Thus, when $\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F^2 \geq \varepsilon_2^2$, we have $\Pr(\mathtt{T}_n > \tau) \geq 2/3$ and the i^{th} test outcome will be correctly an Reject with probability at least $2/3$. □

Lemma 10.6 (Tolerant covariance tester). *Given $\varepsilon_2 > \varepsilon_1 > 0$, $\delta \in (0,1)$, and $d \geq \varepsilon_2^2$, there is a tolerant tester that uses $\mathcal{O}\left(d \cdot \max\left\{\frac{1}{\varepsilon_1^2}, \left(\frac{\varepsilon_2^2}{\varepsilon_2^2 - \varepsilon_1^2}\right)^2, \left(\frac{\varepsilon_2}{\varepsilon_2^2 - \varepsilon_1^2}\right)^2\right\} \log\left(\frac{1}{\delta}\right)\right)$ i.i.d. samples from $N(\mathbf{0}, \mathbf{\Sigma})$ and satisfies the following two conditions:*

1. *If $\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F \leq \varepsilon_1$, then the tester outputs* Accept *with probability at least $1 - \delta$.*

2. *If $\|\mathbf{\Sigma} - \boldsymbol{I}_d\|_F \geq \varepsilon_2$, then the tester outputs* Reject *with probability at least $1 - \delta$.*

The tester is allowed to output Accept *or* Reject *arbitrarily when $\varepsilon_1 < \|\mathbf{\Sigma} - \boldsymbol{I}_d\|_2 < \varepsilon_2$.*

Proof. Use the guarantee of Lemma C.5 on TOLERANTZMGCT (Algorithm 31) with parameters $\varepsilon_1^2 = \varepsilon^2$ and $\varepsilon_2^2 = 2\varepsilon^2$. □

C.2.2 Deferred derivation

Here, we show how to derive Eq. (10.3) from Eq. (10.2).

For any two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^d$, observe that $\|\boldsymbol{a} - \boldsymbol{b}\|_2^2 = \langle \boldsymbol{a} - \boldsymbol{b}, \boldsymbol{a} - \boldsymbol{b} \rangle = (\boldsymbol{a} - \boldsymbol{b})^\top (\boldsymbol{a} - \boldsymbol{b}) = \boldsymbol{a}^\top \boldsymbol{a} - 2\boldsymbol{a}^\top \boldsymbol{b} + \boldsymbol{b}^\top \boldsymbol{b}$, since $\boldsymbol{a}^\top \boldsymbol{b} = \boldsymbol{b}^\top \boldsymbol{a}$ is just a number. So,

$$\frac{1}{n}\sum_{i=1}^{n} \|\boldsymbol{y}_i - \widehat{\boldsymbol{\mu}}\|_2^2 = \frac{1}{n}\sum_{i=1}^{n} \left(\boldsymbol{y}_i^\top \boldsymbol{y}_i - 2\boldsymbol{y}_i^\top \widehat{\boldsymbol{\mu}} + \widehat{\boldsymbol{\mu}}^\top \widehat{\boldsymbol{\mu}}\right)$$

$$\frac{1}{n}\sum_{i=1}^{n} \|\boldsymbol{y}_i - X\boldsymbol{\mu}\|_2^2 = \frac{1}{n}\sum_{i=1}^{n} \left(\boldsymbol{y}_i^\top \boldsymbol{y}_i - 2\boldsymbol{y}_i^\top \boldsymbol{\mu} + \boldsymbol{\mu}^\top \boldsymbol{\mu}\right)$$

Therefore,

$$\begin{aligned}
\|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2^2 &= \frac{1}{n}\sum_{i=1}^{n} \|\widehat{\boldsymbol{\mu}} - \boldsymbol{\mu}\|_2^2 \\
&= \frac{1}{n}\sum_{i=1}^{n} \left(\widehat{\boldsymbol{\mu}}^\top \widehat{\boldsymbol{\mu}} - 2\boldsymbol{\mu}^\top \widehat{\boldsymbol{\mu}} + \boldsymbol{\mu}^\top \boldsymbol{\mu}\right) \\
&\leq \frac{1}{n}\sum_{i=1}^{n} \left(2\boldsymbol{y}_i^\top \widehat{\boldsymbol{\mu}} - 2\boldsymbol{y}_i^\top \boldsymbol{\mu} + \boldsymbol{\mu}^\top \boldsymbol{\mu} - 2\boldsymbol{\mu}^\top \widehat{\boldsymbol{\mu}} + \boldsymbol{\mu}^\top \boldsymbol{\mu}\right) \\
&\qquad \text{(Since Eq. (10.3) tells us that } \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \|\boldsymbol{y}_i - \widehat{\boldsymbol{\mu}}\|_2^2 \leq \tfrac{1}{n}\sum_{i=1}^{n} \|\boldsymbol{y}_i - \boldsymbol{\mu}\|_2^2\text{)}
\end{aligned}$$

$$
\begin{aligned}
&= \frac{2}{n}\sum_{i=1}^{n}\left((\boldsymbol{\mu}+\boldsymbol{g}_i)^\top(\widehat{\boldsymbol{\mu}}-\boldsymbol{\mu}) - \boldsymbol{\mu}^\top\widehat{\boldsymbol{\mu}} + \boldsymbol{\mu}^\top\boldsymbol{\mu}\right) && \text{(Since } \boldsymbol{y}_i = \boldsymbol{\mu}+\boldsymbol{g}_i\text{)}\\
&= \frac{2}{n}\sum_{i=1}^{n}\left(\boldsymbol{g}_i^\top(\widehat{\boldsymbol{\mu}}-\boldsymbol{\mu})\right)\\
&= \frac{2}{n}\sum_{i=1}^{n}\langle \boldsymbol{g}_i, \widehat{\boldsymbol{\mu}}-\boldsymbol{\mu}\rangle\\
&= \frac{2}{n}\langle \sum_{i=1}^{n}\boldsymbol{g}_i, \widehat{\boldsymbol{\mu}}-\boldsymbol{\mu}\rangle && \text{(Linearity of inner product)}
\end{aligned}
$$

establishing Eq. (10.3) as desired.

C.2.3 The adjustments for the general covariance setting

Here, we provide the deferred proofs of Lemma 10.11 and Lemma 10.12 from Section 10.5.1.

Lemma 10.11. *For any $\delta \in (0,1)$, there is an explicit preconditioning process that uses d i.i.d. samples from $N(\mathbf{0},\boldsymbol{\Sigma})$ and succeeds with probability at least $1-\delta$ in constructing a matrix $\boldsymbol{A} \in \mathbb{R}^{d\times d}$ such that $\lambda_{\min}(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}) \geq 1$. Furthermore, for any full rank PSD matrix $\widetilde{\boldsymbol{\Sigma}} \in \mathbb{R}^{d\times d}$, we have $\|(\boldsymbol{A}\widetilde{\boldsymbol{\Sigma}}\boldsymbol{A})^{-1/2}\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}(\boldsymbol{A}\widetilde{\boldsymbol{\Sigma}}\boldsymbol{A})^{-1/2} - \boldsymbol{I}_d\| = \|\widetilde{\boldsymbol{\Sigma}}^{-1/2}\boldsymbol{\Sigma}\widetilde{\boldsymbol{\Sigma}}^{-1/2} - \boldsymbol{I}_d\|$.*

Proof. Suppose $\widehat{\boldsymbol{\Sigma}} \in \mathbb{R}^{d\times d}$ be the empirical covariance constructed from $n = d$ i.i.d. samples from $N(\mathbf{0},\boldsymbol{\Sigma})$. Let $\lambda_1 \leq \ldots \leq \lambda_d$ and $\widehat{\lambda}_1 \leq \ldots \leq \widehat{\lambda}_d$ be the eigenvalues of $\boldsymbol{\Sigma}$ and $\widehat{\boldsymbol{\Sigma}}$ respectively. By Lemma 2.26, we know that:

- With probability 1, we have that $\widehat{\boldsymbol{\Sigma}}$ and $\boldsymbol{\Sigma}$ share the same eigenspace.
- With probability at least $1-\delta$, we have $\frac{\widehat{\lambda}_1}{\lambda_1} \leq 1 + c_0 \cdot \sqrt{\frac{d+\log 1/\delta}{d}}$ for some absolute constant c_0.

Let $\widehat{\boldsymbol{v}}_1,\ldots,\widehat{\boldsymbol{v}}_d$ be the eigenvectors corresponding to the eigenvalues $\widehat{\lambda}_1,\ldots,\widehat{\lambda}_d$. Define the following terms:

- $\boldsymbol{V}_{\text{small}} = \{i \in [d] : \widehat{\lambda}_i < 1\}$ and $\boldsymbol{V}_{\text{big}} = [d] \setminus \boldsymbol{V}_{\text{small}}$
- $\boldsymbol{\Pi}_{\text{small}} = \sum_{i\in \boldsymbol{V}_{\text{small}}} \widehat{\boldsymbol{v}}_i\widehat{\boldsymbol{v}}_i^\top$ and $\boldsymbol{\Pi}_{\text{big}} = \sum_{i\in \boldsymbol{V}_{\text{big}}} \widehat{\boldsymbol{v}}_i\widehat{\boldsymbol{v}}_i^\top$
- $\boldsymbol{A} = \sqrt{k}\boldsymbol{\Pi}_{\text{small}} + \boldsymbol{\Pi}_{\text{big}}$, where $k = \left(1 + c_0 \cdot \sqrt{\frac{d+\log 1/\delta}{n}}\right) \cdot \frac{1}{\widehat{\lambda}_1}$

We first argue that the smallest eigenvalue of $\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}$ is at least 1, i.e. $\lambda_{\min}(\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}) \geq 1$. To show this, it suffices to show that $\boldsymbol{u}^\top\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}\boldsymbol{u} \geq 1$ for any unit vector $\boldsymbol{u} \in \mathbb{R}^d$. By definition,

$$\boldsymbol{u}^\top\boldsymbol{A}\boldsymbol{\Sigma}\boldsymbol{A}\boldsymbol{u} = k\boldsymbol{u}^\top\boldsymbol{\Pi}_{\text{small}}\boldsymbol{\Sigma}\boldsymbol{\Pi}_{\text{small}}\boldsymbol{u} + \boldsymbol{u}^\top\boldsymbol{\Pi}_{\text{big}}\boldsymbol{\Sigma}\boldsymbol{\Pi}_{\text{big}}\boldsymbol{u}$$

since the cross terms are zero because $\boldsymbol{u}^\top \boldsymbol{\Pi}_{\text{small}} \boldsymbol{\Sigma} \boldsymbol{\Pi}_{\text{big}} \boldsymbol{u} = \boldsymbol{u}^\top \boldsymbol{\Pi}_{\text{big}} \boldsymbol{\Sigma} \boldsymbol{\Pi}_{\text{small}} \boldsymbol{u} = 0$. Now, observe that $\boldsymbol{u}^\top \boldsymbol{\Pi}_{\text{small}} \boldsymbol{\Sigma} \boldsymbol{\Pi}_{\text{small}} \boldsymbol{u} \geq \lambda_1 \cdot \|\boldsymbol{\Pi}_{\text{small}} \boldsymbol{u}\|_2^2$ and $\boldsymbol{u}^\top \boldsymbol{\Pi}_{\text{big}} \boldsymbol{\Sigma} \boldsymbol{\Pi}_{\text{big}} \boldsymbol{u} \geq \|\boldsymbol{\Pi}_{\text{big}} \boldsymbol{u}\|_2^2$. Meanwhile, by Pythagoras theorem, we know that $\|\boldsymbol{\Pi}_{\text{small}} \boldsymbol{u}\|_2^2 + \|\boldsymbol{\Pi}_{\text{big}} \boldsymbol{u}\|_2^2 = 1$. Therefore,

$$\begin{aligned} \boldsymbol{u}^\top \boldsymbol{A} \boldsymbol{\Sigma} \boldsymbol{A} \boldsymbol{u} =& k \boldsymbol{u}^\top \boldsymbol{\Pi}_{\text{small}} \boldsymbol{\Sigma} \boldsymbol{\Pi}_{\text{small}} \boldsymbol{u} + \boldsymbol{u}^\top \boldsymbol{\Pi}_{\text{big}} \boldsymbol{\Sigma} \boldsymbol{\Pi}_{\text{big}} \boldsymbol{u} \\ \geq& k \lambda_1 \cdot \|\boldsymbol{\Pi}_{\text{small}} \boldsymbol{u}\|_2^2 + \|\boldsymbol{\Pi}_{\text{big}} \boldsymbol{u}\|_2^2 \\ \geq& \left(\|\boldsymbol{\Pi}_{\text{small}} \boldsymbol{u}\|_2^2 + \|\boldsymbol{\Pi}_{\text{big}} \boldsymbol{u}\|_2^2 \right) \\ =& 1 \end{aligned}$$

where the last inequality is because $k = \left(1 + c_0 \cdot \sqrt{\frac{d + \log 1/\delta}{n}}\right) \cdot \frac{1}{\lambda_1} \geq \frac{1}{\lambda_1}$.

To complete the proof, note that for any full rank PSD matrix $\widetilde{\boldsymbol{\Sigma}} \in \mathbb{R}^{d \times d}$, we have

$$\begin{aligned} \|(\boldsymbol{A} \widetilde{\boldsymbol{\Sigma}} \boldsymbol{A})^{-1/2} \boldsymbol{A} \boldsymbol{\Sigma} \boldsymbol{A} (\boldsymbol{A} \widetilde{\boldsymbol{\Sigma}} \boldsymbol{A})^{-1/2} - \boldsymbol{I}_d\| &= \|(\boldsymbol{A} \widetilde{\boldsymbol{\Sigma}} \boldsymbol{A})^{-1} \boldsymbol{A} \boldsymbol{\Sigma} \boldsymbol{A} - \boldsymbol{I}_d\| \\ &= \|\boldsymbol{A}^{-1} \widetilde{\boldsymbol{\Sigma}}^{-1} \boldsymbol{\Sigma} \boldsymbol{A} - \boldsymbol{I}_d\| \\ &= \|\widetilde{\boldsymbol{\Sigma}}^{-1} \boldsymbol{\Sigma} \boldsymbol{A} \boldsymbol{A}^{-1} - \boldsymbol{I}_d\| \\ &= \|\widetilde{\boldsymbol{\Sigma}}^{-1} \boldsymbol{\Sigma} - \boldsymbol{I}_d\| \\ &= \|\widetilde{\boldsymbol{\Sigma}}^{-1/2} \boldsymbol{\Sigma} \widetilde{\boldsymbol{\Sigma}}^{-1/2} - \boldsymbol{I}_d\| \end{aligned}$$

□

Lemma 10.12. *Fix dimension $d \geq 2$ and group size $k \leq d$. Consider the $q = 2$ setting where $\boldsymbol{T} \in \mathbb{R}^{d \times d}$ is a matrix. Define $w = \frac{10d(d-1)\log d}{k(k-1)}$. Pick sets $\boldsymbol{B}_1, \ldots, \boldsymbol{B}_w$ each of size k uniformly at random (with replacement) from all the possible $\binom{d}{k}$ sets. With high probability in d, this is a $(q = 2, d, k, a = 1, b = \frac{30(d-1)\log d}{(k-1)})$-partitioning scheme.*

Proof. By definition, we have $|\boldsymbol{B}_1|, \ldots, |\boldsymbol{B}_w| = k$. Let us define $\mathcal{E}_{1,i,j}$ as the event that the cell (i,j) of $\boldsymbol{T}$ *never* appears in any of the submatrices $\boldsymbol{T}_{\boldsymbol{B}_1}, \ldots, \boldsymbol{T}_{\boldsymbol{B}_w}$, and $\mathcal{E}_{2,i,j}$ as the event that the cell (i,j) of $\boldsymbol{T}$ appears in strictly more than b submatrices. In the rest of this proof, our goal is to show that $\Pr[\mathcal{E}_1]$ and $\Pr[\mathcal{E}_2]$ are small, where $\mathcal{E}_1 = \cup_{(i,j) \in [d] \times [d]} \mathcal{E}_{1,i,j}$ and $\mathcal{E}_2 = \cup_{(i,j) \in [d] \times [d]} \mathcal{E}_{2,i,j}$.

Fix any two *distinct* $i, j \in [d]$. For $\ell \in [w]$, let us define $X_\ell^{i,j}$ as the indicator event that the cell (i,j) in $\boldsymbol{T}$ appears in the ℓ^{th} principal submatrix $\boldsymbol{T}_{\boldsymbol{B}_\ell}$ when $i, j \in \boldsymbol{B}_\ell$. By construction,

$$\Pr[X_\ell^{i,j} = 1] = \begin{cases} \frac{\binom{d-2}{k-2}}{\binom{d}{k}} = \frac{k(k-1)}{d(d-1)} & \text{if } i \neq j \\ \frac{\binom{d-1}{k-1}}{\binom{d}{k}} = \frac{k}{d} & \text{if } i = j \end{cases}$$

To analyze $\mathcal{E}_1$, we first consider $i, j \in [d]$ where $i \neq j$. We see that

$$\begin{aligned}\Pr[\mathcal{E}_{1,i,j}] = \prod_{\ell=1}^{w} \Pr[X_\ell^{i,j} = 0] &= \left(1 - \frac{k(k-1)}{d(d-1)}\right)^w \\ &\leq \exp\left(-\frac{wk(k-1)}{d(d-1)}\right) = \exp\left(-10 \log d\right) = \frac{1}{d^{10}}\end{aligned}$$

Meanwhile, when $i = j$,

$$\Pr[\mathcal{E}_{1,i,i}] = \prod_{\ell=1}^{w} \Pr[X_\ell^{i,i} = 0] = \left(1 - \frac{k}{d}\right)^w \leq \exp\left(-\frac{wk}{d}\right) \leq \exp\left(-10 \log d\right) = \frac{1}{d^{10}}$$

Taking union bound over $(i, j) \in [d] \times [d]$, we get

$$\Pr[\mathcal{E}_1] \leq \sum_{(i,j) \in [d] \times [d]} \Pr[\mathcal{E}_{1,i,j}] \leq \frac{d^2}{d^{10}} = \frac{1}{d^8}$$

To analyze $\mathcal{E}_2$, let us first define $Z^{i,j} = \sum_{\ell=1}^{w} X_\ell^{i,j}$ for any $i, j \in [d]$. Since the $X_\ell^{i,j}$ variables are indicators, linearity of expectations tells us that

$$\mathbb{E}[Z^{i,j}] = \sum_{\ell=1}^{w} \mathbb{E}[X_\ell^{i,j}] = \begin{cases} \sum_{\ell=1}^{w} \frac{k(k-1)}{d(d-1)} = \frac{wk(k-1)}{d(d-1)} & \text{if } i \neq j \\ \sum_{\ell=1}^{w} \frac{k}{d} = \frac{wk}{d} & \text{if } i = j \end{cases}$$

For $i \neq j$, applying Chernoff bound yields

$$\begin{aligned}\Pr[Z^{i,j} > (1+2) \cdot \mathbb{E}[Z^{i,j}]] &\leq \exp\left(-\frac{\mathbb{E}[Z^{i,j}] \cdot 2^2}{2+2}\right) \leq \exp\left(-\mathbb{E}[Z^{i,j}]\right) \\ &= \exp\left(-\frac{wk(k-1)}{d(d-1)}\right) = \exp\left(-10 \log d\right) = \frac{1}{d^{10}}\end{aligned}$$

Meanwhile, when $i = j$,

$$\begin{aligned}\Pr[Z^{i,i} > (1+2) \cdot \mathbb{E}[Z^{i,i}]] &\leq \exp\left(-\frac{\mathbb{E}[Z^{i,i}] \cdot 2^2}{2+2}\right) \leq \exp\left(-\mathbb{E}[Z^{i,i}]\right) \\ &= \exp\left(-\frac{wk}{d}\right) \leq \exp\left(-10 \log d\right) = \frac{1}{d^{10}}\end{aligned}$$

By defining

$$b = 3 \cdot \max_{i,j \in [d]} \mathbb{E}[Z^{i,j}] = \frac{3wk}{d} = \frac{30(d-1)\log d}{(k-1)} \ ,$$

we see that $\Pr[E_{2,i,j}] = \Pr[Z^{i,j} > b] \leq \Pr[Z^{i,j} > (1+2) \cdot \mathbb{E}[Z^{i,j}]] \leq \frac{1}{d^{10}}$ and $\Pr[E_{2,i,i}] = \Pr[Z^{i,j} > b] \leq \Pr[Z^{i,i} > (1+2) \cdot \mathbb{E}[Z^{i,i}]] \leq \frac{1}{d^{10}}$. Therefore, taking union bound over

$(i,j) \in [d] \times [d]$, we get

$$\Pr[\mathcal{E}_2] \leq \sum_{(i,j) \in [d] \times [d]} \Pr[\mathcal{E}_{2,i,j}] \leq \frac{d^2}{d^{10}} = \frac{1}{d^8}$$

In conclusion, this construction satisfy all 3 conditions of Definition 10.7 with high probability in d. □

Polynomial running time of Eq. (10.6)

In this section, we show that Eq. (10.6) in Lemma 10.15 can be reformulated as a semidefinite program (SDP) that is polynomial time solvable. Recall that we are given n samples $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \sim N(\mathbf{0}, \boldsymbol{\Sigma})$ under the assumption that $\|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1 \leq r$ for some $r > 0$, and Eq. (10.6) was defined as follows:

$$\widehat{\boldsymbol{\Sigma}} = \operatorname*{argmin}_{\substack{\boldsymbol{A} \in \mathbb{R}^{d \times d} \text{ is p.s.d.} \\ \|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq r \\ \lambda_{\min}(\boldsymbol{A}) \geq 1}} \sum_{i=1}^{n} \|\boldsymbol{A} - \boldsymbol{y}_i \boldsymbol{y}_i^\top\|_F^2$$

To convert our optimization problem to the standard SDP form, we “blow up” the problem dimension into some integer $n' \in \mathrm{poly}(d)$. Let m be the number of constraints and n' be the problem dimension. For symmetric matrices $\boldsymbol{C}, \boldsymbol{D}_1, \ldots, \boldsymbol{D}_m \in \mathbb{R}^{n' \times n'}$ and values $b_1, \ldots, b_m \in \mathbb{R}$, the standard form of a SDP is written as follows:

$$\begin{aligned} \min_{\boldsymbol{X} \in \mathbb{R}^{n' \times n'}} \quad & \langle \boldsymbol{C}, \boldsymbol{X} \rangle \\ \text{subject to} \quad & \langle \boldsymbol{D}_1, \boldsymbol{X} \rangle &= b_1 \\ & \vdots \\ & \langle \boldsymbol{D}_m, \boldsymbol{X} \rangle &= b_m \\ & \boldsymbol{X} &\succeq 0 \end{aligned} \tag{C.5}$$

where the inner product between two matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n' \times n'}$ is written as

$$\langle \boldsymbol{A}, \boldsymbol{B} \rangle = \sum_{i=1}^{n'} \sum_{j=1}^{n'} \boldsymbol{A}_{i,j} \boldsymbol{B}_{i,j}$$

For further expositions about SDPs, we refer readers to [VB96, BV04, Fre04, GM12]. In this section, we simply rely on the following known result to argue that our optimization problem will be polynomial time (in terms of n, d, and r) after showing how to frame Eq. (10.6) in the standard SDP form.

Theorem C.9 (Implied by [HJS$^+$22]). *Consider an SDP instance of the form Eq. (C.5). Suppose it has an optimal solution $\boldsymbol{X}^* \in \mathbb{R}^{n' \times n'}$ and any feasible solution $\boldsymbol{X} \in \mathbb{R}^{n' \times n'}$*

satisfies $\|\boldsymbol{X}\|_2 \leq R$ *for some* $R > 0$*. Then, there is an algorithm that produces* $\widehat{\boldsymbol{X}}$ *in* $\mathcal{O}(\mathrm{poly}(n, d, \log(1/\varepsilon)))$ *time such that* $\langle \boldsymbol{C}, \widehat{\boldsymbol{X}} \rangle \leq \langle \boldsymbol{C}, \boldsymbol{X}^* \rangle + \varepsilon R \cdot \|\boldsymbol{C}\|_2$.

Remark C.10. Apart from notational changes, Theorem 8.1 of [HJS+22] actually deals with the maximization problem but here we transform it to our minimization setting. They also guarantee additional bounds on the constraints with respect to $\widehat{\boldsymbol{X}}$, which we do not use.

In the following formulation, for any indices i and j, we define $\delta_{i,j} \in \{0, 1\}$ as the indicator indicating whether $i = j$. This will be useful for representation of the identity matrix.

Re-expressing the objective function

Observe that for any $i \in [n]$, we have

$$\begin{aligned} \|\boldsymbol{A} - \boldsymbol{y}_i\boldsymbol{y}_i^\top\|_F^2 &= \mathrm{Tr}\left((\boldsymbol{A} - \boldsymbol{y}_i\boldsymbol{y}_i^\top)^\top(\boldsymbol{A} - \boldsymbol{y}_i\boldsymbol{y}_i^\top)\right) \\ &= \mathrm{Tr}\left(\boldsymbol{A}^\top\boldsymbol{A}\right) - 2\,\mathrm{Tr}\left(\boldsymbol{y}_i\boldsymbol{y}_i^\top\boldsymbol{A}\right) + \mathrm{Tr}\left(\boldsymbol{y}_i\boldsymbol{y}_i^\top\boldsymbol{y}_i\boldsymbol{y}_i^\top\right) \end{aligned}$$

Since $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n \in \mathbb{R}^d$ are constants with respect to the optimization problem, we can ignore the $\mathrm{Tr}\left(\boldsymbol{y}_i\boldsymbol{y}_i^\top\boldsymbol{y}_i\boldsymbol{y}_i^\top\right)$ term and instead minimize $n\,\mathrm{Tr}\left(\boldsymbol{A}^\top\boldsymbol{A}\right) - 2\sum_{i=1}^n \mathrm{Tr}\left(\boldsymbol{y}_i\boldsymbol{y}_i^\top\boldsymbol{A}\right)$. As $\boldsymbol{A}^\top\boldsymbol{A}$ is a quadratic expression, let us define an auxiliary matrix $\boldsymbol{B} \in \mathbb{R}^{d\times d}$ which we will later enforce $\mathrm{Tr}(\boldsymbol{B}) \geq \mathrm{Tr}(\boldsymbol{A}^T\boldsymbol{A})$. Defining a symmetric matrix $\boldsymbol{Y} = \sum_{i=1}^n \boldsymbol{y}_i\boldsymbol{y}_i^\top \in \mathbb{R}^{d\times d}$, the minimization objective becomes

$$n\,\mathrm{Tr}\left(\boldsymbol{B}\right) - 2\,\mathrm{Tr}\left(\boldsymbol{Y}\boldsymbol{A}\right) = n\boldsymbol{B}_{1,1} + \ldots + n\boldsymbol{B}_{d,d} - 2\langle \boldsymbol{Y}, \boldsymbol{A} \rangle \tag{C.6}$$

Defining the variable matrix $\boldsymbol{X}$

Let $n' = 2d^2 + 3d + 2$ and let us define the SDP variable matrix $\boldsymbol{X} \in \mathbb{R}^{n'\times n'}$ as follows:

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{B} & \boldsymbol{A}^\top & & & & & \\ \boldsymbol{A} & \boldsymbol{I}_d & & & & & \\ & & \boldsymbol{A} - \boldsymbol{I}_d & & & & \\ & & & \boldsymbol{U} & & & \\ & & & & \boldsymbol{S} & & \\ & & & & & s_{\boldsymbol{U}} & \\ & & & & & & s_{\boldsymbol{B}} \end{bmatrix} \in \mathbb{R}^{n'\times n'}$$

where the empty parts of $\boldsymbol{X}$ are zero matrices of appropriate sizes, $\boldsymbol{B} \in \mathbb{R}^{d\times d}$ is an auxiliary matrix aiming to capture $\boldsymbol{A}^\top\boldsymbol{A}$, and $\boldsymbol{U}$ and $\boldsymbol{S}$ are diagonal matrices of size d^2:

$$\boldsymbol{U} = \mathrm{diag}(u_{1,1}, u_{1,2}, \ldots, u_{1,d}, \ldots, u_{d,1}, \ldots, u_{d,d}) \in \mathbb{R}^{d^2\times d^2}$$

$$\boldsymbol{S} = \operatorname{diag}(s_{1,1}, s_{1,2}, \ldots, s_{1,d}, \ldots, s_{d,1}, \ldots, s_{d,d}) \in \mathbb{R}^{d^2 \times d^2}$$

For convenience, we define

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{B} & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{I}_d \end{bmatrix} \in \mathbb{R}^{2d \times 2d}$$

so we can write

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{M} & & & & & \\ & \boldsymbol{A} - \boldsymbol{I}_d & & & & \\ & & \boldsymbol{U} & & & \\ & & & \boldsymbol{S} & & \\ & & & & s_{\boldsymbol{U}} & \\ & & & & & s_{\boldsymbol{B}} \end{bmatrix} \in \mathbb{R}^{n' \times n'} \tag{C.7}$$

In the following subsections, we explain how to ensure that submatrices in $\boldsymbol{X}$ model the desired notions and constraints on $\boldsymbol{A}$, $\boldsymbol{B}$, and so on. For instance, we will use $\boldsymbol{U}$ to enforce $\|\operatorname{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq r$ in an element-wise fashion and use $\boldsymbol{S}$ and $s_{\boldsymbol{U}}$ for slack variables to transform inequality constraints to equality ones. The slack variable $s_{\boldsymbol{B}}$ is used for upper bounding the norm of $\boldsymbol{B}$ later, so that we can argue that the feasible region is bounded.

Defining the cost matrix $\boldsymbol{C}$

To capture the objective function Eq. (C.6), let us define a symmetric cost matrix $\boldsymbol{C} \in \mathbb{R}^{n' \times n'}$ as follows:

$$\boldsymbol{C} = \begin{bmatrix} \operatorname{diag}(n, \ldots, n) & -\boldsymbol{Y} & \\ -\boldsymbol{Y} & \boldsymbol{0}_{d \times d} & \\ & & \boldsymbol{0}_{(2d^2+d+2) \times (2d^2+d+2)} \end{bmatrix} \in \mathbb{R}^{n' \times n'} \tag{C.8}$$

One can check that $\langle \boldsymbol{C}, \boldsymbol{X} \rangle = n\boldsymbol{B}_{1,1} + \ldots + n\boldsymbol{B}_{d,d} - 2\langle \boldsymbol{Y}, \boldsymbol{A} \rangle$.

Enforcing zeroes, ones, and linking $\boldsymbol{A}$ entries with $\boldsymbol{A} - \boldsymbol{I}_d$

To enforce that the empty parts of $\boldsymbol{X}$ always solves to zeroes, we can define a symmetric constraint matrix $\boldsymbol{D}_{i,j}^{zero} \in \mathbb{R}^{n' \times n'}$ such that

$$(\boldsymbol{D}_{i,j}^{zero})_{i',j'} = \begin{cases} 1 & \text{if } i' = i \text{ and } j' = j \\ 0 & \text{otherwise} \end{cases}$$

and $b^{zero}_{i,j} = 0$. Then, $\langle \boldsymbol{D}^{zero}_{i,j}, \boldsymbol{X} \rangle = b^{zero}_{i,j}$ resolves to $\boldsymbol{X}_{i,j} = \langle \boldsymbol{D}^{zero}_{i,j}, \boldsymbol{X} \rangle = b^{zero}_{i,j} = 0$. We can similarly enforce that the appropriate part of $\boldsymbol{X}$ in $\boldsymbol{M}$ resolves to $\boldsymbol{I}_d$.

Now, to ensure that the $\boldsymbol{A}$ submatrices within $\boldsymbol{M}$ are appropriately linked to $\boldsymbol{A} - \boldsymbol{I}_d$, we can define a symmetric constraint matrix $\boldsymbol{D}^{\boldsymbol{A}}_{i,j} \in \mathbb{R}^{n' \times n'}$ such that

$$\boldsymbol{D}^{\boldsymbol{A}}_{i,j} = \begin{bmatrix} \mathbf{0}_{d\times d} & * & & & & & \\ * & \mathbf{0}_{d\times d} & & & & & \\ & & \dagger & & & & \\ & & & \mathbf{0}_{d^2\times d^2} & & & \\ & & & & \mathbf{0}_{d^2\times d^2} & & \\ & & & & & 0 & \\ & & & & & & 0 \end{bmatrix} \in \mathbb{R}^{n' \times n'}$$

and $b^{\boldsymbol{B}}_{i,j} = 0$, where $*$ contains $\frac{1}{4}$ at the (i,j)-th and (j,i)-th entries and $\dagger$ contains $\delta_{i,j} - \frac{1}{2}$ at the (i,j)-th and (j,i)-th entries, with 0 everywhere else; if $i = j$, we double the value. So, $\langle \boldsymbol{D}^{\boldsymbol{A}}_{i,j}, \boldsymbol{X} \rangle = b^{\boldsymbol{A}}_{i,j}$ would enforce that the (i,j)-th and (j,i)-th entries between the $\boldsymbol{A}$ submatrices within $\boldsymbol{M}$ and those in $\boldsymbol{A} - \boldsymbol{I}_d$ are appropriately linked.

Modeling the ℓ_1 constraint

To encode $\|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq r$ in SDP form, let us define auxiliary variables $\{u_{i,j}\}_{i,j\in[d]}$ and define the linear constraints:

- $-A_{i,j} - u_{i,j} \leq -\delta_{i,j}$, for all $i,j \in [d]$
- $A_{i,j} - u_{i,j} \leq \delta_{i,j}$, for all $i,j \in [d]$
- $\sum_{i=1}^{d} \sum_{j=1}^{d} u_{i,j} \leq r$

The first two constraints effectively encode $|A_{i,j} - \delta_{i,j}| \leq u_{i,j}$ and so the third constraint captures $\|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq r$ as desired. To convert the inequality constraint to an equality one, we use the slack variables $\{s_{i,j}\}_{i,j\in[d]}$ in $\boldsymbol{S}$. For instance, we can define symmetric constraint matrices $\boldsymbol{D}^{+}_{i,j} \in \mathbb{R}^{n' \times n'}$, $\boldsymbol{D}^{-}_{i,j} \in \mathbb{R}^{n' \times n'}$, and $\boldsymbol{D}^{r}_{i,j} \in \mathbb{R}^{n' \times n'}$ with $b^{+}_{i,j} = b^{-}_{i,j} = 0$ and $b^{r} = r$ as follows:

$$\boldsymbol{D}^{+}_{i,j} = \begin{bmatrix} \mathbf{0}_{d\times d} & * & & & & & \\ * & \mathbf{0}_{d\times d} & & & & & \\ & & \mathbf{0}_{d\times d} & & & & \\ & & & \dagger & & & \\ & & & & \ddagger & & \\ & & & & & 0 & \\ & & & & & & 0 \end{bmatrix} \qquad \boldsymbol{D}^{-}_{i,j} = \begin{bmatrix} \mathbf{0}_{d\times d} & -* & & & & & \\ -* & \mathbf{0}_{d\times d} & & & & & \\ & & \mathbf{0}_{d\times d} & & & & \\ & & & \dagger & & & \\ & & & & \ddagger & & \\ & & & & & 0 & \\ & & & & & & 0 \end{bmatrix}$$

$$\boldsymbol{D}^r_{i,j} = \begin{bmatrix} \mathbf{0}_{2d\times 2d} & & & & & \\ & \mathbf{0}_{d\times d} & & & & \\ & & \mathbf{1}_{d^2\times d^2} & & & \\ & & & \mathbf{0}_{d^2\times d^2} & & \\ & & & & 1 & \\ & & & & & 0 \end{bmatrix}$$

where $*$ contains $\frac{\delta_{i,j}-1}{4}$ at the (i,j)-th and (j,i)-th entries, † contains $-\frac{1}{2}$ at the (i,j)-th and (j,i)-th entries, and ‡ contains $\frac{1}{2}$ at the (i,j)-th and (j,i)-th entries, with 0 everywhere else; if $i = j$, we double the value. So, $\langle \boldsymbol{D}^+_{i,j}, \boldsymbol{X} \rangle = b^+_{i,j}$ models $\delta_{i,j} - A_{i,j} - u_{i,j} + s_{i,j} = 0$, $\langle \boldsymbol{D}^-_{i,j}, \boldsymbol{X} \rangle = b^-_{i,j}$ models $A_{i,j} - \delta_{i,j} - u_{i,j} + s_{i,j} = 0$, and $\langle \boldsymbol{D}^r_{i,j}, \boldsymbol{X} \rangle = b^r_{i,j}$ models $s_{\boldsymbol{S}} + \sum_{i=1} \sum_{j=1} u_{i,j} = r$.

Positive semidefinite constraints

By known properties of the (generalized) Schur complement [Zha05, Section 1.4 and Section 1.6], it is known that $\boldsymbol{X} \succeq \mathbf{0}$ if and only if the following properties hold simultaneously:

1. $\boldsymbol{M} \succeq \mathbf{0}$
2. $\boldsymbol{A} - \boldsymbol{I}_d \succeq \mathbf{0} \iff \boldsymbol{A} \succeq \boldsymbol{I}_d \iff \lambda_{\min}(\boldsymbol{A}) \geq 1$, which also implies that $\boldsymbol{A}$ is psd
3. $\boldsymbol{U} \succeq \mathbf{0} \iff u_{1,1}, u_{1,2}, \ldots, u_{1,d}, \ldots, u_{d,1}, \ldots, u_{d,d} \geq 0$
4. $\boldsymbol{S} \succeq \mathbf{0} \iff s_{1,1}, s_{1,2}, \ldots, s_{1,d}, \ldots, s_{d,1}, \ldots, s_{d,d} \geq 0$
5. $s_{\boldsymbol{U}} \geq 0$
6. $s_{\boldsymbol{B}} \geq 0$

For the first property, since $\boldsymbol{I}_d \succ \mathbf{0}$, Schur complement tells us that $\boldsymbol{M} = \begin{bmatrix} \boldsymbol{B} & \boldsymbol{A}^\top \\ \boldsymbol{A} & \boldsymbol{I}_d \end{bmatrix} \succeq 0$ if and only if $\boldsymbol{B} \succeq \boldsymbol{A}^\top \boldsymbol{A}$. Observe that $\boldsymbol{B} \succeq \boldsymbol{A}^\top \boldsymbol{A}$ implies $\operatorname{Tr}(\boldsymbol{B}) \geq \operatorname{Tr}(\boldsymbol{A}^\top \boldsymbol{A})$, which aligns with our intention of modeling $\boldsymbol{A}^\top \boldsymbol{A}$ by $\boldsymbol{B}$. Note that the objective function is $n \operatorname{Tr}(\boldsymbol{B}) - 2 \operatorname{Tr}(\boldsymbol{Y}\boldsymbol{A})$ and we have that $\operatorname{Tr}(\boldsymbol{B}) \geq \operatorname{Tr}(\boldsymbol{A}^\top \boldsymbol{A})$ for all feasible matrices $\boldsymbol{B}$. Thus, for any pair $(\boldsymbol{A}^*, \boldsymbol{B}^*)$ that minimizes of the objective function, it has to be that $\operatorname{Tr}(\boldsymbol{B}^*) = \operatorname{Tr}((\boldsymbol{A}^*)^\top \boldsymbol{A}^*)$, since otherwise, the pair $(\boldsymbol{A}^*, \boldsymbol{B}^{**} = (\boldsymbol{A}^*)^\top \boldsymbol{A}^*)$ would have a smaller value.

Enforcing an upper bound on $\|\boldsymbol{B}\|_2$

To apply Theorem C.9, we need to argue that the feasible region of our SDP is bounded and non-empty, so that $\|\boldsymbol{X}\|_2$ is upper bounded. To do so, we need to enforce an upper bound on $\|\boldsymbol{B}\|_2$.

Since $\|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq r$, by triangle inequality and standard norm inequalities, we see that

$$\begin{aligned} \|\boldsymbol{A}\|_2 &\leq \|\boldsymbol{A} - \boldsymbol{I}_d\|_2 + \|\boldsymbol{I}_d\|_2 \leq \|\boldsymbol{A} - \boldsymbol{I}_d\|_F + \|\boldsymbol{I}_d\|_2 \\ &= \|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_2 + d \leq \|\mathrm{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 + d \leq r + d \end{aligned} \tag{C.9}$$

As $\boldsymbol{B}$ is supposed to model $\boldsymbol{A}^T\boldsymbol{A}$ and is constrained only by $\boldsymbol{B} \succeq \boldsymbol{A}^T\boldsymbol{A}$, it is feasible to enforce $\mathrm{Tr}(\boldsymbol{B}) \leq \|\boldsymbol{B}\|_F^2 \leq d \cdot (r+d)^4$ because

$$\|\boldsymbol{A}^T\boldsymbol{A}\|_F^2 \leq d \cdot \|\boldsymbol{A}^T\boldsymbol{A}\|_2^2 = d \cdot \|\boldsymbol{A}\|_2^4 \leq d \cdot (r+d)^4$$

To this end, let us define a symmetric constraint matrix $\boldsymbol{D}_{i,j}^{\boldsymbol{B}} \in \mathbb{R}^{n' \times n'}$ such that

$$\boldsymbol{D}^{\boldsymbol{B}} = \begin{bmatrix} \boldsymbol{I}_d & & \\ & \boldsymbol{0}_{(2d^2+2d+1)\times(2d^2+2d+1)} & \\ & & 1 \end{bmatrix} \in \mathbb{R}^{n' \times n'}$$

and $b^{\boldsymbol{B}} = d \cdot (r+d)^4$. Then, $\langle \boldsymbol{D}^{\boldsymbol{B}}, \boldsymbol{X} \rangle = b^{\boldsymbol{B}}$ resolves to $\mathrm{Tr}(\boldsymbol{B}) + s_{\boldsymbol{B}} = \langle \boldsymbol{D}^{\boldsymbol{B}}, \boldsymbol{X} \rangle = b^{\boldsymbol{B}} = d \cdot (r+d)^4$. In other words, since the slack variable $s_{\boldsymbol{B}}$ is non-negative, i.e. $s_{\boldsymbol{B}} \geq 0$, we have

$$\|\boldsymbol{B}\|_2 \leq \mathrm{Tr}(\boldsymbol{B}) \leq \|\boldsymbol{B}\|_F^2 \leq d \cdot (r+d)^4 \tag{C.10}$$

Bounding $\|\boldsymbol{C}\|_2$ and $\|\boldsymbol{X}\|_2$

Recalling the definition of $\boldsymbol{C}$ in Eq. (C.8), we see that

$$\|\boldsymbol{C}\|_2 \leq \left\| \begin{bmatrix} \mathrm{diag}(n, \ldots, n) & -\boldsymbol{Y} \\ -\boldsymbol{Y} & \boldsymbol{0}_{d \times d} \end{bmatrix} \right\|_2 \leq n + \|\boldsymbol{Y}\|_2$$

Meanwhile, we know from Lemma 2.26 that

$$\|\boldsymbol{Y}\|_2 \leq \|\boldsymbol{\Sigma}\|_2 \cdot \left(1 + \mathcal{O}\left(\sqrt{\frac{d + \log 1/\delta}{n}}\right)\right)$$

with probability at least $1 - \delta$.

Recall from Algorithm 24 that when we solve the optimization problem of Eq. (10.6), we have that $\|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I})\|_1 \leq r$. So, by a similar chain of arguments as Eq. (C.9), we see that

$$\begin{aligned} \|\boldsymbol{\Sigma}\|_2 &\leq \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_2 + \|\boldsymbol{I}_d\|_2 \leq \|\boldsymbol{\Sigma} - \boldsymbol{I}_d\|_F + \|\boldsymbol{I}_d\|_2 \\ &= \|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_2 + d \leq \|\mathrm{vec}(\boldsymbol{\Sigma} - \boldsymbol{I}_d)\|_1 + d = r + d \end{aligned}$$

Therefore,

$$\begin{aligned}\|\boldsymbol{C}\|_2 &\leq n + \|\boldsymbol{\Sigma}\|_2 \cdot \left(1 + \mathcal{O}\left(\sqrt{\frac{d + \log 1/\delta}{n}}\right)\right)\\ &\leq n + (r+d) \cdot \left(1 + \mathcal{O}\left(\sqrt{\frac{d + \log 1/\delta}{n}}\right)\right) \in \operatorname{poly}(n, d, r)\end{aligned}$$

Meanwhile, recalling definition of $\boldsymbol{X}$ from Eq. (C.7), we see that for *any* feasible solution $\boldsymbol{X}$,

$$\|\boldsymbol{X}\|_2 \leq \max\left\{\|\boldsymbol{M}\|_2, \|\boldsymbol{A} - \boldsymbol{I}_d\|_2, \|\boldsymbol{U}\|_2, \|\boldsymbol{S}\|_2, s_{\boldsymbol{U}}, s_{\boldsymbol{B}}\right\}$$

By Eq. (C.10), we have that $\|\boldsymbol{B}\|_2 \leq \sqrt{d} \cdot (r+d)^2$. So,

$$\|\boldsymbol{M}\|_2 \leq \|\boldsymbol{B}\|_2 + \|\boldsymbol{A}\|_2 + 1 \leq d \cdot (r+d)^4 + r + d + 1 \in \operatorname{poly}(d, r)$$

Also, all the remaining terms are in $\operatorname{poly}(r, d)$ since $\|\operatorname{vec}(\boldsymbol{A} - \boldsymbol{I}_d)\|_1 \leq r$. Therefore, $\|\boldsymbol{X}\|_2 \in \operatorname{poly}(d, r)$ with probability $1 - \delta$. So, $\|\boldsymbol{X}\|_2 \leq R$ for some $R \in \operatorname{poly}(d, r)$.

Putting together

Suppose we aim for an additive error of $\varepsilon' > 0$ in Eq. (10.7) when we solve Eq. (10.6). From above, we have that $\|\boldsymbol{C}\|_2, R \in \operatorname{poly}(n, d, r)$. Let us define $\varepsilon = \frac{\varepsilon'}{R \cdot \|\boldsymbol{C}\|_2}$ in Theorem C.9. Then, the algorithm of Theorem C.9 produces $\widehat{\boldsymbol{X}} \in \mathbb{R}^{n' \times n'}$ in $\operatorname{poly}(n, d, \log(1/\varepsilon)) \subseteq \operatorname{poly}(n, d, \log(\frac{R \cdot \|\boldsymbol{C}\|_2}{\varepsilon'})) \subseteq \operatorname{poly}(n, d, r, \log(1/\varepsilon'))$ time such that $\langle \boldsymbol{C}, \widehat{\boldsymbol{X}} \rangle \leq \langle \boldsymbol{C}, \boldsymbol{X}^* \rangle + \varepsilon R \cdot \|\boldsymbol{C}\|_2 = \langle \boldsymbol{C}, \boldsymbol{X}^* \rangle + \varepsilon'$ as desired.

C.3 Addendum for Chapter 11

C.3.1 Path essential graph

In this section, we explain why our algorithm (Algorithm 27) is simply the classic "binary search with prediction"[15] when the given essential graph $\mathcal{E}(\mathcal{G}^*)$ is an undirected path on n vertices. So, another way to view our result is a *generalization* that works on essential graphs of arbitrary moral DAGs.

When the given essential graph is a path $\mathcal{E}(\mathcal{G}^*)$ on n vertices, we know that there are n possible DAGs in the Markov equivalence class where each DAG corresponds to choosing a single root node and having all edges pointing away from it. Observe that a verifying set of any DAG is then simply the root node as the set of of covered edges in any rooted

[15] e.g. see https://en.wikipedia.org/wiki/Learning_augmented_algorithm#Binary_search

tree are precisely the edges incident to the root.

Therefore, given any $\widetilde{\mathcal{G}} \in [\mathcal{G}^*]$, we se that $h(\mathcal{G}^*, \widetilde{\mathcal{V}})$ measures the number of hops between the root of the advice DAG $\widetilde{\mathcal{G}}$ and the root of the true DAG $\mathcal{G}^*$. Furthermore, by Meek rule R1, whenever we intervene on a vertex U on the path, we will fully orient the "half" of the path that points away from the root while the subpath between U and the root remains unoriented (except the edge directly incident to U). So, one can see that Algorithm 27 is actually mimicking exponential search from the root of $\widetilde{\mathcal{G}}$ towards the root of $\mathcal{G}^*$. Then, once the root of $\mathcal{G}^*$ lies within the r-hop neighborhood $\mathcal{H}$, `SubsetSearch` uses $\mathcal{O}(\log |\boldsymbol{V}(\mathcal{H})|)$ interventions, which matches the number of queries required by binary search within a fixed interval over $|\boldsymbol{V}(\mathcal{H})|$ nodes.

C.3.2 Ratio of verification numbers

Lemma 11.9 (Covered edge status changes due to covered edge reversal)**.** *Let $\mathcal{G}^*$ be a moral DAG with MEC $[\mathcal{G}^*]$ and consider any DAG $\mathcal{G} \in [\mathcal{G}^*]$. Suppose $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$ has a covered edge $X \to Y \in \boldsymbol{C}(\mathcal{G})$ and we reverse $X \to Y$ to $Y \to X$ to obtain a new DAG $\mathcal{G}' \in [\mathcal{G}^*]$. Then, all of the following statements hold:*

1. *$Y \to X \in \boldsymbol{C}(\mathcal{G}')$. Note that this is the covered edge that was reversed.*

2. *If an edge E does not involve X or Y, then $E \in \boldsymbol{C}(\mathcal{G})$ if and only if $E \in \boldsymbol{C}(\mathcal{G}')$.*

3. *If $X \in \mathrm{Ch}_{\mathcal{G}}(A)$ for some $A \in \boldsymbol{V} \setminus \{X, Y\}$, then $A \to X \in \boldsymbol{C}(\mathcal{G})$ if and only if $A \to Y \in \boldsymbol{C}(\mathcal{G}')$.*

4. *If $B \in \mathrm{Ch}_{\mathcal{G}}(Y)$ and $X \to B \in \boldsymbol{E}(\mathcal{G})$ for some $B \in \boldsymbol{V} \setminus \{X, Y\}$, then $Y \to B \in \boldsymbol{C}(\mathcal{G})$ if and only if $X \to B \in \boldsymbol{C}(\mathcal{G}')$.*

Proof. The only parental relationships that changed when we reversing $X \to Y$ to $Y \to X$ are $\mathrm{Pa}_{\mathcal{G}'}(Y) = \mathrm{Pa}_{\mathcal{G}}(Y) \setminus \{X\}$ and $\mathrm{Pa}_{\mathcal{G}'}(X) = \mathrm{Pa}_{\mathcal{G}}(X) \cup \{Y\}$. For any other vertex $U \in \boldsymbol{V} \setminus \{X, Y\}$, we have $\mathrm{Pa}_{\mathcal{G}'}(U) = \mathrm{Pa}_{\mathcal{G}}(U)$.

The first two points have the same proof: as parental relationships of both endpoints are unchanged, the covered edge status is unchanged. We now prove the other two points.

3. Since $X \to Y \in \boldsymbol{C}(\mathcal{G})$ is a covered edge in $\mathcal{G}$ and $X \in \mathrm{Ch}_{\mathcal{G}}(A)$ means $A \to X \in \boldsymbol{E}(\mathcal{G})$, we must have $A \to Y \in \boldsymbol{E}(\mathcal{G})$. We prove both directions separately.

   Suppose $A \to X \in \boldsymbol{C}(\mathcal{G})$ is a covered edge in $\mathcal{G}$. Then, $\mathrm{Pa}_{\mathcal{G}}(A) = \mathrm{Pa}_{\mathcal{G}}(X) \setminus \{A\}$. Since $X \to Y \in \boldsymbol{C}(\mathcal{G})$ is a covered edge in $\mathcal{G}$, we have $\mathrm{Pa}_{\mathcal{G}}(X) = \mathrm{Pa}_{\mathcal{G}}(Y) \setminus \{X\} = \mathrm{Pa}_{\mathcal{G}'}(Y)$. Therefore, $\mathrm{Pa}_{\mathcal{G}'}(A) = \mathrm{Pa}_{\mathcal{G}}(A) = \mathrm{Pa}_{\mathcal{G}}(X) \setminus \{A\} = \mathrm{Pa}_{\mathcal{G}'}(Y) \setminus \{A\}$, and so $A \to Y \in \boldsymbol{C}(\mathcal{G}')$ is a covered edge in $\mathcal{G}$.

   Suppose $A \to X \not\in \boldsymbol{C}(\mathcal{G})$ is not a covered edge in $\mathcal{G}$. Then, one of the two cases must occur:

(a) There exists some vertex U such that $U \to A$ and $U \not\to X$ in $\mathcal{G}$.
Since $X \to Y \in \boldsymbol{C}(\mathcal{G})$ is a covered edge in $\mathcal{G}$, $U \not\to X$ implies $U \not\to Y$ in $\mathcal{G}$. Therefore, $A \to Y \not\in \boldsymbol{C}(\mathcal{G}')$ due to $U \to A$.

(b) There exists some vertex V such that $V \to X$ and $V \not\to A$ in $\mathcal{G}$.
There are two possibilities for $V \not\to A$: $V \not\mathrel{-} A$ or $V \leftarrow A$. If $V \not\mathrel{-} A$, then $V \to X \leftarrow A$ is a v-structure, but $\mathcal{G}$ is a moral DAG. If $V \leftarrow A$, then $X \not\in \mathrm{Ch}(A)$ since we have $A \to V \to X$. Both possibilities lead to contradictions.

The first case implies $A \to Y \not\in \boldsymbol{C}(\mathcal{G}')$ while the second case cannot happen.

4. We prove both directions separately.

Suppose $Y \to B \in \boldsymbol{C}(\mathcal{G})$ is a covered edge in $\mathcal{G}$. Then, $\mathrm{Pa}_{\mathcal{G}}(B) = \mathrm{Pa}_{\mathcal{G}}(Y) \cup \{Y\}$. Since $X \to Y \in \boldsymbol{C}(\mathcal{G})$ is a covered edge in $\mathcal{G}$, we have $\mathrm{Pa}_{\mathcal{G}}(X) = \mathrm{Pa}_{\mathcal{G}}(Y) \setminus \{X\}$. So, we have $\mathrm{Pa}_{\mathcal{G}'}(B) \setminus \{X\} = \mathrm{Pa}_{\mathcal{G}}(B) \setminus \{X\} = \mathrm{Pa}_{\mathcal{G}}(Y) \cup \{Y\} \setminus \{X\} = \mathrm{Pa}_{\mathcal{G}}(X) \cup \{Y\} = \mathrm{Pa}_{\mathcal{G}'}(X)$. Thus, $X \to B \in \boldsymbol{C}(\mathcal{G}')$ is a covered edge in $\mathcal{G}'$.

Suppose $Y \to B \not\in \boldsymbol{C}(\mathcal{G})$. Then, one of the two cases must occur:

- There exists some vertex $U \to Y$ and $U \not\to B$ in $\mathcal{G}$.
  Since $X \to Y \in \boldsymbol{C}(\mathcal{G})$ is a covered edge in $\mathcal{G}$, $U \to Y$ implies $U \to X$. Therefore, $X \to B \not\in \boldsymbol{C}(\mathcal{G}')$ due to $U \not\to B$.
- There exists some vertex $V \to B$ and $V \not\to Y$ in $\mathcal{G}$.
  There are two possibilities for $V \not\to Y$: $V \not\mathrel{-} Y$ or $V \leftarrow Y$. If $V \not\mathrel{-} Y$, then $V \to B \leftarrow Y$ is a v-structure, but $\mathcal{G}$ is a moral DAG. If $V \leftarrow Y$, then $B \not\in \mathrm{Ch}(Y)$ since we have $Y \to V \to B$. Both possibilities lead to contradictions.

The first case implies $X \to B \not\in \boldsymbol{C}(\mathcal{G}')$ while the second case cannot happen. □

We use the following simple lemma in our proof of Lemma 11.11.

Lemma C.11. *For any covered edge $X \to Y$ in a DAG $\mathcal{G} = (\boldsymbol{V}, \boldsymbol{E})$, we have $Y \in \mathrm{Ch}(X)$. Furthermore, each vertex only appears as an endpoint of some covered edge at most once.*

Proof. For the first statement, suppose, for a contradiction, that $Y \not\in \mathrm{Ch}(X)$. Then, there exists some $Z \in \boldsymbol{V} \setminus \{X, Y\}$ such that $Z \in \mathrm{De}(X) \cap \mathrm{An}(Y)$. Fix an arbitrary ordering π for $\mathcal{G}$ and let $Z^* = \mathrm{argmax}_{Z \in \mathrm{De}(X) \cap \mathrm{An}(Y)} \{\pi(Z)\}$. Then, we see that $Z^* \to Y$ while $Z^* \not\to X$ since $Z^* \in \mathrm{De}(X)$. So, $X \to Y$ *cannot* be a covered edge. Contradiction.

For the second statement, suppose, for a contradiction, that there are two covered edges $U \to X, V \to X \in \boldsymbol{C}(\mathcal{G})$ that ends with X. Since $U \to X \in \boldsymbol{C}(\mathcal{G})$, we must have $V \to U$. Since $V \to X \in \boldsymbol{C}(\mathcal{G})$, we must have $U \to V$. We cannot have both $U \to V$ and $V \to U$ simultaneously. Contradiction. □

Lemma 11.11 (Formal version of Lemma 11.7)**.** *Consider two moral DAGs $\mathcal{G}_1$ and $\mathcal{G}_2$ from the same MEC such that they differ only in one covered edge direction: $X \to Y \in \boldsymbol{E}(\mathcal{G}_1)$ and $Y \to X \in \boldsymbol{E}(\mathcal{G}_2)$. Let $\boldsymbol{S} \subseteq \boldsymbol{E}$ be a subset such that $X \to Y, Y \to X \not\in \boldsymbol{S}$. If X has a direct parent $A \in \boldsymbol{V}$ in $\mathcal{G}_1$, we further require $A \to X \in \boldsymbol{S}$. When $\pi_{\mathcal{G}_1}$ is an ordering for $\mathcal{G}_1$ such that $Y = \operatorname{argmin}_{Z \in \boldsymbol{V} : X \to Z \in \boldsymbol{C}(\mathcal{G}_1)}\{\pi_{\mathcal{G}_1}(Z) + n^2 \cdot \mathbb{1}_{X \to Z \in \boldsymbol{S}}\}$ with CRG maximal matching $\boldsymbol{M}_{\mathcal{G}_1, \pi_{\mathcal{G}_1}, \boldsymbol{S}}$, one can transform $\pi_{\mathcal{G}_1}$ to $\pi_{\mathcal{G}_2}$ and $\boldsymbol{M}_{\mathcal{G}_1, \pi_{\mathcal{G}_1}, \boldsymbol{S}}$ to another CRG maximal matching $\boldsymbol{M}_{\mathcal{G}_2, \pi_{\mathcal{G}_2}, \boldsymbol{S}}$ for $\boldsymbol{C}(\mathcal{G}_2)$ such that $|\boldsymbol{M}_{\mathcal{G}_1, \pi_{\mathcal{G}_1}, \boldsymbol{S}}| = |\boldsymbol{M}_{\mathcal{G}_2, \pi_{\mathcal{G}_2}, \boldsymbol{S}}|$.*

Proof. Define $U = \operatorname{argmin}_{Z \in \mathrm{Ch}_{\mathcal{G}_1}(X)}\{\pi_{\mathcal{G}_1}(Z)\}$ as the lowest ordered child of X. Note that Algorithm 26 chooses $X \to Y$ instead of $X \to U$ by definition of Y. This implies that $X \to U \in \boldsymbol{S}$ whenever $U \neq Y$.

Let us define $\pi_{\mathcal{G}_2}$ as follows:

$$\pi_{\mathcal{G}_2}(V) = \begin{cases} \pi_{\mathcal{G}_1}(X) & \text{if } V = Y \\ \pi_{\mathcal{G}_1}(U) & \text{if } V = X \\ \pi_{\mathcal{G}_1}(Y) & \text{if } V = U \\ \pi_{\mathcal{G}_1}(V) & \text{else} \end{cases}$$

Clearly, $\pi_{\mathcal{G}_1}(X) < \pi_{\mathcal{G}_1}(Y)$ and $\pi_{\mathcal{G}_2}(X) > \pi_{\mathcal{G}_2}(Y)$. Meanwhile, for any other two adjacent vertices V and V', observe that $\pi_{\mathcal{G}_1}(V) < \pi_{\mathcal{G}_1}(V') \iff \pi_{\mathcal{G}_2}(V) < \pi_{\mathcal{G}_2}(V')$ so $\pi_{\mathcal{G}_2}$ agrees with the arc orientations of $\pi_{\mathcal{G}_1}$ except for $X - Y$. See Fig. 11.3 for an illustrated example.

Define vertex B as follows:

$$B = \operatorname*{argmin}_{Z \in \boldsymbol{V} \,:\, Z \in \mathrm{De}(X) \text{ and } Y \to Z \in \boldsymbol{C}(\mathcal{G}_1)} \{\pi_{\mathcal{G}_1}(Z) + n^2 \cdot \mathbb{1}_{X \to Z \in \boldsymbol{S}}\}$$

If vertex B exists, then we know that $B \in \mathrm{Ch}_{\mathcal{G}_1}(Y)$ and $X \to B \in \boldsymbol{C}(\mathcal{G}_2)$ by Lemma C.11 and Lemma 11.9. By minimality of B, Definition 11.10 will choose $Y \to B$ if picking a covered edge starting with Y for $M_{\mathcal{G}_1, \pi_{\mathcal{G}_1}, \boldsymbol{S}}$. So, we can equivalently define vertex B as follows:

$$B = \operatorname*{argmin}_{Z \in \boldsymbol{V} \,:\, Z \in \mathrm{De}(Y) \text{ and } X \to Z \in \boldsymbol{C}(\mathcal{G}_2)} \{\pi_{\mathcal{G}_2}(Z) + n^2 \cdot \mathbb{1}_{X \to Z \in \boldsymbol{S}}\}$$

By choice of $\pi_{\mathcal{G}_2}$, Definition 11.10 will choose $X \to B$ if picking a covered edge starting with X for $M_{\mathcal{G}_2, \pi_{\mathcal{G}_2}, \boldsymbol{S}}$.

We will now construct a same-sized maximal matching $M_{\mathcal{G}_2, \pi_{\mathcal{G}_2}, \boldsymbol{S}}$ from $M_{\mathcal{G}_1, \pi_{\mathcal{G}_1}, \boldsymbol{S}}$ (Step 1), argue that it is maximal matching of $\boldsymbol{C}(\mathcal{G}_2)$ (Step 2), and that it is indeed a conditional-root-greedy matching for $\boldsymbol{C}(\mathcal{G}_2)$ with respect to $\pi_{\mathcal{G}_2}$ and $\boldsymbol{S}$ (Step 3). There are three cases that cover all possibilities:

Case 1 Vertex A exists, $A \to X \in M_{\mathcal{G}_1, \pi_{\mathcal{G}_1}, \boldsymbol{S}}$, and vertex B exists.

Case 2 Vertex A exists, $A \to X \in M_{\mathcal{G}_1, \pi_{\mathcal{G}_1}, \boldsymbol{S}}$, and vertex B does not exist.

Case 3 $A \to X \notin M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$.

This could be due to vertex A not existing, or $A \to X \notin \boldsymbol{C}(\mathcal{G}_1)$, or $M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$ containing a covered edge ending at A so $A \to X$ was removed from consideration.

Step 1: Construction of $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ such that $|M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}| = |M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}|$.

By Lemma 11.9, covered edge statuses of edges whose endpoints do not involve X or Y will remain unchanged. By definition of Y, we know that Definition 11.10 will choose $X \to Y$ if picking a covered edge starting with X for $M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$.

Since $A \to X \in M_{\mathcal{G}_1,\pi_{\mathcal{G}_1}}$ in cases 1 and 2, we know that there is no arcs of the form $X \to \cdot$ in $M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$. Since there is no arc of the form $\cdot \to X$ in $M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$ in case 3, we know that $X \to Y \in M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$.

Case 1 Define $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}} = M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}} \cup \{A \to Y, X \to B\} \setminus \{A \to X, Y \to B\}$.

Case 2 Define $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}} = M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}} \cup \{A \to Y\} \setminus \{A \to X\}$.

Case 3 Define $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}} = M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}} \cup \{Y \to X\} \setminus \{X \to Y\}$.

By construction, we see that $|M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}| = |M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}|$.

Step 2: $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is a maximal matching of the covered edge $\boldsymbol{C}(\mathcal{G}_2)$ of $\mathcal{G}_2$.

To prove that $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is a maximal matching of $\boldsymbol{C}(\mathcal{G}_2)$, we argue in three steps:

2(i) Edges of $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ belong to $\boldsymbol{C}(\mathcal{G}_2)$.

2(ii) $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is a matching of $\boldsymbol{C}(\mathcal{G}_2)$.

2(iii) $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is maximal matching of $\boldsymbol{C}(\mathcal{G}_2)$.

Step 2(i): Edges of $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ belong to $\boldsymbol{C}(\mathcal{G}_2)$.

By Lemma 11.9, covered edge statuses of edges whose endpoints do not involve X or Y will remain unchanged. Since $M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$ is a matching, it has at most one edge E involving endpoint X and at most one edge E' involving endpoint Y (E' could be E).

Case 1 Since B exists, the edges in $M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$ with endpoints involving $\{X, Y\}$ are $A \to X$ and $Y \to B$. By Lemma 11.9, we know that $A \to Y, X \to B \in \boldsymbol{C}(\mathcal{G}_2)$.

Case 2 Since B does not exist, the only edge in $M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$ with endpoints involving $\{X, Y\}$ is $A \to X$. By Lemma 11.9, we know that $A \to Y \in \boldsymbol{C}(\mathcal{G}_2)$.

Case 3 Since $A \to X \notin M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$, we have $X \to Y \in M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$ by minimality of Y.

In all cases, we see that $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}} \subseteq \boldsymbol{C}(\mathcal{G}_2)$.

Step 2(ii): $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is a matching of $\boldsymbol{C}(\mathcal{G}_2)$.

It suffices to argue that there are *no* two edges in $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ sharing an endpoint. Since $M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$ is a matching, this can only happen via newly added endpoints in $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$.

Case 1 The endpoints of newly added edges are exactly the endpoints of removed edges.

Case 2 Since we removed $A \to X$ and added $A \to Y$, it suffices to check that there are no edges in $M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}}$ involving Y. This is true since B does not exist in Case 2.

Case 3 The endpoints of newly added edges are exactly the endpoints of removed edges.

Therefore, we conclude that $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is a matching of $\boldsymbol{C}(\mathcal{G}_2)$.

Step 2(iii): $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is a maximal matching of $\boldsymbol{C}(\mathcal{G}_2)$.

For any $U \to V \in \boldsymbol{C}(\mathcal{G}_2)$, we show that there is some edge in $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ with at least one of U or V is an endpoint. By Lemma 11.9, covered edge statuses of edges whose endpoints do not involve X or Y will remain unchanged, so it suffices to consider $|\{U, V\} \cap \{X, Y\}| \geq 1$.

We check the following 3 scenarios corresponding to $|\{U, V\} \cap \{X, Y\}| \geq 1$ below:

(i) $Y \in \{U, V\}$.

The endpoints of $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ always contains Y.

(ii) $Y \not\in \{U, V\}$ and $X \to V \in \boldsymbol{C}(\mathcal{G}_2)$, for some $V \in \boldsymbol{V} \setminus \{X, Y\}$.

Since $X \to V \in \boldsymbol{C}(\mathcal{G}_2)$ and $Y \to X$ in $\mathcal{G}_2$, it must be the case that $Y \to V$ in $\mathcal{G}_2$. Since $\mathcal{G}_1$ and $\mathcal{G}_2$ agrees on all arcs except $X - Y$, we have that $Y \to V$ in $\mathcal{G}_1$ as well. Since $X \to V \in \boldsymbol{C}(\mathcal{G}_2)$, we know that $V \in \mathrm{Ch}_{\mathcal{G}_2}(X)$ via Lemma C.11. So, we have $Y \to V \in \boldsymbol{C}(\mathcal{G}_1)$ via Lemma 11.9. Since the set $\{V : Y \to V \in \boldsymbol{C}(\mathcal{G}_1)\}$ is non-empty, vertex B exists. In both cases 1 and 3, the endpoints of $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ includes X.

(iii) $Y \not\in \{U, V\}$ and $U \to X \in \boldsymbol{C}(\mathcal{G}_2)$, for some $U \in \boldsymbol{V} \setminus \{X, Y\}$.

By Lemma C.11, we know that $X \in \mathrm{Ch}_{\mathcal{G}_2}(U)$. Meanwhile, since $Y \to X \in \boldsymbol{C}(\mathcal{G}_2)$, we must have $U \to Y$ in $\mathcal{G}_2$. However, this implies that $X \not\in \mathrm{Ch}_{\mathcal{G}_2}(U)$ since $U \to Y \to X$ exists. This is a contradiction, so this situation cannot happen.

As the above argument holds for any $U \to V \in \boldsymbol{C}(\mathcal{G}_2)$, we see that $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is maximal matching for $\boldsymbol{C}(\mathcal{G}_2)$.

Step 3: $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is a conditional-root-greedy maximal matching.

We now compare the execution of Algorithm 26 on $(\pi_{\mathcal{G}_1}, \boldsymbol{S})$ and $(\pi_{\mathcal{G}_2}, \boldsymbol{S})$. Note that $\boldsymbol{S}$ remains unchanged. We know the following:

- Since $\pi_{\mathcal{G}_2}(Y) = \pi_{\mathcal{G}_1}(X)$ and $A \to X \in \boldsymbol{S}$, if A exists and $A \to X$ is chosen by Algorithm 26 on $(\pi_{\mathcal{G}_1}, \boldsymbol{S})$, then it means that there are *no* $A \to V$ arc in $\boldsymbol{C}(\mathcal{G}_1)$ such that $A \to V \not\in \boldsymbol{S}$. So, $A \to Y$ will be chosen by Algorithm 26 on $(\pi_{\mathcal{G}_2}, \boldsymbol{S})$ if A exists.

- Since $\pi_{\mathcal{G}_2}(Y) = \pi_{\mathcal{G}_1}(X)$, X is chosen as a root by Algorithm 26 on $(\pi_{\mathcal{G}_1}, \boldsymbol{S})$ if and only if Y is chosen as a root by Algorithm 26 on $(\pi_{\mathcal{G}_2}, \boldsymbol{S})$.

- By definition of B, if it exists, then $Y \to B \in M_{\mathcal{G}_1,\pi_{\mathcal{G}_1},\boldsymbol{S}} \iff X \to B \in M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$.

- By the definition of $\pi_{\mathcal{G}_2}$, we see that Algorithm 26 makes the "same decisions" when choosing arcs rooted on $V \setminus \{A, X, Y, B\}$.

Therefore, $M_{\mathcal{G}_2,\pi_{\mathcal{G}_2},\boldsymbol{S}}$ is indeed a conditional-root-greedy maximal matching for $\boldsymbol{C}(\mathcal{G}_2)$ with respect to $\pi_{\mathcal{G}_2}$ and $\boldsymbol{S}$. □

Appendix D

List of work done during Ph.D.

The following lists work done by the author during his Ph.D., starting from August 2021.

1. [LCL25] Jia Peng Lim, Davin Choo, Hady W. Lauw. *A partition cover approach to tokenization*, 2025. Under submission. Preprint available at https://arxiv.org/abs/2501.06246.

2. [BCGJG24] Arnab Bhattacharyya, Davin Choo, Philips George John, Themistoklis Gouleakis. *Learning multivariate Gaussians with imperfect advice*, 2024. Under submission. Preprint available at https://arxiv.org/abs/2411.12700.

3. [CSBS25] Davin Choo, Chandler Squires, Arnab Bhattacharyya, David Sontag. *Probably approximately correct high-dimensional causal effect estimation given a valid adjustment set.* Conference on Causal Learning and Reasoning (CLeaR), 2025.

4. [CL24] Davin Choo, Chun Kai Ling. *A short note about the learning-augmented secretary problem*, 2024. Preprint available at https://arxiv.org/abs/2410.06583.

5. [BCGM25] Arnab Bhattacharyya, Davin Choo, Sutanu Gayen, Dimitrios Myrisiotis. *Learnability of Parameter-Bounded Bayes Nets.* AAAI Conference on Artificial Intelligence (AAAI), 2025. Also presented at ICML workshop Structured Probabilistic Inference & Generative Modeling (SPIGM), 2024.

6. [CGLB24] Davin Choo, Themistoklis Gouleakis, Chun Kai Ling, Arnab Bhattacharyya. *Online bipartite matching with imperfect advice.* International Conference on Machine Learning (ICML), 2024.

7. [CLS$^+$24] Davin Choo, Yan Hao Ling, Warut Suksompong, Nicholas Teh, Jian Zhang. *Envy-free house allocation with minimum subsidy.* Operations Research Letters (ORL), 2024.

8. [CSBS25] Davin Choo, Kirankumar Shiragur, Caroline Uhler. *Causal discovery under off-target interventions.* International Conference on Artificial Intelligence and Statistics (AISTATS), 2024.

9. [CYBC24] Davin Choo, Joy Qiping Yang, Arnab Bhattacharyya, Clément L. Canonne. *Learning bounded degree polytrees with samples.* International Conference on Algorithmic Learning Theory (ALT), 2024.

10. [BCR24] Simina Brânzei, Davin Choo, Nicholas Recker. *The Sharp Power Law of Local Search on Expanders.* Symposium on Discrete Algorithms (SODA), 2024.

11. [DDKC23] Yuval Dagan, Constantinos Daskalakis, Anthimos-Vardis Kandiros, Davin Choo. *Learning and Testing Latent-Tree Ising Models Efficiently.* Conference on Learning Theory (COLT), 2023.

12. [CS23a] Davin Choo, Kirankumar Shiragur. *Adaptivity Complexity for Causal Graph Discovery.* Uncertainty in Artificial Intelligence (UAI), 2023.

13. [CS23b] Davin Choo, Kirankumar Shiragur. *New metrics and search algorithms for weighted causal DAGs.* International Conference on Machine Learning (ICML), 2023.

14. [CGB23] Davin Choo, Themistoklis Gouleakis, Arnab Bhattacharyya. *Active causal structure learning with advice.* International Conference on Machine Learning (ICML), 2023.

15. [CS23c] Davin Choo, Kirankumar Shiragur. *Subset verification and search algorithms for causal DAGs.* Artificial Intelligence and Statistics (AISTATS), 2023.

16. [CSB22] Davin Choo, Kirankumar Shiragur, Arnab Bhattacharyya. *Verification and search algorithms for causal DAGs.* Conference on Neural Information Processing Systems (NeurIPS), 2022.

17. [BCG$^+$22] Arnab Bhattacharyya, Davin Choo, Rishikesh Gajjala, Sutanu Gayen, Yuhao Wang. *Learning Sparse Fixed-Structure Gaussian Bayesian Networks.* Artificial Intelligence and Statistics (AISTATS), 2022.

Bibliography

[AAZ19] Bryon Aragam, Arash Amini, and Qing Zhou. Globally optimal score-based learning of directed acyclic graphs in high-dimensions. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4450–4462, 2019. 71

[ABDH$^+$20] Hassan Ashtiani, Shai Ben-David, Nicholas J. A. Harvey, Christopher Liaw, Abbas Mehrabian, and Yaniv Plan. Near-optimal Sample Complexity Bounds for Robust Learning of Gaussian Mixtures via Compression Schemes. *Journal of the ACM (JACM)*, 67(6):1–42, 2020. 4, 19, 20, 44

[ABDK18] Jayadev Acharya, Arnab Bhattacharyya, Constantinos Daskalakis, and Saravanan Kandasamy. Learning and Testing Causal Models with Interventions. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9469–9481, 2018. 71, 132

[ABG$^+$22] Priyank Agrawal, Eric Balkanski, Vasilis Gkatzelis, Tingting Ou, and Xizhi Tan. Learning-Augmented Mechanism Design: Leveraging Predictions for Facility Location. In *ACM Conference on Economics and Computation (EC)*, pages 497–528. Association for Computing Machinery (ACM), 2022. 36

[ABGLP19] Martin Arjovsky, Léon Bottou, Ishaan Gulrajani, and David Lopez-Paz. Invariant Risk Minimization. *arXiv preprint arXiv:1907.02893*, 2019. 5, 75

[ACE$^+$23] Antonios Antoniadis, Christian Coester, Marek Eliáš, Adam Polak, and Bertrand Simon. Online Metric Algorithms with Untrusted Predictions. *ACM Transactions on Algorithms*, 19(2):1–34, 2023. 36

[ACI22] Anders Aamand, Justin Chen, and Piotr Indyk. (Optimal) Online Bipartite Matching with Degree Information. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5724–5737, 2022. 146, 204

[ADJ$^+$20] Spyros Angelopoulos, Christoph Dürr, Shendan Jin, Shahin Kamali, and Marc Renault. Online Computation with Untrusted Advice. In *Innovations* *in Theoretical Computer Science Conference (ITCS)*, pages 52:1–52:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. 36

[AGKK23] Antonios Antoniadis, Themis Gouleakis, Pieter Kleer, and Pavel Kolev. Secretary and online matching problems with machine learned advice. *Discrete Optimization*, 48(2), 2023. 36, 146, 204

[AGU72] Alfred V. Aho, Michael R. Garey, and Jeffrey D. Ullman. The Transitive Reduction of a Directed Graph. *Society for Industrial and Applied Mathematics (SIAM) Journal on Computing*, 1(2):131–137, 1972. 98

[AGZ19] Bryon Aragam, Jiaying Gu, and Qing Zhou. Learning Large-Scale Bayesian Networks with the sparsebn Package. *Journal of Statistical Software*, 91(11):1–38, 2019. 39

[AIW18] Susan Athey, Guido W. Imbens, and Stefan Wager. Approximate Residual Balancing: Debiased Inference of Average Treatment Effects in High Dimensions. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 80(4):597–623, 2018. 141

[AJS22] Antonios Antoniadis, Peyman Jabbarzade, and Golnoosh Shahkarami. A Novel Prediction Setup for Online Speed-Scaling. In *Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, pages 9:1–9:20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. 36

[AKN06] Pieter Abbeel, Daphne Koller, and Andrew Y. Ng. Learning Factor Graphs in Polynomial Time and Sample Complexity. *Journal of Machine Learning Research (JMLR)*, 7:1743–1788, 2006. 71

[AMK21] Mohammad Ali Alomrani, Reza Moravej, and Elias Boutros Khalil. Deep Policies for Online Bipartite Matching: A Reinforcement Learning Approach. *Transactions on Machine Learning Research*, 2021. 247

[AMP97] Steen A. Andersson, David Madigan, and Michael D. Perlman. A characterization of Markov equivalence classes for acyclic digraphs. *The Annals of Statistics*, 25(2):505–541, 1997. 29, 82

[Arj20] Martin Arjovsky. *Out of Distribution Generalization in Machine Learning*. PhD thesis, New York University, 2020. 5

[ASC20] Bryan Andrews, Peter Spirtes, and Gregory F. Cooper. On the Completeness of Causal Discovery in the Presence of Latent Confounding with Tiered

Background Knowledge. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 4002–4011. Proceedings of Machine Learning Research (PMLR), 2020. 188

[AST90] Noga Alon, Paul Seymour, and Robin Thomas. A Separator Theorem for Nonplanar Graphs. *Journal of the American Mathematical Society*, 3(4):801–808, 1990. 28

[AST⁺10a] Constantin F. Aliferis, Alexander Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D. Koutsoukos. Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part I: Algorithms and Empirical Evaluation. *Journal of Machine Learning Research (JMLR)*, 11:171–234, 2010. 75, 139

[AST⁺10b] Constantin F. Aliferis, Alexander Statnikov, Ioannis Tsamardinos, Subramani Mani, and Xenofon D. Koutsoukos. Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part II: Analysis and Extensions. *Journal of Machine Learning Research (JMLR)*, 11(7):235–284, 2010. 139

[ATS03] Constantin F. Aliferis, Ioannis Tsamardinos, and Alexander Statnikov. HITON: a novel Markov Blanket algorithm for optimal variable selection. In *AMIA Annual Symposium proceedings*, pages 21–25. American Medical Informatics Association (AMIA), 2003. 75

[AZ15] Bryon Aragam and Qing Zhou. Concave Penalized Estimation of Sparse Gaussian Bayesian Networks. *Journal of Machine Learning Research (JMLR)*, 16(69):2273–2328, 2015. 39

[Bal20] Maria-Florina Balcan. Data-Driven Algorithm Design. In *Beyond Worst Case Analysis of Algorithms*, pages 626–645. Cambridge University Press, 2020. 7

[BCD20] Johannes Brustle, Yang Cai, and Constantinos Daskalakis. Multi-Item Mechanisms without Item-Independence: Learnability via Robustness. In *ACM Conference on Economics and Computation (EC)*, pages 715–761. Association for Computing Machinery (ACM), 2020. 4, 72

[BCG⁺22] Arnab Bhattacharyya, Davin Choo, Rishikesh Gajjala, Sutanu Gayen, and Yuhao Wang. Learning Sparse Fixed-Structure Gaussian Bayesian Networks. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 9400–9429, 2022. 3, 4, 43, 69, 203, 280

[BCGJG24] Arnab Bhattacharyya, Davin Choo, Philips George John, and Themistoklis Gouleakis. Learning multivariate Gaussians with imperfect advice. *arXiv preprint arXiv:2411.12700*, 2024. 3, 166, 279

[BCGM25] Arnab Bhattacharyya, Davin Choo, Sutanu Gayen, and Dimitrios Myrisiotis. Learnability of Parameter-Bounded Bayes Nets. In *AAAI Conference on Artificial Intelligence (AAAI)*, 2025. 3, 72, 279

[BCH14] Alexandre Belloni, Victor Chernozhukov, and Christian Hansen. Inference on Treatment Effects after Selection among High-Dimensional Controls. *The Review of Economic Studies*, 81(2):608–650, 2014. 141

[BCII22] Elias Bareinboim, Juan D. Correa, Duligur Ibeling, and Thomas Icard. On Pearl's Hierarchy and the Foundations of Causal Inference. In *Probabilistic and Causal Inference: The Works of Judea Pearl*, page 507–556. Association for Computing Machinery (ACM), 2022. 2

[BCR24] Simina Branzei, Davin Choo, and Nicholas Recker. The Sharp Power Law of Local Search on Expanders. In *Symposium on Discrete Algorithms (SODA)*, pages 1792–1809, 2024. 280

[BFPM21] Stephan Bongers, Patrick Forré, Jonas Peters, and Joris M. Mooij. Foundations of structural causal models with cycles and latent variables. *The Annals of Statistics*, 49(5):2885–2915, 2021. 5

[BGGJ$^+$24] Arnab Bhattacharyya, Sutanu Gayen, Philips George John, Sayantan Sen, and N. V. Vinodchandran. Distribution Learning Meets Graph Structure Sampling. *arXiv preprint arXiv:2405.07914*, 2024. 7, 69, 203

[BGK$^+$22] Arnab Bhattacharyya, Sutanu Gayen, Saravanan Kandasamy, Vedant Raval, and N. V. Vinodchandran. Efficient Interventional Distribution Learning in the PAC Framework. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 7531–7549. Proceedings of Machine Learning Research (PMLR), 2022. 134

[BGMV20] Arnab Bhattacharyya, Sutanu Gayen, Kuldeep S Meel, and N. V. Vinodchandran. Efficient Distance Approximation for Structured High-Dimensional Distributions via Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 14699–14711, 2020. 58, 64, 70

[BGP$^+$23] Arnab Bhattacharyya, Sutanu Gayen, Eric Price, Vincent Y. F. Tan, and N. V. Vinodchandran. Near-Optimal Learning of Tree-Structured Distributions by Chow and Liu. *Society for Industrial and Applied Mathematics (SIAM)*

Journal on Computing, 52(3):761–793, 2023. 4, 25, 55, 56, 57, 58, 60, 64, 67, 70, 211, 212

[BH20] Adarsh Bank and Jean Honorio. Provable Efficient Skeleton Learning of Encodable Discrete Bayes Nets in Poly-Time and Sample Complexity. In *IEEE International Symposium on Information Theory (ISIT)*, pages 2486–2491. IEEE, 2020. 64

[BKP20] Allan Borodin, Christodoulos Karavasilis, and Denis Pankratov. An Experimental Study of Algorithms for Online Bipartite Matching. *ACM Journal of Experimental Algorithmics (JEA)*, 25:1–37, 2020. 147

[BLMS$^+$22] Giulia Bernardini, Alexander Lindermayr, Alberto Marchetti-Spaccamela, Nicole Megow, Leen Stougie, and Michelle Sweering. A Universal Error Measure for Input Predictions Applied to Online Graph Problems. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3178–3190, 2022. 36

[BM08] Benjamin Birnbaum and Claire Mathieu. On-line bipartite matching made simple. *ACM SIGACT News*, 39(1):80–87, 2008. 145

[BMRS20] Etienne Bamas, Andreas Maggiori, Lars Rohwedder, and Ola Svensson. Learning Augmented Energy Minimization via Speed Scaling. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 15350–15359, 2020. 36

[BMS20] Etienne Bamas, Andreas Maggiori, and Ola Svensson. The Primal-Dual method for Learning Augmented Algorithms. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 20083–20094, 2020. 36

[Box79] George E. P. Box. Robustness in the Strategy of Scientific Model Building. In *Robustness in Statistics*, pages 201–236. Elsevier, 1979. 55

[BP93] Jean R. S. Blair and Barry Peyton. An Introduction to Chordal Graphs and Clique Trees. In *Graph Theory and Sparse Matrix Computation*, pages 1–29. Springer, 1993. 27

[BSSX20] Brian Brubach, Karthik Abinav Sankararaman, Aravind Srinivasan, and Pan Xu. Online Stochastic Matching: New Algorithms and Bounds. *Algorithmica*, 82(10):2737–2783, 2020. 149

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004. 266

[BWZ19] Jelena Bradic, Stefan Wager, and Yinchu Zhu. Sparsity Double Robust Inference of Average Treatment Effects. *arXiv preprint arXiv:1905.00744*, 2019. 141

[BY02] Ziv Bar-Yossef. *The complexity of massive data set computations*. PhD thesis, University of California at Berkeley, 2002. 67

[Can19] Clément L. Canonne. A short note on Poisson tail bounds, 2019. 23

[Can20a] Clément L Canonne. A short note on learning discrete distributions. *arXiv preprint arXiv:2002.11457*, 2020. 25

[Can20b] Clément L. Canonne. *A Survey on Distribution Testing: Your Data is Big. But is it Blue?* Number 9 in Graduate Surveys. Theory of Computing Library, 2020. 24, 141

[Can22] Clément L. Canonne. Topics and Techniques in Distribution Testing: A Biased but Representative Sample. *Foundations and Trends® in Communications and Information Theory*, 19(6):1032–1198, 2022. 23, 24

[CCD$^+$18] Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal*, 21(1):C1–C68, 2018. 141

[CCF$^+$24] José Correa, Andrés Cristi, Laurent Feuilloley, Tim Oosterwijk, and Alexandros Tsigonias-Dimitriadis. The Secretary Problem with Independent Sampling. *Management Science*, 2024. 203

[CDKS17] Clément L. Canonne, Ilias Diakonikolas, Daniel M. Kane, and Alistair Stewart. Testing bayesian networks. In *Conference on Learning Theory (COLT)*, pages 370–448. Proceedings of Machine Learning Research (PMLR), 2017. 70, 71

[CDKS18] Clément L Canonne, Ilias Diakonikolas, Daniel M. Kane, and Alistair Stewart. Testing Conditional Independence of Discrete Distributions. In *Symposium on Theory of Computing (STOC)*, pages 735–748. Association for Computing Machinery (ACM), 2018. 25

[CDW19] Wenyu Chen, Mathias Drton, and Y. Samuel Wang. On causal discovery with an equal-variance assumption. *Biometrika*, 106(4):973–980, 2019. 39

[CGB23] Davin Choo, Themis Gouleakis, and Arnab Bhattacharyya. Active causal structure learning with advice. In *International Conference on Machine Learning (ICML)*, pages 5838–5867, 2023. 3, 7, 135, 190, 203, 280

[CGG01] Mary Cryan, Leslie Ann Goldberg, and Paul W. Goldberg. Evolutionary Trees Can be Learned in Polynomial Time in the Two-State General Markov Model. *Society for Industrial and Applied Mathematics (SIAM) Journal on Computing*, pages 375–397, 2001. 71

[CGLB24] Davin Choo, Themistoklis Gouleakis, Chun Kai Ling, and Arnab Bhattacharyya. Online bipartite matching with imperfect advice. In *International Conference on Machine Learning (ICML)*, pages 8762–8781, 2024. 3, 7, 203, 206, 279

[CGM24] Ting-Hsuan Chang, Zijian Guo, and Daniel Malinsky. Post-selection inference for causal effects after causal discovery. *arXiv preprint arXiv:2405.06763*, 2024. 139

[Chi95] David Maxwell Chickering. A Transformational Characterization of Equivalent Bayesian Network Structures. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 87–98. Morgan Kaufmann, 1995. 27, 30, 192

[Chi96] David Maxwell Chickering. Learning Bayesian Networks is NP-Complete. In *Learning from Data: Artificial Intelligence and Statistics V*, pages 121–130. Springer, 1996. 72

[Chi03] David Maxwell Chickering. Optimal Structure Identification With Greedy Search. *Journal of Machine Learning Research (JMLR)*, 3:507–554, 2003. 71, 135

[CHM04] David Maxwell Chickering, David Heckerman, and Christopher Meek. Large-Sample Learning of Bayesian Networks is NP-Hard. *Journal of Machine Learning Research (JMLR)*, 5:1287–1330, 2004. 72

[CL68] C. K. Chow and C. N. Liu. Approximating Discrete Probability Distributions with Dependence Trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968. 55, 57

[CL24] Davin Choo and Chun Kai Ling. A short note about the learning-augmented secretary problem. *arXiv preprint arXiv:2410.06583*, 2024. 3, 206, 279

[CLL$^+$22] Debo Cheng, Jiuyong Li, Lin Liu, Kui Yu, Thuc Duy Le, and Jixue Liu. Toward Unique and Unbiased Causal Effect Estimation From Data With Hidden Variables. *IEEE Transactions on Neural Networks and Learning Systems*, 34(9):6108–6120, 2022. 135

[CLO07] James Clause, Wanchun Li, and Alessandro Orso. Dytan: A Generic Dynamic Taint Analysis Framework. In *International Symposium on Software*

Testing and Analysis (ISSTA), pages 196–206. Association for Computing Machinery (ACM), 2007. 188

[CLS$^+$24] Davin Choo, Yan Hao Ling, Warut Suksompong, Nicholas Teh, and Jian Zhang. Envy-free house allocation with minimum subsidy. *Operations Research Letters (ORL)*, 54(C), 2024. 279

[CLV03] Fan Chung, Linyuan Lu, and Van Vu. Spectra of random graphs with given expected degrees. *Proceedings of the National Academy of Sciences (PNAS)*, 100(11):6313–6318, 2003. 204

[CM02] David Maxwell Chickering and Christopher Meek. Finding Optimal Bayesian Networks. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 94–102. Morgan Kaufmann, 2002. 71

[CM13] T. Tony Cai and Zongming Ma. Optimal hypothesis testing for high dimensional covariance matrices. *Bernoulli*, 19(5B):2359–2388, 2013. 257

[CMKR12] Diego Colombo, Marloes H. Maathuis, Markus Kalisch, and Thomas S. Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, 40(1):294–321, 2012. 35, 135

[Cop95] Nicolaus Copernicus. *On the Revolutions of the Heavenly Spheres*. Prometheus Books, 1995. 9

[CS23a] Davin Choo and Kirankumar Shiragur. Adaptivity Complexity for Causal Graph Discovery. In *Uncertainty in Artificial Intelligence (UAI)*, pages 391–402, 2023. 3, 6, 135, 141, 280

[CS23b] Davin Choo and Kirankumar Shiragur. New metrics and search algorithms for weighted causal DAGs. In *International Conference on Machine Learning (ICML)*, pages 5868–5903, 2023. 3, 6, 135, 141, 142, 280

[CS23c] Davin Choo and Kirankumar Shiragur. Subset verification and search algorithms for causal DAGs. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 4409–4442, 2023. 3, 6, 132, 135, 280

[CSB22] Davin Choo, Kirankumar Shiragur, and Arnab Bhattacharyya. Verification and search algorithms for causal DAGs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 12787–12799, 2022. 3, 5, 33, 132, 135, 280

[CSBS25] Davin Choo, Chandler Squires, Arnab Bhattacharyya, and David Sontag. Probably approximately correct high-dimensional causal effect estimation given a valid adjustment set. In *Conference on Causal Learning and Reasoning (CLeaR)*, 2025. 3, 6, 132, 279, 280

[CSS19] Moses Charikar, Kirankumar Shiragur, and Aaron Sidford. A General Framework for Symmetric Property Estimation. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 12447–12457, 2019. 141

[CSU24] Davin Choo, Kirankumar Shiragur, and Caroline Uhler. Causal discovery under off-target interventions. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1621–1629, 2024. 3, 6, 142

[CSVZ22] Justin Chen, Sandeep Silwal, Ali Vakilian, and Fred Zhang. Faster Fundamental Graph Algorithms via Learned Predictions. In *International Conference on Machine Learning (ICML)*, pages 3583–3602. Proceedings of Machine Learning Research (PMLR), 2022. 36, 205

[CV22] Sourav Chatterjee and Mathukumalli Vidyasagar. Estimating large causal polytrees from small samples. *arXiv preprint arXiv:2209.07028*, 2022. 64

[CYBC24] Davin Choo, Joy Qiping Yang, Arnab Bhattacharyya, and Clément L Canonne. Learning bounded-degree polytrees with known skeleton. In *International Conference on Algorithmic Learning Theory (ALT)*, pages 402–443, 2024. 3, 4, 58, 69, 280

[Das97] Sanjoy Dasgupta. The Sample Complexity of Learning Fixed-Structure Bayesian Networks. *Machine Learning*, 29:165–180, 1997. 4, 43, 58, 64, 70

[Das99] Sanjoy Dasgupta. Learning polytrees. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 134–141. Morgan Kaufmann, 1999. 55

[dCJ11] Cassio P. de Campos and Qiang Ji. Efficient Structure Learning of Bayesian Networks using Constraints. *Journal of Machine Learning Research (JMLR)*, 12:663–689, 2011. 188

[DDF$^+$21] Alexander D'Amour, Peng Ding, Avi Feller, Lihua Lei, and Jasjeet Sekhon. Overlap in observational studies with high-dimensional covariates. *Journal of Econometrics*, 221(2):644–654, 2021. 140

[DDKC23] Yuval Dagan, Constantinos Daskalakis, Anthimos-Vardis Kandiros, and Davin Choo. Learning and Testing Latent-Tree Ising Models Efficiently. In

Conference on Learning Theory (COLT), pages 1666–1729, 2023. 3, 71, 280

[Dia16] Ilias Diakonikolas. Learning Structured Distributions. In *Handbook of Big Data*, pages 267–288. CRC Press, 2016. 39, 163

[DIL$^+$21] Michael Dinitz, Sungjin Im, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Faster Matchings via Learned Duals. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 10393–10406, 2021. 36, 205

[DIL$^+$22] Michael Dinitz, Sungjin Im, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Algorithms with Prediction Portfolios. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 20273–20286, 2022. 205

[DK14] Constantinos Daskalakis and Gautam Kamath. Faster and Sample Near-Optimal Algorithms for Proper Learning Mixtures of Gaussians. In *Conference on Learning Theory (COLT)*, pages 1183–1213. Proceedings of Machine Learning Research (PMLR), 2014. 17

[DKS17] Ilias Diakonikolas, Daniel M. Kane, and Alistair Stewart. Statistical Query Lower Bounds for Robust Estimation of High-Dimensional Gaussians and Gaussian Mixtures. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 73–84. IEEE, 2017. 164, 253

[DL01] Luc Devroye and Gábor Lugosi. *Combinatorial Methods in Density Estimation*. Springer, 2001. 17, 167

[DLPLV21] Paul Dütting, Silvio Lattanzi, Renato Paes Leme, and Sergei Vassilvitskii. Secretaries with Advice. In *ACM Conference on Economics and Computation (EC)*, pages 409–429. Association for Computing Machinery (ACM), 2021. 36

[DMR18] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. The total variation distance between high-dimensional Gaussians with the same mean. *arXiv preprint arXiv:1810.08693*, 2018. 174

[DP21] Constantinos Daskalakis and Qinxuan Pan. Sample-optimal and efficient learning of tree Ising models. In *Symposium on Theory of Computing (STOC)*, pages 133–146. Association for Computing Machinery (ACM), 2021. 4, 55, 56

[DW22] Hao Dong and Yuedong Wang. Nonparametric Neighborhood Selection in Graphical Models. *Journal of Machine Learning Research (JMLR)*, 23(1):14231–14266, 2022. 139

[Ebe07] Frederick Eberhardt. *Causation and Intervention*. PhD thesis, Carnegie Mellon University, 2007. 32, 135

[Ebe10] Frederick Eberhardt. Causal Discovery as a Game. In *Workshop on Causality: Objectives and Assessment*, pages 87–96. Proceedings of Machine Learning Research (PMLR), 2010. 5, 136

[EGS05] Frederick Eberhardt, Clark Glymour, and Richard Scheines. On the Number of Experiments Sufficient and in the Worst Case Necessary to Identify All Causal Relations Among N Variables. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 178–184. AUAI Press, 2005. 5, 31, 135, 136

[EGS06] Frederick Eberhardt, Clark Glymour, and Richard Scheines. N-1 Experiments Suffice to Determine the Causal Relations Among N Variables. In *Innovations in Machine Learning: Theory and Applications*, pages 97–112. Springer, 2006. 5, 135, 136

[EHS13] Doris Entner, Patrik Hoyer, and Peter Spirtes. Data-driven covariate selection for nonparametric estimation of causal effects. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 256–264. Proceedings of Machine Learning Research (PMLR), 2013. 134

[Ein22] Albert Einstein. *Sidelights on relativity*. Methuen & Company Limited, 1922. 144

[Eri19] Jeff Erickson. *Algorithms*. Jeff Erickson, 2019. 84, 99

[ES07] Frederick Eberhardt and Richard Scheines. Interventions and Causal Inference. *Philosophy of Science*, 74(5):981–995, 2007. 5

[Eve99] Brian Everitt. *Chance Rules: An Informal Guide to Probability, Risk and Statistics*. Copernicus, 1999. 163

[FD08] Shunkai Fu and Michel C. Desmarais. Fast Markov Blanket Discovery Algorithm Via Local Learning within Single Pass. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 96–107. Springer, 2008. 139, 140

[FFT$^+$03] Lewis Frey, Douglas Fisher, Ioannis Tsamardinos, Constantin F. Aliferis, and Alexander Statnikov. Identifying Markov Blankets with Decision Tree Induction. In *IEEE International Conference on Data Mining (ICDM)*, pages 59–66. IEEE, 2003. 139

[FH20] Zhuangyan Fang and Yangbo He. IDA with Background Knowledge. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 270–279. Proceedings of Machine Learning Research (PMLR), 2020. 188

[FLM21] Max H. Farrell, Tengyuan Liang, and Sanjog Misra. Deep Neural Networks for Estimation and Inference. *Econometrica*, 89(1):181–213, 2021. 141

[FMMM09] Jon Feldman, Aranyak Mehta, Vahab Mirrokni, and Shan Muthukrishnan. Online Stochastic Matching: Beating 1-1/e. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 117–126. IEEE, 2009. 147, 149

[FMT$^+$21] Chris J. Frangieh, Johannes C. Melms, Pratiksha I. Thakore, Kathryn R. Geiger-Schuller, Patricia Ho, Adrienne M. Luoma, Brian Cleary, Livnat Jerby-Arnon, Shruti Malu, Michael S. Cuoco, Maryann Zhao, Casey R. Ager, Meri Rogava, Lila Hovey, Asaf Rotem, Chantale Bernatchez, Kai W. Wucherpfennig, Bruce E. Johnson, Orit Rozenblatt-Rosen, Dirk Schadendorf, Aviv Regev, and Benjamin Izar. Multimodal pooled Perturb-CITE-seq screens in patient models define mechanisms of cancer immune evasion. *Nature Genetics*, 53(3):332–341, 2021. 75

[FNB$^+$11] M. Julia Flores, Ann E. Nicholson, Andrew Brunskill, Kevin B. Korb, and Steven Mascaro. Incorporating expert knowledge when learning Bayesian network structure: A medical case study. *Artificial Intelligence in Medicine*, 53(3):181–204, 2011. 188

[FNP99] Nir Friedman, Iftach Nachman, and Dana Peér. Learning Bayesian Network Structure from Massive Datasets: The “Sparse Candidate” Algorithm. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 206—215. Morgan Kaufmann, 1999. 71

[FNS21] Yiding Feng, Rad Niazadeh, and Amin Saberi. Two-stage stochastic matching with application to ride hailing. In *Symposium on Discrete Algorithms (SODA)*, pages 2862–2877. Society for Industrial and Applied Mathematics (SIAM), 2021. 204

[FR13] Simon Foucart and Holger Rauhut. *A Mathematical Introduction to Compressive Sensing*. Birkhäuser, 2013. 165

[Fre04] Robert M. Freund. Introduction to Semidefinite Programming (SDP), 2004. MIT OpenCourseWare. 182, 266

[FY96] Nir Friedman and Zohar Yakhini. On the Sample Complexity of Learning Bayesian Networks. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 274–282. Morgan Kaufmann, 1996. 71

[GA21] Ming Gao and Bryon Aragam. Efficient Bayesian network structure learning via local Markov boundary search. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4301–4313, 2021. 55, 64, 71, 139, 140

[GCL23] Shantanu Gupta, David Childers, and Zachary Chase Lipton. Local Causal Discovery for Estimating Causal Effects. In *Conference on Causal Learning and Reasoning*, pages 408–447. Proceedings of Machine Learning Research (PMLR), 2023. 139

[GDA20] Ming Gao, Yi Ding, and Bryon Aragam. A polynomial-time algorithm for learning nonparametric causal graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 11599–11611, 2020. 39, 139

[GH17] Asish Ghoshal and Jean Honorio. Learning Identifiable Gaussian Bayesian Networks in Polynomial Time and Sample Complexity. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6460–6469, 2017. 39, 64, 71

[Gho21] Malay Ghosh. Exponential Tail Bounds for Chisquared Random Variables. *Journal of Statistical Theory and Practice*, 15(35), 2021. 22

[GHT84] John R. Gilbert, Joan P. Hutchinson, and Robert Endre Tarjan. A Separator Theorem for Graphs of Bounded Genus. *Journal of Algorithms*, 5(3):391–407, 1984. 28

[GJ15] Tian Gao and Qiang Ji. Local Causal Discovery of Direct Causes and Effects. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2512–2520, 2015. 139

[GJ17] Tian Gao and Qiang Ji. Efficient Markov Blanket Discovery and Its Application. *IEEE Transactions on Cybernetics*, 47(5):1169–1179, 2017. 139, 140

[GK90] Daniel H. Greene and Donald E. Knuth. Mathematics for the Analysis of Algorithms, 1990. 10

[GKS$^+$19] Kristjan Greenewald, Dmitriy Katz, Karthikeyan Shanmugam, Sara Magliacane, Murat Kocaoglu, Enric Boix-Adserà, and Guy Bresler. Sample Efficient Active Learning of Causal Trees. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 14313–14323, 2019. 135, 136

[GKST22] Vasilis Gkatzelis, Kostas Kollias, Alkmini Sgouritsa, and Xizhi Tan. Improved Price of Anarchy via Predictions. In *ACM Conference on Economics and Computation (EC)*, pages 529–557. Association for Computing Machinery (ACM), 2022. 36

[GLS23] Themistoklis Gouleakis, Konstantinos Lakis, and Golnoosh Shahkarami. Learning-Augmented Algorithms for Online TSP on the Line. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 11989–11996. AAAI Press, 2023. 36

[GM08] Gagan Goel and Aranyak Mehta. Online budgeted matching in random input models with applications to Adwords. In *Symposium on Discrete Algorithms (SODA)*, pages 982–991, 2008. 145, 148, 149

[GM12] Bernd Gärtner and Jiri Matousek. *Approximation Algorithms and Semidefinite Programming*. Springer Science & Business Media, 2012. 182, 266

[GP19] Sreenivas Gollapudi and Debmalya Panigrahi. Online Algorithms for Rent-or-Buy with Expert Advice. In *International Conference on Machine Learning (ICML)*, pages 2319–2327. Proceedings of Machine Learning Research (PMLR), 2019. 36

[GP21] Richard Guo and Emilija Perkovic. Minimal enumeration of all possible total effects in a Markov equivalence class. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 2395–2403. Proceedings of Machine Learning Research (PMLR), 2021. 206

[GPP90] Dan Geiger, Azaria Paz, and Judea Pearl. Learning causal trees from dependence information. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 770–776. AAAI Press, 1990. 71

[GR20] Rishi Gupta and Tim Roughgarden. Data-driven algorithm design. *Communications of the ACM*, 63(6):87–94, 2020. 7

[GRE84] John R. Gilbert, Donald J. Rose, and Anders Edenbrandt. A Separator Theorem for Chordal Graphs. *Society for Industrial and Applied Mathematics (SIAM) Journal on Algebraic Discrete Methods*, 5(3):306–313, 1984. 28, 83, 85

[GSK21] Kristjan Greenewald, Karthikeyan Shanmugam, and Dmitriy Katz. High-Dimensional Feature Selection for Sample Efficient Treatment Effect Estimation. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 2224–2232. Proceedings of Machine Learning Research (PMLR), 2021. 141

[GSKB18] AmirEmad Ghassami, Saber Salehkaleybar, Negar Kiyavash, and Elias Bareinboim. Budgeted Experiment Design for Causal Structure Learning. In *International Conference on Machine Learning (ICML)*, pages 1724–1733. Proceedings of Machine Learning Research (PMLR), 2018. 33, 34, 82, 87, 138

[GTA22] Ming Gao, Wai Ming Tai, and Bryon Aragam. Optimal Estimation of Gaussian DAG Models. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 8738–8757. Proceedings of Machine Learning Research (PMLR), 2022. 64, 139

[GUA$^+$16] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario March, and Victor Lempitsky. Domain-Adversarial Training of Neural Networks. *Journal of Machine Learning Research (JMLR)*, 17(59):1–35, 2016. 5

[Gut09] Allan Gut. *An Intermediate Course in Probability*. Springer, 2009. 21, 22

[GZ20] Jiaying Gu and Qing Zhou. Learning Big Gaussian Bayesian Networks: Partition, Estimation and Fusion. *Journal of Machine Learning Research (JMLR)*, 21(158):1–31, 2020. 39

[GZS19] Clark Glymour, Kun Zhang, and Peter Spirtes. Review of Causal Discovery Methods Based on Graphical Models. *Frontiers in Genetics*, 10, 2019. 35, 135

[Han24] Yanjun Han. Personal communication via email (20 jan 2024), 2024. 24, 158, 250

[HB12] Alain Hauser and Peter Bühlmann. Characterization and greedy learning of interventional Markov equivalence classes of directed acyclic graphs. *Journal of Machine Learning Research (JMLR)*, 13(1):2409–2464, 2012. 33, 82, 98, 135

[HB14] Alain Hauser and Peter Bühlmann. Two optimal strategies for active learning of causal models from interventional data. *International Journal of Approximate Reasoning (IJAR)*, 55(4):926–939, 2014. 33, 136

[HDMM18] Christina Heinze-Deml, Marloes H. Maathuis, and Nicolai Meinshausen. Causal Structure Learning. *Annual Review of Statistics and Its Application*, 5:371–391, 2018. 135

[HEH13] Antti Hyttinen, Frederick Eberhardt, and Patrik O. Hoyer. Experiment Selection for Causal Discovery. *Journal of Machine Learning Research (JMLR)*, 14(93):3041–3071, 2013. 32, 78, 137

[HJ12] Roger A. Horn and Charles R. Johnson. *Matrix Analysis*. Cambridge University Press, 2012. 11, 12, 13, 14

[HJM$^+$08] Patrik Hoyer, Dominik Janzing, Joris Mooij, Jonas Peters, and Bernhard Schölkopf. Nonlinear causal discovery with additive noise models. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 689–696, 2008. 71

[HJS$^+$22] Baihe Huang, Shunhua Jiang, Zhao Song, Runzhou Tao, and Ruizhe Zhang. Solving SDP Faster: A Robust IPM Framework and Efficient Implementation. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 233–244, 2022. 266, 267

[HK99] Monika R. Henzinger and Valerie King. Randomized Fully Dynamic Graph Algorithms with Polylogarithmic Time per Operation. *Journal of the ACM*, 46(4):502–516, 1999. 230

[HLV14] Huining Hu, Zhentao Li, and Adrian Vetta. Randomized Experimental Design for Causal Graph Discovery. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 2339–2347, 2014. 32, 135, 136, 138

[Hoe94] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *The collected works of Wassily Hoeffding*, pages 409–426, 1994. 15

[Hol86] Paul W. Holland. Statistics and Causal Inference. *Journal of the American Statistical Association*, 81(396):945–960, 1986. 74

[Hoo90] Kevin D. Hoover. The Logic of Causal Inference: Econometrics and the Conditional Analysis of Causation. *Economics and Philosophy*, 6(2):207–234, 1990. 5

[HTW15] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. CRC Press, 2015. 4, 167

[HV06] Yimin Huang and Marco Valtorta. Pearl's Calculus of Intervention Is Complete. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 217–224. AUAI Press, 2006. 134

[ICM$^+$22] Azam Ikram, Sarthak Chakraborty, Subrata Mitra, Shiv Kumar Saini, Saurabh Bagchi, and Murat Kocaoglu. Root Cause Analysis of Failures in Microservices through Causal Discovery. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 31158–31170, 2022. 188, 189

[JBT$^+$19] Saurabh Jha, Subho Banerjee, Timothy Tsai, Siva K. S. Hari, Michael B. Sullivan, Zbigniew T. Kalbarczyk, Stephen W. Keckler, and Ravishankar K. Iyer. ML-Based Fault Injection for Autonomous Vehicles: A Case for Bayesian Fault Injection. In *Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 112–124. IEEE, 2019. 188

[Jen06] Johan Ludwig William Valdemar Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30:175–193, 1906. 238

[JHW18] Jiantao Jiao, Yanjun Han, and Tsachy Weissman. Minimax Estimation of the L_1 Distance. *IEEE Transactions on Information Theory*, 64(10):6672–6706, 2018. 23, 24, 158, 250

[JKSB20] Amin Jaber, Murat Kocaoglu, Karthikeyan Shanmugam, and Elias Bareinboim. Causal Discovery from Soft Interventions with Unknown Targets: Characterization and Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9551–9561, 2020. 135

[JL14] Patrick Jaillet and Xin Lu. Online Stochastic Matching: New Algorithms with Better Bounds. *Mathematics of Operations Research*, 39(3):624–646, 2014. 149

[JM22] Billy Jin and Will Ma. Online Bipartite Matching with Advice: Tight Robustness-Consistency Tradeoffs for the Two-Stage Model. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 14555–14567, 2022. 146, 204, 205

[JN07] Finn V. Jensen and Thomas D. Nielsen. *Bayesian Networks and Decision Graphs*. Springer, 2007. 4

[JNG$^+$19] Chi Jin, Praneeth Netrapalli, Rong Ge, Sham M. Kakade, and Michael I. Jordan. A Short Note on Concentration Inequalities for Random Vectors with SubGaussian Norm. *arXiv preprint arXiv:1902.03736*, 2019. 210

[JRZB22] Amin Jaber, Adele Ribeiro, Jiji Zhang, and Elias Bareinboim. Causal Identification under Markov equivalence: Calculus, Algorithm, and Completeness. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3679–3690, 2022. 32

[Kar72] Richard M. Karp. Reducibility among Combinatorial Problems. In *Complexity of Computer Computations*, pages 85–103. Springer, 1972. 83, 84, 99

[KB07] Markus Kalisch and Peter Bühlmann. Estimating High-Dimensional Directed Acyclic Graphs with the PC-Algorithm. *Journal of Machine Learning Research (JMLR)*, 8, 2007. 139

[KBC$^+$18] Tim Kraska, Alex Beutel, Ed H. Chi, Jeffrey Dean, and Neoklis Polyzotis. The Case for Learned Index Structures. In *International Conference on Management of Data (SIGMOD)*, pages 489–504. Association for Computing Machinery (ACM), 2018. 36

[KCG$^+$23] Neville Kenneth Kitson, Anthony C. Constantinou, Zhigao Guo, Yang Liu, and Kiattikun Chobtham. A survey of Bayesian Network structure learning. *Artificial Intelligence Review*, 56(8):8721–8814, 2023. 65

[KDV17] Murat Kocaoglu, Alexandros G. Dimakis, and Sriram Vishwanath. Cost-Optimal Learning of Causal Graphs. In *International Conference on Machine Learning (ICML)*, pages 1875–1884. Proceedings of Machine Learning Research (PMLR), 2017. 32, 135, 138

[KF09] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009. 4, 70

[KJSB19] Murat Kocaoglu, Amin Jaber, Karthikeyan Shanmugam, and Elias Bareinboim. Characterization and Learning of Causal Graphs with Latent Variables from Soft Interventions. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 14369–14379, 2019. 132

[KLSU19] Gautam Kamath, Jerry Li, Vikrant Singhal, and Jonathan Ullman. Privately Learning High-Dimensional Distributions. In *Conference on Learning Theory (COLT)*, pages 1853–1902. Proceedings of Machine Learning Research (PMLR), 2019. 19, 168

[KMR$^+$94] Michael Kearns, Yishay Mansour, Dana Ron, Ronitt Rubinfeld, Robert E. Schapire, and Linda Sellie. On the learnability of discrete distributions. In *Symposium on Theory of Computing (STOC)*, pages 273–282. Association for Computing Machinery (ACM), 1994. 4, 39, 108

[KMT11] Chinmay Karande, Aranyak Mehta, and Pushkar Tripathi. Online bipartite matching with unknown distributions. In *Symposium on Theory of Computing (STOC)*, pages 587–596. Association for Computing Machinery (ACM), 2011. 148

[KNR20] Haim Kaplan, David Naori, and Danny Raz. Competitive Analysis with a Sample and the Secretary Problem. In *Symposium on Discrete Algorithms (SODA)*, pages 2082–2095. Society for Industrial and Applied Mathematics (SIAM), 2020. 203

[KNR22] Haim Kaplan, David Naori, and Danny Raz. Online Weighted Matching with a Sample. In *Symposium on Discrete Algorithms (SODA)*, pages 1247–1272. Society for Industrial and Applied Mathematics (SIAM), 2022. 203

[Knu76] Donald E. Knuth. Big Omicron and big Omega and big Theta. *ACM SIGACT News*, 8(2):18–24, 1976. 10

[KPS18] Ravi Kumar, Manish Purohit, and Zoya Svitkina. Improving Online Algorithms via ML Predictions. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9684–9693, 2018. 36

[KR10] Ken-ichi Kawarabayashi and Bruce Reed. A Separator Theorem in Minor-Closed Classes. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 153–162. IEEE, 2010. 28

[KS96] Daphne Koller and Mehran Sahami. Toward optimal feature selection. In *International Conference on Machine Learning (ICML)*, page 284–292. Morgan Kaufmann, 1996. 139

[KSSU19] Dmitriy Katz, Karthikeyan Shanmugam, Chandler Squires, and Caroline Uhler. Size of Interventional Markov Equivalence Classes in Random DAG Models. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 3234–3243. Proceedings of Machine Learning Research (PMLR), 2019. 138

[KSV24] Adam Klivans, Konstantinos Stavropoulos, and Arsen Vasilyan. Testable Learning with Distribution Shift. In *Conference on Learning Theory (COLT)*, pages 2887–2943. Proceedings of Machine Learning Research (PMLR), 2024. 165

[KVV90] Richard M. Karp, Umesh V. Vazirani, and Vijay V. Vazirani. An Optimal Algorithm for On-line Bipartite Matching. In *Symposium on Theory of Computing (STOC)*, pages 352–358. Association for Computing Machinery (ACM), 1990. 144, 145, 148, 151, 251

[KWJ$^+$04] Ross D. King, Kenneth E. Whelan, Ffion M. Jones, Philip G. K. Reiser, Christopher H. Bryant, Stephen H. Muggleton, Douglas B. Kell, and Stephen G. Oliver. Functional genomic hypothesis generation and experimentation by a robot scientist. *Nature*, 427(6971):247–252, 2004. 5

[Lam23] Wai Yin Lam. *Causal Razors and Causal Search Algorithms*. PhD thesis, Carnegie Mellon University, 2023. 31, 139

[LB18] Andrew Li and Peter Beek. Bayesian Network Structure Learning with Side Constraints. In *International Conference on Probabilistic Graphical Models (PGM)*, pages 225–236. Proceedings of Machine Learning Research (PMLR), 2018. 188

[LCL25] Jia Peng Lim, Davin Choo, and Hady W. Lauw. A partition cover approach to tokenization. *arXiv preprint arXiv:2501.06246*, 2025. 279

[LD05] Michael Levine and Eric H. Davidson. Gene regulatory networks for development. *Proceedings of the National Academy of Sciences (PNAS)*, 102(14):4936–4942, 2005. 75

[Leu04] Rebecca Leung. Carrey: 'Life Is Too Beautiful', 2004. Accessed: 2024-09-23. 144

[LKC17] Gilles Louppe, Michael Kagan, and Kyle Cranmer. Learning to Pivot with Adversarial Networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 982–991, 2017. 5

[LKDV18] Erik M. Lindgren, Murat Kocaoglu, Alexandros G. Dimakis, and Sriram Vishwanath. Experimental Design for Cost-Aware Learning of Causal Graphs. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5284–5294, 2018. 135, 138

[LLMV20] Silvio Lattanzi, Thomas Lavastida, Benjamin Moseley, and Sergei Vassilvitskii. Online Scheduling via Learned Weights. In *Symposium on Discrete Algorithms (SODA)*, pages 1859–1877. Society for Industrial and Applied Mathematics (SIAM), 2020. 36

[LMRX21a] Thomas Lavastida, Benjamin Moseley, R. Ravi, and Chenyang Xu. Learnable and Instance-Robust Predictions for Online Matching, Flows and Load

Balancing. In *Annual European Symposium on Algorithms (ESA)*, pages 59:1–59:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. 205

[LMRX21b] Thomas Lavastida, Benjamin Moseley, R. Ravi, and Chenyang Xu. Using Predicted Weights for Ad Delivery. In *SIAM Conference on Applied and Computational Discrete Algorithms (ACDA)*, pages 21–31. Society for Industrial and Applied Mathematics (SIAM), 2021. 205

[LT79] Richard J. Lipton and Robert Endre Tarjan. A Separator Theorem for Planar Graphs. *Society for Industrial and Applied Mathematics (SIAM) Journal on Applied Mathematics*, 36(2):177–189, 1979. 28

[LV21] Thodoris Lykouris and Sergei Vassilvitskii. Competitive Caching with Machine Learned Advice. *Journal of the ACM (JACM)*, 68(4):1–25, 2021. 36

[LWHLS22] Chaochao Lu, Yuhuai Wu, José Miguel Hernández-Lobato, and Bernhard Schölkopf. Invariant Causal Representation Learning for Out-of-Distribution Generalization. In *International Conference on Learning Representations (ICLR)*, 2022. 75

[LYR23] Pengfei Li, Jianyi Yang, and Shaolei Ren. Learning for Edge-Weighted Online Bipartite Matching with Robustness Guarantees. In *International Conference on Machine Learning (ICML)*, pages 20276–20295. Proceedings of Machine Learning Research (PMLR), 2023. 146, 204, 205

[LYW$^+$20] Zhaolong Ling, Kui Yu, Hao Wang, Lei Li, and Xindong Wu. Using Feature Selection for Local Causal Structure Learning. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(4):530–540, 2020. 139

[Mal24] Daniel Malinsky. A cautious approach to constraint-based causal model selection. *arXiv preprint arXiv:2404.18232*, 2024. 139

[MC04] Subramani Mani and Gregory F. Cooper. Causal discovery using a Bayesian local causal discovery algorithm. In *Studies in Health Technology and Informatics (HTI)*, pages 731–735. IOS Press, 2004. 75, 139

[MC15] Marloes H Maathuis and Diego Colombo. A generalized back-door criterion. *The Annals of Statistics*, 43(3):1060–1088, 2015. 109, 111, 134

[MDLW18] Marloes Maathuis, Mathias Drton, Steffen Lauritzen, and Martin Wainwright. *Handbook of Graphical Models*. CRC Press, 2018. 243

[Mee95] Christopher Meek. Causal Inference and Causal Explanation with Background Knowledge. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, page 403–410. Morgan Kaufmann, 1995. 28, 60, 188

[Meh13] Aranyak Mehta. Online Matching and Ad Allocation. *Foundations and Trends® in Theoretical Computer Science*, 8(4):265–368, 2013. 148, 149, 247

[MGS12] Vahideh H. Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online Stochastic Matching: Online Actions Based on Offline Statistics. *Mathematics of Operations Research*, 37(4):559–573, 2012. 148, 149, 251

[Mil55] William Miller. Death of a Genius: His Fourth Dimension, Time, Overtakes Einstein. *LIFE*, pages 62–64, 1955. 1

[Mit18] Michael Mitzenmacher. A Model for Learned Bloom Filters, and Optimizing by Sandwiching. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 462–471, 2018. 36

[MKB09] Marloes H. Maathuis, Markus Kalisch, and Peter Bühlmann. Estimating high-dimensional intervention effects from observational data. *The Annals of Statistics*, 37(6):3133–3164, 2009. 139

[MNS12] Mohammad Mahdian, Hamid Nazerzadeh, and Amin Saberi. Online Optimization with Uncertain Information. *ACM Transactions on Algorithms (TALG)*, 8(1):1–29, 2012. 36

[MV22] Michael Mitzenmacher and Sergei Vassilvitskii. Algorithms with Predictions. *Communications of the ACM*, 65(7):33–35, 2022. 7

[MY11] Mohammad Mahdian and Qiqi Yan. Online Bipartite Matching with Random Arrivals: An Approach Based on Strongly Factor-Revealing LPs. In *Symposium on Theory of Computing (STOC)*, pages 597–606. Association for Computing Machinery (ACM), 2011. 148, 251

[Nor97] James R. Norris. *Markov Chains*. Cambridge University Press, 1997. 2

[Par20] Gunwoong Park. Identifiability of Additive Noise Models Using Conditional Variances. *Journal of Machine Learning Research (JMLR)*, 21(75):1–34, 2020. 39

[PB14] Jonas Peters and Peter Bühlmann. Identifiability of Gaussian structural equation models with equal error variances. *Biometrika*, 101(1):219–228, 2014. 39, 71

[Pea86] Judea Pearl. Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, 29(3):241–288, 1986. 55

[Pea88] Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988. 2, 29, 38

[Pea95] Judea Pearl. Causal diagrams for empirical research. *Biometrika*, 82(4):669–688, 1995. 109, 111, 112, 121, 134

[Pea09a] Judea Pearl. Causal inference in statistics: An overview. *Statistics Surveys*, 3:96–146, 2009. 2, 108, 114

[Pea09b] Judea Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, 2009. 4, 5, 32

[Per20] Emilija Perkovic. Identifying causal effects in maximally oriented partially directed acyclic graphs. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 530–539. Proceedings of Machine Learning Research (PMLR), 2020. 134, 206

[PK83] Judea Pearl and Jin H. Kim. A Computational Model for Causal and Diagnostic Reasoning in Inference Systems. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 190–193. International Joint Conferences on Artificial Intelligence (IJCAI), 1983. 55

[PK20] Gunwoong Park and Youngwhan Kim. Identifiability of gaussian linear structural equation models with homogeneous and heterogeneous error variances. *Journal of the Korean Statistical Society*, 49:276–292, 2020. 39

[PKM17] Emilija Perkovic, Markus Kalisch, and Marloes H. Maathuis. Interpreting and using CPDAGs with background knowledge. In *Conference on Uncertainty in Artificial Intelligence (UAI)*. AUAI Press, 2017. 206

[PM18] Judea Pearl and Dana Mackenzie. *The Book of Why: The New Science of Cause and Effect*. Basic Books, 2018. 2

[PNBT07] Jose M. Peòa, Roland Nilsson, Johan Björkegren, and Jesper Tegnér. Towards scalable and data efficient learning of Markov boundaries. *International Journal of Approximate Reasoning (IJAR)*, 45(2):211–232, 2007. 139

[POE21] Anne H. Petersen, Merete Osler, and Claus T Ekstrøm. Data-Driven Model Building for Life-Course Epidemiology. *American Journal of Epidemiology*, 190(9):1898–1907, 2021. 188

[POS$^+$18] Jean-Baptiste Pingault, Paul F. O'Reilly, Tabea Schoeler, George B. Ploubidis, Frühling Rijsdijk, and Frank Dudbridge. Using genetic data to strengthen causal inference in observational research. *Nature Reviews Genetics*, 19(9):566–580, 2018. 5

[PR18] Gunwoong Park and Garvesh Raskutti. Learning Quadratic Variance Function (QVF) DAG Models via OverDispersion Scoring (ODS). *Journal of Machine Learning Research (JMLR)*, 18(224):1–44, 2018. 71

[PSS22] Vibhor Porwal, Piyush Srivastava, and Gaurav Sinha. Almost Optimal Universal Lower Bound for Learning Causal DAGs with Atomic Interventions. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 5583–5603. Proceedings of Machine Learning Research (PMLR), 2022. 78, 79, 138

[PTKM18] Emilija Perković, Johannes Textor, Markus Kalisch, and Marloes H. Maathuis. Complete Graphical Characterization and Construction of Adjustment Sets in Markov Equivalence Classes of Ancestral Graphs. *Journal of Machine Learning Research (JMLR)*, 18(220):1–62, 2018. 109, 111, 134, 239

[Ram06] Joseph D. Ramsey. A PC-style Markov blanket search for high-dimensional datasets. Technical report, Carnegie Mellon University, 2006. Technical Report, CMU-PHIL-177. 139

[Rei91] Hans Reichenbach. *The Direction of Time*. University of California Press, 1991. 5

[Rey15] Douglas Reynolds. Gaussian Mixture Models. In *Encyclopedia of Biometrics*, pages 827–832. Springer, 2015. 2

[RH23] Philippe Rigollet and Jan-Christian Hütter. High-Dimensional Statistics. *arXiv preprint arXiv:2310.19244*, 2023. 23

[RHT$^+$17] Maya Rotmensch, Yoni Halpern, Abdulhakim Tlimat, Steven Horng, and David Sontag. Learning a Health Knowledge Graph from Electronic Medical Records. *Scientific Reports*, 7(1):5994, 2017. 5

[RJ86] Lawrence R. Rabiner and Biing-Hwang Juang. An introduction to hidden Markov models. *IEEE ASSP Magazine*, 3:4–16, 1986. 2

[Roh20] Dhruv Rohatgi. Near-Optimal Bounds for Online Caching with Machine Learned Advice. In *Symposium on Discrete Algorithms (SODA)*, pages

1834–1845. Society for Industrial and Applied Mathematics (SIAM), 2020. 36

[RP88] George Rebane and Judea Pearl. The recovery of causal poly-trees from statistical data. *International Journal of Approximate Reasoning (IJAR)*, 2(3):341, 1988. 55, 65, 71

[Rub74] Donald B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of Educational Psychology*, 66(5):688–701, 1974. 2, 108

[RV09] Mark Rudelson and Roman Vershynin. Smallest singular value of a random rectangular matrix. *Communications on Pure and Applied Mathematics*, 62(12):1707–1739, 2009. 12

[RV23] Ronitt Rubinfeld and Arsen Vasilyan. Testing distributional assumptions of learning algorithms. In *Symposium on Theory of Computing (STOC)*, pages 1643–1656. Association for Computing Machinery (ACM), 2023. 165

[RW06] Donald B. Rubin and Richard P. Waterman. Estimating the Causal Effects of Marketing Interventions Using Propensity Score Methodology. *Statistical Science*, 21(2):206–222, 2006. 5

[SC17] Yuriy Sverchkov and Mark Craven. A review of active learning approaches to experimental design for uncovering biological networks. *PLoS Computational Biology*, 13(6), 2017. 5

[SC21] Jonathan Scarlett and Volkan Cevher. An Introductory Guide to Fano's Inequality with Applications in Statistical Estimation. In *Information-Theoretic Methods in Data Science*, page 487–528. Cambridge University Press, 2021. 18

[Sch22] Bernhard Schölkopf. Causality for Machine Learning. In *Probabilistic and Causal Inference: The Works of Judea Pearl*, pages 765–804. Association for Computing Machinery (ACM), 2022. 5

[Scu10] Marco Scutari. Learning Bayesian Networks with the bnlearn R Package. *Journal of Statistical Software*, 35(3):1–22, 2010. 39

[SE17] Susan M. Shortreed and Ashkan Ertefaie. Outcome-adaptive lasso: Variable selection for causal inference. *Biometrics*, 73(4):1111–1122, 2017. 141

[Sek09] Jasjeet Sekhon. The Neyman-Rubin Model of Causal Inference and Estimation Via Matching Methods. In *The Oxford Handbook of Political Methodology*, pages 271–299. Oxford University Press, 2009. 2, 108

[SG91] Peter Spirtes and Clark Glymour. An Algorithm for Fast Recovery of Sparse Causal Graphs. *Social Science Computer Review*, 9(1):62–72, 1991. 71

[SGS00] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT press, 2000. 5, 35, 113, 135

[SHHK06] Shohei Shimizu, Patrik O. Hoyer, Aapo Hyvärinen, and Antti Kerminen. A Linear Non-Gaussian Acyclic Model for Causal Discovery. *Journal of Machine Learning Research (JMLR)*, 7(72):2003–2030, 2006. 71

[SKDV15] Karthikeyan Shanmugam, Murat Kocaoglu, Alexandros G. Dimakis, and Sriram Vishwanath. Learning Causal Graphs with Small Interventions. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3195–3203, 2015. 32, 78, 86, 135, 136, 137, 138

[SMG$^+$20] Chandler Squires, Sara Magliacane, Kristjan Greenewald, Dmitriy Katz, Murat Kocaoglu, and Karthikeyan Shanmugam. Active Structure Learning of Causal DAGs via Directed Clique Trees. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 21500–21511, 2020. 33, 77, 78, 79, 83, 84, 96, 135, 138

[SMH$^+$15] Alexander Statnikov, Sisi Ma, Mikael Henaff, Nikita Lytkin, Efstratios Efstathiadis, Eric R. Peskin, and Constantin F. Aliferis. Ultra-Scalable and Efficient Methods for Hybrid Observational and Experimental Local Causal Pathway Discovery. *Journal of Machine Learning Research (JMLR)*, 16(100):3219–3267, 2015. 75

[SN90] Jerzy Splawa-Neyman. On the Application of Probability Theory to Agricultural Experiments. Essay on Principles. Section 9. *Statistical Science*, 5(4):465–472, 1990. 2, 108

[SO22] Shinsaku Sakaue and Taihei Oki. Discrete-Convex-Analysis-Based Framework for Warm-Starting Algorithms with Predictions. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 20988–21000, 2022. 205

[SP06] Ilya Shpitser and Judea Pearl. Identification of Joint Interventional Distributions in Recursive Semi-Markovian Causal Models. In *AAAI Conference on Artificial Intelligence (AAAI)*, page 1219–1226. AAAI Press, 2006. 134

[SPU20] Basil Saeed, Snigdha Panigrahi, and Caroline Uhler. Causal Structure Discovery from Distributions Arising from Mixtures of DAGs. In *International Conference on Machine Learning (ICML)*, pages 8336–8345. Proceedings of Machine Learning Research (PMLR), 2020. 5

[SR17] Megan S. Schuler and Sherri Rose. Targeted Maximum Likelihood Estimation for Causal Inference in Observational Studies. *American Journal of Epidemiology*, 185(1):65–73, 2017. 140

[SSA22] Abhin Shah, Karthikeyan Shanmugam, and Kartik Ahuja. Finding Valid Adjustments under Non-ignorability with Minimal DAG Knowledge. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 5538–5562. Proceedings of Machine Learning Research (PMLR), 2022. 135

[SSG$^+$98] Richard Scheines, Peter Spirtes, Clark Glymour, Christopher Meek, and Thomas Richardson. The TETRAD project: Constraint based aids to causal model specification. *Multivariate Behavioral Research*, 33(1):65–117, 1998. 188

[SSK23] Abhin Shah, Karthikeyan Shanmugam, and Murat Kocaoglu. Front-door Adjustment Beyond Markov Equivalence with Limited Graph Knowledge. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 43800–43825, 2023. 135

[SU23] Chandler Squires and Caroline Uhler. Causal Structure Learning: A Combinatorial Perspective. *Foundations of Computational Mathematics*, 23(5):1781–1815, 2023. 135

[SVR10] Ilya Shpitser, Tyler VanderWeele, and James M. Robins. On the Validity of Covariate Adjustment for Estimating Causal Effects. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 527–536. AUAI Press, 2010. 109, 111, 134

[SWU21] Liam Solus, Yuhao Wang, and Caroline Uhler. Consistency Guarantees for Greedy Permutation-Based Causal Inference Algorithms. *Biometrika*, 108(4):795–814, 2021. 135, 139

[TA03] Ioannis Tsamardinos and Constantin F. Aliferis. Towards Principled Feature Selection: Relevancy, Filters and Wrappers. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 300–307. Proceedings of Machine Learning Research (PMLR), 2003. 75

[Tal07] Nassim Nicholas Taleb. *The Black Swan: The Impact of the Highly Improbable*. Random House, 2007. 163

[TAS03] Ioannis Tsamardinos, Constantin F. Aliferis, and Alexander Statnikov. Algorithms for Large Scale Markov Blanket Discovery. In *International Florida*

Artificial Intelligence Research Society Conference (FLAIRS), pages 376–380. AAAI Press, 2003. 139, 140

[TBA06] Ioannis Tsamardinos, Laura E. Brown, and Constantin F. Aliferis. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 65:31–78, 2006. 139

[Tib96] Robert Tibshirani. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. 167

[Tib97] Robert Tibshirani. The lasso method for variable selection in the Cox model. *Statistics in Medicine*, 16(4):385–395, 1997. 167

[TJG$^+$19] Cheng Tan, Ze Jin, Chuanxiong Guo, Tianrong Zhang, Haitao Wu, Karl Deng, Dongming Bi, and Dong Xiang. NetBouncer: Active Device and Link Failure Localization in Data Center Networks. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, pages 599–614. USENIX Association, 2019. 188

[TP02] Jin Tian and Judea Pearl. A General Identification Condition for Causal Effects. In *Eighteenth National Conference on Artificial Intelligence*, pages 567–573, 2002. 134

[Tsy09] Alexandre B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009. 16

[TV84] Robert Endre Tarjan and Uzi Vishkin. Finding biconnected componemts and computing tree functions in logarithmic parallel time. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 12–20. IEEE, 1984. 230

[URBY13] Caroline Uhler, Garvesh Raskutti, Peter Bühlmann, and Bin Yu. Geometry of the faithfulness assumption in causal inference. *The Annals of Statistics*, 41(2):436–463, 2013. 139

[Val84] Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984. 4, 16, 39, 108

[Val08] Paul Valiant. *Testing Symmetric Properties of Distributions*. PhD thesis, Massachusetts Institute of Technology, 2008. 24

[Vas24] Arsen Vasilyan. *Enhancing Learning Algorithms via Sublinear-Time Methods*. PhD thesis, Massachusetts Institute of Technology, 2024. 165

[Vaz22] Vijay V. Vazirani. Online Bipartite Matching and Adwords. In *International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 241, pages 5:1–5:11. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. 145

[VB96] Lieven Vandenberghe and Stephen Boyd. Semidefinite Programming. *SIAM Review*, 38(1):49–95, 1996. 182, 266

[VCB22] Matthew J. Vowels, Necati Cihan Camgoz, and Richard Bowden. D'ya Like DAGs? A Survey on Structure Learning and Causal Discovery. *ACM Computing Surveys*, 55(4):1–36, 2022. 35, 135

[vdLR06] Mark J. van der Laan and Daniel Rubin. Targeted Maximum Likelihood Learning. *The International Journal of Biostatistics*, 2(1), 2006. 140

[vdZLT14] Benito van der Zander, Maciej Liśkiewicz, and Johannes Textor. Constructing Separators and Adjustment Sets in Ancestral Graphs. In *Conference on Uncertainty in Artificial Intelligence (UAI) Workshop on Causal Inference: Learning and Prediction*, page 11–24. CEUR-WS.org, 2014. 134

[Ver10] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010. 20

[Ver12] Roman Vershynin. Lectures in Geometric Functional Analysis, 2012. 18

[Ver18] Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge University Press, 2018. 14, 18

[VP90] Thomas Verma and Judea Pearl. Equivalence and Synthesis of Causal Models. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 255—-270. Elsevier, 1990. 29

[VV11] Gregory Valiant and Paul Valiant. The Power of Linear Estimators. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 403–412. IEEE, 2011. 250

[VV17] Gregory Valiant and Paul Valiant. Estimating the Unseen: Improved Estimators for Entropy and Other Properties. *Journal of the ACM (JACM)*, 64(6):1–41, 2017. 162

[Wai19] Martin J. Wainwright. *High-Dimensional Statistics: A Non-Asymptotic Viewpoint*. Cambridge University Press, 2019. 12, 22

[WBL21] Marcel Wienöbst, Max Bannach, and Maciej Liśkiewicz. Extendability of causal graphical models: Algorithms and computational complexity. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 1248–1257. Proceedings of Machine Learning Research (PMLR), 2021. 29, 100, 202

[WD19] Janine Witte and Vanessa Didelez. Covariate selection strategies for causal inference: Classification and comparison. *Biometrical Journal*, 61(5):1270–1289, 2019. 140

[WD21] Samir Wadhwa and Roy Dong. On the Sample Complexity of Causal Discovery and the Value of Domain Expertise. *arXiv preprint arXiv:2102.03274*, 2021. 139

[Wei20] Alexander Wei. Better and Simpler Learning-Augmented Online Caching. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 60:1–60:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. 36

[WLW20] Shufan Wang, Jian Li, and Shiqiang Wang. Online Algorithms for Multi-shop Ski Rental with Machine Learned Advice. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8150–8160, 2020. 36

[WN11] Christian Wulff-Nilsen. Separator Theorems for Minor-Free and Shallow Minor-Free Graphs with Applications. In *Annual Symposium on Foundations of Computer Science (FOCS)*, pages 37–46. IEEE, 2011. 28

[Woo05] James Woodward. *Making Things Happen: A Theory of Causal Explanation*. Oxford University Press, 2005. 5

[WRJ$^+$23] Qing Wang, Jesus Rios, Saurabh Jha, Karthikeyan Shanmugam, Frank Bagehorn, Xi Yang, Robert Filepp, Naoki Abe, and Larisa Shwartz. Fault Injection Based Interventional Causal Learning for Distributed Applications. In *AAAI Conference on Artificial Intelligence (AAAI)*, pages 15738–15744. AAAI Press, 2023. 188, 189

[WS20] Yuhao Wang and Rajen D. Shah. Debiased Inverse Propensity Score Weighting for Estimation of Average Treatment Effects with High-Dimensional Confounders. *arXiv preprint arXiv:2011.08661*, 2020. 141

[WSYU17] Yuhao Wang, Liam Solus, Karren Dai Yang, and Caroline Uhler. Permutation-based Causal Inference Algorithms with Interventions. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5824–5833, 2017. 135

[WY19] Yihong Wu and Pengkun Yang. Chebyshev polynomials, moment matching, and optimal estimation of the unseen. *The Annals of Statistics*, 47(2):857–883, 2019. 162

[WZZG14] Changzhang Wang, You Zhou, Qiang Zhao, and Zhi Geng. Discovering and orienting the edges connected to a target variable in a DAG via a sequential local learning approach. *Computational Statistics & Data Analysis*, 77:252–266, 2014. 139

[YGL$^+$20] Kui Yu, Xianjie Guo, Lin Liu, Jiuyong Li, Hao Wang, Zhaolong Ling, and Xindong Wu. Causality-based Feature Selection: Methods and Evaluations. *ACM Computing Surveys*, 53(5):1–36, 2020. 140

[YNB$^+$22] Steve Yadlowsky, Hongseok Namkoong, Sanjay Basu, John Duchi, and Lu Tian. Bounds on the conditional and average treatment effect with unobserved confounding factors. *The Annals of Statistics*, 50(5):2587–2615, 2022. 140

[YZW$^+$08] Jianxin Yin, You Zhou, Changzhang Wang, Ping He, Cheng Zheng, and Zhi Geng. Partial orientation and local structural learning of causal networks for prediction. In *Workshop on the Causation and Prediction Challenge at IEEE World Congress on Computational Intelligence (WCCI)*, pages 93–105. Proceedings of Machine Learning Research (PMLR), 2008. 139

[ZARX18] Xun Zheng, Bryon Aragam, Pradeep Ravikumar, and Eric P. Xing. DAGs with NO TEARS: Continuous Optimization for Structure Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 9492–9503, 2018. 135

[ZBHK24] Zhenghao Zeng, Sivaraman Balakrishnan, Yanjun Han, and Edward H. Kennedy. Causal Inference with High-dimensional Discrete Covariates. *arXiv preprint arXiv:2405.00118*, 2024. 109, 111, 132, 238

[Zha05] Fuzhen Zhang. *The Schur Complement and Its Applications*. Springer, 2005. 270

[Zha07] Jiji Zhang. Generalized Do-Calculus with Testable Causal Assumptions. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 667–674. Proceedings of Machine Learning Research (PMLR), 2007. 32

[ZPX$^+$19] Xiang Zhou, Xin Peng, Tao Xie, Jun Sun, Chao Ji, Dewei Liu, Qilin Xiang, and Chuan He. Latent error prediction and fault localization for microservice applications by learning from system trace logs. In *ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, pages 683–694. Association for Computing Machinery (ACM), 2019. 188