Spectral Techniques for Average-Case Complexity

Jeff (Sichao) Xu

CMU-CS-25-137
August 2025

Department of Computer Science
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania

Thesis Committee
Pravesh K. Kothari (Chair, CMU/ Princeton University)
Aayush Jain
Ryan O'Donnell
Madhur Tulsiani (TTIC & the University of Chicago)

This research was sponsored by the National Science Foundation under award numbers CCF2047933 and CCF2211971, and a CyLab Presidential Fellowship. 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 U.S. government or any other entity.

Submitted in partial fulfillment of the Requirements for the degree of Doctor of Philosophy

Keywords: Average-Case Complexity, Spectral Techniques, Sum-of-Squares Algorithms, Random Matrix Theory

To Sarah and my parents.

Acknowledgements

First and foremost, I would like to thank my advisor Pravesh Kothari - his wavering support and overflow of energy and optimism help sustain me through the darkest hours. Beyond that, I am grateful to the freedom he grants me for exploring my own interests, while he serves as a constant reminder of the broader picture in research and in life to ensure that I stay somewhat on track, despite being a reckless driver myself. I hope to continue drawing inspiration and guidance from him in all my endeavors, even after I depart from the shelter of his wings.

At the same time, I owe a heartfelt thanks to Aaron Potechin, who has been like a pseudo-advisor to me over the years and is undoubtedly one of my closest collaborators. Though we have met in person for at best a few times over the years, thanks to Skype and later Zoom. our routine weekly meetings have been a steadfast part of my life for the past 5 years. Needless to say, I would not have sticked so far in lower bounds without Aaron's guidance, support and companionship along the journey.

I am also deeply grateful to Aayush Jain for all his support, especially after Pravesh's move, as well as for being a constant source of fascinating problems from cryptography. I would also like to thank Ryan O' Donnell and Madhur Tulsiani for serving on my thesis committee and for their time and patience during my proposal and dissertation talks. This thesis would not have been possible without their encouragement and invaluable feedback. In addition, many thanks go to Elaine Shi for stepping in at the last minute to supervise my writing requirement.

Last but not least, thanks are due to my mentors and advisors Siu On Chan and Prasad Raghavendra during my undergrad years who intrigued a young kid into this particular area of research, and gave him a glimpse of the broader landscape of ideas to explore. Admittedly, Prasad's casual remark on our way to lunch "lower bounds are more interesting, right?" stuck with me and has stayed in my heart ever since.

Throughout the years, I am indebted to many friends and collaborators who have made research both approachable and fun. I would like to express my gratitude to all of my collaborators over the year, even if we might have not written a paper together yet. In particular, I am grateful to Sidhanth Mohanty for guiding me into research hand by hand, and to my neighbor Tim Hsieh for the countless technical discussions that always managed to intertwine research, NBA, politics, and most importantly, love advice. Special thanks go to Tarun Kathuria and Goutham Rajendran for the many late-night conversations that instilled in me faith in both research and life, and more recently, for encouraging and urging me to explore machine learning instead. I am also grateful to Mitali Bafna, Chris Jones, and Xinyu Wu for bearing with me over the years. Outside immediate collaborations and even research, I would like to thank my friends Bill Cai, Antares Chen, Benny Jiang, Praneeth Kacham, Lingjing Kong, Bingbin Liu, Xiao Ming, Mesut Yang, and Fred Zhang. In addition, I am grateful for having met everybody in Berkeley and in Pittsburgh.

Finally, I thank my parents for their unconditional support at every step along this journey, and Sarah for all her understanding and support. Thanks for coming into my life and lightening up my world.

Abstract

In recent years, algorithmic challenges across diverse areas including statistical physics, machine learning and cryptography have centered around statistical inference problems, i.e., computational problems with average-case inputs. For many of these problems, the best-known efficient algorithms are often suboptimal, giving rise to information vs. computation gaps, discrepancies between what is theoretically possible given the amount of information and what can be attained via efficient algorithms. One fundamental question is how we can provide rigorous evidence of hardness to show that such gaps are insurmountable for efficient computation.

In this thesis, we illustrate that many of these questions boil down to the study of random matrices that have entries being polynomials of the underlying input. More concretely, we highlight the recent advances in the past few years that lead to a significantly more refined understanding of these ostensibly complicated matrices, and some intriguing questions around them that still remain after years of attacks. The sharper understanding of these matrices ultimately allows us to provide rigorous evidence via the lens of the Sum-of-Squares (SoS) algorithms, a hierarchy of semidefinite programmings. Unlike several other popular models in the average-case setting (eg. low-degree polynomials/statistical-query/ overlap-gap). Sum-of-Squares algorithms are known to capture various optimal algorithms in both the average and worst-case setting, and therefore provide one of the strongest form of hardness in average-case complexity.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation from Average-Case Complexity

The major goal of this thesis is to investigate the terrain of average-case complexity, and provide rigorous hardness evidence for information-computaton gap, *the regime in which a computational problem can be solved information theoretically without runtime concern, yet we believe there is no efficient algorithm to do so*, via hardness result against the Sum-of-Squares hierarchy of convex relaxations as a proxy.

Independent-Set on Sparse Random Graph This is a canonical problem in average-case complexity that this thesis focused on for a large part. An Erdős-Renyi random graph with average degree d has an independence number of $(1+o_n(1))\frac{2n\log d}{d}$ and a chromatic number of $(1+o_n(1))\frac{d}{2\log d}$ with high probability [COE15, DM11, DSS16]. There is a long line of work focused on finding the best *efficiently certifiable* bounds on these quantities. When the average-degree of the graphs $d = \Omega(\mathrm{poly}(\log n))$, the spectral norm of the (centered) adjacency matrix [FO05] certifies an upper bound on the independence number of $O(\frac{n}{\sqrt{d}})$. These bounds are off from the "ground-truth" by a factor of $\frac{1}{\sqrt{d}}$. Upgrading

eigenvalue based certificates to those based on the "basic" semidefinite programming relaxation yields no asymptotic improvement [CO05a]. Investigating whether efficient certificates that improve on the above bounds exist has been a longstanding and major open question.

More generally speaking, independent-set problem can be viewed as a restricted version of graph-coloring, and community-detection problem in more modern day's terms. For readers familiar with the literature in community detection and Kesten-Stigum threshold from statistical physics [AS15, Abb17a, KMM$^+$13, DKMZ11a, DKMZ11c] , the goal of most work in this thesis, particularly on the hardness side, is to answer the following question

Question 1.1.1. *Can we provide hardness evidence for the regime below the KS-threshold, i.e., can we show efficient Sum-of-Squares algorithms also fail in the possible-yet-hard regime?*

Before we return to this question, let us approach the question of why we might want to consider Sum-of-Squares algorithms as a restricted model of computation to focus on in this work.

Sum-of-Squares as a unified algorithm A unified framework for algorithm design, Sum-of-Squares semidefinite programming hierarchy has proven to be one of the most powerful candidates in recent years. With its root in Positivestellensatz which traces back to Hilbert for more than a century ago, the SoS hierarchy is a family of algorithms of increasingly powerful algorithms parameterized by the corresponding *SoS degree* for polynomial optimization problems [Par00, Las01]. We refer interested reader to [BS14, RSS19, FKP19] for a thorough introduction to Sum-of-Squares algoritms. On the classical side, the SoS reasoning has either produced new state-of-art algorithms [BRS11, GS11, HSS15, MSS16, RRS17] or is known to capture various best-known algorithms that are born prior to the SoS age [GW94, ARV04]. On the other hand, Sum-of-Squares has

also procured various success in high-dimensional statistical inference problems inspired from modern machine learning [MSS16, KSS18, HL18, KKM20, Hop19, BK21]. More than the problem-specific successes, the power of Sum-of-Squares have been shown to be no serendipity as it captures the strongest among all semidefinite programming relaxations among a large class of problems.

SoS as a lens for average-case complexity With greater power comes great responsibility. The power of Sum-of-Squares has also built itself into an outpost for algorithmic intractability as impediment against such strong algorithms may hinge upon fundamental barriers [HKP$^+$17, RSS19]. In the worst-case realm, much remains widely open, even in the degree-4 case: for example, current results do not rule out a $(2-\epsilon)$-approximation for Vertex Cover or a $(0.878+\epsilon)$-approximation for Max Cut, each of which would refute the Unique Game Conjecture in the central stage of worst-case complexity.

In the average-case setting, the Sum-of-Squares paradigm has by now served as a canonical test-bed for algorithmic hardness where the classic theory of NP-hardness no longer applies. Unlike worst-case complexity theory, the web of reductions approach is of limited reach in understanding hardness of average-case problems as it is usually difficult to preserve the distributions under reduction - the central toolkit from worst-case complexity. In such contexts, the smallest degree of Sum-of-Squares (i.e. SoS degree) to solve the question arises as a compelling proxy for computational hardness as SoS is known to capture various the best-known algorithms for combinatorial optimization problems.

On the other hand, the prominence and versatility of Sum-of-Squares has also conferred tremendous difficulty for proving impossibility results against it. The terrain of average-case complexity is drastically different from that in the worst-case setting because life is much less pessimistic on this land: unlike many NP-Hardness results that render

the simplest (eg. greedy) combinatorial techniques optimality, average-case problems such as planted clique or the stochastic block model often admit non-trivial algorithms based on spectral methods and semidefinite programming in comparison. As a result, designing techniques to ascertain average-case algorithmic thresholds the regime where such problems are easy vs where they are computationally hard is extremely challenging while fruitful as it would allow us to pronounce futility for a large class of algorithms among which surprising algorithmic discoveries may be ensconced.

1.2 Meeting Graph Matrix

In the investigation of the above question, it became clear to us that a major technical ingredient that was missing at the time is a more fine-grained understanding of graph matrix, and in particular, the spectral norm of these matrices. On a high level, *graph matrices* is a family of random matrices with each entry being polynomial of the input from the graph sample. We will defer their formal definitions to the latter section.

Prior Results and Bottleneck to SoS Lower Bounds The low-degree polynomials themselves are naturally described by graphs. All prior works beginning with [BHK+16] rely on elegant statements that relate the spectral norms of such graph matrices to natural combinatorial quantities associated with the graphs. However, the precise bounds they achieve turn out to be rather coarse and lose poly $\log n$ factors. Such losses still turn out to give non-trivial results for problems on dense random graphs. However, they trivialize in the setting of random graphs with average degree $\ll \log n$ — our principal interest in this work. While this might appear to be a technical issue, finding methods that do not lose such poly $\log n$ factors was understood to be a major bottleneck in the area. Indeed, even resolving the case of polylog n degree random graphs took several new ideas in the

earlier work [JPR+22] that is not included in this thesis.

Analyzing lower bound witnesses constructed via pseudo-calibration crucially relies on decomposing certain correlated random matrices into a "Fourier-like" basis of *graph matrices* and understanding bounds on their spectral norms. Each entry of such graph matrices is a polynomial in the underlying edge-indicator variables of the input random graph. There has been considerable progress in the past few years in understanding the spectra of such matrices when the input is a *dense* random graph [AMP20, CP20, JPR+22, RT23]. However, when the input is an "ultra-sparse" random graph with a constant average degree, known tools turn out to be rather blunt. At a high level, given the rather complicated structure of graph matrices, prior methods only yield a *coarse* bound on their spectral norms. Such coarse bounds nevertheless suffice for analyzing the lower bound witnesses for problems on dense random graphs (with some polylogarithmic factor losses). On ultra-sparse graphs, however, it turns out to be an absolute non-starter.

Indeed, the limited progress seen so far in the ultra-sparse regime has come about via rather ad hoc techniques. For the "basic SDP" (i.e., degree 2 sum-of-squares), prior works[DMO+19, BKM19, BMR19] relied on the fact that the spectrum of the underlying matrices (that turns out to have essentially independent entries) can be completely understood via some powerful tools from random matrix theory.

More generally, the ultra-sparse regime has been a challenge for understanding algorithmic problems on graphs for a long time. For example, it took a long sequence of works and some sophisticated tools in random matrix theory (e.g., the Ihara-Bass formula and spectral understanding of the *non-backtracking walk matrix* of random graphs) to resolve the algorithmic threshold for *community detection* on the stochastic block model [ABH14, Abb17a, AS18, BKM19, BMR19].

One basic issue that makes the ultra-sparse setting challenging is that the spectrum of random matrices that arise in such settings suffers from a large variance that makes

the analysis tricky. For example, for $d = \omega(1)$, the Erdős-Rényi random graph is almost regular (deviations in degree are negligible compared to the average). When $d = O(1)$, however, this is no longer true: there are bound to be vertices of super-constant degree $\Omega(\sqrt{\log n})$ on the one end and vertices of degree 0 (i.e. isolated vertices) on the other. A trickier issue arises from events with small but non-negligible probability that makes the analysis based on the moment method (the workhorse in such analyses) tricky. For example, it can be shown that for the adjacency matrix A of an Erdős-Rényi graph, even after removing rows and columns corresponding to high degree vertices, for large k, $\mathbb{E}\left[\mathrm{tr}\left((A - \mathbb{E}[A])^k\right)\right]$ is much larger than its typical value due to an inverse polynomial chance of the presence of dense subgraphs.

While such issues have been tackled in the past, with significant effort [MNS18, KMM+13, Mas14, BLM15, MNS15, HS17, LMR22, DdNS21, ZK16, KZ09, DKMZ11a, DKMZ11c] for analyzing the spectra of *basic* matrices associated with ultra sparse random graphs (e.g., adjacency matrices, non-backtracking walk matrices), in our setting the problem is significantly more involved because of our need to understand substantially more complicated *graph matrices* built from sparse random graphs — matrices of polynomial dimension with entries that are low-degree polynomials in the entries of the adjacency matrix of $G(n, d/n)$ and which are invariant (as a function of the adjacency matrix of $G(n, d/n)$) under permutations of $[n]$.

1.3 Results on Spectral Norm Bounds, and Applications

Sharp Norm Bounds in the Ultra-Sparse Regime In the joint work with Pravesh Kothari and Aaron Potechin [KPX24], we build methods that overcome the bottlenecks in the prior works and obtain bounds that sharp *up to absolute constants* on the spectral norms of graph matrices. At a high-level, the trace moment method for bounding the spectral norm of a

matrix relies on counting the contributions of closed walks on the entries of the matrix. A key new high-level idea in our analysis is a certain *localization* of the walk that allows understanding the contribution of a walk from a single step. This localization allows us to obtain a tractable method to bound total contributions to the weighted count of closed walks with negligible losses. This is the focus of chapter 2.

In retrospect, the particular norm bound machinery we develop serves as a starting point of this thesis, as the investigation into sharp bounds of Graph Matrices subsequently leads us to surprising results beyond Sum-of-Squares lower bounds and even average-case complexity at large.

Switching to Random d-Regular Graphs One related question to the above we later explore in this thesis is the algorithmic threshold and hardness on random regular graphs, moving away from the i.i.d. setting of Erdős-Rényi distribution $G(n, \frac{d}{n})$. Concretely,

Question 1.3.1. *How much does the mild correlation among input from random regular graph matter? Do we expect the norm bounds (of Graph Matrix) to be the same as from the i.i.d. setting?*

A priori, this question could go either way as we know the norm bounds of the most basic graph matrix-adjacency matrix are the same, while we are at the same time intrigued by the existence of low-degree polynomial distinguishers that can easily tell these two distributions apart. Since spectral norm, via the trace moment method, is inherently connected to low-degree polynomial of the input, this casts doubts into whether the norm bounds of graph matrices would carry over in a black-box manner.

Based on our work of [Xu25], we answer this question by identifying specific criterions of the underlying "shapes" of graph matrix that govern whether the norm bound would be the same in these two distributions, and thereby give norm bounds for input from random regular graph (even in the case they have different norm bounds from the i.i.d. setting). As an application of our spectral norm bounds, we show that higher-degree

Sum-of-Squares lower bounds for the independent set problem on Erdős-Rényi random graphs can be switched into lower bounds on random d-regular graphs. Our result is the first to address the general open question of analyzing higher-degree Sum-of-Squares on random regular graphs. This is discussed in chapter 4.

Sharp Norm Bounds for Select Graph Matrices from Gaussian Input At this point, it should be pointed out that the graph matrix results mentioned before all apply generally to arbitrary shapes, i.e. graph matrices with general polynomials of the input. The full generality of our norm bounds is in some sense mandated by our motivation from Sum-of-Squares lower bounds, where we are essentially considering all possible low-degree polynomials of the input (eg. consider the independent-set/clique indicator of a graph that is simply a sum of all possible subset of monomials of edges within the set). That said, the full generality of norm bounds is no longer needed in other context. On the , we describe an application of our norm bounds to a high-dimension geometry question that asks the following,

Question 1.3.2. *Given m i.i.d. Gaussian points $v_1, ..., v_m$ in d-dimension, what's the largest m such that we can find an ellipsoid that passes through all these points on the boundary?*

For this particular question, we are able to focus on a specific family of graph matrices (yet still exponentially many thus we do not know these matrices explicitly beyond certain structural properties) from Gaussian input. The conjectural threshold is that one can afford to take $m \sim \frac{d^2}{4}$. In retrospect, proving a certain construction of ellipsoid is "truly" an ellipsoid boils down to proving the associated matrix representation is positive-semidefinite, and can thus be viewed equivalently as a lower bound for basic SDP. By giving norm bounds sharp up to a constant for the matrices we consider , we are able to establish for $m = \frac{d^2}{C}$ for some constant $C > 4$, hence pinning down the question to within

a constant factor. This is based on a joint work with Tim Hsieh, Pravesh Kothari, and Aaron Potechin [HKPX23], and discussed in chapter 3 .

Sharp Norm Bounds for Gaussian Kikuchi Matrices Another example of selected graph matrices that we consider is the family of Kikuchi matrices with Gaussian input in the context of Tensor PCA problem. Given an order-r tensor of the form $T = G + \lambda \cdot v^{\otimes r}$ where G is a random symmetric Gaussian tensor with unit variance entries and v is an unknown boolean vector in $\{\pm 1\}^n$, what's the minimum λ at which one can distinguish T from a random Gaussian tensor and more generally, recover v? As a result of a long line of work, we know that for any $\ell \in \mathbb{N}$, there is a $n^{O(\ell)}$ time algorithm that succeeds when the signal strength $\lambda \gtrsim \sqrt{\log n} \cdot n^{-r/4} \cdot \ell^{1/2-r/4}$. The question of whether the logarithmic factor is necessary turns out to be crucial to understanding whether larger polynomial time allows recovering the signal at a lower signal strength. Such a smooth trade-off is necessary for tensor PCA being a candidate problem for quantum speedups [SOKB25]. It was first conjectured by [WAM19] and then, more recently, with an eye on smooth trade-offs, reiterated in a blogpost of Bandeira [Ban24b, Ban24a].

Based on the joint work with Pravesh Kothari [KX25], we resolve these conjectures and show that spectral algorithms based on the Kikuchi hierarchy [WAM19] succeed whenever $\lambda \geq \Theta_r(1) \cdot n^{-r/4} \cdot \ell^{1/2-r/4}$ where $\Theta_r(1)$ only hides an absolute constant independent of n and ℓ. A sharp bound such as this was previously known only for $\ell \leq 3r/4$ via non-asymptotic techniques in random matrix theory inspired by free probability [BCSvH24]. This is discussed in chapter 5.

Our main technical contribution is applying the aforementioned framework to proving spectral norm bounds on Kikuchi matrices that are tight up to an absolute constant. This is the first example of Kikuchi matrices that we've identified sharp norm bounds so far. Along the way to our result, we also confirm a suspicion that Kikuchi matrices are, in

general, not intrinsically free – a property necessary for application of the recent advances on sharp spectral norm bounds for correlated random matrices.

Application for Condition Number Analysis Another work in which graph matrix arises, *somewhat surprisingly*, is the study of condition number analysis for random input in the context of polynomial decomposition, based on the joint work with Mitali Bafna, Tim Hsieh and Pravesh Kothari in the early years of my graduate school. For establishing polynomial stability of our algorithm in average-case, we prove inverse polynomial bounds on the *smallest* singular value of certain correlated random matrices with low-degree polynomial entries that arise in our analyses. Since previous techniques only yield significantly weaker bounds, we analyze the *smallest* singular value of matrices by studying the *largest* singular value of certain deviation matrices via *graph matrix decomposition*. We include the overview of the strategy in chapter 6 to showcase the application of graph matrices.

1.4 Results in Sum-of-Squares Lower Bounds

We split this thesis into two components, with the first three chapters focusing on obtaining sharp spectral norm bounds of structured random matrices in different settings. That said, in the analysis of Sum-of-Squares lower bounds, a sharp understanding of spectral norm of various graph matrices is usually only a starting point with more technical work in exploring how to use spectral norm bound as a primitive to understand the positivity of the dual witness matrix. In our work with Chris Jones, Aaron Potechin, Goutham Rajendran and Madhur Tulsiani, we establish Sum-of-Squares Lower Bounds for canonical average-case problem including the aforementioned independent-set problem and Densest-k-Subgraph problem on sparse random graphs [JPR+22, JPRX23]. We refer

interested reader to the original arXiv version for these two results as we decide not to include them into this thesis as the result of [JPR+22] is in some sense subsumed by the latter work of [KPX24] for the ultra-sparse regime. On the other hand, the key technical idea of matrix decomposition via Sparse Minimum Vertex Separator in [JPRX23] is also analogous to the latter work of [PX25] that we include into this thesis.

Sum-of-Squares Lower Bounds for Independent-Set in Ultra-Sparse Random Graphs In the joint work with Pravesh Kothari and Aaron Potehcin [KPX24] , we establish SoS lower bound for some higher-yet-constant degree Sum-of-Squares fail to certify the ground truth of independent-set of size $O(\frac{n \log d}{d})$ in ultra-sparse random graph of $G(n, \frac{d}{n})$ for $d = O(1)$. Concretely, we prove that for every $D \in \mathbb{N}$, and large enough constant $d \in \mathbb{N}$, with high probability over the choice of $G \sim G(n, d/n)$, the Erdős-Rényi random graph distribution, the canonical degree $2D$ Sum-of-Squares relaxation fails to certify that the largest independent set in G is of size $o(\frac{n}{\sqrt{d}D^4})$. In particular, degree D sum-of-squares strengthening can reduce the integrality gap of the classical Lovász theta SDP relaxation by at most a $O(D^4)$ factor.

This builds upon the prior work of [JPR+22] that applies for polylogarithmic average degree, and it is the first lower bound for > 4-degree Sum-of-Squares (SoS) relaxation for any problems on *ultra sparse* random graphs (i.e. average degree of an absolute constant). The major ingredient of our work is the norm bounds machinery discussed in chapter 2, while we defer the technical details of application to SoS lower bounds to chapter 7.

Sum-of-Squares Lower Bounds for Exact Coloring While the related problem of Sum-of-Squares bounds for the independence numbers of random graphs has been progressively resolved over the past few years, similar progress on coloring requires tackling the challenge that the natural planted distribution is easily distinguishable from $G(n, \frac{1}{2})$ as

well as new challenges in handling multiple *exact* equality constraints. This leaves the following question open, especially for an optimistic algorithmist,

Question 1.4.1. *Can higher-yet-constant degree SoS somehow utilize the global constraint to deduce that there cannot be simultaneously too many large-ish independent sets in the graph sample, and therefore certify a better bound than the basic SDP?*

Based on the joint work with Aaron Potechin [PX25], we refute the above hope and prove Sum-of-Squares lower bounds for the coloring problem on random graphs. In particular, we show that for all $\epsilon > 0$, if $G \sim G(n, \frac{1}{2})$ then with high probability, SoS requires degree $\Omega(\operatorname{poly}\log n)$ in order to prove that the chromatic number of G is at least $n^{\frac{1}{2}+\epsilon}$. Our result extends analogously for sparse random graphs from $G(n, \frac{d}{n})$ for $\operatorname{poly}\log n \leq d \ll \sqrt{n}$. Despite being a major goal in the study of integrality gaps for the Sum-of-Squares relaxation, before this work, such a result was known only for the basic Lovász-Theta SDP relaxation (equivalently, degree 2 Sum-of-Squares).

In this work, we introduce new technical tools to tackle these challenges. We design a new principled "fix" to the pseudo-expectation values given by pseudo-calibration in order to address the failure of low-degree indistinguishability while still respecting the exact equality constraints. Our analysis of the new construction relies on an approximate matrix factorization technique via a new type of vertex separator which we call rainbow separators. This is the focus of chapter 8.

Chapter 2

A Machinery for Tight Matrix Norm Bounds

2.1 Overview of Our Machinery

The main challenge in analyzing graph matrices is that their entries are highly correlated. In particular, an $N \times N$ graph matrix generally has entries that are polynomial functions of $N^{o(1)}$ bits of independent randomness. This makes analyzing the trace powers of graph matrices complicated. As a result, previous analyses of the norms of graph matrices lose poly-logarithmic factors even when the input graph is a dense random graph. Our main contribution is developing a new method for conducting such an analysis that somewhat surprisingly yields estimates that are sharp up to absolute constant factors.

To describe our new techniques for obtaining the above improved spectral norm bounds, we will focus this overview on two specific and simple examples of graph matrices and then discuss how our ideas extend more generally. Let's start with the formal definition of graph matrices (specialized to the ultra-sparse setting).

Definition 2.1.1 (Fourier character for $G_{n,d/n}$)**.** *Let χ denote the p-biased Fourier character,*

$$\chi_G(1) = \sqrt{\frac{1-p}{p}} = (1+o_n(1))\sqrt{\frac{n}{d}}, \quad \chi_G(0) = -\sqrt{\frac{p}{1-p}} = -(1+o_n(1))\sqrt{\frac{d}{n}}$$

For a subset of edges $H \subseteq \binom{n}{2}$, we write $\chi_G(H) = \prod_{e\in H} \chi_G(e)$.

Definition 2.1.2 (Shape)**.** *A shape α is a graph on vertices $V(\alpha)$ with edges $E(\alpha)$ and two ordered tuples of vertices U_α (left boundary) and V_α (right-boundary) We denote a shape α by $(V(\alpha), E(\alpha), U_\alpha, V_\alpha)$.*

Definition 2.1.3 (Shape transpose)**.** *For each shape α, we use α^T to denote the shape obtained by flipping the boundary U_α and V_α labels. In other words, $\alpha^T = (V(\alpha), E(\alpha), U_{\alpha^T} = V_\alpha, V_{\alpha^T} = U_\alpha)$.*

Definition 2.1.4 (Embedding)**.** *Given an underlying random graph sample G, a shape α and an injective function $\psi(\alpha) : V(\alpha) \to [n]$, we define $M_{\psi(\alpha)}$ to be the matrix of size $\frac{n!}{(n-|U_\alpha|)!} \times \frac{n!}{(n-|V_\alpha|)!}$ with rows and columns indexed by ordered tuples of $[n]$ of size $|U_\alpha|$ and $|V_\alpha|$ with a single nonzero entry*

$$M_{\psi(\alpha)}[\psi(U_\alpha), \psi(V_\alpha)] = \prod_{(i,j)\in E(\alpha)} \chi_G\left(\psi(i), \psi(j)\right).$$

and 0 everywhere else.

Definition 2.1.5 (Graph matrix of a shape)**.** *For a shape α, the graph matrix M_α is*

$$M_\alpha = \sum_{\text{injective } \psi: V(\alpha)\to[n]} M_{\psi(\alpha)}.$$

When analyzing the sum of squares hierarchy, we extend M_α to have rows and columns indexed by all tuples of vertices of size at most $\frac{\mathsf{dsos}}{2}$ by filling in the remaining entries with 0.

Example 2.1.6 (Line-graph graph matrix M_{line})**.** *Define $M_{line} \in \mathbb{R}^{n\times n}$ to be a matrix with zeros on the diagonal and off-diagonal entries $M_{line}[u,v] = \chi_G(u,v)$.*

M_{line} is, up to rescaling, the centered adjacency matrix of a random graph from $G_{n,d/n}$. We will also use the following Z-shape matrix as a slightly complicated example in our discussion in this section.

Example 2.1.7 (Z-shape graph matrix M_Z). *Define $M_Z \in \mathbb{R}^{n(n-1)\times n(n-1)}$ by $M_Z[(i,j),(k,\ell)] = \chi_G(i,k)\chi_G(j,k)\chi_G(j,\ell)$ whenever i,j,k,ℓ are distinct and 0 otherwise.*

Observe that M_Z has $n^2(n-1)^2$ entries that are low-degree polynomials in the underlying $\ll n^2$ bits of randomness and are thus highly correlated. Note also that M_Z is not a tensor product of M_{line} and thus does not admit an immediate description of its spectrum in terms of M_{line}[1].

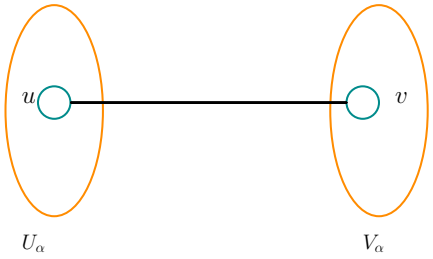


Figure 2.1: Line-graph $M_{line}[u,v] = \chi_G(u,v)$

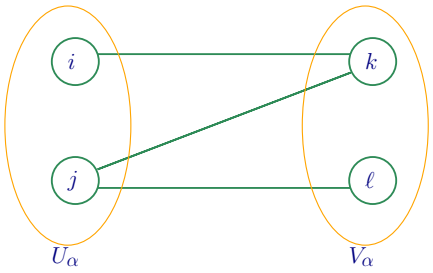


Figure 2.2: Z-shape $M_Z[(i,j),(k,\ell)] = \chi_G(i,k)\cdot\chi_G(j,k)\cdot\chi_G(j,\ell)$

In this section, we will sketch the following two lemmas that are special cases of our main result on spectral norms of graph matrices and illustrate some of our key new ideas. Our spectral bounds hold when M_Z and M_{line} are defined as function of G' obtained from $G \sim G(n,d/n)$ by removing all edges incident on vertices of degree $\geq O(d)$.

Lemma 2.1.8 (Informal version of Lemma 2.3.9, 2.3.10). *With high probability over the random graph sample $G \sim G_{n,d/n}$, we have*

$$\|M_{line}(G')\| \leq O(\sqrt{n}),$$

[1]For the dense case, Cai and Potechin [CP22] showed that surprisingly, the spectrum of the singular values of M_Z has a relatively simple description in terms of the spectrum of the singular values of M_{line}

and

$$\|M_Z(G')\| \leq O(\sqrt{\frac{n}{d}} \cdot \sqrt{n^2}) .$$

where $O(\cdot)$ hides an absolute constant independent of d. Here, G' is obtained from G by removing all edges incident to vertices of degree $\geq O(d)$.

We note that the above bounds are tight up to absolute constant factors (independent of d).

Trace moment method Our proof proceeds by analyzing trace moments of matrices using that for any A, the spectral norm $\|A\|_{spec} \leq \mathrm{tr}\big((AA^\top)^q\big)^{1/2q}$. Taking q to be logarithmic in the dimension of A suffices to get a bound on $\|A\|_{spec}$ sharp up to $1+\delta$ for an arbitrarily small $\delta > 0$. The trace power can be expanded as a sum of weighted walks of length $2q$ over the entries of M_α. Notice that each walk can be viewed as a labeling of a graph composed of q blocks of α and α^T, in particular, it suffices for us to consider the following labelings of walks.

Each term in the expansion of $\mathrm{tr}\big((M_\alpha M_\alpha^\top)^q\big)$ corresponds to a labeling of the vertices in q copies of α and $\alpha^\top$ with labels from $[n]$ (i.e, vertices of the graph G) satisfying some additional constraints that correspond to a valid walk. The following definition captures these constraints.

Definition 2.1.9 (Shape walk and its valid labelings)**.** *Let α be a shape and $q \in \mathbb{N}$. For each walk P, let $G_P(\alpha, 2q)$ be the* shape-walk *graph of the shape-walk P vertices on vertices $V_P(\alpha, 2q)$ formed by the following process,*

1. *Take q copies of $\alpha_1, \ldots, \alpha_q$ of α, and q copies of $\alpha_1^T, \ldots, \alpha_q^T$ of α^T;*

2. *For each copy of α_i (and α_i^T), we associate it with a labeling, i.e., an injective map $\psi_i : V(\alpha) \to [n]$ (and respectively ψ_i' for α^T);*

3. *For each $i \in [q]$, we require the boundary labels to be consistent as ordered tuples, i.e. $\psi_i(U_\alpha) = \psi'_{i-1}(V_\alpha)$ and $\psi'_i(U_{\alpha^T}) = \psi_i(V_{\alpha_i})$;*

4. *Additionally, each such walk must be closed, i.e., $\psi'_q(V_{\alpha^T}) = \psi_1(U_\alpha)$ as a tuple-equality;*

5. *We call each α_i and α_i^T block a block-step in the walk.*

For each walk-graph , we associate it with a natural decomposition $P = \psi_1 \circ \psi'_1 \circ \psi_2 \circ \cdots \circ \psi_q \circ \psi'_q$.

2.2 Motivating our machinery: walk global, think local

Global vs local counts The trace moments are typically analyzed by inferring global constraints on contributing walks. For example, in the analysis for tight norm bound for Gaussian random matrices, the dominant contributing walks have a one-to-one correspondence with the Dyck walk with exactly half of the steps going to “new" vertices, and the other half going to "old" vertices. However, such global analyses become unwieldy once applied to slightly more complicated graph matrices e.g., the Z-shape matrix we defined earlier. Our main idea is a new *local* bound that holds on each step of the walk. This local bound generalizes naturally to graph matrices with more complicated shapes while still giving bounds sharp up to an absolute constant.

For each block-step (a step corresponds to a labeling of a shape α when bounding weighted walks on a graph matrix M_α) in a walk that contributes to eq. (2.1), we will associate two types of charges:

1. vtxcost that is used to identify the labels of the vertices appearing (counting);

2. edgeval that captures the expectation of random variables of edges traversed in the walk;

In total, we will bound the contribution due to block step in the walks contributing to eq. (2.1) as

$$\mathsf{vtxcost} \cdot \mathsf{edgeval} \leq B_q(\alpha)$$

where $B_q(\alpha)$ is some desired upper bound for shape α, which we shall refer to as block-value bound. An assignment of valid block cost $B_q(\alpha)$ for each block translates immediately into a bound for the final trace moment. Formally, we define the block-value function as the following,

Definition 2.2.1. *For any shape α, and $q \in \mathbb{N}$, for any vertex/edge-factor assignment scheme, we call $B_q(\alpha)$ a valid block-value function of the given scheme if*

$$\mathbb{E}[\mathrm{tr}\left(M_\alpha M_\alpha^T\right)^q] \leq \textit{matrix-dimension} \cdot \mathsf{auxcost} \cdot B_q(\alpha)^{2q}$$

for some auxiliary function $\mathsf{auxcost}$ *such that*

$$(\textit{matrix-dimension} \cdot \mathsf{auxcost})^{1/2q} \leq 1 + o_n(1)\,,$$

and for each block-step $\mathsf{BlockStep}_i$ *throughout the walk,*

$$\mathsf{vtxcost}(\mathsf{BlockStep}_i) \cdot \mathsf{edgeval}(\mathsf{BlockStep}_i) \leq B_q(\alpha)$$

We stress that the block-value function $B_q(\alpha)$ depends on the shape α, the length-q, and the vertex/edge-factor assignment scheme. The bulk of our work is in finding a vertex/edge-factor assignment scheme that produces a minimal valid $B_q(\alpha)$ — the *local* component of our argument.

Labeling the steps Our strategy starts by identifying the ”status” of the random variable used in a given block, which we coin "step-label".

Definition 2.2.2 (Edge and step)**.** *We use "edge" to refer to the undirected edge in the underlying random graph sample, and "step" to refer to a directed edge when mentioned in the context of a walk.*

The following vertex labeling scheme is a key component of our new local argument. Each step in the walk corresponds to a vertex labeling of the shape. Each such labeling yields a block contributing equal to the product of the characters on edges appearing in the labeled shape. For each such edge, we will use four types of labels in order to help us construct vtxcost and edgeval.

Definition 2.2.3 (Step-label)**.** *We categorize the status of each step as the following. For a step whose underlying edge/random variable appears at least twice throughout the walk,*

1. *F (a fresh step): an edge (or random variable) appearing for the first time, and the destination vertex is appearing for the* first *time in the walk;*
2. *S (a surprise step/visit): an edge (or random variable) appearing for the first time, and the destination vertex is* not *appearing for the first time in the walk;*
3. *R (a return step): an edge (or random variable) appearing for the last time;*
4. *H (a high-mul step) : an edge (or random variable) appearing for neither the first nor last time.*

For a step whose underlying edge/random variable appears only once throughout the walk, we call the step a singleton *step, and additionally call its underlying edge a* singleton edge*. We will be able to consider a singleton step as a subclass of F steps in our accounting.*

Observation 2.2.4. *Each step receives a single step-label among* $\{F, R, S, H, Singleton\}$.

When working with a graph matrix, a single block-step corresponds to several edge steps (random variables). Hence, we extend the step label to a labeling for an entire block-step,

Definition 2.2.5 (Block-step labeling)**.** *Given a shape α, a labeling $\mathcal{L} : E(\alpha) \to \{F, R, S, H, Singleton\}$ for a block-step is a collection of step labels for for each edge random variable in α.*

Vertex appearance and redistribution We will use a different cost for a vertex label depending on how it appears in a walk. A careful choice of such costs is important.

For example, in the standard argument for bounding the spectral norm of $n \times n$ Gaussian random matrices, each step, if leading to a new vertex, contributes a value of n as the destination vertex takes a label in $[n]$, while on the other hand, if going to a "seen" vertex, contributes a value of 1. In this case, if one applies the block-value bound naively, observe that it would yield a bound of $O(n)$ as opposed to a bound of $O(\sqrt{n})$ with the contribution being dominated by the steps leading to new vertices. For this particular example, this bound can be improved by observing at most half of the blocks attaining a value of n while the other half has a value of 1 thus geometrically averaging out to $\sim \sqrt{n}$.

However, such reasoning does not readily fit into our local reasoning framework. We will instead introduce a vertex redistribution scheme. In the walk, we first formally place vertex's "appearance" into three categories.

Definition 2.2.6 (Vertex appearance in a block-step)**.** *Let v be a vertex in the current block-step. We say that v is making its first appearance if v is not contained in any previous block-steps. We say that v is making its last appearance if v is not contained in any later block-steps. We say that v is making a middle appearance if v appears in both an earlier block-step and a later block-step. Note that we consider a vertex appearing on the block-step boundary (U_α or V_α) as appearing in both adjacent blocks.*

Example 2.2.7. *If $v \in V_i = U_{i+1}$, $v \in V_{j-1} = U_j$ for some $j > i + 2$, and v does not appear anywhere else in the walk then v makes its first appearance in block i, makes middle appearances in blocks $i + 1$ and $j - 1$, and makes its last appearance in block j.*

To handle the disparity in the magnitude due to the step-choices, we adopt the following vertex-factor redistribution scheme.

Vertex-factor redistribution scheme
Each vertex, when it first appears, requires a label in $[n]$. We redistribute this factor as follows.

1. Assign a $\sqrt{n}$ factor to the block-step in which the vertex first appears;
2. Assign a $\sqrt{n}$ factor to the block-step in which the vertex last appears.

Observe that each vertex picks up in the end a factor of n throughout the walk as it gets $\sqrt{n}$ factor each from its first and last appearance, and thus the vertex factor (for identifying new vertices) is preserved. To illustrate the vertex redistribution scheme, we apply the above machinery to get a loose bound for random ± 1 matrix that is tight up to polylogarithmic factors.

Warm up: a loose bound for random ± 1 matrix For the following discussion, let G be an $n \times n$ symmetric matrix with each (off-diagonal) entry sampled i.i.d. from $\{\pm 1\}$.

Lemma 2.2.8 (Naive bound on symmetric random matrices via local argument)**.** *We can bound its block-value function by*

$$B_q(G) \leq \sqrt{n \operatorname{polylog} n},$$

As a corollary, with probability at least $1 - o_n(1)$*, we have* $\|G\| \leq O(\sqrt{n} \operatorname{polylog} n)$.

Proof. Since each edge contributes a value 1 in expectation whenever the walk is non-zero, bounding the trace value reduces from bounding weighted closed walks to simply bounding walk-counts. Hence, it suffices for us to focus on vertex-factors for this example. We start by casing the step-label, and suppose we are traversing from (left) U to (right) V,

1. F-step: the left vertex cannot be making its first appearance by definition, and it cannot be last appearance as otherwise the edge appears only once throughout the walk. The destination vertex is making a first appearance, hence contributing a vertex-cost of $\sqrt{n}$;

2. S/H-step: the left vertex cannot be making its first appearance by definition, and it cannot be last appearance as otherwise the edge appears only once throughout the walk. The destination vertex is making a middle appearance, hence contributing a vertex-cost of $2q = \text{poly}\log n$;

3. R-step: The vertex on the left boundary is potentially making a last appearance, hence contributing a vertex-cost of at most $\sqrt{n}$; the right vertex cannot be making its first appearance as the edge appears in prior steps, and the vertex is not making its last appearance by definition. It can be further specified at a cost of $2q = \text{poly}\log n$;

4. Summing the above, we can bound $B_q(G) \leq 2\sqrt{n} \cdot q = \sqrt{n}\,\text{polylog}(n)$ by setting $q = \text{polylog}\, n$ where we remind the reader that we need to set $q = \Omega(\log n)$ to obtain a bound that holds with probability $1 - o_n(1)$.

□

The challenge of a local argument We note that previous techniques [AMP20, JPR+22, RT23] for analyzing graph matrices give norm bounds that are tight up to $O(\log^{O(|V(\alpha)|)} n)$ loss for any M_α. The main challenge in our new local argument is to obtain bounds tight up to an absolute constant factor.

For readers familiar with trace-method calculation for the example of G.O.E. matrices, or non-backtracking walk matrices for sparse random graph, observe that the bound for the R-step in the above example is lossy when we use a label in $2q$ to specify the destination of an R-step. This is indeed the source of the polylog gap in the above argument. In contrast,

should one imagine the walk proceeds in a tree-like manner, we may consider the F/R steps as either walking towards or away from the root and for the case of an R step walking towards the root, the destination of an R-step is *fixed* as each vertex has only one edge that is moving closer to the root, avoiding a naive cost of $2q$ to specify the destination.

With the above intuition, one still needs to be careful as there is no reason a priori to focus exclusively on walks that are strictly tree-like, and the R-destination is no longer fixed once we start seeing cycles, or more formally, surprise/high-mul steps in the walk. However, as the tight argument for G.O.E. matrices shows, the effect of surprise steps is in some sense *local* as it can be shown that each surprise step can only "confuse" at most 2 different R-steps, i.e., R-steps whose destinations are not unique. To exploit this observation, one may observe that there is a tremendous *gap* opened up by an S-step in the warm-up example, in particular, we have a budget of $\sqrt{n}$ per step, while an S-step only requires a cost of $2q$ to specify the destination. That said, as opposed to paying just a cost of $2q$ to specify the destination, one may additionally pay a cost of $(2q)^2$ to specify the destination of the 2 R-confusions that the S/H-step may incur. By paying this extra cost for S-step (as well as for H-step in a similar fashion), one can assume that *R-destination is fixed* throughout the walk, shaving the polylog overhead. Broadly speaking, we will call the cost used to settle R-confusion Pur-factors/costs (*potentially unforced-return*). The above argument is then formalized as the following,

Lemma 2.2.9 (Tight bound for random ± 1 matrix)**.** *Consider G an $n \times n$ symmetric matrix with each off-diagonal entry sampled i.i.d. from random ± 1 entry. Assuming each S/H-step incurs at most 2 R-confusions, we can bound its block-value function by*

$$B_q(G) \leq (1 + o_n(1)) \cdot 2\sqrt{n}\,,$$

for $q \ll n^{O(1)}$. As a corollary, with probability at least $1 - o_n(1)$, we have $\|G\| \leq (1 + o_n(1)) \cdot$

$2\sqrt{n}$. *This bound is tight even in the leading constant.*

Proof. Identical to above, we focus on the vertex factors, and consider the step-label when we traverse the walk from left to right (i.e. from U_α to V_α).

1. F-step: the left vertex cannot be making its first appearance by definition, and it cannot be last appearance as otherwise the edge appears only once throughout the walk. The destination vertex is making a first appearance, hence contributing a vertex-cost of $\sqrt{n}$;

2. S/H-step: the left vertex cannot be making its first appearance by definition, and it cannot be last appearance as otherwise the edge appears only once throughout the walk. The destination vertex is making a middle appearance, hence can be specified by a cost of $2q$. Moreover, under our assumption that each S/H step incurs at most 2 R-confusions, each of which can be specified by label in $2q$, we have a total cost of

$$\underbrace{2q}_{\text{destination of } S \text{ step}} \cdot \underbrace{(2q)^2}_{\text{destination of potential R-confusions}} \ll \sqrt{n}$$

3. R-step: the vertex on the left boundary is potentially making a last appearance, hence contributing a vertex-cost of at most $\sqrt{n}$; the right vertex cannot be making its first appearance as the edge appears in prior steps, and the vertex is not making its last appearance by definition.

4. Summing the above, we can bound $B_q(G) \leq \sqrt{n} + (2q)^3 + \sqrt{n} \leq (1 + o_n(1)) \cdot 2\sqrt{n}$ by setting $q = \operatorname{polylog} n$.

□

Remark 2.2.10. *In our general analysis, we will utilize that a bound $O(1)$ in the exponent of q would have sufficed for us in the above argument.*

The key to the argument above is that each S/H step incurs at most 2 different R-confusions. We note that this is where our machinery departs from a completely "local" argument, which allows us to improve from prior bounds for graph matrices. On a high level, a key component of our analysis relies upon showing *return-is-fixed*, i.e., there are $O(1)$ choices for the destination of an R-step, and our machinery follows a rule of thumb inspired by the above example,

"If the walk gets too messy to encode, there must be “savings" in the combinatorial factor from previous blocks, that allows us to encode auxiliary information for decoding."

To exploit the loss in previous steps, it is no longer sufficient for us to *narrowly* consider a single local step. Towards this end, we employ a global potential function argument to identify the source of such loss throughout the walk, and more importantly, we incorporate the global potential function into our local block-value bound such that our machinery is ultimately a "local" argument. In particular, the only "global" component of our argument is establishing the assumption that *each S/H step incurs at most* 2 *R-confusions*, and its further generalizations to other steps that come with *gaps* when compared with the dominant block-step (a.k.a. *maximal-value labeling*).

2.3 Technical Overview: Challenges from Ultra-Sparsity

2.3.1 Pruning and Conditioning

A complication that arises from our setting is that the high moments of the trace are in fact too large. As mentioned in the introduction, there are two fundamental sources responsible for this issue: 1) fluctuations in the vertex degrees which have a large effect on the norm (so the improved norm bound is false without pruning high degree vertices); and 2) rare events happening with inverse-polynomial probability that dominate the

contribution to the expected trace power, eg., the event of having a small dense subgraph. To address these two challenges, we work with $\mathcal{G}$ which is a pruned subgraph of a graph sample from $G \sim G_{n,d/n}$, which is further conditioned on the high probability event that the graph has 2-cycle free radius $\Theta(\log_d n)$.

Definition 2.3.1 (2-cycle freeness)**.** *Given a graph $G = (V, E)$, we call it a 2-cycle free graph if for any pair of vertices $u, v \in V(G)$, there are at most 2 simple paths between u and v. Equivalently, each connected component of G has at most one cycle.*

Definition 2.3.2 (2-cycle free radius)**.** *The 2-cycle free radius of a graph is the largest r such that for any vertex v, the induced subgraph of vertices within distance r of v is a 2-cycle free graph.*

Definition 2.3.3 (Indicator functions)**.** *Given a set of vertices $V \subseteq V(G)$, let $1_{\mathcal{F}}[V]$ be the indicator function for each vertex in V having bounded degree, i.e., $1_{\mathcal{F}}[V] = 1$ if every vertex in V has degree at most $c_{degree} \cdot d$ and 0 otherwise. For a shape walk P, we let $1_{\mathcal{F}}[P] = 1_{\mathcal{F}}[V(P)]$ where $V(P)$ is the set of vertices visited by the shape walk.*

We set $\kappa = 0.3 \log_d n$. Let $1_{\mathcal{E}}[G] = 1$ if G has 2-cycle free radius at least κ and $1_{\mathcal{E}}[G] = 0$ otherwise.

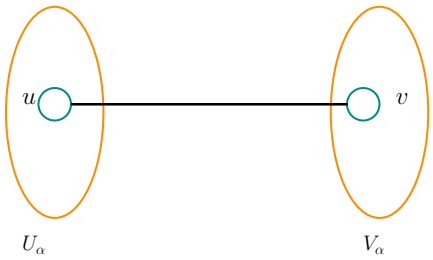
Fact 2.3.4 ([Fri03, BLM15, FM17, Bor19])**.** *A random graph from $G_{n,d/n}$ has 2-cycle free radius at least $\kappa = 0.3 \log_d n$ with high probability $1 - o_n(1)$.*

Therefore, the distribution we work with needs to to be considerably modified, and we are ultimately interested in the following value,

$$\begin{aligned}\mathbb{E}_{\mathcal{G}}\left[\mathrm{Tr}\left((M_\alpha M_\alpha^T)^{2q}\right) \cdot 1_{\mathcal{E}}\right] &= \sum_{\substack{\text{shape walk} P: V(\alpha,2q)\to[n] \\ \text{locally injective at each piece}}} \mathbb{E}_{\mathcal{G}}[\mathrm{val}_{\mathcal{G}}(P) \cdot 1_{\mathcal{E}}\,] \\ &= \sum_{P:\text{shape-walk}} \mathbb{E}_{G\sim G_{n,d/n}}\left[\mathrm{val}_G(P) \cdot 1_{\mathcal{E}} \cdot 1_{\mathcal{F}}[P]\right] \qquad (2.1)\end{aligned}$$

Further challenges from ultra-sparsity The above showcases the bulk of our ideas while extra care is required for our applications for *ultra*-sparse random graphs. To illustrate these challenges, we will use line-graph as an running example and showcase the new ideas needed to migrate the local analysis to sparse random graphs.

For the following discussions, let M_G be an $n \times n$ symmetric matrix with each entry an independent sample from the p-biased distribution. For readers familiar with the $O(\sqrt{d})$ spectral radius bound for sparse random graph with average degree-d, the desired bound translates into a bound of $O(\sqrt{n})$ under the p-biased basis. That said, our goal is to recover an $O(\sqrt{n})$ bound for the following "shape" within our machinery.



Handling singleton edges due to pruning and conditioning For staters, one immediate challenge from the ultra-sparse regime is that we are not working directly with a genuine random graph sample in which each edge is an independent sample: instead, we work with random graphs with pruned vertices, as well as conditioning on the event of 2-cycle-free subgraphs. With this alternative distribution, we need to take into account of *singleton* edges, since it is no longer true the p-biased character has mean-0, as opposed to typical trace-method calculation for mean-0 random variable. This effect is manifest within our framework when we consider a singleton F-step. To illustrate the challenge posed by singleton edges, we first formally describe our edge-factor assignment scheme, inspired by the previous work of [JPR+22].

Edge-value assignment scheme

For each random variable x,

1. F-step: for the first time the random variable appears,
   - Singleton: for random variable x that appears only once throughput the walk, we assign a factor $\sqrt{\frac{n}{d}} \cdot \frac{d}{n} \cdot \mathsf{Singletondecay}$ for $\mathsf{Singletondecay} \leq \exp(-d)$;
   - F: for random variable x that appears at least twice throughout the walk, this gets assigned a factor of 1;
2. H-step: we assign a factor of $\sqrt{\frac{1-p}{p}} \approx \sqrt{\frac{n}{d}}$.
3. R-step: we assign a factor of 1.

At this point, we would like to emphasize that the key to handling singleton edges is that we are able to identify Singletondecay, an extra decay term coming from edge-value for each singleton edge, which we call *singleton-decay*. Furthermore, we show $\mathsf{Singletondecay} \ll \exp(-d)$, providing us with enough slack to offset potential combinatorial blow-up from singleton edges. To highlight the importance of this extra decay term, we consider the local block-value for a singleton F step within our framework.

Hypothetically, suppose there is no singleton decay, we consider the block-step value for the singleton step: the source vertex may be now be making a last appearance, while the destination vertex may be making as well its first appearance. This shall be contrasted with the non-singleton F case in which we pick up only one first appearance factor. Combining the above, we pick a factor of

$$\underbrace{\sqrt{n}}_{\substack{\text{redistributed vertex cost} \\ \text{for the destination's first appearance}}} \cdot \underbrace{\sqrt{n}}_{\substack{\text{redistributed vertex cost} \\ \text{for the source's last appearance}}} \cdot \underbrace{\sqrt{\frac{n}{d}} \cdot \frac{d}{n}}_{\text{singleton-edge value without decay}} = \sqrt{nd}\,.$$

Notice that this yields a block value of $\sqrt{nd}$, which gives us a bound equivalent to the simple row-sum bound. That said, it is crucial to improve the above analysis for singleton step, and in this case, it is clear that the extra decay of $\mathsf{Singletondecay} \leq \exp(-d)$ comes in handy, and allows us to bound the above by $\sqrt{n}$ as desired.

The key technical lemma concerning this component is to show the above is a valid assignment scheme, i.e., we indeed pick up extra decay by unpacking the trace-bound for singleton edges.

Proposition 2.3.5. *The above is a valid edge-value assignment scheme.*

Challenges from unbounded higher-order moments Another challenge due to ultra-sparsity is that p-biased character has unbounded higher moment. In our edge-value assignment scheme described above, each H-step gets assigned a value of $\sqrt{\frac{1-p}{p}} \approx \sqrt{\frac{n}{d}}$ and this poses extra challenges for our previous argument of each S/H step incurring at most 2 R-confusions. In particular, we can no longer afford each H edge incurring unforced return since it no longer comes with any extra gap. We address this issue by improving upon the unforced-return bound, and show that each H-step does not incur any R-confusion.

Besides the effect of H-steps on R-confusion, we also need to be careful in our encoding for destination of an H-step. In particular, even though each vertex has degree at most $c_{trunc} \cdot d$ under the pruned subgraph, a cost of $O(d)$ is still too much for specifying the destination of an H-step. Again, we illustrate this issue by zooming into the block-step analysis for an H-step. With a naive bound of $O(d)$, observe that we would pick up a factor of

$$O(d) \cdot \sqrt{\frac{n}{d}} = O(\sqrt{nd})$$

once combined with the edge-value. This is again a trivial bound comparable to a row-sum bound, however, it also prompts us to improve the encoding cost from $O(d)$ to $O(\sqrt{d})$

for the desired block-value of $O(\sqrt{n})$. In particular, we show that

Proposition 2.3.6. *A cost of $O(\sqrt{d})$ is sufficient for identifying the destination of an H-step.*

To show that a cost of $\sqrt{d}$ is sufficient, we crucially rely upon the assumption that the random graph sample is 2-cycle free in any $\Omega(\log_d n)$-radius neighborhood. The underlying idea also hinges upon another crucial observation for vertex factors, that there are occasions the cost of $O(\sqrt{d})$ may even be spared. To identify such vertices, we make the following formal definition,

Definition 2.3.7 (Doubly constrained)**.** *We call a vertex v doubly constrained by a set of vertices S if there is a path between $i, j \in S$ that passes through v.*

Proposition 2.3.8 (Informal: doubly-constrained vertices are fixed)**.** *Any vertex that is doubly-constrained by a set of known vertices via H steps can be identified at a cost of $O(1)$.*

Combining the above components together, we are ready to introduce our desired vertex-factor assignment scheme,

Vertex-factor assignment scheme: simplified version

Each vertex requires a label in $[n]$ when it first appears. We redistribute factors as follows:

1. Assign a $\sqrt{n}$ factor to the block-step in which the vertex first appears;
2. For any middle appearance,
    - if it is arrived via an S step: assign the corresponding Pur cost and the cost of specifying the vertex (which is again at most $2|V(\alpha)|q$);
    - if it is arrived via an H step and it is not doubly constrained: assign a cost of $O(\sqrt{d})$;
    - if it is reached via an R step or it is doubly constrained by an H-path: assign

a cost of $O(1)$;

3. Assign a $\sqrt{n}$ factor to the block-step in which the vertex last appears.

There are several other technical challenges that arise once we start working with more complicated graph matrices and with matrices that are a sum of a collection of shapes as opposed to a single shape. We defer these discussions to the latter part after we introduce the connection between vertex separators and matrix norm bounds. However, with the ideas already highlighted, we are already able to recover the desired $O(\sqrt{n})$ bound for the line-graph and even get tight norm bounds for some less trivial graph matrices.

2.3.2 Toy examples for sparse matrix norm bounds

Lemma 2.3.9 (Tight bound for scaled adjacency matrix/line graph for $G_{n,d/n}$)**.** *Let G be a random graph sample from $G_{n,d/n}$, and let M_G be its corresponding matrix in the p-biased basis once vertices of degree more than $c_{trunc} \cdot d$ are pruned away. Conditioning on the event G has 2-cycle free radius at least $\kappa = \Omega(\log_d n)$, we can bound the block-value by*

$$B_q(M_G) \leq O(\sqrt{n})$$

for $q \ll n^{O(1)}$. As a corollary, conditioned on the event of 2-cycle free radius being $\Omega(\log_d n)$, with probability at least $1 - o_n(1))$, we have $\|M_G\| \leq O(\sqrt{n})$ where $O(\cdot)$ hides some absolute constant independent of d .

Proof. We analyze the block-value by assuming the technical lemmas highlighted in the above section. Again, we start by casing on the step-label, and assume we are traversing from left (U_α) to right (V_α).

1. Singleton-F step: the vertex on the left boundary may be making last but not first appearance, and the vertex on the right boundary may be making first but not last

appearance. The edge is a singleton edge that gets assigned a value of $\sqrt{\frac{n}{d}} \cdot \frac{d}{n} \cdot$ Singletondecay, combining the above gives,

$$\sqrt{n}^2 \cdot \sqrt{\frac{n}{d}} \cdot \frac{d}{n} \cdot \mathsf{Singletondecay} \leq \sqrt{n} \cdot \exp(-d)\,;$$

2. Non-singleton F and R step: this is identical to their analogs in the warm-up example of random ± 1 matrix, and gets a value of $\sqrt{n}$ each;

3. H-step: neither vertices are making first nor last appearances. Since it does not incur R-confusions, we do not incur any cost of $2q$ (to be contrasted with the warm-up example). Moreover, a cost of $O(\sqrt{d})$ is sufficient to identify the destination, combining with the edge value gives us a bound of

$$O(\sqrt{d}) \cdot \sqrt{\frac{n}{d}} = O(\sqrt{n})\,;$$

4. S-step: this is identical to the analog in warm-up example, and gives a value of $(2q)^3$;

5. Summing the above gives us a bound of $O(\sqrt{n})$.

□

Z-shape: an entry-level graph matrix Now that we have seen how this strategy applies to arguably the most familiar case of the adjacency matrix, we will now power up the above machinery with the Z-shape matrix as illustrated in the following diagram.

As defined, M_Z is a matrix in dimension $n(n-1) \times n(n-1)$, and has its rows/columns indexed by ordered pairs of vertices in $[n]$.

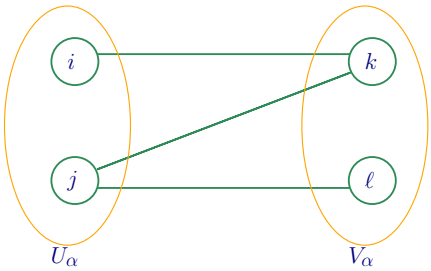


Figure 2.3: Z-shape
$M_Z[(i,j),(k,\ell)] = \chi_G(i,k) \cdot \chi_G(j,k) \cdot \chi_G(j,\ell)$

Lemma 2.3.10 (Norm bound for Z-shape)**.** *We can take the block-value of Z shape to be*

$$B_q(M_Z) \leq O(\sqrt{n}^2 \cdot \sqrt{\frac{n}{d}})$$

As a corollary, with probability at least $1 - o_n(1)$, we have $\|M_Z\| \leq O\left(\sqrt{n}^2 \cdot \sqrt{\frac{n}{d}}\right)$.

Proof. Generalizing from the walk for the adjacency matrix, for the Z-shape, we consider a step in the walk starting from U_α and analyze the factor-assignment scheme introduced above.

1. Observe that i, j cannot be making their first appearance since they are on the left boundary, and k, ℓ cannot be making their last appearance since they are on the right boundary;

2. For simplicity, we ignore the case of singleton edges as they are analogous to the previous case;

3. If vertex k is new, we pick up at most a factor of $\sqrt{n}$ for the vertex factor from k, and an $O(1)$ factor of Pur since one of the edges $(i,k),(j,k)$ is a surprise visit; since both are non-singleton F edges, i, j do not contribute vertex factors as they are not making their last appearances; we pick up at most another factor of $\sqrt{n}$ from ℓ when j is making a middle appearance (either via an F edge to ℓ, or an H edge). This combines to a factor of $\sqrt{n}^2 q^{O(1)}$ for $q \sim \log^2 n$.

4. If vertex k is old, i.e., making a middle appearance,

   - In the case $(i,k) = R, (j,k) = H$, i contributes a factor of $\sqrt{n}$ from its last appearance, and the edge (i,j) is "fixed" when called upon from i and hence does not contribute any factor; the edge j,k contributes a factor of $\sqrt{\frac{n}{d}}$ and no vertex factor since k has already been identified from the R edge on (i,j); this combines to a total of $\sqrt{n}^2 \cdot \sqrt{\frac{n}{d}}$ assuming ℓ is making its first appearance (similar to the case in adjacency matrix), and this is the dominant term;
   - For the case of $(i,k) = H, (j,k) = R$, it is almost analogous, and we pick up at most a factor of $\sqrt{\frac{n}{d}}$ from the H edge, and k is identified for free since it is reached by an R edge;
   - However, we need to further case on whether j is making its last appearance, if not, ℓ may be new and contributes a factor of $\sqrt{n}$ (either via vertex factor, or H edge-value combined with vertex factor of $\sqrt{d}$), and this gives a total of $\sqrt{n} \cdot \sqrt{\frac{n}{d}}$.
   - In the case where j is making its last appearance, there is no vertex factor from j and ℓ cannot be new, hence, it does not contribute a factor of $\sqrt{n}$ (notice (j,ℓ) must be R). This combines to a total of $\sqrt{\frac{n}{d}}$ (notice the tremendous gap from $\sqrt{n}^2\sqrt{\frac{n}{d}}$ even though there may not be any surprise visit locally);
   - In the case both $(i,k), (j,k)$ are H edges, we observe that vertex k can be identified at a cost of $\sqrt{d}$ and we additionally pick up an edge-value of $\sqrt{\frac{n}{d}}^2$ from two H edges; moreover, we may pick up an extra factor of $\sqrt{n}$ from ℓ since j is making a middle appearance; that said, this combines to a factor of $\sqrt{\frac{n}{d}}^2 \cdot \sqrt{n} \cdot O(\sqrt{d}) \leq O(\sqrt{\frac{n}{d}}\sqrt{n}^2)$.

5. Summing over the above choices gives us a bound of $O(\sqrt{n}^2 \cdot \sqrt{\frac{n}{d}})$ as there are $O(1)$ step-labelings.

□

It should be pointed out that this toy example conveys an essential feature coming from our edge-value assignment scheme: despite the lucrative vertex factor of n for a new vertex, it is not always optimal to go to one! Moreover, we want to emphasize this is a local argument as we are traversing the walk and deciding the next step from the current boundary.

2.3.3 Bridging vertex separator, step-labeling and norm bound

In the following section, we will show how we rediscover the polynomial dependence as predicted from prior norm bounds in the ultra-sparse regime, especially in the context of our new technique via step-labeling. We first remind the reader that our main theorem (using the vanilla version) gives the following bound for any shape α,

$$B_q(\alpha) = \max_{S:\text{separator}} c_{\mathsf{norm}}^{|V(\alpha)|} \cdot c_{\mathsf{norm}}^{|E(S)|} \cdot \sqrt{n}^{|V(\alpha)\setminus S|} \left(\sqrt{\frac{n}{d}}\right)^{|E(S)|} \sqrt{n}^{|I(\alpha)|}$$

Let's now zoom back into the examples that we just discuss, and see how the previous bound may fall out of an application of the main theorem in its full generality. We first consider the following construction of vertex-separators from the edge-labeling for the upcoming block,

1. Include any vertex incident to both *non-singleton* F and R edges into S;
2. Include any vertex in U_α incident to some *non-singleton* F edge into S;
3. Include any vertex in V_α incident to some R edge into S;
4. Include any vertex incident to an H edge into S.

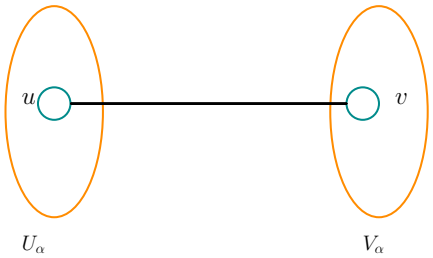


Figure 2.4: Line-graph $M_{line}[u, v] = \chi_G(u, v)$

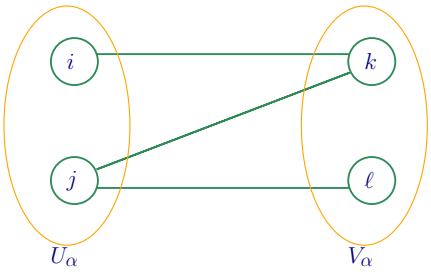


Figure 2.5: Z-shape $M_Z[(i, j), (k, \ell)] = \chi_G(i, k) \cdot \chi_G(j, k) \cdot \chi_G(j, \ell)$

Adjacency matrix revisited In the case of adjacency matrix, the maximal-value edge-labeling is either a (non-singleton) F or an R edge, in which case, one (but not both) of the vertices is included into the separator, while the other vertex remains outside and thus gives a factor of $\sqrt{n}$. In the case of H edges, we note that this can be reduced into the F/R case by turning the edge into an F/R which yields the same block-value while we now have exactly one vertex outside the separator again. As a result, in all these edge-labelings, there is a vertex separator either $u \in U_\alpha$ or $v \in V_\alpha$ that can be constructed and the block-value is dictated by the corresponding vertex separator: each vertex outside the separator continues to contribute a factor of $\sqrt{n}$ as they do in the dense case. In this case, applying the main theorem gives a bound of

$$B_q(M_{line}) = \max_{S:\text{separator}} c_{\text{norm}}^{|V(\alpha)|} \cdot c_{\text{norm}}^{|E(S)|} \cdot \sqrt{n}^{|V(\alpha)\setminus S|} \left(\sqrt{\frac{n}{d}}\right)^{|E(S)|} \sqrt{n}^{|I(\alpha)|} = c_{\text{norm}}^3 \cdot \sqrt{n} = O(\sqrt{n})$$

Z-shape revisited The example of adjacency matrix, despite its simplicity, falls short in conveying another important message that each edge inside the separator contributes a factor of $\sqrt{\frac{n}{d}}$. However, this phenomena has already manifested itself in the Z-shape matrix. Following the dominant edge-labeling in the Z shape that gives labels R, H, F to all three edges from top to bottom, the separator is thus given by (j, k) and the edge inside contributes a factor of $\sqrt{\frac{n}{d}}$ while the two vertices outside contribute a factor of $\sqrt{n}$ each.

In this case, applying the main theorem gives a bound of

$$B_q(M_Z) = \max_{S:\text{separator}} c_{\text{norm}}^{|V(\alpha)|} \cdot c_{\text{norm}}^{|E(S)|} \cdot \sqrt{n}^{|V(\alpha)\setminus S|} \left(\sqrt{\frac{n}{d}}\right)^{|E(S)|} \sqrt{n}^{|I(\alpha)|} = O\left(\sqrt{n}^2 \cdot \sqrt{\frac{n}{d}}\right)$$

Singleton edges affecting the separator? It should be acknowledged that the idea of vertex separators falls slightly short in capturing the potential effect due to singleton edges. In the prior settings, any edge that appears only one throughout the walk would immediately zero out the contribution from the particular walk as this is taking expectation over symmetric random variables with mean 0. However, this is not true in our ultra-sparse setting since we apply conditioning on the degree boundedness of each vertex and the absence of dense subgraph, which unfortunately destroys the perfect balance of the random variable- it is no longer mean 0. That said, as we observe in the previous examples, the bound continue to hold as long as we get a singleton decay that is at most $\frac{1}{\sqrt{d}}$. Put in the context of vertex separators, this reveals that despite the fact that the above construction of S from any edge-labeling does not immediately produce a vertex separator for the shape, its value may still be bounded if we can consider an alternate edge-labeling that 1) yields a higher block-value, and 2) corresponds to an honest vertex separator of the shape if it does not contain any singleton edge.

In our main technical analysis, we show that this is indeed true for components that are connected to U_α and V_α, while singleton edges do indeed have a non-trivial influence when the shape contains floating components, in particular, tree-like floating components. Fortunately in our setting for SoS lower bounds, this turns out to influence a collection of shapes that are previously already "small" in norm, and thus, does not pose qualitively significant differences for our application.

$R-H-F$ **structure lemma** More than just illustrating the power of our factor assignment scheme, these two examples we showcase above may also serve for the broader theme in exposing the connection between step-labeling and vertex separator, and ultimately how this connection enables us to obtain tight norm bounds. As a by-product of our analysis that proceeds by understanding the status of each edge, as opposed to simply by casing on the status of each vertex whether inside the separator, we in fact obtain a more fine-grained understanding of how a walk that maximizes the norm should behave. This is encapsulated in the following structure lemma that we discover. For the clarify of presentation, we restrict our attention to the most interesting collection of shapes that do not contain floating and dangling component, while it can be extended to capture those shapes in a straightforward fashion.

Lemma 2.3.11 (Structure lemma for graph matrix)**.** *Given a shape α, and for a fixed separator S of α, fix the traversal direction from U_α to V_α, the maximum-edge-labeling is given by*

1. *Label each edge that can be reached from U_α without passing through S an R edge;*
2. *Label each edge inside the separator an H edge;*
3. *Label each edge that can be reached from V_α without passing through S an F edge.*

As evident in the two examples we showcase, any F edge before reaching the separator in fact corresponds to a surprise visit so that we can immediately improve the value locally by flipping it to an R edge, assuming there is a large enough gap created by surprise visits, which is $2|V(\alpha)|q_\alpha \ll n^\delta$ for some tiny constant $\delta > 0$ versus the $\sqrt{n}$ contribution if we avoid the surprise visit. Analogously, any R edge that is used after we pass through the separator render a gap we can improve as flipping it to an F edge can be shown to give us an additional vertex contribution of $\sqrt{n}$. Finally, each edge in $E(S)$, if turned to an H edge from F/R, now contributes a factor of $\sqrt{\frac{n}{d}}$ which again allows us to attain a higher block-value.

2.3.4 Overview for unforced-return bounds and high-mul balance

At this point, it should be noted that there is an adjustment which we make to this high-level picture. For the polynomial factors, it is most convenient to have the R edges be the last time the edge appears. However, for analyzing the factors of d and q, it is more convenient to instead have the R edge be the second time that the edge appears and have the subsequent appearances be H edges. We make this switch for the remainder of the analyzing vertex-encoding-cost in the norm bound analysis. With this in mind, we now return to fill in the technical details that we defer from the earlier section, in particular,

Why can an R-step be specified in $O(1)$, i.e., why is an edge being used the second-time forced?

and additionally,

Why is a cost of $\sqrt{d}$ sufficient to specify an H edge, which has already appeared ≥ 2 times?

Forced-return bound

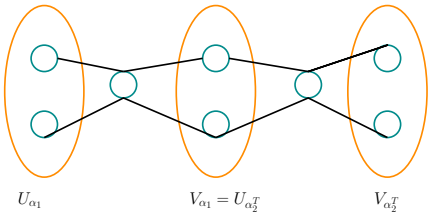
Starting from line-graph To see that an edge being used the second-time is fixed, we may first trace back to what happens if the walk takes place on a tree where things are considerably simpler. In this case, each step in the walk either goes away from the root or backtracks towards it, and moreover, each vertex has only 1 edge which backtracks to the root (which is the edge that discovers the vertex). If we further know that each edge appears only twice in the walk, i.e., no branch is traversed repeatedly, then this is indeed the second-appearance of the edge . As a result, return is forced if the walk proceeds on a tree, or more generally, on an arbitrary graph where the walk proceeds in a tree-like manner.

However, although a sparse random graph is locally tree-like, since we take a long walk of length $q = \Theta(\log n)$, it is unrealistic for us to impose a tree-like assumption on the walk. That said, to generalize the argument for cycles, we observe that each cycle formed in the walk results from an edge leading to a visited vertex, which we call a *surprise visit*, and such an edge can be much more succinctly encoded using a label in $q = \Theta(\log n)$ than its counterpart of a new edge leading to a new vertex. From this perspective, the tremendous gap opened up prompts us to show that as long as each surprise visit does not create "too many" unforced-returns, as we we can then use the gap from corresponding surprise visits to encode auxiliary data so that each return, despite the possible confusion caused by surprise visits, continues to be fixed.

Towards this end, we focus on how the number of unforced return legs from a vertex v may change if we depart and arrive back at the vertex again assuming there is no H edge involved in the walk and each edge appears at most two times. Suppose we depart the vertex by using a new F edge, the subsequent arrival either closes an R edge, or it is a surprise visit which comes with a gap. Moreover, the subsequent surprise visit may see an increment of at most 2 unforced-return from the particular vertex v: with one being the most recent departure F edge, and another one being the surprise visit in the current arrival. That said, each surprise visit can be used to charge at most 2 unforced-returns. This argument turns out to be surprisingly powerful beyond the simple line-graph case, and this formalized in Section 2.4.3 and further extended to capture the influence from H edges by considering H-component as opposed to a single vertex.

Extending to branching vertices with mirror copy The above argument addressees unforced-return restricted to vertices that push out at most 1 F edge when it is on the boundary, however, this is not always the case for graph matrices. For example, consider the middle vertex in the following shape: fix the traversal from left to right, it pushes

out 2 edges in each block. For convenience, we call each such vertex a *branching* vertex. They fall out of reach of the argument we outline above, and we indeed need an upgraded argument to take into account of such vertices, as they are also crucially why the graph matrix norm bounds in the dense setting is governed by *Minimum Vertex Separator*, i.e., the separator of the shape with the fewest number of vertices.



It turns out that such vertices can be easily addressed, at least in the case when apply trace-method for a single shape, since we know that each block of α is followed by its symmetric copy α^T in the subsequent block, and this enables us to use a completely local argument for understanding the influences of such vertices.

For each branching vertex, look at its copy in the subsequent block, either all the paths are mirror-copies of each other, or there is a surprise visit (since locally we have cycles).

With the above observation, in the case each path is a mirror-copy of each other, i.e., each path goes to the boundary and immediately backtracks, each F edge that gets opened up is immediately closed, and hence does not contribute any confusion to unforced-returns. On the other hand, suppose some path does not backtrack immediately, there is a surprise visit along the path that can be assigned to *charge* the corresponding confusion it may incur, and we formalize this via the *missed-immediate-return* function.

High-mul balance: $\sqrt{d}$ is sufficient for H-step

With the unforced return bound settled, we may now proceed to potentially the last missing component to complete our matrix norm bounds for graph matrix of a fixed

shape. Under our bounded-degree graph assumption of the underlying random graph sample, it is not immediately true that we only consider walks such that each vertex has bounded degree. Throughout this section, for convenience, let d be the threshold of degree truncation. It is not true that we constrain our walks to use at most d edges for each vertex: in fact, each vertex may have $q \sim \Theta(\text{poly}\log(n))$ edges in the walk, however, the bounded degree assumption allows us to deduce a decay whenever some vertex has degree more than d: in particular, our edge-value bound holds by assuming each edge appears in the random graph, and thus gives a spike of $\sqrt{\frac{n}{d}}$. However, given the prior that some edge is missing, we can in fact assign a decay for each edge that is missing, and creates a tremendous gap as each non-edge contributes a factor of $-\sqrt{\frac{d}{n}}$ instead. That said, it still suffices for us to assume each vertex to be incident to at most d edges in the walk.

However, this is not quite sufficient: for example, in the case of adjacency matrix, this allows us to avoid a pessimistic bound of $O(\text{poly}\log n)$, while not entirely sufficient towards getting a meaningful bound of $O(\sqrt{d})$, as a priori, a label in $[d]$ is needed for each H edge which would then lead to the trivial bound of $O(d)$ equivalent to a row-sum bound. That said, getting the extra *square-root* gain is crucial in the ultra-sparse regime. We now restate the proposition from earlier section,

Proposition 2.3.12 (Restatement of proposition 2.3.6)**.** *A cost of* $\sqrt{c_{degree} \cdot d} = O(\sqrt{d})$ *is sufficient for identifying the destination of an H-step.*

Towards this goal, we apply our conditioning that the graph has no nearby-2-cycle at radius $c \log_d n$ for some constant $c > 0$, and give a potential-function argument for showing a cost of $\sqrt{d}$ is sufficient. For the example of the adjacency matrix, assuming there is no surprise visit involved (as otherwise we get an extra gap), we can split the walks into intervals of at most $\frac{c}{2} \log_d n$. For each interval, suppose all edges used throughout the interval have been revealed by the beginning of the interval, it suffices for us to identify

a label in $[q]$ that represents the end of the interval, which is possible as we assume each edge has been revealed. Since the walk is locally 2-cycle free within the interval, a label in $[2]$ is sufficient to identify the non-backtracking segment from the start to the end of the interval, and observe that

1. We can partition the chunk of H edges into either main-path (non-backtracking segment) and off-track segment attached to the main-path (backtracking segment);

2. Each H edge on the main-path is fixed once the start and end point is identified;

3. Each backtracking edge requires d going away off-track. This is fixed when traversing back towards the main path and thus give an average of $\sqrt{d}$;

4. The additional label in $[q]$ is then offset by the long chunk as we have $q^{O(1/\log_d)n} \leq 1+\epsilon$ for $q \leq n^{\delta}$ for some constant $\delta \leq O(\frac{1}{\log d})$.

2.3.5 Graph matrix norm bound for grouped shapes

Single trace-method for a collection of various shapes As we shed light upon in the overview for our PSDness analysis, our moment matrix consists of $\Omega(\exp(\mathsf{dsos}\log n))$ many shapes. Despite the intuition in prior works that apply conditioning to reduce dense shapes to sparse shapes, the analysis in [JPR+22] inevitably loses extra polylog(n) factors due to its edge-reservation idea. On a high-level, for ease of the technical analysis, it is helpful in the prior works to assume the shapes that arise in the charging to have various "nice" properties that preserve the prescribed *minimum vertex separator*. This proceeds in the manner of putting aside $O(V(\alpha))$ edges, while applying conditioning on the rest. Unfortunately, this inevitably leads to a loss of polylog factors as there may still be $O(|V(\alpha)|)$ excess edges, as each of which requires a potential $|V(\alpha)| = \Theta(\text{poly}\log n)$ factor for encoding.

Though it is foreseeable that a more powerful conditioning lemma equipped with combinatorial charging argument would carry through, we observe in this work that it is in fact more convenient to group shapes together (as well as with their shape coefficients from pseudo-calibration) and apply one single trace-method calculation as our spectral norm bounds machinery is amenable to be applied on a sum of graph matrices. In particular, for our final charging, we consider graph matrix for grouped shapes defined as the following,

Definition 2.3.13 (Graph matrix for grouped shapes)**.** *Consider $\mathcal{P}$ a collection of shapes, we define the graph matrix of grouped shapes of $\mathcal{P}$ as*

$$M_{\mathcal{P}} = \sum_{\alpha \in \mathcal{P}} M_{\alpha}$$

The main motivation behind this grouping is the observation that the cost of identifying "excess edges" in the outer union bound is in fact the fundamentally the same as having *surprise visits* in our trace-method calculation, which we know by now that each comes with a gap! That said, we can hope to offset the drawing cost by the gap from surprise visit, and as we will soon see, the cost of encoding these "excess edges" is in the end $O(1)$!

To shed light into our analysis, we note that as we apply trace-method calculation on $M_{\mathcal{P}}$ directly without partitioning on the shapes, a block-walk may indeed behave as the following,

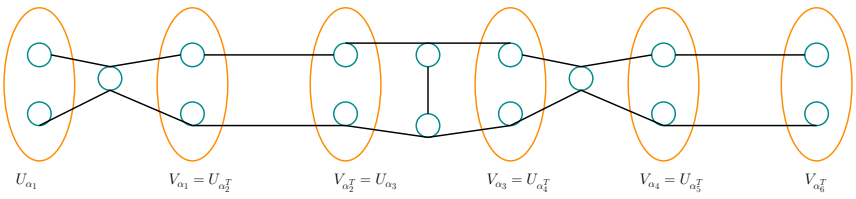


Figure 2.6: Block walk for grouped shapes

Strategy overview for bounding drawing factor At this point, we are ready to extend the block-value analysis to capture additionally the cost of drawing out a shape, and avoid the "potential" encoding cost should one try to apply a triangle inequality in a naive way.

We remind the reader that the encoder knows the shape to-be drawn out, and the decoder has a map of the random graph sample being explored up-to the current timestamp in the walk as well as the edge-labeling of the upcoming block. The task for the decoder is to assign each active vertex in the current boundary a label to help draw out the upcoming shape.

Out strategy proceeds as the following,

1. We extend the block-value bound to additionally take into account of the cost of drawing out the upcoming shape;

2. Given the labeling $\mathcal{L}_\alpha$ for the current α block (which is known to the encoder), we consider a subshape α_H obtained by removing edges from α that receive an F label in $\mathcal{L}$, and remove isolated vertices not in U_{α_H}, and this produces us a list of wedges $W = \{W_i\}$;

3. (Bundle) For each wedge W, we first identify it as an edge set $W \subseteq \binom{n}{2}$ (via claim 2.5.22) ;

4. (Draw) The rest of the shape can be easily drawn out as it either requires a single constant (i.e. going to a new vertex), or it leads to a seen vertex with a new edge and hence is accounted for as a surprise visit.

Branch-chasing: forced-return bound without mirror copy Interestingly, as we extend our block-value bound in order to apply graph matrix norm bounds on a collection of shapes as opposed to a single shape, some component of our prior analysis in fact slightly falls short: in particular, our forced return argument for branching vertex. Note that

previously as we apply trace method for a single shape, we know that each branching vertex that pushes out multiple edges in the current block inevitably faces a pull-back in the subsequent block, and we use this observation to give a lightweight local argument that such branching vertex does not interefer with out forced return bound. However, as shown in the example of fig. 2.6, this is not true for branching vertex anymore (for example, the middle vertex in α_1 pushes out two edges while they don't get immediately pulled back in the subsequent block). In fact, not only is it not true that pull-backs do not immediately happen in the subsequent block, they may never happen either: consider the example where the first block consists of branching vertices pushing out, while any subsequent block proceeds via parallel disjoint paths, and therefore, the paths never rejoin.

Chase confusion We now make this issue concrete. Consider the following example at fig. 2.7, suppose v at the first block branches out to a and b, even though the path to a immediately backtracks, the path to b keeps pushing forward.

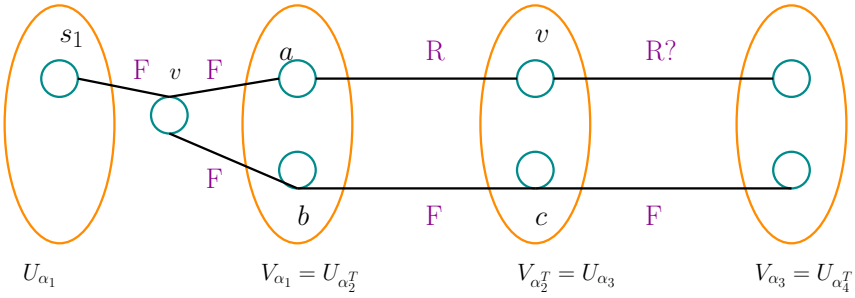


Figure 2.7: Chase confusion

At the end of the second block, if an R edge is used from v for the third-block, it is potentially unclear which edge it points to, as it can be either pointing to s_1 or b, which is then considered a R-chase as we picture it as a walker chasing another.

Put simply in the example of tree, our prior unforced return bound can be seen as saying for each edge, when traversed the second time, it must be traversed in the opposite direction from when it is first created, as seen in the example of backtracking towards the

root (unless there is a cycle that can be charged to it in the general case). However, in this setting, this is no longer true as we may traverse the edge (v, b) twice in the same direction, while there is no surprise visit.

To address this issue, we note that unless there is a surprise visit that can again be charged to this, we can ask each walker on the matrix boundary a single question "whether you are being chased?" Note that for a single walker, this is a question that can be answered in [2] as it is unnecessary to specify which particular walker is chasing. As we additionally maintain a branch-chasing graph that is a tree, we show in section 2.4.5 that a single constant cost per active walker on the boundary (i.e. per vertex on the matrix boundary), is sufficient to determine which edge is being used to chase. As a result, R-chase remains to be fixed up to factor of an absolute constant per vertex in each block.

2.4 Formal Analysis for Bounding Combinatorial Factors

As illustrated in our technical overview, our starting point here is step(edge)-labeling,

Definition 2.4.1 (Swapped Step label for vertex-encoding cost)**.** *A labeling $\mathcal{L}$ for a walk assigns each edge in the length-q walk a label in $\{F, R, S, H\}$, with*

- *F (a fresh step): an edge (or random variable) appearing for the first time, and the destination vertex is appearing for the* first *time in the walk;*
- *R (a return step): an edge (or random variable) appearing for the second time;*
- *S (a surprise step/visit): an edge (or random variable) appearing for the first time, and the destination vertex is* not *appearing for the first time in the walk;*
- *H (a high-mul step) : an edge (or random variable) appearing for neither the first nor the second time.*

By default, we assume $\mathcal{L}$ assigns the labels according to the traversal from U_α to V_α; analogously, we denote $\mathcal{L}^T$ if we traverse $\mathcal{L}$ in a reverse direction from V_α to U_α.

It should be noted here that as we work with vertex-encoding cost in this section, we consider R-step for an edge making its second-appearance.

Remark 2.4.2. *With some abuse of notation, we will also use $\mathcal{L}$ (and analogously $\mathcal{L}^T$) to denote the labeling of a walk restricted to a a single block.*

We now describe our encoding scheme given the edge-labeling. The encoding works by maintaining a set W_t of the vertices in the current block revealed to us already, and walk along edges greedily to expand W_t until all vertices in the walk are encoded, i.e., $W_t = V((\alpha \circ \alpha^T)^q)$.

2.4.1 Vertex encoding scheme

For a fixed set W_t at time-stamp t, and a given shape α, we consider the edges across the cut produced by W_t, let $B_t \subseteq V((\alpha \circ \alpha^T)^q) \setminus W_t$ be the set of unknown vertices incident to some cut edge, and let (i, j) be some edge across the cut with $i \in W_t, j \notin W_t$,

Vertex encoding procedure

1. By default, we set $W_0 = U_\alpha$: any vertex in U_α is the starting point revealed to us from the last block, hence no encoding needed;

2. if there's an R edge (i, j) across the cut, we expand W along this edge o.w.;

3. if there is a path of H edges connected to W across the cut of length at least $\log_d 2q|V(\alpha)|$, split the path into chunks of length at most κ and encode each chunk's boundary in $2q|V(\alpha)|$ with an extra constant in $[2]$ indicating cycle orientation using 2-cycle freeness;

4. if there is some vertex $v \in B_t$ that has (at least) two paths of H edges connected to W, the vertex is doubly-constrained (unless the path crosses U_α or V_α), and all vertices along this path can be encoded using a label in $[2]$ to indicate the orientation of the cycle altogether by using 2-cycle freeness; (specifically, if the path has a combined length of $\ell \gg \kappa$, we need to encode additional vertices in $|2qV(\alpha)|$ with a label in $[2]$ for every κ vertices to identify the entire cycle, however, this is $(1 + o(1))$ per vertex on average); o.w.,

5. if there are multiple high-mul paths crossing U_α or V_α, and it is a mirror copy, we expand along these high-mul paths until the entire mirror copy is encoded into W; o.w.,

6. if there is an H edge (i, j) across the cut, we expand W along this edge;

7. if there is an F edge (i, j) across the cut for $j \notin U_\alpha \cup V_\alpha$, and j is incident to some H edge or R edge outside the cut, or $j \in U_\alpha \cup V_\alpha$ while j is incident to some H edge outside the cut, we expand along this edge; o.w.,

8. expand along the F edges;

9. we additionally require all vertices in the current length-2 block to be decoded before decoding vertices in the new block (except the vertices decoded by doubly-constrained vertices crossing the boundary or by mirror copy);

10. (Excess F edge clean-up) for each length-2 block, if there is any F edge that does not get encoded as the cut edge, we assign an extra label in $|2qV(\alpha)|$ for each such edge indicating their second-time being used (if any).

Remark 2.4.3. *The above encoding scheme applies in a straightforward way for traversing walks in a reversed direction from V_α to U_α.*

We would like to remind the reader that cases 2-5 allow the next vertex to be encoded in $O(1)$ factor, case 6 allows us to specify the next vertex using a factor of d , case 7 allows us to specify the next vertex using a factor of $O(q_\alpha|V(\alpha)|))$, and case 8 requires the full factor of n .

In the remaining of this section, we bound the vertex-encoding cost of the above encoding scheme.

2.4.2 (Non-dangling) Second-use return is forced: overview

In this subsection, we identify the cost of identifying a vertex arrived using some R edge, i.e., an edge that appears for the second-time. At a high-level, the vertex-choice for the destination of such an edge is fixed: for example, when we are traversing a tree from some root, each vertex only has 1 edge that goes towards the root. That said, things might get slightly more complicated when the walk is no longer tree-like. Intuitively, any confusion for determining the vertex reached by an R edge can be settled with a cost of $O(d)$ using a degree bound, however, this would render us a trivial bound of $\sqrt{nd}$ analogous to a row-sum bound.

To get around with this, we note that any such confusion can be charged with a gap in the previous walk, which offers us a gap to encode auxiliary data for settling such confusion. Concretely, we relate the potential confusion in determining destination of a return edge via *surprise visits* [2] , i.e., edges leading to visited vertices in the walk which correspond to some gap in our walk.

Definition 2.4.4 (Potential-unforced-return factor Pur)**.** *At time t, for each vertex v, let $\mathsf{Pur}_t(v)$ be defined as the following,*

1. *Each return leg closed from v contributes 1 to $\mathsf{Pur}_t(v)$;*

[2]The argument in this section also relies upon another quantity *missed-immediately-return* which can be seen as a proxy to surprise visits

2. *Each unclosed F edge currently incident to v contributes* 1;

3. *An extra* -1 *as we can maintain a counter for each vertex such that there is always one potential return-leg presumed to be forced.*

Let $H_{v,t}$ be the component reachable using subsequent H edges from v at time t, and define

$$\mathsf{Pur}_t(H_v) := \sum_{i \in H_{v,t}} \mathsf{Pur}_t(i)$$

Remark 2.4.5. *The definition of $\mathsf{Pur}_t(v)$ does not require v to be on the boundary at time t (similarly for $H_{v,t}$), while we are only interested at the desired invariant when they are at the boundary.*

Definition 2.4.6 (Surprise visit counter)**.** *For a time-stamp t, for each vertex v and $H_{v,t}$ defined above, let $s_t(v)$ be the number of surprise visits arriving at v revealed by time t; we analogously define $s_t(H_v) = \sum_{i \in H_v} s_t(i)$.*

2.4.3 Starting from the basics: a forced-return argument for 1-in-1-out

Proposition 2.4.7. *Throughout the walk where any vertex opens up at most* 1 *edge at the boundary, we maintain the invariant*

$$\mathsf{Pur}_t(H_{v,t}) \leq 2 \cdot s_t(H_{v,t})$$

for $H_{v,t}$ the H-component reachable from the current boundary vertex v.

Overview: unforced-return ignoring H edges For starters, we observe for a single vertex v, if there is no H edge being used throughout the walk and each vertex opens up at most 1 edge on the boundary,

1. For the base case, when v is first revealed, the return from v must be using the edge that explores v, and we have $\mathsf{Pur}(v) = 0$;

2. Each departure from v may add 1 potential-unforced-return edge as it adds an unclosed incidental F edge to v, and departure from v using R edge does not change the Pur factor at v;

3. However, the subsequent arrival may close an F edge that is previously contributing as an unclosed F edge incident to v (though not necessarily the F edge opened by the last departure), and hence $\mathsf{Pur}(v)$ does not change from the last time when v is at the boundary if arriving via an R edge;

4. The arrival, if using a new F edge, is a surprise visit arriving at v and contributes an 1 to s_t and an 1 to Pur on LHS; the bound holds if we assign an increment of 2 via the one single surprise visit.

Incorporating H edges To extend our above argument for using H edges, we give an argument based on H-components which may evolve over time. We first appeal to the following two claims building upon the prior observation that arriving at a vertex via an R edge does not increase its Pur factor compared to when it last appears on the boundary (for the case only a single edge may be opened up),

Claim 2.4.8 (Calling an R edge does not change Pur on vertices already in the current H-comp)**.** *Let v be the current boundary at time t, let $H_{v,t}$ be the H-component of the boundary at time t, and an R edge is called from $H_{v,t}$ (concretely from v)*

$$\sum_{i \in H_{v,t}} \mathsf{Pur}_{t+1}(i) \leq \sum_{i \in H_{v,t}} \mathsf{Pur}_t(i)$$

In other words, despite $H_{v,t}$ is not necessarily the same as $H_{v,t+1}$ since calling an R edge may expand the current H component, it does not increase Pur *factor restricted to the vertices in the current H component.*

Proof. To see the upper bound, observe that calling an R edge from an H component, restricted to the vertices already in the H component prior to the closing of this R edge, we simply have an edge contributing as an unclosed F edge shifted to a closed return leg called from the component. The inequality follows when we close an R edge between vertices already in the same H component. □

Claim 2.4.9 (Getting merged into an H component via an R edge does not change Pur on the annexed vertices)**.** *Let v be the boundary at time t, and let t' be the subsequent time-mark such that $H_{v,t}$ is H-connected to the boundary (at t') again and an R edge is used at this step, i.e., $H_{v,t} \subseteq H_{u,t'}$, the $H_{v,t}$ component gets merged into a H component via an R edge at the current step for u the boundary vertex at t' followed by an R edge*

$$\sum_{i \in H_{v,t}} \mathsf{Pur}_{t'}(i) = \sum_{i \in H_{v,t}} \mathsf{Pur}_{t}(i)$$

Proof. Since we depart from $H_{v,t}$ at time t via an F edge, we have $\mathsf{Pur}_{t_m}(H_{v,t}) = \mathsf{Pur}_t(H_{v,t_m}) + 1$ for any $t < t_m < t'$; however, since we arrive back at a vertex in $H_{v,t}$ at time t' by closing an R edge, we have $\sum_{j \in H_{v,t}} \mathsf{Pur}_{t'}(j) = \sum_{j \in H_{v,t}} \mathsf{Pur}_{t'-1}(j) - 1 = \sum_{j \in H_{v,t}} \mathsf{Pur}_t(j)$, giving us the desired. □

We now proceed to prove the main proposition.

Proof to Proposition 2.4.7. Suppose this is true throughout the walk so far, and we are currently at $H_{v,t}$, and move to u from v at time t, we need to show the above for $H_{u,t+1}$. By construction, it suffices for us to consider whether this is an F/R edge. Notice this is immediate if v is a new vertex, as $H_{u,t+1} = \{u\}$, and there is no potential-unforced-return from u at this time. Otherwise, if the $v \to u$ leg opens up a new surprise visit,

1. If $u \in H_{v,t}$, the new F edge that gets open may contribute 1 to both $\mathsf{Pur}_{t+1}(v)$ and $\mathsf{Pur}_{t+1}(u)$ while Pur factor of other vertices in $H_{v,t}$ does not get changed. That said,

we pick up a gain of 2 on the LHS, while the surprise visit also now contributes to $s_{t+1}(H_{v,t+1})$, giving an increment of 2 on the RHS as well;

2. If $u \notin H_{v,t}$, let t_u be the last time vertex u appears at a boundary H component, and call this component H_{u,t_u}, in addition to the factors contributing to $\mathsf{Pur}_{t_u}(H_{u,t})$, the departure at time t_u opens up 1 F edge from H_{u,t_u}. Plus the new surprise visit that we just open up, we have

$$\begin{aligned}\mathsf{Pur}_{t+1}(H_{u,t+1}) &= \mathsf{Pur}_{t_u}(H_{u,t_u}) + 2 \\ &\leq 2 \cdot s_{t_u}(H_{u,t_u}) + 2 \\ &= 2 \cdot s_{t+1}(H_{u,t})\end{aligned}$$

where we observe that $s_{t+1}(H_{u,t}) = s_{t_u}(H_{u,t_u}) + 1$ as we just pick up a new surprise visit, and $H_{u,t_u} = H_{u,t}$ as a set of vertices.

If $v \to u$ is an R leg that closes some F edge, let t_u be defined as earlier,

1. $u \notin H_{v,t}$: This yields a merge of H components, and we can partition $H_{u,t+1}$ into vertices H-reachable either via u or v at time t,

$$\mathsf{Pur}_{t+1}(H_{u,t+1}) = \sum_{i \in H_{v,t}} \mathsf{Pur}_{t+1}(i) + \sum_{j \in H_{u,t}} \mathsf{Pur}_{t+1}(j)$$

By our claim that calling R edge does not increase Pur on the vertices in the current H component,

$$\sum_{i \in H_{v,t}} \mathsf{Pur}_{t+1}(i) \leq \sum_{i \in H_{v,t}} \mathsf{Pur}_t(i) = \mathsf{Pur}_t(H_{v,t}) \leq 2 \cdot s_t(H_{v,t})$$

Since $u \notin H_{v,t}$, for component $H_{u,t} = H_{u,t_u}$, it gets annexed into a new H-comp via

an F departure and a R return,

$$\sum_{j \in H_{u,t}} \mathsf{Pur}_{t+1}(j) = \sum_{j \in H_{u,t}} \mathsf{Pur}_{t_u}(j) = \mathsf{Pur}_{t_u}(H_{u,t_u}) \leq 2 \cdot s_{t_u}(H_{u,t_u})$$

Combining the above yields

$$\mathsf{Pur}_{t+1}(H_{u,t+1}) \leq 2 \cdot s_{t+1}(H_{v,t}) + 2 \cdot s_{t+1}(H_{u,t_u}) = 2 \cdot s_{t+1}(H_{u,t+1})$$

as $H_{v,t}$ and H_{u,t_u} form a partition for $H_{u,t+1}$ for $u \notin H_{v,t}$, and the surprise visits inherit over time.

2. $u \in H_{v,t}$: This is a return within an H-component, and we have

$$\mathsf{Pur}_{t+1}(H_{u,t}) \leq \mathsf{Pur}_t(H_{v,t})$$

since calling a return leg within the same H component does not increase Pur factor (in fact, a decrease by 1 as the corresponding F edge contributes a factor 1 to both endpoints in the H component, while after being closed, it only contributes as a closed R edge to one endpoint).

□

2.4.4 Branching with mirror copy: a heuristic argument

Branching The above argument crucially relies upon that each vertex can only encode at most 1 edge when it is on the boundary each time, however, this is not true in the general setting for graph matrix, and particularly why rough matrix norm bounds depend crucially on *minimum vertex separator* of the underlying graph. That said, we note that the argument can be extended in a lightweight manner to the general case when the matrix is indexed

by order-tuple, i.e., there is no permutation-jump involved in the walk.

For concreteness, we call a vertex an *anchor* vertex if it encodes multiple (out-going) edges on the boundary at a single time in the walk. The crux of the argument observes that we can further constrain each such vertex to have each of its F edge opens up in the branching to be closed before the vertex appears again on the boundary, as otherwise there is a gap from surprise visit that we can identify locally in the constraint graph, and this is captured by the counter mir to be defined momentarily.

Return bound for tuple-indexed walk Let's start with some definitions in this setting, and fix the traversal direction from U_α to V_α.

Definition 2.4.10 (Anchor vertex)**.** *For a vertex v in the shape α (or α^T) and an edge-labeling $\mathcal{L}$, we call it an* anchor *vertex if it pushes out more than* 1 *non-dangling edge, i.e., an edge on a simple path from v to the next boundary.*

Definition 2.4.11 (Mirror copy)**.** *For any vertex v in α (or in α^T), we let $\sigma(v) \in V(\alpha^T)$ or (in $V(\alpha)$) be the counterpart of v obtained by shape-transpose.*

Definition 2.4.12 (Missed-immediate-return factor counter mir)**.** *For a vertex v that opens up multiple F edges from the boundary, let $\mathsf{mir}_t(v)$ be the number of F edges it opens up as an anchor vertex so far in the walk that are not immediately closed when v appears on the boundary again. Analogously, we define $\mathsf{mir}_t(H_{v,t})$ for the corresponding H-component.*

Remark 2.4.13. *For adjacency matrix or line graph, $\mathsf{mir}_t(v) = 0$ throughout the walk as each vertex may open up at most* 1 *F edge when at boundary.*

Proposition 2.4.14. *For each block, process the anchor-vertices pair according to their distance in increasing order in $\alpha \circ \alpha^T$. For each starting anchor vertex being processed, each path-segment crossing the boundary starting with F edges is either immediately backtracking, or we can assign*

a surprise visit from the path segment (or its adjacent one) to the starting anchor vertex for its corresponding mir *factor. Moreover, each surprise visit is assigned to at most* 2 mir *factors.*

Proof. We first observe that for any path-segment that departs from the starting anchor vertex with F edge may only return to the anchor vertex (with label in $[n]$) within the path-segment at the corresponding mirror copy location (aka. the ending anchor vertex) if the starting anchor vertex (labeled in $[n]$) ever appears again in the path segment. This follows from the injectivity assumption: the starting anchor vertex may only appear in the current block, and unless there is a surprise visit in the segment, the path returning to the starting anchor vertex lands the starting anchor vertex (its label in $[n]$) at the ending anchor vertex location in the constraint graph. That said, for any anchor vertex that gets returned too early, aka. prior to the corresponding mirror copy location, the branch that returns to the current vertex gets assigned a surprise visit and each other path-segment that opens up new F branches departing from the anchor vertex also gets assigned a surprise visit.

Next, we observe that the return to the starting anchor vertex at its mirror location is mandatory for any segment starting with F. Suppose the current path-segment does not return, the starting anchor-vertex picks up 1 mir factor from the path-segment. Suppose there is no surprise visit in the current segment (i.e., the mir factor from the segment is uncharged yet), we claim that this must be the first processed path-segment between the current pair of starting and ending vertices, as otherwise the current path-segment has to arrive at some known vertex at the mirror-copy location which gives a surprise visit to assign to the corresponding mir factor. However, notice for any anchor vertex with path-segment added to its mirror copy in its subsequent block, there are at least two edge-disjoint paths (given by distinct path segments) to its mirror copy. For concreteness, call the second path-segment processed the adjacent of the first segment. Notice these two path segments are disconnected in the graph sample once the starting anchor vertex is removed, while these two paths need to merge eventually at the mirror copy location.

By assumption, the mirror copy is not the starting anchor vertex, hence there must be a surprise visit in the second path-segment and we can charge it to the mir factor of both the first and second path-segments.

Furthermore, to see that a factor of 2 is sufficient, notice each subsequent path-segment between the current pair of starting and ending anchor vertex either comes with a surprise visit or it is an immediate backtracking segment itself that does not contribute mir factor to the starting anchor vertex. This completes the proof of our proposition. □

2.4.5 Branch-chasing using R is free

Troubles for yje immediate-return based argument To recap our prior argument, we essentially give a special treatment for *branching* vertices, i.e., those in the constraint graph that may encode multiple edges across the boundary, and we claim that for any such vertex, each F edge that gets opened up when departing from this vertex must be immediately closed before this vertex appears again in the boundary: in particular, the branching vertex needs to match in its mirror-copy location unless there is a surprise visit that offers a gap for charging potential unforced return due to the vertex opening up multiple F edges simultaneously in a single-shot.

However, this heuristic falls apart when we apply it for grouped shapes, as we no longer have injectivity assumption across the boundary: in particular, mirror copy is no longer well-defined. That said, as highlighted in the overview section, we can consider a branching vertex that simultaneously opens up multiple F branches, and one of them returns sooner than others, causing a potential confusion when calling an R edge at the anchor vertex, that it may refer to any of the F edges leading to the alive F branch (and the F edge leading to the anchor vertex before any of the F branches get opened up).

Branch-chasing graph We start by considering an auxiliary graph constructed as the following. It should be noted that *boundary* in the following section typically refers to the boundary of the walk unless otherwise specified. Notice that each vertex on the boundary gives us a tiny constant decay via their edges. However, the core of our argument relies upon coloring the edges such that red edges are the only possible source of confusions, and furthermore, we maintain the invariants that each vertex is not incident to *too many red edges* and each black directed edge leads to a path to some walker on the current boundary.

Constructing and maintaining the branch-chasing graph

1. This is a directed graph with each vertex receiving a label in $[n]$, moreover, we can color the edges into *red* and *black*, such that we are interested in the desired invariant each vertex is incident to only one out-going red edge unless there are further surprised visits assigned to this vertex;
2. Observe that there is no confusion if there the incident red-edge is *unique*;
3. We start with an empty graph and build the graph as the walk proceeds;
4. For each F edge, we open up a *directed black* edge from the original vertex to the vertex it arrives at;
5. If the F edge that gets opened up is a surprise visit, i.e., it leads to a previously explored vertex, we flip all the *black* unclosed F edges that have a directed path in the branch-chasing graph to the surprise destination to *red*.

With this graph in hand, we first see how it may help us in deciding R edges that arise in the chasing process when there are multiple walkers.

Observation 2.4.15. *If there is no dangling branch, any black edge is on a directed path to the current boundary of the walk, i.e., the boundary of the branch-chasing graph is equivalent to the walk boundary.*

Corollary 2.4.16. *Each chase using a black edge is fixed.*

Proof. Even though each vertex may be incident to multiple black edges, observe that the subgraph cut across by any black edge is a rooted tree, in particular, with some *active* walker in the current boundary. We then consider the following process, at each (block) step in the walk, it suffices for us to go through the vertices on the current boundary and query

1. whether it is a *branching* vertex looking for R chase;

2. whether it is a vertex assigned to *beacon guidance* for R chase;

For each step, for each walker at some branching vertex looking for R chase, the encoder can assign the boundary vertex on the path promised by the out-going R edge as the following,

1. For vertex v looking for beacon-guidance, consider the subtree reached by the desired outgoing edge, assign beacon guidance for any branching vertex with R chase query contained in the subtree;

2. Assign an arbitrary vertex in its subtree reached by the desired out-going edge not assigned for beacon-guidance to v.

To decode, since the branch-chasing graph (restricted to black edges) is a directed tree (forest), there exists some query node whose (out-going) subtree is query-node free; by out encoding, there is also promised to be at least one guidance node in that subtree, assign the outgoing edge pointing to the (potentially multiple) guidance beacon to the query-node. Remove their colors (temporarily), and repeat this process. □

Remark 2.4.17. *It should be pointed out that we only aim to identify the edge-set used by R chase from this process, as any further confusion is then resolved using edge-decay and automorphism factor.*

We now proceed to handle the potential confusion caused by surprise visit in the branch-chasing graph, i.e., red edges. Throughout the process, a surprise visit may turn black edges into red edges, and call this path the corresponding flip path of the surprise visit.

Proposition 2.4.18. *Consider the path of edges that have their colors flipped, this* flip path *passes through at most* 2 *vertices that are incident to* out-going *red edge before the current flip.*

Proof. Without loss of generality, suppose the surprise visit proceeds from $a \to b$, we observe that any edge that gets flipped by this surprise visit. Let u_a be the closest vertex on the flipped path to a that is already incident to some out-going red edge before the current flip, and suppose there is some other vertex v_a that also has a black path to a before the current flip. We observe that it needs to pass through u_a as the subgraph restricted to black edges is a directed tree. However, consider the supposedly black path from v_a to u_a, we claim that it must already be red before the current flip as u_a has an out-going red-edge, which by construction, requires the v_a to u_a path to be flipped to red. Therefore, there can be at most 1 vertex incident to an *out-going* red edge before the current flip that has a flip-path to a; applying the above argument to the flip-path leading to b then gives us the desired bound. □

Corollary 2.4.19. *Assigning each surprise visit to two of the vertices its flipped path passes through that may be incident to more than* 1 *out-going red edge captured by the factor redsurprise, we can maintain the following invariant for each vertex*

$$\textit{out-going red edges incident to } v \leq 1 + w_t(v)$$

where $w_t(v)$ accounts the number of surprise visits assigned to vertex v for creating new excess red edge originating from v.

2.4.6 Return is still "forced" with dangling branches

Before we dig into our final bound, we first see how having dangling edges may affect our forced return bound. Notice that having dangling edges indeed brings Pur factor that does not get assigned to any surprise visit, however, our goal in this section is to show that beyond the "anticpated" Pur factors, dangling edges do not contribute further confusion.

Attaching dangling branches and dangling factor assignment Recall that the return for each dangling branch is not necessarily fixed, and our focus here is to see each dangling branch, despite how many vertices it may pass through, only incur 1 factor of potential-unforced-return.

Let's start by recalling dangling branches (in a shape) can be encoded in the following manner: we can recursively attach a dangling branch to some identified vertex in the shape. And this begets our dangling factor assignment scheme:in each block-step, we assign each dangling factor to the vertex it attaches to, and update the Pur function for each vertex on the newly attached segment via the 1-in-1-out argument. At this point, it should be clear that the Pur factor is associated with the number of dangling branches as opposed to dangling vertices. To formalize this intuition, we introduce attachment edge, as this is the edge part of a dangling branch that comes with extra Pur factor in our argument. Notice not all the edges in a dangling branch come with extra Pur factor, and this is crucial for us in obtaining a norm bound with extra factor only depending on the number of dangling branches as opposed to the number of dangling vertices.

Definition 2.4.20 (Attachment edge)**.** *Given a shape and a block-step walk of the corresponding shape, for each dangling branch, call the edge an attachment edge of the corresponding branch if its origin vertex is identified using R/H edge while it spits out some F edge part of the dangling branch.*

Observation 2.4.21. *For any dangling branch, at any block-step of the walk, there is at most one attachment edge unless there is a surprise visit locally along the branch.*

We now upgrade our edge-coloring scheme in the branch-chasing graph to take into account dangling branch. For each dangling branch that gets encoded, we will color its F edges as the following scheme,

Coloring scheme for dangling branches

1. Following the encoding order that we start from some non-dangling vertex or landing vertex if dangling, and encode the edges along the dangling branch;
2. We restrict our attention to F edges along the dangling branch, and
3. Color each attachment edge *yellow*;
4. Color each non-attachment edge *green*.

Remark 2.4.22. *An out-going green edge is an edge opened as part of dangling branched while it does not get assigned dangling factor.*

Claim 2.4.23. *For any vertex v, let $s_{\mathsf{dangling},t}(v)$ be the number of surprise visits arriving at v by time t that is part of any dangling branch, we have*

$$\textit{out-going green edges from } v \textit{ by time } t \leq 1 + \textit{green}_t(v)$$

and furthermore,

$$\textit{green}_t(v) \leq s_{\mathsf{dangling},t}(v)$$

2.4.7 Ultimate unforced return bound

We now put the above arguments together, concretely, we combine the argument for 1-in-1-out and R-chase being free argument for walks with permutation-jump. Towards

that end, we need to modify the definition of Pur factor to take into account our prior observation that return is still forced if it is an R chase,

Definition 2.4.24 (Time t)**.** *We can associate a particular time-mark to each edge . That said, we use time t to represent the moment before the edge at the corresponding time-mark is used.*

Definition 2.4.25 (Upgraded Pur factor)**.** *At time t (before the edge of time t is used), for each vertex v, let* $\mathsf{Pur}_t(v)$ *be defined as the following,*

1. *Each return leg closed from v contributes* 1 *to* $\mathsf{Pur}_t(v)$*;*
2. *Each unclosed* red *or* green *F edge currently incident to v edge* in the branch-chasing graph *contributes* 1*;*
3. *A* -1 *as we can maintain a counter for each vertex such that there is always one potential return-leg presumed to be forced;*
4. *An extra* $-1 \cdot 1_{v \text{ has seen outgoing red edge by } t}$ *as we can maintain a counter for each vertex such that there is always one outgoing red edge presumed to be forced, and this is only needed when v has outgoing red edge;*
5. *An extra* $-1 \cdot 1_{v \text{ has seen outgoing green edge by } t}$ *as again we can maintain a counter for green edges such that there is always one out-going green edge presumed to be forced, and this is also only needed when v has outgoing green edge.*

Let $H_{v,t}$ *be the component reachable using subsequent H edges from v at time t, and define*

$$\mathsf{Pur}_t(H_v) := \sum_{i \in H_{v,t}} \mathsf{Pur}_t(i)$$

Remark 2.4.26. *We highlight that the only modification is at that each F edge can now be potentially unforced only if it is a red edge in the branch-chasing graph at the moment.*

Definition 2.4.27. *Call an edge protected if it appears as a* black *edge in the branch-chasing graph or it is attached as an attachment edge for some dangling branch.*

Lemma 2.4.28 (Ultimate forced-return bound)**.** *Throughout the walk, we maintain the invariant*

$$\mathsf{Pur}_t(H_{v,t}) \leq s_t(H_{v,t}) + w_t(H_{v,t}) + \mathsf{dangling}_t(H_{v,t}) + green_t(H_{v,t})$$

for $H_{v,t}$ the H-component reachable from the boundary vertex v at the end of time t (after the edge at time t is used), where

1. *s_t is the surprise visit counter that counts the number of surprise visits arriving at $H_{v,t}$;*
2. *w_t is the counter that assigns surprise visits to vertices with multiple red edges in the branch-chasing graph in $H_{v,t}$;*
3. *$\mathsf{dangling}_t$ is the counter that counts the number of dangling branches attached to $H_{v,t}$, i.e., yellow edges;*

Proof. For clarity, in the case a vertex splits out multiple walkers at the block-step boundary at time t, we can process them in an arbitrary order, and notice each F edge splits out does not affect the desired invariant at time t since each such edge that appears in the branch-chasing graph with a walker on the boundary via local injectivity of graph matrix.

Suppose the current edge being used goes from u to v, let time t be the last time mark v appears in the H component at the boundary (before the edge at time t gets used), and call the component $H_{v,t}$. We start by partitioning the unclosed F edges incident to $H_{v,t}$ at time t according to whether they contribute to $\mathsf{Pur}_t(v)$, i.e. whether they appear as a directed edge going out from some vertex in $H_{v,t}$ in the branch-chasing graph at time t. Let those that contribute (i.e., unprotected) be in $O_t(v)$ and the others (i.e., protected edges) be in $B_t(v)$. Notice the last departure from $H_{v,t}$ (not necessarily from vertex v) at time t may create one edge that is either in B or in O at time $t+1$. Our argument relies upon keeping

track of how B and O may change between these two time-marks, and we now case on the status of the leg $u \to v$ at the current time-mark t' (after the edge at t' is used),

1. Again, by construction, it suffices for us to consider the edge used at time t' be $u \to v$ as either F or R, since H edge is automatically captured when we consider H-components;

2. If the walker arrives via an R leg, and assume v, u not in the same H component before this leg as the other case is immediate via prior argument that calling an R edge within the same component cannot increase Pur factor. Therefore, it suffices for us to assume u and v being in disjoint H components at time $t' - 1$, i.e., before the use of the leg $u \to v$, and we can further restrict our attention to $H_{v,t'-1}$ as

$$\mathsf{Pur}_{t'+1}\left(H_{v,t'}\right) = \mathsf{Pur}_{t'+1}(H_{u,t'}) + \mathsf{Pur}_{t'+1}(H_{v,t'}) \leq \mathsf{Pur}_{t'}(H_{u,t'}) + \mathsf{Pur}_{t'+1}(H_{v,t'})$$

   since calling an R edge from an H-component cannot increase the Pur factor on the original vertices, i.e.

$$\mathsf{Pur}_{t'+1}(H_{u,t'}) \leq \mathsf{Pur}_{t'}(H_{u,t'})$$

3. Let E_t be the edge that gets opened up at time t when departing from $H_{v,t}$;

4. Consider the edges that may now contribute to $\mathsf{Pur}_{t'}(H_{t,v})$ while not accounted for at time t, let this set of edges be $\Delta_{t'-t}(H_{v,t})$: they are either the edge E_t if it is opened as a dangling edge, or an edge that shifts from B_t to $O_{t'}$ (as well as potentially non-dangling E_t that shifts from black to red), i.e., they are protected on the branch-chase graph at some time between t and $t' - 1$ as *black* edges, while no longer so at t' since they are now *red* edges;

5. For the edges that shift from black to red within the time interval $[t, t']$, we process

them in an arbitrary order of their out-vertex in $a \in H_{v,t}$, and for each vertex's first edge being processed, we additionally case on whether the out-vertex a has out-going red edge at time t: if not, the first edge flips $1_{a \text{ ever has outgoing red edge}}$ and this offsets the gain to the LHS; otherwise, any such edge assigns a a factor of surprise visit via a gain in $w_{t'}(a)$ over $w_t(a)$, yielding

$$|\Delta_{t,t}(a)| \leq -1 \cdot \left(1_{a \text{ has seen out-going red edge by } t'} - 1_{a \text{ has seen out-going red edge by } t}\right) + (w_{t'}(a) - w_t(a))$$

6. If E_t was opened up as dangling edge, and moreover, as a dangling attachment edge, we pick up a gain in $\mathsf{dangling}_{t'}(H_{v,t})$ over $\mathsf{dangling}_t(H_{v,t})$ that can offset the gain from edge. Otherwise, E_t is a a green edge, i.e., part of some dangling branch while not an attachment edge, E_t is then captured by the factor of $\text{green}_t(v)$: it is either forced as the sole green edge, or it gets assigned surprise visit factors via the function $\text{green}_t(v)$, letting E_t's out-going vertex be a, we have

$$\begin{aligned}
&1_{E_t \text{ is opened as a dangling edge}} \\
&\leq \left(\underbrace{\mathsf{dangling}_{t'}(a) - \mathsf{dangling}_t(a)}_{\text{dangling attachment}} \right) + \\
&\left(-1 \cdot 1_{E_t \text{ is the first out-going green edge from } a} + (\text{green}_{t'}(a) - \text{green}_t(a)) \right)
\end{aligned}$$

7. From the above, for the H-component restricted to vertices H-connected to v at time t', suppose an R edge is used, (and noticing $H_{v,t'} = H_{v,t}$)

$$\begin{aligned}
&\mathsf{Pur}_{t'+1}(H_{v,t'}) - \mathsf{Pur}_t(H_{v,t'}) \leq \mathsf{Pur}_{t'}(H_{v,t'}) - \mathsf{Pur}_t(H_{v,t'}) \\
&\leq |\Delta_{t'-t}(H_{v,t})| + 1_{E_t \text{ is opened up as a dangling edge}}
\end{aligned}$$

$$\leq \sum_{a \in H_{v,t}} -1 \cdot \Big(1_{a \text{ has seen out-going red edge by } t'} - 1_{a \text{ has seen out-going red edge by } t} \Big) + (w_{t'}(a) - w_{t+1}(a))$$

$$+ \underbrace{\mathsf{dangling}_{t'}(H_{v,t}) - \mathsf{dangling}_{t}(H_{v,t})}_{\text{added dangling attachment for } E_t} +$$

$$\Big(-1 \cdot 1_{E_t \text{ is the first out-going green edge from } a} + (\mathrm{green}_{t'}(a) - \mathrm{green}_{t}(a)) \Big)$$

$$\leq \sum_{a \in H_{v,t}} (w_{t'}(a) - w_t(a)) + (\mathsf{dangling}_{t'}(a) - \mathsf{dangling}_t(a)) + (\mathrm{green}_{t'}(a) - \mathrm{green}_t(a))$$

$$= \sum_{a \in H_{v,t}} \Delta_{t'-t} \circ w(a) + \Delta_{t'-t} \circ \mathsf{dangling}(a) + \Delta_{t'-t} \circ \mathrm{green}(a)$$

where we introduce the shorthand

$$\Delta_{t_1,t_2} \circ f(v) \coloneqq f_{t_1}(a) - f_{t_2}(a)$$

for $f \in \{s, w, \mathsf{dangling}, \mathrm{green}\}$ and $t_1 > t_2$. Combining with the component connected to u at time t' gives, (assuming u, v again not in the same H-component at time t')

$$\mathsf{Pur}_{t'+1}(H_{v,t}) = \mathsf{Pur}_{t'+1}(H_{u,t'}) + \mathsf{Pur}_{t'+1}(H_{v,t'})$$

$$\leq \mathsf{Pur}_{t'}(H_{u,t'}) + \mathsf{Pur}_t(H_{v,t'}) + (\mathsf{Pur}_{t'+1}(H_{v,t}) - \mathsf{Pur}_t(H_{v,t}))$$

$$\leq s_{t'}(H_{u,t'}) + w_{t'}(H_{u,t'}) + \mathrm{green}_{t'}(H_{u,t'}) + s_t(H_{v,t}) + w_t(H_{u,t}) + \mathrm{green}_t(H_{u,t'})$$

$$+ s_{t'+1}(H_{v,t'}) - s_t(H_{v,t'}) + \sum_{a \in H_{v,t}} \Big(\Delta_{(t'+1),t} \circ w(a) + \Delta_{(t'+1),t} \circ \mathsf{dangling}(a) + \Delta_{(t'+1),t} \circ \mathrm{green}(a) \Big)$$

$$\leq s_{t'+1}(H_{v,t'}) + w_{t'+1}(H_{v,t'+1}) + \mathsf{dangling}_{t'+1}(H_{v,t'+1}) + \mathrm{green}_{t'+1}(H_{v,t'+1})$$

where we use the property that the counter functions $\{s, w, \mathrm{green}, \mathsf{dangling}\}$ is non-decreasing over time and factorizes over connected components.

8. On the other hand, if the leg $u \to v$ is an F edge, again assuming u, v not being in

the same H-component at time t' and the other case is analogous, restricted to the $H_{v,t'}$, we have

$$\begin{aligned}
&\mathsf{Pur}_{t'+1}(H_{v,t'}) - \mathsf{Pur}_t(H_{v,t}) \leq |\Delta_{(t'+1),t'}(H_{v,t})| + |\Delta_{t',t}(H_{v,t})| + 1_{E_t \text{is opened up as a dangling edge}} \\
&\leq |\Delta_{(t'+1),t'}(H_{v,t})| + \sum_{a \in H_{v,t}} \Delta_{t',t} \circ w(a) + \Delta_{t',t} \circ \mathsf{dangling}(a) + \Delta_{t',t} \circ \mathrm{green}(a)
\end{aligned}$$

where we notice the factors restricted to the last two terms are identical to the previous calculation when R edge is used, and we can focus on the first term $|\Delta_{(t'+1)-t'}(H_{v,t})|$. Notice this set of edges are either the surprise visit we just opened up at time t', or the edges that switch from black to red due to the current surprise visit (eg. potentially the edge E_t that remains black at time t' while now becoming red), and

$$|\Delta_{(t'+1),t'}(H_{v,t})| \leq s_{t'+1}(H_{v,t'}) - s_{t'}(H_{v,t'}) + \sum_{a \in H_{v,t'}} |\Delta_{(t'+1),t'}(a)|$$

Combining the above yields

$$\begin{aligned}
\mathsf{Pur}_{t'+1}(H_{v,t'}) - \mathsf{Pur}_t(H_{v,t}) \leq & s_{t'+1}(H_{v,t'}) - s_t(H_{v,t'}) \\
& + \sum_{a \in H_{v,t}} \Delta_{(t'+1),t} \circ w(a) + \Delta_{(t'+1),t} \circ \mathsf{dangling}(a) + \Delta_{(t'+1),t} \circ \mathrm{green}(a)
\end{aligned}$$

Combining with the invariant for $H_{v,t}$ at time t yields,

$$\begin{aligned}
\mathsf{Pur}_{t'+1}(H_{v,t'+1}) = \mathsf{Pur}_{t'+1}(H_{v,t'}) &\leq \mathsf{Pur}_t(H_{v,t}) + s_{t'+1}(H_{v,t'}) - s_t(H_{v,t'}) \\
&+ \sum_{a \in H_{v,t}} \Delta_{(t'+1),t} \circ w(a) + \Delta_{(t'+1),t} \circ \mathsf{dangling}(a) + \Delta_{(t'+1),t} \circ \mathrm{green}(a) \\
&\leq s_{t'+1}(H_{v,t'+1}) + w_{t'+1}(H_{v,t'+1}) + \mathsf{dangling}_{t'+1}(H_{v,t'+1}) + \mathrm{green}_{t'+1}(H_{v,t'+1})
\end{aligned}$$

□

Putting things together for the final unforced return bound throughout the walk It suffices for us to consider each H-connected component when they last appear in the walk. Let $\{H_i, t_i\}$ be a sequence of H-connected components at the end of the walk, and time t_i their last appearance time-mark, for a shape-walk W, we have

$$\begin{aligned}\mathsf{Pur}(W) &\leq \sum_{H_i,t_i} \mathsf{Pur}_{t_i}(H_{t_i}) \\ &\leq s_{t_i}(H_i) + w_{t_i}(H_i) + \mathsf{dangling}_{t_i}(H_i) + \mathrm{green}_{t_i}(H_i) \\ &\leq 4 \cdot s(W) + \mathsf{dangling}(W)\end{aligned}$$

where we observe that each surprise visit may contribute a 1 to the final bound via s counter, and a factor of 2 via w counter, a factor of 1 via green, and each dangling attachment contributes a 1 via dangling.

2.4.8 (Local) 2-cycle-freeness can be useful

Proposition 2.4.29. *Given the starting vertex, a path of H edges of length t can be encoded at a cost of*

$$(2 \cdot 2q|V(\alpha)|)^{\lceil \frac{t}{\kappa} \rceil}$$

Proof. Since the graph is 2-cycle-free at every κ neighborhood, it suffices to split the long walk into $\lceil \frac{t}{\kappa} \rceil$ chunks, and encode the vertex at the end of each chunk; additionally, since there are at most 2 paths between the start and end of each chunk, a label in $[2]$ to identify the orientation of the cycle is sufficient to recover the entire path. □

Proposition 2.4.30. *For any un-encoded vertex that has at least two paths of H edges to some*

known vertices, the vertex is doubly-constrained, and the entire paths can be encoded at a cost of

$$2 \cdot (2 \cdot 2q|V(\alpha)|)^{\lfloor \frac{t}{\kappa} \rfloor}$$

Proof. The proof is identical to the above, except we no longer need to identify an extra endpoint for the end of the last chunk to use local 2-cycle-freeness. □

Remark 2.4.31. *Throughout our work, we will take* $2q \cdot |V(\alpha)| \approx n^{\epsilon}$ *and* $\kappa \approx 0.3 \log_d n$*, which would give us*

$$(2q \cdot |V(\alpha)|)^{1/\kappa} \leq c$$

for some constant $c > 0$ *independent of* d *provided* $\epsilon < O(\frac{1}{\log d})$ *where we take*

$$q = \Theta(\sqrt{|U_\alpha||V_\alpha|} \log^2 n)$$

In addition to the main claim that gives an additional factor of d for each dangling vertex, we observe 2-cycle freeness can be exploited to improve that bound.

Proposition 2.4.32 (Improved vertex factor for dangling R/H vertices)**.** *Let* α *be a given shape, and let* S *be a separator for some irreducible locally optimal labeling, we let* $\mathsf{dangling}(\alpha \setminus S)$ *be the dangling factor for* S*,*

$$\mathsf{dangling}(\alpha \setminus S) \leq \prod_{i \in \mathsf{branch}(\alpha \setminus S)} \min(2|V(\alpha)|q, d^{|\mathsf{branch}(i)|})$$

Observe that by possibly removing some edges incident to some dangling vertex if necessary, $\mathsf{dangling}(\alpha \setminus S)$ can be partitioned into a collection of trees rooted at some non-dangling vertex in α, and moreover, we call each vertex dangling in $\mathsf{dangling}(\alpha \setminus S)$ a branch point if it is either a leaf, or has degree bigger than 1. For each branching point $i \in \mathsf{branch}(\alpha \setminus S)$, let $\mathsf{branch}(i)$ denote the vertices on the path from last branching point

(or the nearest non-dangling vertex) to i. We are now ready to prove the proposition.

Proof. Notice for each branch, a label in $2|V(\alpha)|q$ is sufficient using Proposition 2.4.29; alternatively, a label in d is sufficient for identifying each "explored" dangling vertex. □

Balance in d A priori, we need to show for each high-mul (non-separator) leg in the walk, only a $\sqrt{d}$ cost is sufficient for encoding, so that we can assign a cost of at most $\sqrt{n}$ for each step as

$$\sqrt{\frac{n}{d}} \cdot \sqrt{d} = \sqrt{n}$$

Potential function for an individual walker Let's start with a potential function argument for individual walker.

Proposition 2.4.33 ($\sqrt{d}$-balance). *Each H edge requires at most a label in $O(\sqrt{d})$ for identifying its destination vertex.*

Proof. We first give a brief argument for the case when there is a single walker in the graph matrix (eg. line graphs), and extend this argument to the case when multiple walkers are involved.

Assigning a potential function for a single walker First, we observe that it suffices for us to consider an interval of the walk of length $0.1 \log_d n$ using $F/R/H$ edges, and let s be the starting point of the interval and t be the destination. Observe that t, as it is reached by H edge at the end of the current interval and since there is only a single walker, there are two cases: either it is already revealed in the walk and can be specified with a label in $2|V_\alpha|q$, or it is explored via an F/R segment in the subsequent interval.

In the first case, pick a pit-stop vertex B to be the destination t. Otherwise, pick the pit-stop B to be instead the last vertex on the $s - t$ path known to the walker by the beginning of the interval. That said, for any interval of length $0.1 \log_d n$, the non-backtracking path

from s to B is fixed using 2-cycle freeness. In the case when $B \neq t$, note that the walker opens up the path from B to t in the subsequent interval via an F/R segment before it uses these edges as an H-step. That said, it suffices for the walker to specify a direction towards or away from t on the main-path when a new edge is opened up after B, which is again $O(1)$ per-step.

The length $0.1 \log_d n$ is picked so that we can specify the destination vertex for free once we have $(2|V(\alpha)|q_\alpha)^{1/0.1 \log_d n} < O(1)$ for $|V(\alpha)| \leq n^\delta$. That said, we can decompose the upcoming interval of H edges as either walking on the main-path (non-backtracking segment from s to t), and off-track path attached the main-path (backtracking segment). Each H edge on the main-path does not require any encoding cost, and each H edge on the backtracking segment contributes d when it is heading outward of the main-path and $O(1)$ when heading inwards.

Assigning a potential function for a group of walkers In the trace-walk for general shapes, the walker is no longer unique as we may have arms splitting out from *branching* vertices, and similarly, arms merging as well. In other words, this corresponds to a walker being spawned and dying in the walk. We now show the potential function argument for individual walker applies well for this general setting.

For starters, recall that constraint graph can be linearized and decomposed as the following,

Decomposing (linearized) constraint graph into path-segments

1. We start with at most $U_\alpha = V_\alpha$ *directed* vertex-disjoint paths from U_1 to V_q;
2. Attach *directed* path-segments from a *branching* vertex to a vertex on a revealed path-segment, or a vertex in V_q.
3. Attach path-segments between vertices in the same block;

4. Attach *directed*(dangling) path-segment from an anchor vertex to a dangling vertex in the same block.

Notice that it suffices for the encoder to know such assignment of potential function. For each walker, if it remains alive at the end of its subsequent κ-interval, i.e., it does not intercept other revealed path-segments, then the potential function can be assigned as an individual walker.

On the one hand, for a walker that intercepts a revealed path-segment (i.e. it is a short-lived walker who does not remain in the walk for a κ-interval), it suffices for us partition the segment into κ-intervals again, and it suffices for us to restrict our attention to a short-lived walker, that it only takes a walk in an interval of length much less than κ, i.e., we do not have sufficient edge-decay to identify a label in $2|V(\alpha)|q_\alpha$. However, we note that the edge-set of non-backtracking H edges can be identified in $O(1)$ via our wedge-bundling argument in the forthcoming lemma 2.5.19, and this fixes the potential function for the short-lived walker before it *dies*, i..e., joins another walker's segment.

On the other hand, for a walker that remains alive at the end of its subsequent κ-interval, i.e., it does not intercept other revealed path-segments, then the potential function can be assigned as an individual walker by extending the argument for the single walker. It is no longer true that the path from s to t must be revealed by this particular walker, however, this happens when another walker explores the main-path from the revealed pit-stop to t and we observe that we may allow the pit-stop to "move" along the direction towards t throughout this interval, and thereby specifying the edges on the main-path from s to t.

To move the pit-stop vertex for a specific walker, we apply the wedge-bundling argument in again. Concretely, apply this argument where each walker looking for a new pit-stop engages in the process as its current pit-stop vertex (as opposed to its current vertex location), and for the walker that explores the segment on the path from the pit-stop to t for some other walker, it participates as the current vertex location. Now observe that

it suffices for us to identify the short paths between these walkers, which can be done in $O(1)$ per walker/vertex at each step.

□

2.5 Tight matrix norm bounds: block value and maximal-value labeling

In this section, we show that the block value is determined by the vertex separator. On the one hand, this identifies the dominant term of the trace being governed by the SMVS; on the other hand, the structure lemma also provides us with a primitive to identify the gap from our trace-walk that offers slack for encoding auxiliary data.

Towards this end, we start by constructing a vertex separator out of any locally-optimal labeling, and then relate the value of the labeling to the desired norm bound. However, before we proceed to construct a vertex separator, we need to understand the structure of a locally optimal labeling.

2.5.1 Set-up for block-value bounds

Before we dig into the core of our block-value bounds, we first make some (re-)definitions of our vertex-factors. In the previous section, we use R edge to represent the second-time an edge appears, and crucially, we show that, a vertex, if pushing out an R edge, the choice (unless dangling) is *fixed*. On the other hand, it would be convenient for us to consider R edge as the last appearance of an edge appearance due to potential assymmetries of the shape, and we note that for any edge, we can assume exchange its potential vertex factors for the second time and last time appearance, and thus assume each vertex, when pushing out an R edge that corresponds to the edge's last appearance, is fixed.

In the following section, we will now think of F / H / R corresponding to the first/middle/last appearance of an edge, which also motivates the following assignment scheme for edge-value depending on edge-labels.

Edge-value assignment scheme

For each random variable x,

1. F-step: for the first time the random variable appears,
   - Singleton: for random variable x that appears only once throughput the walk, we assign a factor $\sqrt{\frac{n}{d}} \cdot \frac{d}{n} \cdot \mathsf{Singletondecay}$ for $\mathsf{Singletondecay} \leq \exp(-d)$;
   - F: for random variable x that appears at least twice throughout the walk, this gets assigned a factor of 1;
2. H-step: we assign a factor of $\sqrt{\frac{n}{d}}$.
3. R-step: we assign a factor of 1.

Vertex-factor redistribution scheme Up front, each new vertex requires a label in $[n]$ when it first appears in the walk. However, we note that we can redistribute this factor throughout the walk and

1. Assign a factor of $\sqrt{n}$ when the vertex appears for the first time;
2. Assign another factor of $\sqrt{n}$ when the vertex appears for the last time.

It can be readily verified that each vertex still gets assigned a factor of $[n]$, modulo the cost of identifying destination of R edges and H edges pointing to this vertex.

Building the separator For each block, given an edge-labeling $\mathcal{L}$ (assumed to be traversing from $U \to V$), we build the separator $S(\mathcal{L})$ as the following (and ignore the dependence on $\mathcal{L}$ when it is clear),

1. Include any vertex in $U_\alpha \cap V_\alpha$ into S;

2. Include any vertex incident to both non-singleton F and R edges into S;

3. Include any vertex in U_α incident to some non-singleton F edge into S;

4. Include any vertex in V_α incident to some R edge into S;

5. Include any vertex incident to an H edge into S.

2.5.2 Bounding the block-value from the separator

Given a separator S and its corresponding edge-labeling $\mathcal{L}$, it suffices for us to bound

$$\mathsf{edgeval}(\mathcal{L}) \cdot \mathsf{vtxcost}(\mathcal{L})$$

for a given block.

Factors outside the separator Notice any vertex outside S is making its first or last appearance, and any vertex making both first and last appearance must be either isolated or it is a vertex outside $U_\alpha \cup V_\alpha$ that is also incident to singleton edges only, that said, restricting the vertex cost to the vertices outside S, we have

$$\mathsf{vtxcost}(\mathcal{L})[V(\alpha) \setminus S] \leq \sqrt{n}^{|V(\alpha)\setminus S|}\sqrt{n}^{|I(\alpha)\cup F_S(\mathcal{L})|}$$

where we let $I(\alpha)$ be the set of isolated vertices in α, and $F_S(\mathcal{L})$ the set of vertices outside $U_\alpha \cup V_\alpha$ that are incident to only singleton F edges in $\mathcal{L}$.

Lemma 2.5.1 (Singleton-edges come with decay unless floating)**.** *For a given edge-labeling $\mathcal{L}$ with non-floating singleton edges outside the separator S, there is an edge-labeling $\mathcal{L}'$ with*

separator S' such that $S \subseteq S'$, $\mathcal{L}'$ has no non-singleton floating edges, and

$$\mathsf{edgeval}(\mathcal{L}) \cdot \mathsf{vtxcost}(\mathcal{L}) \leq \mathsf{edgeval}(\mathcal{L}') \cdot \mathsf{vtxcost}(\mathcal{L}')$$

Moreover, S' is a separator for α, and restricting to the factors outside the new separator,

$$\begin{aligned}&\mathsf{edgeval}(\mathcal{L}')[E(\alpha) \setminus E(S')] \cdot \mathsf{vtxcost}(\mathcal{L})[V(\alpha) \setminus S] \leq \\ &\sqrt{n}^{|V(\alpha)\setminus S'|} \cdot \sqrt{n}^{I(\alpha)} \cdot \left(\sqrt{n}\exp(-d)\right)^{|\mathsf{float}(\alpha \setminus S')|} \cdot \mathsf{Pur}(\mathcal{L}')[V(\alpha) \setminus S'] \,.\end{aligned}$$

where with some abuse of notation, we let $|\mathsf{float}(\alpha \setminus S)|$ be the number floating components in α that are not reachable from S, and $\mathsf{Pur}(\mathcal{L}')[V(\alpha) \setminus S]$ denote the cost to settle Pur factor incurred by edges leading to vertices outside the separator S'

Proof. Before we dig into the analysis, it would be helpful for us to remind the reader of the following definitions.

Definition 2.5.2 (U-wedge and V-wedge)**.** *We call any connected component that is only reachable from U_α (or V_α) a U-wedge (or a V-wedge).*

We adopt a two-step strategy, let $V_c(\alpha \setminus S)$ be the set of non-floating vertices outside the separator, and let $V_f(\alpha \setminus S)$ be the set of floating vertices that are also not reachable from S.

1. If $\mathcal{L}$ has singleton-F edge in any U-wedge, we can locally improve the value by turning the F edge into R edge in $\mathcal{L}'$;

2. Furthermore, for the non-floating component that are neither U/V-wedges, we modify $\mathcal{L}'$ from $\mathcal{L}$ by turning any non-floating singleton F edge into a non-singleton edge;

3. We then bound the value by considering the factors on 1) U/V-wedges, 2) components reachable from some $U - V$ path, 3) floating components.

We now proceed to bound the first inequality such that

$$\mathsf{edgeval}(\mathcal{L}) \cdot \mathsf{vtxcost}(\mathcal{L}) \leq \mathsf{edgeval}(\mathcal{L}') \cdot \mathsf{vtxcost}(\mathcal{L}')$$

We observe the following,

1. For vertices not incident to any singleton edges, it suffices for us to adopt the same encoding for it in $\mathcal{L}'$ as in $\mathcal{L}$, and thus it contributes the same vertex factor to both sides;

2. $S \subseteq S'$ since any vertex in S remains in S', while S' may potentially include more vertices as there may be $U_\alpha \setminus S$ vertices which get pushed into S' since its incidental F edge gets changed into a non-singleton edge;

3. For any edges inside $E(S)$, it remains as a separator edge in $E(S')$ and thus contributes the same edge-value;

4. Any non-singleton edge in $\mathcal{L}$ contributes the same value in $\mathcal{L}'$;

5. Any singleton edge in $\mathcal{L}$ contributes a value of $\sqrt{\frac{d}{n}} \cdot \mathsf{Singletondecay}$ to $\mathcal{L}$, while a value of 1 to $\mathcal{L}'$;

6. Any Pur factor's contribution to vtxcost remain the same on both sides.

The lemma follows by the following two claims. □

Claim 2.5.3 (Non-floating vertices are offset by singleton decay)**.**

$$\sqrt{n}^{|V_c(\alpha)\setminus S|} \cdot \left(\sqrt{\frac{n}{d}} \cdot \frac{d}{n} \cdot \mathsf{Singletondecay}\right)^{|\#\textit{of non-floating singleton edges}(\mathcal{L})|} \cdot \sqrt{n}^{|F_S(\mathcal{L}) \cap V_c(\alpha \setminus S)|} \ll \sqrt{n}^{|V_c(\alpha)\setminus S'|}$$

Proof. This follows by considering a traversal process from U_α and V_α, as any non-floating vertex is reachable from $U \cup V$. Any such vertex is reachable using a singleton F edge, and assign its edge-value to the particular singleton vertex it leads to gives us

$$\left(\sqrt{\frac{n}{d}} \cdot \frac{d}{n} \cdot \mathsf{Singletondecay}\right) \cdot \sqrt{n} = \sqrt{d} \cdot \exp(-d) \ll 1$$

Moreover, for each vertex in $S' \setminus S$, note that up front it gives a factor of $\sqrt{n}$ to the LHS and 1 on the RHS. However, notice that any such vertex has a path to V_α (otherwise it is not added to the separator S'), there is at least one edge along the path that goes from some singleton vertex that remains uncharged (as we charge each edge to the vertex it leads to, and vertex in V_α by definition does not contribute a $\sqrt{n}$ blow-up as it is not making both first and last appearance). Combining the uncharged edge's value with the vertex factor gives us a factor of

$$\sqrt{\frac{n}{d}} \cdot \frac{d}{n} \cdot \mathsf{Singletondecay} \ll 1$$

□

As a result of this claim, we observe that for a locally optimal edge-labeling, each path between U, V does not use singleton-F edges, and thus it necessarily through some vertex in the separator we construct from the above out the edge-labeling. Therefore, we have recovered the connection between vertex separators and norm bounds of in the dense regime, and moreover, this allows us to bound the block value of each separator. Moreover, it is sufficient for us to switch to the vertex-separator perspective from edge-labeling as any locally optimal edge-labeling must now correspond to some "honest" separator of the shape.

Corollary 2.5.4. *Any non-isolated vertex outside the separator contributes a vertex factor of $\sqrt{n}$ (modulo their corresponding floating and dangling factors via* Pur*). They contribute an extra factor*

of $\sqrt{n}$ if isolated.

Claim 2.5.5 (Tree-like floating component gives singleton blow-up)**.**

$$\left(\sqrt{\frac{n}{d}}\cdot\frac{d}{n}\cdot\mathsf{Singletondecay}\right)^{|\#\textit{of floating singleton edges}(\mathcal{L})|}\cdot\sqrt{n}^{|V_f(\alpha\setminus S)|}\leq(\sqrt{n}\cdot\mathsf{Singletondecay})^{|\mathsf{float}(\alpha\setminus S)|}$$

Proof. The previous argument for handling singleton edges crucially replies upon being able to assign each vertex to an edge that comes with singleton decay. However, this is no longer true for floating component not reachable from the separator, especially tree-like floating component: for example, consider an edge floating around in α. Should the edge be a singleton edge in the walk outside the separator, it contributes a factor

$$n^2\cdot\sqrt{\frac{d}{n}}\cdot\mathsf{Singletondecay}\approx n^{1.5}\exp(-d)\gg\sqrt{n^2}\cdot(2|V(\alpha)|q_\alpha)$$

where we recall $\sqrt{n^2}\cdot(2|V(\alpha)|q_\alpha)$ is the cost we would expect if there is no conditioning. That being said, we may have at most one $\sqrt{n}\cdot\mathsf{Singletondecay}$ blow-up for each component only for the first vertex in the floating component outside the separator, as the rest of vertices in the floating component can be charged by the singleton edge leading to it by the previous argument, and this proves the claim above. □

Remark 2.5.6. *A natural question is whether the $\sqrt{n}$ blow-up from conditioning is tight. We remark that it is possible as it comes down to bounding $\mathbb{E}[\chi_e\cdot 1_{\mathcal{F}}]$ where we recall $\mathcal{F}$ is the conditioning on bounded-degree, and we can ignore the conditioning on 2-cycle for this part.*

$$\begin{aligned}\mathbb{E}[\chi_e\cdot 1_{\mathcal{F}}]&=\mathbf{Pr}\left[1_{\mathcal{F}}\cap\left(\chi_e=\sqrt{\frac{1-p}{p}}\right)\right]\cdot\sqrt{\frac{1-p}{p}}+\mathbf{Pr}\left[1_{\mathcal{F}}\cap\left(\chi_e=-\sqrt{\frac{p}{1-p}}\right)\right]\cdot(-\sqrt{\frac{p}{1-p}})\\&=\mathbf{Pr}\left[1_{\mathcal{F}}|\chi_e=\sqrt{\frac{1-p}{p}}\right]\sqrt{\frac{1-p}{p}}\cdot p-\mathbf{Pr}\left[1_{\mathcal{F}}|\chi_e=-\sqrt{\frac{p}{1-p}}\right]\cdot\sqrt{\frac{p}{1-p}}\cdot(1-p)\end{aligned}$$

$$= (\sqrt{(1-p)p}) \left(\mathbf{Pr}\left[1_{\mathcal{F}}|\chi_e = \sqrt{\frac{1-p}{p}}\right] - \mathbf{Pr}\left[1_{\mathcal{F}}|\chi_e = -\sqrt{\frac{p}{1-p}}\right] \right)$$

To bound the difference in the probability, we note that this is equivalent to resampling all the edges but e, what's the probability that at least one of the two endpoints of e hit degree $10d-1$ (suppose we truncate at degree $10d$), and an $\exp(-d)$ bound here is the best we can hope for, which would give a bound of $\mathbb{E}[\chi_e \cdot 1_{\mathcal{F}}] \approx \sqrt{\frac{d}{n}} \cdot \exp(-d)$. Notice we need an extra $\frac{1}{\sqrt{n}}$ decay to offset the blow-up for a singleton F component.

Bounding the floating factor

Claim 2.5.7. *It suffices for us to assign a factor of $\sqrt{n}\exp(-d)$ for each floating component, i.e., $\mathsf{float}(\alpha) \leq \left(\sqrt{n}\exp(-d)\right)$ or a factor of $(2|V(\alpha)|q_\alpha)^{O(1)}$, eithe of which can be bounded by $\left(\sqrt{n}\exp(-d)\right)$.*

Proof. We case on whether the floating component contains a cycle. Observe that the edges in the same floating component have to receive the same $F/R/H$ label under the edge-labeling, otherwise it can be locally improved.

1. **Singleton floating component** We first consider F-component, where some extra care is warranted as singleton edges may now be troublesome. The previous argument for handling singleton edges crucially replies upon being able to assign each vertex to an edge that comes with singleton decay. However, this is no longer true for floating component, especially tree-like floating component: for example, consider an edge floating around in α: should the edge be a singleton edge in the walk, it can contribute

$$n^2 \cdot \sqrt{\frac{d}{n}} \cdot \mathsf{Singletondecay} \approx n^{1.5}\exp(-d) \gg \sqrt{n^2} \cdot (2|V(\alpha)|q_\alpha)$$

where we recall $\sqrt{n^2} \cdot (2|V(\alpha)|q_\alpha)$ is the cost we would expect if there is no conditioning. That being said, we may have one $\sqrt{n}$ blow-up, giving a bound of

$$\sqrt{n}^{|V(C_i)|} \cdot \underbrace{\sqrt{n} \cdot \mathsf{Singletondecay}^{V(C_i)-1}}_{\mathsf{float}(C_i)} \cdot 1_{C_i \cap S = \emptyset}$$

where we have the indicator $1_{C_i \cap S = \emptyset}$ as this can only happen when C_i is outside the separator.

2. **Non-singleton floating component** The argument is identical to the non-floating case except for the first vertex in the component (when being traversed the second-time or later), giving us a bound of

$$\sqrt{n}^{V|(C_i)|} \cdot \underbrace{\mathsf{dangling}(C_i) \cdot 2|V(\alpha)|q_\alpha}_{\mathsf{float}(C_i)}$$

Remark 2.5.8. *We bound* $\mathsf{dangling}(C_i) \cdot 2|V(\alpha)|q_\alpha \leq |2|V(\alpha)q_\alpha|^{O(1)}$ *loosely in the final bound as we consider each landing at the floating component a surprise visit, which comes with* $O(1)$ *factors of* $2|V(\alpha)|q_\alpha$.

Combing the above, for each floating component, it suffices for us to take

$$\begin{aligned}\mathsf{float}(C_i) &\leq \max(\sqrt{n} \cdot (\mathsf{Singletondecay})^{V(C_i)-1} \cdot 1_{C_i \cap S = \emptyset}, (2|V(\alpha)|q_\alpha)^{O(1)}) \\ &\leq \max(\sqrt{n} \cdot \exp(-d) \cdot 1_{C_i \cap S = \emptyset}, (2|V(\alpha)|q_\alpha)^{O(1)})\end{aligned}$$

□

Bounding the block value for any separator of the shape From the above, we have established any labeling corresponds to a separator of the shape with higher block value

if the labeling does not produce a separator of the shape itself due to singleton edges. That said, to bound the value of the maximal value-labeling, it suffices for us to consider separators of the shape.

Let's now try to bound the block value $\mathsf{edgeval}(\mathcal{L}) \cdot \mathsf{vtxcost}(\mathcal{L})$ for any locally optimal $\mathcal{L}$, let S be the corresponding separator, let S_L be the vertices in S that are reachable from U_α without passing through any vertex in S,

1. Each vertex outside S contributes a factor of $\sqrt{n}$;

2. Each vertex in S_L is reachable via an R edge, hence there is no vertex factor;

3. Each vertex in S that is not doubly constrained in the encoding process is arrived along an H edge, and hence requires at most a $\sqrt{d}$ vertex factor if non-dangling, and a factor of d if dangling;

4. Each edge receiving an H label is in $E(S)$ and contributes a value of at most $\sqrt{\frac{n}{d}}$;

5. Each dangling branch contributes a factor of $\mathsf{dangling}$, that is at most a factor of $2|V(\alpha)|q_\alpha$ per dangling branch via either Pur or H-visit;

6. Each floating component, depending on whether it's locally tree-like, either contributes a factor of at most $\sqrt{n}\exp(-d)$, or a factor of $(2|V(\alpha)|q_\alpha)^{O(1)}$ via Pur but not both.

For intuition, we recommend the reader to ignore factors from excess F edges, and dangling vertices. However as stated, the above bound is not easy to apply as it is not immediately clear what vertex factors are needed for encoding vertices inside S; fortunately, this can be simplified once we restrict our attention to a class of separators, *irreducible* separators, without any loss in the SMVS value.

We now unpack the factors from the above in a more illustrative manner, and show that it suffices for us to assume each vertex in S is doubly constrained by S_L, and moreover, there is no dangling vertex inside S.

Not too many surprise visits leading to vertices outside S We bound the contribution to the block-value via Pur factors, and notice unless floating components and dangling components are involved, each such factor corresponds to a surprise visit (i.e. an excess edge in the case it is inevitably surprised due to shapes containing cycles). We remind the reader that $\sqrt{d}^{|\mathsf{excess}(F)|}$ is an artifact of our encoding since we encode a potential direction for future unforced return that may be incurred due to the excess edge, however, we point out that this is not too terrible a cost to pay as in any optimal labeling, there cannot be too many $\mathsf{excess}(F)$ edges, and we can handle the blow-up via edge-decay in the optimal labeling.

Proposition 2.5.9. *Let $\mathcal{L}$ be a locally optimal labeling, and $\mathsf{excess}(F)$ be the number of F edges that lead to visited vertices when we consider a BFS from S,*

$$\mathsf{excess}(F) < \lceil \frac{|F \setminus \mathsf{excess}(F)|}{\log_d n} \rceil$$

Proof. This follows from the optimality of SMVS, consider the new separator formed by including all vertices reached by F edges, which gives us a change of $\sqrt{n}^{-|F \setminus \mathsf{excess}(F)|} \cdot \sqrt{\frac{n}{d}}^{|F|} < 1$. □

Surprise visits arriving at S can be locally improved Fixing the traversal direction from U_α to V_α, we case on whether the edge comes from outside the separator. Suppose so, an edge from $V(\alpha) \setminus S$ to a vertex $v \in S_L$ may be a surprise visit if it is an F edge, that said, we observe that we can locally improve the value by traversing the path from $v \in S_L$ to $a \in U_\alpha$ (recall that v is reachable from U_α without passing through v, hence such a path is

well-defined), and turn each F edge into an R along the path. Noice we have the following change of value,

1. Let $\mathcal{L}$ be the edge-labeling before the flip, and $\mathcal{L}'$ after the flip;

2. By our previous clean-up, we can assume there is no singleton edge involved;

3. Any vertex not in U_α along the path outside the separator contributes a factor of $\sqrt{n}$ in both labelings;

4. The edge leading to v is a surprise visit, and contributes a factor of $(2|V(\alpha)|q_\alpha)^{c_S}$ via Pur factor in $\mathcal{L}$ but not in $\mathcal{L}'$;

5. The vertex $a \in U_\alpha$ (as well as any other vertex in U_α reachable from v) contributes a factor of $\sqrt{n}$ to $\mathcal{L}'$ but not $\mathcal{L}$

Therefore, we pick up a change of at least

$$\frac{\mathsf{edgeval}(\mathcal{L}') \cdot \mathsf{vtxcost}(\mathcal{L}')}{\mathsf{edgeval}(\mathcal{L}) \cdot \mathsf{vtxcost}(\mathcal{L})} \geq \frac{\sqrt{n}}{(2|V(\alpha)|q_\alpha)^{c_S}} \gg 1\,.$$

On the other hand, suppose the edge is contained in $E(S)$, observe that flipping the edge from F to H does not affect the vertex factor of its origin endpoints, while its destination may originally contribute a cost of $2|V(\alpha)|q_\alpha$, and it may now be doubly constrained and does not contribute any vertex cost in $\mathcal{L}'$. That said, the H edge is now contributing a factor of $\sqrt{\frac{n}{d}}$ through its edge-value, and hence, we pick up a change of

$$\frac{\mathsf{edgeval}(\mathcal{L}') \cdot \mathsf{vtxcost}(\mathcal{L}')}{\mathsf{edgeval}(\mathcal{L}) \cdot \mathsf{vtxcost}(\mathcal{L})} \geq \frac{\sqrt{\frac{n}{d}}}{(2|V(\alpha)|q_\alpha)^{c_S}} \gg 1\,.$$

Pruning superfluous H edges

Definition 2.5.10 (Irreducible SMVS)**.** *We call a separator S an irreducible SMVS for a shape α if for any vertex $v \in S$ of degree 1 in $E(S)$,*

1. *There are two vertex-disjoint paths P_U, P_V from v to U_α and v to V_α*
2. *Both paths do not pass through any other vertex in S*

Claim 2.5.11. *For each locally optimal $\mathcal{L}$ labeling that produces separator vertices in $S(\mathcal{L})$ not doubly constrained by $S_L(\mathcal{L})$, there is a labeling $\mathcal{L}'$ with at least the same SMVS value while each vertex in $S(\mathcal{L}')$ is doubly constrained by $S_L(\mathcal{L}')$. Moreover, there is no dangling vertex in $S(\mathcal{L}')$.*

Proof. We start by considering non-dangling vertices in S. We first notice for any vertex in S_L of degree 1 in S, unless it is incident to an F edge, we can pop it out of the separator and obtain the same value by turning the incident H edge into an R edge: the H edge contributes at most a $\sqrt{d}$ vertex factor to indicate the destination of the edge and it contributes a factor of $\sqrt{\frac{n}{d}}$ through the edge-value, while we obtain the same value by picking the $\sqrt{n}$ value from the vertex being outside the separator.

Let $\mathcal{L}'$ be the labeling obtained by applying the above procedure on $\mathcal{L}$, and let $S = S(\mathcal{L}')$ be its corresponding separator. Consider a BFS from S_L to traverse vertices in $S \setminus S_L$, and note that each vertex in the separator not doubly constrained by S_L has only one simple path to S_L, and it is not on any path from S_L to a cycle. In other words, given S_L, each non-doubly constrained vertex forms a tree rooted at S_L. For each branch, starting from the leaf vertex, notice each vertex (if non-dangling) contributes to the block value via the edge that leads to that vertex from S_L a factor of $\sqrt{\frac{n}{d}}$ and a vertex factor $\sqrt{d}$ for indicating the destination of the H edge, giving a total of $\sqrt{\frac{n}{d}} \cdot \sqrt{d} = \sqrt{n}$. However, the same value can be attained by flipping the H edge to an F edge and pop this vertex out of the separator.

For each dangling vertex, the pop-out procedure is rather identical except we now need to take into account the dangling factor. Suppose it is indicated using a label in d, it contributes a value of $\sqrt{\frac{n}{d}} \cdot d = \sqrt{nd}$ while we can again pop the vertex out of the

separator and obtain a value of $\sqrt{nd}$ with the extra $\sqrt{d}$ coming from the dangling factor dangling; analogously, we can generalzie this argument for long dangling branch inside the separator, we can again use the idea in Proposition 2.4.29, and obtain a $\sqrt{2|V(\alpha)|q}$ bound as we would have

$$\begin{aligned}\sqrt{\frac{n}{d}}^{\mathsf{branch}(i)} \min(2|V(\alpha)|q, d^{\mathsf{branch}(i)}) &\leq \sqrt{n}^{\mathsf{branch}(i)} \min(\frac{2|V(\alpha)q}{\sqrt{d}^{\mathsf{branch}(i)}}, \sqrt{d}^{\mathsf{branch}(i)}) \\ &\leq \sqrt{n}^{\mathsf{branch}(i)} \min(\sqrt{2|V(\alpha)|q}, \sqrt{d}^{\mathsf{branch}(i)})\end{aligned}$$

Note that this is the same value we would pick up from vertex factors if we pop out the entire dangling branch, and this completes our proof to the claim. □

Wrapping up block value bound for a single shape We are now ready to wrap up our bound for the block value. Recall that under our setup via edge-labeling, for a particular block of shape α, our bound proceeds as

$$B(\alpha) = \sum_{\mathcal{L}} B(\mathcal{L}) = \sum_{S:\text{separator}} B(\mathcal{L}_S)$$

and we first determine the dependence of a fixed labeling.

Proposition 2.5.12. *For any irreducible locally optimal labeling $\mathcal{L}$, let S be its corresponding separator,*

$$\mathsf{vtxcost}(\mathcal{L}) \cdot \mathsf{edgeval}(\mathcal{L}) \leq c_{\mathsf{norm}}^{|V(\alpha)\setminus S|} \cdot \sqrt{n}^{|V(\alpha)\setminus S|} \cdot \left(\sqrt{\frac{n}{d}}\right)^{|E(S)|} \cdot \mathsf{dangling}(\alpha \setminus S) \cdot \mathsf{float}(\alpha) \cdot \sqrt{n}^{I(\alpha)}$$

with

$$\mathsf{dangling}(\alpha \setminus S) \leq \prod_{b_i \in \mathsf{branch}(\alpha\setminus S)} \min\left(\sqrt{d}^{|b_i|}, 2|V(\alpha)|q_\alpha\right)$$

and

$$\mathsf{float}(\alpha) = \prod_{C_i \in F(\alpha)} \left(2|V(\alpha)|q_\alpha, \sqrt{n} \cdot \mathsf{Singletondecay}^{|E(C_i)|} \cdot 1_{V(C_i)\cap S=\emptyset}\right)$$

for $F(\alpha)$ the collection of floating components in α, and for $I(\alpha)$ the number of isolated vertices in α.

Proof. This follows by recalling the factors accounted from above,

1. Each vertex outside S contributes a factor of $\sqrt{n}$ (via section 2.5.2);
2. Each vertex in S is doubly constrained by the boundary of S (via section 2.5.2), unless floating which is captured by the factor in $|\mathsf{float}(\alpha)|$;
3. Each edge receiving an H label is in $E(S)$ and contributes a value of at most $\sqrt{\frac{n}{d}}$;
4. For dangling vertices, we pick up an additional dangling factor $\mathsf{dangling}(\alpha \setminus S)$; each floating component comes an extra factor captured by $\mathsf{float}(\alpha)$ via Proposition 2.4.32 and Claim 2.5.5;
5. Each excess F edge that leads to visited vertices incur an extra cost of $|2V(\alpha)q_\alpha|^{O(1)}$ via $O(1)$ of the Pur factor it may incur, however, this is subsumed by vertex decay outside the separator (via section 2.5.2).

□

Finally, we obtain the bounds by summing over the choice of separators, and notice that each vertex in $U_\alpha \cap V_\alpha$ is in the mandatory separator,

Corollary 2.5.13. *For any shape α, we can bound its block-value function by*

$$B(\alpha) = \sum_{S:\mathit{separator}} \mathsf{vtxcost}(\mathcal{L}_S) \cdot \mathsf{edgeval}(\mathcal{L}_S)$$

$$\leq \sum_{S:\text{separator}} c_{\mathsf{norm}}^{|V(\alpha)\setminus S|} \cdot \sqrt{n}^{|V(\alpha)\setminus S|} \cdot \left(\sqrt{\frac{n}{d}}\right)^{|E(S)|} \cdot \mathsf{dangling}(\alpha \setminus S) \cdot \mathsf{float}(\alpha) \cdot \sqrt{n}^{|I(\alpha)|}$$

$$\leq (c'_{\mathsf{norm}})^{|V(\alpha)\setminus U_\alpha \cap V_\alpha|} \sqrt{n}^{|V(\alpha)\setminus S|} \cdot \left(\sqrt{\frac{n}{d}}\right)^{|E(S)|} \cdot \mathsf{dangling}(\alpha \setminus S) \cdot \mathsf{float}(\alpha) \cdot \sqrt{n}^{|I(\alpha)|}$$

where we define

$$\mathsf{dangling}(\alpha \setminus S) \leq \prod_{b_i \in \mathsf{branch}(\alpha \setminus S)} \min\left(\sqrt{d}^{|b_i|}, 2|V(\alpha)|q_\alpha\right)$$

and

$$\mathsf{float}(\alpha) = \prod_{C_i \in F(\alpha)} \left(2|V(\alpha)|q_\alpha, \sqrt{n} \cdot \mathsf{Singletondecay}^{|E(C_i)|} \cdot 1_{V(C_i)\cap S=\emptyset}\right)$$

where $F(\alpha)$ is the collection of floating components in α, and $I(\alpha)$ the set of isolated vertices.

2.5.3 Bounding the drawing costs for unknown shapes

Definition 2.5.14 (Wedge bundling)**.** *Given a set of vertices $S \subseteq [n]$, and a wedge W, we call it bundled if U_W is labeled as a subset of S. Moreover, edges that are part of the wedge also receive labels in $\binom{n}{2}$ as a subset.*

To bound the encoding cost for bundling wedges, we give a query algorithm for bundling vertices in U_α (and respectively, in V_α).

Set-up for bundling Let's start with a set-up that allows us to turn the permutation question into a query problem.

Definition 2.5.15 (Wedge and its legs)**.** *Wedge W is a shape with only one labeled set $U_{\mathcal{W}}$(as opposed to U_α and V_α), and each vertex in $i \in U_W$ has a path to some $j \neq i \in U_W$ in $\mathcal{W}$, and each vertex in $V(\mathcal{W}) \setminus U_{\mathcal{W}}$ is connected to the labeled set U_W.*

Definition 2.5.16. *We additionally call a wedge an* irreducible *wedge if every leg is of degree* 1*, i.e., for any* $i \in U_W$, $\deg_W(i) = 1$.

Remark 2.5.17. *Given a wedge* W *that is not irreducible, it can be reduced to some irreducible wedge by removing non-degree-1 vertex from* U_W.

We are now ready to introduce the central question for bounding the combinatorial factor of wedge-bundling:

Question 2.5.18. *Given* G *a random graph sample, particularly it is guaranteed to be* 2*-cycle free at* $\kappa \approx 0.4 \log_d n$ *neighborhood of any vertex, and given a list of special vertices* S *that we want to match, and a list of vertex-disjoint wedges* $\{\mathcal{W}_i\}$ *that are subgraphs of* G*, can we ask each edge in the wedge* $O(1)$ *Yes/No questions and identify each wedge* $\{\mathcal{W}_i\}_{i\in t}$ *as an edge set?*

A query algorithm for wedge bundling

We introduce our bundling procedure via a query algorithm.

Lemma 2.5.19. *Given a sequence of wedges* $\mathcal{W} = \{\mathcal{W}_i\}$*, and their legs given as a set of vertices* $\bigcup_i U_{\mathcal{W}_i}$*, the wedges can be bundled (identified as an edge set) at a cost*

$$\mathsf{cost}\mathcal{B} \leq (c_{matched})^{\sum_i |E(\mathcal{W}_i)|}$$

for some absolute constant $c_{matched}$ *independent of* d.

For starters, we first consider a *merging* process for a irreducible wedge as the following,

1. For each wedge, pick an arbitrary vertex t as a root, and traverse the wedge from the root; notice by removing cycle edges (edges leading to visited vertices in the traversal) from the wedge, this is a tree with leaves at the labeled set $U_{\mathcal{W}}$, and the traversal from the root allows us to assign a direction for each tree edge;

2. For each non-leaf vertex traversed that has degree more than 2, we call this a *branch* vertex;

3. For each leaf node v, consider its closest branch node on its path to t, and let the branch node be B; let $\{b_i\}_B \subseteq U_{\mathcal{W}}$ be the labeled leaf nodes of B, let b_1, b_2 be the two closest leaf nodes to B among its children $\{b_i\}_B$ (with ties broken arbitrarily).

4. For each branch node $b \in V(\mathcal{W})$, let c_b be its number of downward (children) branches, we associate it with

   - $\mathsf{Up}[b] \in \mathbb{N}$: the upward-bundle query distance of b;
   - $\mathsf{Down}[b] \in \mathbb{N}^{c_b-2}$: a list of downward-bundle query distance of b

5. For each leaf node $u \in U_{\mathcal{W}}$, we associate it with $\mathsf{Up}[u]$ its upward-bundle query distance.

Bundling process using upward/downward-bundle distance Before we proceed to bound the auxiliary data required and the query number, let's first see how these auxiliary data (Up and Down) can be helpful for bundling vertices from the given labeled set into wedges. In the beginning of the bundling process, no branch vertex is revealed to the decoder, while each leg vertex in $U_{\mathcal{W}}$ comes with an upward-bundle distance, which we claim is sufficient to for the decoder to identify the branch nodes throughout the process.

For the sake of illustration, we restrict to a single wedge case while its generalization to multiple wedges is straightforward. Throughout the process, we maintain W as a set of to-be-bundled vertices and start with $W = U_{\mathcal{W}}$ (for the general case, we have $W = \cup_{\mathcal{W}} U_{\mathcal{W}}$). With the sequence of upward bundle distance given to the decoder, the decoder may attempt to bundle revealed vertices from W by querying the decoder each pair of possible

bundling (i.e. two vertices in W that have the same upward [3]-bundle query distance, and there is a path in the revealed walk of the same length connecting these two vertices). Any bundled path may give rise to at most one branch node, and it can be specified by traversing the matched path, and we include the branch node into the active set W while removing vertices that are bundled using upward match query distance. Once a branch node is identified along the process, the decoder may encode the auxiliary data for the branch node: the data may be either the next unbundled downward-match query distance (if any), or the upward-match query distance of the current branch node.

Observation 2.5.20. *The sequence of upwards matched distance for each leg vertex in $U_{\mathcal{W}} \setminus V_\alpha$ can be specified to the decoder when each its incident edge is specified from the last block (which exits guaranteed by permissible shape).*

Bounding auxiliary data cost and bundle-attempts

We now proceed to bound the cost of the auxiliary data and the number of bundle-attempts by the decoder. Let's start with the auxiliary data used for specifying upwards/downwards matched query distance for each leg/branch vertex. For intuition, recall that we pick up a constant edge decay for each edge in $E(\alpha) \setminus E(U_\alpha \cap V_\alpha)$, and our goal is to show these auxiliary data can be offset by the edge decays from each block.

Claim 2.5.21. *For any sequence of wedges $\{\mathcal{W}_i\}$,*

$$\sum_{\mathcal{W}_i} \sum_{i \in V(\mathcal{W}_i)} \mathsf{Up}[i] + \mathsf{Down}[i] \leq 2 \sum_{\mathcal{W}_i} |E(\mathcal{W}_i)|$$

Proof. It suffices for us to prove this for a fixed wedge, and notice Up is only defined for branch nodes and leaf nodes, while Down only for branch nodes. First consider

[3] or upward-downward if matching a branch node with a leaf node

$E'(\mathcal{W}) \subseteq E(\mathcal{W})$ that is obtained by removing cycle edges, and notice it suffices for us to consider edges in $E'(\mathcal{W})$ as edges in $E(\mathcal{W}) \setminus E'(\mathcal{W})$ only contribute to the RHS.

The claim then follows by noticing $E'(\mathcal{W})$ is a tree with vertices being branch nodes and leaf nodes, and each edge contributes at most 2 to the LHS. □

Claim 2.5.22. *For a sequence of wedges $\{\mathcal{W}_i\}$, the number of match queries is at most $O(\sum_i E(\mathcal{W}_i))$.*

It suffices for us to bound the number of "No" queries throughout the bundling process. Towards this end, notice we can bound the number of "No" queries at each search level D (either "Upwards" or "Downwards"-bundle distance) and charge the "No" queries to the number of edges matched at the particular search level.

Definition 2.5.23 (Wedge-search graph for level D)**.** *Given a number D, we define its corresponding search graph as,*

1. *Each vertex in the graph is a labeled (in $[n]$) vertex (either leaf or branch node) in the execution of the bundling process that is to-be-matched with some other vertex at distance D in the graph;*

2. *Each edge corresponds to a query of the decoder asking whether the edge is part of a wedge connecting the labeled vertices.*

To bound the number of queries at each level D, we notice any subgraph (even of size polynomial in n) of a random graph is sparse despite potentially denser than $G_{n,d/n}$.

Proposition 2.5.24. *(Sparsity of small subgraph in $G_{n,p}$) For $G \sim G_{n,p}$ for $p < \frac{1}{2}$, with probability at least $1 - O(\frac{1}{n^6})$, every subgraph S of G such that $|V(S)| \leq (\frac{1}{p})^{1/2}$,*

$$|E(S)| \leq 3|V(S)| \log_{1/p}(n)$$

Proof. This follows by first moment. Let $e^*(v) = 3v \log_{1/p}(n)$,

$$\begin{aligned}\mathbf{Pr}[\exists \text{a dense subgraph}] &\leq \sum_{v=2}^{(1/p)^{1/2}} \sum_{e=e^*(v)}^{\binom{v}{2}} \binom{n}{v} \cdot (2v)^e p^e \\ &\leq \sum_{v>2} \sum_{e=e^*(v)}^{\binom{v}{2}} 2^{v\log n - v\log v + e\log p} \\ &\leq \frac{1}{n^6}\end{aligned}$$

□

Corollary 2.5.25. *W.h.p., for $D < 0.2\log_d n$ and $p = \frac{d^D}{n}$, and $|V(S)| \leq \sqrt{\frac{n}{d^D}} \leq n^{0.4}$, we have $|E(S)| \leq 5|V(S)|$.*

Proof to claim 2.5.22. For starters, observe for $D \geq 0.2\log_d n$, edge decay along the wedge is sufficient for us to assign two label in T_D to hardwire the match, and it suffices for us to focus on short wedges. For each bundle query distance D, let T_D be the number of vertices querying at distance D at any stage in the algorithm, notice the auxiliary graph is a random graph on T_D vertices with each edge present with probability at most $(\frac{d}{n})^D \cdot \binom{n}{D-1} \leq \frac{d^D}{n}$ as this is a path of length D with both endpoints fixed in a $G_{n,d/n}$. By the edge bound from above, we have at most $5T_D$ queries (with each query requires a yes/no answer) , while we have edge decay from $T_D \cdot D/2$ many edges from the edges of the wedges matched. To see this, notice each vertex at this level matches some other vertex via a path of length D, and charging the edges on the path to both endpoint gives the desired bound,

$$2^{5\cdot T_D} \ll c^{O(T_D \cdot D)}$$

for some $c > 1$

Summing over all levels of D give us the desired bound. □

2.5.4 Matching bundled wedges

In this subsection, we show the cost for assigning each vertex a label in $\mathcal{W}$, and matching wedge's legs from the bundled edge set can be handled by vertex decay. As a thought process, let us momentarily forget the shape α being given to us before the current block. Instead, we are going to encode (in the current block) the shape α whose norm we are computing.

Let cost$\mathcal{W}$ be the cost of matching shape α from the sequence of bundled wedges,

Lemma 2.5.26. *For some absolute constant $c_{matched} > 0$, we have*

$$\text{cost}\mathcal{W} \leq 2^{|V(\alpha)\setminus S|} \cdot (c_{matched})^{|E(S)|}$$

Bounding the drawing cost As a reminder, throughout this section, we restrict our attention to maximal value labelings, as if not, we can hard-code its corresponding permutation-jump factor via a gap in the adjacent block-value. Given that we are looking at an optimal labeling, we first show that the first idea of identifying via vertices is sufficient when the wedges are far apart as the label for identifying a vertex for each wedge can be handled via edge-decay.

Wedges are either disconnected or far-apart Let $\mathcal{L}$ be an edge-labeling for a shape α, let $\alpha_H(\mathcal{L})$ be the subshape obtained by removing F edges (and isolated vertices connected by F edges) in $\mathcal{L}$, and recall that $\mathcal{W}$ is a list of wedges (i.e. connected components connected to U_α) in α_H.

We first observe that wedges not connected in α_H may still be connected in α, however, we can observe that they must be "far-apart" with each other in α to appear disconnected in α_H.

Proposition 2.5.27. *For any wedge $W_i, W_j \in \mathcal{W}$ for $\mathcal{W}$ the list of wedges obtained from $\alpha_H(\mathcal{L})$ with some maximal-value labeling $\mathcal{L}$, we have*

$$\mathrm{dist}_\alpha(W_i, W_j) > \Omega(\log_d n)$$

Proof. Suppose not and W_i, W_j are connected in α, by definition of α_H, there is at least an all-F path connecting W_i and W_j. Considering the labeling obtained by flipping all F path to all H gives the desired logarithmic tradeoff. □

Proof to Lemma 2.5.26. We can now prove our main lemma. Using the standard BFS encoding for α, we can start from the sequence of bundled wedges, consider the BFS traversal on the wedges, and add edges if needed: in particular,

1. An edge leading to a new vertex can be succinctly encoded by a label in $[2]$, and another label in $[2]$ to specify the vertex-status of the new vertex (whether in V_α);

2. An edge leading to a visited vertex is more delicate as a label in $V(\alpha)|$ may be needed, however, via the above proposition, any such label can be charged to the long path;

3. The bundling data requires c_{matched} factor from each edge that participates in the process, and this is only needed for H edges, hence a factor of $(c_{\text{matched}})^{|E(S)|}$.

Therefore, this bounds the cost of drawing out α by $2^{|V(\alpha)\setminus S|} \cdot (c_{\text{matched}})^{|E(S)|}$. □

Combining the above gives us the desired Lemma 2.5.26.

2.5.5 Wrapping up block-value for grouped graph matrix

We are now ready to wrap up this section by our main lemma,

Lemma 2.5.28. *For a fixed active profile $\mathcal{P}$, for any sequence of shapes $\tau_{\mathcal{P}} = \{\tau\}$ with active profiles prescribed by $\tau_{\mathcal{P}}$, let B be the block-value bound, we have*

$$B(\sum_{\tau\in\tau_{\mathcal{P}}} M_\tau) \leq \max_{\substack{\tau\in\tau_{\mathcal{P}} \\ S_\tau:\text{separator for } \tau}} c_{\mathsf{norm}}^{|V(\tau)\setminus(U_\tau\cap V_\tau)|}\cdot c_{\mathsf{norm}}^{|E(S)|}\cdot\sqrt{\frac{n}{d}}^{|E(S_\tau)|}\cdot\sqrt{n}^{|V(\tau)\setminus S_\tau|}\cdot \mathsf{dangling}(\tau\setminus S)\cdot\mathsf{float}(\tau)$$

for some absolute constant c_{norm}.

Proof. This follows by our drawing cost that for any permissible shape, the wedges can be bundled at a cost of $c_{matched}^{|E(S)|}$ since for wedge bundling process, it suffices for us to identify for each active profile its distance to the vertex in S_R (the right boundary of the separator) that it can reach, and notice each vertex passed through the path before arriving at the destination at S_R contributes 1, hence the cost can be loosely bounded by $c_{matched}^{|V(\alpha)\setminus S_R|}$.

Once the shape is identified, the norm bound is then given by our block-value bound for a single shape from Corollary 2.5.13 , which is

$$c_{\mathsf{norm}}^{|V(\tau)\setminus(U_\tau\cap V_\alpha)|}\cdot c_{\mathsf{norm}}^{|E(S)|}\cdot\sqrt{\frac{n}{d}}^{|E(S_\tau)|}\cdot\sqrt{n}^{|V(\tau)\setminus S_\tau|}\cdot \mathsf{dangling}(\tau\setminus S)\cdot\mathsf{float}(\tau)\,.$$

Combining the above yields the desired. □

Remark 2.5.29. *This bound is potentially less meaningful if the sequence of grouped shapes contains shapes of different sizes, as the max will simply be dominated by the largest one due to different scaling of n. However, it will be more transparent in our application to bounding middle shapes and intersection shapes soon as we will account for the different sizes by combining the shapes with their pseudo-calibrated coefficients.*

Chapter 3

Ellipsoid Fitting: An Application to Dense Input

3.1 Introduction

What's the largest m so that for m points $v_1, v_2, \ldots, v_m \in \mathbb{R}^d$ sampled independently from the d-dimensional standard Gaussian distribution $\mathcal{N}(0, I_d)$, there exists an ellipsoid that passes through each of the v_is with high probability? This latter condition is equivalent to asking for a positive semidefinite matrix Λ such that $v_i^\top \Lambda v_i = 1$ for $1 \leq i \leq m$ and thus, equivalently, the question asks for the largest m such that the basic stochastic semidefinite program above remains feasible with high probability.

It is not hard to prove that for any $m \leq d + 1$, an ellipsoid as above exists with high probability over the v_is [SCPW12]. On the other hand, since the dimension of the smallest linear subspace that contains the positive semidefinite cone of $d \times d$ matrices is $\binom{d+1}{2} \sim d^2/2$, it is easy to prove that for $m \gg d^2/2$, there cannot be an ellipsoid passing through v_is with high probability.

In 2013, Saunderson, Parrilo and Willsky [SPW13] studied this basic geometric ques-

tion and conjectured that there is a sharp phase transition for the problem (from feasibility/existence of an ellipsoid to non-existence of an ellipsoid) as m crosses $d^2/4$.

Conjecture 3.1.1 (SCPW conjecture)**.** *Let $\epsilon > 0$ be a constant and $v_1, \ldots, v_m \sim \mathcal{N}(0, I_d)$ be i.i.d. standard Gaussian vectors in $\mathbb{R}^d$. Then,*

1. *If $m \leq (1-\epsilon)\frac{d^2}{4}$, then $v_1, \ldots, v_m$ have the ellipsoid fitting property with probability $1 - o_d(1)$.*

2. *If $m \geq (1+\epsilon)\frac{d^2}{4}$, then $v_1, \ldots, v_m$ have the ellipsoid fitting property with probability $o_d(1)$.*

This bound is $1/2$ of the dimension of the smallest linear subspace containing the positive semidefinite cone of $d \times d$ matrices. Said differently, the SCPW conjecture (developed in a sequence of works [Sau11, SCPW12, SPW13]) posits that the positive semidefiniteness constraint "forces" a drop of a factor 2 in the threshold m for infeasibility. The SCPW conjecture was motivated by results of numerical experiments (see also the experiments presented in [PTVW22]).

Early on, [Sau11, SPW13] established the existence of a feasible ellipsoid for any $m \leq O(d^{6/5-\epsilon})$ whp. Recently, there has been a new wave of progress on this bound. A recent result [GJJ+20b] on establishing Sum-of-Squares lower bounds for the Sherrington-Kirkpatrick model, as a corollary yields an estimate of $m \leq O(d^{3/2-\epsilon})$. In fact, though not explicitly stated, their work already contains ideas that imply a significantly stronger bound of $m \leq d^2/\operatorname{polylog}(d)$. Very recently, two independent works [PTVW22] and [KD22] analyzed two slightly different explicit constructions for Λ to recover a similar bound of $m \leq d^2/\operatorname{polylog}(d)$. In their works [PTVW22, KD22], the authors ask the question of analyzing their construction (or a different one) to obtain an improved and almost optimal estimate of $m = d^2/C$ for some absolute constant $C > 0$.

The main result of this work achieves this goal. Specifically, we prove:

Theorem 3.1.2 (Main result). *For $m \leq c \cdot d^2$ for some universal constant $c > 0$ and $v_1, \ldots, v_m \sim \mathcal{N}(0, I_d)$ drawn independently, with probability at least $1 - o_d(1)$, there exists an ellipsoid passing through each v_i.*

We establish Theorem 3.1.2 by analyzing the construction of Kane and Diakonikolas [KD22] (which is a variant of the construction proposed in [PTVW22]). Our argument can be used to recover a bound of $c \sim 1/10^8$. We have not tried to optimize this bound. Numerical experiments suggest that the [KD22] construction we analyze cannot approach $c = 1/4$, so establishing the sharp constant in the SCPW conjecture will likely need new ideas. Table 3.1 shows a summary of our result compared to prior work.

Construction	**Bound on m**
Conjectured	$d^2/4$
[Sau11, SPW13]	$O(d^{6/5-\epsilon})$
[GJJ+20b]	$O(d^{3/2-\epsilon})$ *
[PTVW22]	$O(d^2/\operatorname{polylog}(d))$
[KD22]	$O(d^2/\log^4(d))$
this paper	$O(d^2)$

*The bound $O(d^2/\operatorname{polylog}(d))$ is implicit in their work.

Table 3.1: Comparison of our result with prior work.

Our key idea departs from the analysis technique of [KD22] and instead relies on the method of *graphical matrix decomposition*. This method decomposes a random matrix with correlated entries into a sum of structured random matrices called *graph matrices*. The key technical work in the analysis then becomes understanding the smallest and the largest singular values of graph matrices. All prior works rely on arguments that establish bounds on the largest singular values that are accurate up to polylogarithmic factors in the underlying dimension of the matrices.

Concurrent Work We note that a concurrent work of Bandeira et. al. [BMMP23] also obtains a sharper analysis of [KD22] to establish a similar result as this work. They analyze

the same construction of identity perturbation as us. In their work, [BMMP23] ask the question of obtaining estimates that hold with inverse exponential failure probability (as opposed to inverse polynomial failure probability that their work establishes). They also outline a proof strategy that could potentially achieve this goal. We note that our proof does indeed recover an inverse exponential failure probability naturally.

3.2 Construction of Ellipsod

Given $v_1, v_2, \ldots, v_m$ that are i.i.d. samples from $\mathcal{N}(0, \frac{1}{d} I_d)$, recall that we must construct a matrix Λ such that (1) $v_i^T \Lambda v_i = 1$ for any $i \in [m]$, and (2) $\Lambda \succeq 0$.

In this section, we describe our candidate matrix (Definition 3.2.1). To prove that it satisfies the two conditions above, we need to analyze certain random matrices (and their inverses) that arise in the construction, which involves decomposing the matrices into simpler components. We will state our key spectral norm bounds (Lemma 3.2.7 and Lemma 3.2.11) whose proofs are deferred to later sections, and complete the proof of Theorem 3.1.2 in Section 3.2.4.

3.2.1 Candidate construction

The following is our candidate matrix Λ, which is the same one as [KD22].

Definition 3.2.1 (Candidate matrix)**.** *Given $v_1, \ldots, v_m \sim \mathcal{N}(0, \frac{1}{d} I_d)$, we define the matrix $\Lambda \in \mathbb{R}^{d \times d}$ to be*

$$\Lambda := I_d - \sum_{i=1}^{m} w_i v_i v_i^T \tag{3.1}$$

where we take $w = (w_1, w_2, \ldots, w_m)$ to be the solution to the following linear system,

$$Mw = \eta$$

for $\eta \in \mathbb{R}^m$ given by

$$\eta_i := \|v_i\|_2^2 - 1, \quad \forall i \in [m] \tag{3.2}$$

and $M \in \mathbb{R}^{m \times m}$ with entries given by

$$M[i,j] := \langle v_i, v_j \rangle^2, \quad \forall i,j \in [m] \tag{3.3}$$

We first make the following simple observation.

Observation 3.2.2. *For any $i \in [m]$, the constraint $v_i^T \Lambda v_i = 1$ is satisfied.*

Proof. For any $i \in [m]$,

$$v_i^T \Lambda v_i = v_i^T I_d v_i - \sum_{j \in [m]} w_j \langle v_i, v_j \rangle^2 = \|v_i\|_2^2 - \langle M[i], w \rangle = \|v_i\|_2^2 - \eta_i = 1\,.$$

Here $M[i]$ is the i-th row of M, and the equality above follows from $Mw = \eta$ and $\eta_i = \|v_i\|_2^2 - 1$ from Equation (3.2). □

Structure of subsequent sections For Λ to be well-defined, we require that M is full rank (hence invertible). Note that it is easy to see that M is positive semidefinite, since M is a Gram matrix with $M[i,j] = \langle v_i^{\otimes 2}, v_j^{\otimes 2} \rangle$. To analyze M, we will show a decomposition of M in Section 3.2.2 that allows us to more easily analyze its inverse. In Section 3.2.3, we will prove that M is in fact positive definite (Lemma 3.2.10).

Next, to prove that $\Lambda \succeq 0$, we will write $\Lambda = I_d - \mathcal{R}$ where

$$\mathcal{R} := \sum_{i=1}^m w_i v_i v_i^T = \sum_{i=1}^m \left(M^{-1}\eta\right)[i] \cdot v_i v_i^T\,, \tag{3.4}$$

and prove that $\|\mathcal{R}\|_{spec}$ is bounded by 1. This is done in Section 3.4. Finally, combining the analyses, we finish the proof of Theorem 3.1.2 in Section 3.2.4.

3.2.2 Decomposition of M

The proof of Theorem 3.1.2 requires a careful analysis of the matrix M from Equation (3.3) and its inverse. To this end, we first decompose M as $M = A + B$ such that intuitively, A is perturbation of a (scaled) identity matrix and B has rank 2. We will later see how this decomposition allows us to analyze M^{-1} more conveniently.

Proposition 3.2.3 (Decomposition of M)**.**

$$M = \underbrace{M_\alpha + M_\beta + M_D + 1 + \frac{1}{d} I_m}_{:=A} + \underbrace{\frac{1}{d} J_m + \frac{1}{d}\left(1_m \cdot \eta^T + \eta \cdot 1_m^T\right)}_{:=B} \tag{3.5}$$

where J_m is the all-ones matrix, $M_\alpha, M_\beta \in \mathbb{R}^{m\times m}$ are matrices with zeros on the diagonal and $M_D \in \mathbb{R}^{m\times m}$ is a diagonal matrix, defined as follows:

- $M_\alpha[i,j] \coloneqq \sum_{a\neq b\in[d]} v_i[a]\cdot v_i[b]\cdot v_j[a]\cdot v_j[b]$ *for* $i \neq j \in [m]$,
- $M_\beta[i,j] \coloneqq \sum_{a\in[d]} \left(v_i[a]^2 - \frac{1}{d}\right)\left(v_j[a]^2 - \frac{1}{d}\right)$ *for* $i \neq j \in [m]$,
- $M_D[i,i] \coloneqq \|v_i\|_2^4 - \frac{2}{d}\|v_i\|_2^2 - 1$ *for* $i \in [m]$.

Proof. For any off-diagonal entry $i \neq j \in [m]$, on the right-hand side we have

$$\begin{aligned} M[i,j] = \langle v_i, v_j\rangle^2 &= \Big(\sum_{a\in[d]} v_i[a]v_j[a]\Big)^2 \\ &= \sum_{a\neq b\in[d]} v_i[a]\cdot v_i[b]\cdot v_j[a]\cdot v_j[b] + \sum_{a\in[d]} v_i[a]^2\cdot v_j[a]^2\,. \end{aligned}$$

The first term is exactly $M_\alpha[i,j]$. For the second term,

$$\sum_{a\in[d]} v_i[a]^2\cdot v_j[a]^2 = \underbrace{\sum_{a\in[d]}\left(v_i[a]^2 - \frac{1}{d}\right)\left(v_j[a]^2 - \frac{1}{d}\right)}_{M_\beta[i,j]} + \underbrace{\frac{\|v_i\|_2^2 - 1}{d}}_{\frac{1}{d}\eta_i} + \underbrace{\frac{\|v_j\|_2^2 - 1}{d}}_{\frac{1}{d}\eta_j} + \frac{1}{d}\,.$$

Thus, $M[i,j] = M_\alpha[i,j] + M_\beta[i,j] + \frac{1}{d} + \frac{1}{d}\left(1_m \cdot \eta^T + \eta \cdot 1_m^T\right)[i,j]$.

For the diagonal entries, the right-hand side of the (i,i) entry is

$$\begin{aligned} M_D[i,i] + (1 + \frac{1}{d}) + \frac{1}{d} + \frac{2}{d}\eta_i &= (\|v_i\|_2^4 - \frac{2}{d}\|v_i\|_2^2 - 1) + 1 + \frac{2}{d} + \frac{2}{d}(\|v_i\|_2^2 - 1) \\ &= \|v_i\|_2^4 = M[i,i]\,. \end{aligned}$$

This completes the proof. □

Remark 3.2.4. *The intention behind this decomposition is so that for $v_i \sim \mathcal{N}(0, \frac{1}{d} I_d)$, M_α, M_β, M_D are all mean 0 (while not the same variance) since $\mathbb{E}\|v_i\|_2^2 = 1$ and $\mathbb{E}\|v_i\|_2^4 = 1 + \frac{2}{d}$. Therefore, we expect $\|M_\alpha + M_\beta + M_D\|_{spec}$ to be small, which implies that A is positive definite and well-conditioned. Furthermore, observe that B has rank 2:*

$$B = \frac{1}{d} J_m + \frac{1}{d}\left(1_m \cdot \eta^T + \eta \cdot 1_m^T\right) = \frac{1}{d}\begin{bmatrix} 1_m & \eta \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1_m \\ \eta \end{bmatrix}. \tag{3.6}$$

3.2.3 Inverse of M

The decomposition of M into A and a rank-2 matrix B (Equation (3.5)) allows us to apply the Woodbury matrix identity about the inverse of low-rank corrections of invertible matrices.

Fact 3.2.5 (Matrix Invertibility)**.** *Suppose $A \in \mathbb{R}^{n_1 \times n_1}$ and $C \in \mathbb{R}^{n_2 \times n_2}$ are both invertible matrices, and $U \in \mathbb{R}^{n_1 \times n_2}$ and $V \in \mathbb{R}^{n_2 \times n_1}$ are arbitrary. Then, $A + UCV$ is invertible if and only if $C^{-1} + VA^{-1}U$ is invertible.*

Fact 3.2.6 (Woodbury matrix identity)**.** *Suppose $A \in \mathbb{R}^{n_1 \times n_1}$ and $C \in \mathbb{R}^{n_2 \times n_2}$ are both invertible matrices, and $U \in \mathbb{R}^{n_1 \times n_2}$ and $V \in \mathbb{R}^{n_2 \times n_1}$ are arbitrary. Then*

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1} VA^{-1}.$$

In light of Fact 3.2.6, we can write B in Equation (3.6) as $B = UCU^T$ where $U = V^T = \frac{1}{\sqrt{d}}\begin{bmatrix} 1_m & \eta \end{bmatrix} \in \mathbb{R}^{m\times 2}$ and $C = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$, and $M = A + UCU^T$. Note that $C^{-1} = \begin{bmatrix} 0 & 1 \\ 1 & -1 \end{bmatrix}$, and we have

$$C^{-1} + U^T A^{-1} U = \begin{bmatrix} \frac{1_m^T A^{-1} 1_m}{d} & 1 + \frac{\eta^T A^{-1} 1_m}{d} \\ 1 + \frac{\eta^T A^{-1} 1_m}{d} & -1 + \frac{\eta^T A^{-1} \eta}{d} \end{bmatrix} =: \begin{bmatrix} r & s \\ s & u \end{bmatrix}. \tag{3.7}$$

We first need to show that A is invertible. Recall from Equation (3.5) that $A = (1 + \frac{1}{d})I_m + M_\alpha + M_\beta + M_D$. We will prove the following lemma, whose proof is deferred to Section 3.3.4.

Lemma 3.2.7 (M_α, M_β, M_D are bounded)**.** *Suppose $m \le cd^2$ for a small enough constant c. With probability $1 - o_d(1)$, we have*

1. $\|M_\alpha\|_{spec} \le 0.1$,
2. $\|M_\beta\|_{spec} \le 0.1$,
3. $\|M_D\|_{spec} \le O(\sqrt{\frac{\log d}{d}})$.

As an immediate consequence, we get the following:

Lemma 3.2.8 (A is well-conditioned)**.** *With probability $1 - o_d(1)$, the matrix A from Equation (3.5) is positive definite (hence full rank), and*

$$0.5 I_m \preceq A \preceq 1.5 I_m .$$

Proof. Since $A = (1 + \frac{1}{d})I_m + M_\alpha + M_\beta + M_D$, by Lemma 3.2.7 the eigenvalues of A must lie within $1 \pm 0.2 \pm \widetilde{O}(1/\sqrt{d}) \in (0.5, 1.5)$ (we assume d is large). □

Next, from Fact 3.2.5, we can prove that M is invertible (Lemma 3.2.10) by showing that the 2×2 matrix $C^{-1} + U^T A^{-1} U$ is invertible, which is in fact equivalent to $ru - s^2 \neq 0$. We first need the following bound on the norm of η.

Claim 3.2.9. *With probability at least* $1 - o_d(1)$,

$$\|\eta\|_2^2 \leq (1 + o_d(1)) \frac{2m}{d}.$$

Lemma 3.2.10 (Bounds on r, s, u; M is invertible)**.** *Suppose* $m \leq cd^2$ *for a small enough constant* c. *Let* $r, s, u \in \mathbb{R}$ *be defined as in Equation* (3.7). *With probability at least* $1 - o_d(1)$, *we have*

1. $r \in \frac{m}{d} \cdot [2/3, 2]$,
2. $|s| \leq 1 + o_d(1)$,
3. $u \in [-1, -1/2]$.

Thus, we have

$$s^2 - ru \geq \Omega(\frac{m}{d})$$

As a consequence, M *is invertible.*

Proof. By lemma 3.2.8, we know that $\frac{2}{3} I_m \preceq A^{-1} \preceq 2 I_m$. Thus, $r = \frac{1}{d} 1_m^T A^{-1} 1_m \in \frac{1}{d} \|1_m\|_2^2 \cdot [2/3, 2]$, hence $r \in \frac{m}{d} \cdot [2/3, 2]$.

For u, we have

$$\frac{1}{d}(\eta^T A^{-1} \eta) \leq \frac{1}{d} \|A^{-1}\|_{spec} \cdot \|\eta\|_2^2 < (1 + o_d(1)) \cdot \frac{4m}{d^2} < \frac{1}{2}.$$

where the last inequality follows for some $m < cd^2$ for small enough c. Thus, $u = -1 + \frac{\eta^T A^{-1} \eta}{d} \in [-1, -1/2]$.

We defer the proof for s to the original work. With the bounds on r, s and u, we immediatley get $s^2 - ru \geq \Omega(\frac{m}{d})$.

To prove that M is invertible, let us first recall that we write $M = A + UCU^T$ where A is defined in Equation (3.5) and $U = V^T = \frac{1}{\sqrt{d}} \begin{bmatrix} 1_m & \eta \end{bmatrix} \in \mathbb{R}^{m \times 2}$ and $C = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$.

By Lemma 3.2.8, A is invertible. Then by Fact 3.2.5, we know that M is invertible if and only if $C^{-1} + U^T A^{-1} U := \begin{bmatrix} r & s \\ s & u \end{bmatrix}$ (see Equation (3.7)) is invertible, which is equivalent to $ru - s^2 \neq 0$. Thus, $s^2 - ru \geq \Omega(\frac{m}{d})$ suffices to conclude that M is invertible.

□

3.2.4 Finishing the proof of Theorem 3.1.2

The final piece of proving Theorem 3.1.2 is to show that $\mathcal{R} = \sum_{i=1}^{m} w_i v_i v_i^T$ has spectral norm bounded by 1, which immediately implies that the candidate matrix $\Lambda = I_d - \mathcal{R} \succeq 0$.

Lemma 3.2.11 ($\mathcal{R}$ is bounded). *There exists some absolute constant C_R such that for $m \leq \frac{d^2}{C_R}$,*

$$\|\mathcal{R}\|_{spec} \leq \frac{1}{2}.$$

The proof is deferred to Section 3.4. In particular, we will write an expanded expression of M^{-1} and obtain a decomposition of $\mathcal{R}$ (Proposition 3.4.4). Then, in Section 3.4.2, we prove tight spectral norm bounds for matrices in the decomposition, which then completes the proof of Lemma 3.2.11.

Combining Lemma 3.2.10 and lemma 3.2.11 we can finish the proof of Theorem 3.1.2.

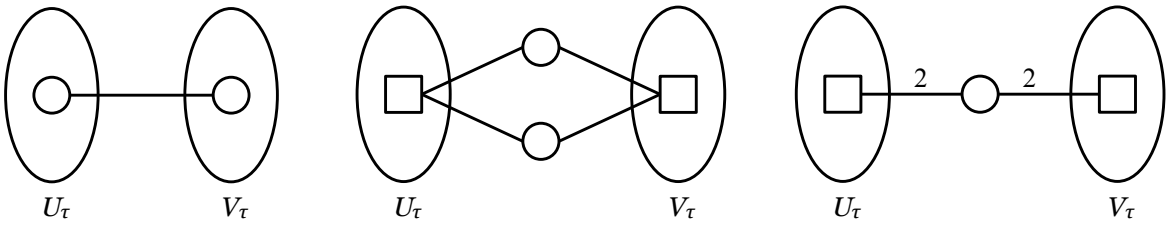
Proof. The matrix M (recall Equation (3.3)) is invertible due to Lemma 3.2.10, thus our candidate matrix $\Lambda = I_d - \mathcal{R}$ matrix defined in Definition 3.2.1 is well-defined. Furthermore, by Lemma 3.2.11 we have that $\|\mathcal{R}\|_{spec} < 1$. This proves that $\Lambda \succ 0$. □

3.3 Graph Matrix Application and Analysis

3.3.1 Adaptation in this Work

Figure 3.1 show the shapes for matrices M_α and M_β defined in Proposition 3.2.3. For these shapes, there are two vertex-types (square and circle). The two ovals in each shape indicate the left and right boundaries U_τ and V_τ.

We next describe how to associate a shape to a matrix (given the underlying matrix G).



(a) GOE, zero diagonal. (b) M_α. (c) M_β.

Figure 3.1: Graph matrix representation of a $d \times d$ GOE matrix with zero diagonal, and the $m \times m$ matrices M_α and M_β as defined in Proposition 3.2.3. Square vertices take labels in $[m]$ and circle vertices take labels in $[d]$. The two ovals indicate the left and right boundaries U_τ, V_τ of the shapes. If an edge e is not labeled with an index, then $t(e) = 1$ by default.

Definition 3.3.1 (Mapping of a shape)**.** *Given a shape τ, we call a function $\sigma : V(\tau) \to \mathbb{N}$ a mapping of the shape if*

1. *σ assigns a label for each vertex according to its specified vertex-type;*
2. *σ is an injective mapping for vertices of the same type.*

Definition 3.3.2 (Graph matrix for shape)**.** *Given a shape τ, we define its graphical matrix M_τ to be the matrix indexed by all possible boundary labelings of S, T, and for each of its entry, we define*

$$M_\tau[S,T] = \sum_{\substack{\sigma: V(\tau)\to\mathbb{N} \\ \sigma(U_\tau)=S,\, \sigma(V_\tau)=T}} \prod_{e\in E(\tau)} \chi_{t(e)}(G[\sigma(e)])\,.$$

Observe that for each entry $M_\tau[S,T]$, since σ must map U_τ and V_τ to S and T, $M_\tau[S,T]$ is simply a sum over labelings of the "middle" vertices $V(\tau) \setminus (U_\tau \cup V_\tau)$. Take Figure 3.1 for example. Suppose $G \in \mathbb{R}^{m\times d}$ and square and circle vertices take labels in $[m]$ and $[d]$ respectively, then we can write out the entries of the matrix: for $i \neq j \in [m]$,

$$M_\alpha[i,j] = \sum_{a\neq b\in[d]} \chi_1(G[i,a]) \cdot \chi_1(G[i,b]) \cdot \chi_1(G[j,a]) \cdot \chi_1(G[j,b])\,,$$
$$M_\beta[i,j] = \sum_{a\in[d]} \chi_2(G[i,a]) \cdot \chi_2(G[j,a])\,.$$

Note also that since σ must be injective for vertices of the same type and $U_\tau \neq V_\tau$ in both examples, there is no mapping such that $\sigma(U_\tau) = \sigma(V_\tau)$. Thus, by Definition 3.3.2, both matrices have zeros on the diagonal.

In this work, we specialize to the following setting:

- $G \in \mathbb{R}^{m\times d}$ is a random Gaussian matrix whose rows are $v_1,\dots,v_m \sim \mathcal{N}(0,\frac{1}{d}I_d)$.
- The Fourier characters $\{\chi_t\}_{t\in\mathbb{N}}$ are the (scaled) Hermite polynomials.
- For all graph matrices that arise in our analysis,
  - $|S| = |T| = 1$,
  - There are two vertex-types: square vertices take labels in $[m]$ and circle vertices take labels in $[d]$.

3.3.2 Technical Verification of Factor Assignment

Vertex factor assignment scheme We now proceed to bound the vertex factors for each step-label. We note that in this section, "vertices" refer to "labeled vertices" in the walk (having labels in $[m]$ or $[d]$). First, we define the weight of a square (resp. circle) vertex to

be m (resp. d), since we need an element in $[m]$ (resp. $[d]$) to specify which vertex to go to in the walk.

We first show a "naive" vertex factor assignment scheme. In the following scheme, we use a *potential unforced return* factor, denoted Pur, to specify the destination of any R step. We will defer the specific details of Pur to Section 3.3.3.

Vanilla vertex factor assignment scheme

1. For each vertex i that first appears via an F step, a label in $\mathsf{weight}(i)$ is required;
2. For each vertex i that appears beyond the first time:
   - If it is arrived via an R step, the destination may need to be specified, and this is captured by the Pur factor.
   - If it is *not* arrived via an R step, then it must be an S or H step. A vertex cost in $2q \cdot D_V$ is sufficient to identify the destination, where we recall $2q$ is the length of our walk, and D_V the size upper bound of each block.

The first thing to check is that this scheme combined with an step-labeling uniquely identifies a closed walk (given the start of the walk). This is immediate for F and R steps by definition. For S and H steps, since the destination is visited before in the walk, $2q \cdot D_V$ is sufficient as it is an upper bound on the number of vertices in the walk.

A potential complication with analyzing the above assignment scheme directly is that it exhibits a significant difference in the vertex factors. For example, consider a vertex that appears only twice in the walk on a tree. Its first appearance requires a label in $[n]$, while its subsequent appearance does not require any cost if it is reached using an R step because backtracking from a tree is fixed (since there is only one parent). This disparity can result in a very loose upper bound for the trace; in fact, the norm bound for M_τ obtained in this manner is equivalent to using the naive row-sum bound.

Redistribution One of our main technical insights is to split the factors such that both first and last appearance contributes a factor of comparable magnitude; we call this *redistribution*.

We first formally define "appearance" in a block-step to clarify our terminology,

Definition 3.3.3 (Vertex appearance in block-step)**.** *Each labeled vertex appearance can be "first", "middle" and "last". Moreover, each vertex on the block-step boundary (U_τ or V_τ) appears in both adjacent blocks.*

For example, suppose a vertex first appears in the right-boundary of block i and last appears in the left-boundary of block j, then it will make middle appearances in the left-boundary of block $i+1$ and right-boundary of block $j-1$ as well.

We are now ready to introduce the following vertex-factor assignment scheme with redistribution that assigns vertex-factor to each vertex's appearance to handle the disparity.

Vertex factor assignment scheme with redistribution

1. For each vertex i that makes its first appearance, assign a cost of $\sqrt{\mathsf{weight}(i)}$;
2. For any vertex's middle appearance, if it is not arrived at via an R step, assign a cost of $2q \cdot D_V$ (where we recall $2q$ is the length of our walk, and D_V the size constraint of each block);
3. For any vertex's middle appearance, if it is arrived at via an R step, its cost is captured by Pur;
4. For each vertex i that makes its last appearance, assign a cost of $\sqrt{\mathsf{weight}(i)}$ that serves as a *backpay*.

Deducing vertex factor from local step-labeling As presented, the vertex factor assignment scheme requires knowing which vertex is making first/middle/last appearance. We

further show that the vertex appearances, or more accurately, an upper bound of the vertex factors, can be deduced by a given step-labeling of the block. Fix traversal direction from U to V,

Localized vertex factor assignment from step-labeling

1. For any vertex v that is on the left-boundary U, it cannot be making the first appearance since it necessarily appears in the previous block;
2. For any vertex v that is on the right-boundary V, it cannot be making the last appearance since it necessarily appears in the subsequent block;
3. For any vertex v reached via some $S/R/H$ step, it cannot be making its first appearance;
4. For any vertex v that incident to some $F/S/H$ step, it cannot be making its last appearance since the edge necessarily appears again.

The first two points are due to Definition 3.3.3. The last point is because each labeled edge (i.e., Fourier character) must be traversed by an R step to close it.

Bounding edge-factors To bound the contribution of the walks, we need to consider factors coming from the edges traversed by the walk. Recall that each edge e in a closed walk P gets a factor $\mathbb{E}[\chi_{t(e)}^{\mathrm{mul}_P(e)}]$, where $t(e)$ is the Fourier index associated with the edge.

In our case, the Fourier characters are the scaled Hermite polynomials. Recall that we assume that our vectors are sampled as $v_i \sim \mathcal{N}(0, \frac{1}{d} I_d)$. Thus, we define the polynomials $\{h_t\}_{t\in\mathbb{N}}$ such that they are orthogonal and $\mathbb{E}_{x\sim\mathcal{N}(0,1/d)}[h_t(x)^2] = t! \cdot d^{-t}$. Specifically,

1. $h_1(x) = x$,
2. $h_2(x) = x^2 - \frac{1}{d}$.

We first state the following bound on the moments of h_t, which follows directly from standard bounds on the moments of Hermite polynomials:

Fact 3.3.4 (Moments of Hermite polynomials)**.** *Let $d \in \mathbb{N}$. For any $t \in \mathbb{N}$ and even $k \in \mathbb{N}$,*

$$\mathbb{E}_{x \sim \mathcal{N}(0,1/d)}[h_t(x)^k] \leq \frac{1}{d^{kt/2}}(k-1)^{kt/2}(t!)^{k/2} \leq (t!)^{k/2}(\frac{k}{d})^{kt/2}.$$

For now, we consider matrices that either contain only h_1 or only h_2 edges (the edge factors for graph matrices with "mixed" edges will be handled in Section 3.4.3). The following is our edge-factor assignment scheme to account for contributions from the Fourier characters.

Edge-factor assignment scheme

For an h_1 edge,

1. F/S: assign a factor of $\frac{1}{\sqrt{d}}$ for its first appearance;
2. H: assign a factor of $\frac{2q}{\sqrt{d}}$ for its middle appearance;
3. R: assign a factor of $\frac{1}{\sqrt{d}}$ for its last appearance.

For an h_2 edge,

1. F/S: assign a factor of $\frac{\sqrt{2}}{d}$ for its first appearance (alternatively, we can view a single h_2 edge as two edge-copies of h_1 and assign each a factor of $\frac{\sqrt{2}}{\sqrt{d}}$, which is a valid upper bound);
2. H: assign a factor of $\frac{8q^2}{d}$ for its middle appearance;
3. R: assign a factor of $\frac{\sqrt{2}}{d}$ for its last appearance (alternatively, we can view a single h_2 edge as two edge-copies of h_1 and assign each a factor of $\frac{\sqrt{2}}{\sqrt{d}}$ which is a valid upper bound).

Proposition 3.3.5. *The above scheme correctly accounts for the edge factors from h_1 and h_2 edges.*

Proof. If an edge has multiplicity 2 then it must be traversed by one F/S step and one R step.

- If it is an h_1 edge, then the scheme assigns a factor $\frac{1}{d}$, which equals $\mathbb{E}_{x\sim\mathcal{N}(0,1/d)}[h_1(x)^2]$.
- If it is an h_2 edge, then the scheme assigns a factor $\frac{2}{d^2}$, which equals $\mathbb{E}_{x\sim\mathcal{N}(0,1/d)}[h_2(x)^2]$.

For an edge with multiplicity $k > 2$, it must be traversed by one F step (including S), one R step and $k-2$ H steps. Moreover, since k is even and $2q$ is the length of the walk, we have $4 \le k \le 2q$.

- If it is an h_1 edge, then the scheme assigns a factor $\frac{1}{d} \cdot (\frac{2q}{\sqrt{d}})^{k-2} \ge d^{-k/2}(2q)^{k/2} \ge (\frac{k}{d})^{k/2}$. By Fact 3.3.4, it is an upper bound on $\mathbb{E}_{x\sim\mathcal{N}(0,1/d)}[h_1(x)^k]$.
- If it is an h_2 edge, then the scheme assigns a factor $\frac{2}{d^2} \cdot (\frac{8q^2}{d})^{k-2} \ge d^{-k}2^{k/2}(2q)^k \ge 2^{k/2}(\frac{k}{d})^k$. By Fact 3.3.4, it is an upper bound on $\mathbb{E}_{x\sim\mathcal{N}(0,1/d)}[h_2(x)^k]$.

This shows that the edge factor assignment scheme above is correct. □

3.3.3 Bounding return cost (Pur factors)

In our vertex factor assignment scheme, we use a *potential unforced return* factor, denoted Pur, to specify the destination of any return (R) step. Note that the term "unforced return" is adopted from [Tao12] as well. In this section, we complete the bound of vertex factors by bounding the Pur factor.

For starters, we will define a potential function for each vertex at time t, which measures the number of returns R pushed out from the particular vertex by time t that may require a label in $2q \cdot D_V$. Notice that a label in $2q \cdot D_V$ is sufficient for any destination vertex arrived via an R step because the vertex appears before; however, this may be a loose bound.

We observe the following: a label in $2q \cdot D_V$ may be spared if the vertex is incident to only one unclosed F/S edge; we call this a *forced* return. Formally, we define a return step as *unforced* if it does not fall into the above categories,

Definition 3.3.6 (Unforced return)**.** *We call a return (R) step an* unforced return *if the source vertex is incident to more than* 1 *(or* 2 *in the case of a square vertex) unclosed edge.*

We now proceed to formalize the above two observations by introducing a potential function to help us bound the number of unforced returns from any given vertex throughout the walk. The number of unforced returns throughout the walk would then be immediately given once we sum over all vertices in the walk.

Definition 3.3.7 (Potential-unforced-return factor Pur)**.** *For any time t and vertex v, let $\mathsf{Pur}_t(v)$ be defined as the number of potential unforced return from v throughout the walk until time t.*

Pur bound for circle vertices

In our setting, each circle vertex pushes out at most 1 edge during the walk, analogous to the case of typical adjacency matrix. This serves as a starting point for our Pur bound for circle vertices.

Lemma 3.3.8 (Bounding Pur_t for circle vertices)**.** *For any time t, suppose the walker is currently at a circle vertex v, then*

$$\begin{aligned}\mathsf{Pur}_t(v) &\leq \#(R \text{ steps closed from } v) + \#(\text{unclosed edges incident to } v \text{ at time } t) - 1 \\ &\leq 2 \cdot s_t(v) + h_t(v)\,,\end{aligned}$$

where we define the following counter functions:

1. *$s_t(v)$ is the number of S steps arriving at v by time t;*

2. *$h_t(v)$ is the number of H steps arriving at v by time t.*

Proof. We first prove the first inequality. The R steps closed from v may all be unforced returns, and the unclosed edges incident to v may be closed by unforced returns in the future. Note that we have a -1 in the above bound because for each vertex we may by default assume the return is using a particular edge, hence at each time we know there is an edge presumed-to-be forced.

We prove the second inequality by induction. Define $P_t(v) :=$ #(R steps closed from v) + #(unclosed edges incident to v at time t) $- 1$ for convenience. At the time when v is first created by an F step, $P_t(v) = 0$ (1 open edge minus 1) and $s_t(v) = h_t(v) = 0$.

At time t, suppose the last time v was visited was at time $t' < t$, and suppose that the inequality holds true for t'. Note that at time $t' + 1$, $P_{t'+1}(v) = P_{t'}(v) + 1$ if a new edge was created by an F or N step leaving v, otherwise $P_{t'+1}(v) = P_{t'}(v)$ (for R step it adds 1 to the number of closed edges closed from v, but decreases 1 open edge). On the other hand, $s_{t'}(v)$ and $h_{t'}(v)$ remain the same (we don't count out-going steps for $s_t(v), h_t(v)$).

When we reach v at time t, we case on the type of steps:

- Arriving by an R step: the edge is now closed, but the R step was not from v. So $P_t(v) = P_{t'+1}(v) - 1 \leq P_{t'}(v)$, while $s_t(v) = s_{t'}(v)$ and $h_t(v) = h_{t'}(v)$.

- Arriving by an S step: the edge is new, so $P_t(v) = P_{t'+1}(v) + 1 \leq P_{t'}(v) + 2$, and we have $s_t(v) = s_{t'}(v) + 1$.

- Arriving by an H step: $P_t(v) = P_{t'+1}(v) \leq P_{t'}(v) + 1$, and $h_t(v) = h_{t'}(v) + 1$.

In all three cases, assuming $P_{t'}(v) \leq 2 \cdot s_{t'}(v) + h_{t'}(v)$, we have $P_t(v) \leq 2 \cdot s_t(v) + h_t(v)$, completing the induction. □

Pur bound for square vertices

The argument of Lemma 3.3.8 does not apply well for vertices incident to multiple edges in a single step. In particular, this may happen for square vertices in M_α as each is arrived via 2 edges and each pushes out 2 edges (recall Figure 3.1). This is not an issue for M_β, but we will treat square vertices in M_β the same way to unify the analysis; in the context of Pur for square vertices, one may think of M_β as collapsing the two circle vertices in M_α.

To handle this issue, we observe that it suffices for us to pay an extra cost of [2] for each square vertex, which would allow us to further presume 2 edges being forced. We then generalize the prior argument to capture this change.

Lemma 3.3.9 (Bounding Pur_t for square vertices)**.** *For any time t, suppose the walker is currently at a square vertex v, then*

$$\begin{aligned}\mathsf{Pur}_t(v) &\le \#(R \text{ steps closed from } v) + \#(\text{unclosed edges incident to } v \text{ at time } t) - 2\\ &\le 2(s_t(v) + h_t(v))\,.\end{aligned}$$

where $s_t(v)$ and $h_t(v)$ are the number of S and H steps arriving at v by time t, respectively.

Proof. We prove this by induction. Note that this is immediate for the base case when v first appears since a square vertex is incident to 2 edges. Define $P_t(v) := \#(\text{R steps closed from } v) + \#(\text{unclosed edges incident to } v \text{ at time } t) - 2$ for convenience. Suppose the inequality is true at time t', and assume vertex v appears again at time t. The departure at time $t'+1$ from v may open up at most 2 edges, hence $P_{t'+1}(v) \le P_{t'}(v) + 2$.

When we reach v at time t (via 2 edges), we case on the type of steps:

- Arriving by two R steps: the two edges closed by the R steps are not closed from v. So $P_t(v) = P_{t'+1} - 2 \le P_{t'}(v)$, while $s_t(v) = s_{t'}(v)$ and $h_t(v) = h_{t'}(v)$.

- Arriving by one S/H and one R step: in this case, $P_t(v) = P_{t'+1}(v) \leq P_{t'}(v) + 2$ and $s_t(v) + h_t(v) = s_{t'}(v) + h_{t'}(v) + 1$.

- Arriving by two S/H steps: in this case, $P_t(v) = P_{t'+1}(v) + 2 \leq P_{t'}(v) + 4$, whereas $s_t(v) + h_t(v) = s_{t'}(v) + h_{t'}(v) + 2$.

In all three cases, we have $P_t(v) \leq 2(s_t(v) + h_t(v))$, completing the induction. □

Corollary 3.3.10. *For each surprise/high-mul step, it suffices for us to assign* 2 Pur *factors, which is a cost of* $(2q \cdot D_V)^2$ *so that each* Pur *factor throughout the walk is assigned. Moreover, for* M_α*, we pay a cost of* 2 *for any R step leaving a square vertex so that we can presume* 2 *edges being forced in Lemma 3.3.9.*

3.3.4 Wrapping up with examples

Recall that for a graph matrix of shape τ,

$$B_q(\tau) = \sum_{\mathcal{L}\text{: step-labelings for } E(\tau)} \mathsf{vtxcost}(\mathcal{L}) \cdot \mathsf{edgeval}(\mathcal{L}) \tag{3.8}$$

is a valid block-value function for τ Moreover, we can take $q \geq d^\epsilon$ and conclude that with probability $1 - 2^{-d^\epsilon}$,

$$\|M_\tau\|_{spec} \leq (1 + o(1)) \cdot B_q(\tau)\,.$$

For each given shape, it suffices for us to bound the block-value for each edge-labeling. And we demonstrate how this may be readily done given the above bounds.

Bound for M_β

We now prove a bound on $\|M_\beta\|_{spec}$. Figure 3.1c shows the associated shape.

Lemma 3.3.11 (Norm bound for M_β)**.** *Let $m \geq \omega(d)$ and $q = \Omega(\log^2 d)$. Then with probability $1 - 2^{-q/\log d}$,*

$$\|M_\beta\|_{spec} \leq (1 + o_d(1))\frac{2m}{d^2}.$$

Proof. Let β be the shape associated with the matrix M_β. We bound $B_q(\beta)$ via our vertex/edge factor assignment schemes combined with Pur factors. Recall that each square vertex has weight m and each circle vertex has weight d. We case on the step-labels of the two edges,

1. $F \to F$: we have an F edge leading to a square vertex and circle vertex each. The first square vertex must be making a middle appearance (Definition 3.3.3), while the circle and the other square vertex make first appearances, giving a vertex factor of $\sqrt{m} \cdot \sqrt{d}$. Furthermore, there is no Pur factor incurred. Finally, the edge factor is $\frac{2}{d^2}$.

2. $R \to R$: we have both R edges departing from a square and circle vertex. There is no vertex making first appearance and the square vertex on the right must be making middle appearance. The other two vertices may be making last appearances. Furthermore, there is no Pur factor incurred, while we assume each R edge from a square vertex can be identified at a cost of $[2]$ modulo the ones with assigned Pur factor, giving a total vertex factor of $2\sqrt{m} \cdot \sqrt{d}$. Finally, the edge factor is $\frac{2}{d^2}$.

3. $R \to F$: the circle vertex must be making a middle appearance, since the F edge must be closed later. The square vertex on the left may be making a last appearance, and the square vertex on the right must be making a first appearance. This gives a vertex factor of $\sqrt{m} \cdot \sqrt{m} = m$. There is no Pur factor, and the edge factor is $\frac{2}{d^2}$.

4. $F \to R$: this cannot happen. The F step means that the circle vertex is making its first appearance, but the R step means that it must have appeared before.

5. For any other step-labelings involving S and H step-labels, both vertices of the S/H edge must be making middle appearances. Thus, the vertex factor is at most $\sqrt{m}$. By Lemmas 3.3.8 and 3.3.9, the Pur factor is at most $(2q)^4$. Finally, the edge factor is at most $O(\frac{q^4}{d^2})$.

Summing over all possible step-labelings, we get

$$B_q(\beta) \leq \sqrt{md} \cdot \frac{2}{d^2} + 2\sqrt{md} \cdot \frac{2}{d^2} + m \cdot \frac{2}{d^2} + \frac{q^{O(1)}}{d^2} \leq \frac{2m}{d^2}(1 + o_d(1)),$$

provided that $m \gg d$. Therefore, we have that with probability $1 - 2^{-q/\log d}$, $\|M_\beta\|_{spec} \leq (1 + o_d(1))\frac{2m}{d^2}$. □

Bound for M_α

We now prove a bound on $\|M_\alpha\|_{spec}$. Figure 3.1b shows the associated shape.

Lemma 3.3.12 (Norm bound for M_α)**.** *Let $m \geq \omega(d)$ and $q = \Omega(\log^2 d)$. Then with probability $1 - 2^{-q/\log d}$,*

$$\|M_\alpha\|_{spec} \leq (1 + o_d(1)) \cdot \frac{1}{d^2}(3d\sqrt{m} + 2m).$$

Proof. Let α be the shape associated with the matrix M_α. Similar to the proof of Lemma 3.3.11, we bound $B_q(\alpha)$ by casing on the step-labelings. There are two paths from U_α to V_α, 4 edges in total. t

1. In the case of all F or all R, one of the square vertices must be making middle appearance, hence we get a vertex factor of $\sqrt{m} \cdot (\sqrt{d})^2 = d\sqrt{m}$. There is no Pur factor, and the edge factor is $(\frac{1}{\sqrt{d}})^4 = \frac{1}{d^2}$. For the case of all R, by Corollary 3.3.10 we pick up an additional factor of [2] since we assume each R edge from a square vertex can be identified at a cost [2] modulo those with assigned Pur factor.

2. If both paths are $R \to F$, then both circle vertices are making middle appearances, hence we get a vertex factor of $\sqrt{m} \cdot \sqrt{m} = m$. There is no Pur factor while we pick up a factor [2] for the return from the square vertex. Finally, the edge factor is $(\frac{1}{\sqrt{d}})^4 = \frac{1}{d^2}$.

3. Analogous to M_β, an $F \to R$ path cannot happen.

4. For any other step-labelings involving S and H step-labels, there must be at least one square and one circle vertex making middle appearances, so the vertex factor is at most $\sqrt{m} \cdot \sqrt{d}$. The Pur factor is $(2q \cdot D_V)^{O(1)} = \text{polylog}(d)$, and the edge factor is $(\frac{2q}{\sqrt{d}})^4$.

Summing over all possible step-labelings, we get

$$B_q(\alpha) \leq d\sqrt{m} \cdot \frac{1}{d^2} + 2d\sqrt{m} \cdot \frac{1}{d^2} + 2m \cdot \frac{1}{d^2} + \sqrt{md} \cdot \frac{q^{O(1)}}{d^2} \leq (1 + o_d(1)) \cdot \frac{1}{d^2}(3d\sqrt{m} + 2m)\,.$$

Therefore, we have that with probability $1 - 2^{-q/\log d}$, $\|M_\alpha\|_{spec} \leq (1 + o_d(1)) \cdot \frac{1}{d^2}(3d\sqrt{m} + 2m)$. □

3.4 Matrix decomposition of $\mathcal{R}$ and Positivity Analysis

In this section, we work towards analyzing $\mathcal{R}$ and proving Lemma 3.2.11. Recall our candidate matrix $\Lambda = I_d - \mathcal{R} \in \mathbb{R}^{d \times d}$ from Definition 3.2.1, where

$$\mathcal{R} := \sum_{i=1}^{m} w_i v_i v_i^T = \sum_{i=1}^{m} \left(M^{-1}\eta\right)[i] \cdot v_i v_i^T\,.$$

See Equations (3.2) and (3.3) for a reminder of the definitions of $M \in \mathbb{R}^{m \times m}$ and $\eta \in \mathbb{R}^m$.

To analyze $\mathcal{R}$, we begin by obtaining an explicit expression for M^{-1} using the Woodbury matrix identity (Fact 3.2.6), as discussed in Section 3.2.3. Recall that we write

$M = A + UCU^T$ where A is positive definite with high probability by Lemma 3.2.8, and $U = \frac{1}{\sqrt{d}} \begin{bmatrix} 1_m & \eta \end{bmatrix} \in \mathbb{R}^{m \times 2}$ and $C = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$. Restating Equation (3.7), the scalars $r, s, u \in \mathbb{R}$ are defined as follows,

$$\begin{bmatrix} r & s \\ s & u \end{bmatrix} := C^{-1} + U^T A^{-1} U = \begin{bmatrix} \frac{1_m^T A^{-1} 1_m}{d} & 1 + \frac{\eta^T A^{-1} 1_m}{d} \\ 1 + \frac{\eta^T A^{-1} 1_m}{d} & -1 + \frac{\eta^T A^{-1} \eta}{d} \end{bmatrix}. \tag{3.9}$$

Our next step is to show the following expansion of $M^{-1}\eta$.

Proposition 3.4.1 (Expansion of $M^{-1}\eta$)**.** *Let $r, s, u \in \mathbb{R}$ be defined as in Equation (3.9). Then,*

$$M^{-1}\eta = \frac{r+s}{s^2 - ru} \cdot A^{-1}\eta - \frac{u+s}{s^2 - ru} \cdot A^{-1} 1_m \,. \tag{3.10}$$

Proof. The inverse of Equation (3.9) is as follows,

$$(C^{-1} + U^T A^{-1} U)^{-1} = \begin{bmatrix} r & s \\ s & u \end{bmatrix}^{-1} = \frac{1}{ru - s^2} \begin{bmatrix} u & -s \\ -s & r \end{bmatrix}.$$

Then, applying the Woodbury matrix identity (Fact 3.2.6), we have

$$\begin{aligned} M^{-1} &= A^{-1} - \frac{1}{ru - s^2} A^{-1} U \begin{bmatrix} u & -s \\ -s & r \end{bmatrix} U^T A^{-1} \\ &= A^{-1} + \frac{1}{s^2 - ru} A^{-1} \left(u \cdot \frac{1_m 1_m^T}{d} - s \cdot \frac{\eta 1_m^T + 1_m \eta^T}{d} + r \cdot \frac{\eta \eta^T}{d} \right) A^{-1} \,. \end{aligned}$$

Next, using the above, we have

$$M^{-1}\eta = A^{-1}\eta + \frac{1}{s^2 - ru} \Big((u \cdot \frac{1_m^T A^{-1} \eta}{d} - s \cdot \frac{\eta^T A^{-1} \eta}{d}) \cdot A^{-1} 1_m$$

$$+ (-s \cdot \frac{1_m^T A^{-1}\eta}{d} + r \cdot \frac{\eta^T A^{-1}\eta}{d}) \cdot A^{-1}\eta \Big) .$$

Plugging in the definition of r, s, u in Equation (3.9), we get

$$\begin{aligned} M^{-1}\eta &= A^{-1}\eta + \frac{1}{s^2 - ru}(u(s-1) - s(u+1))A^{-1}1_m + \frac{1}{s^2 - ru}(-s(s-1) + r(u+1))A^{-1}\eta \\ &= A^{-1}\eta - \frac{u+s}{s^2 - ru} \cdot A^{-1}1_m + \frac{-s^2 + ru + r + s}{s^2 - ru} \cdot A^{-1}\eta \\ &= \frac{r+s}{s^2 - ru} \cdot A^{-1}\eta - \frac{u+s}{s^2 - ru} \cdot A^{-1}1_m , \end{aligned}$$

finishing the proof. □

3.4.1 Inverse of A: Neumann series and truncation

In light of Proposition 3.4.1, we proceed to analyze A^{-1}. Recall from Proposition 3.2.3 that $A = I_m + M_\alpha + M_\beta + M_D + \frac{1}{d} I_m$. A useful tool to obtain inverses is to apply Neumann series (a.k.a. matrix Taylor expansion of $\frac{1}{1-x}$), which allows us to write

$$(I - T)^{-1} = \sum_{k=0}^{\infty} T^k$$

for $\|T\|_{spec} < 1$. In our case, let $T := -(M_\alpha + M_\beta + M_D + \frac{1}{d} I_m)$, then $\|T\|_{spec} < 1$ is guaranteed by Lemma 3.2.7. Thus, we can write

$$A^{-1} = \sum_{k=0}^{\infty} T^k = \sum_{k=0}^{\infty} (-1)^k \sum_{(Q_1, \ldots, Q_k) \in \{M_\alpha, M_\beta, M_D, \frac{1}{d} I_m\}^k} Q_1 Q_2 \cdots Q_k .$$

With the norm bounds from Lemma 3.2.7, we can truncate the series, i.e., capping the number of occurrences of M_α, M_β, M_D and $\frac{1}{d} I_m$ by certain thresholds, such that the error is small.

Definition 3.4.2 (Truncated A^{-1})**.** *We define thresholds $\tau_1 = \tau_2 = O(\log d)$, $\tau_3 = 3$, and $\tau_4 = 1$, and define the truncation of A^{-1} as*

$$T_0 := \sum_{\substack{k_1,k_2,k_3,k_4 \in \mathbb{N} \\ k_i \le \tau_i,\ \forall i}} (-1)^{k_1+\cdots+k_4} \sum_{\substack{(Q_1,\ldots,Q_k) \in \{M_\alpha, M_\beta, M_D, \frac{1}{d} I_m\}^k \\ \text{i-th matrix occurring k_i times}}} Q_1 Q_2 \cdots Q_k \,. \tag{3.11}$$

In the next lemma, we upper bound the truncation error.

Lemma 3.4.3 (Truncation error of A^{-1})**.** *Suppose $m \le cd^2$ for a small enough constant c. Let T_0 be the truncated series defined in Definition 3.4.2 with thresholds $\tau_1 = \tau_2 = O(\log d)$, $\tau_3 = 3$ and $\tau_4 = 1$. Then, with probability $1 - o_d(1)$, the truncation error $E = A^{-1} - T_0$ satisfies $\|E\|_{spec} \le O(\frac{\log d}{d})^2$.*

Proof. From Lemma 3.2.7, we know that $\|M_\alpha\|_{spec}, \|M_\beta\|_{spec} \le 0.1$, $\|M_D\|_{spec} \le O(\sqrt{\log d / d})$, and $\|\frac{1}{d} I_m\|_{spec} = \frac{1}{d}$ with high probability. We can bound the contribution of $\|M_\alpha\|_{spec}$ in the truncation error by

$$\begin{aligned}
&\sum_{i=\tau_1+1}^{\infty} \sum_{j=0}^{\infty} \binom{i+j}{i} \cdot \|M_\alpha\|_{spec}^i \cdot (\|M_\beta\|_{spec} + \|M_D\|_{spec} + \frac{1}{d})^j \\
\le &\sum_{i=\tau_1+1}^{\infty} \sum_{j=0}^{\infty} 2^{i+j} \cdot \|M_\alpha\|_{spec}^i \cdot (0.1 + o(1))^j \\
\le &(2\|M_\alpha\|_{spec})^{\tau_1+1} \cdot O(1) \,.
\end{aligned}$$

Similarly for M_β, M_D and $\frac{1}{d} I_m$. Therefore, we can bound the total truncation error:

$$\begin{aligned}
\|E\|_{spec} &\le ((2\|M_\alpha\|_{spec})^{\tau_1+1} + (2\|M_\beta\|_{spec})^{\tau_2+1} + (2\|M_D\|_{spec})^{\tau_3+1} + (2/d)^{\tau_4+1}) \cdot O(1) \\
&\le O(\frac{\log d}{d})^2
\end{aligned}$$

by our choice of thresholds $\tau_1, \tau_2, \tau_3, \tau_4$. □

3.4.2 Decomposition of $\mathcal{R}$ via truncated A^{-1}

We shift our attention back to $\mathcal{R} = \sum_{i=1}^{m} w_i v_i v_i^T$. Using the expansion of $M^{-1}\eta$ in Proposition 3.4.1 (Equation (3.10)) and the truncation of A^{-1}, we decompose $\mathcal{R}$ as follows.

Proposition 3.4.4 (Decomposition of $\mathcal{R}$)**.** *Suppose $m \le cd^2$ for a small enough constant c. Let T_0 be the truncated series of A^{-1} defined in Definition 3.4.2. Then,*

$$\mathcal{R} = \mathcal{R}_1 + \mathcal{R}_2 + E_{\mathcal{R}}\,,$$

where

$$\mathcal{R}_1 := \frac{r+s}{s^2-ru} \sum_{i\in[m]} (T_0\eta)[i] \cdot v_i v_i^T\,,$$
$$\mathcal{R}_2 := \frac{-u-s}{s^2-ru} \sum_{i\in[m]} (T_0 1_m)[i] \cdot v_i v_i^T\,,$$

and $\|E_{\mathcal{R}}\|_{spec} \le o(1)$ with probability $1 - o_d(1)$.

We first state the following claim which is needed for the error analysis.

Claim 3.4.5. *With probability $1 - e^{-d}$, $\|\sum_{i=1}^{m} v_i v_i^T\|_{spec} \le (1 + o_d(1))\frac{m}{d}$.*

Proof. We can write $\sum_{i=1}^{m} v_i v_i^T = VV^T$ where $V \in R^{d\times m}$ has i.i.d. $\mathcal{N}(0, \frac{1}{d})$ entries, and note that $\|\sum_{i=1}^{m} v_i v_i^T\|_{spec} = \sigma_{\max}(V)^2$. Since we assume $m \ge \omega(d)$, by standard concentration of the largest singular value of rectangular Gaussian matrices **[?]**, we have that $\sigma_{\max}(V) \le (1 + o_d(1))\sqrt{\frac{m}{d}}$ with probability at least $1 - e^{-d}$. □

Proof of Proposition 3.4.4. Recall that $\mathcal{R} = \sum_{i\in[m]} (M^{-1}\eta)[i] \cdot v_i v_i^T$. Unpacking the expression of $M^{-1}\eta$ in Proposition 3.4.1 and plugging in $A^{-1} = T_0 + E$, we have $\mathcal{R} = \mathcal{R}_1 + \mathcal{R}_2 +$

$E_{\mathcal{R}}$ where

$$E_{\mathcal{R}} = \sum_{i\in[m]} \delta_i v_i v_i^T\,, \quad \delta_i := \frac{r+s}{s^2-ru}\langle E[i],\eta\rangle - \frac{u+s}{s^2-ru}\langle E[i],1_m\rangle\,.$$

We first upper bound $|\delta_i|$ for all $i \in [m]$. First, observe that $|\langle E[i],\eta\rangle| \le \|E\|_{spec}\cdot\|\eta\|_2$ and $|\langle E[i],1_m\rangle| \le \|E\|_{spec}\cdot\|1_m\|_2$. By Claim 3.2.9 and lemma 3.4.3, we have $\|E\|_{spec} \le O(\frac{\log d}{d})^2$ and $\|\eta\|_2 \le O(\sqrt{\frac{m}{d}})$ with high probability. Moreover, by Lemma 3.2.10, we have that $r = \Theta(\frac{m}{d})$, $|s| \le O(\sqrt{d})$ and $-1 \le u \le -1/2$, hence $s^2 - ru \ge \Omega(\frac{m}{d})$. Thus,

$$|\delta_i| \le O(\frac{\log^2 d}{\sqrt{md}})$$

as we assume that $m \le O(d^2)$.

Finally, Claim 3.4.5 states that $\|\sum_{i=1}^m v_i v_i^T\|_{spec} \le O(\frac{m}{d})$. Since

$$-\sum_{i\in[m]} |\delta_i| v_i v_i^T \preceq E_{\mathcal{R}} \preceq \sum_{i\in[m]} |\delta_i| v_i v_i^T\,,$$

we may conclude that $\|E_{\mathcal{R}}\|_{spec} \le O(\frac{\log^2 d}{\sqrt{md}})\cdot O(\frac{m}{d}) \le O(\frac{\log^2 d}{\sqrt{d}})$ as $m \le O(d^2)$. □

3.4.3 Overview: each term is a dangling path of injective gadgets

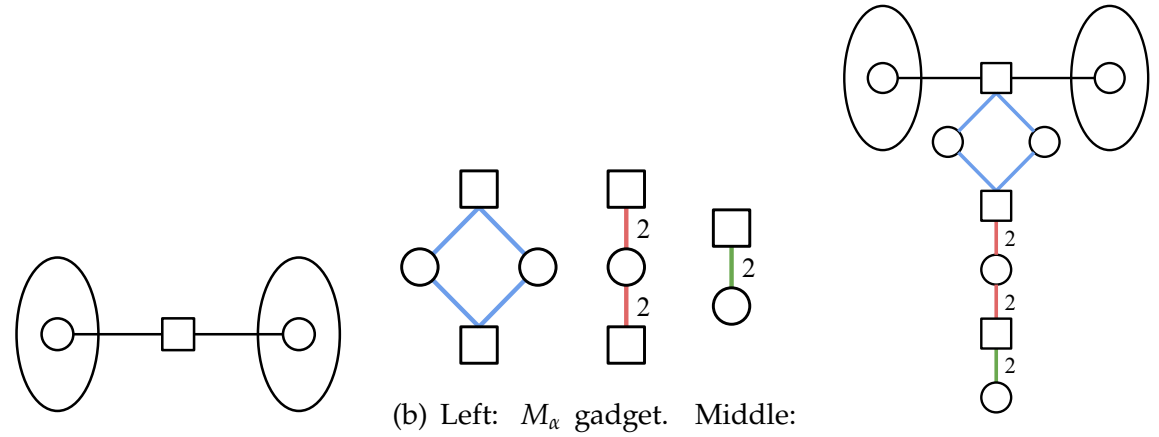
By writing A^{-1} as a truncated series (Definition 3.4.2) and the decomposition of $\mathcal{R}$ (Proposition 3.4.4), we may view $\mathcal{R}_1$ and $\mathcal{R}_2$ as linear combinations of $d \times d$ matrices of the forms

$$\sum_{i\in[m]} (Q_1 Q_2 \cdots Q_k \eta)[i]\cdot v_i v_i^T \quad \text{and} \quad \sum_{i\in[m]} (Q_1 Q_2 \cdots Q_k 1_m)[i]\cdot v_i v_i^T \tag{3.12}$$

respectively, where $Q_1,\ldots,Q_k \in \{M_\alpha, M_\beta, M_D, \frac{1}{d}I_m\}$ are $m \times m$ matrices.

Using the machinery of graph matrices, we can systematically represent these matrices

as shapes (with underlying input $\{v_1, \ldots, v_m\} \subseteq \mathbb{R}^d$). For starters, the off-diagonal part of the matrix $\sum_{i=1}^m v_i v_i^T$ is represented by the shape in Figure 3.2a; it serves as the "base" for other matrices in Equation (3.12). Each matrix in Equation (3.12) can be represented by attaching "gadgets" (Figure 3.2b) to the square vertex in the middle. For the case of $\mathcal{R}_1$, since $\eta_j = \|v_j\|_2^2 - spec1 = \sum_{a=1}^d h_2(v_j[a])$ for $j \in [m]$ (see Equation (3.2)), each shape has an extra h_2 attached at the end. Figure 3.2c shows an example of such a shape.



(a) Off-diagonal part of $\sum_{i=1}^m v_i v_i^T$.

(b) Left: M_α gadget. Middle: M_β gadget. Right: final h_2 gadget.

(c) Off-diagonal part of $\sum_{i=1}^m (M_\alpha M_\beta \eta)[i] \cdot v_i v_i^T$.

Figure 3.2: Examples of graph matrices in the decomposition of $\mathcal{R}$. Recall that square vertices take labels in $[m]$ and circle vertices take labels in $[d]$. Unlabeled edges have Hermite index 1.

In the following, we formally define such shapes which we call *dangling* shapes.

Definition 3.4.6 (Injective gadgets)**.** *We call each shape in* $\{M_\alpha, M_\beta, M_D, \frac{1}{d} I_m\}$ *an* injective gadget *in the dangling shape.*

Definition 3.4.7 (Dangling shapes)**.** *We further separate the matrices in the decomposition of* $\mathcal{R}$ *(Equation* (3.12)*) into diagonal and off-diagonal terms:*

1. *For the off-diagonal terms, we start with a path from* U *to* V *(each containing a circle vertex receiving labels in* $[d]$*) passing through a middle square vertex that receives a label in* $[m]$*.*

2. *For diagonal terms, we have a double edge (that shall be distinguished from an h_2 edge as the double edge here stands for $v_i[a]^2$) connected to a middle square vertex that receives a label in $[m]$.*

For each term, we specify a length-$k \geq 0$ dangling path that starts from the middle square vertex such that

- *for any $k > 0$, each step comes from one of the following gadgets in $\{M_\alpha, M_\beta, M_D, \frac{1}{d} I_m\}$;*
- *The dangling path is not necessarily injective, that we may have each vertex appearing at multiple locations along the path. However, since it is a walk along the above gadgets, the path is* locally injective *within each gadget.*

For matrices from $\mathcal{R}_1$, there is an additional η factor. Thus, we attach an h_2 gadget (Figure 3.2b) to the end of the dangling path. We call this the "final h_2 gadget".

Definition 3.4.8 (Gadget incursion)**.** *Let A, B be two gadgets along the dangling path. We call it a gadget-incursion if there are unnecessary vertex intersections between $V(A)$ and $V(B)$ beyond the "necessary intersection": when A, B are adjacent, $V_A = U_B$ is the necessary boundary intersection, any intersection in $V(A) \setminus V_A$ and $V(B) \setminus U_B$ is a gadget incursion; similarly, when A, B are not adjacent, any intersection in $V(A)$ and $V(B)$ is a gadget incursion.*

Vertex and edge appearance Recall that we bound the norm of a graph matrix by analyzing length-$2q$ "block-walks" of the shape and bounding the vertex/edge factor of each "block-step". To this end, we need to consider both *global* and *local* appearances of a labeled vertex. We remind the reader that a labeled square (resp. circle) vertex is an element in $[m]$ (resp. $[d]$), and a labeled edge is an element in $[m] \times [d]$.

Definition 3.4.9 (Local versus global appearance)**.** *Given a block-walk, we call each labeled vertex's appearance within the given block a local appearance, and each vertex's appearance throughout the walk a global appearance.*

Moreover, we say that a labeled vertex/edge is making its global *first/last appearance if it is the first/last appearance of that labeled vertex throughout the walk. Similarly, we say it is making its* local *first/last appearance if it is the first/last appearance within the given block in the walk.*

We also need the following definition, which is a special case that we need to handle. The term "reverse-charging" will be clear once we describe our edge charging scheme in Section 3.4.4.

Definition 3.4.10 (Reverse-charging step/edge)**.** *For a given walk, and a block in the walk, we call a step $u \to v$ reverse-charging if*

1. *the underlying edge is making its last* global *appearance throughout the walk;*
2. *the underlying edge's first* global *appearance is also in the current block;*
3. *(Reverse) the first appearance of the underlying edge goes from v to u.*

Further set-up for step-labeling and edge factor scheme

Our argument for tight matrix norm bounds requires assigning each edge (or step) a step-label that represents whether it is making first/middle/last appearance, and assigning edge factors based on the edge type (recall our edge factor scheme in Section 3.3.2). However, further care is warranted for dangling shapes when an edge appears with different Hermite indices in the walk (e.g., appears both as an h_1 and h_2 edge). In this case, it is no longer true that an h_k edge needs to appear at least twice in the walk for the random variable to be non-vanishing. For example, suppose an edge appears as h_1, h_1, and h_2 in the walk. Then, even though h_2 only appears once, this term is non-vanishing under expectation:

$$\mathbb{E}_{x\sim\mathcal{N}(0,1/d)}(h_1(x)^2h_2(x)) = \mathbb{E}\left[x^2(x^2-\frac{1}{d})\right] = \mathbb{E}\left[x^4\right] - \mathbb{E}\left[\frac{x^2}{d}\right] = \frac{2}{d^2}\,. \tag{3.13}$$

The matrices that arise in our analysis may contain h_1, h_2, h_3, h_4 edges. For $i \leq 4$, we treat an h_i edge as i edge-copies in our edge factor assignment scheme.

Definition 3.4.11 (Step-labeling scheme for mixed edges)**.** *For each step regardless of the Hermite index, assign a step-label to all its edge-copies as follows,*

1. *Assign an F step if it is making its first appearance;*
2. *Assign an H step if it is making its middle appearance*
3. *Assign an R step if it is making its last appearance.*

We next describe our edge factor assignment scheme.

Lemma 3.4.12 (Edge factor assignment scheme)**.** *For any graph matrix of size at most D_V that contains h_1, h_2, h_3, h_4 edges and walks of length $2q \geq \Omega(\log d)$, we can assign values to each edge-copy among the edge's appearance throughout the walk such that*

1. *Each edge-copy of an h_1, h_2 edge with F / R step-label gets assigned a value $\frac{2^{1/4}}{\sqrt{d}}$.*
2. *Each edge-copy of an edge with step-label H and any edge-copy of an h_3, h_4 edge gets assigned a value $\frac{32qD_V}{\sqrt{d}}$.*
3. *In the case of a random variable appearing as h_1, h_1, h_2 (in an arbitrary order), it gets assigned value of $\frac{2}{d^2}$ in total, in particular, each edge-copy of the h_2 edge gets assigned a value $\frac{\sqrt{2}}{\sqrt{d}}$.*

Proof. We first note that $\mathbb{E}_{x\sim\mathcal{N}(0,1/d)}[h_k(x)^2] = \frac{k!}{d^k}$. For $k = 1, 2$, if an h_k edge only appears twice and no other Hermite index occurs, then it must have $2k$ edge-copies with step-labels F and R, giving an edge factor of $(\frac{2^{1/4}}{\sqrt{d}})^{2k} = \frac{2^{k/2}}{d^k}$, which is larger than $\frac{k!}{d^k}$ for $k = 1, 2$.

Next, we consider the case when h_3, h_4 edges are involved or when an edge appears more than twice, i.e., some edge-copies are assigned step-label H. Let a_k be the number

of times h_k appears in the walk, and let $t := \sum_{k\le 4} k \cdot a_k$ be the total number of edge-copies. Applying Cauchy-Schwarz twice and Fact 3.3.4,

$$\left|\mathbb{E}_{x\sim\mathcal{N}(0,1/d)}[\prod_{k\le 4} h_k(x)^{a_k}]\right| \le (4t/d)^{t/2}.$$

We next show that the edge factors assigned to the t edge-copies upper bound the above. Let t_0 be the number of edge-copies that get assigned $\frac{2^{1/4}}{\sqrt{d}}$. We must have $0 \le t_0 < t$ and $t_0 \le 4$. Then, the assignment scheme gives

$$(\frac{2^{1/4}}{\sqrt{d}})^{t_0} \cdot (\frac{32qD_V}{\sqrt{d}})^{t-t_0} \ge d^{-t/2}(32qD_V)^{t-t_0}.$$

Since the length of the walk is $2q$ and the size of the graph matrix (shape) is $\le D_V$, we have $t \le 8qD_V$. Thus, if $t \le 8$, then clearly $(32qD_V)^{t-t_0} \ge (4t)^{t/2}$; otherwise, $t - t_0 \ge t/2$ and $(32qD_V)^{t-t_0} \ge (4t)^{t/2}$. This shows that the edge factors correctly account for the values from the Hermite characters.

For the special case when an edge appears as h_1, h_1, h_2, the factor $\frac{2}{d^2}$ follows from Equation (3.13). This completes the proof. □

3.4.4 Local Analysis for $\mathcal{R}$

Lemma 3.4.13. *For some constant $\epsilon > 0$, for any $q < d^\epsilon$, the block-value function for R is bounded by*

$$B_q(\mathcal{R}) \le \frac{1}{10}.$$

We first state our vertex factor assignment scheme (with redistribution) which assigns vertex factors to labeled vertices according to their global appearances.

Vertex factor assignment scheme

1. For each labeled vertex i making its first or last *global* appearance, assign a factor of $\sqrt{\mathsf{weight}(i)}$;

2. For each labeled vertex i making its middle *global* appearance, assign a factor of 1 if it is reached via an R step, otherwise assign a factor $2q \cdot D_V$ as well as its corresponding $O(1)$ Pur factors.

We next describe the scheme that assigns edge-copies to vertices. In most cases, we charge edge-copies (on the dangling path) to the vertex it leads to, unless it is a *reverse-charging* edge (Definition 3.4.10). Recall that we define a step $u \to v$ to be reverse-charging if it is making its last global appearance and if its first global appearance is also in the current block going from $v \to u$.

"Top-down" edge-copy charging primitive

1. Assign both edges on the $U - V$ path to the first square vertex in the middle;

2. For any step $u \to v$ before the final h_2 gadget, assign the edge to v unless this is a reverse-charging edge (Definition 3.4.10), in which case we assign to u;

3. For the final h_2 gadget (if any), we reserve its assignment from the current scheme.

See Section 3.4.5 for an illustration of the above scheme. We next show the following invariant throughout the walk,

Proposition 3.4.14. *We can assign each edge-copy to at most one vertex,*

1. *for a circle vertex, it is assigned* 1 *edge-copy if it is making first/last* global *appearance, and* 2 *edge-copies if both;*

2. *for a square vertex on the dangling path, it is assigned* 2 *edge-copies if it is making first/last* global *appearance, and* 4 *edge-copies if both;*

3. *for any square vertex's global middle appearance yet local first appearance in the block, it is assigned at least* 1 *edge-copy;*

4. *no surprise/high-mul step is assigned to any vertex's* global *first/last appearance.*

Proof. We start by giving the argument when $U \neq V$, and then show it can be modified into an argument for $U = V$.

Charging vertices outside $U \cup V$ and the final h_2 gadget. The above scheme applies immediately to vertices that appear in the current block while not in $U \cup V$ or the last square vertex as well as the final circle vertex it connects to in the h_2 gadget.

It follows by observing that any circle (resp. square) vertex that makes its first *global* appearance is the destination of 1 (resp. 2) edge-copies that are not reverse-charging, in which case the edge-copies are assigned to the particular vertex. This holds analogously for vertices making the last *global* appearance but not the first *global* appearance in the block, since the edges are not reverse-charging. We note that this is true for the first square vertex as well since we assign both edges on the $U - V$ path to the first square vertex.

On the other hand, for vertices making both the first and the last *global* appearances, consider the assignment until the final h_2 gadget, the charging above gives us 1 and 2 edges for each such vertex when it first appears *locally*. Furthermore, notice that each of these edges need to be closed, and they are assigned to the destination vertex, i.e., the particular vertex in inspection, and this completes the proof of our assignment restricted to vertices outside $U \cup V$ and the final h_2 gadget.

Charging the final gadget and finding "block-reserve". The goal is to identify an edge-copy as block-reserve, and assign vertex factors to corresponding edges in the final h_2 gadget (if any). We first consider the case when there is no final h_2 attachment, i.e. the term from $\frac{u+s}{s^2-ru}A^{-1}1_m$. Recalling our bounds from Lemma 3.2.10 that $s^2 - ru = \Omega(\frac{m}{d})$,

and $s = O(\sqrt{d}), |u| = O(1)$ (here the weaker bound on s suffices), we observe that

$$|\frac{u+s}{s^2-ru}| = O(\frac{\sqrt{d}}{m/d}) = O(\frac{1}{\sqrt{d}}),$$

and the normalizing constant can be regarded as an edge-copy. This is assigned as the block-reserve factor if there is no final h_2 gadget, and there is no vertex factor involved when there is no final h_2 gadget since this may be the last appearance of a square vertex while its factor has already been assigned to the edges that first lead to this vertex in the current block.

We now proceed to assign edges from the final h_2 gadget to vertex factors involved in the final h_2 gadget, moreover, we will identify one edge-copy of assigned factor at most $\frac{\sqrt{2}}{\sqrt{d}}$ as "block-reserve" that allows us to further charge vertices in $U \cup V$.

For the vertices in the final h_2 gadget, we observe the following,

1. in the final gadget, the last square vertex cannot be making its first *global* appearance yet it may be making its last appearance (since it is on the gadget boundary, we consider it appears in both the final h_2 attachment gadget and the gadget that precedes it);

2. the last circle vertex may be making either its first or last *global* appearance but it cannot be both;

3. the h_2 edge in-between has not yet been assigned, and we can split it as two edge-copies, each of weight $\frac{2^{1/4}}{\sqrt{d}}$ if F/R under the *global* step-labeling.

We now observe the following,

1. If the final h_2 attachment is not a reverse-charging edge assigned to the source square vertex, the square vertex has also been assigned by 2 factors if making its first/last *global* appearance, and 4 if both, in the current block by the previous charging. In

this case, note that we can reserve one of the two edge-copies, that is a factor of at most $\frac{2^{1/4}}{\sqrt{d}}$,

- If the circle vertex is making first/last appearance, we either have both edge-copies receiving F/R labeling, or a mix of H, R labeling. In the first case, we have two assigned factors of $\frac{2^{1/4}}{\sqrt{d}}$. Assign one to the circle vertex's first/last appearance, and another to the block-reserve. In the second case, we have a mix of H, R edges, that we at least have the underlying random variable appears twice as h_1 edges and at least once as an h_2 edge, by moving $\tilde{O}(1)$ polylog factors to the second appearance of the underlying random variable locally, we have again two edge copies of value $\frac{2^{1/4}}{\sqrt{d}}$, and the assignment follows from above.
- If the circle vertex is making a middle appearance, we may assign related Pur factors and vertex-factor cost to one edge-copy such that one edge-copy is of value $\tilde{O}(\frac{1}{\sqrt{d}})$ while the other is still at most $\frac{2^{1/4}}{\sqrt{d}}$; assign the edge-copy with $\tilde{O}(\frac{1}{\sqrt{d}})$ to the middle appearance factor and assign the edge-copy with factor $\frac{2^{1/4}}{\sqrt{d}}$ for block-reserve.

2. If the final attachment h_2 edge is a reverse-charging step that corresponds to an edge whose F copy leads to the final square vertex, we assign both h_2 edges to the final square vertex as the square vertex may be making its last *global* appearance. Note that if the square vertex makes its first *global* appearance in the current block, it has already been assigned 2 edges.

   This leaves the last circle vertex potentially uncharged as it may be making its last appearance as well. Furthermore, we need to find one more "reserve" edge-copy for the block which would then be used to charge $U \cup V$.

3. **Finding critical edge:** we now give a procedure to identify the critical edge, that is, an h_2 edge assigned to a circle vertex in the above top-down charging process. The

process maintains a current circle vertex and its current gadget along the top-down path. In particular, the gadget considered is an M_β gadget throughout. The process starts from vertex s, that is the circle vertex involved in the final h_2 attachment, and the gadget considered is the one in which s opens up the F step of the current edge e^*, the h_2 edge in the final gadget. Note that this is an M_β gadget. We now case on the step-labeling of the top half of the M_β gadget in inspection, in particular, the edges leading to the circle vertex s in that particular gadget,

- This is an F, non-reverse-charging R, or H step assigned to s, this is the critical edge we aim to find, as we have two edge-copies assigned to a circle vertex, and the process terminates.
- This is a reverse-charging R edge: update the circle vertex to be the top vertex of the current M_β gadget s, and update the edge e^* to be the h_2 edge in the top half of the current gadget, and repeat the above process. Note that the updated gadget must appear, and additionally, on top of the current gadget in the dangling path.

We now observe that this process ultimately terminates since each time we move up towards the top of the dangling path. Once the critical edge is found, note that it contains two edge-copies, assign one to the vertex's factor, and another to the block-reserve. In the case of H edges involved, locally assign the edge-value as $\frac{1}{\sqrt{d}}$ and $\tilde{O}(\frac{1}{\sqrt{d}})$, and assign the copy with $\frac{1}{\sqrt{d}}$ for block-reserve, while the other for vertex's factor.

Reserving surprise/high-mul step from vertex-factor assignment outside $U \cup V$ We first observe that in the above assignment, it is clear that the edges assigned to vertex's first appearance cannot be surprise-visit nor high-mul step. That said, it suffices for us to

consider the assignment for vertex's last appearance factor (whose first appearance is not in the same block). Consider such vertex's first local appearance, they may either be H/S edges if not R under the global edge-labeling. If so, since the vertex is making the last appearance in the current block, each corresponds to an R-step in this block. Moreover, observe that any such R-step is intended for "reverse-charging" if the vertex in inspection is making both first and last global appearance in the block, and thus it is not assigned to the vertex factor of the source vertex. That said, it suffices for us to swap the H/S step with the R step so that no surprise/high-mul step is assigned to vertex's (polynomial) factor.

At this point, for a block with $U \neq V$, we have assigned

1. 1 or 2 edge for each vertex's first/last appearance in the current block outside $U_\alpha \cup V_\alpha$ depending on the vertex's type;

2. 1 edge-copy of value $O(\frac{1}{\sqrt{d}})$ has been identified as the block-reserve.

3. it is also straightforward to observe that any edge assigned for vertex's first/last appearance cannot be a surprise visit nor high-mul visit.

Charging circle vertices in $U \cup V$

Proposition 3.4.15 (Unbreakable $U - V$ path)**.** *There is always a $U - V$ path such that each edge along the path is of odd multiplicity. Call this path P_{safe}.*

Proof. It suffices for us to restrict our attention to paths of only h_1 edges. By our gadget property, each vertex except $U_\alpha \cup V_\alpha$ is incident to an even number of h_1 edges, while only vertex copy in $U_\alpha \cup V_\alpha$ is of odd degree. Suppose the path is broken, consider the (maximal) component C_U connected to U_α (but not V_α), we first observe that the edge-multiplicity of $E(C_U, V(\alpha) \setminus C_U)$ must be even, as otherwise, some edge is of odd multiplicity, and the

edge is included inside C_U as opposed to be across the cut. That said,

$$\sum_{v \in C_U} \deg(v) = 2 \cdot \mathrm{mul}(E(C_U)) + \mathrm{mul}(E(C_U, V(\alpha) \setminus C_U)) .$$

Observe that the LHS is odd as any vertex copy except U_α has even degree, and we have exactly 1 odd-degree vertex copy. On the other hand, RHS follows as each edge-multiplicity in $E(C_U)$ contributes a factor 2 to degree of vertices in C_U while edges across the cut contribute 1 for each multiplicity; since $E(C_U, V(\alpha) \setminus C_U)$ is even by connectivity argument above, the RHS is also even and we have a contradiction. □

Corollary 3.4.16. *There is at least one vertex v^* making middle-block appearance, i.e. it is a vertex that appears in previous blocks, and is appearing again in future blocks.*

We recall our charging so far that we have one edge-copy reserved, while we have two circle vertices in $U \cup V$ uncharged. And we now show that the single-edge copy is sufficient. i.e., at most one vertex factor is picked up among these two vertices.

Consider the path P_{safe}, and observe that it passes through both vertices in $U \cup V$. Take v^* to be the first vertex on the path that pushes out an $F/H/S$ edge. In the case $v^* = U$, note that it suffices for us to assign the reserve factor for the vertex in V; similarly, assign the reserve factor for U for $v^* = V$. In the case $v^* \notin U \cup V$, note that the previous scheme assigns at least one edge for v^* if it is a circle, and 2 if it is a square when v^* first appears in the current block, and this is in fact not needed since v^* is making a middle appearance. That said, we have at least 2 factors, 1 from the reserve factor, and 1 from the edge-assignment for v^* in the previous scheme, we now assign these two factors for $U \cup V$.

Remark 3.4.17. *v^* is picked such that it is either reached by an R edge, or it is the boundary vertex U_α. In other words, no factor is needed for specifying v^* of the block.*

and this completes the proof to our proposition when $U \neq V$.

Analysis for $U = V$ We now consider the case when $U = V$.

Definition 3.4.18 (SQ_1: first square vertex in R term)**.** *For each R term, call the first square vertex on top of the dangling path the SQ_1 vertex.*

Remark 3.4.19. *This vertex is an exception from any other square vertex as its first appearance might be the destination of two edge-copies that correspond to the same underlying random variable. This happens when an edge making its first and last appearance at the same time for the $U = V$ term.*

For $U = V$, the charging for vertices outside is identical except for the first square vertex SQ_1: since the two edges assigned to it now correspond to the same underlying edge-copy, and may now receive F, R-labeling. That said, both edges are now assigned to the first square vertex. If the square vertex is making either first/last appearance but not both in the current block, the current assignment is sufficient. However, there are no reverse-charging edge copies protecting SQ_1, and that warrants a further analysis.

We follow the previous strategy in identifying block-reserve: even though for $U = V$, the vertex $U = V$ is by definition not contributing any vertex-appearance factor, we now intend the block-reserve factor to pay for either additional Pur factor due to the dormant step stemming from $U = V$, or the last appearance factor of SQ_1 but not both.

1. SQ_1 makes both first and last appearance in the current block: in this case, the edges assigned to SQ_1 in the beginning of the top-down charging process are assigned to the first appearance of SQ_1. That said, it remains for us to identify two extra edge-copies for the last appearance factor.

   Charging for terms without final h_2 attachment: these terms come with a normalization of $|\frac{u+s}{s^2-ru}| = O(\frac{1}{d})$, and these are two edge copies that we assign to the last

appearance of SQ_1.

Charging for terms with final h_2 attachment: we first note that any edge incident to SQ_1 must be closed, and further case in whether there is any surprise visit arriving at SQ_1 throughout the dangling path.

(a) Suppose that there is no surprise visit arriving at SQ_1 throughout the dangling path.

Consider the final departure from SQ_1, and note that it pushes out at least two edge-copies.

- If this is an M_β gadget, let r be the circle vertex it connects to via an h_2 edge. Suppose the random variable corresponding to (SQ_1, r) first appears as an h_2 edge, in this case, the circle vertex has been assigned two edge-copies the first time it appears as it is reached using h_2 edge, that said, we can reserve the other two edge copies from the final attachment for the missing last appearance factor of SQ_1;
- If this is an M_β gadget, yet the random variable (SQ_1, r) first appears as an h_1 edge in the current block. Note that there is at least one more edge-copy of h_1 edge that is currently assigned an H-label, observe that we can replace its label to be R, and assign both F/R copies to the vertex factor of the circle vertices. We now relabel the final h_2 step to be H^*, and note that these two edges get assigned a value at most $\frac{2}{d}$, and assign them both to the missing last appearance factor of SQ_1.
- To see that H^* does not require extra $D_V \cdot q$ factor, we observe that this is a circle vertex traversed before in the block, and for each block, H^* is unique. That said, it suffices for us to use a special label in $[2]$ when H^* first appears in the block to identify the edge. Moreover, note that swapping H^* with

the R step does not add to Pur factor of the destination circle vertex as this edge no longer appears;

- If this is an M_α gadget, let r_1, r_2 be the circle vertices the square vertex is connected to. Note that for each r_i, there must be at least two more edge-copies of random variable of (SQ_1, r_i) that receive H label, i.e., we have a factor of $\tilde{O}(\frac{1}{d})$. That said, we may reassign the factors and extract one full factor of $\frac{1}{\sqrt{d}}$ (with the remaining being $\tilde{O}(\frac{1}{\sqrt{d}})$, and since we have two of them that combine to a factor of $\frac{1}{d}$, assign it to the missing last appearance factor of SQ_1.

(b) There is a surprise visit arriving at SQ_1. We first observe this must be either an h_2 edge, or a pair of distinct h_1 edges. The surprise visit must be closed already, and their underlying R copies are intended for reverse-charging in the top-down charging scheme for the last appearance of SQ_1. In this case, the factor for the last appearance has already been assigned edges.

2. **Finding "block-reserve" for $U = V$:** SQ_1 is not making both first and last appearance in the current block, we first note that the departure from $U \cup V$ is special as this is the only vertex that can appear on the boundary again without being reached by any edge, and this is not a dormant gadget M_D. That said, the most recent departure may contribute Pur factor to the vertex in $U \cup V$, and we now identify an edge-copy for this factor.

   (a) **Charging for terms without final h_2 attachment:** for $A^{-1}1_m$ term where the final h_2 gadget is missing, we pick up a normalization constant of $O(\frac{1}{d})$ and we designate this as the block-reserve.

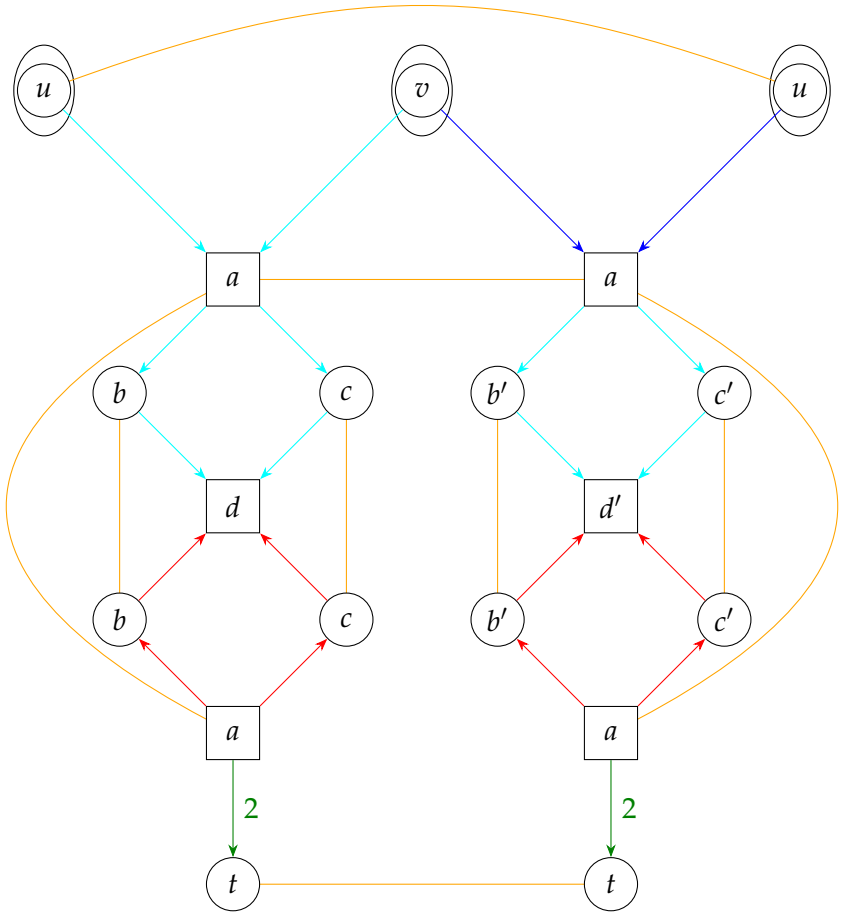
   (b) **Charging for terms with final h_2 attachment:** for terms with the final h_2 gadget, the analysis of finding block-reserve from the case $U \neq V$ applies identically for

finding an h_2 edge that gets assigned to a circle vertex in the top-down traversal process. In particular, we may designate one edge-copy among the two copies of the identified h_2 edge as a block-reserve to charge the corresponding $\tilde{O}(1)$ factors from Pur.

Remark 3.4.20. *Note that the edge identified from the above process is in fact stronger, as it carries a factor of* $O(\frac{1}{\sqrt{d}})$ *as opposed to* $\tilde{O}(\frac{1}{\sqrt{d}})$. *That said, since for* $U \neq V$ *terms, the block-reserve is only assigned for* $\tilde{O}(1)$ *factors from* Pur, *either is sufficient.*

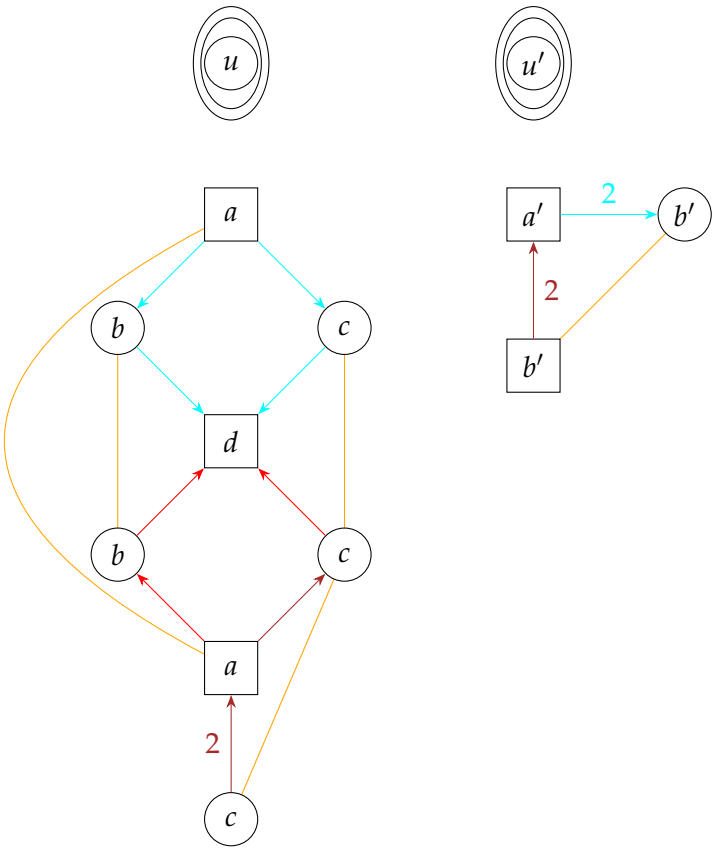
□

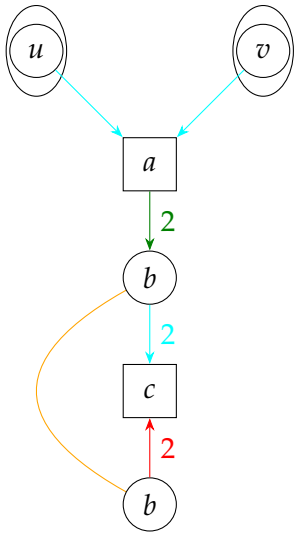
3.4.5 Illustration via diagrams



This figure illustrates the charging scheme. The light blue edges are F edges and we assign their factor to their destination. The dark blue edges are R edges whose first appearance is in a different block. We also assign these edges to their destination. The red edges are reverse-charging R edges whose first appearance is in the current block. We assign these edges to the destination of the corresponding F edge (which is generally the source for this edge).

Note that two edge factors are missing from v. We obtain these factors from the two green edges pointing towards t.





3.4.6 Pur bound for square vertices in $\mathcal{R}$

The prior Pur bound does not apply well for square vertices in $\mathcal{R}$, in particular, it should be pointed out that even before non-trivial intersection within each block along the dangling path, the Pur argument based on 1-in-1-out (or its slightly generalized version of 2-in-2-out) falls apart in the analysis for $\mathcal{R}$.

In particular, one may consider a walk on R_S with the square vertex along the $U - V$ path fixed throughout the walk. It is easy to verify the the fixed square vertex may have growing unclosed F edges without any surprise visit/high-mul step. That said, it is not sufficient to use the slack from such factors to offset the potential confusion due to R edges.

In this section, we give a new argument to handle the Pur factor in the vanilla setting of R when there is no non-trivial intersection along the dangling gadget-path, and then extend it for the general cases of R where each block is not necessarily injective due to intersection across gadgets.

Gap from square middle appearance For starters, we observe that a vertex may only push out unforced returns if it is making a middle appearance given that it maintains

a list of incident edges, including the additional information which are closed. When it is making the last appearance, any currently unclosed edge needs to be closed, and therefore shall be pushed out, giving us a fixed set of edges being pushed-out (where we momentarily ignore the question whether one needs to distinguish among the edges in the edge-set). This immediately renders us the following bound on Pur readily,

Lemma 3.4.21 (Pur factor via middle appearance: beginner version)**.** *For any vertex v, let* $\mathsf{MidApp}(v)$ *be the number of middle appearances of v throughout the walk, we have*

$$\mathsf{Pur}(v) \leq 3 \cdot \mathsf{MidApp}(v)$$

provided each square vertex pushes out at most 3 *edges in each block throughout the walk, and each block is vertex-injective.*

With the above Pur bound, it may not be meaningful if we cannot obtain a slack from MidApp. Fortunately, this is indeed the case for our setting, and we first observe this in the vanilla setting where there is no gadget-incursion within each block, (in particular, this already applies immediately if the edges are injective witin each block),

1. We have assigned each square vertex at least 1 edge when it is making a middle appearance: to see this, the top-down charging scheme assigns each square vertex 2 edges when the square-vertex appears for the first time in the block; in the case of charging $U \cup V$, 1 edge may be re-routed from a square vertex making a *global* middle appearance. That said, at least 1 edge is assigned to each square vertex when it makes a middle appearance.

2. Note that for a fixed vertex v in a given block, its *local* first appearance at the given block may be corresponding to a *global* middle appearance, and such *mismatch* is the source of our slack;

3. Observe that each middle-appearance corresponds to a mismatch described above (provided each block is injective), and each such mismatch of local-global first appearance assigns a vertex making global middle appearance one edge-copy, that is a factor of $O(\frac{1}{\sqrt{d}})$;

4. However, since each vertex's middle appearance does not get assigned any vertex factor in our scheme, we may use the $\frac{1}{\sqrt{d}}$ gap to offset the 3 Pur factors corresponding to the particular *global* middle appearance, that is

$$O(\frac{1}{\sqrt{d}}) \cdot (q \cdot D_V)^3 = o_d(1)\,.$$

Extension to gadget intersections To handle dangling paths with potentially intersecting gadgets, it should be pointed out the Pur bound goes through as stated while we do not necessarily have a gap from middle-appearance. In particular, it is possible now that in a given block, a square vertex appears for various times and only has 2 edges assigned to it for its first local appearance, as any of its subsequent appearance in the given block follow via closing some F edge that gets opened up earlier in the work, and thus assigned to the destination as opposed to the given square vertex.

Towards generalizing the prior argument, we consider a specific subclass of *global* middle appearance of a vertex through the block walk,

Definition 3.4.22 (Middle appearance for a square vertex)**.** *For a given walk and a given block-step* $\mathsf{BlockStep}_i$*, we say a labeled vertex* $v \in [m]$ *makes a middle appearance in* $\mathsf{BlockStep}_i$ *if*

1. *v makes appearance at* $\mathsf{BlockStep}_i$*;*

2. *v makes appearance at some* $\mathsf{BlockStep}_j$ *for $j < i$, and at some* $\mathsf{BlockStep}_{j'}$ *for $j' > i$.*

With some abuse of notation, we continue to let $\mathsf{MidApp}(v)$ denote the number of middle appearances of v. Note that this is clear when we are working with blocks that

do not have block-injectivity, while it is equivalent to the previous definition in vertex-injective blocks.

In particular, we emphasize that following the above definition, in the case of a labeled vertex first appears at $\mathsf{BlockStep}_i$, and appears multiple times in various gadgets in the dangling-path of $\mathsf{BlockStep}_i$, it is not considered as making a middle appearance at $\mathsf{BlockStep}_i$.

Lemma 3.4.23 (Pur bound for square vertices)**.** *For any square vertex v that does not push out dormant gadgets, at any time-t,*

$$\mathsf{Pur}_t(v) \leq P_t(v) \leq 3 \cdot \mathsf{MidApp}_t(v) + 3(s_t(v) + h_t(v))$$

where we define

$$P_t(v) := \#(R \text{ steps closed from } v \text{ by time } t) + \#((\text{unclosed edges incident to } v \text{ at } t\ )) - 4\,.$$

In other words, by assuming at 4 possible return legs to be fixed each time a vertex is on the boundary, the number of unforced returns from a vertex by time t is at most $3 \cdot \mathsf{MidApp}_t(v) + 3(s_t(v) + h_t(v))$.

Proof. The first inequality is definitional as we assume each vertex may have 4 edges being fixed, which incurs a cost of $[4]$ for each vertex each time it pushes out an R step. Analogous to previous Pur bounds, the base case is immediate when vertex first appears in the walk. In particular, we note that the above bound can be strengthened for the $\mathsf{BlockStep}_i$ in which vertex v makes (globally) its first appearance.

Claim 3.4.24. *For any square vertex v, for the $\mathsf{BlockStep}_i$, at any time-t within the $\mathsf{BlockStep}_i$,*

$$P_t(v) \leq 3(s_t(v) + h_t(v)) - 2\,.$$

Proof. Notice this is immediate when vertex v first appears, as it is incident to 2 edges, and thus we have $P_t^{(1)} = -2$ (as we have a -2 term since we maintain 3 edges to be fixed instead of just 2). The invariant holds as any subsequent departure opens up at most 2 F edges, and any subsequent arrival either closes both edges, or either arrival is along a surprise/high-mul visit, and give a net-gain of at most 3 in the number of unclosed F edges. This proves our claim. □

It remains for us to consider the appearance of v in subsequent blocks, in particular, we start with the *locally* first appearance of the subsequent block. Let t be the time-mark in which v is making its first *local* appearance in the current-block. Notice at time t when the vertex first appears, it may be arrived via a single edge (as opposed to 2 due to the $U - V$ path), and therefore it may push out at most 3 edges.

Appearance at the second block If not, this is currently the second block in which v appears: applying the claim on the first block, and observe that the most recent departure opens up at most 2 F edges, while the current arrival closes 1 (unless H or S, in which case a net-gain of 3 suffices), we have

$$P_t(v) \leq 3(s_t(v) + h_t(v)) - 2 + 2 - 1 = 3(s_t(v) + h_t(v)) - 1$$

where the $+2$ corresponds to the 2 F edges opened up at the most recent departure, and the -2 term corresponds to the term in the hypothesis on first-block, and the -1 comes from the current arrival closing at least one R edge.

For any subsequent appearance of v in the current block, if any, the following is immediate,

$$P_{t'}(v) - P_t(v) \leq 3(s_{t'}(v) - s_t(v)) + 3(h_{t'}(v) - h_t(v)) + 1$$

as we observe that

1. The departure from the first vertex may open up at most 3 edges instead of 2;
2. Any subsequent departure and arrival closes 2 edges, and opens up at most 2 edges, hence the previous argument applies, giving a net-gain of $+1$ due to the extra opening in the first departure.

That said, for any appearance of v in the second block at time t', we have

$$P_{t'}(v) = P_{t(v)} + (P_{t'}(v) - P_t(v)) \leq 3(s_{t'}(v) + h_{t'}(v))$$

Appearance at the future blocks For any block, let t_0 be the local first appearance, and t_1 be the local final appearance, applying the above argument gives

$$P_{t_1}(v) - P_{t_0}(v) \leq 3(s_{t_1}(v) - s_{t_0}(v)) + 3(h_{t_1}(v) - h_{t_0}(v)) + 1\,.$$

That said, it suffices for us to bound $P_{t_0}(v)$. This is bounded by

$$P_t(v) \leq 3 \cdot \mathsf{MidApp}_t(v) + 3(s_t(v) + h_t(v))$$

Consider the base case when v appears at the third block, the most recent departure opens up at most 2 new F edges. To offset this, we use the gain in $\mathsf{MidApp}_t(v)$, as the appearance of v in the second block is now counted as a middle appearance once v appears in the third block, which gives a $+3$ on the RHS. The bound extends to any subsequent block immediately. This completes our proof of Pur bound.

Extension to dormant gadgets To capture the Pur change due to dormant gadget, we note that for each square vertex at each block, we can assume one dormant edge it pushes out being fixed while assign a Pur factor for any other dormant gadget it pushes out at that block. Notice this is a cost at most 2 for each square vertex throughout the walk, as

we may need to assume 1 dormant edge for its first appearance, and another for its last appearance. For any middle appearance, it suffices for us to assign a Pur factor for any dormant edge it pushes out, as opposed to all-but-one in the case of first/last appearance. This prompts to define the following counter,

Definition 3.4.25 (Dormant-excess)**.** *For each square vertex v, at any block* $\mathsf{BlockStep}_i$*, let D_i be the number of dormant-gadgets it pushes out at this block, define $D_i(v) := D_i - 1[\textit{first/last appearance at } i]$, and additionally, the counter function is defined for the whole walk by taking*

$$D(v) = \sum_{i:v \textit{ appears in } \mathsf{BlockStep}_i} D_i(v)\,.$$

Remark 3.4.26. *By using an extra additive constant of* 2*, each return using dormant leg is either forced or accounted for in $D(v)$.*

Claim 3.4.27. *Each dormant-excess gadget corresponds to a circle vertex reached using an h_2 edge, and gets assigned a factor of at most $\tilde{O}(\frac{1}{\sqrt{d}})$ in the combinatorial charging argument.*

Proof. Note that in the combinatorial charging argument, we have assigned both h_2 edges to the circle vertex's vertex factor, unless the square vertex is potentially making its first and last appearance at the same block, which is not ruled out by the definition of dormant excess. That said, for the circle vertex, it gets assigned at most a factor of $\sqrt{d}$ in our scheme, while both edges assigned at least $\tilde{O}(\frac{1}{\sqrt{d}})$ each, and combining the above yields the desired. □

Corollary 3.4.28. *By assuming at most* 6 *return legs to be forced, the number of unforced-return from v throughout the walk is at most*

$$\mathsf{Pur}(v) \le 3 \cdot \mathsf{MidApp}_t(v) + 3(s_t(v) + h_t(v)) + D(v)\,.$$

This allows us to effectively ignore Pur factor in the block-value analysis, as each Pur factor can be distributed among surprise visit, high-mul visit, or middle appearance such that each is assigned at most 3 Pur factors. Moreover, since each comes with a $O(\frac{1}{\sqrt{d}})$ factor, combining with the assigned Pur factor contributes an $o_d(1)$ term provided $\frac{q^3}{\sqrt{d}} = o_d(1)$, which is sufficient for us as we set $q = d^\epsilon$ for a small enough constant ϵ. □

3.4.7 Wrapping up

Given the edge-assignment scheme to vertex appearance, we now show why this immediately gives the desired bound on $B(\mathcal{R})$: we have the following factors,

1. Each circle vertex gives a factor of $\sqrt{d}$ for their first/last appearance; the assignment gives each vertex one *global* F/R edge-copy each, that is a factor of $\frac{\sqrt{2}}{\sqrt{d}}$, giving a bound of
$$\sqrt{d} \cdot \frac{\sqrt{2}}{\sqrt{d}} = \sqrt{2}\,;$$

2. Each square vertex gives a factor of $\sqrt{m}$ for their first/last appearance, and the assignment above gives *global* F/R each two edge-copies each, that is a factor of
$$\sqrt{m} \cdot \frac{2}{d} \leq \frac{2\sqrt{m}}{d}\,;$$

3. Each square vertex that makes a *global* middle appearance gives a $O(1)$ factors of Pur, while each such vertex appearance is assigned one edge, that is a factor of
$$(2q \cdot D_V)^{O(1)} \cdot O\left(\frac{\sqrt{2}q^2}{\sqrt{d}}\right) \ll 1\,.$$

Corollary 3.4.29. *For each vertex's appearance and edges assigned to it, we have a factor of at most*

$$(1+o_d(1))\cdot\sqrt{d}\cdot\frac{\sqrt{2}}{\sqrt{d}}\cdot 12\leq(1+o_d(1))\cdot 6\sqrt{2}$$

for a circle vertex, and

$$(1+o_d(1))\cdot 6\cdot\sqrt{m}\cdot\frac{2}{d}\leq(1+o_d(1))\frac{12\sqrt{m}}{d}$$

for a square vertex where the factor 6 *for each vertex comes from our bound that each vertex arrived using a forced R edge can be specified at a cost of* [6].

Proof to lemma 3.4.13. By our edge-copy charging scheme and Proposition 3.4.14, we observe the following,

1. Treat the $U-V$ path as a gadget, and we have 2 choices depending on whether $U=V$ for the upcoming block;
2. For each gadget-step, we sum over the edge-labelings;
3. For each gadget along the dangling path, we pick up at most the following factor,
   - M_α: it gives 2 circle vertices and 1 square vertex, which is a factor of at most
   $$B_\alpha:=(1+o_d(1))\cdot 3^4\cdot\frac{4}{d^2}\cdot\sqrt{md^2}\cdot 6^3\,;$$
   - M_β: it gives 1 circle vertex and 1 square vertex, which is a factor of at most
   $$B_\beta:=(1+o_d(1))\cdot 3^4\cdot\frac{2}{d^2}\cdot\sqrt{md}\cdot 6^2\,;$$
   - M_D: it gives 1 circle vertex, which is a factor of at most $(1+o_d(1))\cdot 3^2$ if it is the first M_D gadget of the block with the factor 3 counting the step-label

of the edge-copies in the gadget, or a factor of at most $B_D := \tilde{O}(\frac{1}{\sqrt{d}})$ for any subsequent M_D gadgets.

4. For the $U - V$ path, the square vertex SQ_1 gives a factor of at most

$$(1 + o_d(1)) \cdot 6 \cdot \sqrt{\frac{m}{d^2}}$$

and the circle vertex combined gives a factor of at most

$$(1 + o_d(1)) \cdot (\sqrt{2})^2 \cdot 6$$

where the factor of 6 comes from at most one circle vertex being reached using R edge.

Therefore, combining the above bounds, we have

$$B(\mathcal{R}) \leq (1 + o_d(1)) \cdot 6(\sqrt{\frac{m}{d^2}} \cdot 6 \cdot \sqrt{2}^2) \cdot 3^2 \cdot \prod_{\leq \text{polylog } d \text{ gadgets}} (B_\alpha + B_\beta + B_D) < \frac{1}{2}$$

provided

$$B_\alpha, B_\beta < \frac{1}{2}$$

and

$$6(\sqrt{\frac{m}{d^2}} \cdot 6 \cdot \sqrt{2}^2) \cdot 3^2 < \frac{1}{2}$$

It can be verified that it suffices for us to take $m < \frac{1}{7000000} \cdot d^2$ though we do not emphasize upon the particular constant as we believe a more careful argument by tracking our above bounds can render an improved constant without much work. □

Lemma 3.4.30 (Restatement of lemma 3.2.11)**.** *With probability at least* $1 - 2^{-d^\epsilon}$ *for some*

constant $\epsilon > 0$,

$$\|\mathcal{R}\|_{spec} < \frac{1}{2}.$$

Proof. This follows by setting $q = d^{\epsilon}$ for some constant $\epsilon > 0$, and combines with our block-value function for small enough constant c. □

Chapter 4

Switching Matrix Norm Bounds: An Application to Random Regular Graph

4.1 Introduction

It is well-known in random graph theory that random d-regular graph denoted as $G_d(n)$ and Erdős-Rényi random graph $G(n, \frac{d}{n})$ share various phase transition thresholds at least in the leading order, eg. connectivity threshold, independence/chromatic number (see [FK15] for more examples) . For algorithmists, one intriguing question to ask is whether its algorithmic variant holds: for average-case computational problems (of both algorithms and hardness), can the result and the analysis be transferred from one distribution to another under some general condition? Crucially, this question remains unclear in various situations as analysis of average-case problems (eg. [HSS15, MW19, JP24] and see more examples in [AMP20]), especially those appealing to a spectral reasoning, is usually brittle in the presence of correlation of the underlying input.

Motivated by the above question, we study the certification question of independent sets in sparse random graphs. In general, a certification question refers to the algorithmic

task of finding an *efficiently certifiable* upper bound on the given input. Via moment methods, the largest independent set have its size tightly concentrated at $\Theta(\frac{n \log d}{d})$ on both distributions of Erdős-Rényi and d-regular. However, on the algorithmic side, the best known algorithm [HH70] employ spectral techniques, and for both distributions yield a bound of $O(\frac{n}{\sqrt{d}})$ which is essentially off by an $O(\sqrt{d})$ factor from the ground-truth. It is a major open question whether the gap of $O(\sqrt{d})$ may be closed, and the independent set problem has been serving as a prototypical example of information-v.-computation gap in the statistics inference community. In this context, Sum-of-Squares semidefinite programming hierarchy arises as a testbed for algorithmic intractability for average-case problems as the general theory of NP-hardness does not apply. Capturing spectral techniques that give the sated $O(\frac{n}{\sqrt{d}})$ bound, SDP hierarchy is a natural candidate to turn to for a possibly improved upper bound. The Sum-of-Squares hierarchy of SDP relaxations [Las00, Par00] is a hierarchy of increasingly powerful families parameterized by the *SoS-degree*: it captures the best known algorithms for many worst-case combinatorial optimization problems [GW94, ARV04] and yield improved and even optimal algorithms for statistical inference problems in the average-case setting [BRS11, GS11, HSS15, MSS16, RRS17, KSS18, HL18, KKM20, Hop19, BK21]. As a result of the power of Sum-of-Squares in algorithmic design, establishing lower bounds against it for statistical inference problems has been a major research direction, and such hardness is usually seen as strong evidence for algorithmic intractability.

Starting from the seminal work of [BHK+16] that resolves the planted clique problem within the Sum-of-Sqaures framework in the last decade, there is a line of work that continues to investigate the limit of these algorithms on random graphs [MRX20, PR20, Pan21, JPR+22, JPRX23, KPX24]: specifically, our work is directly inspired by [JPR+22, KPX24] that study the independent set problem on $G(n, \frac{d}{n})$, and a concrete question we ask is whether one can switch Sum-of-Squares lower bounds from Erdős-Rényi to random

regular graphs.

The particular aforementioned results do not extend to the d-regular graph setting and it is cast as an explicit open question whether one may obtain similar results in $G_d(n)$. Despite several results for basic SDP on random d-regular graphs [BKM19, DMO+19] and such results being anticipated given the results on Erdős-Rényi, not much is known for higher degree Sum-of-Squares in the regular graph setting (beyond degree-4). In fact, it was far from clear prior to this work how existing techniques from the i.i.d. setting may translate for the regular graph distribution, and what kind of new technical obstacles may arise. For example, it takes considerably technical efforts to improve the original result of [BHK+16] to satisfy the clique-size constraint exactly [Pan21], and it is conceivable that similar challenges may appear for the regular graph setting.

From this perspective, our work completes the missing piece in the paradigm of Sum-of-Squares hardness by extending its coverage to random regular graphs, and develops tools to enable a smooth transfer within the known framework for applications in contexts beyond max independent set. In particular, we believe our norm bound result would also enable a smooth transfer for applications outside Sum-of-Squares lower bounds by considering spectral algorithms for average-case problems, and we leave this direction for future work.

Works for Low-Degree Polynomial Hardness and Level-2 Local Statistics Closely related to works in Sum-of-Squares lower bounds, low-degree polynomial framework is also another active platform for examining average-case hardness [KWB19a, Wei20, SW22, GJW22, RSWY22, KVWX23] . Random regular graphs have been investigated in this model [BBK+20], however, as far as we understand, a general theory for regular instances in the low-degree framework remains mostly at large due to the absence of an explicit orthogonal basis. That said, the recent work of [KY24] considers a close variant of our problem in the

local statistics hierarchy of degree $LoSt(2, D)$, and asks explicitly for results of hierarchy of higher degree $LoSt$ in regular graphs. On a high level, the local-statistics hierarchy may be viewed as an adaptation of Sum-of-Squares hierarchy to distinguishing problems, and $LoSt(2, D)$ may be seen as a hybrid of hardness against degree-D polynomials and the basic SDP (which is itself captured by degree-2 SoS). Our result addresses their question in the context of independence number.

Previous Works for Graph Matrix Norm Bounds The notion of graph matrix is first formally coined in [BHK+19] where its norm bounds are given in [BHK+19] and generalized in [AMP20] for the dense setting for random variables with bounded Orlicz norm. In the sparse random graph setting, for random graphs of average degree at least polylog(n), rough norm bounds are studied in [JPR+22] and [RT22]. More recently, our previous work [KPX24] optimizes the subpolynomial dependence of the norm bounds which are crucial for the sparse setting, and gives almost tight bounds in the setting of bounded average degree. Our work showcases the versatility of the framework in [KPX24] by adapting it to the regular graph setting. We believe that following our analysis here, a careful combination of some analog of the main lemma from [Sar23] in the sparse regime (implicit in [Bor19]) and [KPX24] would yield norm bounds for the sparse regime for $d = O(1)$ (albeit in a slightly different model for random regular graphs), and we leave this for future work.

Distribution of Random d-Regular Graphs $G_d(n)$ The spectrum and the spectral norm of random regular graph is an important topic in random matrix theory, and there have been a long line of works spanning over decades (eg. [HH70, BS87, Fri03, TVW13, BLM15, TY16, Bor19, HY21] and we refer reader to see references therein). Before we dive into technical analysis for Sum-of-Squares, let us first make clear the specific distribution of random regular graphs $G_d(n)$ that we are working with throughout this work. We start

by considering the set of all graphs on n vertices such that each vertex has degree d. Since such set is non-empty whenever nd is even and $0 < d < n$, we may then impose a uniform measure on this set, and define $G_d(n)$ the uniform distribution of graphs drawn from this set.

Our choice of distribution is slightly different from the usual choices of configuration/lifting model studied in the previous works for low-degree polynomial analysis. Interestingly, these two alternatives are usually used for the ease of moment calculation as it is possible to employ direct probabilistic calculations in these models. However, we believe such distinction is rather technical and should not be conceptually significant: both distributions are both contiguous to the uniform distribution of random d-regular graphs. In fact, our specific choice is picked so that we may adopt the moment calculation from previous works in a blackbox manner, while we believe similar calculation on either model is true, and can be extracted so that our results for norm bounds may be plugged in mechanically again. That said, to keep the presentation of our work concise, and to maintain the focus on our switching analysis, we opt to restrict our attention to the distribution $G_d(n)$.

4.1.1 Informal Statement of Our Results

We now give a quick overview and an informal statement of results. We give matrix norm bounds for random d-regular graphs. The results are two-fold.

On the one hand, we identify the "crucial" combinatorial structure of the underlying shape that dictates whether the input distribution of Erdős-Rényi matters, i.e., when the same norm bound holds on both distributions. On the other hand, we identify the tight (up to lower order dependence) dependence of graph matrix norm bound for shapes that have different spectral norm upper bounds in these two distributions.

Theorem 4.1.1 (Informal of theorem 4.1.3)**.** *For shapes τ that do not have any edge "floating", i.e., edge-component not connected to the left/right matrix boundary , the same graph matrix norm bound (in [JPR+22]) for Erdős-Rényi continues to hold with high probability for the random d-regular graph distribution $G_d(n)$.*

For shapes τ that have some floating component that is additionally tree-like, the graph matrix norm bounds for Erdős-Rényi no longer hold in $G_d(n)$, and our norm bound for $G_d(n)$ gives a $\sqrt{n}$ factor blow-up over the prior norm bound is for each such component .

4.1.2 Main Theorems of Graph Matrix Norm Bounds on Random d-Regular Graphs

For some $\delta > 0$ and for any shape τ with $|V(\alpha)| \leq n^{\delta}$, and $G_d(n)$ the distribution of random d-regular graphs for any $\log^{10} n \ll d = o(n)$ and $p = \frac{d}{n}$, we obtain the following theorems,

Theorem 4.1.2 (User-friendly version for shapes without floating component)**.** *For a shape τ without floating component, we define*

$$B_q(\tau) := \max_{S:\text{separator for } \tau} (\sqrt{n} \cdot q)^{|V(\tau)\setminus S|} \cdot \left(\sqrt{\frac{1-p}{p}}\right)^{|E(S)|} \cdot \sqrt{n}^{|I(\tau)|} \cdot (c_{\mathsf{norm}})^{|E(\tau)|}$$

for some constant $c_{\mathsf{norm}}, c > 0$ and $I(\tau)$ the set of isolated vertices in τ. For any $\epsilon > 0$, there exists some large enough constant $c > 0$ such that for any $q = \Omega(|U_\tau| |\log^c n)$ and $q < \min(d^{1/10}, n^{O(\delta)})$,

$$\mathbb{E}_{G\sim G_d(n)} \left[\mathrm{Tr}\Big((M_\tau M_\tau^\top)^q\Big)\right] \leq \big((1+\epsilon)\cdot B_q(\tau)\big)^{2q}$$

Theorem 4.1.3 (Full version for shapes containing floating component)**.** *For a shape τ*

possibly containing floating components, we define

$$\mathsf{float}(\tau \setminus S) = \prod_{C_i\text{: floating component not connected to } S} \sqrt{n} \cdot 1[C_i \text{ is tree-like}]\,,$$

for any $S \subseteq V(\tau)$ where $1[C_i$ is tree-like$]$ is the indicator for whether edge-induced component C_i is tree-like, and define

$$B_q(\tau) := \max_{S\text{:separator for } \tau} (\sqrt{n} \cdot q)^{|V(\tau)\setminus S|} \cdot \left(\sqrt{\frac{1-p}{p}}\right)^{|E(S)|} \cdot \sqrt{n}^{|I(\tau)|} \cdot \mathsf{float}(\tau \setminus S) \cdot (c_{\mathsf{norm}})^{|E(\tau)|}$$

for some constant $c_{\mathsf{norm}}, c > 0$ and $I(\tau)$ the set of isolated vertices in τ. For any $\epsilon > 0$ and for any $q = \Omega(|U_\tau| \log^c n)$ and $q < \min(d^{1/10}, n^{O(\delta)})$,

$$\mathbb{E}_{G\sim G_d(n)} \left[\mathrm{Tr}\Big((M_\tau M_\tau^\top)^q\Big)\right] \le \big((1+\epsilon) \cdot B_q(\tau)\big)^{2q}$$

Let us now unpack the above norm bounds for clarity. Towards the end, we recall the norm bound from [JPR+22] for the i.i.d. setting as

$$\tilde{O}\Big(\max_{S\text{:separator}} \sqrt{n}^{|V(\tau)\setminus S|} \cdot \sqrt{n}^{|I(\tau)|} \cdot \sqrt{\frac{1-p}{p}}^{|E(S)|}\Big)$$

The main difference between the norm bound for shape with and without floating component is the extra factor captured by $\mathsf{float}(\tau \setminus S)$, that is each tree-like floating component disconnected from the separator S requires an extra factor of $\sqrt{n}$ in the candidate upper bound. Despite the difference incurred by floating component, the candidate bound stated above are essentially identical as that for the i.i.d. setting in [JPR+22] up to subpolynomial dependence.

As a corollary, we obtain the following matrix norm bound that holds with probability at least $1 - o_n(1)$, $\|M_\tau\| \le (1 + o_n(1)) \cdot B_q(\tau)$.

4.2 What's the Same and what's Different?

A natural starting point for our work is to build upon our machinery for input from i.i.d. input of sparse random graphs as discussed in the first chapter. For concreteness, we recap our definition for graph matrix while adapted to the setting of random regular graphs.

4.2.1 Definition of Graph Matrix via the Basis for Erdős-Rényi

Let us now start with the formal definition of graph matrix. Usually in the case of i.i.d. input, graph matrix is defined using the corresponding orthogonal basis for the null distribution, which is the p-biased basis for the Erdos-Renyi distribution $G(n, \frac{d}{n})$. The specific basis is defined using the following Fourier characters,

Definition 4.2.1 (Fourier character for $G(n, \frac{d}{n})$)**.** *Let χ denote the p-biased Fourier character, we have*

$$\chi(1) = \sqrt{\frac{1-p}{p}} \approx \sqrt{\frac{n}{d}},$$

and

$$\chi(0) = -\sqrt{\frac{p}{1-p}} \approx -\sqrt{\frac{d}{n}}.$$

For a subset of edges $S \subseteq \binom{n}{2}$, we denote its corresponding basis element as $\chi_S(G) = \prod_{e \in S} \chi_e(G)$. Throughout this work, we use p and $\frac{d}{n}$ interchangeably unless otherwise specified.

To see that this forms the an orthogonal basis for Erdős-Rényi $G(n, \frac{d}{n})$, it is straightforward to verify that each edge is independent of each other, and each edge character has mean 0 and variance 1. In this work, even though we are working with input coming random d-regular graph, we define the underlying edge-character as the above Fourier character for Erdős-Rényi graph of $G(n, \frac{d}{n})$. Crucially, we want to point out that that the edge characters are no longer, even though close to being, orthogonal. From this point on,

we will use G to refer to the graph input in general, and as we use the same edge character, we shall not make the distinction of the distribution from which G is sampled.

Next, we are ready to introduce *shape*, a representation of structured matrices whose entires are polynomial functions of the underlying graph input G.

Definition 4.2.2 (Shape)**.** *A shape τ is a graph on vertices $V(\tau)$ with (possibly multi-)edges $E(\tau)$ and two ordered tuples of vertices U_τ (left boundary) and V_τ (right-boundary). We denote a shape τ by $(V(\tau), E(\tau), U_\tau, V_\tau)$.*

We remind the reader that V_τ should be distinguished from $V(\tau)$, where V_τ is the right boundary set, while $V(\tau)$ is the set of all vertices in the graph.

Definition 4.2.3 (Shape transpose)**.** *For each shape τ, we use τ^T to denote the shape obtained by flipping the boundary U_τ and V_τ labels. In other words, $\tau^T = (V(\tau), E(\tau), U_{\tau^T} = V_\tau, V_{\tau^T} = U_\tau)$.*

Definition 4.2.4 (Embedding)**.** *Given an underlying random graph sample G, a shape α and an injective function $\psi(\alpha) : V(\alpha) \to [n]$, we define $M_{\psi(\alpha)}$ to be the matrix of size $\frac{n!}{(n-|U_\alpha|)!} \times \frac{n!}{(n-|V_\alpha|)!}$ with rows and columns indexed by ordered tuples of $[n]$ of size $|U_\alpha|$ and $|V_\alpha|$ with a single nonzero entry*

$$M_{\psi(\alpha)}[\psi(U_\alpha), \psi(V_\alpha)] = \prod_{(i,j)\in E(\alpha)} \chi_G\left(\psi(i), \psi(j)\right).$$

and 0 everywhere else.

Definition 4.2.5 (Graph matrix of a shape)**.** *For a shape α, the graph matrix M_α is*

$$M_\alpha = \sum_{\text{injective } \psi: V(\alpha)\to[n]} M_{\psi(\alpha)}.$$

When analyzing the Sum of Squares hierarchy, we extend M_α to have rows and columns indexed by all tuples of vertices of size at most $\frac{\mathsf{dsos}}{2}$ by filling in the remaining entries with 0.

In words, for a graph matrix of shape τ, its entry of $M_\tau[S,T]$ for set S,T agreeing with the boundary condition of τ is given by the summation over the injective labeling of middle vertices in $V(\tau) \setminus U_\tau \cup V_\tau$.

Before we state our main result for graph matrix norm bounds, we make one further definition for graph matrix that will crucially reflect the difference of norm bounds for the i.i.d. distribution and for the regular graphs.

Definition 4.2.6 (Isolated vertex)**.** *For a shape τ and a vertex $v \in V(\tau) \setminus U_\tau \cup V_\tau$, we call it* isolated *if it is not incident to any edge. However, vertices in $U_\tau \cup V_\tau$ are never considered isolated.*

Definition 4.2.7 (Floating component)**.** *For a shape τ, and an connected component C induced by edges, we call C* floating *if there is no path from C to $U_\tau \cup V_\alpha$.*

Importantly, observe that an isolated vertex of degree 0 in $V(\alpha) \setminus U_\alpha \cup V_\alpha$ is not floating as we consider edge-induced component in the above definition. Finally, we make clear of the definition of a component being tree-like even though it may already be clear as its name suggests,

Definition 4.2.8 (Tree-like component)**.** *We call a connected component C with vertices $V(C)$ and edges $E(C)$ tree-like if $V(C) = E(C) + 1$. In other words, there is no cycle in the connected component.*

Trace Moment Method: Same Machinery yet with a New Focus Again, we apply trace moment method here to study the spectral norm of random matrices.

Taking the expectation over the randomness of the underlying input G, and reorganizing the terms on the righthand side by grouping steps by their underlying labeled-edge,

we can write the expected trace as

$$\mathbb{E}_G \operatorname{tr}\Big((M_\tau M_\tau^\top)^q\Big) = \mathbb{E}_G \sum_{\substack{S_1,T_1,S_2,\ldots,S_{q-1},T_{q-1} \\ S_i\text{:labeling of left boundaries of } \tau \\ T_i\text{:labeling of right boundaries of } \tau}} M_\tau[S_1,T_1]M_\tau^\top[T_1,S_2]\ldots M_\tau^\top[T_{q-1},S_1]$$

$$= \mathbb{E}_G \sum_{\substack{P \\ \text{shape-walk of length } q}} \prod_{e\in E(P)} \chi_G(e)^{\mathrm{mul}_P(e)}$$

where we use $\mathrm{mul}_P(e)$ to denote the multiplicity of labeled-edge e in the walk P. In the i.i.d. setting, the above quantity further factorizes over edges as

$$\mathbb{E}_G \sum_{\substack{P \\ \text{shape-walk of length } q}} \prod_{e\in E(P)} \chi_G(e)^{\mathrm{mul}_P(e)} = \sum_{\substack{P \\ \text{shape-walk of length} q}} \prod_{e\in E(P)} \mathbb{E}_G[\chi_G(e)^{\mathrm{mul}_P(e)}],$$

and a crucial observation at this point is that for each walk to give non-zero contribution, each labeled edge needs to appear in at least two steps, as otherwise the expectation of the walk factorizes over the edges and edges appearing a single time gives mean 0.

Noting that such immediate factorization is not true in our setting of random regular graphs, however, we will proceed with the i.i.d. setting to describe the machinery of obtaining matrix norm bound from block-value bound from [AMP20, JPR+22, HKPX23, KPX24] , and then outline challenges and solution posed by regularity in the subsequent section.

In particular, our norm bound techniques may be viewed as a showcase for the versatility of the machinery developed in [KPX24] with simplification as we do not optimize over polylog dependence which is the main target of their work, and with adaptation to random regular graphs. On a high level, we are able to apply the combinatorial component (bound on vertex-factor) from our prior machinery in an essenly blackbox manner, while our focus is on the edge-value component.

4.2.2 Bounding Walk Value: putting in the regularity

Before we consider how the edge-value may be factorized into each BlockStep, a crucial starting point is to understand what the edge-value bound globally for a whole walk. In particular, we want to find an upper bound of edge-value for $\mathbb{E}_{G\sim G_d(n)} \underbrace{\prod_{e\in P} \chi_G(e)^{\mathrm{mul}_P(e)}}_{:=\mathsf{edgeval}_{\mathrm{scaled}}(P)}$ for a "not-too-large" subset of edges $P \subseteq \binom{n}{2}$ with various multiplicities in a random regular graph.

Fortunately, such question is well-understood in the context of random regular graphs, and we use result from the previous work of [Sar23] in the context of the spectral gap for random d-regular graph. Before we state the known result, we first make the following additional definitions,

Definition 4.2.9 (Surprise step/visit)**.** *In a shape walk, we call a step a* surprise visit *if it leads to a vertex that has already been visited/explored in the walk.*

Definition 4.2.10 (Singleton step/edge)**.** *In a shape walk P, we call a step along labeled edge e (of the underlying randomness) a singleton-edge/step if e appears only once throughout the whole walk.*
On the other hand, we call a step using along a labeled edge e that appears more than once a non-singleton step and the underlying edge a non-singleton edge.

We are now ready to recall the previous known bound, which bounds the unscaled value of a walk P of length-$2q$ in a random d-regular as the following.

Lemma 4.2.11 (Restated edge-value bound from Corollary 3.6 in [Sar23])**.** *Let $\log n \ll q \ll d^{1/10}$, and $d < cn$ for some constant $c > 0$, define*

$$\mathsf{edgeval}_{unscaled}(P) := \prod_{e\in P} \underbrace{(G(e) - \frac{d}{n})^{mul_P(e)}}_{unscaled\ character},$$

where $G(e)$ is the $0/1$ indicator variable for edge in graph sample G, the following bound holds

$$|\mathbb{E}_{G\sim G_d(n)}\mathsf{edgeval}_{unscaled}(P)| \leq (2+o(1))\cdot(2q)^{2\cdot|surprise(P)|}\cdot \\ (p(1-p)/n)^{|singleton(P)|/2}(p(1-p))^{|NonSingleton(P)|}$$

where $surprise(P)$ is the set of surprise visits in the walk P, and $Singleton(P)$ is the set of singleton edges (i.e., edges that appear only once in the walk), $NonSingleton(P)$ is the set of edges that appear at least twice, and importantly, $|NonSingleton(P)|$ denotes the number of such edges (not counting multiplicities).

At this point, we shall emphasize that the most important aspect of the above bound is the factor of $(p(1-p)/n)^{|\text{singleton(P)}|/2}$ which accounts for the change that we are working in a non i.i.d. setting, and each edge appearing only once no longer has zero expected value as in the i.i.d. setting. This lemma does not follow from an immediate magnitude bound, and requires a careful analysis using cancellation of the terms that appear in the moment expansion which is arguably the main technical component of [Sar23]: compared with magnitude bound, it comes with an extra $\frac{1}{\sqrt{n}}$ decay for each singleton edge without which our spectral norm bounds would degrade into a trivial Frobenius norm bound for various shapes.

Before we combine this edge factor with combinatorial counting, let us switch back to the Fourier basis in which edge has their character given, and note that the scaling is picked such that

$$\mathsf{edgeval}_{\text{scaled}}(P) = \mathsf{edgeval}_{\text{unscaled}}(P)\cdot\sqrt{\frac{1}{p(1-p)}}^{\text{mul}(P)}$$

where $\text{mul}(P) := \sum_{e\in P}\text{mul}_P(e)$ is the total edge multiplicity of walk P, and $\text{mul}_P(e)$ is the multiplicity of edge e in P. Combining with the magnitude bound for the unscaled value,

we have

$$\begin{aligned}
&\left|\mathbb{E}_{G\sim G_d(n)}\mathsf{edgeval}_{\text{scaled}}(P)\right| \\
&\leq (2+o(1))\cdot(2q)^{2\cdot|\text{surprise(P)}|}\cdot(p(1-p)/n)^{|\text{singleton(P)}|/2}\cdot(p(1-p))^{|\text{NonSingleton}(P)|} \\
&\cdot\sqrt{\frac{1}{p(1-p)}}^{\text{mul}(P)}
\end{aligned}$$

With the above bound for the walk value, we arrive at our edge-value assignment scheme that assigns factors from the above upper bound to each edge's multiplicity in the walk. We follow the notation in previous works and label each edge's multiplicity (i.e. step) as the following,

At this point, it is crucial for the reader to distinguish a *Singleton* step and an *S* step of surprise visit. In fact, we advise for the readers to skip upon *S* step of surprise visit as they only concern for subpolynomial dependence of the final norm bound which we do not optimize in this work.

Proposition 4.2.12. *The following step-value assignment scheme gives an upper bound for the edge-value up to a factor of* $O(1)$*:*

1. *For singleton step/edge, assign a value of* $\sqrt{\frac{p(1-p)}{n}}\cdot\sqrt{\frac{1}{p(1-p)}}=\sqrt{\frac{1}{n}}$*;*
2. *For an* $F/S/R$ *step of an edge that appears more than once, assign a value of* $\sqrt{p(1-p)}\cdot\sqrt{\frac{1}{p(1-p)}}=1$*;*
3. *For an S step, i.e., the underlying edge is non-singleton and appearing for the first time wile the step leads to a visited vertex in the walk, we additionally assign a value of* $(2q)^2$*. That said, an S step of a surprise visit gets assigned a value of* $(2q)^2$ *in total;*
4. *For an H step of an edge that appears more than twice, assign a value of* $\sqrt{\frac{1-p}{p}}$*.*

Let $f(s)$ be the assigned value of step s in according to the above scheme,

$$(2+o(1)) \prod_{s \in P:step} |f(s)| \geq |\mathsf{edgeval}_{scaled}(P)|$$

In words, up to the factor of $(2+o(1))$, the above step-value assignment scheme assigns the upper bound factor of the walk to each step.

Proof. We first observe that the factor $\sqrt{\frac{1}{p(1-p)}}$ gets assigned to each step regardless of label, which is consistent to the fact that we have $\sqrt{\frac{1}{p(1-p)}}^{\mathrm{mul}(P)}$ in the original upper bound. Additionally, for each edge that appears more than once, we pick up a factor of $p(1-p)$ and this is distributed among the $F//R$ steps of the edge. For edges that appear only once, the whole factor is assigned to the singleton edge. For the factor of $(2q)^{|surprsie(P|)}$, we assign them to the S-step in which the edge is making the first appearance. □

4.2.3 Explicit Application of BlockValue Machinery

In this section, we apply the above machinery to the explicit graph matrices of line graph (see following for diagrams) to familiarize readers with our described scheme, and show one can easily recover the $\tilde{O}(\sqrt{n})$ bound for centered adjacency matrix, which is equivalent to a $\tilde{O}(\sqrt{d})$ bound for the spectral gap of a random graph.

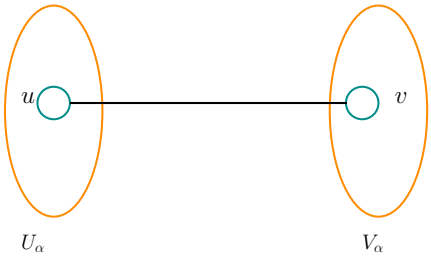


Figure 4.1: Line-graph
$M_{line}[u,v] = \chi_G(u,v)$

Lemma 4.2.13. *For the line-graph τ_{line}, we have*

$$B(\tau_{line}) \leq \tilde{O}(\sqrt{n}),$$

as a corollary, we have $\|M_{line}\| \leq O(\sqrt{n}\,\mathrm{poly}\log(n))$ w.h.p.

Proof. We first take $q = \mathrm{polylog}\, n$. To apply block-value bound, fix a traversal direction from U_τ to V_τ, and given current boundary at U_τ, we start by casing the step-label. If this is an F-step

1. V_τ is making a first appearance and a factor of $\sqrt{n} \cdot q$ is assigned;
2. U_τ is making a middle appearance and at most a cost of q is assigned to the vertex;
3. The edge is non-singleton and making the first appearance, which gets value 1;
4. This combines to a value of $\sqrt{n} \cdot q$ in total.

Similarly if this is an R-step,

1. V_τ is potentially making a last appearance and at most a factor of $\sqrt{n} \cdot q$ is assigned (a factor of q for middle appearance);
2. U_τ is making a middle appearance and at most a cost of q is assigned to the vertex;
3. The edge is non-singleton and making the first appearance, which gets value 1;
4. This combines to a value of $\sqrt{n} \cdot q$ in total.

If this is an S/H-step, we have at most a vertex factor of q^2 in total, and an edge-value of 1 or in the case of H-step, a value of $\sqrt{\frac{1-p}{p}} \approx \sqrt{\frac{n}{d}}$. This gives a total factor of q^2 and $\sqrt{\frac{1-p}{p}} \cdot q^2$.

Lastly, if this is a singleton-step, we have vertex factor of at most n in total when the left vertex is making a last appearance and the right vertex a first appearance. Crucially, the singleton-step gives a decay of $\frac{1}{\sqrt{n}}$, and therefore we have a combined bound of $\sqrt{n}$.

Summing over the above step-labels, we have

$$B(\tau) \leq 2 \cdot \sqrt{n} \cdot q^2 + q^2 + \sqrt{\frac{1-p}{p}} \cdot q^2 + \sqrt{n} \leq \tilde{O}(\sqrt{n})$$

as we take $q = \text{poly} \log n$. □

4.2.4 Connecting back to Vertex Separator

The crucial quantity governing graph matrix norm bounds is the notion of vertex separator, defined as the following,

Definition 4.2.14 (Vertex separator)**.** *For a shape τ, we say $S \subseteq V(\tau)$ is a vertex separator for τ if any path from U_τ to V_τ passes through some vertex in S.*

In the previous analysis, the notion of vertex separator pops up as each edge needs to appear twice in the walk, and therefore, there is always some vertex in a given BlockStep that is making a middle appearance. However, this is no longer true for us as we may non-zero value walk that contain singleton edge and a priori, it is not at all clear whether vertex separator continues to control matrix norm bound in the regular setting.

To relate the block value to vertex separator, we start with a step-labeling of edges in the given block τ. In fact, we restrict to step-labeling that does not involve singleton-step first. This would enable us to recover the usual graph matrix norm bounds. For concreteness, assume this is an τ block (as opposed to $\tau^\top$) and we are traversing from U_τ (left boundary) to V_τ (right boundary). Importantly the vertices in U_τ are already specified for us.

Building the Separator from BlockStep-Labels For a block of τ traversing from U_τ and V_τ, and let $L : E(\tau) \to \{F, R, H, \text{singleton}\}$ be a step-labeling of the given block, we build S_B the separator for the given block as the following,

1. Include any vertex in $U_\tau \cap V_\tau$ into S_B;
2. Include any vertex incident to H-step into S_B;
3. Include any vertex incident to both F, R edges into S_B;
4. Include any vertex in U_τ incident to some F/S-step (note that S is a surprise step as opposed to a singleton-step) into S_B;
5. Include any vertex in V_τ incident to some R-step into S_B;
6. Additionally include any other vertex making middle appearance into S_B in the given block.

At this point, it is not clear whether S_B built from the above process is a vertex separator for any block-labeling, not to mention whether it governs the matrix norm. In fact, it is possible in our case that S_B is not a vertex separator for τ at all. That said, we first consider the case where there are no singleton-steps, and in this case, one can see the set S_B built from above is indeed a vertex separator. This in part explains why matrix norm bounds are connected to vertex separator in the previous works for i.i.d. setting.

Claim 4.2.15. *In the absence of singleton-steps, S_B is a vertex separator, i.e., any path from U_τ to V_τ passes through some vertex in S_B.*

Proof. To show S_B is a vertex separator, it suffices for us to show any path from U_τ to V_τ passes through some vertex in S_B. Consider any such path, we first observe that if the path is trivial (i.e., no step is involved), such path is blocked by $U_\tau \cap V_\tau \subseteq S_B$. For the non-trivial path, such path cannot contain any H-step, as any H-step has both its endpoints included

in S_B, and therefore blocked by the separator. Similarly, since each vertex incident to both F and R-steps are included into the separator, any path from U_τ to V_τ must be of either all F or all R-steps.

In the case of a path being of all F-steps, let u be a vertex in which path intersects U_α, this is a U_τ vertex incident to F-step and therefore included into the set S_B. Analogously, an all R-step patg passes through some vertex in V_τ incident to an R-step and included into S_B. Therefore, any path from U_τ to V_τ passes through the set S_B, and S_B is indeed a vertex separator. □

From the above proof, we can also arrive at the following corollary immediately,

Corollary 4.2.16. *In the presence of singleton-steps, S_B is not necessarily a vertex separator, however, any path not blocked by S_B is all-singleton.*

Observation 4.2.17. *Any vertex included into S_B in the above process is a vertex making a middle appearance in the given block.*

With the connection between vertex separator and step-labeling established, we are now ready to bound the block-value of a given step-labeling, and see why it is indeed the norm bound given by vertex separator in the previous works.

4.2.5 Deducing Block-Value from Step-Labeling

In this section, we combine the edge-value assignment scheme and our vertex-factor assignment scheme introduced earlier to bound the block-value of a given step-labeling . On a high-level, we first consider the case where there is no singleton-edge, and show in this case we recover the usual graph matrix norm bounds.

However, as we show in the last section, a step-labeling in the presence of singleton-edges is not necessarily a vertex separator, therefore it is not quite sufficient to consider

step-labeling free of singleton-step. To handle such discrepancy, we show that any step-labeling L with singleton-step, we can construct an alternative BlockStep-labeling L' that is singleton-step free, and furthermore we have the BlockValue bound $B(L) \leq B(L')$. Therefore, we show that for the sake of maximizer of the block-value (and thereby of the matrix norm bound), it suffices for us to consider L that is free of singleton-step, and for these step-labelings, we have a "genuine" vertex-separator as usual.

Lemma 4.2.18. *For a shape τ, for any step-labeling L of τ that does not contain singleton-step, let S be its associated vertex separator built from the prescribed process, we can bound its block-value by*

$$\begin{aligned} B(L) &:= \mathsf{vtxcost}(L) \cdot \mathsf{edgeval}(L) \\ &\leq (\sqrt{n} \cdot q_\tau)^{|V(\tau))\setminus S|} \cdot \sqrt{\frac{1-p}{p}}^{|E(S)|} \cdot \sqrt{n}^{|I(\tau)|} \cdot q_\tau^{|V(S)\setminus U_\tau \cap V_\tau|} \cdot \max(1, q_\tau^{|E(\tau)\setminus E(S)|-|V(\tau)\setminus V(S)|}) \end{aligned}$$

With some abuse of notation, we also write $B(S) := B(L)$ defined as above as we observe the RHS quantity only has dependence on L through S.

Proof. The bound follows from observing the following,

1. By our vertex factor assignment scheme, any vertex outside the separator is making a first or last appearance in this block but not both since L is free of singleton-step, and gets assigned a vertex cost of $\sqrt{nq_\tau} \leq \sqrt{n} \cdot q_\tau$;

2. Any H edge gets assigned an edge-value of $\sqrt{\frac{1}{p(1-p)}} \approx \sqrt{\frac{n}{d}}$, and any such edge is inside the separator, therefore, we have at most a factor of $\sqrt{\frac{n}{d}}^{|E(S)|}$ in total;

3. Any excess edge gets assigned a factor of q_τ in its F/R-step, therefore we have a factor of $q_\tau^{|E(\tau)\setminus E(S)|-|V(\tau)\setminus S|}$ in total;

4. Any vertex inside the separator other than $|U_\tau \cap V_\tau|$ is making a middle appearance and gives a factor of q_τ, where we notice that a factor is spared for $|U_\tau \cap V_\tau|$ as no encoding is needed for such vertices.

□

Lemma 4.2.19. *For a shape τ, for any of its step-labeling L that contains singleton-step, if τ does not contain any floating component, there exists a separator S whose corresponding block-value bound that at least matches the block-value bound of L. In other words, there exists vertex separator S such that*

$$B(L) \leq B(S) = (\sqrt{n} \cdot q_\tau)^{|V(L)\setminus S|} \cdot \sqrt{\frac{1-p}{p}}^{|E(S)|} \cdot q_\tau^{|V(S)\setminus U_\tau \cap V_\tau|} \cdot \max(1, q_\tau^{|E(\tau)\setminus E(S)|-|V(\tau)\setminus V(S)|})$$

Proof. For starters, notice that our block-value bound factorizes over connected component, and it suffices for us to consider a single connected component. We first observe given a step-labeling, we can decompose τ into singleton-components by only considering singleton-step edges in τ.

We now case on whether $S(L)$, the candidate vertex separator built from L containing singleton-step, is a vertex-separator for L. If so, it suffices for us to verify our previous bound of

$$B(L) \leq (\sqrt{n} \cdot q_\tau)^{|V(L)\setminus S|} \cdot \sqrt{\frac{1-p}{p}}^{|E(S)|} \cdot q_\tau^{|V(S)\setminus U_\tau \cap V_\tau|} \cdot \max(1, q_\tau^{|E(\tau)\setminus E(S)|-|V(\tau)\setminus V(S)|})$$

continues to hold for the given block when we take $S = S(L)$ even in the presence of singleton-step. To see this, notice the previous argument bounding vertex factors in identical way for vertices outside the separator that are not making both first and last appearances, and therefore, it suffices for us to focus on the "new" vertices making both first and last appearance due to the non-repetitiveness of singleton-edges. Any such vertex

is a non-isolated vertex (otherwise if isolated, its double factor has already been accounted for in the previous proof by the factor of $\sqrt{n}^{|I(\tau))|}$), and each such vertex gives an extra factor of $\sqrt{n}$. Call such vertex a *flip vertex*, and *unflipped* otherwise. That said, we can restrict our attention to these vertices and note that are all incident to some singleton edge that comes with an extra $\frac{1}{\sqrt{n}}$ decay. Therefore, the task is to assign a singleton edge for each such vertex that is making both first and last appearance in the block.

Consider the graph induced by singleton edges, we observe that any flipped vertex has a path of singleton edges to an unflipped vertex in the case there is no floating component since it has a path to $U_\alpha \cup V_\alpha$. Therefore we may consider a BFS to traverse flipped vertex from some unflipped vertex via singleton edges, and assign the singleton edge to the vertex it leads to, and observe that

1. Each singleton edge, if non-excess, comes with a value $\frac{1}{\sqrt{n}}$ and gets assigned to the destination vertex, which is indeed the target we start off with;

2. Otherwise, if an excess edge (i.e. a singleton edge that is also a surprise step leading to visited vertex), we incur $q^{O(1)}$ factor but it has an unassigned $\frac{1}{\sqrt{n}}$ factor as there must be at least another edge that is assigned to the destination vertex: if the vertex makes first appearance in the block, either it is not flipped and an extra $\sqrt{n}$ factor is not needed , or it is a flipped vertex but the extra $\sqrt{n}$ factor has been offset by the first singleton step that explores it ,

and this completes the proof for the case $S(L)$ remains as a separator for τ.

On the other hand, if $S(L)$ is not a separator for τ, it suffices for us to construct a separator S and go through the above argument to show the desired block value bound of $B(L) \leq B(S)$. Towards this end, observe that there must be a path from U_τ to V_τ that does not intersect $S(L)$. By our construction of $S(L)$, any such path must be all singleton-step. Consider all such singleton-paths from U_τ to V_τ and let the union of all such paths be T.

Let X be the MVS for T and now we consider $S := S(L) \cup X$ as the new candidate vertex separator.

Observe that $S = S(L) \cup X$ is indeed a vertex separator now as any path not blocked by $S(L)$ is in T while any such path in T from U_τ to V_τ is blocked by X. It now remains to verify the block S value of S upper bounds the BlockValue of L.

Note that as the block-value bound factorizes over connected components, the factors of vertices/edges connected to $S(L)$ can be bounded by the previous argument, it suffices for us to focus on factors on the singleton path from U_τ to V_τ. Towards this end, we first consider a BFS from X using the singleton-step in L to keep track of the change of factors in the BlockValue due to the addition of X.

1. We start by observing for any vertex in X, it may either come from $U_\tau \Delta V_\tau$, or it is outside $U_\tau \cup V_\tau$. Any such vertex cannot be in $U_\tau \cap V_\tau$ since such vertices are automatically included into $S(L)$ already;

2. Any vertex in $X \setminus (U_\tau \cup V_\tau)$ is making a first and last appearance in L, which gives a factor of n . However, it now gives a factor of $\sqrt{n}$ in the bound on the RHS by being a vertex outside the separator, and this is a deficit of $\frac{1}{\sqrt{n}}$;

3. Any vertex in $X \cap (U_\tau \Delta V_\tau)$ is making either a first or last appearance in L (but not both) which gives a factor of $\sqrt{n}$. Similarly to the above, it gives a factor of 1 in the block-value bound of $B(S)$, which gives a deficit of $\frac{1}{\sqrt{n}}$;

4. As we consider the BFS from X using singleton-step, any such singleton-step corresponds to a gain of $\sqrt{n}$ as it gives value 1 for block-value bound of $B(S)$while a factor of $\frac{1}{\sqrt{n}}$ to $B(L)$;

5. Any vertex outside $U_\tau \cup V_\tau$ reached by a singleton-step gives a factor of n in $B(L)$, while a factor of $\sqrt{n}$ to $B(S)$ Combining the change of vertex factor with the edge

assigned to it, this is a change of

$$\frac{\sqrt{n}}{n} \cdot \sqrt{n} = 1\,;$$

6. Any vertex in $U_\tau \Delta V_\tau$ reached by a singleton-step gives a change of $\sqrt{n}$ to both bounds, therefore there is no change in the vertex factor restricted to these vertices. Importantly, since it is still reached by a singleton-step, we pick up a change of $\sqrt{n}$ on these vertices;

7. Combining the above, notice for any vertex i$v \in X \setminus U_\tau \cup V_\tau$, we can assign at least one vertex in $U_\tau \setminus V_\tau$ and $V_\tau \setminus U_\tau$ reached by a BFS from v without passing through any other vertex in X. As otherwise, $X \setminus \{v\}$ continues being a separator of T while this contradicts with the minimality assumption of X to start with. With at least two vertices in $U_\tau \cup V_\tau$ assigned to each vertex v, we have a total change of

$$\frac{1}{n} \cdot (\sqrt{n})^2 = 1;$$

8. Similarly, for any vertex $v \in X \cap (U_\tau \Delta V_\tau)$ (W.L.O.G. consider $v \in X \cap U_\tau$) ,we can assign at least one vertex in V_τ reached by the BFS via a singleton-step from v without passing through other vertex in X, as again otherwise removing v from X preserves a separator for T while contradicting with the minimality. This is a change of

$$\frac{1}{\sqrt{n}} \cdot \sqrt{n} = 1\,.$$

This proves that the block-value bound $B(S) \geq B(L)$ for $S = X \cup S(L)$ for step-labeling L □

To wrap up, we have shown that any step-labeling has its block-value bounded by some

separator, therefore, using a loose bound to crudely bound the number of step-labeling, the final bound follows as

$$B_q(\tau) := \sum_{L\text{: step-labelings for } E(\tau)} B(L) = \sum_{L\text{: step-labelings for } E(\tau)} \mathsf{vtxcost}(L) \cdot \mathsf{edgeval}(L)$$
$$\leq c^{|E(\tau)|} \cdot \max_{S\text{:separator}} B_{\tau,q}(S)$$

for some constant $c > 0$, and we introduce the short-hand $B_{\tau,q}(S)$ to refer to $B(S)$ for shape τ when the dependence on shape τ may be unclear from the context.

One may then observe that $c^{|E(\tau)|} \cdot B_{\tau,q}(S)$ are indeed the norm bound given for graph matrix in the i.i.d. setting by [JPR+22], and in particular, the polynomial factors in $B_{\tau,q}(S)$ match exactly with the polynomial factors in the SMVS bound therein. Since the $c^{|E(\tau)|}$ factor is dominated by the factors of q (i.e. polylog factors in the B_q bound) in the $B(S)$ bound, we have shown that these two bounds match up to lower-order dependence.

However, as introduced in our technical overview, such comparison no longer holds for the scalar example and potentially more. We next show that these are the only examples that a blow-up is incurred in the norm bounds. Formally, we show that a $\sqrt{n}$ blow-up is incurred for each tree-like floating component, i.e. edge-component not connected to neither of the left/right boundary.

Before we get started, we first point out how floating component captures the scalar illustrated by the scalar random variable $p(x) = \sum_{i<j\in[n]} \chi_G(\{i,j\})$. This can be viewed as a shape with $U_\tau = V_\tau = \emptyset$ and $E(\tau) = p(x)$. Thus, with the boundary condition being empty, we indeed have a floating component and it is easy to see that is additionally tree-like.

We are now ready to augment the previous lemma to take into account of floating components.

Lemma 4.2.20. *For any shape τ, for any step-labeling L, there exits a separator S whose corresponding block-value bound upper bounds that of L. In other words, there exists vertex separator S such that for some constant $c > 0$,*

$$B(L) \leq (\sqrt{n} \cdot q_\tau)^{|V(\tau)\setminus S|} \cdot \sqrt{\frac{1-p}{p}}^{|E(S)|} \cdot \sqrt{n}^{|I(\tau)|} \cdot \sqrt{n}^{|float_{tree}(V(\tau)\setminus S)|}$$
$$\cdot \max(1, q_\tau^{|E(\tau)\setminus E(S)|-|V(\tau)\setminus V(S)|}) \cdot q_\tau^{|V(S)\setminus U_\tau \cap V_\tau|} \cdot c^{|E(\tau)|}$$

where $|float_{tree}(V(\tau) \setminus S)|$ is the number of tree-like floating components not connected to S.

Proof. The key of this lemma is that each floating component gives at most a $\sqrt{n}$ blow-up. Again, we start with the observation that our block-value bound factorizes over connected components, and it suffices for us consider C a floating component. In particular, it suffices for us focus on the components not connected to S and those that are of all singleton-steps, as otherwise the previous proof applies: the component of vertices that make both first and last appearances are connected via singleton-step to vertices that are not making both appearances.

It now suffices to bound the block-value restricted to all-singleton floating components, and its value is given by $n^{|V(C)|} \cdot \frac{1}{\sqrt{n}}^{|E(C)|}$ by our vertex-factor and edge-value assignment scheme. Notice in the case of tree-like floating components, we can bound it by

$$n^{|V(C)|} \cdot \frac{1}{\sqrt{n}}^{|E(C)|} \leq \sqrt{n}^{|V(C)|} \cdot \sqrt{n} \leq (\sqrt{n} \cdot q_\tau)^{|V(C)|} \cdot \sqrt{n}$$

and in the case of $|E(C)| \geq |V(C)|$, we can simply bound it by $(\sqrt{n}q_\tau)^{|V(C)|}$. Note that each bound on the RHS is the block-value factors in $B(S)$ associated with such floating component C, and this proves our desired lemma. □

4.2.6 Wrapping Up

We are now ready to conclude by completing the proof to our main theorem.

Proof. By lemma 4.2.20 , we have

$$B(\tau) \leq \sum_{L} B(L) \leq c^{|E(\tau)|} \max_{S} B_{\tau}(S)$$

By design of our block-value function (and grouping the extra factor of 2 from edge-value to the cost of identifying start of the walk), we have

$$\mathbb{E}_{G \sim G_d(n)} \operatorname{tr}\left(M_\tau M_\tau^\top\right)^q \leq O(n^{|U_\alpha|}) \cdot B_q(\tau)^{2q} \leq O(n^{|U_\alpha|}) B(\tau)^{2q}$$

To obtain concentration of the matrix norm bound, we then appeal to the following proposition and this completes our result of matrix norm bounds for random regular graphs. □

4.3 Application to Switching Sum-of-Squares Lower Bounds

In this section, we show the same construction from i.i.d. setting continues to work in the regular setting. The main observation of our switching is the insight that as the matrix for i.i.d. case is largely based on moment method, and in the previous section of graph matrix norm bounds, we have shown the moments of these two distributions are largely the same (up to floating components), we can now plug in the new norm bounds to the existing analysis with some more care on the blow-up incurred by floating components.

Our Results For our application we observe that it suffices for us to focus on shapes that do not have any floating component (in the dominant term) for the analysis of higher-

degree Sum-of-Squares lower bounds, and therefore we can essentially plug in the new norm bound to the existing SoS analysis. To motivate our results, it is most helpful for us to recap the SoS formulation for the independent set problem and recap the previous result for Erdos-Renyi random graph.

Definition 4.3.1 (Pseudo-expectation of degree dsos). *For any* dsos $\in \mathbb{N}$, *a degree-*dsos *pseudoexpectation in variables* $x = (x_1, x_2, \ldots, x_n)$ *(denoted by* $\widetilde{\mathbb{E}}$*) is a linear map that that assigns a real number to every polynomial* f *of degree* $< d$ *in* x *and satisfies: 1) normalization:* $\widetilde{\mathbb{E}}[1] = 1$, *and, 2) positivity:* $\widetilde{\mathbb{E}}[f^2] \geq 0$ *for every polynomial* f *with degree at most* dsos . *For any polynomial* $g(x)$, *a pseudoexpectation satisfies a constraint* $g = 0$ *if* $\widetilde{\mathbb{E}}[f \cdot g] = 0$ *for all polynomials* f *of degree at most* $\mathsf{dsos} - deg(g)$.

Definition 4.3.2 (Axioms for independent set)**.** *Let* G *be a* n*-vertex graph. The following axioms describe the* $0-1$ *indicators* x *of independent sets in* G*:*

$$\forall i \in [n], x_i^2 = x_i \quad \textit{(Booleanity);}$$
$$\forall (i,j) \in E(G), x_i x_j = 0 \quad \textit{(Independent set)} \ .$$

Working with the above set-up, the existing result reads as the following,

Theorem 4.3.3 ([JPR+22])**.** *With high probability over* G *sampled from* $G(n, \frac{d}{n})$, *there is an absolute constant* $c_0 \in \mathbb{N}$ *such that for sufficiently large* $n \in \mathbb{N}$ *and* $d \in [(\log n)^2, n^{0.5}]$, *and parameters* k, dsos *satisfying*

$$k \leq \frac{n}{\sqrt{d}} \cdot \frac{1}{\mathsf{dsos}^{c_0} \cdot \log n},$$

there is a pseudo-expectation of degree dsos *for independent set with objective value*

$$\widetilde{\mathbb{E}}[\sum_{i \in [n]} x_i] \geq (1 - o(1))k \, .$$

We are now ready to introduce our main result of switching the input distributions, which states the main result of theorem 4.3.3 holds with essentially the same parameter for inputs from $G_d(n)$ (up to lower order dependence of dsos).

Theorem 4.3.4 (Switching to Lower bounds for $G_d(n)$)**.** *With high probability over G sampled from the regular graph distribution $G_d(n)$, there is an absolute constant $c_1 \in \mathbb{N}$ such that for sufficiently large $n \in \mathbb{N}$ and $d \in [(\log n)^2, n^{0.5}]$, there is a pseudo-expectation of degree $\mathsf{dsos} < d^{1/10}$ for independent set with objective value*

$$\widetilde{\mathbb{E}}[\sum_{i\in[n]} x_i] \geq (1 - o(1))k\,.$$

for

$$k \leq \frac{n}{\sqrt{d}} \cdot \frac{1}{\mathsf{dsos}^{c_1} \log n}\,.$$

Remark 4.3.5. *Verbatim, our result for the regular graph distribution may seem to require an extra assumption of $\mathsf{dsos} < d^{1/10}$, however, we point out that produces makes minimal effect to the strength of our lower bounds in comparison the i.i.d. one as it is offset by the implicit $\mathrm{poly}(\mathsf{dsos})$ dependence of the objective value. In other words, the extra assumption can be dropped taking a slightly larger constant $c_1 > c_0$ for c_0 promised in the i.i.d. bound and set the objective value to be marginally smaller by* poly dsos *factor.*

4.3.1 Technical Overview of Application to SoS Lower Bounds

For higher-degere Sum-of-Squares lower bounds, the framework of "pseudo-calibration + PSD analysis" pioneered by the seminal work of [BHK+16] has been by now a standard (if not the only known) route, and has been proven successful in several different contexts. Therefore, with its success in related problems, pseduo-calibration is a natural starting point for us.

"Just Use Pseudo-calibration" Fails Again The recipe of pseudo-calibration, and closely related to it the framework of low-degree polynomial hardness, calls for a pair of null and planted distribution, and an explicitly known orthogonal basis for null. To show that algorithm or analysis may be transported from the planted distribution to the null distribution, the low-degree polynomial framework show that it suffices to study the change of measure in the Fourier basis for the null-distribution basis. And this is the beginning for various low-degree polynomial analysis and higher-degree Sum-of-Squares lower bounds. However, this is also where one immediately runs into trouble. Unlike the prior scenario in [JPR+22] where the difficulty is to find a quiet planted distribution, the regularity of our null distribution poses challenge to what an explicit Fourier basis would be for the regular graph distribution.

To address the absence of an explicitly known Fourier basis for the regular graph distribution, we settle with the closest known orthogonal basis we know, the p-biased Fourier basis for $G(n, \frac{d}{n})$. In fact, we make further compromise by recycling the moment matrix constructed for $G(n, \frac{d}{n})$, and plug in the input from random regular graph. In words, the candidate moment matrix construction in [JPR+22] shows that there exists a matrix-valued function

$$\Lambda : \text{Graph} \to \text{SoS Moment Matrix of degree-dsos}$$

such that for typical graph sample G comes from $G(n, \frac{d}{n})$, $\Lambda(G)$ is a valid pseudo-expectation of large objective value. However dangerous a move it may seem, we continue to apply the same matrix-valued function on graphs coming the regular-graph distribution, and observe that the independent-set constraints and large-value constraints continue to hold. We then defer further possible complications to the PSDness analysis.

Applying the Approximate Factorization Machinery We briefly recap the overall idea of PSD analysis from approximate factorization here while we refer the interested reader to [JPR+22, **?**] for a more thorough treatment of this topic. In general, the machinery applies when the Fourier coefficient factorizes over edges and vertices, which also applies to our moment matrix construction as we borrow the candidate matrix from the i.i.d. setting.

To get started, the machinery starts by factorizing the target matrix as $M = LQL^T$ for some graph matrix L (left-shapes), Q (middle-shapes) and L^T (also viewed as right-shapes). The intention of the decomposition is that Λ may have various eigenspaces corresponding to large eigenvalues and L serves as projection matrix to each eigenspace of large eigenvalues. The PSDness then amounts to showing the middle matrix Q is PSD. However, it turns out that the definition of Q is less explicit due to intersection terms that arise from three-way product of L, Q and L^T, and therefore, Q needs to be defined in a recursive manner. Despite the technicality of constructing Q, specifically in further factorizing intersection terms, the up-shot of the decomposition remains the same that Q is a matrix that does not contain large non-trivial eigenvalues. In particular, for the application to independent set, Q may be seen as a perturbed identity matrix and showing its PSDness amounts to showing $Q - Id$, in particular, via a decomposition into a sum of graph matrices, having spectral norm $o(1)$.

Handling the Discrepancy from Switching: Graph Matrix Norm Bounds for Regular Graphs The PSDness analysis of Q is usually where moment method and spectral norm bounds comes into play as the proof strategy, and where the bulk of our switching work rests. To enable a smooth transfer via moment method, one natural idea is to show that if the low-degree moments do not differ much between these two distributions , and one may then establish similar spectral norm upper bounds for graph matrices for both input distribution by using polynomial approximation for the spectral norm upper bound. Since

most of the prior analysis rely primarily on spectral norm upper bounds, existing analysis with the new norm bounds would follow with ease if we can show comparable spectral norm upper bounds for input from the regular graph distribution.

A technical caveat of the above strategy is that not all the bounds are the same for these two distributions. In particular, low-degree polynomial can easily distinguish whether a graph comes from Erdős-Rényi or the regular distribution. In fact, even degree-1 polynomial suffices to distinguish such.

Example 4.3.6 (Low-degree distinguisher for Erdős-Rényi and Regular Distribution)**.** *Consider the degree-*1 *of a single edge in the p-biased basis,* $p(G) = \sum_{i,j,k\in[n]} \chi_G(\{i,j\})\chi_G(\{k,j\})$*, one can bound w.h.p.* $|p(G)| \leq \tilde{O}(\sqrt{n}^3)$ *easily in Erdős-Rényi however, as a result of the non-orthogonality of the p-biased biases in the regular distribution, the scalar in regular distribution has typical magnitude* $\tilde{\Omega}(\sqrt{n}^4)$ *ignoring potential* $\mathrm{poly}(-d)$ *dependence.*

Remark 4.3.7. *We do not give a direct proof for this lower bound in* $G_d(n)$*, while we give a heuristic argument based on the distribution* $G(n, \frac{d}{n})$ *conditioned on vertices having degree exactly d. It can be verified by direct moment calculation that a blow up of at least* $O(\sqrt{n}\,\mathrm{poly}(-d))$*.*

Surprisingly, the existence of such low-degree polynomial distinguisher does not rule out the above strategy of showing comparable spectral norm upper bounds for graph matrices for both distributions, and applying prior PSD analysis. Analogously to the well-known example of adjacency matrix in both distributions having roughly the same norm bound, we show that comparable norm bounds continue to for a large family of graph matrices, i.e., those of shapes that do not contain floating components. We defer the formal definitions to the subsequent section, but in short, the polynomial deviation for the scalar bound is then reflected in our spectral norm upper bounds through the discussion on floating components.

For our application, the only tight case (in $\mathrm{poly}(n)$ order) comes from shapes that do

not have floating components. On the other hand, as a result of our specially chosen truncation rule of connected truncation, floating components only appear at intersection terms where we show a more careful control of the coefficient allows us extra slack to tolerate the increase in the spectral norm upper bound compared with that of the i.i.d. setting.

Moment matrix construction To get started, we formally define our moment matrix as given by the following moments via the connected truncation for the vanilla pseudo-calibration.

Definition 4.3.8 (Moment matrix for independent set with connected truncation)**.** *We define the moments to be*

$$\widetilde{\mathbb{E}}[x_S] := \sum_{\substack{R\subseteq\binom{n}{2} \\ R\text{connected to } S \\ |V(R)\cup S|\le D_V}} (\frac{k}{n})^{|V(R)\cup S|}\cdot(-\sqrt{\frac{p}{1-p}})^{|E(R)|}\cdot\chi_R(G)$$

for any $S\subseteq[n]$ *s.t.* $|S|\le$ dsos*. We follow the notation of* $\chi_R(G)=\prod_{e\in R}\chi_e(G)$*, and* $D_V\approx$ dsos $\log n$ *is a truncation parameter to be specified later.*

Preliminaries for PSD Analysis We refer the reader for the prior work of [JPR+22] for the full analysis while we strive to give a brief introduction to highlight the main lemmas that employ moment methods, and illustrate the necessary changes due to floating components. We start by recalling the following notations,

Definition 4.3.9 (Moment matrix in the graph matrix basis)**.** *The moment matrix defined above as a linear operator can be equivalently written as*

$$\widetilde{\Lambda}=\sum_{\alpha:\text{connected},|V(\alpha)|\le D_V}\widetilde{\lambda}_\alpha\frac{M_\alpha}{|Aut(\alpha)|}$$

where we define

$$\widetilde{\lambda}_\alpha := \left(\frac{k}{n}\right)^{|V(\alpha)|} \cdot \sqrt{\frac{p}{1-p}}^{|E(\alpha)|}$$

and $|Aut(\alpha)|$ is the size of automorphisnm group defined for shape α used to ensure each edge-set contributes once in the decomposition.

For convenience, instead of working with Λ directly, we usually work with a rescaled version of it obtained by left/right multiplying a diagonal matrix with entry being $\sqrt{\frac{n}{k}}^{|U_\alpha|}$, and we will primarily work with the rescaled coefficient defined as

$$\lambda_\alpha = \sqrt{\frac{n}{k}}^{|U_\alpha|+|V_\alpha|} = \left(\frac{k}{n}\right)^{|V(\alpha)|-|\frac{|U_\alpha|+|V_\alpha|}{2}} \cdot (-\sqrt{\frac{p}{1-p}})^{|E(\alpha)|}$$

Definition 4.3.10 (Π independent-set indicator)**.** *Let Π be a diagonal matrix for appropriate dimension for* dsos *with entries $\Pi[S,S] = 1[S \text{ is an IS in } G]$.*

At the backbone of the PSDness analysis is the notion of middle shape defined as. the following,

Definition 4.3.11 (Middle shape)**.** *τ is a middle shape if U_τ and V_τ are the Minimum-Vertex-Separator of τ, i.e., for any separator $S \subseteq V(\tau)$ that separates paths between U_τ and V_τ, we have $|S| \geq |U_\tau| = |V_\tau|$.*

Before we dig into the PSDness analysis, we first verify that this continues to give valid pseudo-moments when the underlying graph comes from random d-regular graph. We note that several proofs are in fact identical to the i.i.d. case while we re-verify them with the regular graph distribution for completeness.

4.3.2 Verification of Moment Constraints

Claim 4.3.12 (Satisfying independent set constraint)**.** *For any set S with $|S| \leq$ dsos , if S is not an independent-set in G, we have*

$$\widetilde{\mathbb{E}}[x_S] = 0$$

Proof. Note that for any set of edges R outside S, we may group together the edges inside S,

$$\begin{aligned}
\widetilde{\mathbb{E}}[x_S] &= \sum_{\substack{R \subseteq \binom{n}{2} \\ R\text{:connected to S} \\ |V(R) \cup S| \leq D_V}} (\frac{k}{n})^{|V(R)\cup S|} \cdot (-\sqrt{\frac{p}{1-p}})^{|E(R)|} \cdot \chi_R(G) \\
&= \sum_{\substack{R \subseteq \binom{n}{2} \setminus \binom{S}{2} \\ R\text{:connected to S} \\ |V(R) \cup S| \leq D_V}} (\frac{k}{n})^{|V(R)\cup S|} \cdot (-\sqrt{\frac{p}{1-p}})^{|E(R)|} \cdot \chi_R(G) \cdot \left(\sum_{H \subseteq \binom{S}{2}} (-\sqrt{\frac{p}{1-p}})^{|E(H)|} \cdot \chi_H(G) \right) \\
&= \sum_{\substack{R \subseteq \binom{n}{2} \setminus \binom{S}{2} \\ R\text{:connected to S} \\ |V(R) \cup S| \leq D_V}} (\frac{k}{n})^{|V(R)\cup S|} \cdot (-\sqrt{\frac{p}{1-p}})^{|E(R)|} \cdot \chi_R(G) \cdot \frac{1}{(1-p)^{\binom{S}{2}}} \cdot 1[S \text{is an ind-set in } G]
\end{aligned}$$

where we observe that subset of edges inside S group to be a multiple of independent-set indicator of S. □

Claim 4.3.13 (Large-value)**.** *W.h.p., the pseudo-moment matrix gives objective value $\sum_{i\in[n]} \widetilde{\mathbb{E}}[x_i] \geq (1 - o(1))k$.*

Proof. We highlight the observation that most calculations can be reduced to moment-calculations and our new norm bounds, in particular, graph matrix norm bounds for

shapes τ with $U_\tau = V_\tau$. Notice this is equivalent to be bounding

$$\sum_{i\in[n]} \left|\widetilde{\mathbb{E}}[x_i] - \frac{k}{n}\right| = \sum_i \sum_{\substack{\alpha\subseteq\binom{n}{2} \\ E(\alpha)\neq\emptyset\text{: connected to } i \\ |V(\alpha)\cup\{i\}|\leq D_V}} \left\|\widetilde{\lambda}_\alpha M_\alpha\right\|$$

for α viewed as a shape with $U_\alpha = V_\alpha = \{i\}$ and $\tilde{\lambda}_\alpha = \left(\frac{k}{n}\right)^{|V(\alpha)|} \cdot (-\sqrt{\frac{p}{1-p}})^{|E(\alpha)|}$ the unscaled shape coefficient of α. Crucially, we want to point out that in the calculation of the above, each shape α dose not containing any floating component, thus the norm bound applies in an identical way as that of the i.i.d. setting. We first switch the unscaled coefficient to the scaled coefficient of

$$\lambda_\alpha = \sqrt{\frac{n}{k}}^{(|U_\alpha|+|V_\alpha|)} \cdot \widetilde{\lambda}_\alpha$$

Since $U_\alpha = V_\alpha = \{i\}$, we have

$$\sum_i \sum_{\substack{\alpha\subseteq\binom{n}{2} \\ E(\alpha)\neq\emptyset\text{: connected to } i \\ |V(\alpha)\cup\{i\}|\leq D_V}} \left\|\tilde{\lambda}_\alpha M_\alpha\right\| \leq n \cdot \frac{k}{n} \sum_{\alpha:|U_\alpha|=|V_\alpha|,\text{connected},|V(\alpha)|\leq D_V, E(\alpha)\neq\emptyset} \|\lambda_\alpha M_\alpha\| = o(k)$$

where the last equation of $\cdot \frac{k}{n} \sum_{\alpha:|U_\alpha|=|V_\alpha|,\text{connected},|V(\alpha)|\leq D_V,E(\alpha)\neq\emptyset} \left\|\lambda_\alpha M_\alpha\right\| = o(1)$ is proven in our main lemma for PSDness concerning middle shapes in the subsequent section. □

Chapter 5

Smooth Trade-off for Tensor PCA: An Application to Hypergraph Input

5.1 Introduction and Our Results

In this work, we study the tensor principal component analysis (PCA) problem. In this setting, the input is a tensor T obtained by adding a "noise" tensor G to a symmetric rank-1 tensor (the "signal") on a boolean vector $v \in \{\pm 1\}^n$. Every entry of G is an independent centered Gaussian random variable with variance 1.

$$T = G + \lambda \cdot v^{\otimes r}.$$

The central algorithmic question is to understand the minimum λ — a measure of the *signal strength* — at which it is possible to distinguish T from the background "noise" G and, more generally, recover the planted vector v.

The first interesting case of $r = 2$ is the classical single-spiked Gaussian PCA problem, which, in addition to being the fundamental question in algorithmic statistics, has also been

of great interest in statistical physics. A celebrated result [BAP05] establishes the famous BBP *sharp* phase transition for this model: the planted signal is efficiently recoverable for any $\lambda > \frac{1}{\sqrt{n}}$[1] and informationally-theoretically impossible to recover when $\lambda < \frac{1}{\sqrt{n}}$. This phenomenon is often described as the existence of a *sharp algorithmic threshold* (the minimum strength at which the planted signal can be detected/recovered) for the PCA problem.

The case of $r > 2$ has been extensively studied as a canonical problem in statistical inference, starting with the work of Montanari and Richard [MR14]. On the one hand, the planted signal is information-theoretically detectable[2] whenever $\lambda \gtrsim n^{-\frac{r+1}{2}}$. The best known polynomial-time algorithms[HSS15, HSSS16] (which in fact run in *near-linear* time!) , however, are known to succeed only when $\lambda \gtrsim \sqrt{\log n} \cdot n^{-r/4}$. This potential gap (which arises only for $r \geq 3$) between the signal strength required for information-theoretic and efficient detection/recovery is a major topic of study in average-case algorithm design. [PWB16, ZK16, LML+17, MNS18, BBH18, JLM20]

Can the minimum signal strength be improved? Whether there are algorithms for tensor PCA that succeed at a lower signal strength has been a central question in average-case algorithm design. Indeed, investigations of this question have already led to new techniques that have profound consequences for algorithm design and beyond. About a decade ago, researchers [BGL17, BGG+16, RRS17] found higher order spectral methods that run in subexponential-time algorithms and improve the above algorithmic results. In their landmark work, Wein, Alaoui, and Moore [WAM19] introduced spectral methods based on *Kikuchi Matrices* to substantially simplify algorithms and their analyses. Kikuchi matrices have since had a deep impact on algorithm design and beyond [GKM22, HKM23,

[1] In particular, this threshold is a factor $1/2$ smaller than the spectral norm of the noise matrix G, a finding that may be surprising at first.

[2] We use $\gtrsim$ to denote an inequality that holds up to a multiplier that is an absolute constant independent of n but may depend on r.

AGKM23, KM23, KM24].

Theorem 5.1.1 ([BGL17, BGG$^+$16, RRS17, WAM19]). *For any $1 \le \ell \le n$, there is a $n^{O(\ell)}$-time algorithm for detection and recovery in tensor PCA model that succeeds with probability tending to 1 whenever*

$$\lambda \gtrsim \sqrt{\log n} \cdot n^{-r/4} \cdot \ell^{1/2-r/4},$$

for an absolute constant $C > 0$.

On the flip side, lower bounds in various restricted models provide evidence that no algorithm could improve the guarantees on λ above by more than a polylogarithmic factor[PR22, KWB19b, DH21, MR14, AGJ20].

Smooth tradeoffs: can higher polynomial times help? The best polynomial-time guarantee for tensor PCA is currently achieved by a near-linear time algorithm. Does allowing higher polynomial time help? If true, such a result establishes that there is no sharp algorithmic threshold for tensor PCA for $r > 2$ unlike the case of $r = 2$ and other well-studied statistical inference problems such as community detection in stochastic block models [DKMZ11b, Abb17b, GJJ$^+$20a, Wei22, BBK$^+$21b].

While such a smooth trade-off has been suspected by researchers for half a decade now, rigorously establishing it has proved challenging. The key technical difficulty is in the analyses of the spectral norms of highly correlated random matrices that arise in the spectral methods for the problem. Even with the simplifications offered by the Kikuchi matrices, proving sharp bounds has proven difficult. Indeed, in their original work that introduced these matrices, Wein, Alaoui, and Moore [WAM19] conjectured an improved bound on the minimum signal strength that incurs no logarithmic loss. The significance of this result and its consequences for establishing a smooth trade-off in algorithmic thresholds for tensor PCA was recently highlighted in a conjecture of Bandeira. We state

his conjecture below and refer the reader to the accompanying blogpost for a more detailed discussion of the context.

Conjecture 5.1.2 ([Ban24a, Ban24b]). *For any even r and $\ell \geq \frac{r}{2}$ and any n, let $\lambda_{r,\ell}$ be the threshold given by the spectral norm of the Level-ℓ Kikuchi matrix. For any fixed r, one has*

$$n^{r/4} \cdot \lambda_{r,\ell} \to 0$$

as $\ell \to \infty$ (crucially noting that this is after one has taken $n \to \infty$).

In addition to being a question of central interest in statistical inference, establishing such a smooth trade-off has consequences for establishing potential quantum speedups. Indeed, a recent work [SOKB25] shows a *quartic* (i.e., a factor four improvement, in contrast to the expected "Grover-search type" factor two improvement, in the exponent) speedup in estimating the top eigenvector of Kikuchi matrices for tensor PCA over the classical power iteration. Their result suggests tensor PCA as a candidate example for quartic quantum speedups provided one could establish that higher polynomial running times strictly lower the signal strength required for success of the Kikuchi spectral methods.

Spectral Norm Bound for Random Matrices as the Bottleneck. Let us describe the key technical barrier in establishing the conjecture above in more detail now. At a high-level, the main issue is that the level ℓ Kikuchi matrix defined as following has size $\Theta(n^{2\ell})$ but only $O(n^r)$ independent bits of randomness. When $\ell \gg r$ – the previously unexplored setting – this matrix has highly correlated entries.

Definition 5.1.3 (Kikuchi Matrix for Even-r. See definition 5.2.2 for the definition for odd r). *For even r, and any $\ell \in \mathbb{N}$, and given tensor G of order-r, M_ℓ is the Kikuchi matrix of size $n^{\binom{n}{\ell}} \times n^{\binom{n}{\ell}}$ for level-ℓ indexed by subsets $I, J \subseteq \binom{n}{\ell}$ with entry $M_\ell(G)[I,J] := G_{I \Delta J}$ for $|I \Delta J| = r$, and 0 otherwise..*

We usually drop the dependence on G when it is clear. For the bulk of our work, unless otherwise specified, we will be working with G being a random symmetric Gaussian tensor of order-r.

Example 5.1.4. *For indices* $I = \{v_1, v_2, v_3, v_4, v_5, v_6\}, J = \{v_1, v_2, v_3, v_4, v_7, v_8\}$*, in the case of* $r = 4$ *and* $\ell = 6$*, we have the corresponding entry of the Kikuchi matrix be*

$$M_\ell[I, J] \coloneqq G_{I \Delta J} = G_{v_5, v_6, v_7, v_8} .$$

More precisely, the analyses of the Kikuchi spectral method for tensor PCA relates a high probability estimate on the spectral norm of the Kikuchi matrix built from G to the threshold signal strength at which detection is possible as follows:

Claim 5.1.5. *For a given upper bound* $B(G)$ *on a random tensor G that holds with high probability, one has a distinguishing algorithm for*

$$\lambda \geq \Theta_r(1) \cdot \frac{B(G)}{n^{r/2} \cdot \ell^{r/2}} .$$

In the original line of work establishing the "coarse" thresholds, the spectral norm bounds for Kikuchi matrices all suffer a loss of polylog factors, yielding bounds

$$B_{folklore}(G) = O_r(1) \cdot n^{r/4} \cdot \ell^{r/4+1/2} \cdot \sqrt{\log n} ,$$

as an immediate application of black-box matrix concentration bounds (eg. Matrix Bernstein).

Let us make two comments about the above bound. First, note that the $\sqrt{\log n}$ factor dominates the effect of increasing ℓ so long as it is bounded by an absolute constant. Thus, one cannot rigorously infer a smooth trade-off in polynomial time regime of the algorithm. Secondly, the dependence of ℓ should also be tightly controlled apart from

the $\sqrt{\log n}$ factor - restricted to the dependence of ℓ, a bound of $o(\ell^{r/2})$ is necessary to establish any trade-off between ℓ and λ.

It is important to point out that controlling the dependence on the two key parameters simulataneously – $\log n$ and the ℓ – turns out to be difficult. For example, the recent work of d'Orsi and Trevisan [dT23] introduced an IharaBass-type formula that yields sharper spectral norm estimates for the basic spectral algorithm in the random k-xor setting. However, while their bound does get rid of the $\log n$ factor, their analyses, when run for larger ℓ has a worse polynomial dependence on ℓ. In retrospect, their strategy relies on analyzing the non-backtracking walk matrix and is tied to the setting where the input tensor is sparse. In contrast, our input tensor is fully dense.

Recent Progress and Barriers in Free Probability. A series of works [BBvH23, BCSvH24, BLNNvH25] introduce techniques from free probability to prove sharp bounds on correlated random matrices via improved noncommutative Khintchine inequalities. These works have used Kikuchi matrices arising in tensor PCA as a testbed for applications. In particular, a recent work [BCSvH24] makes non-trivial progress on conjecture 5.1.2.

Theorem 5.1.6 ([BCSvH24]). *For any even r and any $\frac{r}{2} \leq \ell < \frac{3r}{4}$, let G be a random symmetric tensor of order-r as defined above, and let $M_\ell(G)$ be the Kikuchi-ℓ matrix, with high probability*

$$\|M_\ell(G)\|_{sp} \leq \underbrace{O_r(1) \cdot n^{r/4}\ell^{r/4}}_{:=B_{fp}(G)} .$$

For even r (they do not analyze their algorithm for odd r that often tends to be similar but more technically challenging), they prove the conjecture for all $\ell \leq 3r/4$.

The bound above is based on non-asymptotic techniques in random matrix theory from free probability. In particular, the techniques end up establishing not only a spectral norm bound as above but also a semicircular shape of the spectrum as in Wigner random

matrices. This property is called *intrinsic freeness*. In their work[BCSvH24], the authors suggest that intrinsic freeness is likely false for the Kikuchi matrices above when $\ell \gg r$.

> (Remark 3.4) When ℓ is large compared to r, the dependence structure of M_ℓ is so strong that it is unclear whether it could be accurately modeled by (the corresponding free matrix) M_{free}.

That said, if the matrix is indeed intrinsically free, its spectrum would have a small deviation to that of M_{free}, predicting a norm bound of B_{fp}. As we will see, along the way to our main result, we confirm their suspicion and show that an additional $\sqrt{\ell}$ factor is necessary via a lower bound for the expected high trace, establishing the the matrix does not exhibit semicircle spectrum. We highlight the lower bound construction below lemma 5.2.9 with formal verification in section 5.4.

In an incomparable improvement on the above bound, a recent work of Bandeira and Nizic-Nikolac shows the following bound that removes the logarithmic factor for all ℓ but has a worse polynomial dependence on ℓ. As we discussed above, finding a proof strategy that simultaneously eliminates the logarithmic factor while preserving the right dependence on ℓ has proven challenging despite significant efforts.

Theorem 5.1.7 ([BNN25]). *In the same set-up as above except now for any $\ell \in \mathbb{N}$,*

$$\|M_\ell(G)\|_{sp} \leq \underbrace{O_r(1) \cdot n^{r/4} \cdot \ell^{r/2}}_{:=B_{pc}(G)} .$$

To summarize, the first bound $B_{fp}(G)$ establishes a smooth trade-off but only for running time regimes where $r/2 \leq \ell \leq 3r/4$. The second bound B_{pc}, on the other hand, removes the additional constraint on ℓ at the cost of a worse dependence on ℓ and does not imply a smooth trade-off between running time and signal strength for tensor PCA in polynomial time.

In this work, we prove a sharp bound on $\|M_\ell(G)\|$ (for both even and odd r) for all $\ell \ll n^{\Omega(1)}$ and as a special case, resolve conjecture 5.1.2.

5.1.1 Our Results

We are now ready to describe our main result. Firstly, we state our main result on sharp spectral norm bounds for Kikuchi matrices, and defer the formal definition of the odd-r case to the subsequent section.

Theorem 5.1.8 (Improved Spectral Norm Bound for Kikuchi Matrix for Even-r)**.** *For any even $r > 2$, and G a random symmetric tensor of order-r with entry $G_S \sim N(0,1)$ for any $S \in \binom{n}{r}$, let $M_\ell(G)$ be the level-ℓ Kikuchi matrix of G defined in definition 5.1.3, there exists some constant $\delta > 0$ such that for any $\ell \in \mathbb{N}$ such that $\ell < n^\delta$, with probability at least $1 - o_n(1)$,*

$$\|M_\ell(G)\|_{sp} \leq O_r(1) \cdot \left(\sqrt{n \cdot \ell}\right)^{r/2} \cdot \sqrt{\ell}\,.$$

where $O_r(1)$ hides constants independent of n and ℓ but possibly depending on r.

As noted before, the prior bound of[BCSvH24] worked only when $\ell \leq 3r/4$ and was established only for even r. The case of odd r has been challenging in all analyses of Kikuchi matrix method [GKM22, HKM23, AGKM23, KM23, KM24].

Theorem 5.1.9 (Improved Spectral Norm Bound for Kikuchi Matrix for Odd-r)**.** *For any odd $r > 2$, and G a random symmetric tensor of order-r in the above normalization, let $M_\ell(G)$ be the level-ℓ Kikuchi matrix of G defined in definition 5.2.2, there exists some constant $\delta > 0$ such that for any $\ell \in \mathbb{N}$ such that $\ell \leq n^\delta$, with probability at least $1 - o_n(1)$,*

$$\|M_\ell(G)\|_{sp} \leq O_r(1) \cdot \left(\sqrt{n \cdot \ell}\right)^{r}\,.$$

Remark 5.1.10. *Our bound works to $\ell = n^\delta$ for some constant $\delta > 0$ and gives stronger concentration bound that holds with error probability at most $\exp(n^\delta)$ as opposed to the polynomial tail bound from black-box non-commutative Khintchine.*

From our improved norm bounds for Kikuchi matrices, we show that the Kikuchi hierarchy gives a smooth trade-off for the Tensor PCA problem in the full polynomial-time regime, proving the conjecture of [Ban24b].

Theorem 5.1.11 (Smooth Tradeoff for Tensor PCA)**.** *There exists some constant δ such that for any $\ell \leq n^\delta$, there is an algorithm with runtime $n^{O(\ell)}$ that solves the detection problem of Tensor PCA with probability at least $1 - o_n(1)$ for*

$$\lambda > C_r \cdot n^{-r/4} \cdot \ell^{1/2-r/4}$$

for some absolute constant C_r that depends solely on r.

Concretely, there is an algorithm $f(T) : \mathbb{R}^{n^r} \to \{0,1\}$ running in time $n^{O(\ell)}$ that satisfies $\mathbf{Pr}_G[f(G) = 1] = o_n(1)$ if $\lambda = 0$ and G is a random symmetric tensor, while $\mathbf{Pr}_{G,v}[f(G + \lambda \cdot v^{\otimes r}) = 1] = 1 - o_n(1)$ if $\lambda > C_r \cdot n^{-r/4} \cdot \ell^{1/2-r/4}$ and v is a uniformly chosen boolean vector $\{\pm 1\}^n$.

Tightness of Our Bounds: Kikuchi Matrix is Not Free for $\ell \gg r$. Besides the algorithmic applications, we also give a lower bound for the trace calculation to demonstrate the tightness of our spectral norm bounds.

Lemma 5.1.12. *For any even $r > 2$, there exists some constant $\delta > 0$ such that for any $\ell, q \in \mathbb{N}$ and $\ell, q \leq n^\delta$, and G is a random symmetric tensor of order r with normalization prescribed as above,*

$$\mathbb{E}\,\mathrm{Tr}\left[(M_\ell)^{2q}\right] \geq \binom{n}{\ell} \cdot \left(\Theta_r(1) \cdot \sqrt{n^{r/2} \cdot \ell^{r/2}} \cdot \sqrt{\ell}\right)^{2q}.$$

with probability at least $1 - o_n(1)$ *where* $\Theta_r(1)$ *hides constants solely depending on* r *but independent of* n, q *and* ℓ.

Remark 5.1.13. *Since we allow* $\Theta_r(1)$ *slack in the above, this does not pose a conflict for the bound* $B_{fp}(G)$ *for the regime* $\ell < r$. *Moreover, this bound is most interesting for first picking* r *and then choosing* $\ell > C \cdot r$ *for some large constant* C.

As a by-product of our techniques, by ruling out the semicircle spectrum, this lower bound confirms the suggestion of remark 3.4 in [BCSvH24] that Kikuchi matrix of larger-ℓ is not intrinsically free. To highlight this discrepancy, observe that B_{fp} does not contain the extra factor of $\sqrt{\ell}$ in our final norm bound, while as we will show in the technical sections, this extra factor in fact admits an intuitive combinatorial interpretation. Moreover, our technique also explains why the additional factor of $\sqrt{\ell}$ is absent for the small-ℓ regime of $\ell < r$.

Furthermore, we showcase that in the case the underlying tensor drawn from i.i.d. Rademacher distribution, the above lower bound on the expected trace is sufficient to imply a lower tail bound for spectral norm.

Lemma 5.1.14. *For any even* r, *and for* G *a random symmetric tensor with each entry being i.i.d. from* $G_S \sim \{\pm 1\}$ *for* $S \in \binom{n}{r}$, *with probability at least* $1 - o_n(1)$,

$$\|M\|_{sp} \geq \Theta_r(1) \cdot \sqrt{n^{r/2} \cdot \ell^{r/2}} \cdot \sqrt{\ell}\,.$$

5.2 Technical Overview: Challenges from Hypergraph Input

In this section, we start with some technical preliminaries and set the stage for the second half of this section that serves as a technical overview of our work. We will be primarily

focusing on the even-r case to showcase our techniques. As Kikuchi matrices are the primary focus of our study, let's begin by formally introducing them. To make our technical analysis intuitive, we will also introduce the diagram perspective of the Kikuchi matrices.

5.2.1 Kikuchi Matrices: Definitions and Diagrams.

For completeness, we reiterate the definition for Kikuchi matrix for even-r.

Definition 5.2.1 (Kikuchi Matrix for Even-r)**.** *For even r, and any $\ell \in \mathbb{N}$, and given tensor G of order-r, M_ℓ is the Kikuchi matrix of size $n^{\binom{n}{\ell}} \times n^{\binom{n}{\ell}}$ for level-ℓ indexed by subsets $I, J \subseteq \binom{n}{\ell}$ with entry $M_\ell(G)[I,J] := G_{I \Delta J}$ for $|I \Delta J| = r$, and 0 otherwise..*

We usually drop the dependence on G when it is clear. For the bulk of our work, unless otherwise specified, we will be working with G being a random symmetric Gaussian tensor of order-r.

For odd r, albeit less straightforwardly, one may still define Kikuchi matrix in the following way as inspired by the "Cauchy-Schwarz" trick that is standard in this line of works.

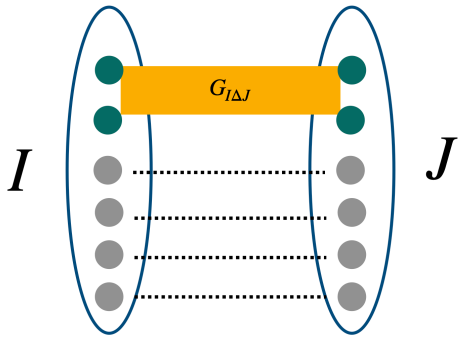
Definition 5.2.2 (Kikuchi Matrix for Odd-r)**.** *For odd r, and any $\ell \in \mathbb{N}$, and given tensor G of order-r, we let M_ℓ denote the corresponding Kikuchi matrix indexed by subsets $I, J \subseteq \binom{n}{\ell}$ with entry*

$$M_\ell[I,J] = \sum_{t \in [n] \setminus (S \cup T)} \sum_{\substack{S_1, S_2 \\ \text{partition of } S \setminus T \\ |S_1| = |S_2|}} \sum_{\substack{T_1, T_2 \\ \text{partition of } T \setminus S \\ |T_1| = |T_2|}} G_{S_1 \cup T_1 \cup \{t\}} \cdot G_{S_2 \cup T_2 \cup \{t\}}$$

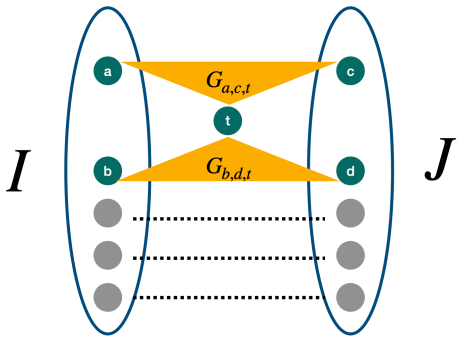
for $|I \Delta J| = 2r - 1$, and 0 otherwise.

As a sanity check, Kikuchi matrix is striaghforwardly a graph-matrix as defined in prior sections- one can view the two oval on the left and right as row and column index for the corresponding matrix entry, and each circle inside the oval as an index in $[n]$. In

this walk, we will refer to them as *step-boundaries* in the walk as they can be viewed as start or destination of some step in the walk. Moreover, we represent the underlying (hyper-)edge [3] (random variable) via the orange rectangle (in the case $r = 4$), and the dotted line connecting two vertices in the row and column index that share the same index in $[n]$, i.e. any gray vertex is a vertex from the intersection of the row and column indices.



(a) Kikuchi Matrix for $(r = 4, \ell = 6)$



(b) Kikuchi Matrix for $(r = 3, \ell = 5)$

5.2.2 Trace Moment Method and Spectral Norm Bounds

Our starting point is again the trace moment method, the classic technique from random matrix theory for understanding the spectrum of random matrices. Specializing to our setting of Kikuchi matrix for even-r and taking expectation on both sides, we have

$$\mathbb{E}_G \operatorname{Tr}\left(M_\ell^{2q}\right) = \sum_{\substack{P:\text{closed-walk of length-2q} \\ P=(S_1,S_2,\ldots,S_{2q}=v_1) \\ S_i \in \binom{n}{\ell}}} \mathbb{E}_G \left[\prod_{i\in[2q]} M_\ell[S_i, S_{i+1}]\right].$$

Since each hyper-edge corresponds to a Gaussian random variable with odd-moment being 0, one can also infer that each hyper-edge must appear for an even number of times

[3] With some abuse of notation, we will not distinguish between hyperedge and edge in our technical analysis.

in the walk (i.e. used by an even number of steps). Therefore, any closed walk with non-zero contribution in the summand admits the following combinatorial description.

Definition 5.2.3 (Trace-walk for even-r)**.** *A trace-walk P for Kikuchi matrix of level-ℓ at length-$2q+1$ is a sequence of ℓ-sized subsets $S = \{S_t\}_{t\in[2q+1]}$ such that*

1. *$S_i \in \binom{n}{\ell}$ for any $i \in [2q]$;*
2. *$|S_i \Delta S_{i+1}| = r$ for any $i \in [2q]$, and the i-th step is along the hyperedge (equivalently viewed as the random variable) $G_{S_i \Delta S_{i+1}}$;*
3. *(Closed-walk) $S_1 = S_{2q+1}$;*
4. *(Evenness) Each hyper-edge appears in an even number of steps.*

Equivalently, we view it as a walk of length-$(2q)$ such that at each step-t, we move from the left boundary $U_t = S_t$ to the right boundary $V_t = S_{t+1} = U_{t+1}$.

For intuition, the following example is a snippet of a trace-walk restricted to the first 3 steps for $r = 4$. This example also highlights a property of working with matrix indices being subsets as opposed to order-tuples-adjacent steps may not be incident to any common vertex. It prompts us to make the following definition for the each step of the walk.

Definition 5.2.4 (Vertex Status for a Given Step)**.** *For any step in the trace-walk in which we move from $U_t = S_t \in \binom{n}{\ell}$ to $V_t = S_{t+1} \in \binom{n}{\ell}$, we call*

1. *each vertex in $S_{t+1} \cap S_t = U_t \cap V_t$* dormant*;*
2. *each vertex in $S_{t+1} \Delta S_t = U_t \Delta V_t$* active*.*

In particular, we additionally call each active vertex in $S_{t+1} \setminus S_t$ incoming-active*, and each vertex $S_t \setminus S_{t+1}$* outgoing-active*. In short, we also refer to each such vertex as "incoming" and "outgoing".*

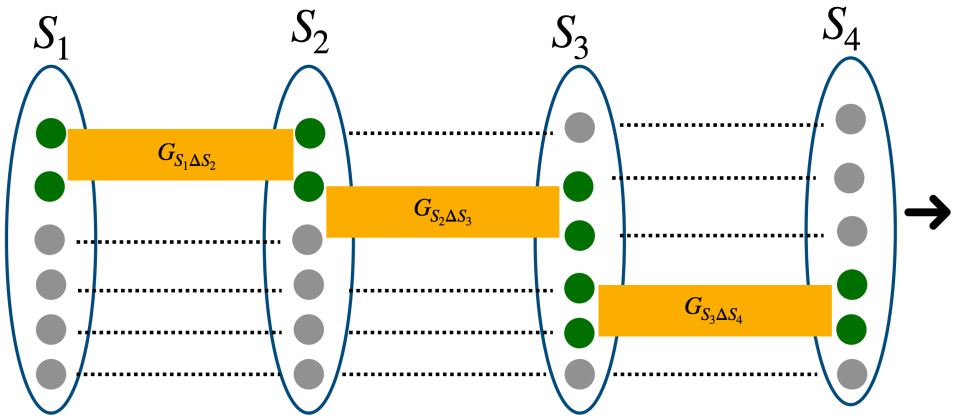


Figure 5.2: Snippet of a Trace-Walk

In the above diagram, each gray vertex is a "dormant" vertex in both of the adjacent steps, while a vertex is green if it is "active" in either of the adjacent steps.

Additionally, for each walk P, we additionally refer to its corresponding expected value as walk-value as the following,

Definition 5.2.5 (Walk-value). *For each walk $P = \{S_1, S_2, \ldots, S_{2q-1}, S_{2q} = S_1\}$, we define its walk-value as*

$$val(P) := \mathbb{E}_G \left[\prod_{i \in [2q]} M_\ell[S_i, S_{i+1}] \right] = \prod_{e=(I,J) \in E(P)} \mathbb{E}_G[(G_e)^{mul_P(e)}] .$$

And a crucial observation at this point is that there are two kinds of factors one needs to control when trying to upper bound the expected trace,

1. Combinatorial (Vertex) Factor: this keeps track of the factor that arises from counting the walks (i.e., the cost of identifying each vertex's label in $[n]$ throughout the trace-walk).

2. Analytical (Edge-value) Factor: this corresponds to the walk-value, i.e., the expectation of random variables traversed through the walk.

Recovering the Folklore Bounds for Rademacher Tensor. To shed light on the local factor-assignment scheme that we will be working with, we start with a lightweight application in this fashion that allows us to recover the vanilla bound of $B_{folkdlore}(G) = O_r(1) \cdot n^{r/4} \cdot \ell^{r/4+1/2} \cdot \sqrt{\log n}$. Undoubtedly, this bound can be readily obtained by a straightforward application of anyone's favorite matrix concentration bound, while we resort to the heavy machinery of the trace moment method with the hope that we may identify what needs to be improved when we seek a tighter bound that shaves off the log factor.

Claim 5.2.6. *For any even-$r > 3$ and $\ell \in \mathbb{N}$, and G a random symmetric tensor such that $G_S \sim \{\pm 1\}$ i.i.d., let $M := M_\ell(G)$ be the associated Kikuchi matrix, and any $q = \Theta(\ell \log n)$, with probability at least $1 - o_n(1)$,*

$$\|M\|_{sp} \leq O(1) \cdot \sqrt{\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2}} \cdot \sqrt{\ell} \cdot \sqrt{\log n}\,.$$

Before we describe the precise factor-assignment scheme, we make the following observation. Since we are working with Rademacher random variables as opposed to Gaussians, one convenient simplification is that we can effective ignore the analytical factor that comes from the walk-value: for any walk P in which each hyperedge is traversed an even number of times, we have

$$\mathbb{E}_G[\mathrm{val}(P)] = \mathbb{E}_G[\prod_{e \in E(P)} (G_e)^{mul_P(e)}] = 1\,,$$

and 0 otherwise where $mul_P(e)$ is the number of times edge e appears walk P, i.e., the number of times the walk traverses along the edge e. From now on, it suffices for us to focus on the combinatorial factor that arises from walk-counting.

We next observe the following. Given any index $S \in \binom{n}{\ell}$, it has at most $\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2}$ neighbors in the Kikuchi matrices. Therefore, at time-t in the walk in which we move

from S_t to S_{t+1}, we have at most $\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2}$ choices for identifying the hyper-edge, and equivalently, the destination S_{t+1}. Moreover, in the case that the hyper-edge has been traversed in some previous step of the walk, such an (hyper-)edge can be more efficiently specified at a cost of at most $2q$. This culminates in the following bound that accounts for the "global" contribution of a hyper-edge to the combinatorial counting,

Proposition 5.2.7. *In the global combinatorial factor of the walk, for a hyper-edge e appearing in the walk for $mul_P(e)$ times, it incurs a total cost of at most $B_{global}(G_e) \leq \binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2} \cdot (2q)^{mul_P(e)-1}$.*

With the above observation, we are now ready to describe the factor-assignment scheme that assigns the factors of each hyper-edge to individual steps that traverse along the particular hyper-edge.

(Combinatorial) Factor Assignment Scheme for Hyper-edges

Each hyper-edge appears at least twice in the walk, and the first time it appears, a cost of $\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2}$ is sufficient, and a cost of $2q$ is sufficient for any subsequent appearance.

1. Assign a factor of $\sqrt{\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2}} \cdot \sqrt{2q}$ for the first and the last[a] step in which the hyper-edge appears in the walk;
2. Assign a factor of $2q$ for any step in which the edge is making a middle (non-first/last) appearance.

[a] equivalently, the final

We start by showing that this is a complete scheme, and each (combinatorial counting) factor in the global bound is accounted for via the local assignment.

Proposition 5.2.8. *For any hyper-edge e that appears in the walk, let $B_{local}(e)$ be the total factor assigned* by the above scheme *to this hyper-edge e throughout the walk across its various appearances, we have*

$$B_{local}(e) \geq B_{global}(e) .$$

Proof. By our assignment-scehme, we have

$$B_{local}(e) = \left(\underbrace{\sqrt{\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2} \cdot 2q}}_{\text{First}} \cdot \underbrace{\sqrt{\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2} \cdot 2q}}_{\text{Last}} \right) \cdot \underbrace{(2q)^{mul_P(e)-2}}_{\text{Middle}} .$$

Since each hyper-edge appears at least twice in a contributing walk, we have $\text{mul}_P(e) \geq 2$. And notice that in the base case of $mul_P(e) = 2$, the inequality holds as $B_{global}(e) \leq \binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2} \cdot (2q)$. Moreover, any increment in $mul_P(e)$ gives a factor of $2q$ to the both sides. This completes the proof to our proposition. □

This then allows us to complete the naive bound with $\sqrt{\ell \log n}$-factor for Kikuchi matrix of random Rademacher tensor.

Proof to claim 5.2.6 . Since each step uses exactly 1 hyper-edge, for $\alpha \in \{F, H, L\}$, let $B_q(\alpha)$ be the local bound for each individual step using a hyper-edge that appears for first (F), middle (H), and last[4] time (L). By construction of the above scheme, we have

$$B_q(F) = B_q(L) = \sqrt{\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2} \cdot 2q} ,$$

and

$$B_q(H) = 2q = o_n(1) \cdot (B(F) + B(L)) .$$

provided $q \ll \binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2}$.

Summing over all possible choices for a given step gives us a local bound of

$$B_q(M_\ell) \leq B_q(F) + B_q(H) + B_q(L) = (2 + o_n(1)) \sqrt{\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2} \cdot 2q} .$$

[4] equivalently, the final time

By construction of our factor assignment scheme, this yields

$$\mathbb{E}[\mathrm{Tr}\left(M_\ell^{2q}\right)] \leq n^\ell \cdot (B_q)^{2q}\,.$$

Finally, plugging the chosen value of $q = \Theta(\ell \cdot \log n)$, taking $1/2q$-th root of the above and applying Markov's then gives us a norm bound that holds with probability $1 - o_n(1)$ of

$$\begin{aligned}\|M_\ell\|_{sp} \leq (1+\epsilon) \cdot B_q &= O(1) \cdot \sqrt{\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2} \cdot 2q} \\ &= O(1)\sqrt{\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2}} \cdot \sqrt{\ell} \cdot \sqrt{\log n}\,.\end{aligned}$$

□

Let us now pause to examine a key bottleneck in the above scheme: where exactly does the extra polylog factor arise in the global accounting? The culprit is the factor of $2q$ incurred when an edge appears for the second (and the last) time. For ease of the discussion, we call it the *last-cost*. To understand why it suffices to focus on cases where the second and the last appearances coincide, observe that any other non-first appearance is treated as a middle appearance in the schemeand such steps are assigned only negligible local contributions, recalling $B(H) = o(1) \cdot B(F)$ (and analogously $B(L)$).

For readers familiar with the trace moment calculation for G.O.E. matrix, it is well-anticipated that the last-cost can be improved: in fact, it may be just 1 as in the case of G.O.E. or graph adjacency matrix. This is almost correct except that is is also slightly over-optimistic, and it turns out that we are in an intriguing in-between regime: the cost can be improved while the last-cost would not just be 1 either!

Identifying the Target Upper Bound from a Lower Bound. To start with, notice that suppose last-cost is $O_r(1)$, we would obtain final bound of $\|M_\ell\|_{sp} \leq O_r(1) \cdot \sqrt{\binom{n-\ell}{r/2} \cdot \binom{\ell}{r/2}}$. To rule out this hypothetical bound in the framework of trace moment method, it is instructive to take a detour into the lower bound and consider the corresponding lower bound for the expected high trace.

Lemma 5.2.9 (Lower Bounding the High Trace)**.** *A factor of $\Theta_r(\ell)$ is necessary for the Last-Cost. As a result,*

$$\mathbb{E}\,\mathrm{Tr}\left(M_\ell^{2q}\right) \geq \textit{Matrix-Dimension} \cdot \left(\Theta_r(1) \cdot \sqrt{\binom{n-\ell}{r/2}\binom{\ell}{r/2}} \cdot \sqrt{\ell}\right)^{2q}$$

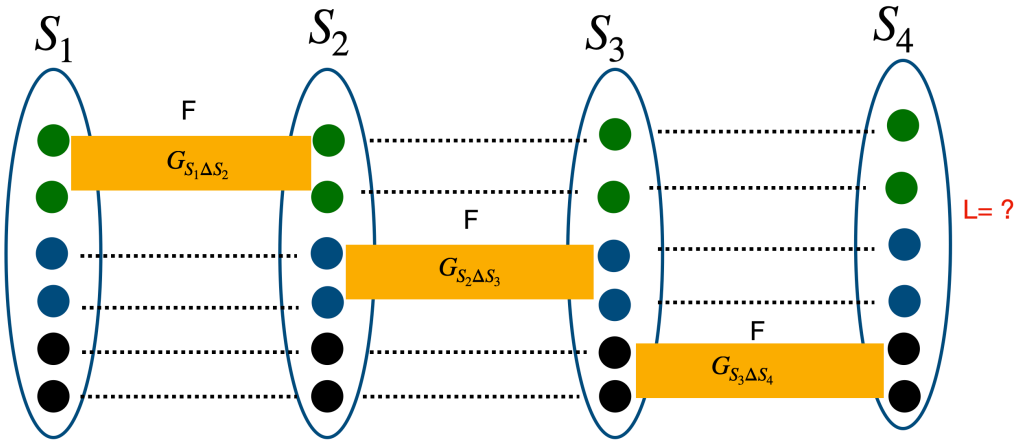


Figure 5.3: Confusion about the Upcoming L-Step

We now give a diagram explanation for the extra factor ℓ in the last-cost. For simplicity, assume r is even and $r/2$ divides ℓ. Lett us consider the first 3-steps of some walk as described in the above diagram,

Question 5.2.10. *Suppose that we are at the index S_4 and we are to use an edge for the second (and last) time in the next step, which edge could we be using? Equivalently, what could S_5 be if this is an L-step?*

It's fairly straightforward to see that in the above example, any edge among the first three edges can be making a second (and last) appearance in the next step. And more generally, for larger ℓ, this reveals that we would have a last-cost of $\Theta(\frac{\ell}{r/2}) = \Theta_r(\ell)$ at some particular step. Moreover, this bound in fact holds for a typical step along edges appearing for the second (and the last) time. Observe that we have a total last-cost for $t := \frac{\ell}{r/2}$ edges being $\frac{\ell}{r/2}!$ if we arrange them in the manner of 1) first using $\frac{\ell}{r/2}$ new edges, and 2) subsequently re-use all of the $\frac{\ell}{r/2}$ new edges to return to the start yet in a potentially different ordering. By Stirling approximation, this gives an individual bound of$(t!)^{1/t} = \Theta(1) \cdot t = \Theta_r(\ell)$. We defer the formal verification of the lower bound to section 5.4 .

Zooming back, the lower bound has identified a clear target for the upper bound, that is showing a cost of ℓ for each step we are using an edge appearing for the second-and-last time in the walk.

When is ℓ Sufficient for last-cost? It'd be straightforward if one can establish a worst-case bound of last-cost being ℓ for any step of any walk. However, that is false. However, it turns out that at least in the ideal situation below, one can show that ℓ is indeed sufficient.

Definition 5.2.11 (Ideal Last-Step)**.** *Suppose at some step-t in the walk, we are walking from S_t to S_{t+1} along some edge that is promised to be making its last-appearance in the walk, we call the step from S_t to S_{t+1} an ideal last-step if* some out-going vertex in $S_t \setminus S_{t+1} \subseteq [n]$ is appearing for the final time in the walk, i.e., it is not to-be-revisited upon the departure from S_t.

Let's now see why this is sufficient to guarantee a cost ℓ for specifying the destination of S_{t+1} from S_t. Note that throughout the walk, we may can easily keep track of which of the "seen" edges so far in the walk have made their final appearances by incurring a $O(1)$ bound per-step,

Observation 5.2.12. *As the walk proceeds, it is a cost of* 2 *per-step to identify whether the edge being traveled along at the given step is appearing for the final time.*

Therefore, in an ideal last-step, it suffices for us to identify one of the outgoing vertices from S_t that is not-to-be revisited as guaranteed by our assumption. This incurs a cost of ℓ since S_t is an ℓ-sized subset. Once such a vertex is identified, the edge used in the next step to move to S_{t+1} is also identified since any such vertex must be incident to a unique edge that has not made their final appearance.

We have now shown that the cost of ℓ is sufficient under the ideal condition. However, a crucial question remains: why is it sufficient to bound the contribution in the ideal case alone? Recall that, in the worst case, the length of the walk can lead to a bound of $2q$, which we aim to avoid.

Final Strategy. We begin by revisiting the factor-assignment scheme used in the vanilla analysis, which assigns a $2q$-factor to steps along edges appearing for the middle time. However, such middle-time edges can be effectively ignored, as they contribute only lower-order terms compared to edges appearing for the first or final time. This observation motivates our final strategy: to design a new factor-assignment scheme that penalizes last-steps failing to meet the ideal condition. To this end, we go beyond the earlier edge-based approachwhich uniformly assigned factors to all L-steps regardless of the underlying edgeand instead decompose the edge-based cost into vertex-level contributions. This shift to a vertex-based analysis provides the flexibility to prioritize vertices appearing for the final time, guiding the scheme to favor steps aligned with the ideal condition. We formally introduce this new scheme in the next section.

5.3 Sharp Norm Bounds from Factor Assignment Scheme

5.3.1 Preliminaries for Trace-Walk

To facilitate the discussion, we begin by reintroducing several key definitions. While some of these were previously presented in the technical overview for the special case of even r, we now restate them in a more general and formal form to ensure that our arguments can be swiftly extended to the case of odd r as well. For convenience, we will highlight the distinction each time a definition is reintroduced and generalized.

For starters, let's reintroduce the definition of trace-walks which would immediately apply to trace walks of Kikuchi matrices for odd-r as well to incorporate vertices at each step of the walk getting summed over while outside the left/right boundaries. For readability purpose, we suggest the reader to focus on the even-r case in the first pass and simply adhere to the simplified definition introduced in definition 5.2.3.

Definition 5.3.1 (Generalized Definitions of Trace-walk)**.** *For any $r > 2$, A trace-walk for Kikuchi matrix of level-ℓ at length-$2q$ is a sequence of sets $(S_1, W_1, S_2, W_2, \ldots, S_{2q}, W_{2q}, S_{2q+1})$ such that*

1. *Walk-boundary sets: $S_i \in \binom{n}{\ell}$ for any $i \in [2q+1]$;*
2. *Intermediate sets : $W_i \subseteq [n]$ for any $i \in [2q]$, and additionally, we assume W_i to come with an two equal-partitions U_A, U_B for $S_i \setminus S_{i+1}$ and V_A, V_B for $S_i \setminus S_{i+1}$;*
3. *For even-r, $|S_i \Delta S_{i+1}| = r$ for any $i \in [2q]$ and $W_i = \emptyset$ for any $i \in [2q]$;*
4. *For odd-r, $|S_i \Delta S_{i+1}| = 2r - 1$, and $|W_i| = 1$ for any $i \in [2q]$;*
5. *(Closed-walk) $S_1 = S_{2q+1}$;*
6. *(Evenness) Each hyper-edge appears in even number of steps.*

Equivalently, we view it as a walk of length-$(2q)$ such that

1. *at each step-$t \in [2q]$, we move from the left boundary $U_t = S_t$ to the right boundary $V_t = S_{t+1} = U_{t+1}$ with potential intermediate-set W_t;*
2. *For even-r, each step travers along exactly 1 hyper-edge given by $S_t \Delta S_{t+1}$;*
3. *for the odd-r, each step travers along 2 hyper-edges given by $U_A \cup V_A \cup W_t$ and $U_B \cup V_B \cup W_t$ where we assume that U_A, U_B are equal partitions for $S_t \setminus S_{t+1}$ and V_A, V_B for $S_{t+1} \setminus S_t$ as prescribed by the intermediate set.*
4. *Each hyper-edge appears for an even number of times.*

With the generalized definition of trace-walk, we may now relate the expected trace of the Kikuchi matrix to the weighted sum of trace-walks as defined above.

Proposition 5.3.2. *For any $r \geq 3, \ell \in \mathbb{N}$ and any $q \in \mathbb{N}$,*

$$\mathbb{E}[\mathrm{Tr}(M_\ell)^{2q}] \leq \sum_{P:\textit{trace-walk of length } 2q} \mathit{val}(P)\,.$$

where we recall that

$$\mathit{val}(P) := \mathbb{E}_G \left[\prod_{i \in [2q]} M_\ell[S_i, S_{i+1}] \right] = \prod_{e=(I,J) \in E(P)} \mathbb{E}_G[(G_e)^{mul_P(e)}]$$

We consider the following factor-assignment scheme. For starters, we focus on the "combinatorial" factor that arises from counting of the walk, and a crucial observation to get us started is to split the cost of specifying random variables (viewed as hyper-edges) among vertices, in particular, among the incoming active vertices.

Definition 5.3.3 (Generalized Vertex Status)**.** *For any step in the trace-walk in which we move from $U_t = S_t \in \binom{n}{\ell}$ to $V_t = S_{t+1} \in \binom{n}{\ell}$ potentially through intermediate vertex W_t, we call*

1. *each vertex in* $S_{t+1} \cap S_t = U_t \cap V_t$ dormant*;*

2. *each vertex in* $(S_{t+1} \Delta S_t) \cup W_t = (U_t \Delta V_t) \cup W_t$ active*.*

In particular, we call each active vertex outside S_t incoming-active *(i.e. vertices from* $(W_t \cup V_t) \setminus U_t$ *, and each active vertex from* S_t outgoing-active *(i.e., vertices from* $U_t \setminus V_t$*). In short, we also refer to each such vertex as "incoming" and "outgoing".*

As a remark, each vertex from the intermediate set W_t is considered active at the corresponding step by the above definition. With this definition in hand, one can observe that each random variable (viewed as a hyper-edge) only depends on the active-vertices while independent of the dormant vertices. And this is precisely why we call them "active" and "dormant".

Definition 5.3.4 (Vertex-appearance at a given step)**.** *For any step-t in a trace-walk in which we move from* $U_t := S_t \in \binom{n}{\ell}$ *to* $V_t := S_{t+1} \in \binom{n}{\ell}$*, we call a vertex in* $S_t \cup S_{t+1} \cup W_t$ *making an* appearance *at step-t. Moreover,*

1. *a vertex* $v \in S_t \cup S_{t+1} \cup W_t$ *with label in* $[n]$ *is making a* first *appearance at step-t if there is no* $t' < t$ *such that any vertex that makes appearance at step-*t' *has the same label in* $[n]$*;*

2. *a vertex* $v \in S_t \cup S_{t+1} \cup W_t$ *with label in* $[n]$ *is making a* last *appearance if there is no* $t' > t$ *such that any vertex that makes appearance at step-*t' *has the same label in* $[n]$*;*

3. *a vertex* $v \in S_t \cup S_{t+1} \cup W_t$ *with label in* $[n]$ *is making a* middle *appearance at step-t if it is making neither a first nor middle appearance.*

When the step-t is clear, we ignore the dependence of the appearance at t and simply call it a vertex making appearance.

As a sanity check, we observe that following the above definition, at each step-$t > 1$ from $S_t = U_t$ to $S_{t+1} = V_t$ (possibly through some intermediate set W_t, a vertex in the

(left) boundary-set S_t makes an appearance in step-t as a vertex from U_t, and also an appearance in the previous step-$(t-1)$ as a vertex from $V_{t-1} = S_t$. Analogously, vertices in the right-boundary set of step-t $S_{t+1} = V_t$ also make appearances in both step t and step $t+1$.

Proposition 5.3.5. *By paying a one-time cost of n^ℓ throughout the walk, we can assume the following: at each step-t from $S_t = U_t$ to $V_t = S_{t+1}$, no vertex in U_t is making a first appearance, and no vertex in V_t is making a last appearance.*

Remark 5.3.6. *This is immediate for any step except the left-boundary of the first and the right-boundary of the final step of the walk following the above discussion. For the special set S_1 being the start and final destination of the walk, we pay a cost of n^ℓ up front which would then allow us to assume they are not making "first" appearance in step-1 for local accounting purpose (and similarly "last" appearance in step-2q).*

Analogously, we extend the definition of "appearance" for vertices to hyper-edges, and thereby steps. The partition of steps according to the step-status will later on be the cornerstone of our local perspective in the factor-assignment scheme.

Definition 5.3.7 (Step-Status/Edge-Appearance)**.** *We consider the hyper-edge (or hyoeredges in the case of odd-r) being traversed along at step-t as making an appearance at the particular step. Moreover, a hyper-edge is making* first *appearance if it does not appear in previous steps; and analogously it is making a* final *appearance if it does not appear in subsequent steps.*

For each step, we define its status based on the edge-appearance of the edge(s) in the current step. In particular, for even-r, a step-status is equivalent to the edge-appearance of the edge it travers along, and takes a label in $\{F, H, L\}$. For odd-r in which each step is composed of 2 edges, a step-status is a label in $\{F, H, L\} \times \{F, H, L\}$.

With these perliminaries settled, we are ready to describe our final factor-assignment scheme. In the subsequent subsection, we describe the more lightweight component

concerning edge-values, and then we describe the most technical component of our work concerning combinatorial-factor in the subsection after.

5.3.2 Edge-Value Assignment

Recall that in our toy-example showcased in the previous section for $\{\pm 1\}$ random input, each walk gives expected value 0 or 1 through the defined quantity of walk-value $\mathrm{val}(P)$. However, this is no longer true when we consider Gaussian input, more generally, inputs with higher moments. That said, it can be readily handled via an additional component - edge-value assignment - in our factor assignment scheme.

Fact 5.3.8. *For $g \in \mathbb{N}(0,1)$ (and $\{\pm 1\}$), we have $E[g] = 0$, and $E[g^2] = 1$, and moreover, for $t > 2$,*

$$|\mathbb{E}[g^t]| \leq (2t)^{t/2}\,.$$

Given the higher-moment bound for edges, we consider the following edge-value assignment to steps,

Edge -Value Assignment

Each edge (random variable g_e that appears for t times in the walks contributes an analytical value of $|\mathbb{E}[(g_e)^t]| \leq (2t)^{t/2}$, and we assign this value to each individual step that traverses along edge e via factor $B_{ev}(i,e)$-the factor assigned to step-i from edge-e- as following.

1. For each step i in which the edge e is making the first or final appearance (F/L) , we assign a factor $B_{ev}(i,e) = 1$;

2. For each step i in which the edge e is making a middle appearance (H), we assign a factor of $B_{ev}(i,e) = 2 \cdot 2q = 4q$.

Definition 5.3.9 (Edge-value Bound for Steps)**.** *For each step-i, we define*

$$B_{ev}(i) := \prod_{\substack{e \\ \text{edge appearing in step-}i}} B_{ev}(i,e)$$

where $B_{ev}(i,e)$ is the factor assigned to step-i from edge-e in the above scheme.

We can now verify that all the analytical factors throughout the global walk is accounted locally across steps in the walk from the perspective of each edge e that appears in the walk.

Claim 5.3.10 (Edge-value assignment is complete)**.** *For each edge $e \in \binom{n}{r}$, let $\tilde{B}_{ev}(e)$ be the factor assigned to it across its appearances throughout the walk in the above scheme, i.e., $B_{ev}(e) := \prod_{\substack{i \in [2q] \\ e \text{ appears in step-}i}} B(i,e)$, we have*

$$\tilde{B}_{ev}(e) \geq |\mathbb{E}[(g_e)^t]| \,.$$

Proof. This is immediate for edges that make 2 appearances, and edges appearing only once (or more generally odd number of times) is zero. For number of appearances $t > 2$ with, we have

$$B_{ev}(e) = (2 \cdot 2q)^{t-2} \geq (2t)^{t/2}$$

as we observe that $t \leq 2q$ trivially, and verify that $x^{t-2} \geq x^{t/2}$ for $t \geq 4$. Finally, the claim is proven as we recall that

$$|\mathbb{E}[g^t]| \leq (2t)^{t/2}$$

for $t > 2$ for $g \in \mathbb{N}(0,1)$ and $\{\pm 1\}$, and noting that for $t = 3$ the RHS is trivially 0 while the LHS is $2q$. □

From the above, it is straightforward to verify that we can upper bound the global

walk-value as following once we switch to a step-perspective. This concludes our edge-value component of the factor-assignment scheme.

Proposition 5.3.11 (Verification of Global Value Bound from Steps)**.** *For any walk P, recall that $val(P) := \mathbb{E}_G\left[\prod_{i\in[2q]} M_\ell[S_i, S_{i+1}]\right] = \prod_{e=(I,J)\in E(P)} \mathbb{E}_G[(G_e)^{mul_P(e)}]$, we have*

$$val(P) \leq \prod_{e\in E(P)} \tilde{B}_{ev}(e) = \prod_{\substack{\text{step-}i \\ i\in[2q]}} B_{ev}(i)\,.$$

where we emphasize $\tilde{B}_{ev}(e)$ is the accounting from edge-perspective for edge e, and $B_{ev}(i)$ is the accounting from the step-perspective for step-i.

5.3.3 Combinatorial-Factor Assignment via Vertices

To identify each step, it is sufficient to identify the following for each edge appearing for the first time,

1. What are the *outgoing* active vertices from the current boundary? This is a cost of ℓ for each such vertex.

2. What are the *incoming* active vertices? Each of them could take a cost of $[n]$ if they have not been seen, or a cost of $O(q \cdot r)$ otherwise.

On the other hand, for any edge that has been seen, at most a cost of $O_r(q)$ is sufficient while we note that it can be improved in the ideal-last-step situation to be $O_r(\ell)$. Motivated by the above observation, we design a factor-assignment for each of the above factors.

Factor of Outgoing Active Vertices

For starters, we note that the factor regarding *outgoing* active vertices can be rerouted and split over the edge's first and last appearance by the following scheme,

Combinatorial Factor Assignment for Factors of Outgoing Active Vertex in ℓ

For each edge appearing for the first time, it incurs a factor of $\ell^{\lfloor r/2 \rfloor}$ from identifying the (incidental) outgoing active vertices. We assign them via the following,

1. For the step in which the edge first appears (F), assign a factor of $\sqrt{\ell}$ for each of its incident *incoming* active vertex on the step-boundary;
2. For the step in which the edge appears for the last time (L), assign a factor of $\sqrt{\ell}$ for each of its incident *outgoing* active vertex on the step-boundary;

Remark 5.3.12. *Crucially, notice that no $\sqrt{\ell}$ factor is assigned to the intermediate vertex W for the odd-r case as the vertex is not on the step boundary!*

Since each edge is incident to the same number $\lfloor r/2 \rfloor$ of incoming and outgoing active vertices on the boundary by definition of our matrices, it is straightforward to observe that we indeed pick up a total of $2 \cdot \lfloor r/2 \rfloor$ many $\sqrt{\ell}$ factors for each hyperedge across its first and last appearance. That is a total of $\sqrt{\ell}^{2 \cdot \lfloor r/2 \rfloor} = \ell^{\lfloor r/2 \rfloor}$ which is precisely the combinatorial factor incurred for identifying the outgoing active vertices for each edge when it appears for the first time!

Factor of Incoming Active Vertices

Next, we focus on the factor from incoming vertex for each edge that appears for the first time. In fact, our assignment will be more general that allows us to handle factor for edges making a middle (non-first/last) appearance and edges making last appearance but not in ideal-condition altogether , as they both involved factors in $O_r(q)$ as well.

Combinatorial Factor Assignment for Vertex Factor in n and $O(q)$

Each vertex requires a factor n to be specified when it first appears in the walk, and a subsequent factor of $O(q)$ when it appears as an incoming active-vertex of some F

edge in the walk. We assign its corresponding factor as the following,

1. Assign a factor of $\sqrt{n}$ for the step if the vertex is appearing for the first time;
2. Assign a factor of $\sqrt{n}$ for the step if the vertex is appearing for the last (final) time;
3. Assign a factor of $O(q \cdot r)$ if the vertex is making a middle appearance as an *incoming* active vertex.

Moreover, for completeness, for vertices incident to some H-edge, i.e. edge appearing for the middle time, or an L-edge but not in ideal-step condition, assign a total cost of $O_r(q)$ to all the (incidental) incoming active vertices.

The crux of the above scheme is that we redistribute the factor of n for a vertex incurred up front when it first appears among its first and last appearance, so that we attain the usual square-root saving that we anticipate for the spectral norm bound.

Ideal last-cost, and Redistribution With the bounds for vertices incident to some edge making the first appearances, we may now focus on the last-cost, the cost of specifying an edge appearing for the last time. We formally restate the ideal last-cost bound as highlighted in the overview, and show that it can be analogously extended to the odd-r case in which we use a *single* factor of ℓ to identify 2 edges simultaneously in the analogous ideal-condition.

For convenience, we recall the condition for an ideal last-step.

Definition 5.3.13 (Ideal Last-Step for even-r)**.** *Suppose at some step-t in the walk, we are walking from S_t to S_{t+1} along some edge that is promised to be making the last-appearance in the walk, we call the step from S_t to S_{t+1} an ideal last-step if* some out-going vertex in $S_t \setminus S_{t+1} \subseteq [n]$ is appearing for the final time in the walk, i.e., it is not to-be-revisited upon the departure

from S_t.

We now re-state our observation earlier, that for any fixed vertex, if it is appearing for the last time in the boundary, its incident edge can be effortlessly specified.

Claim 5.3.14 (Final-Appearance Edges is fixed for any *given* Final-Appearance Vertex)**.** *Given any vertex with label in $[n]$ at step-t via some edge making its final appearance, if the vertex is additionally making its final-appearance, the edge is fixed (i.e., there is a unique edge).*

Proof. Throughout the walk, we can keep track of the edges that have appeared yet not made its final appearance. To do this, for each subsequent appearance of an edge, one may use a cost of 2 to identify whether this is the final appearance.

With this record, suppose there is any "outgoing"-active vertex making its final appearance in U_t, we observe that such candidate "final"-appearance edge is unique given the "outgoing"-active vertex. □

Corollary 5.3.15. *For even-r, from the U_t-boundary at step-t, a cost of ℓ is sufficient to specify a final-appearance edge provided some "out-going" vertex of the edge is making its final appearance.*

Let's now extend the above to the odd-r case in which each step now contains 2-edges by definition of our Kikuchi matrix. We first modify the ideal last-step condition as the following.

Definition 5.3.16 (Ideal Last-Step for odd-r)**.** *For any odd r, suppose at some step-t in the walk, we are walking from S_t to S_{t+1} along some edges via some intermediate vertex W_t, we call the step from S_t to S_{t+1} an ideal last-step if*

1. some out-going vertex in $S_t \setminus S_{t+1} \subseteq [n]$ is appearing for the final time in the walk, i.e., it is not to-be-revisited upon the departure from S_t;
2. *the intermediate vertex W_t (incident to both edges) is appearing for the final time in the current step-t, i.e., it is not to be revisited once we arrive at S_{t+1}.*

Note that for odd-r, the only distinction in the definition of an ideal last-step is that we additionally impose the constraint that the intermediate vertex W_t is also making a final appearance. As shown below, this condition is crucial for us to identify both edges of a single-step via a single factor of $O_r(\ell)$.

Proposition 5.3.17. *For odd-r, a cost of $\ell \cdot r = O_r(1) \cdot \ell$ is sufficient to identify both edges in an ideal last-step.*

Proof. This follows by a double application of claim 5.3.14 as the following. Notice that by construction of our matrix, and thereby definition of our walks, the intermediate vertex is incident to both edges. Therefore, it suffices for us to specify the first edge via a cost of ℓ as in the even-r case via applying claim 5.3.14 on some out-going vertex making its final appearance. Once the edge is identified, the intermediate vertex can be specified again by a cost of r. Finally, apply claim 5.3.14 again but on the specified intermediate vertex identifies to us the second edge that is making its final appearance. This incurs a total cost of $r \cdot \ell = O_r(\ell)$. □

Finally, analogously to splitting the factor of $q = \ell \log n$ equally to the first and last step in which an edge appears, we redistribute the factor of $O_r(\ell)$ from the ideal last-step. Towards this end, we first observe the following bound for number of ideal steps,

Proposition 5.3.18. *There can be at most q ideal last-steps in a length-$2q$ walk.*

Proof. This bound is immediate for even-r as each edge needs to appear at least twice in the walk, therefore, at most half of the steps can be last-steps. To extend to the odd case, observe that we would have used at least $2q + 2$ distinct edges if there are $\geq q + 1$ ideal last steps. However, we use at most $4q$ edges (counting multiplicities) in a length-$2q$ walk, and each distinct edge needs to appear at least twice, leading to a contradiction. □

We may now "naively" split the factor of ℓ as following. It should be emphasized that the cost of $O_r(\sqrt{\ell})$ is assigned to the whole step as opposed to an edge alone. This does

not make a distinction for the even-r case, while importantly for odd-r, each step of *two edges* gets assigned a total cost of $O_r(\sqrt{\ell})$.

Last-Step Cost Assignment

For each step that uses some edge making the first or final appearance (F/L) , we assign a factor $O_r(\sqrt{\ell})$ to the whole-step.

Finally, we wrap up this section by recapping our factor assignments and verify that this is a complete factor-assignment scheme for the combinatorial factor.

Proposition 5.3.19. *Let $B_{global}(P)$ be the sufficient cost to identify the whole walk, and $B_{local}(P)$ be the cost assigned by our scheme to each step across the walk,*

$$B_{global}(P) \geq B_{local}(P)\,,$$

for

$$B_{local}(P) \coloneqq MatrixDimension \cdot \prod_i B_{local}(i)\,.$$

In other words, each combinatorial factor for counting the walk is assigned to some local step.

Proof. To start with, notice that we use a cost of n^ℓ to identify the walk-start in both the global and local accounting. Next, we prove the the combinatorial factor regarding edges that make the first appearance, the cost in ℓ for specifying the out-going active vertex is accounted in section 5.3.3, and the cost of specifying incoming active vertices is accounted by lemma lemma 5.3.20. For combinatorial cost regarding edges that make H appearance, or non-ideal L-step, we assign a factor of $O_r(q)$ to each such edge in section 5.3.3. Finally, for each edge making last appearance in the step in ideal-step, a total factor of ℓ is sufficient and we pick up two $\sqrt{\ell}$-factors from the first and last appearance according to section 5.3.3.

□

Lemma 5.3.20. *This is a complete vertex-assignment scheme regarding combinatorial factor of edges appearing for the first time. In other words, for any vertex v, let $B_F(v)$ be the total factor assigned to this vertex throughout the walk across its various appearances as an incoming active vertex of some edge appearing for the first time, we have*

$$B_F(v) \geq (n) \cdot (2q \cdot r)^{s(v)}$$

where we recall that $s(v)$ is the number of steps in which v is specified as an incoming vertex of some edge beyond the vertex's first appearance.

Proof. To start with, notice the RHS is the combinatorial factor incurred by F edges that arise from specifying the label of v. In the case $s(v) = 0$ and the vertex appears only twice in the walk via first and last appearance, we have $B_{vtx}(v) = (\sqrt{n})^2 = n$. For $s(v) \neq 0$, notice we still pick up n from the first and last appearance v which offset the contribution of n on the RHS. For the factor depending on $s(v)$, consider each time v appears as an incoming active edge beyond the first time, we assign to its particular appearance step a factor of $2q \cdot r$, which matches the cost assigned to vertex v via that particular edge. Taking product over all edges that contribute to $s(v)$ gives the desired.

□

5.3.4 Step-Bound from Factor Assignments

We are now ready to combine the above components and deduce our local bound in the factor assignment scheme. Before that, we observe the following meta-claim about vertex-appearance that allows us to exploit the evenness assumption of our walks such that each edge (i.e. random variable) needs to appear at least twice.

Claim 5.3.21. *In the case the edge is via an F (or L)-step, any incidental out-going vertex (or any incoming vertex) cannot be making* last *(or* first *appearance).*

Proof. This follows from the observation that edge needs to appear at least twice throughout the walk. Observe that it suffices for us to consider active vertices depending on whether they are outgoing or incoming. For an F-step, if any "outgoing" active vertex makes the final appearance at step t, the edge used by step-t would only appear only once as otherwise any active vertex of this edge would make a subsequent appearance.

Similarly, for an L-step while any incoming vertex is making a first appearance, the edge at step-t would have appeared only once throughout the walk. □

Lemma 5.3.22 (Step Bound for Even-r)**.** *For some fixed constant $\delta > 0$ and let $B_q(\alpha)$ be cost assigned to step-t with step-status $\alpha \in \{F, H, L\}$, we have*

$$B_q(F) = B_q(L) \leq O_r(1) \cdot \left(\sqrt{n \cdot \ell}\right)^{r/2} \cdot \sqrt{\ell},$$

and

$$B_q(H) \leq o_n(1) \left(\sqrt{n \cdot \ell}\right)^{r/2}.$$

provided $q \ll n^{\delta}$.

Proof. We case on the step-status from $\{F, H, L\}$. For notational convenience, consider the step-t move from boundary set $U_t = S_t$ to $V_t = S_{t+1}$. In the case this is an F-step, we observe the following,

1. Each dormant vertex in the intersection of $U_t \cap V_t$ not contribute any factor;

2. By claim 5.3.21, any outgoing vertex in $U_t \setminus V_t$ cannot be making its last appearance. Moreover, since each such vertex appears in the previous step and thus cannot be making their first appearance. Finally, note that none of such vertex may be specified via the middle appearance cost for "incoming" active vertex, hence there is no cost for any such vertex.

3. Any incoming vertex in $V_t \setminus U_t$ cannot be making its last appearance, while it can be making its first or middle appearance. In the case this is a first-appearance, we assign it a cost of $\sqrt{n}$; and for any subsequent middle appearance, we assign a cost of $2q \cdot r$ via section 5.3.3,

4. Additionally, any incoming vertex in $V_t \setminus U_t$ is assigned a factor of $\sqrt{\ell}$ via section 5.3.3 for the "split" cost of specifying the out-going active vertex of the edge;

5. Therefore, each vertex contributes a cost of

$$(\sqrt{n} + \ell \cdot 2q \cdot r)\sqrt{\ell} = O_r(1) \cdot \sqrt{n\ell}\,.$$

6. Edge-value is 1 for each F step via section 5.3.2;

7. Each F step gets assigned a factor of $O_r(\sqrt{\ell})$ from (potential) ideal last-step cost via section 5.3.3;

8. Combining the cost for all vertices in this step gives us a bound of

$$B_q(F) \leq O_r(1) \cdot (\sqrt{n\ell})^{r/2} \cdot \sqrt{\ell}$$

   as there are at most $r/2$ vertices in $V \setminus U$.

In the case this is an L-step, the factor is essentially the same as the F-step except vertices in U may now contribute as last-appearance vertex (as opposed to those in V contributing as first-appearance vertex). Formally, we have

1. Each dormant vertex in the intersection of $U_t \cap V_t$ not contribute any factor;

2. By claim 5.3.21, any incoming vertex in $V_t \setminus U_t$ cannot be making its first appearance. Moreover, since each such vertex appears in the subsequent step, and thus cannot be

making their last appearance. Finally, note that none of such vertex may be specified via the middle appearance cost for "incoming" active vertex since such a cost is only incurred for edges appearing for the first time. Therefore, no cost is assigned for any vertex in $V_t \setminus U_t$.

3. Any outgoing vertex in $U_t \setminus V_t$ cannot be making its first appearance, while it can be making its last or middle appearance. In the case this is a last-appearance, we assign it a cost of $\sqrt{n}$; and for any subsequent middle appearance, we assign a cost of $2q \cdot r$ via section 5.3.3.

4. Any outgoing vertex in $U_t \setminus V_t$ is on the step-boundary of an edge appearing for the last time, and therefore assigned a factor of $\sqrt{\ell}$ via section 5.3.3 for the "split" cost of specifying the out-going active vertex of the edge;

5. Therefore, each vertex $U_t \setminus V_t$ contributes a cost of

$$(\sqrt{n} + 2q \cdot r) \cdot \sqrt{\ell} = O_r(1) \cdot \sqrt{n\ell}\,.$$

6. Edge-value is 1 for each L step via section 5.3.2;

7. Each L step gets assigned a factor of $O_r(\sqrt{\ell})$ from (potential) ideal last-step cost via section 5.3.3;

8. Combining the above, notice that the factor of $2q$ is only needed if all vertices from $U \setminus V$ are making middle appearance, in which case we do not pick up any final-appearance $\sqrt{n\ell}$ factor. Therefore, we have the total bound of

$$B_q(L) \leq \sqrt{\ell} \cdot ((1 + o_n(1)\sqrt{n\ell})^{r/2} + \sqrt{\ell} \cdot 2q \cdot (\sqrt{n\ell})^{r/2-1} = O_r(1) \cdot (\sqrt{n\ell})^{r/2} \cdot \sqrt{\ell}\,.$$

provided $q \leq n^{\delta}$ for some $\delta > 0$.

Finally, for the case of H-step, it is immediate to observe that all vertices must be making a middle appearance with no extra vertex-cost as the edge can be specified at a cost of $2q$. Combining with the edge-value gives us a bound of

$$B_q(H) \leq \underbrace{(2q)}_{\text{edge-val}} \cdot \underbrace{(2q)}_{\text{combinatorial factor}} = (2q)^2 .$$

□

Lemma 5.3.23 (Step Bound for Odd-r)**.** *For some fixed constant $\delta > 0$, for any $q < n^{\delta}$, let $B_q(\alpha)$ be cost assigned to step-t with step-status $\alpha \in \{F, H, L\} \times \{F, H, L\}$, we have*

$$B_q(F \times F) = B_q(L \times L) \leq O_r(1) \left(\sqrt{n \cdot \ell}\right)^r ,$$

and

$$B_q(\beta) \leq o_n(1) \left(\sqrt{n \cdot \ell}\right)^r .$$

for any step-status $\beta \in \{F, H, L\} \times \{F, H, L\} \setminus \{F \times F, L \times L\}$. In other words, we treat any step other than both edges being simultaneously F or L as a lower-order term.

Proof. The proof is largely identical to the even-r case for the per-vertex factor of each active vertex except that the number of active vertices differs, except the factor of ℓ from outgoing active vertices of an F-edge warrants extra care as discussed in section 5.3.3 .

In the case of $F \times F$ status, notice the following change,

1. There are at most r incoming active vertices that could be making first appearance, so we get assigned a factor of at most $O_r(1) \cdot \sqrt{n}^r$

2. There are $r - 1$ incoming active vertices on the boundary, each of which gets an assigned factor of $\sqrt{\ell}$ from section 5.3.3

3. Each step gets assigned a total of $O_r(\sqrt{\ell})$ from section 5.3.3 .

Combining the above gives us $B_q(F \times F) \leq O_r(1) \left(\sqrt{n} \cdot \sqrt{\ell}\right)^r$.

The case of $L \times L$ status also warrants attention on its own,

1. There are at most r outgoingg active vertices that could be making last appearance, so we get assigned a factor of at most $O_r(1) \cdot \sqrt{n}^r$

2. There are $r-1$ outgoing active vertices on the boundary, each of which gets an assigned factor of $\sqrt{\ell}$ from section 5.3.3 ;

3. Each step gets assigned a total of $O_r(\sqrt{\ell})$ from section 5.3.3 .

Combining the above gives us $B_q(L \times L) \leq O_r(1) \left(\sqrt{n} \cdot \sqrt{\ell}\right)^r$. The bounds for other step-status are lower-order term, and follow immediately from the above discussion. □

Wrapping Up We are now ready to prove our main theorem.

Proof to theorem 5.1.8 and theorem 5.1.9. We focus on the even-r case while the odd-r case holds verbatim. Let B_q be our final step-value bound for each step by our factor-assignment scheme, summing over all possible step-status, we have

$$B_q \leq B_q(F) + B_q(H) + B_q(L) \leq O_r(1)\sqrt{n \cdot \ell}^{r/2} \cdot \sqrt{\ell}$$

By construction of our factor-assignment scheme, this gives an upper bound of

$$\mathbb{E}[\mathrm{Tr}(M_\ell)^{2q}] \leq \text{MatrixDimension} \cdot (B_q)^{2q} .$$

Finally, this translates to a matrix norm bound immediately by Markov's as we consider

for any constant $\epsilon > 0$.

$$\begin{aligned}\mathbf{Pr}\left[\|M_\ell\|_{sp} \geq (1+\epsilon)\cdot B_q\right] &\leq \frac{\mathbb{E}[\mathrm{Tr}(M_\ell)^{2q}]}{(1+\epsilon)^{2q}\cdot (B_q)^{2q}} \\ &\leq \frac{n^\ell \cdot (B_q)^{2q}}{(1+\epsilon)^{2q}\cdot (B_q)^{2q}} \\ &\leq c^{-q/\log n}\end{aligned}$$

for some constant $c > 0$ since our q can be taken as $q < n^\delta$ for some constant $\delta > 0$. □

5.4 Lower Bounding Spectral Norm: Refuting Intrinsic Freeness

We now complement our upper bound result by a lower bound. For this section, we restrict our attention to the even-r case and note that an analogous argument extends to odd-r case.

Lemma 5.4.1.

$$\begin{aligned}\mathbb{E}[\mathrm{Tr}\left(M_\ell^{2q}\right)] &\geq \textit{matrix-dimension}\cdot\left((2\ell/r)^{r/4}\cdot(2\ell/r)\cdot\sqrt{\binom{n-\ell}{r/2}}\right)^{2q} \\ &= \textit{matrix-dimension}\cdot\left(\Theta_r(1)\cdot\sqrt{n^{r/2}\cdot\ell^{r/2}}\cdot\sqrt{\ell}\right)^{2q}.\end{aligned}$$

Proof. To give a lower bound, we observe that each term in the expected trace is non-negative, and therefore it suffices for us to focus on a specific type of walks in the Kikuchi graph (such that it uses each edge twice so that it has expected-value 1), and then show that the combinatorial count of such walks is large. Consider the walks of the following type,

1. It would be useful to consider $m := \ell/0.5r = 2\ell/r$ buckets, and consider splitting the length-$2q$ walk into a walk of $2q/(2m)$ chunks, with each chunk being a walk of length-$2m$.

2. For each chunk of length-$2m$, this is a walk consist of m many F-steps and m many L steps (moreover, we focus on F-step leading to "new" vertices only).

3. For each chunk, there are $\binom{\ell}{r/2,r/2,r/2....,r/2}$ ways to bucket $r/2$ vertices among ℓ vertices, and each bucket will get to walk twice, once via an F-step and the other via an L-step.

4. Observe that we have $\binom{2m}{2,2,2...,2}$ many orders to pick which bucket "walks" at each step, and regardless of the ordering, we are guaranteed to return to the start at the end of this chunk. In other words, each arbitrary ordering gives a valid walk.

5. For each F-step we have the usual factor of $\binom{n-\ell}{r/2}$, while notice we no longer have the factor of $\binom{\ell}{r/2}$ since the bucket has been determined.

Next, we recall the following fact about the center coefficient of multinomials,

Fact 5.4.2. *There exists some constant $c_k > 0$ such that*

$$\log \binom{kn}{n} \geq c_k \cdot kn \log k$$

for sufficiently large n.

Thus, we have a total count of the length-$2m$ chunk as

$$\binom{\ell}{r/2, r/2, r/2...., r/2} \cdot \binom{2m}{2, 2, 2 \ldots, 2} = \exp(\ell \log(2\ell/r)) \cdot \exp(2m \log m) \cdot \binom{n-\ell}{r/2}^m$$

Taking the $2m$-th root gives us the cost per step as

$$\Theta(1) \cdot 2^{\frac{\ell}{2m} \cdot \log(2\ell/r)} \cdot m \cdot \sqrt{\binom{n-\ell}{r/2}},$$

Recall that $m = 2\ell/r$, we have

$$\Theta(1) \cdot (2\ell/r)^{r/4} \cdot (2\ell/r) \cdot \sqrt{\binom{n-\ell}{r/2}}.$$

Recall that our upper bound reads as (per step)

$$O_r(1) \cdot \sqrt{n^{r/2} \cdot \ell^{r/2+1}},$$

this is tight up to the hidden constant in $O_r(1)$. □

Next, analogously to lower bounding spectral norm of a Wigner matrix, to deduce the concentration of $\|M_\ell\|$ within $\Theta_r(1) \cdot \sqrt{n^{r/2} \cdot \ell^{r/2}} \cdot \sqrt{\ell}$, we use the usual relation between the spectral norm and trace to lower bound $\mathbb{E}[\|M_\ell\|^{2q}]$. For simplicity, we focus on the setting when G is a random symmetric $\{\pm 1\}$ tensor, while note that the analogous bound can be extended to Gaussian via more complicated usage of Gaussian concentration inequality. Crucially , since $\|M_\ell\|_{sp}$ is sub-gaussian, we can further deduce

Corollary 5.4.3. *For G an random symmetric tensor of Bernoulli input,*

$$\mathbb{E}[|\|M_\ell(G)\|_{sp}] \geq \Theta_r(1) \cdot \sqrt{n^{r/2} \cdot \ell^{r/2}} \cdot \sqrt{\ell},$$

and moreover, with probability at least $1 - o_n(1)$,

$$\|M_\ell(G)\|_{sp} \geq \Theta_r(1) \cdot \sqrt{n^{r/2} \cdot \ell^{r/2}} \cdot \sqrt{\ell}.$$

Chapter 6

Singular Value Lower Bounds for Polynomial Decomposition

This is based on the joint work with Mitali Bafna, Tim Hsieh, and Pravesh Kothari. We opt to include a rough overview of the analysis for singular-value lower bounds analysis to highlight the application of graph matrix. We defer interested reader to the original paper for the full algorithm for power-sum decomposition of random polynomials [BHKX22], and for the full combinatorial analysis that carries out the below strategy.

6.1 Technical Overview for Singular Value Lower Bounds

A two-line strategy Before we delve into the "real" challenge of what might come up in our analysis, for the sake of exposition, let's recall our G.O.E. toy example $A \in \mathbb{R}^{m \times n}$ for $m \ll n$. To obtain a singular value lower bound for A when $m \ll n$, we notice it can be reduced to showing

$$A^\top A \approx (1 + o(1))n \cdot \mathrm{Id}\,,$$

which amounts to checking

- Large diagonal: $\mathrm{diag}(A^\top A) = (1 + o(1))n$;
- Small off-diagonal: $\|\text{off-diagonal}(A^\top A)\|_{sp} \leq o(n)$.

For the above example, one may find both components rather immediate as the large diagonal follows via standard Gaussian concentration. While establishing a small spectral norm bound for the off-diagonal block might require slightly more work, there are several techniques to establish the nearly optimal bound of $O(\sqrt{m}\sqrt{n})$ on its spectral norm. That said, we may hope to carry out the above strategy on potentially more complicated matrices (with correlated entries), and let's dig into one example that arises in our analysis for the simplest setting in our main result for decomposing cubics of quadratic.

Example of structured random matrices To describe the matrix, let $A_1, \ldots, A_m$ be random matrices of size $n \times n$ with each entry i.i.d. $\mathbb{N}(0, 1)$. We will describe a new matrix whose entries are functions of A_is. Let $A_t[i] \in \mathbb{R}^n$ denote the i-th column of matrix A_t, and let $S \subseteq [n]$ be a subset of size $\ell \approx \sqrt{n}$. Finally, let G be the matrix defined by:

$$G := \begin{bmatrix} \cdots \\ A_t[i] \otimes A_t[j] \\ \cdots \end{bmatrix} \in \mathbb{R}^{m\binom{\ell}{2} \times n^2}.$$

In other words, each row of G is indexed by (t, i, j) for $t \in [m]$ and $i \neq j \in [\ell]$ with the corresponding row vector be $A_t[i] \otimes A_t[j]$. And a question we would like to answer is the following,

Question 6.1.1. *What is the largest m for which G is full (row) rank with inverse-polynomially bounded smallest singular value?*

For starters, $m \sim n$ is a natural upper bound as G becomes a roughly square matrix when we have $m\ell^2 \approx n^2$. Let's now implement the above strategy. The diagonal entry

(t, i, j) of $GG^\top$ is given by $\|A_t[i] \otimes A_t[j]\|_2 \approx n^2$ by standard Gaussian concentration, and the two-step strategy now prompts us to bound the spectral norm of the off-diagonal part of $GG^\top$, i.e. the spectral norm of $GG^\top - n^2 \cdot \mathrm{Id}_{m\binom{\ell}{2}}$, by $o(n^2)$.

Observe that easy estimates such as the maximum ℓ_1 norm of any row do not give a useful bound: we expect the off-diagonal entries to have typical magnitude n and that can be "charged" to the diagonal entries only if $m\binom{\ell}{2} \cdot n \lesssim n^2$ or $m \leq O(1)$. However, in this very case, one may notice that we are ignoring the potential cancellation from the signs of the entries (as each entry is a mean 0 random variable) and we can hope to obtain a tighter bound by exploiting this kind of cancellation. Towards this end, we appeal to the *trace moment method* to obtain tight bounds for these structured random matrices, and on a high level, for ease of our spectral analysis, we adopt *graph matrix decomposition* to systematically represent such random matrices of correlated entries.

Application of Graph Matrix Decomposition The theory of graphical matrices generally applies to a matrix whose entry is a polynomial of some underlying input. In our case, the off-digonal part of $GG^\top$ is described as:

$$GG^\top[(s,a,b),(t,c,d)] = \langle A_s[a], A_t[c]\rangle \cdot \langle A_s[b], A_t[d]\rangle = \sum_{i,j\in[n]} A_s[i,a]A_t[i,c]A_s[j,b]A_t[j,d]$$

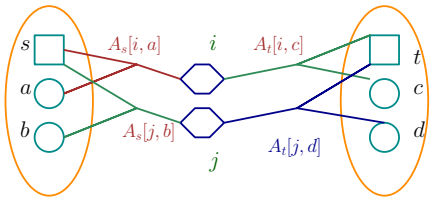
for $(s,a,b) \neq (t,c,d)$. For illustration, we restrict to the case for $s \neq t \in [m]$ and a, b, c, d distinct elements in $[\ell]$, and we remark that in general these cases warrant a careful analysis. Notice we can further decompose the above polynomial entry as

$$\begin{aligned} GG^\top[(s,a,b),(t,c,d)] = &\sum_{i\neq j\in[n]} A_s[i,a]A_t[i,c]A_s[j,b]A_t[j,d] \\ &+ \sum_{i=j\in[n]} A_s[i,a]A_t[i,c]A_s[j,b]A_t[j,d] \end{aligned}$$

depending on whether i, j "collide" with each other. Pictorially, each term in the above matrix corresponds to one of the following diagrams using graph matrix language:

each vertex in the left/right orange oval corresponds to a variable in the row/column index for the matrix;

and each vertex in the middle (outside the orange ovals) corresponds to an indeterminate in the summation with each hyperedge corresponding to a random variable in the polynomial.



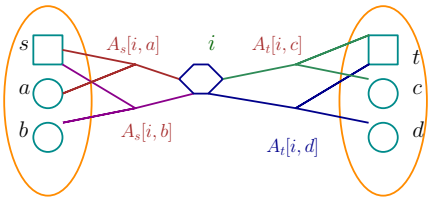


Figure 6.1: Diagram for $GG^\top[(s,a,b),(t,c,d)] = \sum_{i \neq j \in [n]} A_s[i,a]A_t[i,c]A_s[j,b]A_t[j,d]$.

Figure 6.2: Diagram for $GG^\top[(s,a,b),(t,c,d)] = \sum_{i = j \in [n]} A_s[i,a]A_t[i,c]A_s[i,b]A_t[i,d]$.

Once these diagrams are drawn out, we can apply off-the-shelf matrix norm bounds for graph matrices. Therefore, our job for spectral analysis is essentially estimating the combinatorial quantity for each of the diagrams that may arise in the decomposition. In the setting above, a direct application of this idea in a black-box manner gives us a spectral norm bound $\|\text{offdiag}(GG^\top)\|_{sp} \leq \tilde{O}(\sqrt{m\ell^2 n^2}) = o(n^2)$ assuming $m\ell^2 \ll n^2$ and $\ell \approx \sqrt{n}$, giving us the $m \approx n$ bound as we anticipate in the beginning.

6.2 Singular value lower bounds for analysis of V

Recall that given $B_1, \ldots, B_m$ which are degree-2 homogeneous polynomials in ℓ variables with i.i.d. standard Gaussian coefficients, the matrix $V \in \mathbb{R}^{\ell_{4D} \times m\ell_{2D}}$ is the matrix whose columns represent the polynomials $B_t(y)^D y_{j_1} \cdots y_{j_{2D}}$ for $t \in [m]$ and $j_1, \ldots, j_{2D} \in [\ell]$. Furthermore, recall the matrix $N \in \mathbb{R}^{\binom{m}{2} \times m\ell_{2D}}$ defined in Definition 4.20 of [BHKX22] that

satisfies $VN^\top = 0$, i.e. each row of N represents a collection of m degree-$2D$ polynomials $(p_1, \ldots, p_m)$ such that

$$\sum_{t=1}^{m} B_t(y)^D p_t(y) = 0\,,$$

and for row index (s_1, s_2) of N, $p_t(y) = B_{s_2}(y)^D$ if $t = s_1$, $-B_{s_1}(y)^D$ if $t = s_2$, and 0 otherwise.

Proof of Lemma 4.21: rank of N

Lemma 6.2.1 (Restatement of Lemma 4.21: Rank of N)**.** *Let $m, \ell, D \in \mathbb{N}$ such that $m \leq (\frac{\ell}{\text{polylog}(\ell)})^{2D}$. Let $N \in \mathbb{R}^{\binom{m}{2} \times m\ell_{2D}}$ be the matrix defined in Definition* **??***. Then, with probability $1 - \ell^{-\Omega(D)}$,*

$$\sigma_{\binom{m}{2}}(N) \geq \Omega(\ell^D)\,.$$

Proof. We will show that $NN^\top \in \mathbb{R}^{\binom{m}{2} \times \binom{m}{2}}$ is full rank and that $\lambda_{\min}(NN^\top) \geq \Omega(\ell^{2D})$. Notice that the columns of N are indexed by (t, I) where $t \in [m]$ and $I \in [\ell]^{2D}$ is a multiset. By Fact **??**, we can delete some columns of N and prove that the resulting matrix N' satisfies $\lambda_{\min}(N'N'^\top) \geq \Omega(\ell^{2D})$, which implies $\lambda_{\min}(NN^\top) \geq \Omega(\ell^{2D})$. Thus, we will assume that the matrix N only consists of columns (t, I) where I is a set (no repeated elements), i.e., each row consists of multilinear polynomials.

We decompose $NN^\top$ into diagonal and off-diagonal components.

Diagonal entries. The diagonal entry at $s_1 < s_2$ can be written as

$$NN^T[(s_1, s_2), (s_1, s_2)] = \|multilinear(B_{s_1}^D)\|^2 + \|multilinear(B_{s_2}^D)\|^2\,,$$

where we take the multilinear parts of $B_{s_1}^D$ and $B_{s_2}^D$ because we deleted the columns corresponding to non-multilinear monomials. This however does not change the norms up to constant factors, hence by Claim 3.8, the diagonal entries are $\Omega(\ell^{2D})$.

Off-diagonal part. $NN^\top[(s_1,s_2),(t_1,t_2)]$ is zero when s_1,s_2,t_1,t_2 are all different, and if $s_1 \neq t_1$ but $s_2 = t_2$ (for e.g.), then

$$\begin{aligned} NN^T[(s_1,s_2),(t_1,t_2)] &= \langle multilinear(B_{s_1}^D), multilinear(B_{t_1}^D)\rangle \\ &= \Theta(1) \sum_{I\in[\ell]^{2D}} Sym(B_{s_1}^{\otimes D})[I]\cdot Sym(B_{t_1}^{\otimes D})[I] \\ &= \Theta(1) \sum_{\pi\in\mathbf{S}_{2D}} \sum_{\substack{I\in[\ell]^{2D} \\ \text{all distinct}}} B_{s_1}^{\otimes D}[I]\cdot B_{t_1}^{\otimes D}[\pi(I)]\,. \end{aligned}$$

We view the above as a summation of graphical matrices, each corresponding to a permutation π. We then apply graphical matrix norm bounds to bound the spectral norms of them. Given π, the shape that arises can be described as the following,

1. A tripartite graph with exactly $2D$ circle vertices $(i_1),\ldots,(i_{2D})$ in the middle (that take labels in $[\ell]$) , i.e., $V(\alpha)\setminus(U_\alpha\cup V_\alpha)$;

2. U_α, V_α both have two square vertices (that take labels in $[m]$): $U_\alpha = \{\boxed{s},\boxed{u}\}$ and $V_\alpha = \{\boxed{t},\boxed{u}\}$, where $\boxed{u}$ is in the intersection $U_\alpha\cap V_\alpha$;

3. On the left, we add hyperedges in order: $(\boxed{s},(i_1),(i_2))$, $(\boxed{s},(i_3),(i_4))$, and so on; on the right, we add $(\boxed{t},(i_{\pi(1)}),(i_{\pi(2)}))$, $(\boxed{t},(i_{\pi(3)}),(i_{\pi(4)}))$, and so on.

Now we analyze the minimum vertex separator. Observe that for every (i) in the middle, there is a path $\boxed{s}\to(i)\to\boxed{t}$. Thus, any vertex separator of such shape must contain either $\boxed{s}$, $\boxed{t}$, or all of the circle vertices. Clearly, when $m \ll \ell^{2D}$, the minimum weight vertex separator is $\{\boxed{s}\}$ or $\{\boxed{t}\}$. Thus, we get a norm bound of $\tilde{O}\left(\sqrt{m\ell^{2D}}\right)$.

Therefore, we can write

$$NN^TT \succeq \Omega(\ell^{2D})\cdot \mathrm{Id} + offdiag(NN^\top)\,,$$

where $\|offdiag(NN^\top)\|_{sp} \leq \tilde{O}(\sqrt{m}\ell^D) = o(\ell^{2D})$ given that $m \leq (\frac{\ell}{\text{polylog}(\ell)})^{2D}$. Thus, the diagonal dominates, and we have $\lambda_{\min}(NN^\top) \geq \Omega(\ell^{2D})$, hence $\sigma_{\binom{m}{2}}(N) \geq \Omega(\ell^D)$. $\square$

Chapter 7

Sum-of-Squares Lower Bounds for Independent Set in Ultra-Sparse Random Graphs

7.1 Our Results

In this work, we prove that for every $d \in \mathbb{N}$, constant degree sum-of-squares strengthening of the independent set axioms fail to certify a bound of $o(n/\sqrt{d})$ on the maximum independent set with high probability when $G \sim G(n, d/n)$. To formulate this precisely, let us recall the notion of pseudo-expectations consistent with a set of polynomial equations:

Definition 7.1.1 (Pseudo-expectation of degree-dsos)**.** *For any $d_{sos} \in \mathbb{N}$, a degree* dsos*-pseudoexpectation in variables $x = (x_1, x_2, \ldots, x_n)$ (denoted by $\widetilde{\mathbb{E}}_{\mathsf{dsos}}$) is a linear map that assigns a real number to every polynomial f of degree $\leq d$ in x and satisfies: 1)* ***normalization:*** *$\widetilde{\mathbb{E}}[1] = 1$, and, 2)* ***positivity:*** *$\widetilde{\mathbb{E}}[f^2] \geq 0$ for every polynomial f with degree at most $\frac{\mathsf{dsos}}{2}$. For any polynomial $g(x)$, a pseudo-expectation satisfies a constraint $g = 0$ if $\mathbb{E}[f \cdot g] = 0$ for all polynomials f of*

degree at most $d_{sos} - deg(g)$.

We can now describe the sum-of-squares relaxation for independent set in a graph G.

Definition 7.1.2 (Independent Set Axioms)**.** *Let G be a n-vertex graph. The following axioms describe the* 0-1 *indicators x of independent sets in G:*

$$\forall v \in V, x_v^2 = x_v \quad \textit{(Booleanity)} \ ;$$

$$\forall (u,v) \in E, x_u x_v = 0 \quad \textit{(Independent set)}$$

The degree d_{sos}-sum-of-squares relaxation for independent set in G maximizes $\widetilde{\mathbb{E}}[\sum_i x_i]$ over all pseudo-expectations of degree d_{sos} satisfying the above two axioms.

More precisely, we prove:

Theorem 7.1.3 (Main result)**.** *There is a $c > 0$ such that for every $d \in \mathbb{N}$, with probability $1 - o_n(1)$ over $G \sim G_{n,d/n}$, there exists a degree-*dsos *pseudo-expectation satisfying the independent set axioms (Definition 7.1.2) and*

$$\sum_{i \in [n]} \widetilde{\mathbb{E}}[x_i] \geq (1 - o(1)) \frac{n}{c \cdot \sqrt{d} \cdot \mathsf{dsos}^4} \, .$$

Remark 7.1.4. *We note that our techniques likely allow improving the dependence on d_{sos} in the denominator to a linear (instead of quartic) bound. We believe that removing this dependence altogether is possible but requires new ideas.*

Our construction of the lower bound witness (aka pseudo-moment matrix) is based on pseudo-calibration. Informally speaking, pseudo-calibration provides a guess for the pseudo-expectation with a certain "truncated" probability density function of a planted distribution (a distribution over graphs containing a large independent set) that is indistinguishable for low-degree polynomials from $G(n, d/n)$. However, as observed in prior

works [JPR+22], natural planted distributions are *not* low-degree indistinguishable from $G(n, d/n)$ when $d = O(1)$. Nevertheless, it turns out that one can use a certain ad hoc truncation that drops certain carefully chosen terms to obtain a pseudo-expectation that allows us to prove theorem 7.1.3 above. Our truncation strategy is an upgraded variant of the *connected truncation* first utilized in [JPR+22].

In the ultra-sparse regime, our analysis needs to separately consider the vertices that have too high a degree. This trimming of the graph inevitably introduces non-trivial correlations in the graph matrices – a significant challenge – that we show how to overcome in our analysis.

7.2 Moment matrix construction

7.2.1 Trimming the graph

Let G be a random graph on n vertices sampled from $G_{n,d/n}$, instead of working directly with the graph sample, we work with a nice subgraph of G of n' vertices that is obtained by removing "not-too-many" $o(n)$vertices, and show that SoS continues to think there is an independent set of size $o(\frac{n'}{\sqrt{d}})$, we can translate the lower bound for $\widetilde{G}$ to a lower bound for the untruncated graph G. Let's start with the definition of our "trimmed" subgraph.

Definition 7.2.1 (Trimmed subgraph)**.** *We call $\widetilde{G}$ the* trimmed *subgraph of G obtained by removing any "ultra-high degree" vertex, i.e. a vertex that has degree at least $c_{deg} \cdot d$ in G. For this work, we set $c_{deg} = 10$.*

By standard Poisson concentration, we have the following bound on the probability of a vertex begin degree-bad.

Proposition 7.2.2. *Each vertex is degree-bad with probability at most* $\exp\left(-c_{\text{deg}}^2 \cdot d\right)$.

Gluing pseudo-distributions for the whole graph Let $\widetilde{\mathbb{E}}_{\text{trimmed}}$ be a degree-dsos pseudo-expectation for independent set of size k for G_{trimmed} on n' vertices, and let v_B be an (integral) coloring assignment for G_B, we note that they can be merged into a degree-dsos pseudo-expectation $\widetilde{\mathbb{E}}_G$ for independent set of the whole graph G on n vertices.

Definition 7.2.3. *For $S = S_G \cup S_H \subseteq [n]$ where S_H is the set of high degree vertices and S_G the vertices in the trimmed graph, we define*

$$\widetilde{\mathbb{E}}_G[x_S] \coloneqq \widetilde{\mathbb{E}}_{trimmed}[x_S]$$

for $S \cap S_H = \emptyset$, and 0 otherwise.

Claim 7.2.4. *$\widetilde{\mathbb{E}}_G$ is a valid independent-set pseudo-expectation for G at degree* dsos *of value k. Moreover, $\widetilde{\mathbb{E}}_G$ gives an independence number of G at least $\frac{k}{n} \geq (1 - o(1)) \cdot \frac{k}{n'}$.*

Proof. The constraints for independent-set pseudo-expectation are immediate, and the value of the independent set size follows from $n - n' = o(n)$. □

With this reduction, the bulk of our work boils down into constructing $\widetilde{\mathbb{E}}_{\text{trimmed}}$ for the graph restricted to the nice vertices. From this point on, we will drop the subscript "trimmed" and refer to it simply by $\widetilde{\mathbb{E}}$. Let's now proceed to its construction.

7.2.2 Constructing pseudo-expectation for the truncated subgraph

Pseudocalibration for Independent Set Similar to previous works, we consider one of the most straightforward planted distribution D_{pl} for any independent set of size k in $G_{n,d/n}$:

(1) Sample a random graph $G \sim G_{n,d/n}$;

(2) Sample a subset $S \subseteq [n]$ by picking each vertex with probability $\frac{k}{n}$;

(3) Let $\tilde{G}$ be G with edges inside S removed, and output $(S, \tilde{G})$.

It can then be computed that we have

Lemma 7.2.5 (Lem 4.4 in [JPR+22]). *Let $x_T(S)$ be the indicator function for T being in the planted solution i.e. $T \subseteq S$. Then, for all $T \subseteq [n]$ and $\alpha \subseteq \binom{[n]}{2}$,*

$$\mathbb{E}_{(S,\tilde{G})\sim\mathcal{D}_{pl}}[x_T(S) \cdot \chi_\alpha(\tilde{G})] = \left(\frac{k}{n}\right)^{|V(\alpha)\cup T|} \left(-\frac{p}{1-p}\right)^{\frac{|E(\alpha)|}{2}}$$

Non-dangling truncation: an upgrade from [JPR+22] A major insight from [JPR+22] in designing moments for sparse random graph is that one may employ the rule "connected truncation", and ignore any graph (or more precisely, "shape") that is not connected to S.

On a high level, the connected truncation removes polynomial distinguisher based on subgraph-statistics, and a natural question is whether their that alone is sufficient here in the ultra-sparse regime should one ask for an independent set lower bound. Surprisingly, it is not! At least not within the current framework of analysis.

We now recall the definition of dangling vertex from our prior discussion for tight matrix norm bounds.

Definition 7.2.6 (Dangling and non-dangling vertex). *For a given shape α, we call a vertex $v \in V(\alpha)$ a "dangling" vertex if it is not on any simple path from U_α to V_α; otherwise, we call the vertex "non-dangling". Equivalently, any dangling vertex is a vertex of degree at most 1 outside $S = U_\alpha \cup V_\alpha$.*

It can be verified that without removing dangling shapes, some shape indeed has large norm, and break apart the charging strategy in their analysis. Therefore, removing such shapes turns out to be surprisingly necessary within our current framework.

Definition 7.2.7 (Truncation of $\widetilde{\mathbb{E}}$). *Let $T(S)$ be the set of graphs with labeled vertices $S \subseteq [n]$ s.t. for any graph $\alpha \in T(S)$,*

1. *(Size-constraint)* $|V(\alpha) \cup S| \leq D_V$;

2. *(Connectivity) Any vertex in* $V(\alpha)$ *is reachable from* S;

3. *(Non-dangling) Any vertex in* $V(\alpha) \setminus S$ *has degree at least* 2;

for $D_V = c_{trunc} \cdot \mathsf{dsos} \log n$ *for some absolute constant* $c_{trunc} > 0$.

Remark 7.2.8. *When it is clear, we also drop the dependence on* x_S *and let* $\mathcal{S}$ *be the set of shapes (or ribbons) that appear in our moment matrix* $\widetilde{\mathbb{E}}_{trimmed}$.

This lands us at the following definition of the moment matrix in the language of linear operator, as we will defer the matrix's language to the latter section after formally introducing ribbons,

Definition 7.2.9 (Pseudo-expectation for Independent-Set as a linear operator)**.**

$$\begin{aligned}\widetilde{\mathbb{E}}[x_S](G) &= \sum_{\alpha \in T(S)} \mathbb{E}_{D_{pl}}[x_S \cdot \chi_\alpha] \cdot \chi_\alpha(G) \\ &= \sum_{\alpha \in T(S)} \left(\frac{k}{n}\right)^{|V(\alpha) \cup S|} \cdot \left(-\sqrt{\frac{p}{1-p}}\right)^{|E(\alpha)|} \cdot \chi_\alpha(G)\end{aligned}$$

Definition 7.2.10 (Setting objective value k)**.** *Throughout this work, we will pick* $k = \frac{n}{c_\eta \sqrt{d} \cdot \mathsf{dsos}^4}$ *for some absolute constant* $c_\eta > 1$.

Formally, we define our candidate moment matrix for $\widetilde{\mathbb{E}}$ as the following.

Definition 7.2.11 (Moment matrix $\widetilde{\Lambda}$)**.** *We define the moment matrix as*

$$\widetilde{\Lambda} := \sum_{\alpha \in \mathcal{S}} \left(\frac{k}{n}\right)^{|V(\alpha)|} \cdot \left(-\sqrt{\frac{p}{1-p}}\right)^{|E(\alpha)|} \cdot \frac{M_\alpha}{|\mathsf{Aut}(\alpha)|}$$

where we remind the reader that $|\mathsf{Aut}(\alpha)|$ *is the size of automorphisnm group used to ensure each edge-set contributes once in the decomposition, and* $\mathcal{S}$ *is the set of shapes such that*

1. *(Size-constraint)* $|V(\alpha)| \leq D_V$;

2. *(Degree-constraint)* $|U_\alpha \cup V_\alpha| \leq \mathsf{dsos}$;

3. *(Connectivity) Any vertex in* $V(\alpha)$ *is reachable from* S;

4. *(Non-dangling) Any vertex in* $V(\alpha) \setminus S$ *has degree at least* 2.

7.2.3 Everything but PSDness for Independent Set

Since our truncation is largely inspired by the "connected truncation", it continues to inherit several nice properties from connected truncation that allows us to readily verify the non-PSDness properties of $\widetilde{\mathbb{E}}$.

Claim 7.2.12 (Independent-set constraint)**.** *For any* $S \subseteq [n]$*,*

$$\widetilde{\mathbb{E}}[x_S] = 0$$

if S *is not an independent set in* G*.*

Proof. This follows by grouping the subgraphs contributing to $\widetilde{\mathbb{E}}[x_S]$ according to the independent-set indicator in S as

$$\begin{aligned}\widetilde{\mathbb{E}}[x_S] &= \sum_{\alpha \in T(S)} \left(\frac{k}{n}\right)^{|V(\alpha)\cup S|} \cdot \left(-\sqrt{\frac{p}{1-p}}\right)^{|E(\alpha)|} \cdot \chi_\alpha(G) \\ &= \sum_{\alpha \in T(S): E(\alpha)=\emptyset} \left(\frac{k}{n}\right)^{|V(\alpha)\cup S|} \cdot \left(-\sqrt{\frac{p}{1-p}}\right)^{|E(\alpha)|} \cdot \chi_\alpha(G) \cdot \left(\sum_{H \subseteq \binom{V(S)}{2}} \sqrt{\frac{p}{1-p}} \cdot \chi_H(G)\right)\end{aligned}$$

Proposition 7.2.13 (Independent-set indicator)**.** *For* $e \in \{0,1\}$*,* $1 - \sqrt{\frac{p}{1-p}} \cdot \chi(e) = \frac{1}{1-p}(1 -$

e); *furthermore, for any set of edges* $H \subseteq \binom{n}{2}$,

$$\sum_{T \subseteq H} \left(-\sqrt{\frac{p}{1-p}}\right)^{|T|} \chi_T(G) = \frac{1}{(1-p)^{|H|}} \cdot \prod_{e \in H} (1-e)$$

Combining the above proves our claim. □

Claim 7.2.14 (Normalization)**.** *For any graph* G, $\widetilde{\mathbb{E}}[1] = 1$.

Proof. This follows immediately by connected truncation, that the only shape contributing to $S = \emptyset$, i.e., $\widetilde{\mathbb{E}}[1]$, is the emptyset, which comes with coefficient 1. □

Claim 7.2.15. *The candidate moment matrix of* $\widetilde{\Lambda}$ *is SoS-symmetric.*

Proof. This follows by observing for a fixed $S = U_\alpha \cup V_\alpha$, the coefficient does not depend on the partition of S into U_α and V_α. □

Claim 7.2.16 (Large independent set (Lemma 4.11 in [JPR+22]))**.** *With probability at least* $1 - o_n(1)$, *we have*

$$\widetilde{\mathbb{E}}[\sum_i x_i] \geq (1 - o(1))k$$

7.3 PSDness Analysis from Norm Bounds

As foreshadowed in our overview, this section follows the PSDness analysis strategy as [JPR+22], and we refer interested reader to it for a more thorough exposure to the approximate factorization machinery for proving high-degree Sum-of-Squares lower bounds. In this work, we build upon the technical analysis from [JPR+22] to give a more delicate combinatorial analysis to control the lower-order dependence of norm bounds unique to us.

7.3.1 Decomposition of the SoS moment matrix into the graph matrix basis

With these technical definitions ready, we can now observe that the SoS moment matrix we define earlier can be rewritten as

$$\widetilde{\Lambda} = \sum_{\text{shape } \alpha \in \mathcal{S}} \frac{\widetilde{\lambda}_\alpha \cdot \mathsf{M}_\alpha}{|\mathsf{Aut}(\alpha)|}$$

where we define

$$\widetilde{\lambda}_\alpha := \left(\frac{k}{n}\right)^{|V(\alpha)|} \cdot (-\sqrt{\frac{p}{1-p}})^{|E(\alpha)|}$$

and recall that $\mathcal{S}$ is the set of ribbons/shapes under our truncation rule.

Definition 7.3.1 ((Scaled) Shape coefficient)**.** *Given a shape α, we define its shape-coefficient*

$$\lambda_\alpha = \left(\frac{k}{n}\right)^{|V(\alpha)| - \frac{|U_\alpha|+|V_\alpha|}{2}} \cdot \left(-\sqrt{\frac{p}{1-p}}\right)^{|E(\alpha)|}$$

For convenience, it is convenient to work with a rescaling of the above matrix Λ that can be obtained by left and right multiplying a diagonal matrix with diagonal being $\sqrt{k}^{|U_\alpha|}$, giving us the matrix

$$\Lambda = \sum_{\alpha \in \mathcal{S}:\text{shape}} \frac{\lambda_\alpha \cdot M_\alpha}{|\mathsf{Aut}(\alpha)|}.$$

7.3.2 Recursive factorization overview: decomposition

Recall that our goal is to show

$$\Lambda = \sum_{\alpha \in \mathcal{S}} \lambda_\alpha M_\alpha \succeq 0$$

Towards this goal, we will decompose each shape α into $\sigma \circ \tau \circ \sigma'^T$, for $\sigma, \sigma'^T \in \mathsf{LeftShape}$ some left shape, and $\tau \in \mathsf{MidShape}$ some middle shape based on *Minimum Vertex Separator*

(MVS), which corresponds to writing the matrix (modulo intersection terms)

$$\mathsf{M}_{\alpha} \approx \mathsf{M}_{\sigma}\mathsf{M}_{\tau}\mathsf{M}_{\sigma'^{T}}$$

Analogously, we will also factorize the coefficients of λ_{α}, and with the decomposition in hand, we can simply write

$$\Lambda \approx (\sum_{\sigma \in \mathsf{LeftShape}} \lambda_{\sigma} M_{\sigma})(\sum_{\tau \in \mathsf{MidShape}} \lambda_{\tau} M_{\tau})(\sum_{\sigma'^{T} \in \mathsf{LeftShape}} \lambda_{\sigma'^{T}} M_{\sigma'^{T}}) = LQL^{T}$$

Our goal now reduces to showing $Q \succeq 0$, and this would allow us to restrict our attention to the middle shapes. Unfortunately, the above is an over-simplification as the equality only holds in an approximate sense, in which we hide out terms arising from ribbon intersection that warrants some extra care.

Consider ribbons of shape σ, τ, σ' intersect into some shape ζ we follow the usual strategy of recursive factorization and factor out γ, γ' s.t. γ is the leftmost MVS separating U_{σ} and $V_{\sigma} \cup \mathrm{Int}(P)$, and γ' the rightmost MVS separating $V_{\sigma'^{T}}$ and $U_{\sigma'^{T}} \cup \mathrm{Int}(P)$, i.e., we decompose

$$M_{\zeta} \approx M_{(\sigma - \gamma)} M_{\tau_P} M_{(\sigma'^{T} - \gamma'^{T})}$$

to further handle the intersection terms. The recursive factorization will create more intersection terms, however, the key is to notice the leftmost/rightmost MVS will move closer towards U_{σ} and $V_{\sigma'^{T}}$ after each intersection, and hence this process would terminate. Another technical detail that we ignore so far is the truncation error, and we defer this to a later section. To summarize the above, our decomposition strategy can be captured by

$$\begin{aligned}\Lambda &= L(Q_0 - Q_1 + Q_2 + \cdots + Q_{\mathsf{dsos}})L^{T} \\ &= (\sum_{\sigma \in \mathsf{LeftShape}} \lambda_{\sigma} M_{\sigma})(\sum_{\tau \in \mathsf{MidShape}} \lambda_{\tau} M_{\tau} + \sum_{\tau_P \in \mathrm{Int}} \lambda_{\tau_P} M_{\tau_P})(\sum_{\sigma' \in \mathsf{LeftShape}} \lambda_{\sigma'^{T}} M_{\sigma'^{T}})\end{aligned}$$

$+$ Truncation

At this point, a natural next step is to show $Q = Q_0 - Q_1 + \cdots + Q_{\mathsf{dsos}} \succeq 0$. To this end, we observe that there is a trivial identity component in Q which may be pulled out and serve as our PSD mass, and the subsequent goal is to show

$$\sum_{\tau \in \mathsf{MidShape}} \lambda_\tau M_\tau + \sum_{\tau_P \in \mathrm{Int}} \lambda_{\tau_P} M_{\tau_P}$$

to be of small norm (after appropriate factoring out the kernel due to indicators).

Concretely, we follow the previous works and adopt a two step strategy, we first identify the matrix norm of each shape that arises in the above, and then combine it with the given coefficient and show that it can be charged to the appropriate diagonal with a charging argument.

7.3.3 Main lemmas for PSDness

Lemma 7.3.2 (Independent-set indicator is PSD)**.** *Consider* Π *the diagonal matrix indexed by any subsets* S *such that* $|S| \leq \mathsf{dsos}$ *with* $\Pi[S, S] = 1[S \text{ is an Independent Set in } G]$*,*

$$\Pi \succeq 0$$

Proof. This is a diagonal matrix with non-negative entries. □

Lemma 7.3.3 (Middle shape is bounded)**.**

$$Q_0 = \sum_{\tau \in \mathsf{MidShape}} \lambda_\tau \cdot M_\tau \preceq \frac{1}{10}\Pi$$

Lemma 7.3.4 (Intersection is bounded)**.**

$$\sum_{0<i\leq \mathsf{dsos}} Q_i = \sum_i \sum_{\tau_P \in Int_i} \lambda_{\tau_P} \cdot M_{\tau_P} \preceq \frac{1}{10}\Pi$$

where Int_i is the set of intersection shapes of level i.

Lemma 7.3.5 (Truncation error)**.**

$$\textit{truncation error} \preceq n^{-\Omega(\mathsf{dsos}\log n)}\P\,.$$

Lemma 7.3.6 (Left-shape is well conditioned)**.**

$$\left(\sum_{\sigma\in \mathsf{LeftShape}} \lambda_\sigma M_\sigma\right)\left(\sum_{\sigma\in \mathsf{LeftShape}} \lambda_\sigma M_\sigma\right)^T \succeq n^{-O(\mathsf{dsos})}\Pi\,.$$

Assuming the above lemmas, we are able to complete the proof to our main theorem by filling in the missing piece about PSDness from previous section,

Proof of main theorem. Following our recursive factorization, we have

$$\begin{aligned}
\Lambda &= L(Q_0 - Q_1 + Q_2 + \cdots + Q_{\mathsf{dsos}})L^T \\
&= L\cdot\Pi^{1/2}(Id + Q_0 + \sum_i (-1)^i Q_i)\Pi^{1/2}\cdot L^T + \text{truncation} \\
&\succeq \Omega(1)\left(\sum_{\sigma\in \mathsf{LeftShape}} \lambda_\sigma M_\sigma\right)\left(\sum_{\sigma\in \mathsf{LeftShape}} \lambda_\sigma M_\sigma\right)^T + \text{truncation} \\
&\succeq (\Omega(1) - n^{-O(\mathsf{dsos})})\Pi \\
&\succeq 0
\end{aligned}$$

where the first psd inequality we plug in the norm bound for the middle shape and

intersection term, and then apply the bounds from singular value lower bounds and truncation error bound. □

7.3.4 Bounding the middle shapes

In this subsection, we show we can charge the middle shape MidShape the set of canonical middle shapes with the corresponding diagonal. We restate the main lemma of the section below,

We first identify what a middle shape is that arises in the approximate factorization machinery based on *minimum vertex separator*.

Definition 7.3.7 (Middle shape)**.** *A shape α is a middle shape if U_α and V_α are both minimum vertex separators for α.*

Overall strategy for middle shape Recall that the ultimate goal of this section is to show

$$Q_0 = \sum_{\tau \in \mathsf{MidShape}} \lambda_\tau \cdot M_\tau \preceq \frac{1}{10}$$

(ignoring the kernel from independent set indicator), we factorize out edges in $E(U\tau \cap V_\tau)$ as they form a corresponding independent-set indicator in our context, captured by a diagonal matrix Π, landing us at

$$\begin{aligned}\sum_{\tau \in \mathsf{MidShape}} \lambda_\tau M_\tau &= \Pi^{1/2} \cdot \left(\sum_{\substack{\tau: \text{middle shape} \\ E(U_\tau \cap V_\tau) = \emptyset}} \lambda_\tau \cdot M_\tau \right) \Pi^{1/2} \\ &= \Pi^{1/2} \cdot \left(\mathrm{Id} + \sum_{\substack{\tau: \text{ middle shape} \\ E(U_\tau \cap V_\tau) = \emptyset \\ \text{non-trivial}}} \lambda_\tau \cdot M_\tau \right) \Pi^{1/2}\end{aligned}$$

At this point, the PSDness of the above matrix boils down to the following lemmas,

Lemma 7.3.8 (Charging non-trivial middle shape)**.**

$$\left\| \sum_{\substack{\tau \\ \text{middle shape} \\ \text{non-trivial} \\ E(U_\tau \cap V_\tau)=\emptyset}} \lambda_{\tau_c} \cdot M_{\tau_c} \right\| \leq \frac{1}{10}$$

Overview of middle shape charging

1. For each middle shape, it can be charged to the appropriate diagonal (via section 7.3.5);

2. For each diagonal, there are not "too-many" middle shapes charged to it. A naive triangle inequality and union bound does not work well here, and we appeal to the grouping via active-profile of shapes(via section 7.3.6).

We first give an argument for charging a particular middle shape with certain gap, and then combine it with our counting of middle shapes.

7.3.5 Charging a single middle shape

The heart of this section is to show

Lemma 7.3.9. *For any middle shape τ,*

$$\lambda_\tau \cdot \|M_\tau\| \leq (c_\eta)^{|V(\tau) - \frac{|U_\tau|+|V_\tau|}{2}} \cdot (\frac{1}{\mathsf{dsos}^2})^{|V(\tau) - \frac{|U_\tau|+|V_\tau|}{2}|}$$

where we defer the specific $o(1)$ slack for counting at the moment. Recall from our graph matrix norm bounds,

Proposition 7.3.10. *For each shape α, it has norm bound*

$$\|M_\alpha\| \leq c_{\mathsf{norm}}^{|V(\tau)|} \cdot \max_{S:\text{separator}} \sqrt{n}^{|V(\tau)\setminus S|} \cdot \left(\sqrt{\frac{1-p}{p}}\right)^{|E(S)|} \cdot \mathsf{float}(\alpha) \cdot \mathsf{dangling}(\alpha)$$

Furthermore, recall that our middle shape coefficient is given by

$$\lambda_\tau = \left(\frac{k}{n}\right)^{|V(\tau)|-\frac{|U_\tau|+|V_\tau|}{2}} \cdot \left(\eta \cdot \sqrt{\frac{p}{1-p}}\right)^{E(\tau)}$$

Combining the above gives us

$$\lambda_\tau \cdot \|M_\tau\| \le c_{\mathsf{norm}}^{|V(\tau)|} \cdot (\frac{k}{n})^{|V(\tau)|-|U_\tau|} \max_{S:\text{separator}} \cdot \sqrt{n}^{|V(\alpha)\setminus S|} \cdot \left(\sqrt{\frac{1-p}{p}}\right)^{|E(S)|} \cdot \left(\eta \cdot \sqrt{\frac{p}{1-p}}\right)^{E(\tau)}$$
$$\cdot \mathsf{float}(\alpha) \cdot \mathsf{dangling}(\alpha)$$

Proof. Momentarily, let's ignore the dependence of dsos factor: the core of the argument relies upon the observation that we are able to assign a $\frac{1}{k}$ factor for each vertex outside the separator from the shape coefficient via a decomposition into left/middle/right part according to MVS,

1. We have a total vertex coefficient $\left(\frac{k}{n}\right)^{|V(\tau)|-|U_\tau|}$, and U_τ is by construction the MVS of τ while S is a separator, so $|S| \ge |U_\tau|$ and hence we have $\left(\frac{k}{n}\right)^{|V(\tau)\setminus S|} \le \left(\frac{k}{n}\right)^{|V(\tau)\setminus U_\tau|}$;

2. By connectivity of middle shape, we can consider a BFS from S, and we note the $\frac{k}{n}$ vertex coefficient shown from above is sufficient for charging the factors on vertices outside S;

3. For each vertex outside S, it is reachable from S, and we can consider the edge that explores the vertex in the BFS process, combining the factor of that particular edge with the vertex factor of the vertex gives,

$$\left|\sqrt{n} \cdot \left(\eta \cdot \sqrt{\frac{p}{1-p}}\right) \cdot \frac{k}{n}\right| \le c_\eta$$

where we recall our hardness assumption and $\frac{\sqrt{d}}{k} \le c_\eta$ (ignoring the extra decay

from $\frac{1}{\mathsf{dsos}}$ factors which are reserved for counting shapes) ;

4. Since each vertex outside the separator gives us a decay of c_η, and observe that by middle shape property, $|V(\tau) \setminus U_\tau \cap V_\tau| \geq \frac{1}{2}|V(\tau) \setminus S|$ as both U_τ and V_τ are the MVS, thus setting adjusting c_η to appropriate constant gives us

$$c_{\mathsf{norm}}^{|V(\tau)|} \cdot \frac{1}{c_\eta^{|V(\alpha)\setminus S|}} \leq (\frac{1}{10})^{V(\tau)\setminus U_\tau \cap V_\tau|}$$

5. For each edge leading to an explored vertex outside S, we pick up an edge factor of

$$\left| \left(\eta \cdot \sqrt{\frac{p}{1-p}} \right) \right| = O\left(\frac{1}{\sqrt{n}} \right)$$

6. For floating factor, we note that there is no floating component due to connected truncation;

7. For dangling factor, notice we pick up potentially one such factor for any dangling branch outside the separator, however, since each vertex outside U and V is of degree at least two, for each dangling branch, there is at least one edge unused for vertex connectivity to the separator in the prior charging, and hence we have a gap of at most

$$\left| \sqrt{\frac{p}{1-p}} \cdot (2|V(\alpha)|q_\alpha)^{O(1)} \right| \ll 1 ;$$

8. Finally, for edges inside the SMVS S, we have

$$\left| (k-1) \cdot \left(\eta \cdot \sqrt{\frac{p}{1-p}} \right) \cdot \sqrt{\frac{1-p}{p}} \right| = 1$$

□

7.3.6 Grouping middle shapes

To complete a bound for middle shape, we still need a count of them. Departing from the prior works that apply a union bounds over various shapes which leads to inevitable loss of poly log factors, we circumvent this issue by grouping middle shapes and apply trace-method on the gruped martrix.

Grouping shapes into permissible shapes

Definition 7.3.11 (Permissible shapes)**.** *We call a shape permissible if it can be obtained from the following process,*

1. *Start from any shape α such that any vertex not in $U_\alpha \cup V_\alpha$ has degree at least* 2*, and remove any edge between $V(\alpha) \setminus U_\alpha \cup V_\alpha$ and $U_\alpha \cap V_\alpha$;*
2. *For any dangling vertex, put in an edge from the dangling vertex to the lowest-ordered (in* dsos*) vertex in $U_\alpha \cap V_\alpha$;*
3. *For any floating component, it must have at least two edges that get removed, and put back the the two edges connecting to the lowest ordered vertices in $U_\alpha \cap V_\alpha$.*

Lemma 7.3.12. *For any set of potential edges E on vertices $V(E)$,*

$$\left| \sum_{E' \subseteq E} \prod_{e \in E'} \left(-\sqrt{\frac{p}{1-p}} \chi_E \right) \right| \leq (1 + \frac{p}{1-p})^{|E|} .$$

Proof.

$$\begin{aligned} \left| \sum_{E' \subseteq E} \prod_{e \in E'} \left(-\sqrt{\frac{p}{1-p}} \chi_e \right) \right| &= \prod_{e \in E'} \left| 1 - \sqrt{\frac{p}{1-p}} \chi_e \right| \\ &\leq (1 + \frac{1}{k-1})^{|E|} \end{aligned}$$

where the second inequality follows by casing on the edge present-status ,

1. If the edge is present,
$$1 - \sqrt{\frac{p}{1-p}}\chi_e = 1 - 1 = 0\,;$$

2. If the edge is missing,
$$1 - \sqrt{\frac{p}{1-p}}\chi_e = 1 + \frac{p}{1-p}\,;$$

□

Lemma 7.3.13 (Bounding the blow-up from hidden potential-edges)**.** *For each permissible shape τ, let E be the set of edges that can be added to τ such that τ remains a valid middle shape,*

$$\left|\sum_{E' \subseteq E} \prod_{e \in E'} \left(-\sqrt{\frac{p}{1-p}}\chi_e\right)\right| \leq 1 + o_n(1)$$

Proof. Notice each such edge has to go from a vertex outside of $U_\tau \cap V_\tau$ to $U_\tau \cap V_\tau$, and this is at most dsos choices for each edge outside, therefore $|E| \leq \mathsf{dsos} \cdot |V(\tau) \setminus U_\tau \cap V_\tau|$. Combining with the bound from previous, we have

$$\begin{aligned}\left|\sum_{E' \subseteq E} \prod_{e \in E'} \left(-\sqrt{\frac{p}{1-p}}\chi_e\right)\right| &\leq (1 + \frac{p}{1-p})^{|E|} \\ &\leq 1 + \frac{p}{1-p}\mathsf{dsos}|V(\tau) \setminus U_\tau \cap V_\tau| \\ &\leq 1 + o_n(1)\end{aligned}$$

where the last bound follows from our choice of $\mathsf{dsos} = o(n)$. □

The above prompts to define the following hidden edge indicator for any permissible shape,

Definition 7.3.14 (Hidden edge indicator)**.** *For a given permissible shape τ, let $E_{h(\tau)}$ be the set of hidden edges that can be added between $V(\tau) \setminus (U_\tau \cup V_\tau)$ and $U_\tau \cap V_\tau$ that does not violate the*

permissibility of τ (due to ordering of edge-removing), and we define

$$q(\tau) := \sum_{E' \subseteq E_{h(\tau)}} \prod_{e \in E'} \left(-\sqrt{\frac{p}{1-p}} \chi_e \right)$$

Furthermore, for our counting scheme to apply, it is also convenient for us to group the shapes according to which vertices in U_τ and in V_τ are incident to some vertices outside. Towards this end, we appeal to active-profile defined as the following,

Definition 7.3.15 (Active-profile)**.** *For any (permissible) middle shape τ, we call the set of indices $\mathcal{P} = \{U_{active}, V_{active}\} \subseteq [\mathsf{dsos}]^2$ an active profile for τ if any vertex $v \in U_{active}, V_{active}$ is incident to some vertex outside $U_\tau \cap V_\tau$.*

Lemma 7.3.16 (Bounding active-profiles)**.** *For each diagonal U, each active profile that gets charged to it can be identified at a cost of $\mathsf{dsos}^{|U_{active}|+|V_{active}}$.*

Definition 7.3.17 (Graph matrix with prescribed active-profile)**.** *Given an active profile $\mathcal{P}$, we define the corresponding graph matrix for the given profile as*

$$M_{\mathcal{P}} = \sum_{\tau: \text{shapes with prescribed active-profile}} M_\tau$$

Proof. Since active profile is a collection of middle shapes, by middle shape property, we have $|U_{active}| = |V_{active}|$. It then suffices for us to identify a label in dsos for each index in active-profile. □

7.3.7 Wrapping up middle shape bound

We are now ready to apply our matrix norm bounds on each active-profile.

Lemma 7.3.18 (Grouped middle shape bound)**.**

$$\sum_{\mathcal{P}:\textit{active profiles}:E(U_\tau)\cap E(V_\tau)=\emptyset} \|\lambda_{\mathcal{P}} M_{\mathcal{P}}\| \leq \frac{1}{10}$$

Proof. Applying our "grouped" norm bound on $\lambda_{\mathcal{P}}\mathsf{M}_{\mathcal{P}}$ for active-profile $\mathcal{P}$ gives us

$$\begin{aligned}\|\lambda_{\mathcal{P}}\mathcal{M}_{\mathcal{P}}\| &\leq \max_{\tau\in\tau_{\mathcal{P}}\text{non-trivial, permissible}} \|\lambda_\tau \cdot M_\tau \cdot q(\tau)\| \\ &\leq \max_{\tau\in\tau_{\mathcal{P}}\text{non-trivial, permissible}} \|\lambda_\tau \cdot M_\tau\| \cdot |q(\tau)|\end{aligned}$$

where we observe the following,

1. The coefficient is given by

$$\lambda_\tau = (\frac{k}{n})^{|V(\tau)|-\frac{|U_\tau|+|V_\tau|}{2}} \cdot \left(-\sqrt{\frac{p}{1-p}}\right)^{|E(\tau)|};$$

2. The norm bound, via the block-value bound, is given by

$$\max_{\tau,S} c_{\mathsf{norm}}^{|V(\tau)|} \cdot \sqrt{n}^{|V(\tau)\setminus S|} \left(\sqrt{\frac{n}{d}}\right)^{|E(S)|} \sqrt{n}^{|I(\alpha)|} \cdot \mathsf{dangling}(\alpha \setminus S) \cdot \mathsf{float}(\alpha);$$

3. The hidden edge indicator has magnitude bounded by

$$\|q(\tau)\| \leq (1+o_n(1))^{|V(\tau)\setminus U_\tau\cup V_\tau|};$$

We now mimic our charging strategy for a single middle shape, and note that the factor other than dsos dependence follows from the charging for a single shape, and we make clear the dsos dependence here as,

1. Assuming $\frac{k}{n} = O(\frac{1}{c_\eta\sqrt{d}\cdot\mathsf{dsos}^2})$;

2. Each vertex outside $U_\tau \cup V_\tau$ contributes a single factor of dsos via the blow-up from hidden-edge indicator, while each comes with a full factor of decay $\frac{1}{\mathsf{dsos}^2}$;

3. Each vertex in $U_\tau \Delta V_\tau$ does not contribute any dsos factor from hidden-edge indicator, and each comes with a factor of $\frac{1}{\mathsf{dsos}}$ since each comes with half a vertex coefficient;

4. By our sparsity bound, $|E(S)| \leq 5|V(S)|$, and moreover, notice since we look at permissible shape, there are no edges inside $E(U_\tau \cap V_\tau)$; and the edges between $U_\tau \cap V_\tau$ and $V(\tau) \setminus (U_\tau \cap V_\tau)$ are at most $2 \cdot |V(\tau) \setminus (U_\tau \cap V_\tau)|$, and therefore, we have $|E(S)| \leq 7|V(\tau) \setminus U_\tau \cap V_\tau| \leq c^{|V(\tau) \setminus U_\tau \cap V_\tau|}$ for some constant c subsumed by c_η factor of vertex-decay;

5. Combining the above gives us a factor of $\left(\frac{1}{\mathsf{dsos}}\right)^{|U_\tau \Delta V_\tau|} = \left(\frac{1}{\mathsf{dsos}}\right)^{|U_{active}(\mathcal{P})+|V_{active}(\mathcal{P})|}$;

Summing over all active-profiles gives us

$$\sum_{\mathcal{P}} \|\lambda_{\mathcal{P}} \mathcal{M}_{\mathcal{P}}\| \leq \max_{\mathcal{P}} c(\mathcal{P}) \cdot \left(\frac{1}{\mathsf{dsos}}\right)^{|U_{active}(\mathcal{P})+|V_{active}(\mathcal{P})|} \ll \frac{1}{10}$$

where we recall that the identification cost of $\mathcal{P}$ is bounded by $\mathsf{dsos}^{|U_{active}(\mathcal{P})|+|V_{active}(\mathcal{P})|}$. □

Remark 7.3.19. *Notice the above argument carries through for* $\frac{k}{n} = O(\frac{1}{\sqrt{d} \cdot \mathsf{dsos}^2})$ *while we in fact have a larger decay of* $\frac{1}{\sqrt{d} \cdot \mathsf{dsos}^4}$. *This is reserved for the extra* dsos*-factor needed for each vertex in intersection shape.*

This completes our proof to Lemma 7.3.3 by noting that putting edges from $E(U_\tau \cap V_\tau)$ back into anuy shape for any active-profile groups out the independent set indicator Π again.

7.3.8 Bounding intersection terms

Preliminaries for intersection term

Definition 7.3.20 (Ribbon composition)**.** *Given ribbons R_1, R_2, we call them* composable *if $V_{R_1} = U_{R_2}$. For ribbons with boundary set indexed by subgraphs, we additionally constrain $E(V_{R_1}) = E(U_{R_2})$. We use $R_1 \circ R_2$ to represent the ribbon from the composition of R_1 and R_2.*

Definition 7.3.21 (Proper composition)**.** *Given R_1, R_2 composable ribbons, we call them a proper composition if there is no surprise intersection beyond the boundary, i.e., $(V(R_1) \setminus V_{R_1}) \cap (V(R_2) \setminus U_{R_2}) = \emptyset$.*

Remark 7.3.22. *For properly composable ribbons R_1, R_2, we have $M_{R_1} M_{R_2} = M_{R_1 \circ R_2}$.*

Definition 7.3.23 (Improper shape and phantom graph)**.** *We call a shape improper if it contains multi-vertices (repeated vertices) or multi-edges. The underlying multigraph of an improper shape is also called a* phantom graph*, and we usually use τ_P to refer to both the improper shape and its underlying multigraph.*

Definition 7.3.24 (Shape composition)**.** *Given shapes α, β, we call them* composable *if $V_\alpha = U_\beta$. For composable shapes, we write $\zeta = \alpha \circ \beta$ for ζ any (possibly improper) shape whose multigraph can be obtained by composing ribbons of shape α and β.*

Unfortunately, it is no longer true that $M_\alpha M_\beta = M_{\alpha \circ \beta}$ as ribbon injectivity dose not hold beyond the anticipated boundary, and collision outside the boundary would inevitably produce improper shapes that are the main subject of this section.

Factorizing intersection terms We now make concrete our charging strategy for the intersection terms. Recall from our big picture of LQL^T decomposition, intersection can happen whenever shapes from L, Q, L^T intersect with one and another. For a left, middle, right shape σ, τ, σ' that intersect to some improper shape τ_P, we follow the recursive factorization framework from planted clique where we factorize the left/right shape from intersection term $\lambda_{\tau_P} M_{\tau_P}$ and attempt to charge to the corresponding diagonal of the new left/right shape.

With this big picture in mind, we are ready for more technical definitions.

Definition 7.3.25 (Intersected vertices $\mathrm{Int}(\tau_P)$)**.** *Given a composable pair of σ, τ, σ' that intersect to some improper shape τ_P, let $Int(\tau_P)$ be the set of vertices that gets intersected in τ_P, i.e., the vertices that appear with multiplicity greater than 1 in the associated multigraph of τ_P.*

Definition 7.3.26 (Left-intersection shape γ)**.** *A shape γ is a left-intersection color-shape if*

1. *γ comes with an an additional label for each vertex in $V(\gamma)$ specifying whether this vertex is in* to be intersected*, i.e., whether the vertex is in $Int(\gamma)$*
2. *V_γ. is the unique MVS for separating V_γ and U_γ;*
3. *U_γ is a MVS that separates $V_\gamma \cup Int(\gamma)$ and U_γ;*
4. *We follow the convention where we put edges in $E(V_\gamma)$ to the middle shape $E(U_\tau)$, hence, $E(V_\gamma) = \emptyset$.*

Consequently, we define the corresponding shape coefficient for γ to be

$$\lambda_\gamma := \left(\frac{k}{n}\right)^{|V(\gamma)| - \frac{|U_\gamma| + |V_\gamma|}{2}} \cdot \left(\eta \cdot \sqrt{\frac{p}{1-p}}\right)^{|E(\gamma)|}$$

Remark 7.3.27. *For our application, we will further factorize out the edges in $E(U_\gamma \cap V_\gamma)$ as they form the independent-set indicator, hence we actually have $E(U_\gamma \cap V_\gamma) = \emptyset$ (and similarly for γ') in our analysis.*

Analogously, we define the right-intersection color shape as the following.

Definition 7.3.28 (Right-intersection shape γ')**.** *A shape γ' is a left-intersection color-shape if*

1. *γ' comes with an an additional label for each vertex in $V(\gamma')$ specifying whether this vertex is in* to be intersected*, i.e., whether the vertex is in $Int(\gamma')$*

2. *$V_{\gamma'}$ is a MVS that separates $V_{\gamma'} \cup Int(\gamma')$ and $U_{\gamma'}$;*

3. *We follow the convention where we put edges in $E(U_{\gamma'})$ to the middle shape $E(V_\tau)$, hence, $E(U_{\gamma'}) = \emptyset$.*

Consequently, we define the corresponding shape coefficient for γ' to be

$$\lambda_{\gamma'} = \left(\frac{k}{n}\right)^{|V(\gamma')| - \frac{|U_{\gamma'}|+|V_{\gamma'}|}{2}} \cdot \left(\eta \cdot \sqrt{\frac{p}{1-p}}\right)^{|E(\gamma')|}$$

Definition 7.3.29 (Intersectable tuple (γ, τ, γ'))**.** *We call a composable tuple (γ, τ, γ') intersectable if there exists some intersection pattern τ_P s.t. any to-be-intersected vertex in γ and γ' gets intersected in τ_P, i.e.,*

$$Int(\gamma) \cup Int(\gamma') = Int(\tau_P)$$

For an intersectable tuple, we define its coefficient

$$\lambda_{\gamma\circ\tau\circ\gamma'} \coloneqq \lambda_\gamma \cdot \lambda_\tau \cdot \lambda_{\gamma'} = \left(\frac{k}{n}\right)^{|V(\gamma\circ\tau\circ\gamma')| - \frac{|U_\gamma|+|V_{\gamma'}|}{2}} \cdot \left(\eta \cdot \sqrt{\frac{p}{1-p}}\right)^{|E(\gamma\circ\tau\circ\gamma')}$$

Overview of intersection-term charging The charging strategy is largely similar to the 2-step charging strategy for middle-shape charging, despite each step may contain more technical twists.

1. For each intersection term $\gamma \circ \tau \circ \gamma'$ it can be charged to the appropriate diagonal U_γ or $V_{\gamma'}$ (via lemma 7.3.30) ;

2. For each diagonal, "not too many" intersection terms are being charged to it so that we can afford a union bound.

7.3.9 Charging a particular intersection shape

Lemma 7.3.30. *For intersection-tuple (γ, τ, γ') that intersect to τ_P,*

$$\lambda_{\gamma\circ\tau\circ\gamma'} \cdot \|M_{\tau_P}\| \leq o(1)$$

Since τ_P is a multigraph that arises from intersection, before we unpack the above equation, we need to first understand what our norm bounds $\|M_{\tau_P}\|$ yield for improper graphs coming from intersections.

Bounding $\|\mathsf{M}_{\tau_P}\|$

Fact 7.3.31. *For χ_e a p-biased Fourier character, we have*

$$\chi_e^k = \mathbb{E}[\chi_e^k] + \mathbb{E}[\chi_e^{k+1}] \cdot \chi_e$$

Definition 7.3.32 (Linearization of τ_P)**.** *Given a multigraph τ_P, we call ψ a linearization of τ_P if*

1. *Each edge in ψ is a (multi-)edge;*
2. *Each multiedge in τ_P has multiplicity 0 or 1 in ψ.*

Definition 7.3.33 (Vapor coefficient vap for mul-1 edge)**.** *Fix τ_P a multigraph, and $\psi(\tau_P)$ its linearization, for any $e \in \psi(E(\tau_P))$, we define its vapor coefficient to be*

$$vap(e) = mul_{\tau_P}(e) - 1$$

For intuition, any edge of multiplicity 2 in τ_P that linearizes to a mul-1 edge in ψ receives a vapor coefficient of 1.

Definition 7.3.34 (Vapor and phantom coefficient θ for $\psi(\tau_P)$)**.** *Fix τ_P a multigraph, and $\psi(\tau_P)$ its linearization, we define the corresponding vapor coefficient of ψ to be*

$$\theta_{\tau_P}(\psi) = \sum_{e \in E(\psi)} vap(e)$$

Furthermore, let $phantom_{\tau_P}(\psi)$ be the number of edges that disappear from $E(\tau_P)$ in the linearization ψ. The dependence on τ_P is dropped whenever it is clear.

Definition 7.3.35 (Floating stick)**.** *Given an improper graph τ_P and its linearization ψ, we call a component $C \in \psi$ a* floating stick *if*

1. *C is floating;*
2. *$V(C) = E_\psi(C) - 1$, i.e., this is path.*

Definition 7.3.36 (Floating cycle)**.** *Given an improper graph τ_P and its linearization ψ, we call a component $C \in \psi$ a* floating stick *if*

1. *C is floating;*
2. *$V(C) \geq E_\psi(C)$, i.e., this is at least a cycle.*

We remark that the above floating components are further distinguished as they have different contribution to matrix norm bounds: a floating stick C contributes norm $\sqrt{n}^{V(C)} \exp(-d \cdot (|V(C)| - 1))$ while a floating cycle would contribute norm $\max\left(\sqrt{n}^{V(C)}, (\sqrt{\frac{n}{d}})^{E_\psi(C)}\right) \cdot (2|V(\tau_P)|q_{\tau_P})$ depending on whether the floating cycle is part of the separator.

Proposition 7.3.37 (Norm bounds for improper shape from $\gamma \circ \tau \circ \gamma'$ intersection)**.** *Let $\gamma \circ \tau \circ \gamma'$ be an intersection-tuple that intersect to τ_P,*

$$\|M_{\tau_P}\| \leq c_{\mathsf{norm}}^{|V(\tau_P) \setminus U_{\tau_P} \cap V_{\tau_P}|} 2^{|E(\tau_P)|} \max_{\substack{(\psi, S) \\ \psi: \text{linearization of } \tau_P \\ S: \text{a separator for } \psi}} \left(\sqrt{\frac{1-p}{p}}\right)^{\theta(\psi)} \left(\sqrt{\frac{1-p}{p}}\right)^{E_\psi(S)} \sqrt{n}^{V(\tau_P) \setminus V(S)} \sqrt{n}^{I_\psi}$$

$$\cdot \mathsf{dangling}(\psi \setminus S) \cdot \mathsf{float}(\psi)$$

Proof. We highlight the difference from graph matrix norm bounds for usual color-shape, **Identifying the linearization:** The linearization ψ is fixed by identifying for each multi-edge whether it becomes mul-0 or 1 in ψ, hence $2^{|E(\tau_P)|}$ is sufficient where $|E(\tau_P)|$ is the number of distinct edges in τ_P;

Factors of n: To bound the factor of n, we note that its factor comes from the vapor coefficient via linearization and the norm bound factor for the linearization of ψ. Via our linearizartion, each multiplicity of edge that vaporizes gives a factor of $\sqrt{\frac{1-p}{p}}$, captured by the factor of $\left(\sqrt{\frac{1-p}{p}}\right)^{\theta(\psi)}$, combining with the norm bound factor for ψ gives us

$$\sqrt{\frac{1-p}{p}}^{\theta(\psi)} \cdot \max_{S\text{:separator for } \psi} \left(\sqrt{\frac{1-p}{p}}\right)^{E_\psi(S)} \sqrt{n}^{V(\tau_P)\setminus V(S)} \sqrt{n}^{I_\psi}$$
$$\cdot \mathsf{float}(\psi) \cdot \mathsf{dangling}(\psi \setminus S)$$

Combining the above gives us the desired norm bound. □

Now that we have identified the quantity of interest for $\|\mathsf{M}_{\tau_P}\|$, it is time to move on and see why the norm bound factor can be offset by $\lambda_{\gamma\circ\tau\circ\gamma'}$, and hopefully, be left with some gap (at least a constant per edge) for counting. Let's proceed by unpacking the desired equation,

$$\lambda_{\gamma\circ\tau\circ\gamma'}\|\mathsf{M}_{\tau_P}\| \leq \left(\frac{k}{n}\right)^{|V(\gamma\circ\tau\circ\gamma')|-\frac{|U_\gamma|+|V_{\gamma'}|}{2}} \cdot \left(\eta \cdot \sqrt{\frac{p}{1-p}}\right)^{|E(\gamma\circ\tau\circ\gamma')|} \cdot 2^{|E(\tau_P)|}$$
$$\cdot \max_{\substack{(\psi,S)\\ \psi\text{:linearization of } \tau_P\\ S\text{:a separator for } \psi}} \left(\sqrt{\frac{n}{d}}\right)^{\theta(\psi)} \left(\sqrt{\frac{n}{d}}\right)^{E_\psi(S)} \sqrt{n}^{V(\tau_P)\setminus V(S)} \sqrt{n}^{I_\psi} \cdot \mathsf{float}(\psi) \cdot \mathsf{dangling}(\psi \setminus S)$$

$$\leq \left(\frac{1}{c_\eta \cdot \sqrt{d}}\right)^{|V(\gamma\circ\tau\circ\gamma')|-\frac{|U_\gamma|+|V_{\gamma'}|}{2}} \cdot \left(\sqrt{\frac{d}{n}}\right)^{|E(\gamma\circ\tau\circ\gamma')|} \cdot 2^{|E(\tau_P)|}$$

$$\cdot \max_{\substack{(\psi,S)\\ \psi\text{:linearization of } \tau_P\\ S\text{:a separator for } \psi}} \left(\sqrt{\frac{n}{d}}\right)^{\theta(\psi)} \left(\sqrt{\frac{n}{d}}\right)^{E_\psi(S)} \sqrt{n}^{V(\tau_P)\setminus V(S)} \sqrt{n}^{I_\psi}$$

$$\cdot \mathsf{float}(\psi) \cdot \mathsf{dangling}(\psi \setminus S)$$

where we transfer the $\frac{1}{k}$ vertex factor to $\frac{1}{c_\eta\sqrt{d}}$ using our hardness assumption.

Controlling the factor of d and n for intersection term

In this subsection, the quantity of interest would be the factor of d and n. Our goal of this section is to show the following lemma,

Lemma 7.3.38. *For τ_P that arises from intersection of $\gamma \circ \tau \circ \gamma'$,*

$$\left(\frac{1}{c_\eta \cdot \sqrt{d}}\right)^{|V(\gamma\circ\tau\circ\gamma')|-\frac{|U_\gamma|+|V_{\gamma'}|}{2}} \cdot \left(\sqrt{\frac{d}{n}}\right)^{|E(\gamma\circ\tau\circ\gamma')|} \cdot 2^{|E(\tau_P)|} \cdot \max_{\substack{(\psi,S)\\ \psi\text{:linearization of } \tau_P\\ S\text{:a separator for } \psi}} \left(\sqrt{\frac{n}{d}}\right)^{\theta(\psi)} \left(\sqrt{\frac{n}{d}}\right)^{E_\psi(S)} \sqrt{n}^{V($$

$$\cdot \sqrt{n}^{I_\psi} \cdot \mathsf{float}(\psi) \cdot \mathsf{dangling}(\psi) \leq o(1)$$

Proof overview The above equation lies at the core of our argument for bounding intersection terms. It would be helpful to unpack and classify the terms from the equation into a couple of components (ordered by their relative magnitude).

To start with, each vertex (outside the separator) contributes a factor of $\sqrt{n}$ to the above equation (ignoring its potential dangling/floating factor captured in a separate argument), it suffices for us to assign a factor of $\sqrt{d}$ for each such vertex, and suppose the vertex is

connected to the separator S in τ_P, we can offset its contribution via

$$\frac{1}{\sqrt{d}} \cdot \sqrt{n} \cdot \sqrt{\frac{d}{n}} \leq 1$$

Similarly, for isolated vertex, we need to assign an extra factor of $\sqrt{d}$ and an extra multiplicity of edges in $E(\gamma \circ \tau \circ \gamma')$. Besides connectivity in τ_P that requires some work, the major ingredient for this component is the vertex-intersection-tradeoff that says we have enough $\frac{1}{\sqrt{d}}$ factors for vertices outside the new separator after the intersection, captured by the following lemma,

Lemma 7.3.39 (Vertex factor for intersection (Intersection trade-off lemma from [JPR+22, PR20]))**.** *For $\gamma \circ \tau \circ \gamma'$ that intersects to τ_P,*

$$|V(\gamma \circ \tau \circ \gamma')| - \frac{|U_\gamma| + |V_{\gamma'}|}{2} \geq |V(\tau_P) \setminus V(S)| + |I_\psi|$$

for any linearization ψ and separator S of ψ;

Moreover, we need to assign at least one edge in $E(\gamma \circ \tau \circ \gamma')$ for each vertex in $V(\tau_P) \setminus V(S)$, and potentially more if they are isolated outlined by the above argument. Additionally, we need to identify extra $\sqrt{\frac{d}{n}}$ factor to handle dangling and floating components. The extra gap here relies upon the k-wise symmetry of our coefficients, that we start with γ,τ,γ' that do not have dangling branches in the very beginning; hence, any component that becomes floating/dangling is a result of over-linearization (i.e., multi-edges become mul-0). That being said, multi-edges turning mul-0 can in general be tricky as they are the reason we pick up extra factors for isolated vertices (and dang/floating factors). Towards this end, we first observe the following,

Proposition 7.3.40 (One phantom edge-mul for one dangling/floating factor)**.** *It suffices for us to assign one edge-multiplicity to each dangling/floating factor.*

Proof. Observe that we pick up a coefficient $\sqrt{\frac{d}{n}}$ for each edge's phantom multiplicity, and each dangling/floating factor is either $2|V(\psi)|q_\psi$ or $\sqrt{n}\exp(-d)$ (in which case the component is outside the separator). □

At this point, we would like an edge-factor assignment scheme that allows us to assign edges (specifically their multiplicities) in $E(\gamma \circ \tau \circ \gamma')$ such that

Edge-factor assignment target: necessary condition

1. Assign one multiplicity to each multiplicity that vaporizes in $\theta(\psi)$ and separator edges in $E_\psi(S)$;
2. Assign at least one edge's multiplicity to each non-isolated vertex in $V(\tau_P) \setminus V(S)$;
3. Assign at least two multiplicities to each isolated vertex;
4. Assign at at least multiplicity $c > 0$ (where c may be fractional if needed) to each floating component or dangling branch to offset the dangling/floating factor.

We note that the following target would also be sufficient, as it strengthens the last condition to each multiplicity being charged to at most one dangling/floating factor.

Edge-factor assignment target: sufficient condition

1. Assign one multiplicity to each multiplicity that vaporizes in $\theta(\psi)$ and separator edges in $E_\psi(S)$;
2. Assign at least one edge's multiplicity to each non-isolated vertex in $V(\tau_P) \setminus V(S)$;
3. Assign at least two multiplicities to each isolated vertex;

4. Assign at at least one multiplicity to each floating component or dangling branches to offset the dangling/floating factor.

Before we delve into the argument, we remind the reader that phantom(ψ)) is the multiplicity of edges that vanish from $E(\tau_P)$, i.e., of multiedges that become multiplicity 0 in ψ. And instead of working with $E(\gamma \circ \tau \circ \gamma')$ that counts multiplicity of edges, it is easier for us to consider the (non multi-)graph ψ and phantom(ψ) that allows us to avoid worrying about vapor edges and their multiplicities as well.

Claim 7.3.41. *Recall $\theta(\psi)$ is the multiplicity of edges that vaporize in the linearization to ψ from τ_P,*

$$|E_\psi| + \mathit{phantom}(\psi) = |E(\gamma \circ \tau \circ \gamma')| - |\theta(\psi)|$$

Lemma 7.3.42 (Edge-assignment)**.** *For $\gamma \circ \tau \circ \gamma'$ that intersects to τ_P and let ψ be a linearization of τ_P, s.t.*

$$|E_\psi| + \mathit{phantom}(\psi) \geq |V(\tau_P) \setminus V(S)| + |E_\psi(S)| + |I_\psi| + |\mathsf{dangling}_\psi| + |\mathsf{float}_\psi| \tag{7.1}$$

where we define

$$|\mathsf{float}_\psi| := |\mathsf{float\text{-}stick}_\psi| + |\mathsf{float\text{-}cyc}_\psi|$$

and remind the reader that phantom(ψ) counts the multiplicity of edges that completely vanish in ψ from linearization τ_P.

Proof to lemma 7.3.42. We start by considering ψ the linearized graph (recall ψ may have phantom edges removed) and put phantom edges back along the way. Take S to be the SMVS for ψ, we then consider the following recursive process to traverse ψ via edges in $E(\tau_P)$.

Throughout the process, vertices in the graph $V(\psi)$ can be partitioned as the following,

1. $V_W \subseteq V(\tau_P)$: those reachable from W via edges in E_ψ or already in W;

2. For a connected component $C \subseteq V(\psi) \setminus V_W$ in E_ψ while not yet reachable from W, it is either a non-floating component, a floating-stick, or a floating-cyc;

3. For isolated vertices in $V(\psi) \setminus V_W$, we can group them according to the phantom edges into a collection of iso-connected components.

W-exploration procedure

1. Let W be the current set of vertices visited (initialized to be S an arbitrary SMVS of ψ);

2. Explore the vertices (not yet in W) while connected to W via edges in $E(\psi)$, i.e., assign the edge to each vertex it leads to;

3. Process the dangling/floating factor for vertices in W, and this may potentially explore component connected to W via phantom edge;

4. Explore a component connected to W via some phantom edge;

5. For vertices outside W and not reachable via phantom from W, there must be a phantom-edge connecting two different components, process that phantom edge and explore both components;

6. Repeat this process until all vertices are pushed into W.

For starters, we first notice for the component C_S that is connected to the separator of ψ, we have

$$|E_\psi(C_S)| \geq |V(C_S) \setminus V(S)| + |E_\psi(S)|$$

as each vertex is connected to the separator, and we can charge it to the edge leading to the vertex when we BFS from S, and each separator edge in $E_\psi(S)$ contributes an 1 to both sides. That being said, we may have dangling/floating factors not yet accounted in C_S, not to mention there may be components not reachable from S in ψ.

For the subsequent steps involving dangling/floating factor , we need to repeatedly apply the following proposition which allows us to identify either there is a new phantom edge incident or there is some excess edge not used in the *exploration* process in the component,

Proposition 7.3.43 (Degree-at-least-two)**.** *In τ_P where phantom edges are yet to vanish, each vertex in $V(\tau_P) \setminus (U_{\tau_P} \cup V_{\tau_P})$ have degree at least 2 (not counting multiplicities).*

Proof. Each vertex in $V(\tau_P) \setminus (U_{\tau_P} \cup V_{\tau_P})$ comes from at least one of the following two categories, $V(\gamma) \setminus U_\gamma, V(\gamma') \setminus V_{\gamma'}$ and $V(\tau) \setminus (U_\tau \cup V_\tau)$. Any vertex in the first category has degree at least 2 in γ from γ shape property that V_γ is the MVS for γ (analogously for γ'), and any vertex in $V(\tau) \setminus U_\tau \cup V_\tau$ has degree at least 2 in τ by k-wise symmetry. □

Let W be the current set, and let $E_\psi(W), \text{phantom}_W(\psi)$ be the multiplicity of edges traversed so far, and $I_\psi, \mathsf{dangling}_\psi, \mathsf{float}_\psi$ be the factors processed so far. We maintain the invariant,

$$|E_\psi(W)| + \text{phantom}_W(\psi) \geq |V(W) \setminus V(S)| + |E_\psi(S)| + |I_\psi(W)| + |\mathsf{dangling}_\psi(W)| + |\mathsf{float}_\psi(W)|$$

In the base case, we have $W = S$, and the inequality holds immediately. As noted above, it is also immediate for $V_W = V(C_S)$. We now start processing potential dangling/floating factors on vertices in V_W. Notice in the base case, the only factors involved are dangling factors, and we can restrict our attention to its end-point, and notice it is either incident to an uncharged edge in E_ψ, or a phantom edge via Proposition 7.3.43. For a phantom edge, we discuss by cases depending on where it leads to, as it may incur new floating factors for which we need to either identify there is either some excess edge in the new component, or a new phantom edge incident to the component.

1. The phantom edge leads to another vertex in W, and the vertex it leads to may be

assigned another dangling/floating factor as well, in which case we pick up at least 2 (from phantom multiplicities) and 2 from RHS for 2 dangling/floating factors;

2. The phantom edge leads to a component C not yet explored, i.e., not in W, in which case we pick up at least 2 phantom multiplicities from the LHS. We assign one multiplicity to the current dangling/floating factor being processed with one remaining phantom multiplicity yet-to-assign . It now remains to assign this multiplicity and identify an excess edge for the floating factor of the new component,

   - If C is a floating-stick, assign the above multiplicity to the vertex in C incident to the current phantom edge, which also puts C into W. Moreover, we claim that $V(C)$ must have at least one more degree-1 vertex in ψ that is incident to some not yet processed phantom edge by Proposition 7.3.43 (notice we get extra gap if C has more than 2 deg-1 vertices);
   - If C is a floating-cycle, it must be outside S, while $E_\psi(C) \geq V_\psi(C)$, hence we can use the remaining multiplicity from earlier to assign to the floating factor of C, and use the edges in $E_\psi(C)$ to *explore* vertices in $V_\psi(C)$;
   - If C is non-floating, (eg. $V(C)$ intersects with $U_\psi \cup V_\psi$), there is no dangling/floating factor on C, and we assign the above multiplicity to the vertex in C incident to the current phantom edge;
   - If C is an iso-connected component, i.e., a connected component in τ_P that become isolated under ψ, since each vertex in $V(C)$ is of degree at least 2 in τ_P, assign the remaining multiplicity from above to the first vertex in $V(C)$. Note that since it is isolated, it remains for us to find assign this *source* vertex another edge-multiplicity. Putting that aside, we can traverse the iso-connected component, and charge each phantom edge (and its 2 multiplicities) to the new vertex it explores. Since each vertex has degree at least 2, there must be an

incidental phantom edge, and we can assign one of its multiplicities to the *source* vertex of this component, filling the gap from earlier. The other multiplicity is then reserved to explore the potentially new component.

For components not yet explored, and they are not connected to the current W via any phantom edge, we claim that there must be a phantom edge that connects two unexplored components. This follows via our intersection-tuple property, that in τ_P, each vertex has a path to U_ψ and a path to V_ψ. Hence it is either reachable from the separator in ψ, or reachable from some phantom edge. We can then use this two multiplicities to *explore* these two components into W, and notice each of them may be floating, while we can follow the above argument and identify a new phantom edge or an excess edge for each floating component. This completes our proof of Lemma 7.3.42. □

7.3.10 Bounding intersection count

Analogous to our charging in the base level for middle shapes, as we have shown each single intersection term can be bounded, we may now complete the proof by bounding the count. That said, we first notice that each intersection term can be specified as the following.

Identifying what to bound

Definition 7.3.44 (Intersection-pattern)**.** *We call a tuple* $(\gamma_t, \dots, \gamma_1, \tau, \gamma'_1, \dots, \gamma', P_{Int})$ *an intersection pattern such that*

1. P_{Int} *specifies which vertices get intersected and how they are intersected, and let* $P_{Int,L,R}$ *be* P_{Int} *restricted to the first* (L, R)*-levels* $\gamma_L, \dots, \gamma_1, \tau, \gamma'_1, \dots, \gamma'_R$.
2. *For any* $\ell, r \in \mathbb{N}$, $\left(\gamma_\ell, (\gamma_{\ell-1}, \dots, \gamma_1, \tau, \gamma'_1, \dots, \gamma'_{r-1}), \gamma_r\right)$ *is a intersection-tuple with* $P_{\ell,r}$;

3. *Additionally, call* $\text{depth}(P_{Int}) = l$ *the intersection-depth of the corresponding pattern;*

4. *For technical reasons, we allow* $\gamma_i = \gamma_{i-1}$ *or* $\gamma'_i = \gamma'_{i-1}$ *for* $i > 1$ *to represent the case when there is no* new *left/right intersection. However, both cannot hold at the same time.*

Definition 7.3.45 (Intersection shape τ_P)**.** *We call a shape an intersection shape if it it a shape that can be obtained by ribbon intersections of at most* dsos *levels.*

To bound the count of intersection terms, we first observe that τ_P when viewed as shape can be counted in the identical way as for middle shape, with the only difference being for a fixed shape, there may be various ways of ribbon interesctions (intersection patterns) that obtain the same underlying multi-graph as a shape. That said, to the number of intersection patterns for a fixed shape, it suffices for us to identify a dsos label for each edge specifying which level the edge comes from (i.e. the depth of γ_i where the edge is from).

As a result, given the final shape τ_P viewed as a multi-graph with information of where each edge comes from, the intersection pattern is immediately fixed as one may

1. Draw out each $\gamma_i, \gamma_i'^T$ for each $i \in [\text{dsos}]$ and τ (the base-level) according to the depth-information from each edge;

2. The vertex intersection, i.e. which vertex colliding with each other, is also fixed as the final shape of τ_P is also given.

Proposition 7.3.46. *Given a shape* τ_P*, the number of intersection levels can be identified at a cost of*

$$c_{int}^{|V(\tau_P)\setminus U_{\tau_P}\cap V_{\tau_P}|} \cdot \text{dsos}^{|V(\tau_P)|}$$

for some absolute constant c_{int} *where we remind the reader that* $|V(\tau_P)|$ *counts multiplicity of intersected vertices.*

Proof. We apply this bound alongside our norm bound for grouped matrix. In addition to the block-value bound for middle shape, it suffices for us to go through the shape in an extra pass and identify where each edge comes from (i.e., identifying the particular intersection pattern).

First, identify a level of [dsos] for each vertex-multiplicity identifying which dsos level it appears, and this is subsumed by the extra decay factor of $\frac{1}{\text{dsos}}$ from our coefficient; once all the dsos levels are identified, we appeal to the following structural left/right-intersection shape property: each γ-shape is U-wedge.

That said, once τ_P is identified as an edge-set in the trace-method calculation alongside the [dsos] label for each vertex-multiplicity we are provided now, it suffices for us to apply wedge-bundling idea for each γ_i, γ_i^T level recursively from outermost level to the base level of τ, and applying lemma 2.5.19, this is a single constant per vertex and this completes our bound. □

7.3.11 Wrapping up the intersection shape bound

Proof to Lemma 7.3.4. This section is in parallel with the summary for middle shape bound. Notice that for a single active-profile $\mathcal{P}$, apply our graph matrix bound on

$$\lambda_{\mathcal{P}}\mathcal{M}_{\mathcal{P}} := \sum_{\substack{\tau_P:\text{intersection shapes with active profile}\mathcal{P} \\ \text{non-trivial, permissible} \\ E(U_\tau \cap V_\tau)=\emptyset}} \lambda_{\tau_P} \cdot M_{\tau_P} \cdot q(\tau)$$

Applying our "grouped" norm bound on $\lambda_{\mathcal{P}}\mathsf{M}_{\mathcal{P}}$ for active-profile $\mathcal{P}$ while now $\mathcal{P}$ is the collection of intersection shapes τ_P gives us

$$\|\lambda_{\mathcal{P}}\mathcal{M}_{\mathcal{P}}\| \leq \max_{\tau \in \tau_P(\mathcal{P})\text{non-trivial, permissible}} \|\lambda_{\tau_P} \cdot M_{\tau_P} \cdot q(\tau_P)\|$$

$$\leq \max_{\tau_P \in \tau_P(\mathcal{P}) \text{non-trivial, permissible}} \|\lambda_{\tau_P} \cdot M_{\tau_P}\| \cdot |q(\tau_P)|$$

We now mimic our charging strategy for middle shapes , and we note that the factor other than dsos dependence follows from the charging for a single intersection shape in Lemma lemma 7.3.38, and we make clear the dsos dependence here as,

1. Assuming $\frac{k}{n} = O(\frac{1}{c_\eta \sqrt{d} \cdot \text{dsos}^4})$, we first consider the following dsos dependence that inheir from that of middle shapes.;

2. Each vertex outside $U_\tau \cup V_\tau$ contributes a single factor of dsos via the blow-up from hidden-edge indicator, while each comes with a full factor of decay $\frac{1}{\text{dsos}^2}$;

3. Each vertex in $U_\tau \Delta V_\tau$ does not contribute any dsos factor from hidden-edge indicator, and each comes with a factor of $\frac{1}{\text{dsos}}$ since each comes with half a vertex coefficient;

4. By our sparsity bound, $|E(S)| \leq 5|V(S)|$, and moreover, notice since we look at permissible shape, there are no edges inside $E(U_\tau \cap V_\tau)$; and the edges between $U_\tau \cap V_\tau$ and $V(\tau) \setminus (U_\tau \cap V_\tau)$ are at most $2 \cdot |V(\tau) \setminus (U_\tau \cap V_\tau)|$, and therefore, we have $|E(S)| \leq 7|V(\tau) \setminus U_\tau \cap V_\tau| \leq c^{|V(\tau) \setminus U_\tau \cap V_\tau|}$ for some constant c subsumed by c_η factor of vertex-decay;

5. The difference in comparing to the middle shape charging is that we incur an overhead to identify intersection patterns, and this is a factor of an extra dsos for each vertex from Proposition 7.3.46;

6. Recall that we have an $\frac{1}{\text{dsos}}$ excess from charging middle shapes, which can now be used to offset the extra dsos overhead in Rmemark 7.3.19;

7. Combining the above gives us a factor of $\left(\frac{1}{\text{dsos}}\right)^{|U_\tau \Delta V_\tau|} = \left(\frac{1}{\text{dsos}}\right)^{|U_{active}(\mathcal{P})| + |V_{active}(\mathcal{P})|}$.

Summing over all active-profiles gives us

$$\|\sum_{\mathcal{P}} \lambda_{\mathcal{P}} M_{\mathcal{P}}\| \leq \max_{\mathcal{P}} c(\mathcal{P}) \cdot \left(\frac{1}{\mathsf{dsos}}\right)^{|U_{active}(\mathcal{P})+|V_{active}(\mathcal{P})|} \ll \frac{1}{10}$$

where we recall that the identification cost of $\mathcal{P}$ is bounded by $\mathsf{dsos}^{|U_{active}(\mathcal{P})|+|V_{active}(\mathcal{P})|}$. Putting back the edges in $E(U_{\tau_p} \cap V_{\tau_p})$ gives us back the indepednent set indicator on the left and right again, and this completes our bound for the intersection terms.

□

Chapter 8

Sum-of-Squares Lower Bounds for Coloring Random Graphs

8.1 Our Results

In this work, we give higher-degree Sum-of-Squares lower bounds, and confirm that the Sum-of-Squares bound for coloring essentially matches what one would naturally expect from the independent-set bound. To set the stage for our results, we recall the generic formulation of coloring in the SoS framework.

Definition 8.1.1 (Axioms for graph-coloring)**.** *Let G be an n-vertex graph and let k be the number of colors. The following axioms describe the $\{0,1\}$ indicators x of k-coloring in G:*

$$\forall (i,c) \in [n] \times [k],\ x_{i,c}^2 = x_{i,c} \quad \textit{(Booleanity);}$$
$$\forall c \in [k],\ \forall (i,j) \in E(G),\ x_{i,c}x_{j,c} = 0 \quad \textit{(Independent set in the same color)};$$
$$\forall i \in [n],\ \sum_{c \in [k]} x_{i,c} = 1 \quad \textit{(Exact-coloring)}\ .$$

Theorem 8.1.2 (Dense regime)**.** *There is an absolute constant $c_0 \in \mathbb{N}$ such that for sufficiently large $n \in \mathbb{N}$, and parameters k,* dsos *satisfying*

$$k \leq \Theta\left(\sqrt{n} \cdot (2^{\mathsf{dsos}^{c_0}} \cdot \log^{c_0} n)\right) ,$$

*there is a pseudo-expectation of degree-*dsos *for k-coloring with high probability over G sampled from $G(n, \frac{1}{2})$.*

Despite our focus being the dense regime throughout the work, our result also translates to the sparse regime,

Theorem 8.1.3 (Sparse regime)**.** *There is an absolute constant $c_0 \in \mathbb{N}$ such that for sufficiently large $n \in \mathbb{N}$ and $d \in [(\log n)^2, n^{0.5}]$, and parameters k,* dsos *satisfying*

$$k \leq \Theta\left(\frac{n}{\sqrt{d}} \cdot (\mathsf{dsos} \cdot \log n)^{c_0}\right) ,$$

*there is a pseudo-expectation of degree-*dsos *for k-coloring with high probability over G sampled from $G(n, \frac{d}{n})$.*

Remark 8.1.4. *Our bound for coloring, when restricted to a single color, can be viewed as a result for independent set, and matches the best known SoS lower bound for the independent set problem up to low-order dependence of* dsos *in both regimes.*

8.2 Introduction and Motivation: Why Another SoS Lower Bound?

Among various classic computational problems, graph coloring and independent set are arguably the most intimate duo. Both are in Karp's original 21 NP-Complete problems,

and algorithms or hardness results for one of these problems usually correspond to algorithms or hardness results for the other problem. Putting aside their computational complexity, graph colorings and independent sets have also been extensively investigated in probabilistic combinatorics. For random graphs, it is well-known that the thresholds for these two problems are closely related: Given a random graph $G \sim G(n, \frac{1}{2})$, with high probability the size of the largest independent set in G is $\Theta(\log n)$ and the chromatic number of G is $\Theta(\frac{n}{\log n})$.

It is natural to wonder whether such a close connection continues to hold in the computational lens for the questions of certifying an upper bound on the maximum size of an independent set and certifying a lower bound on the chromatic number for a random graph $G \sim G(n, \frac{1}{2})$. With high probability, a spectral bound certifies that the maximum size of an independent set of G is $O(\sqrt{n})$ which implies that the chromatic number of G is at least $\Omega(\sqrt{n})$. That said, it is possible that more powerful proof systems can give a better bound for one or both problems.

For independent set, this question has been analyzed through the seminal planted clique problem [Jer92, Kuc95] which asks if we can distinguish a random graph from a graph where we start with a random graph and then plant a large clique. A long line of work has shown that various restricted computational models including Lovász-Schrijver [FK03], statistical queries [FGR+17], Markov Chains [GZ19, CMZ23] and Sum-of-Squares [BHK+16, JPR+22, KPX24] cannot distinguish these graphs when the size of the planted clique is much smaller than $\sqrt{n}$. This implies that none of these models can give a bound which is much better than $\sqrt{n}$ for the size of the largest independent set in a random graph (the clique and independent set problems are equivalent by taking the complement of the graph). However, we have few results about the hardness of certifying a lower bound on the chromatic number of a random graph. For the basic semidefinite program (aka. degree-2 SoS/ Lovász-Theta function), we know that the corresponding threshold

of $\Theta(\sqrt{n})$-colors is essentially tight [Juh82, Co05b]. Prior to our work, we did not even know whether this hardness result extends to degree 4 SoS.

There were two reasons for this gap in the hardness results. First, if we try and plant a k-coloring in a random graph by dividing the graph into k independent sets evenly and removing all edges within these independent sets, it is actually easy to distinguish between the resulting graph and a random graph for any $k \leq \Theta(n^{3/4})$. Finding a quieter way to plant a coloring in a random graph is an interesting open problem. The recent work of [KVWX23] gives a more refined planted distribution that is quiet for $k \geq \Theta(n^{2/3})$ but remains a polynomial factor away from the desired $\Theta(\sqrt{n})$-threshold. Secondly, the global constraint of each vertex receiving exactly one color also poses significant technical challenges for the analysis. In fact, we generally don't know how to prove SoS lower bounds for problems with global constraints. The two methods which have been found so far for analyzing SoS with global constraints are reducing the global constraint to local constraints [KOS19] and using a certain nullspace trick [GJJ$^+$20b]. A prior result by [KM21] ignores the global constraint, and gives higher degree SoS hardness for multi-coloring in $G(n, \frac{1}{2})$ via reductions from independent-set [BHK$^+$16]. In this paper, we overcome both of these challenges to prove an SoS lower bound for the coloring problem.

8.2.1 Background and Prior Work

The Sum-of-Squares hierarchy The Sum-of-Squares hierarchy (SoS), which was independently studied by Shor [Sho87], Parillo [Par00], Nesterov [Nes00], Lasserre [Las01], and Grigoriev [Gri01], is a hierarchy of convex relaxations for polynomial optimization problems which is based on the Positivstellensatz/sum of squares proof system studied by Hilbert more than a century ago. SoS is parametrized by the *SoS degree* d where as d increases, degree d SoS becomes more accurate but corresponds to a larger semidefinite

program. Roughly speaking[1], degree d SoS can be evaluated in time $n^{O(d)}$.

SoS is surprisingly powerful. SoS (together with a rounding algorithm to round a solution to the convex relaxation to an actual solution) captures the best known algorithms for max cut, sparsest cut, and unique games [GW95, ARV04, ABS15] and has been used to give algorithms for many more problems including dictionary learning, planted sparse vector, tensor decomposition, tensor completion, quantum separability, solving unique games on certified small set expanders, and many high dimensional statistical inference problems and robust estimation problems [BRS11, BBH$^+$12, MSS16, BKS17, KSS18, HL18, KKM20, Hop19, BK21, BBK$^+$21a, BHKL22]. We refer interested readers to [BS14, RSS19, FKP19] for a thorough introduction to Sum-of-Squares algorithms.

Sum-of-Squares Lower Bounds for Average Case Problems Classical complexity theory analyzes the complexity of problems on worst case inputs. However, problems which are hard in the worst case may still be easy on random instances or instances which appear in practice. Thus, it is also important to analyze average case problems where the input is random rather than worst case. These average case problems are generally not NP-hard so to show average case hardness results, we need to either use different assumptions such as the planted clique conjecture (which says that the planted clique problem is hard) or restrict ourselves to specific models.

Over the past decade or so, there has been quite a bit of progress in analyzing the complexity of average case problems using both of these approaches. Brennan et al. showed that planted clique can be reduced to many other problems so these problems are hard assuming the planted clique conjecture [BBH18, BB20, BABB21, HS21]. On the lower bound side, lower bounds have been shown for several different models including statistical queries, low-degree polynomials, the Sherali-Adams hierarchy, and the sum of

[1]As observed by O'Donnell [O'D17], this is not always true as the optimal solution to an SDP may involve coefficient of doubly exponential size.

squares hierarchy. Of these models, SoS is the strongest[2] and thus the hardest to prove lower bounds for.

Since the pioneering work of Barak et al. [BHK+16] showing SoS lower bounds for planted clique[3], SoS lower bounds have been shown for several other fundamental average case problems, namely tensor and sparse PCA [HKP+17, PR20], the Sherrington-Kirkpatrick problem [GJJ+20b], independent set on sparse graphs[JPR+22, KPX24], densest k-subgraph [JPRX23], and non-Gaussian component analysis [DKPP24]. That said, prior to our work, proving even a degree 4 SoS lower bound for k-coloring was open.

Remark 8.2.1. *Grigoriev's SoS lower bound for 3-XOR and the vast generalization of this bound to all CSPs where the predicate supports a pairwise uniform distribution of solutions by Barak, Chan and Kothari [BCK15], and subsequently Kothati, Mori, O'Donnell, and Wittmer [KMOW17] are also average-case lower bounds, though these lower bounds use different techniques.*

Low-Degree Polynomial and Basic Level Local Statistics Hardness Closely related to the Sum-of-Squares lower bounds, the low-degree polynomial framework is another active platform for examining average-case hardness[KWB19a, BMR19, BBK+20, Wei20, SW22, GJW22, RSWY22, KVWX23]. It provides a common framework for establishing preliminary evidence of algorithmic hardness, and has even found its application in the recovery and search variants of the problem beyond the vanilla distinguishing question where the Sum-of-Squares formulation is less clear. However, for the distinguishing problems that are applicable to both platforms, low-degree hardness is strictly captured by variants of Sum-of-Squares lower bounds, and it remains a major research agenda to understand their potential gap.

[2]More precisely, SoS is strictly more powerful than Sherali-Adams, an SoS lower bound proved via pseudo-calibration implies a low-degree polynomial lower bound, and low-degree polynomials and [BBH+20, BEAH+24] showed that statistical queries and low-degree polynomials are equivalent under mild technical conditions

[3]This lower bound did not include a constraint on the exact size of the clique. Pang [Pan21] fixed this by showing an SoS lower bound for planted clique with a constraint on the exact size of the clique

Most pertinent to us is the work of [KVWX23] where they consider the same question of coloring in $G(n, \frac{1}{2})$, and give low-degree polynomial hardness for refuting $k \leq O(n^{\frac{2}{3}})$-coloring of the graph. Most importantly, they leave the problem of closing the gap to $O(\sqrt{n})$ as an open problem. As we shall soon elaborate, the absence of low-degree hardness is also an exciting challenge for us, and the starting point of our main technical work. This is the focus of section 8.3.1.

Another line of work closely related to ours is that of [BKM19, BBK+20] which considers the coloring question in the sparse regime. However, their hardness result is essentially limited to the degree-2 Sum-of-Squares (basic level Local Statistics) and low-degree polynomials. Besides the restriction to degree-2 SoS, their hardness result is specific to pinning down the threshold to the tight constant in the sparse regime of $G(n, \frac{d}{n})$ for constant d, and does not apply for even the slightly denser regime of $d = o(n)$, nor the dense regime in $G(n, \frac{1}{2})$: the game in the sparse regime is all about the right constant, whereas our focused dense regime had a polynomial gap in the threshold prior to this work. It is also an important question on its own whether in the sparse regime one can obtain hardness result against higher-degree Sum-of-Squares, and pinning down to the right constant against degree-4 SoS is already fairly exciting at this point. Along this line, it is conceivable that combining [KPX24] and our result, one can get a lower bound in the regime of $d = O(1)$ against higher constant-degree SoS though with an objective value which is lossy in the leading constant.

8.3 Technical Overview

8.3.1 The Absence of Low-degree Hardness

When we have a natural planted distribution which is hard to distinguish from the random distribution, we can obtain the moment matrix M using the pseudo-calibration framework

pioneered by [BHK+16]. The main technical challenge is then showing that M is PSD with high probability. However, for k-coloring, we do not know of a natural planted distribution which is hard to distinguish from a random distribution using low-degree polynomials. To illustrate this challenge, we consider the planted distribution from the Stochastic Block Model. We fix the random graph to be in the dense-regime of $G(n, \frac{1}{2})$ as this is also where the gap of known low-degree polynomial hardness is the most drastic.

Definition 8.3.1 (Stochastic Block Model for k-coloring with Average-Degree $d = \frac{n}{2}$)**.** *We consider the following distribution of random graphs.*

1. *Sample a k-coloring by picking a color $\beta(v) \in [k]$ for each vertex $v \in [n]$ uniformly at random. Let $\beta(x)$ be the indicator vector for coloring of x;*

2. *Sample a random graph on n vertices by sampling edges for each pair of vertices i.i.d. as the following*

   - *For a pair of vertices with the same color, set the edge to be non-existent;*
   - *For a pair of vertices with different color, sample an edge with probability $\frac{k}{k-1} \cdot \frac{1}{2}$.*

3. *Let G be the graph obtained from the above process, and output $(G, \beta(x))$.*

Essentially equivalent to verifying $\widetilde{\mathbb{E}}[1] = 1 + o(1)$ with high probability for the construction from pseudo-calibration in SoS lower bounds, the low-degree polynomial likelihood ratio asks for verifying the following sum-of-sqaures of coefficients,

$$\sum_{\substack{R \subseteq \binom{n}{2} : |R| \leq D \\ R \neq \emptyset}} \left| \mathbb{E}_{planted}[\chi_R(G)] \right|^2 = o(1) \tag{8.1}$$

for $\chi_R(G) = \prod_{e \in R} \chi_e(G)$ and $\chi_e(G)$ is the Fourier basis for the null distribution. In our case, we can restrict ourselves to the standard $\{\pm 1\}$ basis for $G(n, \frac{1}{2})$, and subsequently

the p-biased basis in $G(n, \frac{d}{n})$. We now zoom in to the Fourier coefficients when we fix the planted distribution to the SBM model. For any fixed $R \subseteq \binom{n}{2}$,

$$\begin{aligned}\mathbb{E}_{SBM}[\chi_R] &= \mathop{\mathbf{Pr}}_{x \text{ uniform coloring}}[x_{S,\beta_S} = 1] \cdot \mathbb{E}_{G|x\sim sbm}[\chi_R(G)] \\ &= \sum_{\beta: V(R)\to[k]} \left(\frac{1}{k}\right)^{|V(R)|} \cdot \mathsf{Colorval}[R]_{\beta\beta}\end{aligned}$$

for Colorval defined as the following,

Definition 8.3.2 ($\mathsf{Colorval}[E]_\beta$)**.** *For an edge-set E, and β a coloring of its vertices in $[k]$, we define*

$$\mathsf{Colorval}[E]_\beta = \prod_{e\in E: \textit{pure-color under } \beta} (-1) \cdot \prod_{e\in E: \textit{cross-color under } \beta} (\frac{1}{k-1}) .$$

where we additionally call each edge pure-color if $\beta(i) = \beta(j)$, and cross-color otherwise. In words, the above function assigns a value of -1 for each edge between vertices of the same color and a value of $\frac{1}{k-1}$ for each edge between vertices of different colors.

We adopt the convention that for $E = \emptyset$, we have $\mathsf{Colorval}[\emptyset] = 1$.

At this point, we observe that color-subgraph (R, β_R) appears naturally as a basis in the above calculation. For convenience of our discussion, we call an (edge-induced) subgraph R with vertices labeled in $[n]$ a ribbon, and a vertex-labeled subgraph (R, β) with additional color-labeling $\beta : V(R) \to [k]$ a color-ribbon. Moreover, when the explicit vertex labeling in $[n]$ is abstracted out, we call any unlabeled subgraph a "shape", and define color-shape analogously. [4]

We are now ready to see the issue with the SBM distribution. Since each quantity is non-negative in the eq. (8.1), it suffices for us to focus on the contribution from ribbon R

[4]See formal definitions in section 8.3.4 that ultimately relate the subgraph to a matrix, while it suffices to consider its scalar analog as presented for the discussions below.

obtained by vertex-labelings of some specific shape of α.

$$LHS \text{ of eq. (8.1)} = \sum_{\alpha:\text{shape}} \sum_{R:V(\alpha)\to[n]} \left(\left(\frac{1}{k}\right)^{|V(\alpha)|} \cdot \underbrace{\left(\sum_{\beta:V(R)\to[k]} \mathsf{Colorval}[R]_\beta \right)}_{:=Ev(R)} \right)^2$$

Since the particular Ev defined through $\mathsf{Colorval}[E]_\beta$ does not depend on vertex-labels of the edge-set, we may simply write $Ev(\alpha) = Ev(R)$ for any ribbon of shape α. Moreover, for any shape α, there are $\Theta(n^{|V(\alpha)|})$ ribbons obtained by labeling its vertices, thus for the desired low-degree hardness to hold, one would require $\left(\frac{1}{k}\right)^{|V(\alpha)|} \cdot |Ev(\alpha)| = O(\frac{1}{k})^{|V(\alpha)|}$ so that one may set $k \approx \sqrt{n}$. However, it turns out that such bound is false as $|Ev(\alpha)| = \Theta(k)$ for a large family of subgraphs (e.g. any subgraph that does not contain a degree-1 vertex) . For concreteness, we can simply restrict our attention to α being a triangle, in which case we have

$$\begin{aligned} Ev(\text{triangle}) &= \sum_{\beta:V(R)\to[k]} \mathsf{Colorval}[\text{triangle}]_\beta \\ &= \sum_{\substack{\beta:V(\alpha)\to[k] \\ \text{all vertices have the same color}}} \mathsf{Colorval}[\text{triangle}]_\beta + \sum_{\substack{\beta:V(\alpha)\to[k] \\ \text{not all vertices have the same color}}} \mathsf{Colorval}[\text{triangle}]_\beta \\ &= (-k) + O(1) \end{aligned}$$

where we split the sum over all colorings into colorings that assign all three vertices to have the same-color (aka. pure-colorings), and other colorings, and notice the value (in magnitude) is dominated by the pure-coloring. More generally, triangles are not the only troublesome shapes, and the best bound one may hope for via this natural planted

distribution would be a lower bound for $k \approx \Theta(n^{\frac{3}{4}}) \gg \Theta(\sqrt{n})$. To bypass this particular barrier, [KVWX23] gives a planted distribution that matches the global statistic for triangles and obtains an improved bound of $k = \Theta(n^{\frac{2}{3}})$. They ask the question of finding a planted distribution approaching the threshold of $\Theta(\sqrt{n})$ as an explicit open question.

8.3.2 The Challenge from Global Constraints

The failure of a planted distribution is by no means unfamiliar for designing lower bounds for statistical inference problems. One natural idea to cope with such failure is simply to ask for a better "quiet" planting, and such a quest is an exciting challenge that warrants merit on its own. On the other hand, such a daring undertaking may appear "unnecessary" to us provided we are ultimately interested in the refutation problem as opposed to the distinguishing problem.

In contrast to distinguishing hardness where we need to satisfy extra polynomial constraints from local statistics [BMR19], the set of coloring-axioms are all the constraints we need to satisfy for the refutation problem. That said, we are allowed to depart from the set of moments from pseudo-calibration of some known planted distribution, and the shift back towards refutation hardness in fact opens up new avenues for us to circumvent the absence of a planted distribution. To the best of our knowledge, in the context of Sum-of-Squares lower bounds and more broadly average-case hardness for statistical inference problems, non-trivial deviation from pseudo-calibration has only been done in the context of the independent-set problem on sparse random graphs [JPR+22].

The Lure of Connected Truncation [JPR+22] shows that one may still salvage the doomed planted distribution for independent set by a non-standard truncation rule in the Fourier expansion, *the connected-truncation rule*. Recall that vanilla pseudo-calibration truncates edge-set solely based on cardinality, the connected-truncation rule asks for fur-

ther removal of edge-set that is not connected to S, and set

$$\widetilde{\mathbb{E}}[x_{S,\beta_S}] = \sum_{\substack{R \subseteq \binom{n}{2} : |R| \le D \\ R \text{ connected to } S}} \mathbb{E}_{planted}[x_{S,\beta_S} \cdot \chi_R(G)] \cdot \chi_R(G)\,.$$

On a high-level, one intuitive justification for such truncation rule is that there is no explicit polynomial constraint from a refutation lower bound for satisfying global statistics constraint, and therefore one may hope to simply remove them when designing the moments to start with.

In particular, our above example of a triangle for $\widetilde{\mathbb{E}}[1]$ is indeed an example of low-degree global statistic that gets removed by connected truncation! More generally, it can be observed that should one employ connected truncation for coloring, the bulk of the analysis does go through for the desired threshold of $\sqrt{n}$ (put simply, via a standard concentration argument, one can easily show $\widetilde{\mathbb{E}}[x_{S,\beta_S}] \ge 0$ for any (S, β_S) which is a scalar analog for matrix positivity). [5]

Global Constraints Strike Back However, for coloring, it turns out that we cannot quite close our eyes to these global statistics! In fact, the exact-sum constraint is implicitly imposing some global constraints that present inherent barriers for simple truncation modifications like connected-truncation.

To see this challenge, we observe that one natural way of satisfying the sum-color constraint is to use Fourier expansion and verify for any $S \subseteq [n]$, for any edge-set $R \subseteq \binom{n}{2}$, one has $\langle \chi_R(G), \sum_{\beta_S : S \to [k]} \widetilde{\mathbb{E}}[x_{S,\beta_S}] \rangle = \langle \chi_R(G), \widetilde{\mathbb{E}}[1] \rangle$ where $\langle \chi_R(G), \widetilde{\mathbb{E}}[x_{S,\beta_S}] \rangle$ denotes the Fourier inversion formula, i.e., the coefficient of $\chi_R(G)$ in the Fourier expansion of $\widetilde{\mathbb{E}}[x_{S,\beta_S}]$. However, this is plainly untrue with connected-truncation.

[5] We believe similar result to [KM21] can also be obtained by connected truncation+ direct PSD analysis, and this highlights that the essence of coloring (given known results for independent set) is in the exact-sum constraints.

For example, taking S as a single vertex v, and R some triangle containing v, say a triangle on $\{v, i, j\}$, it is clear that the LHS is exactly $Ev(\text{triangle}) \approx -k$ as we show earlier, while the RHS is 0 under connected-truncation.

This leads to the following question. If we use the coefficients from pseudo-calibration as a starting point for proving refutation hardness for k-coloring, what are the coefficients that we can modify and be creative with?

8.3.3 Rainbow Correction: an Ad-hoc Fix Made Systematic

Set-up for Our Correction Before we describe our modification, we first observe the following general form of pseudo-moment coefficient that we will be working with inspired by pseudo-calibration of the SBM distribution. The SBM distribution gives pseudo-calibrated coefficient as the following: for any edge-set R, let $\widetilde{\lambda}_{S,\beta(S)}(R) = \langle \widetilde{\mathbb{E}}[x_{S,\beta_S}], \chi_R \rangle$ denote the coefficient of χ_R in the Fourier expansion of $\widetilde{\mathbb{E}}[x_{S,\beta_S}]$,

$$\begin{aligned}\widetilde{\lambda}_{S,\beta(S)}(R) \coloneqq \mathbb{E}_{SBM}[x_{S,\beta_S} \cdot \chi_R(G)] &= \sum_{\substack{\beta: V(R)\to[k] \\ \beta(S)=\beta_S}} (\frac{1}{k})^{|V(R)\cup S|} \sqrt{\frac{p}{1-p}}^{|E(R)|} \mathsf{Colorval}[R]_\beta \\ &= \sum_{\substack{\beta: V(R)\to[k] \\ \beta(S)=\beta_S}} \widetilde{\lambda}_{S,\beta(S)}(R)_\beta\end{aligned}$$

for $\widetilde{\lambda}_{S,\beta(S)}(R)_\beta \coloneqq (\frac{1}{k})^{|V(R)\cup S|} \sqrt{\frac{p}{1-p}}^{|E(R)|} \mathsf{Colorval}[R]_\beta$.

We would like to think of the color-ribbon coefficient as having two components, a shape component from vertex/edge-factor of $(\frac{1}{k})^{|V(R)\cup S|} \cdot (\sqrt{\frac{p}{1-p}})^{|E(R)|}$ that does not depend on the coloring β, and an edge-value component $Ev(R)_\beta$ (motivated from $\mathsf{Colorval}[R]_\beta$). From this perspective, we will consider the following generic form for Fourier coefficients

in the pseudo-moment we are going to design,

$$\widetilde{\lambda}_{S,\beta_S}(R)_\beta \coloneqq (\frac{1}{k})^{|V(R)\cup S|}\sqrt{\frac{p}{1-p}}^{|E(R)|} \cdot Ev(R)_\beta$$

for some function $Ev(R)_\beta$ to be chosen. As a result, we will then set the final pseudo-moment (viewed as a function of graph sampe G) as

$$\widetilde{\mathbb{E}}[x_{S,\beta_S}](G) = \sum_{\substack{R\subseteq\binom{n}{2}\\ |V(R)\cup S|\le D_V}} \sum_{\substack{\widehat{\beta}:V(R)\to[k]\\ \beta(S)=\beta_S}} \widetilde{\lambda}_{S,\beta_S}(R)_\beta \cdot \chi_R(G)\,.$$

Remark 8.3.3. *We remind the reader that Ev stands for edge-value, as opposed to pseudo-expectation value.*

In the above definition, we write $Ev(R)_\beta$ with an explicit dependence on the coloring of β. For convenience, for any given $Ev(R)_\beta$ function, we also introduce the notation $Ev(R)$ without dependence of β denoting the sum of the $Ev(R)_\beta$ of all coloring β. This is formally defined as the following,

Definition 8.3.4 (Edge-value $Ev(\cdot)$ as a sum of all colors)**.** *For any ribbon (i.e. vertex-labeled subgraph) R, we define Ev function without explicit dependence on coloring β as the sum over all colorings, formally, we define*

$$Ev(R) \coloneqq \sum_{\beta:V(R)\to[k]} Ev(R)_\beta\,.$$

The definition extends to unlabeled subgraph "shape".

Notice in the case of fixing the edge-value function as $Ev(R)_\beta = \mathsf{Colorval}[R]_\beta$, we simply recover the set of coefficients from vanilla pseudo-calibration from the usual Stochastic

Block Model. However, as the reader may have noticed from our language in defining $Ev(R)_\beta$, this equality does not have to be fixed! And this is where we are going to depart from the vanilla value from pseudo-calibration.

To describe our modification, we make the following definitions for convenience that partition the collections of colorings for a given subgraph.

Definition 8.3.5 (Pure-color edge v. cross-color edge)**.** *For each edge with both endpoints receiving some color, we call it a pure-color edge if both endpoints have the same color, and cross-color/rainbow-color if otherwise. For short, we also call such edges pure edges and cross/rainbow edges.*

Definition 8.3.6 (Subgraph coloring β)**.** *For a subgraph R with edge-set, and β a coloring for $V(R)$, we call it a*

1. *Pure-coloring: there is no rainbow-color edge in R under β. Additionally define $Ev_{pure}(R) := \sum_{\substack{\beta:V(R)\to[k] \\ pure}} Ev(R)_\beta$;*
2. *Mixed-coloring: there is some edge that is pure-color and some edge that is rainbow-color in R under β. Additionally define $Ev_{mixed}(R) := \sum_{\substack{\beta:V(R)\to[k] \\ mixed}} Ev(R)_\beta$*
3. *Rainbow-coloring: there is no pure-color edge in R under β. Additionally define $Ev_{rainbow}(R) := \sum_{\substack{\beta:V(R)\to[k] \\ rainbow}} Ev(R)_\beta$*

In the case of R is a single connected component, we also call R a pure/mixed/rainbow component under β depending on the category of β.

This categorization prompts us to view

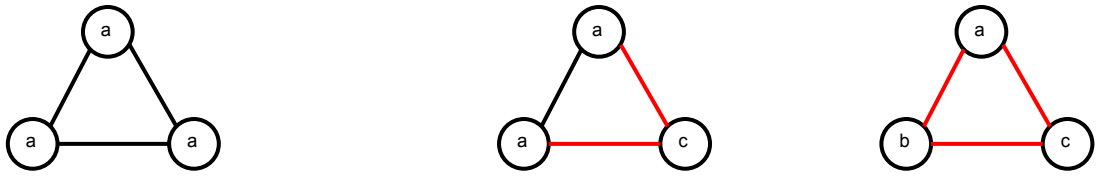
$$Ev(R) := \sum_{\beta:V(R)\to[k]} Ev(R)_\beta$$

as

$$Ev(R) = Ev_{pure}(R) + Ev_{mixed}(R) + Ev_{rainbow}(R)\,.$$

Remark 8.3.7. *As an edge-case of our definition, coloring for a single vertex without any edge is considered both a pure-coloring and a rainbow-coloring, and the corresponding vertex is considered both a pure-component and a rainbow-component.*

See the following for an example for coloring partitions on a triangle, where we use a black edge to represent a pure-color edge, and a red edge to represent a cross-color edge. For this section, let a, b, c be colors of the vertices while we ignore the vertex-label in $[n]$ as they are not the focus of our discussion here (for simplicity, consider them as being fixed).



(a) Pure-coloring (b) Mixed-coloring (c) Rainbow-coloring

Figure 8.1: Example of Pure/Mixed/Rainbow-Coloring

Triangle as a Starting Point To further motivate our correction, let us revisit the example of triangle. Recall that our ideal planted distribution is simply one that gives

$$|Ev(R)| = \left| \sum_{\beta: V(R) \to [k]} Ev(R)_\beta \right| = 0\,,$$

recovering connected truncation de facto, while per the prior discussion, for R being a triangle, the construction from SBM gives

$$|Ev_{sbm}(R)| = \left|Ev_{pure}(R) + Ev_{mixed}(R) + Ev_{rainbow}(R)\right| = |-k + O(1) + O(1)| = \Theta(k)\,.$$

where we write $Ev_{sbm}(R)$ to denote the $Ev(R)_\beta$ function of taking $\mathsf{Colorval}[R]_\beta$ for any β.

With this set-up, instead of trying to find an ideal planted distribution, we simply want to modify the coefficient (through $Ev(R)_\beta$) so that we ultimately have $Ev(R) = 0$. But how can we do so? On the one hand, we observe that we have essentially no freedom in picking coefficients for pure-coloring as they are constrained by the independent-set constraints, and therefore we leave the edge-value for pure-coloring touched, i.e. $Ev_\beta(R) = \mathsf{Colorval}[R]_\beta$.

On the other hand, there is no more independent-set constraint on any rainbow-coloring! If we modify its coefficient such that

$$Ev_{rainbow}(R) = (-1) \cdot (Ev_{pure}(R) + Ev_{mixed}(R)) .$$

we would then have our desired $Ev(R)$ bound for triangle!

To illustrate this, for each rainbow-ribbon (R, β) of a triangle, we may simply modify its coefficient as

$$Ev(R)_\beta \coloneqq \mathsf{Colorval}[R]_\beta + \frac{D_C}{|RS_R|}$$

where the second term can be viewed as a correction term with

$$D_C = (-1) \cdot \left(Ev_{pure}(R) + Ev_{mixed}(R) + \sum_{\beta:\text{rainbow}} \mathsf{Colorval}[R]_\beta \right)$$

in the case of triangle, and $|RS_R|$ (rainbow-size of R) is simply the set of rainbow-colorings for R to ensure that

$$\sum_{\beta:\text{rainbow}} Ev(R) = D_C + \sum_{\beta:\text{rainbow}} \mathsf{Colorval}[R]_\beta = -\left(Ev_{pure}(R) + Ev_{mixed}(R)\right) .$$

In other words, we would like to pick up an extra D_C term from the total contribution of rainbow-colorings, and we achieve so by evenly distributing the D_C term among all

rainbow-colorings.

We defer the modification for mixed-coloring, while note that in the example of triangle, we may just leave it unmodified from the vanilla SBM value for illustration purpose, i.e. $Ev(R)_\beta = \mathsf{Colorval}[R]_\beta$ for any mixed-coloring β. With this new modified coefficient for rainbow-colorings, we claim that R being a triangle is no longer an issue for the scalar calculation of $\widetilde{\mathbb{E}}[1]$ (aka. SoS analog for low-degree likelihood ratio) as it would have

$$|Ev(R)| = |Ev_{pure}(R) + Ev_{mixed}(R) + Ev_{rainbow}(R)| = 0\,.$$

To further motivate our correction scheme, consider the subgraph/shape being a line or more generally a tree or a forest. In these case, we have $Ev(\cdot) = 0$ holds straightforwardly by vanilla pseudo-calibration. From this perspective, our correction can be seen as modifying the rainbow-value so that every shape looks tree-like as a global statistic when we sum over all colors of its vertices.

Note that the example of triangle is in some sense an edge-case as the value of mixed-coloring has a small magnitude. However, this is not necessarily true for general cases, and extra care is warranted for designing coefficients for mixed-colorings.

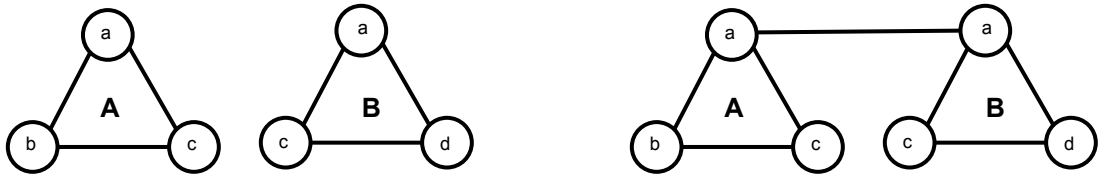
Factorizability over Connected Components, and Mixed Coloring In the above discussion, we show that there is extra freedom in designing the coefficient for rainbow ribbon while the coefficient for pure ribbon is fixed by the independent-set constraints. However, one question we did not answer is what happens for the in-between colorings, the mixed-colorings.

To answer this question, we need to first describe an ideal property that we would like of modified coefficient to satisfy, that it should factorize over connected components. Formally, for a subgraph α that may contain multiple connected components $C_1, \ldots, C_t$,

we would like our modified coefficient to satisfy

$$Ev(\alpha)_\beta = \prod_{i:C_i} Ev(C_i)_{\beta(C_i)} .$$

That said, we now observe the factorizability over connected components alongside independent-set constraint dictates that mixed-colorings should be carefully treated similarly as pure-colorings as they may now also participate in independent-set constraints. We now illustrate this with the example of 2 disjoint triangles.



(a) α: 2 Disjoint Triangles A and B

(b) α': Connected Triangles

Let β be the given vertex-coloring for $\alpha = A \cup B$, and $\beta(A)$ the coloring for A (and analogously for B). Following our rule of factorization over connected components, we define

$$Ev(\alpha)_\beta := Ev(A)_{\beta(A)} \cdot Ev(B)_{\beta(B)} .$$

Note that $\beta(A), \beta(B)$ restricted to each single component is a rainbow-color, and each shape is a triangle and hence each can be defined via the previous rule. That said, $Ev(\alpha)_\beta$ is also well-defined by taking the product of the above. However, consider the diagram α' which contains an added edge on the top between two vertices of the same color a. This is no longer a single connected component, and β is now a mixed-coloring for α'. However, applying the independent-set constraint on color a, we can deduce that the sum of the following two terms gives a scalar product of independent-set indicator on vertices with

color-a, i.e.

$$Ev(\alpha')_\beta + Ev(\alpha) = 0\,.$$

That said, the coefficient of mixed-value coloring is not completely unconstrained like rainbow-colors due to the internal independent-set constraints on the pure-color edges. With these constraints, its value ultimately depends on the underlying rainbow component after the removal of pure-color edges (in the above example, α' is fixed by α). This observation culminates in a recursive process for defining color-ribbon coefficients.

For a mixed-value ribbon (R, β) for R a single connected component, consider the subgraph obtained by removing pure-color edge in R under β denoted by $\mathcal{G}(R, \beta)$, and we observe that under such removal, $\mathcal{G}(R, \beta)$ is potentially a collection of rainbow components under β, and let $\{C_i\}$ denote the collection of edge-components. We then define the final value as

$$Ev(R)_\beta := (-1)^{|pure(R)_\beta|} \cdot \prod_{C_i \text{connected comp. in } \mathcal{G}(R,\beta)} Ev(C_i)_{\beta(C_i)}\,,$$

where we use $pure(R)_\beta$ to denote the set of pure-edge in R under β, and note that $(-1)^{|pure(R)_\beta|}$ is the factor given by independent-set constraints on pure-color edges.

We formally describe our recursive scheme for defining the Ev coefficients in section 8.4 and verify the hard-constraints in section 8.4.4.

8.3.4 Approximate Matrix Factorization via Rainbow Separator

With the coefficient value defined from the above discussion, we are ready to discuss the challenges from the PSD analysis. At the first sight, one may anticipate[6] that such analysis should largely follow from the known independent set analysis except with some obvious translation (for example, consider a larger moment matrix that is additionally indexed by colors). That said, it is true in the sense that our analysis would give a re-proof for

[6] as we suspected so ourselves in the beginning

independent set lower bound, there are several obvious issues should one tries to carry out the apply the prior work. In particular, a major conceptual issue arises from our departure from pseudo-calibration, which is arguably inevitable in the absence of a (known) quiet planting.[7]

We try to give a lightweight introduction to the approximate matrix factorization machinery here, while we still would like to refer the reader to prior works for a more thorough treatment of the subject (in particular, see [AMP20, KPX24] for introduction of graph matrices, and [BHK+16, JPR+22, JPRX23] for approximate matrix factorization). To this end, we introduce "shape" that is at the heart of SoS lower bounds as it forms the basis for a moment matrix.

Preliminaries for Matrix Factorization

Definition 8.3.8 (Fourier character for $G_{n,d/n}$)**.** *Let χ denote the $p = \frac{d}{n}$-biased Fourier character,*

$$\chi_e(1) = \sqrt{\frac{1-p}{p}}, \quad \chi_e(0) = -\sqrt{\frac{p}{1-p}}$$

For a graph G, and a subset of edges $H \subseteq \binom{n}{2}$, we write $\chi_H(G) = \prod_{e \in H} \chi_e(e \in G)$.

For the dense regime of $G(n, \frac{1}{2})$, the above simplifies to $\chi_e = 1$ in the case of edge e is present in G, and $\chi_e = -1$ if not.

Definition 8.3.9 (Shape α)**.** *Shape is a (multi)-graph $(V(\alpha), E(\alpha))$ additionally associated with two specially labeled set of vertices (left-boundary) U_α and (right-boundary) V_α with $U_\alpha, V_\alpha \subseteq V(\alpha)$. We also additionally define the operation shape transpose such that α^T denotes the shape obtained by flipping the left/right boundary of α.*

Additionally, we call each graph with vertices labeled in $[n]$ a ribbon.

[7] Such challenge is also likely inevitable for a quiet planted distribution that does not have an immediate product structure.

Definition 8.3.10 (Graph matrix for a ribbon)**.** *For a ribbon R, the matrix M_R has rows and columns indexed by all order-tuples of $[n]$ of size at most $\frac{\mathsf{dsos}}{2}$ and has a single nonzero entry,*

$$M_R[I,J] = \begin{cases} \chi_{E(R)}(G) & I = U_R, J = V_R \\ 0 & \textit{Otherwise} \end{cases}$$

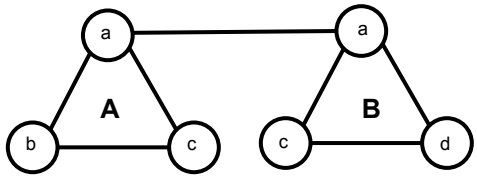
Definition 8.3.11 (Graph matrix for a shape α)**.** *For any graph G, and any given shape α, the graph matrix M_α is a matrix with rows and columns indexed by all order-tuples in $[n]$ of size at most $\frac{\mathsf{dsos}}{2}$ defined as the sum of all ribbons of the prescribed shape, formally,*

$$M_\alpha := \sum_{R:\textit{ribbon of shape } \alpha} M_R$$

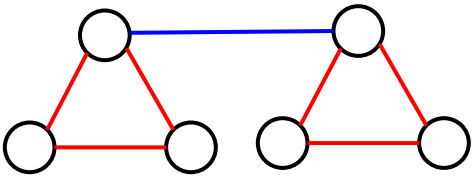
Definition 8.3.12 (Pure-color edge v. cross-color edge)**.** *For a subgraph R, and $\beta : V(R) \to [k]$ a coloring of its vertices, we call each edge in R a pure-color edge if both endpoints have the same color, and cross-color/rainbow-color if otherwise. For short, we also call each such edge pure-edge and cross/rainbow edge.*

Definition 8.3.13 (Ribbon/Shape coloring β and color-partition CP)**.** *For a shape α, we will consider $\beta : V(\alpha) \to [k]$ a coloring. For our application, we further group the collection of colorings based on each edge being pure-color/cross-color, and towards this end we define CP by partitioning the collection of colorings based on each edge of $E(\alpha)$ being pure-color or cross-color (i.e. rainbow-color).*

Example 8.3.14 (Example of Color-Partition)**.** *Take the following shape for example. Recall that a, b, c, d are color-labels in $[k]$. Let red edge denote the constraint that the edge must be rainbow-color for coloring β in the prescribed color-partition CP, and blue edge a pure-color edge. The coloring β on the left can be seen as an element in the collection prescribed by CP that constrains each triangle edge to be rainbow-color.*



(a) A Coloring β



(b) A Color-Partition CP such that $\beta \in \text{CP}$

For the bulk of analysis, we will use coloring β and color-partition CP interchangeably

.

Definition 8.3.15 (Color-shape (α, CP))**.** *We call (α, CP) for a shape α and its color-partition CP a color-shape. Additionally, for a coloring $\beta \in CP$ that assigns each vertex's color in $[k]$, and R a graph obtained by labeling $V(\alpha)$ in $[n]$, we call (R, β) a color-ribbon.*

Remark 8.3.16. *When it is clear that we are referring to a color-shape, we usually make the dependence of CP implicit, and simply refer to it via α.*

Definition 8.3.17 (Graph Matrix for a Color-Ribbon)**.** *For a color-ribbon (R, β), the matrix $\mathsf{M}_{R,\beta}$ has rows and columns indexed by all oreder-tuples of $[n] \times [k]$ of size at most $\frac{\mathsf{dsos}}{2}$ and has a single nonzero entry,*

$$M_{R,\beta}[(I, \beta_I), (J, \beta_J)] = \begin{cases} \chi_{E(R)}(G) & (I, \beta_I) = (U_R, \beta(U_R)), (J, \beta_J) = (V_R, \beta(V_R)) \\ 0 & \textit{Otherwise} \end{cases}$$

Definition 8.3.18 (Graph matrix for Color-Shape)**.** *Given a color-shape (α, CP), we define its corresponding graph $\mathsf{M}_{(\alpha,CP)}$ as a matrix with rows and columns indexed by any order-tuple of $[n] \times [k]$ of size at most $\frac{\mathsf{dsos}}{2}$, and define it as the sum of graph matrix of color-ribbon of shape α*

and coloring in CP, formally

$$\mathsf{M}_{\alpha,CP} := \sum_{R:\textit{ribbon of shape } \alpha} \sum_{\beta:V(R)\to[k]} \mathsf{M}_{R,\beta}\,.$$

Recap of the Prior Strategy Throughout this work, we follow the convention of referring to a color-indexed graph matrix via M. With the setup above, we are ready to recap the prior strategy see the challenge in matrix factorization. For the moment, let us return to the independent-set setting and simply consider shape without any coloring. The main idea of approximate factorization is to observe the candidate moment may be rewritten in the basis of graph matrix as

$$\Lambda = \sum_{\substack{\alpha:\text{shape} \\ \text{subject to truncation constraints}}} \widetilde{\lambda}(\alpha)\cdot M_\alpha = LQL^T$$

for some choice of L, and action boils down towards showing Q is PSD. Put simply for a single shape, this roughly says that for any shape α, we can decompose it into three pieces $\tau = \sigma \circ \tau \circ \sigma'$ such that

$$\widetilde{\lambda}(\alpha)\cdot M_\alpha \approx \left(\widetilde{\lambda}(\sigma)\cdot M_\sigma\right)\cdot\left(\widetilde{\lambda}(\tau)\cdot M_\tau\right)\cdot\left(\widetilde{\lambda}(\sigma')\cdot M_{\sigma'}\right)$$

where the approximate equality ignore terms that arise in the approximation of $M_\alpha \approx M_\sigma \cdot M_\tau \cdot M_{\sigma'}$, i.e., "error" terms due to vertex intersections beyond the boundary. Moreover, crucial to our decomposition scheme is the notion of separator and minimum-vertex-separator in the context of independent-set defined as the following,

Definition 8.3.19 (Separator and Minimum-Vertex-Separator)**.** *Given a graph α with two labeled set of vertices U, V, a set $S \subseteq V(\alpha)$ is a separator for α if any path between U and V passes through some vertex in S. Additionally, S is a* Minimum Vertex Separator *if $|S| \le |T|$ for any*

T that is a separator for α.

See the following diagram of a decomposition based on Minimum Vertex Separators for an example where $\sigma = L$ is the left-part, $\tau = M$ is the middle-part, and $\sigma' = R$ is the right-part. Intuitively speaking, one may view the left/right shape as projection matrix to the subspace a shape lives in. After zooming into the "correct" eigenspace, the crux of the argument amounts to verifying

$$\widetilde{\lambda}(\tau) \cdot \|M_\tau\| = o(1)$$

where we use $\|M\|$ to denote matrix's spectral norm of matrix M throughout the work. The above can be viewed as a matrix analog of the low-degree hardness (scalar calculation of $\widetilde{\mathbb{E}}[1]$) in which the key component is to verify for each shape α has small deviation $\widetilde{\lambda}(\alpha) \cdot |\sum_{R:V(\alpha)\to[n]} \chi_R(G)| \leq o(1)$.

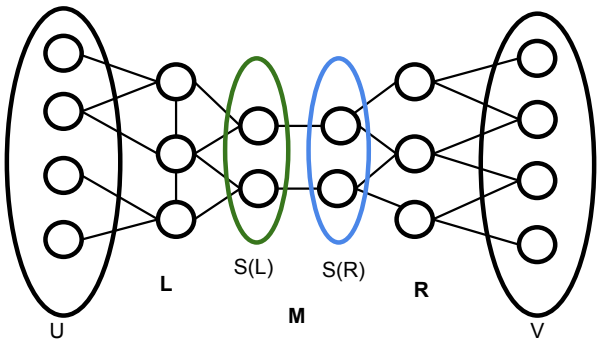


Figure 8.4: Three-way Decomposition of a Shape

Failure of MVS-Decomposition With the prior strategy recapped, it is not too hard to verify that for any shape τ arising from the decomposition, i.e. after applying MVS decomposition, the natural matrix analog for coloring holds and one has

$$\widetilde{\lambda}(\tau) \cdot \|\mathsf{M}_{\tau,\mathrm{CP}}\| = o(1)\,.$$

However, there belies a detrimental issue in the above strategy from applying the MVS-decomposition. The keystone of the above idea is that in the decomposition of $\alpha = \sigma \circ \tau \circ \sigma'$, the coefficient of α factorizes exactly according to the leftmost/rightmost MVS $S(L)$ and $S(R)$. This condition has held among all known higher-degree SoS lower bounds as the planted distribution usually comes with an immediate product structure so that the coefficient factorizes over vertices and edges. Unfortunately, this is not true in our setting due to our rainbow correction.

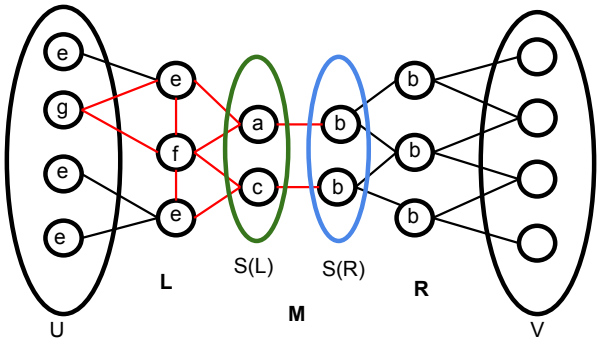


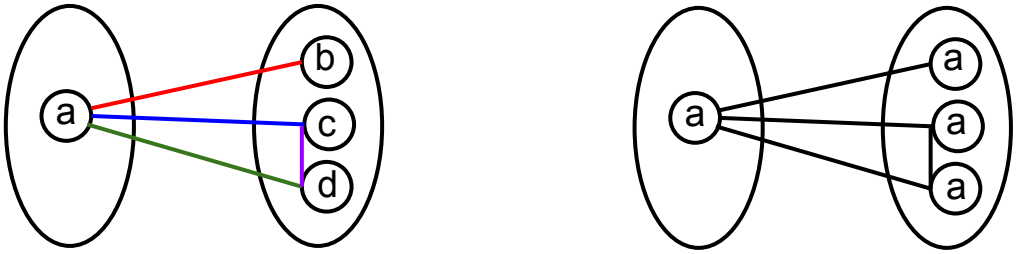
Figure 8.5: Issue with MVS

Consider again the above shape but this time vertices additionally comes with colors. The red edges in the diagram form a rainbow component (as there is no pure-edge in the component despite some vertices have the same color), and by our coefficient-scheme, it receives Ev coefficient as a whole-component, and it seems extremely unlikely whether one may factorize its value according to vertices in $S(L)$ and $S(R)$. From another perspective, this issue is consistent with the folklore understanding that approximate factorization scheme applies well only when the underlying coefficient factorizes over vertices and edges.

Can we avoid factorization? With the challenge of factorizability, it is tempting for one to ask whether we can avoid factorization at all: using prior experience in independent-set, one may hope that factorization is only needed in the case when every edge is pure-color,

which is also when the coefficient factorizes well. And whenever correction is involved (eg. via rainbow-color), one may try to use a trivial factorization, i.e., use the trivial separator of U and V and write the shape α as $\alpha = Id \circ \alpha \circ Id$.

Guided by this intuition, it is encouraging to verify that we can indeed ignore MVS decomposition if the component is a rainbow component since the Ev coefficient essentially offers an extra decay of $\frac{k}{k^{|V(\tau)|}}$ assuming τ is a single rainbow-component. Even though some extra factor in k may be incurred (comparing with the pure-edge case) as we need to additionally give vertex-label in $[k]$, there is in the end some extra slack when combined with the coefficient that ultimately allows one to bypass the MVS decomposition for rainbow-component.



(a) Rainbow-color Shape with norm $\tilde{O}(\sqrt{n}^3\sqrt{k}^4)$ (b) Pure-color Shape with norm $\tilde{O}(\sqrt{n}^3)$

Figure 8.6: Norm for Color Shape

We showcase this by the above color-shape with all 4 vertices having different color. By our coefficient scheme, it receives a coefficient of $(\frac{1}{k})^{4-\frac{4}{2}} \cdot Ev_{rainbow}(\alpha) \approx (\frac{1}{k})^{4-\frac{4}{2}} \cdot \frac{k}{k^4} = \frac{1}{k^5}$ where we point out that $(\frac{1}{k})^{4-\frac{4}{2}}$ is the scaling such that diagonal is 1, and our focus is the extra decay from Ev. Note that such color-shape has norm $\tilde{O}(\sqrt{n}^3 \cdot \sqrt{k}^4)$ and therefore we have the desired charging of $\frac{1}{k^5} \cdot \tilde{O}(\sqrt{n}^3 \cdot \sqrt{k}^4) \leq 1$ provided $k \geq \tilde{\Omega}(\sqrt{n})$.

To give a comparison, in the case of all 4 vertices having the same color $a \in [k]$, i.e., we are back at the previous independent-set setting, the coefficient is $(\frac{1}{k})^{4-\frac{4}{2}}$ and the norm is $\tilde{O}(\sqrt{n}^3)$. Therefore, we can no longer set $k \approx \sqrt{n}$ for this shape, and we need to appeal to

matrix factorization to handle this shape.

However, this is not quite the end of the story, as the above argument skips upon the in-between regime, the mixed-coloring. Unfortunately, for this collection of color-shapes, the analog of the above inequality falls apart and further matrix decomposition is inevitable within our framework.

A Smooth Interpolation via Rainbow Separator Prompted by the above example, we want to find a smooth interpolation in the decomposition-scheme between pure-color for which MVS-decomposition is needed, and rainbow-color for which the trivial decomposition is sufficient. Towards this end, we propose a new decomposition criterion based on (leftmost/rightmost) *Rainbow Separator*, and show that the product structure can be relaxed to some extent. To motivate this decomposition, consider again the above example, and note that MVS decomposition in fact applies well for the rightmost separator, as $S(R)$ is not incident to any rainbow-color edge to its right, and the coefficient factorizes for the right-component. However, $S(L)$ is troublesome as it is incident to a rainbow edge to its left. That said, we may "carefully" move $S(L)$ to the left such that it is not incident to any more rainbow edge to its left. This lands us at the following updated $S(L)$ given by the green vertices.

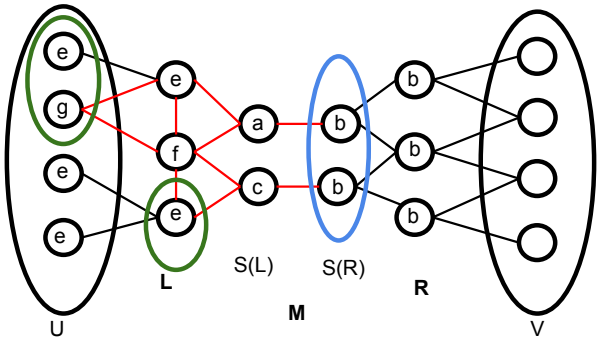


Figure 8.7: Decomposition via Rainbow Separator

For readers familiar with the recent works in Sum-of-Squares lower bounds, the notion

of leftmost/rightmost rainbow separator is reminiscent of the Positive MVS (PMVS) in the context of Densest-k-Subgraph problem that it does not admit a simple one-shot definition like MVS does. Instead, the definition of leftmost/rightmost rainbow separator is formalized via an iterative process in section 8.5.2 that starts from the original leftmost/rightmost MVS, and keeps moving left/rightwards until the candidate set satisfies certain condition: in our case, the set being the unique MVS to its left/right (for uniqueness of the decomposition), and moreover, not incident to any rainbow-edge to its left/right (for factorizability of coefficient).

Remark 8.3.20. *In the case of every vertex receives the same color, the above process does not need to move to left/right, and recovers the decomposition based on left/rightmost MVS.*

8.3.5 Recursive Factorization for the Kernel

Finally, there is the well-anticipated challenge we need to address, the sum-color constraint, and the kernel it incurs. Before we get started, it is useful to see how the kernel arises. Roughly speaking there are two different angles to seeing this. On one hand, one natural and intuitive way is to notice that the polynomial constraint $\sum_{c\in[k]} \widetilde{\mathbb{E}}[x_{i,c}] - \widetilde{\mathbb{E}}[1] = 0$ simply corresponds to a kernel for the moment matrix via the natural vector representation of the above polynomial. On the other hand, an arguably more twisted way is to notice that as we apply PSD analysis on $\Lambda = LQL^T$, one typically wishes to show $\|Q - I\| = o(1)$ and conclude PSDness of Q then via a triangle inequality.

However, such argument is bound to fail due to the kernel. Interestingly, the failure is manifested through intersection terms, in particular, via trivial intersection terms τ on the diagonal $U_\tau = V_\tau = V(\tau)$ and $E(\tau) = \emptyset$. To elaborate on this, we make the following definitions,

Definition 8.3.21 (Kernel Partition with Partial Coloring)**.** *Given a set $B \subseteq [n]$, we consider*

a kernel partition $S(B) \cup T(B)$ as the following,

1. *There is a partial coloring for B such that vertices in $S(B)$ receive color under $\beta_{S(B)}$;*
2. *Vertices in $T(B)$ are uncolored, and we sum over all colors of vertices in $T(B)$.*

This is naturally motivated by the kernel constraint from

$$\widetilde{\mathbb{E}}\left[x_{S(B),\beta_{S(B)}} \cdot \left(\sum_{\beta_{T(B)}} x_{T,\beta_{T(B)}}\right)\right] - \widetilde{\mathbb{E}}[x_{S(B),\beta_{S(B)}}] = 0$$

Definition 8.3.22 (Kernel Partition)**.** *Given a set $B \subseteq [n]$, we call $S(B) \cup T(B)$ a kernel partition for B if vertices in $S(B)$ receives some fixed color, and $T(B)$ are the vertices whose colors are being summed over. In other words, a kernel partition is a kernel-partition with partial-coloring except with $\beta_{S(B)}$ the fixed colors on $S(B)$ unspecified.*

Definition 8.3.23 (Kernel matrix $K_{S\cup T}$ for Kernel-Partition)**.** *For any kernel partition $S \cup T \subseteq [n]$ such that S (possibly empty) are the vertices with color fixed to some β_S, and $T \neq \emptyset$ the vertices without color. We define*

$$K_{S\cup T} := \otimes_{i\in S\cup T} K_i$$

where $K_i = Id_k$ is a $k \times k$ identity matrix for $i \in S$ and $K_i = -\frac{J_k}{k}$ for $i \in T$ is the the all-1 matrix of dimension $k \times k$ scaled by a factor of $-\frac{1}{k}$,

In other words, this is a coloring matrix of dimension $([n] \times [k])^{|S\cup T|} \times ([n] \times [k])^{|S\cup T|}$, and has non-zero entry at

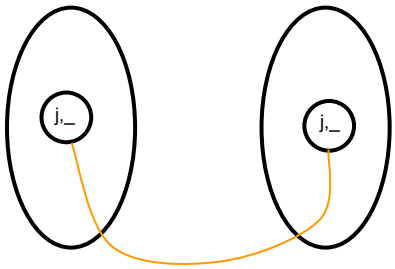
$$K_{S\cup T}[(S \cup T, \beta_S \cup \beta_T), (S \cup T, \beta_S \cup \beta'_T)] = (-\frac{1}{k})^{|T|}.$$

for any $\beta_S : S \to [k]$ and $\beta_T, \beta'_T : T \to [k]$. .

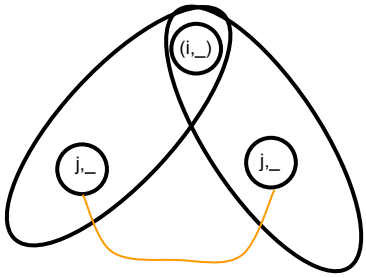
Notice crucially that for $K_{S\cup T}$ matrix, each non-zero entry must have consistent colors

on S in both row and columns, while the row-coloring and column-coloring for T do not necessarily coincide.

Example 8.3.24. *Consider vertices $i \neq j \in [n]$ and color $a \in [k]$, For set $B = \{(i,a),(j,\_)\}$ corresponds the kernel partition with $S(B) = \{(i,a)\}$ and $T(B) = j$. The corresponding matrix $K_{S\cup T}$ is a coloring matrix with $(-\frac{J}{k})$ on the block restricted to columns and rows of the form $\{(i,a),(j,\_)\}$ (viewed as sets). Summing over color of i gives us a coloring with non-zero entries given by $Id_k \otimes (-\frac{J_k}{k})$. on the block of indices $\{(i,\_),(j,\_)\}$, i.e., blocks restricted to coloring for vertex-set (i,j).*



(a) $S = \emptyset, T = \{i\}$
$-\frac{J_k}{k}$ at Block for Coloring of vertex j



(b) $S = \{(i,\_)\}, T = \{j\}$
$Id_k \otimes \frac{-J_k}{k}$ at block for Coloring of vertices (i,j).

Figure 8.8: Kernel Partition from Intersection Shape

For intuition, consider the above intersection shape in section 8.3.5, and observe their corresponding graph matrix in Q is exactly given by $-\frac{J}{k}$ and $Id \otimes \frac{-J_k}{k}$ when restricted to the block of coloring for the corresponding vertices (where the minus sign comes from such shapes being intersection terms).

More generally, consider a set of vertices $U \subseteq [n]$. It turns out that we can consider any $S \cup T$ partition of U, and observe that the corresponding $K_{S(U)\cup T(U)}$ matrix is part of Q: ultimately, for the block restricted to colorings of U, we have

$$Id[U] + \sum_{S(U)\cup T(U)} K_{S(U)\cup T(U)}$$

where with some abuse of notation we let $Id[U]$ as the coloring matrix restricted to colors on vertices of U, and this is exactly why the magnitude bound argument fails. As each $K_{S(U)\cup T(U)}$ has norm exactly 1 (and not necessarily PSD), and therefore extra care is need to take care of this component.

For ease of notation, we define the following matrix for a global analog of $K_{S(U)\cup T(U)}$ by summing over all sets of size at most dsos and all kernel-partition.

Definition 8.3.25 ("Trivial" Kernel Matrix K_{trivial})**.** *We define the candidate kernel matrix as*

$$K_{\mathsf{trivial}} = \sum_{\substack{W\subseteq[n]\\ |W|\leq \mathsf{dsos}}} \sum_{\substack{S(W)\cup T(W)\\ \textit{kernel-partition}}} K_{S(W)\cup T(W)}$$

To handle the kernel, we use both of the above views interchangeably. On a high-level, we employ the kernel-trick introduced in [GJJ+20b] recapitulated as the following,

Claim 8.3.26 (Kernel-trick)**.** *For any symmetric matrix M with kernel $Ker(M)$, M is PSD if*

$$\widetilde{M} = M + \sum_{u_i\in Ker(M)} v_i u_i^T + u_i v_i^T \succeq 0\,.$$

This prompts to generate $-K_{trivial}$ via vectors in the kernel of Q, and fill in the hole incurred by K_{trivial} on the diagonal blocks of Q. As a result of the filling, we would then have a genuine Id on the diagonal so that we can continue to apply a magnitude-bound based argument for PSDness.

Our work amounts to identifying the right choice of $u_i \in Ker(Q)$, however, this turns out to be surprisingly a challenging task as applying L^T on the natural kernel-vector additionally induces large-norm components that warrant extra care. On a high-level, this can be seen as undoing the eigenspace zooming operation of L. Towards this end, analogous to the recursive factorization scheme via LQL^T, we employ a similar

recursive factorization scheme to identify the kernel-vector such that the resulting matrix is amenable to a magnitude-bound analysis for PSDness. The decomposition scheme is the focus of section 8.8.

8.3.6 Final Roadmap for PSD Analysis

To wrap up our overview section, we give a roadmap for PSD analysis before we enter the combinatorial charging analysis, and describe our key technical lemmas that would allow us to complete the proof of PSDness analysis. We formally define our moment matrix in definition 8.4.5. Then, we apply the new decomposition machinery based on rainbow separator and obtain a new set of (color)-left shapes encapsulated in the L matrix, plugging this into the by-now standard factorization machinery, we have

$$\Lambda = L\widetilde{Q}L^T + M_{\mathsf{truncation}}$$

where $M_{\mathsf{truncation}}$ is the error term that arises from the three-way product in $L\widetilde{Q}L^T$ yet beyond our chosen truncation limit due to size-bound, and some Q recursively constructed with the new L matrix of color-left-shape. The recursive approximation process of obtaining $\widetilde{Q}$ is formalized in section 8.5.2. [8]

However, it turns out the above decomposition is not quite sufficient for us. To handle the kernel coming from k-wise symmetry, it is important for us to apply approximate factorization on the truncation term as well, and ultimately write

$$\Lambda = LQL^T$$

for $Q = \widetilde{Q} + Q_{\mathsf{trunc}}$ for some Q_{trunc} with negligible norm.

[8] Note that we use the symbol $\widetilde{Q}$ as opposed to Q at this stage as we intend to reserve Q for the final middle matrix before putting in the kernel-filling.

We now focus on the PSDness of $\widetilde{Q}$. Unlike previous setting where we may apply a magnitude bound directly on Q_i and charge it to the identity contained in Q_0, we have large intersection term due to our kernel that does not offer us any slack in the charging argument. This are the terms corresponding to the trivial shape in Q, and formally identified in lemma 8.8.1. Group out these terms, and write

$$Q = Id + K_{\mathsf{trivial}} + Q_{\mathsf{nontrivial}} + Q_{\mathsf{trunc}}$$

for some $Q_{\mathsf{nontrivial}} \coloneqq Q - (Id + K_{\mathsf{trivial}})$ be the collection of all non-trivial shapes defined as the following,

Definition 8.3.27 (Trivial Shape and Nontrivial Shape)**.** *We call a color-shape (τ, CP) trivial if $U_\tau = V_\tau = V(\tau)$ and $pure(E(\tau))_{CP} = \emptyset$. Any shape that is not a trivial shape is a non-trivial shape.*

We then apply a magnitude bound argument on $Q_{\mathsf{nontrivial}}$ to show $\|Q_{\mathsf{nontrivial}}\| = o(1)$. This is shown in section 8.6 and section 8.7.

Next, we apply the kernel trick to handle the damaged identity on the diagonal, and construct a kernel matrix K_Q via the recursive process in section 8.8 to approximates K_{trivial}, i.e., $\|K_{\mathsf{trivial}} - K_Q\| = o(1)$ and $Q \cdot K_Q = 0$. The PSDness of Q is then translated to that of $Q - K_Q$.

Last but not least, identical to the prior works on independent-set [BHK+16, JPR+22, KPX24], the independent-set indicator can be readily grouped out in $\widetilde{Q} + K_Q$, and give us

$$Q + K_Q = \Pi_{indset}^{1/2}(Id + Q_{\mathsf{nontrivial}} + K_{\mathsf{nontrivial}} + Q_{\mathsf{trunc}})(\Pi_{indset}^{1/2})^T$$

for $K_{\mathsf{nontrivial}} \coloneqq K_{\mathsf{trivial}} - K_Q$ and Π_{indset} is the diagonal matrix such that

$$\Pi_{indset}[(S,\beta_S),(S,\beta_S)] = p(S,\beta_S) \cdot \prod_{c\in[k]} \mathbf{1}[\beta_S(c) \text{ is an Independent Set in } G]$$

where $\beta_S(c)$ denotes the set of vertices in S with color c, and some scalar $p(S,\beta_S) \geq 0$. Finally we conclude the PSDness by applying magnitude bound on $Q_{\mathsf{non-trivial}}$ and $K_{\mathsf{nontrivial}}$ respectively.

We wrap up this section by completing the proof to main theorem.

Proof to Main Theorem. As discussed above, it suffices for us to show $Q + K_Q \succeq 0$. Since Π_{indset} is a diagonal matrix with non-negative entries, we have $\Pi_{indset} \succeq 0$, and it suffices to show

$$Id + Q_{\mathsf{nontrivial}} + K_{\mathsf{nontrivial}} + Q_{\mathsf{trunc}} \succeq 0$$

This then follows by the slack function bounds we establish through the combinatorial analysis in section 8.6 and section 8.7 and section 8.9, we show that they altogether imply with probability at least $1 - O(\frac{1}{n})$,

$$\|Q_{\mathsf{nontrivial}} + K_{\mathsf{nontrivial}}\| = o(1)$$

Additionally, $\|Q_{\mathsf{trunc}}\| = o(1)$ by our bound on truncation term and well-conditioness bound for L via lemma 8.9.28. Finally, we can conclude

$$\begin{aligned} Q + K_Q &= \Pi_{indset}^{1/2}(Id + Q_{\mathsf{nontrivial}} + K_{\mathsf{nontrivial}} + Q_{\mathsf{trunc}})(\Pi_{indset}^{1/2})^T \\ &\succeq \Pi_{indset}^{1/2}((1-o(1)) \cdot Id)(\Pi_{indset}^{1/2})^T \\ &\succeq 0 \end{aligned}$$

□

8.4 Correcting Coefficients for the Exact Sum-Color Constraints

8.4.1 Correction via Rainbow Colors

In this section, we describe our scheme of picking coefficient for correction terms. In particular, we will pick coefficient for rainbow colors to offset the generic coefficient from pseudo-calibration for pure-coloring, and additionally, *mandatorily chosen* coefficients of mix-coloring and the vanilla $\mathsf{Colorval}[\cdot]_\beta$ component for rainbow-coloring from pseudo-calibration .

A recursive process for defining correction-coefficient

For a shape α that is decomposed into $\{C_i\}$ with coloring β, we define its color-coefficient as

$$Ev(\alpha)_\beta := \prod_{i:C_i} Ev(C_i)_{\beta(C_i)} .$$

In particular, for each connected component, we will pick $\kappa(C_i)_{\beta(C_i)}$ as a correction coefficient, and set

$$Ev(C_i)_{\beta(C_i)} := \mathsf{Colorval}[C_i]_{\beta(C_i)} + \kappa(C_i)_{\beta(C_i)} ,$$

for each component C_i and $\beta(C_i)$ its coloring. Given a connected component C, we define its correction-coefficient in the following recursive process.

1. For any C such that $E(C) = \emptyset$ or tree-like (i.e. $\sum_{\beta:V(C)\to[k]} \mathsf{Colorval}[C]_\beta = 0$ from vanilla pseudo-calibration), set $\kappa(C)_\beta := 0$ for any β coloring of C. In other words, correction is not needed and we leave the coefficient untouched, formally,

for any $\beta : V(C) \to [k]$,

$$Ev(C)_\beta = \mathsf{Colorval}[E(C)]_\beta \,.$$

2. From this point on, we focus on any component C that is not tree-like.

3. For any β that is a pure-coloring (i.e. each edge is a pure-color edge), set $\kappa(C)_\beta := 0$. In other words, the final value is completely determined by the generic component, i.e., $Ev(C)_\beta = \mathsf{Colorval}[C]_\beta$.

4. For any β that is a mixed-coloring (i.e. it contains both pure-color and cross-color edge), let $\mathcal{G}(C, \beta)$ be the subgraph obtained by removing pure-color edge under β. Note that $\mathcal{G}(C, \beta)$ may not be a single connected component, while each component in $\mathcal{G}(C, \beta)$ is a rainbow-component (with some abuse of notation, we consider an isolated vertex as a rainbow component as well). Let $\mathcal{G}(C, \beta) = \{\cup_i G_i\}$ be its decomposition into connected components after removal of pure-color edges such that each G_i is a rainbow component,

$$Ev(C)_\beta = (-1)^{|\text{pure}(C)_\beta|} \prod_i Ev(G_i)_\beta \,,$$

where we define $|\text{pure}(C)_\beta|$ to be the number of pure-color edge removed in the process.

5. Finally, notice the above process settles the coefficient for pure-color and mixed-value assignments, and the only color-assignments with undefined color-value are the rainbow-colors. Let $|RS_C|$ denote the rainbow size of component C, i.e. the number of rainbow-colorings for C, define the coefficient for each rainbow-

color β as

$$\kappa(C)_\beta \coloneqq \frac{1}{|RS_C|} \cdot D_C$$

for

$$D_C \coloneqq (-1) \cdot \Big(\sum_{\substack{\beta \\ \text{:pure-coloring of C}}} Ev(C)_\beta + \sum_{\substack{\beta \\ \text{mix-coloring of } C}} Ev(C)_\beta + \sum_{\substack{\beta: \\ \text{rainbow of } C}} \mathsf{Colorval}[C]_\beta \Big)$$

and recall we take

$$Ev(C)_\beta \coloneqq \mathsf{Colorval}[C]_\beta + \kappa(C)_\beta$$

8.4.2 Bounding the Magnitude of a Color Ribbon

In the following discussion, we proceed to bound the magnitude of a single correction ribbon, in particular, we are interested in the magnitude of each rainbow-coloring, as that of a mixed-coloring can then be bounded in magnitude by a product of rainbow-coloring.

Since the rainbow-coloring gets assigned the same correction term $\frac{D_C}{|RS_C|}$, to bound the magnitude of a single rainbow-ribbon, it suffices for us to upper bound

$$|D_C| = \left| \left(\sum_{\beta\text{:pure-coloring of } C} Ev(C)_\beta + \sum_{\beta\text{:mix-coloring of}C} Ev(C)_\beta + \sum_{\beta\text{:rainbow of C}} Ev(C)_\beta \right) \right|,$$

Towards this end, we define the following candidate upper bound function,

$$\mathsf{Rb}(C) \coloneqq \sum_{\beta\text{:rainbow}} |\mathsf{Colorval}[C]_\beta| + \sum_{\beta\text{:pure}} |\mathsf{Colorval}[C]_\beta| + \sum_{\beta\text{:mixed}} \big(|\mathsf{Colorval}[C]_\beta| + |\kappa(C)_\beta| \big)$$

To motivate the above function, notice that we have

$$\sum_{\beta:\text{rainbow}} |\mathsf{Colorval}[C]_\beta| + |D_C|$$
$$\leq \sum_{\beta:\text{rainbow}} |\mathsf{Colorval}[C]_\beta| + \sum_{\beta:\text{pure}} |\mathsf{Colorval}[C]_\beta| + \sum_{\beta:\text{mixed}} |\mathsf{Colorval}[C]_\beta| + |\kappa(C)_\beta|\,,$$

and the RHS is exactly our upper bound function $\mathsf{Rb}(C)$ as defined. Moreover, we will work with a similar function $\mathsf{Rb}_{\mathsf{pinned}}(C)$ that pins down an arbitrary vertex to an arbitrary color defined as

$$\mathsf{Rb}_{\mathsf{pinned}}(C) := \sum_{\substack{\beta:\text{rainbow-coloring for } C \\ \beta(v)=a}} |\mathsf{Colorval}[C]_\beta| + \sum_{\substack{\beta:\text{mixed/pure coloring for } C \\ \beta(v)=a}} |\mathsf{Colorval}[C]_\beta| + |\kappa(C)_\beta|$$

Note that the above remains well-defined even without picking v and a as the particular value is independent of the vertex and color choice. With the above definition, we have

$$\mathsf{Rb}(C) = k \cdot \mathsf{Rb}_{\mathsf{pinned}}(C)$$

Before we dig in further, we first observe that the above function is easy to bound for C of the following structures.

Proposition 8.4.1 (Ground-component)**.** *For any connected component G_i such that G_i is either a cycle, tree-like, or trivial (i.e. $E(G_i) = \emptyset$), with some vertex $v \in G_i$,having its color given ("pinned"), we have*

$$|\mathsf{Rb}_{\mathsf{pinned}}| \leq 2^{|V(G_i)|-1}$$

Proof. The key is to observe that for any mix-coloring for β for G_i with these structure assumptions,

$$Ev(G_i)_\beta = \mathsf{Colorval}[G_i]_\beta$$

Therefore,

$$\mathsf{Rb}_{\mathsf{pinned}}(G_i) = \sum_{\beta:\text{rainbow}/\text{mixed},\beta(v)=a} |\mathsf{Colorval}[G_i]_\beta| + \sum_{\beta:\text{pure},\beta(v)=a} |\mathsf{Colorval}[G_i]_\beta| \leq 2^{|V(C)|-1},$$

□

Proposition 8.4.2. *For any component C,*

$$|\mathsf{Rb}_{\mathsf{pinned}}(C)| \leq 2^{|V(C)|} \cdot (2 \cdot |E(C)|)^{|E(C)|}.$$

We defer the proofs to the above proposition and lemma to section 8.11.1 , while we note that these control the blow-up incurred by our correction term relative to the original color-value. It is most crucial to us that it does not give extra blow up in k. We incur a slight blow-up of $O(|E(C)|)^{|E(C)|}$ that does not pose substantial change to our result as this is ultimately subsumed by slack for $2^{\mathsf{dsos}^{O(1)}}$ dependence in the dense regime (or $\mathrm{poly}(\mathsf{dsos})$ in the sparse regime) which we do not opt to optimize in this work.

Lemma 8.4.3 (Color-value bound for floating component C)**.** *For any component C, we have*

$$Ev(C) \coloneqq \sum_{\beta:V(C)\to[k]} Ev(C)_\beta = 0$$

Proof. This follows immediately by our choice of correction term D_C,

$$\begin{aligned} Ev(C) &= \sum_{\beta:pure} Ev(C)_\beta + \sum_{\beta:mixed} Ev(C)_\beta + \sum_{\beta:rainbow} Ev(C)_\beta \\ &= \left(\sum_{\beta:pure} Ev(C)_\beta + \sum_{\beta:mixed} Ev(C)_\beta + \sum_{\beta:rainbow} \mathsf{Colorval}[C]_\beta \right) + D_C \\ &= 0 \end{aligned}$$

□

As our coefficient construction factorizes over connected components, we also have

Corollary 8.4.4. *For any subgraph τ that contains a collection of connected components $\{C_i\}$,*

$$Ev(\tau) = \sum_{\beta: V(\tau) \to [k]} Ev(\tau)_\beta = 0$$

8.4.3 Construction of Final Moment Matrix

With the above scheme of identifying edge-value Ev for color-ribbon, we are ready to formally define our moment matrix. Fix $D_V = \Theta(\mathsf{dsos} \log n)$ to be the truncation parameter, we define our moment matrix as

Definition 8.4.5 (Defining final moment)**.** *For any $S \subseteq [n]$, and $\beta_S : S \to [k]$, we define*

$$\widetilde{\mathbb{E}}[x_{S,\beta_S}](G) \coloneqq \sum_{\substack{R \subseteq \binom{n}{2} \\ |V(R) \cup S| \le D_V}} \sum_{\substack{\widehat{\beta}: V(R) \to [k] \\ \widehat{\beta}(S) = \beta_S}} \left(\frac{1}{k}\right)^{|V(R) \cup S|} \cdot Ev(R)_{\widehat{\beta}} \cdot \chi_R(G)$$

Switching back to the language of moment matrix, it is equivalent to define our moment matrix in the following expansion of graph matrices,

Definition 8.4.6 (Moment Matrix $\widetilde{\Lambda}$ in Graph Matrix Basis)**.** *We define $\widetilde{\Lambda}$ as*

$$\widetilde{\Lambda}(G) = \sum_{\substack{(\tau, CP) \textit{color-shape} \\ |V(\tau)| \le D_V}} \underbrace{\left(\frac{1}{k}\right)^{|V(\tau)|} \cdot \sqrt{\frac{1-p}{p}}^{|E(\tau)|} \cdot Ev(\tau)_{CP}}_{\widetilde{\lambda}(\tau)_{CP}} \cdot \mathsf{M}_{\tau, CP}(G)\,.$$

We ignore the dependence on G throughout, and additionally call $\widetilde{\lambda}(\tau)$ the unscaled coefficient.

8.4.4 Verifying the Hard Constraints

Claim 8.4.7 (Normalization)**.** $\widetilde{\mathbb{E}}[1] = 1$.

Proof. Observe that trivial shape gives 1 to $\widetilde{\mathbb{E}}[1]$, and any non-trivial shape is floating for $\widetilde{\mathbb{E}}[1]$ and gives total $Ev = 0$ by correction scheme in lemma 8.4.3. □

Claim 8.4.8 (Booleanity)**.** $\widetilde{\mathbb{E}}[x_{S,\beta_S}^2] = \widetilde{\mathbb{E}}[x_{S,\beta_S}]$ *holds for any* $S \subseteq [n]$ *with* $|S| \leq \mathsf{dsos}$ *and any* β_S.

Proof. This is immediate by our definition of moment that does not depend on the multiplicity degree of x in (S, β_S). □

Claim 8.4.9 (Exact color constraint)**.** *For any* $S \subseteq [n]$ *and* $|S| \leq \mathsf{dsos}$*, we have*

$$\widetilde{\mathbb{E}}[\sum_{\beta_S : S \to [k]} x_{S,\beta_S}] = \widetilde{\mathbb{E}}[1]\,.$$

Proof. Observe the RHS is 1 as the trivial shape is the only contributing shape, while any subset of edges is floating and therefore have a color-coefficient of 0 under our correction by our bound on floating shape in lemma 8.4.3. On the other hand, the trivial shape gives 1 in total to the LHS after summing over all color-assignment, and any non-trivial shape again becomes floating after summing over all vertices and gives value 0 as on the RHS. □

Claim 8.4.10 (Independent-set constraint)**.** *For any* $(S, \beta(S))$ *of size* $|S| \leq \mathsf{dsos}$*, we have*

$$\widetilde{\mathbb{E}}[x_{S,\beta(S)}] = 0$$

if for any color in $f \in \beta(S)$*,* $f(S)$ *the set of vertices in* S *receiving color* f *is not an independent set.*

Proof. Observe that for any color-ribbon (R, β) viewed as a edge-set that contributes to $\widetilde{\mathbb{E}}[x_{S,\beta(S)}]$, w.l.o.g., assume that R does not have any pure-color edge under β, let U be the

set of possible pure-edges in $V(S)$ under β_S, by construction of our Ev coefficient,

$$\sum_{P \subseteq U} Ev(R \cup P, \beta) \cdot \chi_{R \cup P}(G) = \sum_{P \subseteq U} (-1)^{|P|} Ev(R, \beta) \cdot \chi_{R \cup P}(G) = Ev(R, \beta) \cdot \chi_R(G) \cdot \prod_{e \in U} (\chi_e - 1)$$

And note that the above is a missing edge indicator on pure-color edges in R. Summing over all color-ribbon R (that does not contain any pure-color edge in S under β_S) gives us the desired independent set indicator constraint on (S, β_S). □

8.5 Decomposition via Rainbow Minimum Vertex Separator

In this section, we discuss the core strategy of applying approximate matrix factorization for our candidate matrix.

8.5.1 Factorization of Color Shapes into Rainbow Components

In the previous works, a core strategy is to decompose according to a shape's underlying leftmost/rightmost Minimum Vertex Separator which allows us to effectively zoom into the "right" eigenspace. This is again our strategy for the bulk of the analysis, in particular, when we restrict to pure color-shapes. However, one immediate challenge arises should one try to adopt this strategy for correction terms: the coefficient for correction term, when restricted to a single component, does not have any nice product structure, and this poses novel challenges for applying the aforementioned techniques.

For a given color-ribbon (R, β) of shape α with $\alpha = \{C_i\}$ being its decomposition into connected components, recall that it has final coefficient for the rescaled moment matrix

Λ given by

$$\widetilde{\lambda}(\alpha)_\beta = \prod_i \lambda(C_i)_{\beta(C_i)} = \prod_i \left(\frac{1}{k}\right)^{|V(C_i)|} \sqrt{\frac{p}{1-p}}^{|E(C_i)|} \cdot Ev(C_i)_{\beta(C_i)}$$

for $Ev(C_i)$ defined from our recursive process in section 8.4.

We now zoom into the Ev term, as the other two factors are standard in SoS lower bounds, and factorize nicely according to vertices and edges. However, Ev does not immediately factorize due to the (-1) sign in the potential correction term on rainbow-coloring. That said, by our construction, it further factorizes over pure-color edges into rainbow components (notice we may see each isolated vertex with no edge as a rainbow component as well as it does not contain any rainbow color edge).

For convenience, we make the following definition.

Definition 8.5.1 (Rainbow-collection $\mathcal{G}(\tau, \beta)$)**.** *Given a subgraph (or broadly, a shape viewed as a subgraph) τ and coloring β, let $\mathcal{G}(\tau, \beta) = \{G_i\}_i$ denote the collection of connected components in τ after removing pure-color in $E(\tau)$ under β where each G_i is a rainbow component under β. This definition extends to color-partition and color-shape (τ, CP) in a straightforward manner.*

By our coefficient scheme, its edge-value coefficient of a component C_i is given by

$$Ev(C)_{\beta(C)} = (-1)^{pure(C)_\beta} \cdot \prod_{G_i \in \mathcal{G}} Ev(G_i)\,,$$

thus, the final coefficient of a color-ribbon may be rewritten as

$$\begin{aligned}
\widetilde{\lambda}(\alpha)_\beta &= \prod_{\substack{i:C_i \\ \text{connected comp. in } \alpha}} \left(\frac{1}{k}\right)^{|V(C_i)|} \sqrt{\frac{p}{1-p}}^{|E(C_i)|} \cdot Ev(C_i)_{\beta(C_i)} \\
&= \prod_{\substack{i:C_i \\ \text{connected comp. in } \alpha}} \left(\frac{1}{k}\right)^{|V(C_i)|} \left(-\sqrt{\frac{p}{1-p}}\right)^{|pure(C_i)_{\beta(C_i)}|} \cdot
\end{aligned}$$

$$\prod_{j:G_j \in \mathcal{G}(\tau,\beta)} \left(\sqrt{\frac{p}{1-p}}^{|E(G_j)|} \cdot Ev(G_j) \right)$$

and observe that the above extends to a color-shape as well.

We now observe that the RHS factorizes over connected component in α, and moreover within each connected component C of α, it further factorizes according to pure-edge into rainbow components. With this observation, our goal now is to design a factorization scheme that respects the above.

8.5.2 Rainbow MVS Subroutine for Proper Color-Shapes

To this end, we present the following recursive process for finding the rainbow-separator S_L, S_R. We defer the discussion on intersection shapes to later section, and focus on decomposition for (proper) color-shapes without any intersection.

Finding Leftmost/Rightmost Rainbow Separator S_L and S_R for a (Proper) Color-Shape

For a shape α that does not contain any intersection, and its coloring $\beta : V(\alpha) \to [k]$, we describe how to identify the leftmost/rightmost rainbow separator S_L and S_R.

1. **Initialization**: We start from the following sets. Let S_L be the leftmost MVS separating U_α and V_α, and S_R the rightmost MVS separating U_α and V_α;

2. **Recursive Leftwards/Rightwards Extensions**: We alternately apply the following operations of **Rainbow-Extension Operation** and **MVS-Extension Operation** (including their updates) on S_L and S_R until convergence, i.e. S_L does not change after a consecutive application of both extension operations, and analogously for S_R.

3. **Rainbow-Extension Operation**: Let $Rv(L)$ be the rainbow components incident to L under β, i.e., the set of vertices that can be reached from L using cross-color edge; similarly let $Rv(R)$ be the rainbow components incident to R;

   Boundary Update: Update S_L (and S_R) to be the vertices in $Rv(L)$ (and $Rv(R)$) reachable from U_α (analogously, from V_α) without passing through any other vertex in $Rv(L)$ (and $Rv(R)$). In words, move S_L leftwards to the new boundary of $Rv(L)$ reachable from the left-boundary U_α, and move S_R rightwards to the new boundary of $Rv(R)$ reachable from right-boundary V_α.

4. **MVS-Extension Operation**: Find S'_L be the leftmost MVS separating U_α and the current S_L, find S'_R be the rightmost MVS separating V_α and the current S_R;

   Boundary Update: Update $S_L \leftarrow S'_L$, and $S_R \leftarrow S'_R$. In words, move S_L leftwards to the new leftmost MVS identified as S'_L, and move S_R rightwards to the new rightmost MVS identified as S'_R.

(S_L, S_R)-Factorization for a Color-Shape

With S_L, S_R constructed from the above, we then define $L(\alpha), M(\tau)$ and $R(\alpha)$ such that $\alpha = L(\alpha) \circ M(\alpha) \circ R(\alpha)$. For the following, we suppress the dependence on α, and write $L = L(\alpha), R = R(\alpha)$.

1. Let $E(L) \subseteq E(\alpha) \setminus E(S_L)$ be the subset of edges that can be reached from U_α without passing through S_L, and let $U_L = U_\alpha, V_L = S_L$. We follow the convention that edges in $E(S_L)$ is reserved to the middle, and therefore not included into $E(L)$.

2. Let $E(R) \subseteq E(\alpha) \setminus E(S_R)$ be the subset of edges that can be reached from V_α without passing through S_R, and let $V_R = V_\alpha, U_L = S_R$. Similarly edges in $E(S_R)$

is reserved to the middle, and not included in R.

3. Let $E(M) := E(\alpha) \setminus (E(L) \cup E(M))$. Let $U_M = S_L$ and $V_M = S_R$.

Remark 8.5.2. *The above procedure is defined for a shape α and a coloring $\beta : V(\alpha) \to [k]$. However, observe it extends to color-partition CP immediately as the only dependence over β is whether some subset of edges in $E(\alpha)$ being pure/rainbow-color.*

8.5.3 Decomposition of Coefficients via Rainbow MVS

The upgrade from generic MVS to rainbow MVS is that it allows us to factorize the coefficient for each color-shape into left, middle, right shapes. We start by the following definition of shape coefficient that applies to all three of the above.

Definition 8.5.3 (Left/Right Shape coefficient)**.** *For a left shape σ and $\beta : V(\sigma) \to [k]$ its coloring , we let its coefficient be*

$$\lambda(\sigma)_\beta = \left(\frac{1}{k}\right)^{|V(\sigma)|-\frac{|V_\sigma|}{2}} \cdot \sqrt{\frac{p}{1-p}}^{|E(\sigma)|} \cdot Ev(\sigma)_\beta \,.$$

where we recall that

$$Ev(\sigma)_\beta = (-1)^{|pure(\sigma)_\beta|} \prod_{G_i : \mathcal{G}(\sigma)_\beta} Ev(G_i)_{\beta(G_i)}$$

We define similarly for right-shape by taking transpose of a left-shape.

It should be noted that we do not work frequently with left/right shape coefficient directly. That said, we also write the left/right shape coefficient as $\lambda_L(\sigma)$ to emphasize that that these are the left/right shape coefficients. These are only used in the kernel discussion in section 8.8 where we focus on the action of L on the kernel.

Particularly in section 8.6 and section 8.7, we do not work with the left/right shape coefficient defined as above. Instead, our analysis there will be primarily focusing on

shapes with left/right shape factored out, and we call these middle coefficients defined as the following.

Definition 8.5.4 (Scaled Coefficient for (Color) Middle Shape τ)**.** *For a middle shape τ and $\beta : V(\sigma) \to [k]$ its coloring , we let its coefficient be*

$$\lambda(\tau)_\beta = \left(\frac{1}{k}\right)^{|V(\tau)|-\frac{|U_\tau|+|V_\tau|}{2}} \cdot \sqrt{\frac{p}{1-p}}^{|E(\tau)|} \cdot Ev(\tau)_\beta \,.$$

where we recall that

$$Ev(\tau)_\beta = (-1)^{|pure(\tau)_\beta|} \prod_{G_i : \mathcal{G}(\tau)_\beta} Ev(G_i)_{\beta(G_i)}$$

The definition extends to intersection-left/right-shape γ and γ' to be defined in definition 8.5.17. The particular vertex factor scaling also applies to extension pieces to be defined in definition 8.5.10 and definition 8.5.9.

Again, observe the only dependence of the coloring β is whether each edge is pure-color or rainbow-color, while their exact label in $[k]$ does not matter. Therefore, the above definition extends to color partition CP and color-shape written as a tuple of (τ, CP) immediately, and we write its coefficient as $\lambda(\tau)_{\mathrm{CP}}$.

With the above factorization scheme, we verify that the coefficient factorizes according to the $L/M/R$ components designed above.

Proposition 8.5.5 (Factorization of matrix entry)**.** *For any ribbon of shape α with coloring β with decomposition $\alpha = \sigma \circ \tau \circ \sigma'$, we have*

$$\widetilde{\lambda}(\alpha)_\beta = \lambda(\sigma)_{\beta(V(\sigma))} \cdot \lambda(\tau)_{\beta(V(\tau))} \cdot \lambda(\sigma')_{\beta(V(\sigma'))} \,.$$

Proof. We first note that restricted to vertex factors and edge factors of $\sqrt{\frac{p}{1-p}}$ factorizes, the above holds in a straightforward manner as in previous factorizations (eg. see [JPR+22])

. We shall focus on the term $Ev(\alpha)$ term as these do not factorize over vertices or edges immediately.

By our construction, the Ev coefficient factorizes according to rainbow components by removing pure-color edge, and we have

$$Ev(\alpha) = (-1)^{|pure(\alpha)_\beta|} \prod_{G_i \in \mathcal{G}_{(\alpha)_\beta}} Ev(G_i)_{\beta(G_i)}$$

It now remains to show

$$Ev(\sigma)_{\beta(\sigma)} \cdot Ev(\tau)_{\beta(\tau)} \cdot Ev(\sigma')_{\beta(\sigma')} = (-1)^{|pure(\alpha)_\beta|} \prod_{i \in \mathcal{G}_{(\alpha)_\beta}} Ev(G_i)_{\beta(G_i)}$$

Unpacking the definition for Ev on the LHS, let $\mathcal{G}(\sigma)$ be the collection of rainbow-components in σ after removing pure-color edge under β , and define that analogously for τ and σ', we have

$$LHS = \prod_{G_x \in \mathcal{G}(\sigma)} Ev(G_x)_{\beta(G_x)} \cdot \prod_{G_y \in \mathcal{G}(\tau)} Ev(G_y)_{\beta(G_y)} \cdot \prod_{G_z \in \mathcal{G}(\tau)} Ev(G_z)_{\beta(G_z)} \cdot (-1)^{|pure(\sigma)_\beta|+|pure(\tau)_\beta|+|pure(\sigma')_\beta|},$$

Comparing with the RHS, it is easy to observe that the (-1) factor equal on both sides as

$$|pure(\sigma)_\beta| + |pure(\tau)_\beta| + |pure(\sigma')_\beta| = |pure(\alpha)|_\beta$$

since all three sets on the LHS are pairwise disjoint and their union is exactly $pure(\alpha)_\beta$. It remains for us to show a bijection between the collection of rainbow components on the LHS and the collection on the RHS.

To show from left to right, this direction is immediate as we notice each rainbow component in σ, τ, σ' is also a rainbow component in $\alpha = \sigma \circ \tau \circ \sigma'$. For the other direction, it is crucial to observe that each component in α is in exactly one of σ, τ, σ'. This

is immediate for rainbow components that do not touch on vertices in $U_\tau \cup V_\tau$ as they are either in $\mathcal{G}(\sigma)$ or $\mathcal{G}(\sigma')$ if they are connected to U_α or V_α, and if neither, it is in $\mathcal{G}(\tau)$. Finally, for any rainbow component G_i that touches on vertices in $U_\tau \cup V_\tau = S_L \cup S_R$, each such component is in $\mathcal{G}(\tau)$ only: the rainbow component does not have edges in σ as the $V(G_i) \cap S_L$ is not incident to any rainbow-color edge in σ as otherwise the rainbow (leftwards) extension can be further applied, violating the assumption that σ is the final left-shape at the end of the factorization process. The proof holds identically for S_R and σ'. □

Structure of Left/Right Shapes

Definition 8.5.6 (Color-Left/Right-Shape)**.** *For a left-shape σ with coloring CP, it is a color-left-shape if*

1. *V_σ is not incident to any cross-color edge in CP;*
2. *V_σ is the unique MVS for σ (equiv. σ is a left-shape) .*

Analogously, for a right-shape σ with coloring CP, it is a color-right-shape if

1. *σ is a right shape;*
2. *U_σ is not incident to any cross-color edge in CP.*

Similar to how we abstract out labels in $[n]$ from ribbons to shapes, the above process can be abstracted out to color partition by identifying the collection of vertices that have the same color and ignoring the specific label in $[k]$. Therefore, for the following combinatorial charging analysis, we will work in the level of color-partition.

We are now ready to define our desired left-shape matrix L to write our moment matrix $\Lambda = LQL^T$ in the three-way decomposition.

Proposition 8.5.7. *For any shape α with color CP, let $\alpha = \sigma \circ \tau \circ \sigma'$, σ and σ' with color $CP(\sigma), CP(\sigma')$ satisfy the above property of left/right shape.*

Proof. We verify the two conditions for a left-shape. First, if V_σ is incident to some vertex $a \in V(\sigma) \setminus V_\sigma$ via a cross-color edge, there is a path from U_α to a without passing through V_σ. However, a can be reached via a cross-color edge in V_σ, and by construction, we require V_σ to be the MVS between U_σ and any vertex reachable from V_σ without using pure-color edge, including a, and therefore we have arrived at a contradiction. The second property follows as we choose the candidate leftmost MVS each time before we extend for cross-color edges. □

Crucial to our three-way factorization is the following left-shape matrix L that we are ready to define now,

Definition 8.5.8 (Color-Left-Shape Matrix L)**.** *We define $L = \sum_{(\sigma,CP):\textit{left-color-shape}} \lambda(\sigma)_{CP} \cdot \mathsf{M}_{\sigma,CP}$.*

Extension Pieces As we define our rainbow MVS subroutine recursively, it is also useful for us to additionally define extension pieces/shapes to help us keep track of how a middle shape arises in the decomposition.

Definition 8.5.9 (Left/rightwards Rainbow Extension Shape/Piece γ^{rb})**.** *We call a color-shape γ with color-partition CP a leftwards rainbow extension shape/piece if*

1. *V_γ is the unique MVS for γ;*
2. *Any vertex in U_γ is in the rainbow component of some vertex in V_γ under CP.*
3. *Every vertex is connected to U_γ, and to V_γ.*

Taking the shape transpose allows us to define right-rainbow extension.

Definition 8.5.10 (Left/rightwards MVS Extension Shape/Piece γ^{mvs})**.** *We call a color-shape γ with color-partition CP a leftwards MVS extension shape/piece if*

1. *U_γ is an MVS for γ;*

2. *Any edge in $E(\gamma)$ incident to V_γ is pure-color under CP.*

Proposition 8.5.11 (Decomposition of Middle Shape via Tracking Rainbow-MVS Decomposition Process)**.** *For any shape α with color CP , let $\alpha = \sigma \circ \tau \circ \sigma'$ be the decomposition from the above process, τ further admits the following decomposition*

$$\tau = \gamma_{t_l}^{mvs} \circ \gamma_{t_l}^{rb} \circ \cdots \circ \gamma_1^{mvs} \circ \gamma_1^{rb} \circ \tau_{base} \circ (\gamma')_1^{rb} \circ (\gamma')_1^{mvs} \circ \dots \gamma_{t_r}^{rb} \circ \gamma_{t_r}^{mvs}$$

for some collection of $\{\gamma_i^{rb}, (\gamma')_i^{rb}\}$ rainbow-extension shapes and $\{\gamma_i^{mvs}, (\gamma')_i^{mvs}\}$ mvs-extension shapes for $t_l, t_r \leq$ dsos, and τ_{base} some generic middle shape (i.e. $U_{\tau_{base}}, V_{\tau_{base}}$ are the MVS of τ_{base}).

Proof. Consider applying the prescribed process of finding rainbow separators for α. The base level starts at S_L, S_R the leftmost and rightmost MVS of α, and the component in between gives τ_{base} that is a generic middle shape.

Each time we apply the process, it starts from the given boundary, and first extends to left/right via rainbow-edges and then check for MVS extension. For each iteration of rainbow-extension, we verify that each piece γ, γ^T being factorized out satisfies the properties rainbow-extension. Consider γ a piece obtained from leftwards rainbow-extension. Firstly, note that V_γ is the unique MVS for γ, as otherwise the prior iteration of finding the leftmost MVS can be further moved to the left. Secondly, observe any vertex in U_γ is in the rainbow component of some vertex in V_γ in the prescribed CP by definition of our decomposition process. This gives us the left/rightwards rainbow-extension shape in each level.

At each level, notice we further find X, Z the component between U_α and the U_γ

boundary, which component we call X (and similarly for Z). The component between the MVS and U_X in X is then further pulled out and put in the middle (similarly for Z), it is immediate to observe the component in between is also a left/rightwards MVS extension piece .

Finally, observe that we have at most dsos levels as each time the candidate boundary moves to the left/right via MVS-extension piece, at least some vertex is added compared to the previous MVS-extension. Therefore, we have at most dsos levels of MVS-extensions.

□

8.5.4 Rainbow MVS Decomposition for Intersection Shapes

Similar to all prior works that utilize the machinery of approximate matrix factorization, we incur error terms in matrix multiplication of graph matrix of shapes, and we call these error terms "intersection shapes". We refer reader to [JPR+22, JPRX23] for a more thorough introduction of intersection terms, while we focus on the adaptation of discussion and terminologies therein to our setting.

Preliminaries for Intersection Shapes In the above decomposition for proper shape, we intend to write a shape $\alpha = \sigma \circ \tau \circ \sigma'$ via matrix multiplication of

$$\mathsf{M}_\alpha = \mathsf{M}_\sigma \cdot \mathsf{M}_\tau \cdot \mathsf{M}_{\sigma'}$$

However, this equality does not hold exactly, and instead we encounter intersection shapes defined as the following.

Definition 8.5.12 (General Intersection Shape/Profile)**.** *We call a shape τ_P a general intersection shape if it can be obtained from the following*

1. *It is associated with a t-way products of graph matrices from $\alpha_1, \alpha_2, \ldots, \alpha_t$ such that each consecutive shape is composable (sharing with the same boundary condition);*

2. *Additionally associated with it is a intersection profile P_{τ_P} that specifies how each vertex in $V(\alpha_i)$ share the same vertex label in $[n]$, i.e.* intersect *with vertices in other parts of $V(\alpha_j)$ for $j \neq i$ while preserving local injectivity within each piece/shape.*

With some abuse of notation, we may also refer to the intersection profile as τ_P. Note that this also allows us to view proper shape as an intersection shape as well with trivial intersection profile.

It is intuitive to picture $P(\tau_P)$ as additionally drawing edges between intersected vertices in $\sigma \circ \tau \circ \sigma'$, as captured by the following example.

Example 8.5.13. *The following is an example of intersection pattern, where the orange curves specify how different set of vertices in different parts may intersect by receiving the same label in $[n]$.*

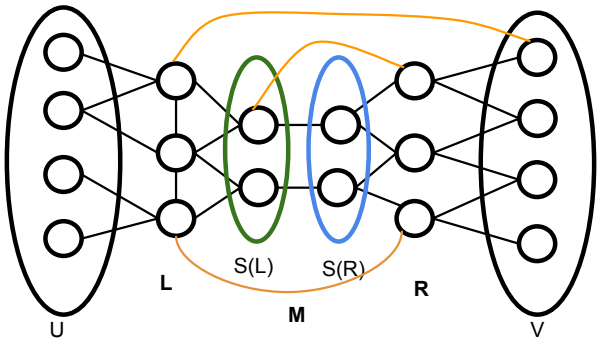


Figure 8.9: Intersection Patten

With intersection shapes considered, we have instead the following equality from matrix multiplication of graph matrices that extend beyond 3-way product,

Proposition 8.5.14. *For composable shapes $\alpha_1, \alpha_2, \ldots, \alpha_t$,*

$$\mathsf{M}_{\alpha_1} \cdot \mathsf{M}_{\alpha_2} \cdot \ldots \mathsf{M}_{\alpha_t} = \sum_{\substack{\tau_P = (\alpha_1 \circ \alpha_2 \circ \cdots \circ \alpha_k) \\ \textit{intersection profile}}} \mathsf{M}_{\tau_P}$$

Proof. The proposition follows immediately by a direct expansion of the LHS, and note that each term is a shape under some intersection profile that specifies how vertices intersect. □

As we shift from proper shapes to improper shapes, a technical issue that comes up is that we have multi-edge in the definition for graph matrix, and this incurs some necessary adjustment for graph matrix norm bounds, captured by the notion of isolated vertices.

Definition 8.5.15 (Isolated vertex)**.** *Given a (possibly with intersection) shape α, we call a vertex $v \in V(\alpha) \setminus (U_\alpha \cup V_\alpha)$ an isolated vertex if it is not incident to any edge in $E(\alpha)$ with odd-multiplicity.*

We are now ready to wrap up the preliminaries for our intersection shapes, and describe the decomposition for them. A priori, these are error terms in the approximate matrix factorization and they are of small norm. However, similar to how matrix factorization is needed, intersection shapes are also of small norm only if we zoom into the "correct" eigenspace (i.e. decomposition).

Recap of Prior Strategy From the previous works, a three-way decomposition of ribbons $L \circ M \circ R$ may incur intersections from the ribbon boundary, and the usual strategy employed is to factor out the the uninersected part from the left and right, and identify the diagonal being charged to as U_γ the leftmost MVS separating U_L from $V_R \cup$ intersection, and $V_{\gamma'}$ the rightmost MVS separating V_R from $U_L \cup$ intersection. This ultimately comes down to verifying the shape $\gamma \circ \tau \circ \gamma'$ can be appropriately charged where γ is the shape

obtained from edges between U_γ and V_L in L, and γ'' the shape obtained from edges between U_R and $V_{\gamma'}$.

Upgrade to Rainbow Separators via Modified Start For our case, we employ the strategy inspired by the above, and incorporate it to our Rainbow MVS subroutine for proper shape by making the likely anticipated change to our initialization,

Instead of starting from the left/rightmost MVS separating U_α and V_α, we start building S_L from the leftmost MVS separating U_α and $V_\alpha \cup Int(\alpha)$, and analogously start building S_R from the rightmost MVS separating V_α and $U_\alpha \cup Int(\alpha)$.

With the modified starting point, the recursive stage remains the same as in the process described in section 8.5.2: we alternately apply the below operations of **Rainbow-Extension Operation** and **MVS-Extension Operation** (including their updates) on S_L and S_R, until each converges, i.e. S_L does not change after a consecutive application of both extension operations, and analogously for S_R.

Crucial to our recursive scheme is to notice that by further decomposing intersection shape, we are effectively factorizing out the non-intersected component in the three-way product. Consider the three-way product of $\sigma \circ \tau \circ \sigma'$ with some intersection incurred to ψ, applying (S_L, S_R)-decomposition from our intersection-operation, we may factor out the intersecting part γ from σ, and γ' from σ', and write

$$\psi = (\sigma - \gamma) \circ \tau_P \circ (\sigma - \gamma)$$

where τ_P is the result of intersection from γ, τ, γ'. Formally, for an intersection shape from the three-way product, we define its corresponding γ and γ' shape within σ and σ' via the following process.

Finding Intersection Shapes γ and γ'

For an intersection shape ψ obtained from three-way product of (σ, τ, σ') , and with (S_L, S_R) identified from the intersection decomposition operation, we define shape γ and γ' as the following.

1. γ shape is defined as the following: let $U_\gamma \coloneqq S_L$, and $V_\gamma = V_\sigma = U_\tau$. Consider $E(\sigma - \gamma)$ be the set of edges in $E(\sigma)$ that can be reached from U_σ without passing through $S_L = U_\gamma$, and we define $E(\gamma) \coloneqq E(\sigma) \setminus E(\sigma - \gamma)$. Under this definition, we follow the convention such that edges between the vertices in $E(U_\gamma)$ are put in $E(\gamma)$.

2. γ' shape is defined analogously as the following: let $U_\gamma \coloneqq U_{\sigma'} = V_\tau$, and $V_{\gamma'} = S_R$. Consider $E(\sigma' - \gamma')$ be the set of edges in $E(\sigma')$ that can be reached from $V_{\sigma'}$ without passing through $S_R = V_{\gamma'}$, and we define $E(\gamma') \coloneqq E(\sigma') \setminus E(\sigma' - \gamma')$.

3. Additionally, we call (γ, τ, γ')'s intersection result τ_P as a three-way intersection.

4. For each γ, γ' shape, we additionally associate with each of them $Int(\gamma)$ and $Int(\gamma')$ the set of vertices that intersect with some other vertex in the intersection profile prescribed by ψ.

Remark 8.5.16. *Though we did not make the coloring-dependence explicit, it should be noted that each shape discussed above is a color-shape that comes with some color-partition.*

Definition 8.5.17 (Intersection-left/right-shape)**.** *We call each color-shape (γ, CP_γ), $(\gamma', CP_{\gamma'})$ obtained from the above process a left/right intersection (color-)shape.*

Useful to our analysis of intersection term is the following structural property of intersection left/right shape.

Proposition 8.5.18 (γ-shape property)**.** *For any left-intersection γ-shape obtained from the above process, it satisfies the following properties:*

1. *γ is a left-color-shape, i.e. V_γ is the unique MVS, and additionally a color MVS that is only incident to pure-color edge;*

2. *U_γ is a color MVS separating U_γ and $Int(\gamma) \cup V_\gamma$: in other words, it is the result of leftwards extensions from the MVS separating U_γ and $Int(\gamma) \cup V_\gamma$.*

Analogous property hold for right-intersection shape γ' by considering its shape transpose into a left-shape.

Proof. The first property holds from σ being a color-left-shape, hence $V_\sigma = V_\gamma$ is a unique MVS that also satisfies the coloring constraint. The second property follows from our construction of S_L from the decomposition process for intersection shape. □

Remark 8.5.19. *As with previous SoS lower bounds using graph matrices, a key observation is that the intersection operation and the intersection term decomposition operation are unaffected by replacing a different shape of $\sigma - \gamma$ and $\sigma' - \gamma'$ provided they share the same boundary condition of $V_{\sigma-\gamma} = S_L = U_\gamma$ and $U_{\sigma-\gamma} = S_R = V_{\gamma'}$.*

With the definition of γ shape, we are also able to define the collection of intersection shapes that come up in our analysis obtained after factoring out the non-intersection part.

Definition 8.5.20 (Factored Intersection Shape/Profile with Level-t)**.** *We call a shape τ_P a general intersection shape if it can be obtained from the following*

1. *It is associated with a t-way products of shapes from $\gamma_t \circ \gamma_{t-1} \circ \tau \circ \gamma'_1 \circ \dot{\circ}\gamma'_t$ such that each consecutive shape is composable (sharing with the same boundary condition);*

2. *For each level $i \in [t]$ γ_i and γ_i' are intersection left/right-shapes respectively, moreover, at least one of them is non-trivial;*

3. *τ is a proper middle shape (i.e. U_τ and V_τ are obtained under left/right-extensions from the leftmost/rightmost MVS of τ).*

4. *Additionally associated with it is a intersection profile P_{τ_p} that specifies how each vertex in $V(\alpha_i)$ share the same vertex label in $[n]$, i.e.* intersect *with vertices in other parts of $V(\alpha_j)$ for $j \neq i$ while preserving local injectivity within each piece/shape.*

8.5.5 Summary of the Overall Decomposition: Finding Middle Matrix Q

We now employ the recursive factorization machinery using the new L matrix defined via rainbow MVS decomposition as in [BHK+16, Pan21, JPR+22, JPRX23, KPX24]. Again, our notation reserves the use of Q for the final middle matrix, and use $\widetilde{Q}$ for the intermediate steps.

Definition 8.5.21 (Recursive Factorization via L)**.** *The recursive factorization of the candidate matrix Λ proceeds as the following. Starting from $i = 0$, and set*

$$\widetilde{Q} = Q_0 := \sum_{(\tau,CP):\text{proper middle shape}} \lambda(\tau)_{CP} \cdot \mathsf{M}_{\tau,CP}$$

For each level of $i \in [\mathsf{dsos}]$,

1. *Consider $M_{target} = \Lambda - L\widetilde{Q}L^T + truncation_i$ where we have*

$$truncation_i := \sum_{\substack{(\tau,CP)\in L\circ Q_i\circ L^T \\ |V(\tau)|>D_V}} \lambda(\tau)_{CP}\mathsf{M}_{\tau,CP}$$

be the collection of shapes from LQ_iL^T that exceed our truncation threshold of $|V(\alpha)| \leq D_V$

2. *Apply intersection operation on M_{target} via its expansion into intersection shapes, and this gives a new collection of (intersected) middle shapes from the decomposition process by processing τ_P at the i-th level of intersections, and we write*

$$Q_i := (-1)^i \cdot \sum_{(\tau_P, CP): \textit{intersection-shape with i-levels}} \lambda(\tau_P)_{CP} \cdot \mathsf{M}_{\tau_P, CP}$$

   for $i \geq 1$.

3. *Update $\widetilde{Q} \leftarrow Q_0 + Q_1 + \ldots Q_i$ and continue this process until $M_{target} = 0$.*

Via the standard machinery for matrix approximate factorization from [JPR+22, JPRX23], it is guaranteed that the process terminates within dsos levels. Immediately, this allows us to write

$$\Lambda = L\widetilde{Q}L^T + \text{truncation}$$

for $\widetilde{Q} = \sum_{i \leq 2\mathsf{dsos}} Q_i$ and $M_{trunc} = \sum_{i \leq 2\mathsf{dsos}} \text{truncation}_i$. As foreshadowed in our technical overview, we have a non-trivial kernel in Q, and for our kernel trick to apply, it is important for us to continue to apply factorization for the truncation term. This is formally captured by the following.

Proposition 8.5.22 (Furrther Factorization of Truncation Error Term)**.** *There is a matrix Q_{trunc} one can find such that*

$$M_{trunc} = LQ_{trunc}L^T$$

Moreover,

$$\|Q_{trunc}\| \leq \|M_{trunc}\| \cdot n^{O(\mathsf{dsos})}$$

Proof. This follows by our well-conditionedness bound of L, that we have $L \succeq n^{-O(\mathsf{dsos})} Id$,

and therefore, we can find

$$Q_{trunc} \coloneqq (L^{-1})M_{trunc}(L^{-1})^T$$

□

As a corollary of the above, we obtain the following approximate factorization for our moment matrix,

$$\Lambda = L(\widetilde{Q} + Q_{trunc})L^T .$$

At this point, we are ready to dive into the magnitude bound for Q. Towards this end, standard to works on independent set, we first factor out pure-color edges in the diagonal block of Q, and write

$$(\widetilde{Q} + Q_{trunc}) = \Pi_{indset}^{1/2} \cdot Q \cdot \Pi_{ind}^{1/2}$$

for the diagonal matrix Π_{ind} defined as the following

$$\Pi_{ind}[(S, \beta_S), (S, \beta_S)] = \prod_{c \in [k]} (\frac{1}{1-p})^{\binom{|\beta_S(c)|}{2}} \cdot \mathbf{1}[\beta_S(c) \text{ is an Independent Set in } G]$$

where we write $\beta_S(c)$ be the set of vertices in S that receive color $c \in [k]$ under β_S. With this factorization, we may now assume each shape in $\widehat{Q}$ to not have any pure color edge in $E(U_\tau \cap V_\tau)$. However, as hinted upon by our kernel issue due to sum-color constraint, it is not quite sufficient to apply magnitude bound yet.

As we shall elaborate on in section 8.8, the kernel from sum-color constraint is evident in the trivial shapes on the diagonal. This includes the usual PSD mass of Id_U, however, it also contains non-trivial intersection profiles that can be grouped to a diagonal matrix of $(Id - \frac{J_k}{k})^{\otimes |U|}$ on the block of coloring for U.

Definition 8.5.23 (Trivial)**.** *We call a color-shape (τ, CP) (potentially with non-trivial intersection) trivial if*

1. $U_\tau = V_\tau = V(\tau)$;
2. $pure(E(\tau))_\beta = \emptyset$.

In the next two sections, we put away this issue and show that we may apply a magnitude bounds for non-trivial shapes with pure-color edges in $E(U_\tau \cap V_\tau)$ removed. Specifically, the next section focus on non-intersection shapes in Q_0, and the subsequent section focuses on shapes with intersections in Q_i for $i \geq 1$.

Lemma 8.5.24 (Informal version of lemma 8.6.2)**.** *For any (non-trivial) shape $(\tau, CP) \in Q_0$, we have*

$$\|\lambda(\tau)_{CP} \cdot \mathsf{M}_{\tau,CP}\| = o(1)$$

Lemma 8.5.25 (Informal version of lemma 8.7.3)**.** *For any non-trivial intersection shape (τ_P, CP) in Q, we have*

$$\|\lambda(\tau_P)_{CP} \cdot \mathsf{M}_{\tau_P,CP}\| = o(1)$$

8.6 Combinatorial Charging Analysis for Non-intersected Shapes

For clarity of our presentation, we focus on the dense regime of $G(n, \frac{1}{2})$ in the subsequent sections, and we defer the necessary changes to the appendix for the application to the sparse regime of $G(n, \frac{d}{n})$ for $d = o(n)$.

To get started, recall from proposition 8.5.11, a middle shape $\tau \in Q_0$ admits the following decomposition

$$\tau = A_t \circ X_t \circ A_{t-1} \circ X_{t-1} \circ \cdots \circ A_1 \circ X_1 \circ Y \circ Z_1 \circ B_1 \circ \cdots \circ Z_{t'} \circ B_{t'}$$

where we abbreviate the color-partition dependence of each shape as it is clear each is a color-shape, each A_i, B_j is a left/right-rainbow-extension shape in the decomposition, and X_i, Z_j is a left/right-MVS-extension shape and Y is a base middle shape.

Motivation for Slack Function Finally, we handle the discrepancy between a single shape and summing over shapes as ultimately we would be interested in the summation of shapes while the above discussion focus on single-shape. To cope with the summation over color-shapes, we introduce the slack function for each shape defined as the following.

Definition 8.6.1 (Slack function)**.** *For each shape (τ, CP), we define*

$$\mathit{slack}(\tau, CP) = |\lambda_{\tau,CP}| \cdot \|\mathsf{M}_{\tau,CP}\|$$

For intuition, one should think of slack$(\cdot)$ as a priori some factor less than 1 we want to control via combinatorial analysis. Moreover, we ideally want to upper bound it away from 1 so that we continue to get a gap after summing over all shapes. This is the main task of our combinatorial analysis, culminated at the following lemma,

Lemma 8.6.2 (Main lemma for Q_0)**.** *For any non-trivial shape $(\tau, CP) \in Q_0$, we have*

$$\|\lambda(\tau)_{CP} \cdot \mathsf{M}_{\tau,CP}\| \leq o(1)$$

in our regime of k. Formally, we show

$$\|\lambda(\tau)_{CP} \cdot \mathsf{M}_{\tau,CP}\| \leq \mathit{slack}(\tau_p, CP)$$

for

$$slack(\tau, CP) \coloneqq \left(\frac{\sqrt{n}}{k}\right)^{\frac{1}{2}|V(\tau)\setminus U_\tau \cap V_\tau|} \cdot \left(\frac{1}{k}\right)^{\frac{1}{2}\cdot|rainbow(U_\tau \cap V_\tau)|} \cdot O(|rainbow(E(\tau))_{CP}|)^{|rainbow(E(\tau))_{CP}|} \cdot O(|V(\tau) \setminus U_\tau \cap V_\tau|)^{2|V(\tau)\setminus U_\tau \cap V_\tau|)}$$

To unpack the formal statement for the reader, we point out the first two factors are our focus and they are $o(1)$ once we set $k > \sqrt{n}$, and the last two factors are subpolynomial (per vertex) in our regime of focus and handled via the extra slack of our chosen k from the $\sqrt{n}$ threshold.

Towards this end, we first show factors in $\|\mathsf{M}_{\tau,\mathrm{CP}}\|$ can be decomposed according to each piece they appear in. We introduce the definition for weight function and coefficient in section 8.6.1, and show that each piece can be properly charged in the subsequent sections.

8.6.1 Set-up of Weight, Coefficient Function

Since any floating component comes with $Ev(C) = 0$, we can assume τ to not contain any floating component. We start by unpacking the factors in $\|\lambda(\tau)_{\mathrm{CP}} \cdot \mathsf{M}_{\tau,\mathrm{CP}}\|$ from our norm bound via trace-method in corollary 8.10.4,

For the following analysis, it would be helpful for us to first consider a fixed shape, and rewrite the upper bound of the norm bound factor in the above proposition as

$$\|\mathsf{M}_{\tau,\mathrm{CP}}\|^{\approx} \leq \cdot w(\tau)_{\mathrm{CP}} \cdot O(|V(\tau)|q)^{|V(\tau)\setminus U_\tau \cap V_\tau|}$$

for proxy function

$$w(\tau)_{\mathrm{CP}} \coloneqq \max_{S:\text{separator}} \sqrt{n}^{|V(\tau)\setminus S|} \cdot \max_{\beta\in\mathrm{CP}} k^{|\mathrm{CP}(\tau)|-\frac{|\mathrm{CP}(U_\tau)|+|\mathrm{CP}(V_\tau)|}{2}},$$

where one should observe that $w(\cdot)$ ignores the subpolynomial factor from the norm bound intuitvely.

Factorization of weight function into pieces For τ a shape with color-partition CP, recall that the corresponding weight function as defined is given by

$$w(\tau)_{\mathrm{CP}} = \max_{S:\mathrm{sep}} \sqrt{n}^{|V(\tau)\setminus S|} \cdot k^{|\mathrm{CP}(\tau)| - \frac{|\mathrm{CP}(U_\tau)|+|\mathrm{CP}(V_\tau)|}{2}} .$$

In this section, we show that this function can be further factorized into pieces for

$$\tau = A_t \circ X_t \circ A_{t-1} \circ X_{t-1} \circ \cdots \circ A_1 \circ X_1 \circ Y \circ Z_1 \circ B_1 \circ \cdots \circ Z_{t'} \circ B_{t'} .$$

Since U_Y, V_Y are MVS of τ, we can rewrite the weight function factor as

$$w(\tau)_{\mathrm{CP}} \coloneqq \sqrt{n}^{|V(\tau)| - \frac{|U_Y|+|V_Y|}{2}} \cdot k^{|\mathrm{CP}(\tau)| - \frac{|\mathrm{CP}(U_\tau)|+|\mathrm{CP}(V_\tau)|}{2}}$$

For intuition, consider the simplest case of $\tau = X \circ Y \circ Z$, we now observe that the above bound factorizes according to the X, Y, Z pieces as we may rewrite $|V(\tau)|$ as

$$|V(\tau)| = |V(X) \setminus U_Y| + |V(Y)| + |V(Z) \setminus V_Y| .$$

as the partition into X, Y, Z are piecewise vertex disjoint beyond S_L, S_R, and we follow the convention of reserving the factors of S_L, S_R into $V(Y)$. Similarly, recall that with some abuse of notation, we fix $\arg\max_{\beta \in \mathrm{CP}} k^{|\beta(\tau)| - \frac{|\beta(U_\tau)|+|\beta(V_\tau)|}{2}}$ for CP, and we can rewrite $|\mathrm{CP}(\tau)|$ as

$$|\mathrm{CP}(\tau)| \leq |\mathrm{CP}(X)| + |\mathrm{CP}(Y)| + |\mathrm{CP}(Z)| - |\mathrm{CP}(U_Y)| - |\mathrm{CP}(V_Y)|$$

$$= \left(|\mathrm{CP}(X)| - \frac{|\mathrm{CP}(S_L)|}{2}\right) + \left(|\mathrm{CP}(Y)| - \frac{|\mathrm{CP}(U_Y)| + |\mathrm{CP}(V_Y)|}{2}\right) + \left(|\mathrm{CP}(Z) - \frac{|\mathrm{CP}(S_R)|}{2}\right)$$

where the first line follows as each color of U_Y (and respectively C_Y) contributes to X, Y (and Y, Z).

Definition 8.6.3 (Weight function for pieces)**.** *Define weight function for each piece as*

1. *For left/rightwards MVS/rainbow-extension piece X (and Z, A, B analogously), define*

$$w(X)_{CP(X)} \coloneqq \sqrt{n}^{|V(X)\setminus V_X|} \cdot k^{|CP(X)| - \frac{|CP(U_X)|+|CP(V_X)|}{2}};$$

2. *For the base piece Y, define*

$$w(Y)_{CP(Y)} \coloneqq \sqrt{n}^{|V(Y)| - \frac{|U_Y|+|V_Y|}{2}} \cdot k^{|CP(Y)| - \frac{|CP(U_Y)|+|CP(V_Y)|}{2}}.$$

With the above definition, we observe that in the simplest case of $\tau = X \circ Y \circ Z$, we have

$$w(\tau)_{\mathrm{CP}} \leq w(X)_{\mathrm{CP}(X)} \cdot w(Y)_{\mathrm{CP}(Y)} \cdot w(Z)_{\mathrm{CP}(Z)}.$$

We point out that the color dependence on conn is simplified out for extension piece as they do not contain floating components by construction.

We further observe that the above definition extends to various levels of left/right extensions,

Proposition 8.6.4. *For $\tau = A_t \circ X_t \circ A_{t-1} \circ X_{t-1} \circ \cdots \circ A_1 \circ X_1 \circ Y \circ Z_1 \circ B_1 \circ \cdots \circ Z_{t'} \circ B_{t'}$*

$$w(\tau)_{CP} \leq \prod_i^t \left(w(A_i)_{CP(A_i)} \cdot w(X_i)_{CP(X_i)}\right) \cdot w(Y)_{CP(Y)} \cdot \prod_j^{t'} \left(w(Z_j)_{CP(Z_j)} \cdot w(B_j)_{CP(B_j)}\right)$$

where the inequality comes from assuming $CP(\tau)$ to be piecewise disjoint beyond the colors in the

mandatory boundary.

Proof. For the general setting, notice that each vertex $v \notin V(Y)$, it either appears to the left or the right of Y. Suppose it appears to the left, its contribution to norm bound comes from the smallest i such that the vertex appears in L_i for $L \in \{A, X\}$; and for the norm bound factor of each color, it is either picked up as a whole factor via some piece that contains it in the interior, or we split its factor to two adjacent piece if it appears on the boundary of any piece. □

Factorization of coefficient function into pieces We now define piece-function for $\lambda(\cdot)$ in the above fashion for the weight function. A priori, this is less immediate as the factor Ev may span across pieces. For example, in the simplest case of $\tau = X \circ Y \circ Z$, i.e. we have one level of rainbow extension via X and Z starting from the MVS of U_Y and V_Y. Since U_Y, V_Y are only guaranteed to be MVS instead of rainbow MVS, a floating component incident to U_Y, V_Y may have edges across two pieces. However, such dependence can be controlled once we observe each rainbow component cannot stretch "too far" beyond each rainbow-extension piece.

Claim 8.6.5. *Each rainbow component may have edges in at most two adjacent pieces beyond the base level. In other words, any rainbow component with edges in X_i (resp. Z_i), its edges is completely contained in both X_i and A_{i+1} (resp. Z_i and B_{i+1}) for $i \neq 0$. For the base level, we may have rainbow component with edges in both X_1, Y, Z_1 but there are no more pieces that contain its edges.*

Proof. For any rainbow component C with edges in X_i that additionally has edges in A_{i+1}, any rainbow edge incident to $U_{A_{i+1}}$ is contained in A_{i+1}, that said, C as a rainbow component cannot have edges to the left of $U_{A_{i+1}}$. The same argument applies for Z shape by flipping the boundary. □

Inspired by our approximate weight function, we also define its analog for $Ev(\tau)_{\text{CP}}$.

Definition 8.6.6 (Approximate Ev Coefficient)**.** *Define*

$$|Ev^{\approx}(\tau)_{CP}| \coloneqq \prod_{\substack{G_i \\ \textit{rainbow comp in}(\tau,CP)}} (\frac{1}{k})^{|V(G_i)|-1}$$

We show that $Ev^{\approx}(\cdot)$ approximates $Ev(\cdot)$ up to factor of $n^{o(1)}$ in lemma 8.11.2. Intuitively, this follows from the observation that any floating component C gives a factor $O(k)$ blow up, which we correct via distributing across rainbow colors, and there are $k \cdot (k-1) \cdot \cdots \cdot (k-|V(C)|)$ many such rainbow colorings, and we apply approximation on the falling factorials in our regime of $|V(C)| \ll k$.

As a corollary of the above, we define the proxy coefficient function for shape τ as the following.

Definition 8.6.7 (Proxy Coefficient Function for τ)**.** *For color-shape (τ, CP) , we define*

$$\lambda^{\approx}_{\tau,CP} \coloneqq \left(\frac{1}{k}\right)^{|V(\tau)|-\frac{|U_\tau|+|V_\tau|}{2}} \cdot |Ev^{\approx}(\tau)_{CP}|$$

Our proxy coefficient function extends to pieces naturally,

Definition 8.6.8 (Proxy coefficient function for pieces)**.** *We define coefficient function for each piece as the following,*

1. *For X (and analogously Z by taking shape-transpose) left/right rainbow extension piece , we define*

$$\lambda^{\approx}(X)_{CP} \coloneqq (\frac{1}{k})^{|V(X)|-\frac{|U_X|+|V_X|}{2}} \cdot Ev^{\approx}(X)_{CP(X)}$$

for

$$Ev^{\approx}(X)_{CP} \coloneqq \prod_{\substack{G_i \in \mathcal{G}(X,CP) \\ G_i \text{incident to} V_X \\ E(G_i) \neq \emptyset}} (\frac{1}{k})^{|V(G_i) \setminus V_X|} \cdot \prod_{\substack{G_i \in \mathcal{G}(X,CP) \\ G_i \text{not incident to} V_X \\ E(G_i) \neq \emptyset}} (\frac{1}{k})^{|V(G_i) \setminus V_X)|-1};$$

2. *For A (and analogously B) left/right MVS extension piece, we define*

$$\lambda^{\approx}(A)_{CP} \coloneqq (\frac{1}{k})^{|V(A)| - \frac{|U_A|+|V_A|}{2}} \cdot Ev^{\approx}(Y)_{CP}$$

for

$$Ev^{\approx}(Y)_{CP} \coloneqq \prod_{\substack{G_i \in \mathcal{G}(A,CP) \\ E(G_i) \neq \emptyset}} (\frac{1}{k})^{|V(G_i)|-1};$$

3. *For the base piece Y, we define*

$$\lambda^{\approx}(Y)_{CP} \coloneqq (\frac{1}{k})^{|V(Y)| - \frac{|U_Y|+|V_Y|}{2}} \cdot Ev^{\approx}(Y)_{CP}$$

for

$$Ev^{\approx}(Y)_{CP} \coloneqq \prod_{\substack{G_i \in \mathcal{G}(Y,\tau) \\ E(G_i) \neq \emptyset}} (\frac{1}{k})^{|V(G_i)|-1}.$$

With these proxy functions, we may then verify

$$\lambda^{\approx}(\tau)_{\mathrm{CP}} \leq \prod_{i=t}^{1} (\lambda^{\approx}(A_i)_{\mathrm{CP}(A_i)} \cdot \lambda^{\approx}(X_i)_{\mathrm{CP}(X_i)}) \cdot \lambda^{\approx}(Y)_{\mathrm{CP}} \cdot \prod_{j=t}^{1} (\lambda^{\approx}(Z_j)_{\mathrm{CP}(Z_j)} \cdot \lambda^{\approx}(B_j)_{\mathrm{CP}(B_j)})$$

To see this, observe the first factor holds identically to the usual three-way partition, while the focus shall be the Ev term.

Claim 8.6.9 (Decomposition of Rainbow Component into Pieces). *For $\tau = A_t \circ X_t \circ A_{t-1} \circ$*

$X_{t-1} \circ \cdots \circ A_1 \circ X_1 \circ Y \circ Z_1 \circ B_1 \circ \cdots \circ Z_{t'} \circ B_{t'}$

$$\sum_{\substack{G_i:\textit{rainbow comp. in } (\mathsf{conn}(\tau),CP)\\ E(G_i)\neq\emptyset}} (|V(G_i)|-1) \geq$$

$$\sum_{X_t} \sum_{\substack{G_i\\ \textit{rainbow comp. in } (X_t,CP(X_t))\\ E(G_i)\neq\emptyset\\ V(G_i)\cap V_{X_t}\neq\emptyset}} |V(G_i)\setminus V_{X_t}| + \sum_{\substack{G_i\\ \textit{rainbow comp. in } (X_t,CP(X_t))\\ E(G_i)\neq\emptyset\\ V(G_i)\cap V_{X_t}=\emptyset}} (|V(G_i)|-1) +$$

$$\sum_{Z_t} \sum_{\substack{G_i\\ \textit{rainbow comp. in } (Z_t,CP(Z_t))\\ E(G_i)\neq\emptyset\\ V(G_i)\cap U_{X_t}\neq\emptyset}} |V(G_i)\setminus V_{Z_t}| + \sum_{\substack{G_i\\ \textit{rainbow comp. in } (Z_t,CP(Z_t))\\ E(G_i)\neq\emptyset\\ V(G_i)\cap U_{Z_t}=\emptyset}} (|V(G_i)|-1) +$$

$$\sum_{A\in\{A_j,B_j,Y\}} \sum_{\substack{G_i\\ \textit{rainbow comp. in } A\\ E(G_i)\neq\emptyset}} (|V(G_i)|-1)$$

Proof. For starters, we remind the reader that each rainbow component by definition includes no isolated vertex after removal of pure-edge while we focus on rainbow component with edges for this section as isolated vertex trivially gives $Ev = 1$ here. We prove this by processing rainbow component in $\mathsf{conn}(\tau)$. The base case holds trivially with no rainbow component added.

Let C be a rainbow component in τ, by claim 8.6.5, there are at most two adjacent pieces containing its edges X_i and A_{i-1} suppose it appears to the left of Y if not intersecting Y, or it may have edges contained in X_1, Y, Z_1. We first observe that in the case of C having all its edges contained in a single piece of $\{A_j, B_j, Y\}$, the above is immediate as the LHS gets a factor of $|V(C)|-1$ and the RHS picks up the same factor from the piece that contains the component C. Analogously, this further holds for component C that appears in X_t (or Z_t piece) yet not incident to V_{X_t} (or U_{Z_t} respectively).

In the case C is incident to V_{X_t} for some X_t (for some $t > 1$), its vertices may be

decomposed into a collection of rainbow components in X_t and A_{t-1}. Observe that $V(C) \cap V(A_{t-1})$ is a collection of at least 1 single rainbow component, writing $RHS(C)$ as the gain from component C on the RHS, we pick up a factor of

$$RHS_A(C) \leq |V(C) \cap V(A_{t-1})| - 1 \leq |V(C) \cap V(A_{t-1})| - 1$$

from the edge factors in A_{t-1}. On the other hand, from X_t, we pick up a factor of

$$RHS_X(C) = |V(C) \cap V(X_t) \setminus V_{X_t}| = |V(C) \cap V(X_t)| - |V(C) \cap U_{A_{t-1}}|$$

where we use $V_{X_t} = U_{A_{t-1}}$. Summing the above gives us the gain on the RHS being

$$RHS(C) = RHS_A(C) + RHS_Y(C) \leq |V(C) \cap V(A_{t-1})| - 1 + |V(C) \cap V(X_t)| - |V(C) \cap U_{A_{t-1}}| = |V(C)|$$

where we use $|V(C) \cap V(A_{t-1})| + |V(C) \cap V(X_t)| = |V(C)| - |V(C) \cap U_{A_{t-1}}|$. And note that the LHS factor is exactly $|V(C)| - 1$ and this proves the case when C is a rainbow component that does not touch upon the base piece.

For C touching upon the base piece, the pieces of focus are now X_1, Y, Z_1. Notice $V(C) \cap V(Y)$ under CP(Y) is a collection of at least a single rainbow component, which gives us a factor of at most

$$RHS_Y(C) \leq |V(C) \cap V(Y)| - 1$$

from the piece Y. Combining with the factors in X_1, Z_1, we have a total of at most

$$RHS(C) \leq |V(C) \cap V(X_1) \setminus V_{X_1}| + |V(C) \cap V(Z_1) \setminus U_{Z_1}| + |V(C) \cap V(Y)| - 1$$

Since $|V(C) \cap V(X_1) \setminus V_{X_1}| + |V(C) \cap V(Z_1) \setminus U_{Z_1}| + |V(C) \cap V(Y)| = |V(C)|$, we can

bound the above again by $|V(C)| - 1$ as desired.

□

Finally, the above shows the Ev component of both sides holds as the desired inequality, and for the vertex-factor of $\frac{1}{k}$, notice that this component holds with the usual strategy of splitting the factor into two pieces across the boundaries of pieces,

1. Each vertex in $V(\tau) \setminus U_\tau \cup V_\tau$ comes with a factor of $\frac{1}{k}$: it either has its factor split across two adjacent pieces, in which case we pick up $(\frac{1}{\sqrt{k}})^2 = \frac{1}{k}$, or it is a vertex that appears in the interior of some piece, which is a whole factor of $\frac{1}{k}$;

2. Vertices in $U_\tau \Delta V_\tau$ comes with a factor of $\frac{1}{\sqrt{k}}$: it either starts from the left (or right) piece and gets popped out of the boundary ultimately, and picks up the corresponding factor $\frac{1}{\sqrt{k}}$ from the piece it gets popped out;

3. Vertices in $U_\tau \cap V_\tau$ remain dormant throughout the decomposition, and does not receive any factor.

Corollary 8.6.10. *We have*

$$\lambda^{\approx}(\tau)_{CP} \leq \prod_{i=t}^{1}(\lambda^{\approx}(A_i)_{CP(A_i)}| \cdot \lambda^{\approx}(X_i)_{CP(X_i)}) \cdot \lambda^{\approx}(Y)_{CP} \cdot \prod_{j=t}^{1}(\lambda^{\approx}(Z_j)_{CP(Z_j)} \cdot \lambda^{\approx}(B_j)_{CP(B_j)})$$

8.6.2 Charging for Base Middle Shape

Lemma 8.6.11 (Charging for Base Piece)**.** *For the base piece (Y, CP), i.e. when U_Y, V_Y are the MVS for Y,*

$$\lambda^{\approx}(Y)_{CP} \cdot w(Y)_{CP} \leq \mathit{slack}(Y, CP)$$

for

$$\mathit{slack}(Y, CP) := (\frac{\sqrt{n}}{k})^{|V(Y)| - \frac{|U_Y|+|V_Y|}{2}} \cdot (\frac{1}{k})^{\frac{1}{2}|\mathit{rainbow}(U_Y \cap V_Y)_{CP}|}.$$

and $rainbow(U_Y \cap V_Y)_{CP}$ *is the set of vertices incident to rainbow edges in* $E(U_Y \cap V_Y)$ *under CP.*

Again, let $\mathcal{G}(Y, \text{CP})$ be the collection of rainbow components in Y under CP after removal of pure-color edges, We first observe that $\mathcal{G}(Y, \text{CP})$ may contain rainbow-color edges in $U_Y \cap V_Y$, and it is convenient for us to consider $\mathcal{G}'(Y, \text{CP})$ the collection of rainbow components obtained after removing both pure-color edges and crucially, rainbow edges in $U_Y \cap V_Y$. The removal of edges allow us to reserve an extra decay factor for vertices in $U_\tau \cap V_\tau$ incident to internal rainbow edges,

Proposition 8.6.12 (Decay for Reservation for $rainbow(U_Y \cap V_Y)_{\text{CP}}$)**.** *Consider* $\mathcal{G}'(Y, CP)$ *obtained from* $\mathcal{G}$ *by additionally removing rainbow edges in* $U_Y \cap V_Y$ *under CP, we have*

$$Ev^{\approx}(Y, CP) = \prod_{G_i \in \mathcal{G}(Y,CP)} (\frac{1}{k})^{|V(G_i)|-1} \leq \prod_{G_i \in \mathcal{G}'(Y,CP)} (\frac{1}{k})^{|V(G_i)|-1} \cdot (\frac{1}{k})^{\frac{1}{2}|rainbow(U_Y \cap V_Y)_{CP}|}.$$

Next, we show that the decay from $\mathcal{G}'$ alone is sufficient to offset the CP factor in Y,

Proposition 8.6.13 (Lower Bounding Ev Decay)**.**

$$\sum_{G_i \in \mathcal{G}'(Y,CP)} (|V(G_i)| - 1) \geq |CP(Y)| - \frac{|CP(U_Y)| + |CP(V_Y)|}{2}$$

Proof to lemma 8.6.11. We are now ready to show the slack bound Y by unpacking the factors and using the above results,

$$\begin{aligned}
&\lambda^{\approx}(Y)_{\text{CP}} \cdot w(Y)_{\text{CP}} \\
&= \sqrt{n}^{|V(Y)| - \frac{|U_Y|+|V_Y|}{2}} \cdot k^{|\text{CP}(Y)| - \frac{|\text{CP}(U_Y)|+|\text{CP}(V_Y)|}{2}} \cdot (\frac{1}{k})^{|V(Y)| - \frac{|U_Y|+|V_Y|}{2}} \cdot Ev^{\approx}(Y)_{\text{CP}} \\
&\leq \underbrace{\sqrt{n}^{|V(Y)| - \frac{|U_Y|+|V_Y|}{2}} \cdot (\frac{1}{k})^{|V(Y)| - \frac{|U_Y|+|V_Y|}{2}} \cdot (\frac{1}{k})^{\frac{1}{2}|rainbow(U_Y \cap V_Y)_{\text{CP}}|}}_{=\text{slack}(\cdot)} \cdot .
\end{aligned}$$

$$\underbrace{k^{|\mathrm{CP}(Y)|-\frac{|\mathrm{CP}(U_Y)|+|\mathrm{CP}(V_Y)|}{2}}\cdot\prod_{G_i\in\mathcal{G}'(Y,\mathrm{CP})}(\frac{1}{k})^{|V(G_i)|-1}}_{\leq 1 \text{via proposition 8.6.13}}$$

$$\leq \mathrm{slack}(Y,\mathrm{CP})$$

where we use proposition 8.6.12 in the first inequality, and proposition 8.6.13 in the second one. We complete the charing for base piece Y by proving the above propositions.

□

Proof to proposition 8.6.12. Oserve that each rainbow component has a path to U_τ and V_τ, hence we may consider a traversal from U_τ to assign the $\frac{1}{k}$ factors: in particular, each vertex reached some rainbow edge within the rainbow component receives a $\frac{1}{k}$ decay. We then observe the vertices outside $U_\tau \cap V_\tau$ receive the factor on both sides in $\mathcal{G}$ and $\mathcal{G}'$. For vertices in $U_\tau \cap V_\tau$, $\mathcal{G}'$ does not assign any vertex decay for vertices, while for rainbow component C within $U_\tau \cap V_\tau$, $\mathcal{G}$ contains an extra decay of $|V(C)|-1 \geq \frac{|V(C)|}{2}$. Summing over all rainbow components within $U_\tau \cap V_\tau$ yields the desired.

□

Proof to proposition 8.6.13. We prove the proposition by showing

$$\sum_{G_i\in\mathcal{G}(Y,\mathrm{CP})}(|V(G_i)|-1) \geq |\mathrm{CP}(Y)|-|\mathrm{CP}(U_Y)| = |\mathrm{CP}(Y)\setminus\mathrm{CP}(U_Y)|\,,$$

and similarly

$$\sum_{G_i\in\mathcal{G}(Y,\mathrm{CP})}(|V(G_i)|-1) \geq |\mathrm{CP}(Y)|-|\mathrm{CP}(V_Y)| = |\mathrm{CP}(Y)\setminus\mathrm{CP}(V_Y)|\,.$$

The main proposition then follows by taking the average of the above two lines. We now focus on the inequality restricted on U_Y. Unpacking the definition, we remind the reader

that $|\mathrm{CP}(Y) \setminus \mathrm{CP}(U_Y)|$ is simply the number of colors that appear in Y but not in U_Y. Consider a traversal from U_Y, we bound the contribution to the RHS, a.k.a., the new color explored, in each rainbow component. In particular, we show that we can assign a factor of 1 on the LHS for each vertex outside S_L that takes a new color in the traversal.

We now recall that for Y, this may be decomposed into a collection of rainbow components in $\mathcal{G}$ connected via pure-color edge to U_Y (and equivalently V_Y) since there is no floating component in τ. For any rainbow-component C incident to U_Y, any such component contributes at most $V(C) - 1$ new colors to the RHS, while we pick up a factor of $|V(C)| - 1$. For each rainbow component C not incident to U_Y, it must be reached via some pure-color edge, that said, it contributes a factor of $|V(C)| - 1$ factor to the LHS via its coefficient, and again it contributes at most $|V(C)| - 1$ colors. Summing over rainbow components in $\mathcal{G}(Y, \mathrm{CP})$ proves the desired inequality. □

8.6.3 Charging for Rainbow-Extension Shape

Lemma 8.6.14. *For left-rainbow-extension X (and similarly right-rainbow-extension Z),*

$$\lambda^{\approx}(X)_{CP} \cdot w(X)_{CP} \leq \mathit{slack}(X, CP)$$

for

$$\mathit{slack}(X, CP) \coloneqq \left(\frac{\sqrt{n}}{k}\right)^{|V(X) \setminus V_X|}$$

Proof. We start by recalling the structure property of X that will be useful.

1. V_X is a MVS of X (even unique);

2. Any vertex in U_X is is the same rainbow component as some vertex in V_X, i.e., connected to V_X via a rainbow-path .

Unpacking the factors, we have

$$\begin{aligned}
\lambda^{\approx}(X)_{\mathrm{CP}} \cdot w(X)_{\mathrm{CP}} &= (\frac{1}{k})^{|V(X)|-\frac{|U_X|+|V_X|}{2}} \cdot Ev^{\approx}(X)_{\mathrm{CP}} \cdot \sqrt{n}^{|V(X)\setminus V_X|} \cdot k^{|\mathrm{CP}(X)|-\frac{|\mathrm{CP}(U_X)|+|\mathrm{CP}(V_X)|}{2}} \\
&= (\frac{1}{k})^{|V(X)|-\frac{|U_X|+|V_X|}{2}} \cdot \sqrt{n}^{|V(X)\setminus V_X|} \cdot Ev^{\approx}(X)_{\mathrm{CP}} \cdot k^{|\mathrm{CP}(X)|-\frac{|\mathrm{CP}(U_X)|+|\mathrm{CP}(V_X)|}{2}} \\
&\leq \underbrace{\left(\frac{\sqrt{n}}{k}\right)^{|V(X)\setminus V_X|}}_{\mathrm{slack}(\cdot)} \cdot k^{\frac{|U_X|-|V_X|}{2}} \cdot k^{|\mathrm{CP}(X)|-\frac{|\mathrm{CP}(U_X)|+|\mathrm{CP}(V_X)|}{2}} \cdot Ev^{\approx}(X)_{\mathrm{CP}} \\
&\leq \mathrm{slack}(X, \mathrm{CP})
\end{aligned}$$

where in the first inequality we rewrite

$$\begin{aligned}
(\frac{1}{k})^{|V(X)|-\frac{|U_X|+|V_X|}{2}} \cdot \sqrt{n}^{|V(X)\setminus V_X|} &= (\frac{1}{k})^{|V(X)|-|V_X|} \cdot k^{\frac{|U_X|-|V_X|}{2}} \cdot \sqrt{n}^{|V(X)\setminus V_X|} \\
&= k^{\frac{|U_X|-|V_X|}{2}} \cdot (\frac{\sqrt{n}}{k})^{|V(X)\setminus V_X|}
\end{aligned}$$

and the last inequality holds by the following proposition by considering the logarithm with base k. □

Proposition 8.6.15.

$$|CP(X)| - \frac{|CP(U_X)| + |CP(V_X)|}{2} + \frac{|U_X| - |V_X|}{2} \leq \sum_{\substack{G_i \in \mathcal{G}(X,CP) \\ \textit{incident to} V_X}} |V(G_i) \setminus V_X| + \sum_{\substack{G_i \in \mathcal{G}(X,CP) \\ \textit{not incident to} V_X}} (|V(G_i)| - 1)$$

Proof. Let $|V(\mathcal{I})|$ be the number of vertices in rainbow component incident to V_X. We drop the dependence of $G_i \in \mathcal{G}(X, \mathrm{CP})$ as it is clear throughout. Rearranging the RHS, we have

$$\sum_{\substack{G_i \\ \text{incident to} V_X}} |V(G_i) \setminus V_X| + \sum_{\substack{G_i \\ \text{not incident to} V_X}} (|V(G_i)| - 1) = |V(\mathcal{I})| - |V_X| + \sum_{\substack{G_i \\ \text{not incident to} V_X}} (|V(G_i) \cap V(X)| -$$

Therefore, it suffices to show

$$|\mathrm{CP}(X)| - \frac{|\mathrm{CP}(U_X)| + |\mathrm{CP}(V_X)|}{2} \leq |V(\mathcal{I})| - \frac{|U_X| + |V_X|}{2} + \sum_{\substack{G_i \\ \text{not incident to} V_X}} (|V(G_i)| - 1)$$

Let $\mathcal{D}$ be the collection of rainbow components not connected to V_X, and write

$$|\mathrm{CP}(X)| = |\mathrm{CP}(\mathcal{I})| + |\mathrm{CP}(\mathcal{D}) \setminus \mathrm{CP}(\mathcal{I})|.$$

The above equation then follows by the subsequent two claims which establish

$$|\mathrm{CP}(\mathcal{I})| - \frac{|\mathrm{CP}(U_X)| + |\mathrm{CP}(V_X)|}{2} \leq |V(\mathcal{I})| - \frac{|U_X| + |V_X|}{2},$$

and

$$|\mathrm{CP}(\mathcal{D}) \setminus \mathrm{CP}(\mathcal{I})| \leq \sum_{\substack{G_i \\ \text{not incident to} V_X}} (|V(G_i)| - 1)$$

□

Claim 8.6.16.

$$|CP(\mathcal{I})| - \frac{|CP(U_X)| + |CP(V_X)|}{2} \leq |V(\mathcal{I})| - \frac{|U_X| + |V_X|}{2}$$

Proof. Partition the vertices according to their color, for each group of vertices with color c, we count the contribution of this group to both sides of the equation. Process the vertices of color c in the following order,

1. If some vertex v in with color c appears in both of U, V boundary, the group contributes 0 to the LHS, and we pick up at least 0 on the RHS via this vertex;

2. Otherwise, if some vertex v in c appears in one of the U, V boundaries but not both,

we get a factor of $\frac{1}{2}$ on the LHS, and via this vertex v's contribution to the RHS, we pick up a factor of $\frac{1}{2}$.

3. If no vertex of color c appears in U, V boundary while some vertex v takes color c outside the boundary, we get a $+1$ from the LHS, and again a $+1$ on the RHS.

4. Any extra vertex with color c can only bring non-negative contribution to the RHS.

Processing all color c in $\mathcal{I}$ yields the desired. □

Claim 8.6.17.

$$|CP(\mathcal{D}) \setminus CP(\mathcal{I})| \leq \sum_{\substack{G_i \\ \text{not incident to} V_X}} (|V(G_i)| - 1)$$

Proof. We first observe that $\mathcal{D}$ is connected to $\mathcal{I}$ by some pure-color edge, hence we can consider a traversal to bound the CP factor of each rainbow component in $\mathcal{D}$: each rainbow component is reached via a pure-color edge, and contributes at most $|V(G_i)| - 1$ unexplored colors. □

8.6.4 Charging for MVS Extension Shape

Lemma 8.6.18. *For left-MVS-extension A (and similarly right-MVS-extension B) with color-partition CP,*

$$\lambda^{\approx}(A)_{CP} \cdot w(A)_{CP} \leq slack(A, CP)$$

for

$$slack(A, CP) := (\frac{\sqrt{n}}{k})^{|V(A) \setminus V_A|}.$$

Proof. Unpacking the factors, we have

$$w(A)_{\text{CP}} = \sqrt{n}^{|V(A) \setminus V_A|} \cdot k^{|\text{CP}(A)| - \frac{|\text{CP}(U_A)| + |\text{CP}(V_A)|}{2}},$$

and

$$\lambda(A)_{\text{CP}} = (\frac{1}{k})^{|V(A)|-\frac{|U_A|+|V_A|}{2}} \cdot \prod_{\substack{G_i \\ \text{rainbow comp. in}(A,\text{CP})}} (\frac{1}{k})^{|V(G_i)\cap V(A)|-1}$$

The crux of our argument is in observing again

$$|\text{CP}(A)| - |\text{CP}(U_A)| = |\text{CP}(A) \setminus \text{CP}(U_A)| \leq \sum_{G_i:\text{rainbow comp. in}(A,\text{CP})} |V(G_i)| - 1$$

and also

$$|\text{CP}(A)| - |\text{CP}(V_A)| = |\text{CP}(A) \setminus \text{CP}(V_A)| \leq \sum_{G_i:\text{rainbow comp. in}(A,\text{CP})} (|V(G_i)| - 1)$$

We now prove each line. For the first line, consider a traversal from U_A, and process each rainbow component reachable either via a pure-edge to a known vertex or via an incidental vertex in U_A, each such component C gives at most $|V(C_i)| - 1$ new colors to the LHS, while it also contributes the same factor to the RHS. Since each rainbow component will be explored by the end of the process, i.e. no floating component, this completes the proof to the first line. The second inequality follows identically by considering a traversal from V instead. Taking the average gives us

$$|\text{CP}(A)| - \frac{|\text{CP}(U_A)| + |\text{CP}(V_A)|}{2} \leq \sum_{G_i\text{rainbow comp. in}(A,\text{CP})} |V(G_i)| - 1\,.$$

On the other hand, for the factor of $\sqrt{n}$, notice we have $|U_A| < |V_A|$ as U_A is the unique MVS of A, therefore

$$\sqrt{n}^{|V(A)\setminus V_A|} \cdot (\frac{1}{k})^{|V(A)|-\frac{|U_A|+|V_A|}{2}} \leq \sqrt{n}^{|V(A)\setminus V_A|} \cdot (\frac{1}{k})^{|V(A)|-|V_A|} = (\frac{\sqrt{n}}{k})^{|V(A)\setminus V_A|}$$

Combining the above we have

$$\lambda^{\approx}(A)_{\mathrm{CP}} \cdot w(A)_{\mathrm{CP}} = \sqrt{n}^{|V(A)\setminus V_A|} \cdot k^{|\mathrm{CP}(A)|-\frac{|\mathrm{CP}(U_A)|+|\mathrm{CP}(V_A)|}{2}} \cdot (\frac{1}{k})^{|V(A)|-\frac{|U_A|+|V_A|}{2}} \cdot \prod_{\substack{G_i \\ \text{rainbow comp. in}(A,\mathrm{CP})}} (\frac{1}{k})^{|V(}$$

$$= \underbrace{\sqrt{n}^{|V(A)\setminus V_A|}(\frac{1}{k})^{|V(A)|-\frac{|U_A|+|V_A|}{2}}}_{\text{slack}(\cdot)} \cdot \underbrace{k^{|\mathrm{CP}(A)|-\frac{|\mathrm{CP}(U_A)|+|\mathrm{CP}(V_A)|}{2}} \cdot \prod_{\substack{G_i \\ \text{rainbow comp. in}(A,\mathrm{CP})}} (\frac{1}{k})^{|V(G_i}}_{\leq 1}$$

$$\leq \text{slack}(A)_{\mathrm{CP}}\,.$$

□

8.6.5 Wrapping Up

Proof to lemma 8.6.2. We now prove the main lemma of the section. Using the approximate weight bound and coefficient bounds, via the bound on each piece lemma 8.6.11, lemma 8.6.14, and lemma 8.6.18, we have the following corollary,

Corollary 8.6.19 (Final approximate slack for proper middle shape)**.**

$$\lambda^{\approx}(\tau)_{CP} \cdot w(\tau)_{CP} \leq \prod_i slack(A_i) \cdot slack(X_i) \cdot slack(Y) \cdot \prod_j slack(Y_j) \cdot slack(B_j)$$

$$\leq (\frac{\sqrt{n}}{k})^{|V(\tau)|-\frac{|U_\tau|+|V_\tau|}{2}} \left(\frac{1}{k}\right)^{\frac{1}{2}\cdot|rainbow(U_\tau \cap V_\tau)|}$$

Finally, we move from approximate slack to the actual bound by putting back the subpolynomial factors ignored due to our approximation for Ev and norm bounds, which is in in the end

$$\begin{aligned}\|\lambda(\tau)_{\text{CP}} \cdot \mathsf{M}_{\tau,\text{CP}}\| &\leq \lambda^{\approx}(\tau)_{\text{CP}} \cdot w(\tau)_{\text{CP}} \cdot |\frac{\lambda_\tau(\text{CP})}{\lambda^{\approx}_\tau(\text{CP})}| \cdot \frac{\|\mathsf{M}_{\tau,\text{CP}}\|}{w(\tau)_{\text{CP}}} \\ &\leq \left(\frac{\sqrt{n}}{k}\right)^{\frac{1}{2}|V(\tau)\setminus U_\tau \cap V_\tau|} \cdot \left(\frac{1}{k}\right)^{\frac{1}{2}\cdot|rainbow(U_\tau \cap V_\tau)|} \cdot O(|rainbow(E(\tau))_{\text{CP}}|)^{|rainbow(E(\tau))_{\text{CP}}|} \\ &\cdot O(|V(\tau) \setminus U_\tau \cap V_\tau|)^{2|V(\tau)\setminus U_\tau \cap V_\tau|)}\end{aligned}$$

where we plug in the bound from lemma 8.11.2 and remind the reader that $rainbow(E(\tau))_{\text{CP}}$ is the set of rainbow edges in $E(\tau)$ under CP. □

8.7 Combinatorial Charging Analysis for Intersection Shapes

8.7.1 Analysis Set-up

We now introduce some lemmas to set up for our charging analysis, and our starting point is a piecewise decomposition for γ and γ' shape in our three-way intersection.

Piecewise Decomposition for γ-Shape For our analysis, identical to our decomposition of middle shape into various pieces via left/rightwards extensions, we apply the same decomposition for γ shape.

Proposition 8.7.1. *For each (color) left-intersection γ shape, we can decompose it as*

$$\gamma = A_t \circ X_t \circ \cdots \circ A_2 \circ X_2 \circ A_1 \circ X_1 \circ Y$$

where A_i is a left-rainbow-extension shape, and X_i is a left-MVS-extension shape, and Y is a generic γ-shape such that

1. *U_Y is an MVS separating U_Y and $Int(Y) \cup V_Y$;*

2. *Y is an left shape (i.e. V_Y is the unique MVS).*

Proof. This is analogous to the decomposition of middle shape into pieces, and the base shape γ_{base} satisfies the generic property as it is starting from the leftmost MVS separating U_σ and $V_\sigma \cup \text{Int}(\sigma) = V_\gamma \cup \text{Int}(\gamma)$. □

Next, we recall define the weight function and coefficient function extension pieces defined as the following (see definition 8.6.8).

For $\gamma \circ \tau \circ \gamma'$ a tuple of color-shapes that intersect to τ_P, let U_Y be the leftmost MVS separating U_γ and $V_\gamma \cup \text{Int}(\gamma)$, and $V_{Y'}$ be the rightmost MVS separating $V_{\gamma'}$ and $U_{\gamma'} \cup \text{Int}(\gamma')$, i.e. these are the original separator in the decomposition process before any left/rightwards extension. Notice the component between U_γ and U_Y, between $V_{\gamma'}$ and V_Y are untouched by the intersection, that said, we can factor these components out by finding L_γ (and respectively $R_{\gamma'}$) be the subset of edges in $E(\gamma)$ (and $E(\gamma')$) reachable from U_γ (and $V_{\gamma'}$) without passing through U_Y and $V_{Y'}$. This allows us to define $\widetilde{\tau}_P$ as the shape obtained from τ_P yet with the same intersection pattern,

1. $\widetilde{\tau}_P$ is obtained from intersecting vertices in $Y \circ \tau \circ Y'$ according to the intersection pattern in τ_P;

2. $U_{\widetilde{\tau}_P} = U_Y$, and $V_{\widetilde{\tau}_P} = V_{Y'}$.

In other words, this is the usual intersection shape before any left/rightwards extensions in γ and γ'.

Proposition 8.7.2. *Let $\gamma = A_t \circ X_t \circ \dots A_1 \circ X_1 \circ Y$, and $\gamma' = Y' \circ Z_1 \circ B_1 \circ \cdots \circ Z_{t'} \circ B_{t'}$, for $\gamma \circ \tau \circ \gamma'$ a tuple of color-shapes that intersect to τ_P, let $\widetilde{\tau}_P$ be the result of intersection in $Y \circ \tau \circ Y'$ according to τ_P, we have*

$$w(\tau_P)_{CP} = \prod_i \left(w(A_i) \cdot w(X_i)\right) \cdot w(\widetilde{\tau}_P) \cdot \prod_j \left(w(B_j) \cdot w(Z_j)\right)$$

and

$$\lambda^{\approx}(\tau_P)_{CP} \leq \prod_i (\lambda^{\approx}(A_i) \cdot \lambda^{\approx}(X_i)) \cdot \lambda^{\approx}(\widetilde{\tau}_P) \cdot \prod_j \left(\lambda^{\approx}(B_j) \cdot \lambda^{\approx}(Z_j)\right)$$

for

$$\lambda^{\approx}(\widetilde{\tau}_P) \coloneqq \lambda^{\approx}(Y \circ \tau \circ Y') = \lambda^{\approx}(Y)_{CP} \cdot \lambda^{\approx}(\tau)_{CP} \cdot \lambda^{\approx}(Y')_{CP}$$

and

$$w(\widetilde{\tau}_P)_{CP} = \max_{S:\textit{separator for } \widetilde{\tau_P}} \sqrt{n}^{|V(\widetilde{\tau}_P)|-|S|+I(\widetilde{\tau}_P)} \cdot k^{|CP(Y\circ\tau\circ Y')|}$$

Proof. The coefficient bound is identical to our piecewise decomposition in middle shape, and we apply approximate factorization for Ev of each rainbow component, hence the approximate coefficient. For the norm bound, the factor k from CP follows identically as the coefficient bound; to see the factor n, rewrite

$$|V(\tau_P)| = |V(A_t \circ \cdots \circ X_1) \setminus V_{X_1}| + |V(\widetilde{\tau}_P)| + |V(Z_1 \circ \cdots \circ B_{t'}) \setminus U_{Z_1}|,$$

and notice $I(\tau_P) = I(\widetilde{\tau}_P)$ as there are no intersection in $A_t \circ \cdots \circ X_1)$ and in $Z_1 \circ \cdots \circ B_{t'}$, hence there is no isolated vertex. Moreover, since each extension piece is a left/right shape, observe that we have $\min_{S:\text{separator for } \tau_P} |S| = \min_{S:\text{separator for } \widetilde{\tau}_P} |S|$. This allows us to write

$$\begin{aligned}
&|V(\tau_P)| - \min_{S:\text{separator for } \tau_P} |S| + |I(\tau_P)| \\
&= |V(A_t \circ \cdots \circ X_1) \setminus V_{X_1}| + |V(Z_1 \circ \cdots \circ B_{t'}) \setminus U_{Z_1}| + |V(\widetilde{\tau}_P)| - \min_{S:\text{separator for} \widetilde{\tau}_P} |S| + |I(\widetilde{\tau}_P)|
\end{aligned}$$

and notice the first two terms are exactly the $\sqrt{n}$ factor from the weight function of the extension pieces, and remaining term is exactly the $\sqrt{n}$ factor from $w(\tau_P)$.

Finally, rearranging the factor of k in $w(\widetilde{\tau_P})$, it can be equivalently written as

$$\begin{aligned}
w(\widetilde{\tau_P}) &= \max_{S\text{:separator for } \widetilde{\tau_P}} \sqrt{n}^{|V(\widetilde{\tau_P})|-|S|+I(\widetilde{\tau_P})} \cdot k^{|\mathrm{CP}(Y\circ\tau\circ Y')|} \\
&= \max_{S\text{:separator for } \widetilde{\tau_P}} \sqrt{n}^{|V(\widetilde{\tau_P})|-|S|+I(\widetilde{\tau_P})} \\
&\cdot k^{|\mathrm{CP}(Y)|-\frac{|\mathrm{CP}(U_Y)|+|\mathrm{CP}(V_Y)|}{2}} \cdot k^{|\mathrm{CP}(\tau)|-\frac{|\mathrm{CP}(U_\tau)|+|\mathrm{CP}(V_\tau)|}{2}} \cdot k^{|\mathrm{CP}(Y')|-\frac{|\mathrm{CP}(U_{Y'})|+|\mathrm{CP}(V_{Y'})|}{2}} \\
&= \max_{S\text{:separator for } \widetilde{\tau_P}} \sqrt{n}^{|V(\widetilde{\tau_P})|-|S|+I(\widetilde{\tau_P})} \\
&\cdot k^{|\mathrm{CP}(Y)|-\frac{|\mathrm{CP}(U_Y)|+|\mathrm{CP}(V_Y)|}{2}} \cdot k^{|\mathrm{CP}(\tau)|-\frac{|\mathrm{CP}(U_\tau)|+|\mathrm{CP}(V_\tau)|}{2}} \cdot k^{|\mathrm{CP}(Y')|-\frac{|\mathrm{CP}(U_{Y'})|+|\mathrm{CP}(V_{Y'})|}{2}}
\end{aligned}$$

□

8.7.2 Combinatorial Analysis for Intersection Shapes

We now introduce the main lemma of our combinatorial charging.

Lemma 8.7.3 (Slack for Intersection Shape)**.** *For a tuple of color-shapes* $(\gamma_t, \ldots, \gamma_2, \gamma_1, \tau, \gamma'_1, \gamma'_2, \ldots, \gamma'_t)$ *that intersect to* τ_P*, we have*

$$\|\lambda(\tau_P)_{CP(\tau_P)} \cdot \mathsf{M}_{(\tau_P,CP)}\| = o(1)$$

Formally, we have

$$\|\lambda(\tau_P)_{CP(\tau_P)} \cdot \mathsf{M}_{(\tau_P,CP)}\| \leq \mathit{slack}(\tau_P)_{CP}$$

for

$$\mathit{slack}(\tau_P)_{CP} := (\frac{\sqrt{n}}{k})^{\sum_i |V(\gamma_i)-\frac{|U_{\gamma_i}|+|V_{\gamma_i}|}{2}+|V(\tau_0)|-\frac{|U_{\tau_0}|+|V_{\tau_0}|}{2}+\sum_i |V(\gamma'_i)-\frac{|U_{\gamma'_i}|+|V_{\gamma'_i}|}{2}} \cdot \left(\frac{1}{k}\right)^{\frac{1}{2}\cdot|\mathit{rainbow}(\mathit{dormant}(\tau_P))|}$$

$$\cdot O(|\mathit{rainbow}(E(\tau))_{CP}|)^{|\mathit{rainbow}(E(\tau))_{CP}|} \cdot O(|V(\tau) \setminus U_\tau \cap V_\tau|)^{2|V(\tau)\setminus U_\tau \cap V_\tau|}$$

Towards this end, we consider the intersection-shape $\widetilde{\tau}_P$ obtained before any left/right extension in γ, γ', and the technical bulk of this section is in showing the following lemma that states $\widetilde{\tau}_P$ can be charged, if we get to "pretend" that the rainbow-component value factorizes.

Lemma 8.7.4. *For a tuple of color-shapes (γ, τ, γ') that intersect to τ_P with $\gamma = A_t \circ X_t \circ \dots A_1 \circ X_1 \circ Y$, and $\gamma' = Y' \circ Z_1 \circ B_1 \circ \dots \circ Z_{t'} \circ B_{t'}$, let $\widetilde{\tau}_P$ be the intersection result in $Y \circ \tau \circ Y'$,*

$$\lambda^{\approx}(\widetilde{\tau}_P)_{CP(\widetilde{\tau}_P)} \cdot w(\widetilde{\tau}_P) \leq slack(\widetilde{\tau}_P, CP(\widetilde{\tau}_P)) \cdot \lambda^{\approx}(\tau)_{CP(\tau)} \cdot w^{\approx}(\tau)$$

for

$$slack(\widetilde{\tau}_P, CP(\widetilde{\tau}_P)) := (\frac{\sqrt{n}}{k})^{|V(Y)| - \frac{|U_{Y'}|+|V_{Y'}|}{2}} \cdot (\frac{\sqrt{n}}{k})^{|V(Y)| - \frac{|V_{Y'}|+|U_{Y'}|}{2}}$$

In other words, the above lemma shows that not only can we charge to the corresponding diagonal in the polynomial factor, we in fact obtain an extra slack for each vertex introduced in Y, Y' provided they are not in $U_Y \cap V_Y, U_{Y'} \cap V_{Y'}$.

With the above lemma in hand, we then incorporate the factorization issue by left/rightwards rainbow/MVS extension, whose combinatorial weight are controlled identically to the analysis for middle shape. This then allows to prove the main lemma assuming lemma 8.7.4.

Proof to lemma 8.7.3. From our set-up of piecewise weight and coefficient function (proposition 8.7.2), for each level of intersection of $i \in [t]$ that intersects into τ_i, $(\gamma_i, \tau_{i-1}, {\gamma'}_i)$, we may further decompose $\gamma_i = A_{i,t'} \circ X_{i,t'} \circ \dots A_{t,1} \circ X_i 1 \circ Y_i$, and $\gamma' = Y'_i \circ Z_{i,1} \circ B_{i,1} \circ \dots \circ Z_{i,t''} \circ B_{i,t'}$. For each level i, let $\widetilde{\tau}_i$ denote the intersection with extension part in $\gamma_i, {\gamma'}_i$ factored out as usual, ignoring the dependence on CP as it is clear below, we have

$$\begin{aligned}\frac{\lambda^{\approx}(\gamma_i \circ \tau_{i-1} \circ {\gamma'}_i) \cdot w(\tau_i)}{\lambda^{\approx}(\tau_{i-1}) \cdot w(\tau_{i-1})} \cdot &\leq \frac{\prod_{t'} (\lambda^{\approx}(A_{i,t}) \cdot \lambda^{\approx}(X_{i,t})) \cdot \lambda^{\approx}(\widetilde{\tau}_i) \cdot \prod_{t''} (\lambda(B_{i,t''}) \cdot \lambda(Z_{i,t''})) \cdot w(\tau_i)}{\lambda^{\approx}(\tau_{i-1}) \cdot w(\tau_{i-1})} \\ &= \frac{\lambda^{\approx}(\widetilde{\tau}_i) \cdot w(\widetilde{\tau}_i)}{\lambda^{\approx}(\tau_{i-1}) \cdot w(\widetilde{\tau}_{i-1})} \cdot \prod_{E_i : \text{extension-pieces of the level for} \gamma_i, {\gamma'}_i} \lambda^{\approx}(E_i) \cdot w(E_i)\end{aligned}$$

$$\leq (\frac{\sqrt{n}}{k})^{|V(\gamma_i)|-\frac{|U_{\gamma_i}|+|V_{\gamma_i}|}{2}} \cdot (\frac{\sqrt{n}}{k})^{|V(\gamma'_i)|-\frac{|U_{\gamma'_i}|+|V_{\gamma'_i}|}{2}}$$

where we use the bound from lemma 8.6.14 for rainbow-extension pieces and lemma 8.6.18 for MVS-extension pieces. Taking the product across all intersection levels gives us

$$\lambda(\tau_P) \cdot w(\tau_P) \leq (\frac{\sqrt{n}}{k})^{\sum_i |V(\gamma_i)|-\frac{|U_{\gamma_i}|+|V_{\gamma_i}|}{2}} \cdot (\frac{\sqrt{n}}{k})^{\sum_i |V(\gamma'_i)|-\frac{|U_{\gamma'_i}|+|V_{\gamma'_i}|}{2}} \cdot \lambda^{\approx}(\tau_0) \cdot w(\tau_0)$$

Since for the base level of intersection τ_0 is simply an non-intersection shape captured by our bound in the previous section, plugging in corollary 8.6.19 gives us

$$\lambda^{\approx}(\tau_P) \cdot w(\tau_P) \leq (\frac{\sqrt{n}}{k})^{\sum_i |V(\gamma_i)-\frac{|U_{\gamma_i}|+|V_{\gamma_i}|}{2}+|V(\tau_0)|-\frac{|U_{\tau_0}|+|V_{\tau_0}|}{2}+\sum_i |V(\gamma'_i)-\frac{|U_{\gamma'_i}|+|V_{\gamma'_i}|}{2}} \cdot \left(\frac{1}{k}\right)^{\frac{1}{2}\cdot|rainbow(dormant(\tau_P))|}$$

where we let $rainbow(dormant(\tau_P))$ be the set of vertices in $U_X \cap V_X$ for all $X \in \{\gamma_i, \tau_0, \gamma'_i\}_{i\in[t]}$ that are incident to internal rainbow edges, and note that this factor comes from the analogous factor in the base piece of τ_0.

Putting back the approximate slack from our coefficient and norm bound gives us

$$\begin{aligned}
&\|\lambda(\tau_P)_{\text{CP}} \cdot \mathsf{M}_{\tau_P,\text{CP}}\| \\
&\leq \lambda^{\approx}(\tau_P)_{\text{CP}} \cdot w(\tau_P)_{\text{CP}} \cdot \frac{|\lambda(\tau_P)_{\text{CP}}|}{\lambda^{\approx}(\tau_P)} \cdot \frac{\|\mathsf{M}_{\tau_P,\text{CP}}\|}{w(\tau_P)_{\text{CP}}} \\
&\leq (\frac{\sqrt{n}}{k})^{\sum_i |V(\gamma_i)-\frac{|U_{\gamma_i}|+|V_{\gamma_i}|}{2}+|V(\tau_0)|-\frac{|U_{\tau_0}|+|V_{\tau_0}|}{2}+\sum_i |V(\gamma'_i)-\frac{|U_{\gamma'_i}|+|V_{\gamma'_i}|}{2}} \cdot \left(\frac{1}{k}\right)^{\frac{1}{2}\cdot|rainbow(dormant(\tau_P))|} \\
&\cdot O(|rainbow(E(\tau))_{\text{CP}}|)^{|rainbow(E(\tau))_{\text{CP}}|} \cdot O(|V(\tau) \setminus U_\tau \cap V_\tau|)^{2|V(\tau)\setminus U_\tau \cap V_\tau|)}
\end{aligned}$$

□

We now proceed to prove our main technical lemma. For starters, we recall the intersection tradeoff lemma from prior works,

Proposition 8.7.5 (Intersection Trade-off Lemma (as usual))**.** *For a three-way intersection pattern (Y, τ, Y') that intersects to $\widetilde{\tau}_P$, let $S(\widetilde{\tau}_P)$ be an MVS for $\widetilde{\tau}_P$, and $S(\tau)$ an MVS for τ,*

$$|V(\widetilde{\tau}_P)| + |I(\widetilde{\tau}_P)| - |S(\widetilde{\tau}_P)| \leq |V(\tau)| + |I(\tau)| - |S(\tau)| + |V(Y) \setminus U_Y| + |V(Y') \setminus V_{Y'}|$$

Proof to lemma 8.7.4. For convenience, we restate the lemma inequality

$$\lambda^{\approx}(\widetilde{\tau}_P)_{\mathrm{CP}} \cdot w(\widetilde{\tau}_P)_{\mathrm{CP}} \leq \lambda^{\approx}(\tau)_{\mathrm{CP}} \cdot w(\tau)_{\mathrm{CP}} \cdot \mathrm{slack}(\widetilde{\tau}_P, \mathrm{CP}(\widetilde{\tau}_P))\,.$$

Unpacking its factors, on the LHS, we have

$$\begin{aligned}
&\lambda^{\approx}(\widetilde{\tau}_P)_{\mathrm{CP}} \cdot w(\widetilde{\tau}_P)_{\mathrm{CP}} \\
&= \lambda^{\approx}(Y)_{\mathrm{CP}} \cdot \lambda^{\approx}(\tau)_{\mathrm{CP}} \cdot \lambda^{\approx}(Y')_{\mathrm{CP}} \cdot w(\widetilde{\tau}_P)_{\mathrm{CP}} \\
&= (\frac{1}{k})^{|V(Y \circ \tau \circ Y')| - \frac{|U_Y| + |V_{Y'}|}{2}} \cdot Ev^{\approx}(Y \circ \tau \circ Y')_{\mathrm{CP}} \cdot \max_{S:\text{separator for } \widetilde{\tau_P}} \sqrt{n}^{|V(\widetilde{\tau}_P)| - |S| + I(\widetilde{\tau}_P)} \cdot k^{|\mathrm{CP}(Y \circ \tau \circ Y')| - \frac{|U_Y| + |V_{Y'}|}{2}}
\end{aligned}$$

By construction, V_Y and $U_{Y'}$ are color MVS, hence there is no rainbow component across pieces of Y and τ, and similarly across τ and Y'. We can rewrite

$$Ev^{\approx}(Y \circ \tau \circ Y') = Ev^{\approx}(Y) \cdot Ev(\tau) \cdot Ev^{\approx}(Y')\,;$$

Further unpacking the factors of $\mathrm{CP}(Y \circ \tau \circ Y')$gives us

$$\begin{aligned}
\lambda(\widetilde{\tau}_P)_{\mathrm{CP}} \cdot w(\widetilde{\tau}_P)_{\mathrm{CP}} = \left(\frac{1}{k}\right)^{|V(Y \circ \tau \circ Y')| - \frac{|U_Y| + |V_{Y'}|}{2}} &\cdot Ev^{\approx}(Y)_{\mathrm{CP}(Y)} \cdot Ev(\tau)_{\mathrm{CP}(\tau)} \\
&\cdot Ev^{\approx}(Y')_{\mathrm{CP}(Y')} \cdot \max_{S:\text{separator for } \widetilde{\tau}_P} \sqrt{n}^{|V(\widetilde{\tau}_P)| - |S| + |I(\widetilde{\tau}_P)|} \\
&\cdot k^{|\mathrm{CP}(Y)| - \frac{|\mathrm{CP}(U_Y)| + |\mathrm{CP}(V_Y)|}{2}} \cdot k^{|\mathrm{CP}(\tau)| - \frac{|\mathrm{CP}(U_\tau)| + |\mathrm{CP}(V_\tau)|}{2}} \cdot k^{|\mathrm{CP}(Y')| - \frac{|\mathrm{CP}(U_{Y'})| + |\mathrm{CP}(V_{Y'})|}{2}}
\end{aligned}$$

Next, we show how to wrap up this bound with the following inequalities proven in the subsequent claims.

1. From claim 8.7.6, for any separator S of τ_P,

$$\begin{aligned}
&\left(\frac{1}{k}\right)^{|V(Y\circ\tau\circ Y')|-\frac{|U_Y|+|V_{Y'}|}{2}} \cdot \sqrt{n}^{|V(\widetilde{\tau}_P)|-|S|+|I(\widetilde{\tau}_P)|} \\
&= (\frac{\sqrt{n}}{k})^{|V(Y\circ\tau\circ Y')|-\frac{|U_Y|+|V_{Y'}|}{2}} \cdot \sqrt{n}^{|V(\widetilde{\tau}_P)|-|S|+|I(\widetilde{\tau}_P)|-(|V(Y\circ\tau\circ Y')|-\frac{|U_Y|+|V_{Y'}|}{2})} \\
&= (\frac{\sqrt{n}}{k})^{|V(Y\circ\tau\circ Y')|-\frac{|U_Y|+|V_{Y'}|}{2}} \cdot \sqrt{n}^{\frac{|U_\tau|+|V_\tau|}{2}-|S(\tau)|+|I(\tau)|-\frac{|I_{dc}(U_Y)|+|I_{dc}(V_{Y'})|}{2}}
\end{aligned}$$

   where $I_{dc}(U_Y)$ is the subset of vertices in U_Y not connected to V_Y, and $I_{dc}(V_{Y'})$ the subset of vertices in $V_{Y'}$ not connected to $U_{Y'}$.

2. From claim 8.7.8

$$Ev(Y)_{\mathrm{CP}(Y)} \cdot k^{|\mathrm{CP}(Y)|-\frac{|\mathrm{CP}(U_Y)|+|\mathrm{CP}(V_Y)|}{2}} \leq (\frac{1}{k})^{\frac{1}{2}\cdot(\#\text{ components in } U_Y \text{ disconnected from } V_Y\ )};$$

   and similarly,

$$Ev(Y')_{\mathrm{CP}(Y')} \cdot k^{|\mathrm{CP}(Y')|-\frac{|\mathrm{CP}(U_{Y'})|+|\mathrm{CP}(V_{Y'})|}{2}} \leq (\frac{1}{k})^{\frac{1}{2}\cdot(\#\text{ components in } V_{Y'} \text{ disconnected from } U_Y\ )}$$

3. Rewriting the slack bound as

$$\begin{aligned}
&(\frac{\sqrt{n}}{k})^{|V(Y\circ\tau\circ Y')|-\frac{|U_Y|+|V_{Y'}|}{2}} \\
&= (\frac{\sqrt{n}}{k})^{|V(Y)|-\frac{|U_Y|+|V_Y|}{2}} \cdot (\frac{\sqrt{n}}{k})^{|V(\tau)|-\frac{|U_\tau|+|V_\tau|}{2}} \cdot (\frac{\sqrt{n}}{k})^{|V(Y')|-\frac{|U_{Y'}|+|V_{Y'}|}{2}}
\end{aligned}$$

With these bounds, we can simplify the above equation as

$$
\begin{aligned}
&\lambda^{\approx}(\widetilde{\tau}_P)_{\mathrm{CP}} \cdot w(\widetilde{\tau}_P)_{\mathrm{CP}} \\
&\leq \sqrt{n}^{|I(\tau)|+\frac{|U_\tau|+|V_\tau|}{2}-|S(\tau)|-\frac{|I_{dc}(U_Y)|+|I_{dc}(V_{Y'})|}{2}} \cdot Ev(\tau)_{\mathrm{CP}(\tau)} \cdot k^{|\mathrm{CP}(\tau)|-\frac{|\mathrm{CP}(U_\tau)|+|\mathrm{CP}(V_\tau)|}{2}} \\
&\cdot k^{\frac{1}{2}\cdot(\#\text{ components in } U_Y \text{ disconnected from } V_Y\ )} \cdot k^{\frac{1}{2}\cdot(\#\text{ components in } V_{Y'} \text{ disconnected from } U_Y\ )} \\
&= \left(\sqrt{n}^{|I(\tau)|+\frac{|U_\tau|+|V_\tau|}{2}-|S(\tau)|} \cdot Ev(\tau)_{\mathrm{CP}(\tau)} \cdot k^{|\mathrm{CP}(\tau)|-\frac{|\mathrm{CP}(U_\tau)|+|\mathrm{CP}(V_\tau)|}{2}}\right) \cdot (\frac{\sqrt{n}}{k})^{|V(Y\circ\tau\circ Y')|-\frac{|U_Y|+|V_{Y'}|}{2}} \\
&\cdot \left(k^{\frac{1}{2}\cdot(\#\text{ components in } U_Y \text{ disconnected from } V_Y\ )} \cdot k^{\frac{1}{2}\cdot(\#\text{ components in } V_{Y'} \text{ disconnected from } U_Y\ )} \cdot (\frac{1}{\sqrt{n}})^{\frac{|I_{dc}(U_Y)|+|I_{dc}(V_{Y'})|}{2}}\right) \\
&\leq \left(\sqrt{n}^{|I(\tau)|+\frac{|U_\tau|+|V_\tau|}{2}-|S(\tau)|} \cdot Ev(\tau)_{\mathrm{CP}(\tau)} \cdot k^{|\mathrm{CP}(\tau)|-\frac{|\mathrm{CP}(U_\tau)|+|\mathrm{CP}(V_\tau)|}{2}} \cdot (\frac{\sqrt{n}}{k})^{|V(\tau)|-\frac{|U_\tau|+|V_\tau|}{2}}\right) \\
&\cdot (\frac{\sqrt{n}}{k})^{|V(Y)|-\frac{|U_Y|+|V_Y|}{2}} \cdot (\frac{\sqrt{n}}{k})^{|V(Y)|-\frac{|U_{Y'}|+|V_{Y'}|}{2}}
\end{aligned}
$$

where the last inequality follows by observing

$$(\#\text{ components in } U_Y \text{ disconnected from } V_{Y'}\ ) \leq |I_{dc}(U_Y)|\,,$$

and analogously

$$(\#\text{ components in } V_{Y'} \text{ disconnected from } U_{Y'}\ ) \leq |I_{dc}(V_{Y'})|$$

as any disconnected component C in U_Y (respectively in $U_{Y'}$) gives contribution $|C \cap U_Y| \geq 1$ (respectively $|C \cap V_{Y'}| \geq 1$) to the set in $I_{dc}(U_Y)$ (respectively $I_{dc}(V_{Y'})$), and we ignore the extra slack from I_{dc} in the final bound .

Finally, noticing the first term is exactly $\lambda^{\approx}(\tau)_{\mathrm{CP}(\tau)} \cdot w(\tau)_{\mathrm{CP}(\tau)}$ from the previous level of intersection, we have

$$\frac{\lambda^{\approx}(\widetilde{\tau}_P)_{\mathrm{CP}} \cdot w(\widetilde{\tau}_P)_{\mathrm{CP}}}{\lambda^{\approx}(\tau)_{\mathrm{CP}(\tau)} \cdot w(\tau)_{\mathrm{CP}(\tau)}} \leq (\frac{\sqrt{n}}{k})^{|V(Y)|-\frac{|U_{Y'}|+|V_{Y'}|}{2}} \cdot (\frac{\sqrt{n}}{k})^{|V(Y)|-\frac{|V_{Y'}|+|U_{Y'}|}{2}}$$

In other words, we pick up decay factor for each newly introduced vertex in $V(Y)$ and $V(Y')$ provided it is not in the intersection of $U_Y \cap V_Y$ and $U_{Y'} \cap V_{Y'}$. $\square$

Key Claims for Intersection Term Analysis We now state and prove the claims used in the above analysis.

Claim 8.7.6. *Recall $I(\tau)$ denote the set of isolated vertices, with some abuse of notation, we additionally denote $I_{dc}(U_Y) \subseteq U_Y$ be the set of vertices not connected to V_Y , and similarly let $I_{dc}(V_{Y'}) \subseteq V_{Y'}$ be the set of vertices not connected to $U_{Y'}$,*

$$\begin{aligned}&|V(\widetilde{\tau}_P)| - |S(\widetilde{\tau}_P)| + |I(\widetilde{\tau}_P)| - |I(\tau)| - \left(|V(Y \circ \tau \circ Y')| - \frac{|U_Y| + |V_{Y'}|}{2}\right) \\ &\leq \frac{|U_\tau| + |V_\tau|}{2} - |S(\tau)| - \frac{|I_{dc}(U_Y)| + |I_{dc}(V_{Y'})|}{2}\end{aligned}$$

Proof. Applying the intersection tradeoff lemma, we have

$$|V(\widetilde{\tau}_P)| + |I(\widetilde{\tau}_P)| - |S(\widetilde{\tau}_P)| \leq |V(\tau)| + |I(\tau)| - |S(\tau)| + |V(Y) \setminus U_Y| + |V(Y') \setminus V_{Y'}|\,.$$

Rearranging, we have

$$|V(\widetilde{\tau}_P)| + |I(\widetilde{\tau}_P)| - |S(\widetilde{\tau}_P)| - |I(\tau)| \leq |V(\tau)| - |S(\tau)| + |V(Y) \setminus U_Y| + |V(Y') \setminus V_{Y'}|$$

Note that

$$\begin{aligned}|V(\tau)| + |V(Y) \setminus U_Y| + |V(Y') \setminus V_{Y'}| &= |V(\tau)| + |V(Y)| - |U_Y| + |V(Y')| - |V_{Y'}| \\ &= |V(Y \circ \tau \circ Y')| + |U_\tau| + |V_\tau| - |U_Y| - |V_{Y'}|\end{aligned}$$

since each vertex in $U_\tau = V_Y, V_\tau = U_{Y'}$ gets counted an extra time each. Plugging this to

the above and rearranging gives us

$$|V(\widetilde{\tau}_P)| + |I(\widetilde{\tau}_P)| - |S(\widetilde{\tau}_P)| - |V(Y \circ \tau \circ Y')| \leq |U_\tau| + |V_\tau| - |U_Y| - |V_{Y'}| - |S(\tau)|$$

Adding $\frac{|U_Y|+|V_{Y'}|}{2}$ gives us the desired LHS from the claim , and on the right-hand-side we have

$$\begin{aligned} RHS &= |U_\tau| + |V_\tau| - |U_Y| - |V_{Y'}| - |S(\tau)| + \frac{|U_Y| + |V_{Y'}|}{2} \\ &= (|U_\tau| - \frac{|U_Y|}{2}) + (|V_\tau| - \frac{|V_{Y'}|}{2}) - |S(\tau)| \\ &\leq \frac{|U_\tau| - |I_{dc}(U_Y)|}{2} + \frac{|V_\tau| - |I_{dc}(V_{Y'})|}{2} - |S(\tau)| \end{aligned}$$

where the last line uses $|U_\tau| + |I_{dc}(U_Y)| \leq |U_Y|, |V_\tau| + |I_{dc}(V_{Y'})| \leq |V_{Y'}|$ from the subsequent claim. Rearranging the above equation completes the proof. □

Claim 8.7.7. *For Y a left-shape,*

$$|V_Y| + |I_{dc}(U_Y)| \leq |U_Y|$$

Proof. Recall that $I_{dc}(U_Y) \subseteq U_Y \setminus V_Y$ is the set of edges not incident to any edge in Y, any path from V_Y to $I_{dc}(U_Y)$ is trivially blocked. Next we observe that V_Y is also an MVS (in fact, unique) separating $U_Y \setminus I_{dc}(U_Y)$ and V_Y : otherwise, suppose $T \neq V_Y$ is such an MVS with $|T| \leq |V_Y|$, T is also a separator for Y and we have a contradiction that Y is a left-shape in which V_Y is the unique MVS. Since $U_Y \setminus I_{dc}(U_Y)$ is also a separator separating $U_Y \setminus I_{dc}(U_Y)$ and V_Y, we have

$$|V_Y| \leq |U_Y \setminus I_{dc}(U_Y)|$$

Rearranging gives us the desired. □

Claim 8.7.8. *For a left-shape Y,*

$$Ev(Y)_{CP(Y)} \cdot k^{|CP(Y)|-\frac{|CP(U_Y)|+|CP(V_Y)|}{2}} \leq k^{\frac{1}{2}\cdot(\#\textit{ components in } U_Y \textit{ disconnected from } V_Y\ )}$$

The same claim holds for right-shape Y' by taking its transpose.

Proof. Recall that

$$Ev(Y)_{\mathrm{CP}(Y)} \coloneqq \prod_{\substack{G_i\text{rainbow comp. in } Y \\ \text{connected to } U_Y \cup V_Y}} (\frac{1}{k})^{|V(G_i)|-1}$$

and we drop the connectivity constraint to $U_Y \cup V_Y$ as any component in Y is connected to $U_Y \cup V_Y$. Furthermore, any component in V_Y is connected to U_Y, while there may be isolated component in U_Y not connected to V_Y.

We now show that

$$|\mathrm{CP}(Y)| - |\mathrm{CP}(U_Y)| \leq \sum_{\substack{G_i\text{rainbow comp. in } Y \\ \text{connected to } U_Y \cup V_Y}} (|V(G_i)| - 1)$$

and

$$|\mathrm{CP}(Y)| - |\mathrm{CP}(V_Y)| \leq \sum_{G_i\text{rainbow comp. in } Y} (|V(G_i)| - 1) + (\#\text{ components in } U_Y \text{ disconnected from } V_Y\ )$$

We highlight the proof for the second inequality. This is equivalent to bounding $|\mathrm{CP}(Y) \setminus \mathrm{CP}(V_Y)|$, and consider a traversal from V_Y, among the components connected to V_Y, each rainbow component is discovered in the traversal via a pure-color edge (or via its incidental vertex in V_Y), and therefore we explore at most $|V(G_i)| - 1$ new colors from the vertices while we pick up a factor $|V(G_i)| - 1$ from the RHS.

For components not connected to V_Y, any such component is connected to U_Y, and for any such component, we pick one of its incidental vertices in U_Y and apply the above

process. This proves the second inequality. The first inequality follows as any component is connected to U_Y in Y.

Taking the average of the above gives us

$$|\mathrm{CP}(Y)| - \frac{|\mathrm{CP}(U_Y)| + |\mathrm{CP}(V_Y)|}{2}$$
$$\leq \sum_{\substack{G_i \text{rainbow comp. in } Y \\ \text{connected to } U_Y \cup V_Y}} (|V(G_i)| - 1) + \frac{1}{2} \cdot (\#\text{ components in } U_Y \text{ disconnected from } V_Y\ )$$

□

8.8 Handling the Kernel

8.8.1 Overview of the Kernel Analysis

In this section, we address the kernel issue in LQL^T factorization. In the previous section, we focus on terms that appear in Q without intersections. That said, in our context, since we have a non-trivial kernel coming from the coloring constraint, it turns out that there are terms that cannot be charged to the diagonal of Id by a magnitude argument. In particular, instead of getting Id^{dsos} on the diagonal, we have $(Id - \frac{J}{k})^{\mathsf{dsos}}$.

A Candidate Kernel from the Trivial Term in Q

Lemma 8.8.1. *Following the LQL^T decomposition of Λ, the trivial shape (edgeset being $\emptyset$) appears in Q as $(I - \frac{J_k}{k})^{\otimes u}$ for $U = V$ and $|U| = |V|$.*

Proof. Let $Q_\emptyset \coloneqq \langle Q, \emptyset \rangle$ be the matrix of coefficient of empty edge-set in Q where $\langle \cdot, \cdot \rangle$ denotes the Fourier inversion operator. In other words, for any entry $(S, \beta_S), (T, \beta_T)$,

$$Q_\emptyset[(S, \beta_S), (T, \beta_T)] \coloneqq \langle \emptyset, Q[(S, \beta_S), (T, \beta_T)] \rangle\ .$$

It suffices for us to show the entry of $Q_\emptyset[(U,\beta_U),(U,\beta'_U)] = (1-\frac{1}{k})^u \cdot (-\frac{1}{k})^q$. for u the number of vertices with the same color in both β_U and β'_U, and q be the number of vertices with different colors under β, β'. For convenience, we denote any such entry by $F(u,q)$.

Claim 8.8.2. *For any $u, q \geq 0$, $F(u,q)$ satisfies the following recurrence*

$$F(u,q) = \mathbf{1}[q=0] - \sum_{a,b:a+b>0}^{a\leq u, b\leq q} \binom{u}{a}\binom{q}{b} \cdot (\frac{1}{k})^{a+b} \cdot F(u-a, q-b)$$

with base case $F(0,0) = 1, F(1,0) = 1-\frac{1}{k}$, and $F(0,1) = -\frac{1}{k}$.

Proof. The base case is equivalent to looking at $Q[\emptyset,\emptyset]$ and we have 1 from the trivial shape, and observe that there is no intersection term at this entry, i.e., $Q_i[\emptyset,\emptyset] = 0$ and hence the base case. The further condition of $F(1,0)$ and $F(0,1)$ holds as $\langle Q_1[(i,\_),(i,\_)],\emptyset\rangle = -\frac{1}{k}$, while $F(1,0)$ contains the extra trivial shape without intersection from $Q_0[(i,a),(i,a)] = 1$.

For the general case, observe that the term $\mathbf{1}[q=0]$ controls the contribution from Q_0, i.e. we only pick non-intersected trivial shape from Q_0 when every vertex has the same color in both coloring. For contribution from Q_i for $i > 0$, we case on the outmost level of intersections,

1. Let a be the number of trivial intersection from vertices with the same color, and b be the number of trivial intersection from vertices with different color;

2. Since the pure-color/cross-color vertices are fixed to start with, there are simply $\binom{u}{a}$ and $\binom{q}{b}$ ways of choosing intersection vertices at the outer level, provided $a+b>0$ (i.e., some vertex intersection happens);

3. Notice that each vertex intersection gives us a factor of $\frac{1}{k}$ along with a (-1) sign for any level of intersection; once choosing the action of the outmost level, the

contribution is then multiplied by the coefficient of the subsequent level, given by $F(u-a, q-b)$ by our hypothesis. Alternatively, given any term contributing to $F(u-a, q-b)$, there are $\binom{u}{a} \cdot \binom{q}{b}$ of picking it to contribute $F(u,q)$ by padding non-trivial intersection at the outmost level.

□

Finally, inducting on u, q such that $F(u,q) = (1-\frac{1}{k})^u \cdot (-\frac{1}{k})^q$, the we complete the proof to the lemma by the next two equalities from the Binomial Theorem.

$$\sum_{a=0}^{u} \binom{u}{a} \left(\frac{1}{k}\right)^a (1-\frac{1}{k})^{u-a} = \left((1-\frac{1}{k}) + \frac{1}{k}\right))^u,$$

and

$$\sum_{b=0}^{q} \binom{q}{b} \left(\frac{1}{k}\right)^b \cdot (-\frac{1}{k})^b = (\frac{1}{k} - \frac{1}{k})^q.$$

In the case $q = 0$, we verify

$$\begin{aligned}
F(u,0) &= 1 - \sum_{a,b:a+b>0}^{a\le u, b\le q} \binom{u}{a}\binom{q}{b} \cdot (\frac{1}{k})^{a+b} \cdot F(u-a, q-b) \\
&= 1 - \sum_{a>0}^{u} \binom{u}{a} (\frac{1}{k})^a \cdot F(u-a, 0) \\
&= 1 - \sum_{a>0}^{u} \binom{u}{a} (\frac{1}{k})^a \cdot (1-\frac{1}{k})^{u-a} \quad \text{(By induction hypothesis)} \\
&= 1 - \left((1-\frac{1}{k}) + \frac{1}{k})^u - (1-\frac{1}{k})^u\right) \\
&= (1-\frac{1}{k})^u
\end{aligned}$$

as desired where we rewrite

$$\sum_{a>0}^{u} \binom{u}{a} (\frac{1}{k})^a \cdot (1-\frac{1}{k}) = \left((1-\frac{1}{k}) + \frac{1}{k}\right)^u - \underbrace{(1-\frac{1}{k})^u}_{a=0}$$

For the case $q > 0$, we have

$$\begin{aligned}
F(u,q) &= -\sum_{a,b:a+b>0}^{a\le u, b\le q} \binom{u}{a}\binom{q}{b} \cdot (\frac{1}{k})^{a+b} \cdot F(u-a, q-b) \\
&= -\sum_{a,b:a+b>0}^{a\le u, b\le q} \binom{u}{a}\binom{q}{b} \cdot (\frac{1}{k})^{a+b} \cdot (1-\frac{1}{k})^{u-a} \cdot (-\frac{1}{k})^{q-b} \quad \text{(By induction hypothesis)} \\
&= -\left(\sum_{a=0}^{u} \binom{u}{a} (\frac{1}{k})^{a} (1-\frac{1}{k})^{q-a}\right) \cdot \left(\sum_{b=0}^{q} \binom{q}{b} (\frac{1}{k})^{b} \cdot (-\frac{1}{k})^{q-b}\right) \\
&= -\left(0 - (1-\frac{1}{k})^{u} \cdot (-\frac{1}{k})^{q}\right) \\
&= (1-\frac{1}{k})^{u} \cdot (-\frac{1}{k})^{q}
\end{aligned}$$

as desired. □

Candidate Kernel of Q: a first try Before we formally show the kernel vectors, it is enticing at the first attempts to suspect the following subspace be the kernel of Q as given by the trivial shape on the diagonal block corresponding to colorings of the same set of vertices.

To get started, we define the coloring-vector that naturally comes up when working with the color-indexed moment matrix,

Definition 8.8.3 (Coloring-vector)**.** *For any* $\mathsf{dsos} \in \mathbb{N}$*, we define coloring vector as the vector in the appropriate dimension for the corresponding color-moment matrix, i.e., each vector's entry is indexed by*

- $S \subseteq [n]$*, and* $|S| \le \mathsf{dsos}$*;*
- $\beta_S : S \to [k]$*: the coloring for* S*.*

With some abuse of notation, we consider this vector to be of dimension $\binom{[n]\times[k]}{\le \mathsf{dsos}}$*.*

Definition 8.8.4 (Candidate kernel of $\mathbf{1}_{(S,\beta_S)\cup T}$)**.** *For any set $\emptyset \neq T \subseteq [n]$, we define $K_T = \mathsf{Span}\{\mathbf{1}_{S,\beta_S} \otimes 1_T\}_{S\cap T=\emptyset}$ where we write*

1. *$\mathbf{1}_{S,\beta_S}$ is the (coloring) indicator-vector of dimension $\binom{[n]\times[k]}{\leq \mathsf{dsos}}$ that only has non-zero entry 1 at the entry for (S,β_S);*

2. *$\mathbf{1}_T = \sum_{\beta_T:V(T)\to[k]} \mathbf{1}_{T,\beta_T}$, i.e., the vector with non-zero entry 1 on any coloring of T.*

Additionally, for any choice of (S,β_S) such that $S\cap T = \emptyset$, we let $\mathbf{1}_{(S,\beta_S),T} := \mathbf{1}_{S,\beta(S)} \otimes \mathbf{1}_T$ denote the following vector

1. *This is a coloring-vector with non-zero entries indexed by $(S\cup T, \beta_S \cup \beta_T)$ for any coloring $\beta_T : T \to [k]$,*

2. *and each non-zero entry is 1.*

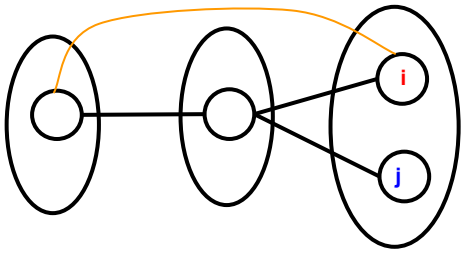
For notational clarity, we also write $\mathbf{1}_{(S,\beta_S),T} = \mathbf{1}_{(S,\beta_S)\cup T}$ to emphasize that S are the vertices with fixed colors while we are summing over colors of T, consistent to our definition of kernel partition.

Example 8.8.5. *Plugging in $S = \emptyset$ and $T = \{i\}$ gives us the k-dimensional all-1 vector on color of i.*

However, this natural guess turns out to be false, unlike the lower bound for deg-2 SoS, it can be verified that $v = e_{i,1} \otimes 1_j$, i.e. the vector v that has all 1-entry on $v[(i,1),(j,c_j)]$ and 0 everywhere else is not *exactly* in the kernel of Q. See the following diagram for illustration. Another way to see this issue is that there are also additional truncation terms in the final matrix Q that are of small magnitude but present, hence we need a more robust way of handling the kernel.

This motivates us to adjust the proof strategy for the kernel as the following inspired by the kernel trick from [GJJ+20b]. Our proof strategy is two-step,

1. Identify the Kernel of Q;



(a) A Surviving Term from $Qv \neq 0$

2. "Filling in" the kernel of Q so that the final matrix has genuine identity on the diagonal as PSD mass to enable a magnitude argument.

Identifying the Kernel of Q We first observe the kernel of the moment matrix Λ (after diagonal rescaling of factor $k^{\frac{|S|}{2}}$ from $\widetilde{\Lambda}$ so that Λ has diagonal 1 instead of $(\frac{1}{k})^{|S|}$ for vertex-set S.) , and apply L^T on the kernel to identify the (right) kernel of Q. Since Q is a symmetric matrix, this would also give us the kernel of Q. Let's now observe the kernel for Λ, which follows from the sum-constraint for our moment matrix.

Definition 8.8.6 ((Unscaled) Kernel basis vector of Λ)**.** *For any set of vertices $S \in [n]$, and for any $T \cap S = \emptyset$, $T \neq \emptyset$, we define $\hat{r}_{(S,\beta_S),T}$ as a coloring-vector with the following non-zero entries,*

1. $\hat{r}_{(S,\beta_S),T}[S, \beta_S] := -1$;

2. *For any coloring $\beta_T : T \to [k]$, $\hat{r}_{(S,\beta_S),T}[S \cup T, \beta_S \cup \beta_T] := 1$.*

We also use the notation $\hat{r}_{(S,\beta_S),T} = \hat{r}_{(S,\beta_S)\cup T}$ interchangeably.

Let $K_{\Lambda}^{\leq \mathsf{dsos}}$ be the set of $\hat{v}_{(S,\beta_S),T}$ vectors defined above for any S and T with $|S \cup T| \leq \mathsf{dsos}$ and any β_S, and we drop the dependence of dsos when it is clear.

Example 8.8.7. *Consider $T = \{i\}$, we have non-zero entries at $\hat{r}_{\emptyset,T}[\emptyset] = -1$ and $\hat{r}_{\emptyset,T}[(i, c)] = 1$ for any $c \in [k]$.*

We will more often work with the basis vectors in the following scaling,

Definition 8.8.8 ((Scaled) Kernel basis of Λ). *For any set of vertices $S \in [n]$, and for any $T \cap S = \emptyset$, $T \neq \emptyset$, we define the scaled basis vector as*

$$r_{(S,\beta_S),T} = \left(\frac{1}{\sqrt{k}}\right)^{|T|} \cdot \hat{r}_{(S,\beta_S),T}$$

For intuition, one can view $r_{(S,\beta_S),T}$ as the basis vector for the kernel arises by pinning the vertices S into coloring β_S, and then sum over all colors of T.

Claim 8.8.9 (Verifying the kernel for Λ). *For any $S, T \subseteq [n]$ and β_S, $r_{(S,\beta_S),T} \in K_\Lambda$, we verify the kernel property, i.e., we have*

$$\Lambda r_{(S,\beta_S),T} = 0$$

Proof. This follows by the constraint for any subset of vertices $S \subseteq [n]$, the moment matrix satisfies

$$\widetilde{\mathbb{E}}[\sum_{\beta:T\to[k]} x_{T,\beta}] = \widetilde{\mathbb{E}}[1]\,,$$

and more generally within the polynomial system,

$$\widetilde{\mathbb{E}}[x_{S,\beta_S} \cdot (\sum_{\beta:T\to[k]} x_{T,\beta_T})] = \widetilde{\mathbb{E}}[\sum_{\beta_T:T\to[k]} x_{S\cup T,\beta_S\cup\beta_T}] = \widetilde{\mathbb{E}}[x_{S,\beta_S}]\,.$$

To examine the entries of $\Lambda r_{S,\beta_S,T}$, note this is a vector indexed by (A,β_A) for $A \subseteq [n]$ and $\beta_A : A \to [k]$, and for any (A,β_A), we have

$$\begin{aligned}
\frac{1}{\sqrt{k}^{|S|}}\Lambda r_{(S,\beta_S),T}[A,\beta_A] &= -\Lambda[(A,\beta_A),(S,\beta_S)] + \sum_{\beta_T:T\to[k]} \Lambda[(A,\beta_A),(S\cup T,\beta_S\cup\beta_T)] \\
&= -\widetilde{\mathbb{E}}[x_{A\cup S,\beta_A\cup\beta_S}] + \widetilde{\mathbb{E}}[\sum_{\beta_T:T\to[k]} x_{A\cup S\cup T,\beta_A\cup\beta_S\cup\beta_T}] \\
&= \widetilde{\mathbb{E}}[x_{A\cup S,\beta_A\cup\beta_S} \cdot (\sum_{\beta_T} x_{T,\beta_T} - 1)]
\end{aligned}$$

$$= 0$$

where the last equality follows from the aforementioned sum-color constraint for T. $\square$

Recall from definition 8.5.8 that L the left-shape matrix with rows indexed by (U_σ, β_U) and columns indexed by (V_σ, β_V), we next show that L does not have non-trivial right kernel, and therefore, for any vector $v \in Ker(\Lambda)$, we also have $L^T v \in Ker(Q)$.

Lemma 8.8.10. *L does not have non-trivial right kernel, i.e., for any vector $v \neq 0$, we have $Lv \neq 0$; similarly, L does not have non-trivial left-kernel, i.e. $L^T v \neq 0$ for any $v \neq 0$.*

Proof. We first prove that L does not have non-trivial right-kernel. For any $v \neq 0$, consider the minimal support of v, i.e., the index (S, β_S) such that $v[(S, \beta_S)] \neq 0$ for the smallest $|S|$, and we claim that $Lv[(S, \beta_S)] = v[(S, \beta_S)] \neq 0$.

To see this observe that $L[(S, \beta_S), (S, \beta_S)] = (\frac{1}{\sqrt{k}})^{|S|}$, and moreover, for any T with $|T| < |S|$, we have $v[(T, \beta_T)] = 0$ for any β_T by our minimal assumption. That said, it suffices for us to examine the entries of $L[(S, \beta_S), (T, \beta_T)]$ for $|T| \geq |S|$. Note that for $|T| = |S|$ yet $S \neq T$, there is no non-zero term as the only left shape with $|U_\sigma| = |V_\sigma|$ is the identity shape; on the other hand, for $|T| > |S|$, there is no left-shape of this form as for any left shape σ we need to have $|U_\sigma| \geq |V_\sigma|$. Therefore, we have $Lv \neq 0$ for non-zero v.

Similarly, for $L^T v$, consider the largest index $|S|$ such that $v[(S, \beta_S)] \neq 0$, and observe that $L^T v[(S, \beta_S)] = v[(S, \beta_S)] \neq 0$. $\square$

Claim 8.8.11 ($L^T K(\Lambda) \subseteq Ker(Q)$)**.** *For any $v_{(S,\beta_S),T} \in K(\Lambda)$,*

$$Q \underbrace{L^T v_{(S,\beta_S),T}}_{:= u_{(S,\beta_S),T}} = 0\,.$$

Proof. By claim 8.8.9, we know

$$\Lambda v_{(S,\beta_S),T} = 0\,,$$

and since $\Lambda = LQL^T$, we have

$$LQL^T v_{(S,\beta_S),T} = LQ \cdot u_{(S,\beta_S),T} = 0.$$

By lemma 8.8.10, L does not have non-trivial left-kernel, hence we have

$$Q \cdot u_{(S,\beta_S),T} = 0\,.$$

Moreover, L does not have non-trivial right-kernel, thus $u_{(S,\beta_S),T} \neq 0$ and is in the kernel of Q. □

PSD Analysis via Filling in the Kernel Now that we have identified the kernel of matrix Q, we note that we may still apply a magnitude-bound based argument except on a closely related matrix to show the PSDness of Q.

Claim 8.8.12. *For any symmetric matrix M with kernel $Ker(M)$, M is PSD if*

$$\widetilde{M} = M + \sum_{u_i \in Ker(M)} v_i u_i^T + u_i v_i^T \succeq 0$$

Proof. To verify the PSDness of M, it is sufficient to check for any $x \perp Ker(M)$,

$$x^T M x = x^T M x + \sum_{i:u_i \in Ker(M)} \langle x, v_i\rangle\langle x, u_i\rangle + \langle x, u_i\rangle\langle x, v_i\rangle = x^T \widetilde{M} x \geq 0$$

where the second inequality inequality follows $u_i \in Ker(M)$, and the last one from PSDness of $\tilde{M}$. □

The kernel trick motivates us now to find a matrix K_Q such that $R = \sum_i c_i(v_i u_i^T + u_i v_i^T)$ for some $u_i \in Ker(Q)$ and show $\widetilde{Q} = Q + K_Q$ being PSD instead. Our work amounts to identifying the right choice of $u_i \in Ker(Q)$, however, this turns out to be a surprisingly challenging task as K constructed from the natural kernel-vector (coloring-vector) contains extra large-norm components that warrant extra care. Towards this end, analogous to the recursive factorization scheme via LQL^T, we iteratively identify the kernel-vector such that the resulting matrix is amenable to a magnitude-bound analysis for PSDness. This is the focus of our next subsection, and we give a summary at the end of this section to wrap up our kernel analysis.

8.8.2 Overview for Kernel-Factorization

Recall our definition of kernel partition defined from the overview section,

Definition 8.8.13 (Kernel Partition)**.** *Given a set $B \subseteq [n]$, we call $S(B) \cup T(B)$ a kernel partition for B if vertices in $S(B)$ receives some fixed color, and $T(B)$ are the vertices whose colors are being summed over. In other words, a kernel partition is a kernel-partition with partial-coloring except with $\beta_{S(B)}$ the fixed colors on $S(B)$ unspecified.*

Definition 8.8.14 (Kernel matrix $K_{S\cup T}$ for Kernel-Partition)**.** *For any kernel partition $S \cup T \subseteq [n]$ such that S (possibly empty) are the vertices with color fixed to some β_S, and $T \neq \emptyset$ the vertices without color. We define*

$$K_{S\cup T} := \otimes_{i\in S\cup T} K_i$$

where $K_i = Id_k$ is a $k \times k$ identity matrix for $i \in S$ and $K_i = -\frac{J_k}{k}$ for $i \in T$ is the the all-1 matrix of dimension $k \times k$ scaled by a factor of $-\frac{1}{k}$,

In other words, this is a coloring matrix of dimension $([n] \times [k])^{|S\cup T|} \times ([n] \times [k])^{|S\cup T|}$, and

has non-zero entry at

$$K_{S\cup T}[(S\cup T,\beta_S\cup\beta_T),(S\cup T,\beta_S\cup\beta'_T)]=(-\frac{1}{k})^{|T|}.$$

for any $\beta_S:S\to[k]$ *and* $\beta_T,\beta'_T:T\to[k]$. .

Note that $K_{(S,\beta_S)\cup T}$ corresponds to the damage of identity in Q due to intersection terms inside T (where S are the non-intersected vertices with fixed colors as β_S for any β_S) . The goal of this section is to identify a construction of kernel-filling vectors $f_{S\cup T}$ for $S\cup T$ such that $L^T f_{S\cup T}$ can be used as a proxy to generate matrix $K_{S\cup T}$. Formally, we construct K_Q as the following from defined $f_{S\cup T}$ vectors,

Definition 8.8.15 (Construction of K_Q from Kernel-Filling Vectors)**.** *Given a collection of coloring vectors* $\{f_{(S,\beta_S),T}\}$ *, we define* K_Q *to be a coloring matrix of corresponding dimension for* dsos

$$K_Q:=\sum_{B\subseteq[n]}\sum_{\substack{S(B)\cup T(B)\\ \text{Kernel-Partition for } B}}\sum_{\beta_S:S(B)\to[k]}\left(\frac{1}{2}f_{(S,\beta_S),T}\otimes\frac{\mathbf{1}_{(S,\beta_S),T}}{\sqrt{k}^{|T|}}+\frac{1}{2}\cdot(\frac{1}{\sqrt{k}})^{|T|}\mathbf{1}_{(S,\beta_S),T}\otimes f_{(S,\beta_S),T}\right)$$

The collection of coloring-vectors $\{f_{(S,\beta_S),T}\}$ *is to be defined in definition 8.8.33.*

Note that the particular form of $\frac{1}{2}f_{(S,\beta_S),T}\otimes\mathbf{1}_{(S,\beta_S),T}+\frac{1}{2}\otimes\mathbf{1}_{(S,\beta_S),T}\otimes f_{(S,\beta_S),T}$ is to ensure symmetry of the matrix K_Q, and the extra $(\frac{1}{\sqrt{k}})^{|T|}$ normalization is to ensure that we are aligning to $\frac{I}{k}$ factor for each vertex in T for the chosen normalization in f. For our magnitude bound analysis, it is intuitive to simply consider $\widetilde{K}_Q=f_{(S,\beta_S),T}\otimes\mathbf{1}_{(S,\beta_S),T}$ without symmetrizing as we pick up the same desired term on $\mathbf{1}_{(S,\beta_S),T}$ and the magnitude bound holds under symmetrizations as we do not pay attention to constant slack.

Recall that

$$K_{\text{trivial}}=\sum_{\substack{W\subseteq[n]\\ |W|\leq\text{dsos}}}\sum_{\substack{S(W)\cup T(W)\\ \text{kernel-partition}}}K_{S(W)\cup T(W)}$$

and the upshot of our construction is in finding the "correct" kernel-filling vectors such that we can take

$$\|K_{\mathsf{nontrivial}}\| = \|K_Q - K_{\mathsf{trivial}}\| = o(1)$$

for some matrix $K_{\mathsf{nontrivial}} \coloneqq K_Q - K_{\mathsf{trivial}}$ that may be ultimately viewed as an error term in our PSD analysis. It should be noted that natural construction of kernel-filling factors fall short in satisfying the above constraint, and this is motivates our recursive construction scheme for kernel-vectors.

On a high level, our recursive construction scheme starts from the natural kernel-vector of $r_{(S,\beta_S),T} = \sqrt{k}^{|S|}(-\mathbf{1}_{(S,\beta_S)} + \mathbf{1}_{S(,\beta_S),T})$, and note that this gives a sequence of (kernel) right-shapes in K_Q with $V_\sigma = |S \cup T|$ (ignoring the symmetry, otherwise we have a collection of left shapes obtained by the transpose of the above as well). However, this is not the correct construction as the collection of right-shapes may contain large norm terms. And the crux of our recursive construction scheme is to show that they are again lying in the kernel of Q, and can be removed with some other careful choice of kernel-factors.

En route to establishing this, our crude goal, as a guiding heuristic from observing the structural property of large norm shape obtained under the construction from the natural kernel vectors (after summing overall $S, T \subseteq [n]$), is to find a collection of vectors f in the kernel of Q corresponding to each kernel partition such that the terms (i.e. shapes) appearing in $L^T f_{(S,\beta_S),T}$ admit the following structural characterization (informally, ignoring color dependence):

1. Each term is a right-shape σ with $V_\sigma = S \cup T$;

2. S is an MVS that separates U_σ and S;

3. Each vertex in $S \cup T$ either has path to U_σ, or is a component connected to $T(V_\sigma)$ but not connected to $U_\sigma \cup S(V_\sigma)$;

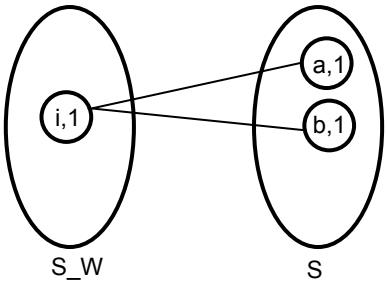
And to ensure $f_{(S,\beta_S),T}$ lies in the kernel of Q, our strategy is to write f as a linear combination of our kernel-basis vectors $r_{(S,\beta_S),T}$. We adopt a recursive construction for $f_{(S,\beta_S),T}$, and take

$$f_{(S,\beta_S)\cup T} \coloneqq r_{(S,\beta_S)\cup T} - \sum_{(S_W,\beta_{S_W})\cup T_W} f_{(S_W,\beta_{S_W})\cup T_W} \cdot \underbrace{R[(S_W,\beta_{S_W})\cup T_W, (S,\beta_S)\cup T]}_{\text{coefficient}}$$

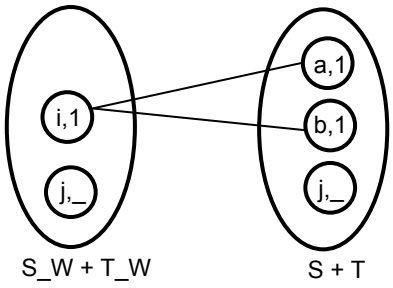
for some coefficient-function $R[(S_W,\beta_{S_W})\cup T_W, (S,\beta_S)\cup T]$ to be defined momentarily.

Concretely, we will take the coefficient function $R[(S_W,\beta_{S_W})\cup T_W, (S,\beta_S)\cup T]$ to be a sum of right-shapes that is additionally oblivious of the color in T_W. For intuition, consider the following example. Note the oval boundaries on the right corresponding to S and $S+T$ are intended to emphasize the kernel-partition in inspection.

Example 8.8.16. *Consider the kernel from fixing vertex i to have color 1, and summing over vertex j's colors. This can be written as $\mathbf{1}_{(i,1),j} - \mathbf{1}_{(i,1)}$ in our language. The multiplication with coefficient $R[(i,1)\cup j, (a,1,b,1)\cup j] = (\chi_{i,a}(G)\cdot\chi_{i,b}(G))$ for $a \neq b \in [n]$ (and $a,b \notin \{i,j\}$) can be seen as constructing the following vector $(\mathbf{1}_{(i,1),j} - \mathbf{1}_{(i,1)})\cdot(\chi_{i,a}(G)\cdot\chi_{i,b}(G))$ that is also in the kernel of Λ.*



(a) Vector Entry on S_W



(b) Vector Entry on $(S_W,\beta_{S_W})\cup T_W$

Figure 8.11: Kernel Vector with Kernel-Coefficient Multiplication

Remark 8.8.17. *We illustrate the above kernel vector in the diagram. In the diagram to the right,*

note that $(j, \_)$ vertex appears twice in both sides as to highlight that it appears in both T_W and T in our application.

Description of Kernel Shape for K_Q To motivate our recursive scheme for constructing the vectors, it is important that we understand what K_Q is to start with, and consider its expansion in the graph matrix basis. For simplicity, we restrict our attention to the unsymmetrized version $\widetilde{K}_Q$ obtained by $f_{(\cdot)} \otimes \mathbf{1}_{(\cdot)}$ alone. For starters, we observe that any term in $L^T f_{(S,\beta_S)\cup T}$ is a right shape σ with $V_\sigma = S \cup T$.

Consider the contribution of a fixed right shape σ to K_Q after summing over labels for its right boundary. Formally, putting back the color-dependence, for any color-right-shape (σ, CP) with $V_\sigma = S \cup T$ with β_S color of S fixed, its corresponding graph matrix of $\sum_{S,T\subseteq[n]} \sum_{\beta_S} \mathsf{M}_{\sigma,\mathrm{CP}}[\_, S \cup T] \otimes \mathbf{1}_{(S,\beta_S),T} = \mathsf{M}_{K(\sigma)}$ for the following kernel shape $K(\sigma)$.

Definition 8.8.18 (Kernel-shape $K(\sigma, \mathrm{CP})$)**.** *Given a right-shape σ with kernel-partition on $V_\sigma = S(V_\sigma) \cup T(V_\sigma)$, we define its corresponding kernel shape $K(\sigma, CP)$ as the following,*

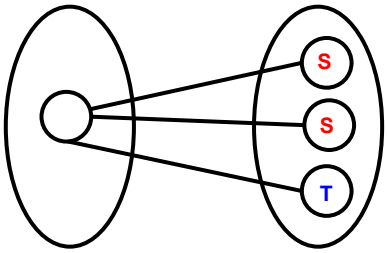
1. *We start with a shape σ and set $E(K(\sigma)) = E(\sigma)$ and $U_{K(\sigma)} \coloneqq U_\sigma$;*
2. *Include each vertex in $S(V_\sigma)$ into $V_{K(\sigma)}$;*
3. *The coloring of vertices from $V(\sigma)$ satisfy constraints from the color-partition of CP;*
4. *For each vertex v in $T(V_\sigma)$, create an extra copy of the vertex v' and connect it via a* dotted *line, include v' into $V_{K(\sigma)}$ where the dotted line corresponds to a* color-jump*, the two vertices connected have the same vertex label in $[n]$ though not necessarily of the same color.*

Claim 8.8.19. *For any right-color-shape (σ, CP) shape with $V_\sigma = S \cup T$,*

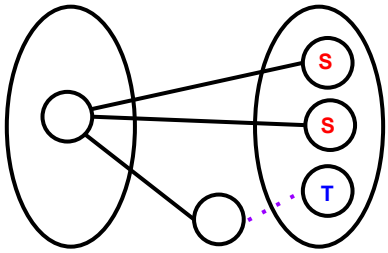
$$\sum_{S,T\subseteq[n]} \sum_{\beta_S : S\to[k]} \mathsf{M}_{\sigma,CP}[\_, S \cup T] \otimes \mathbf{1}_{(S,\beta_S),T} = \mathsf{M}_{K(\sigma),CP(\sigma)}$$

Proof. This follows by straightforward expansion of matrix multiplication as illustrated in the definition of $K(\sigma)$. □

We highlight the above construction with the following diagram for intuition. We start with a right-shape σ with V_σ containing 2 S vertices that have their colors fixed, and 1 T vertex whose color is being summed over. After applying the vector outer-product with $1_{(S,\beta_S),T}$, the vertices of S remain untouched, while we note that the vertex T has its label in $[n]$ pinned on the right-boundary while its color from σ may now vary (as we are summing over its color).



(a) Original right-shape σ



(b) Kernel-shape $K(\sigma)$ from vector outer-product

Troublesome Kernel Shape Illustrated For illustration, we claim the above shape $K(\sigma)$ appears at the natural construction of $r_{(S,\beta_S),T} = -\mathbf{1}_{S,\beta_S} + \mathbf{1}_{(S,\beta_S),T}$ for $|S| = 2, |T| = 1$. Importantly, note that we are interested in the resulting matrix after summing over all labels of $S, T \subseteq [n]$ as well as any coloring on S. For simplicity, we ignore the color-dependence and prescribe each edge in σ to be a pure-color edge, i.e., every vertex in σ receives the same color. Note that this constraint does not affect the T vertex in $T(V_{K(\sigma)})$, and it may receive an arbitrary color in $[k]$.

In our scaling so that we are constructing $Id_k \otimes Id_k \otimes \frac{J_k}{k}$ from the vector outer-product,

we have coefficient $\sqrt{k}^{|S|} \cdot \lambda_L(\sigma)_{\text{CP}}$ for the above kernel shape $K(\sigma)$, i.e.

$$\lambda_K(K(\sigma))_{\text{CP}} = \sqrt{k}^{|S(V_\sigma)|} \cdot \lambda_L(\sigma)_{\text{CP}} \cdot (\frac{1}{\sqrt{k}})^{|T|} = \sqrt{k}^2 \cdot (-1)^3 \cdot (\frac{1}{k})^{4-\frac{1}{2}} \cdot \frac{1}{\sqrt{k}}$$

where we recall that for a right-shape σ, we have a vertex factor of $(\frac{1}{k})^{|V(\sigma)|-\frac{|U_\sigma|}{2}}$, and note that crucially we have only a half vertex factor for vertices U_σ (hence the extra scaling of $\sqrt{k}^{|S|}$), and the extra $\frac{1}{\sqrt{k}}$ comes from vector outer-product with $(\frac{1}{\sqrt{k}})^{|T|}\mathbf{1}_{(S,\beta_S),T}$.

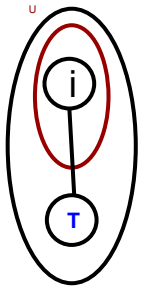
That said, we can see that kernel shape $K(\sigma)$ has matrix norm bound given by $\tilde{O}(\sqrt{n}^3\sqrt{k})$ where the extra factor of $\sqrt{k}$ comes from the freedom in picking color for the T vertex on the right boundary. Combining the above and setting the regime to be the dense regime of $p = \frac{1}{2}$, we have

$$\lambda_K(K(\sigma))_{\text{CP}} \cdot \tilde{\Theta}(\sqrt{n}^3\sqrt{k}) = \tilde{\Theta}(1) \cdot (\frac{1}{k})^{4-\frac{3}{2}} \cdot \sqrt{n}^3 \gg 1$$

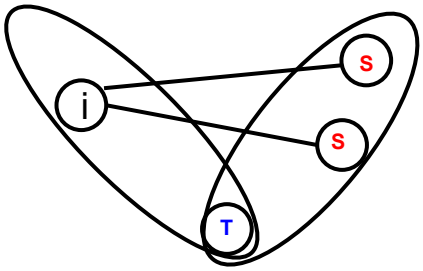
for $k \approx \sqrt{n}$ where the RHS is picked to be 1 as the candidate PSD mass from Q is Id. Therefore, this is a troublesome right-shape which we call reducible, i.e., we need to remove such using other kernel vectors. And this is showcased in the following.

Cancellation of Troublesome Right Shape In this case, we observe that the particular kernel-shape $K(\sigma)$ can be further decomposed as a product of right-shapes. For illustration, fix the vertex label on the left to be i, we may rewrite σ as the product of $\sigma = \sigma_1 \circ \sigma_2$. Note that $U_{\sigma_1} = \{i\}$.

Important to us is that σ_1 is a shape that satisfies our crude heuristic if we consider the kernel partition for σ_1 as $S(V_{\sigma_1}) = (i, \_)$ and $T(V_{\sigma_1}) = T$: in this case, $S(V_{\sigma_1})$ is indeed a MVS for $U_{\sigma_1} = i$ and $S(V_{\sigma_1}) = i$! That said, we can expect to find kernel-vector of the smaller size (reduced from 3 vertices to 2) on $\{i, T\}$ to generate the particular σ_1 under $L^T f_{(i,\_),T)}$ and view σ_2 as a coefficient!



(a) (Irreducible) Right-shape $\sigma_1 \in L^T f_{(i,\_),T}$



(b) Right-shape σ_2 as Coefficient

Figure 8.13: Cancellation for $K(\sigma) = \sigma_1 \circ \sigma_2$

Observing this example, we now want to show that any right-shape that violates the some crude structural property (capturing the above herustic) may be cancelled out by some term $\sum_{S_W,T_W} f_{S_W \cup T_W} \cdot R[S_W \cup T_W, S \cup T]$. In fact, we will need to show a bijection between terms in $L^T r_{S \cup T}$ (restricted to those that violate the above properties) and the terms from $\sum_{(S_W,\beta_{S_W}) \cup T_W} f_{(S_W,\beta_{S_W}) \cup T_W} \cdot R[(S_W,\beta_{S_W}) \cup T_W, (S,\beta_S) \cup T]$.

At this point, we remark that the above cancellation strategy is essentially asking for a further factorization of the component that makes the shape large norm, i.e., this exactly what approximate factorization via MVS is intended to achieve! Thus, we want to find a decomposition scheme for right shape that allows us to show the bijection. Another issue that we ignore so far is that we need to put back the color-dependence as our left-shape matrix L is consist of color-shape in this setting. This is formalized via the following decomposition scheme for right shape.

8.8.3 Irreducible Decomposition of Right Shape

We now describe a decomposition procedure that applies to any shape σ with kernel partition on $V_\sigma = S \cup T$. Our goal is to identify $\sigma = \sigma_A \circ \sigma_B$ such that $V_{\sigma_A} = U_{\sigma_B} = S_W \cup T_W$ for some $T_W \subseteq T$, and moreover, T_W is not incident to any edge in σ_B .

Finding Kernel (Rainbow) Separator (aka. Kernel Right Shape Decomposition)
Given a color-shape σ with color-partition CP_σ such that $(\sigma, \mathrm{CP}_\sigma)$ is a (color-)right shape in L^T, (i.e. σ is a right shape and any edge incident to U_σ is a pure-color edge under CP) and V_σ is additionally partitioned into $V_\sigma = S(V_\sigma) \cup T(V_\sigma)$, we define its irreducible rainbow separator and decomposition $(\sigma, \mathrm{CP}_\sigma) = (\sigma_A, \mathrm{CP}_A) \circ (\sigma_B, \mathrm{CP}_B)$ for σ_A, σ_B both color-right-shapes as the following:

1. Initialize the shape $B = \sigma$ with edges $E(B) = E(\sigma)$, and $U_B = U_\sigma, V_B = V_\sigma$.
2. Apply alternately the following operation of *Rightwards MVS Subtraction* or *Rightwards Rainbow Extension*, and *Update B Operation* until B converges (i.e. the shape B does not change after applying both extension operations)
    - *Rightwards MVS Extension*: let S_R be the rightmost MVS separating U_B and $S(V_\sigma)$;
    - *Rightwards Rainbow Extension*: let $Rv(U_B)$ be the vertices of rainbow components incident to U_B, and let S_R be the set of vertices reachable from $S(V_\sigma)$ without passing through any other vertex in $Rv(U_B)$.
    - *Update B*: Update $E(B) \subseteq E(\sigma)$ to be the subset of edges reachable from $S(V_\sigma)$ without passing through any vertex in S_R. Update $U_B \coloneqq S_R$.

We are now ready to define the kernel-separator W and shape σ_A, σ_B alongside their color-partitions in the kernel-decomposition as the following,

1. Let $E(\sigma_B) = E(B)$ be the set of edges that can be reached from $S(V_\sigma)$ without passing through any vertex in S_W, and let $E(\sigma_A) = E(\sigma) \setminus E(B)$;
2. *Update* T_W: Initialize $T_W \coloneqq \emptyset$. Add a vertex v from $T(V_\sigma) \setminus S_W$ into T_W

- if it is a vertex reachable from U_σ without using any edge in $E(B)$; [a]
- it is a vertex not connected to $U_\sigma \cup S(V_\sigma)$ but incident to edges. (In other words, the vertex is part of an edge-component around $T(V_\sigma)$).

3. Call $W \coloneqq S_W \cup T_W$ the kernel-separator for σ. And note that $S_W \cup T_W$ is also the kernel-partition for W.

4. Completing their boundary definition as the following: let $U_{\sigma_A} \coloneqq U_\sigma, V_{\sigma_A} = U_{\sigma_B} \coloneqq W = S_W \cup T_W$ (with kernel partition given by $S_W \cup T_W$), and $V_{\sigma_B} = V_\sigma$. [b]

[a] crucially, this shall be contrasted to vertices reachable without passing through S_W. See diagram.
[b] Shape B and σ_B have the same set of edges and right-boundary, except that the left-boundary U_{σ_B} contains vertices from $U_B = S_W$ and extra vertices in $T_W \subseteq U_{\sigma_B} \cap T(V_B)$ that are held dormant in B.

With the above procedure for decomposition, we make the following observations.

Observation 8.8.20. *Both S_W and T_W may be empty. Moreover, W may be empty.*

Observation 8.8.21 (Vertex's kernel partition status may flip)**.** *It is possible for a vertex to start from $v \in T(V_\sigma)$ and becomes an S-vertex $v \in S_W$ in the decomposition, however T_W remains a subset of T.*

Observation 8.8.22 (Vertices in T_W are dormant in σ_B)**.** *$T_W \subseteq U_{\sigma_B} \cap V_{\sigma_B}$ is not incident to any edge in σ_B.*

Claim 8.8.23 (Vertices in T_W are dormant in σ_B)**.** *$T_W \subseteq T(V_\sigma)$. Moreover, $T_W \subseteq U_{\sigma_B} \cap V_{\sigma_B}$ is not incident to any edge in σ_B.*

Proof. $T_W \subseteq T(V_\sigma)$ is immediate as each vertex is added to T_W from $T(V_\sigma) \setminus S_W$. Recall that any edge in σ_B is an edge that can be reached from $S(V_\sigma)$ without passing through S_W, hence this is also immediate for a vertex added to T_W as a vertex not connected to $U_\sigma \cup S(V_\sigma)$ but incident to edges. Finally, since S_W is a separator for U_σ and $S(V_\sigma)$, any vertex reachable from U_σ without using any edge in $E(B)$ is in S_W. However, a vertex may only get added from $T(V_\sigma) \setminus S_W$. This proves the claim. □

Proposition 8.8.24 (Self-cancellation property for trivial T_W)**.** *For any right color-shape $(\sigma, \beta_\sigma) = (A, \beta_A) \circ (B, \beta_B)$ with $W = V_A = S_W \cup T_W$ as its kernel partition, if $T_W = \emptyset$, then we have U_σ being the unique MVS separating U_σ and $S(V_\sigma)$.*

Moreover, let σ' be the shape with

1. *$E(\sigma') \coloneqq E(\sigma)$, and identical left-boundary $U_\sigma = U_{\sigma'}$;*
2. *The right-boundary is obtained by removing $T(V_\sigma)$ vertices, i.e., $V_{\sigma'} \coloneqq V_\sigma \setminus T(V_\sigma)$;*
3. *Remove any deg-0 vertex in $V(\sigma') \setminus U_{\sigma'} \cup V_{\sigma'}$ from $V(\sigma')$;*

σ' continues to be a (color-)right shape in L^T.

Proof. We first give a proof that highlights the main idea by ignoring the color-dependence, and then show it continues to hold with color-dependence. In the former case, $W = S_W$ is taken to be the rightmost MVS separating $S(V_\sigma)$ and U_σ (with no further rightwards-extensions). Since U_σ is also separating U_σ and $S(V_\sigma)$, we have $|U_\sigma| \geq |W|$.

We now observe that S_W is also a separator for U_σ and $T(V_\sigma)$: for any vertex $v \in T(V_\sigma) \setminus S_W$, if there is a path from U_σ to v without passing through v, such vertex v is by construction included into T_W, giving us a contradiction. For vertex $v \in T(V_\sigma) \cap S_W$, S_W by definition separates any path from U_σ to v. Therefor, S_W is also a separator for U_σ and $T(V_\sigma)$, hence a separator for U_σ and $V_\sigma = S(U_\sigma) \cup T(V_\sigma)$ as well. Since σ is a right-shape, U_σ is the unique MVS separating U_σ and V_σ, hence we have $|W| \geq |U_\sigma|$ with equality only if $W = U_\sigma$. Combining the above yields us $W = U_\sigma$. Since W is taken to be the Rightmost MVS separating U_σ and $S(V_\sigma)$, U_σ is also the leftmost separator separating U_σ and $S(V_\sigma)$, we conclude. that U_σ is the unique MVS separating U_σ and $S(V_\sigma)$.

To incorporate the color-dependence, it suffices to notice U_σ is also a rainbow-MVS as σ is a right-shape, and hence U_σ is not incident to any cross-color edge in (σ, CP). To see the uniqueness, we first observe that S_W remains as a separator for U_σ and $T(V_\sigma)$

by the same argument as above. Let S_1 be the original RMVS separating U_σ and $S(V_\sigma)$ that ultimately evolves into S_W after potentially non-trivial rightwards extensions. Notice rightwards extension may only unblock path from U_σ to $T(V_\sigma)$, hence if S_W is a separator for U_σ and $T(V_\sigma)$, S_1 is also a separator for U_σ and $T(V_\sigma)$. By the same argument earlier, we can see we have $S_1 = U_\sigma$, and thus we can deduce there is no non-trivial rightwards extension, and again U_σ is the unique MVS separating U_σ and $S(V_\sigma)$.

Finally to see σ' is a right-shape, it suffices to verify there is no floating component in σ', as otherwise $U_{\sigma'}$ being the unique color MVS and any edge-component being connected to $V_{\sigma'}$ are sufficient condition to guarantee σ' being a right-shape. And there is no floating component in σ' as component connected to $U_\sigma \cup S(U_\sigma)$ remains connected to the left/right boundary in σ', while there cannot be edge-component connected to only $T(V_\sigma)$ while not to $U_\sigma \cup S(U_\sigma)$ as any such component's incidental T vertex is automatically put in T_W, and we have a contradiction here.

□

Corollary 8.8.25. *Any color-shape $(\sigma, \beta_\sigma) = (A, \beta_A) \circ (B, \beta_B)$ with kernel-separator $T_W = \emptyset$ also has $S_W = U_A$ and $A = Id$.*

Canonical Decomposition of a Kernel Shape We now introduce some properties of the shapes obtained from the decomposition.

Definition 8.8.26 (Irreducible right shape via decomposition)**.** *Call a right color-shape $(\sigma, \beta_\sigma) = (A, \beta_A) \circ (B, \beta_B)$ irreducible if $B = Id_{(B,\beta_B)}$ from the above process.*

Observation 8.8.27. *In the simple case of no color-dependence, i.e., each edge is pure-color, σ is irreducible if $S(V_\sigma)$ is an MVS separating U_σ and $S(V_\sigma)$.*

We would like to point out that with the above definition of irreducible right-shapes, *Id* is considered to be irreducible. That said, we may further distinguish it with the following definition.

Definition 8.8.28 (Trivial and non-trivial irreducible right-shapes)**.** *The Id-shape with $U_\sigma = V_\sigma = V(\sigma)$ and $rainbow(E(U_\sigma \cap V_\sigma)) = \emptyset$ is called trivial. Any other shape is called non-trivial.*

Definition 8.8.29 (Kernel coefficient (right-) shape)**.** *For any right-shape σ with kernel partition on both $U_\sigma = S(U_\sigma) \cup T(U_\sigma)$ and $V_\sigma = S(V_\sigma) \cup T(V_\sigma)$, and $T(U_\sigma), T(V_\sigma) \neq \emptyset$, , we call it a kernel-coefficient shape if*

1. *It is a right-shape, U_σ is the unique MVS in σ; moreover, $S(U_\sigma)$ is the unique MVS separating $S(U_\sigma)$ and $S(V_\sigma)$;*

2. *It is not identity (the trivial shape);*

3. *Any vertex in $T(V_\sigma) \setminus U_\sigma$ is either a deg-0 vertex, or can be reached from $S(V_\sigma)$ without passing through U_σ ;*

4. *$T(U_\sigma) \subseteq T(V_\sigma)$.*

When a color-partition CP is given, we additionally require there to be no rainbow edge incident to U_σ.

We additionally let $\mathcal{R}$ be the collection of kernel-coefficient-right-shapes, and define its matrix as

$$R[(S_W, \beta_{S_W}) \cup T_W, (S, \beta_S) \cup T] :=$$
$$\sum_{\substack{(\sigma, CP) \in \mathcal{R} \\ \text{kernel-coefficient shape for} \\ U_\sigma = (S_W, \beta_{S_W}) \cup T_W \\ V_\sigma = (S, \beta_S) \cup T}} \sum_{\beta_T : T \to [k]} \lambda_R(\sigma)_{CP} \cdot \mathsf{M}_{\sigma, CP}[(S_W, \beta_{S_W}) \cup (T_W, \beta_T(T_W)), (S, \beta_S) \cup (T, \beta_T)]$$

Notice that R has indices that does not depend on color of T_W and T throughout, while we sum over the colors of T in its definition for notational correctness due to the matrix $\mathsf{M}_{\sigma,\mathrm{CP}}$ are indexed with each vertex receiving some color.

Proposition 8.8.30 (Decomposition of a right shape)**.** *For any right (color-)shape* $(\sigma, \beta_\sigma) = (A, \beta_A) \circ (B, \beta_B)$ *decomposed by the above process with color-partition,* (A, β_A) *is an irreducible right-shape (possibly trivial), and* (B, β_B) *is either identity (in the case of* σ *being irreducible) or a kernel-coefficient right-shape.*

Proof. We first show A is an irreducible right-shape. By definition of an irreducible right shape, we are to show $(A, \beta_A) = (A_L, \beta_{A_L}) \circ (A_R, \beta_{A_R})$ from our kernel-decomposition process gives $A_R = Id$. That said, consider the track of the rightwards extension process, and observe that in the base level $A = A_0$ with S_{A_0} being the rightmost MVS for U_A and $S(V_A)$. The subsequent rightwards extensions move S_{A_0} to the right so that we arrive at $S_A = V_A$ at the end of the process, and consider the decomposition according to the iterations

$$A = \underbrace{A_0 \circ X_1 \circ A_1 \circ X_2 \circ \cdots \circ A_t \circ X_t}_{A_L} \circ (A \setminus A_L)$$

for A_i, X_i rightwards-extension-shapes. We then observe that applying decomposition process on σ undergoes the same iterative process up until obtaining A_L, as we are applying the same process with the same starting point of A_0 (rightmost MVS for U_A and $S(V_A)$ remains invariant under composition with a kernel-coefficient shape). Since the process does not converge until $S_A = V_A$, the same extension process for A does not converge until we have $V_{A_L} = V_A$, giving us $A_R = A \setminus A_L$ being identity as desired.

To see B being either an identity or kernel-coefficient-right-shape, it suffices to verify the conditions that B is a right-color-shape, and any vertex in $T(V_B) \setminus T(U_B)$ is a degree-0 vertex or a vertex reachable from $S(V_B)$ in B without passing through U_B.

We start by verifying any vertex in $T(V_B) \setminus U_B$ is either a deg-0 vertex from $T(V_B) \setminus U_B$, or a vertex reachable from $S(V_\sigma)$ without passing through U_B. Notice if the vertex is not connected to $S(V_\sigma) \cup U_B$, yet not degree-0, such vertex is put into T_A and added to U_B. Therefore, it suffices for us to consider the vertex connected to $S(V_\sigma) \cup U_B$. If the vertex

cannot be reached from $S(V_\sigma)$ without passing through U_B yet connected to $S(V_\sigma)$, there must be a path from from U_B to the vertex, moreover, the path does not use any edge reachable from $S(V_\sigma)$ without passing through S_W. However, we have a contradiction here as any such T vertex is added to T_A, and this proves the desired connectivity condition.

Finally we show B is a color-right-shape. Recall that U_B is obtained from adding $T(V_B) \setminus S_A$ as T vertices to S_A. By construction, S_A is the unique MVS separating in $E(B)$ S_A and $S(V_\sigma)$, therefore it is also the unique MVS separating S_A and $S(V_\sigma) \cup T_S$ where $T_S \subseteq T(V_\sigma)$ is the set of vertices reachable from $S(V_\sigma)$ in B without passing through S_A. Since any added vertex to both boundaries is in the mandatory separator, $S_A \cup T_A$ is also a unique MVS for $S(V_\sigma) \cup T_S \cup T_A$. Finally, observe that $T_S \cup T_A = T(V_\sigma)$ as any vertex from $T(V_\sigma)$ not added to T_A is either a degree-0 vertex in $T(V_\sigma)$, or a vertex that can be reached from $S(V_\sigma)$ without passing through S_B (in which case added via T_A). Moreover, note that adding T_I a set of deg-0 vertex to $S(V_\sigma) \cup T_S \cup T_A \cup T_I$ preserves $S_A \cup T_A = U_B$ being a unique MVS for both sides, and note that the right boundary is exactly $S(V_\sigma) \cup T(V_\sigma)$ now. Therefore, we have shown B is a right-shape. (It is immediate that it also satisfies color-constraints and hence a right-color-shape) □

Definition 8.8.31 (Scaling for Kernel-Coefficient)**.** *Let $\lambda_R(\sigma)_{CP}$ be the kernel-coefficient for (σ, CP) in R, we define it as*

$$\lambda_R(\sigma)_{CP} \coloneqq (\frac{1}{k})^{|V(\sigma)|-|U_\sigma|} \cdot Ev(\sigma)_{CP}$$

We pay attention to the vertex-factor scaling here.

Claim 8.8.32. *For $(\sigma, \beta_\sigma) = (A, \beta_A) \circ (B, \beta_B)$ in the kernel-decomposition,*

$$\lambda_L(\sigma)_{CP} = \lambda_L(A)_{CP(A)} \cdot \lambda_R(B)_{CP(B)},$$

where λ_L is the coefficient in the left/right-shape L matrix.

Proof. Recall that in the right-shape matrix σ, its coefficient is given by

$$\lambda_L(\sigma)_{\mathrm{CP}} = (\frac{1}{k})^{|V(\sigma)|-\frac{|U_\sigma|}{2}} \cdot Ev(\sigma)_{\mathrm{CP}}$$

Similarly, we have

$$\lambda_L(A)_{\mathrm{CP}(A)} = (\frac{1}{k})^{|V(A)|-\frac{|U_A|}{2}} \cdot Ev(A)_{\mathrm{CP}(A)}$$

By the above definition, we have

$$\lambda_R(B)_{\mathrm{CP}(B)} \coloneqq (\frac{1}{k})^{|V(B)|-|U_B|} \cdot Ev(B)_{\mathrm{CP}(B)}$$

Since $V_A = U_B$ is a rainbow separator, there is no rainbow component across A and B, and since our coefficient factorizes over rainbow component and pure-color edges, we immediately have

$$Ev(A)_{\mathrm{CP}(A)} \cdot Ev(B)_{\mathrm{CP}(B)} = Ev(\sigma)_{\mathrm{CP}}\,,$$

and our focus shall be on the vertex-factors. This comes down to verifying

$$(\frac{1}{k})^{|V(A)|-\frac{|U_A|}{2}} (\frac{1}{k})^{|V(B)|-|U_B|} = (\frac{1}{k})^{|V(\sigma)|-\frac{|U_\sigma|}{2}}\,.$$

This follows as we have

$$|V(A)| + |V(B)| - \frac{|U_A|}{2} - |U_B| = |V(\sigma)| - \frac{|U_A|}{2} = |V(\sigma)| - \frac{|U_\sigma|}{2}\,,$$

where we use $|V(A)| + |V(B)| = |V(\sigma)| + |U_B|$ and $U_A = U_\sigma$. □

Defining Kernel-Filling Vectors Finally, with the construction of R matrix that serves as the coefficient for our the kernel-vectors, we are now ready to define our kernel-filling vectors for any kernel-partition.

Definition 8.8.33 (Defining Kernel-Filling Vectors f)**.** *We give a recursive construction based on the kernel-partition, for any $((S, \beta_S), T)$, we define*

$$f_{(S,\beta_S),T} \coloneqq r_{(S,\beta_S),T} - \sum_{\substack{W \subseteq [n], \\ W = S_W \cup T_W \\ \beta_S : S_W \to [k] \\ |W| \leq |S \cup T| \\ \emptyset \neq T_W \subseteq T}} f_{(S_W,\beta_S),T_W} \cdot R[(S_W, \beta_{S_W}) \cup T_W), (S, \beta_S) \cup T]$$

where we remind the reader that R is the kernel-coefficient matrix defined as

$$R[(S_W, \beta_{S_W}) \cup T_W, (S, \beta_S) \cup T] \coloneqq$$
$$\sum_{\substack{(\sigma, CP) \in \mathcal{R} \\ \text{kernel-coefficient shape for} \\ U_\sigma = (S_W, \beta_{S_W}) \cup T_W \\ V_\sigma = (S, \beta_S) \cup T}} \sum_{\beta_T : T \to [k]} \lambda_R(\sigma)_{CP} \cdot \mathsf{M}_{\sigma,CP}[(S_W, \beta_{S_W}) \cup (T_W, \beta_T(T_W)), (S, \beta_S) \cup (T, \beta_T)]$$

and

$$r_{(S,\beta_S),T} = \sqrt{k}^{|S|} \cdot (-\mathbf{1}_{(S,\beta_S)} + \mathbf{1}_{(S,\beta_S),T}) \,.$$

Remark 8.8.34. *We remind the reader that the $\sqrt{k}^{|S|}$ scaling is picked such that $L^T f_{(S,\beta_S),T} \approx (\frac{1}{\sqrt{k}})^{|T|} 1_{(S,\beta_S),T}$. Since the identity shape has $\lambda_L(Id_{S \cup T}) = (\frac{1}{\sqrt{k}})^{|S \cup T|}$, we use the extra scaling of $\sqrt{k}^{|S|}$ to align back to the candidate kernel from $(\frac{1}{\sqrt{k}})^{|T|} 1_{(S,\beta_S),T}$ whose outer product corresponds to the kernel due to $Id^S \otimes (\frac{J}{k})^{|T|}$ (up to ordering).*

Before we show the desired equality, we first observe that the above construction process is well-defined. In particular, when defining $f_{(S,\beta_S),T}$, we observe that the coefficient from matrix R is 0 in most places, and ultimately the construction only depends on the

f function with $|W| < |S \cup T|$. By inducting on $|S \cup T|$, we ensure each involved $f_{S_W \cup T_W}$ vector is constructed before defining $f_{(S,\beta_S),T}$.

With this definition, we are ready to introduce our main lemma that says $L^t f_{(S,\beta_S),T}$ gives us the following collection of right-shapes that would ultimately allow us to obtain a good approximation of K_{trivia} via K_Q.

Proposition 8.8.35 (Informal of lemma 8.8.43 by ignoring intersection shapes on both sides)**.** *We have*

$$L^T f_{(S,\beta_S),T} = \sum_{\substack{\sigma,CP:\text{non-trivial, irreducible} \\ V_\sigma = S \cup T}} \lambda_L(\sigma)_{CP} \cdot \sqrt{k}^{|S|} \cdot \mathsf{M}_{\sigma,CP}[\star, (S,\beta_S) \cup T] + \left(\frac{1}{\sqrt{k}}\right)^{|T|} \mathbf{1}_{(S,\beta_S),T}$$

holds up to intersection terms on both sides for the f vector defined in definition 8.8.33, i.e.,

$$f_{(S,\beta_S),T} := r_{(S,\beta_S),T} - \sum_{\substack{W \subseteq [n], \\ W = S_W \cup T_W \\ \beta_{S_W}: S_W \to [k] \\ |W| \leq |S \cup T| \\ T_W \neq \emptyset}} f_{(S_W,\beta_S),T_W} \cdot R[(S_W \cup T_W, \beta_{S_W \cup T_W}), (S \cup T, \beta_{S \cup T})] \,.$$

In words, the RHS of the equality of $L^T f_{(S,\beta_S),T}$ is best understood by considering it as a collection of right-shapes σ satisfying the irreducible property (including the trivial shape on $S \cup T$) with fixed right boundary $V_\sigma = S \cup T$ for any right-boundary coloring $\beta : V_\sigma \to [k]$ such that the colors restricted to S is given by β_S, i.e. $\beta(S) = \beta_S$.

We refer the reader to the formal statement in lemma 8.8.43 for the inclusion of intersection terms, and give the proof when restricted to proper shapes without intersections at this section for clarity of our presentation.

Proof. For convenience, we call $L^T \sum_{\substack{W \subseteq [n], \\ W = S_W \cup T_W \\ \beta_S: S_W \to [k] \\ |W| \leq |S \cup T| \\ T_W \neq \emptyset}} f_{(S_W,\beta_S),T_W} \cdot R[(W, \beta_W), (S \cup T, \beta_{S \cup T})]$ *com-*

posable terms. Throughout this section, it is convenient to consider $S, T \subseteq [n]$ as fixed with color β_S also fixed while we sum over the color of T. Unpacking the definition for $r_{(S,\beta_S)\cup T} = \sqrt{k}^{|S|}\mathbf{1}_{(S,\beta_S),T} - \sqrt{k}^{|S|}\mathbf{1}_{(S,\beta_S)}$, it is equivalent to showing

$$\sqrt{k}^{|S|} \cdot L^T \mathbf{1}_{(S,\beta_S),T} = \sqrt{k}^{|S|} \cdot L^T \mathbf{1}_{(S,\beta_S)} + \text{Composable Terms} + \text{Irreducible Terms}$$

Remark 8.8.36. *For intuition, the reader should think of Irreducible Terms as the desired kernel for the corresponding kernel-partition. It contains the desired all-*1 *component, but also contains extra terms that are necessary.*

Set-up We first point out that we are considering vector equality here, and each term may be viewed as obtained from starting with a ribbon R with fixed right-boundary, i.e., $V_R = (S \cup T, \beta_S \cup \beta_T)$ or $V_R = (S, \beta_S)$ in the case of $L^T 1_{S,\beta_S}$ and remove its right boundary V_R. Therefore, it is equivalent for the purpose of the vector equality that we transform the ribbon with original right boundary $V_R = (S, \beta_S)$ into a ribbon with right boundary of $V_R = (S \cup T, \beta_S \cup \beta_T)$, and this allows us to focus solely on ribbons with the same right boundary.

To handle this discrepancy, for any ribbon R with $V_R = S$, consider the set of vertices $T_I \subseteq T(V_\sigma) = T$ such that T_I are not incident to any edge in $E(R)$, add T_I as deg-0 vertex into $V(R)$, and finally modify the right boundary from (S, β_S) to $(S \cup T, \beta_S \cup \beta_T)$. Formally we have

$$\lambda_L(R)_{\beta_R} \mathsf{M}_{R,\beta(R)}[(U_R, \beta_{U_R}), (S, \beta_S)] = \sum_{\beta(T_I)} \lambda_L(R \cup T_I) \cdot \mathsf{M}_{R\cup T_I, \beta(R)\cup\beta(T_I)}[(U_R, \beta_{U_R}), (S \cup T, \beta_S \cup \beta_T)]$$

where we remind the reader that adding deg-0 vertex outside the left/right boundary

does not affect the graph matrix for a ribbon, i.e.

$$\mathsf{M}_{R,\beta(R)} = \mathsf{M}_{R\cup T_I,\beta(R)\cup\beta(T_I)}$$

With this transformation, it remains for us to show a bijection among ribbons with right boundary being $(S \cup T, \beta_S \cup \beta_T)$ in the above equality. We prove this by strong induction by inducting on $|S \cup T|$. At this point, it is convenient for us to ignore the labeling of vertices in $[n]$ and consider shapes, except noting the right boundary have labels given by $S \cup T$.

Partitioning the RHS by W Recall our desired equality of

$$(\frac{1}{\sqrt{k}})^{|T|} \cdot L^T \mathbf{1}_{(S,\beta_S),T} = (\frac{1}{\sqrt{k}})^{|T|} \cdot L^T \mathbf{1}_{(S,\beta_S)} + \text{Composable Terms} + \text{Irreducible Terms}\,.$$

Viewing the RHS as shapes with right boundary being $(S \cup T, \beta_S \cup \beta_T)$, we note that the three terms in the equation can be viewed as partitioning based on the intermediate separator $W = S_W \cup T_W$ by the right-shape-decomposition process as the following,

1. $L^T \mathbf{1}_{S,\beta_S}$ corresponds to right shapes with intermediate separators W such that $T_W = \emptyset$;

2. For right shapes with $T_W \neq \emptyset$, we further case on whether $W = S \cup T$, if so, this is a irreducible shape;

3. Otherwise, we have $W \neq S \cup T$ with $T_W \neq \emptyset$, and this is a shape in the composable term.

And at the same time, the LHS can be viewed as right-shapes from L^T with right boundary being $(S \cup T, \beta_S \cup \beta_T)$.

The bijection restricted to the first two terms can be proven directly as the following, and we prove the bijection for composable term via an induction on $|S \cup T|$.

Bijection for Shapes with $T_W = \emptyset$ and Intermediate Separator $W = S \cup T$

Claim 8.8.37. *There is a bijection between shapes on the LHS with $T_W = \emptyset$, and shapes in $L^T \mathbf{1}_{S,\beta_S}$.*

Proof. The injective map from the LHS to the RHS follows by the self-cancellation property in proposition 8.8.24. The other direction follows by observing that for each right-shape with $V_\sigma = S$, let σ' be the shaped obtained by adding vertices from T to $V_{\sigma'}$ preserves the shape being a right-shape: as any separator with added vertices into V_σ is also a separator for the original right-shape. Finally, notice that applying the decomposition process on σ' yields an intermediate separator W with $T_W = \emptyset$, since $U_{\sigma'} = S_W$ is also the unique MVS for $U_{\sigma'}$ and $S(V_{\sigma'}) = V_\sigma$), and each vertex in $T(V_{\sigma'})$ is either reachable from $S(V_\sigma)$ without passing U_σ since it appears as a vertex in $V(\sigma) \setminus (U_\sigma \cup V_\sigma)$, or it is a deg-0 vertex outside the boundary in σ (i.e. it is a vertex that gets added), and such vertex is not in T_W by our decomposition (factored out by the kernel-coefficient shape). Hence, we have shown a bijection between $T_W = \emptyset$ and $L^T \mathbf{1}_{S,\beta_S}$.

□

Claim 8.8.38. *There is also a bijection between ribbons on the LHS with $W = S \cup T$ and (non-intersected) shapes in the irreducible terms.*

Proof. The bijection on shapes with $W = S \cup T$ follows by definition of irreducible shape with $B = Id$ in the decomposition. □

Bijection for $W \neq S \cup T$ with $T_W \neq \emptyset$ and Composable Terms

Lemma 8.8.39. *There is a bijection between shapes in the composable terms on the RHS, and shapes with intermediate kernel separator $W \neq S \cup T$ with $T_W \neq \emptyset$ on the LHS.*

Proof. We prove this by induction on $|S \cup T|$. The base case is when $|S \cup T| = 1$, in particular, $S = \emptyset$, and the hypothesis holds trivially as

1. Any right shape with $V_\sigma = T$ has the intermediate separator $W = T = S \cup T$, and any shape of this form is in the irreducible term. In other words, there are no shape with $W \neq S \cup T$ and $T_W \neq \emptyset$.

2. As for the composable terms, notice there are no kernel-coefficient term with $|V_\sigma| = |S \cup T| = 1$ as we require $T_W \subseteq W \neq \emptyset$ and $|W| < |S \cup T|$, hence there are no composable terms either.

For the induction case, we assume the hypothesis appleid on $L^T f_{S_W, T_W}$ holds for any W with $T_W \neq \emptyset$ and $|W| < |S \cup T|$. From LHS to RHS, notice this holds by construction as for a shape $(\sigma, \beta) = (A, \beta_A) \circ (B, \beta_B)$ from the kernel-decomposition process with intermediate separator $W = V_A \neq V_\sigma$ and $T_W \neq \emptyset$, there is a unique term on the RHS from the corresponding W where A is obtained from $L^T f_{S_W, T_W}$ as an irreducible term and B is a kernel-coefficient term with $U_B = W$ as $W \neq V_\sigma$.

For the other direction, for any irreducible term A and kernel-coefficient-shape B with A, B additionally consistent on the kernel-partition on W, observe trivially that $R = A \circ B$ is also a ribbon on the LHS while it remains to verify that R is only obtained from A and B on the RHS, i.e., applying the decomposition process on $A \circ B$ gives us the decomposition via the intermediate separator W with prescribed kernel partition.

Recall that intermediate separator is built from the rainbow-MVS separating U_A and $S(V_B)$, it suffices for us to show $S_W = S_{W'}$. In fact, we will show that these two separators are built from the same base separator $S_{W_0} = S_{W_0'}$ before any rightward extension process. Let X be the set of vertices in W' that can be reached from U_A without passing through

S_{W_0}, and Z the vertices in W that can be reached without passing through $S_{W_0'}$, and let $Y = S_{W_0} \cap S_{W_0'}$. Analogously, let C be the set of vertices in W that can be reached from $S(V_B)$ without passing through $S_{W_0'}$, and let D be the set of vertices in $S_{W_0'}$ that can be reached from $S(V_B)$ without passing through W. We observe the following. Note that we have $S_{W_0} = C \cup Y \cup Z$ and $S_{W_0'} = X \cup Y \cup D$.

1. $|X| \leq |C|$ with equality only if $X = C$: $S_{W_0'} = X \cup Y \cup D$ is a unique MVS separating $S_{W'}$ and $S(V_B)$. At the same time, $C \cup Y \cup D$ is also separating $S_{W_0'}$ and $S(V_B)$;

2. $|Z| \leq |D|$ with equality only $Z = D$: $S_W = C \cup Y \cup Z$ is a unique MVS separating S_{W_0} and $S(V_B)$. At the same time, $C \cup Y \cup D$ is also separating these two sets;

3. Unless both of the above hold as equalities, $X \cup Y \cup Z$ is a smaller separator separating U_A and $S(V_B)$, and we have a contradiction here as S_{W_0} and $S_{W_0'}$ both start as the RMVS separating U_A and $S(V_B)$. Therefore, we have shown $X = C, Z = D$ and $S_{W_0} = S_{W_0'}$.

4. We then observe that by construction of our rightwards extension process, this guarantees $S_W = S_{W'}$, and additionally $T_W = T_{W'}$.

This completes the proof to our bijection. □

□

Intersection Patterns Put Back As foreshadowed in the previous proposition, there are intersection terms that arise in the construction of $L^T f$ as we are essentially applying graph matrix multiplication again, and intersection beyond the boundary injectivity are expected. More interestingly, it turns out that our constructed kernel $L^T f_{(S,\beta_S),T}$ will contain intersection terms as they are truly part of the kernel of Q.

We now give the full characterization for shapes that appear in our final result of $L^T f_{(S,\beta_S),T}$.

Formal Irreducible Kernel for Q

Definition 8.8.40 (Intersection Shape, and Intersection Pattern)**.** *Given a (possibly improper) shape α obtained from t-way product of shape $\tau \circ \gamma_1 \circ \gamma_2 \cdots \circ \gamma_t$, we call α an intersection shape. For each shape α, we let $P(\alpha)$ denote its corresponding intersection pattern defined s the following. Intersection pattern $P(\alpha)$ is a graph on $V(\tau \circ \gamma_1 \circ \gamma_2 \cdots \circ \gamma_t)$ such that for each γ_i, it specifies how vertices in $V(\gamma_i) \setminus U_{\gamma_i}$ intersects with vertices in prior level, i.e. vertices in $V(\tau \circ \gamma_1 \circ \cdots \circ \gamma_{i-1})$*

.

Remark 8.8.41. *Any proper shape α is also trivially an intersection shape.*

Definition 8.8.42 (Valid Intersection Profile for Kernel)**.** *For each intersection pattern P_α with intersection shape $\alpha = \tau \circ \gamma_1 \circ \cdots \circ \gamma_t$. We call it a valid intersection profile if each γ_i is a right-intersection shape such that ,*

1. *$S(U_{\gamma_i})$ is the unique color-MVS separating $S(U_{\gamma_i})$ and $S(V_{\gamma_i})$ for γ_i;*
2. *and $S(V_{\gamma_i})$ is a color-MVS for $S(U_\gamma) \cup Int(\gamma_i)$ and $S(V_{\gamma_i})$ where $Int(\gamma_i)$ is the set of vertices in γ_i that intersect with vertices in previous level beyond the necessary boundary in U_{γ_i};*
3. *$T(U_{\gamma_i}) \subseteq T(V_{\gamma_i})$, and any vertex in $T(U_{\gamma_i})$ is not incident to any edge; any vertex in $T(V_{\gamma_i}) \setminus T(U_{\gamma_i})$ is reachable from $S(V_{\gamma_i})$ without passing through $S(U_{\gamma_i})$.*
4. *In other words, γ_i is a kernel-coefficient shape in $\mathcal{R}$ for $(S(U_{\gamma_i}), \beta_{S(U_{\gamma_i})}) \cup T(U_{\gamma_i})$ and $(S(V_{\gamma_i}), \beta_{S(V_{\gamma_i})}) \cup T(V_{\gamma_i})$*

and additionally,

1. *τ is a proper irreducible right-shape, i.e., $S(V_\tau)$ is the rightmost color-MVS separating $U_\tau = U_\alpha$ and $S(V_\tau)$.*

Lemma 8.8.43. *Define*

$$f_{(S,\beta_S),T} := \sqrt{k}^{|S|}(\mathbf{1}_{(S,\beta_S)\cup T} - \mathbf{1}_{(S,\beta_S)}) - \sum_{\substack{S_W,T_W \\ T_W\subseteq T}} f_{(S_W,\beta_{S_W}),T_W} \cdot R[(S_W,\beta_{S_W})\cup T_W),(S,\beta_S)\cup T]\,,$$

we have

$$L^T f_{(S,\beta_S),T} = \sum_{\substack{(\alpha,CP):\text{(color) intersection-shape} \\ P_\alpha:\text{valid intersection-profile}}} (-1)^{|P_\alpha|}\sqrt{k}^{|S(V_\alpha)|}\lambda_L(\alpha)_{CP}\cdot \mathsf{M}_\alpha[\star,(S,\beta_S)\cup T]$$

In words, the RHS reads as the collection of shapes satisfying the property prescribed by valid intersection profile with V_σ evaluated on $(S,\beta_S)\cup(T,\beta_T)$ for any coloring $\beta_T : T\to[k]$.

For convenience, we assume (α, CP) to come with P_α specified within and ignore the extra dependence on P_α.

Proof. We focus on ribbons with non-trivial intersection pattern here, as the bijection between ribbons with trivial intersection is proven in earlier section. We note that ribbons with non-trivial section pattern on the LHS come from term in $L^T\sum_{\substack{S_W,T_W \\ T_W\subseteq T}} f_{(S_W,\beta_{S_W}),T_W}\cdot R[(S_W\cup T_W,\beta_{(S_W\cup T_W)}),(S\cup T,\beta_{S\cup T})]$. Note that the direction from RHS to LHS is immediate as any irreducible right shape via intersection records the intersection profile, i.e., how it appears in the product of $L^T f$. IT suffices for us to show the direction from LHS to RHS, in particular, the cancellation of intersections that does not satisfy the condition identified by definition 8.8.42.

Let σ be a ribbon of kernel-coefficient shape. Observe that any term coming from $L^T f\cdot R$ may be viewed as $\alpha\circ\sigma$, where α is a shape with valid intersection-profile for the intermediate separator $S_W\cup T_W$ via the induction hypothesis. Fix an intersection profile restricted to σ be P_σ. In the case σ is not a proper-intersection right-shape under P_σ, i.e., $S(V_\sigma)$ is not the color-MVS for $S(U_\sigma)\cup\mathrm{Int}(\sigma)$, we claim that we can another term of the

same shape on the LHS (that comes with a flipped sign) so that they cancel and do not contribute to the RHS of $L^T f_{(S,\beta_S),T}$. To see this, let γ be the proper-intersection shape on (σ, P_σ), and note that we can write $\sigma = \gamma \circ (\sigma - \gamma)$ where we take V_γ to be the color-MVS for σ under intersection pattern P_σ. By induction hypothesis, the shape $\alpha \circ \gamma$ with intersection profile $P(\alpha) \cup P_\gamma$ where we let P_γ be the intersection pattern of P_σ restricted to the γ part appears in $L^T f_{S(U_\gamma),T(U_\gamma)}$ with coefficient $(-1)^{|P_\alpha|+1}\lambda_L(\alpha \circ \gamma)$ where we note that the additional $+1$ comes from the outmost intersection level of γ. Therefore, these two shapes cancel each other and do not contribute the final RHS.

On the other hand, if σ is a proper intersection-right-shape, such term appears on the RHS with an extra factor of (-1) from the minus sign ahead of the composable terms. And importantly, observe that such shape with the prescribed intersection pattern continues to admit the characterization of a valid intersection profile for kernel term. □

8.8.4 Summary of Kernel Operations

Recall from our roadmap in the PSD analysis, the goal of this section is to find a matrix K_Q that approximates

$$K_{\mathsf{trivial}} = \sum_{\substack{W \subseteq [n] \\ |W| \leq \mathsf{dsos}}} \sum_{\substack{S(W) \cup T(W) \\ \text{kernel-partition}}} K_{S(W) \cup T(W)} \,.$$

The K_Q matrix is formally defined definition 8.8.15 as

$$K_Q \coloneqq \sum_{B \subseteq [n]} \sum_{\substack{S(B) \cup T(B) \\ \text{Kernel-Partition for } B}} \sum_{\beta_S : S(B) \to [k]} \left(\frac{1}{2} f_{(S,\beta_S),T} \otimes (\frac{1}{\sqrt{k}})^{|T|} \mathbf{1}_{(S,\beta_S),T} + \frac{1}{2} (\frac{1}{\sqrt{k}})^{|T|} \mathbf{1}_{(S,\beta_S),T} \otimes f_{(S,\beta_S),T} \right) .$$

To enable a combinatorial analysis, we will shift to the description of K_Q via right-shape obtained by our vectors $L^T f$ described in definition 8.8.42. Plugging in our main lemma

from construction f vectors in lemma 8.8.43, we have

$$L^T f_{(S,\beta_S),T} = \sum_{\substack{(\alpha,\text{CP}):\text{(color) intersection-shape} \\ P_\alpha:\text{intersection-pattern}}} (-1)^{|P_\alpha|} \lambda_L(\alpha)_{\text{CP}} \cdot \sqrt{k}^{|S(V_\alpha)|} \cdot \mathsf{M}_\alpha[\star, (S,\beta_S) \cup T]$$

Via the claim 8.8.19, vector outer-prouct with a left/right shape produces a corresponding kernel-shape, we now expand K_Q in the kernel-shape basis. For convenience, we define

$$\lambda_K(\alpha)_{\text{CP}} \coloneqq \lambda_L(\alpha)_{\text{CP}} \cdot \sqrt{k}^{|S(V_\alpha)|} \cdot (\frac{1}{\sqrt{k}})^{T(V_\alpha)}$$

Therefore, we have

$$\begin{aligned} K_Q &= \sum_{(\alpha,\text{CP}):\text{irreducible right-shape}} (-1)^{|P_\alpha|} \cdot \lambda_L(\alpha)_{\text{CP}} \sqrt{k}^{|S(V_\alpha)|} \cdot (\frac{1}{2} \cdot \mathsf{M}_{K(\alpha),\text{CP}} + \mathsf{M}^T_{K(\alpha),\text{CP}}) \cdot (\frac{1}{\sqrt{k}})^{T(V_\alpha)} \\ &= K_{\mathsf{trivial}} + \sum_{(\alpha,\text{CP}):\text{non-trivial irreducible right-shape}} (-1)^{|P_\alpha|} \cdot \lambda_K(\alpha)_{\text{CP}} (\frac{1}{2}\mathsf{M}_{K(\alpha),\text{CP}} + \frac{1}{2}\mathsf{M}^T_{K(\alpha),\text{CP}}) \end{aligned}$$

where we observe that the trivial shape is the only shape that is both a left and right shape, hence we can combine the coefficient of $\frac{1}{2}$ (i.e. it is invariant under left/right symmetrization), and their coefficient groups to be K_{trivial} by construction. Note that the second term is exactly the matrix $K_{\mathsf{nontrivial}}$. To show that K_Q is a good approximation as desired, we now proceed to show

$$\begin{aligned} \|K_{\mathsf{nontrivial}}\| &= \| \sum_{(\alpha,\text{CP}):\text{non-trivial irreducible right-shape}} (-1)^{|P_\alpha|} \cdot \lambda_K(\alpha)_{\text{CP}} (\frac{1}{2}\mathsf{M}_{K(\alpha),\text{CP}} + \frac{1}{2}\mathsf{M}^T_{K(\alpha),\text{CP}}) \| \\ &\leq 2 \| \sum_{(\alpha,\text{CP}):\text{non-trivial irreducible right-shape}} (-1)^{|P_\alpha|} \cdot \lambda_K(\alpha)_{\text{CP}} \mathsf{M}_{K(\alpha),\text{CP}} \| \end{aligned}$$

Similar to charging analysis for middle shapes in Q, we next perform combinatorial analysis in the next section to lower bound the slack of each shape in the above summation

that ultimately allows us to deduce

$$\|K_Q - K_{\text{trivial}}\| = \sum_{(\alpha,\text{CP}):\text{non-trivial irreducible right-shape}} \|\lambda_K(\alpha)_{\text{CP}} \cdot \mathsf{M}_{K(\alpha),\text{CP}}\| = o(1)$$

8.9 Combinatorial Charging Analysis for Irreducible Right Shape

Lemma 8.9.1. *For (σ, CP) a non-trivial irreducible color-right-shape with kernel partition $S \cup T$ on V_σ from lemma 8.8.43, consider its corresponding kernel shape $K(\sigma, CP)$, we have*

$$\|\lambda_K(\sigma)_{CP} \cdot \mathsf{M}_{K(\sigma,CP)}\| \leq slack(\sigma, CP)$$

for

$$\begin{aligned} slack(\sigma, CP) := \left(\frac{\sqrt{n}}{k}\right)^{\frac{1}{2}|V(\sigma)\setminus U_\sigma \cap S(V_\sigma)|} \cdot \left(\frac{1}{k}\right)^{\frac{1}{2}\cdot|rainbow(U_\sigma \cap S(V_\sigma))|} \cdot \\ O(|rainbow(E(U_\sigma \cap S(V_\sigma))))_{CP}|)^{|rainbow(E(U_\sigma \cap S(V_\sigma)))_{CP}|} \cdot \\ O(|V(\sigma) \setminus U_\sigma \cap S(V_\sigma)|)^{2|V(\tau)\setminus U_\tau \cap V_\tau|)} \end{aligned}$$

and

$$\lambda_K(\sigma) := \lambda_L(\sigma)_{CP}\sqrt{k}^{|S(V_\sigma)|} \cdot (\frac{1}{\sqrt{k}})^{|T(V_\sigma)|}$$

The analysis in this section is largely identical to the analysis in the fashion of section 8.6 and section 8.7 with minor modification adjusted to the structural properties. We start by applying piecewise decomposition of each irreducible right shape,

8.9.1 Piecewise Decomposition of Irreducible Right Shape

A color-right-shape $(\sigma, \mathrm{CP}_\sigma)$ is called irreducible if its kernel-decomposition $(\sigma, \mathrm{CP}_\sigma) = (A, \mathrm{CP}_A) \circ (B, \mathrm{CP}_B)$ via our prescribed process gives $B = Id$. That said, by tracking our recursive process that yields the decomposition of $A \circ B$, we also obtain a finer decomposition of A as the following. Recall that the process keeps track of intermediate separator S_{W_t} throughout, even though the process does not append T_{W_t} vertices for the intermediate level, observe that we may apply the T_W vertices for intermediate separators, and obtain $W_t = S_{W_t} \cup T_{W_t}$.

Formally, for each (rightwards) extension, we factor out edges and vertices involved, and define the extension piece as

Definition 8.9.2. *A kernel (rightwards) extension piece γ from adjacent separators S_{W_t} and $S_{W_{t+1}}$ is a shape defined as the following. Assume each intermediate separators comes with additional T_{W_t} vertices,*

1. *Let $U_\gamma = S_{W_t} \cup T_{W_t}$, and $V_\gamma = S_{W_{t+1}} \cup T_{W_{t+1}}$ with inherited kernel partition. In other words, $S(U_\gamma) = S_{W_t}$, and $T(V_\gamma) = T_{W_t}$. and similarly for V_γ.*

2. *Let $E(\gamma)$ be the set of edges that can be reached from $S(V_\gamma)$ without passing through any vertex in U_γ.*

Definition 8.9.3 (Kernel rightwrds-extension shape: rainbow)**.** *A shape (X, CP_X) with kernel partitions on both boundaries is a kernel rightwrads-MVS-extension shape if*

1. *U_X is a unique MVS for X ;*

2. *Any vertex in $S(V_X)$ is in the same rainbow component as $S(U_X)$: each vertex in $S(V_X)$ is in the rainbow component of some vertex in $S(U_X)$ under CP_X.*

3. *$T(U_X) \subseteq T(V_X)$ and any vertex in $T(U_X) \cap T(V_X)$ is dormant in X.*

4. *Any vertex in $T(V_X) \setminus T(U_X)$ is reachable from $S(V_X)$ without passing through U_X.*

Additionally there are kernel-partitions for both U_X and V_X, and any vertex in $T(U_X)$ is dormant in X, i.e., it is in $T(U_X) \cap T(V_X)$ and degree-0.

Remark 8.9.4. *Notice we may have an T vertex in $T(V_\sigma) \setminus T(U_\sigma)$ joining the boundary yet it is connected via a pure-color edge.*

Definition 8.9.5 (Kernel rightwrds-extension shape: MVS)**.** *A shape (Y, CP_Y) with kernel partition on both boundaries is a kernel rightwrads-MVS-extension shape if*

1. *$S(V_\gamma)$ is the unique MVS separating $S(U_\gamma)$ and $S(V_\gamma)$, moreover, V_Y is a unique MVS for Y.*

2. *Any edge in $E(Y)$ incident to U_Y is pure-color under CP.*

3. *$T(U_X) \subseteq T(V_X)$ and any vertex in $T(U_X) \cap T(V_X)$ is dormant in X.*

4. *Any vertex in $T(V_X) \setminus T(U_X)$ is reachable from $S(V_X)$ without passing through U_X.*

Let $\widetilde{\tau}_P$ be the intersection result in $\tau \circ Y$.

Proposition 8.9.6. *For an irreducible color-right-shape (A, CP_A), we have*

$$(A, CP_A) = (\tau, CP_\tau) \circ (X_1, CP_{X_1}) \circ (Y_1, CP_{Y_1}) \circ \cdots \circ (X_t, CP_{X_t}) \circ (Y_t, CP_{Y_t})$$

where

1. *τ is a base piece such that $S(V_\tau)$ is an MVS separating U_τ and $S(V_\tau)$;*

2. *(X_i, CP_{X_i}) for $i \in [t-1]$ is a kernel-rightwards-rainbow-extension shape;*

3. *(Y_i, CP_{Y_i}) for $i \in [t-1]$ is a kernel-rightwards-MVS-extension shape;*

With some abuse of notation, we allow the final extension Y_t to be trivial (i.e. identity).

Proof. Notice may view the base piece τ as an MVS-extension, thus it suffices for us to show each γ piece factored out is an MVS/rainbow-extension piece satisfy the desired properties.

Let γ be a rightwards rainbow-extension from S_t to S_{t+1} (along with T_t and T_{t+1} their corresponding T vertices). By construction of rainbow extension from S_t to S_{t+1}, any vertex in S_{t+1} is the boundary of rainbow components incident to S_t that can be reached from V_A, and hence $S(V_\gamma) = S_{t+1}$ are in the same rainbow components as $S(U_\gamma) = S_t$.

For the other case, let γ be a rightwards MVS-extension S_t to S_{t+1} (along with T_t and T_{t+1} their corresponding T vertices). The condition $S(V_\gamma)$ being the unique MVS separating $S(U_\gamma)$ and $S(V_\gamma)$ follows immediately by definition from MVS-extension. To see that U_γ is a unique MVS, notice $U_\gamma = S_t \cup T_t$ and as we show below, $T_t \subseteq T(U_\gamma) \cap T(V_\gamma)$ is dormant in γ, therefore U_γ is the MVS separating U_γ and $V_\gamma \cup T(U_\gamma)$. Finally adding vertices in $T(V_\gamma) \setminus T(U_\gamma)$ preserves the MVS and proves the claim.

To verify $T(U_X) \subseteq T(V_X)$, note that any vertex is added to $T(U_X)$ either has a path to U_A in X without using edges to the right of S_t (i.e. edges reachable from $S(V_A)$ without passing through S_A), or a T vertex not connected to $U_A \cup S(V_A)$ yet incident to edges. In the former case, any such path is a also a path from U_A without using edges to the right of S_{t+1}, hence such vertex continues to be in T_{t+1}. The latter case is immediate as any such component is in T_{W_t} regardless of the rightwards-extensions. Additionally, observe that any vertex in $T(U_\gamma) \cap T(V_\gamma)$ is dormant in γ as any T vertex in U_γ is not incident to edges to the right of $S_t = S(U_\gamma)$ while any edge in γ is to the right of S_t by definiton.

□

Norm Bound for Kernel Shape Next, we observe the following norm bound for kernel shape.

Proposition 8.9.7 (Weight function for $K(\sigma)$)**.** *For any right shape (σ, CP) with kernel partition on V_σ, and its corresponding kernel-shape $K(\sigma, CP)$, we have*

$$\|\mathsf{M}_{K(\sigma),CP}\| \leq w_K(\sigma)_{CP} \cdot (|V(\sigma) \setminus U_\sigma \cap V_\sigma|)^{O(1)|V(\sigma)\setminus U_\sigma \cap V_\sigma|}$$

for

$$w_K(\sigma)_{CP} := \max_{S:\text{separator for } \sigma} \sqrt{n}^{|V(\sigma)|-|S|+|I(\sigma)|} \cdot k^{|CP(\sigma)|-\frac{|CP(U_\sigma)|+|CP(S(V_\sigma))|}{2}} \cdot \sqrt{k}^{|T(V_\sigma)|}$$

Proof. We first focus on the norm bound factor of n, which is given by

$$\max_{H:\text{separator for } K(\sigma)} \sqrt{n}^{|V(\sigma)\setminus H|} = \max_{S:\text{separator for } \sigma} \sqrt{n}^{|V(\sigma)|-|S|+|I(\sigma)|}$$

where we observe that $K(\sigma)$ has the same separator as σ, and the MVS is given by U_σ as any σ is a right-shape. In other words, $K(\sigma)$ and σ share the same norm bound factor from $[n]$.

For the color factors, the factor on $[k]$ coming from the connected component is therefore bounded by

$$k^{|\mathrm{CP}(K(\sigma))|-\frac{|\mathrm{CP}(U_{K(\sigma)})|+|\mathrm{CP}(S(V_{K(\sigma)}))|}{2}} \leq k^{|\mathrm{CP}(\sigma)|-\frac{|\mathrm{CP}(U_\sigma)|+|\mathrm{CP}(S(V_\sigma))|}{2}+\frac{|T(V_\sigma)|}{2}}$$

To see this bound, notice we continue to apply the factor assignment scheme that we a factor of $\sqrt{k}$ for each vertex making its first, or last appearance in the trace calculation, and this implies that

1. Each color that only appears in $V(K(\sigma))$ gives at most a factor k;
2. Each color that appears in $U_{K(\sigma)} = U_\sigma$ has half of its k factor assigned to the previous block, which means it gets at most a factor of $\sqrt{k}$ for the current block;

3. Each vertex that appears in $V_{K(\sigma)}$ has half its k factor assigned to the subsequent block, which means it gets at most a factor of $\sqrt{k}$ for the current block.

To see the upper bound on the RHS which connects back to the color factor of σ as opposed to $K(\sigma)$, we observe that it is equivalent to apply the above scheme on σ, by ignoring the original right boundary condition for vertices in $T(V_\sigma)$, as its color is not guaranteed to appear in the subsequent block. Moreover, for each vertex connected by a dotted line in $V_{K(V_\sigma)}$, an extra factor of $\sqrt{k}$ is sufficient as this color is bound to appear in the subsequent block via being in the block boundary. This gives us

$$k^{|\mathrm{CP}(\sigma)|-\frac{|\mathrm{CP}(U_\sigma)|+|\mathrm{CP}(S(V_\sigma))|}{2}} \cdot \underbrace{\sqrt{k}^{|T(V_\sigma)|}}_{\text{color-jump}}$$

This completes our bound for $\|\mathsf{M}_{K(\sigma),\mathrm{CP}}\|$ by the identical argument in the norm bound for color-shape. □

With the above weight function, we apply the same idea of factorizing the weight function into pieces.

8.9.2 Charging Analysis for Proper Irreducible Right Shapes

We start by considering proper irreducible right shapes without any intersection, and note that the norm bound conveniently simplifies for a proper right shape (σ, CP) as U_σ is by definition the (unique) MVS, and we have

$$w_K(\sigma)_{\mathrm{CP}} = \sqrt{n}^{|V(\sigma)\setminus U_\sigma|} k^{|\mathrm{CP}(\sigma)|-\frac{|\mathrm{CP}(U_\sigma)|+|\mathrm{CP}(S(V_\sigma))|}{2}} \cdot \sqrt{k}^{|T(V_\sigma)|}.$$

Set-up of Piecewise Weight Function and Coefficient Function Similar to middle shapes and intersection shapes in the analysis for Q , irreducible color-right-shapes are

obtained via a sequence of rightwards extensions starting from the MVS for U_σ and $S(V_\sigma)$. We apply the weight function and coefficient function from middle-shape charging, and show that the desired quantity is small throughout the rightwards extension process.

Towards that end, we define weight function for each piece as the following.

Definition 8.9.8 (Weight function for kernel pieces)**.** *Define weight function for each kernel piece as*

1. *For (τ, CP) the base piece, define*

$$w_K(\tau)_{CP} = \sqrt{n}^{|V(\tau)\setminus U_\tau|} \cdot k^{|CP(\tau)|-\frac{|CP(U_\tau)|+|CP(S(V_\tau)|)}{2}} \cdot \sqrt{k}^{|CP(T(V_\tau))|}$$

2. *For (X, CP) the rightwards extension piece (including both rainbow and MVS-extension), for each piece, let D_X be the dormant vertices in $T(U_X) \cap T(V_X)$, define*

$$w_K(X)_{CP} = \sqrt{n}^{|V(X)\setminus U_X|} \cdot \sqrt{k}^{|CP(T(V_X)\setminus T(U_X))|} \cdot k^{|CP(X\setminus D_X)|-\frac{|CP(U_X\setminus T(U_X))|+CP(S(V_X))|}{2}}.$$

We ignore the subscript dependence of $w_K(\cdot)$ in this section as it is clear we are working with kernel pieces throughout unless otherwise specified.

Proposition 8.9.9. *Recall that*

$$w_K(\sigma)_{CP} \coloneqq \sqrt{n}^{|V(\sigma)\setminus U_\sigma|} \cdot k^{|CP(\sigma)|-\frac{|CP(U_\sigma)|+|CP(S(V_\sigma))|}{2}+\frac{|T(V_\sigma)|}{2}},$$

For $\sigma = \tau \circ X_1 \circ Y_1 \circ \cdots \circ X_t \circ Y_t$, we have

$$w(\sigma) \leq w(\tau) \prod_i w(X_i) \cdot w(Y_i).$$

Proof. Observe that the $\sqrt{n}$ factor of each vertex not in U_σ comes from the leftmost piece

it first appears in, and for the color-factors of k not via dotted line, i.e. color-jump factor, follow identically to the piecewise decomposition for middle shape; on the other hand, we note that the color-jump factor of $\sqrt{k}$ is assigned to the piece in which the T vertex first appears (i.e., the leftmost piece). □

For clarity, we define the coefficient function for kernel piece in essentially the same fashion as that for middle shape,

Definition 8.9.10 (Approximate coefficient function for kernel piece)**.** *Define the coefficient function for kernel piece as the following,*

1. *For the base piece τ, we define*

$$\lambda_K^{\approx}(\tau)_{CP} \coloneqq (\frac{1}{k})^{|V(\tau)|-\frac{|S(V_\tau)|+|U_\tau|}{2}} \cdot Ev^{\approx}(\tau) \cdot (\frac{1}{\sqrt{k}})^{|T(V_\tau)|}.$$

   for

$$Ev^{\approx}(\tau) \coloneqq \prod_{\mathcal{G}_i \in \mathcal{G}(\tau, CP)} (\frac{1}{k})^{|V(\mathcal{G}_i)|-1}$$

2. *For a rightwards-rainbow-extension shape X, we define*

$$\lambda_K^{\approx}(X)_{CP} \coloneqq (\frac{1}{k})^{|V(X)\setminus D_X|-\frac{|S(U_X)|+|S(V_X)|}{2}} \cdot Ev^{\approx}(X)_{CP} \cdot (\frac{1}{\sqrt{k}})^{|T(V_X)\setminus T(U_X)|};$$

   for

$$Ev^{\approx}(X)_{CP} \coloneqq \prod_{\substack{\mathcal{G}_i \in \mathcal{G}(X,CP) \\ \mathcal{G}_i \text{incident to} S(U_X)}} (\frac{1}{k})^{|V(\mathcal{G}_i)|} \cdot \prod_{\substack{\mathcal{G}_i \in \mathcal{G}(X,CP) \\ \mathcal{G}_i \text{not connected to } S(U_X)}} (\frac{1}{k})^{|V(\mathcal{G}_i)|-1}.$$

3. *For a rightwards-MVS-extension shape Y, we define*

$$\lambda_K^{\approx}(Y)_{CP} \coloneqq (\frac{1}{k})^{|V(Y)\setminus D_Y|-\frac{|S(U_Y)|+|S(V_Y)|}{2}} \cdot Ev^{\approx}(Y)_{CP} \cdot (\frac{1}{\sqrt{k}})^{|T(V_Y)\setminus T(U_Y)|}$$

for

$$Ev^{\approx}(Y)_{CP} \coloneqq \prod_{\substack{\mathcal{G}_i \in \mathcal{G}(Y,CP) \\ \mathcal{G}_i \text{connected to} S(U_X) \cup S(U_\tau)}} (\frac{1}{k})^{|V(\mathcal{G}_i)|-1}.$$

Proposition 8.9.11. *Recall that*

$$\lambda_K^{\approx}(\sigma)_{CP} = (\frac{1}{k})^{|V(\sigma)| - \frac{|U_\sigma|+|S(V_\sigma)|}{2}} \cdot (\frac{1}{\sqrt{k}})^{|T(V_\sigma)|} \cdot Ev^a pprox(\sigma)_{CP}$$

we have

$$\lambda_K^{\approx}(\sigma)_{CP} \leq \lambda_K^{\approx}(\tau)_{CP(\tau)} \cdot \prod_i \lambda_K^{\approx}(X_i)_{CP(X_i)} \cdot \lambda_K^{\approx}(Y_i)_{CP(Y_i)}$$

Proof. The proof is identical to that of middle shape, except noting that the additional factor of $(\frac{1}{\sqrt{k}})^{|T(V_\sigma)|}$ is assigned to the piece in which each vertex in $T(V_\sigma)$ first appears (i.e., the leftmost piece). □

8.9.3 Charging for the Base Irreducible Right Shape

Lemma 8.9.12 (Charging for base kernel piece)**.** *Let (τ, CP) be a base piece such that $S(V_\tau)$ is MVS for U_τ and $S(V_\tau)$, we have*

$$\lambda_K^{\approx}(\tau)_{CP} \cdot w(\tau)_{CP} \leq (\frac{\sqrt{n}}{k})^{|V(\tau) \setminus U_\tau|}$$

Similar to our bound for the base piece Y in the middle shape, any left shape in L originally does not contain any floating components that act as a scalar should be factored out. Here, even though every component is connected to $S(V_\tau) \cup T(V_\tau)$, components connected to $T(V_\tau)$ alone while disconnected from $U_\tau \cup S(V_\tau)$ act as floating components in the kernel-matrix of $K(\sigma)$. That said, by construction of our correction scheme, any shape without such floating component have vanishing value. Therefore, it suffices for us to consider shapes with each component connected to $U_\tau \cup S(V_\tau)$.

Unpacking the factors, we have

$$\begin{aligned}
&\lambda(\tau)_{\mathrm{CP}} \cdot w(\tau)_{\mathrm{CP}} \\
&= (\frac{1}{k})^{|V(\tau)|-\frac{|S(V_\tau)|+|U_\tau|}{2}} \cdot \prod_{\mathcal{G}_i \in \mathcal{G}(\tau,\mathrm{CP})} (\frac{1}{k})^{|V(\mathcal{G}_i))|-1} \cdot (\frac{1}{\sqrt{k}})^{|T(V_\tau)|} \\
&\cdot \sqrt{n}^{|V(\tau)\setminus U_\tau|} \cdot k^{|\mathrm{CP}(\tau)|-\frac{|\mathrm{CP}(U_\tau)|+|\mathrm{CP}(S(V_\tau)|)}{2}} \cdot \sqrt{k}^{|\mathrm{CP}(T(V_\tau))|}
\end{aligned}$$

It then follows by the subsequent claims.

Claim 8.9.13.

$$|T(V_\tau)| \geq |CP(T(V_\tau))|$$

Proof. This is immediate as $|\mathrm{CP}(T(V_\tau))|$ the number of distinct colors on $T(V_\tau)$ is at most $|T(V_\tau)|$. □

Claim 8.9.14.

$$(\frac{1}{k})^{|V(\tau)|-\frac{|S(V_\tau)|+|U_\tau|}{2}} \sqrt{n}^{|V(\tau)\setminus U_\tau|} \leq (\frac{1}{\sqrt{k}})^{|U_\tau|-|S(V_\tau)|} \cdot (\frac{\sqrt{n}}{k})^{|V(\tau)\setminus U_\tau|}$$

Proof. Rearranging the above,

$$\begin{aligned}
(\frac{1}{k})^{|V(\tau)|-\frac{|S(V_\tau)|+|U_\tau|}{2}} \sqrt{n}^{|V(\tau)\setminus U_\tau|} &= k^{\frac{|S(V_\tau)|-|U_\tau|}{2}} \cdot (\frac{1}{k})^{|V(\tau)|-|U_\tau|} \cdot \sqrt{n}^{|V(\tau)\setminus U_\tau|} \\
&\leq (\frac{1}{\sqrt{k}})^{|U_\tau|-|S(V_\tau)|} \cdot (\frac{\sqrt{n}}{k})^{|V(\tau)\setminus U_\tau|}
\end{aligned}$$

□

It should be noted here that since we assume $S(V_\tau)$ to be the MVS for U_τ and $S(V_\tau)$, this guarantees the above quantity is at most 1. However, it gives us an extra gap of factor $\sqrt{k}$ that would be useful for the following bound on color-factors.

Claim 8.9.15.

$$(\frac{1}{\sqrt{k}})^{|U_\tau|-|S(V_\tau)|} \cdot \underbrace{\prod_{\mathcal{G}_i \in \mathcal{G}(\tau,CP)} (\frac{1}{k})^{|V(\mathcal{G}_i)|-1}}_{Ev^{\approx}(\tau)_{CP}} \cdot k^{|CP(\mathsf{conn}(\tau))|-\frac{|CP(U_\tau)|+|CP(S(V_\tau)|)}{2}} \leq 1$$

Proof. Taking logarithm with base k, it suffices for us to show

$$\frac{|U_\tau| - |S(V_\tau)|}{2} + \sum_{\mathcal{G}_i \in \mathcal{G}(\tau,\mathrm{CP})} (|V(\mathcal{G}_i)| - 1) \geq |\mathrm{CP}(\tau)| - \frac{|\mathrm{CP}(U_\tau)| + |\mathrm{CP}(S(V_\tau))|}{2}$$

To show this, we first apply process each component in τ, and then remove possible pure-edges it may contain and process the rainbow components therein. To begin with, observe that for any component connected to $S(V_\tau) \cup U_\tau$, it must also be connected to U_τ: suppose not, we have a component connected to $S(V_\tau)$ but not U_τ, consider the alternate separator obtained by removing the incident vertex from $S(V_\tau)$ preserves a separator separating U_τ and $S(V_\tau)$, and this contradicts the assumption that $S(V_\tau)$ is an MVS for U_τ and $S(V_\tau)$. That said, it suffices for us to consider components connected to U_τ.

Proposition 8.9.16.

$$(\#\textit{components connected to } U_\tau \textit{ but not } S(V_\tau)) \leq |U_\tau| - |S(V_\tau)|$$

Proof. For each component C, since $S(V_\tau)$ is an MVS separating U_τ and $S(V_\tau)$, we also have $|V(C) \cap U_\tau| \geq |V(C) \cap S(V_\tau)|$ as otherwise we can obtain a smaller MVS by considering $S(V_\tau) - (V(C) \cap S(V_\tau)) + |V(C) \cap U_\tau|$. That said, summing over components C connected to U_τ ,

1. Component C connected to $S(V_\tau)$ gives a 0 to the LHS, while a non-negative factor on the RHS;

2. Component C not connected to $S(V_\tau)$ gives an 1 to the LHS, while $|U_\tau \cap V(C)| - |S(V_\tau) \cap V(C)| = |U_\tau \cap V(C)| \geq 1$, hence the inequality continues to hold.

This proves our proposition as we have

$$|U_\tau| = \sum_{C\text{:connected to } U_\tau} |U_\tau \cap V(C)|,$$

and

$$|S(V_\tau)| = \sum_{C\text{:connected to } U_\tau} |S(V_\tau) \cap V(C)|$$

as each vertx/component is connected to U_τ. □

Proposition 8.9.17.

$$\sum_{\mathcal{G}_i \in \mathcal{G}(\tau, CP)} (|V(\mathcal{G}_i)| - 1) \geq |CP(\tau)| - |CP(U_\tau)|,$$

and additionally,

$$(\#\textit{components connected to } U_\tau \textit{ but not } S(V_\tau)) + \sum_{\mathcal{G}_i \in \mathcal{G}(\tau, CP)} (|V(\mathcal{G}_i)| - 1) \geq |CP(\tau)| - |CP(S(V_\tau))|.$$

Proof. The proof to the first line is identical to that of proposition 8.6.13, as any component is connected to U_τ.

For the second line, it is not true that any component in τ is connected to $S(V_\tau)$ here. However, for each such component, it suffices for us to identify a vertex in $|V(C) \cap U_\tau|$ and apply the induction process on the rainbow components therein, this change is accounted for by the additional factor on the LHS. □

We are now ready to prove the desired claim. Taking average of the sum of the two

equations in the proposition, we have

$$\begin{aligned}
&|\mathrm{CP}(\tau)| - \frac{|\mathrm{CP}(U_\tau)| + |\mathrm{CP}(S(V_\tau))|}{2} \\
&\leq \frac{(\#\text{components connected to } U_\tau \text{ but not } S(V_\tau))}{2} + \sum_{\mathcal{G}_i \in \mathcal{G}(\tau, Y))} (|V(\mathcal{G}_i))| - 1) \\
&\leq \frac{|U_\tau| - |S(V_\tau)|}{2} + \sum_{\mathcal{G}_i \in \mathcal{G}(\tau, \mathrm{CP})} (|V(\mathcal{G}_i)| - 1)\,.
\end{aligned}$$

□

Proof to lemma 8.9.12. We may now complete the proof to the lemma combining the above claims,

$$\begin{aligned}
&\lambda^{\approx}_K(\tau)_{\mathrm{CP}} \cdot w_K(\tau)_{\mathrm{CP}} \\
&= (\frac{1}{k})^{|V(\tau)| - \frac{|S(V_\tau)| + |U_\tau|}{2}} \cdot \prod_{\mathcal{G}_i \in \mathcal{G}(\tau, \mathrm{CP})} (\frac{1}{k})^{|V(\mathcal{G}_i))| - 1} \cdot (\frac{1}{\sqrt{k}})^{|T(V_\tau)|} \\
&\cdot \sqrt{n}^{|V(\tau) \setminus U_\tau|} \cdot k^{|\mathrm{CP}(\tau)| - \frac{|\mathrm{CP}(U_\tau)| + |\mathrm{CP}(S(V_\tau))|}{2}} \cdot \sqrt{k}^{|\mathrm{CP}(T(V_\tau))|} \\
&\leq (\frac{1}{\sqrt{k}})^{|U_\tau| - |S(V_\tau)|} \cdot (\frac{\sqrt{n}}{k})^{|V(\tau) \setminus U_\tau|} \cdot k^{|\mathrm{CP}(\tau)| - \frac{|\mathrm{CP}(U_\tau)| + |\mathrm{CP}(S(V_\tau))|}{2}} \cdot \underbrace{(\frac{1}{\sqrt{k}})^{|T(V_\sigma)|} \cdot \sqrt{k}^{|\mathrm{CP}(T(V_\sigma))|}}_{\leq 1 \text{via claim } 8.9.13} \\
&\leq (\frac{\sqrt{n}}{k})^{|V(\tau) \setminus U_\tau|}
\end{aligned}$$

where the last line follows by claim 8.9.15. □

8.9.4 Charging for Rainbow Extension

Lemma 8.9.18. *For (X, CP) a rightwards-rainbow kernel-extension piece, we have*

$$\lambda^{\approx}_K(X)_{CP} \cdot w(X)_{CP} \leq (\frac{\sqrt{n}}{k})^{|V(X) \setminus U_X|}$$

Proof. Unpacking the factors, we have

$$\begin{aligned}
&w_K(X)_{\mathrm{CP}} \cdot \lambda_K^{\approx}(X)_{\mathrm{CP}} \\
&= (\frac{1}{k})^{|V(X)\setminus D_X| - \frac{|S(U_X)|+|S(V_X)|}{2}} \cdot Ev^{\approx}(X)_{\mathrm{CP}} \cdot \underbrace{(\frac{1}{\sqrt{k}})^{|T(V_X)\setminus T(U_X)|} \cdot \sqrt{k}^{|\mathrm{CP}(T(V_X)\setminus T(U_X))|}}_{\leq 1} \\
&\cdot \sqrt{n}^{|V(X)\setminus U_X|} \cdot k^{|\mathrm{CP}(X\setminus D_X)| - \frac{|\mathrm{CP}(U_X\setminus T(U_X))|+\mathrm{CP}(S(V_X))|}{2}}
\end{aligned}$$

for

$$Ev^{\approx}(X)_{\mathrm{CP}} := \prod_{\substack{\mathcal{G}_i \in \mathcal{G}(X,\mathrm{CP}) \\ \mathcal{G}_i \text{incident to} S(U_X)}} (\frac{1}{k})^{|V(\mathcal{G}_i)|} \cdot \prod_{\substack{\mathcal{G}_i \in \mathcal{G}(X,\mathrm{CP}) \\ \mathcal{G}_i \text{not incident to } S(U_X)}} (\frac{1}{k})^{|V(\mathcal{G}_i)|-1}$$

where we first observe $|\mathrm{CP}(T(V_X) \setminus T(U_X))| \leq |T(V_X) \setminus T(U_X)|$ by definition. We next focus on the vertex factors, and observe that

$$\begin{aligned}
(\frac{1}{k})^{|V(X)\setminus D_X| - \frac{|S(U_X)|+|S(V_X)|}{2}} \cdot \sqrt{n}^{|V(X)\setminus U_X|} &= (\frac{1}{k})^{|V(X)\setminus D_X| - \frac{|S(U_X)|+|S(V_X)|}{2}} \sqrt{n}^{|V(X)\setminus D_X| - |S(U_X)|} \\
&\leq (\frac{\sqrt{n}}{k})^{|V(X)\setminus U_X|} \cdot k^{\frac{|S(V_X)|-|S(U_X)|}{2}}
\end{aligned}$$

Combining with the remaining factors, it remains for us to show

$$k^{\frac{|S(V_X)|-|S(U_X)|}{2}} \cdot Ev(X)_{\mathrm{CP}} \cdot k^{|\mathrm{CP}(X\setminus D_X)| - \frac{|\mathrm{CP}(U_X\setminus T(U_X))|+\mathrm{CP}(S(V_X))|}{2}} \leq 1\,.$$

We complete the proof by the following claim which considers the above equation under logarithm with k.

Claim 8.9.19.

$$\begin{aligned}
&\frac{|S(V_X)| - |S(U_X)|}{2} + |CP(X \setminus D_X)| - \frac{|CP(U_X \setminus T(U_X))| + CP(S(V_X))|}{2} \\
&\leq \sum_{\mathcal{G}_i : \textit{incident to } U_X} |V(\mathcal{G}_i)| + \sum_{\mathcal{G}_i : \textit{not incident to } U_X} (|V(\mathcal{G}_i)| - 1)
\end{aligned}$$

□

Proof to claim 8.9.19. The proof is identical to that of proposition 8.6.15 where we crucially observe that each vertex in $S(V_X)$ is in the rainbow component of some vertex of some vertex in $S(U_X)$. □

8.9.5 Charging for MVS Extension

Lemma 8.9.20. *For (X, CP) a rightwards-MVS kernel-extension piece, we have*

$$\lambda^{\approx}_K(X)_{CP} \cdot w_K(X)_{CP} \leq slack(X, CP)$$

for

$$slack(X, CP) := (\frac{\sqrt{n}}{k})^{|V(X)\setminus U_X|}$$

Proof. Start by unpacking the factors, we have

$$\lambda^{\approx}_K(X)_{\mathrm{CP}} \cdot w_K(X)_{\mathrm{CP}} =$$
$$\sqrt{n}^{|V(X)\setminus U_X|} \cdot k^{|\mathrm{CP}(X)\setminus \mathrm{CP}(D_X)| - \frac{|\mathrm{CP}(S(U_X))|+|\mathrm{CP}(S(V_X))|}{2}} \cdot Ev(X)_{\mathrm{CP}} \cdot (\frac{1}{k})^{|V(X)\setminus D_X| - \frac{|S(U_X)|+|S(V_X)|}{2}}$$

Since this is a rightwards-MVS-extension piece, we have V_X being an MVS for X. Hence,

$$\begin{aligned}\sqrt{n}^{|V(X)\setminus U_X|} \cdot (\frac{1}{k})^{|V(X)\setminus D_X| - \frac{|S(U_X)|+|S(V_X)|}{2}} &= \sqrt{n}^{|V(X)\setminus D_X| - |S(U_X)|} \cdot (\frac{1}{k})^{|V(X)\setminus D_X| - \frac{|S(U_X)|+|S(V_X)|}{2}} \\ &\leq k^{\frac{|S(V_X)|-|S(U_X)|}{2}} \cdot (\frac{\sqrt{n}}{k})^{|V(X)\setminus U_X|}.\end{aligned}$$

Rewrite $|V_X| = |S(V_X)| + |T(U_X) \cap T(V_X)| + |T(V_X) \setminus (T(U_X) \cap T(V_X))|$ as these three sets are piecewise-disjoint, by V_X being an MVS for X, we have

$$|V_X| \leq |U_X| = |S(U_X)| + |T(U_X) \cap T(V_X)|,$$

rearranging gives us

$$|S(V_X)| + |T(V_X) \setminus (T(U_X) \cap T(V_X))| \leq |S(U_X)|,$$

and hence $|S(V_X)| \leq |S(U_X)|$.

We now consider the remaining color-factors of the above equation,

$$k^{|\mathrm{CP}(X) \setminus \mathrm{CP}(D_X)| - \frac{|\mathrm{CP}(S(U_X))| + |\mathrm{CP}(S(V_X))|}{2}} \cdot Ev(X)_{\mathrm{CP}} \leq 1$$

by the following claim.

Claim 8.9.21.

$$|CP(X) \setminus CP(D_X)| - \frac{|CP(S(U_X))| + |CP(S(V_X))|}{2} \leq \sum_{\substack{\mathcal{G}_i \in \mathcal{G}(CP) \\ \mathcal{G}_i}} (|V(\mathcal{G}_i)| - 1)$$

This follows by bounding

$$|\mathrm{CP}(X) \setminus \mathrm{CP}(D_X)| - |\mathrm{CP}(S(U_X))| \leq \sum_{\substack{\mathcal{G}_i \in \mathcal{G}(X,\mathrm{CP}) \\ \mathcal{G}_i E(\mathcal{G}_i) \cap E(X) \neq \emptyset}} (|V(\mathcal{G}_i)| - 1),$$

and

$$|\mathrm{CP}(X) \setminus \mathrm{CP}(D_X)| - |\mathrm{CP}(S(V_X))| \leq \sum_{\substack{\mathcal{G}_i \in \mathcal{G}(\mathrm{X},\mathrm{CP}) \\ \mathcal{G}_i E(\mathcal{G}_i) \cap E(\mathrm{X}) \neq \emptyset}} (|V(\mathcal{G}_i)| - 1),$$

respectively by considering a traversal from U_X and V_X, and notice each vertex is connected to U_X and to V_X for an MVS-extension piece. Taking the average of the above proves the claim and this completes the charging for MVS extension piece. □

8.9.6 Charging for Irreducible Right Shapes with Intersections

We consider the slack for irreducible right shapes with non-trivial intersections. Similar to our intersection analysis for Q, for any level of intersection τ and γ that intersects into τ_P, we factorize out the intersection component in $\gamma = Y \circ A_i \circ X_i \dots A_t \circ X_t$ for Y a proper intersection-left-shape, and $\{A_i, X_i\}$ the collection of rightwards rainbiw/MVS extension pieces from the process of identifying rainbow MVS. This allows us to write $\tau_P = \widetilde{\tau}_P \circ A_1 \circ X_1 \circ \cdots \circ A_t \circ X_T$, and apply weight function and approximate coefficient for $\widetilde{\tau}_P$.

Formally, we define the base piece for γ as the following while the extension pieces are identical to the kernel extension pieces for proper irreducible right shape.

Definition 8.9.22 (Base Piece for Kernel Intersection γ-shape)**.** *The base piece for intersection of $\tau \circ \gamma$ into τ_P is a shape Y obtained at the first stage of applying rainbow MVS decomposition process without applying any further rightwards extensions. In other words, it is defined via the following process,*

1. *$S(V_Y)$ is a MVS separating $U_Y \cup Int(\gamma)_{\tau_P}$ and $S(V_\gamma) = S(V_{\tau_P})$ where $Int(\gamma)_{\tau_P}$ is the set of vertices in $V(\gamma)$ intersecting with vertices in τ beyond the boundary;*

2. *Let $E(\gamma - Y)$ be the set of edges that can be reached from $S(V_\gamma)$ without passing through $S(V_Y)$, and define edges $E(Y) = E(\gamma) \setminus E(E(\gamma - Y)$;*

3. *We additionally apply the T_W update operation to give kernel partition on Y: $U_Y = V_\tau$ with inherited kernel partition, and $T(V_Y)$ is any vertex from $T(V_\gamma) \setminus S(V_Y)$ that is either reachable from $U_Y \cup Int(\gamma)_{\tau_P}$ without using any edge in $E(\gamma - Y)$ or vertex not connected to $U_Y \cup Int(\gamma)_{\tau_P} \cup S(V_\gamma)$ but incident to edges.*

With the base piece defined, observe it satisfies the following structural property by our decomposition,

Definition 8.9.23. *A shape Y is a base piece for kernel interesction if*

1. *U_Y and V_Y both come with kernel partitions;*
2. *$S(V_Y)$ is the MVS separating $U_Y \cup Int(Y)$;*
3. *$T(U_Y) \subseteq T(V_Y)$ are dormant in Y, and any vertex in $T(V_Y) \setminus T(U_Y)$ is reachable from $S(V_Y)$ without passing through U_Y.*

Recall that $D_Y = T(U_Y) \cap T(V_Y)$ is the set of dormant vertices in T, we define the approximate coefficient for Y as

$$\lambda_K^{\approx}(Y) := (\frac{1}{k})^{|V(Y)\setminus D_Y| - \frac{|S(V_Y)|+|U_Y|}{2}} \cdot Ev(\tau)_{CP}^{\approx} \cdot (\frac{1}{\sqrt{k}})^{|T(V_Y)\setminus T(U_Y)|}$$

Proposition 8.9.24.

$$\lambda_L(\gamma) \leq \lambda_K^{\approx}(Y) \cdot \prod_i \lambda_K^{\approx}(A_i) \cdot \lambda_K^{\approx}(X_i)$$

Proof. This follows by analogous proof to our decomposition of coefficient of irreducible right shape into pieces. □

Lemma 8.9.25 (Slack for Intersection Base Piece)**.**

$$\lambda_K^{\approx}(\widetilde{\tau}_P)_{CP(\widetilde{\tau}_P)} \cdot w_K(\widetilde{\tau}_P) \leq \lambda_K^{\approx}(\tau)_{CP(\tau)} \cdot w_K(\tau)_{CP(\tau)} \cdot slack(Y, CP(Y))$$

for

$$slack(Y, CP(Y)) := (\frac{\sqrt{n}}{k})^{|V(Y)| - \frac{|U_Y|+|V_Y|}{2}}$$

Proof. This is analogous to lemma 8.7.4 by considering the left-intersection-shape to be trivial. while we contain the full proof here for completeness On the LHS, we have

$$\lambda_K^{\approx}(\widetilde{\tau}_P)_{\mathrm{CP}(\widetilde{\tau}_P)} \cdot w_K(\widetilde{\tau}_P) = (\frac{1}{k})^{|V(\widetilde{\tau}_P)| - \frac{|U_Y|+S(V_Y)|}{2}} \cdot Ev(\tau)_{\mathrm{CP}(\tau)} \cdot Ev^{\approx}(Y)_{\mathrm{CP}(Y)}$$

$$\cdot \max_{S:\text{sep. for}\widetilde{\tau}_P} \sqrt{n}^{|V(\widetilde{\tau}_P)|-|S|+|I(\widetilde{\tau}_P)|} \cdot k^{|\text{CP}(\tau\circ Y)|-\frac{|U_Y|+|S(V_Y)|}{2}} \cdot \sqrt{k}^{|\text{CP}(T(V_Y))|} \cdot (\frac{1}{\sqrt{k}})^{|T(V_Y)|}$$

Apply intersection-trade-off lemma from claim 8.7.6, for any separator S of $\widetilde{\tau}_P$, and write $S(\tau)$ as the MVS for τ (to be distinguished from kernel-partition), we have

$$\begin{aligned}
&(\frac{1}{k})^{|V(\widetilde{\tau}_P)|-\frac{|U_Y|+S(V_Y)|}{2}} \cdot \sqrt{n}^{|V(\widetilde{\tau}_P)|-|S|+|I(\widetilde{\tau}_P)|} \\
&\leq (\frac{\sqrt{n}}{k})^{|V(\tau\circ Y)|-\frac{|U_\tau|+|V_{Y'}|}{2}} \cdot \sqrt{n}^{\frac{|U_\tau|+|V_\tau|}{2}-|S(\tau)|+|I(\tau)|-\frac{|I_{dc}(V_Y)|}{2}}
\end{aligned}$$

Plugging into the original equation and writing

$$|\text{CP}(T(V_Y))| \leq |\text{CP}(T(V_Y) \setminus T(U_Y))| + |\text{CP}(T(V_\tau))|$$

as $T(V_\tau) = T(U_Y) \subseteq T(V_Y)$, we have

$$\begin{aligned}
\lambda^{\approx}_K(\widetilde{\tau}_P)_{\text{CP}(\widetilde{\tau}_P)} \cdot w_K(\widetilde{\tau}_P) \leq\ & (\frac{\sqrt{n}}{k})^{|V(\tau\circ Y)|-\frac{|U_\tau|+|V_{Y'}|}{2}} \cdot Ev(\tau)_{\text{CP}(\tau)} \cdot \sqrt{n}^{\frac{|U_\tau|+|V_\tau|}{2}-|S(\tau)|+|I(\tau)|-\frac{|I_{dc}(V_Y)|}{2}} \\
&\cdot \sqrt{k}^{|\text{CP}(T(V_\tau))|} \cdot (\frac{1}{\sqrt{k}})^{|T(U_\tau)|} \cdot \underbrace{\sqrt{k}^{|\text{CP}(T(V_Y)\setminus T(U_Y))|} \cdot (\frac{1}{\sqrt{k}})^{|T(V_Y)\setminus T(U_Y)|}}_{\leq 1} \\
&\cdot Ev(Y)_{\text{CP}(Y)} \cdot k^{|\text{CP}(Y)|-\frac{|\text{CP}(U_Y)|+|\text{CP}(V_Y)|}{2}}
\end{aligned}$$

Finally, and notice $Ev(Y)_{\text{CP}(Y)} \cdot k^{|\text{CP}(Y)|-\frac{|\text{CP}(U_Y)|+|\text{CP}(V_Y)|}{2}} \leq \sqrt{k}^{I_{dc}(V_Y)}$ from claim 8.7.8, plugging the bound from the previous level for τ, we have

$$\frac{\lambda^{\approx}(\widetilde{\tau}_P)_{\text{CP}} \cdot w(\widetilde{\tau}_P)_{\text{CP}}}{\lambda^{\approx}(\tau)_{\text{CP}(\tau)} \cdot w(\tau)_{\text{CP}(\tau)}} \leq (\frac{\sqrt{n}}{k})^{|V(Y)|-\frac{|U_Y|+|V_Y|}{2}}$$

□

Multiplying our slack across each intersection-level, and the extension pieces within,

our base piece bounds from lemma 8.9.25, lemma 8.9.12, and extension bounds from lemma 8.9.18 and lemma 8.9.20, and our slack bound for approximating Ev (section 8.11.1) and norm bound gives us

Lemma 8.9.26. *For any irreducible right shape with t-level of intersections σ, we have*

$$\|\lambda_K(\sigma)_{CP(\sigma)} \cdot \mathsf{M}_{K(\sigma),cp}\| \leq slack(\sigma, CP)$$

for

$$slack(\sigma, CP) := (\frac{\sqrt{n}}{k})^{\Sigma_i |V(\gamma_i) \setminus D(\gamma_i)| - \frac{|U_{\gamma_i}|+|S(V_{\gamma_i})|}{2} + |V(\tau)| - \frac{|U_\tau|+|S(V_\tau)|}{2}}$$
$$\cdot O(|rainbow(E(\sigma))_{CP}|)^{|rainbow(E(\sigma))_{CP}|} \cdot O(|V(\tau) \setminus U_\tau \cap V_\tau|)^{2|V(\sigma) \setminus U_\sigma \cap S(V_\sigma)|)}$$

8.9.7 Well-conditionedness of L

Lemma 8.9.27. *For the left-(color)-shape matrix,*

$$L := \sum_{\substack{(\sigma,CP):\text{color-left-shape} \\ |V(\sigma)| \leq D_V}} \lambda_L(\sigma, CP) \cdot \mathsf{M}_{(\sigma,CP)},$$

we have

$$LL^T \succeq n^{-O(\mathsf{dsos})} Id$$

Proof. This is identical to analogous lemmas in prior works while we include the proof here for completeness. For clarity, we suppress the dependence on truncation limit $|V(\sigma)| \leq D_V$ as it holds throughout, and we write $\sigma = (\sigma, \mathrm{CP}_\sigma)$ for a color-shape without making the dependence on CP explicit while noting that σ, σ' are color-shapes throughout . Observe

the block-structure in LL^T,

$$LL^T = \sum_{j=0}^{\text{dsos}} \left(\sum_{\substack{(\sigma,\text{CP}):\text{color-left-shape} \\ |V(\sigma)| \le D_V \\ |V_\sigma| = j}} \lambda_L(\sigma,\text{CP}) \cdot \mathsf{M}_{(\sigma,\text{CP})} \right) \left(\sum_{\substack{(\sigma,\text{CP}):\text{color-left-shape} \\ |V(\sigma)| \le D_V \\ |V_\sigma| = j}} \lambda_L(\sigma,\text{CP}) \cdot \mathsf{M}_{(\sigma,\text{CP})} \right)^T$$

In particular, using Lemme E.56 [JPRX23] , we note that it is sufficient to find non-negative weights $\{w_j\}$ for $|V_\sigma| = j$ that rescales columns of L, and show that

$$\sum_{j=0}^{\text{dsos}} \left(\sum_{\substack{(\sigma,\text{CP}):\text{color-left-shape} \\ |V(\sigma)| \le D_V \\ |V_\sigma| = j}} \sqrt{w_j} \cdot \lambda_L(\sigma,\text{CP}) \cdot \mathsf{M}_{(\sigma,\text{CP})} \right) \left(\sum_{\substack{(\sigma,\text{CP}):\text{color-left-shape} \\ |V(\sigma)| \le D_V \\ |V_\sigma| = j}} \sqrt{w_j} \lambda_L(\sigma,\text{CP}) \cdot \mathsf{M}_{(\sigma,\text{CP})} \right)^T \succeq Id$$

which would then imply in the unscaled version

$$LL^T \succeq \frac{1}{\max_j w_j} Id\,.$$

Unpacking the above rescaled columns, we have

$$\sum_{j=0}^{\text{dsos}} \left(\sum_{\substack{(\sigma,\text{CP}):\text{color-left-shape} \\ |V(\sigma)| \le D_V \\ |V_\sigma| = j}} \sqrt{w_j} \cdot \lambda_L(\sigma,\text{CP}) \cdot \mathsf{M}_{(\sigma,\text{CP})} \right) \left(\sum_{\substack{(\sigma,\text{CP}):\text{color-left-shape} \\ |V(\sigma)| \le D_V \\ |V_\sigma| = j}} \sqrt{w_j} \lambda_L(\sigma,\text{CP}) \cdot \mathsf{M}_{(\sigma,\text{CP})} \right)^T$$
$$= \sum_{j=0}^{\text{dsos}} w_j \left(Id_j + \sum_{\sigma:\text{non-trivial}} \lambda_\sigma (\mathsf{M}_\sigma + \mathsf{M}_{\sigma^T}) + \sum_{\substack{\sigma,\sigma' \\ \text{non-trivial}}} \lambda_\sigma \mathsf{M}_\sigma \lambda_{\sigma'} \mathsf{M}_{\sigma'}^T \right)$$

Since the last two term is PSD, it suffices to focus on the first term. Next we observe that for each term in the summand above, i.e. for any fixed j,

$$\sum_{\sigma:\text{non-trivial}} \lambda_L(\sigma)(\mathsf{M}_\sigma + \mathsf{M}_{\sigma^T}) \succeq \sum_{|U|<j} -2\lambda_L(\sigma)\|\mathsf{M}_\sigma\| \cdot Id_U$$

Summing over all j gives us

$$\sum_{j=0}^{\mathsf{dsos}} w_j \left(Id_j + \sum_{\sigma:\text{non-trivial}} \lambda_\sigma(\mathsf{M}_\sigma + \mathsf{M}_{\sigma^T}) \right) \succeq \sum_v w_v \cdot Id_v - \sum_\sigma w_v \lambda_L(\sigma)\|\mathsf{M}_\sigma\| Id_U$$

Picking $w_u = O((\frac{1}{k})^{u/2})$ gives us the desired lower bound of Id by applying Lemma E.57. That said, this implies a final lower bound of

$$LL^T \succeq \frac{1}{\max w_u} Id = (\frac{1}{k})^{O(\mathsf{dsos})} \cdot Id$$

via observing the largest $|V_\sigma|$ has size at most dsos in the decomposition. □

Lemma 8.9.28. *For Q_{trunc} the term for truncation terms such that $LQ_{trunc}L^T = M_{trunc}$, we have*

$$\|Q_{trunc}\| \leq o(1)$$

This is standard as in prior works, picking truncation parameter $D_V = \Theta(\mathsf{dsos} \log n)$ ensures us that

$$\|\lambda(\tau)_{\text{CP}} \cdot \mathsf{M}_{\tau,\text{CP}}\| \leq n^{-\Omega(\mathsf{dsos})}$$

where the constant slack in the exponent can be tweaked by modifying the constant in D_V. Combining with our count function gives the corresponding bound for $\|M_{trunc}\| \leq O(n^{-\Omega(\mathsf{dsos})})$. The proof is identical to its analog in [BHK+16, JPR+22]. As a corollary,

combining with our well-conditiondess bound, we have

$$\|Q_{trunc}\| \leq \|L^{-1}\|^2 \cdot \|M_{trunc}\| \leq o(1)$$

by taking a large enough constant for $D_V = c \cdot \mathsf{dsos} \log n$.

8.10 Spectral Norm Bound for Graph Matrix for Color Shapes

We recall the definition of graph matrix for color-shape,

Definition 8.10.1 (Graph matrix for Color-Shape)**.** *Given a color-shape (α, CP), we define its corresponding graph $\mathsf{M}(\alpha)_{CP}$ as a matrix with rows and columns indexed by any order-tuple of $[n] \times [k]$ of size at most $\frac{\mathsf{dsos}}{2}$ with entries given by*

$$\mathsf{M}_{(\alpha,CP)}[(U,\beta_U),(V,\beta_V)] := \sum_{\substack{\psi:V(\alpha)\to[n] \\ \text{injective} \\ U=\psi(U_\alpha),V=\psi(V_\alpha)}} \sum_{\substack{\widehat{\beta}:V(\alpha)\to[k] \\ \widehat{\beta}\in CP \\ \widehat{\beta}(U)=\beta_U,\widehat{\beta}(V)=\beta_V}} \prod_{(i,j)\in E(\alpha)} \chi_{\psi(i),\psi(j)}(G)\,.$$

Again, this is the usual graph matrix except each vertex receives an additional label in $[k]$ for its color, and CP is a grouping of coloring for vertices in the shape.

Trace moment method Our proof proceeds by analyzing trace moments of matrices using that for any A, the spectral norm $\|A\|_{spec} \leq \mathrm{tr}\big((AA^\top)^q\big)^{1/2q}$. Taking q to be logarithmic in the dimension of A suffices to get a bound on $\|A\|$ sharp up to $1+\delta$ for an arbitrarily small $\delta > 0$.

The trace power can be expanded as a sum of weighted walks of length $2q$ over the entries of M_α, and each walk can be viewed as a labeling of a graph composed of q blocks of α and α^T: each term in the expansion of $\mathrm{tr}\big((M_\alpha M_\alpha^\top)^q\big)$ corresponds to a labeling of the vertices in q copies of α and $\alpha^\top$ with labels from $[n]$ (i.e, vertices of the graph G) and a

label from $[k]$ satisfying some additional constraints that correspond to a valid walk. In particular, we recover the usual graph matrix without colors if we restrict ourselves to labels in $[n]$. The following definition captures these constraints.

Definition 8.10.2 (Color-Shape walk and its valid labelings)**.** *Let α be a shape and $q \in \mathbb{N}$. For each walk P, let $G_P(\alpha, 2q)$ be the* Color-Shape Walk *graph of the shape-walk P vertices on vertices $V_P(\alpha, 2q)$ formed by the following process,*

1. *Take q copies of $\alpha_1, \ldots, \alpha_q$ of α, and q copies of $\alpha_1^T, \ldots, \alpha_q^T$ of α^T;*
2. *For each copy of α_i (and α_i^T), we associate it with a labeling, i.e., an injective map $\psi_i : V(\alpha) \to [n]$ (and respectively ψ_i' for α^T);*
3. *For each $i \in [q]$, we require the boundary labels to be consistent as ordered tuples, i.e. $\psi_i(U_\alpha) = \psi_{i-1}'(V_\alpha)$ and $\psi_i'(U_{\alpha^T}) = \psi_i(V_{\alpha_i})$;*
4. *Additionally, each such walk must be closed, i.e., $\psi_q'(V_{\alpha^T}) = \psi_1(U_\alpha)$ as a tuple-equality;*
5. *Each vertex in $V_P(\alpha, 2q)$ receive a color-label in $[k]$;*
6. *We call each α_i and α_i^T block a block-step in the walk.*

For each walk-graph , we associate it with a natural decomposition $P = \psi_1 \circ \psi_1' \circ \psi_2 \circ \cdots \circ \psi_q \circ \psi_q'$.

Proposition 8.10.3. *For shape τ and any color-partition CP, for any $q > \Theta((|U_\tau| + |V_\tau|) \operatorname{poly}\log n)$, for any $\epsilon > 0$, we have*

$$\mathbb{E}[\mathrm{Tr}\Big(\mathsf{M}_{\tau,CP}\mathsf{M}_{\tau,CP}^T\Big)^q] \leq \big((1+\epsilon)\cdot B_q(\tau, CP)\big)^{2q}$$

for candidate norm bound

$$B_q(\tau, CP) := \max_{S:separator} \big(\sqrt{n}\cdot(2q\cdot|V(\tau)|)\big)^{|V(\tau)\setminus S|} \cdot \Big(\frac{\sqrt{n}}{2q\cdot|V(\tau)|}\Big)^{|I(\tau)|} \left(\sqrt{\frac{1-p}{p}}\right)^{|E(S)|} \cdot (2q\cdot|V(\tau)|)^{|S\setminus U_\tau\cap}$$

$$\cdot \max_{\beta \in CP} k^{|\beta(\tau)| - \frac{|\beta(U_\tau)|+|\beta(V_\tau)|}{2}} \cdot (2|V(\tau)|q)^{|V(\tau)\setminus U_\tau \cap V_\tau|}$$
$$\leq \max_{S:separator} \sqrt{n}^{|V(\tau)\setminus S|} \cdot \sqrt{n}^{I(\tau)|} \cdot \left(\sqrt{\frac{1-p}{p}}\right)^{|E(S)|}$$
$$\cdot (2q \cdot |V(\tau) \setminus U_\tau \cap V_\tau|)^{2|V(\tau)\setminus U_\tau \cap V_\tau|} \cdot k^{|CP(\tau)| - \frac{|CP(U_\tau)|+|CP(V_\tau)|}{2}}$$

where as usual $I(\tau)$ is the set of isolated vertices, and $|\beta(\tau)|$ is the number of distinct colors of $V(\tau)$ under β, and analogously $|\beta(U_\tau)|, |\beta(V_\tau)|$ the number of distinct colors under β in U_τ and V_τ.

With some abuse of notation, we also ignore the dependence on β and write

$$k^{|CP(\tau)| - \frac{|CP(U_\tau)|+|CP(V_\tau)|}{2}} := \max_{\beta \in CP} k^{|\beta(\tau)| - \frac{|\beta(U_\tau)|+|\beta(V_\tau)|}{2}}$$

by fixing CP to be the coloring maximizer $\beta \in CP$, so that we can view CP also a coloring with label in $[k]$ abstracted out, and have $|CP(\tau)| = |\beta(\tau)|$ and analogously $|CP(U_\tau)| = \beta(U_\tau)|$, $|CP(V_\tau)| = |\beta(V_\tau)|$.

As a corollary, we obtain the following norm bound for graph matrix that holds with high probability,

Corollary 8.10.4. *With probability at least , for any $\epsilon > 0$, we have*

$$\mathbf{Pr}\left[\|M_{\tau,CP}\| > B_q(\tau, CP)\right] \leq (\frac{1}{1+\epsilon})^{2q} = (\frac{1}{c_{\text{norm}}})^q$$

for some constant $c_{\text{norm}} > 1$.

For application to our SoS lower bounds, standard to prior works in [BHK+16, GJJ+20b, JPR+22, JPRX23, KPX24] we pick a global parameter $q = \Theta(D_V)$ so that the bound holds for all graph matrices after a union bound over all shapes of concerned. We are now ready to prove the block-value bound via race moment calculation.

Proof. We apply trace moment method as introduced. The dependence of norm bound in the factor of $[n]$ follows straightforwardly from [JPR+22] and [KPX24] while we restate the proof here for completeness and to highlight its adaptation to the factor of $[k]$, i.e., factors incurred by identifying color of each vertex. Using ideas from Sec 2.1 in [KPX24] and [HKPX23], the crux of finding $B_q(\tau, \mathrm{CP})$ is to identify a vertex/edge-factor assignment scheme for each block-step in the trace expansion, and verify that

$$\mathbb{E}[\mathrm{Tr}\left(\mathsf{M}_{\tau,\mathrm{CP}}\mathsf{M}_{\tau,\mathrm{CP}}^T\right)^q] \leq \text{Cost of Identifying the Start} \cdot B_q(\tau, \mathrm{CP})^{2q}\,.$$

for some candidate upper bound $B_q(\tau, \mathrm{CP})$ that upper bounds the contribution of each step (or more formally, block-step). We refer interested reader to [KPX24] for a thorough introduction, while we highlight the adaptation to our setting. In our case, the cost of identifying the start is at most $(nk)^{|U_\tau|}$, and we note that we have $(nk)^{|U_\tau|} \leq (1+\epsilon)^{2q}$ for any $\epsilon > 0$ provided $q \geq \Theta(|U_\tau| \operatorname{poly}\log n)$. That said, our focus is simply to identify a candidate upper bound function B_q for each block, which would then give us

$$\mathbb{E}[\mathrm{Tr}\left(\mathsf{M}_{\tau,\mathrm{CP}}\mathsf{M}_{\tau,\mathrm{CP}}^T\right)^q] \leq \left((1+\epsilon)\cdot B_q(\tau, \mathrm{CP})\right)^{2q}\,.$$

Each block-function is associated with a factor-assignment scheme, and we consider the following scheme.

1. For each vertex appearing for the first/last time in the given block, assign a factor of $\sqrt{n}\cdot(2q|V(\tau)\setminus U_\tau \cap V_\tau|)$;
2. For each vertex appearing for the middle time (but also not in $U_\tau \cap V_\tau$) in the given block, assign a factor of $2q|V(\tau)\setminus U_\tau \cap V_\tau|$;
3. For each edge making a middle appearance (i.e. an H-step in the language of [KPX24]), assign a factor of $\sqrt{\frac{1-p}{p}}$.

Claim 8.10.5. *The above is a valid assignment scheme.*

Proof. To see that this is a valid assignment scheme, note that one can write

$$\mathbb{E}[\mathrm{Tr}\Big(\mathsf{M}_{\tau,\mathrm{CP}}\mathsf{M}_{\tau,\mathrm{CP}}^{T}\Big)^{q}] = \sum_{P:walk}\prod_{e\in P}\mathbb{E}[\chi_e(G)^{\mathrm{mul}_P(e)}]$$

by straightforward trace expansion, and there are two kinds of factors in the above, vertex-factor (for combinatorial counting) that is used to identify the walk, and analytical edge-factor from expectation of the random variables.

In our case, as we are working with normalized Fourier variables, observe that the above is 0 if any edge appears only once, and each edge gives value 1 if appearing twice, and moreover, a value $\sqrt{\frac{1-p}{p}}^{t-2}$ for any edge that appears t-times in the walk for $t > 2$. That said, we distribute the $\sqrt{\frac{1-p}{p}}^{t-2}$ factor of each edge to each block it makes a middle appearance, and crucially, notice that we simply assign a factor 1 if the edge is making first/last time.

For vertex factors (restricted to labels in $[n]$), notice to identify the walk, it suffices for us to put down a label in $[n]$ the first time a vertex appears, and a cost of $2|V(\tau)|q$ each time the vertex appears beyond the first time. Therefore, we crucially distribute the factor n to the first and last time of which the vertex appears, which is therefore a $\sqrt{n}$ for each appearance. Moreover, we give an additional factor $2|V(\tau)|q$ for each time the vertex appears as a loose upper bound. □

We next show how to relate the block-value bound $B_q(\tau)$ to vertex separators. For each block-step, include each vertex making middle appearance in the block into the set S.

Claim 8.10.6. *S is a separator for τ unless some edge appears only once in the walk.*

Proof. To see this, suppose S is not a separator, we have a path from U_τ to V_τ that does not pas through S. By construction of our set S, no vertex along the path is making a

middle appearance. Since by definition, a vertex cannot be making first appearance in U_τ or last appearance in V_τ, starting from the vertex making last appearance in U_τ, at some point the path must pass through a switch-edge, an edge with one endpoint making last appearance and one endpoint making its first appearance. Any such edge appears only once throughout the walk, and this proves the claim. □

That said, to upper bound the contribution of each BlockStep, it is sufficient for us to upper bound its corresponding value from the set S as a separator. For each separator S, notice the value of the blockstep is upper bounded by its corresponding factors on the RHS. This completes the proof to our factors in n.

To see the factor in k, consider the following factor-assignment scheme similar to the above,

1. For each color appearing for the first/last time, assign a factor of $\sqrt{k}$;
2. For each color used by vertex outside $U_\tau \cap V_\tau$ appearing for the middle time, assign a factor of $2|V(\tau)|q$.
3. Vertices in $U_\tau \cap V_\tau$ have their color fixed, and therefore no factor is needed.

Therefore, for each block with coloring β for τ, its contribution of color-factor is at most $k^{|\beta(\tau)|-\frac{|\beta(U_\tau)|+|\beta(V_\tau)|}{2}} \cdot (2|V(\tau)|q)^{|V(\tau)\setminus U_\tau \cap V_\tau|}$ where we use a loose upper bound as only vertices in $V(\tau) \setminus U_\tau \cap V_\tau$ may require encoding for its color. Since we do not optimize the dependence of subpolynomial factors in this work, we subsume this factor into the dependence of $2|V(\tau)|q$ in our final bound. □

8.11 Deferred Proofs

8.11.1 Correction Magnitude Bound

Proposition 8.11.1 (Restatement of proposition 8.4.2)**.** *For any component C,*

$$|\mathsf{Rb}_{\mathsf{pinned}}(C)| \leq 2^{|V(C)|} \cdot (2 \cdot |E(C)|)^{|E(C)|}.$$

Proof. We focus on component C that is not a ground-component, i.e., a tree or a single cycle. For non-ground component C, we first observe that

$$\sum_{\beta:\text{rainbow},\beta(v)=a} |\mathsf{Colorval}[C]_\beta| \leq \frac{1}{k-1}$$

as we can reserve a cross-color edge (with a factor of $\frac{1}{k-1}$) unused in the above process. and apply the prior argument to get a bound of $\frac{1}{k-1}(k-1) = 1$ for each vertex beyond the first one. Moreover, since the value for pure-color (once the first vertex's color is pinned) is 1, we shall now focus on the factors from mixed-color.

By our recursive process, any mixed-coloring β corresponds to $\mathit{pure}(C,\beta) \neq \emptyset$ be the subset of edges that get removed. Let $\mathcal{G} = \{G_i\}_i$ be the sequence of connected components obtained by removing $\mathit{pure}(C,\beta)$ from $E(C)$. Note that any of the resulting components be G_i under β is a rainbow-color component, and the value is defined as

$$\left|Ev(C)_\beta\right| = |\prod_{G_i \in \mathcal{G}} Ev(G_i)_{\beta(G_i)}|$$

Let us now group the collection of mixed-colorings β based on the pure-edge subset E_P and observe the following. For any $E_P \neq \emptyset$, we note that once we pin down an arbitrary vertex v in C to an arbitrary color, there is a path to traverse the collection of connected components using pure-color edge, such that each component (including the first one being pinned down by color for v) has at least one vertex's color pinned.

We have

$$\sum_{\substack{\beta:\text{mixed} \\ Pure(C,\beta)=E_P \\ \beta(v)=a}} \left|Ev(C)_\beta\right| \leq \sum_{\substack{\beta::\text{mixed} \\ Pure(C,\beta)=E_P \\ \beta(v)=a}} \prod_{G_i\in\mathcal{G}} \left|Ev(G_i)_{\beta(G_i)}\right|$$

$$\leq \prod_{G_i\in\mathcal{G}} \left(\sum_{\substack{\beta_i:V(G_i)\to[k] \\ \text{rainbow} \\ \beta(v_i)\text{pinned for some } v_i\in V(G_i)}} \left|Ev(G_i)_{\beta_i}\right| \right)$$

where we observe the second line inequality as given the first color $\beta(v) = a$, any component has its root vertex pinned while potentially more, hence there are potentially more terms on the RHS while any term on the LHS can be mapped injectively to a decomposition into $\{\beta(R_i)\}$ on the RHS. Finally, notice

$$\left(\sum_{\substack{\beta_i:V(G_i)\to[k] \\ \text{rainbow} \\ \beta(v_i)\text{pinned for some } v_i\in V(G_i)}} \left|Ev(G_i)_{\beta_i}\right| \right) \leq \mathsf{Rb}_{\mathsf{pinned}}(G_i)$$

by recalling the original definition, we have arrived at

$$\sum_{\substack{\beta:\text{mixed} \\ Pure(C,\beta)=E_P \\ \beta(v)=a}} \left|Ev(C)_\beta\right| \leq \prod_{G_i\in\mathcal{G}} \mathsf{Rb}_{\mathsf{pinned}}(G_i)$$

where $\mathcal{G}$ is the collection of components obtained from removing E_P. Let us now return

to the original equation bounding $\mathsf{Rb}_{\mathsf{pinned}}(C)$,

$$\begin{aligned}\mathsf{Rb}_{\mathsf{pinned}}(C) &:= \underbrace{\sum_{\beta:\text{rainbow}} |\mathsf{Colorval}[C]_\beta|}_{\leq \frac{1}{k-1}} + \underbrace{\sum_{\beta:\text{pure}} |\mathsf{Colorval}[C]_\beta|}_{=1} + \sum_{\beta:\text{mixed}} |\mathsf{Colorval}[C]_\beta| + |\kappa(C)_\beta| \\ &\leq 1 + \frac{1}{k-1} + \sum_{\beta:\text{mixed}} |Ev(C)_\beta| \\ &\leq 1 + \frac{1}{k-1} + \sum_{\emptyset \neq E_P \subseteq E(C)} \prod_{G_i \in \mathcal{G}(E_P)} \mathsf{Rb}_{\mathsf{pinned}}(G_i)\end{aligned}$$

Applying the expansion recursively for the Rb on the RHS until G_i is a ground-component. By the sum-product structure, expanding out all the mixed-value terms, each term in the ultimate expansion corresponds to a monomial given by a product of a collection of ground-level components $\mathcal{G} = \{G_i\}$ obtained by removing edges in C, and each monomial product has magnitude at most

$$\prod_{G_i \in \mathcal{G}} |\mathsf{Rb}_{\mathsf{pinned}}(G_i)| \leq 2^{\sum_i (|V(G_i)|-1)} \leq 2^{|V(C)|}$$

and the coefficient of such monomial is at most $(1 + \frac{1}{k-1})^E \leq e^{\frac{E}{k-1}} \leq (1 + o(1))^{|E|}$ for the dense regime when $|E| = o(k)$.

It now remains to bound the number of such product terms. Notice each product corresponds to an ordering of edges in $E(C)$ being removed that gives rise to the final decomposition of $\mathcal{G} = \{G_i\}$, there are $2^{|E(C)|}$ many of such collections. For any collection of ground-components, notice we have at most $|E(C)|$ levels as each level some edge gets removed, and since each term such collection appears corresponds to some ordering of edge-removal, we have at most $|E(C)|^{|E(C)|}$ many such terms. Therefore, we have proven a bound of

$$\mathsf{Rb}_{\mathsf{pinned}}(C) \leq (1 + o_n(1)2)^{|E(C)|} \cdot 2^{|V(C)|} \cdot |E(C)|^{|E(C)|}.$$

As a corollary, we also have

$$\mathsf{Rb}(C) \leq 2^{|V(C)|} \cdot (2 \cdot |E(C)|)^{|E(C)|}.$$

□

Lemma 8.11.2 (Slack Bound for Proxy Coefficient)**.** *For a shape τ, we have*

$$\left| \frac{Ev(\tau)_{CP}}{Ev^{\approx}(\tau)} \right| \leq O(|V(\tau)|^{|V(\tau)|^2})$$

where we remind the reader that $Ev^{\approx}(\tau) \coloneqq \prod_{C_i \in \mathcal{G}(\tau, CP)} \left(\frac{1}{k}\right)^{|V(C_i)|-1}$.

Proof. Observe that the approximation is only applied on each rainbow component, and for rainbow component C the exact coefficient is given by

$$\prod_{C_i} |\mathsf{Colorval}[C_i]_{rainbow} + \kappa(C+i)_{rainbow}| \leq \prod_{C_i} \frac{k \cdot |Rb_{pinned}(C)|}{|Rs(C)|}$$

where we remind the reader that $|Rs(C)|$ is the collection of rainbow coloring for C, and notice that there are at least $|k - D_V|^{|V(C)|}$ many rainbow coloring for each component C. That said, for each component, we have factor

$$\frac{k}{|Rs(C)|} \leq k \cdot (\frac{1}{k})^{|V(C)|} \cdot \left(\frac{k}{k - |D_V|}\right)^{|V(C)|} \leq (1 + \frac{D_V}{k})^{|V(C)|} = (1 + o(1))^{|V(C)|}$$

as we consider $D_V = \Theta(\mathsf{dsos} \log n)$ and $\mathsf{dsos} \ll k^{1/c_\eta}$ for some constant $c_\eta > 0$ in our hardness result.

Plugging our bound for $|Rb_{pinned}(C)|$, the slack from $|Rb_{pinned}(C)|$ ultimately domi-

nates, and we have

$$\left| \frac{Ev(\tau)_{\mathrm{CP}}}{Ev^{\approx}(\tau)} \right| \leq O(|V(\tau)|^{2^{|V(\tau)|^2}})$$

□

Bibliography

[Abb17a] Emmanuel Abbe. Community detection and stochastic block models: Recent developments. *J. Mach. Learn. Res.*, 18(1):64466531, jan 2017.

[Abb17b] Emmanuel Abbe. Community detection and stochastic block models: Recent developments. *Journal of Machine Learning Research*, 18(177):1–86, 2017.

[ABH14] Emmanuel Abbe, Afonso S. Bandeira, and Georgina Hall. Exact recovery in the stochastic block model, 2014.

[ABS15] Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. *J. ACM*, 62(5), November 2015.

[AGJ20] Gérard Ben Arous, Reza Gheissari, and Aukosh Jagannath. Algorithmic thresholds for tensor pca. *The Annals of Probability*, 48(4):2052–2087, 2020.

[AGKM23] Omar Alrabiah, Venkatesan Guruswami, Pravesh K. Kothari, and Peter Manohar. A near-cubic lower bound for 3-query locally decodable codes from semirandom csp refutation. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing (STOC)*, pages 1438–1448, Orlando, FL, USA, June 2023. ACM.

[AMP20] Kwangjun Ahn, Dhruv Medarametla, and Aaron Potechin. Graph matrices: Norm bounds and applications. abs/1604.03423, 2020.

[ARV04] Sanjeev Arora, Satish Rao, and Umesh Vazirani. Expander flows and a $\sqrt{\log n}$-approximation to sparsest cut. 2004.

[AS15] Emmanuel Abbe and Colin Sandon. Community detection in general stochastic block models: Fundamental limits and efficient algorithms for recovery. In *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 670–688, 2015.

[AS18] Emmanuel Abbe and Colin Sandon. Proof of the achievability conjectures for the general stochastic block model. *Communications on Pure and Applied Mathematics*, 71, 2018.

[BABB21] Enric Boix-Adserà, Matthew Brennan, and Guy Bresler. The average-case complexity of counting cliques in Erdős–Rényi hypergraphs. *SIAM Journal on Computing*, pages FOCS19–39, 2021.

[Ban24a] Afonso S. Bandeira. Ten open problems involving matrices, randomness, graphs, and more. Oberwolfach Report for Workshop 2417 (21.04.202426.04.2024), 2024. https://publications.mfo.de/handle/mfo/4149.

[Ban24b] Afonso S. Bandeira. Tensor pca and the kikuchi algorithm (problemă9). Randomstrasseă101: Open Problems 2024, December 2024. https://randomstrasse101.math.ethz.ch/posts/Kikuchi-TensorPCA/#MontanariRichardTensorPCA.

[BAP05] Jinho Baik, Gérard Ben Arous, and Sandrine Péché. Phase transition of the largest eigenvalue for nonnull complex sample covariance matrices. *The Annals of Probability*, 33(5):1643–1697, 2005.

[BB20] Matthew Brennan and Guy Bresler. Reducibility and statistical-computational gaps from secret leakage. In *Conference on Learning Theory*, pages 648–847. PMLR, 2020.

[BBH+12] Boaz Barak, Fernando G. S. L. Brandão, Aram Wettroth Harrow, Jonathan A. Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. *CoRR*, abs/1205.4484, 2012.

[BBH18] Matthew Brennan, Guy Bresler, and Wasim Huleihel. Reducibility and computational lower bounds for problems with planted sparse structure. In *Conference On Learning Theory*, pages 48–166. PMLR, 2018.

[BBH+20] Matthew Brennan, Guy Bresler, Samuel B. Hopkins, Jerry Zheng Li, and Tselil Schramm. Statistical query algorithms and low-degree tests are almost equivalent. *ArXiv*, abs/2009.06107, 2020.

[BBK+20] Afonso S. Bandeira, Jess Banks, Dmitriy Kunisky, Cristopher Moore, and Alexander S. Wein. Spectral planting and hardness of refuting cuts, colorability, and communities in random graphs. *arXiv preprint arXiv:2008.12237*, 2020.

[BBK+21a] Mitali Bafna, Boaz Barak, Pravesh K. Kothari, Tselil Schramm, and David Steurer. Playing unique games on certified small-set expanders. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2021, page 16291642, New York, NY, USA, 2021. Association for Computing Machinery.

[BBK+21b] Afonso S. Bandeira, Jess Banks, Dmitriy Kunisky, Christopher Moore, and Alexander S. Wein. Spectral planting and the hardness of refuting cuts, colorability, and communities in random graphs. In Mikhail Belkin and Samory Kpotufe, editors, *Proceedings of the 34th Conference on Learning Theory (COLT)*, volume 134 of *Proceedings of Machine Learning Research*, pages 410–473. PMLR, August 2021.

[BBvH23] Afonso S. Bandeira, March T. Boedihardjo, and Ramon van Handel. Matrix concentration inequalities and free probability. *Inventiones mathematicae*, 234(1):419487, June 2023.

[BCK15] Boaz Barak, Siu On Chan, and Pravesh K. Kothari. Sum of Squares Lower Bounds from Pairwise Independence. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 97–106, 2015.

[BCSvH24] Afonso S. Bandeira, Giorgio Cipolloni, Dominik Schröder, and Ramon van Handel. Matrix concentration inequalities and free probability ii. two-sided bounds and applications, 2024.

[BEAH+24] Afonso S. Bandeira, Ahmed El Alaoui, Samuel B. Hopkins, Tselil Schramm, Alexander S. Wein, and Ilias Zadik. The franz-parisi criterion and computational trade-offs in high dimensional statistics. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA, 2024. Curran Associates Inc.

[BGG+16] Vijay Bhattiprolu, Mrinalkanti Ghosh, Venkatesan Guruswami, Euiwoong Lee, and Madhur Tulsiani. Weak decoupling, polynomial folds, and approximate optimization over the sphere. *arXiv preprint*, 2016. Accepted at FOCS 2017.

[BGL17] Vijay Bhattiprolu, Venkatesan Guruswami, and Euiwoong Lee. Sum-of-squares certificates for maxima of random tensors on the sphere. In *Proceedings of the 11th International Conference on Approximation, Randomization and Combinatorial Optimization (APPROX/RANDOM 2017)*, volume 81 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 31:1–31:20, 2017.

[BHK+16] B. Barak, S. B. Hopkins, J. Kelner, P. Kothari, A. Moitra, and A. Potechin. A nearly tight sum-of-squares lower bound for the planted clique problem. pages 428–437, 2016.

[BHK+19] Boaz Barak, Samuel Hopkins, Jonathan Kelner, Pravesh K Kothari, Ankur Moitra, and Aaron Potechin. A Nearly Tight Sum-of-Squares Lower Bound for the Planted Clique Problem. *SIAM Journal on Computing*, 48(2):687–735, 2019.

[BHKL22] Mitali Bafna, Max Hopkins, Tali Kaufman, and Shachar Lovett. High dimensional expanders: Eigenstripping, pseudorandomness, and unique games. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1069–1128. SIAM, 2022.

[BHKX22] Mitali Bafna, Jun-Ting Hsieh, Pravesh K. Kothari, and Jeff Xu. Polynomial-time power-sum decomposition of polynomials. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 956–967, 2022.

[BK21] Ainesh Bakshi and Pravesh K. Kothari. List-decodable subspace recovery: Dimension independent error in polynomial time. In *Proceedings of the Thirty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '21, page 12791297, USA, 2021. Society for Industrial and Applied Mathematics.

[BKM19] Jess Banks, Robert Kleinberg, and Cristopher Moore. The lovász theta function for random regular graphs and community detection in the hard regime. *SIAM Journal on Computing*, 48(3):1098–1119, 2019.

[BKS17] Boaz Barak, Pravesh K. Kothari, and David Steurer. Quantum entanglement, sum of squares, and the log rank conjecture. pages 975–988. ACM, 2017.

[BLM15] Charles Bordenave, Marc Lelarge, and Laurent Massoulié. Non-backtracking spectrum of random graphs: community detection and non-regular Ramanujan graphs. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 1347–1357. IEEE, 2015.

[BLNNvH25] Afonso S. Bandeira, Kevin Lucca, Petar Nizic-Nikolac, and Ramon van Handel. Matrix chaos inequalities and chaos of combinatorial type. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, page 795805, New York, NY, USA, 2025. Association for Computing Machinery.

[BMMP23] Afonso S. Bandeira, Antoine Maillard, Shahar Mendelson, and Elliot Paquette. Fitting an ellipsoid to a quadratic number of random points. 2023.

[BMR19] Jess Banks, Sidhanth Mohanty, and Prasad Raghavendra. Local statistics, semidefinite programming, and community detection. *CoRR*, abs/1911.01960, 2019.

[BNN25] Afonso S. Bandeira and Petar Nizi-Nikolac. Norm bounds for kikuchi random matrices, 2025. Personal communication.

[Bor19] Charles Bordenave. A new proof of Friedman's second eigenvalue theorem and its extension to random lifts. Technical Report 1502.04482v4, arXiv, 2019. To appear in Annales scientifiques de l'École normale supérieure.

[BRS11] Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. pages 472–481, 2011.

[BS87] Andrei Broder and Eli Shamir. On the second eigenvalue of random regular graphs. In *28th Annual Symposium on Foundations of Computer Science (sfcs 1987)*, pages 286–294, 1987.

[BS14] Boaz Barak and David Steurer. Sum-of-squares proofs and the quest toward optimal algorithms. *Electronic Colloquium on Computational Complexity (ECCC)*, 21:59, 2014.

[CMZ23] Zongchen Chen, Elchanan Mossel, and Ilias Zadik. *Almost-Linear Planted Cliques Elude the Metropolis Process*, pages 4504–4539. 2023.

[CO05a] Amin Coja-Oghlan. The Lovász number of random graphs. *Combinatorics, Probability and Computing*, 14(4):439–465, 2005.

[Co05b] Amin Coja-oghlan. The lovász number of random graphs. *Comb. Probab. Comput.*, 14(4):439465, July 2005.

[COE15] Amin Coja-Oghlan and Charilaos Efthymiou. On independent sets in random graphs. *Random Structures & Algorithms*, 47(3):436–486, 2015.

[CP20] Wenjun Cai and Aaron Potechin. The spectrum of the singular values of z-shaped graph matrices, 2020.

[CP22] Wenjun Cai and Aaron Potechin. On mixing distributions via random orthogonal matrices and the spectrum of the singular values of multi-z shaped graph matrices, 2022.

[DdNS21] Jingqiu Ding, Tommaso d'Orsi, Rajai Nasser, and David Steurer. Robust recovery for stochastic block models, 2021.

[DH21] Rishabh Dudeja and Daniel Hsu. Statistical query lower bounds for tensor pca. *Journal of Machine Learning Research*, 22(83):1–51, 2021.

[DKMZ11a] Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Phys. Rev. E*, 84:066106, Dec 2011.

[DKMZ11b] Aurélien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106, December 2011.

[DKMZ11c] Aurelien Decelle, Florent Krzakala, Cristopher Moore, and Lenka Zdeborová. Inference and phase transitions in the detection of modules in sparse networks. *Phys. Rev. Lett.*, 107:065701, Aug 2011.

[DKPP24] Ilias Diakonikolas, Sushrut Karmalkar, Shuo Pang, and Aaron Potechin. Sum-of-squares lower bounds for non-gaussian component analysis, 2024.

[DM11] Varsha Dani and Cristopher Moore. Independent sets in random graphs from the weighted second moment method. In *Approximation, randomization, and combinatorial optimization*, volume 6845 of *Lecture Notes in Comput. Sci.*, pages 472–482. Springer, Heidelberg, 2011.

[DMO+19] Yash Deshpande, Andrea Montanari, Ryan O'Donnell, Tselil Schramm, and Subhabrata Sen. The threshold for sdp-refutation of random regular nae-3sat. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2305–2321. SIAM, 2019.

[DSS16] Jian Ding, Allan Sly, and Nike Sun. Maximum independent sets on random regular graphs. *Acta Math.*, 217(2):263–340, 2016.

[dT23] Tommaso d'Orsi and Luca Trevisan. A Ihara-Bass Formula for Non-Boolean Matrices and Strong Refutations of Random CSPs. In Amnon Ta-Shma, editor, *38th Computational Complexity Conference (CCC 2023)*, volume 264 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 27:1–27:16, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[FGR+17] Vitaly Feldman, Elena Grigorescu, Lev Reyzin, Santosh S. Vempala, and Ying Xiao. Statistical algorithms and a lower bound for detecting planted cliques. *J. ACM*, 64(2), April 2017.

[FK03] Uriel Feige and Robert Krauthgamer. The probable value of the Lovász–Schrijver relaxations for maximum independent set. *SIAM J. Comput.*, 32(2):345–370, 2003.

[FK15] Alan Frieze and Micha Karoski. *Introduction to Random Graphs*. Cambridge University Press, 2015.

[FKP19] N. Fleming, P. Kothari, and T. Pitassi. *Semialgebraic Proofs and Efficient Algorithm Design*. 2019.

[FM17] Zhou Fan and Andrea Montanari. How well do local algorithms solve semidefinite programs? In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 604–614. ACM, 2017.

[FO05] Uriel Feige and Eran. O. Ofek. Spectral techniques applied to sparse random graphs. *Random Structures & Algorithms*, 27, 2005.

[Fri03] Joel Friedman. A proof of alon's second eigenvalue conjecture. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 720–724. ACM, 2003.

[GJJ+20a] Mrinalkanti Ghosh, Fernando Granha Jeronimo, Chris Jones, Aaron Potechin, and Goutham Rajendran. Sum-of-squares lower bounds for sherrington–kirkpatrick via planted affine planes. In *61st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 954–965. IEEE Computer Society, 2020.

[GJJ+20b] Mrinalkanti Ghosh, Fernando Granha Jeronimo, Chris Jones, Aaron Potechin, and Goutham Rajendran. Sum-of-squares lower bounds for Sherrington-Kirkpatrick via planted affine planes. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science—FOCS 2020*, pages 954–965. IEEE Computer Soc., Los Alamitos, CA, [2020] ©2020.

[GJW22] David Gamarnik, Aukosh Jagannath, and Alexander S. Wein. Hardness of random optimization problems for boolean circuits, low-degree polynomials, and langevin dynamics, 2022.

[GKM22] Venkatesan Guruswami, Pravesh K. Kothari, and Peter Manohar. Algorithms and certificates for boolean csp refutation: Smoothed is no harder than random. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 678–689, Rome, Italy, June 2022. ACM.

[Gri01] Dima Grigoriev. Linear lower bound on degrees of Positivstellensatz calculus proofs for the parity. *Theoretical Computer Science*, 259(1-2):613–622, 2001.

[GS11] Venkatesan Guruswami and Ali Kemal Sinop. Lasserre hierarchy, higher eigenvalues, and approximation schemes for graph partitioning and quadratic integer programming with psd objectives. In *FOCS*, pages 482–491, 2011.

[GW94] M. X. Goemans and D. P. Williamson. .878-approximation algorithms for MAX CUT and MAX 2SAT. pages 422–431, 1994.

[GW95] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *J. ACM*, 42:1115–1145, 1995.

[GZ19] David Gamarnik and Ilias Zadik. The landscape of the planted clique problem: Dense subgraphs and the overlap gap property. *CoRR*, abs/1904.07174, 2019.

[HH70] A. J. Hoffman and Leonard Howes. On eigenvalues and colorings of graphs, ii. *Annals of the New York Academy of Sciences*, 175, 1970.

[HKM23] Jun-Ting Hsieh, Pravesh K. Kothari, and Sidhanth Mohanty. A simple and sharper proof of the hypergraph moore bound. In *Proceedings of the 34th ACMSIAM Symposium on Discrete Algorithms (SODA)*, pages 2324–2344, Florence, Italy, January 2023. SIAM.

[HKP+17] Samuel B Hopkins, Pravesh K Kothari, Aaron Potechin, Prasad Raghavendra, Tselil Schramm, and David Steurer. The power of sum-of-squares for detecting hidden structures. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 720–731. IEEE, 2017.

[HKPX23] Jun-Ting Hsieh, Pravesh K. Kothari, Aaron Potechin, and Jeff Xu. Ellipsoid Fitting up to a Constant. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming (ICALP 2023)*, volume 261 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 78:1–78:20, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[HL18] Samuel B. Hopkins and Jerry Li. Mixture Models, Robustness, and Sum of Squares Proofs. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 1021–1034, New York, NY, USA, 2018. ACM.

[Hop19] Samuel B. Hopkins. Mean estimation with sub-gaussian rates in polynomial time, 2019.

[HS17] Samuel B. Hopkins and David Steurer. Efficient Bayesian Estimation from Few Samples: Community Detection and Related Problems. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 379–390, 2017.

[HS21] Shuichi Hirahara and Nobutaka Shimizu. Nearly optimal average-case complexity of counting bicliques under seth. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2346–2365. SIAM, 2021.

[HSS15] Samuel B Hopkins, Jonathan Shi, and David Steurer. Tensor principal component analysis via sum-of-squares proofs. In *Conference on Learning Theory*, pages 956–1006, 2015.

[HSSS16] Samuel B. Hopkins, Tselil Schramm, Jonathan Shi, and David Steurer. Fast spectral algorithms from sum-of-squares proofs: Tensor decomposition and planted sparse vectors. In *Proceedings of the 48th Annual ACM Symposium*

on Theory of Computing (STOC), pages 178–191. ACM, 2016. Also available as arXiv:1512.02337.

[HY21] Jiaoyang Huang and Horng-Tzer Yau. Spectrum of random d-regular graphs up to the edge, 2021.

[Jer92] Mark Jerrum. Large cliques elude the metropolis process. *Random Struct. Algorithms*, 3(4):347–360, 1992.

[JLM20] Aukosh Jagannath, Patrick Lopatto, and Léo Miolane. Statistical thresholds for tensor pca. *Annals of Applied Probability*, 30(4):1910–1933, 2020.

[JP24] Chris Jones and Lucas Pesenti. Diagram analysis of iterative algorithms, 2024.

[JPR+22] C. Jones, A. Potechin, G. Rajendran, M. Tulsiani, and J. Xu. Sum-of-squares lower bounds for sparse independent set. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 406–416, Los Alamitos, CA, USA, feb 2022. IEEE Computer Society.

[JPRX23] Chris Jones, Aaron Potechin, Goutham Rajendran, and Jeff Xu. Sum-of-squares lower bounds for densest k-subgraph. In *Proceedings of the 55th Annual ACM Symposium on Theory of Computing*, STOC 2023, page 8495, New York, NY, USA, 2023. Association for Computing Machinery.

[Juh82] Ferenc Juhász. The asymptotic behaviour of lovász' theta function for random graphs. *Combinatorica*, 2(2):153–155, 1982.

[KD22] Daniel M Kane and Ilias Diakonikolas. A Nearly Tight Bound for Fitting an Ellipsoid to Gaussian Random Points. *arXiv preprint arXiv:2212.11221*, 2022.

[KKM20] Adam Klivans, Pravesh K. Kothari, and Raghu Meka. Efficient algorithms for outlier-robust regression, 2020.

[KM21] Pravesh K. Kothari and Peter Manohar. A stress-free sum-of-squares lower bound for coloring, 2021.

[KM23] Pravesh K. Kothari and Peter Manohar. An exponential lower bound for linear 3-query locally correctable codes. *CoRR*, abs/2311.00558, 2023.

[KM24] Pravesh K. Kothari and Peter Manohar. Superpolynomial lower bounds for smooth 3-lccs and sharp bounds for designs. *CoRR*, abs/2404.06513, 2024.

[KMM+13] Florent Krzakala, Cristopher Moore, Elchanan Mossel, Joe Neeman, Allan Sly, Lenka Zdeborová , and Pan Zhang. Spectral redemption in clustering sparse networks. *Proceedings of the National Academy of Sciences*, 110(52):20935–20940, nov 2013.

[KMOW17] Pravesh Kothari, Ryuhei Mori, Ryan O'Donnell, and David Witmer. Sum of squares lower bounds for refuting any CSP. 2017.

[KOS19] Pravesh K. Kothari, Ryan O'Donnell, and Tselil Schramm. SOS Lower Bounds with Hard Constraints: Think Global, Act Local. In Avrim Blum, editor, *10th Innovations in Theoretical Computer Science Conference (ITCS 2019)*, volume 124 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 49:1–49:21, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[KPX24] Pravesh K. Kothari, Aaron Potechin, and Jeff Xu. Sum-of-squares lower bounds for independent set on ultra-sparse random graphs. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 19231934, New York, NY, USA, 2024. Association for Computing Machinery.

[KSS18] Pravesh K. Kothari, Jacob Steinhardt, and David Steurer. Robust moment estimation and improved clustering via sum of squares. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 10351046, New York, NY, USA, 2018. Association for Computing Machinery.

[Kuc95] Ludek Kucera. Expected complexity of graph partitioning problems. *Discrete Appl. Math.*, 57(2-3):193–212, 1995.

[KVWX23] Pravesh Kothari, Santosh S. Vempala, Alexander S. Wein, and Jeff Xu. Is planted coloring easier than planted clique? In *Annual Conference Computational Learning Theory*, 2023.

[KWB19a] Dmitriy Kunisky, Alexander S. Wein, and Afonso S. Bandeira. Notes on computational hardness of hypothesis testing: Predictions using the low-degree likelihood ratio, 2019.

[KWB19b] Dmitriy Kunisky, Alexander S. Wein, and Afonso S. Bandeira. Notes on computational hardness of hypothesis testing: Predictions using the lowdegree likelihood ratio. CoRR abs/1907.11636, arXiv preprint, 2019. Presented at ISAAC Congress 2019, later published in *Mathematical Analysis, its Applications and Computation* (ISAAC 2019), Springer PROMS 385 (2022).

[KX25] Pravesh K. Kothari and Jeff Xu. Smooth trade-off for tensor pca via sharp bounds for kikuchi matrices. in submission, 2025.

[KY24] Dmitriy Kunisky and Xifan Yu. Computational hardness of detecting graph lifts and certifying lift-monotone properties of random regular graphs, 2024.

[KZ09] Florent Krzakala and Lenka Zdeborová . Hiding quiet solutions in random constraint satisfaction problems. *Physical Review Letters*, 102(23), jun 2009.

[Las00] Jean B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM J. on Optimization*, 11(3):796–817, March 2000.

[Las01] Jean B. Lasserre. An explicit exact SDP relaxation for nonlinear 0-1 programs. pages 293–303, London, UK, 2001. Springer-Verlag.

[LML+17] Thibault Lesieur, Léo Miolane, Marc Lelarge, Florent Krzakala, and Lenka Zdeborová. Statistical and computational phase transitions in spiked tensor estimation. In *2017 IEEE International Symposium on Information Theory (ISIT)*, pages 511–515. IEEE, 2017.

[LMR22] S. Liu, S. Mohanty, and P. Raghavendra. On statistical inference when fixed points of belief propagation are unstable. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 395–405, Los Alamitos, CA, USA, feb 2022. IEEE Computer Society.

[Mas14] Laurent Massoulié. Community detection thresholds and the weak ramanujan property. In *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '14, page 694703, New York, NY, USA, 2014. Association for Computing Machinery.

[MNS15] Elchanan Mossel, Joe Neeman, and Allan Sly. Reconstruction and estimation in the planted partition model. *Probability Theory and Related Fields*, 162(3-4):431–461, 2015. Supported by NSF grants DMS-1106999, CCF-1320105 and DOD ONR grant N000141110140. Supported by Sloan Research Fellowship in mathematics and by NSF award DMS-1208339.

[MNS18] Elchanan Mossel, Joe Neeman, and Allan Sly. A proof of the block model threshold conjecture. *Combinatorica*, 38(3):665708, jun 2018.

[MR14] Andrea Montanari and Emile Richard. A statistical model for tensor pca. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 28972905, Cambridge, MA, USA, 2014. MIT Press.

[MRX20] Sidhanth Mohanty, Prasad Raghavendra, and Jeff Xu. Lifting sum-of-squares lower bounds: degree-2 to degree-4. pages 840–853, 2020.

[MSS16] Tengyu Ma, Jonathan Shi, and David Steurer. Polynomial-time tensor decompositions with sum-of-squares. pages 438–446. IEEE, 2016.

[MW19] Ankur Moitra and Alexander S. Wein. Spectral methods from tensor networks. *SIAM Journal on Computing*, 52(2):STOC19–354–STOC19–384, 2019.

[Nes00] Yurii Nesterov. Squared functional systems and optimization problems. In *High performance optimization*, pages 405–440. Springer, 2000.

[O'D17] Ryan O'Donnell. SOS Is Not Obviously Automatizable, Even Approximately. In Christos H. Papadimitriou, editor, *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*, volume 67 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 59:1–59:10, Dagstuhl, Germany, 2017. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[Pan21] Shuo Pang. SOS lower bound for exact planted clique. In *36th Computational Complexity Conference*, volume 200 of *LIPIcs. Leibniz Int. Proc. Inform.*, pages Art. 26, 63. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2021.

[Par00] Pablo A Parrilo. *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. PhD thesis, California Institute of Technology, 2000.

[PR20] Aaron Potechin and Goutham Rajendran. Machinery for proving sum-of-squares lower bounds on certification problems. abs/2011.04253, 2020.

[PR22] Aaron Potechin and Goutham Rajendran. Sub-exponential time sum-of-squares lower bounds for principal components analysis. *Advances in Neural Information Processing Systems*, 35, 2022. Appeared in NeurIPS 2022.

[PTVW22] Aaron Potechin, Paxton Turner, Prayaag Venkat, and Alexander S Wein. Near-optimal fitting of ellipsoids to random points. *arXiv preprint arXiv:2208.09493*, 2022.

[PWB16] Amelia Perry, Alexander S. Wein, and Afonso S. Bandeira. Statistical limits of spiked tensor models. *arXiv preprint arXiv:1612.07728*, 2016.

[PX25] Aaron Potechin and Jeff Xu. Sum-of-squares lower bounds for coloring random graphs. In *Proceedings of the 57th Annual ACM Symposium on Theory of Computing*, STOC '25, page 8495, New York, NY, USA, 2025. Association for Computing Machinery.

[RRS17] Prasad Raghavendra, Satish Rao, and Tselil Schramm. Strongly Refuting Random CSPs Below the Spectral Threshold. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 121–131, New York, NY, USA, 2017. ACM.

[RSS19] Prasad Raghavendra, Tselil Schramm, and David Steurer. High-dimensional estimation via sum-of-squares proofs, 2019.

[RSWY22] Cynthia Rush, Fiona Skerman, Alexander S. Wein, and Dana Yang. Is it easier to count communities than find them?, 2022.

[RT22] Goutham Rajendran and Madhur Tulsiani. Concentration of polynomial random matrices via efron-stein inequalities, 2022.

[RT23] Goutham Rajendran and Madhur Tulsiani. *Concentration of polynomial random matrices via Efron-Stein inequalities*, pages 3614–3653. 2023.

[Sar23] Amir Sarid. The spectral gap of random regular graphs. *Random Structures & Algorithms*, 63(2):557–587, 2023.

[Sau11] James Saunderson. *Subspace identification via convex optimization*. PhD thesis, Massachusetts Institute of Technology, 2011.

[SCPW12] James Saunderson, Venkat Chandrasekaran, Pablo A Parrilo, and Alan S Willsky. Diagonal and low-rank matrix decompositions, correlation matrices, and ellipsoid fitting. *SIAM Journal on Matrix Analysis and Applications*, 33(4):1395–1416, 2012.

[Sho87] Naum Zuselevich Shor. An approach to obtaining global extremums in polynomial mathematical programming problems. *Cybernetics*, 23(5):695–700, 1987.

[SOKB25] Alexander Schmidhuber, Ryan O'Donnell, Robin Kothari, and Ryan Babbush. Quartic quantum speedups for planted inference. *Phys. Rev. X*, 15:021077, Jun 2025.

[SPW13] James Saunderson, Pablo A Parrilo, and Alan S Willsky. Diagonal and low-rank decompositions and fitting ellipsoids to random points. In *52nd IEEE Conference on Decision and Control*, pages 6031–6036. IEEE, 2013.

[SW22] Tselil Schramm and Alexander S. Wein. Computational barriers to estimation from low-degree polynomials. *The Annals of Statistics*, 50(3), jun 2022.

[Tao12] Terence Tao. *Topics in random matrix theory*, volume 132. American Mathematical Soc., 2012.

[TVW13] Linh V. Tran, Van H. Vu, and Ke Wang. Sparse random graphs: Eigenvalues and eigenvectors. *Random Structures & Algorithms*, 42(1):110–134, 2013.

[TY16] Konstantin Tikhomirov and Pierre Youssef. The spectral gap of dense random regular graphs. *arXiv e-prints*, page arXiv:1610.01765, October 2016.

[WAM19] Alexander S. Wein, Ahmed El Alaoui, and Cristopher Moore. The Kikuchi Hierarchy and Tensor PCA. *CoRR*, abs/1904.03858, 2019.

[Wei20] Alexander S Wein. Optimal low-degree hardness of maximum independent set. *arXiv preprint arXiv:2010.06563*, 2020.

[Wei22] Alexander S. Wein. Optimal lowdegree hardness of maximum independent set. *Mathematics of Statistics and Learning*, 4(3/4):221–251, 2022.

[Xu25] Jeff Xu. Switching Graph Matrix Norm Bounds: From i.i.d. to Random Regular Graphs. In Srikanth Srinivasan, editor, *40th Computational Complexity Conference (CCC 2025)*, volume 339 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11:1–11:23, Dagstuhl, Germany, 2025. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[ZK16] Lenka Zdeborová and Florent Krzakala. Statistical physics of inference: thresholds and algorithms. *Advances in Physics*, 65(5):453–552, aug 2016.