New Results in the Theory of Approximation

Fast Graph Algorithms and Inapproximability

Sushant Sachdeva

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Doctor of Philosophy

Recommended for Acceptance
by the Department of
Computer Science
Advisor: Sanjeev Arora

September 2013

Abstract

For several basic optimization problems, it is **NP**-hard to find an exact solution. As a result, understanding the best possible trade-off between the running time of an algorithm and its approximation guarantee, is a fundamental question in theoretical computer science, and the central goal of the theory of approximation.

There are two aspects to the theory of approximation : (1) efficient approximation algorithms that establish trade-offs between approximation guarantee and running time, and (2) inapproximability results that give evidence against them. In this thesis, we contribute to both facets of the theory of approximation.

In the first part of this thesis, we present the first near-linear-time algorithm for Balanced Separator - given a graph, partition its vertices into two roughly equal parts without cutting too many edges - that achieves the best approximation guarantee possible for algorithms in its class. This is a classic graph partitioning problem and has deep connections to several areas of both theory and practice, such as metric embeddings, Markov chains, clustering, etc.

As an important subroutine for our algorithm for Balanced Separator, we provide a near-linear-time algorithm to simulate the heat-kernel random walk on a graph, equivalent to computing $e^{-L}v$, where L is the Laplacian of the graph, and v is a vector. This algorithm combines techniques from approximation theory and numerical linear algebra to reduce the problem of approximating the matrix exponential to solving a small number of Laplacian systems. We also give a reduction in the reverse direction, from matrix inversion to matrix exponentiation, hence justifying the use of Laplacian system solvers.

In the second part of this thesis, we prove inapproximability results for several basic optimization problems. We address some classic scheduling problems, *viz.* Concurrent Open Shop and the Assembly Line problem, and variants of the Hypergraph Vertex Cover problem. For all these problems, optimal inapproximability results were previously known under the Unique Games Conjecture. We are able to prove near-optimal inapproximability results for these problems without using the conjecture.

Acknowledgements

I owe a lot of gratitude to my advisor Sanjeev Arora. This thesis would have been impossible without his unwavering support and guidance. His grand vision of theoretical computer science, and an open-minded approach to research driven by scientific curiosity are truly inspiring. I aspire to his simple yet crisp intuitive understanding of complex concepts. He has helped me in innumerable ways over these years to develop into an independent researcher.

My academic journey has been influenced positively by a number of wonderful mentors. I would specially like to thank Nisheeth Vishnoi for his guidance. I have greatly benefitted from our discussions, both technical, and about research and writing in general. I would also like to thank him and Microsoft Research India for hosting me during Summer 2011. Large parts of this thesis are based on my work with Nisheeth.

I am also extremely grateful to Boaz Barak, Moses Charikar, and Madhur Tulsiani. I have turned to each of them for support during several stages of my graduate life, and have always received insightful advice for my numerous conundrums. I can only aspire to achieve the degree of clarity their thoughts embody. I would like to thank my undergraduate advisor, Sundar Vishwanathan, for having fostered in me an appreciation of theory and its beauty. I wish to also thank my uncle, Suresh Sachdeva, for having instilled in me the aspiration to be a researcher at an early age.

I wish to thank Toyota Technological Institute at Chicago for hosting me as an intern during Summer 2012, and to Yury Makarychev for serving as my mentor during the stay.

Thanks to Boaz Barak and Bernard Chazelle – I learned a lot about teaching while serving as a teaching assistant for their courses. Many thanks to my entire thesis committee: Sanjeev Arora, Mark Braverman, Moses Charikar, Bernard Chazelle, and Nisheeth Vishnoi, for their time and their valuable suggestions.

I have been very fortunate to have collaborated with a lot of terrific researchers over the years and would like to thank them for everything I have learnt from them: Sanjeev Arora, Amey Bhangale, Arnab Bhattacharyya, Julia Chuzhoy, Mark Braverman, Rong Ge, Venkatesan Guruswami, Pooya Hatami, Swastik Kopparty, James Lee, Yury Makarychev, Rajsekar Manokaran, Mohammad Moharrami, Ankur Moitra, Ofer Neiman, Lorenzo Orecchia, Rishi Saket, Grant Schoenebeck, Nikhil Srivastava, Madhur Tulsiani, Nisheeth Vishnoi, and Pratik Worah. I would also like to acknowledge the support of NSF grants CCF-0832797, CCF-1117309, and Sanjeev Arora's Simons Investigator Grant.

I would like to thank all the members of the theory groups at Princeton University and at the Institute of Advanced Study for creating such an exciting environment for research. Thanks also to the members of the Center for Computational Intractability for the engaging meetings and workshops that were great opportunities to learn.

Thanks to the wonderfully helpful administrative staff at the Computer Science department at Princeton University. Special thanks to Mitra Kelly and Melissa Lawson for putting up with a myriad of administrative questions and requests from me.

I have shared some wonderful times with the students in theory group at the department and would like to thank all of them for making the five years so enjoyable, especially with all the discussions over coffee. I owe my gratitude to Aditya Bhaskara, Moritz Hardt, Mohammad Mahmoody, Rajsekar Manokaran, Siddhartha Sen, David Steurer, and Aravindan Vijayaraghavan, for all the advice during various stages of graduate life. Thanks also to Rong Ge, Daniel Larkin, Shi Li, Yonatan Naamad, Huy Nguyễn, and V. Anuradha.

Life in Princeton outside the theory group too had hardly any dull moments due to many wonderful friends. Special thanks to my apartment mates: Sreechakra Goparaju, for being a great companion, and for all the discussions we had on every topic under the Sun, and Arnab Sinha, for his contagious enthusiasm for food and travel. Thanks also to several friends who made Princeton feel like home: Harish B., Pathikrit Bhattacharya, Parthav Desai, Aahana Ganguly, Jaya Khanna, Radha Kumar, Shruti Patil, Katharina Schwartz, Vinay Sitapati, V. Anuradha, and Arjun Vijayakumar.

I have been fortunate to have some great friends nearby and would like to thank them for several memorable weekends: Neeraja Dashaputre, Rohan Joshi, Jyotsna Joshi, Vignesh Kasinath, Kartik Mohta, Sanika Mokashi, and Sangram Raje. I would also like to thank several other wonderful friends, who have stayed in touch despite the distances: Sukhada Fadnavis, Shantanu Gangal, Priya Gupta, Sonal Jain, Sudeep Kamath, Tejas Kulkarni, Aquila Mavalankar, Neha Mhaisalkar, Aishwarya Palorath, Soumya Venugopal, Arunanshu Roy, and Anisha Zaveri.

I would like to thank my uncle Shiv Raj Kathpalia, my aunt Lalita Kathpalia, and my cousins Tanuj Kathpalia and Yatin Kathpalia, for being family for me in this land far away from home.

Lastly, but most importantly, I would like to thank my family - my parents, Indra Sachdeva and Surinder Pal Sachdeva, and my brother Prashant Sachdeva. I owe more to their constant love, support, and understanding over the years, than words can express. It is to them that I dedicate my thesis.

To my family.

Contents

List of Figures

Chapter 1

Introduction

One of the central goals of theoretical computer science is to study the amount of time required for basic computational questions. Several questions of interest can be framed as combinatorial optimization problems, where the goal is to find one solution from a discrete set of feasible solutions that optimizes a specified objective.

Classically, the focus has been to study whether the optimum solution can be found in time polynomial in the size of the input. Though proving unconditional, super-polynomial lower bounds for optimization problems has remained out of reach, for most natural problems of interest where we do not have polynomial time algorithms, the theory of **NP**-completeness tells us that they are almost as hard to solve as 3SAT (which is widely believed to require super-polynomial, maybe even exponential amount of time [IPZ01]).

However, proving **NP**-completeness for an optimization problem is usually not satisfactory since it only rules out polynomial time algorithms for computing the optimum solution. How about computing solutions that are close to being optimal? One widely used approach to measuring the quality of a near-optimal solution is by its approximation ratio, where we say that a solution is an c-approximation if its objective value is within a factor c of the optimum (approximation ratio will be larger than 1 throughout). Thus, a natural question is, what is the best approximation ratio that can be guaranteed by efficient algorithms. This is the focus of the theory of approximation that has been developed ever since the theory of **NP**-completeness was developed in the 1970's.

One of the most important ways of coping with **NP**-hardness of most problems of interest is to look for approximately optimal solutions. The ultimate goal of a complete theory of approximation is to answer the following question for every optimization problem of interest:

> *Given an input of size n, what is the best approximation factor that can be guaranteed by an algorithm running in time $T(n)$?*

Here $T(n)$ is any function of n, e.g., n^{10}. In other words, what is the best possible trade-off that can be achieved between approximation guarantee and running time.

Over the last few decades, much progress has been made towards this challenging goal. On one hand, a rich theory of approximation algorithms has arisen that gives efficient algorithms with provable approximation guarantees. On the other hand, the field of hardness of approximation, beginning in its earnest with the breakthrough PCP theorem [ALM+98, AS98], has sought to prove limits on approximability achievable by efficient algorithms. However, we are still quite far from a complete understanding of the approximability of several fundamental problems such as Balanced Separator and Minimum Vertex Cover.

In this thesis, we contribute to both these facets of theory of approximation, as detailed below.

1.1 Near-Linear-time Graph algorithms

Since graphs offer a convenient abstraction for all kinds of networks, it's no surprise that optimization problems on graphs are ubiquitous, and particularly in fields such as computer networks, operations research, and social sciences. Today, the sizes of the graphs that we would like to study are huge, e.g. the web graph has $\sim$100 billion nodes, and the Facebook graph has $\sim$1 billion nodes. With such large graphs, any algorithm that runs in time significantly more than linear in the size of its input is essentially impractical. This has motivated the search for near-linear-time algorithms for fundamental graph problems.

With the idea of designing near-linear-time graph algorithms that achieve the best possible algorithmic guarantees, several fundamental graph problems such as $s-t$ Max-Flow and balanced graph partitioning have been revisited over the past few years. The classic combinatorial techniques seem to no longer be sufficient, and the new algorithms often combine a host of techniques from diverse areas including semi-definite programming, exponential weights method, and linear algebra. In this thesis, we add new techniques to this toolbox from approximation theory [1] and numerical linear algebra.

In the first part of this thesis, we give a near-linear-time approximation algorithm for Balanced Separator, a fundamental graph partitioning problem where we seek to partition the vertices of the input graph into two roughly equal sets while minimizing the number of edges cut. This problem has been intensely studied in both theory and practice, and has deep connections to several areas of mathematics. It is also known to be **NP**-hard to solve exactly, and several approximation algorithms with provable guarantees are known. Our algorithm achieves an approximation guarantee that is optimal for any algorithm of its kind in near-linear time, and combines techniques from approximation theory, numerical linear algebra, and rounding semi-definite programs.

We discuss the problem in more detail, and our contributions, in Chapter 2.

[1] Approximation theory is the study of approximating a given function $f(x)$ by polynomial or rational functions, not to be confused with the theory of approximation.

1.2 Hardness of Approximation

The goal of hardness of approximation is to complement approximation algorithms with evidence that certain approximation guarantees cannot be achieved by efficient algorithms, for instance, by proving that such a guarantee is **NP**-hard to achieve. The celebrated PCP theorem [ALM$^+$98, AS98] proved that, for Max-3SAT, it is **NP**-hard to achieve an approximation factor that is arbitrarily close to 1. In fact, it showed that achieving such arbitrarily good approximation for a wide class of problems called MAX-SNP-complete problems, that include Max-3SAT, Minimum Vertex Cover, and Minimum Set Cover, is **NP**-hard.

Though the PCP theorem by itself does not imply hardness results matching the known approximation results, as a result of several influential works [BGS98, Raz98, Hås01], the approximability of a few problems, such as Max-3SAT, is quite well understood by now, and we have optimal hardness results for some of them. However, the picture is far from complete for several important problems such as Minimum Vertex Cover and Balanced Separator.

Despite a significant amount of effort devoted towards proving improved inapproximability results, there remained a significant gap between upper and lower bounds on the approximability of problems such as Max-Cut and Minimum Vertex Cover. Towards proving optimal inapproximability results for these problems, Khot [Kho02] introduced the Unique Games Conjecture. The UGC conjectures that a specific constraint satisfaction problem (CSP) is **NP**-hard. We now know that the UGC implies new optimal inapproximability results for several problems, including Max-Cut, Minimum Vertex Cover, and all CSP's. However, we lack strong evidence in favor of the conjecture, and our belief in the conjecture isn't as strong as other classic complexity assumptions, such as $\mathbf{P} \neq \mathbf{NP}$.

Since the UGC is so delicately poised, and we seem to be quite far from a proof of the conjecture, there is considerable interest in obtaining these new inapproximability results without assuming the conjecture. In the second part of this thesis, we show near-optimal **NP**-hardness results for approximating some classic scheduling problems, and variants of Minimum Vertex Cover. All these results were previously known only under the UGC, and our main contribution is to modify the reductions to obtain these results just assuming $\mathbf{P} \neq \mathbf{NP}$.

The problems of interest, and our contributions are discussed in detail in Chapter 7.

1.3 Organization of this thesis

This thesis is divided into two parts. We outline the structure of the two parts below.

Part I

In the first part, we present a near-linear-time approximation algorithm for graph partitioning problems, building on rational approximations for the exponential, and algorithms to approximate the matrix exponential. This part is based on joint work with Lorenzo Orecchia and Nisheeth Vishnoi. We first describe the problems of interest, and place our results in context in Chapter 2.

In Chapter 3, we describe a novel spectral algorithm for Balanced Separator, a fundamental graph partitioning problem, that runs in near-linear time and achieves an approximation guarantee that is optimal for spectral algorithms. This algorithm requires, as a subroutine, an algorithm to approximate a matrix-exponential-vector product.

In Chapter 4, we give an efficient reduction from approximating the matrix-exponential-vector product to approximating the matrix-inverse-vector product. Combining with the near-linear-time Laplacian solver of Spielman and Teng, this will provide us with the subroutine required for our algorithm for Balanced Separator, along with its running time guarantee. We also give a simpler algorithm that avoids the heavy machinery of the Spielman-Teng solver, but still results in a Balanced Separator algorithm that compares favorably to all previous spectral algorithms.

In Chapter 6, we discuss uniform polynomial approximations to the exponential, and prove that there exist polynomial approximations to the exponential that offer a quadratic improvement over the naive Taylor series approximation. We also prove that this result is essentially optimal. The existence of these improved polynomial approximations is required for the guarantees on the simpler algorithm to approximate the matrix-exponential-vector product from the previous chapter.

In Chapter 5, we show that there is also a reduction from approximating the matrix-inverse-vector product to the matrix-exponential-vector product. This justifies the use of the Spielman-Teng result for obtaining a near-linear-time algorithm in Chapter 4.

Part II.

In the second part, we prove close to optimal inapproximability results for variants of hypergraph vertex cover, and some classic scheduling problems. This part is based on joint work with Rishi Saket. In Chapter 7, we describe the problems, and put our results in context.

In Chapter 8, we prove that some classic scheduling problems that have been long known to have simple 2-approximations, are in fact **NP**-hard to approximate better than 2. These results were previously known under a stronger assumption known as the Unique Games Conjecture. Our results are based on a reduction from a structured hardness result for Minimum Vertex Cover on hypergraphs.

In Chapter 9, we prove that the Minimum Vertex Cover problem is **NP**-hard to approximate on k-uniform k-partite hypergraphs better than $\frac{k}{2} - 1$. Since there is a $\frac{k}{2}$ approximation algorithm due to Lovász, this is off by at most 1 from the best possible. Here again, the optimal result is known to hold under the Unique Games Conjecture.

Chapter 2

The Balanced Separator Problem and the Matrix Exponential

2.1 The Balanced Separator problem, and Spectral Algorithms

The Balanced Separator problem is the quintessential graph partitioning question. Informally, the problem can be stated as follows:

> *Can the given graph be divided into two roughly equal parts without cutting too many edges?*

To formally specify the problem, say we have an unweighted graph $G = (V, E)$, where $V = [n]$, and $|E| = m$. Given a set $S \subseteq V$, define its volume, denoted by $\mathsf{vol}(S)$, to be the sum of the degrees of the vertices in the set S. We measure the quality of the cut $(S, \overline{S})$ by its conductance $\phi(S)$, defined as, $\phi(S) \overset{\text{def}}{=} \frac{|E(S,\bar{S})|}{\min\{\mathsf{vol}(S),\mathsf{vol}(\bar{S})\}}$. Moreover, we say that a cut $(S, \overline{S})$ is b-balanced if $\min\{\mathsf{vol}(S), \mathsf{vol}(\overline{S})\} \geq b \cdot \mathsf{vol}(V)$. The Balanced Separator problem asks the following decision question:

Given an unweighted graph G, a constant balance parameter $b \in (0, 1/2]$, and a target conductance value $\gamma \in (0,1]$, does G have a b-balanced cut S such that $\phi(S) \leq \gamma$?

This is a classic **NP**-hard problem and a central object of study for the development of approximation algorithms, both in theory and in practice. On the theoretical side, the Balanced Separator problem has far reaching connections to several areas of mathematics such as spectral graph theory, the study of random walks, and metric embeddings. In practice, algorithms for Balanced Separator play a crucial role in the design of recursive algorithms [Shm97], clustering [KVV00], and scientific computation [SKK+00].

Spectral Algorithms. Spectral methods are an important set of techniques in the design of graph-partitioning algorithms and are fundamentally based on the study of the behavior of random walks over the instance graph. Spectral algorithms tend to be conceptually appealing, because of the intuition based on the underlying diffusion process, and are easy to implement, as many of the primitives required, such as eigenvector computation, already appear in highly-optimized software packages. The most important spectral algorithm for graph partitioning is the Laplacian Eigenvector (LE) algorithm of Alon and Milman [AM85], which, given a graph of conductance at most γ, outputs a cut of conductance at most $O(\sqrt{\gamma})$, an approximation guarantee that is asymptotically optimal for spectral algorithms.

A consequence of the seminal work of Spielman and Teng [ST04, ST06] is that the LE algorithm can be implemented in $\tilde{O}(m)$ time using the Spielman-Teng Laplacian system solver. Hence, LE is an asymptotically optimal spectral algorithm for the minimum-conductance problem, both in terms of running time (up to polylog factors) and approximation quality.

The simplest spectral algorithm for Balanced Separator iteratively uses the LE algorithm to remove small conductance cuts from G, working with the residual graph at each step, until either a balanced cut or an induced γ-expander is found (see, *e.g.*, [KVV00]). We'll denote this algorithm by Recursive Eigenvector Algorithm (RLE). However, observe that the cut found at each iteration may consist of only a constant number of vertices, and hence this algorithm may run for $\Omega(n)$ iterations. Since each iteration requires an eigenvalue computation, RLE may require $\Omega(mn)$ time in the worst case.

Our Results. We present a simple random-walk-based algorithm that is the first such asymptotically optimal spectral algorithm for Balanced Separator. Our algorithm can be seen as analogous to the RLE algorithm for Balanced Separator, and settles the question of designing spectral algorithms for Balanced Separator. It combines techniques from rounding algorithms for semi-definite programs (SDP), numerical linear algebra, and approximation theory. The following is our main theorem for Balanced Separator.

Theorem 2.1 (Spectral Algorithm for Balanced Separator)**.** *Given an unweighted graph $G = (V, E)$, a balance parameter $b \in (0, 1/2]$, $b = \Omega(1)$ and a conductance value $\gamma \in (0, 1]$, we give an algorithm called* BalSep(G, b, γ)*, that either outputs an $\Omega(b)$-balanced cut $S \subset V$ such that $\phi(S) \leq O(\sqrt{\gamma})$, or outputs a certificate that no b-balanced cut of conductance γ exists.* BalSep *runs in time $O(m\ \mathrm{poly}(\log n))$.*

This settles the question of designing asymptotically optimal spectral algorithms for Balanced Separator. The algorithm for Theorem 2.1 is based on a variant of the heat kernel random walk and requires, as a subroutine, an algorithm to compute the product of the matrix-exponential of a matrix and an arbitrary vector in time essentially proportional to the sparsity of the matrix. Our contribution to the problem of computing the matrix-exponential-vector product appear in detail in Section 2.2.

The algorithm for computing the matrix-exponential-vector product required for Theorem 2.1 runs in time $\tilde{O}(m)$ and, notably, makes use of the Spielman-Teng Laplacian system solver in a non-trivial way. Combined with the new simple Laplacian system solver due to Kelner *et al.* [KOSZ13], we obtain an algorithm that is almost as simple and practical as the LE algorithm.

We also prove an alternative result on how to perform this matrix-exponential computation, which relies just on matrix-vector products. This result, when combined with our algorithm for Balanced Separator, yields a theorem identical to Theorem 2.1 except that the running time now increases to $\tilde{O}(m/\sqrt{\gamma})$, see Theorem 3.2.

2.1.1 Comparison to Previous work on Balanced Separator

The best known approximation guarantee for Balanced Separator is $O(\sqrt{\log n})$ achieved by the seminal work of Arora, Rao and Vazirani [ARV09] that combines SDPs and flow ideas. A rich line of research has centered on reducing the running time of this algorithm [KRV06, AK07, OSVV08]. This effort culminated in Sherman's work [She09], which brings down the required running time to $O(n^{\varepsilon})$ s-t maximum-flow computations [1].

However, these algorithms are based on advanced theoretical ideas that are not easy to implement, or even capture in a principled heuristic. Moreover, they fail to achieve a nearly-linear [2] running time, which is crucial in many of today's applications that involve very large graphs. In particular, even with the recent algorithms for approximating s-t max-flows in $O(mn^{o(1)})$ time [She13, KOLS13], the total time required by Sherman's algorithm is $O(mn^{\epsilon+o(1)})$. To address these issues, researchers

[1]Even though the results of [ARV09] and [She09] are stated for the Sparsest Cut problem, the same techniques apply to the conductance problem, e.g. by modifying the underlying flow problems. See for example [AL08].

[2]Following the convention of [ST08], we denote by nearly-linear a running time of $\tilde{O}(m/\mathrm{poly}(\gamma))$.

have focused on the design of simple, nearly-linear-time algorithms for Balanced Separator based on spectral techniques.

Spectral Algorithms for Balanced Separator. We already noted that the simplest spectral algorithm for Balanced Separator, the RLE algorithm, requires quadratic time in the worst case. Spielman and Teng [ST04, ST08] were the first to design nearly-linear-time algorithms that output an $\Omega(b)$-balanced cut of conductance $O(\sqrt{\gamma}\,\mathrm{poly}(\log n))$, if a b-balanced cut of conductance less than γ exists. Their algorithmic approach is based on local random walks, which are used to remove unbalanced cuts in time proportional to the size of the cut removed, hence avoiding the quadratic dependence of RLE. Using similar ideas, Andersen, Chung and Lang [ACL06], and Andersen and Peres [AP09] improved the approximation guarantee to $O(\sqrt{\gamma \log n})$ and the running time to $\tilde{O}(m/\sqrt{\gamma})$.

More recently, Orecchia and Vishnoi [OV11] employed an SDP formulation of the problem, together with the Matrix Multiplicative Weights Update method of Arora and Kale [AK07] and a new SDP rounding, to obtain an output conductance of $O(\sqrt{\gamma})$ with running time $\tilde{O}(m/\gamma^2)$, effectively removing unbalanced cuts in $O(\log n/\gamma)$ iterations. In Section 3.7, we give a more detailed comparison with the work of Orecchia-Vishnoi.

Finally, our algorithm should be compared to the remarkable results of Mądry [Mąd10] for Balanced Separator, which build up on Räcke's work [Räc08], and on the low-stretch spanning trees from Abraham *et al.* [ABN11], to achieve a trade-off between running time and approximation. For every integer $k \geq 1$, he achieves roughly $O((\log n)^k)$ approximation in time $\tilde{O}(m + 2^k \cdot n^{1+2^{-k}})$. Calculations show that for $\gamma \geq 2^{-(\log\log n)^2}$, our algorithm achieves strictly better running time and approximation than Mądry's for sparse graphs. More importantly, we believe that our algorithm is significantly simpler, and likely to find applications in practical settings.

2.2 Approximating the Matrix Exponential

Given a symmetric $n \times n$ matrix A, its matrix exponential, denoted by e^A or $\exp(A)$, is defined to be $e^A \stackrel{\text{def}}{=} \sum_{i \geq 0} \frac{A^i}{i!}$. This operator is of fundamental interest in several areas of mathematics, physics, and engineering, and has recently found important applications in algorithms, optimization, and quantum complexity. Roughly, these latter applications are manifestations of the Matrix Multiplicative Weights Update method of Arora and Kale [AK07], and its deployment to solve semi-definite programs efficiently (see [AK07, OV11, JJUW11]).

At the core of most algorithms based on the Matrix Multiplicative Weights Update method, and in particular, our algorithm for Balanced Separator, lies an algorithm to quickly compute $\exp(-A)v$ for a symmetric, positive semi-definite (PSD) matrix A and a unit vector v. It is sufficient to compute an approximation u, to $\exp(-A)v$, in time which is as close as possible to t_A, where t_A denote the time required to multiply the matrix A with a given vector v [3].

It can be shown that using about $\|A\|$ terms in the Taylor series expansion of $\exp(-A)$, one can find a vector u that approximates $\exp(-A)v$. Hence, this method runs in time roughly $O(t_A \cdot \|A\|)$. In our application, and certain others [AK07, Kal07, IPS11, IPS05], this dependence on the norm is prohibitively large.

Since polynomial approximations to the exponential do not suffice to approximate the matrix exponential fast (see Section 2.4), a number of works focussed on obtaining good rational approximations to the exponential [CMV69, Ehl73, UW73, SSV75,

[3]In general, t_A depends on how A is represented and can be $\Theta(n^2)$. However, it is possible to exploit the special structure of A if given as an input appropriately. e.g., It is possible to just multiply the non-zero entries of A, giving $t_A = O(m_A)$, where m_A denotes the number of non-zero entries in A.

And81]. The work of Saff, Schönhage, and Varga [SSV75] reduces the computation of $\exp(-A)v$ to a number of $(I + cA)^{-1}v$ computations for some $c > 0$. Note that this is insufficient for a fast algorithm since exact computation of $(I + cA)^{-1}v$ is a costly operation. Eshof and Hochbruck [EH05] propose the use of iterative methods for approximating $(I + cA)^{-1}v$ [4]. However, to the best of our knowledge, there is no known proof of using rational approximations to approximate the matrix exponential under approximate inverse computation.

Reduction to Matrix Inversion and SDD Matrices. We give a rigorous proof of a reduction from (approximate) matrix exponentiation to (approximate) matrix inversion. Though based on the rational approximation result from Saff, Schönhage, and Varga [SSV75], our proposed procedure differs from earlier approaches in significant ways, and we outline the differences in detail in Section 4.3. Appealing to techniques from numerical linear algebra and approximation theory, we prove the following theorem in chapter 4:

Theorem 2.2 (Informal). *Given a PSD matrix A, and a vector v, the approximation of $\exp(-A)v$ can be reduced to poly-logarithmic number of approximations of the form $(I + cA)^{-1}u$, where $c > 0$, and u is another vector.*

For fast graph algorithms, the quantity of interest is $e^{-L}v$, where L is the combinatorial Laplacian of a graph, and v is a vector. The vector $e^{-L}v$ can also be interpreted as the resulting distribution of a certain continuous-time random walk on the graph with starting distribution v. Thus, concretely, using a the near-linear-time Laplacian system solver[5] [ST04, KMP11, KOSZ13], this gives an $\tilde{O}(m)$-time algorithm for approximating $e^{-L}v$ for graphs with m edges.

[4]Their method was cited in a number of works [AK07, Kal07, IPS11, IPS05].

[5]A Laplacian solver is an algorithm that (approximately) solves a given system of linear equations $Lx = b$, where L is a graph Laplacian and b lies in the image of L, *i.e.*, it (approximately) computes $L^{-1}b$, see [Vis12].

Theorem 2.3 (SDD Matrix Exponential Computation)**.** *Given an $n \times n$ SDD* [6] *matrix A, a vector v and a parameter $\delta \leq 1$, there is an algorithm that can compute a vector u such that $\|\exp(-A)v - u\| \leq \delta \|v\|$ in time $\tilde{O}((m_A + n)\log(2 + \|A\|))$. Here the tilde hides poly$(\log n)$ and poly$(\log {}^1\!/\!_\delta)$ factors.*

For our application to Balanced Separator, firstly, the dependence of the running time on the $\log(2 + \|A\|)$ turns out to just contribute an extra $\log n$ factor. Also, we will have $\delta = {}^1\!/\!_{\text{poly}(n)}$. Secondly, the matrix we need to invert is neither SDD nor sparse. Fortunately, we can combine the Laplacian system solver with the Sherman-Morrison formula to invert our matrices; see Theorem 4.4. A significant effort goes into analyzing the effect of the error introduced due to approximate matrix inversion. This error can cascade due to the iterative nature of our algorithm that proves this theorem.

When the only guarantee we know on the matrix is that it is symmetric and PSD, we can use the Conjugate Gradient method to approximate the matrix inverse and obtain the following theorem, which is the best known algorithm to compute $\exp(-A)v$ for an arbitrary symmetric PSD matrix A, when $\|A\| = \omega(\text{poly}(\log n))$.

Theorem 2.4 (PSD Matrix Exponential Computation)**.** *Given an $n \times n$ symmetric PSD matrix A, a vector v and a parameter $\delta \leq 1$, there is an algorithm that computes a vector u such that $\|\exp(-A)v - u\| \leq \delta \|v\|$ in time $\tilde{O}\left((t_A + n)\sqrt{1 + \|A\|}\log(2 + \|A\|)\right)$. Here the tilde hides poly$(\log n)$ and poly$(\log {}^1\!/\!_\delta)$ factors.*

In the symmetric PSD setting, we also prove the following theorem, which gives a result comparable to Theorem 2.4. This result is comparable to, and implied by an earlier independent work of Hochbruck and Lubich (Theorem 2 in [HL97]). In fact, both Theorem 2.5 and the result of Hochbruck and Lubich follow immediately

[6] A is said to be *Symmetric and Diagonally Dominant* (SDD) if, $A_{ij} = A_{ji}$, for all i, j and $A_{ii} \geq \sum_{j \neq i} |A_{ij}|$, for all i. In particular, every Laplacian matrix is SDD.

by combining the Lanczos method with improved polynomial approximations for the exponential function. We discuss polynomial approximations to e^{-x} in Section 2.4.

Theorem 2.5 (Simple PSD Matrix Exponential Computation)**.** *Given an $n \times n$ symmetric PSD matrix A, a vector v and a parameter $\delta \leq 1$, there is an algorithm that computes a vector u such that $\|\exp(-A)v - u\| \leq \delta \|v\|$, in time $O((t_A + n) \cdot k + k^2)$, where $k \stackrel{\text{def}}{=} \tilde{O}(\sqrt{1 + \|A\|})$. Here the tilde hides poly$(\log {}^1/_\delta)$ factors.*

As noted above, t_A can be significantly smaller than m_A. Moreover, it only uses multiplication of a vector with the matrix A as a primitive and does not require matrix inversion. Consequently, it does not need tools like the SDD solver or Conjugate Gradient method, thus obviating the error analysis required for the previous algorithms. Furthermore, this algorithm is very simple and when combined with our random walk-based BalSep algorithm, results in a very simple and practical $O(\sqrt{\gamma})$ approximation algorithm for Balanced Separator that runs in time $\tilde{O}({}^m/_{\sqrt{\gamma}})$.

2.3 Matrix Inversion is as easy as Exponentiation

The results on approximating the matrix exponential from the last section show that (approximate) matrix exponentiation can be reduced to a small number of (approximate) matrix inversions. For our application to Balanced Separator, it was sufficient to combine this result with the near-linear-time SDD solver by Spielman and Teng.

However, these results leave a few important questions unanswered. The first question arises because all known SDD or Laplacian system solvers require tools beyond simple linear algebra operations, such as the computation of a low-stretch spanning tree (*e.g.* [ST04, KMP11, KOSZ13]). Hence we should ask whether such a solver is necessary in order to compute $e^{-L}v$ in near-linear time (see [Vis12, Chapter 9]). Secondly, and more obviously, these results fail to resolve whether there is a fast algorithm for approximating the matrix exponential for all PSD matrices.

We answer the first question in the affirmative, and give evidence that a fast algorithm that approximates the exponential for all PSD matrices would require techniques beyond our current reach, by presenting a reduction in the other direction, again relying on analytical techniques. The following is our main result.

Theorem 2.6. *Given $\varepsilon, \delta \in (0,1]$, there exist poly*$(\log {}^1\!/_{\delta\varepsilon})$ *numbers* $0 < w_j, t_j = O($*poly*$({}^1\!/_{\delta\varepsilon}))$, *such that for all symmetric matrices A satisfying $\delta I \preceq A \preceq I$, $(1-\varepsilon)A^{-1} \preceq \sum_j w_j e^{-t_j A} \preceq (1+\varepsilon)A^{-1}$.*

This proves that the problems of (approximate) matrix exponentiation and (approximate) matrix inversion are equivalent up to polylogarithmic factors. Versions of this lemma were proved by Beylkin and Mònzon [BM05, BM10]. Our proof is simple and self-contained.

Let us consider the implications of this result to the two questions raised above. Firstly, it justifies the somewhat surprising use of Laplacian solvers for approximating the matrix exponential, and in particular, for algorithms based on the Matrix Multiplicative Weights Update method for problems like Balanced Separator.

Secondly, since this equivalence does not require the matrix A to be a Laplacian, but only that it be a symmetric positive-definite matrix, this implies that a fast algorithm for approximating the matrix exponential could be used to approximate $A^{-1}v$ with only a loss of polylogarithmic factors in the running time. While such fast solvers are known for linear systems arising from SDD matrices, as discussed earlier, extending these results to all PSD matrices is an important open problem, and it would be interesting to investigate if this approach can be used to make some progress.

2.4 Uniform approximations to e^{-x}

A straightforward approach to approximate the matrix-exponential-vector product $\exp(-A)v$ is to consider a polynomial $p(x)$ that approximates e^{-x} on the interval $[0, \|A\|]$ up to a point-wise error of δ, and return the approximation $p(A)v$. As a concrete instantiation of this strategy, we already observed that we could truncate the Taylor expansion of e^{-x} to obtain such approximating polynomials. The degree of these polynomials required in order to obtain a good approximation is roughly $\|A\|$. For our application to the Balanced Separator problem, $\|A\|$ is of the order of $1/\gamma$ which can be polynomially large in n.

This serves as additional motivation for the following natural question: Do there exist polynomials approximating e^{-x} on the interval $[0, \|A\|]$, with their dependence on $\|A\|$ asymptotically better than linear?

We answer this question in the affirmative and show the existence of such polynomials with degree roughly $\sqrt{\|A\|}$. We also show a matching lower bound that proves that, asymptotically, this is the best dependence possible. This gives further evidence that significantly improving the running time of Theorem 2.5 for general PSD matrices requires more advanced techniques. The following theorem states both these results formally:

Theorem 2.7 (Uniform Approximation to e^{-x}). ·

- **Upper Bound.** *For every* $0 < b$, *and* $0 < \delta \leq 1$, *there exists a polynomial* p *that satisfies,* $\sup_{x \in [0,b]} |e^{-x} - p(x)| \leq \delta$, *and has degree* $\tilde{O}(\sqrt{b})$, *where the tilde hides* poly$(\log {}^1/_\delta)$ *factors.*

- **Lower Bound.** *For every* $\log_e 4 \leq b$, *and* $\delta \in (0, {}^1/_8]$, *any polynomial* $p(x)$ *that approximates* e^{-x} *uniformly over the interval* $[0, b]$ *up to an error of* δ, *must have degree at least* $\frac{1}{2} \cdot \sqrt{b}$.

We note that the upper bound from the above theorem is implicit in the work of Hochbruck and Lubich [HL97] (This was pointed out to us by an anonymous reviewer for [OSV12]). We give a different proof of this result, one that starts from the rational approximation result of Saff, Schönhage, and Varga [SSV75] mentioned earlier, and avoids complex analysis, unlike the result from [HL97].

The upper bound from the above theorem is required for Theorem 2.5. A concern to address though, is that the polynomials given by Theorem 2.7 are not known explicitly. We address this issue by using the Lanczos method from numerical linear algebra that allows us to convert guarantees about polynomial approximation from scalars to matrices, without explicit knowledge of the polynomials. Thus, just the existence of good polynomial approximations allows us to prove Theorem 2.5.

Chapter 3

Fast Spectral Algorithm for Balanced Separator

In this chapter, we give a novel spectral approximation algorithm for the Balanced Separator problem that, given a graph G, a constant balance $b \in (0, 1/2]$, and a parameter γ, either finds an $\Omega(b)$-balanced cut of conductance $O(\sqrt{\gamma})$ in G, or outputs a certificate that all b-balanced cuts in G have conductance at least γ, and runs in time $\tilde{O}(m)$ [1]. This settles the question of designing asymptotically optimal spectral algorithms for Balanced Separator. Our algorithm relies on a variant of the heat kernel random walk and requires, as a subroutine, an algorithm to compute $\exp(-L)v$ where L is the Laplacian of a graph related to G, and v is a vector. In Chapter 4, we describe the algorithms for computing $\exp(-L)v$, which, when combined with the spectral algorithm described in this chapter, prove Theorems 3.1 and 3.2.

3.1 Our Results

In this chapter, we describe our algorithm for the Balanced Separator problem, called BalSep, that underlies theorem 2.1. Our algorithm requires the ability to approx-

[1] The tilde notation hides polylogarithmic factors in the argument, unless otherwise specified.

imate $\exp(-A)v$ for a given matrix A, and a vector v. Combining the algorithm from this chapter, with the near-linear-time algorithm for approximating the matrix-exponential-vector product for the required class of matrices given by Theorem 4.4, and described in Chapter 4, we are able to complete the proof of Theorem 2.1. We restate the theorem here for completeness:

Theorem 3.1 (Theorem 2.1 restated)**.** *Given an unweighted graph $G = (V, E)$, a balance parameter $b \in (0, 1/2]$, $b = \Omega(1)$ and a conductance value $\gamma \in (0, 1]$, we give an algorithm called* $\textsc{BalSep}(G, b, \gamma)$*, that either outputs an $\Omega(b)$-balanced cut $S \subset V$ such that $\phi(S) \leq O(\sqrt{\gamma})$, or outputs a certificate that no b-balanced cut of conductance γ exists.* $\textsc{BalSep}$ *runs in time $O(m\ \mathit{poly}(\log n))$.*

As pointed out in the introduction, our algorithm, $\textsc{BalSep}$, can be combined with other algorithms to approximate the matrix-exponential-vector product. In particular, when combined with the algorithm in Theorem 2.5, we obtain a very simple and practical algorithm for Balanced Separator that matches the running time, but improves on the approximation guarantee of the Evolving-Sets-based algorithm by Andersen and Peres [AP09]. We record the theorem here for completeness and then move on to the overview of $\textsc{BalSep}$ and its proof. The proof of this theorem appears in Section 3.5.

Theorem 3.2 (Simple Spectral Algorithm for Balanced Separator)**.** *Given an unweighted graph $G = (V, E)$, a balance parameter $b \in (0, 1/2]$, $b = \Omega(1)$ and a conductance value $\gamma \in (0, 1]$, we give an algorithm, which runs in time $\tilde{O}(m/\sqrt{\gamma})$, that either outputs an $\Omega(b)$-balanced cut $S \subset V$ such that $\phi(S) \leq O(\sqrt{\gamma})$ or outputs a certificate that no b-balanced cut of conductance γ exists.*

Throughout this chapter, we fix G to denote the input graph, γ to denote the target conductance value, and b to denote the target balance parameter.

3.1.1 Comparison with the Recursive Eigenvector Algorithm

Before we explain our algorithm, it is useful to review the Recursive Laplacian Eigenvector (RLE) algorithm. Recall that given G, γ and b, the goal of the Balanced Separator problem is to either certify that every b-balanced cut in G has conductance at least γ, or produce a $\Omega(b)$ balance cut in G of conductance $O(\sqrt{\gamma})$. RLE does this by applying LE iteratively to remove unbalanced cuts of conductance $O(\sqrt{\gamma})$ from G. The iterations stop and the algorithm outputs a cut, when it either finds a $(b/2)$-balanced cut of conductance $O(\sqrt{\gamma})$ or the union of all unbalanced cuts found so far is $(b/2)$-balanced. Otherwise, the algorithm terminates when the residual graph has spectral gap at least 2γ. In the latter case, any b-balanced cut must have at least half of its volume lie within the final residual graph, and hence, has conductance at least γ in the original graph. Unfortunately, this algorithm may require $\Omega(n)$ iterations in the worst case. For instance, this is true if the graph G consists of $\Omega(n)$ components loosely connected to an expander-like core through cuts of low conductance. This example highlights the weakness of the RLE approach: the second eigenvector of the Laplacian may only be correlated with one low-conductance cut and fail to capture at all even cuts of slightly larger conductance. This limitation makes it impossible for RLE to make significant progress at any iteration. We now proceed to show how to fix RLE and present our algorithm at a high level.

3.1.2 High-Level Idea of Our Algorithm

Rather than working with the vertex embedding given by the eigenvector, at iteration t, we will consider the multi-dimensional vector embedding represented by the transition probability matrix $P^{(t)}$ of a certain random walk over the graph. We refer to this kind of walk as an *Accelerated Heat Kernel Walk* (AHK) and we describe it formally in Section 3.3.

At each iteration $t = 1, 2, \ldots$, the current AHK walk is simulated for $\tau = {}^{\log n}/_{\gamma}$ time to obtain $P^{(t)}$. For any t, this choice of τ ensures that the walk must mix across all cuts of conductance much larger than γ, hence emphasizing cuts of the desired conductance in the embedding $P^{(t)}$.

The embedding obtained in this way, can be seen as a weighted combination of multiple eigenvectors, with eigenvectors of low eigenvalue contributing more weight. Hence, the resulting embedding captures not only the cut corresponding to the second eigenvector, but also cuts associated with other eigenvectors of eigenvalue close to γ. This enables our algorithm to potentially find many different low-conductance unbalanced cuts at once. Moreover, the random walk matrix is more stable than the eigenvector under small perturbations of the graph, making it possible to precisely quantify our progress from one iteration to the next as a function of the mixing of the current random walk.

For technical reasons, we are unable to show that we make sufficient progress if we just remove the unbalanced cuts found, as in RLE. Instead, if we find a low-conductance unbalanced cut $S^{(t)}$ at iteration t, we perform a *soft* removal, by modifying the current walk $P^{(t)}$ to accelerate the convergence to stationarity on the set $S^{(t)}$. This ensures that a different cut is found using $P^{(t+1)}$ in the next iteration. In particular, the AHK walks we consider throughout the execution of the algorithm will behave like the standard heat kernel walk on most of the graph, except on a small unbalanced subset of vertices, where their convergence will be accelerated.

3.2 Definitions and Preliminaries

We first present some preliminaries for this chapter.

Instance Graph and Edge Volume. We denote by $G = (V, E)$ the unweighted instance graph, where $V = [n]$ and $|E| = m$. We assume G is connected. We let

$d \in \mathbb{R}^n$ be the degree vector of G, i.e. d_i is the degree of vertex i. For a subset $S \subseteq V$, we define the edge volume as $\mathsf{vol}(S) \stackrel{\text{def}}{=} \sum_{i \in S} d_i$. The total volume of V is $2m$. The conductance of a cut $(S, \bar{S})$ is defined to be $\phi(S) \stackrel{\text{def}}{=} {|E(S,\bar{S})|}/{\min\{\mathsf{vol}(S), \mathsf{vol}(\bar{S})\}}$. Moreover, a cut $(S, \bar{S})$ is b-balanced if $\min\{\mathsf{vol}(S), \mathsf{vol}(\bar{S})\} \geq b \cdot \mathsf{vol}(V)$.

Special Graphs. Let K_V denote the complete graph with an edge of weight ${d_i d_j}/{2m}$ between every pair $i, j \in V$. For $S \subseteq V$, let K_S denote the complete graph on S, with an edge of weight ${d_i d_j}/{\mathsf{vol}(S)}$ between every pair $i, j \in S$. For $i \in V$, let Star_i denote the star graph rooted at i, with an edge of weight ${d_i d_j}/{2m}$ between i and j, for all $j \in V$.

Graph matrices. For an undirected graph $H = (V, E_H)$, let $A(H)$ denote the adjacency matrix of H, and $D(H)$ the diagonal matrix of degrees of H. The (combinatorial) Laplacian of H is defined as $L(H) \stackrel{\text{def}}{=} D(H) - A(H)$. Note that for all $x \in \mathbb{R}^V$, $x^\top L(H) x = \sum_{\{i,j\} \in E_H} (x_i - x_j)^2$. By D and L, we denote $D(G)$ and $L(G)$ respectively for the input graph G. Finally, the natural random walk over G has transition matrix $W \stackrel{\text{def}}{=} AD^{-1}$.

Vector and Matrix Notation. We are working within the vector space $\mathbb{R}^n$. For a vector $x \in \mathbb{R}^n$, let $\mathrm{supp}(x)$ be the set of vertices where x is not zero. We denote by $\{e_i\}_{i=1}^n$ the standard basis for $\mathbb{R}^n$. $\mathbf{0}$ and $\mathbf{1}$ will denote the all 0s and all 1s vectors respectively. We will denote by I the identity matrix over this space. For a symmetric matrix A, denote by $\lambda_i(A)$, the i^{th} smallest eigenvalue of A. For a symmetric matrix A, we will use $A \succeq 0$ to indicate that A is positive semi-definite. The expression $A \succeq B$ is equivalent to $A - B \succeq 0$. For two matrices A, B of equal dimensions, let $A \bullet B \stackrel{\text{def}}{=} \mathrm{Tr}\left(A^\top B\right) = \sum_{ij} A_{ij} \cdot B_{ij}$.

Fact 3.3. $L(K_V) = D - {1}/{2m} \cdot D\mathbf{1}\mathbf{1}^\top D = D^{1/2}(I - {1}/{2m} \cdot D^{1/2}\mathbf{1}\mathbf{1}D^{1/2})D^{1/2}$.

Embedding Notation. We will deal with vector embeddings of G, where each vertex $i \in V$ is mapped to a vector $v_i \in \mathbb{R}^d$, for some $d \leq n$. For such an embedding $\{v_i\}_{i\in V}$, we denote by v_{avg} the mean vector, i.e. $v_{\mathsf{avg}} \overset{\text{def}}{=} \sum_{i\in V} {}^{d_i}/_{2m} \cdot v_i$. Given a vector embedding $\{v_i \in \mathbb{R}^d\}_{i\in V}$, recall that X is the Gram matrix of the embedding if $X_{ij} = v_i^\top v_j$. A Gram matrix X is always PSD, *i.e.*, $X \succeq 0$. For any $X \in \mathbb{R}^{n\times n}$, $X \succeq 0$, we call $\{v_i\}_{i\in V}$ the *embedding corresponding to* X if X is the Gram matrix of $\{v_i\}_{i\in V}$. For $i \in V$, let R_i denote the matrix such that $R_i \bullet X = \|v_i - v_{\mathsf{avg}}\|^2$.

Fact 3.4. $\sum_{i\in V} d_i R_i \bullet X = \sum_{i\in V} d_i \|v_i - v_{\mathsf{avg}}\|^2 = {}^1/_{2m} \cdot \sum_{i<j} d_j d_i \|v_i - v_j\|^2 = L(K_V) \bullet X.$

3.3 Accelerated Heat Kernel (AHK) Random Walks

The random-walk processes used by our algorithm are continuous-time Markov processes [Par99] over V. In these processes, state transitions do not take place at specified discrete intervals, but follow exponential distributions described by a *transition rate matrix* $Q \in R^{n\times n}$, where Q_{ij} specifies the rate of transition from vertex j to i. More formally, letting $p(\tau) \in \mathbb{R}^n$ be the probability distribution of the process at time $\tau \geq 0$, we have that ${}^{\partial p(\tau)}/_{\partial\tau} = Qp(\tau)$ Given a transition rate matrix [2] Q, the differential equation for $p(\tau)$ implies that $p(\tau) = e^{\tau Q}p(0)$. We will be interested in a class of continuous-time Markov processes over V that take into account the edge structure of G. The simplest such process is the *heat kernel* process, which is defined as having transition rate matrix $Q = -(I - W) = -LD^{-1}$. The heat kernel can also be interpreted as the probability transition matrix of the following discrete-time random walk: sample a number of steps i from a Poisson distribution with mean τ

[2] A matrix Q is a valid transition rate matrix if its diagonal entries are non-positive and its off-diagonal entries are non-negative. Moreover, it must be that $\mathbf{1}Q = 0$, to ensure that probability mass is conserved.

and perform i steps of the natural random walk over G :

$$p(\tau) = e^{-\tau L D^{-1}} p(0) = e^{-\tau (I-W)} p(0) = e^{-\tau} \sum_{i=0}^{\infty} \frac{\tau^i}{i!} W^i p(0).$$

We generalize the concept of heat kernel to a larger class of continuous-time Markov processes, which we name *Accelerated Heat Kernel* (AHK) processes. A process $\mathcal{H}(\beta)$ in this class is defined by a non-negative vector $\beta \in \mathbb{R}^n$, and the transition rate matrix of $\mathcal{H}(\beta)$ is $Q(\beta) \stackrel{\text{def}}{=} -(L + \sum_{i \in V} \beta_i L(\mathsf{Star}_i))D^{-1}$. As this is the negative of a sum of Laplacian matrices, it is easy to verify that it is a valid transition rate matrix. Adding the star terms to the transition rate matrix has the effect accelerating the convergence of the process to stationarity at vertices i with large value of β_i, as a large fraction of the probability mass that leaves these vertices is distributed uniformly over the edges. We denote by $P_\tau(\beta)$ the probability-transition matrix of $\mathcal{H}(\beta)$ between time 0 and τ, i.e. $P_\tau(\beta) = e^{\tau Q(\beta)}$.

Finally, to connect to the high-level idea described earlier, for each iteration t, we use $P^{(t)} \stackrel{\text{def}}{=} P_\tau(\beta^{(t)})$ for $\tau = {}^{\log n}/_{\gamma}$ with $\beta^{(t)} \approx {}^1/_\tau \sum_{j=1}^{t-1} \sum_{i \in S^{(j)}} e_i$ and starting with $\beta^{(1)} = \mathbf{0}$.

Embedding View.

A useful matrix to study $\mathcal{H}(\beta)$ will be $D^{-1}P_{2\tau}(\beta)$. This matrix describes the probability distribution over the edges of G and has the advantage of being symmetric and positive semidefinite:

$$D^{-1}P_{2\tau}(\beta) = D^{-1/2} e^{-(2\tau) D^{-1/2}(L + \sum_{i \in V} \beta_i L(\mathsf{Star}_i))D^{-1/2}} D^{-1/2},$$

Moreover, we have the following fact:

Fact 3.5. *$D^{-1/2}P_\tau(\beta)$ is a square root of $D^{-1}P_{2\tau}(\beta)$.*

Proof.

$$\left(D^{-1/2}P_\tau(\beta)\right)^\top D^{-1/2}P_\tau(\beta) = e^{\tau(Q(\beta))^\top} D^{-1} e^{\tau Q(\beta)} = D^{-1}e^{\tau Q(\beta)}e^{\tau Q(\beta)} = D^{-1}e^{2\tau Q(\beta)}.$$

□

Hence, $D^{-1}P_{2\tau}(\beta)$ is the Gram matrix of the embedding given by the columns of its square root $D^{-1/2}P_\tau(\beta)$. This property will enable us to use geometric SDP techniques to analyze $\mathcal{H}(\beta)$.

Mixing.

Spectral methods for finding low-conductance cuts are based on the idea that random walk processes mix slowly across sparse cuts, so that it is possible to detect such cuts by considering the starting vertices for which the probability distribution of the process strongly deviates from the stationary distribution. We measure this deviation for vertex i at time t by the ℓ_2^2-norm of the distance between $P_\tau(\beta)e_i$ and the uniform distribution *over the edges of* G. We denote it by $\Psi(P_\tau(\beta), i)$:

$$\Psi(P_\tau(\beta), i) \overset{\text{def}}{=} d_i \sum_{j \in V} d_j \left(\frac{e_j^\top P_\tau(\beta) e_i}{d_j} - \frac{1}{2m}\right)^2 .$$

A fundamental quantity for our algorithm will be the total deviation from stationarity over a subset $S \subseteq V$. We will denote $\Psi(P_t(\beta), S) \overset{\text{def}}{=} \sum_{i \in S} \Psi(P_t(\beta), i)$. In particular, $\Psi(P_\tau(\beta), V)$ will play the role of potential function in our algorithm. The following facts express these mixing quantities in the geometric language of the embedding corresponding to $D^{-1}P_{2\tau}(\beta)$.

Fact 3.6. $\Psi(P_\tau(\beta), i) = d_i R_i \bullet D^{-1}P_{2\tau}(\beta)$.

Proof. By Fact 3.5 and the definition of R_i :

$$\begin{aligned} d_i R_i \bullet D^{-1} P_{2\tau}(\beta) &= d_i \left\| D^{-1/2} P_\tau(\beta) e_i - \sum_{j \in V} \frac{d_j}{2m} D^{-1/2} P_\tau(\beta) e_j \right\|^2 \\ &= d_i \left\| D^{-1/2} P_\tau(\beta) e_i - \frac{D^{1/2}\mathbf{1}}{2m} \right\|^2 \\ &= d_i \left\| D^{1/2} \left(D^{-1} P_\tau(\beta) e_i - \frac{\mathbf{1}}{2m} \right) \right\|^2 = \Psi(P_\tau(\beta), i). \end{aligned}$$

□

The following is a consequence of Fact 3.4:

Fact 3.7. $\Psi(P_\tau(\beta), V) = \sum_{i \in V} d_i R_i \bullet D^{-1} P_{2\tau}(\beta) = L(K_V) \bullet D^{-1} P_{2\tau}(\beta).$

3.4 Algorithm Description

The algorithm BALSEP is formally described in Figure 3.1. Below, we present an overview of the algorithm.

All the random walks in our algorithm will be run for time $\tau \overset{\text{def}}{=} O(\log n)/\gamma$. We will consider embeddings given by the columns of $D^{-1/2} P_\tau(\beta)$ for some choice of β. Since we require our algorithm to run in time $\tilde{O}(m)$, and we are only interested in Euclidean distances between vectors in the embedding, we will use the Johnson-Lindenstrauss Lemma (see Lemma 3.20 in Section 3.6) to obtain an $O(\log n)$-dimensional embedding approximately preserving distances between columns of $D^{-1/2} P_\tau(\beta)$ up to a factor of $(1+\varepsilon)$, where we pick $\varepsilon \overset{\text{def}}{=} 1/7$. (This choice of ε satisfies $1+\varepsilon/1-\varepsilon \leq 4/3$.)

Required Subroutines. Our algorithm BALSEP will call two subroutines FINDCUT and EXPV. FINDCUT is an SDP-rounding algorithm that uses random projections and radial sweeps to find a low-conductance cut, that is either c-balanced, for some constant $c = \Omega(b) \leq b/100$, or obeys a strong guarantee stated in Theorem 3.10.

Such an algorithm is implicit in [OV11] and is described precisely in Section 3.8. ExpV is a generic algorithm that approximately computes products of the form $P_\tau(\beta)u$ for unit vectors u. Expv can be chosen to be either the algorithm implied by Theorem 4.4, which makes use of the Spielman-Teng solver, or that in Theorem 4.5, which just applies the Lanczos method.

Algorithm Overview. The algorithm BalSep will output a c-balanced cut of conductance $O(\sqrt{\gamma})$ or the string No, if it finds a certificate that no b-balanced cut of conductance less than γ exists. BalSep can also fail and output the string Fail. We will show that this only happens with small probability. The constants in the proof have not been optimized and are likely larger than necessary. They can also be modified to obtain different trade-offs between the approximation guarantee and the output balance.

At iteration $t = 1$, we have $\beta^{(1)} = \mathbf{0}$, so that $P^{(1)}$ is just the probability transition matrix of the heat kernel on G for time τ. In general at iteration t, BalSep runs ExpV to compute $O(\log n)$ random projections of $P^{(t)}$ and constructs an approximation $\{v_i^{(t)}\}_{i\in V}$ to the embedding given by the columns of $D^{-1/2}P^{(t)}$. This approximate embedding has Gram matrix $X^{(t)}$.

In Step 2, BalSep computes $L(K_V) \bullet X^{(t)}$, which is an estimate of the total deviation $\Psi(P^{(t)}, V)$ by Fact 3.7. If this deviation is small, the AHK walk $P^{(t)}$ has mixed sufficiently over G to yield a certificate that G cannot have any b-balanced cut of conductance less than γ. This is shown in Lemma 3.8. If the AHK walk $P^{(t)}$ has not mixed sufficiently, we can use FindCut to find a cut $S^{(t)}$ of low conductance $O(\sqrt{\gamma})$, which is an obstacle for mixing. If $S^{(t)}$ is c-balanced , we output it and terminate. Otherwise, $S^{(t)}$ is unbalanced and is potentially preventing BalSep from detecting balanced cuts in G. We then proceed to modify the AHK walk, by increasing the

```
Input: An unweighted connected instance graph $G = (V, E)$, a constant balance value $b \in (0, 1/2]$, a conductance value $\gamma \in [1/m, 1)$.
Set $\tau \stackrel{\text{def}}{=} \log n/12\gamma$ and $\beta^{(1)} \stackrel{\text{def}}{=} \mathbf{0}$.
At iteration $t = 1, \ldots, T \stackrel{\text{def}}{=} 12 \log n$ :

    1. Denote $P^{(t)} \stackrel{\text{def}}{=} P_\tau(\beta^{(t)})$. Pick $k \stackrel{\text{def}}{=} O(\log n/\varepsilon^2)$ random unit vectors $\{u_1^{(t)}, u_2^{(t)}, \ldots, u_k^{(t)} \in \mathbb{R}^n\}$ and use the subroutine ExpV to compute the embedding $\{v_i^{(t)} \in \mathbb{R}^k\}_{i \in V}$ defined as

       $$\left(v_i^{(t)}\right)_j \stackrel{\text{def}}{=} \sqrt{\frac{n}{k}} u_j^\top D^{-1/2} P^{(t)} e_i.$$

       Let $X^{(t)}$ be the Gram matrix corresponding to this embedding.

    2. If $L(K_V) \bullet X^{(t)} = \sum_{i \in V} d_i ||v_i^{(t)} - v_{\mathsf{avg}}^{(t)}||^2 \leq \frac{1+\varepsilon}{n}$, output No and terminate.

    3. Otherwise, run FindCut$(G, b, \gamma, \{v_i^{(t)}\}_{i \in V})$. FindCut outputs a cut $S^{(t)}$ with $\phi(S^{(t)}) \leq O(\sqrt{\gamma})$ or fails, in which case we also output Fail and terminate.

    4. If $S^{(t)}$ is $c$-balanced, output $S^{(t)}$ and terminate.

    5. Otherwise, update $\beta^{(t+1)} \stackrel{\text{def}}{=} \beta^{(t)} + \frac{72\gamma}{T} \sum_{i \in S^{(t)}} e_i$ and proceed to the next iteration.

Output No and terminate.
```

Figure 3.1: The BalSep algorithm

values of $\beta^{(t+1)}$ for the vertices in $S^{(t)}$. This change ensures that $P^{(t+1)}$ mixes faster from the vertices in $S^{(t)}$ and in particular mixes across $S^{(t)}$.

The BalSep algorithm exactly parallels the RLE algorithm discussed earlier, introducing only two fundamental changes. First, we use the embedding given by the AHK random walk $P^{(t)}$ in place of the eigenvector to find cuts in G or in a residual graph. Secondly, rather than fully removing unbalanced low-conductance cuts from the graph, we modify $\beta^{(t)}$ at every iteration t, so $P^{(t+1)}$ at the next iteration mixes across the unbalanced cuts found so far.

3.5 Analysis of the Algorithm

At the heart of the analysis of BalSep is a modification of the Matrix Multiplicative Weights update method from [OV11, AK07], stated in the language of random walks. This modification allows us to deal with the different embedding used by BalSep at each iteration as compared to [OV11].

In the analysis, the quantity $\Psi(P^{(t)}, V)$ plays the role of potential function. Recall that, from a random-walk point of view, $\Psi(P^{(t)}, V)$ is the total deviation from stationarity of $P_\tau(\beta^{(t)})$ over all vertices as starting points. We start by showing that if the potential function is small enough, we obtain a certificate that no b-balanced cut of conductance at most γ exists. In the second step, we show that, if BalSep finds an unbalanced cut $S^{(t)}$ of low conductance, the potential decreases by a constant fraction. Unless explicitly stated otherwise, the proofs of all lemmas have been deferred to Section 3.6.

Potential Guarantee.

We argue that, if $\Psi(P^{(t)}, V)$ is sufficiently small, it must be the case that G has no b-balanced cut of conductance less than γ. A similar result is implicit in OV. This theorem has a simple explanation in terms of the AHK random walk $P^{(t)}$. Notice that in each iteration t, we increase the acceleration of $P^{(t)}$ only by a tiny amount on a small unbalanced set $S^{(t)}$. Hence, if a balanced cut of conductance less than γ existed, its convergence could not be greatly helped. Then, if $P^{(t)}$ is still mixing very well, no such balanced cut can exist.

Lemma 3.8. *If $\Psi(P^{(t)}, V) \leq \frac{4}{3n}$, and for each $i = 1, \ldots, t$, we have $\mathsf{vol}(S^{(i)}) \leq c \cdot 2m \leq {}^{b}/{}_{100} \cdot 2m$, then*

$$L + \sum_{i \in V} \beta_i^{(t)} L(\mathsf{Star}_i) \succeq 3\gamma \cdot L(K_V).$$

Moreover, this implies that no b-balanced cut of conductance less than γ exists in G.

The Deviation of an Unbalanced Cut.

In the next step, we show that, if the walk has not mixed sufficiently, w.h.p. the embedding $\{v_i^{(t)}\}_{i \in V}$, computed by BalSep, has low quadratic form with respect to the Laplacian of G. From a SDP-rounding perspective, this means that the embedding can be used to recover cuts with conductance close to γ. This part of the analysis departs from that in [OV11], as we use our modified definition of the embedding.

Lemma 3.9. *If $\Psi(P^{(t)}, V) \geq \frac{1}{n}$, then w.h.p. $L \bullet X^{(t)} \leq O(\gamma) \cdot L(K_V) \bullet X^{(t)}$.*

This guarantee on the embedding allows us to apply SDP-rounding techniques in the subroutine FindCut. The following result is implicit in [OV11]. Its proof appears in Section 3.8 for completeness.

Theorem 3.10. *Consider an embedding $\{v_i \in \mathbb{R}^d\}_{i \in V}$ with Gram matrix X such that $L \bullet X^{(t)} \leq \alpha L(K_V) \bullet X^{(t)}$, for $\alpha > 0$. On input $(G, b, \alpha, \{v_i\}_{i \in V})$, FindCut runs in time $\tilde{O}(md)$ and w.h.p. outputs a cut C with $\phi(C) \leq O(\sqrt{\alpha})$. Moreover, there is a constant $c = \Omega(b) \leq {}^{b}/_{100}$ such that either C is c-balanced or*

$$\sum_{i \in C} d_i R_i \bullet X \geq {}^{2}/_{3} \cdot L(K_V) \bullet X.$$

The following corollary is a simple consequence of Lemma 3.9 and Theorem 3.10

Corollary 3.11. *At iteration t of BalSep, if $\Psi(P^{(t)}, V) \geq \frac{1}{n}$ and $S^{(t)}$ is not c-balanced, then w.h.p. $\Psi(P^{(t)}, S^{(t)}) \geq {}^{1}/_{2} \cdot \Psi(P^{(t)}, V)$.*

In words, at the iteration t of BalSep, the cut $S^{(t)}$ must either be c-balanced or be an unbalanced cut that contributes a large constant fraction of the total deviation of $P^{(t)}$ from the stationary distribution. In this sense, $S^{(t)}$ is the main reason for the failure of $P^{(t)}$ to achieve better mixing. To eliminate this obstacle and drive the potential further down, $P^{(t)}$ is updated to $P^{(t+1)}$ by accelerating the convergence to stationary from all vertices in $S^{(t)}$. Formally, this is achieved by adding weighted stars rooted at all vertices over $S^{(t)}$ to the transition-rate matrix of the AHK random walk $P^{(t)}$.

Potential Reduction.

The next theorem crucially exploits the stability of the process $\mathcal{H}(\beta^{(t)})$ and Corollary 3.11 to show that the potential decreases by a constant fraction at every iteration in which an unbalanced cut is found. More precisely, the theorem shows that accelerating the convergence from $S^{(t)}$ at iteration t of BalSep has the effect of eliminating at least a constant fraction of the total deviation due to $S^{(t)}$. The proof is a simple application of the Golden-Thompson inequality [Bha96] and mirrors the main step in the Matrix Multiplicative Weights Update analysis.

Theorem 3.12. *At iteration t of* BalSep*, if $\Psi(P^{(t)}, V) \geq \frac{1}{n}$ and $S^{(t)}$ is not c-balanced, then w.h.p.*

$$\Psi(P^{(t+1)}, V) \leq \Psi(P^{(t)}, V) - {}^{1}\!/_{3} \cdot \Psi(P^{(t)}, S^{(t)}) \leq {}^{5}\!/_{6} \cdot \Psi(P^{(t)}, V).$$

We are now ready to prove Theorem 3.1 and Theorem 3.2 by applying Lemma 3.12 to show that after $O(\log n)$ iterations, the potential must be sufficiently low to yield the required certificate according to Lemma 3.8.

Proof of Theorem 2.1. If BalSep outputs a cut $S^{(t)}$ in Step 4, by construction, we have that $\phi(S) \leq O(\sqrt{\gamma})$ and S is $\Omega(b)$-balanced. Alternatively, at iteration t, if $L(K_V) \bullet X^{(t)} \leq {}^{1+\varepsilon}\!/_{n}$, we have by Lemma 3.20 that

$$\Psi(P^{(t)}, V) = L(K_V) \bullet D^{-1} P_{2\tau}(\beta^{(t)}) \leq \frac{1}{1-\varepsilon} L(K_V) \bullet X^{(t)} \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot \frac{1}{n} \leq \frac{4}{3n}.$$

Therefore, by Lemma 3.8, we have a certificate that no b-balanced cut of conductance less than γ exists in G. Otherwise, we must have $L(K_V) \bullet X^{(t)} \geq {}^{1+\varepsilon}/_{n}$, which, by Lemma 3.20, implies that

$$\Psi(P^{(t)}, V) \geq {}^{1}/_{n}.$$

Then, by Lemma 3.9 and Theorem 3.10, we have w.h.p. that FindCut does not fail and outputs a cut $S^{(t)}$ with $\phi(S^{(t)}) \leq O(\sqrt{\gamma})$. As BalSep has not terminated in Step 4, it must be the case that $S^{(t)}$ is not c-balanced and, by Theorem 3.12, we obtain that w.h.p. $\Psi(P^{(t+1)}, V) \leq {}^{5}/_{6} \cdot \Psi(P^{(t)}, V)$. Now,

$$\Psi(P^{(1)}, V) = L(K_V) \bullet D^{-1} P_{2\tau}(\mathbf{0}) \leq I \bullet P_{2\tau}(\mathbf{0}) \leq n.$$

Hence, after ${}^{2\log n}/_{\log(6/5)} \leq 12 \log n = T$ iterations, w.h.p. we have that $\Psi(P^{(T)}, V) \leq {}^{1}/_{n}$ and, by Lemma 3.8, no b-balanced cut of conductance less than γ exists.

We now consider the running time required by the algorithm at every iteration. In Step 1, we compute $k = O(\log n)$, products of the form $D^{-1/2} P^{(t)} u$, where u is an unit vector, using the ExpV algorithm based on the Spielman-Teng solver, given in Theorem 4.4. This application of Theorem 4.4 is explained in Section 4.1.1. By the definition of $\beta^{(t)}$, at iteration t we have:

$$\begin{aligned}\|HMH\| &= \left\| \tau D^{-1/2} (L + \sum_{i \in V} {}^{d_i}/_{2m} \cdot \beta_i D + \sum_{i \in V} {}^{d_i}/_{2m} \cdot \beta_i e_i e_i^{\top}) D^{-1/2} \right\| \\ &\leq \left\| \tau D^{-1/2} (L + 2 \cdot 72 \cdot \gamma D) D^{-1/2} \right\| \leq O(\tau) = \mathrm{poly}(n).\end{aligned}$$

Moreover, it is easy to see that our argument is robust up to an error $\delta = {}^{1}/_{\mathrm{poly}(n)}$ in this computation and the sparsity of M is $O(m)$ so that the running time of a single matrix-exponential-vector product is $\tilde{O}(m)$. Given the embedding produced by Step 1, $L(K_V) \bullet X^{(t)}$ can be computed in time $\tilde{O}(nk) = \tilde{O}(n)$ by computing the distances $||v_i^{(t)} - v_{\mathsf{avg}}^{(t)}||^2$ for all $i \in V$. By Theorem 3.10, Step 3 runs in time $\tilde{O}(mk) = \tilde{O}(m)$.

Finally, both Steps 4 and 5 can be performed in time $\tilde{O}(m)$. As there are at most $O(\log n)$ iterations, the theorem follows. $\square$

Theorem 3.2 is proved similarly. It suffices to show that a single matrix-exponential-vector product requires time $\tilde{O}(m/\sqrt{\gamma})$.

Proof of Theorem 3.2. Using the algorithm of Theorem 2.5, we obtain that $\|A\| \leq O(\tau)$, so that $k = \tilde{O}(\sqrt{\tau}) = \tilde{O}(1/\sqrt{\gamma}) \leq \tilde{O}(n)$. Hence, the running time of a single computation for this method is $\tilde{O}(m\sqrt{\tau}) = \tilde{O}(m/\sqrt{\gamma})$. $\square$

3.6 Remaining Proofs

In this section we provide the remaining proofs from Section 3.5. We start with some preliminaries.

Preliminaries

Fact 3.13. *$L \preceq 2 \cdot D$ and $L(\mathsf{Star}_i) \preceq 2 \cdot D$.*

Fact 3.14. *For all $i \in V$, $L(\mathsf{Star}_i) = d_i/2m \cdot L(K_V) + d_i R_i$. In particular, $L(\mathsf{Star}_i) \succeq d_i R_i$.*

At iteration t of BalSep, we denote

$$C^{(t)} \stackrel{\text{def}}{=} D^{-1/2} Q(\beta^{(t)}) D^{1/2} = D^{-1/2}(L + \sum_{i \in V} \beta_i^{(t)} L(\mathsf{Star}_i)) D^{-1/2}.$$

The following are some useful facts about $C^{(t)}$ that we will require:

Fact 3.15. *The vector $D^{1/2}$ is the eigenvector of $C^{(t)}$ with smallest eigenvalue 0.*

Fact 3.16. *$C^{(t)} \preceq O(1) \cdot I$.*

Useful Lemmas

Lemma 3.17.

$$\Psi(P^{(t)}, V) = L(K_V) \bullet D^{-1} P_{2\tau}(\beta^{(t)}) = \mathit{Tr}\left(e^{-2\tau C^{(t)}}\right) - 1.$$

Proof. By definition, we have

$$\Psi(P^{(t)}, V) = L(K_V) \bullet D^{-1} P_{2\tau}(\beta^{(t)}) = L(K_V) \bullet D^{-1} e^{-2\tau Q(\beta^{(t)})} = L(K_V) \bullet D^{-1/2} e^{-2\tau C^{(t)}} D^{-1/2}.$$

Using Fact 3.3 and the cyclic property of the trace function, we obtain

$$L(K_V) \bullet D^{-1/2} e^{-2\tau C^{(t)}} D^{-1/2} = (I - {}^{1}\!/_{2m} D^{1/2} \mathbf{1}\mathbf{1} D^{1/2}) \bullet e^{-2\tau C^{(t)}}.$$

Finally, by Fact 3.15, we must have that the right-hand side equals $\mathrm{Tr}\left(e^{-2\tau C^{(t)}}\right) - 1$, as required. □

The following lemma is a simple consequence of the convexity of e^{-x}. It is proved in [Ore11].

Lemma 3.18. *For a symmetric matrix $A \in \mathbb{R}^{n \times n}$ such that $\rho I \succeq A \succeq 0$ and $\tau > 0$, we have*

$$e^{-\tau A} \preceq \left(I - \frac{(1 - e^{-\tau\rho})}{\rho} A\right).$$

The following are standard lemmas.

Lemma 3.19 (Golden-Thompson inequality [Bha96])**.** *Let $X, Y \in \mathbb{R}^{n \times n}$ be symmetric matrices. Then,*

$$\mathit{Tr}\left(\left(e^{X+Y}\right)\right) \leq \mathit{Tr}\left(\left(e^{X} e^{Y}\right)\right).$$

Lemma 3.20 (Johnson-Lindenstrauss Dimension Reduction)**.** *Given an embedding $\{v_i \in \mathbb{R}^d\}_{i \in V}$, $V = [n]$, let $u_1, u_2, \ldots, u_k$, be vectors sampled independently uniformly* *from the $n-1$-dimensional sphere of radius $\sqrt{n/k}$. Let U be the $k \times t$ matrix having the vector u_i as i^{th} row and let $\tilde{v}_i \stackrel{\text{def}}{=} U v_i$. Then, for $k \stackrel{\text{def}}{=} O(\log n/\varepsilon^2)$, for all $i, j \in V$,*

$$(1-\varepsilon) \cdot \left\| v_i - v_j \right\|^2 \leq \left\| \tilde{v}_i - \tilde{v}_j \right\|^2 \leq (1+\varepsilon) \cdot \left\| v_i - v_j \right\|^2 .$$

Proof of Lemma 3.8

Proof. Set $\beta \stackrel{\text{def}}{=} \beta^{(t)}$. By Lemma 3.17, we have $\operatorname{Tr}\left(e^{-2\tau C^{(t)}}\right) - 1 \leq 4/3n$. Hence, $\lambda_{n-1}(e^{-2\tau C^{(t)}}) \leq 4/3n$, which implies that, by taking logs,

$$\lambda_2(C^{(t)}) \geq \frac{\log n}{4\tau} \geq 3\gamma.$$

This can be rewritten in matrix terms, by Fact 3.3 and Fact 3.15:

$$L + \sum_{i \in V} \beta_i^{(t)} L(\mathsf{Star}_i) \succeq 3\gamma \cdot L(K_V). \tag{3.1}$$

which proves the first part of the Lemma.

For the second part, we start by noticing that, $\beta^{(t)} = 72\gamma/T \cdot \sum_{j=1}^{t} \sum_{i \in S^{(j)}} e_i$. Now for any b-balanced cut U, with $\mathsf{vol}(U) \leq \mathsf{vol}(\bar{U})$, consider the vector x_U defined as

$$(x_U)_i \stackrel{\text{def}}{=} \begin{cases} \sqrt{\frac{1}{2m} \cdot \frac{\mathsf{vol}(\bar{U})}{\mathsf{vol}(U)}} & \text{for } i \in U, \\ \\ -\sqrt{\frac{1}{2m} \cdot \frac{\mathsf{vol}(U)}{\mathsf{vol}(\bar{U})}} & \text{for } i \in \bar{U}. \end{cases}$$

Applying the guarantee of Equation 3.1, we obtain

$$x_U^\top L x_U + \frac{72\gamma}{T} \cdot \sum_{j=1}^{t} \sum_{i \in S^{(j)}} x_U^\top L(\mathsf{Star}_i) x_U \geq 3\gamma \cdot x_U^\top L(K_V) x_U.$$

Notice that

$$x_U^\top L x_U = \sum_{\{i,j\}\in E} ((x_U)_i - (x_U)_j)^2 = \frac{2m}{\mathsf{vol}(\bar{U})} \cdot \frac{\left|E(U,\bar{U})\right|}{\mathsf{vol}(U)} \leq 2 \cdot \phi(U),$$

$$x_U^\top L(\mathsf{Star}_i) x_U = \sum_{j\in V} \frac{d_j}{2m}((x_U)_i - (x_U)_j)^2) \leq \frac{2m}{\mathsf{vol}(U)} \cdot \frac{d_i}{2m} = \frac{d_i}{\mathsf{vol}(U)},$$

$$x_U^\top L(K_V) x_U = \sum_{i<j\in V} \frac{d_j d_i}{2m}((x_U)_i - (x_U)_j)^2) = 1.$$

Hence, our guarantee becomes

$$2\phi(U) + \frac{72\gamma}{T} \cdot \sum_{j=1}^{t} \frac{\mathsf{vol}(S^{(j)})}{\mathsf{vol}(U)} \geq 3\gamma.$$

Since for $j = 1, \dots, t$, we have $\mathsf{vol}(S^{(j)}) \leq {}^{b}\!/_{100} \cdot 2m \leq {}^{\mathsf{vol}(U)}\!/_{100}$, and $t \leq T$, we obtain $\phi(U) \geq \gamma$.

□

Proof of Lemma 3.9

Proof. Consider $L \bullet D^{-1} P_{2\tau}(\beta^{(t)}) = L \bullet D^{-1/2} e^{-2\tau C^{(t)}} D^{-1/2}$. Using the cyclic property of trace and the definition of $C^{(t)}$, we have that

$$L \bullet D^{-1} P_{2\tau}(\beta^{(t)}) \leq C^{(t)} \bullet e^{-2\tau C^{(t)}}.$$

We now consider the spectrum of $C^{(t)}$. By Fact 3.15, the smallest eigenvalue is 0. Let the remaining eigenvalues be $\lambda_2 \leq \lambda_3 \leq \cdots \leq \lambda_n$. Then, $C^{(t)} \bullet e^{-2\tau C^{(t)}} = \sum_{i=2}^{n} \lambda_i e^{-2\tau\lambda_i}$. We will analyze these eigenvalues in two groups. For the first group, we consider eigenvalues smaller than 24γ and use Lemma 3.17 to obtain,

$$\sum_{i:\lambda_i \leq 24\gamma} \lambda_i e^{-2\tau\lambda_i} \leq O(\gamma) \cdot \sum_{i=2}^{n} e^{-2\tau\lambda_i} \leq O(\gamma) \cdot (\mathrm{Tr}\left(e^{-2\tau C^{(t)}}\right) - 1)$$

$$\leq O(\gamma) \cdot L(K_V) \bullet D^{-1} P_{2\tau}(\beta^{(t)}).$$

For the remaining eigenvalues, by Fact 3.16 and Lemma 3.17, together with $\gamma \geq 1/m \geq 1/n^2$ and $\Psi(P^{(t)}, V) \geq \frac{1}{n}$, we have,

$$\sum_{i:\lambda_i \geq 24\gamma} \lambda_i e^{-2\tau\lambda_i} \leq O(1) \cdot n \cdot e^{-2\tau \cdot 24\gamma} \leq \frac{O(1)}{n^3}$$
$$\leq O(\gamma) \cdot \Psi(P^{(t)}, V) = O(\gamma) \cdot L(K_V) \bullet D^{-1} P_{2\tau}(\beta^{(t)}).$$

Combining these two parts, we have:

$$L \bullet D^{-1} P_{2\tau}(\beta^{(t)}) \leq C^{(t)} \bullet e^{-2\tau C^{(t)}} \leq O(1) \cdot \sum_{i:\lambda_i \leq 24\gamma} \lambda_i e^{-2\tau\lambda_i} \leq O(\gamma) \cdot L(K_V) \bullet D^{-1} P_{2\tau}(\beta^{(t)}).$$

Now, we apply the Johnson-Lindenstrauss Lemma (Lemma 3.20) to both sides of this inequality to obtain:

$$L \bullet X^{(t)} \leq O(\gamma) \cdot L(K_V) \bullet X^{(t)}.$$

□

Proof of Corollary 3.11

Proof. By Lemma 3.9 and Theorem 3.10, we have that $S^{(t)}$ w.h.p. is either c-balanced or $\sum_{i \in S^{(t)}} d_i R_i \bullet X^{(t)} \geq 2/3 \cdot L(K_V) \bullet X^{(t)}$. By Lemma 3.20 and as $1+\varepsilon / 1-\varepsilon \leq 4/3$, we have w.h.p.:

$$\Psi(P^{(t)}, S^{(t)}) = \sum_{i \in S^{(t)}} d_i R_i \bullet D^{-1} P_{2\tau}(\beta^{(t)}) \geq \frac{1}{1+\varepsilon} \cdot \Big(\sum_{i \in S^{(t)}} d_i R_i \bullet X^{(t)}\Big) \geq \frac{2}{3} \frac{1}{1+\varepsilon} \cdot L(K_V) \bullet X^{(t)}$$
$$\geq \frac{2}{3} \cdot \frac{1-\varepsilon}{1+\varepsilon} \cdot L(K_V) \bullet D^{-1} P_{2\tau}(\beta^{(t)}) = \frac{2}{3} \cdot \frac{1-\varepsilon}{1+\varepsilon} \cdot \Psi(P^{(t)}, V) \geq \frac{1}{2} \cdot \Psi(P^{(t)}, V).$$

□

Proof of Theorem 3.12

Proof. By Lemma 3.17 and the Golden-Thompson inequality in Lemma 3.19:

$$\Psi(P^{(t+1)},V) = \operatorname{Tr}\left(e^{-2\tau C^{(t+1)}}\right) - 1 \leq \operatorname{Tr}\left(e^{-2\tau C^{(t)}} e^{-2\tau D^{-1/2}(\frac{72\gamma}{T}\sum_{i\in S^{(t)}} L(\mathsf{Star}_i))D^{-1/2}}\right) - 1.$$

We now apply Lemma 3.18 to the second term under trace. To do this we notice that $\sum_{i\in S^{(t)}} L(\mathsf{Star}_i) \preceq 2L(K_V) \preceq 2D$, so that

$$D^{-1/2}\left(\frac{72\gamma}{T}\sum_{i\in S^{(t)}} L(\mathsf{Star}_i)\right)D^{-1/2} \preceq \frac{144\gamma}{T}I.$$

Hence, we obtain,

$$\Psi(P^{(t+1)},V) \leq \operatorname{Tr}\left(e^{-2\tau C^{(t)}}\left(I - (1-e^{-288\cdot\tau\gamma/T})\cdot\frac{1}{2}D^{-1/2}(\sum_{i\in S^{(t)}} L(\mathsf{Star}_i))D^{-1/2}\right)\right) - 1.$$

Applying the cyclic property of trace, we get,

$$\Psi(P^{(t+1)},V) \leq \Psi(P^{(t)},V) - \frac{(1-e^{-288\cdot\tau\gamma/T})}{2}\sum_{i\in S^{(t)}} L(\mathsf{Star}_i) \bullet D^{-1}P_{2\tau}(\beta^{(t)}).$$

Next, we use Fact 3.14 to replace $L(\mathsf{Star}_i)$ by R_i and notice that $288\cdot\tau\gamma/T = 2:$

$$\Psi(P^{(t+1)},V) \leq \Psi(P^{(t)},V) - \frac{(1-e^{-2})}{2}\sum_{i\in S^{(t)}} d_i R_i \bullet D^{-1}P_{2\tau}(\beta^{(t)}).$$

Then, we apply the definition of $\Psi(P^{(t)},S^{(t)}):$

$$\Psi(P^{(t+1)},V) \leq \Psi(P^{(t)},V) - 1/3\cdot\Psi(P^{(t)},S^{(t)}).$$

Finally, by Corollary 3.11, we know that w.h.p. $\Psi(P^{(t)},S^{(t)}) \geq 1/2\cdot\Psi(P^{(t)},V)$ and the required result follows. □

3.7 SDP Interpretation

In [OV11], the authors presented an algorithm that outputs either a $\Omega(b)$-balanced cut of conductance $O(\sqrt{\gamma})$ or a certificate that no b-balanced cut of conductance γ exists in time $\tilde{O}(m/\gamma^2)$. This algorithm uses the Matrix Multiplicative Weights update method of Arora and Kale [AK07] to approximately solve an SDP formulation of the Balanced Separator problem. The main technical contribution of their work is the routine FindCut (implicit in their Oracle procedure), which plays the role of an approximate separation oracle for their SDP. In each iteration of the algorithm, the Matrix Multiplicative Weights Update method produces a candidate SDP-solution $Y^{(t)}$. In one scenario, $Y^{(t)}$ does not have sufficiently low Laplacian objective value:

$$L \bullet Y^{(t)} \geq \Omega(\gamma) L(K_V) \bullet Y^{(t)}. \tag{3.2}$$

In this case, the algorithm uses Equation 3.2 to produce a candidate solution $Y^{(t+1)}$ with lower objective value. Otherwise, FindCut is run on the embedding corresponding to $Y^{(t)}$. By Theorem 3.10, this yields either a cut of the required balance or a dual certificate that $Y^{(t)}$ is infeasible. This certificate has the form

$$\gamma \cdot \sum_{i \in S^{(t)}} d_i R_i \bullet Y^{(t)} \geq \Omega(\gamma) L(K_V) \bullet Y^{(t)} \tag{3.3}$$

and is used by the update to construct the next candidate $Y^{(t+1)}$. The number of iterations necessary is determined by the width of the two possible updates described above. A simple calculation shows that the width of the update for Equation 3.2 is $\Theta(1)$, while for Equation 3.3, it is only $O(\gamma)$. Hence, the overall width is $\Theta(1)$, implying that $O(\log n/\gamma)$ iteration are necessary for the algorithm of OV to produce a dual certificate that the SDP is infeasible and therefore no b-balanced cut of conductance γ exists.

Our modification of the update is based on changing the starting candidate solutions from $Y^{(1)} \propto D^{-1}$ to $X^{(1)} \propto D^{-1/2} e^{-2\tau D^{-1/2} L D^{-1/2}} D^{-1/2}$. In Lemma 3.8 and Lemma 3.9, we show that this modification implies that all $X^{(t)}$ must now have $L \bullet X^{(t)} \leq O(\gamma) \cdot L(K_V) \bullet X^{(t)}$ or else we find a dual certificate that the SDP is infeasible. This additional guarantee effectively allows us to bypass the update of Equation 3.2 and only work with updates of the form given in Equation 3.3. As a result, our width is now $O(\gamma)$ and we only require $O(\log n)$ iterations.

Another way to interpret our result is that all possible $\tau \approx {}^{\log n}/_{\gamma}$ updates of the form of Equation 3.2 in the algorithm of OV are regrouped into a single step, which is performed at the beginning of the algorithm.

3.8 The FindCut Subroutine

In this section, we present the FindCut routine and prove its guarantees. Most of the following material appears in [OV11, Ore11], and is reproduced here in our notation for completeness. The constants in these proofs are not optimized.

Preliminary Lemmas

Fact 3.21. *For a subset $S \subseteq V$,*

$$\sum_{i \in \bar{S}} d_i R_i \succeq \frac{\mathsf{vol}(S)}{2m} \left(L(K_V) - L(K_S) \right).$$

Proof. By Fact 3.14,

$$\sum_{i \in \bar{S}} d_i R_i = \sum_{i \in \bar{S}} L(\mathsf{Star}_i) - \frac{\mathsf{vol}(\bar{S})}{2m} L(K_V).$$

Moreover, by the definitions of the graphs $\mathsf{Star}_i, K_V, K_S$, it is clear that

$$\sum_{i \in \bar{S}} L(\mathsf{Star}_i) + \frac{\mathsf{vol}(S)}{2m} L(K_S) \succeq L(K_V).$$

Combining these two equations, we obtain the required statement. $\square$

The following is a variant of the sweep cut argument of Cheeger's inequality [Chu97], tailored to ensure that a constant fraction of the variance of the embedding is contained inside the output cut.

Lemma 3.22. *Let $x \in \mathbb{R}^n, x \geq 0$, such that $x^\top L x \leq \lambda$ and $\mathsf{vol}(\mathrm{supp}(x)) \leq {}^{2m}/_{2}$. Relabel the vertices so that $x_1 \geq x_2 \geq \cdots \geq x_{z-1} > 0$ and $x_z = \cdots = x_n = 0$. For $i \in [z-1]$, denote by $S_i \subseteq V$, the sweep cut $\{1, 2, \ldots, i\}$. Further, assume that $\sum_{i=1}^n d_i x_i^2 \leq 1$, and, for some fixed $k \in [z-1]$, $\sum_{i=k}^n d_i x_i^2 \geq \sigma$. Then, there is a sweep cut S_h of x such that $z - 1 \geq h \geq k$ and $\phi(S_h) \leq {}^{1}/_{\sigma} \cdot \sqrt{2\lambda}$.*

We will also need the following simple fact.

Fact 3.23. *Given $u, v, w \in \mathbb{R}^n$, $(\|v - w\| - \|u - w\|)^2 \leq \|v - u\|^2$.*

3.8.1 Roundable Embeddings and Projections

The following definition of *roundable embedding* captures the case in which a vector embedding of the vertices V identifies a balanced cut of conductance close to α in G. Intuitively, in a roundable embedding, a constant fraction of the total variance is spread over a large set R of vertices.

Definition 3.24 (Roundable Embedding)**.** *Given an embedding $\{v_i\}_{i \in V}$ with Gram matrix X, denote by Ψ the total variance of the embedding: $\Psi \stackrel{\text{def}}{=} L(K_V) \bullet X$. Also, let $R = \{i \in V : \|v_i - v_{\mathsf{avg}}\|^2 \leq 32 \cdot {}^{(1-b)}/_{b} \cdot \frac{\Psi}{2m}\}$. For $\alpha > 0$, we say that $\{v_i\}_{i \in V}$ is roundable for (G, b, α) if:*

- $L \bullet X \leq \alpha\Psi$,
- $L(K_R) \bullet X \geq \frac{\Psi}{128}$.

A roundable embedding can be converted into a balanced cut of conductance $O(\sqrt{\alpha})$ by using a standard projection rounding, which is a simple extension of an argument already appearing in [ARV09] and [AK07]. The rounding procedure ProjRound is described in Figure 3.2 for completeness. It is analyzed in [OV11] and [Ore11], where the following theorem is proved.

Theorem 3.25 (Rounding Roundable Embeddings [OV11, Ore11]). *If $\{v_i \in \mathbb{R}^h\}_{i\in V}$ is roundable for (G, b, α), then* ProjRound$(\{v_i\}_{i\in V}, b)$ *produces a $\Omega(b)$- balanced cut of conductance $O(\sqrt{\alpha})$ with high probability in time $\tilde{O}(nh + m)$.*

```
1. Input: An embedding $\{v_i \in \mathbb{R}^h\}_{i\in V}$, $b \in (0, 1/2]$.
2. Let $c = \Omega(b) \leq b/100$ be a constant, fixed in the proof of Theorem 3.25 in [OV11].
3. For $t = 1, 2, \ldots, O(\log n)$:
    a. Pick a unit vector $u$ uniformly at random from $\mathbb{S}^{h-1}$ and let $x \in \mathbb{R}^n$ with $x_i \overset{\text{def}}{=} \sqrt{h} \cdot u^\top v_i$.
    b. Sort the vector $x$. Assume w.l.o.g. that $x_1 \geq x_2 \geq \cdots \geq x_n$. Define $S_i \overset{\text{def}}{=} \{j \in [n] : x_j \geq x_i\}$.
    c. Let $S^{(t)} \overset{\text{def}}{=} (S_i, \bar{S}_i)$ which minimizes $\phi(S_i)$ among sweep-cuts for which $\mathsf{vol}(S_i) \in [c \cdot 2m, (1-c) \cdot 2m]$.
4. Output: The cut $S^{(t)}$ of least conductance over all choices of $t$.
```

Figure 3.2: ProjRound subroutine

3.8.2 Description of FindCut

In this subsection we prove Theorem 3.10. The procedure FindCut is formally described in Figure 3.3.

1. **Input:** Instance graph G, balance b, conductance value α and embedding $\{v_i\}_{i\in V}$, with Gram matrix X.
2. Let $r_i = \|v_i - v_{\mathsf{avg}}\|$ for all $i \in V$. Denote $\Psi \stackrel{\text{def}}{=} L(K_V) \bullet X$ and define the set $R \stackrel{\text{def}}{=} \{i \in V : r_i^2 \leq 32 \cdot {}^{(1-b)}/_b \cdot {}^{\Psi}/_{2m}\}$.
3. Case 1: If $L \bullet X > \alpha\Psi$, output FAIL and terminate.
4. Case 2: If $L(K_R) \bullet X \geq {}^{\Psi}/_{128}$, the embedding $\{v_i\}_{i\in V}$ is roundable for (G, b, α). Run ProjRound, output the resulting cut and terminate.
5. Case 3: Relabel the vertices of V such that $r_1 \geq r_2 \geq \cdots \geq r_n$ and let $S_i = \{1, \ldots, i\}$ be the j^{th} sweep cut of r. Let z the smallest index such that $\mathsf{vol}(S_z) \geq {}^{b}/_{4} \cdot 2m$. Output the most balanced sweep cut C among $\{S_1, \ldots, S_{z-1}\}$, such that $\phi(C) \leq 40 \cdot \sqrt{\gamma}$.

Figure 3.3: FindCut subroutine

Proof. (of Theorem 3.10) By Markov's inequality, $\mathsf{vol}(\bar{R}) \leq {}^{b}/_{(32\cdot(1-b))} \cdot 2m \leq {}^{b}/_{16} \cdot 2m \leq {}^{1}/_{32} \cdot 2m$. By assumption, Case 1 cannot take place. If Case 2 holds, then the embedding is roundable: by Theorem 3.25, ProjCut outputs an $\Omega(b)$-balanced cut C with conductance $O(\sqrt{\alpha})$. If this is not the case, we are in Case 3.

We then have $L(K_R) \leq {}^{\Psi}/_{128}$ and, by Fact 3.21:

$$\sum_{i\in\bar{R}} d_i R_i \bullet X = \sum_{i\in\bar{R}} d_i r_i^2 \geq \frac{\mathsf{vol}(R)}{2m} \cdot \left(1 - \frac{1}{128}\right) \cdot \Psi$$
$$\geq \left(1 - \frac{1}{32}\right) \cdot \left(1 - \frac{1}{128}\right) \cdot \Psi \geq \left(1 - \frac{5}{128}\right) \cdot \Psi.$$

It must be the case that $\bar{R} = S_g$ for some $g \in [n]$, with $g \leq z$ as $\mathsf{vol}(S_g) \leq \mathsf{vol}(S_z)$. Let $k \leq z$ be the the vertex in $\overline{R}$ such that $\sum_{j=1}^{k} d_j r_j^2 \geq {}^{3}/_{4} \cdot (1 - {}^{5}/_{128})$ and $\sum_{j=k}^{g} d_j r_j^2 \geq {}^{1}/_{4} \cdot (1 - {}^{5}/_{128})$. By the definition of z, we have $k \leq g < z$ and $r_z^2 \leq {}^{4}/_{b} \cdot {}^{\Psi}/_{2m} \leq 8 \cdot {}^{(1-b)}/_{b} \cdot {}^{\Psi}/_{2m}$. Hence, we have $r_z \leq {}^{1}/_{2} \cdot r_i$, for all $i \geq g$. Define the vector x as $x_i \stackrel{\text{def}}{=} (r_i - r_z)$ for $i \in S_z$ and $r_i \stackrel{\text{def}}{=} 0$ for $i \notin S_z$. Notice that:

$$x^\top Lx = \sum_{\{i,j\}\in E} (x_i - x_j)^2 \leq \sum_{\{i,j\}\in E} (r_i - r_j)^2 \overset{\text{Fact 3.23}}{\leq} \sum_{\{i,j\}\in E} \|v_i - v_j\|^2 \leq \alpha\Psi.$$

Also, $x \geq 0$ and $\mathsf{vol}(\text{supp}(x)) \leq {}^b\!/\!_4 \cdot 2m \leq {}^{2m}\!/\!_2$, by the definition of z. Moreover,

$$\sum_{i=1}^{n} d_i x_i^2 = \sum_{i=1}^{z} d_i (r_i - r_z)^2 \leq \sum_{i=1}^{z} d_i r_i^2 \leq \Psi,$$

and

$$\sum_{i=k}^{n} d_i x_i^2 = \sum_{i=k}^{z} d_i (r_i - r_z)^2 \geq \sum_{i=k}^{g} d_i (r_i - {}^1\!/\!_2 \cdot r_i)^2$$
$$= {}^1\!/\!_4 \cdot \sum_{i=k}^{g} d_i r_i^2 \geq {}^1\!/\!_{16} \cdot (1 - {}^5\!/\!_{128}) \cdot \Psi \geq {}^1\!/\!_{20} \cdot \Psi.$$

Hence we can now apply Lemma 3.22 to the vector ${}^1\!/\!_{\sqrt{\Psi}} \cdot x$. This shows that there exists a sweep cut S_h with $z > h \geq k$, such that $\phi(S_h) \leq 40 \cdot \sqrt{\alpha}$. It also shows that C, as defined in Figure 3.3, must exist. Moreover, it must be the case that $S_k \subseteq S_h \subseteq C$. As $h \geq k$, we have,

$$\sum_{i\in C} d_i R_i \bullet X = \sum_{i\in C} d_i r_i^2 \geq \sum_{i=1}^{k} d_i r_i^2 \geq \frac{3}{4} \cdot \left(1 - \frac{5}{128}\right) \cdot \Psi \geq \frac{2}{3} \cdot \Psi = \frac{2}{3} \cdot L(K_V) \bullet X.$$

Finally, using the fact that $\{v_i\}_{i\in V}$ is embedded in d dimensions, we can compute $L \bullet \tilde{X}$ in time $O(dm)$. Moreover, $L(K_V) \bullet X$ can be computed in time $O(nd)$ by using the decomposition $L(K_V) \bullet X = \sum_{i\in V} d_i \|v_i - v_{\mathsf{avg}}\|^2$. By the same argument, we can compute $L(K_R) \bullet X$ in time $O(nd)$. The sweep cut over r takes time $\tilde{O}(m)$. And, by Theorem 3.25, ProjRound runs in time $\tilde{O}(md)$. Hence, the total running time is $\tilde{O}(md)$. □

Notes

The material presented in this chapter is based on the paper "Approximating the Exponential, the Lanczos Method, and an $\tilde{O}(m)$-Time Spectral Algorithm for Balanced Separator" [OSV12], joint with Lorenzo Orecchia and Nisheeth Vishnoi, that appeared at STOC 2012. A full version of the paper is available on arxiv [OSV11].

Chapter 4

Computing $\exp(-A)v$

Our algorithm for the Balanced Separator problem, presented in the last chapter, requires the ability to approximate matrix exponentials. The main result of this chapter is a reduction from approximating the matrix-exponential-vector product $\exp(-A)v$, for any positive semidefinite matrix A and any vector v to approximating polylogarithmic number of matrix-inverse-vector products $(I + cA)^{-1}u$, for some constant $c > 0$ and some vector u. As a warm up, we combine this reduction with the Spielman-Teng solver to obtain an algorithm for the case where the matrix A is symmetric and diagonally-dominant (SDD), such as the Laplacian of a graph, that runs in time roughly $\tilde{O}(m_A)$, where m_A is the number of non-zero entries of A. Finally, combining the SDD solver with the Sherman-Morrison formula, we show how to approximate $\exp(-A)v$ in near-linear time for the matrices A required by our Balanced Separator algorithm.

Our algorithms are based on the Lanczos method from numerical linear algebra, and low degree polynomial and rational approximations to the exponential.

4.1 Our Results

Before we state our results for this chapter, we state the basic definitions used in this chapter. Some of these definitions have been presented before, but are being reproduced here for completeness.

Definitions. We work with square $n \times n$ matrices over $\mathbb{R}$. For a symmetric PSD matrix M, we define the M-norm of a vector x as $\|x\|_M \stackrel{\text{def}}{=} (x^\top M x)^{1/2}$. Given an $n \times n$ symmetric PSD matrix M and a function $f : \mathbb{R} \mapsto \mathbb{R}$, we can define $f(M)$ as follows: Let $u_1, \ldots, u_n$ be eigenvectors of M with eigenvalues $\lambda_1 \geq \ldots \geq \lambda_n$. Define $f(M) \stackrel{\text{def}}{=} \sum_i f(\lambda_i) u_i u_i^\top$. Thus, abusing notation, we denote the matrix exponential by $\exp(-M)$, which can also be defined as $\sum_{i \geq 0} \frac{(-1)^i}{i!} M^i$. Let $\lambda_1(M)$ and $\lambda_n(M)$ denote the largest and the smallest eigenvalues of M respectively, and, let $\Lambda(M)$ denote the smallest interval containing the spectrum of M, *i.e.*, $[\lambda_n(M), \lambda_1(M)]$. $\|M\| \stackrel{\text{def}}{=} \sup_{\|x\|=1} \|Mx\|$ denotes the spectral norm of M.

M is said to be *Symmetric and Diagonally Dominant* (SDD) if, $M_{ij} = M_{ji}$, for all i, j and $M_{ii} \geq \sum_{j \neq i} |M_{ij}|$, for all i. M is called *Upper Hessenberg* if, $(M)_{ij} = 0$ for $i > j + 1$. M is called *tridiagonal* if $M_{ij} = 0$ for $i > j + 1$ and for $j > i + 1$. For a matrix M, let m_M denote the number of non-zero entries in M. Further, let t_M denote the time required to multiply the matrix M with a given vector w. In general t_M depends on how M is given as an input and can be $\Theta(n^2)$. However, it is possible to exploit the special structure of M if given as an input appropriately: It is possible to just multiply the non-zero entries of M, giving $t_M = O(m_M)$. Also, if M is a rank one matrix $ww^\top$, where w is known, we can multiply by M in $O(n)$ time.

For any positive integer k, let Σ_k denote the set of all polynomials with degree at most k. Given a degree k polynomial $p \stackrel{\text{def}}{=} \sum_{i=0}^{k} a_i \cdot x^i$, the ℓ_1 norm of p, denoted as $\|p\|_1$ is defined as $\|p\|_1 = \sum_{i \geq 0}^{k} |a_i|$.

4.1.1 Approximating $\exp(-A)v$

We first describe the well-known Lanczos method (*e.g.* see [Saa92]) in Section 4.2. We also give a proof of a well known theorem about the method that permits us to extend polynomial approximations for a function f over reals to approximating f over matrices (Theorem 4.10). For approximating matrix exponentials, the Lanczos method can be combined with good polynomial approximations to the exponential. However, as we will show in Chapter 6, polynomial approximations do not suffice for our application to the Balanced Separator problem.

Reduction to Matrix Inversion : Approximating $\exp(-A)v$ using Rational Approximations

We give a procedure called ExpRational that approximates matrix exponentials, given black box access to a procedure Invert_A with the following guarantee: given a vector y, a positive integer k and $\varepsilon_1 > 0$, $\mathsf{Invert}_A(y, k, \varepsilon_1)$ returns a vector u_1 such that, $\|(I + {}^{A}\!/_{k})^{-1}y - u_1\| \leq \varepsilon_1 \|y\|$. ExpRational is motivated by the Lanczos method, but it is based on good rational approximations for the exponential, instead of polynomial approximations.

The following theorem summarizes our result about the procedure ExpRational, stated informally in Theorem 2.2.

Theorem 4.1 (Running Time of ExpRational given Invert_A)**.** *Given a symmetric PSD matrix $A \succeq 0$, a vector v with $\|v\| = 1$, an error parameter $0 < \delta \leq 1$ and oracle access to Invert_A, for parameters $k \stackrel{\text{def}}{=} O(\log {}^{1}\!/_{\delta})$ and $\varepsilon_1 \stackrel{\text{def}}{=} \exp(-\Theta(k \log k + \log(2 + \|A\|)))$, ExpRational computes a vector u such that $\|\exp(-A)v - u\| \leq \delta$, in time $O(T^{inv}_{A,k,\varepsilon_1} \cdot k + n \cdot k^2 + k^3)$, where $T^{inv}_{A,k,\varepsilon_1}$ is the time required by $\mathsf{Invert}_A(\cdot, k, \varepsilon_1)$.*

The procedure ExpRational is described in Section 4.3. A proof of Theorem 4.1 appears in Section 4.5. The algorithms for Theorems 2.3 and 2.4 are obtained from ExpRational by giving different implementations of the required procedure Invert_A.

Exponentiating SDD matrices and general PSD matrices.

Using the Spielman-Teng SDD solver to implement the Invert_A procedure for ExpRational, we obtain a procedure for exponentiating SDD matrices. Theorem 2.3, restated below, then follows from Theorem 4.1 (see Section 4.3.1).

Theorem 4.2 (Theorem 2.3 Restated)**.** *Given an $n \times n$ symmetric matrix A which is SDD, a vector v and a parameter $\delta \leq 1$, there is an algorithm that can compute a vector u such that $\|\exp(-A)v - u\| \leq \delta \|v\|$ in time $\tilde{O}((m_A + n)\log(2 + \|A\|))$. The tilde hides poly$(\log n)$ and poly$(\log {}^1\!/\!_\delta)$ factors.*

Similarly, we prove Theorem 2.4 (restated below) by using the Conjugate Gradient method to implement the Invert_A procedure for general PSD matrices (see Section 4.3.2).

Theorem 4.3 (Theorem 2.4 Restated)**.** *Given an $n \times n$ symmetric PSD matrix A, a vector v and a parameter $\delta \leq 1$, there is an algorithm that can compute a vector u such that $\|\exp(-A)v - u\| \leq \delta \|v\|$ in time $\tilde{O}\left((t_A + n)\sqrt{1 + \|A\|}\log(2 + \|A\|)\right)$. Here the tilde hides poly$(\log n)$ and poly$(\log {}^1\!/\!_\delta)$ factors.*

Going beyond SDD Matrices

As mentioned in Section 2.2, our algorithm for Balanced Separator (Theorem 2.1) requires exponentiating matrices that may be neither SDD, nor sparse. Thus, Theorem 2.3 is insufficient for our application. However, they do have some additional structure. We prove the following theorem that is tailored for exponentiating matrices required for Theorem 2.1.

Theorem 4.4 (Matrix Exponential Computation Beyond SDD)**.** *Given an $n \times n$ symmetric matrix $A = \Pi H M H \Pi$ where M is SDD, H is a diagonal matrix with strictly positive entries and Π is a rank $(n-1)$ projection matrix $= 1 - ww^\top$ (w is explicitly known and $\|w\| = 1$), a vector v and a parameter $\delta \leq 1$, there is an algorithm that can compute a vector u such that $\|\exp(-A)v - u\| \leq \delta \|v\|$ in time $\tilde{O}((m_M + n)\log(2 + \|HMH\|))$. The tilde hides poly$(\log n)$ and poly$(\log {}^1\!/\!_\delta)$ factors.*

Recall from Section 3.4 that our algorithm for Balanced Separator requires us to compute $\exp(-A)v$ for a matrix A of the form $D^{-1/2}(L + \sum_i \beta_i L(\mathsf{Star}_i))D^{-1/2}$, where $\beta_i \geq 0$. We first note that if we let $\Pi \stackrel{\text{def}}{=} I - {}^1\!/\!_{2m} \cdot (D^{1/2}1)(D^{1/2}1)^\top$, the projection onto the space orthogonal to ${}^1\!/\!\sqrt{2m} \cdot D^{1/2}1$, then, for each i, $D^{-1/2}L(\mathsf{Star}_i)D^{-1/2} = \Pi({}^{d_i}\!/\!_{2m} \cdot I + e_i e_i^\top)\Pi$. Since $D^{1/2}1$ is an eigenvector of $D^{-1/2}LD^{-1/2}$, we have, $\Pi D^{-1/2}LD^{-1/2}\Pi = D^{-1/2}LD^{-1/2}$. Thus,

$$\begin{aligned} A &= \Pi D^{-1/2}LD^{-1/2}\Pi + \sum_i \beta_i \Pi({}^{d_i}\!/\!_{2m} \cdot I + e_i e_i^\top)\Pi \\ &= \Pi D^{-1/2}\left(L + \sum_i \beta_i \cdot {}^{d_i}\!/\!_{2m} \cdot D + \sum_i \beta_i d_i \cdot e_i e_i^\top\right) D^{-1/2}\Pi. \end{aligned}$$

This is of the form $\Pi H M H \Pi$, where $H \stackrel{\text{def}}{=} D^{-1/2}$ is diagonal and M is SDD. It is worth noting that since A itself may be neither sparse nor SDD, we cannot apply the Spielman-Teng SDD solver to approximate $(I + \alpha A)^{-1}$.

For the above theorem, we combine the SDD solver with the Sherman-Morrison formula (for matrix inverse with rank 1 updates) to implement the Invert_A procedure for ExpRational. The details appear in Section 4.4. Note that in order to obtain a version of Theorem 2.5 for such matrices, we do not have to do anything additional since multiplication by H and Π take $O(n)$ steps and hence, t_A is still $O(m_M + n)$. Finally, note that in our application, $\|HMH\|$ is poly(n).

Approximation Using Our Polynomial Approximation to e^{-x}

More straightforwardly, combining the Lanczos method with the polynomial approximation to e^{-x} that we prove in Theorem 6.1 (a more precise version of Theorem 2.7), we obtain the following theorem.

Theorem 4.5 (Running Time Using Lanczos)**.** *Given a symmetric PSD matrix $A \preceq 0$, a vector v with $\|v\| = 1$ and a parameter $0 < \delta \leq 1$, for k that is*

$$O\left(\sqrt{\max\{\log 1/\delta, \lambda_1(A) - \lambda_n(A)\}} \cdot (\log 1/\delta)^{3/2} \cdot \log\log 1/\delta\right),$$

and $f(x) = e^{-x}$, the procedure Lanczos *computes a vector u such that $\|\exp(-A)v - u\| \leq \|\exp(-A)\|\,\delta$. The time taken by* Lanczos *is $O\left((n + t_A)k + k^2\right)$.*

This result is comparable to, and implied by an earlier independent work of Hochbruck and Lubich (Theorem 2 in [HL97]). This gives us our second method for approximating $\exp(-A)v$ for general PSD matrices. Note that this algorithm avoids any inverse computation and, as a result, the procedure and the proofs are simpler and the algorithm practical.

Remark 4.6. *Note the k^3 term in the running time for Theorem 4.1 and the k^2 term in the running time for Theorem 4.5. This is the time required for computing the eigendecomposition of a $(k+1) \times (k+1)$ symmetric matrix. While this process requires $O(k^3)$ time in general, as in Theorem 4.1; in case of Theorem 4.5, the matrix is tridiagonal and hence the time required is $O(k^2)$ (see [PC99]).*

4.2 Lanczos Method – From Scalars to Matrices

Suppose we are given an $n \times n$ symmetric matrix B, a function $f : \mathbb{R} \to \mathbb{R}$, and we are looking to compute $f(B)v$. Since exact computation of $f(B)$ usually requires diagonalization of B, which could take as much as $O(n^3)$ time (see [PC99]), we seek an approximation to $f(B)v$. The Lanczos method allows us to do exactly that: It looks for an approximation to $f(B)v$ of the form $p(B)v$, where p is a polynomial of small degree, say k. Without loss of generality, we assume throughout that $\|v\| = 1$.

Before we describe how, we note that it computes this approximation in roughly $O((t_B + n)k)$ time plus the time it takes to compute $f(\cdot)$ on a $(k+1) \times (k+1)$ tridiagonal matrix, which can often be upper bounded by $O(k^2)$ (see [PC99]). Hence, the time is reduced to $O((t_B+n)k+k^2)$. What one has lost in this process is accuracy: The candidate vector u output by the Lanczos method, is now only an approximation to $f(B)v$. The quality of approximation, or $\|f(B)v - u\|$, can be upper bounded by the *uniform error* of the best degree k polynomial approximating f in the interval $[\lambda_n(B), \lambda_1(B)]$. Roughly, $\|f(B)v - u\| \approx (\min_{p_k \in \Sigma_k} \sup_{x \in [\lambda_1(B), \lambda_n(B)]} |f(x) - p_k(x)|)$. Surprisingly, one does not need to know the best polynomial and proving *existence* of good polynomials is sufficient. By increasing k, one can reduce this error and, indeed, if one lets $k = n$, there is no error. Thus, the task is reduced to proving existence of low degree polynomials that approximate f to within the desired error.

We give a description of the Lanczos method (*e.g.* see [Saa92]) in Figure 4.1 and give a proof of a well-known theorem about the approximation guarantee and the running time of the method (Theorem 4.10). We then show how to deduce Theorem 4.5 (Simple algorithm for exponentiating PSD matrices) using Theorem 4.10.

4.2.1 Computing the best polynomial approximation

Now, we describe the Lanczos method in detail, and how it achieves the error guarantee described above. Notice that $p(B)v$ lies in the subspace $\mathsf{Span}\{v, Bv, \ldots, B^k v\}$ – called the *Krylov subspace*. We recall the definition of a Krylov subspace.

Definition 4.7 (Krylov Subspace). *Given a matrix B and a vector v, the Krylov subspace of order k, denoted by $\mathcal{K}(B, v, k)$, is defined as the subspace spanned by the vectors $v, Bv, \ldots, B^k v$.*

It will be convenient to work with an orthonormal basis for $\mathcal{K} \stackrel{\text{def}}{=} \mathcal{K}(B, v, k)$. Let $\{v_i\}_{i=0}^k$ be an orthonormal basis for $\mathcal{K}$. Let V_k be the $n \times (k+1)$ matrix with $\{v_i\}_{i=0}^k$ as its columns. Thus, $V_k^\top V_k = I_{k+1}$ and $V_k V_k^\top$ denotes the projection onto the Krylov subspace $\mathcal{K}$. Let T_k be the $(k+1) \times (k+1)$ matrix expressing B as an operator restricted to $\mathcal{K}$ in the basis $\{v_i\}_{i=0}^k$, *i.e.*, $T_k \stackrel{\text{def}}{=} V_k^\top B V_k$. Note that this is not just a change of basis, since vectors in $\mathcal{K}$ can be mapped by B to vectors outside $\mathcal{K}$. Now, since $v, Bv \in \mathcal{K}$, we have,

$$Bv = (V_k V_k^\top) B (V_k V_k^\top) v = V_k (V_k^\top B V_k) V_k^\top v = V_k T_k V_k^\top v.$$

Extending this argument, the following lemma shows that for all $i \leq k$, $B^i v = V_k T_k^i V_k^\top v$, and hence, by linearity, $p(B)v = V_k p(T_k) V_k^\top v$, for any polynomial p of degree at most k.

Lemma 4.8. (Exact Computation with Polynomials. See *e.g.* [Saa92]). *Let V_k and T_k be as defined above. For any polynomial p of degree at most k,*

$$p(B)v = V_k p(T_k) V_k^\top v.$$

Proof. Recall that $V_k V_k^\top$ is the orthogonal projection onto the subspace $\mathcal{K}(B, v, k)$. By linearity, it suffices to prove this when p is x^t for $t \leq k$. This is true for $t = 0$ since $V_k V_k^\top v = v$. For any $j \leq k$, $B^j v$ lies in $\mathcal{K}(B, v, k)$, thus, $\forall\ j \leq k$, $V_k V_k^\top B^j v = B^j v$. Hence,

$$\begin{aligned} B^t v &= (V_k V_k^\top) B (V_k V_k^\top) B \cdots B (V_k V_k^\top) v \\ &= V_k (V_k^\top B V_k)(V_k^\top B V_k) \cdots (V_k^\top B V_k) V_k^\top v = V_k T_k^t V_k^\top v. \end{aligned}$$

□

Thus, T_k can be used to compute $p(B)v$ exactly for any degree k polynomial p. This lemma suggests that a natural candidate for approximating $f(B)v$ is the vector $V_k f(T_k) V_k^\top v$, even when f is not a degree k polynomial. To determine the quality of this approximation, we will bound the norm of the error, *i.e.*, $\left\| f(B)v - V_k f(T_k) V_k^\top v \right\|$.

Writing $r_k(x) \stackrel{\text{def}}{=} f(x) - p_k(x)$, where p_k is any degree k approximation to $f(x)$, and using Lemma 4.8, we get that the error in the approximation is,

$$f(B)v - V_k f(T_k) V_k^\top v = r_k(B)v - V_k r_k(T_k) V_k^\top v,$$

for any choice of p_k. Hence, the norm of the error is at most $(\|r_k(B)\| + \|r_k(T_k)\|)\,\|v\|$, which is bounded by the value of r_k on the eigenvalues of B and T_k. To obtain the best bound, we can minimize this error bound over polynomials p_k of degree at most k. This is summarized in the following lemma.

Lemma 4.9. (Approximation by Best Polynomial. See *e.g.* [Saa92]). *Let V_k and T_k be as defined above. Let $f : \mathbb{R} \to \mathbb{R}$ be any function such that $f(B)$ and $f(T_k)$ are well-defined. Then,*

$$\left\| f(B)v - V_k f(T_k) V_k^\top v \right\| \le \min_{p_k \in \Sigma_k} \left(\max_{\lambda \in \Lambda(B)} |f(\lambda) - p_k(\lambda)| + \max_{\lambda \in \Lambda(T_k)} |f(\lambda) - p_k(\lambda)| \right).$$

Proof. Let p_k be any degree k polynomial. Let $r_k \stackrel{\text{def}}{=} f - p_k$. Then,

$$\begin{aligned} \left\| f(B)v - V_k f(T_k) V_k^\top v \right\| &\le \left\| p_k(B)v - V_k p_k(T_k) V_k^\top v \right\| + \left\| r_k(B)v - V_k r_k(T_k) V_k^\top v \right\| \\ &\le 0 + \|r_k(B)\| + \left\| V_k r_k(T_k) V_k^\top \right\| \qquad \text{(Using Lemma 4.8)} \\ &= \max_{\lambda \in \Lambda(B)} |r_k(\lambda)| + \max_{\lambda \in \Lambda(T_k)} |r_k(\lambda)|. \end{aligned}$$

Minimizing over p_k gives us our lemma. □

Thus, $V_k f(T_k) V_k^\top v$ approximates $f(B)v$ as well as the *best* degree k polynomial that uniformly approximates f. Observe that in order to compute this approximation, we do not need to know the polynomial explicitly. It suffices to prove that there exists a degree k polynomial that uniformly approximates f well on an interval containing the spectrum of B and T_k (For exact computation, $\Lambda(T_k) \subseteq \Lambda(B)$). It remains to show how to compute V_k and T_k. We address this next.

4.2.2 Efficiently Computing a Basis for the Krylov Subspace

In this section, we show how to compute V_k and T_k efficiently. This is done iteratively as follows: Let $v_0 \stackrel{\text{def}}{=} v$. For $i = 0, \ldots, k-1$, we compute Bv_i and remove the components along the vectors $\{v_0, \ldots, v_i\}$ to obtain a new vector that is orthogonal to $v_0, \ldots, v_i$. This vector, scaled to norm 1, is defined to be v_{i+1} (similar to Gram-Schmidt orthonormalization). By construction, these vectors satisfy, for all $i \leq k$, $\mathsf{Span}\{v_0, \ldots, v_i\} = \mathsf{Span}\{v, Bv, \ldots, B^i v\}$.

Since $T_k = V_k^\top B V_k$, we have $(T_k)_{ij} = v_i^\top B v_j$. By construction, $Bv_j \in \mathsf{Span}\{v_0, \ldots, v_{j+1}\}$, and if $i > j+1$, v_i is orthogonal to this subspace, and hence $v_i^\top (Bv_j) = 0$. Thus, T_k is Upper Hessenberg, *i.e.*, $(T_k)_{ij} = 0$ for $i > j+1$. Moreover, if B is symmetric, $v_j^\top (Bv_i) = v_i^\top (Bv_j)$, and hence T_k is symmetric and tridiagonal. This implies that we need to orthogonalize Bv_i only w.r.t v_i and v_{i-1}. Thus, we need to compute only $O(k)$ dot-products while computing the basis, instead of $O(k^2)$; and time required for computing the dot-products is $O(nk)$, instead of $O(nk^2)$. This is crucial for the proof of Theorem 2.5 and Theorem 3.2.

We summarize the procedure below. A formal description of the Lanczos algorithm is given in Figure 4.1.

1. Compute the basis $\{v_i\}_{i=0}^k$ – Start with $v_0 \stackrel{\text{def}}{=} v$. For $i = 0, \ldots, k-1$, compute Bv_i and orthogonalize it to v_i and v_{i-1}. Scale the vector to unit norm to get v_{i+1}.

2. Construct the matrices V_k, T_k, and return the vector $V_k f(T_k) V_k^\top v$ as the candidate approximation to $f(B)v$.

The following theorem summarizes the main result about this procedure.

Theorem 4.10. (Lanczos Theorem. See *e.g.* [Saa92]) *Given a symmetric PSD matrix B, a vector v with $\|v\| = 1$, a function f, and a positive integer parameter k as inputs, the procedure* Lanczos *computes a vector u such that,*

$$\|f(B)v - u\| \leq 2 \cdot \min_{p_k \in \Sigma_k} \max_{\lambda \in \Lambda(B)} |f(\lambda) - p_k(\lambda)| \, .$$

The time taken by Lanczos *is $O\left((n + t_B)k + k^2\right)$.*

Proof. The algorithm Lanczos implements the Lanczos method we've discussed here. The error guarantee follows from Lemma 4.9 and the fact that $\Lambda(T_k) \subseteq \Lambda(B)$. The total running time is dominated by k multiplications of B with a vector, $O(k)$ dot-products and the eigendecomposition of the tridiagonal matrix T_k to compute $f(T_k)$ (which can be done in $O(k^2)$ time [PC99]), giving a total running time of $O\left((n + t_B)k + k^2\right)$. □

Combining the approximation guarantee of the Lanczos algorithm given by Theorem 4.10 for the setting $B \stackrel{\text{def}}{=} A$ and $f(x) \stackrel{\text{def}}{=} e^{-x}$, along with the polynomial approximation to e^{-x} that we prove in Theorem 6.1 (a more precise version of Theorem 2.7), we obtain a proof of Theorem 4.5, which is easily seen to imply Theorem 2.5.

This completes our description of the Lanczos method, and how to deduce Theorem 2.5 using the method.

Input: A symmetric matrix $B \succeq 0$, a vector v such that $\|v\| = 1$, a positive integer k, and a function $f : \mathbb{R} \to \mathbb{R}$.
Output: A vector u that is an approximation to $f(B)v$.

```
1. Initialize $v_0 \stackrel{\text{def}}{=} v$.
2. For $i = 0$ to $k-1$,    (Construct an orthonormal basis to Krylov subspace of order $k$)
    a. If $i = 0$, compute $w_0 \stackrel{\text{def}}{=} Bv_0$. Else, compute $w_i \stackrel{\text{def}}{=} Bv_i - \beta_i v_{i-1}$.
    b. Define $\alpha_i \stackrel{\text{def}}{=} v_i^\top w_i$ and $w'_i \stackrel{\text{def}}{=} w_i - \alpha_i v_i$ *.
    c. Define $\beta_{i+1} \stackrel{\text{def}}{=} \|w'_i\|$ and $v_{i+1} \stackrel{\text{def}}{=} w'_i/\beta_{i+1}$.
3. Let $V_k$ be the $n \times (k+1)$ matrix whose columns are $v_0, \ldots, v_k$ respectively.
4. Let $T_k$ be the $(k+1) \times (k+1)$ matrix such that for all $i$, $(T_k)_{ii} = v_i^\top B v_i = \alpha_i$, $(T_k)_{i,i+1} = (T_k)_{i+1,i} = v_{i+1}^\top B v_i = \beta_{i+1}$ and all other entries are 0.
5. Compute $\mathcal{B} \stackrel{\text{def}}{=} f(T_k)$ exactly via eigendecomposition. Output the vector $V_k \mathcal{B} V_k^\top v$.
* If $w'_i = 0$, compute the approximation with the matrices $T_{i-1}$ and $V_{i-1}$, instead of $T_k$ and $V_k$. The error bounds still hold.
```

Figure 4.1: The LANCZOS algorithm for approximating $f(B)v$

4.3 Approximating $\exp(-A)v$ Using a Rational Approximation to e^{-x}

In this section, we modify the Lanczos method to obtain an algorithm, which we call EXPRATIONAL, that underlies Theorem 4.1 and give a proof of the theorem. Next, we describe the respective Invert_A procedures required to prove Theorem 4.2 (Exponentiating SDD matrices), Theorem 4.3 (Exponentiating PSD matrices) and Theorem 4.4 (Matrix exponentials required for BALSEP). Our starting point is the following, rather surprising result by Saff, Schönhage, and Varga (SSV) [SSV75].

Theorem 4.11. (Rational Approximation [SSV75]). *There exists constants $c_1 \geq 1$ and k_0 such that, for any positive integer $k \geq k_0$, there exists a polynomial $p_k^\star(x)$ of*

Input: A Matrix $A \succeq 0$, a vector v such that $\|v\| = 1$, and an approximation parameter δ.
Output: A vector u such that $\|\exp(-A)v - u\| \leq \delta$.
Parameters: Let $k \stackrel{\text{def}}{=} O(\log {}^1\!/\!_\delta)$ and $\varepsilon_1 \stackrel{\text{def}}{=} \exp(-\Theta(k \log k + \log(1 + \|A\|)))$.

```
1. Initialize $v_0 \stackrel{\text{def}}{=} v$.
2. For $i = 0$ to $k - 1$,    (Construct an orthonormal basis to Krylov subspace of order $k$ )
    a. Call the procedure $\mathsf{Invert}_A(v_i, k, \varepsilon_1)$. The procedure returns a vector $w_i$,
       such that, $\|(I + {}^A\!/\!_k)^{-1} v_i - w_i\| \leq \varepsilon_1 \|v_i\|$.    (Approximate $(I + {}^A\!/\!_k)^{-1} v_i$)
    b. For $j = 0, \ldots, i$,
        i. Let $\alpha_{j,i} \stackrel{\text{def}}{=} v_j^\top w_i$.    (Compute projection onto $w_i$)
    c. Define $w_i' \stackrel{\text{def}}{=} w_i - \sum_{j=0}^{i} \alpha_{j,i} v_j$.    (Orthogonalize w.r.t. $v_j$ for $j \leq i$)
    d. Let $\alpha_{i+1,i} \stackrel{\text{def}}{=} \|w_i'\|$ * and $v_{i+1} \stackrel{\text{def}}{=} w_i' / \alpha_{i+1,i}$.    (Scaling it to norm 1)
    e. For $j = i + 2, \ldots, k$,
        i. Let $\alpha_{j,i} \stackrel{\text{def}}{=} 0$.
3. Let $V_k$ be the $n \times (k + 1)$ matrix whose columns are $v_0, \ldots, v_k$ respectively.
4. Let $T_k$ be the $(k + 1) \times (k + 1)$ matrix $(\alpha_{i,j})_{i,j \in \{0,\ldots,k\}}$ and $\widehat{T}_k \stackrel{\text{def}}{=} {}^1\!/\!_2(T_k^\top + T_k)$.
   (Symmetrize $T_k$)
5. Compute $\mathcal{B} \stackrel{\text{def}}{=} \exp\left(k \cdot (I - \widehat{T}_k^{-1})\right)$ exactly and output the vector $V_k \mathcal{B} V_k^\top v$.
* If $w_i' = 0$, compute the approximation the matrices $T_{i-1}$ and $V_{i-1}$, instead of $T_k$
  and $V_k$. The error bounds still hold.
```

Figure 4.2: The ExpRational algorithm for approximating $\exp(-A)v$

degree k such that $p_k^\star(0) = 0$, and,

$$\sup_{t\in(0,1]} \left|e^{-k/t+k} - p_k^\star(t)\right| = \sup_{x\in[0,\infty)} \left|e^{-x} - p_k^\star\left((1+{}^{x}\!/\!{}_{k})^{-1}\right)\right| \le c_1 k \cdot 2^{-k} \ .$$

This result implies that for any positive integer k, there exists a degree k polynomial $p_k^\star$ such that, $p_k^\star((1+{}^{x}\!/\!{}_{k})^{-1})$ approximates e^{-x} up to an error of $O(k\cdot 2^{-k})$ over the interval $[0,\infty)$. This suggests we can approximate $\exp(-A)v$, using the Lanczos method with $B \stackrel{\text{def}}{=} (I + {}^{A}\!/\!{}_{k})^{-1}$ and $f(x) \stackrel{\text{def}}{=} e^{k(1-{}^{1}\!/\!{}_{x})}$. Essentially, this was the method suggested by Eshof and Hochbruck [EH05]. The strong approximation guarantee of the SSV result along with the guarantee of the Lanczos method from the previous section, would imply that the order of the Krylov subspace for B required would be roughly $\log {}^{1}\!/\!{}_{\delta}$, and hence, independent of $\|A\|$. The running time is then dominated by the computation $Bv = (I + {}^{A}\!/\!{}_{k})^{-1}v$.

Eshof and Hochbruck note that the computation of exact matrix inverse is a costly operation ($O(n^3)$ time in general) and all known faster methods for inverse computation incur some error. They suggest using the Lanczos method with faster iterative methods, e.g. Conjugate Gradient, for computing the inverse (or rather the product of the inverse with a given vector) as a *heuristic*. They make no attempt to give a theoretical justification of why approximate computation suffices. Also note that, even if the computation was error-free, a method such as Conjugate Gradient will have running time which varies with $\sqrt{{}^{\lambda_1(A)}\!/\!{}_{\lambda_n(A)}}$ in general.

We abstract out the required approximate inversion procedure as Invert_A and require the following guarantee: given a vector y, a positive integer k, and an $\varepsilon_1 > 0$, $\mathsf{Invert}_A(y, k, \varepsilon_1)$ returns a vector u_1 such that, $\|(I + {}^{A}\!/\!{}_{k})^{-1}y - u_1\| \le \varepsilon_1 \|y\|$. We will use Invert_A to approximately multiply a given vector with $(I + {}^{A}\!/\!{}_{k})^{-1}$ during each iteration of the Lanczos algorithm. To be able to prove Theorem 2.3 using the SSV guarantee, we have to adapt the Lanczos method in several ways, and hence, deviate from the method suggested by Eshof and Hochbruck.

1. Eshof and Hochbruck construct T_k as a tridiagonal matrix as Lanczos method suggests, but since the computation is no longer exact, the basis $\{v_i\}_{i=0}^k$ is no longer guaranteed to be orthonormal. As a result, the proofs of the Lanczos method break down. Our algorithm, instead, builds an orthonormal basis, which means that at every step, we orthonormalize $w_i \approx (I + {}^A\!/_k)^{-1} v_i$ w.r.t all vectors $v_0, \ldots, v_i$. Thus, T_k becomes an Upper Hessenberg matrix instead of tridiagonal and we need to compute k^2 dot products in order to compute T_k.

2. With T_k being asymmetric, several nice spectral properties are lost, *e.g.* real eigenvalues and an orthogonal set of eigenvectors. We overcome this fact by symmetrizing T_k to construct $\widehat{T}_k = \frac{T_k + T_k^\top}{2}$ and computing our approximation with $\widehat{T}_k$. This permits us to bound the quality of a polynomial approximation applied to $\widehat{T}_k$ by the behavior of the polynomial on the eigenvalues of $\widehat{T}_k$.

3. Our analysis is based on the SSV approximation result, which is better than the variant proved and used by Eshof and Hochbruck . Moreover, for their *shifting* technique, which is the source of the $\|\exp(-A)\|$ factor in the hypothesis, the given proof in [EH05] is incorrect and it is not clear if the given bound could be achieved even under exact computation[1].

4. Most importantly, for our application, we will be able to employ the Spielman-Teng solver (Theorem 4.13) to approximate $(I + {}^A\!/_k)^{-1} v$ in time essentially independent of the norm of A.

[1]They show the existence of degree k polynomials in $(1 + \nu x)^{-1}$ for any *constant* $\nu \in (0, 1)$, that approximate e^{-x} up to an error of $\exp({}^1\!/_{2\nu} - \Theta(\sqrt{k(\nu^{-1} - 1)}))$. In order to deduce the claimed hypothesis, it needs to be used for $\nu \approx {}^1\!/_{\lambda_n(A)}$, in which case, there is a factor of $e^{\lambda_n(A)}$ in the error, which could be huge.

We now summarize our procedure below. A formal description of the ExpRational procedure is given in Figure 4.2.

1. Compute the basis $\{v_i\}_{i=0}^k$ – Start with $v_0 \stackrel{\mathrm{def}}{=} v$. For $i = 0, \ldots, k-1$,
   (a) Use Invert_A to obtain w_i, an approximation to $(I + A/k)^{-1} v_i$.
   (b) Orthogonalize w_i to $v_0, \ldots, v_i$. Scale the vector to unit norm to get v_{i+1}.
2. Construct the matrices V_k, T_k. Compute $\widehat{T}_k$ and return the vector $V_k f(\widehat{T}_k) V_k^\top v$ as the candidate approximation to $f(B)v$.

We will prove Theorem 4.1 that summarizes our main result about this procedure.

Remark 4.12. *Note that there is an $n \cdot k^2$ term in the running time for* ExpRational*, in contrast with $n \cdot k$ in the running time for* Lanczos*. This is because, in* ExpRational*, we can no longer guarantee that the matrix T_k is tridiagonal, in contrast with* Lanczos*. For* ExpRational*, k is small ($O(\log n)$ for our application) and hence the term $n \cdot k^2$ does not hurt. Whereas, for* Lanczos*, k is large ($\tilde{O}(1/\sqrt{\gamma})$ for our application), and a term of $n \cdot k^2$ in the running time would be prohibitive.*

The proof of Theorem 4.1 is quite technical. The main issue is that the error in approximating the matrix inverse at each iteration, propagates to later steps. We need to bound the error introduced in the output vector because of the error at each iteration. We present a proof at the end of this chapter in Section 4.5.

We now give different implementations of Invert_A in order to prove Theorems 4.2, 4.3 and 4.4 in Sections 4.3.1, 4.3.2 and 4.4, respectively.

4.3.1 SDD Matrices – Proof of Theorem 4.2

For Theorem 4.2 about exponentiating SDD matrices, we implement the Invert_A procedure in ExpRational using the Spielman-Teng SDD solver [ST04, ST06]. Here, we state an improvement on the Spielman-Teng result by Koutis, Miller and Peng [KMP11].

Theorem 4.13. (SDD Solver [KMP11]). *Given a system of linear equations $Mx = b$, where the matrix M is SDD, and an error parameter $\epsilon > 0$, it is possible to obtain a vector u that is an approximate solution to the system, in the sense that*

$$\|M^{-1}b - u\|_M \leq \epsilon \|M^{-1}b\|_M \ .$$

The time required for computing u is $\tilde{O}\left(m_M \log n \log {}^1\!/\!_\epsilon\right)$, where M is an $n \times n$ matrix. (The tilde hides $\log \log n$ factors.)

We now give a proof of Theorem 4.2.

Proof. (of Theorem 4.2) We use the ExpRational procedure to approximate the exponential. We only need to describe how to implement the Invert_A procedure for an SDD matrix A. Recall that the procedure Invert_A, given a vector y, a positive integer k and real parameter $\varepsilon_1 > 0$, is supposed to return a vector u_1 such that $\|(I + {}^A\!/\!_k)^{-1}y - u_1\| \leq \varepsilon_1 \|y\|$, in time $T^{\mathrm{inv}}_{A,k,\varepsilon_1}$. Also, observe that this is equivalent to approximately solving the linear system $(I + {}^A\!/\!_k)z = y$ for the vector z.

If the matrix A is SDD, $(I + {}^A\!/\!_k)$ is also SDD, and hence, we can use the Spielman-Teng SDD solver to implement Invert_A. We use Theorem 4.13 with inputs $(I + {}^A\!/\!_k)$, the vector y and error parameter ε_1. It returns a vector u_1 such that,

$$\|(I + {}^A\!/\!_k)^{-1}y - u_1\|_{(I+{}^A\!/\!_k)} \leq \varepsilon_1 \|(I + {}^A\!/\!_k)^{-1}y\|_{(I+{}^A\!/\!_k)} \ .$$

This implies that,

$$\begin{aligned} \|(I + {}^A\!/\!_k)^{-1}y - u_1\|^2 &= ((I + {}^A\!/\!_k)^{-1}y - u_1)^\top ((I + {}^A\!/\!_k)^{-1}y - u_1) \\ &\leq ((I + {}^A\!/\!_k)^{-1}y - u_1)^\top (I + {}^A\!/\!_k)((I + {}^A\!/\!_k)^{-1}y - u_1) \end{aligned}$$

$$\begin{aligned}&\leq \left\|(I + {}^{A}\!/\!_{k})^{-1}y - u_1\right\|^2_{(I+{}^{A}\!/\!_{k})} \leq \varepsilon_1^2 \cdot \|(I + {}^{A}\!/\!_{k})^{-1}y\|^2_{(I+{}^{A}\!/\!_{k})} \\ &= \varepsilon_1^2 \cdot y^\top (I + {}^{A}\!/\!_{k})^{-1} y \leq \varepsilon_1^2 \cdot y^\top y \ ,\end{aligned}$$

which gives us $\|(I + {}^{A}\!/\!_{k})^{-1}y - u_1\| \leq \varepsilon_1 \|y\|$, as required for Invert_A. Thus, Theorem 4.1 implies that the procedure ExpRational computes a vector u approximating $e^{-A}v$, as desired.

The time required for the computation of u_1 is $T^{\text{inv}}_{A,k,\varepsilon_1} = \tilde{O}\left((m_A + n)\log n \log {}^{1}\!/\!_{\varepsilon_1}\right)$, and hence from Theorem 4.1, the total running time is

$$\tilde{O}\left((m_A + n)\log n(\log {}^{1}\!/\!_{\delta} + \log(2 + \|A\|))\log {}^{1}\!/\!_{\delta} + (\log {}^{1}\!/\!_{\delta})^3\right),$$

where the tilde hides polynomial factors in $\log\log n$ and $\log\log {}^{1}\!/\!_{\delta}$. $\square$

4.3.2 General PSD Matrices – Theorem 4.3

The proof of Theorem 4.3 is identical to that of Theorem 4.2 except that we replace the SDD solver by the Conjugate Gradient method for implementing the Invert_A procedure. We use the following theorem.

Theorem 4.14. (Conjugate Gradient Method. See [She94]). *Given a system of linear equations $Mx = b$ and an error parameter $\epsilon > 0$, it is possible to obtain a vector u that is an approximate solution to the system, in the sense that*

$$\|u - M^{-1}b\|_M \leq \epsilon \|M^{-1}b\|_M.$$

The time required for computing u is $O\left(t_M\sqrt{\kappa(M)}\log {}^{1}\!/\!_{\epsilon}\right)$, – where $\kappa(M)$ denotes the condition number of M.

We now give a proof of Theorem 4.3.

Proof. (of Theorem 4.3) We use the ExpRational procedure to approximate the exponential. We run the Conjugate Gradient method with the on input $(I + {}^{A}\!/_{k})$, the vector y and error parameter ε_1. As with the SDD solver, the guarantee on the returned vector u_1 is in terms of $\|\cdot\|_{I+A/k}$, but as observed in the proof of Theorem 2.3, this implies $\|(I + {}^{A}\!/_{k})^{-1}y - u_1\| \leq \varepsilon_1 \|y\|$, as required for Invert_A. Thus, Theorem 4.1 implies that the procedure ExpRational computes a vector u approximating $e^{-A}v$, as desired.

Using Theorem 4.14, the time required by each call to Invert_A, $T^{\text{inv}}_{A,k,\varepsilon_1}$ is, $O\left(t_A\sqrt{\kappa(I + {}^{A}\!/_{k})} \log {}^{1}\!/_{\varepsilon_1}\right) = O\left(t_A\sqrt{1 + \|A\|} \log {}^{1}\!/_{\varepsilon_1}\right)$, and hence, from Theorem 4.1, the total running time is

$$\tilde{O}\left(t_A\sqrt{1+\|A\|}(\log {}^{1}\!/_{\delta} + \log(2 + \|A\|)) \log {}^{1}\!/_{\delta} + (\log {}^{1}\!/_{\delta})^2\right),$$

where the tilde hides polynomial factors in $\log\log n$ and $\log\log {}^{1}\!/_{\delta}$. □

Remark 4.15. *Note that, in comparison to the SDD solver, the Conjugate Gradient method has a significantly larger running time in general, because of the* $\sqrt{\kappa(M)}$ *factor. However, the Conjugate Gradient method only requires multiplication by the matrix* M*, hence the factor* t_M *in the running time, which could be smaller than the* m_M *factor in the running time for the SDD Solver.*

4.4 Beyond SDD Matrices

In this section, we give a proof of Theorem 4.4 that requires us to approximate $\exp(-A)v$, for a given vector v and matrix $A = \Pi H M H \Pi$, where M is an SDD matrix, H is a diagonal matrix with strictly positive entries and Π is a rank $(n-1)$ projection matrix, $\Pi \overset{\text{def}}{=} I - ww^\top$ (w is explicitly known and $\|w\| = 1$). Since A may be neither SDD, nor sparse, Theorem 4.2 does not suffice, whereas the running times in Theorems 4.3 and 2.5 are slow for our requirements.

Fortunately, Lemma 4.16 given below implements the Invert_A procedure for such matrices.

Lemma 4.16. (Invert_A Procedure for Theorem 4.4). *Given a positive integer k, vector y, an error parameter ε_1, a rank $(n-1)$ projection matrix $\Pi = I - ww^\top$ (where $\|w\| = 1$ and w is explicitly known), a diagonal matrix H with strictly positive entries, and an invertible SDD matrix M with m_M non-zero entries, let M_1 denote the matrix $(I + {}^1\!/\!_k \cdot \Pi H M H \Pi)$. We can compute a vector u such that*

$$\left\|M_1^{-1}y - u\right\| \leq \varepsilon_1 \left\|M_1^{-1}y\right\|,$$

in time $\tilde{O}((m_M + n)\log n \log \frac{1+\|HMH\|}{\varepsilon_1})$. (The tilde hides poly$(\log\log n)$ factors.)

Before we give a proof of Lemma 4.16, we complete the proof of Theorem 4.4 assuming Lemma 4.16. We use the ExpRational procedure with the Invert_A procedure given by Lemma 4.16. The time required for each call to Invert_A is $T^{\text{inv}}_{A,k,\varepsilon_1} \stackrel{\text{def}}{=} \tilde{O}((m_M + n)\log n \log \frac{1+\|HMH\|}{\varepsilon_1})$, and hence from Theorem 4.1, we get that we can compute the desired vector u approximating $e^{-A}v$ in total time

$$\tilde{O}\left((m_M + n)\log n(\log {}^1\!/\!_\delta + \log(2 + \|HMH\|))\log {}^1\!/\!_\delta + (\log {}^1\!/\!_\delta)^3\right),$$

where the tilde hides polynomial factors in $\log\log n$ and $\log\log {}^1\!/\!_\delta$. This completes the proof of Theorem 4.4. We now give a proof for Lemma 4.16, which is proven by combining the SDD Solver with the Sherman-Morrison formula.

In order to prove Lemma 4.16, we need to show how to approximate the inverse of a matrix of the form HMH, where H is diagonal and M is SDD. The following lemma achieves this.

Lemma 4.17. *Given a vector y, an error parameter ε_1, a diagonal matrix H with strictly positive entries, and an invertible SDD matrix M with m_m non-zero entries;*

we can compute a vector u such that $\|(HMH)^{-1}y - u\|_{HMH} \leq \varepsilon_1 \|(HMH)^{-1}y\|_{HMH}$, in time $\tilde{O}((m_M + n)\log n \log {}^1\!/\!\varepsilon_1)$. (The tilde hides factors of $\log\log n$.)

Proof. Observe that $(HMH)^{-1}y = H^{-1}M^{-1}H^{-1}y$. Use the SDD solver (Theorem 4.13) with inputs M, vector $H^{-1}y$ and parameter ε_1 to obtain a vector u_1 such that,

$$\left\|M^{-1}(H^{-1}y) - u_1\right\|_M \leq \varepsilon_1 \left\|M^{-1}H^{-1}y\right\|_M .$$

Return the vector $u \overset{\text{def}}{=} H^{-1}u_1$. We can bound the error in the output vector u as follows,

$$\begin{aligned}
\left\|(HMH)^{-1}y - u\right\|_{HMH}^2 &= \left\|H^{-1}M^{-1}H^{-1}y - H^{-1}u_1\right\|_{HMH}^2 \\
&= (H^{-1}M^{-1}H^{-1}y - H^{-1}u_1)^\top (HMH)(H^{-1}M^{-1}H^{-1}y - H^{-1}u_1) \\
&= (M^{-1}H^{-1}y - u_1)^\top M(M^{-1}H^{-1}y - u_1) \\
&= \left\|M^{-1}(H^{-1}y) - u_1\right\|_M^2 \\
&\leq \varepsilon_1^2 \left\|M^{-1}H^{-1}y\right\|_M^2 = \varepsilon_1^2 (M^{-1}H^{-1}y)^\top M(M^{-1}H^{-1}y) \\
&= \varepsilon_1^2 (H^{-1}M^{-1}H^{-1}y)^\top (HMH)(H^{-1}M^{-1}H^{-1}y) \\
&= \varepsilon_1^2 \left\|H^{-1}M^{-1}H^{-1}y\right\|_{HMH}^2 = \varepsilon_1^2 \left\|(HMH)^{-1}y\right\|_{HMH}^2 .
\end{aligned}$$

Thus, $\|(HMH)^{-1}y - u\|_{HMH} \leq \varepsilon_1 \|(HMH)^{-1}y\|_{HMH}$. Since H is diagonal, multiplication by H^{-1} requires $O(n)$ time. Hence, the total time is dominated by the SDD solver, giving a total running time of $\tilde{O}((m_M + n)\log n \log {}^1\!/\!\varepsilon_1)$. ☐

Now, we prove Lemma 4.16.

Proof. (of Lemma 4.16.) We sketch the proof idea first. Using the fact that w is an eigenvector of our matrix, we will split y into two components – one along w and one orthogonal (denote it z). Along w, we can easily compute the component of the required vector. Among the orthogonal component, we will write our matrix as the sum of $I + {}^1\!/_k \cdot HMH$ and a rank one matrix, and use the Sherman-Morrison formula to express its inverse. We can use Lemma 4.17 to compute the inverse of $I + {}^1\!/_k \cdot HMH$. The procedure is described in Figure 4.3 and the proof for the error analysis is given below.

Let $M_1 \stackrel{\text{def}}{=} {}^1\!/_k \cdot HMH$. Then, $I + {}^1\!/_k \cdot \Pi HMH\Pi = I + \Pi M_1 \Pi$. Without loss of generality, we will assume that $\|y\| = 1$. Note that $I + \Pi M_1 \Pi \succ 0$, and hence is invertible. Let $z \stackrel{\text{def}}{=} y - (w^\top y)w$. Thus, $w^\top z = 0$. Since w is an eigenvector of $(I + \Pi M_1 \Pi)$ with eigenvalue 1, we get,

$$(I + \Pi M_1 \Pi)^{-1} y = (I + \Pi M_1 \Pi)^{-1} z + (w^\top y)w. \tag{4.1}$$

Let's say $t \stackrel{\text{def}}{=} (I + \Pi M_1 \Pi)^{-1} z$. Then, $t + \Pi M_1 \Pi t = z$. Left-multiplying by $w^\top$, we get, $w^\top t = w^\top z = 0$. Thus, $\Pi t = t$, and hence $(I + \Pi M_1)t = z$, or equivalently, $t = (I + \Pi M_1)^{-1} z$.

$$\begin{aligned}(I + \Pi M_1 \Pi)^{-1} z &= (I + \Pi M_1)^{-1} z = (I + M_1 - ww^\top M_1)^{-1} z = (I + M_1 - w(M_1 w)^\top)^{-1} z \\ &= \left((I + M_1)^{-1} - \frac{(I + M_1)^{-1} ww^\top M_1 (I + M_1)^{-1}}{1 + w^\top M_1 (I + M_1)^{-1} w} \right) z \\ &\qquad \text{(Sherman-Morrison formula)} \\ &= (I + M_1)^{-1} z - \frac{w^\top M_1 (I + M_1)^{-1} z}{1 + w^\top M_1 (I + M_1)^{-1} w} (I + M_1)^{-1} w. \end{aligned} \tag{4.2}$$

Since we can write $I + M_1 = I + HMH = H(H^{-2} + M)H$, we can use Lemma 4.17 to estimate $(I + M_1)^{-1} z$ and $(I + M_1)^{-1} w$. Using Equation (4.2), the procedure for estimating $(I + \Pi M_1 \Pi)^{-1} x$ is described in Figure 4.3.

We need to upper bound the error in the above estimation procedure. From the assumption, we know that $\beta_1 = (I + M_1)^{-1} z - e_1$, where $\|e_1\|_{(I+M_1)} \leq \frac{\varepsilon_1}{6(1+\|M_1\|)} \|(I + M_1)^{-1} z\|_{(I+M_1)}$, and $\beta_2 = (I + M_1)^{-1} x - e_2$, where $, \|e_2\|_{(I+M_1)} \leq \frac{\varepsilon_1}{6(1+\|M_1\|)} \|(I + M_1)^{-1} w\|_{(I+M_1)}$. Combining Equations (4.1) and (4.2) and subtracting

Input: An SDD matrix M, a diagonal matrix H with positive entries, a unit vector w and a vector y.
Output: A vector u that approximates $(I+1/k\cdot\Pi HMH\Pi)^{-1}y$, where $\Pi \overset{\text{def}}{=} 1-ww^\top$.

1. Compute $z \overset{\text{def}}{=} y-(w^\top y)w$, and $M_1 \overset{\text{def}}{=} 1/k \cdot HMH$.
2. Estimate $(I+M_1)^{-1}z$ with error parameter $\frac{\varepsilon_1}{6(1+\|M_1\|)}$. Denote the vector returned by β_1.
3. Estimate $(I+M_1)^{-1}w$ with error parameter $\frac{\varepsilon_1}{6(1+\|M_1\|)}$. Denote the vector returned by β_2.
4. Compute
$$u_1 \overset{\text{def}}{=} \beta_1 - \frac{w^\top M_1\beta_1}{1+w^\top M_1\beta_2}\beta_2+(w^\top y)w. \tag{4.3}$$
Return u_1.

Figure 4.3: The Invert_A procedure for Theorem 4.4

Equation (4.3), we can write the error as,

$$\begin{aligned}
&(I+\Pi M_1\Pi)^{-1}y-u_1\\
&\quad= (I+M_1)^{-1}z-\beta_1-\frac{w^\top M_1(I+M_1)^{-1}z}{1+w^\top M_1(I+M_1)^{-1}w}(I+M_1)^{-1}w+\frac{w^\top M_1\beta_1}{1+w^\top M_1\beta_2}\beta_2\\
&\quad= e_1-\frac{w^\top M_1(I+M_1)^{-1}z}{1+w^\top M_1(I+M_1)^{-1}w}(I+M_1)^{-1}w\\
&\qquad+\frac{w^\top M_1[(I+M_1)^{-1}z-e_1]}{1+w^\top M_1[(I+M_1)^{-1}w-e_2]}[(I+M_1)^{-1}w-e_2]\\
&\quad= e_1+\frac{w^\top M_1(I+M_1)^{-1}z\cdot w^\top M_1 e_2}{(1+w^\top M_1(I+M_1)^{-1}w)(1+w^\top M_1[(I+M_1)^{-1}w-e_2])}(I+M_1)^{-1}w\\
&\qquad-\frac{w^\top M_1 e_1}{1+w^\top M_1[(I+M_1)^{-1}w-e_2]}[(I+M_1)^{-1}w-e_2]\\
&\qquad-\frac{w^\top M_1(I+M_1)^{-1}z}{1+w^\top M_1[(I+M_1)^{-1}w-e_2]}e_2.
\end{aligned}$$

Let us first bound the scalar terms. Note that $\|z\|\le\|y\|=1$.

$$|w^\top M_1(I+M_1)^{-1}z| \le \|w\|\left\|M_1(I+M_1)^{-1}z\right\| \le \left\|M_1(I+M_1)^{-1}\right\| \le 1\ ,$$

$$w^\top M_1 e_1 \leq \|w\| \, \|M_1\| \, \|e_1\| \leq \|M_1\| \, \|e_1\|_{(I+M_1)} \leq {}^{\varepsilon_1}\!/_6 \cdot \left\|(I+M_1)^{-1} z\right\|_{(I+M_1)}$$
$$= {}^{\varepsilon_1}\!/_6 \cdot \left\|(I+M_1)^{-1/2} z\right\| \leq {}^{\varepsilon_1}\!/_6.$$

Similarly, $w^\top M_1 e_2 \leq {}^{\varepsilon_1}\!/_6$. Also, $M_1(I+M_1)^{-1} \succeq 0$ and hence $w^\top M_1 (I+M_1)^{-1} w \geq 0$. Thus,

$$\begin{aligned}
\left\|(I+\Pi M_1 \Pi)^{-1} y - u_1\right\| &\leq \left\|(I+\Pi M_1 \Pi)^{-1} y - u_1\right\|_{(I+M_1)} \\
&\leq \|e_1\|_{(I+M_1)} + \frac{1 \cdot {}^{\varepsilon_1}\!/_6}{1 \cdot (1 - {}^{\varepsilon_1}\!/_6)} \left\|(I+M_1)^{-1} w\right\|_{(I+M_1)} \\
&\quad + \frac{{}^{\varepsilon_1}\!/_6}{(1 - {}^{\varepsilon_1}\!/_6)} \left(\left\|(I+M_1)^{-1} w\right\|_{(I+M_1)} + \|e_2\|_{(I+M_1)}\right) + \frac{1}{1 - {}^{\varepsilon_1}\!/_6} \|e_2\|_{(I+M_1)} \\
&\leq \frac{\varepsilon_1}{6(1+\|M_1\|)} \left\|(I+M_1)^{-1} z\right\|_{(I+M_1)} + \frac{4 \cdot {}^{\varepsilon_1}\!/_6}{1 - {}^{\varepsilon_1}\!/_6} \left\|(I+M_1)^{-1} w\right\|_{(I+M_1)} \\
&\leq {}^{\varepsilon_1}\!/_6 \left\|(I+M_1)^{-1/2} z\right\| + {}^{4\varepsilon_1}\!/_5 \left\|(I+M_1)^{-1/2} w\right\| \leq \varepsilon_1.
\end{aligned}$$

Other than the estimation of $(I+M_1)^{-1}z$ and $(I+M_1)^{-1}w$, we need to compute a constant number of dot products and a constant number of matrix-vector products with the matrix M_1. Multiplying a vector with $M_1 = {}^1\!/_k \cdot HMH$ takes time $O(m_M + n)$, giving a total time of $\tilde{O}((m_M + n) \log n \log \frac{1+\|HMH\|}{\varepsilon_1})$. □

4.5 Error Analysis for ExpRational

To complete the proof of Theorem 4.1, we need to analyze the role of the error that creeps in due to approximate matrix inversion. The problem is that this error, generated in each iteration of the Krylov basis computation, propagates to the later steps. Thus, small errors in the inverse computation may lead to the basis V_k computed by our algorithm to be quite far from the k-th order Krylov basis for B, v. In this section, we give the proof of Theorem 4.1, except for the proof of a few lemmas, which have been presented in Section 4.5.1 for better readability.

Proof Overview. At a very high-level, the proof follows the outline of the proof for Lanczos method. We first show that assuming the error in computing the inverse is small, $\widehat{T}_k$ can be used to approximate small powers of $B = (I + {}^{A}\!/\!_{k})^{-1}$ when restricted to the Krylov subspace, *i.e.* for all $i \leq k$, $\|B^i v - V_k \widehat{T}_k^i V_k^\top v\| \lessapprox \varepsilon_2$, for some small ε_2. This implies that we can bound the error in approximating $p((I + {}^{A}\!/\!_{k})^{-1})$ using $p(\widehat{T}_k)$, by $\varepsilon_2 \|p\|_1$, where p is a polynomial of degree at most k. This is the most technical part of the error analysis because we need to capture the propagation of error through the various iterations of the algorithm. We overcome this difficulty by expressing the final error as a sum of k terms, with the i^{th} term expressing how much error is introduced in the final candidate vector because of the error in the inverse computation during the i^{th} iteration. Unfortunately, the only way we know of bounding each of these terms is by *tour de force*. A part of this proof is to show that the spectrum of $\widehat{T}_k$ cannot shift far from the spectrum of B.

To bound the error in the candidate vector output by the algorithm, *i.e.* $\|f(B)v - V_k f(\widehat{T}_k) V_k^\top v\|$, we start by expressing e^{-x} as the sum of a degree k-polynomial p_k in $(1 + {}^{x}\!/\!_{k})^{-1}$ and a remainder function r_k. We use the analysis from the previous paragraph to upper bound the error in the polynomial part by $\varepsilon_2 \|p\|_1$. We bound the contribution of the remainder term to the error by bounding $\|r_k(B)\|$ and $\|r_k(\widehat{T}_k)\|$. This step uses the fact that eigenvalues of $r_k(\widehat{T}_k)$ are $\{r_k(\lambda_i)\}_i$, where $\{\lambda_i\}_i$ are eigenvalues of $\widehat{T_k}$. This is the reason our algorithm symmetrizes T_k to $\widehat{T}_k$. To complete the error analysis, we use the polynomials $p_k^\star$ from Theorem 4.11 and bound its ℓ_1 norm. Even though we do not know $p_k^\star$ explicitly, we can bound $\|p_k^\star\|_1$ indirectly by writing it as an interpolation polynomial and using that the values it assumes in $[0, 1]$ have to be small in magnitude.

Proof. For notational convenience, define $B \stackrel{\text{def}}{=} (I + {}^{A}\!/\!_{k})^{-1}$. Since the computation of Bv_i is not exact in each iteration, the eigenvalues of $\widehat{T}_k$ need not be bounded by the eigenvalues of B. Also, Lemma 4.8 no longer holds, *i.e.*, we can't guarantee that

$V_k\widehat{T}_k^t V_k^\top v_0$ is identical to $B^t v_0$. However, we can prove the following lemma that proves bounds on the spectrum of $\widehat{T}_k$ and also bounds the norm of the difference between the vectors $V_k\widehat{T}_k^t V_k^\top v_0$ and $B^t v_0$. This is the most important and technically challenging part of the proof.

Lemma 4.18 (Approximate Computation with $\widehat{T}_k$. Proof in Sec. 4.5.1)**.** *The coefficient matrix $\widehat{T}_k$ generated satisfies the following:*

1. *The eigenvalues of $\widehat{T}_k$ lie in* $\left[\left(1+\frac{\lambda_1(A)}{k}\right)^{-1} - \varepsilon_1\sqrt{k+1}, \left(1+\frac{\lambda_n(A)}{k}\right)^{-1} + \varepsilon_1\sqrt{k+1}\right]$.
2. *For any $t \leq k$, if $\varepsilon_1 \leq \varepsilon_2/(8(k+1)^{5/2})$ and $\varepsilon_2 \leq 1$, we have,* $\left\|B^t v_0 - V_k\widehat{T}_k^t e_1\right\| \leq \varepsilon_2$.

Here is an idea of the proof of the above lemma: Since, during every iteration of the algorithm, the computation of Bv_i is approximate, we will express BV_k in terms of T_k and an error matrix E. This will allow us to express $\widehat{T}_k$ in terms of T_k and a different error matrix. The first part of the lemma will follow immediately from the guarantee of the Invert_A procedure.

For the Second part, we first express $BV_k - V_k\widehat{T}_k$ in terms of the error matrices defined above. Using this, we can write the telescoping sum $B^tV_k - V_k\widehat{T}_k^t = \sum_{j=1}^t B^{t-j}(BV_k - V_k\widehat{T}_k)\widehat{T}_k^{j-1}$. We use triangle inequality and a *tour de force* calculation to bound each term. A complete proof is included in Section 4.5.1.
As a simple corollary, we can bound the error in the computation of the polynomial, in terms of the ℓ_1 norm of the polynomial being computed.

Corollary 4.19 (Approximate Polynomial Computation. Proof in Sec. 4.5.1)**.** *For any polynomial p of degree at most k, if $\varepsilon_1 \leq \varepsilon_2/(2(k+1)^{3/2})$ and $\varepsilon_2 \leq 1$,*

$$\left\|p(B)v_0 - V_k p(\widehat{T}_k)e_1\right\| \leq \varepsilon_2 \left\|p\right\|_1 .$$

Using this corollary, we can prove an analogue of Lemma 4.9, giving error bounds on the procedure in terms of degree k polynomial approximations. The proof is very similar and is based on writing f as a sum of a degree k polynomial and an *error function.*

Lemma 4.20 (Polynomial Approximation for ExpRational. Proof in Sec. 4.5.1)**.** *Let V_k be the orthonormal basis and $\widehat{T}_k$ be the matrix of coefficients generated by* ExpRational. *Let f be any function such that $f(B)$ and $f(T_k)$ are defined. Define $r_k(x) \stackrel{\text{def}}{=} f(x) - p(x)$. Then,*

$$\left\| f(B)v_0 - V_k f(\widehat{T}_k)e_1 \right\| \leq \min_{p \in \Sigma_k} \left(\varepsilon_2 \left\| p \right\|_1 + \max_{\lambda \in \Lambda(B)} \left| r_k(\lambda) \right| + \max_{\lambda \in \Lambda(\widehat{T}_k)} \left| r_k(\lambda) \right| \right). \tag{4.4}$$

In order to control the second error term in the above lemma, we need bounds on the eigenvalues of $\widehat{T}_k$, which are provided by Lemma 4.18.

For our application, $f(t) = f_k(t) \stackrel{\text{def}}{=} \exp\left(k \cdot (1 - {}^{1}\!/_{t})\right)$ so that $f_k((1 + {}^{x}\!/_{k})^{-1}) = \exp(-x)$. This function is discontinuous at $t = 0$. Under exact computation of the inverse, the eigenvalues of $\widehat{T}_k$ would be bounded by the eigenvalues of B and hence would lie in $(0, 1]$. Unfortunately, due to the errors, the eigenvalues of $\widehat{T}_k$ could be outside the interval. Since f is discontinuous at 0, and goes to infinity for small negative values, in order to get a reasonable approximation to f, we will ensure that the eigenvalues of $\widehat{T}_k$ are strictly positive, *i.e.*, $\varepsilon_1 \sqrt{k+1} < (1 + {}^{1}\!/_{k} \cdot \lambda_1(A))^{-1}$.

We will use the polynomials $p_k^\star$ from Theorem 4.11 in Lemma 4.20 to bound the final error. We will require the following lemma to bound the ℓ_1-norm of $p_k^\star$.

Lemma 4.21 (ℓ_1-norm Bound. Proof in Sec. 4.5.1)**.** *Given a polynomial p of degree k such that $p(0) = 0$ and*

$$\sup_{t \in (0,1]} \left| e^{-{}^{k}\!/_{t} + k} - p(t) \right| = \sup_{x \in [0, \infty)} \left| e^{-x} - p\left((1 + {}^{x}\!/_{k})^{-1} \right) \right| \leq 1 ,$$

we must have $\|p\|_1 \leq (2k)^{k+1}$.

This lemma is proven by expressing p as the interpolation polynomial on the values attained by p at the $k+1$ points $0, {}^1\!/\!{}_k, \ldots, {}^k\!/\!{}_k$, which allows us to express the coefficients in terms of these values. We can bound these values, and hence, the coefficients, since we know that p isn't too far from the exponential function. A complete proof is included in Section 4.5.1.

Theorem 4.11 shows that $p_k^\star(t)$ is a good uniform approximation to $e^{-k/t+k}$ over the interval $(0,1]$. Since $\Lambda(B) \subseteq (0,1]$, this will help us help us bound the second error term in Equation (4.4). Since $\widehat{T}_k$ can have eigenvalues larger that 1, we need to bound the error in approximating $f_k(t)$ by $p_k^\star(t)$ over an interval $(0,\beta]$, where $\beta \geq 1$. The following lemma, gives us the required error bound. This proof for this lemma bounds the error over $[1,\beta]$ by applying triangle inequality and bounding the change in f_k and p over $[1,\beta]$ separately.

Lemma 4.22 (Approximation on Extended Interval. Proof in Sec. 4.5.1)**.** *For any $\beta \geq 1$, any degree k polynomial p satisfies,*

$$\sup_{t\in(0,\beta]} |p(t) - f_k(t)| \leq \|p\|_1 \cdot (\beta^k - 1) + (f_k(\beta) - f_k(1)) + \sup_{t\in(0,1]} |p(t) - f_k(t)| \, .$$

We bound the final error using the polynomial $p_k^\star$ in Equation (4.4). We will use the above lemma for $\beta \overset{\text{def}}{=} 1 + \varepsilon_1\sqrt{k+1}$ and assume that $\varepsilon_1\sqrt{k+1} < (1 + {}^1\!/\!{}_k \cdot \lambda_1(A))^{-1}$.

$$\begin{aligned}
\left\| f(B)v_0 - V_k f(\widehat{T}_k)e_1 \right\| &\leq \varepsilon_2 \|p_k^\star\|_1 + \max_{\lambda\in\Lambda(B)} |r_k(\lambda)| + \max_{\lambda\in\Lambda(\widehat{T}_k)} |r_k(\lambda)| \\
&\leq \varepsilon_2 \|p_k^\star\|_1 + \sup_{\lambda\in(0,1]} |(f_k - p_k^\star)(\lambda)| + \sup_{\lambda\in(0,\beta]} |(f_k - p_k^\star)(\lambda)|. \\
&\qquad (\text{Since } \Lambda(B) \subseteq (0,1] \text{ and } \Lambda(\widehat{T}_k) \subseteq (0,\beta]\ ) \\
&\leq \varepsilon_2 \|p_k^\star\|_1 + \sup_{t\in(0,1]} |p_k^\star(t) - f_k(t)| +
\end{aligned}$$

$$\|p_k^\star\|_1 \cdot (\beta^k - 1) + (f_k(\beta) - f_k(1)) + \sup_{t\in(0,1]} |p_k^\star(t) - f_k(t)|$$

$$= \|p_k^\star\|_1 \cdot (\varepsilon_2 + \beta^k - 1) + (\exp\left({}^{k(\beta-1)}/_{\beta}\right) - 1) + 2 \sup_{t\in(0,1]} |p_k^\star(t) - f_k(t)| \ .$$

Given $\delta < 1$, we plug in the following parameters,

$$k \overset{\text{def}}{=} \max\{k_0, \log_2 {}^{8c_1}/_{\delta} + 2\log_2\log_2 {}^{8c_1}/_{\delta}\} = O\left(\log {}^1/_{\delta}\right) \ ,$$

$$\varepsilon_1 \overset{\text{def}}{=} {}^{\delta}/_{32} \cdot (k+1)^{-5/2} \cdot (1 + {}^1/_k \cdot \lambda_1(A))^{-1} \cdot (2k)^{-k-1},\ \beta \overset{\text{def}}{=} 1 + \varepsilon_1\sqrt{k+1},\ \varepsilon_2 \overset{\text{def}}{=} 8(k+1)^{5/2}\varepsilon_1 \ ,$$

where k_0, c_1 are the constants given by Theorem 4.11. Note that these parameters satisfy the condition $\varepsilon_1\sqrt{k+1} < (1 + {}^1/_k \cdot \lambda_1(A))^{-1}$. Corollary 4.11 implies that $p_k^\star(0) = 0$ and

$$\begin{aligned}\sup_{t\in(0,1]} |p_k^\star(t) - f_k(t)| &\le \frac{\delta}{8} \cdot \frac{\log_2 {}^{8c_1}/_{\delta} + 2\log_2\log_2 {}^{8c_1}/_{\delta}}{(\log_2 {}^{8c_1}/_{\delta})^2} \\ &\le \frac{\delta}{8} \cdot \frac{1}{\log_2 {}^{8c_1}/_{\delta}} \left(1 + 2 \cdot \frac{\log_2\log_2 {}^{8c_1}/_{\delta}}{\log_2 {}^{8c_1}/_{\delta}}\right) \le \frac{\delta}{8} \cdot \frac{1}{3} \cdot 3 \le \frac{\delta}{8} \ , \end{aligned} \tag{4.5}$$

where the last inequality uses $\delta \le 1 \le c_1$ and $\log_2 x \le x, \forall x \ge 0$. Thus, we can use Lemma 4.21 to conclude that $\|p_k^\star\|_1 \le (2k)^{k+1}$.

We can simplify the following expressions,

$$\exp\left({}^{k(\beta-1)}/_{\beta}\right) - 1 \le \exp\left(k\varepsilon_1 \cdot \sqrt{k+1}\right) - 1 \le \exp({}^{\varepsilon_2}/_8) - 1 \le (1 + {}^{\varepsilon_2}/_4) - 1 = {}^{\varepsilon_2}/_4 \ ,$$

$$\beta^k - 1 = (1 + \varepsilon_1\sqrt{k+1})^k - 1 \le \exp(k \cdot \varepsilon_1\sqrt{k+1}) - 1 \le {}^{\varepsilon_2}/_4.$$

Thus the total error $\|u - \exp(-A)v\| = \left\|f(B)v_0 - V_k f(\widehat{T}_k)e_1\right\| \le (2k)^{k+1} \cdot 2\varepsilon_2 + \varepsilon_2 + {}^{\delta}/_4 \le \delta$.

Running Time. The running time for the procedure is dominated by k calls to the Invert_A procedure with parameters k and ε_1, computation of at most k^2 dot-products and the exponentiation of $\widehat{T}_k$. The exponentiation of $\widehat{T}_k$ can be done in time $O(k^3)$ [PC99]. Thus the total running time is $O(T^{\text{inv}}_{A,k,\varepsilon_1} \cdot k + n \cdot k^2 + k^3)$. This completes the proof of the Theorem 4.1. □

4.5.1 Remaining Proofs

In this section, we give the remaining proofs in Section 4.5.

Lemma 4.23 (Lemma 4.18 Restated)**.** *The coefficient matrix $\widehat{T}_k$ generated satisfies the following:*

1. *The eigenvalues of $\widehat{T}_k$ lie in the interval*

$$\left[\left(1+\tfrac{\lambda_1(A)}{k}\right)^{-1} - \varepsilon_1\sqrt{k+1}, \left(1+\tfrac{\lambda_n(A)}{k}\right)^{-1} + \varepsilon_1\sqrt{k+1}\right].$$

2. *For any $t \le k$, if $\varepsilon_1 \le \varepsilon_2/(8(k+1)^{5/2})$ and $\varepsilon_2 \le 1$, we have, $\left\| B^t v_0 - V_k \widehat{T}_k^t e_1 \right\| \le \varepsilon_2$.*

Proof. Given a vector y, a positive integer k and real parameter $\varepsilon_1 > 0$, $\mathsf{Invert}_A(y, k, \varepsilon_1)$ returns a vector u_1 such that $\|By - u_1\| \le \varepsilon_1 \|y\|$, in time $T^{\text{inv}}_{A,k,\varepsilon_1}$. Thus, for each i, the vector w_i satisfies $\|Bv_i - w_i\| \le \varepsilon_1 \|v_i\| = \varepsilon_1$. Also define u_i as $u_i \overset{\text{def}}{=} Bv_i - w_i$. Thus, we get, $\|u_i\| \le \varepsilon_1$. Let E be the $n \times (k+1)$ matrix with its columns being $u_0, \ldots, u_k$. We can write the following recurrence,

$$BV_k = V_k T_k + E + \alpha_{k,k+1} v_{k+1} e_{k+1}^\top , \tag{4.6}$$

where each column of E has ℓ_2 norm at most ε_1. Note that we continue to do complete orthonormalization, so $V_k^\top V_k = I_k$. Thus, T_k is not tridiagonal, but rather *Upper Hessenberg*, i.e., $(T_k)_{ij} = 0$ whenever $i > j + 1$.

Multiplying both sides of Equation (4.6) by $V_k^\top$, we get $T_k = V_k^\top B V_k - V_k^\top E$. This implies,

$$\widehat{T}_k = V_k^\top B V_k - 1/2 \cdot (V_k^\top E + E^\top V_k) \tag{4.7}$$

$$= V_k^\top (V_k T_k + E + \alpha_{k,k+1} v_{k+1} e_{k+1}^\top) - 1/2 \cdot (V_k^\top E + E^\top V_k) \qquad \text{(Using (4.6))}$$

$$= T_k + 1/2 \cdot (V_k^\top E - E^\top V_k). \tag{4.8}$$

Define $E_1 \overset{\text{def}}{=} 1/2 \cdot (V_k^\top E + E^\top V_k)$. Thus, using Equation (4.7), $\widehat{T}_k = V_k^\top B V_k - E_1$. Let us first bound the norm of E_1.

$$\|E_1\| \leq 1/2 \cdot (\|V_k^\top E\| + \|E^\top V_k\|) \leq 1/2 \cdot (\|E\| + \|E^\top\|) \leq \|E\|_F \leq \varepsilon_1 \sqrt{k+1}\ .$$

Since $\widehat{T}_k = V_k^\top B V_k - E_1$. We have,

$$\lambda_{\max}(\widehat{T}_k) \leq \lambda_1(B) + \|E_1\| \leq (1 + 1/k \cdot \lambda_n(A))^{-1} + \varepsilon_1 \sqrt{k+1}\ ,$$

$$\lambda_{\min}(\widehat{T}_k) \geq \lambda_n(B) - \|E_1\| \geq (1 + 1/k \cdot \lambda_1(A))^{-1} - \varepsilon_1 \sqrt{k+1}\ .$$

(We use $\lambda_{\max}$ and $\lambda_{\min}$ for the largest and smallest eigenvalues of $\widehat{T}_k$ respectively in order to avoid confusion since $\widehat{T}_k$ is a $(k+1)\times(k+1)$ matrix and not an $n \times n$ matrix.) First, let us compute $BV_k - V_k\widehat{T}_k$.

$$BV_k - V_k \widehat{T}_k \overset{(4.6),(4.8)}{=} V_k T_k + E + \alpha_{k,k+1} v_{k+1} e_{k+1}^\top - V_k \left(T_k + 1/2 \cdot (V_k^\top E - E^\top V_k)\right)$$

$$= \left(I - 1/2 \cdot V_k V_k^\top\right) E + 1/2 \cdot V_k E^\top V_k + \alpha_{k,k+1} v_{k+1} e_{k+1}^\top\ . \tag{4.9}$$

Now,

$$\left\|B^t v_0 - V_k \widehat{T}_k^t e_1\right\| = \left\|\sum_{j=1}^t B^{t-j}(BV_k - V_k \widehat{T}_k)\widehat{T}_k^{j-1} e_1\right\| \qquad \text{(Telescoping sum)}$$

$$
\overset{(4.9)}{=} \left\| \sum_{j=1}^{t} B^{t-j} \left(\left(I - {}^{1}\!/\!_{2} \cdot V_k V_k^\top \right) E + {}^{1}\!/\!_{2} \cdot V_k E^\top V_k + \alpha_{k,k+1} v_{k+1} e_{k+1}^\top \right) \widehat{T}_k^{j-1} e_1 \right\|
$$

$$
\overset{\Delta-\text{ineq.}}{\leq} \sum_{j=1}^{t} \left\| B^{t-j} \left(\left(I - {}^{1}\!/\!_{2} \cdot V_k V_k^\top \right) E + {}^{1}\!/\!_{2} \cdot V_k E^\top V_k \right) \widehat{T}_k^{j-1} e_1 \right\| + \left\| \sum_{j=1}^{t} B^{t-j} \left(\alpha_{k,k+1} v_{k+1} e_{k+1}^\top \right) \widehat{T}_k^{j-1} e_1 \right\| . \tag{4.10}
$$

We can bound the first term in Equation (4.10) as follows.

$$
\sum_{j=1}^{t} \left\| B^{t-j} \left(\left(I - {}^{1}\!/\!_{2} \cdot V_k V_k^\top \right) E + {}^{1}\!/\!_{2} \cdot V_k E^\top V_k \right) \widehat{T}_k^{j-1} e_1 \right\| \tag{4.11}
$$

$$
\leq \sum_{j=1}^{t} \left\| \left(I - {}^{1}\!/\!_{2} \cdot V_k V_k^\top \right) E + {}^{1}\!/\!_{2} \cdot V_k E^\top V_k \right\| \left\| \widehat{T}_k \right\|^{j-1}
$$

$$
(\text{Using } \|B\| \leq 1)
$$

$$
\leq \left(\sum_{j=1}^{t} (1 + \varepsilon_1 \sqrt{k+1})^{j-1} \right) \left\| \left(I - {}^{1}\!/\!_{2} \cdot V_k V_k^\top \right) E + {}^{1}\!/\!_{2} \cdot V_k E^\top V_k \right\|
$$

$$
(\text{Using } \left\| \widehat{T}_k \right\| \leq 1 + \varepsilon_1 \sqrt{k+1})
$$

$$
\leq t(1 + \varepsilon_1 \sqrt{k+1})^{t-1} \left(\left\| \left(I - {}^{1}\!/\!_{2} \cdot V_k V_k^\top \right) \right\| \|E\| + {}^{1}\!/\!_{2} \cdot \|V_k\| \left\| E^\top \right\| \|V_k\| \right)
$$

$$
\leq 2t\varepsilon_1 \sqrt{k+1} (1 + \varepsilon_1 \sqrt{k+1})^{t-1}. \tag{4.12}
$$

The second term in Equation (4.10) can be bounded as follows.

$$
\left\| \sum_{j=1}^{t} B^{t-j} \left(\alpha_{k,k+1} v_{k+1} e_{k+1}^\top \right) \widehat{T}_k^{j-1} e_1 \right\| \leq |\alpha_{k,k+1}| \sum_{j=1}^{t} \|B\|^{t-j} \left\| v_{k+1} e_{k+1}^\top \widehat{T}_k^{j-1} e_1 \right\|
$$

$$
\leq (1 + \varepsilon_1) \sum_{j=1}^{t} \|B\|^{t-j} \left\| v_{k+1} e_{k+1}^\top \left(T_k + {}^{1}\!/\!_{2} \cdot (V_k^\top E - E^\top V_k) \right)^{j-1} e_1 \right\|
$$

$$
(\text{Using } |\alpha_{k,k+1}| \leq \|w_k\| \leq \|B v_k\| + \varepsilon_1 \leq 1 + \varepsilon_1 \text{ and } (4.8))
$$

$$
\leq (1 + \varepsilon_1) \sum_{j=1}^{t} \|B\|^{t-j} \left\| v_{k+1} e_{k+1}^\top \left(\left(T_k + {}^{1}\!/\!_{2} \cdot (V_k^\top E - E^\top V_k) \right)^{j-1} - T_k^{j-1} \right) e_1 \right\|
$$

$$
(\text{Using } e_{k+1}^\top T_k^r e_1 = 0 \text{ for } r < k \text{ as } T_k \text{ is Upper Hessenberg})
$$

$$
\begin{aligned}
&\leq (1+\varepsilon_1)\sum_{j=1}^{t}\left\|v_{k+1}e_{k+1}^{\top}\right\|\left(\sum_{i=1}^{j-1}\binom{j-1}{i}\left\|{}^{1}\!/\!_{2}\cdot(V_k^{\top}E-E^{\top}V_k)\right\|^{i}\|T_k\|^{j-1-i}\right)\\
&\qquad\qquad\text{(Using sub-multiplicity of } \|\cdot\| \text{ and } \|B\|\leq 1)\\
&\leq (1+\varepsilon_1)\sum_{j=1}^{t}\left(\sum_{i=1}^{j-1}\binom{j-1}{i}(\varepsilon_1\sqrt{k+1})^{i}(1+\varepsilon_1\sqrt{k+1})^{j-1-i}\right)\\
&\qquad\qquad\text{(Using } \left\|v_{k+1}e_{k+1}^{\top}\right\|\leq 1, \|T_k\|\leq(1+\varepsilon_1\sqrt{k+1})\\
&\qquad\qquad\qquad\qquad\text{and } \left\|{}^{1}\!/\!_{2}\cdot(V_k^{\top}E-E^{\top}V_k)\right\|\leq\varepsilon_1\sqrt{k+1})\\
&\leq (1+\varepsilon_1)\sum_{j=1}^{t}((1+2\varepsilon_1\sqrt{k+1})^{j-1}-1)\\
&\leq t(1+\varepsilon_1)((1+2\varepsilon_1\sqrt{k+1})^{t-1}-1). \qquad (4.13)
\end{aligned}
$$

Combining Equations (4.10),(4.12) and (4.13), we get,

$$
\begin{aligned}
\left\|B^{t}v_0-V_k\widehat{T}_k^{t}e_1\right\| &\leq 2t\varepsilon_1\sqrt{k+1}(1+\varepsilon_1\sqrt{k+1})^{t-1}+t(1+\varepsilon_1)((1+2\varepsilon_1\sqrt{k+1})^{t-1}-1)\\
&\leq 2t\varepsilon_1\sqrt{k+1}e^{\varepsilon_1(t-1)\sqrt{k+1}}+te^{\varepsilon_1}(e^{2\varepsilon_1(t-1)\sqrt{k+1}}-1) \qquad \text{(Using } 1+x\leq e^{x})\\
&\leq 2t\varepsilon_1\sqrt{k+1}e^{\varepsilon_1(t-1)\sqrt{k+1}}+t(e^{2\varepsilon_1 t\sqrt{k+1}}-1)\\
&\leq {}^{\varepsilon_2}\!/\!_{4}\cdot e^{\varepsilon_2/8(k+1)}+k(e^{\varepsilon_2/4(k+1)}-1) \qquad \text{(Using } \varepsilon_1\leq {}^{\varepsilon_2}\!/\!_{8(k+1)^{5/2}} \text{ and } t\leq k)\\
&\leq {}^{\varepsilon_2}\!/\!_{4}\cdot e^{\varepsilon_2/8(k+1)}+k{}^{\varepsilon_2}\!/\!_{4(k+1)}\cdot(1+{}^{\varepsilon_2}\!/\!_{4(k+1)})\\
&\qquad\qquad\text{(Using } e^{x}\leq 1+x+x^{2} \text{ for } 0\leq x\leq 1)\\
&\leq {}^{\varepsilon_2}\!/\!_{2}+{}^{\varepsilon_2}\!/\!_{2}\leq\varepsilon_2 \qquad \text{(Using } \varepsilon_2\leq 1 \text{ and } k\geq 0).
\end{aligned}
$$

This proves the lemma. □

Corollary 4.24 (Corollary 4.19 Restated)**.** *For any polynomial p of degree at most k, if $\varepsilon_1\leq\varepsilon_2/(2(k+1)^{3/2})$ and $\varepsilon_2\leq 1$,*

$$\left\|p(B)v_0-V_kp(\widehat{T}_k)e_1\right\|\leq\varepsilon_2\|p\|_1.$$

Proof. Suppose $p(x)$ is the polynomial $\sum_{t=0}^{k} a_t \cdot x^t$,

$$\left\| p(B)v_0 - V_k p(\widehat{T}_k)e_1 \right\| = \left\| \sum_{t=0}^{k} a_t \cdot B^t v_0 - V_k \sum_{t=0}^{k} a_t \cdot \widehat{T}_k^t e_1 \right\|$$
$$\leq \sum_{t=0}^{k} |a_t| \cdot \left\| B^t v_0 - V_k \widehat{T}_k^t e_1 \right\| \leq \varepsilon_2 \sum_{t=0}^{k} |a_t| = \varepsilon_2 \left\| p \right\|_1 ,$$

where the last inequality follows from the previous lemma as $\varepsilon_2, \varepsilon_1$ satisfy the required conditions. □

Lemma 4.25 (Lemma 4.20 Restated)**.** *Let V_k be the orthonormal basis and $\widehat{T}_k$ be the matrix of coefficients generated by the above procedure. Let f be any function such that $f(B)$ and $f(T_k)$ are defined. Then,*

$$\left\| f(B)v_0 - V_k f(\widehat{T}_k)e_1 \right\| \leq \min_{p \in \Sigma_k} \left(\varepsilon_2 \left\| p \right\|_1 + \max_{\lambda \in \Lambda(B)} |r_k(\lambda)| + \max_{\lambda \in \Lambda(\widehat{T}_k)} |r_k(\lambda)| \right) . \quad (4.14)$$

Proof. Let p be any degree k polynomial. Let $r_k \stackrel{\text{def}}{=} f - p$. We express f as $p + r_k$ and use the previous lemma to bound the error in approximating $p(B)v_0$ by $V_k f(\widehat{T}_k)e_1$.

$$\left\| f(B)v_0 - V_k f(\widehat{T}_k)e_1 \right\| \leq \left\| p(B)V_k e_1 - V_k p(\widehat{T}_k)e_1 \right\| + \left\| V_k r_k(B)e_1 - V_k r_k(\widehat{T}_k)e_1 \right\|$$
$$\leq \varepsilon_2 \left\| p \right\|_1 + \left\| V_k r_k(B)e_1 \right\| + \left\| V_k r_k(\widehat{T}_k)e_1 \right\|$$
$$\leq \varepsilon_2 \left\| p \right\|_1 + \left\| r_k(B) \right\| + \left\| r_k(\widehat{T}_k) \right\|$$
$$\leq \varepsilon_2 \left\| p \right\|_1 + \max_{\lambda \in \Lambda(B)} |r_k(\lambda)| + \max_{\lambda \in \Lambda(\widehat{T}_k)} |r_k(\lambda)|.$$

Minimizing over p gives us our lemma. □

Lemma 4.26 (Lemma 4.21 Restated)**.** *Given a polynomial p of degree k such that $p(0) = 0$ and*

$$\sup_{t \in (0,1]} \left| e^{-k/t + k} - p(t) \right| = \sup_{x \in [0,\infty)} \left| e^{-x} - p\left((1 + x/k)^{-1} \right) \right| \leq 1 ,$$

we must have $\|p\|_1 \leq (2k)^{k+1}$.

Proof. We know that $p(0) = 0$. Interpolating at the $k+1$ points $t = 0, {}^1\!/\!_k, {}^2\!/\!_k, \ldots, 1$, we can use Lagrange's interpolation formula to give,

$$p(x) \equiv \sum_{i=1}^{k} \frac{\prod_{0\leq j\leq k, j\neq i}(x - {}^j\!/\!_k)}{\prod_{0\leq j\leq k, j\neq i}({}^i\!/\!_k - {}^j\!/\!_k)} p({}^i\!/\!_k).$$

The above identity is easily verified by evaluating the expression at the interpolation points and noting that it is a degree k polynomial agreeing with p at $k+1$ points. Thus, if we were to write $p(x) = \sum_{l=1}^{k} a_l \cdot x^l$ (note that $a_0 = 0$), we can express the coefficients a_l as follows.

$$a_l = \sum_{i=1}^{k} \frac{\sum\limits_{\substack{0\leq j_1<\ldots<j_{k-l}\leq k \\ j_1,\ldots,j_{k-l}\neq i}} (-1)^{k-l} {}^{j_1}\!/\!_k \cdot \ldots \cdot {}^{j_{k-l}}\!/\!_k}{\prod_{0\leq j\leq k, j\neq i}({}^i\!/\!_k - {}^j\!/\!_k)} p({}^i\!/\!_k).$$

Applying triangle inequality, and noting that $p(t)$ is a 1-uniform approximation to $e^{-k/t+k}$ for $t \in (0,1]$, we get,

$$|a_l| \leq \sum_{i=1}^{k} \frac{\binom{k}{k-l}}{({}^1\!/\!_k)^k} |p({}^i\!/\!_k)| \leq \sum_{i=1}^{k} \frac{\binom{k}{k-l}}{({}^1\!/\!_k)^k} \left(e^{-\frac{k(k-i)}{i}} + {}^\delta\!/\!_2\right) \leq 2 \cdot k^{k+1} \binom{k}{k-l}.$$

Thus, we can bound the ℓ_1 norm of p as follows,

$$\|p\|_1 = \sum_{l=1}^{k} |a_l| \leq \sum_{l=1}^{k} 2 \cdot k^{k+1} \binom{k}{k-l} \leq (2k)^{k+1}.$$

□

Lemma 4.27 (Lemma 4.22 Restated)**.** *For any* $\beta \geq 1$*, any degree* k *polynomial* p *satisfies,*

$$\sup_{t\in(0,\beta]} |p(t) - f_k(t)| \leq \|p\|_1 \cdot (\beta^k - 1) + (f_k(\beta) - f_k(1)) + \sup_{t\in(0,1]} |p(t) - f_k(t)|.$$

Proof. Given a degree k polynomial p that approximates f_k over $(0,1]$, we wish to bound the approximation error over $(0,\beta]$ for $\beta \geq 1$. We will split the error bound over $(0,1]$ and $[1,\beta]$. Since we know that $f_k(1) - p(1)$ is small, we will bound the error over $[1,\beta]$ by applying triangle inequality and bounding the change in f_k and p over $[1,\beta]$ separately.

Let $\beta > 0$. First, let us calculate $\sup_{t\in[1,\beta]} |p(t) - f_k(t)|$.

$$\begin{aligned}
\sup_{t\in[1,\beta]} |p(t) - f_k(t)| &\leq \sup_{t\in[1,\beta]} \left(|p(t) - p(1)| + |p(1) - f_k(1)| + |f_k(1) - f_k(t)|\right) \\
&\leq \sup_{t\in[1,\beta]} \left(\|p\|_1 \cdot \max_{0\leq i\leq k} |t^i - 1^i| + |p(1) - f_k(1)| + |f_k(1) - f_k(t)|\right) \\
&\leq \sup_{t\in[1,\beta]} \left(\|p\|_1 \cdot \max_{0\leq i\leq k} |t^i - 1^i|\right) + |p(1) - f_k(1)| + \sup_{t\in[1,\beta]} |f_k(1) - f_k(t)| \\
&\leq \|p\|_1 \cdot (\beta^k - 1) + |p(1) - f_k(1)| + \sup_{t\in[1,\beta]} |f_k(1) - f_k(t)| \\
&\qquad\qquad \text{(Since } t \geq 1 \text{ and } t^k \text{ is increasing for } t \geq 0\text{)} \\
&\leq \|p\|_1 \cdot (\beta^k - 1) + |p(1) - f_k(1)| + (f_k(\beta) - f_k(1)) \\
&\qquad\qquad \text{(Since } f_k(t) \text{ is an increasing function for } t \geq 0\text{).}
\end{aligned}$$

Now, we can bound the error over the whole interval as follows.

$$\begin{aligned}
\sup_{t\in(0,\beta]} |p(t) - f_k(t)| &= \max\{\sup_{t\in(0,1]} |p(t) - f_k(t)|, \sup_{t\in[1,\beta]} |p(t) - f_k(t)|\} \\
&\leq \max\{\sup_{t\in(0,1]} |p(t) - f_k(t)|, \\
&\qquad \|p\|_1 \cdot (\beta^k - 1) + |p(1) - f_k(1)| + (f_k(\beta) - f_k(1))\} \\
&= \|p\|_1 \cdot (\beta^k - 1) + (f_k(\beta) - f_k(1)) + \sup_{t\in(0,1]} |p(t) - f_k(t)| \ .
\end{aligned}$$

$\square$

Notes

The material presented in this chapter is based on the paper "Approximating the Exponential, the Lanczos Method, and an $\tilde{O}(m)$-Time Spectral Algorithm for Balanced Separator" [OSV12], joint with Lorenzo Orecchia and Nisheeth Vishnoi, that appeared at STOC 2012. A full version of the paper is available on arxiv [OSV11].

Chapter 5

Matrix Inversion is as easy as Exponentiation

In this chapter, we prove that the inverse of a positive-definite matrix can be approximated by a weighted sum of a small number of matrix exponentials. Combining this with the result from the previous chapter, we establish an equivalence between matrix inversion and exponentiation up to polylogarithmic factors. In particular, this connection justifies our use of Laplacian solvers for designing fast algorithms for Balanced Separator. The proof relies on the Euler-Maclaurin formula and certain bounds derived from the Riemann zeta function. Versions of this lemma were proved in [BM05, BM10]. We give a simple and self-contained proof in this chapter.

5.1 Introduction

In the last chapter, appealing to techniques from approximation theory, the computation of $e^{-L}v$ was reduced to a small number of computations of the form $L^{-1}u$. Thus, using the near-linear-time Laplacian solver due to Spielman and Teng [ST04], we obtained an $\tilde{O}(m)$-time algorithm for approximating $e^{-L}v$ for graphs with m edges. Whether the Spielman-Teng result is necessary in order to compute $e^{-L}v$ in near-linear time is an interesting question in its own right (see [Vis12, Chapter 9]). We answer this question in the affirmative by presenting a reduction in the other direction, again relying on analytical techniques. We show that the inverse of a positive definite matrix can be represented as a sparse weighted sum of matrix exponentials. The following is our main result.

Theorem 5.1. *Given $\varepsilon, \delta \in (0,1]$, there exist poly*$(\log {}^{1}/_{\delta\varepsilon})$ *numbers* $0 < w_j, t_j = O(\text{poly}({}^{1}/_{\delta\varepsilon}))$, *such that for all symmetric matrices A satisfying $\delta I \preceq A \preceq I$, $(1-\varepsilon)A^{-1} \preceq \sum_j w_j e^{-t_j A} \preceq (1+\varepsilon)A^{-1}$.*

This proves that the problems of matrix exponentiation and matrix inversion are equivalent up to polylogarithmic factors. Note that the numbers w_j, t_j in the above theorem are *independent* of the matrix A, and are given explicitly in the proof. The proof of Theorem 5.1 follows from the lemma below, which gives such an approximation in the scalar world.

Lemma 5.2. *Given $\varepsilon, \delta \in (0,1]$, there exist poly*$(\log {}^{1}/_{\delta\varepsilon})$ *numbers* $0 < w_j, t_j = O(\text{poly}({}^{1}/_{\delta\varepsilon}))$, *such that for all $x \in [\delta, 1]$, $(1-\varepsilon)x^{-1} \leq \sum_j w_j e^{-t_j x} \leq (1+\varepsilon)x^{-1}$.*

Note that as x approaches 0 from the right, x^{-1} is unbounded, where as e^{-tx} is bounded by 1 for any $t > 0$. This justifies the assumption that $x \in [\delta, 1]$. We reiterate that the above lemma is not new, and versions of this lemma were proved in [BM05, BM10]. We present an arguably simpler and self-contained proof in the next few sections. We begin by showing how Lemma 5.2 implies Theorem 5.1.

Proof of Theorem 5.1

Let $\{\lambda_i\}_i$ be the eigenvalues of A with corresponding eigenvectors $\{u_i\}_i$. Since A is symmetric and $\delta I \preceq A \preceq I$, we have $\lambda_i \in [\delta, 1]$, for all i. Let $w_j, t_j > 0$ denote the numbers given by Lemma 5.2 for parameters ε and δ. Thus, Lemma 5.2 implies that all i, $(1-\varepsilon)\lambda_i^{-1} \leq \sum_j w_j e^{-t_j \lambda_i} \leq (1+\varepsilon)\lambda_i^{-1}$. Note that if λ_i is an eigenvalue of A, then λ_i^{-1} is the corresponding eigenvalue of A^{-1} and $e^{-t_j\lambda_i}$ is that of e^{-t_jA} with the same eigenvector. Thus, multiplying the scalar inequalities by $u_iu_i^\top$ and summing up, we obtain the matrix inequality $(1-\varepsilon)\sum_i \lambda_i^{-1}u_iu_i^\top \preceq \sum_j w_j \sum_i e^{-t_j\lambda_i}u_iu_i^\top \preceq (1+\varepsilon)\sum_i \lambda_i^{-1}u_iu_i^\top$. Hence, $(1-\varepsilon)A^{-1} \preceq \sum_j w_j e^{-t_jA} \preceq (1+\varepsilon)A^{-1}$.

5.2 Integral Representation, Discretization and Smoothness

The starting point of the proof of Lemma 5.2 is the easy integral identity $x^{-1} = \int_0^\infty e^{-xt}dt$. Thus, by discretizing this integral to a sum, the fact that one can approximate x^{-1} as a weighted sum of exponentials as claimed Lemma 5.2 is not surprising. The crux is to prove that this can be achieved using a *sparse* sum of exponentials. One way to discretize an integral to a sum is the so called *trapezoidal rule*. If g is the integrand, and $[a,b]$ is the interval of integration, this rule approximates the integral $\int_a^b g(t)dt$ by covering the area under g in the interval $[a,b]$ using *trapezoids* of small width, say h, as follows:

$$\int_a^b g(t)dt \;\approx\; T_g^{[a,b],h} \;\stackrel{\text{def}}{=}\; \frac{h}{2}\cdot\sum_{j=0}^{K-1}\left(g(a+jh)+g(a+(j+1)h)\right),$$

where $K \stackrel{\text{def}}{=} \frac{b-a}{h}$ is an integer. The choice of h determines the discretization of the interval $[a,b]$, and hence K, which is essentially the sparsity of the approximating sum. To apply this to the integral representation for x^{-1}, we have to first truncate the infinite integral $\int_0^\infty e^{-xt}dt$ to a large enough interval $[0,b]$, and then bound the error in the trapezoidal rule. Recall that the error needs to be of the form

$$\left|x^{-1} - \tfrac{h}{2}\sum_j\left(e^{-xjh}+e^{-x(j+1)h}\right)\right| \le \varepsilon x^{-1}.$$

For such an error guarantee to hold, we must have $xh \leq O_\varepsilon(1)$. Thus, if we want the approximation to hold for all $0 < x \leq 1$, we require $h \leq O_\varepsilon(1)$, which in turn implies that $K \geq \Omega_\varepsilon(b)$. Also, if we restrict the interval to $[0, b]$, the truncation error is $\int_b^\infty e^{-xt} dt = x^{-1} e^{-bx}$, forcing $b \geq \delta^{-1} \log {}^1\!/\!_\varepsilon$ for this error to be at most ${}^\varepsilon\!/\!_x$ for all $x \in [\delta, 1]$. Thus, this way of discretizing can only give us a sum which uses poly$({}^1\!/\!_\delta)$ exponentials, which does not suffice for our application.

This suggests that we should pick a discretization such that t, instead of increasing linearly with h, increases much more rapidly. Thus, a natural idea is to allow t to grow geometrically. This can be achieved by substituting $t = e^s$ in the above integral to obtain the identity $x^{-1} = \int_{-\infty}^{\infty} e^{-xe^s+s} ds$. We show that discretizing this integral using the trapezoidal rule does indeed give us the lemma.

For convenience, we define $f_x(s) \stackrel{\text{def}}{=} e^{-xe^s+s}$. First, observe that $f_x(s) = x^{-1} \cdot f_1(s + \ln x)$. Since we also allow the error to scale as x^{-1}, as x varies over $[\delta, 1]$, s needs to change only by an additive $\log {}^1\!/\!_\delta$ to compensate for x. Roughly, this suggests that when approximating this integral by the trapezoidal rule, the dependence on ${}^1\!/\!_\delta$ is likely logarithmic, instead of polynomial. The proof formalizes this intuition and uses the fact that the error in the approximation by the trapezoidal rule can be expressed using the Euler-Maclaurin formula (see Section 5.3.1) which involves higher order derivatives of f_x. We establish the following properties about the derivatives of f_x which, when combined with known estimates on Bernoulli numbers obtained from the Riemann zeta function, allow us to bound this error with relative ease (see Section 5.3.2): (1) All the derivatives of f_x up to any fixed order, vanish at the end points of the integration interval (in the limit). (2) The derivatives of f_x are reasonably *smooth*; the L_1 norm of the k-th derivative is bounded roughly by $x^{-1}k^k$ (see Lemma 5.4). In summary, this allows us to approximate x^{-1} as an *infinite* sum of exponentials. In this sum, the contribution beyond about poly$(\log {}^1\!/\!_{\varepsilon\delta})$ terms turns out to be negligible, and hence we can truncate the infinite sum to obtain our final approximation (see Section 5.3.3).

We now present simple properties of the derivatives of f_x, alluded to above, which underlie the technical intuition as to why an approximation of the kind claimed in Lemma 5.2 should exist. Let $f_x^{(k)}(s)$ denote the k^{th} derivative of the function f_x with respect to s. The first fact relates $f_x^{(k)}(s)$ to $f_x(s)$.

Fact 5.3. *For any non-negative integer k, $f_x^{(k)}(s) = f_x(s)\sum_{j=0}^{k} c_{k,j}(-xe^s)^j$, where $c_{k,j}$ are some non-negative integers satisfying $\sum_{j=0}^{k} c_{k,j} \le (k+1)^{k+1}$.*

Proof. We prove this lemma by induction on k. For $k = 0$, we have $f_x^{(0)}(s) = f_x(s)$. Hence, $f_x^{(0)}$ is of the required form, with $c_{0,0} = 1$, and $\sum_{j=0}^{0} c_{0,j} = 1$. Hence, the claim holds for $k = 0$. Suppose the claim holds for k. Hence, $f_x^{(k)}(s) = f_x(s)\sum_{j=0}^{k} c_{k,j}(-xe^s)^j$, where $c_{k,j}$ are non-negative integers satisfying $\sum_{j=0}^{k} c_{k,j} \le (k+1)^{k+1}$. We can compute $f_x^{(k+1)}(s)$ as follows,

$$\begin{aligned} f_x^{(k+1)}(s) &= \frac{d}{ds}\left(\sum_{j=0}^{k} c_{k,j}(-xe^s)^j f_x(s)\right) = \sum_{j=0}^{k} c_{k,j}(j - xe^s + 1)(-xe^s)^j f_x(s) \\ &= f_x(s)\sum_{j=0}^{k+1}((j+1)c_{k,j} + c_{k,j-1})(-xe^s)^j, \end{aligned}$$

where we define $c_{k,k+1} \overset{\text{def}}{=} 0$, and $c_{k,-1} \overset{\text{def}}{=} 0$. Thus, if we define $c_{k+1,j} \overset{\text{def}}{=} (j+1)c_{k,j} + c_{k,j-1}$, we get that $c_{k+1,j} \ge 0$, and that $f_x^{(k+1)}$ is of the required form. Moreover, we get, $\sum_{j=0}^{k+1} c_{k+1,j} \le (k+2)(k+1)^{k+1} + (k+1)^{k+1} = (k+3)(k+1)(k+1)^k \le (k+2)^2(k+1)^k \le (k+2)^{k+2}$. This proves the claim for $k+1$ and, hence, the fact follows by induction. □

The next lemma uses the fact above to bound the L_1 norm of $f_x^{(k)}$.

Lemma 5.4. *For every non-negative integer k, $\int_{-\infty}^{\infty}\left|f_x^{(k)}(s)\right| ds \le \frac{2}{x}\cdot e^k(k+1)^{2k}$.*

Proof. By Fact 5.3, $\int_{-\infty}^{\infty} \left| f_x^{(k)}(s) \right| ds$ is at most

$$\begin{aligned}
\int_{-\infty}^{\infty} \left| \left(\sum_{j=0}^{k} c_{k,j} (-xe^s)^j \right) \right| e^{-xe^s+s} ds \quad &\overset{t=xe^s}{=} \quad \frac{1}{x} \int_0^{\infty} \left| \left(\sum_{j=0}^{n} c_{k,j} (-t)^j \right) \right| e^{-t} dt \\
&\overset{\text{Fact 5.3}}{\leq} \quad \frac{1}{x} (k+1)^{k+1} \left(\int_0^1 e^{-t} dt + \int_1^{\infty} t^k e^{-t} dt \right) \\
&\leq \quad \frac{1}{x} \cdot (k+1)^{k+1} \cdot (1+k!) \leq \frac{2}{x} \cdot e^k (k+1)^{2k},
\end{aligned}$$

where the last inequality uses $k+1 \leq e^k$, and $1 + k! \leq 2(k+1)^k$. $\square$

We conclude this section by giving a brief comparison of our proof to that from [BM05]. While the authors in [BM05] employ both the trapezoidal rule and the Euler-Maclaurin formula, our proof strategy is different and leads to a shorter and simpler proof. In contrast to the previous proof, we use the Euler-Maclaurin formula in the limit over $[-\infty, \infty]$, and since the derivatives of f_x vanish in the limit, we save considerable effort in bounding the derivatives at the end points of the integral, which is required when using the Euler-Maclaurin formula to bound the error. We manage to use simpler bounds, at the cost of slightly worse parameters. On the way, we obtain an approximation of x^{-1} as an infinite sum of exponentials that holds for all $x > 0$, which we believe is interesting in itself.

5.3 Proof of the Reduction

Before we introduce the Euler-Maclaurin formula which captures the error in the approximation of an integral by the trapezoidal rule, we introduce the Bernoulli numbers and polynomials, bounds on which are derived using a connection to the Riemann zeta function.

5.3.1 Bernoulli Polynomials and Euler-Maclaurin Formula

The Bernoulli numbers, denoted by b_k for any integer $k \geq 0$, are a sequence of rational numbers which, while discovered in an attempt to compute sums of the form $\sum_{i\geq 0}^{n} i^k$, have deep connections to several areas of mathematics, including number theory and analysis.[1] They can be defined recursively as: $b_0 = 1$, and the following equation which is satisfied for all positive integers $k \geq 2$, $\sum_{j=0}^{k-1} \binom{k}{j} b_j = 0$. This implies that $(e^t - 1)\sum_{k=0}^{\infty} b_k \frac{t^k}{k!} = t$. Further, it can be checked that $\frac{t}{2} + \frac{t}{e^t-1}$ is an even function, thus implying that $b_k = 0$ for odd $k \geq 2$. Given the Bernoulli numbers, the Bernoulli polynomials are defined as $B_k(s) \overset{\text{def}}{=} \sum_{j=0}^{k} \binom{k}{j} b_j s^{k-j}$. It follows from the definition that, for all k, and $j \leq k$, $\frac{B_k^{(j)}(s)}{k!} = \frac{B_{k-j}(s)}{(k-j)!}$. We also get $B_0(s) \equiv 1$, $B_1(s) \equiv s - \frac{1}{2}$. Moreover, using the definition of Bernoulli numbers, we get that $B_k(0) = B_k(1) = b_k$ for all $k \geq 2$. We also need the following bounds on the Bernoulli polynomials and the Bernoulli numbers.

Lemma 5.5. *For any non-negative integer k, and for all $s \in [0,1]$, $\frac{|B_{2k}(s)|}{(2k)!} \leq \frac{|b_{2k}|}{(2k)!} \leq \frac{4}{(2\pi)^{2k}}$.*

Proof. The first inequality follows from a well-known fact that $|B_{2k}(s)| \leq |b_{2k}|$ for all $s \in [0,1]$ (see [GKP94]). For the second inequality, we recall the following connection between Bernoulli numbers and the Riemann zeta function for any even positive integer, proved by Euler (see [GKP94]), $\zeta(2k) \overset{\text{def}}{=} \sum_{j\geq 1} j^{-2k} = (-1)^{k+1}\frac{b_{2k}(2\pi)^{2k}}{2\cdot(2k)!}$. Thus, $\frac{|b_{2k}|}{(2k)!} = \frac{2}{(2\pi)^{2k}} \sum_{j\geq 1} j^{-2k} \leq 4(2\pi)^{-2k}$. $\square$

One of the most significant connections in analysis involving the Bernoulli numbers is the *Euler-Maclaurin formula* which describes the error in approximating an integral by the trapezoidal rule.

[1]The story goes that when Charles Babbage designed the Analytical Engine in the 19th century, one of the most important tasks he hoped the Engine would perform was the calculation of Bernoulli numbers.

Lemma 5.6 (Euler-Maclaurin Formula)**.** *Given a function $g : \mathbb{R} \to \mathbb{R}$, for any $a < b$, any positive h, and any positive integer $N \in \mathbb{N}$, we have,*

$$\int_a^b g(s)ds - T_g^{[a,b],h} = h^{2N+1} \int_0^K \frac{B_{2N}(s - [s])}{(2N)!} g^{(2N)}(a + sh)ds \\ - \sum_{j=1}^{N} \frac{b_{2j}}{(2j)!} h^{2j} \left(g^{(2j-1)}(b) - g^{(2j-1)}(a) \right), \qquad (5.1)$$

where $K \overset{\text{def}}{=} \frac{b-a}{h}$ is an integer, and $[\cdot]$ denotes the integer part.

Note that the Euler-Maclaurin formula is really a family of formulae, one each for the choice of N, which we call the *order* of the formula. Also note that this formula captures the error *exactly*. This error can be much less than the naive bound obtained by summing up the absolute value of the error due to each trapezoid. The first term in (5.1), after removing the contribution due to the Bernoulli polynomials via Lemma 5.5, can be bounded by the L_1 norm of $g^{(2N)}$. The second term in (5.1) depends only on $g^{(2N-1)}$ evaluated at the ends of the interval. The choice of N is influenced by how well behaved the higher order derivatives of the function are. For example, if $g(s)$ is a polynomial, when $2N > \text{degree}(g)$, we get an exact expression for $\int_a^b g(s)ds$ in terms of the values of the derivatives of g at a and b.

In the next section, we use the Euler-Maclaurin formula to bound the error in approximating the integral $\int f_x(s)ds$ using the trapezoidal rule. For our application, we pick a and b such that the derivatives up to order $2N-1$ at a and b are negligible. Since the sparsity of the approximation is $\Omega(1/h)$, for the sparsity to depend logarithmically on the error parameter ε, we need to pick N to be roughly $\Omega(\log 1/\varepsilon)$, so that the first error term in (5.1) is comparable to ε.

We end this section by giving a proof sketch for the Euler-Maclaurin formula (see also [Tao]). We'll first prove the formula for $h = 1$ and for the interval $[0, 1]$. Consider the integral $\int_0^1 \frac{B_{2N}^{(2N)}(s)}{(2N)!} g(s)ds$, and apply integration by parts[2] to it repeatedly to

[2] $\int \frac{du}{ds} v ds = uv - \int u \frac{dv}{ds} ds$.

obtain

$$\int_0^1 \frac{B_{2N}^{(2N)}(s)}{(2N)!} g(s)ds = \frac{B_{2N}^{(2N-1)}(s)}{(2N)!} g(s)\Bigg|_0^1 - \frac{B_{2N}^{(2N-2)}(s)}{(2N)!} g^{(1)}(s)\Bigg|_0^1 + \frac{B_{2N}^{(2N-3)}(s)}{(2N)!} g^{(2)}(s)\Bigg|_0^1$$
$$- \cdots - \frac{B_{2N}(s)}{(2N)!} g^{(2N-1)}(s)\Bigg|_0^1 + \int_0^1 \frac{B_{2N}(s)}{(2N)!} g^{(2N)}(s)ds.$$

Using the fact that for all $k \leq 2N$, $\frac{B_{2N}^{(k)}(s)}{(2N)!} = \frac{B_{2N-k}(s)}{(2N-k)!}$, and rearranging, we get,

$$\int_0^1 B_0(s)g(s)ds - B_1(s)g(s)\Bigg|_0^1 = \sum_{k=2}^{2N} (-1)^{k-1} \frac{B_k(s)}{k!} g^{(k-1)}(s)\Bigg|_0^1 + \int_0^1 \frac{B_{2N}(s)}{(2N)!} g^{(2N)}(s)ds.$$

Now, using $B_0(s) \equiv 1$, we get that the first term on the l.h.s. is $\int_0^1 g(s)ds$. Also, since $B_1(1) = 1/2, B_1(0) = -1/2$, we see that the second term on the l.h.s. is $1/2 \cdot (g(0) + g(1)) = T_g^{[0,1],1}$. Finally, using $B_k(0) = B_k(1) = b_k$ for $k \geq 2$, and that $b_k = 0$ when $k \geq 2$ is odd, we get the desired formula for $h = 1$ and the interval $[0, 1]$. By a change of variables, we obtain the formula for an arbitrary h and the interval $[a, a + h]$. Summing the formula for all the intervals $[a + (j - 1)h, a + jh]$, we obtain the final formula.

5.3.2 Approximation Using an Infinite Sum

As mentioned in Section 5.2, we approximate the integral $\int_{-\infty}^{\infty} f_x(s)ds$ using the trapezoidal rule. We bound the error in this approximation using the Euler-Maclaurin formula. Since the Euler-Maclaurin formula applies to finite intervals, we first fix the step size h, use the Euler-Maclaurin formula to bound the error in the approximation over the interval $[-bh, bh]$ (where b is some positive integer), and then let b go to ∞. This allows us to approximate the integral over $[-\infty, \infty]$ by an infinite sum of exponentials. In the next section, we truncate this sum to obtain our final approximation.

We are given $\varepsilon, \delta \in (0,1]$. Fix an $x \in [\delta, 1]$, the step size $h = \Theta\left((\log {}^{1}\!/\!_{\varepsilon})^{-2}\right)$, and the order of the Euler-Maclaurin formula, $N = \Theta\left(\log {}^{1}\!/\!_{\varepsilon}\right)$ (exact parameters to be specified later). For any positive integer b, applying the order N Euler-Maclaurin formula to the integral $\int_{-bh}^{bh} f_x(s)ds$, and using bounds from Lemma 5.5, we get,

$$\begin{aligned}\left|\int_{-bh}^{bh} f_x(s)ds - T_{f_x}^{[-bh,bh],h}\right| \leq & 4\left(\frac{h}{2\pi}\right)^{2N}\int_{-bh}^{bh}\left|f_x^{(2N)}(s)\right|ds \\ & + \sum_{j=1}^{N} 4\left(\frac{h}{2\pi}\right)^{2j}\left(\left|f_x^{(2j-1)}(-bh)\right| + \left|f_x^{(2j-1)}(bh)\right|\right).\end{aligned} \tag{5.2}$$

Now, we can use Fact 5.3 to bound the derivatives in the last term of (5.2). Fact 5.3 implies that for any s and any positive integer k, $\left|f^{(k)}(s)\right| \leq f_x(s)(k+1)^{k+1}\max\{1,(xe^s)^k\}$. Thus, for $b \geq -\frac{1}{h}\log\frac{1}{x}$, we have $xe^{-bh} \leq 1$ and $\left|f^{(k)}(-bh)\right| \leq e^{-bh}(k+1)^{k+1}$, and hence $f^{(k)}(-bh)$ vanishes for any fixed k and h, as b goes to ∞. Also, for any $x > 0$, and $b > \frac{1}{h}\log\frac{1}{x}$, we get, $\left|f^{(k)}(bh)\right| \leq x^k e^{(k+1)bh - xe^{bh}}(k+1)^{k+1}$, which again vanishes for any fixed k and h, as b goes to ∞. Thus, letting b go to ∞ and observing that $T_{f_x}^{[-bh,bh],h}$ converges to $h\sum_{j\in\mathbb{Z}} f_x(jh)$, (5.2) implies,

$$\left|\int_{-\infty}^{\infty} f_x(s)ds - h\sum_{j\in\mathbb{Z}} f_x(jh)\right| \leq 4\left(\frac{h}{2\pi}\right)^{2N}\int_{-\infty}^{\infty}\left|f_x^{(2N)}(s)\right|ds. \tag{5.3}$$

Hence, since the derivatives of the function $f_x(s)$ vanish as s goes to $\pm\infty$, the error in approximating the integral over $[-\infty,\infty]$ is just controlled by its *smoothness.* Since we already know f_x is a very smooth function, we are in good shape. Using Lemma 5.4, we get, $\left(\frac{h}{2\pi}\right)^{2N}\int_{-\infty}^{\infty}\left|f_x^{(2N)}(s)\right|ds \leq \frac{2}{x}\left(\frac{(2N+1)^2eh}{2\pi}\right)^{2N}$. Thus, if we let $h \overset{\text{def}}{=} \frac{2\pi}{e^2(2N+1)^2}$, and $N \overset{\text{def}}{=} \left\lceil \frac{1}{2}\log\frac{24}{\varepsilon}\right\rceil$, (5.3) implies that,

$$\left|x^{-1} - h\sum_{j\in\mathbb{Z}} e^{jh}\cdot e^{-xe^{jh}}\right| = \left|\int_{-\infty}^{\infty} f_x(s)ds - h\sum_{j\in\mathbb{Z}} f_x(jh)\right| \leq 8e^{-2N}\cdot\frac{1}{x} \leq \frac{\varepsilon}{3}\frac{1}{x}. \tag{5.4}$$

Also note that the above approximation holds for all $x > 0$. Thus, in particular, we can approximate the function x^{-1} over $[\delta, 1]$ as an (infinite) sum of exponentials.

5.3.3 Truncating the Infinite Sum and Proof of Lemma 5.2

Now, we want to truncate the infinite sum of exponentials approximating x^{-1} given by (5.4). Since the function $f_x(s) = e^s \cdot e^{-xe^s}$ is non-decreasing for $s < \log {}^1\!/\!_x$, we majorize the lower tail by an integral. For $A \overset{\text{def}}{=} \lfloor -\frac{1}{h} \log \frac{3}{\varepsilon} \rfloor < 0 \le \frac{1}{h} \log \frac{1}{x}$ (since $x \le 1$),

$$h \sum_{j<A} e^{jh} \cdot e^{-xe^{jh}} \le h \int_{-\infty}^{A} e^{jh} \cdot e^{-xe^{jh}} dj = \int_0^{e^{Ah}} e^{-xt} dt = x^{-1}\left(1 - e^{-xe^{Ah}}\right) \le \frac{\varepsilon}{3}\frac{1}{x}. \tag{5.5}$$

Again, for the upper tail, since the function $f_x(s) = e^s \cdot e^{-xe^s}$ is non-increasing for $s \ge \log \frac{1}{x}$, we majorize by an integral. For $B \overset{\text{def}}{=} \lceil \frac{1}{h} \log\left(\frac{1}{\delta} \log \frac{3}{\varepsilon}\right) \rceil \ge \frac{1}{h} \log \frac{1}{x}$ (since $x \ge \delta$ and $\varepsilon \le 1$),

$$h \sum_{j>B} e^{jh} \cdot e^{-xe^{jh}} \le h \int_{B}^{\infty} e^{jh} \cdot e^{-xe^{jh}} dj = \int_{e^{Bh}}^{\infty} e^{-xt} dt = x^{-1} \cdot e^{-xe^{Bh}} \le \frac{\varepsilon}{3}\frac{1}{x}. \tag{5.6}$$

Before we complete the proof, we list here the setting of all parameters for completeness:

$$N = \left\lceil \frac{1}{2} \log \frac{24}{\varepsilon} \right\rceil, \; h = \frac{2\pi}{e^2(2N+1)^2}, \; A = \left\lfloor -\frac{1}{h} \log \frac{3}{\varepsilon} \right\rfloor, \; B = \left\lceil \frac{1}{h} \log\left(\frac{1}{\delta} \log \frac{3}{\varepsilon}\right) \right\rceil.$$

Thus, combining (5.4), (5.5) and (5.6), the final error is given by,

$$\left|\frac{1}{x} - h\sum_{j \ge A}^{B} e^{jh} \cdot e^{-xe^{jh}}\right| \le \left|\frac{1}{x} - h \sum_{j \in \mathbb{Z}} e^{jh} \cdot e^{-xe^{jh}}\right| + h \sum_{j<A} e^{jh} \cdot e^{-xe^{jh}} + h \sum_{j>B} e^{jh} \cdot e^{-xe^{jh}} \le \frac{\varepsilon}{x}.$$

Hence, $(1-\varepsilon)x^{-1} \leq \sum_{j \geq A}^{B} he^{jh} \cdot e^{-xe^{jh}} \leq (1+\varepsilon)x^{-1}$, implying the claim of Lemma 5.2.

Notes

The material presented in this chapter is based on the paper "Matrix Inversion is as easy as Exponentiation", joint with Nisheeth K. Vishnoi [SV13].

Chapter 6

Uniform Approximations to e^{-x}

In this chapter, we prove that e^{-x} can be uniformly approximated up to a small additive error, in a non-negative interval $[a, b]$ with a polynomial of degree roughly $\sqrt{b-a}$. We also give a matching lower bound to prove that this dependence on $(b-a)$ is optimal.

Combined with the Lanczos method from Chapter 4, it immediately implies Theorem 4.5 that gives a simple algorithm to compute $\exp(-A)v$ for symmetric PSD matrices that runs in time roughly $O(t_A \cdot \sqrt{\|A\|})$, where t_A is time required to multiply a given vector v with A. This result, in turn, gives a simple algorithm for Balanced Separator (Theorem 3.2).

6.1 Introduction

We will discuss uniform approximations to e^{-x} and give a proof of Theorem 2.7 that shows the existence of polynomials that approximate e^{-x} uniformly over the interval $[0, b]$, whose degree grows as $\sqrt{b}$ and also gives a lower bound stating that this dependence is necessary. We restate a more precise version of the theorem here for completeness.

Theorem 6.1 (Uniform Approximation to e^{-x}).

- **Upper Bound.** *For every $0 \le a < b$, and a given error parameter $0 < \delta \le 1$, there exists a polynomial $p_{a,b,\delta}$ that satisfies,*

$$\sup_{x\in[a,b]} |e^{-x} - p_{a,b,\delta}(x)| \le \delta \cdot e^{-a},$$

and has degree $O\left(\sqrt{\max\{\log {}^1\!/\!_\delta, (b-a)\}} \cdot (\log {}^1\!/\!_\delta)^{3/2} \cdot \log\log {}^1\!/\!_\delta\right)$.

- **Lower Bound.** *For every $0 \le a < b$ such that $a + \log_e 4 \le b$, and $\delta \in (0, {}^1\!/\!_8]$, any polynomial $p(x)$ that approximates e^{-x} uniformly over the interval $[a, b]$ up to an error of $\delta \cdot e^{-a}$, must have degree at least $\frac{1}{2} \cdot \sqrt{b-a}$.*

We reiterate that the upper bound from the above theorem is implicit in the work of Hochbruck and Lubich [HL97], as pointed out to us by an anonymous reviewer for [OSV12]. This chapter gives a different proof of this result, starting from the rational approximation result of Saff, Schönage, and Varga [SSV75] mentioned earlier, and avoids complex analysis, unlike [HL97].

We give an overview of the proofs of both these results next.

Upper Bound.

We wish to show that there exists a polynomial of degree of the order of $\sqrt{b-a} \cdot \text{poly}(\log {}^1\!/\!_\delta)$ that approximates e^{-x} on the interval $[a, b]$, up to an error of $\delta \cdot e^{-a}$ for any $\delta > 0$. Our approach is to approximate $(1 + {}^x\!/\!_k)^{-1}$ on the interval $[a, b]$, by a polynomial q of degree l, and then compose the polynomial $p_k^\star$ from the SSV result with q, to obtain $p_k^\star(q(x))$ which is a polynomial of degree $k \cdot l$ approximating e^{-x} over $[a, b]$. Thus, we are looking for polynomials q that minimize $|q(x) - {}^1\!/\!_x|$ over $[1 + {}^a\!/\!_k, 1 + {}^b\!/\!_k]$. Slightly modifying the optimization, we consider polynomials q that minimize

$|x \cdot q(x) - 1|$ over $[1 + {}^a\!/\!_k, 1 + {}^b\!/\!_k]$. In Section 6.3, we show that the solution to this modified optimization can be derived from the well-known Chebyshev polynomials. For the right choice of k and l, the composition of the two polynomials approximates e^{-x} to within an error of $\delta \cdot e^{-a}$ over $[a, b]$, and has degree $\sqrt{b-a} \cdot \mathrm{poly}(\log {}^1\!/\!_\delta)$. To bound the error in the composition step, we need to bound the sum of absolute values of coefficients of $p_k^\star$, which we achieve by rewriting $p_k^\star$ as an interpolation polynomial. The details appear in Section 6.3.

Lower Bound.

We prove that the square-root dependence on $b - a$ of the required degree is optimal. The proof is simple and we give the details here: Using a theorem of Markov from approximation theory (see [Che66]), we show that, any polynomial approximating e^{-x} over the interval $[a, b]$ up to an error of $\delta \cdot e^{-a}$, for some constant δ small enough, must have degree of the order of $\sqrt{b-a}$. Markov's theorem says that the absolute value of the derivative of a univariate polynomial p of degree k, which lives in a box of height h over an interval of width w, is upper bounded by ${}^{d^2 h}\!/\!_w$. Let p_k be a polynomial of degree k that $\delta \cdot e^{-a}$-approximates e^{-x} in the interval $[a, b]$. If b is large enough and, δ a small enough constant, then one can get a lower bound of $\Omega(e^{-a})$ on the derivative of p_k using the Mean Value Theorem. Also, one can obtain an upper bound of $O(e^{-a})$ on the height of the box in which p_k lives. Both these bounds use the fact that p_k approximates e^{-x} and is $\delta \cdot e^{-a}$ close to it. Since the width of the box is $b-a$, these two facts, along with Markov's theorem, immediately imply a lower bound of $\Omega(\sqrt{b-a})$ on k. This shows that our upper bound is tight up to a factor of $\mathrm{poly}(\log {}^1\!/\!_\delta)$.

6.1.1 Preliminaries

Given an interval $[a, b]$, we are looking for low-degree polynomials (or rational functions) that approximate the function e^{-x} in the sup norm over the interval.

Definition 6.2 (δ-Approximation)**.** *A function g is called a δ-approximation to a function f over an interval $\mathcal{I}$, if,* $\sup_{x\in\mathcal{I}} |f(x) - g(x)| \leq \delta$*.*

Such approximations are known as *uniform approximations* in approximation theory and have been studied quite extensively. We will consider both finite and infinite intervals $\mathcal{I}$.

For any positive integer k, let Σ_k denote the set of all degree k polynomials. We also need to define the ℓ_1 norm of a polynomial.

Definition 6.3 (ℓ_1 Norm of a Polynomial)**.** *Given a degree k polynomial $p \stackrel{\text{def}}{=} \sum_{i=0}^{k} a_i \cdot x^i$, the ℓ_1 norm of p, denoted as $\|p\|_1$ is defined as* $\|p\|_1 = \sum_{i\geq 0}^{k} |a_i|$*.*

6.2 Known Approximation Results and Discussion

Approximating the exponential is a classic question in Approximation Theory, see *e.g.* [Che66]. We ask the following question:

Question: *Given $\delta \leq 1$ and $a < b$, what is the smallest degree of a polynomial that is an $\delta \cdot e^{-a}$-approximation to e^{-x} over the interval $[a, b]$?*

This qustion has been studied in the following form: Given λ, what is the best low degree polynomial (or rational function) approximation to $e^{\lambda x}$ over $[-1, 1]$? In a sense, these questions are equivalent, as is shown by the following lemma, proved using a linear shift of variables. A proof is included in Section 6.5.

Lemma 6.4 (Linear Variable Shift for Approximation)**.** *For any non-negative integer k and l and real numbers $b > a$,*

$$\min_{p_k\in\Sigma_k, q_l\in\Sigma_l} \sup_{t\in[a,b]} \left| e^{-t} - \frac{p_k(t)}{q_l(t)} \right| = e^{-\frac{b+a}{2}} \cdot \min_{p_k\in\Sigma_k, q_l\in\Sigma_l} \sup_{x\in[-1,1]} \left| e^{\frac{(b-a)}{2}x} - \frac{p_k(x)}{q_l(x)} \right|.$$

Using the above lemma, we can translate the known results to our setting. As a starting point, we could approximate e^{-x} by truncating the Taylor series expansion of the exponential. We state the approximation achieved in the following lemma. A proof is included in Section 6.5.

Lemma 6.5 (Taylor Approximation)**.** *The degree k polynomial obtained by truncating Taylor's expansion of e^{-t} around the point ${}^{b+a}/_2$ is a uniform approximation to e^{-t} on the interval $[a,b]$ up to an error of*

$$e^{-\frac{b+a}{2}} \cdot \sum_{i=k+1}^{\infty} \frac{1}{i!} \left(\frac{b-a}{2}\right)^i,$$

which is smaller than $\delta \cdot e^{-\frac{b+a}{2}}$ for $k \geq \max\{\frac{e^2(b-a)}{2}, \log {}^1/_\delta\}$

A lower bound is known in the case where the size of the interval is fixed, *i.e.*, $b-a=O(1)$.

Proposition 6.6 (Lower Bound for Polynomials over Fixed Interval, [Al′59, Saf73])**.** *For any $a,b \in \mathbb{R}$ such that $b-a$ is* fixed*, as k goes to infinity, the best approximation achieved by a degree k polynomial has error*

$$(1+o(1))\frac{1}{(k+1)!}\left(\frac{b-a}{2}\right)^{k+1} \cdot e^{-\frac{b+a}{2}}.$$

In essence, this theorem states that if the *size of the interval is fixed*, the polynomials obtained by truncating the Taylor series expansion achieve asymptotically the least error possible and hence, the best asymptotic degree for achieving a $\delta \cdot e^{-\frac{b+a}{2}}$-approximation. In addition, Saff [Saf73] also shows that if, instead of polynomials, we allow rational functions where the degree of the denominator is a constant, the degree required for achieving a $\delta \cdot e^{-\frac{b+a}{2}}$-approximation changes at most by a constant.

These results indicate that tight bounds on the answer to our question should be already known. In fact, at first thought, the optimality of the Taylor series polynomials seems to be in contradiction with our results. However, note the two important differences:

1. The error in our theorem is $e^{-a} \cdot \delta$, whereas, the Taylor series approximation involves error $e^{-\frac{b+a}{2}} \cdot \delta$, which is *smaller*, and hence requires larger degree.

2. Moreover, the lower bound applies only when the length of the interval $(b - a)$ is constant, in which case, our theorem says that the required degree is $\mathrm{poly}(\log {}^1\!/\!_\delta)$, which is $\Omega(\log {}^1\!/\!_\delta)$, in accordance with the lower bound.

If the length of the interval $[a, b]$ grows unbounded (as is the case for our applications to the Balanced Separator problem in the previous sections), the main advantage of using polynomials from Theorem 6.1 is the improvement in the degree from linear in $(b - a)$ to $\sqrt{b - a}$.

6.3 Proof of Upper Bound

In this section, we use Theorem 4.11 by Saff, Schönhage, and Varga [SSV75], to give a proof of the upper bound result in Theorem 6.1. We restate Theorem 4.11 for completeness.

Theorem 6.7 (Theorem 4.11 Restated, [SSV75]). *There exists constants $c_1 \geq 1$ and k_0 such that, for any positive integer $k \geq k_0$, there exists a polynomial $p_k^\star(x)$ of degree k such that $p_k^\star(0) = 0$, and,*

$$\sup_{t \in (0,1]} \left| e^{-k/t + k} - p_k^\star(t) \right| = \sup_{x \in [0,\infty)} \left| e^{-x} - p_k^\star\left((1 + x/k)^{-1} \right) \right| \leq c_1 k \cdot 2^{-k} \ .$$

Our approach is to compose the polynomial $p_k^\star$ given by Theorem 6.7 with polynomials approximating $(1 + x/k)^{-1}$, to construct polynomials approximating e^{-x}. We first show the existence of polynomials approximating x^{-1}, and from these polynomials, we will derive approximations to $(1 + x/k)^{-1}$.

Our goal is to find a polynomial q of degree k, that minimizes $\sup_{x \in [a,b]} |q(x) - {}^1\!/\!_x|$. We slightly modify this optimization to minimizing $\sup_{x \in [a,b]} |x \cdot q(x) - 1|$. Note that

$x \cdot q(x) - 1$ is a polynomial of degree $k+1$ which evaluates to -1 at $x = 0$, and conversely every polynomial that evaluates to -1 at 0 can be written as $x \cdot q(x) - 1$ for some q. So, this is equivalent to minimizing $\sup_{x\in[a,b]} |q_1(x)|$, for a degree $k+1$ polynomial q_1 such that $q_1(0) = -1$. By scaling and multiplying by -1, this is equivalent to finding a polynomial q_2, that maximizes $q_2(0)$, subject to $\sup_{x\in[a,b]} |q_2(x)| \leq 1$. If we shift and scale the interval $[a, b]$ to $[-1, 1]$, the optimal solution to this problem is known to be given by the well known Chebyshev polynomials. We put all these ideas together to prove the following lemma. A complete proof is included in Section 6.5.

Lemma 6.8 (Approximating x^{-1})**.** *For every $\epsilon > 0$, $b > a > 0$, there exists a polynomial $q_{a,b,\epsilon}(x)$ of degree $\left\lceil \sqrt{\frac{b}{a}} \log \frac{2}{\epsilon} \right\rceil$ such that $\sup_{x\in[a,b]} |x \cdot q_{a,b,\epsilon}(x) - 1| \leq \epsilon$.*

As a simple corollary, we can approximate $(1 + {}^{x}\!/_{k})^{-1}$, or rather generally, $(1 + \nu x)^{-1}$ for some $\nu > 0$, by polynomials. A proof is included in Section 6.5.

Corollary 6.9 (Approximating $(1 + \nu x)^{-1}$)**.** *For every $\nu > 0, \epsilon > 0$ and $b > a \geq 0$, there exists a polynomial $q^\star_{\nu,a,b,\epsilon}(x)$ of degree $\left\lceil \sqrt{\frac{1+\nu b}{1+\nu a}} \log \frac{2}{\epsilon} \right\rceil$ such that $\sup_{x\in[a,b]} |(1 + \nu x) \cdot q^\star_{\nu,a,b,\epsilon}(x) - 1| \leq \epsilon$.*

The above corollary implies that the expression $(1 + \nu x) \cdot q^\star$ is within $1 \pm \varepsilon$ on $[a, b]$. If ε is small, for a small positive integer t, $[(1 + \nu x) \cdot q^\star]^t$ should be at most $1 \pm O(t\varepsilon)$. The following lemma, proved using the binomial theorem proves this formally. A proof is included in Section 6.5.

Lemma 6.10 (Approximating $(1 + \nu x)^{-t}$)**.** *For all real $\epsilon > 0$, $b > a \geq 0$ and positive integer t; if $t\epsilon \leq 1$, then,*

$$\sup_{x\in[a,b]} |((1 + \nu x) \cdot q^\star_{\nu,a,b,\epsilon}(x))^t - 1| \leq 2t\epsilon,$$

where $q^\star_{\nu,a,b,\epsilon}$ is the polynomial given by Corollary 6.9.

Since $q^\star$ is an approximation to $(1+ {}^{x}\!/_{k})^{-1}$, in order to bound the error for the composition $p_k^\star(q^\star)$, we need to bound how the value of the polynomial $p_k^\star$ changes on small perturbations in the input. We will use the following crude bound in terms of the ℓ_1 norm of the polynomial.

Lemma 6.11 (Error in Polynomial)**.** *For any polynomial p of degree k, and any $x, y \in \mathbb{R}$, $|p(x) - p(y)| \leq \|p\|_1 \cdot \max_{0 \leq i \leq k} |x^i - y^i|$.*

In order to utilize the above lemma, we will need a bound on the ℓ_1 norm of $p_k^\star$, which is provided by Lemma 4.21, that bounds the ℓ_1 norm of any polynomial in $(1+ {}^{x}\!/_{k})^{-1}$ that approximates the exponential function and has no constant term. We restate the lemma here for completeness.

Lemma 6.12 (ℓ_1-norm Bound. Lemma 4.21 Restated)**.** *Given a polynomial p of degree k such that $p(0) = 0$ and*

$$\sup_{t \in (0,1]} \left| e^{-{}^{k}\!/_{t}+k} - p(t) \right| = \sup_{x \in [0,\infty)} \left| e^{-x} - p\left((1+ {}^{x}\!/_{k})^{-1}\right) \right| \leq 1 \ ,$$

we must have $\|p\|_1 \leq (2k)^{k+1}$.

We can now analyze the error in approximating e^{-x} by the polynomial $p_k^\star(q^\star)$ and give a proof for Theorem 6.1.

Proof. Given $\delta \leq 1$, let $k = \max\{k_0, \log_2 {}^{4c_1}\!/_{\delta} + 2\log_2\log_2 {}^{4c_1}\!/_{\delta}\} = O\left(\log {}^{1}\!/_{\delta}\right)$, where k_0, c_1 are the constants given by Theorem 6.7. Moreover, $p_k^\star$ is the degree k polynomial given by Theorem 6.7, which gives, $p_k^\star(0) = 0$, and,

$$\begin{aligned} \sup_{x \in [0,\infty)} \left| e^{-x} - p_k^\star\left((1+ {}^{x}\!/_{k})^{-1}\right) \right| &\leq \frac{\delta}{4} \cdot \frac{\log_2 {}^{4c_1}\!/_{\delta} + 2\log_2\log_2 {}^{4c_1}\!/_{\delta}}{(\log_2 {}^{4c_1}\!/_{\delta})^2} \\ &\leq \frac{\delta}{4} \cdot \frac{1}{\log_2 {}^{4c_1}\!/_{\delta}} \left(1 + 2 \cdot \frac{\log_2\log_2 {}^{4c_1}\!/_{\delta}}{\log_2 {}^{4c_1}\!/_{\delta}}\right) \leq \frac{\delta}{4} \cdot \frac{1}{2} \cdot 3 \leq \frac{\delta}{2} \ , \end{aligned} \tag{6.1}$$

where the last inequality uses $\delta \le 1 \le c_1$ and $\log_2 x \le x, \forall x \ge 0$. Thus, we can use Lemma 6.12 to conclude that $\|p_k^\star\|_1 \le (2k)^{k+1}$.

Let $\nu \overset{\text{def}}{=} 1/k$. Define ε as $\varepsilon \overset{\text{def}}{=} \frac{\delta}{2(2k)^{k+2}}$. Let $p_{a,b,\delta}(x) \overset{\text{def}}{=} e^{-a} \cdot p_k^\star\left(q^\star_{\nu,0,b-a,\epsilon}(x-a)\right)$, where $q^\star_{\nu,0,b-a,\epsilon}$ is the polynomial of degree $\left\lceil \sqrt{1+\nu(b-a)}\log\frac{2}{\epsilon}\right\rceil$ given by Corollary 6.9. Observe that $p_{a,b,\delta}(x)$ is a polynomial of degree that is the product of the degrees of $p_k^\star$ and $q^\star_{\nu,0,b-a,\epsilon}$, *i.e.*, $k\left\lceil \sqrt{1+\nu(b-a)}\log\frac{2}{\epsilon}\right\rceil$. Also note that $k\epsilon < 1$ and hence we can use Lemma 6.10. We show that $p_{a,b,\delta}$ is a uniform δ-approximation to e^{-x} on the interval $[a,b]$.

$$\begin{aligned}
&\sup_{x\in[a,b]} \left|e^{-x} - p_{a,b,\delta}(x)\right| = e^{-a}\cdot \sup_{x\in[a,b]} \left|e^{-(x-a)} - e^{a}\cdot p_{a,b,\delta}(x)\right| \\
&\quad= e^{-a}\cdot \sup_{y\in[0,b-a]} \left|e^{-y} - e^{a}\cdot p_{a,b,\delta}(y+a)\right| \\
&\quad\overset{\text{by def}}{=} e^{-a}\cdot \sup_{y\in[0,b-a]} \left|e^{-y} - p_k^\star\left(q^\star_{\nu,0,b-a,\epsilon}(y)\right)\right| \\
&\quad\overset{\Delta-\text{ineq.}}{\le} e^{-a} \sup_{y\in[0,b-a]} \left(\left|e^{-y} - p_k^\star\left((1+\nu y)^{-1}\right)\right| + \left|p_k^\star\left((1+\nu y)^{-1}\right) - p_k^\star\left(q^\star_{\nu,0,b-a,\epsilon}(y)\right)\right|\right) \\
&\quad\le e^{-a}\cdot \sup_{y\in[0,b-a]} \left|e^{-y} - p_k^\star\left((1+\nu y)^{-1}\right)\right| \\
&\qquad\quad + e^{-a}\cdot \sup_{y\in[0,b-a]} \left|p_k^\star\left((1+\nu y)^{-1}\right) - p_k^\star\left(q^\star_{\nu,0,b-a,\epsilon}(y)\right)\right| \\
&\quad\overset{\textit{Lem. } 6.11}{\le} e^{-a}\cdot \sup_{y\in[0,\infty)} \left|e^{-y} - p_k^\star\left((1+\nu y)^{-1}\right)\right| \\
&\qquad + e^{-a}\cdot \|p_k^\star\|_1 \cdot \max_{0\le i\le k} \sup_{y\in[0,b-a]} \left|(1+\nu y)^{-i} - \left(q^\star_{\nu,0,b-a,\epsilon}(y)\right)^i\right| \\
&\quad\overset{\textit{Eq. } (6.1)}{\le} e^{-a}\cdot\frac{\delta}{2} + e^{-a}\cdot \|p_k^\star\|_1 \cdot \max_{0\le i\le k} \sup_{y\in[0,b-a]} (1+\nu y)^{-i}\left|1 - \left((1+\nu y)\cdot q^\star_{\nu,0,b-a,\epsilon}(y)\right)^i\right| \\
&\quad\overset{\textit{Lem. } 6.10,6.12}{\le} e^{-a}\cdot\frac{\delta}{2} + 2k\epsilon\cdot e^{-a}\cdot (2k)^{k+1} \le \delta\cdot e^{-a}
\end{aligned}$$

The degree of the polynomial $p_{a,b,\delta}$ is

$$k\left\lceil \sqrt{1+\nu(b-a)}\log\frac{2}{\epsilon}\right\rceil = O\left(\sqrt{k^2+k(b-a)}\cdot\left(k\log k + \log 1/\delta\right)\right)$$

$$= O\left(\sqrt{\max\{\log {}^1/\delta, (b-a)\}} \cdot (\log {}^1/\delta)^{3/2} \cdot \log\log {}^1/\delta\right).$$

□

6.4 Proof of Lower Bound

In this section, we will use the following well known theorem of Markov from approximation theory to give a proof of the lower bound result in Theorem 6.1.

Theorem 6.13 (Markov, See [Che66]). *Let $p : \mathbb{R} \to \mathbb{R}$ be a univariate polynomial of degree d such that any real number $a_1 \le x \le a_2$, satisfies $b_1 \le p(x) \le b_2$. Then, for all $a_1 \le x \le a_2$, the derivative of p satisfies $|p'(x)| \le d^2 \cdot \frac{b_2 - b_1}{a_2 - a_1}$.*

The idea is to first use the uniform approximation bound to bound the value of the polynomial within the interval of approximation. Next, we use the approximation bound and the *Mean Value theorem* to show that there must exist a point t in the interval where $|p'(t)|$ is large. We plug both these bounds into Markov's theorem to deduce our lower bound.

Proof. Suppose p is a degree k polynomial that is a uniform approximation to e^{-x} over the interval $[a, b]$ up to an error of $\delta \cdot e^{-a}$. For any $x \in [a, b]$, this bounds the values p can take at x. Since p is a uniform approximation to e^{-x} over $[a, b]$ up to an error of $\delta \cdot e^{-a}$, we know that for all $x \in [a, b]$, $e^{-x} - \delta \cdot e^{-a} \le p(x) \le e^{-x} + \delta \cdot e^{-a}$. Thus, $\max_{x \in [a,b]} p(x) \le e^{-a} + \delta \cdot e^{-a}$ and $\min_{x \in [a,b]} p(x) \ge e^{-b} - \delta \cdot e^{-a}$.

Assume that $\delta \le {}^1/8$, and $b \ge a + \log_e 4 \ge a + \log_e {}^2/(1-4\delta)$. Applying the *Mean Value theorem* on the interval $[a, a + \log_e {}^2/(1-4\delta)]$, we know that there exists $t \in [a, a + \log_e {}^2/(1-4\delta)]$, such that,

$$|p'(t)| = \left|\frac{p(a + \log_e {}^2/(1-4\delta)) - p(a)}{\log_e {}^2/(1-4\delta)}\right| \ge \frac{(e^{-a} - \delta \cdot e^{-a}) - (e^{-a - \log_e {}^2/1-4\delta} + \delta \cdot e^{-a})}{\log_e {}^2/(1-4\delta)}$$

$$\geq e^{-a}\frac{1-2\delta-\frac{(1-4\delta)}{2}}{\log_e {}^2\!/\!{}_{(1-4\delta)}} = e^{-a}\frac{1}{2\log_e {}^2\!/\!{}_{(1-4\delta)}}.$$

We plug this in Markov's theorem (Theorem 6.13) stated above to deduce,

$$e^{-a}\frac{1}{2\log_e {}^2\!/\!{}_{(1-4\delta)}} \leq k^2\frac{(e^{-a}+\delta\cdot e^{-a})-(e^{-b}-\delta\cdot e^{-a})}{b-a} \leq k^2\cdot e^{-a}\cdot\frac{1+2\delta}{b-a}\ .$$

Rearranging, we get,

$$k \geq \sqrt{\frac{b-a}{2\cdot(1+2\delta)\cdot\log_e {}^2\!/\!{}_{(1-4\delta)}}} \geq \sqrt{\frac{b-a}{2\cdot {}^5\!/\!{}_4\cdot\log_e 4}} \geq \frac{1}{2}\cdot\sqrt{b-a},$$

where the second inequality uses $\delta \leq {}^1\!/\!{}_8$. □

6.5 Remaining Proofs

Lemma 6.14 (Lemma 6.4 Restated)**.** *For any non-negative integer k and l and real numbers $b > a$,*

$$\min_{p_k\in\Sigma_k,q_l\in\Sigma_l}\sup_{t\in[a,b]}\left|e^{-t}-\frac{p_k(t)}{q_l(t)}\right| = e^{-\frac{b+a}{2}}\cdot\min_{p_k\in\Sigma_k,q_l\in\Sigma_l}\sup_{x\in[-1,1]}\left|e^{\frac{(b-a)}{2}x}-\frac{p_k(x)}{q_l(x)}\right|$$

Proof. Using the substitution $t \overset{\text{def}}{=} \frac{(b+a)}{2}-\frac{(b-a)}{2}x$,

$$\begin{aligned}\min_{p_k\in\Sigma_k,q_l\in\Sigma_l}\sup_{t\in[a,b]}\left|e^{-t}-\frac{p_k(t)}{q_l(t)}\right| &= \min_{p_k\in\Sigma_k,q_l\in\Sigma_l}\sup_{x\in[1,-1]}\left|e^{-\frac{(b+a)}{2}+\frac{(b-a)}{2}x}-\frac{p_k\left({}^{(b+a)}\!/\!{}_2-{}^{(b-a)}\!/\!{}_{2}x\right)}{q_l\left({}^{(b+a)}\!/\!{}_2-{}^{(b-a)}\!/\!{}_{2}x\right)}\right|\\ &= e^{-\frac{b+a}{2}}\cdot\min_{p'_k\in\Sigma_k,q'_l\in\Sigma_l}\sup_{x\in[-1,1]}\left|e^{\frac{(b-a)}{2}x}-\frac{p'_k(x)}{q'_l(x)}\right|.\end{aligned}$$

□

Lemma 6.15 (Lemma 6.5 Restated)**.** *The degree k polynomial obtained by truncating Taylor's expansion of e^{-t} around the point ${}^{b+a}\!/\!{}_2$ is a uniform approximation to e^{-t} on* *the interval $[a, b]$ up to an error of*

$$e^{-\frac{b+a}{2}} \cdot \sum_{i=k+1}^{\infty} \frac{1}{i!} \left(\frac{(b-a)}{2}\right)^i,$$

which is smaller than δ for $k \geq \max\{\frac{e^2(b-a)}{2}, \log {}^1\!/\!_\delta\}$.

Proof. Let $q_k(t)$ be the degree k Taylor approximation of the function e^{-t} around the point ${}^{(b+a)}\!/\!_2$, *i.e.*, $q_k(t) \overset{\text{def}}{=} e^{-\frac{(b+a)}{2}} \sum_{i=0}^{k} \frac{1}{i!} \left(t - \frac{(b+a)}{2}\right)^i$.

$$\sup_{t \in [a,b]} |e^{-t} - q_k(t)| = \sup_{t \in [a,b]} e^{-\frac{b+a}{2}} \cdot \left| \sum_{i=k+1}^{\infty} \frac{1}{i!} \left(t - \frac{(b+a)}{2}\right)^i \right| = e^{-\frac{b+a}{2}} \cdot \sum_{i=k+1}^{\infty} \frac{1}{i!} \left(\frac{(b-a)}{2}\right)^i.$$

Using the inequality $i! > \left(\frac{i}{e}\right)^i$, for all i, and assuming $k \geq \frac{e^2(b-a)}{2}$, we get,

$$\sum_{i=k+1}^{\infty} \frac{1}{i!} \left(\frac{(b-a)}{2}\right)^i \leq \sum_{i=k+1}^{\infty} \left(\frac{e(b-a)}{2i}\right)^i \leq \sum_{i=k+1}^{\infty} e^{-i} = \frac{1}{e-1} e^{-k},$$

which is smaller than δ for $k \geq \log {}^1\!/\!_\delta$. ◻

Lemma 6.16 (Lemma 6.8 Restated)**.** *For every $\epsilon > 0$, $b > a > 0$, there exists a polynomial $q_{a,b,\epsilon}(x)$ of degree $\left\lceil \sqrt{\frac{b}{a}} \log \frac{2}{\epsilon} \right\rceil$ such that*

$$\sup_{x \in [a,b]} |x \cdot q_{a,b,\epsilon}(x) - 1| \leq \epsilon.$$

Proof. If $T_{k+1}(x)$ denotes the degree $k+1$ Chebyshev polynomial, consider the function,

$$q_{a,b,\epsilon}(x) \overset{\text{def}}{=} \frac{1}{x} \left(1 - \frac{T_{k+1}\left(\frac{b+a-2x}{b-a}\right)}{T_{k+1}\left(\frac{b+a}{b-a}\right)}\right).$$

First, we need to prove that the above expression is a polynomial. Clearly $1 - \frac{T_{k+1}\left(\frac{b+a-2x}{b-a}\right)}{T_{k+1}\left(\frac{b+a}{b-a}\right)}$ is a polynomial and evaluates to 0 at $x = 0$. Thus, it must have x as a factor. Thus $q_{a,b,\epsilon}$ is a polynomial of degree k. Let $\kappa = {}^{b}\!/_{a}$ and note that $\kappa > 1$. Thus,

$$\begin{aligned}
\sup_{x\in[a,b]} |x \cdot q_{a,b,\epsilon}(x) - 1| &= \sup_{x\in[a,b]} \left| \frac{T_{k+1}\left(\frac{b+a-2x}{b-a}\right)}{T_{k+1}\left(\frac{b+a}{b-a}\right)} \right| \\
&\le T_{k+1}\left(\frac{b+a}{b-a}\right)^{-1} && \text{(Since } |T_{k+1}(y)| \le 1 \text{ for } |y| \le 1\text{)} \\
&= 2\left(\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)^{k+1} + \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{k+1}\right)^{-1} && \text{(By def.)} \\
&\le 2\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)^{-k-1} && \text{(Each term is positive since } \sqrt{\kappa} > 1\text{)} \\
&= 2\left(\frac{1 - {}^{1}\!/_{\sqrt{\kappa}}}{1 + {}^{1}\!/_{\sqrt{\kappa}}}\right)^{k+1} \\
&\le 2 \cdot (1 - {}^{1}\!/_{\sqrt{\kappa}})^{k+1} \le 2 \cdot e^{-(k+1)/\sqrt{\kappa}} \le \epsilon,
\end{aligned}$$

for $k = \left\lceil \sqrt{\kappa} \log \frac{2}{\epsilon} \right\rceil$. The first inequality follows from the fact that $|T_{k+1}(x)| \le 1$ for all $|x| \le 1$. □

Corollary 6.17 (Corollary 6.9 Restated)**.** *For every $\nu > 0, \epsilon > 0$ and $b > a \ge 0$, there exists a polynomial $q^{\star}_{\nu,a,b,\epsilon}(x)$ of degree $\left\lceil \sqrt{\frac{1+\nu b}{1+\nu a}} \log \frac{2}{\epsilon} \right\rceil$ such that*

$$\sup_{x\in[a,b]} |(1+\nu x) \cdot q^{\star}_{\nu,a,b,\epsilon}(x) - 1| \le \epsilon.$$

Proof. Consider the polynomial $q^{\star}_{\nu,a,b,\epsilon}(x) \overset{\text{def}}{=} q_{1+\nu a,1+\nu b,\epsilon}\,(1+\nu x)$, where $q_{1+\nu a,1+\nu b,\epsilon}$ is given by the previous lemma.

$$\begin{aligned}
\sup_{x\in[a,b]} \left|(1+\nu x) \cdot q^{\star}_{\nu,a,b,\epsilon}(x) - 1\right| &= \sup_{x\in[a,b]} |(1+\nu x) \cdot q_{1+\nu a,1+\nu b,\epsilon}\,(1+\nu x) - 1| \\
&\overset{t \overset{\text{def}}{=} 1+\nu x}{=} \sup_{t\in[1+\nu a,1+\nu b]} |t \cdot q_{1+\nu a,1+\nu b,\epsilon}\,(t) - 1| \overset{\text{Lem. 6.8}}{\le} \epsilon.
\end{aligned}$$

Since $1+\nu x$ is a linear transformation, the degree of $q^{\star}_{\nu,a,b,\epsilon}$ is the same as that of $q_{1+\nu a,1+\nu b,\epsilon}$, which is, $\left\lceil \sqrt{\frac{1+\nu b}{1+\nu a}} \log \frac{2}{\epsilon} \right\rceil$. □

Lemma 6.18 (Lemma 6.10 Restated)**.** *For all real $\epsilon > 0$, $b > a \geq 0$ and positive integer t; if $t\epsilon \leq 1$, then,*

$$\sup_{x\in[a,b]} |((1+\nu x)\cdot q^{\star}_{\nu,a,b,\epsilon}(x))^t - 1| \leq 2t\epsilon,$$

where $q^{\star}_{\nu,a,b,\epsilon}$ is the polynomial given by Corollary 6.9.

Proof. We write the expression $(1+\nu x)\cdot q^{\star}_{\nu,a,b,\epsilon}(x)$ as 1 plus an error term and then use the Binomial Theorem to expand the t^{th} power.

$$\begin{aligned}
\sup_{x\in[a,b]} |((1+\nu x)\cdot q^{\star}_{\nu,a,b,\epsilon}(x))^t - 1| &= \sup_{x\in[a,b]} \left|\left(1-\left[1-(1+\nu x)\cdot q^{\star}_{\nu,a,b,\epsilon}(x)\right]\right)^t - 1\right| \\
&= \sup_{x\in[a,b]} \left|\sum_{i=1}^{t}\binom{t}{i}\left(1-(1+\nu x)\cdot q^{\star}_{\nu,a,b,\epsilon}(x)\right)^i\right| \\
&\leq \sup_{x\in[a,b]} \sum_{i=1}^{t}\binom{t}{i}\left|1-(1+\nu x)\cdot q^{\star}_{\nu,a,b,\epsilon}(x)\right|^i \\
&\leq \sum_{i=1}^{t}\binom{t}{i}\sup_{x\in[a,b]}\left|1-(1+\nu x)\cdot q^{\star}_{\nu,a,b,\epsilon}(x)\right|^i \\
&\overset{\text{Cor. 6.9}}{\leq} \sum_{i=1}^{t}\binom{t}{i}\epsilon^i = (1+\epsilon)^t - 1 \\
&\leq \exp(t\epsilon) - 1 \leq 1 + t\epsilon + (t\epsilon)^2 - 1 \leq 2t\epsilon,
\end{aligned}$$

where the second last inequality uses $e^x \leq 1 + x + x^2$ for $x \in [0,1]$. [1] ☐

Lemma 6.19 (Lemma 6.11 Restated)**.** *For any polynomial p of degree k, and any $x, y \in \mathbb{R}$*

$$|p(x) - p(y)| \leq \|p\|_1 \cdot \max_{0\leq i\leq k} |x^i - y^i|.$$

[1]For $x \in [0,1]$, $e^x = \sum_{i\geq 0}\frac{x^i}{i!} = 1 + x + x^2\left(\frac{1}{2!}+\frac{x}{3!}+\ldots\right) \leq 1 + x + x^2\left(\frac{1}{2}+\frac{x}{2^2}+\frac{x}{2^3}+\ldots\right) \leq 1 + x + x^2$

Proof. Suppose $p(t)$ is the polynomial $\sum_{i=0}^{k} a_i \cdot t^i$, where $a_i \in \mathbb{R}$. Then,

$$\begin{aligned}|p(x) - p(y)| &= \left|\sum_{i=0}^{k} a_i \cdot x^i - \sum_{i=0}^{k} a_i \cdot y^i\right| \leq \sum_{i=0}^{k} |a_i||x^i - y^i| \\ &\leq \left(\sum_{i=0}^{k} |a_i|\right) \max_{0 \leq i \leq k} |x^i - y^i| = \|p\|_1 \cdot \max_{0 \leq i \leq k} |x^i - y^i|.\end{aligned}$$

□

Notes

The material presented in this chapter is based on the paper "Approximating the Exponential, the Lanczos Method, and an $\tilde{O}(m)$-Time Spectral Algorithm for Balanced Separator" [OSV12], joint with Lorenzo Orecchia and Nisheeth Vishnoi, that appeared at STOC 2012. A full version of the paper is available on arxiv [OSV11].

Chapter 7

Optimal Inapproximability without UGC

In this chapter, we present a brief history of the field of Hardness of Approximation, and our contribution to it. We will describe the problems of interest and our results for them, along with a comparison to previous works.

7.1 Hardness of Approximation – A quick history

Over the last two decades, the field of hardness of approximation has had several notable successes. The PCP [1] theorem [FGL+96, AS98, ALM+98] ruled out polynomial time approximation schemes for several problems including Max-3SAT and Minimum Vertex Cover, assuming **P** $\neq$ **NP**. A framework for proving such hardness results based on composing an outer PCP based on Raz's Parallel Repetition Theorem [Raz98], with an inner PCP based on the Long Code (introduced by Bellare *et al.* [BGS98]) proved to be very successful. This led to optimal inapproximability results for Max-Clique [Hås99], Max-3XOR, and Max-3SAT [Hås01].

[1] PCP stands for Probabilistically Checkable Proofs.

For several problems such as Minimum Vertex Cover and Max-Cut, this led to improved inapproximability results, but they were still far from optimal. Towards obtaining optimal inapproximability results, Khot introduced the Unique Games Conjecture (UGC) [Kho02]. This conjecture proposed that a specific form of the outer PCP was still **NP**-hard. Assuming the UGC led to optimal inapproximability results for Minimum Vertex Cover [KR08], and Max-Cut [KKMO07, MOO10]. Surprisingly, Raghavendra [Rag08] proved that assuming the UGC leads to optimal inapproximability results for all *Constraint Satisfaction Problems* (CSPs – a large class of problems that includes Max-Cut, Max-2SAT, Max-3XOR, Max-3SAT etc.). By now, the UGC is known to imply optimal inapproximability results for a host of other problems, e.g. some classic scheduling problems [BK09, BK10], strict CSPs [KMTV11], ordering CSPs [GMR08, GHM$^+$11].

Given that the UGC implies so many optimal inapproximability results, we would like to believe that the conjecture is true. However, we must ask what evidence do we have supporting the conjecture. There have been several algorithmic attacks on the conjecture (e.g. [Kho02, Tre05, GT06, CMM06a, CMM06b]), but they fail to refute the conjecture. Beginning with the work of Khot and Vishnoi [KV05], a sequence of works [RS09, KS09, BGH$^+$11] showed that a large class of algorithms based on semidefinite programming (SDP) hierarchies cannot refute the conjecture, by constructing hard instances for these algorithms. These results supported the possibility of the UGC being true.

However, recently it was shown that there are sub-exponential time algorithm for Unique Games [ABS10, BRS11, GS11]. This is in contrast to problems such as 3SAT that, we suspect, require $2^{\Theta(n)}$ time [IPZ01]. Further, it was shown that the known hard instances from [KV05, RS09, BGH$^+$11] can be solved in polynomial time by using a more powerful SDP hierarchy, the Lasserre hierarchy [BBH$^+$12]. Thus, overall, the conjecture is poised delicately and we lack strong evidence in either direction.

Given the uncertain status of the UGC, we explore if some of these new inapproximability results can be achieved without assuming the conjecture. We make progress towards this goal for variants of the Minimum Vertex Cover problem on hypergraphs, and some scheduling problems. We discuss these problems in the next few sections.

7.2 Hypergraph Vertex Cover

A k-uniform hypergraph $G = (V, E)$ consists of a set of vertices V and a collection of hyperedges $E \subseteq 2^V$ such that each hyperedge $e \in E$ contains exactly k vertices. A subset of vertices $\mathcal{V} \subseteq V$ such that every hyperedge $e \in E$ contains at least one vertex from $\mathcal{V}$, *i.e.*, $e \cap \mathcal{V} \neq \emptyset$, is called a *Vertex Cover* for G. Equivalently, a vertex cover is a hitting set for the collection of hyperedges E. Thus, the complement of a vertex cover is a subset of vertices $\mathcal{I}$ such that no hyperedge $e \in E$ is contained inside $\mathcal{I}$, *i.e.*, $e \nsubseteq \mathcal{I}$. Such a set is called an *Independent Set*.

The Minimum Vertex Cover problem on k-uniform hypergraphs requires us to compute the minimum size of a vertex cover in a given k-uniform hypergraph G. We denote this problem by k-HypVC. It is an extremely well studied combinatorial optimization problem, especially on graphs ($k = 2$), and is known to be **NP**-hard for all $k \geq 2$. Indeed, the Minimum Vertex Cover problem on graphs was one of Karp's original 21 **NP**-complete problems [Kar72].

On the other hand, the simple greedy algorithm that picks a maximal collection of disjoint hyperedges, and outputs the set of all vertices in the edges gives a k-approximation. A k-approximation can also be obtained using the standard Linear Programming (LP) relaxation for the problem. The best algorithms known today achieve only a marginally better approximation factor of $(1 - o(1))k$ [Kar09, Hal02].

On the intractability side, there have been several results. For Minimum Vertex Cover on graphs ($k = 2$), Håstad showed a $7/6 - \varepsilon$ inapproximability. This was later improved to 1.36 in the work of Dinur and Safra [DS05]. For k-uniform hypergraphs, a hardness factor of $2-\varepsilon$ was given by Goldreich [Gol01] for some constant k. Subsequently, Holmerin [Hol02b] showed an inapproximability factor of $2-\varepsilon$ for $k=4$, and $3/2-\varepsilon$ for $k=3$. For large values of k, Trevisan [Tre01] gave a $k^{1/19}$ hardness factor, which was improved to $k^{1-o(1)}$ by Holmerin [Hol02a], to $k-3-\varepsilon$ by Dinur et al. [DGK02], and to $k-1-\varepsilon$ in the work of Dinur, Guruswami, Khot, and Regev [DGKR05], which gives the best hardness factor known for all $k \geq 3$.

Obtaining tight inapproximability for Minimum Vertex Cover has remained a challenging open problem. However, assuming the Unique Games Conjecture (UGC) formulated by Khot [Kho02] removes fundamental obstructions to further progress. Indeed, Khot and Regev [KR08] showed an optimal $k-\varepsilon$ inapproximability for Minimum Vertex Cover on k-uniform hypergraphs for $k \geq 2$, assuming UGC. This result was strengthened by Austrin, Khot, and Safra [AKS11], to yield optimal inapproximability for Minimum Vertex Cover on d-regular graphs, again assuming UGC.

The Unique Games Conjecture also yields structured hardness results for Hypergraph Vertex Cover which, via reductions, imply optimal inapproximability results for some classic scheduling problems [BK09, BK10], and the Stochastic Vehicle Routing problem [GNS12]. We define these problems in the next section.

7.3 Scheduling and Stochastic Vehicle Routing

For some classic scheduling problems such as Concurrent Open Shop and the Assembly Line problem, optimal inapproximability results were known only under the UGC [BK09, KMTV11]. In this thesis, we prove these optimal inapproximability results without assuming the UGC, *i.e.*, based only on the assumption $\mathbf{P} \neq \mathbf{NP}$. We also give a super-constant inapproximability result for the Stochastic Vehicle Routing problem without assuming the UGC, where again, such a result was previously only known assuming the UGC [GNS12].

All these results are based on reductions from a new structured hardness result for Hypergraph Vertex Cover that we will describe in the next section. We now define the scheduling and stochastic optimization problems mentioned above, along with a brief description of previous works and the statements of the respective hardness results we prove.

Concurrent Open Shop

There is a set of m machines $\mathcal{M} = \{M_1, \ldots, M_m\}$, such that each machine produces a unique kind of component. We have a collection of jobs $\mathcal{J} = \{1, \ldots, n\}$, where each job $j \in \mathcal{J}$ has weight w_j, and for each $i = 1, \ldots, m$, the job j requires an operation of p_{ij} units of processing on machine M_i. For a fixed job j, all the p_{ij} units on machine M_i, for $i = 1, \ldots, m$, can be scheduled independently and in parallel. The completion time of the job i, denoted by T_i is defined to be the least time when all its operations p_{ij} are completed. The objective is to minimize $\sum_i w_i T_i$, *i.e.*, the total weighted completion time of the jobs.

This problem is used to model various applications in manufacturing and supply chain management (see [LLP05] for a survey). A 2-approximation for this problem can be obtained through various techniques [GKP07, LLP07, MQS+10]. Garg *et al.* [GKP07] showed that the problem does not have a polynomial time approximation scheme assuming $\mathbf{P} \neq \mathbf{NP}$. It was shown to be $\mathbf{NP}$-hard to approximate within $\frac{6}{5} - \varepsilon$ by Mastrolilli et al. [MQS+10], who also showed a $\frac{4}{3} - \varepsilon$ hardness factor assuming the UGC. Subsequently, Bansal and Khot [BK10] obtained a tight $2 - \varepsilon$ inapproximability, assuming UGC. The tight UGC hardness result can also be obtained using the framework of Kumar et al. [KMTV11] for hardness results for strict CSPs. We show an optimal hardness of approximation result, bypassing the UGC.

Theorem 7.1. *The Concurrent Open Shop problem is* **NP***-hard to approximate within a factor of* $2 - \varepsilon$*, for any* $\varepsilon > 0$.

Assembly Line Problem

There is a set of $m + 1$ machines $\mathcal{M} = \{M_0, M_1, \ldots, M_m\}$, and a set of jobs $\mathcal{J} = \{1, \ldots, n\}$. The machine M_0 is special, and is called the assembly machine. Each job $j \in \mathcal{J}$ requires an operation O_{ij} of size p_{ij} to be executed on machine M_i, for every $i = 1, \ldots, m$. For a fixed j, the operations $\{O_{ij}\}_{i=1}^m$ can be done in any order. For every j, after all the operations $\{O_{ij}\}_{i=1}^m$ are completed, a final operation O_{0j} of size p_{0j} is required to be performed on machine M_0. The job j is completed when O_{0j} is completed. Denote the time of completion of job j by T_j. The objective is to minimize the makespan, *i.e.*, $\max_j T_j$, or equivalently, the latest completion time of a job.

This is a classic scheduling problem [PSS+95], with a trivial 2-approximation. It is easy to see that any schedule that does not let a machine idle if an operation can be scheduled on it, gives a 2-approximation. The best approximation factor known is $2 - \frac{1}{m}$ [PSS+95], giving only a lower order improvement. It had been observed that the known hardness results for coloring k-colorable graphs implied that the problem does not have a polynomial time approximation scheme assuming $\mathbf{P} \neq \mathbf{NP}$, but these reductions give small hardness factors. Settling this problem was listed in a famous list of open problems in scheduling by Schuurman and Woeginger [SW99]. Assuming the UGC, Bansal and Khot [BK10] showed an optimal hardness of $2 - \varepsilon$. We settle this question by proving an optimal hardness of approximation result, bypassing the UGC.

Theorem 7.2. *The Assembly Line problem is* **NP***-hard to approximate within a factor of* $2 - \varepsilon$*, for any* $\varepsilon > 0$.

Two Stage Stochastic Vehicle Routing

In this problem, we are concerned with the delivery of a single good from a depot to a set of customers. We have a vehicle of capacity Q units, and a metric (V, d) over a set of customers including a depot/root $r \in V$. There is a samplable joint distribution $\mathcal{D}$ on unsplittable customer demands for the good, denoted by $\{q_v\}_{v \in V}$, and an inflation factor $\lambda \geq 1$. The solution consists of two parts:

(a) *First Stage Solution*: The first part is a fixed route τ consisting of multiple round trips – which we refer to as *r-tours* – starting and ending at the depot r. This route is computed by the algorithm without the knowledge of the exact customer demands, but can be used to satisfy as many demands as possible after they are revealed. Note that the vehicle can carry Q amount of the good at the start of each r-tour. A vertex may appear several times in the route, however since its demand is unsplittable, it cannot exceed Q and has to be satisfied all at once. The instantiated demands may not all be satisfied by the first stage solution, in which case, a more expensive second stage solution has to be computed to satisfy the remaining demands.

(b) *Second Stage Solution*: Given the instantiated demands $\overline{q}$ sampled from $\mathcal{D}$, the algorithm satisfies a subset $\overline{q}_A$ by the fixed route. The second stage solution is a recourse route $\sigma(\overline{q})$ which satisfies the rest of the demands.

The algorithm pays the cost $d(\tau)$ of the fixed route τ, plus λ times the expected cost of the recourse route $\mathbf{E}_{\overline{q} \leftarrow \mathcal{D}}[d(\sigma(\overline{q}))]$. The goal is to find a solution consisting of the fixed route and a strategy for the recourse route, in order to minimize the cost.

This problem has recently been studied by Gørtz, Nagarajan, and Saket [GNS12], who gave a poly-logarithmic approximation for it. Using Theorem 7.4 of Bansal and Khot [BK10] as a black-box, [GNS12] also gave a reduction showing an $\omega(1)$ hardness of approximation, assuming UGC. This is in contrast to constant factor approximations that have been obtained for single stage stochastic vehicle routing problems [GNR12]. For this problem too, we are able to bypass the UGC, and yet prove an $\omega(1)$ hardness result.

Theorem 7.3. *The Two Stage Stochastic Vehicle Routing problem is* **NP**-*hard to approximate within any constant factor* $C > 1$.

7.4 Structured Hardness for Vertex Cover

The optimal inapproximability of Minimum Vertex Cover on hypergraphs by Khot and Regev [KR08] was strengthened in the works of Bansal and Khot [BK09, BK10] to show that Minimum Vertex Cover is $(k-\varepsilon)$-hard to approximate even on k-uniform hypergraphs with additional structure (still assuming UGC). The UGC based results for the problems mentioned in the last section were proved via reductions from this structured hardness result [BK10, GNS12]. Our results for the above problems are based on a weaker structured hardness result for Hypergraph Vertex Cover, which however does not assume the UGC. In this section, we first discuss the results of Bansal and Khot [BK09, BK10], followed by our structured hardness for Hypergraph Vertex Cover.

Bansal and Khot [BK09] showed (assuming UGC) an optimal $2-\varepsilon$ inapproximability for Minimum Vertex Cover on graphs ($k = 2$) even when the graph $G(V, E)$ is *almost bipartite*, *i.e.*, it has two disjoint independent sets of size at least $(\frac{1}{2} - \varepsilon)|V|$ each, and thus the complement of each of the independent sets is a vertex cover of size at most $(\frac{1}{2} + \varepsilon)|V|$. More formally, they showed that for any $\varepsilon > 0$, assuming the UGC, it is **NP**-hard to decide whether a given graph is almost bipartite, or if every vertex cover is of size at least $(1 - \varepsilon)|V|$. This is a tight structured hardness result for Minimum Vertex Cover, and is obtained by the construction of a PCP with nearly optimal parameters. This PCP also yielded optimal $2-\varepsilon$ inapproximability for the classic scheduling problem of minimizing the weighted completion time subject to precedence constraints (usually called Precedence Constrained Scheduling), assuming the UGC [2].

In a subsequent work, Bansal and Khot [BK10] generalized the structured hardness result for Minimum Vertex Cover to *almost k-partite* k-uniform hypergraphs ($k \geq 2$). Formally stated, they proved the following result.

Theorem 7.4 (Bansal-Khot [BK10])**.** *Assuming the Unique Games Conjecture, for any integer $k \geq 2$, and constant $\varepsilon > 0$, the following is* **NP***-hard: Given a k-uniform hypergraph $G(V, E)$, decide between the following two cases,*

YES Case. *There is a partition of V into disjoint subsets $V_1, \ldots, V_k, X$, such that $|V_1| = |V_2| = \cdots = |V_k| \geq (\frac{1-\varepsilon}{k})|V|$, with the property that every hyperedge $e \in E$ has at most one vertex from any V_j, ($j \in [k]$). Informally, the hypergraph is almost k-partite.*

NO Case. *There is no vertex cover in G of size at most $(1 - \varepsilon)|V|$,* i.e., *every independent set in G has size at most $\varepsilon|V|$.*

Observe that in the YES case of the above theorem, for any $j \in [k]$, the set $V_j \cup X$ is a vertex cover, and thus, $V \setminus (V_j \cup X)$ is an independent set. Therefore, the graph has k vertex covers of size at most $(\frac{1}{k} + \varepsilon)|V|$ each, and k independent sets, each of size at least $(1 - \frac{1}{k} - \varepsilon)|V|$. For $k = 2$, this is same as the almost bipartite property for graphs. Combining the YES and NO cases also yields a $(k - \delta)$ UGC based inapproximability for Minimum Vertex Cover on k-uniform hypergraphs.

Via reductions from this hardness result, Bansal and Khot [BK10] proved optimal UGC based inapproximability for Concurrent Open Shop and the Assembly Line Problems. Gørtz, Nagarajan, and Saket [GNS12] also used Theorem 7.4 to rule out a constant approximation for Two Stage Stochastic Vehicle Routing, assuming the

[2] To be precise, Bansal and Khot [BK09] assume a stronger version of the conjecture that is not known to be equivalent to the UGC.

UGC. In the next section, we discuss our new structured hardness result that bypasses the UGC, and is a key intermediate step in our results for these problems.

7.4.1 A New Structured Hardness for Hypergraph Vertex Cover

The structural property of almost k-partiteness in the YES case of Theorem 7.4 is crucial for obtaining hardness results for the scheduling and stochastic optimization problems mentioned above. The question we study is whether one can instead use a weaker structural property for which an analogous inapproximability can be shown *without assuming* the UGC.

For example, let us consider the following structural property in the YES case. Say $k = qT$ for some positive integers q and T, and that there is a partition of $|V|$ into subsets $V_1, \dots, V_q, X$ such that $|V_1| = \cdots = |V_q| \geq (\frac{1-\varepsilon}{q})|V|$, and every hyperedge contains at most T vertices from V_j, for any $j \in [q]$. This is equivalent to requiring that: the qT vertices in every hyperedge e can be partitioned into T disjoint blocks of q vertices each, such that each block has at most one vertex from V_j, for any $j \in [q]$. If we call a block that has at most one vertex from any V_j as *good*, then the property requires that the qT vertices of any hyperedge can be partitioned into T good blocks of size q each. The almost k-partiteness corresponds to the case of $T = 1$ and $q = k$. For $T > 1$, this property is weaker than the YES case of Theorem 7.4. In particular, substituting it into Theorem 7.4 yields a weaker inapproximability factor for Hypergraph Vertex Cover. Nevertheless, even with $T = q$, this property would be useful for the above mentioned applications, and interesting in general.

However, it seems difficult to prove an analog of Theorem 7.4 even with the above property in the YES case. Instead, we weaken it further to require that: the qT vertices in every hyperedge e can be partitioned into T disjoint blocks of q vertices each, such that *at least* $T - 1$ of the blocks are good. In other words, the vertices of any hyperedge can be partitioned into T blocks of size q each, such that at least $T-1$ of the blocks each have at most one vertex from V_j, for any $j \in [q]$. We prove the following structural hardness for hypergraph vertex cover with the above property in the YES case.

Theorem 7.5. *For any $\varepsilon > 0$, and positive integers $q, T > 1$, the following is* **NP**-*hard: Given a qT-uniform hypergraph $G(V, E)$, decide between the following two cases,*

YES Case: *There is a partition $V_1, V_2, \ldots, V_q, X$ of V, such that:*

1. $|V_1| = |V_2| = \cdots = |V_q| \geq \left(\frac{1-\varepsilon}{q}\right)|V|$ *and,*
2. *For each hyperedge $e \in E$, the vertices in e can be partitioned into T blocks $B^e_1, \ldots, B^e_T$, each of size q, such that B^e_i contains at most one vertex from any V_j $(1 \leq j \leq q)$, for $i = 1, \ldots, T-1$.*

NO Case: *There is no vertex cover in G of size at most $(1-\varepsilon)|V|$,* i.e., *every independent set in G has size at most $\varepsilon|V|$.*

Note that in the YES case, the vertices in the block B^e_T can be arbitrary. The proof of this theorem has been presented in Chapter 8, followed by black box reductions that give optimal $(2-\varepsilon)$ inapproximability for Concurrent Open Shop and the Assembly Line problem, and an $\omega(1)$ inapproximability for Stochastic Vehicle Routing.

7.5 Vertex Cover on k-uniform k-partite Hypergraphs

Another version of hypergraph vertex cover that we study is the Minimum Vertex Cover problem on k-partite k-uniform hypergraphs, when the underlying partition is given. We denote this problem as k-HypVC-Partite. This is an interesting problem by itself, and its variants have been studied for applications related to databases such as distributed data mining [FMO+03], schema mapping discovery [GS10a], and optimization of finite automata [ISOY05]. On bipartite graphs ($k = 2$), by König's Theorem, computing the minimum vertex cover is equivalent to computing the maximum matching, which can be done efficiently. For general k, the problem was studied by Lovász who, in his doctoral thesis [Lov75], proved the following upper bound.

Theorem 7.6 (Lovász [Lov75])**.** *For every k-partite k-uniform hypergraph G: $\textsc{vc}(G)/\textsc{lp}(G) \leq k/2$, where $\textsc{vc}(G)$ denotes the size of the minimum vertex cover and $\textsc{lp}(G)$ denotes the value of the standard LP relaxation. This yields an efficient $k/2$ approximation for k-HypVC-Partite.*

Note that the standard LP relaxation does not utilize the knowledge of the k-partition and therefore, by Lovász's result, $\textsc{lp}(G)$ is a $k/2$ approximation to $\textsc{vc}(G)$ even when the k-partition is not known. The above upper bound was shown to be tight by Aharoni, Holzman, and Krivelevich [AHK96], who proved the following theorem.

Theorem 7.7 (Aharoni *et al.* [AHK96])**.** *For every $k \geq 3$, there exists a family of k-partite k-uniform hypergraphs G such that $\textsc{vc}(G)/\textsc{lp}(G) \geq k/2 - o(1)$. Thus, the integrality gap of the standard LP relaxation for k-HypVC-Partite is $k/2 - o(1)$.*

The standard LP relaxation for k-HypVC-Partite and a proof of the above theorem describing the integrality gap construction is included in Chapter 9.

On the hardness side, the problem was shown to not have a polynomial time approximation scheme assuming $\mathbf{P} \neq \mathbf{NP}$, in [ISOY05] and [GS10a] for $k = 3$, which can be extended easily to $k \geq 3$. A recent work of Guruswami and Saket [GS10b] showed the following non-trivial hardness of approximation factor for general k.

Theorem 7.8 (Guruswami and Saket [GS10b]). *For any $\epsilon > 0$ and $k \geq 5$, k-HypVC-Partite is* **NP***-hard to approximate within a factor of $\frac{k}{4} - \epsilon$. Assuming the UGC yields an optimal hardness factor of $\frac{k}{2} - \epsilon$ for $k \geq 3$* [3].

We show a nearly optimal **NP**-hardness result for approximating k-HypVC-Partite.

Theorem 7.9. *For any $\epsilon > 0$ and integer $k \geq 4$, it is* **NP***-hard to approximate k-HypVC-Partite to within a factor of $\frac{k}{2} - 1 + \frac{1}{2k} - \epsilon$.*

The **NP**-hardness factor obtained in the above theorem is a significant improvement on that obtained in [GS10b], and for any $k \geq 4$, is off by at most an additive constant of 1 from the optimal. The sub-optimality is due to fundamental limitations of currently known techniques and is analogous to that of the hardness factor for k-HypVC obtained in [DGKR05]. Chapter 9 is devoted to a proof of the above theorem.

[3]The UGC result is based on the framework of Kumar *et al.* [KMTV11]

Chapter 8

Hardness for Scheduling Problems

In this chapter, we study the inapproximability of two well-known scheduling problems: *Concurrent Open Shop* and the *Assembly Line* problem. For both these problems, Bansal and Khot [BK10] obtained tight $(2-\varepsilon)$-factor inapproximability, assuming the Unique Games Conjecture (UGC). We prove optimal $(2-\varepsilon)$-factor **NP**-hardness of approximation for both these problems without assuming UGC. We also prove a super constant **NP**-hardness factor for *Two Stage Stochastic Vehicle Routing*, for which a similar inapproximability was shown by Gørtz, Nagarajan, and Saket [GNS12] assuming the UGC, and based on the result of [BK10].

Our results follow via black-box hardness reductions from a structured hardness result for Minimum Vertex Cover on hypergraphs – a weaker analog of a similar result of Bansal and Khot [BK10] which, however, is based on UGC.

8.1 Main Results

Structured Hypergraph Vertex Cover (HypVCBlk). We rephrase our structured hardness result for Minimum Vertex Cover on hypergraphs as **NP**-hardness of a decision problem HypVCBlk[1], which we define below. For any $\varepsilon > 0$, and pos-

[1]The abbreviation HypVCBlk denotes Hypergraph Vertex Cover with Block structure.

itive integers $q, T > 1$, an instance of HypVCBlk(q, T, ε) consists of a qT-uniform hypergraph. The problem is defined as follows.

Definition 8.1. *For any $\varepsilon > 0$, and positive integers $q, T > 1$, the problem* HypVCBlk*(q, T, ε) is: Given a qT-uniform hypergraph $G(V, E)$ as an instance, decide between the following two cases,*

YES Case: *There is a partition $V_1, V_2, \ldots, V_q, X$ of V, such that:*

1. *$|V_1| = |V_2| = \cdots = |V_q| \geq \left(\frac{1-\varepsilon}{q}\right) |V|$, and,*

2. *For each hyperedge $e \in E$, the vertices in e can be partitioned into T blocks $B_1^e, \ldots, B_T^e$, each of size q, such that the block B_i^e contains at most one vertex from any V_j $(1 \leq j \leq q)$ for $i = 1, \ldots, T-1$.*

NO Case: *There is no vertex cover in G of size at most $(1 - \varepsilon)|V|$,* i.e., *every independent set in G has size at most $\varepsilon|V|$.*

This definition serves as a convenient abstraction for the hardness reduction as well as for proving hardness results for the applications. We prove the following theorem concerning the hardness of HypVCBlk, that immediately implies Theorem 7.5 from Section 7.4.1.

Theorem 8.2. *For any constant $\varepsilon > 0$, and integers $q, T > 1$,* HypVCBlk*(q, T, ε) is* **NP***-hard.*

Applications: Scheduling and Stochastic Routing. Our hardness results for Concurrent Open Shop, Assembly Line problem, and Two stage Stochastic Vehicle routing will follow via black-box reductions from the hardness for HypVCBlk proved in Theorem 8.2. These problems were defined in Section 7.3. We restate our results for these problems here for completeness.

Theorem 8.3. *The Concurrent Open Shop problem is* **NP***-hard to approximate within a factor of* $2 - \varepsilon$*, for any* $\varepsilon > 0$*.*

Theorem 8.4. *The Assembly Line problem is* **NP***-hard to approximate within a factor of* $2 - \varepsilon$*, for any* $\varepsilon > 0$*.*

Theorem 8.5. *The Two Stage Stochastic Vehicle Routing problem is* **NP***-hard to approximate within any constant factor* $C > 1$*.*

Organization. We begin with an outline of the proof and techniques used, in Section 8.2. Next, we give useful definitions and preliminary results for the analysis of Boolean functions, in Sections 8.3.1 and 8.3.2 respectively. Section 8.3.3 describes the variant of Label Cover used in the reduction and states its inapproximability. Section 8.4 contains the main reduction and its analysis, proving Theorem 8.2. Sections 8.5.1, 8.5.2 and 8.5.3 contain the respective hardness reductions and proofs of Theorems 8.3, 8.4, and 8.5.

8.2 Proof Outline and Techniques

The main idea of our hardness reduction for Theorem 8.2 is illustrated by the following *gadget*. Say we are given integers T, m, q, and a small constant $\varepsilon \in (0, 1)$. Consider the following *biased* Long Code: the set $\{*, 1, \ldots, q\}^m$, equipped with a measure μ. We call the elements of this set as colorings of $[m]$. A random coloring is sampled from μ as follows: Sample each coordinate independently, assigning it $*$ with probability ε, and j (for $j \in [q]$) with probability $\frac{1-\varepsilon}{q}$. Given a set of q colorings , we say that they 'collide' in coordinate l, if there is some $j \in [q]$ such that two or more of these colorings have j in coordinate l.

The gadget we construct is a qT-uniform hypergraph with the vertex set consisting of all colorings from T copies of the biased Long Code. The hyperedges are constructed as follows. Choose q colorings from each of the T Long Codes, and add a hyperedge consisting of these qT colorings if for every coordinate $l \in [m]$, there is at most one $i \in [T]$ such that the q colorings from the i^{th} Long Code collide in coordinate l. We outline the two properties of this gadget which are key to the reduction.

For any coordinate $l \in [m]$, we can partition the vertices (colorings) into subsets $V_1, \ldots, V_q, X$ in the following manner. A coloring is in V_j if its l^{th} coordinate is j, for all $j \in [q]$, and in X if it is $*$. Clearly, X has ε fraction of the weight, and each of the subsets V_j have the same weight. Moreover, for each hyperedge e, if we let the block B_i^e be the set of vertices (colorings) in e from the i^{th} Long Code ($1 \leq i \leq T$), we get that at least $T - 1$ blocks each contain at most one vertex from V_j, for any $j \in [q]$ Therefore, for any coordinate, we obtain the desired structural property of the hypergraph. This corresponds to the YES case of HypVCBlk.

For the NO case, suppose we are given a minimal vertex cover in the above hypergraph that has an intersection of measure at most 0.99 with each of the Long Codes. Denote its complementary maximal independent set by $\mathcal{I}$. Consider the family of colorings obtained as the intersection of $\mathcal{I}$ with the i^{th} Long Code. The maximality of $\mathcal{I}$ implies that this family is *monotone*, *i.e.*, a coloring in the family belongs to it even after changing a coordinate from $*$ to any $j \in [q]$. By perturbing ε by a small amount, and applying Russo's lemma [Rus82], we deduce that the family has bounded *average sensitivity*. Applying Friedgut's theorem [Fri98] yields a small set of coordinates called a *core*, such that the membership of a coloring in the family is essentially determined by the coordinates in the core. This allows us to find q colorings in the family that do not collide in coordinates outside the core. Therefore, we obtain, for each of the T families, q colorings along with a core for that family. It easily follows that the cores corresponding to at least two of the families must intersect, otherwise there is a hyperedge in $\mathcal{I}$, which is a contradiction. Thus, we have *decoded* an independent set of significant weight into a small set of coordinates that are *consistent*.

Variants of the above gadget over the boolean domain were used by Dinur *et al.* [DGKR05] in their hardness for Minimum Vertex Cover on k-uniform hypergraphs, and later by Sachdeva and Saket [SS11] for proving hardness for Minimum Vertex Cover on k-uniform k-partite hypergraphs. For analyzing their constructions over the boolean domain, [DGKR05] and [SS11] used techniques from extremal combinatorics which, however, are considerably less amenable for the larger, biased domains used in our reduction. Instead, we are able to adapt and apply techniques from Fourier Analysis which were introduced in the seminal work of Dinur and Safra [DS05] and used by Dinur *et al.* [DGK02] on boolean domains. These techniques were later extended to larger domains in more recent works [DKPS10, KS12].

We note that Bansal and Khot [BK10] in their proof of Theorem 7.4 used powerful tools based on the Invariance Principle [Mos08]. However, their use of these tools requires a *bijective* constraint satisfaction problem, which is currently known to be **NP**-hard only assuming the Unique Games Conjecture [Kho02]. The techniques in our proof – though weaker in terms of the lower bounds yielded – can nevertheless be used, as outlined below, with a constraint satisfaction problem unconditionally known to be **NP**-hard.

We combine our gadget with a variant of Label Cover – used in the work of Gopalan, Khot, and Saket [GKS10] – that is particularly suited for our techniques. It consists of T-variable constraints which are obtained by aggregating subsets of T constraints incident on a variable on the smaller side of the standard Label Cover. We replace each variable of this variant of Label Cover by a Long Code. Based on the structure of each T-variable constraint, we add hyperedges between the Long Codes corresponding to its variables in a manner analogous to the hyperedges of the above gadget. The YES case follows easily using the analysis sketched above and a satisfying assignment to the Label Cover instance. For the NO case, we use the decoding procedure outlined above to find a good assignment to the variables. This gives the desired structured hardness (Theorem 8.2) for Minimum Vertex Cover on hypergraphs. Using this, the hardness results for the scheduling and stochastic vehicle routing problems (Theorems 8.3, 8.4 and 8.5) follow via black-box reductions. These reductions are analogous to the ones used for these problems in earlier work [MQS$^+$10, BK10, GNS12], suitably modified to exploit the structure of the Minimum Vertex Cover instances we construct.

8.3 Preliminaries

Before we describe the reduction, we need a few definitions and preliminary lemmas. We follow notation similar to previous works [DKPS10, KS12].

8.3.1 Definitions

An element of $\{*, 1, \ldots, q\}^n$ is referred to as a *coloring* of $[n]$. We shall refer to subsets of $\{*, 1, \ldots, q\}^n$ as *families*. The measure μ_p on the set $\{*, 1, \ldots, q\}^n$ is a product measure assigning, in each coordinate, probability mass $1 - p$ to $*$ and p/q to each of the remaining q elements. The dimension of the space will often not be stated explicitly, and will be clear from the context. A family $\mathcal{F} \subseteq \{*, 1, \ldots, q\}^n$ is *monotone* if $F \in \mathcal{F}$ implies $F' \in \mathcal{F}$ where F' is any coloring obtained by changing a $*$ in any coordinate of F to some element in $[q]$.

For any coloring $F \in \{*, 1, \ldots, q\}^n$, and any set $C \subseteq [n]$, we let $F|_C$ denote the coloring F restricted to C. For any coloring F, we call F an *extension* of the restricted coloring $F|_C$. A set $C \subseteq [n]$ is a (δ, p)-*core* for a family $\mathcal{F}$, if there exists a family $\mathcal{F}'$ such that $\mu_p(\mathcal{F} \triangle \mathcal{F}') \leq \delta$ and $\mathcal{F}'$ depends only on the coordinates in C, *i.e.*, for any $F \in \{*, 1, \ldots, q\}^n$, changing the value of the coordinates in $[n] \setminus C$ does not affect whether F is in $\mathcal{F}'$. For $t \in (0,1)$, and a subset $C \subseteq [n]$, let a *core-family* $[\mathcal{F}]_C^t$ be defined as,

$$[\mathcal{F}]_C^t \overset{\text{def}}{=} \left\{ F \in \{*, 1, \ldots, q\}^C \,\middle|\, \Pr_{F' \in \mu_p^{[n] \setminus C}} [(F, F') \in \mathcal{F}] > t \right\},$$

where (F, F') is a combined coloring obtained by choosing the coloring assigned by F on coordinates in C, and by F' on $[n] \setminus C$. The *influence* of a coordinate $i \in [n]$ for a family $\mathcal{F}$ is defined as follows:

$$\mathsf{Inf}_i^p(\mathcal{F}) \overset{\text{def}}{=} \mu_p(\{F : F|_{i \leftarrow *} \notin \mathcal{F} \text{ and } \exists j \in [q] \text{ s.t. } F|_{i \leftarrow j} \in \mathcal{F}\}),$$

where $F|_{i \leftarrow *}$ is a coloring identical to F except on the i^{th} coordinate where it is $*$, and similarly for $F|_{i \leftarrow r}$, for any $r \in [q]$. The *average sensitivity* of $\mathcal{F}$ at p is defined as $\mathsf{as}_p(\mathcal{F}) \overset{\text{def}}{=} \sum_{i=1}^n \mathsf{Inf}_i^p(\mathcal{F})$.

Let P_p^q be a distribution on $\{*, 1, \ldots, q\}^q$ defined by the following randomized sampling:

1. Sample a uniformly random permutation $\tau : [q] \mapsto [q]$. Set $(y_1, \ldots, y_q) \overset{\text{def}}{=} (\tau(1), \ldots, \tau(q))$.

2. Independently for each $j \in [q]$, set $x_j = y_j$ with probability p and with probability $1 - p$ set $x_j = *$. Output $(x_1, \ldots, x_q)$ as a uniform sample from P_p^q.

Clearly, P_p^q is supported on tuples with at most one coordinate valued j, for any given $j \in [q]$. It is also easy to see that if we sample $(x_1, \ldots, x_q)$ from P_p^q, each x_j is marginally distributed according to μ_p.

8.3.2 Results from Analysis of Boolean Functions

We begin by stating the following variant of Russo's Lemma proved in [DKPS10] (as Lemma 1).

Lemma 8.6 (Russo's Lemma [Rus82])**.** *Let $\mathcal{F} \subseteq \{*, 1, \ldots, q\}^n$ be monotone, then $\mu_p(\mathcal{F})$ is increasing with p. In fact,*

$$\frac{1}{q} \cdot \mathsf{as}_p(\mathcal{F}) \leq \frac{d\mu_p(\mathcal{F})}{dp} \leq \mathsf{as}_p(\mathcal{F}).$$

The following corollary follows immediately.

Corollary 8.7. *For a monotone family $\mathcal{F} \subseteq \{*, 1, \ldots, q\}^n$, we have,*

1. *For any $p' \geq p$, $\mu_{p'}(\mathcal{F}) \geq \mu_p(\mathcal{F})$.*

2. *For any $\varepsilon > 0$, there is a $p' \in [1 - \varepsilon, 1 - \varepsilon/2]$ such that $\mathsf{as}_{p'}(\mathcal{F}) \leq \frac{2q}{\varepsilon}$.*

Proof. The first property is immediate since Lemma 8.6 implies that $\frac{d\mu_p(\mathcal{F})}{dp} \geq 0$. For the second property, assume to the contrary that for all $p \in [1-\varepsilon, 1-\varepsilon/2]$, $\mathsf{as}_p(\mathcal{F}) > \frac{2q}{\varepsilon}$. By Lemma 8.6, this implies that for all $p \in [1 - \varepsilon, 1 - \varepsilon/2] : \frac{d\mu_p(\mathcal{F})}{dp} > \frac{2}{\varepsilon}$. Integrating over p in the range $[1 - \varepsilon, 1 - \varepsilon/2]$, we obtain that $\mu_{p'}(\mathcal{F}) > 1$ for $p' = 1 - \varepsilon/2$, which is a contradiction. $\square$

For our analysis, we require the following generalization of Friedgut's Theorem [Fri98], the proof of which follows from the results of Sachdeva and Tulsiani [ST11].

Theorem 8.8 (Friedgut's Theorem [Fri98, ST11])**.** *Fix $\delta > 0$. Let $\mathcal{F} \subseteq \{*, 1, \ldots, q\}^n$ be monotone with $a = \mathsf{as}_p(\mathcal{F})$. There exists a function $C_{Friedgut}(p, \delta, a) \leq c_{p,q}^{a/\delta}$, for a constant $c_{p,q}$ depending only on p and q, so that $\mathcal{F}$ has a (δ, p)-core C of size $|C| \leq C_{Friedgut}(p, \delta, a)$.*

We use the above stated Friedgut's theorem in conjunction with the following generalization of Proposition 3 in [DKPS10] and Lemma 3.1 in [DS05].

Proposition 8.9. *Let $t \in (1/2, 1)$ be a constant. If C is a (δ, p)-core of $\mathcal{F}$, then $\mu_p^C\left([\mathcal{F}]_C^t\right) \geq \mu_p(\mathcal{F}) - \frac{t}{1-t}\delta$.*

Proof. Define the families

$$\mathcal{F}_{1/2} \overset{\text{def}}{=} \left\{F \;\middle|\; F|_C \in [\mathcal{F}]_C^{1/2}\right\}, \qquad \text{and} \qquad \mathcal{F}_t \overset{\text{def}}{=} \left\{F \mid F|_C \in [\mathcal{F}]_C^{t}\right\},$$

where $F|_C$ denotes the coloring F restricted to C. For any coloring F, we call F an *extension* of the restricted coloring $F|_C$. Thus, $\mathcal{F}_{1/2}$ and $\mathcal{F}_t$ are the families of all extensions of the core-families $[\mathcal{F}]_C^{1/2}$ and $[\mathcal{F}]_C^t$ respectively. Thus, since μ_p is a product measure, $\mu_p(\mathcal{F}_{1/2}) = \mu_p^C([\mathcal{F}]_C^{1/2})$, and $\mu_p(\mathcal{F}_t) = \mu_p^C([\mathcal{F}]_C^t)$.

It is easy to see that amongst all families $\mathcal{F}'$ that depend only on the coordinates in C, the family $[\mathcal{F}]_{1/2}$ minimizes $\mu(\mathcal{F}\Delta\mathcal{F}')$ (though it may not be the unique minimizing family). In other words, the core-family on the core C, whose extension *best approximates* $\mathcal{F}$, is $[\mathcal{F}]_C^{1/2}$. Since C is a (δ, p)-core, we get, $\mu_p(\mathcal{F}\Delta\mathcal{F}_{1/2}) \leq \delta$.

Moreover, since $t > 1/2$, $[\mathcal{F}]_C^t \subseteq [\mathcal{F}]_C^{1/2}$. Consider any coloring $F \in [\mathcal{F}]_C^{1/2} \setminus [\mathcal{F}]_C^t$. Thus, the fraction of extensions of F that are in $\mathcal{F}$ is at least $\frac{1}{2}$ and at most t, *i.e.*,

$$\frac{1}{2} \cdot \mu_p^C(F) < \mu_p\left(\{F' |\; F' \in \mathcal{F} \text{ and } F'|_C = F\}\right) \leq t \cdot \mu_p^C(F).$$

Thus, the extensions of F that are contained in $\mathcal{F}\Delta\mathcal{F}_{1/2}$ have measure

$$\begin{aligned} &\mu_p\left(\left\{F' \middle|\; F' \in \mathcal{F}\Delta\mathcal{F}_{1/2} \text{ and } F'|_C = F\;\right\}\right) \\ &= \mu_p\left(\{F' |\; F' \notin \mathcal{F} \text{ and } F'|_C = F\;\}\right) \geq (1-t) \cdot \mu_p^C(F). \end{aligned}$$

Thus, the contribution by extensions of F to $\mu_p(\mathcal{F}\Delta\mathcal{F}_{1/2})$ is at least $(1-t)\mu_p^C(F)$. Similarly, the contribution of extensions of F to to $\mu_p(\mathcal{F}\Delta\mathcal{F}_t)$ is at most $t\mu_p^C(F)$.

For any coloring $F' \notin [\mathcal{F}]_C^{1/2} \setminus [\mathcal{F}]_C^t$, its extensions contribute equally to $\mu_p(\mathcal{F}\Delta\mathcal{F}_{1/2})$ and $\mu_p(\mathcal{F}\Delta\mathcal{F}_t)$. Thus, using $\frac{t}{t-1} \geq 1$ for $t \in (1/2, 1)$, and summing over all F, we get $\mu_p(\mathcal{F}\Delta\mathcal{F}_t) \leq \frac{t}{1-t}\mu_p(\mathcal{F}\Delta\mathcal{F}_{1/2}) \leq \frac{t}{1-t}\delta$. Hence, $\mu_p^C\left([\mathcal{F}]_C^t\right) = \mu_p\left(\mathcal{F}_t\right) \geq \mu_p\left(\mathcal{F}\right) - \frac{t}{1-t}\delta$. $\square$

8.3.3 Label Cover

From the work of Gopalan, Khot, and Saket [GKS10], we borrow the definition of the version of Label Cover we will use for our reduction. For positive integers $T, k, m > 1$, an instance of LabelCover(T, k, m) consists of a T-uniform hypergraph $G(V, E)$ with vertex set V and a hyperedge set E, where $\forall e \in E,\ |e| = T$. The hypergraph G is connected, and any subset $S \subset V$ of size $\delta|V|$, for a positive constant δ, induces a constant fraction $\gamma(T, \delta) > 0$ of the hyperedges in E, where the function γ depends only on T and δ. Every hyperedge $e = (v_1^e, \ldots, v_T^e)$ is associated with a T-tuple of projection functions $\{\pi_i^e\}_{i=1}^T$ where $\pi_i^e : [m] \to [k]$ and $k < m$. A vertex labeling $\sigma : V \mapsto [m]$ strongly satisfies hyperedge e if $\pi_i^e(\sigma(v_i^e)) = \pi_j^e(\sigma(v_j^e))$ for every $v_i^e, v_j^e \in e$. It is said to weakly satisfy hyperedge e if $\pi_i^e(\sigma(v_i^e)) = \pi_j^e(\sigma(v_j^e))$ for some pair $v_i^e, v_j^e \in e$.

The following theorem is proved in [GKS10] by a simple reduction from the standard bipartite version of LabelCover.

Theorem 8.10. *For any soundness parameter $\alpha > 0$, and any positive integer $T > 1$, there are positive integers k, m, such that the following is* **NP***-hard: Given an instance of* LabelCover*(T, k, m), decide between the following cases,*

> **YES Case:** *There is some vertex labeling that strongly satisfies every hyperedge.*
>
> **NO Case:** *No vertex labeling weakly satisfies an α fraction of the hyperedges.*

8.4 Hardness Reduction for HypVCBlk

We will show that for any positive constant $\varepsilon > 0$, and positive integers $q, T > 1$, HypVCBlk(q, T, ε) is **NP**-hard, thus proving Theorem 8.2. The reduction we give in this section proves a vertex weighted version of Theorem 8.2, which can easily be converted to an unweighted version by suitable replication of vertices. Our reduction begins with an instance $\mathcal{L}$ of LabelCover(T, k, m) given by Theorem 8.10, where we will specify the soundness parameter α later. The instance $\mathcal{L}$ includes a T-uniform hypergraph $G_{\mathcal{L}}$ with vertex set $V_{\mathcal{L}}$ and hyperedge set $E_{\mathcal{L}}$.

8.4.1 Construction

We construct the instance $G(V, E)$ of HypVCBlk(q, T, ε) as follows: For each vertex $v \in V_{\mathcal{L}}$, there is a copy $\mathcal{F}_v$ of the Long Code, *i.e.*, the set $\{*, 1, \ldots, q\}^m$ equipped with the measure μ_p for $p \overset{\text{def}}{=} 1 - \varepsilon$, as defined in Section 8.3.1.

Vertices: The vertex set V is the union of the Long Codes $\mathcal{F}_v$ for each $v \in V_{\mathcal{L}}$. The weight of $F \in \mathcal{F}_v$ is $\mathrm{wt}(F) \overset{\text{def}}{=} \mu_p(F)/|V_{\mathcal{L}}|$. Thus the total weight of all vertices in V is 1.

Hyperedges: Let $e \in E_{\mathcal{L}}$ be a hyperedge in the graph $G_{\mathcal{L}}$ such that $e = (v_1, \ldots, v_T)$ with the associated tuple of projections $(\pi_1, \ldots, \pi_T)$. Let $\mathcal{F}_i \overset{\text{def}}{=} \mathcal{F}_{v_i}$ for $i = 1, \ldots, T$. Suppose there are colorings $F_i^j \in \mathcal{F}_i$, $j = 1, \ldots, q$, and $i = 1, \ldots, T$, satisfying the following property:

> (*Almost-uniqueness*) For every $l_1, \ldots, l_T \in [m]$ such that $\pi_1(l_1) = \pi_2(l_2) = \cdots = \pi_T(l_T)$, there is a subset $I \subseteq [T]$, $|I| \geq T - 1$ such that for all $i \in I$, for any $j \in [q]$, the tuple $(F_i^1(l_i), F_i^2(l_i), \ldots, F_i^q(l_i))$ has at most one coordinate with value j.

For each such subset of qT colorings $F_i^j \in \mathcal{F}_i$, $j = 1, \ldots, q$, and $i = 1, \ldots, T$, which satisfy the above property, add a hyperedge $\{F_1^1, \ldots, F_1^T, F_2^1, \ldots, F_q^T\}$ to E.

In the rest of the section, we analyze the YES and NO cases separately.

8.4.2 YES Case

In the YES case of Theorem 8.10, there is a labeling $\sigma : V_{\mathcal{L}} \mapsto [m]$ that strongly satisfies every hyperedge in $E_{\mathcal{L}}$. For $j \in [q]$, we define

$$V_j \overset{\text{def}}{=} \{F \in \mathcal{F}_v \mid v \in V_{\mathcal{L}} \text{ and } F(\sigma(v)) = j\},$$

and

$$X \overset{\text{def}}{=} \{F \in \mathcal{F}_v \mid v \in V_{\mathcal{L}} \text{ and } F(\sigma(v)) = *\}.$$

Clearly, $V_1, \ldots, V_q, X$ is partition of V. By the definition of the probability measure μ_p, we have that $\text{wt}(V_1) = \cdots = \text{wt}(V_q) = \frac{1-\varepsilon}{q}$, and $\text{wt}(X) = \varepsilon$.

We need to check the structural property of the YES case of Definition 8.1. Consider a hyperedge $e \in E$. This edge corresponds to a hyperedge $e' = (v_1, \ldots, v_T) \in E_{\mathcal{L}}$ with the associated tuple of projections $(\pi_1, \ldots, \pi_T)$, so that e is incident on qT colorings $F_i^j \in \mathcal{F}_{v_i}$, $j = 1, \ldots, q$, and $i = 1, \ldots, T$, which satisfy the *Almost-uniqueness* property as stated in the construction. Let this set of qT colorings be partitioned into T blocks $B_i = \{F_i^j\}_{j=1}^q$, for $i \in [T]$. Since $\pi_1(\sigma(v_1)) = \pi_2(\sigma(v_2)) = \cdots = \pi_T(\sigma(v_T))$, the *Almost-uniqueness* property implies that there is a subset $I \subseteq [T]$, $|I| \geq T - 1$ such that for all $i \in I$, for all $j \in [q]$, the tuple $(F_i^1(\sigma(v_i)), \ldots, F_i^q(\sigma(v_i)))$ has at most one coordinate with value j. Hence B_i $(i \in I)$ has at most one coloring from any V_j, for any $j \in [q]$. Since this holds for every hyperedge in E, this completes the proof of the YES case of Theorem 8.2.

8.4.3 NO Case

In the NO case, assume that there is a maximal independent set $\mathcal{I} \subseteq V$ of weight $\text{wt}(\mathcal{I}) \geq \varepsilon$. For any vertex v, let the family $\mathcal{I}[v] \overset{\text{def}}{=} \mathcal{F}_v \cap \mathcal{I}$. Observe that the maximality of $\mathcal{I}$ implies that $\mathcal{I}[v]$ is monotone for all $v \in V$. This is because if there is a coloring $F_v \in \mathcal{I}[v]$ such that $F_v(l) = *$ for some $l \in [m]$, then adding $F_v|_{l \leftarrow j}$ for any $j \in [q]$ to $\mathcal{I}$ does not add any new hyperedges.

Good Vertices: Call a vertex $v \in V_{\mathcal{L}}$ 'good' if $\mu_p(\mathcal{I}[v]) \geq \varepsilon/2$.

Thus, the families $\mathcal{I}[v]$ of good vertices have a significant measure. It is easy to see by averaging that at least $\varepsilon/2$ fraction of the vertices in $V_{\mathcal{L}}$ are good. We show the existence of q special colorings in the families $\mathcal{I}[v]$ of good vertices v. The following is the main lemma of the analysis of the NO case.

Lemma 8.11. *For every good vertex $v \in V_{\mathcal{L}}$, there is a subset of coordinates $C_v \subseteq [m]$, and q colorings $F_v^j \in \mathcal{I}[v]$, for $j = 1, \ldots, q$, such that,*

1. *$|C_v| \leq c(q, \varepsilon)$, where $c(q, \varepsilon)$ is a constant depending only on q and ε.*

2. *For all $l \in [m] \setminus C_v$, and $j \in [q]$, the q-tuple $(F_v^1(l), \ldots, F_v^q(l))$ has at most one coordinate set to j.*

Proof. We fix a good vertex v for the proof of this lemma. By the definition of a good vertex, we have that $\mu_p(\mathcal{I}[v]) \geq \varepsilon/2$. By Corollary 8.7, there exists a $p' \in [1-\varepsilon, 1-\varepsilon/2]$ such that $\mathsf{as}_{p'}(\mathcal{I}[v]) \leq \frac{2q}{\varepsilon}$. Moreover, $\mu_{p'}(\mathcal{I}[v]) \geq \mu_p(\mathcal{I}[v]) \geq \varepsilon/2$.

We first show that for such a p', the family $\mathcal{I}[v]$ has a suitably small core. Let $\delta \stackrel{\text{def}}{=} \varepsilon/2q$. Define the constant $c'(q, \varepsilon)$ as,

$$c(q, \varepsilon) \stackrel{\text{def}}{=} \max_{\substack{p' \in [1-\varepsilon, 1-\varepsilon/2] \\ a \in [0, 2q/\varepsilon]}} C_{Friedgut}(p', \delta, a).$$

Applying Theorem 8.8, we obtain that the monotone family $\mathcal{I}[v]$ has a (δ, p')-core $C_v \subseteq [m]$ of size $|C_v| \leq c(q, \varepsilon)$. Further, setting $s \stackrel{\text{def}}{=} 1 - 1/q$, and applying Proposition 8.9, we obtain that,

$$\mu_{p'}([\mathcal{I}[v]]^s_{C_v}) \geq \mu_{p'}(\mathcal{I}[v]) - (q-1)\delta \geq \varepsilon/2 - (q-1)(\varepsilon/2q) = \varepsilon/2q,$$

by our choice of s and δ. Thus $[\mathcal{I}[v]]^s_{C_v}$ is non-empty. Fix some coloring $F^{C_v} \in [\mathcal{I}[v]]^s_{C_v}$. Consider the joint distribution on q extensions $F^1, \ldots, F^q$ of F^{C_v} defined by the following random procedure:

1. Let $F^j|_{C_v} = F^{C_v}$ for all $j \in [q]$. (This must be true for F^j to be an extension of F^{C_v}.)

2. Independently for every $i \in [m] \setminus C_v$, the tuple $(F^1(i), \ldots, F^q(i))$ is sampled uniformly from $P^q_{p'}$, as defined in Section 8.3.1.

The above, along with the definition of $P^q_{p'}$, implies that for a fixed $j \in [q]$, $F^j(i)$ is distributed i.i.d according to $\mu_{p'}$, for $i \in [m] \setminus C_v$. Since $F^{C_v} \in [\mathcal{I}[v]]^s_{C_v}$, we have that,

$$\Pr\left[F^j \in \mathcal{I}[v]\right] > s = 1 - 1/q, \tag{8.1}$$

for each $j \in [q]$. Taking a union bound over all j, we obtain,

$$\Pr\left[F^1, \ldots, F^q \in \mathcal{I}[v]\right] > 0. \tag{8.2}$$

By the definition of $P^q_{p'}$, we also have that for any $i \in [m] \setminus C_v$, $(F^1(i), \ldots, F^q(i))$ has at most one entry set to j, for any $j \in [q]$. Thus, from Equation (8.2), there exists a set of colorings $F^1, \ldots, F^j \in \mathcal{I}[v]$ with the required properties. This completes the proof of the lemma. □

The next step is to find a significant fraction of the hyperedges in $E_{\mathcal{L}}$ which can be weakly satisfied by a suitable labeling which we will construct using the subsets C_v. For this we define good hyperedges.

Good Hyperedges: Call a hyperedge $e = (v_1, \ldots, v_T) \in E_{\mathcal{L}}$ 'good', if all of its vertices $v_1, \ldots, v_T$ are good vertices.

Since at least $\varepsilon/2$ fraction of vertices are good, by the definition of LabelCover(T, k, m), at least $\gamma \overset{\text{def}}{=} \gamma(\varepsilon/2, T) > 0$ fraction of hyperedges in $E_{\mathcal{L}}$ are good, where $\gamma(\varepsilon/2, T)$ depends only on ε and T. The following is a key lemma to complete the analysis.

Lemma 8.12. *Let $e = (v_1, \ldots, v_T) \in E_{\mathcal{L}}$ be a good hyperedge with the associated projections $(\pi_1, \ldots, \pi_T)$. For $i \in [T]$, let C_{v_i} be as obtained by applying Lemma 8.11 on the (good) vertex v_i. Then there is a pair $a, b \in [T]$ with $a \neq b$, such that $\pi_a(C_{v_a}) \cap \pi_b(C_{v_b}) \neq \emptyset$.*

Proof. Assume to the contrary that the projections of the cores $\{\pi_i(C_{v_i})\}_{i=1}^T$ are all pairwise disjoint. Since e is a good hyperedge, the vertices $v_1, \ldots, v_T$ are all good. From the second part of Lemma 8.11, we also obtain the q colorings $\{F_{v_i}^j\}_{j=1}^q \subseteq \mathcal{I}[v_i]$ for each v_i, $i \in [T]$.

Consider any $l_1, \ldots, l_T \in [m]$ such that $\pi_1(l_1) = \cdots = \pi_T(l_T)$. From our assumption of pairwise disjointedness of the projections of the cores, there is at most one $i \in [T]$ such that $\pi_i(l_i) \in \pi_i(C_{v_i})$. Thus there is a subset $I \subseteq [T]$, $|I| \geq T - 1$, such that for all $i \in I$, $l_i \notin C_{v_i}$. Therefore, for each $i \in I$, $(F_{v_i}^1(l_i), \ldots, F_{v_i}^q(l_i))$ has at most coordinate set to j, for any $j \in [q]$. This implies that there is a hyperedge in E over the qT colorings $F_{v_i}^j \in \mathcal{I}[v_i]$, $j = 1, \ldots, q$, and $i = 1, \ldots, T$. This contradicts the fact that $\mathcal{I}$ is an independent set in $G(V, E)$. Thus the lemma is proved. □

We are now ready to define a (randomized) labeling of the good vertices in $V_{\mathcal{L}}$.

> *Labeling of good vertices:* For every good vertex v, define a random labeling $\sigma(v)$ which selects a label uniformly at random from the small set C_v given by Lemma 8.11.

Let $e = (v_1, \ldots, v_T) \in E_{\mathcal{L}}$ be a good hyperedge, and let a, b be as given by Lemma 8.12. The labeling σ weakly satisfies e with probability at least $1/|C_{v_a}||C_{v_b}| \geq 1/c(q, \varepsilon)^2$. Also, since at least $\gamma = \gamma(\varepsilon/2, T)$ fraction of the hyperedges are good, the labeling σ weakly satisfies at least $\gamma/c(q, \varepsilon)^2$ fraction of hyperedges in expectation. Choosing the soundness parameter α in Theorem 8.10 to be small enough, we obtain a contradiction. Thus, in the NO case, there is no independent set in $G(V, E)$ containing at least an ε fraction of the vertices.

8.5 Hardness Reductions for Applications

8.5.1 Concurrent Open Shop

We give a reduction from an instance $G(V, E)$ of $\textsc{HypVCBlk}(q, T, \varepsilon)$, where we shall specify the parameters later. We construct an instance of Concurrent Open Shop as follows. There is a machine M_e for every hyperedge $e \in E$. For every vertex $v \in V$, there is a job j_v which has unit size operation on every machine M_e such that e is incident on v. All other operations are of size 0. Each job has weight 1.

In the YES case, there is a partition $V_1, \ldots, V_q, X$ of V satisfying the property given in Definition 8.1. In this case, consider the following schedule: every machine M_e corresponding to hyperedge e schedules the unit size operations corresponding to jobs j_v where $v \in V_1$ in the first batch, those for vertices in V_2 in the second batch, and so on until those of vertices in V_q in the q^{th} batch, and lastly those of vertices in X. The order within each batch can be arbitrary. This gives a valid schedule for all machines.

We know in the YES case that the vertices of each hyperedge e can be partitioned into T blocks of size q each, such that at least $T - 1$ of the blocks are *good*, *i.e.*, any of the good blocks contains at most one vertex from V_j for any $j \in [q]$. Thus, for any $j \in [q]$ the number of vertices in any hyperedge e from the set $\cup_{i=1}^{j} V_j$ is at most $q + jT$. Thus every job corresponding to vertices in V_j is completed by time $q + jT$. Note that jobs corresponding to vertices in X are completed by time qT. Since each

V_j ($j \in [q]$) is of the same size, the average completion time is at most:

$$\begin{aligned}\frac{1}{q}\left(\sum_{j=1}^{q}(q+jT)\right)+\varepsilon qT &= \frac{1}{q}\left(q^2+T\cdot\frac{q(q+1)}{2}\right)+\varepsilon qT\\ &= qT\left(\frac{1}{T}+\frac{q+1}{2q}+\varepsilon\right).\end{aligned}$$

In the NO case, consider the set of jobs that are completed by time $qT-1$. This set must correspond to an independent set in G, since no hyperedge can complete all its operations in time qT. Thus, except for ε fraction, the rest of the jobs take at least qT time to complete. Thus the average completion time is at least $(1-\varepsilon)qT$.

Combining the above analysis with Theorem 8.2, we deduce that Concurrent Open Shop is **NP**-hard to approximate within a factor of

$$\frac{1-\varepsilon}{(1/T+(q+1)/2q+\varepsilon)}, \tag{8.3}$$

which can be made arbitrarily close to 2 by choosing T, q large enough and $\varepsilon = 1/(qT)^2$.

8.5.2 Assembly Line Problem

As above, we give a reduction from an instance $G(V,E)$ of HypVCBlk(q,T,ε), where the exact parameters will be specified later. For every hyperedge $e \in E$, there is a machine M_e. There is also an additional special machine M_0. For every vertex $v \in V$, there is a job j_v with a unit sized operation on each machine M_e such that e is incident on v. In addition, j_v has a $qT/|V|$ sized operation on M_0.

In the YES case, there is a partition $V_1,\ldots,V_q,X$ of V satisfying the property given in Definition 8.1. As proved in the previous subsection, for any $j \in [q]$, for any hyperedge e, the number of vertices incident on e from the subset $\cup_{k=1}^{j} V_k$ is at most $q+jT$. Also, since $|V_j| \le |V|/q$, the total time required by jobs corresponding to V_j

on M_0 is at most $\frac{qT}{|V|} \cdot \frac{|V|}{q} = T$. Similarly, as $|X| \leq \varepsilon|V|$, the total time required by jobs corresponding to X on M_0 is at most $\frac{qT}{|V|} \cdot \varepsilon|V| = \varepsilon qT$.

Consider the following schedule for any machine M_e : it schedules the unit sized operations corresponding to jobs of vertices in V_1 in the first batch, those of vertices in V_2 in the second batch, and so on till those of vertices in V_q in the q^{th} batch, and lastly those of vertices in X. The order within each batch can be arbitrary. The machine M_0 processes all the jobs corresponding to vertices in V_j in the time interval $(q+Tj, q+T(j+1)]$. This is a valid schedule for each machine M_e. Moreover, under this schedule, for a job corresponding to $v \in V_j$, all its unit size operations on machines $\{M_e\}$, where e is incident on v, are completed by time $q+Tj$. Since the time required by jobs from V_j on M_0 is at most T, it is feasible to schedule them in the time interval $(q+Tj, q+T(j+1)]$. Thus, this is a valid schedule for jobs corresponding to $V \setminus X$. Since the hyperedges are qT-uniform, the jobs corresponding to X complete on the machines $\{M_e\}_e$ by time qT, and are processed on M_0 in the time interval $(q+(q+1)T, q+(q+1)T+\varepsilon qT]$. Thus, the makespan is $qT(1+\varepsilon)+q+T$.

In the NO case, consider the set of vertices whose corresponding jobs complete on all the machines $\{M_e\}_e$ within time $qT-1$. This is an independent set, since no hyperedge can complete all its operations by time by $qT-1$. Thus, at most ε fraction of vertices can start processing on M_0 before time $qT-1$. Therefore, at time $qT-1$, at least $(1-\varepsilon)|V| \cdot (qT/|V|) = (1-\varepsilon)qT$ amount of processing remains on machine M_0. Thus, the makespan is at least $2qT - \varepsilon qT - 1$.

Hence, the hardness of approximation factor obtained is,

$$\frac{2qT - \varepsilon qT - 1}{qT(1+\varepsilon)+q+T}, \tag{8.4}$$

which can be made arbitrarily close to 2 by choosing T, q large enough, and $\varepsilon \leq 1/(qT)^2$ small enough.

8.5.3 Two Stage Stochastic Vehicle Routing

We give a reduction from an instance $G(V, E)$ of $\textsc{HypVCBlk}(q, T, \varepsilon)$ where the parameters will be specified later. We construct the following instance of the Two Stage Stochastic Vehicle Routing problem:

Metric: Define the set of customers to be $U \overset{\text{def}}{=} \{r, x\} \cup V$, where r is the depot/root, and x is a dummy vertex. Define the metric d on U to be the shortest-path metric on a star with x as center, and all other vertices as the leaves. The distance between x and r is $|V|/2q$ and the distance between x and any $v \in V$ is $1/2$.

Capacity: The capacity Q of the vehicle is set to 1.

Inflation factor: The inflation factor λ is set to ∞, so that there is no recourse route and all demands have to be satisfied by the fixed route.

Demand Distribution: For each hyperedge $e = (v_1, \ldots, v_{qT}) \in E$, there is a scenario s_e of demands, where we have a unit demand on each vertex v that is incident on e and zero demand on all other vertices. Let the distribution be uniformly supported on all scenarios s_e, corresponding to hyperedges $e \in E$.

Before proceeding, we note that an r-tour τ that visits the set of vertices $V_\tau \subseteq V$ has cost $d(\tau) = \frac{|V|}{q} + |V_\tau|$. We now analyze the YES and NO cases separately.

In the YES case, there is a partition $V_1, \ldots, V_q, X$ of V as specified in the definition of $\textsc{HypVCBlk}(q, T, \varepsilon)$ (Definition 8.1). Construct a fixed route τ^* as follows: For each $j \in [q]$, τ^* consists of $T+q$ copies of the r-tour that visits the subset $V_j \cup X$. Note that each of the $q(T+q)$ r-tours visits X. We need to show that τ^* can always satisfy all the demands. By the definition of the demand distribution, it suffices to show that for any hyperedge e in E, τ^* can satisfy a unit demand on each of the vertices that e is incident on. The second property specified in the definition of $\textsc{HypVCBlk}(q, T, \varepsilon)$ (Definition 8.1) immediately implies that $e \in E$ is incident on at most $T + q$ vertices from V_j, for any $j \in [q]$. Since for every $j \in [q]$, we have $T + q$ r-tours visiting V_j, we can satisfy the unit demands for all the vertices from V_j that e is incident on by assigning them one distinct r-tour each that visits V_j. The unit demands for vertices in X can be satisfied by assigning them one of the remaining r-tours each. This is possible since all the r-tours visit X and the total number of r-tours is $q(T+q)$, which is more than the number of vertices qT incident on e. The cost of this solution is,

$$\sum_{j=1}^{q}(T+q)(|V|/q+|V_j \cup X|) \leq (T+q)|V|(2+\varepsilon q).$$

In the NO case, we show that any feasible solution has cost at least $qT(1 - f_{q,T}(\varepsilon))|V|$, where $f_{q,T}(x)$ is a function parametrized by q and T such that for any fixed value of q and T, $f_{q,T}(x) \to 0$ as $x \to 0$. Consider a solution consisting of K different r-tours $\tau_1, \ldots, \tau_K$. We may assume that $K \leq q^2T$, otherwise the total cost would exceed $q^2T(|V|/q) = qT|V|$ and we would be done ($f_{q,T}(x) \equiv 0$).

Now consider a subset of indices $I \subseteq [K]$ of size $|I| \leq qT - 1$. Let $V(I)$ be the vertices that appear only in the set of r-tours $\{\tau_j \mid j \in I\}$ and in no other r-tours in the solution. Since every r-tour has capacity 1, the total demand of the vertices in $V(I)$, in any scenario, is at most $qT - 1$. Fix any hyperedge $e \in E$. Since G is qT-uniform, and in the scenario s_e we assign a unit demand to each of the qT vertices that e is incident on, not all these vertices can lie in $V(I)$. Thus, $V(I)$ is an independent set in G, and by the property of the NO case, $|V(I)| \leq \varepsilon|V|$.

By a union bound, the total number of vertices in V, which appear in at most $qT - 1$ r-tours, is at most

$$\sum_{\substack{I \subseteq [K], \\ |I| \leq qT-1}} |V(I)|.$$

Since $K \leq q^2T$, the number of index sets I in the above sum is at most $\binom{q^2T}{qT-1} \leq 2^{q^2T}$. Thus, the total number of vertices in V which appear in at most $qT - 1$ r-tours is bounded by $\varepsilon 2^{q^2 T}|V| \stackrel{\text{def}}{=} f_{q,T}(\varepsilon)|V|$. Therefore, the cost of the solution in the NO case is at least $qT(1 - f_{q,T}(\varepsilon))|V|$.

Thus, the hardness of approximation factor obtained is,

$$\frac{qT(1 - f_{q,T}(\varepsilon))}{(T+q)(2+\varepsilon q)}, \tag{8.5}$$

which can be made an arbitrarily large constant by choosing $T = q$ large enough and then ε to be small enough. This completes the analysis.

Notes

The material presented in this chapter is based on the paper "Optimal Inapproximability for Scheduling Problems via Structural Hardness for Hypergraph Vertex Cover" [SS13] joint with Rishi Saket. A preliminary version of the paper appeared at CCC 2013.

Chapter 9

Hardness for Hypergraph VC on k-partite k-uniform hypergraphs

In this chapter, we study the problem of computing the minimum vertex cover on k-uniform k-partite hypergraphs when the k-partition is given. We show that this problem is **NP**-hard to approximate within a factor of $\frac{k}{2} - 1 + \frac{1}{2k} - \varepsilon$. This hardness factor is off from the optimal by an additive constant of at most 1 for $k \geq 4$. Our reduction relies on the *Multi-Layered PCP* of Dinur *et al.* [DGKR05] and uses a gadget – based on biased Long Codes – adapted from the LP integrality gap construction of Aharoni *et al.* [AHK96]. The nature of our reduction requires the analysis of several Long Codes with different biases, for which we prove structural properties of the so called *cross-intersecting* collections of set families.

9.1 Main Results

We study the Minimum Vertex Cover problem on k-partite k-uniform hypergraphs, when the underlying partition is given. Recall that we denote this problem as k-HypVC-Partite. For completeness, we restate our hardness result for approximating k-HypVC-Partite.

Theorem 9.1. *For any $\epsilon > 0$ and integer $k \geq 4$, it is* **NP**-*hard to approximate* k-HypVC-Partite *to within a factor of* $\frac{k}{2} - 1 + \frac{1}{2k} - \epsilon$.

Organization. In section 9.2, we give an overview of the techniques used in this work. Section 9.3 defines and analyzes the above mentioned cross-intersecting set families. Section 9.4 defines the Multi-Layered PCP of Dinur *et al.* [DGKR05], and states their hardness for it. In Section 9.5.1 we describe our reduction and prove Theorem 9.1. Finally, we give a description of the integrality gap construction of Aharoni *et al.* [AHK96] in Section 9.6.

9.2 Techniques

It is helpful to first briefly review the hardness reduction of [DGKR05] for k-HypVC.

The main idea of their construction can be illustrated by the following *gadget*. Consider a domain R and the set of all its subsets $\mathcal{H} = 2^R$. Sample subsets from $\mathcal{H}$ by choosing each element of R independently with probability $1 - 1/k - \varepsilon$ (for some small $\varepsilon > 0$), and let the weight of each subset in $\mathcal{H}$ be its corresponding sampling probability, thus making the sum of all weights to be 1. The set $\mathcal{H}$ along with the associated weights is an example of a biased Long Code over R. Construct a k-uniform hypergraph over the vertex set $\mathcal{H}$ by adding an edge between any k subsets whose intersection is empty. In this hypergraph it is easy to see that every element $r \in R$ yields a corresponding independent set (in the hypergraph) of weight $(1 - 1/k - \varepsilon)$, by choosing all subsets which contain that element. On the other hand, Dinur *et al.* [DGKR05] show via an analysis based on extremal set theory, that any independent set of weight ε must contain k subsets in $\mathcal{H}$ which have a small intersection, thus yielding a special *small* subset of R. This gap of $1 - 1/k - \varepsilon$ vs ε for independent set corresponds to a gap of $1/k + \varepsilon$ vs $1 - \varepsilon$ for the complementary minimum vertex cover objective.

The construction of [DGKR05] combines the above Long Code based gadget with a new *Multi-Layered PCP*. This is a two variable CSP consisting of several *layers* of variables, and constraints between variables from each pair of layers. The work of [DGKR05] shows that it is NP-hard to find a labeling of the variables which satisfies a small fraction of the constraints between *any* two layers, even if there is a labeling that satisfies all the constraints of the instance. The reduction to a k-uniform hypergraph (as an instance of k-HypVC) involves replacing each variable of the PCP with a biased Long Code and adding the edges of the above gadget across different Long Codes.

The starting point for our hardness reduction for k-HypVC-Partite is – as in [DGKR05] – the Multi-Layered PCP. While we do not explicitly construct a stand-alone Long Code based gadget, our reduction can be thought of as adapting the integrality gap construction of Aharoni *et al.* [AHK96] into a Long Code based gadget in a manner that preserves the k-uniformity and k-partiteness of the integrality gap.

Such transformations of integrality gaps into Long Code based gadgets have recently been studied in the works of Raghavendra [Rag08] and Kumar *et al.* [KMTV11], who show this for a wide class of CSPs and their appropriate SDP and LP integrality gaps respectively. These Long Code based gadgets can be combined with a Unique Games instance to yield tight UGC based hardness results, where the reduction is analyzed via the Mossel's *Invariance Principle* [Mos08]. Indeed, for k-HypVC-Partite the work of Guruswami and Saket [GS10b] combines the integrality gap of [AHK96] with (a slight modification) of the approach of Kumar *et al.* [KMTV11] to obtain an optimal UGC based hardness result.

Our reduction, on the other hand, combines Long Codes with the Multi-Layered PCP instead of Unique Games and so we cannot adopt a Invariance Principle based analysis. Thus, in a flavor similar to that of [DGKR05], our analysis is via extremal combinatorics. However, our *gadget* involves several biased Long Codes with different biases and each hyperedge includes vertices from differently biased Long Codes, unlike the construction in [DGKR05]. The different biases are derived from the LP solution to the integrality gap of [AHK96], in such a way that the gap obtained in the gadget corresponds to the value of the integrality gap.

For our analysis, we use structural properties of a *cross-intersecting* collection of set families. A collection of set families is *cross-intersecting* if any intersection of subsets – each chosen from a different family – is large. Variants of this notion have previously been studied in extremal set theory (see *e.g.* [AL09]). We prove an upper bound on the measure of the smallest family in such a collection. This enables a small vertex cover (in the hypergraph of our reduction) to be *decoded* into a good labeling to the Multi-Layered PCP.

9.3 Cross-Intersecting Set Families

We use the notation $[n] = \{1, \ldots, n\}$ and $2^{[n]} = \{F \mid F \subseteq [n]\}$. A subset of $2^{[n]}$ is called a *set family*. We begin by defining cross-intersecting set families:

Definition 9.2. *A collection of k families $\mathcal{F}_1, \ldots, \mathcal{F}_k \subseteq 2^{[n]}$, is called k-wise t-cross-intersecting if for every choice of sets $F_i \in \mathcal{F}_i$ for $i = 1, \ldots, k$, we have $|F_1 \cap \ldots \cap F_k| \geq t$.*

In this section, we prove that if a collection of k families $\mathcal{F}_1, \ldots, \mathcal{F}_k \subseteq 2^{[n]}$ is k-wise t-cross-intersecting, then at least one of the families is small in size, under an appropriately picked measure. Before we formally state the claim in Lemma 9.4, we define the family of p-biased measures on subsets of $[n]$, that we will work with.

Definition 9.3. *Given a bias parameter $0 < p < 1$, we define the measure μ_p on subsets of $[n]$ as: $\mu_p(F) \overset{\text{def}}{=} p^{|F|} \cdot (1-p)^{n-|F|}$. The measure of a family $\mathcal{F}$ is defined as $\mu_p(\mathcal{F}) = \sum_{F \in \mathcal{F}} \mu_p(F)$.*

We can now state the main result of this section.

Lemma 9.4. *For arbitrary $\epsilon, \delta > 0$, there exists some $t = O\left(\frac{1}{\delta^2}\left(\log\frac{1}{\epsilon} + \log\left(1 + \frac{1}{2\delta^2}\right)\right)\right)$ such that the following holds for all positive integers k: Given k numbers $0 < q_i < 1$ such that $\sum_i q_i \geq 1$ and k families, $\mathcal{F}_1, \ldots, \mathcal{F}_k \subseteq 2^{[n]}$, that are k-wise t-cross-intersecting, there exists a j such that $\mu_{1-q_j-\delta}(\mathcal{F}_j) < \epsilon$.*

Proof. Let $\mathcal{F}_1, \ldots, \mathcal{F}_k \subseteq 2^{[n]}$ be a collection of k-wise t-cross-intersecting families. We will specify our choice of $t(\epsilon, \delta)$ at the end of the proof.

For the proof, we introduce an important technique for analyzing cross-intersecting families – the shift operation (see Definition 4.1, pg. 1298 [GGL95]). Given a family $\mathcal{F}$, define the (i,j)-shift as follows:

$$S_{ij}^{\mathcal{F}}(F) = \begin{cases} (F \cup \{i\} \backslash \{j\}) & \text{if } j \in F,\ i \notin F \text{ and } (F \cup \{i\} \backslash \{j\}) \notin \mathcal{F}, \\ F & \text{otherwise.} \end{cases}$$

Let the (i,j)-shift of a family $\mathcal{F}$ be defined as $S_{ij}(\mathcal{F}) = \{S_{ij}^{\mathcal{F}}(F) \mid F \in \mathcal{F}\}$. Given a family $\mathcal{F} \subseteq 2^{[n]}$, we repeatedly apply (i,j)-shift for $1 \leq i < j \leq n$ to $\mathcal{F}$ until we obtain a family that is invariant under these shifts. Such a family is called a *left-shifted family* and we will denote it by $S(\mathcal{F})$.

The following lemma, whose proof is included later in the section, shows that the cross-intersecting property is preserved under left-shifting.

Lemma 9.5. *Consider families $\mathcal{F}_1, \ldots, \mathcal{F}_k \subseteq 2^{[n]}$ that are k-wise t-cross-intersecting. Then, the families $S(\mathcal{F}_1), \ldots, S(\mathcal{F}_k)$ are also k-wise t-cross-intersecting.*

Moreover, by definition, there is a bijection between the sets in $\mathcal{F}$ and $S(\mathcal{F})$ that preserves the size of the set. Thus, for any fixed p, the measures of $\mathcal{F}$ and $S(\mathcal{F})$ are the same under μ_p, i.e., $\mu_p(\mathcal{F}) = \mu_p(S(\mathcal{F}))$. Combining this observation with Lemma 9.5, it suffices to prove that there exists a j such that $\mu_{1-q_j-\delta}(S(\mathcal{F}_j)) < \varepsilon$. Thus, we can assume that the families $\mathcal{F}_1, \ldots, \mathcal{F}_k$ are left-shifted (if not, we can replace then with $S(\mathcal{F}_1), \ldots, S(\mathcal{F}_k)$).

Next, we prove a key structural property about left-shifted cross-intersecting families which states that for at least one of the families, all of its subsets have a dense prefix. A similar fact for a single left-shifted family was shown in [GGL95] (pg. 1311, Lemma 8.3), which was reproved and used in [DGKR05]. However, our case of multiple families with varying biases is different for which we need to prove the following combinatorial lemma.

Lemma 9.6. *Let $q_1, \ldots, q_k \in (0,1)$ be k numbers such that $\sum_i q_i \geq 1$ and let $\mathcal{F}_1, \ldots, \mathcal{F}_k \subseteq 2^{[n]}$ be left-shifted families that are k-wise t-cross-intersecting for some $t \geq 1$. Then, there exists a $j \in [k]$ such that for all sets $F \in \mathcal{F}_j$, there exists a positive integer $r_F \leq n - t$ such that $|F \cap [t + r_F]| > (1 - q_j)(t + r_F)$.*

We defer the proof of this lemma to the end of this section. Assuming this lemma for now, we conclude that there must exist a j such that for all sets $F \in \mathcal{F}_j$, there exists an r_F such that $|F \cap [t + r_F]| > (1 - q_j)(t + r_F)$. We now apply a Chernoff bound argument to deduce that $\mu_{1-q_j-\delta}(\mathcal{F}_j)$ must be small.

Note that $\mu_{1-q_j-\delta}(\mathcal{F}_j)$ is equal to the probability that a random set F chosen according to $\mu_{1-q_j-\delta}$ lies in $\mathcal{F}_j$. Thus, $\mu_{1-q_j-\delta}(\mathcal{F}_j)$ is bounded by the probability that for a random set F chosen according to $\mu_{1-q_j-\delta}$, there exists an r_F that satisfies $|F \cap [t + r_F]| \geq (1 - q_j)(t + r_F)$.

Chernoff bound states that for a set of m independent bernoulli random variables X_i, with $\Pr[X_i = 1] = 1 - q_j - \tau$,

$$\Pr\left[\sum_{i=1}^{m} X_i \geq (1 - q_j)m\right] \leq e^{-2m\tau^2}.$$

Thus, we get that for any $r \geq 0$, $\Pr[|F \cap [t+r]| \geq (1-q_j)(t+r)] \leq e^{-2(t+r)\delta^2}$. Summing over all r, we get that,

$$\mu_{1-q_j-\delta}(\mathcal{F}_j) \leq \sum_{r \geq 0} e^{-2(t+r)\delta^2} \leq \frac{e^{-2t\delta^2}}{1-e^{-2\delta^2}} \leq e^{-2t\delta^2}\left(1+\frac{1}{2\delta^2}\right).$$

Thus, if t is large enough ($t = \Omega\left(\frac{1}{\delta^2}\left(\log\frac{1}{\epsilon} + \log\left(1+\frac{1}{2\delta^2}\right)\right)\right)$ suffices), then $\mu_{1-q_j-\delta}(\mathcal{F})$ must be smaller than ϵ. This completes the proof of Lemma 9.4. □

We now give a proof of Lemma 9.5 that shows that left-shifting preserves the cross-intersecting property.

Proof. (of Lemma 9.5) Given the assumption, we will prove that $S_{ij}(\mathcal{F}_1), \ldots, S_{ij}(\mathcal{F}_k)$ are k-wise t-cross-intersecting. A simple induction then implies the statement of the lemma.

Consider arbitrary sets $F_i \in \mathcal{F}_i$. By our assumption, $|F_1 \cap \ldots \cap F_k| \geq t$. It suffices to prove that $|S_{ij}^{\mathcal{F}_1}(F_1) \cap \ldots \cap S_{ij}^{\mathcal{F}_k}(F_k)| \geq t$. If $j \notin F_1 \cap \ldots \cap F_k$, the claim is true since the only element being removed is j. Thus, for all $l \in [k]$, $j \in F_k$. If for all $l \in [k]$, $S_{ij}^{\mathcal{F}_l}(F_l) = F_l$, the claim is trivial. Thus, let us assume wlog that $S_{ij}^{\mathcal{F}_1}(F_1) \neq F_1$. Thus, $i \notin F_1$ and hence $i \notin F_1 \cap \ldots \cap F_k$. Now, if $i \in S_{ij}^{\mathcal{F}_1}(F_1) \cap \ldots \cap S_{ij}^{\mathcal{F}_k}(F_k)$, we get that j is replaced by i in the intersection and we are done. Thus, we can assume wlog that $i \notin S_{ij}^{\mathcal{F}_2}(F_2)$. This implies that $i \notin F_2$ and $F_2 \cup \{i\} \backslash \{j\} \in \mathcal{F}_2$. Now consider $F_1 \cap (F_2 \cup \{i\} \backslash \{j\}) \cap F_3 \cap \ldots \cap F_k$. Since we are picking one set from each $\mathcal{F}_i$, it must have at least t elements, but this intersection does not contain j and hence it is a subset of $S_{ij}^{\mathcal{F}_1}(F_1) \cap \ldots \cap S_{ij}^{\mathcal{F}_k}(F_k)$, implying that $|S_{ij}^{\mathcal{F}_1}(F_1) \cap \ldots \cap S_{ij}^{\mathcal{F}_k}(F_k)| \geq t$. □

Finally, we give a proof of Lemma 9.6 that gives a structural property of cross-intersecting families.

Proof. (of Lemma 9.6) Let us assume to the contrary that for every $i \in [k]$, there exists a set $F_i \in \mathcal{F}_i$ such that for all $r \geq 0$, $|F_i \cap [t+r]| \leq (1-q_i)(t+r)$. The following combinatorial argument shows that the families $\mathcal{F}_i$ cannot be k-wise t-cross-intersecting.

Let us construct an arrangement of balls and bins where each ball is colored with one of k colors. Create n bins labeled $1, \dots, n$. For each i and for every $x \in [n] \backslash F_i$, we place a ball with color i in the bin labeled x. Note that a bin can have several balls, but they must have distinct colors. Given such an arrangement, we can recover the sets it represents by defining F_i^c to be the set of bins that contain a ball with color i.

For all r, our initial assumption implies that $|F_i^c \cap [t+r]| \geq q_i(t+r)$. Thus, there are at least $\lceil q_i(t+r) \rceil$ balls with color i in bins labeled $1, \dots, t+r$. The total number of balls in bins labeled $1, \dots, t+r$ is,

$$\sum_{i=1}^{k} |F_i^c \cap [t+r]| \;\geq\; \sum_{i=1}^{k} \lceil q_i(t+r) \rceil \;\geq\; \sum_{i=1}^{k} q_i(t+r) \;\geq\; t+r \;\geq\; r+1,$$

where the last two inequalities follow using $\sum_i q_i \geq 1$ and $t \geq 1$.

Next, we describe a procedure to manipulate the above arrangement of balls.

```
**for** $r := 0$ to $n-t$
    **if** bin $t+r$ is empty
    **then if** a bin labeled from 1 to $t-1$ contains a ball **then** move it to bin $t+r$
        **else if** a bin labeled from $t$ to $t+r-1$ contains two balls
            **then** move one of them to bin $t+r$
            **else** output "error"
```

We need the following lemma.

Lemma 9.7. *The above procedure satisfies the following properties:*

1. The procedure never outputs error*.*

2. At every step, any two balls in the same bin have different colors.

3. At step r, define $G_i^{(r)}$ to be the set of labels of the bins that do not contain a ball *of color i. Then, for all $i \in [k]$, $G_i^{(r)} \in \mathcal{F}_i$.*

4. After step r, the bins t to $t+r$ have at least one ball each.

Proof. 1. If it outputs error at step r, there must be at most r balls in bins 1 to $t+r$. At the start of the procedure, there are at least $r+1$ balls in these bins and during the first r steps, the number of balls in these bins remain unchanged. This is a contradiction.

2. Note that this is true at $r = 0$ and a ball is only moved to an empty bin, which proves the claim.

3. We first note that for all $i \in [k]$, $G_i^{(0)} \in \mathcal{F}_i$. Next, observe that for any left-shifted family $\mathcal{F} \subseteq 2^{[n]}$, and $F \in \mathcal{F}$ such that $i \notin F$ and $j \in F$ where $i < j$, the set $(F \cup \{i\} \backslash \{j\})$ must be in $\mathcal{F}$. Whenever we move a ball from bin i to j, we have $i < j$. Since $\mathcal{F}_i$ are left-shifted, by repeated application of the above observation, we get that at step r, $G_i^{(r)} \in \mathcal{F}_i$.

4. Since the procedure never outputs error, at step r, if the bin $t+r$ is empty, the procedure places a ball in it while not emptying any bin labeled between $[t, t+r-1]$. This proves the claim. □

The above lemma implies that at the end of the procedure (after $r = n - t$), there is a ball in each of the bins labeled from $[t, n]$. Thus, the sets $G_i = G_i^{(n-t)}$ satisfy $\cap_i G_i \subseteq [t-1]$ and hence $|\cap_i G_i| \leq t - 1$. Also, we know that $G_i \in \mathcal{F}_i$. Thus, the families $\mathcal{F}_i$ cannot be k-wise t-cross-intersecting. This completes the proof of Lemma 9.6. □

9.4 Multi-Layered PCP

In this section, we describe the Multi-Layered PCP constructed in [DGKR05] and its useful properties. An instance Φ of the Multi-Layered PCP is parametrized by integers $L, R > 1$. The PCP consists of L sets of variables $X_1, \ldots, X_L$. The label set (or range) of the variables in the l^{th} set X_l is a set R_{X_l}, where $|R_{X_l}| = R^{O(L)}$. For any two integers $1 \leq l < l' \leq L$, the PCP has a set of constraints $\Phi_{l,l'}$ in which each constraint depends on one variable $x \in X_l$, and one variable $x' \in X_{l'}$. The constraint (if it exists) between $x \in X_l$ and $x' \in X_{l'}$ $(l < l')$ is denoted and characterized by a projection $\pi_{x \to x'} : R_{X_l} \to R_{X_{l'}}$. A labeling to x and x' satisfies the constraint $\pi_{x \to x'}$ if the projection (via $\pi_{x \to x'}$) of the label assigned to x coincides with the label assigned to x'.

The following useful 'weak-density' property of the Multi-Layered PCP was defined in [DGKR05].

Definition 9.8. *An instance Φ of the Multi-Layered PCP with L layers is* weakly-dense *if for any $\delta > 0$, given $m \geq \lceil \frac{2}{\delta} \rceil$ layers $l_1 < l_2 < \cdots < l_m$ and given any sets $S_i \subseteq X_{l_i}$, for $i \in [m]$ such that $|S_i| \geq \delta |X_{l_i}|$; there always exist two layers $l_{i'}$ and $l_{i''}$ such that the constraints between the variables in the sets $S_{i'}$ and $S_{i''}$ is at least $\frac{\delta^2}{4}$ fraction of the constraints between the sets $X_{l_{i'}}$ and $X_{l_{l''}}$.*

The following inapproximability of the Multi-Layered PCP was proven by Dinur et al. [DGKR05] based on the PCP Theorem [AS98, ALM+98] and Raz's Parallel Repetition Theorem [Raz98].

Theorem 9.9. *There exists a universal constant $\gamma > 0$ such that, for any integer parameters $L, R > 1$, there is a weakly-dense L-layered PCP $\Phi = \bigcup \Phi_{l,l'}$ such that it is NP-hard to distinguish between the following two cases:*

- **YES** Case: *There exists an assignment of labels to the variables of Φ that satisfies all the constraints.*
- **NO** Case: *For every $1 \leq l < l' \leq L$, not more that $1/R^{\gamma}$ fraction of the constraints in $\Phi_{l,l'}$ can be satisfied by any assignment.*

9.5 Hardness Reduction for HypVC-Partite

9.5.1 Construction of the Hypergraph

Fix a $k \geq 3$, an arbitrarily small parameter $\varepsilon > 0$, and let $r = \lceil 10\varepsilon^{-2} \rceil$. We shall construct a $(k+1)$-uniform $(k+1)$-partite hypergraph as an instance of $(k+1)$-HypVC-Partite. Our construction will be a reduction from an instance Φ of the Multi-Layered PCP with number of layers $L = 32\varepsilon^{-2}$, and parameter R which shall be chosen later to be large enough. It involves creating, for each variable of the PCP, several copies of the Long Code endowed with different biased measures as explained below. Our construction is motivated by the integrality gap constructed by Aharoni *et al.* [AHK96] that has been described in Section 9.6.

Over any domain T, a Long Code $\mathcal{H}$ is a collection of all subsets of T, i.e., $\mathcal{H} = 2^T$. A bias $p \in [0,1]$ defines a measure μ_p on $\mathcal{H}$ such that $\mu_p(v) = p^{|v|}(1-p)^{|T \setminus v|}$ for any $v \in \mathcal{H}$. In our construction, we need several different biased measures defined as follows. For all $j = 1, \ldots, r$, define $q_j \overset{\text{def}}{=} \frac{2j}{rk}$, and biases $p_j \overset{\text{def}}{=} 1 - q_j - \varepsilon$. Each p_j defines a biased measure μ_{p_j} over a Long Code over any domain. Next, we define the vertices of the hypergraph.

Vertices. We shall denote the set of vertices by V. Consider a variable x in the layer X_l of the PCP. For $i \in [k+1]$ and $j \in [r]$, let $\mathcal{H}^x_{ij}$ be a Long Code on the domain R_{X_l} endowed with the bias μ_{p_j}, i.e., $\mu_{p_j}(v) = {p_j}^{|v|}(1-p_j)^{|R_{X_l} \setminus v|}$ for all $v \in \mathcal{H}^x_{ij} = 2^{R_{X_l}}$. The set of vertices corresponding to x is $V[x] \overset{\text{def}}{=} \bigcup_{i=1}^{k+1} \bigcup_{j=1}^{r} \mathcal{H}^x_{ij}$. We define the weights on vertices to be proportional to its biased measure in the corresponding Long Code. Formally, for any $v \in \mathcal{H}^x_{ij}$,

$$\mathrm{wt}(v) \overset{\text{def}}{=} \frac{\mu_{p_j}(v)}{L|X_l|r(k+1)}. \tag{9.1}$$

The above conveniently ensures that for any $l \in [L]$,

$$\sum_{x \in X_l} \mathrm{wt}(V[x]) = 1/L \ , \quad \text{and} \quad \sum_{l \in [L]} \sum_{x \in X_l} \mathrm{wt}(V[x]) = 1 \ .$$

In addition to the vertices for each variable of the PCP, the instance also contains $k+1$ *dummy* vertices $d_1, \ldots, d_{k+1}$, each with a very large weight given by $\mathrm{wt}(d_i) \overset{\text{def}}{=} 2$ for $i \in [k+1]$. Clearly, this ensures that the total weight of all the vertices in the hypergraph is $2(k+1)+1$. As we shall see later, the edges shall be defined in such a way that, together with vertex weights, we would be guaranteed that the maximum weight independent set will contain all the dummy vertices. Before defining the edges we define the $(k+1)$ partition $(V_1, \ldots, V_{k+1})$ of V to be:

$$V_i = \left(\bigcup_{l=1}^{L} \bigcup_{x \in X_l} \bigcup_{j=1}^{r} \mathcal{H}_{ij}^{x} \right) \bigcup \{d_i\}, \tag{9.2}$$

for all $i = 1, \ldots, k+1$. We now define the hyperedges of the instance. In the rest of the section, we shall think of the vertices of the long codes as subsets of their respective domains.

Hyperedges. For every pair of variables x and y of the PCP such that there is a constraint $\pi_{x \to y}$, we construct edges as follows.

(1.) Consider all permutations $\sigma : [k+1] \to [k+1]$ and sequences $(j_1, \ldots, j_k, j_{k+1})$ such that, $j_1, \ldots, j_k \in [r] \cup \{0\}$ and $j_{k+1} \in [r]$ such that: $\sum_{i=1}^{k} \mathbb{1}_{\{j_i \neq 0\}} \ q_{j_i} \geq 1$.

(2.) Add all possible hyperedges e such that for all $i \in [k]$:

(2.a) If $j_i \neq 0$ then $e \cap V_{\sigma(i)} =: v_{\sigma(i)} \in \mathcal{H}^{x}_{\sigma(i), j_i}$, and,

(2.b) If $j_i = 0$ then $e \cap V_{\sigma(i)} = d_{\sigma(i)}$ and,

(2.c) $e \cap V_{\sigma(k+1)} =: u_{\sigma(k+1)} \in \mathcal{H}^{y}_{\sigma(k+1), j_{k+1}}$, which satisfy,

$$\pi_{x\to y}\Bigg(\bigcap_{\substack{i:\ i\in[k]\\ j_i\neq 0}} v_{\sigma(i)}\Bigg)\bigcap u_{\sigma(k+1)} = \emptyset. \tag{9.3}$$

Let us denote the hypergraph constructed above by $G(\Phi)$. From the construction it is clear that $G(\Phi)$ is $(k+1)$-partite with partition $V = \cup_{i\in[k+1]}V_i$.

The role of the dummy vertices $\{d_1,\ldots,d_{k+1}\}$ is to ensure that each hyperedge contains exactly $k+1$ vertices – without them we would have hyperedges with fewer than $k+1$ vertices. Also note that the hyperedges are defined in such a way that the set $\{d_1,\ldots,d_{k+1}\}$ is an independent set in the hypergraph. Moreover, since the weight of each dummy vertex d_i is 2, while the total weight of all except the dummy vertices is 1, this implies that any maximum independent set $\mathcal{I}$ contains all the dummy vertices. Thus, $V\setminus\mathcal{I}$ is a minimum vertex cover that does not contain any dummy vertices. For convenience, the analysis of our reduction, presented in the rest of this section, shall focus on the weight of $(\mathcal{I}\cap V)\setminus\{d_1,\ldots,d_{k+1}\}$.

The rest of this section is devoted to proving the following theorem which implies Theorem 9.1.

Theorem 9.10. *Let Φ be the instance of Multi-Layered PCP from which the hypergraph $G(\Phi)$ is derived as an instance of $(k+1)$-HypVC-Partite. Then,*

- *Completeness: If Φ is a YES instance, then there is an independent set $\mathcal{I}^*$ in $G(\Phi)$ such that,*

$$\mathrm{wt}\left(\mathcal{I}^*\cap(V\setminus\{d_1,\ldots,d_{k+1}\})\right)\geq 1-\frac{1}{k}-2\varepsilon.$$

- *Soundness: If Φ is a NO instance, then for all independent sets $\mathcal{I}$ in $G(\Phi)$,*

$$\mathrm{wt}\left(\mathcal{I}\cap(V\setminus\{d_1,\ldots,d_{k+1}\})\right)\leq 1-\frac{k}{2(k+1)}+\varepsilon.$$

The completeness case of the above theorem is proved in Section 9.5.2, and the soundness case in Section 9.5.3.

9.5.2 Completeness

In the completeness case, the instance Φ is a YES instance, i.e., there is a labeling A which maps each variable x in layer X_l to an assignment in R_{X_l} for all $l = 1, \ldots, L$, such that all the constraints of Φ are satisfied.

Consider the set of vertices $\mathcal{I}^*$ which satisfies the following properties:

(1) $d_i \in \mathcal{I}^*$ for all $i = 1, \ldots, k+1$.

(2) For all $l \in [L]$, $x \in X_l$, $i \in [k+1], j \in [r]$,

$$\mathcal{I}^* \cap \mathcal{H}^x_{ij} = \{v \in \mathcal{H}^x_{ij} : A(x) \in v\}. \tag{9.4}$$

Suppose x and y are two variables in Φ with a constraint $\pi_{x\to y}$ between them. Consider any $v \in \mathcal{I}^* \cap V[x]$ and $u \in \mathcal{I}^* \cap V[y]$. The above construction of $\mathcal{I}^*$ along with the fact that the labeling A satisfies the constraint $\pi_{x\to y}$ implies that $A(x) \in v$ and $A(y) \in u$ and $A(y) \in \pi_{x\to y}(v) \cap u$. Therefore, Equation (9.3) of the construction is not satisfied by the vertices in $\mathcal{I}^*$, and so $\mathcal{I}^*$ is an independent set in the hypergraph. By Equation (9.4), the fraction of the weight of the Long Code $\mathcal{H}^x_{ij}$ which lies in $\mathcal{I}^*$ is p_j, for any variable x, $i \in [k+1]$ and $j \in [r]$. Therefore,

$$\frac{\mathrm{wt}(\mathcal{I}^* \cap V[x])}{\mathrm{wt}(V[x])} = \frac{1}{r}\sum_{j=1}^{r} p_j = 1 - \frac{1}{k}\left(1 + \frac{1}{r}\right) - \varepsilon, \tag{9.5}$$

by our setting of p_j in Section 9.5.1. The above yields that

$$\mathrm{wt}\left(\mathcal{I}^* \cap (V \setminus \{d_1, \ldots, d_{k+1}\})\right) = 1 - \frac{1}{k}\left(1 + \frac{1}{r}\right) - \varepsilon \geq 1 - \frac{1}{k} - 2\varepsilon, \tag{9.6}$$

for a small enough value of $\varepsilon > 0$, and our setting of the parameter r.

9.5.3 Soundness

For the soundness analysis we have that Φ is a NO instance as given in Theorem 9.9, and we wish to prove that the size of the maximum weight independent set in $G(\Phi)$ is appropriately small. For a contradiction, we assume that there is a maximum independent set $\mathcal{I}$ in $G(\Phi)$ such that,

$$\mathrm{wt}(\mathcal{I} \cap (V \setminus \{d_1, \ldots, d_{k+1}\})) \geq 1 - \frac{k}{2(k+1)} + \varepsilon. \tag{9.7}$$

Define the set of variables X' to be as follows:

$$X' \stackrel{\text{def}}{=} \left\{ x \text{ a variable in } \Phi : \frac{\mathrm{wt}(\mathcal{I} \cap V[x])}{\mathrm{wt}(V[x])} \geq 1 - \frac{k}{2(k+1)} + \frac{\varepsilon}{2} \right\}. \tag{9.8}$$

An averaging argument shows that $\mathrm{wt}(\bigcup_{x \in X'} V[x]) \geq \varepsilon/2$. A further averaging implies that there are $\frac{\varepsilon}{4}L = \frac{8}{\varepsilon}$ layers of Φ such that $\frac{\varepsilon}{4}$ fraction of the variables in each of these layers belong to X'. Applying the Weak Density property of Φ given by Definition 9.8 and Theorem 9.9 yields two layers $X_{l'}$ and $X_{l''}$ ($l' < l''$) such that $\frac{\varepsilon^2}{64}$ fraction of the constraints between them are between variables in X'. The rest of the analysis shall focus on these two layers and for convenience we shall denote $X' \cap X_{l'}$ by X and $X' \cap X_{l''}$ by Y, and denote the respective label sets by R_X and R_Y.

Consider any variable $x \in X$. For any $i \in [k+1], j \in [r]$, call a Long Code $\mathcal{H}^x_{ij}$ *significant* if $\mu_{p_j}(\mathcal{I} \cap \mathcal{H}^x_{ij}) \geq \frac{\varepsilon}{2}$. From Equation (9.8) and an averaging argument we obtain that,

$$\left|\{(i,j) \in [k+1] \times [r] : \mathcal{H}^x_{ij} \text{ is } \textit{significant.}\}\right| \geq \left(1 - \frac{k}{2(k+1)}\right)(r(k+1)) = \frac{rk}{2} + r. \tag{9.9}$$

Using an analogous argument we obtain a similar statement for every variable $y \in Y$ and corresponding Long Codes $\mathcal{H}^y_{ij}$. The following structural lemma follows from the above bound.

Lemma 9.11. *Consider any variable $x \in X$. Then there exists a sequence $(j_1, \ldots, j_{k+1})$ with $j_i \in [r] \cup \{0\}$ for $i \in [k+1]$; such that the Long Codes $\{\mathcal{H}^x_{i,j_i} \mid i \in [k+1]$ where $j_i \neq 0\}$, are all* significant. *Moreover,*

$$\sum_{i=1}^{k+1} j_i \geq \frac{rk}{2} + r \ . \tag{9.10}$$

Proof. For all $i \in [k+1]$ choose j_i as follows: if none of the Long Codes $\mathcal{H}^x_{ij}$ for $j \in [r]$ are *significant* then let $j_i \overset{\text{def}}{=} 0$, otherwise let $j_i \overset{\text{def}}{=} \max\{j \in [r] : \mathcal{H}^x_{ij}$ is *significant*$\}$. It is easy to see that j_i is an upper bound on the number of significant Long Codes in $\{\mathcal{H}^x_{ij}\}_j$. Therefore,

$$\begin{aligned} \sum_{i=1}^{k+1} j_i &\geq \left|\{(i,j) \in [k+1] \times [r] : \mathcal{H}^x_{ij} \text{ is } \textit{significant.}\}\right| \\ &\geq \frac{rk}{2} + r \qquad \text{(From Equation (9.9))}, \end{aligned} \tag{9.11}$$

which proves the lemma. □

Next, we define the decoding procedure to define a label for any given variable $x \in X$.

Labeling for variable $x \in X$

The label $A(x)$ for each variable $x \in X$ is chosen independently via the following three step (randomized) procedure.

Step 1. Choose a sequence $(j_1, \ldots, j_{k+1})$ yielded by Lemma 9.11 applied to x.

Step 2. Choose an element i_0 uniformly at random from $[k+1]$.

Before describing the third step of the procedure, we require the following lemma.

Lemma 9.12. *There exist vertices $v_i \in \mathcal{I} \cap \mathcal{H}^x_{ij_i}$ for every $i : i \in [k+1] \setminus \{i_0\}, j_i \neq 0$, and an integer $t := t(\varepsilon)$ satisfying:*

$$\left| \bigcap_{\substack{i:i\in[k+1]\setminus\{i_0\},\\ j_i\neq 0}} v_i \right| < t\ . \tag{9.12}$$

Proof. Since $j_{i_0} \leq r$ it is easy to see,

$$\sum_{i\in[k+1]\setminus\{i_0\}} j_i \geq \frac{rk}{2} \ \Rightarrow \sum_{\substack{i:i\in[k+1]\setminus\{i_0\},\\ j_i\neq 0}} q_{j_i} \geq 1. \tag{9.13}$$

Moreover, since the sequence $(j_1, \dots, j_{k+1})$ was obtained by Lemma 9.11 applied to x, we know that $\mu_{p_{j_i}}(\mathcal{I} \cap \mathcal{H}^x_{ij_i}) \geq \frac{\varepsilon}{2}, \forall i : i \in [k+1] \setminus \{i_0\}, j_i \neq 0$. Combining this with Equation (9.13) and Lemma 9.4 we obtain that for some integer $t := t(\varepsilon)$ the collection of set families $\{\mathcal{H}^x_{ij_i} : i \in [k+1] \setminus \{i_0\}, j_i \neq 0\}$ is not k'-wise t-cross-intersecting, where $k' = |\{i \in [k+1] \setminus \{i_0\} : j_i \neq 0\}|$. This proves the lemma. $\square$

The third step of the labeling procedure is as follows:

Step 3. Apply Lemma 9.12 to obtain the vertices $v_i \in \mathcal{I} \cap \mathcal{H}^x_{ij_i}$ for every $i : i \in [k+1] \setminus \{i_0\}, j_i \neq 0$ satisfying Equation (9.12). Define $B(x)$ as,

$$B(x) \overset{\text{def}}{=} \bigcap_{\substack{i:i\in[k+1]\setminus\{i_0\},\\ j_i\neq 0}} v_i, \tag{9.14}$$

noting that $|B(x)| < t$. Assign a random label from $B(x)$ to the variable x and call the assigned label $A(x)$.

Labeling for variable $y \in Y$

After labeling the variables $x \in X$ via the procedure above, we construct a labeling $A(y)$ for any variable $y \in Y$ by defining,

$$A(y) \overset{\text{def}}{=} \text{argmax}_{a \in R_Y} \left|\{x \in X \cap N(y) \ \mid \ a \in \pi_{x \to y}(B(x))\}\right|, \tag{9.15}$$

where $N(y)$ is the set of all variables that have a constraint with y. The above process selects a label for y which lies in maximum number of projections of $B(x)$ for variables $x \in X$ which have a constraint with y.

The rest of this section is devoted to lower bounding the number of constraints satisfied by the labeling process, and thus obtaining a contradiction to the fact that Φ is a NO instance.

Lower bounding the number of satisfied constraints

Fix a variable $y \in Y$. Let $U(y) \overset{\text{def}}{=} X \cap N(y)$, i.e., the variables in X which have a constraint with y. Further, define the set $P(y) \subseteq [k+1]$ as follows,

$$P(y) = \{i \in [k+1] \ \mid \ \exists j \in [r] \text{ such that } \mu_{p_j}(\mathcal{I} \cap \mathcal{H}_{ij}^{y}) \geq \varepsilon/2\}. \tag{9.16}$$

In other words, $P(y)$ is the set of all those indices in $[k+1]$ such that there is a *significant* Long Code corresponding to each of them. Applying Equation (9.9) to y we obtain that there at least $\frac{r(k+2)}{2}$ *significant* Long Codes corresponding to y, and therefore $|P(y)| \geq \frac{k+2}{2} \geq 1$. Next we define subsets of $U(y)$ depending on the outcome of Step 2 in the labeling procedure for variables $x \in U(y)$. For $i \in [k+1]$ define,

$$U(i, y) \overset{\text{def}}{=} \{x \in U(y) \ \mid \ i \text{ was chosen in Step 2 of the labeling procedure for } x\}, \tag{9.17}$$

and,

$$U^*(y) \overset{\text{def}}{=} \bigcup_{i \in P(y)} U(i, y). \tag{9.18}$$

Note that $\{U(i,y)\}_{i\in[k+1]}$ is a partition of $U(y)$. Also, since $|P(y)| \geq \frac{k+1}{2}$, and the labeling procedure for each variable x chooses the index in Step 2 uniformly and independently at random, we have,

$$\mathbb{E}[|U^*(y)|] \geq \frac{|U(y)|}{2}, \tag{9.19}$$

where the expectation is over the random choice of the indices in Step 2 of the labeling procedure for all $x \in U(y)$. Before continuing, we need the following simple lemma (proved as Claim 5.4 in [DGKR05]).

Lemma 9.13. *Let $A_1, \ldots, A_N$ be a collection of N sets, each of size at most $T \geq 1$. If there are not more than D pairwise disjoint sets in the collection, then there is an element that is contained in at least $\frac{N}{TD}$ sets.*

Now consider any $i' \in P(y)$ such that $U(i', y) \neq \emptyset$, and a variable $x \in U(i', y)$. Since $i' \in P(y)$, there is a *significant* Long Code $\mathcal{H}^y_{i' j'}$ for some $j' \in [r]$. Furthermore, since $\mathcal{I}$ is an independent set there cannot be a $u \in \mathcal{I} \cap \mathcal{H}^y_{i',j'}$ such that $\pi_{x\to y}(B(x)) \cap u = \emptyset$, otherwise the following set of $k+1$ vertices,

$$\{v_i \mid i \in [k+1] \setminus \{i'\}, j_i \neq 0\} \cup \{d_i \mid i \in [k+1] \setminus \{i'\}, j_i = 0\} \cup \{u\}$$

form an edge in $\mathcal{I}$, where v_i, j_i ($i \in [k+1]$) are as constructed in the labeling procedure for x.

Consider the collection of sets $\pi_{x\to y}(B(x))$ for all $x \in U(i', y)$. Clearly, each set is of size less than t. Let D be the maximum number of disjoint sets in this collection. Each disjoint set independently reduces the measure of $\mathcal{I} \cap \mathcal{H}^y_{i',j'}$ by a factor of $(1-(1-p_{j'})^t)$. However, since $\mu_{p_{j'}}(\mathcal{I} \cap \mathcal{H}^y_{i',j'})$ is at least $\frac{\varepsilon}{2}$, this implies that

D is at most $\log(\frac{\varepsilon}{2})/\log(1-(2/rk)^t)$, since $p_{j'} \leq 1-\frac{2}{rk}$. Moreover, since t and r depends only on ε, the upper bound on D also depends only on ε.

Therefore by Lemma 9.13, there is an element $a \in R_Y$ such that $a \in \pi_{x\to y}(B(x))$ for at least $\frac{1}{Dt}$ fraction of $x \in U(i', y)$. Noting that this bound is independent of j' and that $\{U(i', y)\}_{i' \in P(y)}$ is a partition of $U^*(y)$, we obtain that there is an element $a \in R_Y$ such that $a \in \pi_{x\to y}(B(x))$ for at least $\frac{1}{(k+1)Dt}$ fraction of $x \in U^*(y)$. Therefore, in Step 3 of the labeling procedure when a label $A(x)$ is chosen uniformly at random from $B(x)$, in expectation, $a = \pi_{x\to y}(A(x))$ for at least $\frac{1}{(k+1)Dt^2}$ fraction of $x \in U^*(y)$. Combining this with Equation (9.19) gives us that there is a labeling to the variables in X and Y, which satisfies at least $\frac{1}{2(k+1)Dt^2}$ fraction of the constraints between variables in X and Y which is in turn at least $\frac{\varepsilon^2}{64}$ fraction of the constraints between the layers $X_{l'}$ and $X_{l''}$. Since D and t depend only on ε, choosing the parameter R of Φ to be large enough we obtain a contradiction to our assumed lower bound on the size of the independent set. Therefore in the Soundness case, for any independent set $\mathcal{I}$,

$$\mathrm{wt}(\mathcal{I} \cap (V \setminus \{d_1, \ldots, d_{k+1}\})) \leq 1 - \frac{k}{2(k+1)} + \varepsilon.$$

Combining the above with Equation (9.6) of the analysis in the Completeness case yields a factor $\frac{k^2}{2(k+1)} - \delta$ (for any $\delta > 0$) hardness for approximating $(k+1)$-HypVC-Partite .

Thus, we obtain a factor $\frac{k}{2}-1+\frac{1}{2k}-\delta$ hardness for approximating k-HypVC-Partite.

9.6 Integrality Gap construction of Aharoni *et al.*

Our construction of the dictatorship test is motivated by the integrality gap for k-HypVC-Partite constructed by Aharoni *et al.* [AHK96]. Hence, for completeness, we describe the natural linear programming (LP) relaxation for k-HypVC-Partite and the $\frac{k}{2} - o(1)$ integrality gap construction from [AHK96] in this section.

Let $G = (V, E)$ be a k-uniform hypergraph. Let $h(v)$ be a real variable for every vertex $v \in V$. Figure 9.1 describes LP, the natural relaxation for vertex cover in hypergraphs.

$$\begin{array}{lll} \text{minimize} & \sum_{v \in V} h(v) & \\ & & \\ \text{subject to} & \sum_{v_i \in e} h(v_i) \geq 1 & \forall e = \{v_1, \ldots, v_k\} \in E, \\ & 1 \geq h(v) \geq 0 & \forall v \in V. \end{array}$$

Figure 9.1: Relaxation LP for Hypergraph Vertex Cover.

We now restate the result of Aharoni *et al.*

Theorem 9.14 (Aharoni *et al.* [AHK96])**.** *The integrality gap of* LP *for an instance of* k-HypVC-Partite *is at least* $k/2 - o(1)$.

Integrality Gap Construction. The hypergraph that is constructed is unweighted. Let r be a (large) positive integer. The vertex set V of the hypergraph is partitioned into subsets $V_1, \ldots, V_k$ where, for all $i = 1, \ldots, k$,

$$V_i = \{x_{ij} \mid j = 1, \ldots, r\} \cup \{y_{il} \mid l = 1, \ldots, rk + 1\}. \tag{9.20}$$

Before we define the hyperedges, for convenience we shall define the LP solution. The LP values of the vertices are as given by the function $h : V \to [0, 1]$ as follows: for all $i = 1, \ldots, k$,

$$\begin{aligned} h(x_{ij}) &= \frac{2j}{rk}, \qquad \forall j = 1, \ldots, r \\ h(y_{il}) &= 0, \qquad \forall l = 1, \ldots, rk + 1. \end{aligned}$$

The set of hyperedges is naturally defined to be the set of all possible hyperedges, choosing exactly one vertex from each V_i such that the sum of the LP values of the corresponding vertices is at least 1. Formally,

$$E = \{e \subseteq V \mid \forall i \in [k],\ |e \cap V_i| = 1 \text{ and } \sum_{v \in e} h(v) \geq 1\}. \tag{9.21}$$

Clearly the graph is k-uniform and k-partite with $\{V_i\}_{i\in[k]}$ being the k-partition of V.

The value of the LP solution is

$$\sum_{v \in V} h(v) = k \sum_{j \in [r]} \frac{2j}{rk} = r + 1. \tag{9.22}$$

Now let V' be a minimum vertex cover in the hypergraph. To lower bound the size of the minimum vertex cover, we first note that the set $\{v \in V \mid h(v) > 0\}$ is a vertex cover of size rk, and therefore $|V'| \leq rk$. Also, for any $i \in [k]$ the vertices $\{y_{il}\}_{l \in [rk+1]}$ have the same neighborhood. Therefore, we can assume that V' has no vertex y_{il}, otherwise it will contain at least $rk + 1$ such vertices.

For all $i \in [k]$ let define indices $j_i \in [r] \cup \{0\}$ as follows:

$$j_i = \begin{cases} 0 & \text{if: } \forall j \in [r],\ x_{ij} \in V', \\ \max\{j \in [r] \mid x_{ij} \notin V'\} & \text{otherwise.} \end{cases} \tag{9.23}$$

It is easy to see that since V' is a vertex cover,

$$\sum_{i \in [k]} h(x_{ij_i}) < 1,$$

which implies,

$$\sum_{i \in [k]} j_i < \frac{rk}{2}.$$

Also, the size of V' is lower bounded by $\sum_{i\in[k]}(r - j_i)$. Therefore,

$$|V'| \geq \sum_{i\in[k]}(r - j_i) \geq rk - \sum_{i\in[k]} j_i \geq rk - \frac{rk}{2} = \frac{rk}{2}. \tag{9.24}$$

The above combined with the value of the LP solution yields an integrality gap of $\frac{rk}{2(r+1)} \geq \frac{k}{2} - o(1)$ for large enough r.

Notes

The material presented in this chapter is based on the paper "Nearly Optimal **NP**-hardness of Vertex Cover on k-Uniform k-Partite Hypergraphs", joint with Rishi Saket. A preliminary version of this paper appeared at APPROX 2011 [SS11].

Bibliography

[ABN11] Ittai Abraham, Yair Bartal, and Ofer Neiman. Advances in metric embedding theory. *Advances in Mathematics*, 228(6):3026 – 3126, 2011. 12

[ABS10] Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 563–572, 2010. 113

[ACL06] Reid Andersen, Fan R. K. Chung, and Kevin J. Lang. Local graph partitioning using pagerank vectors. In *FOCS'06: Proc. 47th Ann. IEEE Symp. Foundations of Computer Science*, pages 475–486, 2006. 12

[AHK96] R. Aharoni, R. Holzman, and M. Krivelevich. On a theorem of Lovász on covers in r-partite hypergraphs. *Combinatorica*, 16(2):149–174, 1996. 123, 146, 147, 148, 149, 156, 165, 166

[AK07] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. In *STOC '07: Proc. 39th Ann. ACM Symp. Theory of Computing*, pages 227–236, 2007. 11, 12, 13, 14, 31, 41, 44

[AKS11] Per Austrin, Subhash Khot, and Muli Safra. Inapproximability of vertex cover and independent set in bounded degree graphs. *Theory of Computing*, 7(1):27–43, 2011. 115

[Al′59] S. Ja. Al′per. Asymptotic values of best approximation of analytic functions in a complex domain. *Uspehi Mat. Nauk*, 14(1 (85)):131–134, 1959. 101

[AL08] Reid Andersen and Kevin J. Lang. An algorithm for improving graph partitions. In *Proceedings of the nineteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '08, pages 651–660, 2008. 11

[AL09] Noga Alon and Eyal Lubetzky. Uniformly cross intersecting families. *Combinatorica*, 29(4):389–431, 2009. 149

[ALM$^+$98] Sanjeev Arora, Carsten Lund, Rajeev Motwani, Madhu Sudan, and Mario Szegedy. Proof verification and the hardness of approximation problems. *Journal of the ACM*, 45(3):501–555, 1998. 2, 4, 112, 155

[AM85] Noga Alon and V. D. Milman. λ_1, isoperimetric inequalities for graphs, and superconcentrators. *J. Comb. Theory, Ser. B*, 38(1):73–88, 1985. 9

[And81] Jan-Erik Andersson. Approximation of ex by rational functions with concentrated negative poles. *Journal of Approximation Theory*, 32(2):85 – 95, 1981. 14

[AP09] Reid Andersen and Yuval Peres. Finding sparse cuts locally using evolving sets. In *STOC '09: Proc. 41st Ann. ACM Symp. Theory of Computing*, pages 235–244, 2009. 12, 21

[ARV09] S. Arora, S. Rao, and U. V. Vazirani. Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM*, 56(2):1–37, 2009. 11, 44

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of NP. *Journal of the ACM*, 45(1):70–122, 1998. 2, 4, 112, 155

[BBH$^+$12] Boaz Barak, Fernando G.S.L. Brandao, Aram W. Harrow, Jonathan Kelner, David Steurer, and Yuan Zhou. Hypercontractivity, sum-of-squares proofs, and their applications. In *Proceedings of the ACM Symposium on the Theory of Computing*, pages 307–326, 2012. 113

[BGH$^+$11] B. Barak, P. Gopalan, J. Håstad, R. Meka, P. Raghavendra, and D. Steurer. Making the long code shorter, with applications to the unique games conjecture. *CoRR*, abs/1111.0405, 2011. 113

[BGS98] Mihir Bellare, Oded Goldreich, and Madhu Sudan. Free bits, pcps, and nonapproximability—towards tight results. *SIAM J. Comput.*, 27(3):804–915, June 1998. 4, 112

[Bha96] Rajendra Bhatia. *Matrix Analysis (Graduate Texts in Mathematics)*. Springer, 1996. 33, 36

[BK09] N. Bansal and S. Khot. Optimal long code test with one free bit. In *Proceedings of the Annual Symposium on Foundations of Computer Science*, pages 453–462, 2009. 113, 115, 119, 120

[BK10] N. Bansal and S. Khot. Inapproximability of hypergraph vertex cover and applications to scheduling problems. In *Proceedings of the International Colloquium on Automata, Languages and Programming*, pages 250–261, 2010. 113, 115, 116, 117, 118, 119, 120, 125, 129, 130

[BM05] Gregory Beylkin and Lucas Monzón. On approximation of functions by exponential sums. *Applied and Computational Harmonic Analysis*, 19(1):17 – 48, 2005. 17, 85, 86, 90

[BM10] Gregory Beylkin and Lucas Monzón. Approximation by exponential sums revisited. *Applied and Computational Harmonic Analysis*, 28(2):131 – 149, 2010. Special Issue on Continuous Wavelet Transform in Memory of Jean Morlet, Part I. 17, 85, 86

[BRS11] Boaz Barak, Prasad Raghavendra, and David Steurer. Rounding semidefinite programming hierarchies via global correlation. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 472–481, 2011. 113

[Che66] E. W. Cheney. *Introduction to approximation theory / E.W. Cheney.* McGraw-Hill, New York :, 1966. 99, 100, 106

[Chu97] Fan R.K. Chung. *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92).* American Mathematical Society, 1997. 43

[CMM06a] Moses Charikar, Konstantin Makarychev, and Yury Makarychev. Near-optimal algorithms for unique games. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, STOC '06, pages 205–214, 2006. 113

[CMM06b] Eden Chlamtac, Konstantin Makarychev, and Yury Makarychev. How to play unique games using embeddings. In *Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '06, pages 687–696, 2006. 113

[CMV69] W.J Cody, G Meinardus, and R.S Varga. Chebyshev rational approximations to e^{-x} in $[0, \infty)$ and applications to heat-conduction problems. *Journal of Approximation Theory*, 2(1):50 – 65, 1969. 13

[DGK02] I. Dinur, V. Guruswami, and S. Khot. Vertex cover on k-uniform hypergraphs is hard to approximate within factor $(k - 3 - \epsilon)$. *Electronic Colloquium on Computational Complexity (ECCC)*, (027), 2002. 115, 129

[DGKR05] Irit Dinur, Venkatesan Guruswami, Subhash Khot, and Oded Regev. A new multilayered PCP and the hardness of hypergraph vertex cover. *SIAM J. Comput.*, 34:1129–1146, May 2005. 115, 124, 129, 146, 147, 148, 149, 151, 154, 155, 164

[DKPS10] I. Dinur, S. Khot, W. Perkins, and S. Safra. Hardness of finding independent sets in almost 3-colorable graphs. In *Proceedings of the Annual Symposium on Foundations of Computer Science*, pages 212–221, 2010. 129, 130, 131, 132

[DS05] Irit Dinur and Samuel Safra. On the hardness of approximating minimum vertex cover. *Annals of Mathematics*, 162(1):439–485, 2005. 115, 129, 132

[EH05] Jasper vanden Eshof and Marlis Hochbruck. Preconditioning lanczos approximations to the matrix exponential. *SIAM J. Sci. Comput.*, 27:1438–1457, November 2005. 14, 61, 62

[Ehl73] Byron L. Ehle. A-stable methods and Padé approximations to the exponential. *Siam J. on Mathematical Analysis*, 4(4):671–680, 1973. 13

[FGL$^+$96] Uriel Feige, Shafi Goldwasser, László Lovász, Shmuel Safra, and Mario Szegedy. Interactive proofs and the hardness of approximating cliques. *Journal of the ACM*, 43(2):268–292, 1996. 112

[FMO$^+$03] T. Feder, R. Motwani, L. O'Callaghan, R. Panigrahy, and D. Thomas. Online distributed predicate evaluation. Technical Report, Stanford University, 2003. 123

[Fri98] E. Friedgut. Boolean functions with low average sensitivity depend on few coordinates. *Combinatorica*, 18(1):27–35, 1998. 128, 132

[GGL95] R. L. Graham, M. Grötschel, and L. Lovász, editors. *Handbook of combinatorics (vol. 2)*. MIT Press, Cambridge, MA, USA, 1995. 150, 151

[GHM$^+$11] V. Guruswami, J. Hstad, R. Manokaran, P. Raghavendra, and M. Charikar. Beating the random ordering is hard: Every ordering csp is approximation resistant. *SIAM Journal on Computing*, 40(3):878–914, 2011. 113

[GKP94] Ronald L. Graham, Donald E. Knuth, and Oren Patashnik. *Concrete Mathematics: A Foundation for Computer Science*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2nd edition, 1994. 91

[GKP07] N. Garg, A. Kumar, and V. Pandit. Order scheduling models: hardness and algorithms. In *Proceedings of the 27th international conference on Foundations of Software Technology and Theoretical Computer Science*, pages 96–107, 2007. 116

[GKS10] P. Gopalan, S. Khot, and R. Saket. Hardness of reconstructing multivariate polynomials over finite fields. *SIAM Journal of Computing*, 39(6):2598–2621, 2010. 129, 134

[GMR08] Venkatesan Guruswami, Rajsekar Manokaran, and Prasad Raghavendra. Beating the random ordering is hard: Inapproximability of maximum acyclic subgraph. In *Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '08, pages 573–582, 2008. 113

[GNR12] A. Gupta, V. Nagarajan, and R. Ravi. Approximation algorithms for VRP with stochastic demands. *Operations Research*, 60(1):123–127, 2012. 119

[GNS12] I. Gørtz, V. Nagarajan, and R. Saket. Stochastic vehicle routing with recourse. In *Proceedings of the International Colloquium on Automata, Languages and Programming*, pages 411–423, 2012. 115, 116, 118, 119, 120, 125, 130

[Gol01] O. Goldreich. Using the FGLSS-reduction to prove inapproximability results for minimum vertex cover in hypergraphs. *Electronic Colloquium on Computational Complexity (ECCC)*, (102), 2001. 115

[GS10a] G. Gottlob and P. Senellart. Schema mapping discovery from data instances. *J. ACM*, 57(2), January 2010. 123

[GS10b] V. Guruswami and R. Saket. On the inapproximability of vertex cover on k-partite k-uniform hypergraphs. In *ICALP*, pages 360–371, 2010. 123, 124, 148

[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 *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 482–491, 2011. 113

[GT06] Anupam Gupta and Kunal Talwar. Approximating unique games. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, SODA '06, pages 99–106, 2006. 113

[Hal02] E. Halperin. Improved approximation algorithms for the vertex cover problem in graphs and hypergraphs. *SIAM Journal of Computing*, 31(5):1608–1623, 2002. 114

[Hås99] Johan Håstad. Clique is hard to approximate within n to the power 1-epsilon. *Acta Mathematica*, 182:105–142, 1999. 112

[Hås01] Johan Håstad. Some optimal inapproximability results. *J. ACM*, 48(4):798–859, 2001. 4, 112

[HL97] Marlis Hochbruck and Christian Lubich. On krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 34(5):1911–1925, October 1997. 15, 19, 53, 98

[Hol02a] J. Holmerin. Improved inapproximability results for vertex cover on k-uniform hypergraphs. In *Proceedings of the International Colloquium on Automata, Languages and Programming*, pages 1005–1016, 2002. 115

[Hol02b] J. Holmerin. Vertex cover on 4-regular hyper-graphs is hard to approximate within 2-epsilon. In *Proceedings of the ACM Symposium on the Theory of Computing*, pages 544–552, 2002. 115

[IPS05] G. Iyengar, David J. Phillips, and Clifford Stein. Approximation algorithms for semidefinite packing problems with applications to maxcut and graph coloring. In *IPCO'05: Proc. 11th Conf. Integer Programming and Combinatorial Optimization*, pages 152–166, 2005. 13, 14

[IPS11] G. Iyengar, D. J. Phillips, and C. Stein. Approximating semidefinite packing programs. *SIAM Journal on Optimization*, 21(1):231–268, 2011. 13, 14

[IPZ01] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4), December 2001. 1, 113

[ISOY05] L. Ilie, R. Solis-Oba, and S. Yu. Reducing the size of NFAs by using equivalences and preorders. In *Proc. CPM*, pages 310–321, 2005. 123

[JJUW11] Rahul Jain, Zhengfeng Ji, Sarvagya Upadhyay, and John Watrous. QIP = PSPACE. *J. ACM*, 58(6):30:1–30:27, December 2011. 13

[Kal07] Satyen Kale. Efficient algorithms using the multiplicative weights update method. Technical report, Princeton University, Department of Computer Science, 2007. 13, 14

[Kar72] R. M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–103, 1972. 114

[Kar09] George Karakostas. A better approximation ratio for the vertex cover problem. *ACM Trans. Algorithms*, 5(4):41:1–41:8, November 2009. 114

[Kho02] S. Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the ACM Symposium on the Theory of Computing*, pages 767–775, 2002. 4, 113, 115, 129

[KKMO07] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for max-cut and other 2-variable csps? *SIAM J. Comput.*, 37(1):319–357, April 2007. 113

[KMP11] Ioannis Koutis, Gary L. Miller, and Richard Peng. A nearly-m log n time solver for SDD linear systems. In *Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, FOCS '11, pages 590–598, 2011. 14, 16, 64

[KMTV11] Amit Kumar, Rajsekar Manokaran, Madhur Tulsiani, and Nisheeth K. Vishnoi. On LP-based approximability for strict CSPs. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, pages 1560–1573, 2011. 113, 115, 116, 124, 148

[KOLS13] Jonathan A. Kelner, Lorenzo Orecchia, Yin Tat Lee, and Aaron Sidford. An almost-linear-time algorithm for approximate max flow in undirected graphs, and its multicommodity generalizations. *CoRR*, abs/1304.2338, 2013. 11

[KOSZ13] Jonathan A. Kelner, Lorenzo Orecchia, Aaron Sidford, and Zeyuan Allen Zhu. A simple, combinatorial algorithm for solving SDD systems in nearly-linear time. In *Proceedings of the 45th annual ACM symposium on Symposium on theory of computing*, STOC '13, pages 911–920, 2013. 11, 14, 16

[KR08] S. Khot and O. Regev. Vertex cover might be hard to approximate to within $2 - \epsilon$. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008. 113, 115, 119

[KRV06] Rohit Khandekar, Satish Rao, and Umesh Vazirani. Graph partitioning using single commodity flows. In *STOC '06: Proc. 38th Ann. ACM Symp. Theory of Computing*, pages 385–390, 2006. 11

[KS09] S. Khot and R. Saket. SDP integrality gaps with local ℓ_1-embeddability. In *Proceedings of the Annual Symposium on Foundations of Computer Science*, pages 565–574, 2009. 113

[KS12] S. Khot and R. Saket. Hardness of finding independent sets in almost q-colorable graphs. In *Proceedings of the Annual Symposium on Foundations of Computer Science*, pages 380–389, 2012. 129, 130

[KV05] Subhash A. Khot and Nisheeth K. Vishnoi. The unique games conjecture, integrality gap for cut problems and embeddability of negative type metrics into $\ell 1$. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '05, pages 53–62, 2005. 113

[KVV00] R. Kannan, S. Vempala, and A. Vetta. On clusterings-good, bad and spectral. In *FOCS'00: Proc. 41th Ann. IEEE Symp. Foundations of Computer Science*, page 367, 2000. 9, 10

[LLP05] J. Leung, H. Li, and M. Pinedo. *Multidisciplinary Scheduling: Theory and Applications. Chapter "Order Scheduling Models: An Overview"*. 2005. 116

[LLP07] J. Leung, H. Li, and M. Pinedo. Scheduling orders for multiple product types to minimize total weighted completion time. *Discrete Appl. Math.*, 155(8):945–970, April 2007. 116

[Lov75] L. Lovász. On minimax theorems of combinatorics. *Doctoral Thesis, Mathematiki Lapok*, 26:209–264, 1975. 123

[Mąd10] Aleksander Mądry. Fast approximation algorithms for cut-based problems in undirected graphs. In *Proceedings of the 2010 IEEE 51st Annual*

Symposium on Foundations of Computer Science, FOCS '10, pages 245–254, 2010. 12

[MOO10] Elchanan Mossel, Ryan ODonnell, and Krzysztof Oleszkiewicz. Noise stability of functions with low influences: invariance and optimality. *Annals of Mathematics*, 171(1):295–341, 2010. 113

[Mos08] Elchanan Mossel. Gaussian bounds for noise correlation of functions and tight analysis of long codes. In *Proceedings of the 2008 49th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '08, pages 156–165, 2008. 129, 148

[MQS$^+$10] M. Mastrolilli, M. Queyranne, A. S. Schulz, O. Svensson, and N. A. Uhan. Minimizing the sum of weighted completion times in a concurrent open shop. *Oper. Res. Lett.*, 38(5):390–395, September 2010. 116, 130

[Ore11] Lorenzo Orecchia. *Fast Approximation Algorithms for Graph Partitioning using Spectral and Semidefinite-Programming Techniques*. PhD thesis, EECS Department, University of California, Berkeley, May 2011. 36, 42, 44

[OSV11] Lorenzo Orecchia, Sushant Sachdeva, and Nisheeth K. Vishnoi. Approximating the exponential, the lanczos method and an $\tilde{O}(\mathrm{m})$-time spectral algorithm for balanced separator. *CoRR*, abs/1111.1491, 2011. 47, 84, 111

[OSV12] Lorenzo Orecchia, Sushant Sachdeva, and Nisheeth K. Vishnoi. Approximating the exponential, the lanczos method and an $\tilde{O}(m)$-time spectral algorithm for balanced separator. In *Proceedings of the 44th symposium on Theory of Computing*, STOC '12, pages 1141–1160, 2012. 19, 47, 84, 98, 111

[OSVV08] Lorenzo Orecchia, Leonard J. Schulman, Umesh V. Vazirani, and Nisheeth K. Vishnoi. On partitioning graphs via single commodity flows. In *STOC '08: Proc. 40th Ann. ACM Symp. Theory of Computing*, pages 461–470, 2008. 11

[OV11] Lorenzo Orecchia and Nisheeth K. Vishnoi. Towards an sdp-based approach to spectral methods: A nearly-linear-time algorithm for graph partitioning and decomposition. In *SODA'11: Proc. 22nd Ann. ACM-SIAM Symp. Discrete Algorithms*, pages 532–545, 2011. 12, 13, 29, 31, 32, 41, 42, 44

[Par99] Emanuel Parzen. *Stochastic Processes*. Society for Industrial and Applied Mathematics, 1999. 25

[PC99] Victor Y. Pan and Zhao Q. Chen. The complexity of the matrix eigenproblem. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, STOC '99, pages 507–516, 1999. 53, 54, 58, 77

[PSS+95] C. N. Potts, S. V. Sevast'janov, V. A. Strusevich, L. N. Van Wassenhove, and C. M. Zwaneveld. The two-stage assembly scheduling problem: Complexity and approximation. *Operations Research*, 43(2):346–355, March/April 1995. 117

[Räc08] Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proceedings of the 40th annual ACM symposium on Theory of computing*, STOC '08, pages 255–264, 2008. 12

[Rag08] Prasad Raghavendra. Optimal algorithms and inapproximability results for every CSP? In *Proceedings of the 40th annual ACM symposium on Theory of computing*, STOC '08, pages 245–254, New York, NY, USA, 2008. ACM. 113, 148

[Raz98] Ran Raz. A parallel repetition theorem. *SIAM Journal of Computing*, 27(3):763–803, 1998. 4, 112, 155

[RS09] Prasad Raghavendra and David Steurer. Integrality gaps for strong sdp relaxations of unique games. In *Proceedings of the 2009 50th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '09, pages 575–585, 2009. 113

[Rus82] L. Russo. An approximate zero-one law. *Z. Wahrsch. Verw. Gebeite*, 61(1):129–139, 1982. 128, 132

[Saa92] Y. Saad. Analysis of some krylov subspace approximations to the matrix exponential operator. *SIAM J. Numer. Anal.*, 29:209–228, February 1992. 50, 54, 55, 56, 58

[Saf73] E.B Saff. On the degree of best rational approximation to the exponential function. *Journal of Approximation Theory*, 9(2):97 – 101, 1973. 101

[She94] Jonathan Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. 1994. `http://www.cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf`. 65

[She09] Jonah Sherman. Breaking the multicommodity flow barrier for $O(\sqrt{\log n})$-approximations to Sparsest Cut. In *FOCS'09: Proc. 50th Ann. IEEE Symp. Foundations of Computer Science*, 2009. 11

[She13] Jonah Sherman. Nearly maximum flows in nearly linear time. *CoRR*, abs/1304.2077, 2013. 11

[Shm97] D. B. Shmoys. *Approximation Algorithms for NP-hard Problems*, chapter Approximation algorithms for Cut problems and their application to divide-and-conquer, pages 192–235. PWS, 1997. 9

[SKK$^+$00] Kirk Schloegel, George Karypis, Vipin Kumar, J. Dongarra, I. Foster, G. Fox, K. Kennedy, A. White, and Morgan Kaufmann. Graph partitioning for high performance scientific simulations, 2000. 9

[SS11] Sushant Sachdeva and Rishi Saket. Nearly optimal np-hardness of vertex cover on k-uniform k-partite hypergraphs. In *Proceedings of*, APPROX'11/RANDOM'11, pages 327–338, Berlin, Heidelberg, 2011. Springer-Verlag. 129, 168

[SS13] Sushant Sachdeva and Rishi Saket. Optimal inapproximability for scheduling problems via structural hardness for hypergraph vertex cover. *Proceedings of the Annual IEEE Conference on Computational Complexity*, 2013. 145

[SSV75] E. B. Saff, A. Schönhage, and R. S. Varga. Geometric convergence to e^{-z} by rational functions with real poles. *Numerische Mathematik*, 25:307–322, 1975. 13, 14, 19, 59, 98, 102

[ST04] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, STOC '04, pages 81–90, 2004. 9, 12, 14, 16, 63, 85

[ST06] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *CoRR*, abs/cs/0607105, 2006. 9, 63

[ST08] Daniel A. Spielman and Shang-Hua Teng. A local clustering algorithm for massive graphs and its application to nearly-linear time graph partitioning. *CoRR*, abs/0809.3232, 2008. 11, 12

[ST11] S. Sachdeva and M. Tulsiani. Cuts in Cartesian products of graphs. *CoRR*, abs/1105.3383, 2011. 132

[SV13] S. Sachdeva and N. K. Vishnoi. Matrix Inversion Is As Easy As Exponentiation. *ArXiv e-prints*, May 2013. 96

[SW99] P. Schuurman and G. J. Woeginger. Polynomial time approximation algorithms for machine scheduling: Ten open problems. *Journal of Scheduling 2*, pages 203–213, 1999. 117

[Tao] Terence Tao. The Euler-Maclaurin formula, Bernoulli numbers, the zeta function, and real-variable analytic continuation. Available from *What's New*. 92

[Tre01] Luca Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, STOC '01, pages 453–461, New York, NY, USA, 2001. ACM. 115

[Tre05] Luca Trevisan. Approximation algorithms for unique games. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '05, pages 197–205, 2005. 113

[UW73] C. Underhill and A. Wragg. Convergence properties of Padé approximants to $\exp(z)$ and their derivatives. *IMA Journal of Applied Mathematics*, 11(3):361–367, 1973. 13

[Vis12] Nisheeth K. Vishnoi. $Lx = b$. Foundations and Trends in Theoretical Computer Science, 2012. 14, 16, 86