Diversity-Inducing Probability Measures for Machine Learning

by

Chengtao Li

B.S., Tsinghua University (2014)
M.S., Massachusetts Institute of Technology (2016)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2019

Author .... Signature redacted
Department of Electrical Engineering and Computer Science
January 31, 2019

Certified by ... Signature redacted
Suvrit Sra
Assistant Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Certified by . Signature redacted
Stefanie Jegelka
Assistant Professor of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by ... Signature redacted
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Diversity-Inducing Probability Measures for Machine Learning

by

Chengtao Li

Submitted to the Department of Electrical Engineering and Computer Science
on January 31, 2019, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Subset selection problems arise in machine learning within kernel approximation, experimental design, and numerous other applications. In such applications, one often seeks to select diverse subsets of items to represent the population. One way to select such diverse subsets is to sample according to Diversity-Inducing Probability Measures (DIPMs) that assign higher probabilities to more diverse subsets. DIPMs underlie several recent breakthroughs in mathematics and theoretical computer science, but their power has not yet been explored for machine learning. In this thesis, we investigate DIPMs, their mathematical properties, sampling algorithms, and applications.

Perhaps the best known instance of a DIPM is a Determinantal Point Process (DPP). DPPs originally arose in quantum physics, and are known to have deep relations to linear algebra, combinatorics, and geometry. We explore applications of DPPs to kernel matrix approximation and kernel ridge regression. In these applications, DPPs deliver strong approximation guarantees and obtain superior performance compared to existing methods. We further develop an MCMC sampling algorithm accelerated by Gauss-type quadratures for DPPs. The algorithm runs several orders of magnitude faster than the existing ones.

DPPs lie in a larger class of DIPMs called Strongly Rayleigh (SR) Measures. Instances of SR measures display a strong negative dependence property known as negative association, and as such can be used to model subset diversity. We study mathematical properties of SR measures, and construct the first provably fast-mixing Markov chain that samples from general SR measures. As a special case, we consider an SR measure called Dual Volume Sampling (DVS), for which we present the first poly-time sampling algorithm.

While all considered distributions over subsets are unconstrained, those of interest in the real world usually come with constraints due to prior knowledge, resource limitations or personal preferences. Hence we investigate sampling from *constrained* versions of DIPMs. Specifically, we consider DIPMs with cardinality constraints and matroid base constraints and construct poly-time approximate sampling algorithms for them. Such sampling algorithms will enable practical uses of constrained DIPMs in real world.

Thesis Supervisor: Suvrit Sra
Title: Assistant Professor of Electrical Engineering and Computer Science

Thesis Supervisor: Stefanie Jegelka
Title: Assistant Professor of Electrical Engineering and Computer Science

Acknowledgments

First and foremost, I am forever thankful to my advisors, Suvrit Sra and Stefanie Jegelka, for their never-ending guidance and support throughout my PhD. They have always been a strong support for me and gave me the freedom to pursue anything I want. Only with such free environment can I possibly produce any good work. This thesis would not have been possible without them – they were the best advisors that I could have ever asked for.

I am also thankful to my fantastic thesis committee member, Devavrat Shah. He is extremely nice and patient, and has given me very valuable opinions on my work. I am very fortunate to have him on my thesis committee.

Many thanks must also go to my wonderful collaborators, friends and people in my group. I have benefited tremendously from conversations with Yonatan Belinkov, Benson Chen, Hongge Chen, Connor Coley, Wengong Jin, Nate Kushman, Tao Lei, Ruizhi Liao, Ge Liu, Tianren Liu, Hongzhou Lin, Zelda Mariet, David Alvarez Melis, Evan Pu, Tianxiao Shen, Matt Staib, Zi Wang, Jiajun Wu, Keyulu Xu, Zhi Xu, Guowei Zhang, Hongyi Zhang, Jingzhao Zhang, Yu Zhang, Yuan Zhang and Xijia Zheng. I will never forget all those discussions (on any topics) with you at Stata Center, and will always miss those activities we have attended together. I also want to thank staffs at MIT for their help with all administrative things.

A big thank you to all members at MIT-CHIEF (MIT-China Innovation and Entrepreneurship Forum), with whom I have spent the last three wonderful years. When I joined MIT-CHIEF three years ago, I knew nothing about entrepreneurship. During these three years I have met so many fantastic people and have learned so much from them. I will never forget the days and nights when we worked together.

There are several people that I want to show my special thank for:

- Jingkai Chen, for being such a supportive friend. He always says how much he has learnt from me, but what I have learnt from him is much more. He sets a very high standard for himself. Encouraged by this, I have been trying to do the same.

- Ning Jiang, for staying with me during the first two years of my PhD, and continue to be my friend afterwards. The beginning of my PhD was a hard time for me, and I

could not sustain without her support. She has became the one who I turn to when I do not know what to do. She could point me to the correct direction with a few words.

- Qingkai Liang, for being my best friend throughout my PhD. I cannot recall how many times he had helped me out. I can still remember those days and nights when we chatted and got drunk together. I became a ski lover and wine lover because of him. We even went to wine-tasting courses together and both got level-2 certificate of WSET. My PhD life would be much less awesome without him.

- Chu Ma, for countless meals and table tennis together. She is an excellent example of what a solid female researcher looks like. She is one of the very few people who I know could continue exercising almost every single day. I admire her and am encouraged by her.

- Fangchang Ma, for being such a critical friend on mouth but such a helpful friend at heart. He taught me how to view things critically and allow me to see a new form of humor. I also learnt a lot about leadership from him.

- Fanyu Que, for showing up in my life and staying close with me during the last year of my PhD. She has been bearing with my impatience and naiveness with her incredible patience and tolerance. I could act as a boy in front of her. There is countless number of times when she raised me up. I could not be luckier to have her with me.

- Dajiang Suo, for showing me how a mature man should act. I have benefitted tremendously from learning the way he talks. I feel lucky to be one of his friends.

- Yunming Zhang, for being my best roommate for three and a half years, providing me with so much help. I have to say I am not a perfect roommate, but he is always nice and patient to me. I really appreciate it.

Finally, I would like to thank my parents, Songyue and Jun, for their supports and patience to me through the entire PhD process. They always gave their strongest support whenever I do any decisions. I dedicate this thesis to my wonderful family.

Bibliographic Notes

The main ideas in this thesis have been published in peer-reviewed conferences and one preprint. The list of papers is as follows:

- **Fast DPP Sampling for Nyström with Application to Kernel Methods.** *International Conference on Machine Learning (ICML), 2016.*
- **Gaussian Quadrature for Matrix Inverse Forms with Applications.** *International Conference on Machine Learning (ICML), 2016.*
- **Fast Mixing Markov Chains for Strongly Rayleigh Measures, DPPs, and Constrained Sampling.** *Advances in Neural Information Processing Systems (NIPS), 2016*
- **Polynomial Time Algorithms for Dual Volume Sampling.** *Advances in Neural Information Processing Systems (NIPS), 2017*
- **Fast Mixing Markov Chains for Strongly Rayleigh Measures and Variants.** *In Preparation, 2018*

The code for the work presented in this thesis is publicly available at `https://github.com/ChengtaoLi`.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Subset selection problems lie at the heart of many applications where a small subset of items must be selected to represent a larger population. A typical example is kernel matrix approximation [49, 56]. Kernel methods are widely used in machine learning and they need to manipulate kernel matrices with operations like matrix inversion or matrix multiplication. However, the square/cubic time complexity of these operations will be prohibitive when the kernel matrix is large. One solution to large kernel matrix operation is to construct a low-rank approximation of the kernel matrix. Such approximation is done by first selecting a few rows and columns from the kernel matrix. These rows and columns are then multiplied together in certain ways to construct the approximated matrix (See Figure 1-1). Subset selection problems also appear in video summarization [85, 161]. A long video may take tens of hours to watch. To get the main idea of the video quickly, we could create a "trailer" of the video by selecting a subset of scenes and watch the trailer. A similar example is text summarization [119] where we have a large pile of papers to read. Instead of reading the papers page by page, we select a subset of representative paragraphs in the paper and form a summary of all the papers. By reading the summary, we are able understand main ideas in papers within a short time (See Figure 1-1). Another example of subset selection is sensor placement [88, 103]. Here, the task is to monitor a specific area with a certain number of sensors. There are many possible locations to place sensors on. Due to the limited number of sensors, we seek for a subset of locations such that the area monitored by at least one sensor is maximized. (See Figure 1-1).

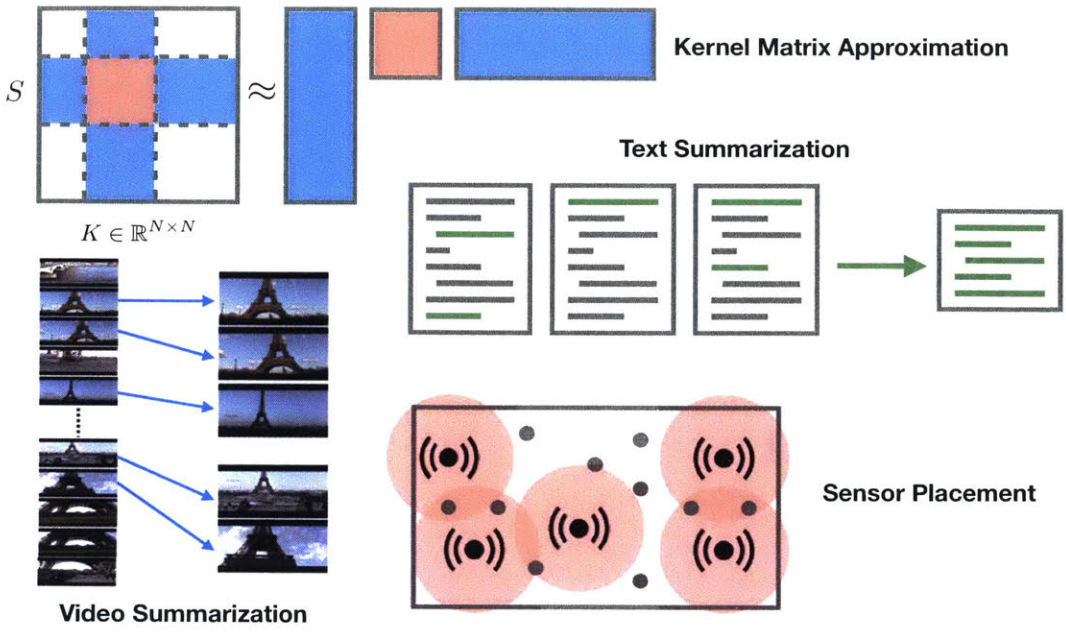


Figure 1-1: Applications examples for subset selection: kernel matrix approximation, video/text summarization and sensor placement.

Typically, the selected subsets are expected to fulfill various criteria such as sparsity or diversity. Our focus is on *diversity*, a criterion that plays a key role in a variety of applications, such as gene network subsampling [19], recommender systems [189], among many others [104, 77, 107, 2, 169, 160].

Diverse subset selection amounts to sampling from the set of all subsets of a ground set according to a measure that places more mass on subsets with qualitatively different items. We call such probability measures *Diversity-Inducing Probability Measures (DIPMs)*. A well-known example of DIPMs is called Determinantal Point Processes (DPPs). DPPs were first introduced in physics to model the repulsive phenomenon of Fermion particles [127]. Later they were referred to as *Determinantal Point Processes* [29] and were introduced to machine learning community [107]. Each DPP is associated with a kernel matrix L which quantifies the similarities between items. A DPP captures diversity by assigning subset S probability that is proportional to submatrix determinant (See Figure 1-2). Specifically, we have $\pi(S) \propto \det(L_{S,S})$.

To illustrate that DPPs capture diversity, we consider sampling a subset of points on a 2-D panel where the dissimilarity between points grows with their distance. We show in

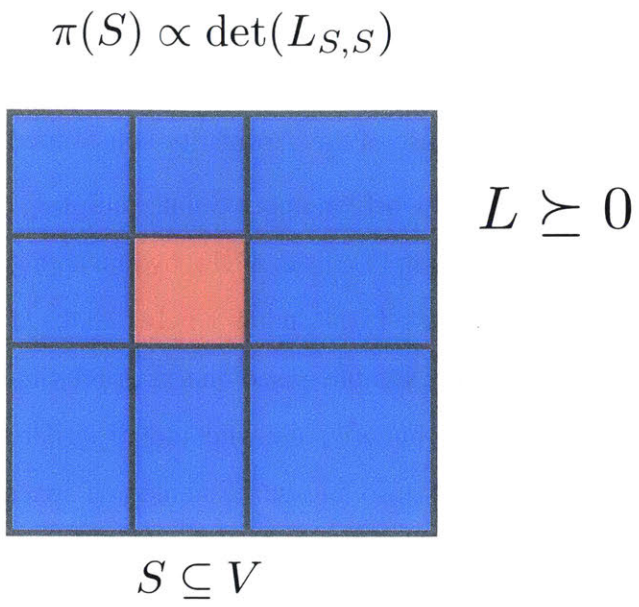


Figure 1-2: DPP definition

Figure 1-3 subsets sampled by DPP versus uniform sampling. If we do uniform sampling where no similarity information between points is considered, we end up with a subset that has clusters at random places. On the other hand, the subset sampled by DPP is spread out, and each sampled point is far from other points, i.e. the subset is diverse.

DPPs enjoy substantial interest in machine learning [104, 106, 100, 85, 132], in part due to computational tractability of basic tasks such as computing partition functions, sampling, and extracting marginals [94, 107]. Despite being polynomial-time, these tasks remain

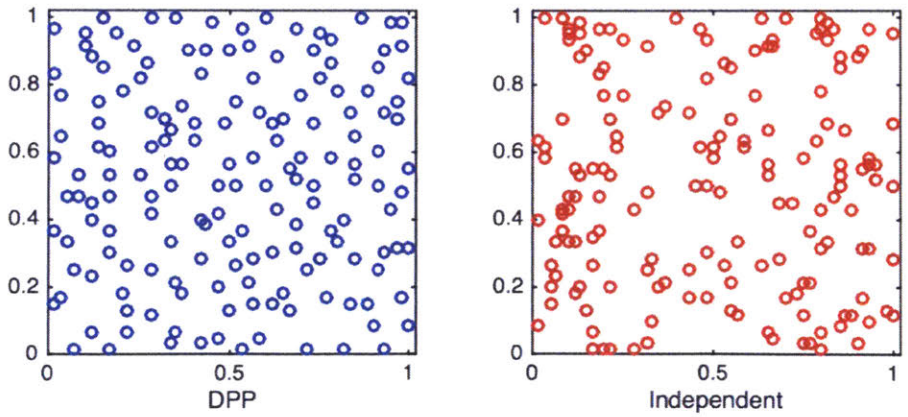


Figure 1-3: Sampled subsets from a 2D panel using DPP and Uniform distribution.

infeasible for large datasets. DPP sampling, for example, relies on an eigendecomposition of the DPP kernel, whose cubic complexity is a huge impediment to scalability. Cubic preprocessing costs also impede wider use of the cardinality-constrained variant k-DPP [105].

These drawbacks have triggered work on approximate sampling methods. Much work has been devoted to approximately sampling from a DPP by first approximating its kernel via algorithms such as the Nyström method [3], Random Kitchen Sinks [153, 1] or matrix ridge approximations [187, 181], and then sample based on this approximation. These methods lead to considerable efficiency gain, but are somewhat inappropriate for sampling because they aim to project the DPP kernel onto a lower dimensional space while minimizing a matrix norm, rather than minimizing an error measure sensitive to determinants. Another approach based on the concept of coresets is proposed to directly minimize the TV distances between the original distribution and the approximated one [112]. This method relies on the structure of the dataset: if the dataset is nicely clustered, the approximation will be both efficient and effective. Alternative approaches use a dual formulation [104], which saves time in preprocessing of the kernel matrix by transforming an eigendecomposition of a large kernel matrix to a smaller one. This is based on the assumption that a low-rank decomposition of kernel matrix is available, which may not always be true.

Another line of work focuses on sampling with Markov chain Monte Carlo [92, 76, 47]. The idea is to maintain an active subset of the ground set, and iteratively update the active set by adding items to or removing items from it. After running certain number of iterations the active subset is viewed as a subset sampled from an approximation of the target distribution. The number of updates needed to produce a good sample from MCMC is low-order polynomial [8, 117]. However, existing Markov chains require storing and updating the inverse of sub-matrices of kernel, which is potentially inefficient. Thus, since its proposal, MCMC sampling from DPP has so far not been used widely in practice.

Nevertheless, DPPs still have huge potential to be applied to subset selection problems where diverse subsets are needed. In the first part of this thesis, we explore the application of DPPs in kernel matrix approximation. We show that with diverse subset of rows and columns selected via DPP, we are able to achieve one of the best performances among existing state-of-art baselines. We further consider applying such approximation to a

downstream application, kernel ridge regression, and show empirically that using DPPs will result in superior performance.

Besides effectiveness, we pursue efficiency in sampling procedure. In the second part of this thesis, we address efficiency of (k-)DPP sampling with MCMC. We accelerate existing MCMC approaches with a retrospective-style algorithmic framework and an ancient technique called *Gauss quadrature*. Our Markov chain samples valid subsets as existing ones, but is much faster when the kernel L of the (k-)DPP is sparse. In large-scale experiments we observe over 10^3 times acceleration with our method.

While DPPs are one common example of DIPMs, there is a broader class of probability measures that is diversity-inducing called Strongly Rayleigh (SR) measures. These measures are intimately related to stable polynomials. This viewpoint is first established in [28], which has proved key to uncovering their remarkable properties, both for modeling as well as for fast sampling. SR measures exhibit *negative association*, a strong, "robust" notion of negative dependence. They have recently emerged as valuable tools in the design of algorithms [6], in the theory of polynomials and combinatorics [28], and in machine learning through DPPs. Despite being important, the mathematical properties of SR measures have largely been unexplored. Only recently in the work of [8] the authors have proved the first poly-time mixing MCMC on a certain class of SR measures called *homogeneous SR measures*, while the mixing time of MCMC for general SR measuresremains unknown.

For the third part of thesis, we study sampling methods for general SR measures and derive a provably fast mixing Markov chain that is novel and may be of independent interest. Our results provide the first polynomial guarantee for Markov chain sampling from a general DPP, and more generally from an SR distribution. This result also indicates an efficient sampling method for *Dual Volume Sampling (DVS)*, whose poly-time sampling method has remained open since 2013 [13]. Specifically, we prove that DVS lies in the class of SR, thus a poly-time MCMC sampling method follows.

While most DIPMs that we have considered have no explicit constraints, real-world applications usually come with various constraints. Take sensor placement for example. We want to have a precise control over the number of locations sampled, so that we do not end up sampling too many locations (See Figure 1-4). However, little has been known in

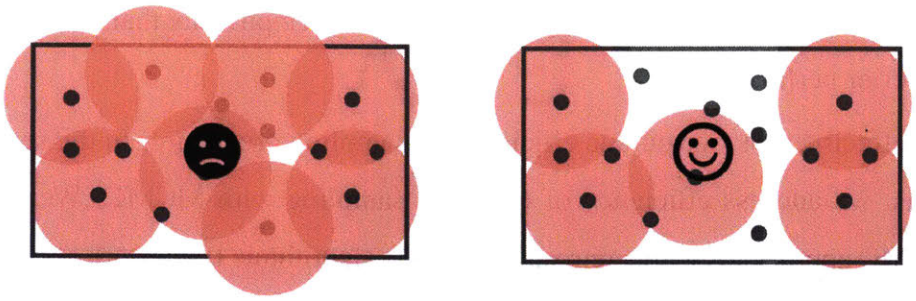


Figure 1-4: In sensor placement problem, we want to control the size of subsets we are sampling so as not to end up with sampling too many locations

efficient sampling methods of constrained DIPMs.

In the last part of this thesis, we study DIPMs with certain constraints. More specifically, we consider matroid base constraints or cardinality constraints. We present theoretical results concerning mixing times of Markov chains for constrained DIPMs, and prove that under certain conditions, all chains mix rapidly. Note that while some of the constrained distributions do lie in the family of SR measures, adding constraints may break the SR property, thus a direct application of fast MCMC for SR measures is not viable, and fast mixing chains have not been known before. We verify empirically that the dependencies of mixing times on several function- or data-related factors are consistent with our analysis.

This thesis is organized as follows: In Chapter 1 we give a brief overview of DIPMs and the problems on which we will focus throughout this thesis. In Chapter 2 we apply DPPs to core machine learning applications including kernel matrix approximation and kernel ridge regressions. We show superior performance of DPP-approximated algorithms. In Chapter 3 we introduce efficient MCMC algorithms for (k-)DPPs accelerated by Gauss-type quadrature. In Chapter 4 we focus on the broader class of SR and show the first poly-time mixing MCMC algorithm. We also revisit Dual Volume Sampling (DVS) and show that DVS lies in the class of SR. In Chapter 5 we consider constrained DIPMs and show efficient MCMC method for these measures. We close this thesis with conclusion and open problems in section 6.

Chapter 2

Determinantal Point Processes for Kernel Methods

In this chapter, we consider applying DPP to kernel methods, including Nyström method and kernel ridge regression. The Nyström method has long been popular for scaling up kernel methods. However, its theoretical guarantees and empirical performance both critically rely on the selection of suitable *landmarks*. We study landmark selection via DPPs. We prove that landmarks selected according to a DPP offer guaranteed approximation errors for Nyström. Subsequently, we analyze implications for kernel ridge regression, where we also prove the approximation guarantees. For efficient implementation, we use Markov chain DPP accelerated by Gauss quadrature to do the sampling, which will be explained in more details in the next chapter. We present empirical results that support the theoretical analysis, and demonstrate the superior performance of DPP-based landmark selection compared against existing approaches. Materials in this chapter are based on [113].

2.1 Introduction

Matrix low-rank approximation is an important ingredient of modern machine learning methods: many methods rely on operations such as multiplication and inversion of matrices. Scaling cubically in the number of data points n, these operations quickly become a bottleneck for large data. In such cases, low-rank approximations promise speedups with a

tolerable loss in accuracy.

A notable example is the *Nyström method* [144, 18], which takes a positive semidefinite matrix $L \in \mathbb{R}^{n \times n}$ as input, selects from it a small subset S of columns $L_{\cdot,S}$, and constructs the approximation $\tilde{L} = L_{\cdot,S} L_{S,S}^{\dagger} L_{S,\cdot}$. The matrix $\tilde{L}$, in its factored form, is then used in place of L. If the number $k = |S|$ of selected columns is small, then using $\tilde{L}$ decreases runtimes from $\mathcal{O}(n^3)$ to $\mathcal{O}(nk^3)$, a substantial saving.

Since its introduction to the machine learning community, the Nyström method has been applied to a wide spectrum of problems, including kernel ICA [15, 162], kernel and spectral methods in computer vision [21, 66], manifold learning [178, 177], regularization [157], and efficient approximate sampling [3]. Recent work [46, 14, 4] has shown risk bounds for Nyström applied to various kernel methods.

The most important step of the Nyström method is the selection of the column subset S, the so-called *landmarks*. This choice governs the approximation error and subsequent performance of the approximated learning methods [46]. The most basic strategy is to sample landmarks uniformly at random [184]. More sophisticated non-uniform selection strategies include deterministic greedy schemes [168], incomplete Cholesky decomposition [65, 16], sampling with probabilities proportional to diagonal values [56], to column norms [55], based on leverage scores [79], via K-means [186], and using submatrix determinants [22].

We study landmark selection using *Determinantal Point Processes* (DPP), discrete probability models that allow tractable sampling of diverse non-independent samples [126, 107]. Our work generalizes the determinantal sampling scheme of [22].[1] We refer to our scheme as DPP-Nyström, and analyze it from several perspectives.

A key quantity in our analysis is the error of Nyström approximation. Suppose c is the target rank; then for selecting $k \geq c$ landmarks, Nyström's error is typically measured using the Frobenius or spectral norm, relative to the best achievable error via rank-c SVD L_c; that is, we measure

$$\frac{\|L - L_{\cdot,S} L_{S,S}^{\dagger} L_{S,\cdot}\|_F}{\|L - L_c\|_F} \quad \text{or} \quad \frac{\|L - L_{\cdot,S} L_{S,S}^{\dagger} L_{S,\cdot}\|_2}{\|L - L_c\|_2}.$$

[1] Surprisingly, they do not make an explicit connection to DPPs

Several authors also use additive instead of relative bounds. However, such bounds are very sensitive to scaling, and become loose even if a single entry of the matrix is large. Thus, we focus on the above relative error bounds.

First, we analyze this approximation error. Previous analysis [22] assumes a cardinality of $k = c$; we go beyond this limitation and analyze the general case of selecting $k \geq c$ columns. Our relative error bounds rely on the properties of characteristic polynomials. Empirically DPP-Nyström is seen to obtain approximations superior to other state-of-art methods.

Second, we consider its impact on kernel methods. Specifically, we address the impact of Nyström-based kernel approximations on kernel ridge regression. This task has been noted as the main application in [14, 4]. We show risk bounds of DPP-Nyström that hold in expectation. Empirically, it achieves the best performance among competing methods.

Third, we consider the efficiency of DPP-Nyström; specifically, its tradeoff between error and running time. Since its proposal in [22], determinantal sampling (also realized as k-DPP) has so far not been used widely in practice due to (valid) concerns about its scalability. We use MCMC for DPP sampling and accelerate it with Gauss quadrature. Empirical results indicate that the chain yields favorable results within a small number of iterations, and the best efficiency-accuracy traedoffs compared to state-of-art methods (Figure 2-6).

2.2 Background and Notation

Let $L \in \mathbb{R}^{n \times n}$ be positive semidefinite (PSD); let it have the eigendecomposition $L = U \Lambda U^\top$ with eigenvalues $\{\lambda_i\}_{i=1}^n$ arranged decreasingly. We use $L_{i,\cdot}$ for the i-th row and $L_{\cdot,j}$ for the j-th column, and, likewise, $L_{S,\cdot}$ for the rows of L and $L_{\cdot,S}$ for the columns of L indexed by $S \subseteq [n]$. Finally, $L_{S,S}$ is the submatrix of L with rows and columns indexed by S. In this notation, $L_c = U_{\cdot,[c]} \Lambda_{[c],[c]} U_{\cdot,[c]}^\top$ is the best rank-c approximation to L in both Frobenius and spectral norm. We write $r(\cdot)$ for the rank and $(\cdot)^\dagger$ for the pseudoinverse, and denote the decomposition of L by $B^\top B$, where $B \in \mathbb{R}^{r(L) \times n}$.

The Nyström Method The *standard Nyström* method selects a subset $S \subseteq [n]$ of

$|S| = k$ *landmarks*, and approximates L with $L_{\cdot,S}L_{S,S}^{\dagger}L_{S,\cdot}$. The actual set of landmarks affects the approximation quality, and has hence been the subject of a substantial body of research [46, 168, 65, 16, 56, 55, 79, 186, 22].

Besides various landmark selection methods, there exist variations to the standard Nyström method, such as the *ensemble Nyström method* [108] that uses a weighted combination of approximations, or the *modified Nyström method* that constructs an approximation $L_{\cdot,S}L_{\cdot,S}^{\dagger}LL_{S,\cdot}^{\dagger}L_{S,\cdot}$ [175]. In this chapter, we focus on the standard Nyström method.

2.3 DPP for the Nyström Method

Next, we consider sampling landmarks $S \subseteq [n]$ from k-DPP(L), and use the approximation $\tilde{L} = L_{\cdot,S}L_{S,S}^{\dagger}L_{S,\cdot}$, referred to as DPP-Nyström. This method was introduced in [22], but without making the explicit connection to DPPs. Our analysis builds on this connection and subsumes existing results which only apply to $k = c$ (recall, c is the rank of the target approximation).

In the remainder of this section, we show following bounds:

Theorem 1 (Relative Error). *If $S \sim$ k-DPP(L), then DPP-Nyström satisfies the relative errors bounds*

$$\mathbb{E}_S\left[\frac{\|L - L_{\cdot,S}(L_{S,S})^{\dagger}L_{S,\cdot}\|_F}{\|L - L_c\|_F}\right] \leq \left(\frac{k+1}{k+1-c}\right)\sqrt{n-c}, \tag{2.3.1}$$

$$\mathbb{E}_S\left[\frac{\|L - L_{\cdot,S}(L_{S,S})^{\dagger}L_{S,\cdot}\|_2}{\|L - L_c\|_2}\right] \leq \left(\frac{k+1}{k+1-c}\right)(n-c). \tag{2.3.2}$$

Our analysis exploits a property of characteristic polynomials observed in [89]. Coefficients of characteristic polynomials are a sum of determinants:

$$e_k(L) = \sum_{|S|=k} \det(B_S^{\top}B_S) = e_k(\Lambda). \tag{2.3.3}$$

The following lemma bounds a ratio of such coefficients.

Lemma 2 ([89]). *For any $k \geq c > 0$, it holds that*

$$\frac{e_{k+1}(\Lambda)}{e_k(\Lambda)} \leq \frac{1}{k+1-c} \sum_{i>c} \lambda_i.$$

With this lemma in hand, we proceed to prove Theorem 1.

Proof (Thm. 1). We begin with the Frobenius norm error, and then show the spectral norm result. Using the decomposition $L = B^\top B$, it holds that

$$\begin{aligned}
&\mathbb{E}_S\left[\|L - L_{\cdot,S} L_{S,S}^\dagger L_{S,\cdot}\|_F\right] = \mathbb{E}_S\left[\|B^\top B - B^\top B_S (B_S^\top B_S)^\dagger B_S^\top B\|_F\right] \\
&= \mathbb{E}_S\left[\|B^\top (I - B_S (B_S^\top B_S)^\dagger B_S^\top) B\|_F\right] = \mathbb{E}_S\left[\|B^\top (I - U_S U_S^\top) B\|_F\right],
\end{aligned}$$

where $U_S \Sigma_S V_S^\top$ is the SVD of B_S. Next, we extend U_S to a full basis $U = [U_S \; U_S^\perp]$. Since U is orthonormal, we have $UU^\top = I$ and $I - U_S U_S^\top = U_S^\perp (U_S^\perp)^\top$. Plugging in this identity and applying Cauchy-Schwartz yields

$$\begin{aligned}
&\mathbb{E}_S\left[\|B^\top (I - U_S U_S^\top) B\|_F\right] = \mathbb{E}_S\left[\|B^\top U_S^\perp (U_S^\perp)^\top B\|_F\right] \\
&= \mathbb{E}_S\left[\sqrt{\textstyle\sum_{i,j} (b_i^\top U_S^\perp (U_S^\perp)^\top b_j)^2}\right] \leq \mathbb{E}_S\left[\sqrt{(\textstyle\sum_{i,j} \|b_i^\top U_S^\perp\|_2^2 \|b_j^\top U_S^\perp\|_2^2)}\right] \\
&= \mathbb{E}_S\left[\textstyle\sum_i \|b_i^\top U_S^\perp\|_2^2\right] = \frac{1}{e_k(L)} \textstyle\sum_{|S|=k} \sum_i \det(B_S^\top B_S) \|b_i^\top U_S^\perp\|_2^2 \\
&= \frac{1}{e_k(L)} \textstyle\sum_{|S|=k} \sum_{i \notin S} \det(B_{S\cup\{i\}} B_{S\cup\{i\}}^\top) \\
&= (k+1) \frac{e_{k+1}(L)}{e_k(L)}.
\end{aligned}$$

By (2.3.3) and Lemma 2 it follows that

$$\begin{aligned}
(k+1)\tfrac{e_{k+1}(L)}{e_k(L)} &\leq \tfrac{k+1}{k+1-c} \textstyle\sum_{i>c} \lambda_i \\
&\leq \tfrac{k+1}{k+1-c} \sqrt{n-c} \sqrt{\textstyle\sum_{i>c} \lambda_i^2} \\
&= \tfrac{k+1}{k+1-c} \sqrt{n-c} \|L - L_c\|_F.
\end{aligned}$$

The bound on the Frobenius norm immediately implies the bound on the spectral norm:

$$\mathbb{E}_S\left[\|L - L_{\cdot,S}(L_{S,S})^\dagger L_{S,\cdot}\|_2\right] \leq \mathbb{E}_S\left[\|L - L_{\cdot,S}L_{S,S}^\dagger L_{S,\cdot}\|_F\right]$$
$$\leq \frac{k+1}{k+1-c}\sqrt{n-c}\|L - L_c\|_F \leq \frac{k+1}{k+1-c}(n-c)\|L - L_c\|_2 \qquad \square$$

Remarks Our bounds are not directly comparable to previous bounds (e.g., [79] on uniform and leverage score sampling) that hold with certain probability since our bounds hold in expectation. However, in Sec. 2.5.1 we extensively experiment with DPP-Nyström on various datasets and observe superior accuracies against various existing state-of-art methods.

We also show the bounds that hold with high probability. To show high probability bounds we employ concentration results on homogeneous strongly Rayleigh measures. Specifically, we use the following theorem.

Theorem 3 ([150]). *Let $\mathbb{P}$ be a k-homogeneous strongly Rayleigh probability measure on $\{0,1\}^n$ and f an ℓ-Lipschitz function on $\{0,1\}^n$, then*

$$\mathbb{P}(f - \mathbb{E}[f] \geq a\ell) \leq \exp\{-a^2/8k\}.$$

It is known that a k-DPP is a homogeneous strongly Rayleigh measure on $\{0,1\}^n$ [28, 8], thus Theorem 3 applies to results obtained with k-DPP. Concretely, for the bound in Theorem 1 that holds in expectation, we have the following bound that holds with high probability:

Corollary 4. *When sampling $S \sim$ k-DPP(L), for any $\delta \in (0,1)$, with probability at least $1-\delta$ we have*

$$\frac{\|L - L_{\cdot,S}(L_{S,S})^\dagger L_{S,\cdot}\|_F}{\|L - L_c\|_F} \leq \left(\frac{k+1}{k+1-c}\right)\sqrt{n-c} + \sqrt{8k\log(1/\delta)}\sqrt{\frac{\sum_{i=1}^n \lambda_i^2}{\sum_{i=c+1}^n \lambda_i^2}},$$
$$\frac{\|L - L_{\cdot,S}(L_{S,S})^\dagger L_{S,\cdot}\|_2}{\|L - L_c\|_2} \leq \left(\frac{k+1}{k+1-c}\right)(N-c) + \sqrt{8k\log(1/\delta)}\frac{\lambda_1}{\lambda_{c+1}},$$

where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ are the eigenvalues of L.

Proof. The Lipschitz constants of the relative errors are upper bounded by $\sqrt{\frac{\sum_{i=1}^{n}\lambda_i^2}{\sum_{i=c+1}^{n}\lambda_i^2}}$ and $\frac{\lambda_1}{\lambda_{c+1}}$, respectively. Applying Theorem 3 yields the results.

□

2.4 Low-rank Kernel Ridge Regression

The theoretical (Section 2.3) and empirical (Section 2.5.1) results suggest that DPP-Nyström may be very suitable for scaling kernel learning methods. In this section, we analyze its implications on kernel ridge regression. The experiments in Section 2.5 confirm our results empirically.

Suppose we have n training samples $\{(x_i, y_i)\}_{i=1}^n$, where $y_i = z_i + \epsilon_i$ are the observed labels under zero-mean noise, with finite noise covariance. We minimize a regularized empirical loss

$$\min_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f(x_i)) + \frac{\gamma}{2}\|f\|^2$$

over an RKHS $\mathcal{F}$; equivalently, we solve the problem

$$\min_{\alpha\in\mathbb{R}^n} \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, (L\alpha)_i) + \frac{\gamma}{2}\alpha^\top L\alpha,$$

for the corresponding kernel matrix L. With the squared loss $\ell(y, f(x)) = \frac{1}{2}(y - f(x))^2$, the resulting estimator is

$$\hat{f}(x) = \sum_{i=1}^{n} \alpha_i k(x, x_i), \quad \alpha = (L + n\gamma I)^{-1} y, \tag{2.4.1}$$

and the prediction for $\{x_i\}_{i=1}^n$ is given by $\hat{z} = L(L + n\gamma I)^{-1}y \in \mathbb{R}^n$. Denoting the noise

covariance by F, we obtain the risk

$$\begin{aligned}\mathcal{R}(\hat{z}) &= \frac{1}{n}\mathbb{E}_{\varepsilon}\|\hat{z}-z\|^2 \\ &= n\gamma^2 z^\top (L+n\gamma I)^{-2} z + \tfrac{1}{n}Tr(FL^2(L+n\gamma I)^{-2}) \\ &= \text{bias}(L) + \text{var}(L). \end{aligned} \tag{2.4.2}$$

Observe that the bias term is matrix-decreasing (in L) while the variance term is matrix-increasing. Since the estimator (2.4.1) requires expensive matrix inversions, it is common to replace L in (2.4.1) by an approximation $\tilde{L}$. If $\tilde{L}$ is constructed via Nyström we have $\tilde{L} \preceq L$, and it directly follows that the variance shrinks with this substitution, while the bias increases. Denoting the predictions from $\tilde{L}$ by $\hat{z}_{\tilde{L}}$, Theorem 5 completes the picture of how using $\tilde{L}$ affects the risk.

Theorem 5. *If $\tilde{L}$ is constructed via DPP-Nyström and $\gamma \geq \frac{1}{n}Tr(L)$, then*

$$\mathbb{E}_S\left[\sqrt{\frac{\mathcal{R}(\hat{z})}{\mathcal{R}(\hat{z}_{\tilde{L}})}}\right] \geq 1 - \frac{(k+1)}{n\gamma}\frac{e_{k+1}(L)}{e_k(L)}.$$

Proof. We build upon [14]. Knowing that $\text{Var}(\tilde{L}) \leq \text{Var}(L)$ as $\tilde{L} \preceq L$, it remains to bound the bias. We write $L = B^\top B$ and $\tilde{L} = B^\top B_S(B_S^\top B_S)^\dagger B_S^\top B$, and bound the difference $L - \tilde{L}$ as

$$\begin{aligned} L - \tilde{L} &= B^\top(I - B_S(B_S^\top B_S)^\dagger B_S^\top)B = B^\top U_S^\perp (U_S^\perp)^\top B \\ &\preceq \|B^\top U_S^\perp (U_S^\perp)^\top B\|_F I = \sqrt{\sum_{i,j}(b_i^\top U_S^\perp (U_S^\perp)^\top b_j)^2} I \\ &\preceq \sqrt{(\sum_{i,j}\|b_i^\top U_S^\perp\|_2^2\|b_j^\top U_S^\perp\|_2^2)} I = \sum_i \|b_i^\top U_S^\perp\|_2^2 I = \nu_S I, \end{aligned}$$

where $\nu_S = \sum_i \|b_i^\top U_S^\perp\|_2^2 \leq \sum_i \|b_i^\top\|_2^2 = Tr(L)$. Since, by assumption, $\frac{1}{n}Tr(L) < \gamma$, we have that $\frac{\nu_S}{n\gamma} < 1$, and

$$(\tilde{L}+n\gamma I)^{-1} \preceq (L - \nu_S I + n\gamma I)^{-1} \preceq (1 - \frac{\nu_S}{n\gamma})^{-1}(L+n\gamma I)^{-1}.$$

Finally, this matrix inequality imples that

$$\sqrt{\frac{\text{bias}(L)}{\text{bias}(\tilde{L})}} \geq (1 - \frac{\nu_S}{n\gamma}).$$

Taking expectation over $S \sim k$-DPP(L) yields

$$\mathbb{E}_S\left[\sqrt{\frac{\text{bias}(L)}{\text{bias}(\tilde{L})}}\right] \geq 1 - \mathbb{E}_S\left[\frac{\nu_S}{n\gamma}\right] = 1 - \frac{(k+1)}{n\gamma}\frac{e_{k+1}(L)}{e_k(L)}.$$

Together with the fact that $\text{var}(\tilde{L}) \geq \text{var}(L)$, we obtain

$$\begin{aligned}
\mathbb{E}_S\left[\sqrt{\frac{\mathcal{R}(\hat{z})}{\mathcal{R}(\hat{z}_{\tilde{L}})}}\right] &= \mathbb{E}_S\left[\sqrt{\frac{\text{bias}(L) + \text{var}(L)}{\text{bias}(\tilde{L}) + \text{var}(\tilde{L})}}\right] \\
&\geq 1 - \frac{(k+1)}{n\gamma}\frac{e_{k+1}(L)}{e_k(L)} \qquad (2.4.3) \\
&\geq 1 - \frac{k+1}{k+1-c}\frac{\sum_{i>c}\lambda_i}{\sum_i \lambda_i} \qquad (2.4.4)
\end{aligned}$$

for any $k \geq c$, where the last inequality follows from Lemma 2 and $\gamma \geq \frac{1}{n}Tr(L)$. □

Remarks. Theorem 5 quantifies how the learning results rely on the decay of the spectrum of L. In particular, the ratio $e_{k+1}(L)/e_k(L)$ closely relates to the effective rank of L: if $\lambda_k > a$ and $\lambda_{k+1} \ll a$, this ratio becomes almost zero, resulting in near-perfect approximations and no loss in learning. This also becomes evident in (2.4.4).

Again for the bound in Theorem 5 that holds in expectation, we have the following bound that holds with high probability:

Corollary 6. *If $\tilde{L}$ is constructed via* DPP-*Nyström, then with probability at least $1 - \delta$, $\sqrt{\frac{\text{bias}(\tilde{L})}{\text{bias}(L)}}$ is upper-bounded by*

$$1 + \frac{1}{n\gamma}\left(\frac{(k+1)e_{k+1}(L)}{e_k(L)} + \sqrt{8k\log(1/\delta)}tr(L)\right).$$

Proof. Consider the function $f_S(L) = \nu_S = \sum_i \|b_i^\top (U^S)^\perp\|_2^2 \leq \sum_i \|b_i^\top\|_2^2 = \text{tr}(L)$. Since $0 \leq f_S(L) \leq \text{tr}(L)$, it follows that the Lipschitz constant for f_S is at most $\text{tr}(L)$. Thus

when $S \sim k$-DPP and $\delta \in (0,1)$, by applying Theorem 3 we see that the inequality $\nu_S \leq \mathbb{E}[\nu_S] + \sqrt{8k\log(1/\delta)}\mathrm{tr}(L)$ holds with probability at least $1-\delta$. Hence

$$\begin{aligned}\mathbb{E}_S\left[\sqrt{\frac{\mathrm{bias}(\tilde{L})}{\mathrm{bias}(L)}}\right] &\leq 1 + \mathbb{E}\left[\frac{\nu_S}{n\gamma}\right] + \sqrt{8k\log(1/\delta)}\frac{\mathrm{tr}(L)}{n\gamma} \\ &= 1 + \frac{1}{n\gamma}\left(\frac{(k+1)e_{k+1}(L)}{e_k(L)} + \sqrt{8k\log(1/\delta)}\mathrm{tr}(L)\right)\end{aligned}$$

holds with probability at least $1-\delta$. □

There exist works considering DPP methods in this scenario [14, 4]. Although our bounds are not directly comparable to existing ones, we do extensive experiments to compare DPP-Nyström against other state-of-art methods in Sec. 2.5.2 and observe superior performance of DPP-Nyström.

2.5 Experiments

In our experiments, we evaluate the performance of DPP-Nyström on both kernel approximation and kernel learning tasks, in terms of running time and accuracy.

We use 8 datasets: Abalone, Ailerons, Elevators, CompAct, CompAct(s), Bank32NH, Bank8FM and California Housing[2]. We truncated each dataset to be 4,000 samples (3,000 training and 1,000 testing). Throughout our experiments we use an RBF kernel and choose the bandwidth parameter σ and regularization parameter λ for each dataset by 10-fold cross-validation. We initialize the Gibbs sampler via Kmeans++ and run for 3,000 iterations. Results are averaged over 3 random subsets of data.

2.5.1 Kernel Approximation

We first explore DPP-Nyström (kDPP in the figures) for approximating kernel matrices. We compare to uniform sampling (`Unif`) and leverage score sampling (`Lev`) [79] as baseline landmark selection methods. We also include AdapFull (`AdapFull`) [50] that performs

[2]The data is available at `http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html`

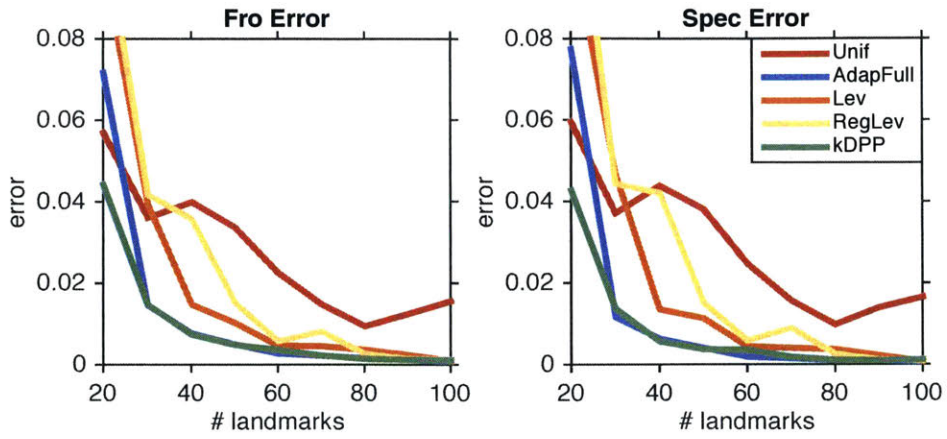


Figure 2-1: Relative Frobenius/Spectral norm errors from different kernel approximation algorithms on Ailerons dataset.

quite well in practice but scales poorly, as $\mathcal{O}(n^2)$, with the size of dataset. Although sampling with regularized leverage score (`RegLev`) [4] is not originally designed for kernel approximation, we include its results to see how regularization affects leverage score sampling.

Figure 2-1 shows example results on the Ailerons data. DPP-Nyström performs well, achieving the lowest error as measured in both spectral and Frobenius norm. The only method that is on par in terms of accuracy is `AdapFull`, which has a much higher running time.

For a different view, Figure 2-2 shows the improvement in error over `Unif`. Relative improvements are averaged over all data sets. Again, the performance of DPP-Nyström almost always dominate those of other methods, and achieves an up to 80% reduction in error.

2.5.2 Kernel Ridge Regression

Next, we apply DPP-Nyström to kernel ridge regression, comparing against uniform sampling (`Unif`) [14] and regularized leverage score sampling (`RegLev`) [4] which have theoretical guarantees for this task. Figure 2-3 illustrates an example result: non-uniform sampling greatly improves accuracy, with kDPP improving over regularized leverage scores

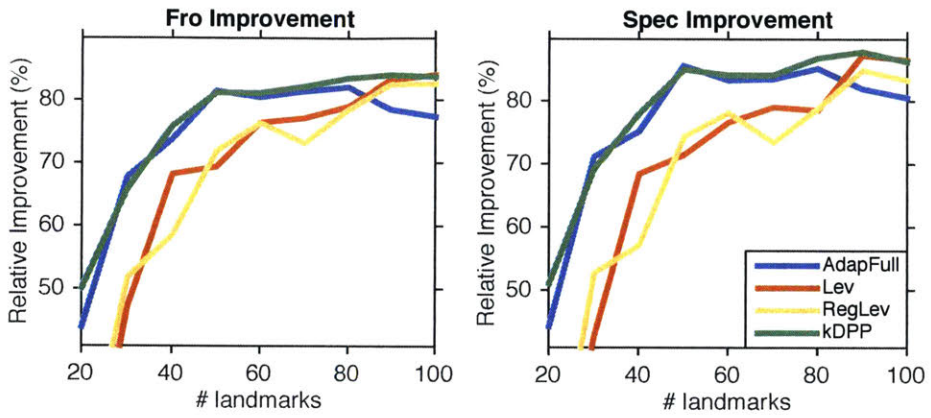


Figure 2-2: Improvement in relative Frobenius/spectral norm errors (%) over `Unif` (with corresponding landmark sizes) for kernel approximation, averaged over all datasets.

in particular for a small number of landmarks, where a single column has a larger effect.

Figure 2-4 displays the average improvement over `Unif`, averaged over 8 data sets. Again the performance of kDPP dominates those of `RegLev` and `Unif` and leads to gains in accuracy. On average kDPP consistently achieve more than 20% improvement over `Unif`.

2.5.3 Mixing of the Markov Chain DPP

In the next experiment, we empirically study the mixing of the Markov chain with respect to matrix approximation errors, the ultimate measure that is of interest in our application of the sampler. We use $k = 50$ and vary n from 500 to 4,000. To exclude impacts of the initialization, we pick the initial state S_0 uniformly at random. We run the chain for 5,000 iterations, monitoring how the error changes with the number of iterations. The example results in Figure 2-5 show that empirically, the error drops very quickly and afterwards fluctuates only little, indicating a fast convergence of the approximation error. Further results may be found in the supplementary material.

Notably, our empirical results suggest that the mixing time does not increase much even if n increases greatly, suggesting that the MCMC sampler remains fast even for large n

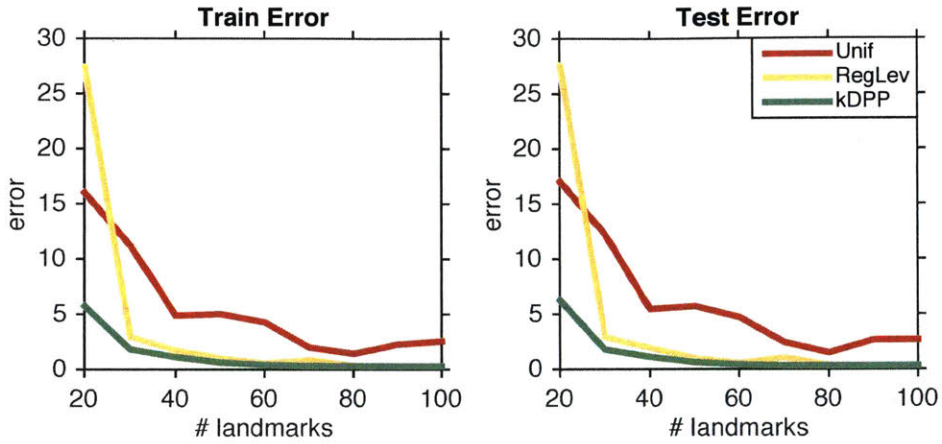


Figure 2-3: Training and testing errors by different Nyström-approximated kernel ridge regression algorithms on Ailerons dataset.

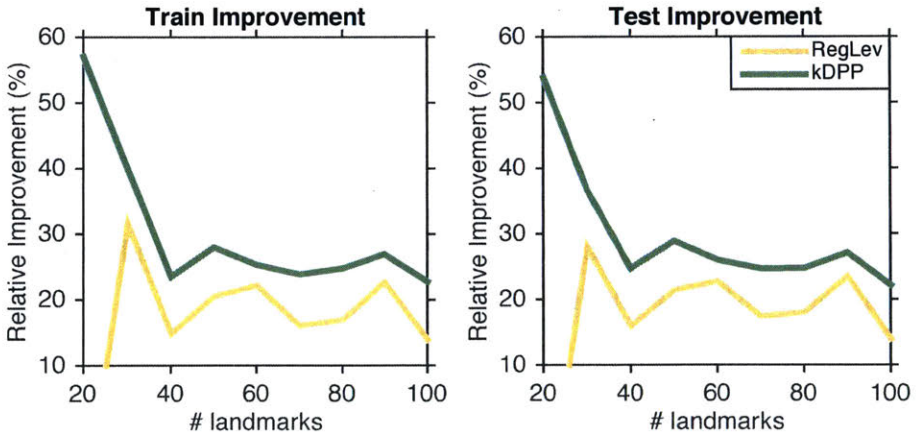


Figure 2-4: Improvements in training/testing errors (%) over uniform sampling (with corresponding landmark sizes) in kernel ridge regression, averaged over all datasets.

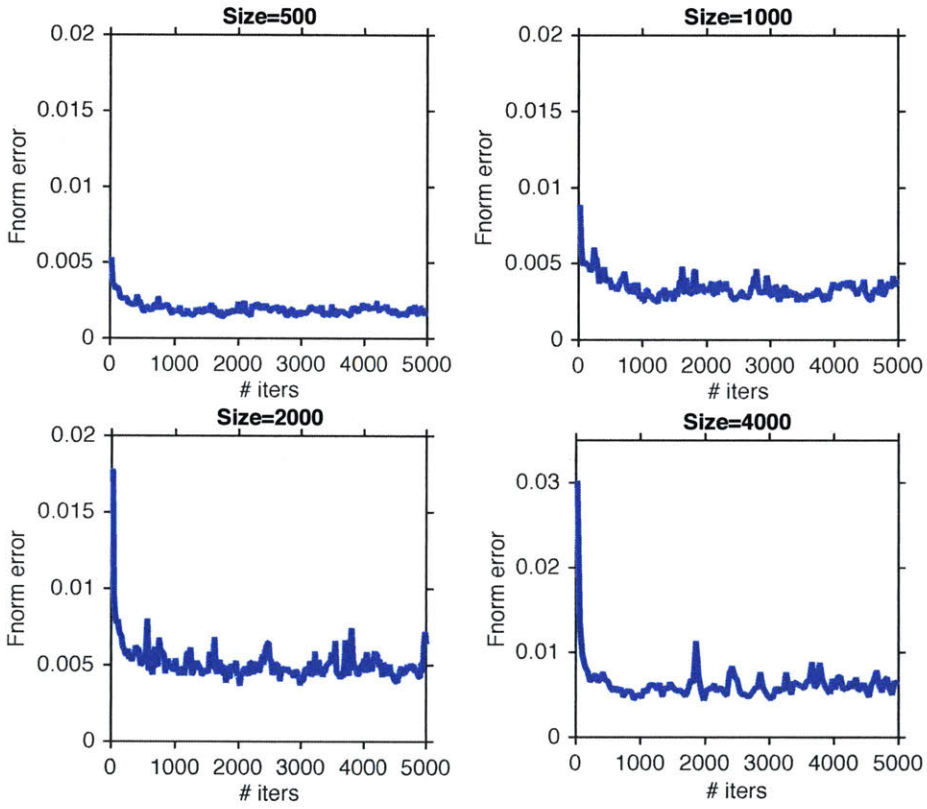
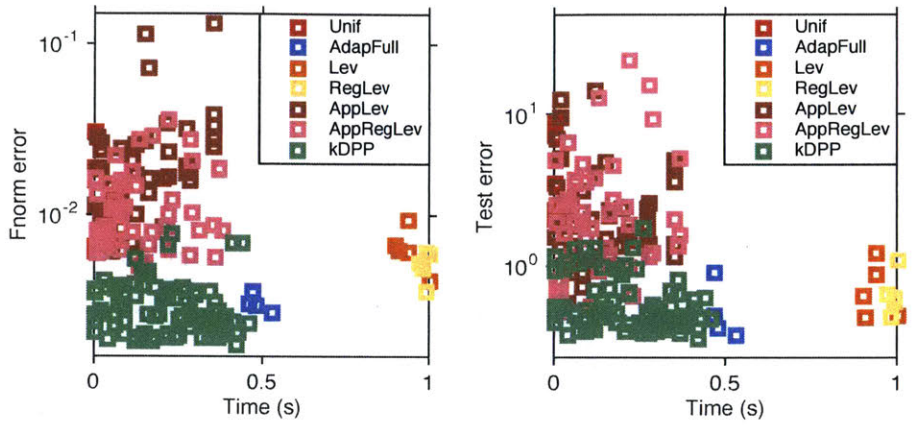


Figure 2-5: Relative Frobenius norm error of DPP-Nyström with 50 landmarks as changing across iterations of the Markov Chain.

2.5.4 Time-Error Tradeoffs



(a) Time vs. Fnorm Error

(b) Time vs. Test Error

Figure 2-6: Time-Error tradeoffs with 50 landmarks on the Ailerons data truncated at 2,000 samples (1,000 training and 1,000 testing). Errors are shown on a log scale. Bottom left is the best (low error, low running time), top right is the worst.

Iterative methods like the MCMC sampler offer tradeoffs between time and error. The longer the Markov Chain runs, the closer the sampling distribution is to the desired DPP, and the higher the accuracy obtained by Nyström. We hence explicitly show the time and accuracy for 0 to 300 iterations of the sampler.

A similar tradeoff occurs with leverage scores. For the experiments in the other sections, we computed the (regularized) leverage scores for `Lev` and `RegLev` exactly. This requires a full, computationally expensive eigendecomposition. For a fast, rougher approximation, we here compare to an approximation mentioned in [4]. Concretely, we sample p elements with probability proportional to the diagonal entries of kernel matrices L_{ii}, and then use a Nyström-like method to construct an approximate low-rank decomposition of L, and compute scores based on this approximation. We vary p from 50 to 500 to show the tradeoff for approximated leverage score sampling (`AppLev`) and regularized leverage score sampling (`AppRegLev`).

Figure 2-6 summarizes and compares the tradeoffs offered by these different methods. The x-axis indicates time, the y-axis error, so the lower left is the preferred corner. We see that exact leverage scores are accurate but expensive, whereas the approximate versions

empirically lose accuracy. `AdapFull` is accurate but needs longer time than kDPP. These results are sharpened as n grows. Overall, DPP-Nyström offers the best tradeoff of accuracy versus efficiency.

2.6 Summary

In this chapter, we explore the use of k-DPP for sampling good landmarks for the Nyström method and its further application to kernel ridge regression. We theoretically and empirically observe its competitive performance, for both matrix approximation and ridge regression, compared to state-of-the-art methods. To make this accurate method scalable to large matrices, we consider sampling via MCMC accelerated with Gauss quadrature. Our results indicate that the iterative approach, an MCMC sampler, achieves good landmark samples quickly. Our empirical results demonstrate that among state of the art methods, the iterative sampler yields the best tradeoff between efficiency and accuracy.

Chapter 3

Sampling DPP by Efficient MCMC with Gauss Quadrature

In this chapter, we focus on Markov chain Monte Carlo sampling for (k-)DPP. We show that the chain involves heavy computation of *bilinear inverse forms (BIFs)* $u^\top A^{-1}u$, where A is a positive definite matrix and u a given vector. To accelerate the chain, we present a framework for accelerating a spectrum of machine learning algorithms that rely on computation of BIFs. Our framework is built on Gauss-type quadrature and can easily scale to large, sparse matrices; it allows retrospective computation of lower and upper bounds on $u^\top A^{-1}u$, which in turn helps accelerate several algorithms. We prove that these bounds improve iteratively, compare their tightness to each other, and show their linear convergence. To our knowledge, our work is the first to demonstrate these key properties of Gauss-type quadrature, which is a classical and exceptionally well-studied topic. We illustrate empirical consequences of our results by using quadrature to accelerate Markov chain (k-)DPP and observe tremendous speedups in several instances. Materials in this chapter are based on [118].

3.1 Bilinear Inverse Forms (BIFs)

Symmetric positive definite matrices abound in machine learning, arising in various guises: covariances, kernels, graph Laplacians, or otherwise. A basic computation with such matrices is the evaluation of the bilinear form $u^T f(A)v$, where f is a matrix function and u,

v are given vectors. If $f(A) = A^{-1}$, we speak of computing the *bilinear inverse form (BIF)* $u^T A^{-1} v$. If, for instance, $u{=}v{=}e_i$ (i^{th} canonical vector), then $u^T f(A) v = (A^{-1})_{ii}$ is the i^{th} diagonal entry of the inverse.

In this chapter, we are specifically interested in computing BIFs due to their great value in several machine learning contexts, including the evaluation of a Gaussian density at a point, the Woodbury matrix inversion lemma, implementation of MCMC samplers for Determinantal Point Processes (DPP), computation of graph centrality measures, or greedy submodular maximization (see Sec. 3.1.3).

When A is large, it is preferable to compute $u^T A^{-1} v$ iteratively rather than first computing $A^{-1} v$ (using Cholesky) at a cost of $O(n^3)$ operations. One idea is to use conjugate gradients to approximately solve $Ax = v$ and to then obtain $u^T A^{-1} v = u^T x$. But several applications require precise bounds on numerical estimates to $u^\top A^{-1} v$ (e.g., in MCMC based DPP samplers such bounds help decide whether to accept or reject a transition in each iteration–see Sec. 3.6.1), so we need a more finessed approach.

Gauss quadrature is such an approach. Originally proposed in [71] for approximating integrals, Gauss- and *Gauss-type quadrature* (i.e., Gauss-Lobatto [122] and Gauss-Radau [152] quadrature) have since been applied to approximating bilinear forms including the BIF $u^T A^{-1} v$ [17]. [17] also show that Gauss and (right) Gauss-Radau quadrature yield lower bounds, while Gauss-Lobatto and (left) Gauss-Radau yield upper bounds on this bilinear inverse form.

Despite its long history and voluminous existing work (see e.g., [82]), our understanding of Gauss-type quadrature for matrix problems is far from complete. For instance, it is not known whether the bounds on BIFs iteratively improve; nor is it known how the bounds obtained from Gauss, Gauss-Radau and Gauss-Lobatto quadrature compare with each other. *We do not even know how fast the iterates of Gauss-Radau or Gauss-Lobatto quadrature converge* to the true value of $u^T A^{-1} v$.

Contributions. We address all the aforementioned problems, and make the following main contributions:

- We show that the lower and upper bounds generated by Gauss-type quadrature monotonically approach the target value (Thm. 12, Thm. 14, Corr. 15). Furthermore, we show

that for the same number of iterations Gauss-Radau quadrature yields bounds superior to those given by Gauss or Gauss-Lobatto (Thm. 12, Thm. 14), but somewhat surprisingly they share the same convergence rate.

- We derive linear convergence rates for Gauss-Radau and Gauss-Lobatto explicitly (Thm. 13, Thm. 16, Corr. 17).
- We demonstrate the implications of our results for scalable Markov chain sampling from a DPP. In this case, quadrature accelerates computations, and the bounds aid early stopping. Notably, on large-scale sparse problems our methods lead to even several orders of magnitude in speedups.

3.1.1 Determinantal Point Processes (DPPs)

A Determinantal Point Process DPP(L) is a distribution over all subsets of a ground set $\mathcal{V}$ of cardinality n. It is determined by a positive semidefinite kernel $L \in \mathbb{R}^{n \times n}$. Let $L_{S,S}$ be the submatrix of L consisting of the entries L_{ij} with $i, j \in S \subseteq \mathcal{V}$. Then, the probability $\pi(S)$ of observing $S \subseteq \mathcal{V}$ given by

$$\pi(S) = \det(L_{S,S}) / \det(L + I) \tag{3.1.1}$$

Conditioning on sampling sets of fixed cardinality k, one obtains a k-DPP [105]:

$$\pi(S \mid |S| = k) = \det(L_{S,S}) e_k(L)^{-1} [\![|S| = k]\!],$$

where $e_k(L)$ is the k-th coefficient of the characteristic polynomial

$$\det(\lambda I - L) = \sum_{k=0}^{n} (-1)^k e_k(L) \lambda^{n-k}.$$

DPPs arise in random matrix theory, combinatorics, machine learning, matrix approximations, and many other areas; see e.g., [126, 123, 124, 41, 29, 170, 107, 94, 31, 30, 118].

One may expect an exponential-time sampling algorithm for DPP since the size of the support for DPP is 2^n, but it has been shown [94] that one could sample from DPP in

polynomial time where the sampling procedure relies on an eigendecomposition of L. The resulting cubic time complexity, however, is still a huge impediment for applications of DPP on real applications.

Such drawback has triggered work on approximate sampling methods. Much work has been devoted to approximately sample from a DPP by first approximating L via algorithms such as the Nyström method [3], Random Kitchen Sinks [1, 153], matrix ridge approximations [181, 187], or block-constant matrix approximation [112], and then sampling based on this approximation. Anther line of work focuses on MCMC [113, 8, 117], which offers a potentially attractive avenue different from the above approaches that all rely on approximation techniques.

While our main focus is discrete version of DPP by default, there is also a line of work considering sampling from continuous DPP, either via approximation to kernel function [90] or via MCMC [74]. Readers could refer to the original paper and references therein. Further discussion of this line of work is beyond the scope of this paper.

3.1.2 MCMC for (k-)DPP

First we show MCMC for general DPP π, i.e., we run a Markov Chain with state space being $S \subseteq \mathcal{V}$. All our chains are ergodic. Previous work used a simple add-delete Metropolis-Hasting chain [100]. Starting with an arbitrary set $S \subseteq \mathcal{V}$, we sample a point $u \in \mathcal{V}$ uniformly at random. If $u \in S$, we remove t with probability $\min\{1, \pi(S \setminus \{u\})/\pi(S)\}$; if $t \notin S$, we add it to S with probability $\min\{1, \pi(S \cup \{u\})/\pi(S)\}$. Algorithm 1 shows the (lazy) Markov chain.

While the aforementioned chain deals with state space of subsets with any sizes, we use another chain to deal with state space of subsets with fixed size. Such chain proves useful when sampling from k-DPP. Such chain takes swapping steps: given a current set $S \subseteq \mathcal{V}$, it picks, uniformly at random, points $v \in S$ and $u \notin S$, and swaps them with probability $\min\{1, \pi(S \cup \{v\} \setminus \{u\})/\pi(S)\}$. Algorithm 2 formalizes this procedure.

Both aforementioned Markov chains converge to the target distributions, which could be easily verified by detailed balance. When sampling from (k-)DPP, if n is large and only a

Algorithm 1 Add/delete Chain

Require: DPP distribution π
  Initialize $S \subseteq \mathcal{V}$
  while not mixed **do**
    Let $b = 1$ with probability $\frac{1}{2}$
    if $b = 1$ **then**
      Pick $u \in \mathcal{V}$ uniformly at random
      if $u \in S$ **then**
        $S = S \backslash \{u\}$ with probability $\min\{1, \pi(S \setminus \{u\})/\pi(S)\}$
      else
        $S = S \cup \{u\}$ with probability $\min\{1, \pi(S \cup \{u\})/\pi(S)\}$
      end if
    else
      Do nothing
    end if
  end while

Algorithm 2 Exchange Chain

Require: k-DPP distribution π
  Initialize $S \subseteq \mathcal{V}$, $|S| = k$ ($i.e.$ $\pi(S) > 0$)
  while not mixed **do**
    Let $b = 1$ with probability $\frac{1}{2}$
    if $b = 1$ **then**
      Pick $v \in S$ and $u \notin S$ uniformly randomly
      $S = S \cup \{u\} \backslash \{v\}$ with probability $\min\{1, \pi(S \cup \{u\} \setminus \{v\})/\pi(S)\}$
    else
      Do nothing
    end if
  end while

few samples are needed, aforementioned MCMC are preferred and state-of-the-art. Therein the core task is to compute transition probabilities – an expression involving BIFs – which are compared with a random scalar threshold. Specifically, For the chain in Algorithm 1, the transition probabilities from a current subset (state) S to S' are

$$\min\{1, \pi(S')/\pi(S)\} = \min\{1, L_{u,u} - L_{u,S}L_{S,S}^{-1}L_{S,u}\}$$

for $S' = S \cup \{u\}$; and

$$\min\{1, \pi(S')/\pi(S)\} = \min\{1, L_{u,u} - L_{u,S'}L_{S',S'}^{-1}L_{S',u}\}$$

for $S' = S\backslash\{u\}$. In a k-DPP, the moves are swaps with transition probabilities

$$\min\{1, \pi(S' \cup \{u\})/\pi(S' \cup \{v\})\} = \min\left\{1, \frac{L_{u,u} - L_{u,S'}L_{S',S'}^{-1}L_{S',u}}{L_{v,v} - L_{v,S'}L_{S',S'}^{-1}L_{S',v}}\right\}$$

for replacing $v \in S$ by $u \notin S$ (and $S' = S\backslash\{v\}$). We illustrate this application in greater detail in Sec. 3.6.1.

3.1.3 Other Motivating Applications

BIFs play a central role in many problems including DPP sampling. Below we recount several notable examples: in all cases, efficient computation of BIFs is key to making the algorithms practical.

Submodular optimization, Sensing. Algorithms for maximizing submodular functions can equally benefit from fast BIF bounds. Given a positive definite matrix $K \in \mathbb{R}^{n\times n}$, the set function $F(S) = \log\det(K_S)$ is *submodular*: for all $S \subseteq T \subseteq [n]$ and $i \in [n] \setminus T$, it holds that $F(S \cup \{i\}) - F(S) \geq F(T \cup \{i\}) - F(T)$.

A key task in applications such as sensing [103], MAP inference for DPPs [78], and matrix column subset selection [34, 176], is to find the set $S^* \subseteq [n]$ maximizing $F(S)$. In sensor placement, this maximization problem arises when modeling spatial phenomena (temperature, pollution) via Gaussian Processes, and selecting locations to maximize the

joint entropy $F_1(S) = H(X_S) = \log\det(K_S) + \text{const}$ or mutual information $F_2(S) = I(X_S; X_{[n]\setminus S})$. [103] use a sparse covariance kernel K for the GP.

Greedy algorithms for maximizing monotone [141] or non-monotone [40] submodular functions rely on marginal gains of the form

$$\begin{aligned}
F_1(S \cup \{i\}) - F_1(S) &= \log(K_i - K_{iS}K_S^{-1}K_{Si}); \\
F_1(T \setminus \{i\}) - F_1(T) &= -\log(K_i - K_{iU}K_U^{-1}K_{Ui}); \\
F_2(S \cup \{i\}) - F_2(S) &= \log \tfrac{K_i - K_{iS}K_S^{-1}K_{Si}}{K_i - K_{i\bar{S}}K_{\bar{S}}^{-1}K_{\bar{S}i}}
\end{aligned}$$

for $U = T\setminus\{i\}$ and $\bar{S} = [n]\setminus S$. The algorithms compare gains to a random threshold, or find an item with the largest gain. In both cases, fast BIF bounds offer speedups. They can be combined with lazy [138] and stochastic greedy algorithms [139].

Network Analysis, Centrality. When analyzing relationships and information flows between connected entities in a network, such as people, organizations, computers, smart hardwares, etc. [159, 111, 12, 64, 61, 24], an important question is to measure popularity, centrality, or importance of a node.

Several existing popularity measures can be expressed as the solution to a large-scale linear system. For example, *PageRank* [145] is the solution to $(I - (1-\alpha)A^\top)x = \alpha\mathbf{1}/n$, and *Bonacich centrality* [26] is the solution to $(I - \alpha A)x = \mathbf{1}$, where A is the adjacency matrix. When computing local estimates, i.e., only a few entries of x, we obtain exactly the task of computing BIFs [183, 110]. Moreover, we may only need local estimates to an accuracy sufficient for determining which entry is larger, a setting where our quadrature based bounds on BIFs will be useful.

Scientific Computing. In computational physics BIFs are used for estimating selected entries of the inverse of a large sparse matrix. More generally, BIFs can help in estimating the trace of the inverse, a computational substep in lattice Quantum Chromodynamics [54, 69], some signal processing tasks [83], and in Gaussian Process (GP) Regression [155] e.g., for estimating variances. In numerical linear algebra, BIFs are used in rational approximations [165], evaluation of Green's function [67], and selective inversion of sparse matrices [120, 121, 110]. A notable use is the design of preconditioners [23] and uncertainty

quantification [20].

Benefiting from fast iterative bounds. Many of the above examples use the BIFs to rank values, identify the largest value, or compare their values to a scalar or between one another. In such cases, we first compute fast, crude lower and upper bounds on the BIF, and refine iteratively, just as far as needed to determine the comparison. Fig. 3-1 in Sec. 3.3.4 illustrates the evolution of those bounds, and Sec. 3.6 explains details.

3.2 Background on Gauss Quadrature

For convenience, we begin by recalling key aspects of Gauss quadrature,[1] as applied to computing $u^\top f(A)v$, for an $n \times n$ symmetric positive definite matrix A that has *simple* eigenvalues, arbitrary vectors u, v, and a matrix function f.

We note that it suffices to consider $u^\top f(A)u$ thanks to the *polarization* identity

$$u^\top f(A)v = \tfrac{1}{4}(u+v)^\top f(A)(u+v) - \tfrac{1}{4}(u-v)^\top f(A)(u-v).$$

Let $A = Q^\top \Lambda Q$ be the eigendecomposition of A where Q is orthonormal. Letting $\tilde{u} = Qu$, we then have

$$u^\top f(A)u = \tilde{u}^\top f(\Lambda)\tilde{u} = \sum\nolimits_{i=1}^{n} f(\lambda_i)\tilde{u}_i^2.$$

Toward computing $u^T f(A)u$, a key conceptual step is to write the above sum as the Riemann-Stieltjes integral

$$I[f] := u^\top f(A)u = \int_{\lambda_{\min}}^{\lambda_{\max}} f(\lambda)d\alpha(\lambda), \tag{3.2.1}$$

where $\lambda_{\min} \in (0, \lambda_1)$, $\lambda_{\max} > \lambda_n$, and $\alpha(\lambda)$ is piecewise constant measure defined by

$$\alpha(\lambda) := \begin{cases} 0, & \lambda < \lambda_1, \\ \sum_{j=1}^{k} \tilde{u}_j^2, & \lambda_k \leq \lambda < \lambda_{k+1}, \quad k < n, \\ \sum_{j=1}^{n} \tilde{u}_j^2, & \lambda_n \leq \lambda. \end{cases}$$

Our task now reduces to approximating the integral (3.2.1), for which we invoke the powerful

[1] The summary in this section is derived from various sources: [72, 17, 82]. Experts can skim this section for collecting our notation before moving onto Sec. 3.3, which contains our new results.

idea of Gauss-type quadratures [71, 152, 122, 72]. We rewrite the integral (3.2.1) as

$$I[f] := Q_n + R_n = \sum_{i=1}^{n} \omega_i f(\theta_i) + \sum_{i=1}^{m} \nu_i f(\tau_i) + R_n[f], \tag{3.2.2}$$

where Q_n denotes the nth degree approximation and R_n denotes a remainder term. The weights $\{\omega_i\}_{i=1}^n$, $\{\nu_i\}_{i=1}^m$ and nodes $\{\theta_i\}_{i=1}^n$ are chosen such that for all polynomials of degree less than $2n+m-1$, denoted $f \in \mathbb{P}^{2n+m-1}$, we have *exact* interpolation $I[f] = Q_n$. One way to compute weights and nodes is to set $f(x) = x^i$ for $i \leq 2n+m-1$ and then use this exact nonlinear system. But there is an easier way to obtain weights and nodes, namely by using polynomials orthogonal with respect to the measure α. Specifically, we construct a sequence of *orthogonal polynomials* $p_0(\lambda), p_1(\lambda), \ldots$ such that $p_i(\lambda)$ is a polynomial in λ of degree exactly k, and p_i, p_j are orthogonal, i.e., they satisfy

$$\int_{\lambda_{\min}}^{\lambda_{\max}} p_i(\lambda) p_j(\lambda) d\alpha(\lambda) = \begin{cases} 1, & i = j \\ 0, & \text{otherwise.} \end{cases}$$

The roots of p_n are distinct, real and lie in the interval of $[\lambda_{\min}, \lambda_{\max}]$, and form the nodes $\{\theta_i\}_{i=1}^n$ for Gauss quadrature (see, e.g., [82, Ch. 6]).

Consider the two *monic polynomials* whose roots serve as quadrature nodes:

$$\pi_n(\lambda) = \prod_{i=1}^{n} (\lambda - \theta_i), \quad \rho_m(\lambda) = \prod_{i=1}^{m} (\lambda - \tau_i),$$

where $\rho_0 = 1$ for consistency. We further denote $\rho_m^+ = \pm\rho_m$, where the sign is taken to ensure $\rho_m^+ \geq 0$ on $[\lambda_{\min}, \lambda_{\max}]$. Then, for $m > 0$, we calculate the quadrature weights as

$$\omega_i = I\left[\frac{\rho_m^+(\lambda)\pi_n(\lambda)}{\rho_m^+(\theta_i)\pi_n'(\theta_i)(\lambda - \theta_i)}\right], \quad \nu_j = I\left[\frac{\rho_m^+(\lambda)\pi_n(\lambda)}{(\rho_m^+)'(\tau_j)\pi_n(\tau_j)(\lambda - \tau_j)}\right],$$

where $f'(\lambda)$ denotes the derivative of f with respect to λ. When $m = 0$ the quadrature degenerates to Gauss quadrature and we have

$$\omega_i = I\left[\frac{\pi_n(\lambda)}{\pi_n'(\theta_i)(\lambda - \theta_i)}\right].$$

Different choices of these parameters yield different quadrature rules: $m = 0$ gives Gauss quadrature [71]; $m = 1$ with $\tau_1 = \lambda_{\min}$ ($\tau_1 = \lambda_{\max}$) gives left (right) Gauss-Radau quadrature [152]; $m = 2$ with $\tau_1 = \lambda_{\min}$ and $\tau_2 = \lambda_{\max}$ yields Gauss-Lobatto quadrature [122]; while for general m we obtain Gauss-Christoffel quadrature [72].

Although we have specified how to select nodes and weights for quadrature, these ideas cannot be applied to our problem in Eq. 3.2.1 because the measure α is unknown. Indeed, calculating the measure explicitly would require knowing the entire spectrum of A, which is as good as explicitly computing $f(A)$, hence untenable for us. The next section shows how to circumvent the difficulties due to unknown α.

The key idea to circumvent our lack of knowledge of α is to recursively construct polynomials called *Lanczos polynomials*. The construction ensures their orthogonality with respect to α. Concretely, we construct Lanczos polynomials via the following three-term recurrence:

$$\begin{aligned} &\beta_i p_i(\lambda) = (\lambda - \alpha_i)p_{i-1}(\lambda) - \beta_{i-1}p_{i-2}(\lambda), \quad i = 1, 2, \ldots, n \\ &p_{-1}(\lambda) \equiv 0; \quad p_0(\lambda) \equiv 1, \end{aligned} \tag{3.2.3}$$

while ensuring $\int_{\lambda_{\min}}^{\lambda_{\max}} d\alpha(\lambda) = 1$. We can express (3.2.3) in matrix form by writing

$$\lambda P_n(\lambda) = J_n P_n(\lambda) + \beta_n p_n(\lambda) e_n,$$

where $P_n(\lambda) := [p_0(\lambda), \ldots, p_{n-1}(\lambda)]^\top$, e_n is nth canonical unit vector, and J_n is the tridiagonal matrix

$$J_n = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \beta_2 & \ddots & \ddots & \\ & & \ddots & \alpha_{n-1} & \beta_{n-1} \\ & & & \beta_{n-1} & \alpha_n \end{bmatrix}. \tag{3.2.4}$$

This matrix is known as the *Jacobi matrix*, and is closed related to Gauss quadrature. The following well-known theorem makes this relation precise.

Theorem 7 ([158, 84]). *The eigenvalues of J_n form the nodes $\{\theta_i\}_{i=1}^n$ of Gauss-type quadratures. The weights $\{\omega_i\}_{i=1}^n$ are given by the squares of the first elements of the normalized eigenvectors of J_n.*

Thus, if J_n has the eigendecomposition $J_n = P_n^\top \Gamma P_n$, then for Gauss quadrature Thm. 7 yields

$$Q_n = \sum\nolimits_{i=1}^{n} \omega_i f(\theta_i) = e_1^\top P_n^\top f(\Gamma) P_n e_1 = e_1^\top f(J_n) e_1. \tag{3.2.5}$$

Specialization. We now specialize to our main focus, $f(A) = A^{-1}$, for which we prove more precise results. In this case, (3.2.5) becomes $Q_n = [J_n^{-1}]_{1,1}$. The task now is to compute Q_n, and given A, u to obtain the Jacobi matrix J_n.

Fortunately, we can efficiently calculate J_n iteratively using the *Lanczos Algorithm* [109]. Suppose we have an estimate J_i, in iteration $(i+1)$ of Lanczos, we compute the tridiagonal coefficients α_{i+1} and β_{i+1} and add them to this estimate to form J_{i+1}. As to Q_n, assuming we have already computed $[J_i^{-1}]_{1,1}$, letting $j_i = J_i^{-1} e_i$ and invoking the Sherman-Morrison identity [163] we obtain the recursion:

$$[J_{i+1}^{-1}]_{1,1} = [J_i^{-1}]_{1,1} + \frac{\beta_i^2 ([j_i]_1)^2}{\alpha_{i+1} - \beta_i^2 [j_i]_i}, \tag{3.2.6}$$

where $[j_i]_1$ and $[j_i]_i$ can be recursively computed using a Cholesky-like factorization of J_i [82, p.31].

For Gauss-Radau quadrature, we need to modify J_i so that it has a prescribed eigenvalue. More precisely, we extend J_i to J_i^{lr} for left Gauss-Radau (J_i^{rr} for right Gauss-Radau) with β_i on the off-diagonal and α_i^{lr} (α_i^{rr}) on the diagonal, so that J_i^{lr} (J_i^{rr}) has a prescribed eigenvalue of $\lambda_{\min}$ ($\lambda_{\max}$).

For Gauss-Lobatto quadrature, we extend J_i to J_i^{lo} with values β_i^{lo} and α_i^{lo} chosen to ensure that J_i^{lo} has the prescribed eigenvalues $\lambda_{\min}$ and $\lambda_{\max}$. For more detailed on the construction, see [80].

For all methods, the approximated values are calculated as $[(J_i')^{-1}]_{1,1}$, where $J_i' \in \{J_i^{\text{lr}}, J_i^{\text{rr}}, J_i^{\text{lo}}\}$ is the modified Jacobi matrix. Here J_i' is constructed at the i-th iteration of the algorithm.

Algorithm 3 Gauss Quadrature Lanczos (GQL)

Require: u and A the corresponding vector and matrix, $\lambda_{\min}$ and $\lambda_{\max}$ lower and upper bounds for the spectrum of A

Ensure: g_i, g_i^{rr}, g_i^{lr} and g_i^{lo} the Gauss, right Gauss-Radau, left Gauss-Radau and Gauss-Lobatto quadrature computed at i-th iteration

Initialize: $u_{-1} = 0$, $u_0 = u/\|u\|$, $\alpha_1 = u_0^\top A u_0$, $\beta_1 = \|(A - \alpha_1 I)u_0\|$, $g_1 = \|u\|/\alpha_1$, $c_1 = 1$, $\delta_1 = \alpha_1$, $\delta_1^{\text{lr}} = \alpha_1 - \lambda_{\min}$, $\delta_1^{\text{rr}} = \alpha_1 - \lambda_{\max}$, $u_1 = (A - \alpha_1 I)u_0/\beta_1$, $i = 2$

while $i \leq n$ **do**

$\alpha_i = u_{i-1}^\top A u_{i-1}$ ▷ Lanczos Iteration

$\tilde{u}_i = A u_{i-1} - \alpha_i u_{i-1} - \beta_{i-1} u_{i-2}$

$\beta_i = \|\tilde{u}_i\|$

$u_i = \tilde{u}_i/\beta_i$

$g_i = g_{i-1} + \frac{\|u\|\beta_{i-1}^2 c_{i-1}^2}{\delta_{i-1}(\alpha_i\delta_{i-1} - \beta_{i-1}^2)}$ ▷ Update g_i with Sherman-Morrison formula

$c_i = c_{i-1}\beta_{i-1}/\delta_{i-1}$

$\delta_i = \alpha_i - \frac{\beta_{i-1}^2}{\delta_{i-1}}$, $\delta_i^{\text{lr}} = \alpha_i - \lambda_{\min} - \frac{\beta_{i-1}^2}{\delta_{i-1}^{\text{lr}}}$, $\delta_i^{\text{rr}} = \alpha_i - \lambda_{\max} - \frac{\beta_{i-1}^2}{\delta_{i-1}^{\text{rr}}}$

$\alpha_i^{\text{lr}} = \lambda_{\min} + \frac{\beta_i^2}{\delta_i^{\text{lr}}}$, $\alpha_i^{\text{rr}} = \lambda_{\max} + \frac{\beta_i^2}{\delta_i^{\text{rr}}}$ ▷ Solve for J_i^{lr} and J_i^{rr}

$\alpha_i^{\text{lo}} = \frac{\delta_i^{\text{lr}}\delta_i^{\text{rr}}}{\delta_i^{\text{rr}} - \delta_i^{\text{lr}}}(\frac{\lambda_{\max}}{\delta_i^{\text{lr}}} - \frac{\lambda_{\min}}{\delta_i^{\text{rr}}})$, $(\beta_i^{\text{lo}})^2 = \frac{\delta_i^{\text{lr}}\delta_i^{\text{rr}}}{\delta_i^{\text{rr}} - \delta_i^{\text{lr}}}(\lambda_{\max} - \lambda_{\min})$ ▷ Solve for J_i^{lo}

$g_i^{\text{lr}} = g_i + \frac{\beta_i^2 c_i^2 \|u\|}{\delta_i(\alpha_i^{\text{lr}}\delta_i - \beta_i^2)}$, $g_i^{\text{rr}} = g_i + \frac{\beta_i^2 c_i^2 \|u\|}{\delta_i(\alpha_i^{\text{rr}}\delta_i - \beta_i^2)}$, $g_i^{\text{lo}} = g_i + \frac{(\beta_i^{\text{lo}})^2 c_i^2 \|u\|}{\delta_i(\alpha_i^{\text{lo}}\delta_i - (\beta_i^{\text{lo}})^2)}$ ▷ Update g_i^{rr}, g_i^{lr} and g_i^{lo} with Sherman-Morrison formula

$i = i + 1$

end while

The algorithm for computing Gauss, Gauss-Radau, and Gauss-Lobatto quadrature rules with the help of Lanczos iteration is called *Gauss Quadrature Lanczos* (GQL) and is shown in [81]. We recall its pseudocode in Alg. 3 to make our presentation self-contained (and for our proofs in Sec. 3.3).

The error of approximating $I[f]$ by Gauss-type quadratures can be expressed as

$$R_n[f] = \frac{f^{(2n+m)}(\xi)}{(2n+m)!} I[\rho_m \pi_n^2],$$

for some $\xi \in [\lambda_{\min}, \lambda_{\max}]$ (see, e.g., [173]). Note that ρ_m does not change sign in $[\lambda_{\min}, \lambda_{\max}]$; but with different values of m and τ_j we obtain different (but fixed) signs for $R_n[f]$ using $f(\lambda) = 1/\lambda$ and $\lambda_{\min} > 0$. Concretely, for Gauss quadrature $m = 0$ and $R_n[f] \geq 0$; for left Gauss-Radau $m = 1$ and $\tau_1 = \lambda_{\min}$, so we have $R_n[f] \leq 0$; for right Gauss-Radau we have $m = 1$ and $\tau_1 = \lambda_{\max}$, thus $R_n[f] \geq 0$; while for Gauss-Lobatto

we have $m = 2$, $\tau_1 = \lambda_{\min}$ and $\tau_2 = \lambda_{\max}$, so that $R_n[f] \leq 0$. This behavior of the errors clearly shows the ordering relations between the target values and the approximations made by the different quadrature rules. Lemma 8 (see e.g., [135]) makes this claim precise.

Lemma 8. *Let g_i, g_i^{lr}, g_i^{rr}, and g_i^{lo} be the approximations at the i-th iteration of Gauss, left Gauss-Radau, right Gauss-Radau, and Gauss-Lobatto quadrature, respectively. Then, g_i and g_i^{rr} provide lower bounds on $u^\top A^{-1}u$, while g_i^{lr} and g_i^{lo} provide upper bounds.*

While Gauss-type quadratures relate to the Lanczos algorithm, Lanczos itself is closely related to conjugate gradient (CG) [93], a well-known method for solving $Ax = b$ for positive definite A.

We recap this connection below. Let x_k be the estimated solution at the k-th CG iteration. If x^* denotes the true solution to $Ax = b$, then the *error* ε_k and *residual* r_k are defined as

$$\varepsilon_k := x^* - x_k, \qquad r_k = A\varepsilon_k = b - Ax_k, \tag{3.2.7}$$

At the k-th iteration, x_k is chosen such that r_k is orthogonal to the k-th *Krylov space*, i.e., the linear space $\mathcal{K}_k$ spanned by $\{r_0, Ar_0, \ldots, A^{k-1}r_0\}$. It can be shown [154] that r_k is a scaled Lanczos vector from the k-th iteration of Lanczos started with r_0. Noting the relation between Lanczos and Gauss quadrature applied to appoximate $r_0^\top A^{-1} r_0$, one obtains the following theorem that relates CG with GQL.

Theorem 9 (CG and GQL; [136]). *Let ε_k be the error as in* (3.2.7)*, and let $\|\varepsilon_k\|_A^2 := \varepsilon_k^T A \varepsilon_k$. Then, it holds that*

$$\|\varepsilon_k\|_A^2 = \|r_0\|^2([J_n^{-1}]_{1,1} - [J_k^{-1}]_{1,1}),$$

where J_k is the Jacobi matrix at the k-th Lanczos iteration starting with r_0.

Finally, the rate at which $\|\varepsilon_k\|_A^2$ shrinks has also been well-studied, as noted below.

Theorem 10 (CG rate, see e.g. [164]). *Let ε_k be the error made by CG at iteration k when started with x_0. Let κ be the condition number of A, i.e., $\kappa = \lambda_1/\lambda_n$. Then, the error norm*

at iteration k satisfies

$$\|\varepsilon_k\|_A \leq 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^k \|\varepsilon_0\|_A.$$

Due to these explicit relations between CG and Lanczos, as well as between Lanczos and Gauss quadrature, we readily obtain the following convergence rate for relative error of Gauss quadrature.

Theorem 11 (Gauss quadrature rate). *The i-th iterate of Gauss quadrature satisfies the relative error bound*

$$\frac{g_n - g_i}{g_n} \leq 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i.$$

Proof. This is obtained by exploiting relations among CG, Lanczos and Gauss quadrature. Set $x_0 = 0$ and $b = u$. Then, $\varepsilon_0 = x^*$ and $r_0 = u$. An application of Thm. 9 and Thm. 10 thus yields the bound

$$\begin{aligned}\|\varepsilon_i\|_A^2 &= \|u\|^2([J_n^{-1}]_{1,1} - [J_i^{-1}]_{1,1}) = g_n - g_i \\ &\leq 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i \|\varepsilon_0\|_A = 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i u^\top A^{-1} u = 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i g_n\end{aligned}$$

where the last equality draws from Lemma 18. □

In other words, Thm. 11 shows that the iterates of Gauss quadrature converge linearly.

3.3 Main Theoretical Results

In this section we summarize our main theoretical results and some empirical evidence that supports our theory, and in the next section we will show detailed proofs for these results. The key questions that we answer are: (i) do the bounds on $u^\top A^{-1} u$ generated by GQL improve monotonically with each iteration; (ii) how tight are these bounds; and (iii) how fast do Gauss-Radau and Gauss-Lobatto iterations converge? Our answers not only fill gaps in the literature on quadrature, but provide the theoretical base for building fast algorithms

for widely used applications (see Sec. 3.1.3 and Sec. 3.6).

3.3.1 Lower Bounds

Our first result shows that both Gauss and right Gauss-Radau quadratures give iteratively better lower bounds on $u^\top A^{-1}u$. Moreover, with the same number of iterations, right Gauss-Radau yields tighter bounds.

Theorem 12. *Let $i < n$. Then, g_i^{rr} yields better bounds than g_i but worse bounds than g_{i+1}; more precisely,*

$$g_i \leq g_i^{rr} \leq g_{i+1}, \quad i < n. \tag{3.3.1}$$

Combining Thm. 12 with the convergence rate of relative error for Gauss quadrature (Thm. 11) we obtain the following convergence rate estimate for right Gauss-Radau.

Theorem 13 (Relative error right Gauss-Radau)**.** *For each i, the right Gauss-Radau iterate g_i^{rr} satisfies*

$$\frac{g_n - g_i^{rr}}{g_n} \leq 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i.$$

3.3.2 Upper Bounds

Our second result compares Gauss-Lobatto with left Gauss-Radau quadrature.

Theorem 14. *Let $i < n$. Then, g_i^{lr} gives better upper bounds than g_i^{lo} but worse than g_{i+1}^{lo}; more precisely,*

$$g_{i+1}^{lo} \leq g_i^{lr} \leq g_i^{lo}, \quad i < n.$$

This shows that bounds given by both Gauss-Lobatto and left Gauss-Radau become tighter with each iteration. For the same number of iterations, left Gauss-Radau provides a tighter bound than Gauss-Lobatto.

Combining the above two theorems, we obtain the following corollary for all four Gauss-type quadratures.

Corollary 15 (Monotonicity). *With increasing i, g_i and g_i^{rr} give increasingly better lower bounds and g_i^{lr} and g_i^{lo} give increasingly better upper bounds, that is,*

$$g_i \leq g_{i+1}; \quad g_i^{rr} \leq g_{i+1}^{rr};$$
$$g_i^{lr} \geq g_{i+1}^{lr}; \quad g_i^{lo} \geq g_{i+1}^{lo}.$$

3.3.3 Convergence rates

Our next two results prove linear convergence rates for left Gauss-Radau quadrature and Gauss-Lobatto quadrature applied to computing the BIF $u^T A^{-1} u$.

Theorem 16 (Relative error left Gauss-Radau). *For each i, the left Gauss-Radau iterate g_i^{lr} satisfies*

$$\frac{g_i^{lr} - g_n}{g_n} \leq 2\kappa^+ \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^i,$$

where $\kappa^+ := \lambda_n / \lambda_{\min}$, $i < n$.

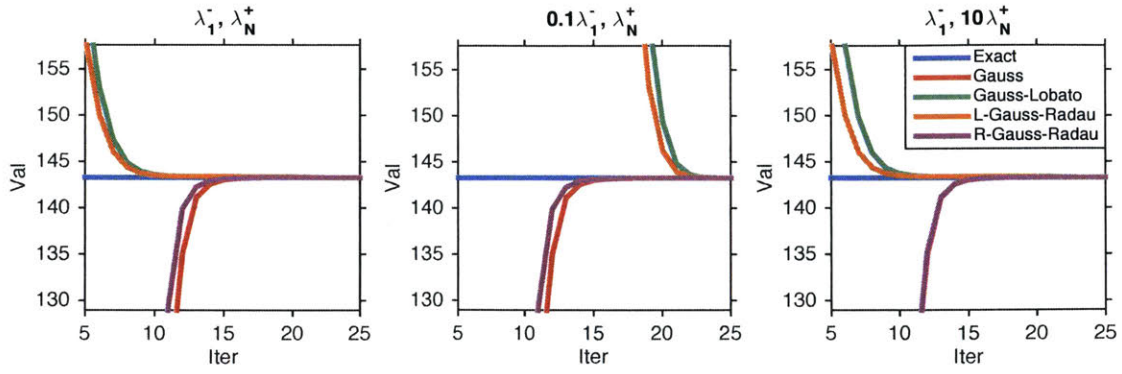
Theorem 16 shows that the error again decreases linearly, and it also depends on $\lambda_{\min}$, our estimate of the smallest eigenvalue that determines the range of integration. Using the relations between left Gauss-Radau and Gauss-Lobatto, we readily obtain the following corollary.

Corollary 17 (Relative error Gauss-Lobatto). *For each i, the Gauss-Lobatto iterate g_i^{lo} satisfies*

$$\frac{g_i^{lo} - g_n}{g_n} \leq 2\kappa^+ \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\right)^{i-1},$$

where $\kappa^+ := \lambda_n / \lambda_{\min}$ and $i < n$.

Remarks All aforementioned results assumed that A is strictly positive definite with simple eigenvalues. In Appendix 3.5, we show similar results for the more general case that A is only required to be symmetric, and u lies in the space spanned by eigenvectors of A corresponding to distinct positive eigenvalues.



(a) (b) (c)

Figure 3-1: Lower and upper bounds computed by Gauss-type quadrature in each iteration on $u^\top A^{-1}u$ with $A \in \mathbb{R}^{100\times 100}$.

3.3.4 Empirical Evidence

We present an empirical verification of the theoretical results proved above. We generate a random symmetric matrix $A \in \mathbb{R}^{100\times 100}$ with density 10%, where each entry is either zero or standard normal, and shift its diagonal entries to make its smallest eigenvalue $\lambda_1 = 10^{-2}$, thus making A positive definite. We set $\lambda_{\min} = \lambda_1^- = (\lambda_1 - 10^{-5})$ and $\lambda_{\max} = \lambda_n^+ = (\lambda_n + 10^{-5})$. We randomly sample $u \in \mathbb{R}^{100}$ from a standard Gaussian distribution. Fig. 3-1 illustrates how the lower and upper bounds given by the four quadrature rules evolve with the number of iterations.

Fig. 3-1 (b) and (c) show the sensitivity of the rules (except Gauss quadrature) to estimating the extremal eigenvalues. Specifically, we use $\lambda_{\min} = 0.1\lambda_1^-$ and $\lambda_{\max} = 10\lambda_n^+$.

The plots in Fig. 3-1 agree with the theoretical results. First, all quadrature rules are seen to yield iteratively tighter bounds. The bounds obtained by the Gauss-Radau quadrature are superior to those given by Gauss and Gauss-Lobatto quadrature (also numerically verified). Notably, the bounds given by all quadrature rules converge very fast – within 25 iterations they yield reasonably tight bounds.

It is valuable to see how the bounds are affected if we do not have good approximations to the extremal eigenvalues λ_1 and λ_n. Since Gauss quadrature does not depend on the approximations $\lambda_{\min} < \lambda_1$ and $\lambda_{\max} > \lambda_n$, its bounds remain the same in (a),(b),(c). Left Gauss-Radau depends on the quality of $\lambda_{\min}$, and, with a poor approximation takes more

iterations to converge (Fig. 3-1(b)). Right Gauss-Radau depends on the quality of $\lambda_{\max}$; thus, if we use $\lambda_{\max} = 10\lambda_n^+$ as our approximation, its bounds become worse (Fig. 3-1(c)). However, its bounds are never worse than those obtained by Gauss quadrature. Finally, Gauss-Lobato depends on both $\lambda_{\min}$ and $\lambda_{\max}$, so its bounds become worse whenever we lack good approximations to λ_1 or λ_n. Nevertheless, its quality is lower-bounded by left Gauss-Radau as stated in Thm. 14.

3.4 Proofs for Main Theoretical Results

In this section we show detailed proofs for main results we mentioned in Section 3.3. We begin by proving an exactness property of Gauss and Gauss-Radau quadrature.

Lemma 18 (Exactness). *With A being symmetric positive definite with simple eigenvalues, the iterates g_n, g_n^{lr}, and g_n^{rr} are exact. Namely, after n iterations they satisfy*

$$g_n = g_n^{lr} = g_n^{rr} = u^\top A^{-1} u.$$

Proof. Observe that the Jacobi tridiagonal matrix can be computed via Lanczos iteration, and Lanczos is essentially essentially an iterative tridiagonalization of A. At the i-th iteration we have $J_i = V_i^\top A V_i$, where $V_i \in \mathbb{R}^{n\times i}$ are the first i Lanczos vectors (i.e., a basis for the i-th Krylov space). Thus, $J_n = V_n^\top A V_n$ where V_n is an $n \times n$ orthonormal matrix, showing that J_n has the same eigenvalues as A. As a result $\pi_n(\lambda) = \prod_{i=1}^n (\lambda - \lambda_i)$, and it follows that the remainder

$$R_n[f] = \frac{f^{(2n)}(\xi)}{(2n)!} I[\pi_n^2] = 0,$$

for some scalar $\xi \in [\lambda_{\min}, \lambda_{\max}]$, which shows that g_n is exact for $u^\top A^{-1} u$. For left and right Gauss-Radau quadrature, we have $\beta_n = 0$, $\alpha_n^{\text{lr}} = \lambda_{\min}$, and $\alpha_n^{\text{rr}} = \lambda_{\max}$, while all other elements of the $(n+1)$-th row or column of J_n' are zeros. Thus, the eigenvalues of J_n' are $\lambda_1, \ldots, \lambda_n, \tau_1$, and $\pi_n(\lambda)$ again equals $\prod_{i=1}^n (\lambda - \lambda_i)$. As a result, the remainder satisfies

$$R_n[f] = \frac{f^{(2n)}(\xi)}{(2n)!} I[(\lambda - \tau_1)\pi_n^2] = 0,$$

from which it follows that both g_n^{rr} and g_n^{lr} are exact. □

The convergence rate in Thm. 10 and the final exactness of iterations in Lemma 18 does not necessarily indicate that we are making progress at each iterations. However, by exploiting the relations to CG we can indeed conclude that we are making progress in each iteration in Gauss quadrature.

Theorem 19. *The approximation g_i generated by Gauss quadrature is monotonically non-decreasing, i.e.,*

$$g_i \leq g_{i+1}, \quad \textit{for } i < n.$$

Proof. At each iteration r_i is taken to be orthogonal to the i-th Krylov space: $\mathcal{K}_i = \text{span}\{u, Au, \ldots, A^{i-1}u\}$. Let Π_i be the projection onto the complement space of $\mathcal{K}_i$. The residual then satisfies

$$\begin{aligned}\|\varepsilon_{i+1}\|_A^2 &= \varepsilon_{i+1}^T A \varepsilon_{i+1} = r_{i+1}^\top A^{-1} r_{i+1} \\ &= (\Pi_{i+1} r_i)^\top A^{-1} \Pi_{i+1} r_i \\ &= r_i^\top (\Pi_{i+1}^\top A^{-1} \Pi_{i+1}) r_i \leq r_i A^{-1} r_i,\end{aligned}$$

where the last inequality follows from $\Pi_{i+1}^\top A^{-1} \Pi_{i+1} \preceq A^{-1}$. Thus $\|\varepsilon_i\|_A^2$ is monotonically nonincreasing, whereby $g_n - g_i \geq 0$ is monotonically decreasing and thus g_i is monotonically nondecreasing. □

Before we proceed to Gauss-Radau, let us recall a useful theorem and its corollary.

Theorem 20 (Lanczos Polynomial [82])**.** *Let u_i be the vector generated by Alg. 3 at the i-th iteration; let p_i be the Lanczos polynomial of degree i. Then we have*

$$u_i = p_i(A) u_0, \quad \textit{where } p_i(\lambda) = (-1)^i \frac{\det(J_i - \lambda I)}{\prod_{j=1}^i \beta_j}.$$

From the expression of Lanczos polynomial we have the following corollary specifying the sign of the polynomial at specific points.

Corollary 21. *Assume $i < n$. If i is odd, then $p_i(\lambda_{\min}) < 0$; for even i, $p_i(\lambda_{\min}) > 0$, while $p_i(\lambda_{\max}) > 0$ for any $i < n$.*

Proof. Since $J_i = V_i^\top A V_i$ is similar to A, its spectrum is bounded by $\lambda_{\min}$ and $\lambda_{\max}$ from left and right. Thus, $J_i - \lambda_{\min}$ is positive semi-definite, and $J_i - \lambda_{\max}$ is negative semi-definite. Taking $(-1)^i$ into consideration we will get the desired conclusions. □

We are ready to state our main result that compares (right) Gauss-Radau with Gauss quadrature.

Theorem 22 (Thm. 12 in Section 3.3). *Let $i < n$. Then, g_i^{rr} gives better bounds than g_i but worse bounds than g_{i+1}; more precisely,*

$$g_i \leq g_i^{rr} \leq g_{i+1}, \quad i < n. \tag{3.4.1}$$

Proof. We prove inequality (3.4.1) using the recurrences satisfied by g_i and g_i^{rr} (see Alg. 3) *Upper bound:* $g_i^{rr} \leq g_{i+1}$. The iterative quadrature algorithm uses the recursive updates

$$g_i^{\text{rr}} = g_i + \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i^{\text{rr}}\delta_i - \beta_i^2)},$$
$$g_{i+1} = g_i + \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_{i+1}\delta_i - \beta_i^2)}.$$

It suffices to thus compare α_i^{rr} and α_{i+1}. The three-term recursion for Lanczos polynomials shows that

$$\beta_{i+1}p_{i+1}(\lambda_{\max}) = (\lambda_{\max} - \alpha_{i+1})p_i(\lambda_{\max}) - \beta_i p_{i-1}(\lambda_{\max}) > 0,$$
$$\beta_{i+1}p^*_{i+1}(\lambda_{\max}) = (\lambda_{\max} - \alpha_i^{\text{rr}})p_i(\lambda_{\max}) - \beta_i p_{i-1}(\lambda_{\max}) = 0,$$

where p_{i+1} is the original Lanczos polynomial, and p^*_{i+1} is the modified polynomial that has $\lambda_{\max}$ as a root. Noting that $p_i(\lambda_{\max}) > 0$, we see that $\alpha_{i+1} \leq \alpha_i^{\text{rr}}$. Moreover, from Thm. 19 we know that the g_i's are monotonically increasing, whereby $\delta_i(\alpha_{i+1}\delta_i - \beta_i^2) > 0$. It follows

that

$$0 < \delta_i(\alpha_{i+1}\delta_i - \beta_i^2) \leq \delta_i(\alpha_i^{\text{rr}}\delta_i - \beta_i^2),$$

and from this inequality it is clear that $g_i^{\text{rr}} \leq g_{i+1}$.

Lower-bound: $g_i \leq g_i^{rr}$. Since $\beta_i^2 c_i^2 \geq 0$ and $\delta_i(\alpha_i^{\text{rr}}\delta_i - \beta_i^2) \geq \delta_i(\alpha_{i+1}\delta_i - \beta_i^2) > 0$, we readily obtain

$$g_i \leq g_i + \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i^{\text{rr}}\delta_i - \beta_i^2)} = g_i^{\text{rr}}. \qquad \square$$

Combining Thm. 22 with the convergence rate of relative error for Gauss quadrature (Thm. 11) immediately yields the following convergence rate for right Gauss-Radau quadrature:

Theorem 23 (Relative error of right Gauss-Radau, Thm. 13 in Section 3.3). *For each i, the right Gauss-Radau g_i^{rr} iterates satisfy*

$$\frac{g_n - g_i^{rr}}{g_n} \leq 2\Big(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\Big)^i.$$

This results shows that with the same number of iterations, right Gauss-Radau gives superior approximation over Gauss quadrature, though they share the same relative error convergence rate.

Our second main result compares Gauss-Lobatto with (left) Gauss-Radau quadrature.

Theorem 24 (Thm. 14 in Section 3.3). *Let $i < n$. Then, g_i^{lr} gives better upper bounds than g_i^{lo} but worse than g_{i+1}^{lo}; more precisely,*

$$g_{i+1}^{lo} \leq g_i^{lr} \leq g_i^{lo}, \quad i < n.$$

Proof. We prove these inequalities using the recurrences for g_i^{lr} and g_i^{lo} from Alg. 3.

$g_i^{lr} \leq g_i^{lo}$: From Alg. 3 we observe that $\alpha_i^{\text{lo}} = \lambda_{\min} + \frac{(\beta_i^{\text{lo}})^2}{\delta_i^{\text{lr}}}$. Thus we can write g_i^{lr} and g_i^{lo} as

$$g_i^{\text{lr}} = g_i + \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i^{\text{lr}}\delta_i - \beta_i^2)} = g_i + \frac{\beta_i^2 c_i^2}{\lambda_{\min}\delta_i^2 + \beta_i^2(\delta_i^2/\delta_i^{\text{lr}} - \delta_i)}$$
$$g_i^{\text{lo}} = g_i + \frac{(\beta_i^{\text{lo}})^2 c_i^2}{\delta_i(\alpha_i^{\text{lo}}\delta_i - (\beta_i^{\text{lo}})^2)} = g_i + \frac{(\beta_i^{\text{lo}})^2 c_i^2}{\lambda_{\min}\delta_i^2 + (\beta_i^{\text{lo}})^2(\delta_i^2/\delta_i^{\text{lr}} - \delta_i)}$$

To compare these quantities, as before it is helpful to begin with the original three-term recursion for the Lanczos polynomial, namely

$$\beta_{i+1}p_{i+1}(\lambda) = (\lambda - \alpha_{i+1})p_i(\lambda) - \beta_i p_{i-1}(\lambda).$$

In the construction of Gauss-Lobatto, to make a new polynomial of order $i+1$ that has roots $\lambda_{\min}$ and $\lambda_{\max}$, we add $\sigma_1 p_i(\lambda)$ and $\sigma_2 p_{i-1}(\lambda)$ to the original polynomial to ensure

$$\begin{cases} \beta_{i+1}p_{i+1}(\lambda_{\min}) + \sigma_1 p_i(\lambda_{\min}) + \sigma_2 p_{i-1}(\lambda_{\min}) & = 0, \\ \beta_{i+1}p_{i+1}(\lambda_{\max}) + \sigma_1 p_i(\lambda_{\max}) + \sigma_2 p_{i-1}(\lambda_{\max}) & = 0. \end{cases}$$

Since $\beta_{i+1}, p_{i+1}(\lambda_{\max}), p_i(\lambda_{\max})$ and $p_{i-1}(\lambda_{\max})$ are all greater than 0, $\sigma_1 p_i(\lambda_{\max}) + \sigma_2 p_{i-1}(\lambda_{\max}) < 0$. To determine the sign of polynomials at $\lambda_{\min}$, consider the two cases:

- Odd i. In this case $p_{i+1}(\lambda_{\min}) > 0$, $p_i(\lambda_{\min}) < 0$, and $p_{i-1}(\lambda_{\min}) > 0$;

- Even i. In this case $p_{i+1}(\lambda_{\min}) < 0$, $p_i(\lambda_{\min}) > 0$, and $p_{i-1}(\lambda_{\min}) < 0$.

Thus, if $S = (\text{sgn}(\sigma_1), \text{sgn}(\sigma_2))$, where the signs take values in $\{0, \pm 1\}$, then $S \neq (1,1)$, $S \neq (-1,1)$ and $S \neq (0,1)$. Hence, $\sigma_2 \leq 0$ must hold, and thus $(\beta_i^{\text{lo}})^2 = (\beta_i - \sigma_2)^2 \geq \beta_i^2$ given that $\beta_i^2 > 0$ for $i < n$.

Using $(\beta_i^{\text{lo}})^2 \geq \beta_i^2$ with $\lambda_{\min} c_i^2 (\delta_i)^2 \geq 0$, an application of monotonicity of the univariate function $g(x) = \frac{ax}{b+cx}$ for $ab \geq 0$ to the recurrences defining g_i^{lr} and g_i^{lo} yields the desired inequality $g_i^{\text{lr}} \leq g_i^{\text{lo}}$.

$g^{lo}_{i+1} \le g^{lr}_i$: From recursion formulas we have

$$g_i^{\text{lr}} = g_i + \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i^{\text{lr}}\delta_i - \beta_i^2)},$$
$$g_{i+1}^{\text{lo}} = g_{i+1} + \frac{(\beta_{i+1}^{\text{lo}})^2 c_{i+1}^2}{\delta_{i+1}(\alpha_{i+1}^{\text{lo}}\delta_{i+1} - (\beta_{i+1}^{\text{lo}})^2)}.$$

Establishing $g_i^{\text{lr}} \ge g_{i+1}^{\text{lo}}$ thus amounts to showing that (noting the relations among g_i, g_i^{lr} and g_i^{lo}):

$$\begin{aligned}
& \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i^{\text{lr}}\delta_i - \beta_i^2)} - \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_{i+1}\delta_i - \beta_i^2)} \quad \ge \quad \frac{(\beta_{i+1}^{\text{lo}})^2 c_{i+1}^2}{\delta_{i+1}(\alpha_{i+1}^{\text{lo}}\delta_{i+1} - (\beta_{i+1}^{\text{lo}})^2)} \\
\iff & \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i^{\text{lr}}\delta_i - \beta_i^2)} - \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_{i+1}\delta_i - \beta_i^2)} \quad \ge \quad \frac{(\beta_{i+1}^{\text{lo}})^2 c_i^2 \beta_i^2}{(\delta_i)^2\delta_{i+1}(\alpha_{i+1}^{\text{lo}}\delta_{i+1} - (\beta_{i+1}^{\text{lo}})^2)} \\
\iff & \frac{1}{\alpha_i^{\text{lr}}\delta_i - \beta_i^2} - \frac{1}{\alpha_{i+1}\delta_i - \beta_i^2} \quad \ge \quad \frac{(\beta_{i+1}^{\text{lo}})^2}{\delta_i\delta_{i+1}(\alpha_{i+1}^{\text{lo}}\delta_{i+1} - (\beta_{i+1}^{\text{lo}})^2)} \\
\iff & \frac{1}{(\alpha_{i+1} - \delta_{i+1}^{\text{lr}}) - \beta_i^2/\delta_i} - \frac{1}{\alpha_{i+1} - \beta_i^2/\delta_i} \quad \ge \quad \frac{1}{\delta_{i+1}(\alpha_{i+1}^{\text{lo}}\delta_{i+1}/(\beta_{i+1}^{\text{lo}})^2 - 1)} \qquad \text{(Lemma 26)} \\
\iff & \frac{1}{\delta_{i+1} - \delta_{i+1}^{\text{lr}}} - \frac{1}{\delta_{i+1}} \quad \ge \quad \frac{1}{\delta_{i+1}(\frac{\lambda_{\min}\delta_{i+1}}{(\beta_{i+1}^{\text{lo}})^2} + \frac{\delta_{i+1}}{\delta_{i+1}^{\text{lr}}} - 1)} \\
\iff & \frac{\lambda_{\min}\delta_{i+1}}{(\beta_{i+1}^{\text{lo}})^2} + \frac{\delta_{i+1}}{\delta_{i+1}^{\text{lr}}} - 1 \ge \frac{\delta_{i+1}}{\delta_{i+1}^{\text{lr}}} - 1 \\
\iff & \frac{\lambda_{\min}\delta_{i+1}}{(\beta_{i+1}^{\text{lo}})^2} \ge 0,
\end{aligned}$$

where the last inequality is obviously true; hence the proof is complete. □

In summary, we have the following corollary for all the four quadrature rules:

Corollary 25 (Monotonicity of Lower and Upper Bounds, Corr. 15 in Section 3.3)**.** *As the iteration proceeds, g_i and g_i^{rr} gives increasingly better asymptotic lower bounds and g_i^{lr} and g_i^{lo} gives increasingly better upper bounds, namely*

$$g_i \le g_{i+1}; \quad g_i^{rr} \le g_{i+1}^{rr}; \quad g_i^{lr} \ge g_{i+1}^{lr}; \quad g_i^{lo} \ge g_{i+1}^{lo}.$$

Proof. Directly drawn from Thm. 19, Thm. 22 and Thm. 24. □

Before proceeding further to our analysis of convergence rates of left Gauss-Radau and Gauss-Lobatto, we note two technical results that we will need.

Lemma 26. *Let α_{i+1} and α_i^{lr} be as in Alg. 3. The difference $\Delta_{i+1} = \alpha_{i+1} - \alpha_i^{lr}$ satisfies $\Delta_{i+1} = \delta_{i+1}^{lr}$.*

Proof. From the Lanczos polynomials in the definition of left Gauss-Radau quadrature we have

$$\begin{aligned}\beta_{i+1}p_{i+1}^*(\lambda_{\min}) &= \big(\lambda_{\min} - \alpha_i^{\text{lr}}\big)p_i(\lambda_{\min}) - \beta_i p_{i-1}(\lambda_{\min}) \\ &= \big(\lambda_{\min} - (\alpha_{i+1} - \Delta_{i+1})\big)p_i(\lambda_{\min}) - \beta_i p_{i-1}(\lambda_{\min}) \\ &= \beta_{i+1}p_{i+1}(\lambda_{\min}) + \Delta_{i+1}p_i(\lambda_{\min}) = 0.\end{aligned}$$

Rearrange this equation to write $\Delta_{i+1} = -\beta_{i+1}\frac{p_{i+1}(\lambda_{\min})}{p_i(\lambda_{\min})}$, which can be further rewritten as

$$\Delta_{i+1} \overset{\text{Thm. 20}}{=} -\beta_{i+1}\frac{(-1)^{i+1}\det(J_{i+1} - \lambda_{\min}I)/\prod_{j=1}^{i+1}\beta_j}{(-1)^{i}\det(J_i - \lambda_{\min}I)/\prod_{j=1}^{i}\beta_j} = \frac{\det(J_{i+1} - \lambda_{\min}I)}{\det(J_i - \lambda_{\min}I)} = \delta_{i+1}^{\text{lr}}.\square$$

Remark 27. Lemma 26 has an implication beyond its utility for the subsequent proofs: it provides a new way of calculating α_{i+1} given the quantities δ_{i+1}^{lr} and α_i^{lr}; this saves calculation in Alg. 3.

The following lemma relates δ_i to δ_i^{lr}, which will prove useful in subsequent analysis.

Lemma 28. *Let δ_i^{lr} and δ_i be computed in the i-th iteration of Alg. 3. Then, we have the following:*

$$\delta_i^{lr} < \delta_i, \tag{3.4.2}$$

$$\frac{\delta_i^{lr}}{\delta_i} \leq 1 - \frac{\lambda_{\min}}{\lambda_n}. \tag{3.4.3}$$

Proof. We prove (3.4.2) by induction. Since $\lambda_{\min} > 0$, $\delta_1 = \alpha_1 > \lambda_{\min}$ and $\delta_1^{\text{lr}} = \alpha - \lambda_{\min}$ we know that $\delta_1^{\text{lr}} < \delta_1$. Assume that $\delta_i^{\text{lr}} < \delta_i$ is true for all $i \leq k$ and considering the

$(k+1)$-th iteration:

$$\delta_{k+1}^{\text{lr}} = \alpha_{k+1} - \lambda_{\min} - \frac{\beta_k^2}{\delta_k^{\text{lr}}} < \alpha_{k+1} - \frac{\beta_k^2}{\delta_k} = \delta_{k+1}.$$

To prove (3.4.3), simply observe the following

$$\frac{\delta_i^{\text{lr}}}{\delta_i} = \frac{\alpha_i - \lambda_{\min} - \beta_{i-1}^2/\delta_{i-1}^{\text{lr}}}{\alpha_i - \beta_{i-1}^2/\delta^{i-1}} \overset{(3.4.2)}{\leq} \frac{\alpha_i - \lambda_{\min}}{\alpha_i} \leq 1 - \frac{\lambda_{\min}}{\lambda_n}. \qquad \square$$

With aforementioned lemmas we will be able to show how fast the difference between g_i^{lr} and g_i decays. Note that g_i^{lr} gives an upper bound on the objective while g_i gives a lower bound.

Lemma 29. *The difference between g_i^{lr} and g_i decreases linearly. More specifically we have*

$$g_i^{lr} - g_i \leq 2\kappa^+ (\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1})^i g_n$$

where $\kappa^+ = \lambda_n/\lambda_{\min}$ and κ is the condition number of A, i.e., $\kappa = \lambda_n/\lambda_1$.

Proof. We rewrite the difference $g_i^{\text{lr}} - g_i$ as follows

$$\begin{aligned} g_i^{\text{lr}} - g_i &= \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i^{\text{lr}}\delta_i - \beta_i^2)} = \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_{i+1}\delta_i - \beta_i^2)} \frac{\delta_i(\alpha_{i+1}\delta_i - \beta_i^2)}{\delta_i(\alpha_i^{\text{lr}}\delta_i - \beta_i^2)} \\ &= \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_{i+1}\delta_i - \beta_i^2)} \frac{1}{\big(\alpha_i^{\text{lr}} - \beta_i^2/\delta_i\big)/\big(\alpha_{i+1} - \beta_i^2/\delta_i\big)} = \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i\delta_i - \beta_i^2)} \frac{1}{1 - \Delta_{i+1}/\delta_{i+1}}, \end{aligned}$$

where $\Delta_{i+1} = \alpha_{i+1} - \alpha_i^{\text{lr}}$. Next, recall that $\frac{g_n - g_i}{g_n} \leq 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i$. Since g_i lower bounds g_n, we have

$$\begin{aligned} \Big(1 - 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i\Big) g_n \leq g_i \leq g_n, \\ \Big(1 - 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^{i+1}\Big) g_n \leq g_{i+1} \leq g_n. \end{aligned}$$

Thus, we can conclude that

$$\frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i\delta_i - \beta_i^2)} = g_{i+1} - g_i \leq 2\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i g_n.$$

Now we focus on the term $\big(1 - \Delta_{i+1}/\delta_{i+1}\big)^{-1}$. Using Lemma 26 we know that $\Delta_{i+1} = \delta_{i+1}^{\text{lr}}$. Hence,

$$1-\Delta_{i+1}/\delta_{i+1} = 1 - \delta_{i+1}^{\text{lr}}/\delta_{i+1} \geq 1 - (1 - \lambda_{\min}/\lambda_n) = \lambda_{\min}/\lambda_n \triangleq \frac{1}{\kappa^+}.$$

Finally we have

$$g_i^{\text{lr}} - g_i = \frac{\beta_i^2 c_i^2}{\delta_i(\alpha_i\delta_i - \beta_i^2)} \frac{1}{1 - \Delta_{i+1}/\delta_{i+1}} \leq 2\kappa^+\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i g_n.$$

□

Theorem 30 (Relative error of left Gauss-Radau, Thm. 16 in Section 3.3). *For left Gauss-Radau quadrature where the preassigned node is $\lambda_{\min}$, we have the following bound on relative error:*

$$\frac{g_i^{lr} - g_n}{g_n} \leq 2\kappa^+\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i,$$

where $\kappa^+ := \lambda_n/\lambda_{\min}$, $i < n$.

Proof. Write $g_i^{\text{lr}} = g_i + (g_i^{\text{lr}} - g_i)$. Since $g_i \leq g_n$, using Lemma 29 to bound the second term we obtain

$$g_i^{\text{lr}} \leq g_n + 2\kappa^+\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^i g_n,$$

from which the claim follows upon rearrangement. □

Due to the relations between left Gauss-Radau and Gauss-Lobatto, we have the following corollary:

Corollary 31 (Relative error of Gauss-Lobatto, Corr. 17 in Section 3.3). *For Gauss-Lobatto quadrature, we have the following bound on relative error:*

$$\frac{g_i^{lo} - g_n}{g_n} \leq 2\kappa^+\Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^{i-1}, \tag{3.4.4}$$

where $\kappa^+ := \lambda_n/\lambda_{\min}$ *and* $i < n$.

3.5 Generalization: Symmetric Matrices

In this section we consider the case where u lies in the column space of several top eigenvectors of A, and discuss how the aforementioned theorems vary. In particular, note that the previous analysis assumes that A is positive definite. With our analysis in this section we relax this assumption to the more general case where A is symmetric with simple eigenvalues, though we require u to lie in the space spanned by eigenvectors of A corresponding to positive eigenvalues.

We consider the case where A is symmetric and has the eigendecomposition of $A = Q\Lambda Q^\top = \sum_{i=1}^n \lambda_i q_i q_i^\top$ where λ_i's are eigenvalues of A increasing with i and q_i's are corresponding eigenvectors. Assume that u lies in the column space spanned by top k eigenvectors of A where all these k eigenvectors correspond to positive eigenvalues. Namely we have $u \in \text{Span}\{\{q_i\}_{i=n-k+1}^n\}$ and $0 < \lambda_{n-k+1}$.

Since we only assume that A is symmetric, it is possible that A is singular and thus we consider the value of $u^\top A^\dagger u$, where $A^\dagger$ is the pseudo-inverse of A. Due to the constraints on u we have

$$u^\top A^\dagger u = u^\top Q \Lambda^\dagger Q^\top u = u^\top Q_k \Lambda_k^\dagger Q_k^\top u = u^\top B^\dagger u,$$

where $B = \sum_{i=n-k+1}^n \lambda_i q_i q_i^\top$. Namely, if u lies in the column space spanned by the top k eigenvectors of A then it is equivalent to substitute A with B, which is the truncated version of A at top k eigenvalues and corresponding eigenvectors.

Another key observation is that, given that u lies only in the space spanned by $\{q_i\}_{i=n-k+1}^n$, the Krylov space starting at u becomes

$$\text{Span}\{u, Au, A^2u, \ldots\} = \text{Span}\{u, Bu, B^2u, \ldots, B^{k-1}u\} \tag{3.5.1}$$

This indicates that Lanczos iteration starting at matrix A and vector u will finish constructing the corresponding Krylov space after the k-th iteration. Thus under this condition, Alg. 3

will run at most k iterations and then stop. At that time, the eigenvalues of J_k are exactly the eigenvalues of B, thus they are exactly $\{\lambda_i\}_{i=n-k+1}^{n}$ of A. Using similar proof as in Lemma 18, we can obtain the following generalized exactness result.

Corollary 32 (Generalized Exactness). *g_k, g_k^{rr} and g_k^{lr} are exact for $u^\top A^\dagger u = u^\top B^\dagger u$, namely*

$$g_k = g_k^{rr} = g_k^{lr} = u^\top A^\dagger u = u^\top B^\dagger u.$$

The monotonicity and the relations between bounds given by various Gauss-type quadratures will still be the same as in the original case in Sec. 3.3, but the original convergence rate cannot apply in this case because now we probably have $\lambda_{\min}(B) = 0$, making κ undefined. This crash of convergence rate results from the crash of the convergence of the corresponding conjugate gradient algorithm for solving $Ax = u$. However, by looking at the proof of, e.g., [164], and by noting that $\lambda_1(B) = \ldots = \lambda_{n-k}(B) = 0$, with a slight modification of the proof we actually obtain the bound

$$\|\varepsilon^i\|_A^2 \leq \min_{P_i} \max_{\lambda \in \{\lambda_i\}_{i=n-k+1}^{n}} [P_i(\lambda)]^2 \|\varepsilon^0\|_A^2,$$

where P_i is a polynomial of order i. By using properties of Chebyshev polynomials and following the original proof (e.g., [82] or [164]) we obtain the following lemma for conjugate gradient.

Lemma 33. *Let ε^k be as before (for conjugate gradient). Then,*

$$\|\varepsilon^k\|_A \leq 2\Big(\frac{\sqrt{\kappa'}-1}{\sqrt{\kappa'}+1}\Big)^k \|\varepsilon_0\|_A, \quad \textit{where } \kappa' := \lambda_n/\lambda_{n-k+1}.$$

Following this new convergence rate and connections between conjugate gradient, Lanczos iterations and Gauss quadrature mentioned in Sec. 3.3, we have the following convergence bounds.

Corollary 34 (Convergence Rate for Special Case). *Under the above assumptions on A and u, due to the connection Between Gauss quadrature, Lanczos algorithm and Conjugate*

Gradient, the relative convergence rates of g_i, g_i^{rr}, g_i^{lr} and g_i^{lo} are given by

$$\frac{g_k - g_i}{g_k} \leq 2\Big(\frac{\sqrt{\kappa'}-1}{\sqrt{\kappa'}+1}\Big)^i; \qquad \frac{g_k - g_i^{rr}}{g_k} \leq 2\Big(\frac{\sqrt{\kappa'}-1}{\sqrt{\kappa'}+1}\Big)^i;$$
$$\frac{g_i^{lr} - g_k}{g_k} \leq 2\kappa'_m\Big(\frac{\sqrt{\kappa'}-1}{\sqrt{\kappa'}+1}\Big)^i; \qquad \frac{g_i^{lo} - g_k}{g_k} \leq 2\kappa'_m\Big(\frac{\sqrt{\kappa'}-1}{\sqrt{\kappa'}+1}\Big)^i$$

where $\kappa'_m = \lambda_n/\lambda'_{\min}$ and $0 < \lambda'_{\min} < \lambda_{n-k+1}$ is a lowerbound for nonzero eigenvalues of B.

3.6 Algorithmic Results and Efficient (k-)DPP Sampling

Our theoretical results show that Gauss-Radau quadrature provides good lower and upper bounds to BIFs. More importantly, these bounds get iteratively tighter at a linear rate, finally becoming exact. However, in many applications motivating our work (see Sec. 3.1.3), we do not need exact values of BIFs; bounds that are tight enough suffice for the algorithms to proceed. As a result, all these applications benefit from our theoretical results that provide iteratively tighter bounds. This idea translates into a *retrospective* framework for accelerating methods whose progress relies on knowing an interval containing the BIF. Whenever the algorithm takes a step (*transition*) that depends on a BIF (e.g., as in the next section, a state transition in a sampler if the BIF exceeds a certain threshold), we compute rough bounds on its value. If the bounds suffice to take the critical decision (e.g., decide the comparison), then we stop the quadrature. If they do not suffice, we take one or more additional iterations of quadrature to tighten the bound. Alg. 4 makes this idea explicit. We illustrate our framework by accelerating Markov chain sampling for (k-)DPPs.

3.6.1 Retrospective Markov Chain (k-)DPP

First, we use our framework to accelerate iterative samplers for Determinantal Point Processes. Specifically, we discuss MH sampling [100]; the variant for Gibbs sampling follows analogously.

The key insight is that all state transitions of the Markov chain rely on a comparison

Algorithm 4 Efficient Retrospective Framework

Require: Algorithm with transitions that depend on BIFs
  while algorithm not yet done **do**
    while no transition request for values of a BIF **do**
      proceed with the original algorithm
    end while
    if exist transition request for values of a BIF **then**
      while bounds on the BIF not tight enough to make the transition **do**
        Retrospectively run one more iteration of left and(or) right Gauss-Radau to obtain tighter bounds.
      end while
      Make the correct transition with bounds
    end if
  end while

between a scalar p and a quantity involving the bilinear inverse form. Given the current set S, assume we propose to add element u to S. The probability of transitioning to state $S \cup \{u\}$ is $q = \min\{1, L_{u,u} - L_{u,S}L_{S,S}^{-1}L_{S,u}\}$. To decide whether to accept this transition, we sample $p \sim (0, 1)$ uniformly at random; if $p < q$ then we accept the transition, otherwise we remain at S. Hence, we need to compute q just accurately enough to decide whether $p < q$. To do so, we can use the aforementioned lower and upper bounds on $L_{u,S}L_{S,S}^{-1}L_{S,u}$.

Let s_i and t_i be lower and upper bounds for this BIF in the i-th iteration of Gauss quadrature. If $p \leq L_{u,u} - t_i$, then we can safely accept the transition, if $p \geq L_{u,u} - s_i$, then we can safely reject the transition. Only if $L_{u,u} - t_i < p < L_{u,u} - s_i$, we cannot make a decision yet, and therefore retrospectively perform one more iteration of Gauss quadrature to obtain tighter upper and lower bounds s_{i+1} and t_{i+1}. We continue until the bounds are sharp enough to safely decide whether to make the transition. Note that in each iteration we make an exact decision without approximation error, and hence the resulting algorithm is an exact Markov chain for DPP. The algorithm is shown in Alg. 5. In each iteration, it calls Alg. 6, which uses step-wise lazy Gauss quadrature for deciding the comparison, while stopping as early as possible.

If we condition the DPP on observing a set of a fixed cardinality k, we obtain a k-DPP. The MH sampler for this process is similar, but a state transition corresponds to swapping two elements (adding u and removing v at the same time). Assume the current set is $S = S' \cup \{v\}$. If we propose to delete v and add u to S', then the corresponding transition

Algorithm 5 Gauss-DPP(L)

Require: L: DPP kernel; $\mathcal{V} = [n]$ the ground set
Ensure: S sampled from exact DPP(L)
  Randomly Initialize $S \subseteq \mathcal{V}$
  while chain not mixed **do**
    Pick $y \in \mathcal{V}$, $p \in (0, 1)$ uniformly randomly
    if $y \in S$ **then**
      $S' = S \backslash \{y\}$
      Compute bounds $\lambda_{\min}$, $\lambda_{\max}$ on the spectrum of $L_{S',S'}$
      if DPPJUDGE($L_{yy} - \frac{1}{p}$, $L_{S',y}$, $L_{S',S'}$, $\lambda_{\min}$, $\lambda_{\max}$) **then**
        $S = S'$
      end if
    else
      $S' = S \cup \{y\}$
      Compute bounds $\lambda_{\min}$, $\lambda_{\max}$ on the spectrum of L_S
      if not DPPJUDGE($L_{yy} - p$, $L_{S,y}$, L_S, $\lambda_{\min}$, $\lambda_{\max}$) **then**
        $S = S'$
      end if
    end if
  end while

Algorithm 6 DPPJUDGE($t, u, A, \lambda_{\min}, \lambda_{\max}$)

Require: t the target value; vector u, matrix A; lower and upper bounds $\lambda_{\min}$ and $\lambda_{\max}$ on the spectrum of A
Ensure: Return **true** if $t < u^\top A^{-1} u$, **false** otherwise
  while true do
    Run one Gauss-Radau iteration to get g^{rr} and g^{lr} for $u^\top A^{-1} u$.
    if $t < g^{\text{rr}}$ **then**
      return true
    else if $t \geq g^{\text{lr}}$ **then**
      return false
    end if
    $i = i + 1$
  end while

probability is

$$q = \min\left\{1, \frac{L_{u,u} - L_{u,S'} L_{S',S'}^{-1} L_{S',u}}{L_{v,v} - L_{v,S'} L_{S',S'}^{-1} L_{S',v}}\right\}. \tag{3.6.1}$$

Again, we sample $p \sim \text{Unif}(0,1)$, but now we must compute two quantities, and hence two sets of lower and upper bounds: s_i^u, t_i^u for $L_{u,S'} L_{S',S'}^{-1} L_{S',u}$ in the i-th Gauss quadrature iteration, and s_j^v, t_j^v for $L_{v,S'} L_{S',S'}^{-1} L_{S',v}$ in the j-th Gauss quadrature iteration. Then if we have $p \leq \frac{L_{u,u} - t_i^u}{L_{v,v} - s_j^v}$, we can safely accept the transition; and if $p \geq \frac{L_{u,u} - s_i^u}{L_{v,v} - t_j^v}$ we can safely reject the transition; otherwise, we tighten the bounds via additional Gauss-Radau iterations.

Refinements. We could perform one iteration for both u and v, but it may be that one set of bounds is already sufficiently tight, while the other is loose. A straightforward idea would be to judge the tightness of the lower and upper bounds by their difference (gap) $t_i - s_i$, and decide accordingly which quadrature to iterate further.

But the bounds for u and v are not symmetric and contribute differently to the transition decision. In essence, we need to judge the relation between p and $\frac{L_{u,u} - L_{u,S'} L_{S',S'}^{-1} L_{S',u}}{L_{v,v} - L_{v,S'} L_{S',S'}^{-1} L_{S',v}}$, or, equivalently, the relation between $pL_{v,v} - L_{u,u}$ and $pL_{v,S'} L_{S',S'}^{-1} L_{S',v} - L_{u,S'} L_{S',S'}^{-1} L_{S',u}$. Since the left hand side is "easy", the essential part is the right hand side. Assuming that in practice the impact is larger when the gap is larger, we tighten the bounds for $L_{v,S} L_{S,S}^{-1} L_{S,v}$ if $p(t_j^v - s_j^v) > (t_i^u - s_i^u)$, and otherwise tighen the bounds for $L_{u,S} L_{S,S}^{-1} L_{S,u}$. Details of the final algorithm with this refinement are shown in Alg. 7 and Alg. 8.

Algorithm 7 Gauss-k-DPP(L, k)

Require: L the kernel matrix we want to sample DPP from, k the size of subset and $\mathcal{V} = [n]$ the ground set
Ensure: S sampled from exact k-DPP(L) where $|S| = k$
  Randomly Initialize $S \subseteq \mathcal{V}$ where $|S| = k$
  while not mixed **do**
    Pick $v \in S$ and $u \in \mathcal{V} \backslash S$ uniformly randomly
    Pick $p \in (0,1)$ uniformly randomly
    $S' = S \backslash \{v\}$
    Get lower and upper bounds $\lambda_{\min}$, $\lambda_{\max}$ of the spectrum of $L_{S',S'}$
    if k-DPP-JudgeGauss($pL_{v,v} - L_{u,u}, p, L_{S',u}, L_{S',v}, \lambda_{\min}, \lambda_{\max}$) = *True* **then**
      $S = S' \cup \{u\}$
    end if
  end while

Algorithm 8 k-DPP-JudgeGauss$(t, p, u, v, A, \lambda_{\min}, \lambda_{\max})$

Require: t the target value, p the scaling factor, u, v and A the corresponding vectors and matrix, $\lambda_{\min}$ and $\lambda_{\max}$ lower and upper bounds for the spectrum of A
Ensure: Return *True* if $t < p(v^\top A^{-1} v) - u^\top A^{-1} u$, *False* if otherwise
  $u_{-1} = 0$, $u_0 = u/\|u\|$, $i^u = 1$, $\beta_0^u = 0$, $d^u = \infty$
  $v_{-1} = 0$, $v_0 = v/\|v\|$, $i^v = 1$, $\beta_0^v = 0$, $d^v = \infty$
  while *True* **do**
    if $d^u > pd^v$ **then**
      Run one more iteration of Gauss-Radau on $u^\top A^{-1} u$ to get tighter $(g^{\text{lr}})^u$ and $(g^{\text{rr}})^u$
      $d^u = (g^{\text{lr}})^u - (g^{\text{rr}})^u$
    else
      Run one more iteration of Gauss-Radau on $v^\top A^{-1} v$ to get tighter $(g^{\text{lr}})^v$ and $(g^{\text{rr}})^v$
      $d^v = (g^{\text{lr}})^v - (g^{\text{rr}})^v$
    end if
    if $t < p\|v\|^2 (g^{\text{rr}})^v - \|u\|^2 (g^{\text{lr}})^u$ **then**
      Return *True*
    else if $t \geq p\|v\|^2 (g^{\text{lr}})^v - \|u\|^2 (g^{\text{rr}})^u$ **then**
      Return *False*
    end if
  end while

3.6.2 Empirical Evidence

We perform experiments on both synthetic and real-world datasets to test the impact of our retrospective quadrature framework in applications. We focus on (k-)DPP sampling.

Synthetic Datasets

We generate small sparse matrices using methods similar to Sec. 3.3.4. We generate 5000×5000 matrices and vary the density of the matrices from 10^{-3} to 10^{-1}. The running time and speedup are shown in Fig. 3-2.

The results suggest that our framework greatly accelerates DPP sampling. The speedups are particularly pronounced for sparse matrices. As the matrices become very sparse, the original algorithms benefit from sparsity too, and the difference shrinks a little.

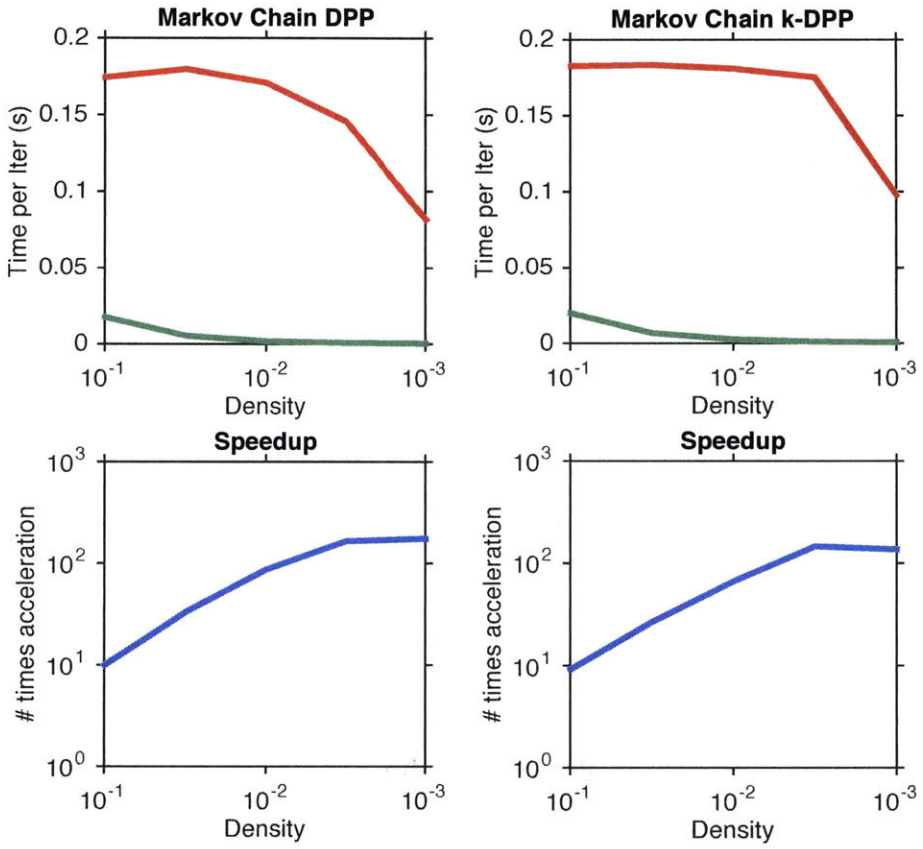


Figure 3-2: Running times (top) and corresponding speedup (bottom) on synthetic data. (k-)DPP is initialized with random subsets of size $n/3$ and corresponding running times are averaged over 1,000 iterations of the chain. All results are averaged over 3 runs of experiments.

Data	Dimension	nnz	Density(%)
Abalone	4,177	144,553	0.83
Wine	4,898	2,659,910	11.09
GR	5,242	34,209	0.12
HEP	9,877	61,821	0.0634
Epinions	75,879	518,231	0.009
Slashdot	82,168	959,454	0.014

Table 3.1: Data. For all datasets we add an 1E-3 times identity matrix to ensure positive definiteness.

Real Datasets

We further test our framework on real-world datasets of varying sizes. We selected 6 datasets, four of them are of small/medium sizes and two are large. The four small/medium-sized datasets are used in [79]. The first two of small/medium-sized datasets, Abalone and Wine[2], are popular datasets for regression, and we construct sparse kernel matrices with an RBF kernel. We set the bandwidth parameter for Abalone as $\sigma = 0.15$ and that for Wine as $\sigma = 1$ and the cut-off parameter as 3σ for both datasets, as in [79]. The other two small/medium-sized datasets are GR (arXiv High Energy Physics collaboration graph) and HEP (arXiv General Relativity collaboration graph), where the kernel matrices are Laplacian matrices. The final two large datasets datasets are Epinions (Who-trusts-whom network of Epinions) and Slashdot (Slashdot social network from Feb. 2009) [3] with large Laplacian matrices. Dataset statistics are shown in Tab. 3.1.

The running times are shown in Tab. 3.2. The results suggest that the quadratures significantly accelerate (k-)DPP sampling even on real data. Our algorithms are lead to speedups up to a thousand times.

To our knowledge, these results are the first time to run (k-)DPP for information gain on such large datasets.

[2]Available at `http://archive.ics.uci.edu/ml/`.
[3]Available at `https://snap.stanford.edu/data/`.

	Abalone		Wine		GR	
DPP	9.6E-3	1x	8.5E-2	1x	9.3E-3	1x
	5.4E-4	**17.8x**	5.9E-3	**14.4x**	4.3E-4	**21.6x**
k-DPP	1.4E-2	1x	0.15	1x	1.7E-2	1x
	7.3E-4	**19.2x**	1.1E-2	**13.6x**	7.3E-4	**23.3x**
	HEP		Epinions		Slashdot	
DPP	6.5E-2	1x	1.46	1x	5.85	1x
	5.9E-4	**110.2x**	3.7E-3	**394.6x**	7.1E-3	**823.9x**
k-DPP	0.13	1x	2.40	1x	11.83	1x
	9.2E-4	**141.3x**	4.9E-3	**489.8x**	1E-2	**1183x**

Table 3.2: Running time and speedup for (k-)DPP. For results on each dataset (occupying two columns), the first column shows the running time (in seconds) and the second column shows the speedup. For each algorithm (occupying two rows), the first row shows results from the original algorithm and the second row shows results from algorithms using our framework.

3.7 Numerical details

Instability. As seen in Alg. 3, the quadrature algorithm is built upon Lanczos iterations. Although in theory Lanczos iterations construct a set of orthogonal Lanczos vectors, in practice the constructed vectors usually lose orthogonality after some iterations due to rounding errors. One way to deal with this problem is to reorthogonalize the vectors, either completely at each iteration or selectively [147]. Also, an equivalent Lanczos iteration proposed in [146] which uses a different expression to improve local orthogonality. Further discussion on numerical stability of the method lies beyond the scope of this thesis.

Preconditioning. For Gauss quadrature on $u^\top A^{-1}u$, the convergence rate of bounds is dependent on the condition number of A. We can use preconditioning techniques to get a well-conditioned submatrix and proceed with that. Concretely, observe that for non-singular C,

$$u^\top A^{-1}u = u^\top C^\top C^{-\top} A^{-1} C^{-1} C u = (Cu)(CAC^\top)^{-1}(Cu).$$

Thus, if $CAC^\top$ is well-conditioned, we can use it with the vector Cu in Gauss quadrature.

There exists various ways to obtain good preconditioners for an SPD matrix. A simple choice is to use $C = [\text{diag}(A)]^{-1/2}$. There also exists methods for efficiently constructing sparse inverse matrix [25]. If L happens to be an SDD matrix, we can use techniques introduced in [44] to construct an approximate sparse inverse in near linear time.

3.8 Summary

In this chapter, we present a general and powerful retrospective computational framework for algorithms including Markov chain (k-)DPP that rely on computations of bilinear inverse forms. The framework is based on Gauss quadrature methods, and supported by our new theoretical results. We analyze properties of the various types of Gauss quadratures for approximating the bilinear inverse forms and show that all bounds are monotonically becoming tighter with the number of iterations; those given by Gauss-Radau are superior to those obtained from other Gauss-type quadratures; and both lower and upper bounds enjoy a linear convergence rate. We empirically verify the efficiency of our framework and are able to obtain speedups of up to thousand times for Markov chain (k-)DPP sampling.

Chapter 4

Sampling from Strongly Rayleigh Measures

While DPP is an instantiation of DIPMs, there is a broader class of probability measures that is diversity-inducing called Strongly Rayleigh (SR) measures. In this chapter, we study the sampling methods for SR measures and derive a provably fast mixing Markov chain that is novel and may be of independent interest. Our results provide the first polynomial guarantee for Markov chain sampling from a general DPP, and more generally from an SR distribution. This result also indicates an efficient sampling method for *Dual Volume Sampling (DVS)*, whose poly-time sampling method remains open since 2013 [13]. Specifically, we prove that DVS is essentially an instantiation of SR, thus a poly-time MCMC sampling method follows. We also show a poly-time exact sampling method for DVS based on matrix computations. Materials in this chapter are based on [114, 115]

4.1 Introduction

Strong Rayleigh (SR) measures were introduced in the landmark paper of [28], who develop a rich theory of negatively associated measures. In particular, we say that a probability

measure π is *negatively associated* if

$$\int F d\pi \int G d\pi \geq \int FG d\pi \tag{4.1.1}$$

for F, G increasing functions on $2^{\mathcal{V}}$ with *disjoint* support. This property reflects a "repelling" nature of π, a property that occurs more broadly across probability, combinatorics, physics, and other fields—see [148, 28, 180] and references therein. The negative association property turns out to be quite subtle in general; the class of SR measures captures a strong notion of negative association and provides a framework for analyzing such measures.

Specifically, SR measures are defined via their connection to real stable polynomials [148, 28, 180]. A multivariate polynomial $f \in \mathbb{C}[z]$ where $z \in \mathbb{C}^n$ is called *real stable* if all its coefficients are real and $f(z) \neq 0$ whenever $\mathfrak{Im}(z_i) > 0$ for $1 \leq i \leq n$. A measure is called an *SR measure* if its multivariate generating polynomial,

$$f_\pi(z) := \sum_{S \subseteq \mathcal{V}} \pi(S) \prod_{i \in S} z_i, \tag{4.1.2}$$

is real stable. It is known (see [28, pg. 523]) that the class of SR measures is exponentially larger than the class of determinantal measures.

4.1.1 SR Instantiations

Strongly Rayleigh measures have been underlying recent progress in approximation algorithms [75, 7, 50, 113], graph sparsification [68, 171], extensions to the Kadison-Singer problem [6], finite extensions to free probability [131], and concentration of measure results [150]. There has been many notable examples of SR measures widely studied in machine learning and theoretical computer science and we list them as follows.

Determinantal Point Processes. A Determinantal Point Process (DPP) is a measure over subsets given by the principal minors of a positive semidefinite matrix $L \in \mathbb{R}^{n \times n}$. Its probabilities satisfy

$$\pi(S) \propto \det(L_{S,S}), \tag{4.1.3}$$

DPPs arise in random matrix theory, combinatorics, machine learning, matrix approximations, and many other areas; see e.g., [126, 123, 124, 41, 29, 170, 107, 94, 31, 30, 118].

(Weighted) regular and balanced matroids. The uniform distribution over the bases of certain matroids (regular matroids and balanced matroids [62, 150]) is SR, most notably, the uniform distribution over spanning trees in a graph. Here, spanning trees are viewed as subsets of edges, and the distribution is over subsets of edges.

Product measures / Bernoullis conditioned on their sum. Assume there is a weight $q_i \in [0,1]$ for each element $i \in V$. The product measure $\pi(S) = \prod_{i \in S} q_i \prod_{j \notin S}(1-q_j)$ is SR, as is its conditioning on sets of a specific cardinality k, i.e., $\pi'(S) = \pi(S \mid |S| = k)$ or $\pi'(S) = 0$ if $|S| \neq k$, and $\pi'(S) \propto \pi(S)$ otherwise.

Uniform distribution on certain matroid (regular matroid and balanced matroid [62, 150]) base is SR, most notably, the uniform distribution over spanning trees in a graph. Here, spanning trees are viewed as subsets of edges, and the distribution is over subsets of edges.

4.1.2 Sampling using MCMC

We sample from π using MCMC, i.e., we run a Markov Chain with state space $2^{\mathcal{V}}$. All our chains are ergodic. The *mixing time* of the chain indicates the number of iterations t that we must perform (after starting from an arbitrary set $S_0 \subseteq \mathcal{V}$) before we can consider S_t as a valid sample from π. Formally, if $\delta_{S_0}(t)$ is the total variation distance between the distribution of S_t and π after t steps, then $\tau_{S_0}(\varepsilon) = \min\{t : \delta_{S_0}(t') \leq \varepsilon,\ \forall t' \geq t\}$ is the mixing time to sample from a distribution ϵ-close to $\pi_{\mathcal{C}}$ in terms of total variation distance. We say that the chain mixes fast if τ_{S_0} is polynomial in n. The mixing time can be bounded in terms of the eigenvalues of the transition matrix, as the following classic result shows:

Theorem 35 (Mixing Time [52]). *Let λ_i be the eigenvalues of the transition matrix, and $\lambda_{\max} = \max\{\lambda_2, |\lambda_n|\} < 1$. Then, the mixing time starting from an initial set $S_0 \subseteq \mathcal{V}$ is bounded as*

$$\tau_{S_0}(\varepsilon) \leq (1-\lambda_{\max})^{-1}(\log \pi_{\mathcal{C}}(S_0)^{-1} + \log \varepsilon^{-1}).$$

Efficient sampling techniques have been studied for instances of SR distributions. A popular method for sampling from Determinantal Point Processes uses the spectrum of the defining kernel [94]. Generic MCMC samplers can also be derived, for example, previous work used a simple add-delete Metropolis-Hasting chain [100]. Starting with an arbitrary set $S \subseteq \mathcal{V}$, we sample a point $t \in \mathcal{V}$ uniformly at random. If $t \in S$, we remove t with probability $\min\{1, \pi(S \setminus \{t\})/\pi(S)\}$; if $t \notin S$, we add it to S with probability $\min\{1, \pi(S \cup \{t\})/\pi(S)\}$. Algorithm 9 shows the (lazy) Markov chain.

Algorithm 9 Add/delete Markov Chain

Require: SR distribution π
  Initialize $S \subseteq \mathcal{V}$
  while not mixed **do**
    Let $b = 1$ with probability $\frac{1}{2}$
    if $b = 1$ **then**
      Pick $t \in \mathcal{V}$ uniformly at random
      if $t \in S$ **then**
        $S = S \setminus \{t\}$ with probability $\min\{1, \pi(S \setminus \{t\})/\pi(S)\}$
      else
        $S = S \cup \{t\}$ with probability $\min\{1, \pi(S \cup \{t\})/\pi(S)\}$
      end if
    else
      Do nothing
    end if
  end while

The add-delete chain can work well in practice [100], however, it was not shown to be always fast mixing. An *elementary* DPP has non-zero measure only on sets of a fixed cardinality; for such a process (or a process close to it), the chain will stall or mix slowly.

Another special case of SR distributions are *homogeneous* SR measures. These measures are nonzero only for some sets of a fixed cardinality k. Examples include Bernoulli distributions conditioned on cardinality, uniform distributions on the bases of balanced matroids [62], and k-DPPs. A natural MCMC sampler for these processes takes swapping steps: given a current set $S \subseteq \mathcal{V}$, it picks, uniformly at random, points $s \in S$ and $t \notin S$, and swaps them with probability $\min\{1, \pi(S \cup \{t\} \setminus \{s\})/\pi(S)\}$. Algorithm 10 formalizes this procedure. Building upon results in [62], [8] recently showed that the mixing time for the swap sampler for homogeneous SR measures is polynomial in n, k, and $\log(\frac{1}{\epsilon\pi(S_0)})$. These

results are restricted to homogeneous SR measures, and do not hold for general SR measures or SR with various constraints.

Algorithm 10 Exchange Markov Chain

Require: Homogeneous SR distribution π
  Initialize $S \subseteq \mathcal{V}$, $\pi(S) > 0$
  while not mixed **do**
    Let $b = 1$ with probability $\frac{1}{2}$
    if $b = 1$ **then**
      Pick $s \in S$ and $t \notin S$ uniformly randomly
      $S = S \cup \{t\} \backslash \{s\}$ with probability $\min\{1, \pi(S \cup \{t\} \setminus \{s\})/\pi(S)\}$
    else
      Do nothing
    end if
  end while

4.1.3 Other Related work

Recent work in machine learning addresses sampling from distributions with sub- or supermodular F [86, 156] and sampling by optimization [60, 128]. Apart from sampling, other related tracts include work on variational inference for combinatorial distributions [32, 53, 167, 185] and inference for submodular processes [95].

4.2 Sampling from General Strongly Rayleigh Measures

In this section, we consider sampling from general SR measures. We show in particular that SR measures are amenable to efficient Markov chain sampling. Our starting point is the observation of [28] on closure properties of SR measures; of these we use *symmetric homogenization*. Given a distribution π on $2^{\mathcal{V}} = 2^{[n]}$, its symmetric homogenization π_{sh} on $2^{[2n]}$ is

$$\pi_{sh}(S) := \begin{cases} \pi(S \cap [n]) \binom{n}{|S \cap \mathcal{V}|}^{-1} & \text{if } |S| = n; \\ 0 & \text{otherwise.} \end{cases} \tag{4.2.1}$$

If π is SR, so is π_{sh}. We use this property below in our derivation of a fast-mixing chain.

We use here a recent result of [8], who show a Markov chain that mixes rapidly for *homogeneous SR* distributions. These distributions are over all subsets $S \subseteq \mathcal{V}$ of some fixed size $|S| = k$, and hence do not include general DPPs. Concretely, for any k-homogeneous SR distribution $\pi : 2^{\mathcal{V}} \to \mathbb{R}_+$, a Gibbs-exchange sampler has mixing time

$$\tau_{S_0}(\varepsilon) \leq 2k(n-k)(\log \pi(S_0)^{-1} + \log \varepsilon^{-1}).$$

This sampler uniformly samples one item in the current set, and one outside the current set, and swaps them with an appropriate probability. Using these ideas we show how to obtain fast mixing chains for *any* general SR distribution π on $2^{\mathcal{V}}$. First, we construct its symmetric homogenization π_{sh}, and sample from π_{sh} using a Gibbs-exchange sampler. This chain is fast mixing, thus we will efficiently get a sample $T \sim \pi_{sh}$. The corresponding sample for π can be then obtained by computing $S = T \cap \mathcal{V}$. Theorem 36, formally establishes the validity of this idea.

Theorem 36. *If π is SR, then the mixing time of a Exchange Markov Chain sampler for π_{sh} is bounded as*

$$\tau_{S_0}(\varepsilon) \leq 2n^2\Big(\log \binom{n}{|S_0|} + \log(\pi(S_0))^{-1} + \log \varepsilon^{-1}\Big). \tag{4.2.2}$$

Proof. Given a general SR measure π, we construct its symmetric homogenization π_{sh} as in Eq. 4.2.1. By closure property of SR we know that π_{sh} is homogeneous SR. Then it follows from [8] that the base exchange Markov chain has its mixing time bounded as

$$\begin{aligned}(\tau_{sh})_{R_0}(\varepsilon) &\leq 2n^2(\log(\pi_{sh}(R_0))^{-1} + \log \varepsilon^{-1}) \\ &= 2n^2\left(\log \binom{n}{|S_0|} + \log(\pi_C(S_0))^{-1} + \log \varepsilon^{-1}\right),\end{aligned}$$

where $R_0 \subseteq [2N]$, $|R_0| = N$ and $S_0 = R_0 \cap \mathcal{V}$.

We construct a base exchange Markov chain on $2n$ variables where we maintain a set $|R| = n$. In each iteration and with probability 0.5 we choose uniformly $s \in R$ and $t \in [2n] \backslash R$ and switch them with certain transition probabilities. Let $S = R \cap \mathcal{V}$, $T = \mathcal{V} \backslash R$,

there are in total four possibilities for locations of s and t:

- With probability $\frac{|S|(n-|S|)}{2n^2}$, $s \in S$ and $t \in T$, and we switch assignment of s and t with probability $\min\{1, \frac{\pi_{sh}(R\cup\{t\}\backslash\{s\})}{\pi_{sh}(R)}\} = \min\{1, \frac{\pi_C(S\cup\{t\}\backslash\{s\})}{\pi_C(S)}\}$. This is equivalent to switching elements between S and T;

- With probability $\frac{|S|(n-|S|)}{2n^2}$, $s \notin S$ and $t \notin T$, and switch with probability $\min\{1, \frac{\pi_C(S\cup\{t\})}{\pi_C(S)} \times \frac{|S|+1}{n-|S|}\}$. This is equivalent to doing nothing to S;

- With probability $\frac{|S|^2}{2n^2}$, $s \in S$ and $t \notin T$, and we switch with probability $\min\{1, \frac{\pi_C(S\backslash\{s\})}{\pi_C(S)} \times \frac{|S|}{n-|S|+1}\}$. This is equivalent to deleting elements from S;

- With probability $\frac{(n-|S|)^2}{2n^2}$, $s \notin S$ and $t \in T$, and switch with probability $\min\{1, \frac{\pi_C(S\cup\{t\})}{\pi_C(S)} \times \frac{|S|+1}{n-|S|}\}$. This is equivalent to adding elements to S.

Constructing the chain in the same manner but only maintaining $S = R \cap [n]$ will result in Algo. 11, while the mixing time stays unchanged.

□

For Theorem 36 we may choose the initial set such that S_0 makes the first term in the sum logarithmic in N ($S_0 = R_0 \cap \mathcal{V}$ in Algorithm 11).

Algorithm 11 Markov Chain for Strongly Rayleigh Distributions

Require: SR distribution π
  Initialize $R \subseteq [2n]$ where $|R| = n$ and take $S = R \cap \mathcal{V}$, $T = \mathcal{V}\backslash R$
  while not mixed **do**
    Draw $q \sim$ Unif $[0, 1]$
    Draw $t \in T$ and $s \in S$ uniformly at random
    if $q \in [0, \frac{(n-|S|)^2}{2n^2})$ **then**
      $S = S \cup \{t\}$ with probability $\min\{1, \frac{\pi(S\cup\{t\})}{\pi(S)} \times \frac{|S|+1}{n-|S|}\}$ ▷ Add t
    else if $q \in [\frac{(n-|S|)^2}{2n^2}, \frac{n-|S|}{2n})$ **then**
      $S = S \cup \{t\}\backslash\{s\}$ with probability $\min\{1, \frac{\pi(S\cup\{t\}\backslash\{s\})}{\pi(S)}\}$ ▷ Exchange s with t
    else if $q \in [\frac{n-|S|}{2n}, \frac{|S|^2+n(n-|S|)}{2n^2})$ **then**
      $S = S\backslash\{s\}$ with probability $\min\{1, \frac{\pi(S\backslash\{s\})}{\pi(S)} \times \frac{|S|}{n-|S|+1}\}$ ▷ Delete s
    else
      Do nothing
    end if
  end while

Efficient Implementation. Directly running a chain to sample n items from a (doubled) set of size $2n$ adds some computational overhead. Hence, we construct an equivalent, more space-efficient chain (Algorithm 11) on the initial ground set $\mathcal{V} = [n]$ that only maintains $S \subseteq \mathcal{V}$. Interestingly, this sampler is a mixture of add-delete and Gibbs-exchange samplers. This combination makes sense intuitively, too: add-delete moves (shown in Algorithm 14) are needed since the exchange sampler (shown in Algorithm 10) cannot change the cardinality of S. But a pure add-delete chain can stall if the sets concentrate around a fixed cardinality (low probability of a larger or smaller set). Exchange moves will not suffer the same high rejection rates. The key idea underlying Algorithm 11 is that the elements in $\{n+1, \ldots, 2n\}$ are indistinguishable, so it suffices to maintain merely the cardinality of the currently selected subset instead of all its indices.

Corollary 37. *The bound* (4.2.2) *applies to the mixing time of Algorithm 11.*

Remarks. By assuming π is SR, we obtain a clean bound for fast mixing. Compared to the bound in [86], our result avoids the somewhat opaque factor $\exp(\beta\zeta_F)$ that depends on F. This advantage comes with a cost of an additive factor, which can be made small via careful initialization, e.g., by choosing S_0 up to a constant size.

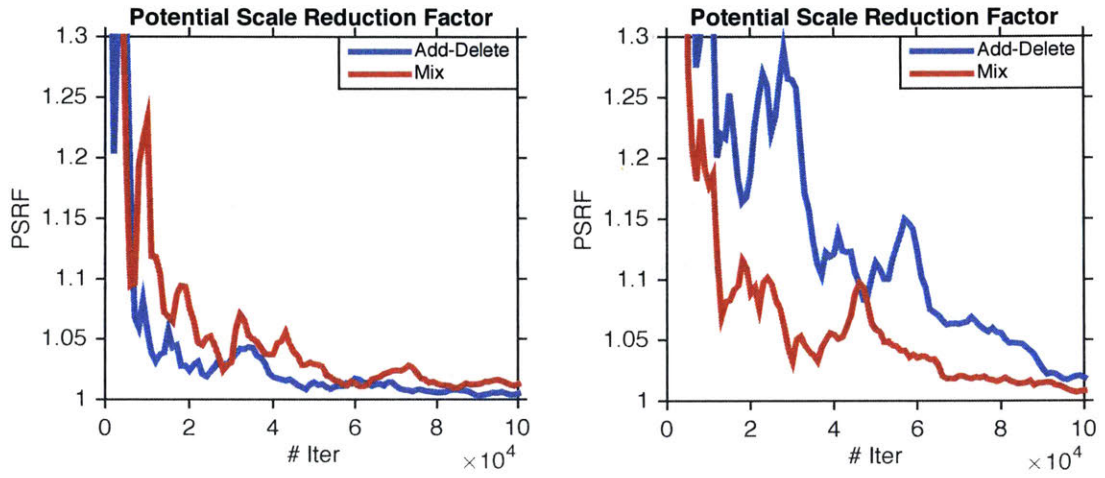
In certain cases, the above chain may mix slower in practice than a pure add-delete chain that was used in previous works [100, 86], since its probability of doing nothing is higher. In other cases, it mixes much faster than the pure add-delete chain; we observe both phenomena in our subsequent experiments. Contrary to a simple add-delete chain, it is *guaranteed* to mix well.

4.2.1 Experiments

We empirically study how fast our sampler on strongly Rayleigh distribution converges. We compare the chain in Algorithm 11 (`Mix`) against a simple add-delete chain (`Add-Delete`). We use a DPP on Ailerons data[1] of size 200, and the corresponding PSRF is shown in Fig. 4-1a. We observe that `Mix` converges slightly slower than `Add-Delete` since it is lazier. However, the Add-Delete chain does not always mix fast. Fig. 4-1b illustrates a different

[1] `http://www.dcc.fc.up.pt/657~ltorgo/Regression/DataSets.html`

setting, where we modify the eigenspectrum of the kernel matrix: the first 100 eigenvalues are 500 and others 1/500. Such a kernel corresponds to almost an elementary DPP, where the size of the observed subsets sharply concentrates around 100. Here, `Add-Delete` moves very slowly. `Mix`, in contrast, has the ability of exchanging elements and thus converges way faster than `Add-Delete`.



(a) (b)

Figure 4-1: (a) Convergence of marginal and conditional probabilities by DPP on uniform matroid, (b,c) comparison between add-delete chain (Algorithm 9) and projection chain (Algorithm 11) for two instances: slowly decaying spectrum and sharp step in the spectrum.

4.3 Dual Volume Sampling

From this section on we will consider an instantiation of DIPM that turns out to be SR measures, thus fast mixing MCMC follows. In real world, a variety of applications share the core task of selecting a subset of columns from a short, wide matrix A with n rows and $m > n$ columns. The criteria for selecting these columns typically aim at preserving information about the span of A while generating a well-conditioned submatrix. Classical and recent examples include experimental design, where we select observations or experiments [151]; preconditioning for solving linear systems and constructing low-stretch spanning trees (here A is a version of the node-edge incidence matrix and we select edges in a graph) [13, 10];

matrix approximation [34, 33, 91]; feature selection in k-means clustering [35, 36]; sensor selection [99] and graph signal processing [43, 179].

The distribution we study holds promise for all of these applications. It relies on sampling columns of A according to a probability distribution defined over its submatrices: the probability of selecting a set S of k columns from A, with $n \leq k \leq m$, is

$$\pi(S; A) \propto \det(A_S A_S^\top), \tag{4.3.1}$$

where A_S is the submatrix consisting of the selected columns. This distribution is reminiscent of *volume sampling*, where $k < n$ columns are selected with probability proportional to the determinant $\det(A_S^\top A_S)$ of a $k \times k$ matrix, i.e., the squared volume of the parallelepiped spanned by the selected columns. (Volume sampling does *not* apply to $k > n$ as the involved determinants vanish.) In contrast, $\pi(S; A)$ uses the determinant of an $n \times n$ matrix and uses the volume spanned by the *rows* formed by the selected columns. Hence we refer to $\pi(S; A)$-sampling as *dual volume sampling (DVS)*.

Despite the ostensible similarity between volume sampling and DVS, and despite the many practical implications of DVS outlined below, efficient algorithms for DVS are not known and were raised as open questions in [13]. In the subsequent, we make the following contributions:

- We establish that $\pi(S; A)$ is a *Strongly Rayleigh* measure [28], a remarkable property that captures a specific form of negative dependence. Our proof relies on the theory of real stable polynomials, and the ensuing result implies a provably fast-mixing, practical MCMC sampler. Moreover, this result implies concentration properties for dual volume sampling.

- We develop polynomial-time randomized sampling algorithms and their derandomization for DVS. Surprisingly, our proofs require only elementary (but involved) matrix manipulations.

In parallel with our work, [48] also proposed a polynomial time sampling algorithm that works efficiently in practice. Our work goes on to further uncover the hitherto unknown

"Strong Rayleigh" property of DVS, which has important consequences, including those noted above.

4.3.1 Connections and implications.

The selection of $k \geq n$ columns from a short and wide matrix has many applications. Our algorithms for DVS hence have several implications and connections; we note a few below.

Experimental design. The theory of optimal experiment design explores several criteria for selecting the set of columns (experiments) S. Popular choices are

$$S \in \operatorname{argmin}_{S \subseteq \{1,\dots,m\}} J(A_S), \text{ with}$$

$$J(A_S) = \|A_S^\dagger\|_F = \|(A_S A_S^\top)^{-1}\|_F \text{ (A-optimal design)}, \tag{4.3.2}$$

$$J(A_S) = \|A_S^\dagger\|_2 \text{ (E-optimal design)}, \tag{4.3.3}$$

$$J(A_S) = -\log\det(A_S A_S^\top) \text{ (D-optimal design)}. \tag{4.3.4}$$

Here, $A^\dagger$ denotes the Moore-Penrose pseudoinverse of A, and the minimization ranges over all S such that A_S has full row rank n. A-optimal design, for instance, is statistically optimal for linear regression [151].

Finding an optimal solution for these design problems is NP-hard; and most discrete algorithms use local search [137]. [13, Theorem 3.1] show that dual volume sampling yields an approximation guarantee for both A- and E-optimal design: if S is sampled from DVS $\pi(S; A)$, then

$$\mathbb{E}\left[\|A_S^\dagger\|_F^2\right] \leq \frac{m-n+1}{k-n+1}\|A^\dagger\|_F^2; \quad \mathbb{E}\left[\|A_S^\dagger\|_2^2\right] \leq \left(1 + \frac{n(m-k)}{k-n+1}\right)\|A^\dagger\|_2^2. \tag{4.3.5}$$

[13] provide a polynomial time sampling algorithm only for the case $k = n$. Our algorithms achieve the bound (4.3.5) in expectation, and the derandomization in Section 4.5.3 achieves the bound deterministically. [182] recently (in parallel) achieved approximation bounds for A-optimality via a different algorithm combining convex relaxation and a greedy method. Other methods include leverage score sampling [125] and predictive length sampling [190].

Low-stretch spanning trees and applications. Objectives 4.3.2 also arise in the con-

struction of low-stretch spanning trees, which have important applications in graph sparsification, preconditioning and solving symmetric diagonally dominant (SDD) linear systems [172], among others [59]. In the node-edge incidence matrix $\Pi \in \mathbb{R}^{n \times m}$ of an undirected graph G with n nodes and m edges, the column corresponding to edge (u, v) is $\sqrt{w(u,v)}(e_u - e_v)$. Let $\Pi = U\Sigma Y$ be the SVD of Π with $Y \in \mathbb{R}^{n-1 \times m}$. The stretch of a spanning tree T in G is then given by $St_T(G) = \|Y_T^{-1}\|_F^2$ [13]. In those applications, we hence search for a set of edges with low stretch.

Network controllability. The problem of sampling $k \geq n$ columns in a matrix also arises in network controllability. For example, [188] consider selecting control nodes S (under certain constraints) over time in complex networks to control a linear time-invariant network. After transforming the problem into a column subset selection problem from a short and wide controllability matrix, the objective becomes essentially an E-optimal design problem, for which the authors use greedy heuristics.

4.4 SR Property and Fast Markov Chain Sampling

Next, we investigate DVS more deeply and discover that it possesses a remarkable structural property, namely, the *Strongly Rayleigh (SR)* [28] property. This property has proved remarkably fruitful in a variety of recent contexts, including recent progress in approximation algorithms [75], fast sampling [7, 117], graph sparsification [68, 171], extensions to the Kadison-Singer problem [6], and certain concentration of measure results [150], among others.

4.4.1 Strong Rayleigh Property of DVS

Theorem 38 establishes the SR property for DVS and is the main result of this section. Here and in the following, we use the notation $z^S = \prod_{i \in S} z_i$.

Theorem 38. *Let $A \in \mathbb{R}^{n \times m}$ and $n \leq k \leq m$. Then the multiaffine polynomial*

$$p(z) := \sum_{|S|=k, S \subseteq [m]} \det(A_S A_S^\top) \prod_{i \in S} z_i \quad = \quad \sum_{|S|=k, S \subseteq [m]} \det(A_S A_S^\top) z^S, \tag{4.4.1}$$

is real stable. Consequently, $\pi(S; A)$ is an SR measure.

The proof of Theorem 38 relies on key properties of real stable polynomials and SR measures established in [28]. Essentially, the proof demonstrates that the generating polynomial of $\overline{P}(S_c; A)$ can be obtained by applying a few carefully chosen stability preserving operations to a polynomial that we know to be real stable. Stability, although easily destroyed, is closed under several operations noted in the important proposition below.

Proposition 39 (Prop. 2.1 [28]). *Let $f : \mathbb{C}^m \to \mathbb{C}$ be a stable polynomial. The following properties preserve stability:*

- ***Substitution**: $f(\mu, z_2, \ldots, z_m)$ for $\mu \in \mathbf{R}$;*
- ***Differentiation**: $\partial^S f(z_1, \ldots, z_m)$ for any $S \subseteq [m]$;*
- ***Diagonalization**: $f(z, z, z_3 \ldots, z_m)$ is stable, and hence $f(z, z, \ldots, z)$; and*
- ***Inversion**: $z_1 \cdots z_n f(z_1^{-1}, \ldots, z_n^{-1})$.*

In addition, we need the following two propositions for proving Theorem 38.

Proposition 40 (Prop. 2.4 [27]). *Let B be Hermitian, $z \in \mathbb{C}^m$ and A_i ($1 \leq i \leq m$) be Hermitian semidefinite matrices. Then, the following polynomial is stable:*

$$f(z) := \det(B + \sum\nolimits_i z_i A_i). \tag{4.4.2}$$

Proposition 41. *For $n \leq |S| \leq m$ and $L := A^\top A$, we have $\det(A_S A_S^\top) = e_n(L_{S,S})$.*

Proof. Let $Y = Diag([y_i]_{i=1}^m)$ be a diagonal matrix. Using the Cauchy-Binet identity we have

$$\det(AYA^\top) = \sum\nolimits_{|T|=n, T\subseteq[m]} \det((AY)_{:,T}) \det((A^\top)_{T,:}) = \sum\nolimits_{|T|=n, T\subseteq[m]} \det(A_T^\top A_T) y^T.$$

Thus, when $Y = I_S$, the (diagonal) indicator matrix for S, we obtain $AYA^\top = A_S A_S^\top$. Consequently, in the summation above only terms with $T \subseteq S$ survive, yielding

$$\det(A_S A_S^\top) = \sum_{|T|=n, T\subseteq S} \det(A_T^\top A_T) = \sum_{|T|=n, T\subseteq S} \det(L_{T,T}) = e_n(L_{S,S}). \qquad \square$$

We are now ready to sketch the proof of Theorem 38.

Proof. (Theorem 38). Notationally, it is more convenient to prove that the "complement" polynomial $p_c(z) := \sum_{|S|=k, S\subseteq[m]} \det(A_S A_S^\top) z^{S_c}$ is stable; subsequently, an application of Prop. 39-(iv) yields stability of (4.4.1). Using matrix notation $W = Diag(w_1, \ldots, w_m)$, $Z = Diag(z_1, \ldots, z_m)$, our starting stable polynomial (this stability follows from Prop. 40) is

$$h(z, w) := \det(L + W + Z), \quad w \in \mathbb{C}^m,\ z \in \mathbb{C}^m,$$

which can be expanded as

$$h(z, w) = \sum\nolimits_{S\subseteq[m]} \det(W_S + L_S) z^{S_c} = \sum\nolimits_{S\subseteq[m]} \Big(\sum\nolimits_{T\subseteq S} w^{S\backslash T} \det(L_{T,T}) \Big) z^{S_c}.$$

Thus, $h(z, w)$ is real stable in $2m$ variables, indexed below by S and R where $R := S\backslash T$. Instead of the form above, We can sum over $S, R \subseteq [m]$ but then have to constrain the support to the case when $S_c \cap T = \emptyset$ and $S_c \cap R = \emptyset$. In other words, we may write (using Iverson-brackets $[\![\cdot]\!]$)

$$h(z, w) = \sum_{S,R\subseteq[m]} [\![S_c \cap R = \emptyset \wedge S_c \cap T = \emptyset]\!] \det(L_{T,T}) z^{S_c} w^R. \tag{4.4.3}$$

Next, we truncate polynomial (4.4.3) at degree $(m - k) + (k - n) = m - n$ by restricting $|S_c \cup R| = m - n$. By [28, Corollary 4.18] this truncation preserves stability, whence

$$H(z, w) := \sum_{\substack{S,R\subseteq[m] \\ |S_c\cup R|=m-n}} [\![S_c \cap R = \emptyset]\!] \det(L_{S\backslash R, S\backslash R}) z^{S_c} w^R,$$

is also stable. Using Prop. 39-(iii), setting $w_1 = \ldots = w_m = y$ retains stability; thus

$$\begin{aligned} g(z, y) :&= H(z, (\underbrace{y, y, \ldots, y}_{m \text{ times}})) = \sum_{\substack{S,R\subseteq[m] \\ |S_c\cup R|=m-n}} [\![S_c \cap R = \emptyset]\!] \det(L_{S\backslash R, S\backslash R}) z^{S_c} y^{|R|} \\ &= \sum_{S\subseteq[m]} \Big(\sum\nolimits_{|T|=n, T\subseteq S} \det(L_{T,T}) \Big) y^{|S|-|T|} z^{S_c} = \sum_{S\subseteq[m]} e_n(L_{S,S}) y^{|S|-n} z^{S_c}, \end{aligned}$$

is also stable. Next, differentiating $g(z, y)$, $k - n$ times with respect to y and evaluating at 0 preserves stability (Prop. 39-(ii) and (i)). In doing so, only terms corresponding to $|S| = k$ survive, resulting in

$$\frac{\partial^{k-n}}{\partial y^{k-n}} g(z, y)\Big|_{y=0} = (k-n)! \sum_{|S|=k, S\subseteq[m]} e_n(L_{S,S}) z^{S_c} = (k-n)! \sum_{|S|=k, S\subseteq[m]} \det(A_S A_S^\top) z^{S_c},$$

which is just $p_c(z)$ (up to a constant); here, the last equality follows from Prop. 41. This establishes stability of $p_c(z)$ and hence of $p(z)$. Since $p(z)$ is in addition multiaffine, it is the generating polynomial of an SR measure, completing the proof. □

4.4.2 Implications: MCMC

The SR property of $\pi(S; A)$ established in Theorem 38 implies a fast mixing Markov chain for sampling S. The states for the Markov chain are all sets of cardinality k. The chain starts with a randomly-initialized active set S, and in each iteration we swap an element $s^{\text{in}} \in S$ with an element $s^{\text{out}} \notin S$ with a specific probability determined by the probability of the current and proposed set. The stationary distribution of this chain is the one induced by DVS, by a simple detailed-balance argument. The chain is shown in Algorithm 12.

Algorithm 12 Markov Chain for Dual Volume Sampling

Input: $A \in \mathbb{R}^{n\times m}$ the matrix of interest, k the target cardinality, T the number of steps
Output: $S \sim \pi(S; A)$
Initialize $S \subseteq [m]$ such that $|S| = k$ and $\det(A_S A_S^\top) > 0$
for $i = 1$ to T **do**
  draw $b \in \{0, 1\}$ uniformly
  if $b = 1$ **then**
    Pick $s^{\text{in}} \in S$ and $s^{\text{out}} \in [m]\backslash S$ uniformly randomly
    $q(s^{\text{in}}, s^{\text{out}}, S) \leftarrow \min\left\{1, \det(A_{S\cup\{s^{\text{out}}\}\backslash\{s^{\text{in}}\}} A_{S\cup\{s^{\text{out}}\}\backslash\{s^{\text{in}}\}}^\top)/\det(A_S A_S^\top)\right\}$
    $S \leftarrow S \cup \{s^{\text{out}}\}\backslash\{s^{\text{in}}\}$ with probability $q(s^{\text{in}}, s^{\text{out}}, S)$
  end if
end for

The convergence of the Markov chain is measured via its mixing time: The *mixing time* of the chain indicates the number of iterations t that we must perform (starting from S_0) before we can consider S_t as an approximately valid sample from $\pi(S; A)$. Formally, if

$\delta_{S_0}(t)$ is the total variation distance between the distribution of S_t and $\pi(S; A)$ after t steps, then

$$\tau_{S_0}(\varepsilon) := \min\{t : \delta_{S_0}(t') \leq \varepsilon,\ \forall t' \geq t\}$$

is the *mixing time* to sample from a distribution ε-close to $\pi(S; A)$ in terms of total variation distance. We say that the chain mixes fast if τ_{S_0} is polynomial in the problem size.

The fast mixing result for Algorithm 12 is a corollary of Theorem 38 combined with a recent result of [8] on fast-mixing Markov chains for homogeneous SR measures. Theorem 42 states this precisely.

Theorem 42 (Mixing time). *The mixing time of Markov chain shown in Algorithm 12 is given by*

$$\tau_{S_0}(\varepsilon) \leq 2k(m-k)(\log P(S_0; A)^{-1} + \log \varepsilon^{-1}).$$

Proof. Since $\pi(S; A)$ is k-homogeneous SR by Theorem 38, the chain constructed for sampling S following that in [8] mixes in $\tau_{S_0}(\varepsilon) \leq 2k(m-k)(\log P(S_0; A)^{-1} + \log \varepsilon^{-1})$ time. □

Implementation. To implement Algorithm 12 we need to compute the transition probabilities $q(s^{\text{in}}, s^{\text{out}}, S)$. Let $T = S \backslash \{s^{\text{in}}\}$ and assume $r(A_T) = n$. By the matrix determinant lemma we have the acceptance ratio

$$\frac{\det(A_{S\cup\{s^{\text{out}}\}\backslash\{s^{\text{in}}\}} A^{\top}_{S\cup\{s^{\text{out}}\}\backslash\{s^{\text{in}}\}})}{\det(A_S A_S^{\top})} = \frac{(1 + A^{\top}_{\{s^{out}\}}(A_T A_T^{\top})^{-1} A_{\{s^{out}\}})}{(1 + A^{\top}_{\{s^{in}\}}(A_T A_T^{\top})^{-1} A_{\{s^{in}\}})}.$$

Thus, the transition probabilities can be computed in $\mathcal{O}(n^2 k)$ time. Moreover, one can further accelerate this algorithm by using the quadrature techniques of [118] to compute lower and upper bounds on this acceptance ratio to determine early acceptance or rejection of the proposed move.

Initialization. A remaining question is initialization. Since the mixing time involves $\log P(S_0; A)^{-1}$, we need to start with S_0 such that $P(S_0; A)$ is sufficiently bounded away

from 0. We show in Appendix B.6 that by a simple greedy algorithm, we are able to initialize S such that $\log \pi(S; A)^{-1} \geq \log(2^n k! \binom{m}{k}) = \mathcal{O}(k \log m)$, and the resulting running time for Algorithm 12 is $\widetilde{\mathcal{O}}(k^3 n^2 m)$, which is *linear* in the size of data set m and is efficient when k is not too large.

4.4.3 Further implications and connections

Concentration. [150] shows concentration results for strong Rayleigh measures. As a corollary of our Theorem 38 together with their results, we directly obtain tail bounds for DVS.

Algorithms for experimental design. Widely used, classical algorithms for finding an approximate optimal design include Fedorov's exchange algorithm [63] (a greedy local search) and simulated annealing [140]. Both methods start with a random initial set S, and greedily or randomly exchange a column $i \in S$ with a column $j \notin S$. Apart from very expensive running times, they are known to work well in practice [142, 182]. Yet so far there is no theoretical analysis, or a principled way of determining when to stop the greedy search.

Curiously, our MCMC sampler is essentially a randomized version of Fedorov's exchange method. The two methods can be connected by a unified, simulated annealing view, where we define $P^\beta(S; A) \propto \exp\{\log\det(A_S A_S^\top)/\beta\}$ with temperature parameter β. Driving β to zero essentially recovers Fedorov's method, while our results imply fast mixing for $\beta = 1$, together with approximation guarantees. Through this lens, simulated annealing may be viewed as initializing Fedorov's method with the fast-mixing sampler. In practice, we observe that letting $\beta < 1$ improves the approximation results, which opens interesting questions for future work.

4.5 Polynomial-time Dual Volume Sampling

We describe in this section our method to sample from the distribution $\pi(S; A)$. Our first method relies on the key insight that, as we show, the marginal probabilities for DVS can be

computed in polynomial time. To demonstrate this, we begin with the partition function and then derive marginals.

4.5.1 Marginals

The partition function has a conveniently simple closed form, which follows from the Cauchy-Binet formula and was also derived in [13].

Lemma 43 (Partition Function [13]). *For $A \in \mathbb{R}^{n\times m}$ with $r(A) = n$ and $n \leq |S| = k \leq m$, we have*

$$Z_A := \sum\nolimits_{|S|=k, S\subseteq[m]} \det(A_S A_S^\top) = \binom{m-n}{k-n} \det(AA^\top).$$

Next, we will need the marginal probability $P(T \subseteq S; A) = \sum_{S:T\subseteq S} \pi(S; A)$ that a given set $T \subseteq [m]$ is a subset of the random set S. In the following theorem, the set $T_c = [m]\backslash T$ denotes the (set) complement of T, and $Q^\perp$ denotes the orthogonal complement of Q.

Theorem 44 (Marginals). *Let $T \subseteq [m]$, $|T| \leq k$, and $\varepsilon > 0$. Let $A_T = Q\Sigma V^\top$ be the singular value decomposition of A_T where $Q \in \mathbb{R}^{n\times r(A_T)}$, and $Q^\perp \in \mathbb{R}^{n\times(n-r(A_T))}$. Further define the matrices*

$$B = (Q^\perp)^\top A_{T_c} \in \mathbb{R}^{(n-r(A_T))\times(m-|T|)},$$

$$C = \begin{bmatrix} \frac{1}{\sqrt{\sigma_1^2(A_T)+\varepsilon}} & 0 & \dots \\ 0 & \frac{1}{\sqrt{\sigma_2^2(A_T)+\varepsilon}} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} Q^\top A_{T_c} \in \mathbb{R}^{r(A_T)\times(m-|T|)}.$$

Let $Q_B \mathrm{diag}(\sigma_i^2(B)) Q_B^\top$ be the eigenvalue decomposition of $B^\top B$ where $Q_B \in \mathbb{R}^{|T_c|\times r(B)}$. Moreover, let $W^\top = \left[I_{T_c}; C^\top\right]$ and $\Gamma = e_{k-|T|-r(B)}(W((Q_B^\perp)^\top Q_B^\perp)W^\top)$. Then the marginal probability of T in DVS is

$$P(T \subseteq S; A) = \frac{\left[\prod_{i=1}^{r(A_T)} \sigma_i^2(A_T)\right] \times \left[\prod_{j=1}^{r(B)} \sigma_j^2(B)\right] \times \Gamma}{Z_A}.$$

We prove Theorem 44 via a perturbation argument that connects DVS to volume sampling. Specifically, observe that for $\epsilon > 0$ and $|S| \geq n$ it holds that

$$\det(A_S A_S^\top + \varepsilon I_n) = \varepsilon^{n-k} \det(A_S^\top A_S + \varepsilon I_k) = \varepsilon^{n-k} \det\left(\begin{bmatrix} A_S \\ \sqrt{\varepsilon}(I_m)_S \end{bmatrix}^\top \begin{bmatrix} A_S \\ \sqrt{\varepsilon}(I_m)_S \end{bmatrix} \right). \tag{4.5.1}$$

Carefully letting $\epsilon \to 0$ bridges volumes with "dual" volumes. The technical remainder of the proof further relates this equality to singular values, and exploits properties of characteristic polynomials. A similar argument yields an alternative proof of Lemma 43. We show the proofs in detail in Appendix B.1 and B.2 respectively.

Complexity. The numerator of $P(T \subseteq S; A)$ in Theorem 44 requires $\mathcal{O}(mn^2)$ time to compute the first term, $\mathcal{O}(mn^2)$ to compute the second and $\mathcal{O}(m^3)$ to compute the third. The denominator takes $\mathcal{O}(mn^2)$ time, amounting in a total time of $\mathcal{O}(m^3)$ to compute the marginal probability.

4.5.2 Sampling

The marginal probabilities derived above directly yield a polynomial-time *exact* DVS algorithm. Instead of k-sets, we sample ordered k-tuples $\overrightarrow{S} = (s_1, \ldots, s_k) \in [m]^k$. We denote the k-tuple variant of the DVS distribution by $\overrightarrow{P}(\cdot; A)$:

$$\overrightarrow{P}((s_j = i_j)_{j=1}^k; A) = \frac{1}{k!} P(\{i_1, \ldots, i_k\}; A) = \prod_{j=1}^k \overrightarrow{P}(s_j = i_j | s_1 = i_1, \ldots, s_{j-1} = i_{j-1}; A).$$

Sampling $\overrightarrow{S}$ is now straightforward. At the jth step we sample s_j via $\overrightarrow{P}(s_j = i_j | s_1 = i_1, \ldots, s_{j-1} = i_{j-1}; A)$; these probabilities are easily obtained from the marginals in Theorem 44.

Corollary 45. *Let $T = \{i_1, \ldots, i_{t-1}\}$, and $P(T \subseteq S; A)$ as in Theorem 44. Then,*

$$\overrightarrow{P}(s_t = i; A | s_1 = i_1, \ldots, s_{t-1} = i_{t-1}) = \frac{P(T \cup \{i\} \subseteq S; A)}{(k - t + 1)\, P(T \subseteq S; A)}.$$

As a result, it is possible to draw an exact dual volume sample in time $\mathcal{O}(km^4)$.

The full proof may be found in the appendix. The running time claim follows since the sampling algorithm invokes $\mathcal{O}(mk)$ computations of marginal probabilities, each costing $\mathcal{O}(m^3)$ time.

Remark A potentially more efficient approximate algorithm could be derived by noting the relations between volume sampling and DVS. Specifically, we add a small perturbation to DVS as in Equation 4.5.1 to transform it into a volume sampling problem, and apply random projection for more efficient volume sampling as in [49]. Please refer to Appendix B.3 for more details.

4.5.3 Derandomization

Next, we derandomize the above sampling algorithm to *deterministically* select a subset that satisfies the bound (4.3.5) for the Frobenius norm, thereby answering another question in [13]. The key insight for derandomization is that conditional expectations can be computed in polynomial time, given the marginals in Theorem 44:

Corollary 46. *Let $(i_1, \ldots, i_{t-1}) \in [m]^{t-1}$ be such that the marginal distribution satisfies $\overrightarrow{P}(s_1 = i_1, \ldots, s_{t-1} = i_{t-1}; A) > 0$. The conditional expectation can be expressed as*

$$\mathbb{E}\left[\|A_S^\dagger\|_F^2 \mid s_1 = i_1, \ldots, s_{t-1} = i_{t-1}\right] = \frac{\sum_{j=1}^n P'(\{i_1, \ldots, i_{t-1}\} \subseteq S | S \sim P(S; A_{[n]\setminus\{j\}}))}{P'(\{i_1, \ldots, i_{t-1}\} \subseteq S | S \sim \pi(S; A))},$$

where P' are the unnormalized marginal distributions, and it can be computed in $\mathcal{O}(nm^3)$ time.

We show the full derivation in Appendix B.4.

Corollary 46 enables a greedy derandomization procedure. Starting with the empty tuple $\overrightarrow{S}_0 = \emptyset$, in the ith iteration, we greedily select $j^* \in \text{argmax}_j \mathbb{E}[\|A_{S \cup j}^\dagger\|_F^2 \mid (s_1, \ldots, s_i) = \overrightarrow{S}_{i-1} \circ j]$ and append it to our selection: $\overrightarrow{S}_i = \overrightarrow{S}_{i-1} \circ j$. The final set is the non-ordered version S_k of $\overrightarrow{S}_k$. Theorem 47 shows that this greedy procedure succeeds, and implies a deterministic version of the bound (4.3.5).

Theorem 47. *The greedy derandomization selects a column set S satisfying*

$$\|A_S^\dagger\|_F^2 \leq \frac{m-n+1}{k-n+1}\|A^\dagger\|_F^2; \quad \|A_S^\dagger\|_2^2 \leq \frac{n(m-n+1)}{k-n+1}\|A^\dagger\|_2^2.$$

In the proof, we construct a greedy algorithm. In each iteration, the algorithm computes, for each column that has not yet been selected, the expectation conditioned on this column being included in the current set. Then it chooses the element with the lowest conditional expectation to actually be added to the current set. This greedy inclusion of elements will only decrease the conditional expectation, thus retaining the bound in Theorem 47. The detailed proof is deferred to Appendix B.5.

Complexity. Each iteration of the greedy selection requires $\mathcal{O}(nm^3)$ to compute $\mathcal{O}(m)$ conditional expectations. Thus, the total running time for k iterations is $\mathcal{O}(knm^4)$. The approximation bound for the spectral norm is slightly worse than that in (4.3.5), but is of the same order if $k = \mathcal{O}(n)$.

4.6 Experiments

We report selection performance of DVS on real regression data (CompAct, CompAct(s), Abalone and Bank32NH[2]) for experimental design. We use 4,000 samples from each dataset for estimation. We compare against various baselines, including uniform sampling (`Unif`), leverage score sampling (`Lev`) [125], predictive length sampling (`PL`) [190], the sampling (`Smpl`)/greedy (`Greedy`) selection methods in [182] and Fedorov's exchange algorithm [63]. We initialize the MCMC sampler with Kmeans++ [11] for DVS and run for 10,000 iterations, which empirically yields selections that are sufficiently good. We measure performances via (1) the prediction error $\|y - X\hat{\alpha}\|$, and (2) running times. Figure 4-2 shows the results for these three measures with sample sizes k varying from 60 to 200. Further experiments (including for the interpolation $\beta < 1$), may be found in the appendix.

In terms of prediction error, DVS performs well and is comparable with `Lev`. Its strength compared to the greedy and relaxation methods (`Smpl`, `Greedy`, `Fedorov`) is running

[2] `http://www.dcc.fc.up.pt/?ltorgo/Regression/DataSets.html`

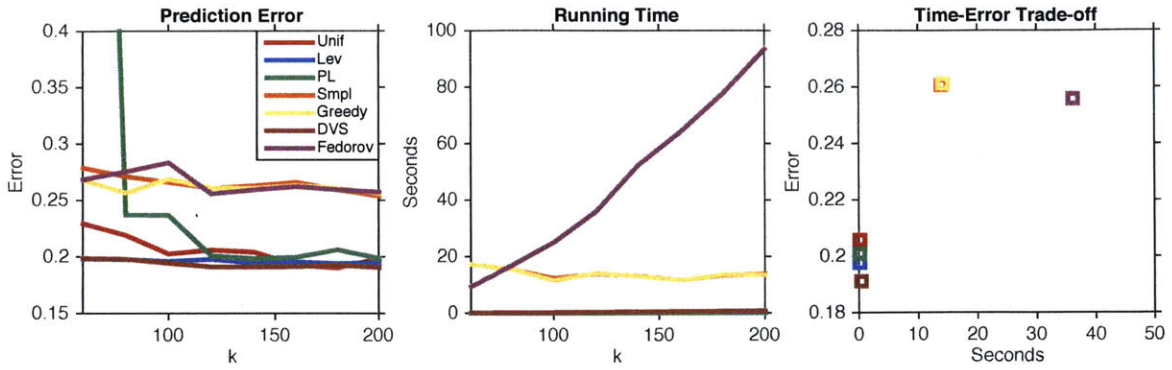


Figure 4-2: Results on the CompAct(s) dataset. Results are the median of 10 runs, except `Greedy` and `Fedorov`. Note that `Unif`, `Lev`, `PL` and `DVS` use less than 1 second to finish experiments.

time, leading to good time-error tradeoffs. These tradeoffs are illustrated in Figure 4-2 for $k = 120$.

In other experiments (shown in Appendix B.7) we observed that in some cases, the optimization and greedy methods (`Smpl`, `Greedy`, `Fedorov`) yield better results than sampling, however with much higher running times. Hence, given time-error tradeoffs, DVS may be an interesting alternative in situations where time is a very limited resource and results are needed quickly.

4.7 Summary

In this chapter, we consider a broader class of DIPMs called strongly Rayleigh measures, which include DPPs as special cases. We obtain an unconditional fast mixing guarantee for MCMC sampling for SR measures. This is the first poly-time mixing MCMC for general SR measures. We further study the problem of DVS via the theory of SR measures and real-stable polynomials and prove that DVS lies in the SR family. This result has remarkable consequences, especially because it implies a provably fast-mixing Markov chain sampler that makes DVS much more attractive to practitioners. Empirical results on experimental design demonstrates the superior performances of DVS over existing methods.

Chapter 5

Constrained Sampling

While general DIPMs have supports on $2^{\mathcal{V}}$, their variants with constrained support typically arise in a variety of real-world settings. Constraints over the support could be imposed from prior knowledge, resource limitations, or other pragmatic considerations. In this chapter, we focus on DIPMs with certain constraints: 1) cardinality constraints, where one wants to have a more precise control over the size of the subsets sampled from DIPMs; 2) matroid base constraints, where one wants to incorporate certain structural information in the sampled subsets. While the unconstrained instances of certain DIPMs have MCMC samplers that are guaranteed to be fast mixing, their constrained variants has no known sampling methods. We develop MCMC samplers for such distributions and identify sufficient conditions under which their chains mix rapidly. Finally, we illustrate our claims by empirically verifying the dependence of mixing times on the key factors governing our theoretical bounds. Materials in this chapter are based on [117, 116]

5.1 Introduction

Distributions over subsets of objects arise in a variety of machine learning applications. They occur as discrete probabilistic models [32, 167, 185, 87, 107] in computer vision, computational biology and natural language processing. They also occur in combinatorial bandit learning [42], as well as in recent applications to neural network compression [133] and matrix approximations [113].

Yet, practical use of discrete distributions can be hampered by computational challenges due to their combinatorial nature. Consider for instance sampling, a task fundamental to learning, optimization, and approximation. Without further restrictions, efficient sampling can be impossible [57]. Several lines of work thus focus on identifying tractable sub-classes, which in turn have had wide-ranging impacts on modeling and algorithms. Important examples include the Ising model [96], matchings (and the matrix permanent) [97], spanning trees (and graph algorithms) [37, 68, 171, 7], and Determinantal Point Processes (DPPs) that have gained substantial attention in machine learning [107, 118, 70, 100, 8, 102].

General distributions on $2^{\mathcal{V}}$ with constrained support typically arise upon incorporating prior knowledge or resource constraints. We focus on resource constraints such as bounds on cardinality and bounds on including limited items from sub-groups. Such constraints can be phrased as a family $\mathcal{C} \subseteq 2^{\mathcal{V}}$ of subsets; we say S satisfies the constraint $\mathcal{C}$ iff $S \in \mathcal{C}$. Then the distribution of interest is of the form

$$\pi_{\mathcal{C}}(S) \propto \exp(\beta F(S))[\![S \in \mathcal{C}]\!], \tag{5.1.1}$$

where $F : 2^{\mathcal{V}} \to \mathbf{R}$ is a set function that encodes relationships between items $i \in \mathcal{V}$, $[\![\cdot]\!]$ is the Iverson bracket, and β a constant (also referred to as the inverse *temperature*). Most prior work on sampling with combinatorial constraints (such as sampling the bases of a matroid), assumes that F breaks up linearly using element-wise weights w_i, i.e., $F(S) = \sum_{i \in S} w_i$. In contrast, we allow generic, nonlinear functions, and obtain a mixing times governed by structural properties of F.

An important thing to note is that, even if $\exp(\beta F(S))$ itself belongs to certain class of distributions like SR, adding a constraint may not lead to a distribution in the same class. Take SR for example, certain constraints on an SR measure may result in an SR measure due to closure properties of SR (see [28] for details), but counter examples exist for more general constraints. Thus even though efficient MCMC sampling method is known for general SR, ones for specific constrained SR measure is not known to be fast mixing.

Our focus is on sampling from $\pi_{\mathcal{C}}$ in (5.1.1), where in our case π is a DIPM; we denote by $Z = \sum_{S \subseteq \mathcal{V}} \exp(\beta F(S))$ and $Z_{\mathcal{C}} = \sum_{S \subseteq \mathcal{C}} \exp(\beta F(S))$. The simplest example of $\pi_{\mathcal{C}}$

is the uniform distribution over sets in $\mathcal{C}$, where $F(S)$ is constant. In general, F may be highly nonlinear. We study MCMC sampling method for SR measures and its various constrained version. We propose different Markov chains for different variants and show they are essentially fast mixing. We summarize the key contributions of this chapter below.

- We propose a general technique for constructing fast mixing Markov chains by combining already fast mixing chains on overlapping subsets of the whole state space. Based on this technique we construct a fast mixing chain for cardinality-constrained SR measures (Theorem 52). Such construction is not restricted to the specific class of SR measures and is more widely applicable.

- We analyze a special case for cardinality constraints, i.e., the case of $|S| \leq k$. We show (in Theorem 56) mixing times of an add-delete chain for such case, which, perhaps surprisingly, turns out to be quite different from $|S| = k$. This constraint can be more practical than the strict choice $|S| = k$, because in many applications, the user may have an upper bound on the budget, but may not necessarily want to expend all k units.

- We analyze (Theorem 57) mixing times of an exchange chain when the constraint family $\mathcal{C}$ is the set of bases a matroid, i.e., $|S| = k$ or S obeys a partition constraint. Both of these constraints have high practical relevance [105, 101, 185].

Finally, a detailed set of experiments illustrates our theoretical results.

Related work. Recent work in machine learning addresses sampling from distributions with sub- or supermodular F [86, 156], determinantal point processes [8, 113], and sampling by optimization [60, 128]. Many of these works (necessarily) make additional assumptions on $\pi_{\mathcal{C}}$, or are approximate, or cannot handle constraints. Moreover, the constraints cannot easily be included in F: an out-of-the-box application of the result in [86], for instance, would lead to an unbounded constant in the mixing time.

5.2 Sampling from SR with Cardinality Constraint

In this section, we consider SR measures with cardinality constraints, namely $\ell \leq |S| \leq u$ where S is the sampled subset. MCMC on general SR measures has already proved to be efficient due to the remarkable properties of SR. However, the SR property is brittle: A restriction of the support like cardinality constraints on the subsets, may destroy it. Notwithstanding, there is now a growing interest in studying the complexity of such combinatorially constrained distributions. Recent work addresses approximations to the partition function, marginal probabilities and mode under various constraints [78, 101, 174, 143]. None of these works, however, considers efficient sampling via MCMC. Our work may be viewed as a first step towards constructing efficient MCMC samplers for potentially non-SR measures by still exploiting SR properties.

To design fast mixing Markov chain samplers for cardinality-constrained SR measures, we develop a *combination Markov chain* that is efficient when the overall state space decomposes into *overlapping* "easy" regions. Assuming that each region has access to an efficient sampler, we show how to use the overlap to obtain an overall fast mixing chain. In contrast to previous work on chain decomposition that was mainly used as a tool for analyzing *given* Markov chains [98, 129], our strategy is *constructive* and explicitly uses decomposition for building a sampler. More importantly, it inherits efficiency from sub-chains.

5.2.1 Chain Combination for Easy Fast-Mixing Chain Construction

Throughout, we assume that the support of $\pi_{\mathcal{C}}$, i.e., the state space $\mathcal{C}$, is covered by m overlapping parts $\mathcal{C}_i$, namely $\mathcal{C} = \bigcup_{i=1}^{m} \mathcal{C}_i$. We assume that for a suitably partially-rescaled version of the distribution $\pi_{\mathcal{C}}$ restricted to each of the $\mathcal{C}_i$, we have a fast mixing chain $\mathcal{M}_{\mathcal{C}_i}$ with transition probabilities Q_i. Such assumption holds for certain cardinality-constrained SR measures as shown in Sec. 5.2.2.

Our approach is motivated by chain decomposition techniques for analyzing Markov chains [134, 98]; we construct a chain that will be easy to analyze with such techniques.

The chain decomposition analysis assumes an already existing (ergodic and time-

reversible) Markov chain with stationary distribution $\pi_{\mathcal{C}}$ and transition probabilities $P(X, Y)$. Given a partition of the state space into m *disjoint* parts $\mathcal{C} = \mathcal{T}_1 \cup \ldots \cup \mathcal{T}_m$, one can decompose the chain into *restriction* chains, one on each $\mathcal{T}_i$, and a *projection* chain across parts. Let $[m] = \{1, \ldots, m\}$, the projection chain represents a distribution $\overline{\pi}_{[m]}(i) := \sum_{X \in \mathcal{T}_i} \pi_{\mathcal{C}}(X)$ over the indices $[m]$ of the parts. Its transition probabilities $\overline{P}$ aggregate the original ones:

$$\overline{P}(i, j) = \overline{\pi}_{[m]}(i)^{-1} \sum_{X \in \mathcal{T}_i, Y \in \mathcal{T}_j} \pi_{\mathcal{C}}(X) P(X, Y). \tag{5.2.1}$$

In addition, we have one restriction chain $\mathcal{M}_{\mathcal{T}_i}$ on each part $\mathcal{T}_i$, whose stationary distribution is the conditional $\pi_{\mathcal{T}_i}(X) = \pi_{\mathcal{C}}(X)/\overline{\pi}_{[m]}(i)$. Its transition probabilities are

$$P_i(X, Y) = \begin{cases} P(X, Y), & X \neq Y \\ 1 - \sum_{Z \in \mathcal{C}_i \setminus \{X\}} P(X, Z) & X = Y, \end{cases} \tag{5.2.2}$$

These transition probabilities are not always easy to compute or even approximate. This issue may arise for support-restricted non-uniform distributions such as constrained DPPs or SR measures.

In contrast to analyzing an existing chain on the entire state space, our approach uses a different, bottom-up approach, that combines chains on sub-parts of the state space.

Case of $m = 2$: We start with the base case of $m = 2$ parts, i.e., $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$ and $\mathcal{C}_1 \cap \mathcal{C}_2 = \mathcal{I} \neq \emptyset$. Let $\mathcal{D}_i = \mathcal{C}_i \backslash \mathcal{C}_{3-i}$ (for $i \in [2]$). We first make two identical copies $\mathcal{I}_1$ and $\mathcal{I}_2$ of $\mathcal{I}$ and let $\mathcal{C}'_i = \mathcal{I}_i \cup \mathcal{D}_i$ be each part with the duplicated intersection. Consider two new measures $\pi_{\mathcal{C}'_1}$ and $\pi_{\mathcal{C}'_2}$ that distribute the mass on the intersection:

$$\pi_{\mathcal{C}'_i}(X) \propto \begin{cases} \pi_{\mathcal{C}}(X) & \text{if } X \in \mathcal{D}_i \\ p^{i-1}(1-p)^{2-i} \pi_{\mathcal{C}}(X) & \text{if } X \in \mathcal{I}_i, \end{cases}$$

for $i \in [2]$ and $p \in (0, 1)$. Let Q_i be the transition probabilities for a Markov chain with stationary distribution $\pi_{\mathcal{C}'_i}$ (that we have by assumption). We create a "lazier" version of this

chain, with transition probabilities

$$P_i(X,Y) := \begin{cases} \frac{1}{4}Q_i(X,Y), & \text{if } X,Y \in \mathcal{C}'_i,\ X \neq Y, \\ 1 - \sum_{Z \in \mathcal{C}'_i, Z \neq X} \frac{1}{4}Q_i(X,Z), & \text{if } X,Y \in \mathcal{C}'_i,\ X = Y. \end{cases} \tag{5.2.3}$$

Each chain $\mathcal{M}_{\mathcal{C}'_i}$ again has stationary distribution $\pi_{\mathcal{C}'_i}$.

Next, we combine $\mathcal{M}_{\mathcal{C}'_1}$ and $\mathcal{M}_{\mathcal{C}'_2}$ by allowing transitions between $\mathcal{I}_1$ and $\mathcal{I}_2$. The combined chain $\mathcal{M}_{\mathcal{C}'_1 \cup \mathcal{C}'_2}$ has transition probabilities

$$P(X,Y) := \begin{cases} \frac{1}{4}p^{2-i}(1-p)^{i-1} & \text{if } X \in \mathcal{I}_i,\ Y \in \mathcal{I}_{1-i},\ X \text{ and } Y \text{ are identical copies in } \mathcal{I}, \\ P_i(X,Y) & \text{if } X,Y \in \mathcal{C}'_i,\ X \neq Y, \\ 1 - \sum_{Z \neq X} P(X,Z) & \text{if } X = Y. \end{cases} \tag{5.2.4}$$

This chain is still lazy; it does not move with probability at least $1/2$. From detailed balance, it follows that $\mathcal{M}_{\mathcal{C}'_1 \cup \mathcal{C}'_2}$ converges to the distribution

$$\pi_{\mathcal{C}'_1 \cup \mathcal{C}'_2} = \begin{cases} \pi_S(X) & \text{if } X \in \mathcal{D}_i, \\ p^{i-1}(1-p)^{2-i}\pi_{\mathcal{C}}(X) & \text{if } X \in \mathcal{I}_i, \end{cases} \quad i \in \{1,2\}.$$

Each $X \in \mathcal{I}$ occurs via its two duplicates in $\mathcal{I}_1$ and $\mathcal{I}_2$ with probability $p\pi_{\mathcal{C}}(X)$ and $(1-p)\pi_{\mathcal{C}}(X)$, respectively. Hence, by re-identifying both copies with X ("projecting"), we obtain a sampler for $\pi_{\mathcal{C}}$. Algorithm 13 makes the above described chain explicit.

Analysis. We must now bound the mixing time of the combined chain. For doing so, we follow the decomposition analysis of [98]. The main idea is to bound a quantity – the *Poincaré constant* or *log-Sobolev constant* – that characterizes the mixing time, once on each part and once for transiting between parts. The final constant and mixing time will follow as a function of these quantities associated with the parts.

These important quantities are defined as follows. Let $f : \mathcal{C} \to \mathbb{R}$ be an arbitrary test function; its expectation and variance with respect to $\pi_{\mathcal{C}}$ are $\mathbb{E}_{\pi_{\mathcal{C}}}[f] = \sum_{X \in \mathcal{C}} \pi_{\mathcal{C}}(X) f(X)$ and $\mathcal{V}_{\pi_{\mathcal{C}}}[f] = \sum_{X \in \mathcal{C}} \pi_{\mathcal{C}}(X)(f(X) - \mathbb{E}_{\pi_{\mathcal{C}}} f)^2$, respectively. The Poincaré constant λ bounds

Algorithm 13 Combined Chain for $m = 2$ parts

Require: Target distribution $\pi_{\mathcal{C}}(\cdot)$, state decomposition $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$, $\mathcal{I} = \mathcal{C}_1 \cap \mathcal{C}_2$, transition probabilities $P_i(\cdot, \cdot)$ defined in (5.2.3), $p \in (0, 1)$
Initialize $T \in \mathcal{C}$. Let $b = i$ if $S \in \mathcal{C}_i \backslash \mathcal{I}$ for $i \in \{1, 2\}$, otherwise set $b = 1$.
while not mixed **do**
  Run chain with transition probabilities P_b for one step
  if the new state is the same as last step **then**
    if $T \in \mathcal{I}$ **then**
      Set $b = 3 - b$ with probability $\frac{1}{4} p^{2-b}(1-p)^{b-1}$
    end if
  end if
end while
Output $T \in \mathcal{C}$

the ratio between the variance and the *Dirichlet form*

$$\mathcal{E}_{\pi_{\mathcal{C}}}(f, f) = \frac{1}{2} \sum_{X,Y \in \mathcal{C}} \pi_{\mathcal{C}}(X) P(X, Y)(f(X) - f(Y))^2;$$

it is the largest constant such that the Poincaré inequality $\lambda \mathcal{V}_{\pi_{\mathcal{C}}}[f] \leq \mathcal{E}_{\pi_{\mathcal{C}}}(f, f)$ holds for all functions $f : \mathcal{C} \to \mathbb{R}$. The log-Sobolev constant replaces the variance by $\mathcal{L}_{\pi_{\mathcal{C}}}(f) = \mathbb{E}_{\pi_{\mathcal{C}}}[f^2(\ln f^2 - \ln(\mathbb{E}_{\pi_{\mathcal{C}}}[f]^2))]$; it is the largest α such that $\alpha \mathcal{L}_{\pi_{\mathcal{C}}}(f) \leq \mathcal{E}_{\pi_{\mathcal{C}}}(f, f)$ for all $f : \mathcal{C} \to \mathbb{R}$.

The decomposition technique assumes an already existing Markov chain, and a partition of the state space into m *disjoint* parts $\mathcal{C} = \mathcal{T}_1 \cup \ldots \cup \mathcal{T}_m$. If it is possible to bound λ (or α) for the restriction chain on each part $\mathcal{T}_i$ individually, and for the global projection chain, then the following result implies a bound on the mixing time.

Theorem 48 (Mixing time [98]). *Consider a finite-state time-reversible Markov chain decomposed into a projection chain and m restriction chains on disjoint parts $\mathcal{T}_i$ of the state space $\mathcal{C}$. Let $\overline{\lambda}$, $\overline{\alpha}$ be Poincaré and log-Sobolev constants for the projection chain, and $\{\lambda_i\}_{i \in [m]}$, $\{\alpha_i\}_{i \in [m]}$ be those for the restriction chains, respectively. Let $\lambda_{\min} := \min_i \lambda_i$ and $\alpha_{\min} := \min_i \alpha_i$, and define*

$$\gamma := \max_{i \in [m]} \max_{X \in \mathcal{C}_i} \sum_{Y \in \mathcal{C} \backslash \mathcal{C}_i} P(X, Y). \tag{5.2.5}$$

Then the original Markov chain satisfies a Poincaré and log-Sobolev inequality with constants

$$\lambda = \min\left\{\frac{\overline{\lambda}}{3}, \frac{\overline{\lambda}\lambda_{\min}}{3\gamma + \overline{\lambda}}\right\}; \quad \alpha = \min\left\{\frac{\overline{\alpha}}{3}, \frac{\overline{\alpha}\alpha_{\min}}{3\gamma + \overline{\alpha}}\right\}.$$

These constants imply upper bounds on mixing time:

$$\tau_{S_0}(\varepsilon) = \mathcal{O}\left(\frac{1}{\lambda}\log\left(\varepsilon\pi_{\mathcal{C}}(S_0)\right)^{-1}\right); \quad \tau_{S_0}(\varepsilon) = \mathcal{O}\left(\frac{1}{\alpha}\log\left(\varepsilon^{-1}\log\pi_{\mathcal{C}}(X_0)^{-1}\right)\right).$$

Our analysis uses this result – note that Theorem 5.2.7 uses a partition into disjoint parts. Moreover, the critical ingredient for using Theorem 5.2.7 are bounds on the local and global Poincaré constants, which are not always easy to obtain. It turns out that, by construction, our chain combination admits such bounds, and we obtain the following bounds for our combination chain.

Theorem 49. *Given a decomposition $\mathcal{C} = \mathcal{C}_1 \cup \mathcal{C}_2$ where $\mathcal{C}_1 \cap \mathcal{C}_2 \neq \emptyset$, define the chains $\mathcal{M}_{\mathcal{C}_1'}$, $\mathcal{M}_{\mathcal{C}_2'}$ and $\mathcal{M}_{\{1,2\}}$ as above. Let $\overline{\lambda}$ and $\{\lambda_i\}_{i\in\{1,2\}}$ be Poincaré constants, $\overline{\alpha}$ and $\{\alpha_i\}_{i\in\{1,2\}}$ be log-Sobolev constants for $\{\mathcal{M}_{\mathcal{C}_i'}\}$ and $\mathcal{M}_{\{1,2\}}$ respectively. With $\lambda_{\min} = \min_i \lambda_i$ and $\alpha_{\min} = \min_i \alpha_i$, we have*

$$\lambda \geq \min\left\{\frac{\overline{\lambda}}{3}, \frac{\overline{\lambda}\lambda_{\min}}{\frac{3}{4}\max\{p, 1-p\} + \overline{\lambda}}\right\}; \quad \alpha \geq \min\left\{\frac{\overline{\alpha}}{3}, \frac{\overline{\alpha}\alpha_{\min}}{\frac{3}{4}\max\{p, 1-p\} + \overline{\alpha}}\right\}.$$

In particular, for $P(X, Y)$ defined as in Equation (5.2.4)*:*

$$\lambda \geq \min\left\{\frac{p(1-p)\pi_{\mathcal{C}}(\mathcal{I})}{3}, \frac{\lambda_{\min}}{\frac{3\max\{p,1-p\}}{4p(1-p)\pi_{\mathcal{C}}(\mathcal{I})} + 1}\right\}.$$

Proof. By definition of $\mathcal{M}_{\mathcal{C}_0' \cup \mathcal{C}_1'}$, we let

$$\overline{\pi}_{[2]}(i) = \sum_{X \in \mathcal{C}_i'} \pi_{\mathcal{C}_i'}(X) = \pi_{\mathcal{C}}(\mathcal{D}_i) + p^{i-1}(1-p)^{2-i}\pi_{\mathcal{C}}(\mathcal{I})$$

and construct the projection chain $\mathcal{M}_{\{1,2\}}$ with transition probabilities

$$\begin{aligned}\overline{P}(i,1-i) &= \frac{\sum_{X\in\mathcal{I}_i, Y\in\mathcal{I}_{1-i}} p^{i-1}(1-p)^{2-i}\pi_{\mathcal{C}}(X)P(X,Y)}{\pi_{\mathcal{C}}(\mathcal{D}_i)+p^{i-1}(1-p)^{2-i}\pi_{\mathcal{C}}(\mathcal{I})} \\ &= \frac{p(1-p)\pi_{\mathcal{C}}(\mathcal{I})}{4(\pi_{\mathcal{C}}(\mathcal{D}_i)+p^{i-1}(1-p)^{2-i}\pi_{\mathcal{C}}(\mathcal{I}))}.\end{aligned}$$

The resulting Poincaré constant is given by [8, Fact 2.1]

$$\overline{\lambda} = \frac{\overline{P}(1,2)}{\overline{\pi}_{[2]}(2)} = \frac{p(1-p)\pi_{\mathcal{C}}(\mathcal{I})}{4(\pi_{\mathcal{C}}(\mathcal{D}_1)+(1-p)\pi_{\mathcal{C}}(\mathcal{I}))(\pi_{\mathcal{C}}(\mathcal{D}_2)+p\pi_{\mathcal{C}}(\mathcal{I}))} \geq p(1-p)\pi_{\mathcal{C}}(\mathcal{I}). \tag{5.2.6}$$

Finally, we have

$$\gamma = \max_{i\in[2]}\max_{X\in\mathcal{C}_i'}\sum_{Y\in\mathcal{C}_{3-i}'} P(X,Y) = \frac{1}{4}\max\{p,1-p\}. \tag{5.2.7}$$

Together with Theorem 48, the bounds (5.2.7) and (5.2.6) imply the results. □

Theorem 49 matches intuition: if the small chain on each part of the state space is fast mixing, the resulting λ_i's are bounded away from 0; if the probability of intersection states, $\pi(\mathcal{I})$, is large, $\overline{\lambda}$ will be bounded away from 0 and it will be easy to transit between chains. Hence λ is large and the resulting combined chain is fast mixing.

A key point in our chain combination is how the transition probabilities between smaller chains are set. They are a constant, resulting in an easy-to-analyze projection chain. This greatly eases the bounding procedure for the whole chain, as shall be seen in Section 5.2.2.

General Case

Next, we extend our analysis from $m = 2$ to arbitrarily many parts $\{\mathcal{C}_i\}_{i=1}^m$ of $\mathcal{C} = \cup_i \mathcal{C}_i$. We assume that the decomposition is such that each part $\mathcal{C}_i$ is *reachable* from any other part $\mathcal{C}_j$, i.e., for any $i, j \in [m]$ there exists a sequence of subsets $\mathcal{C}_{i_1}, \ldots, \mathcal{C}_{i_k}$ such that $\mathcal{C}_i \cap \mathcal{C}_{i_1} \neq \emptyset$, $\mathcal{C}_{i_1} \cap \mathcal{C}_{i_2} \neq \emptyset, \ldots, \mathcal{C}_{i_k} \cap \mathcal{C}_j \neq \emptyset$.

The construction proceeds as for $m = 2$. We create a copy $\mathcal{C}_i'$ of each of the m parts

$\mathcal{C}_i$. The $\mathcal{C}_i'$ are disjoint; they contain copies of the intersections of parts. As a result, each state $X \in \mathcal{C}$ is copied $|C(X)|$ times, where $C(X) = \{\mathcal{C}_i | X \in \mathcal{C}_i\}$ is the number of parts it is contained in. We "spread" X across its copies via a distribution p_X over $C(X)$.

Consider m new distributions $\{\pi_{\mathcal{C}_i'}\}_{i=1}^m$ on state spaces $\mathcal{C}_i'$ with probabilities

$$\pi_{\mathcal{C}_i'}(X) \propto p_X(\mathcal{C}_i)\pi_{\mathcal{C}}(X), \quad X \in \mathcal{C}_i', i \in [m].$$

Again we assume that constructing m smaller chains $\mathcal{M}_{\mathcal{C}_i'}$ on $\mathcal{C}_i'$ with transition probabilities Q_i is easy. We then construct $\mathcal{M}_{\mathcal{C}_i'}$ with transition probabilities P_i as in (5.2.3).

Now we combine $\{\mathcal{M}_{\mathcal{C}_i'}\}$ by allowing transitions between identical copies of elements in $\cup_i \mathcal{C}_i'$. Specifically, we construct the following chain $\mathcal{M}_{\cup_i \mathcal{C}_i'}$ with transition probabilities

$$P(X,Y) = \begin{cases} \frac{1}{4} p_X(\mathcal{C}_j) & \text{if } X \in \mathcal{C}_i' \text{ and } Y \in \mathcal{C}_j' \text{ are identical copies, } i \neq j, \\ P_i(X,Y) & \text{if } X, Y \in \mathcal{C}_i', X \neq Y, \\ 1 - \sum_{Z \neq X} P(X,Z) & \text{if } X = Y. \end{cases}$$

This lazy chain has stationary distribution

$$\pi_{\cup_i \mathcal{C}_i'}(X) = p_X(\mathcal{C}_i)\pi_{\mathcal{C}}(X), \quad X \in \mathcal{C}_i'.$$

When sampling, we again "project" and re-identify all copied samples with the original X; thus any $X \in \mathcal{C}$ is sampled with probability $\pi_{\mathcal{C}}$.

By our construction, the projection chain has transition probabilities

$$\overline{\pi}_{[m]}(i) = \sum_{X \in \mathcal{C}_i'} \pi_{\mathcal{C}_i'}(X) = \sum_{X \in \mathcal{C}_i'} p_X(\mathcal{C}_i)\pi_{\mathcal{C}}(X),$$

and construct the projection chain $\mathcal{M}_{[m]}$ with transition probabilities

$$\overline{P}(i,j) = \frac{\sum_{X \in \mathcal{C}_i', Y \in \mathcal{C}_j'} p_X(\mathcal{C}_i)\pi_{\mathcal{C}}(X)P(X,Y)}{\overline{\pi}_{[m]}(i)}, \quad i,j \in [m].$$

Finally we have

$$\gamma = \max_{i\in[m]} \max_{X\in\mathcal{C}_i'} \sum_{Y\in\cup_{j\neq i}\mathcal{C}_j'} P(X,Y) \leq 1/4. \tag{5.2.8}$$

The Poincaré and log-Sobolev constants are then given by the following theorem.

Theorem 50. *Given decomposition $\mathcal{C} = \mathcal{C}_1 \cup \ldots \cup \mathcal{C}_m$ where $\mathcal{C}_i \cap (\cup_{j\neq i}\mathcal{C}_j) \neq \emptyset$, and chains $\mathcal{M}_{\mathcal{C}_i'}$ and $\mathcal{M}_{[m]}$ defined as above. Let $\overline{\lambda}$ and $\{\lambda_i\}_{i\in[m]}$ be Poincaré constants, $\overline{\alpha}$ and $\{\alpha_i\}_{i\in[m]}$ be log-Sobolev constants for $\{\mathcal{M}_{\mathcal{C}_i'}\}$ and $\mathcal{M}_{[m]}$ respectively. With $\lambda_{\min} = \min_i \lambda_i$ and $\alpha_{\min} = \min_i \alpha_i$, we have*

$$\lambda \geq \min\left\{\frac{\overline{\lambda}}{3}, \frac{\overline{\lambda}\lambda_{\min}}{3/4+\overline{\lambda}}\right\}; \quad \alpha \geq \min\left\{\frac{\overline{\alpha}}{3}, \frac{\overline{\alpha}\alpha_{\min}}{3/4+\overline{\alpha}}\right\},$$

where γ is defined in Eq. 5.2.8.

Discussion. Our construction gives a principled way to tackle the problem of constructing Markov chains on a large and complex state spaces in a bottom-up fashion, where we first construct simple chains on subsets of state space and combine them together. Note that such construction could be made recursive. Once a large chain has been constructed, we can combine it with other large chains to form more complex chains on larger state spaces.

A Toy Example

We consider constructing a Markov chain that samples from a $\pi_{\mathcal{C}}$ where $\mathcal{C} = [3n]$. Each of the first n points $\mathcal{T}_1$ has probability p_1/n, each of the second n points $\mathcal{T}_2$ has probability p_2/n and each of the last n points $\mathcal{T}_3$ has probability $(p_1+p_2)/n$, where $p_1 + p_2 = 1/2$. To construct a Markov chain to sample from this distribution, one obvious way is to decompose $\mathcal{C}$ as $\mathcal{C}_1 \cup \mathcal{C}_2$, where $\mathcal{C}_1 = \mathcal{T}_1 \cup \mathcal{T}_3$ and $\mathcal{C}_2 = \mathcal{T}_2 \cup \mathcal{T}_3$. This is good because each small chain is well studied – it is just a uniform distribution on $\{1,\ldots,2n\}$ and $\{n+1,\ldots,3n\}$. Meanwhile, the constructed projection chain has only 2 states, thus the Poincaré constant is computable immediately.

Note that one can also construct a Metropolis-Hastings or Gibbs-style Markov chain

and use technique based on [98] to bound the mixing time. One straightforward way is to decompose $\mathcal{C} = \mathcal{T}_1 \cup \mathcal{T}_2 \cup \mathcal{T}_3$. However, the resulting projection chain would be a 3-state chain instead of two-state constructed with chain combination, the Poincaré or log-Sobolev constant of which is a bit harder to compute. Such disadvantages may become more pronounced when the state space and distributions are less uniform, and we may wish to decompose the state space into more parts: while using disjoint parts as in [98] results in an $\Omega(M^2)$-state projection chain where M is the number of overlapping components like $\mathcal{C}_i$ above, the chain combination would only construct a projection with an $\mathcal{O}(M)$-state projection chain.

5.2.2 Application to Sampling from SR with Cardinality Constraints

We now apply our chain combination technique to sample from a constrained SR and prove an unconditioned polynomial mixing time bound on constrained SR. Although we focus on DPP in this section, analogous results hold with any SR distribution.

[117] showed a Markov chain for *any* DPP (SR) distribution on $2^{\mathcal{V}}$ that is essentially a chain on an n-homogeneous SR measure on $2^{[2n]}$, thus we obtain the following corollary:

Corollary 51. *The Poincaré constant for the chain for sampling from general SR measures, described in Algorithm 11, is at least* $\frac{1}{2n^2}$.

As we have mentioned, the constrained DPPs we are interested in are not known to be SR. The existing bounds, e.g. in [117], are only polynomial under certain additional conditions.

The concrete constrained DPP that we illustrate below has the following measure:

$$\pi_{\mathcal{C}}(S) := \Pr(S \mid k-1 \leq |S| \leq k+1) \propto \det(L_{S,S}) [\![k-1 \leq |S| \leq k+1]\!]. \quad (5.2.9)$$

Here we have $\mathcal{C} = \{S \mid k-1 \leq |S| \leq k+1\}$. This constraint slightly generalizes the fixed cardinality constraint $|S| = k$, and serves to illustrate our general technique; moreover, it forms the basis of DPPs under interval constraints on cardinality (end of this section). For the constrained DPP of (5.2.9) we have the following result.

Theorem 52 (Mixing Time). *For $k \geq 1$, there is a Markov chain starting with stationary distribution given in* (5.2.9) *and initial state S_0 with a mixing time of*

$$\mathcal{O}(n^2(\log \pi_{\mathcal{C}}(S_0)^{-1} + \log \varepsilon^{-1} + \log \binom{n}{|S_0|})).$$

Remarks. We highlight here the fact that the above mixing time bound holds not only for (5.2.9), but for all SR distributions with the same constraints. Moreover, the associated distribution $\pi_{\mathcal{C}}$ may not be SR, and we are not aware of any fast mixing time bounds for such potentially non-SR distributions in the literature. Therefore, our chain composition technique provides means to extend existing fast mixing Markov chains to sample from distributions for which fast mixing was previously unknown.

The proof of Theorem 52 uses a combination chain that is composed of two parts each with an SR measure. The chains within each part, and their Poincaré constants, follow from properties of SR measures (re-weighted via *rank sequences*).

The proof of the theorem will make use of the following properties of SR measures. Rank sequences will be used for combining the parts via reweighting, and for establishing bounds.

Theorem 53. *[SR under Rank Rescaling [149]] Let $\pi_{\mathcal{C}}$ where $\mathcal{C} \subseteq 2^{\mathcal{V}}$ be SR and $\{b_i : 0 \leq i \leq n\}$ a finite sequence of nonnegative numbers such that $\sum_i b_i x^i$ is stable (namely, it has only real roots). Then the measure $\pi'_{\mathcal{C}}(S) \propto b_{|S|}\pi_{\mathcal{C}}(S)$ is also SR.*

Lemma 54. *[Log-Concavity [28]] For any distribution $\pi_{\mathcal{C}}$ where $\mathcal{C} \subseteq 2^{\mathcal{V}}$, the sequence $\left\{a_k = \sum_{S \in \mathcal{C}, |S|=k} \pi_{\mathcal{C}}(S)\right\}_{k=0}^{n}$ is called the rank sequence of $\pi_{\mathcal{C}}$. If $\pi_{\mathcal{C}}$ is SR, its rank sequence is log-concave, namely $a_k^2 \geq a_{k-1}a_{k+1}$, $1 \leq k \leq n-1$.*

Now we are ready to prove our mixing time bound for the constrained DPP.

Proof. *(of Thm. 52)* Let $\pi_{\mathcal{C}}$ the constrained DPP distribution in Eq. (5.2.9). Let $\mathcal{C}_{(i,i+1)} = \{S \mid S \in \mathcal{C}, i \leq |S| \leq i+1\}$, we have

$$\mathcal{C}_{(k-1,k)} \cup \mathcal{C}_{(k,k+1)} = \mathcal{C}; \quad \mathcal{C}_{(k-1,k)} \cap \mathcal{C}_{(k,k+1)} = \{S \mid S \in \mathcal{C}, |S| = k\}.$$

To construct a chain on $\mathcal{C}$ via chain combination, we make two copies of states $\{S \mid S \in \mathcal{C}, |S| = k\}$ and construct $\mathcal{C}'_{(i-1,i)}$ with elements being identical copies of ones in $\mathcal{C}_{(i-1,i)}$. We consider two new distributions $\{\pi_{\mathcal{C}'_{(i,i+1)}}\}$ on state spaces $\mathcal{C}'_{(i,i+1)}$ for $i \in \{k-1, k\}$ with probabilities

$$\pi_{\mathcal{C}'_{(i,i+1)}}(S) \propto \begin{cases} \frac{\pi_{\mathcal{C}}(S)}{2}; & |S| = k \\ \pi_{\mathcal{C}}(S); & \text{otherwise} \end{cases} \qquad \text{for } S \in \mathcal{C}_{(i,i+1)}.$$

By Theorem 53 we know that for any $a_i, a_{i+1} \geq 0$, the distribution

$$\pi_{\mathcal{C}'_{(i,i+1)}}(S) \propto \det(L_S)(a_i[\![|S| = i]\!] + a_{i+1}[\![|S| = i+1]\!])$$

is still SR, thus $\{\pi_{\mathcal{C}'_{(i,i+1)}}\}$ are SR measures. We construct symmetric homogenizations of $\{\pi_{\mathcal{C}'_{(i,i+1)}}\}$ as $\{\pi_{\mathcal{T}'_{(i,i+1)}}\}$ where $\mathcal{T}'_{(i,i+1)} = \{S \mid S \subseteq \mathcal{Z}, i \leq |S \cap \mathcal{V}| \leq i+1\}$. The resulting distributions are n-homogeneous SR. We construct two Markov chains on $\mathcal{T}'_{(k-1,k)}$ and $\mathcal{T}'_{(k,k+1)}$ with transition probabilities $Q_{(k-1,k)}$ and $Q_{(k,k+1)}$ as in Algorithm 11. Then we construct smaller chains $\mathcal{M}_{\mathcal{T}'_{(k-1,k)}}$ and $\mathcal{M}_{\mathcal{T}'_{(k,k+1)}}$ on $\mathcal{T}'_{(k-1,k)}$ and $\mathcal{T}'_{(k,k+1)}$ with transition probabilities as

$$P_{(i,i+1)}(X,Y) = \begin{cases} \frac{1}{4}Q_{(i,i+1)}(X,Y), & X \neq Y \\ 1 - \sum_{Z \in \mathcal{T}'_{(i,i+1)}, Z \neq X} P_{(i,i+1)}(X,Z), & X = Y \end{cases} \text{for } X, Y \in \mathcal{T}'_{(i,i+1)}, i \in \{k-1, k\}.$$

Let λ_1 and λ_2 be the corresponding Poincaré constants. By Corollary 51 and a simple application of a comparison technique [51] we have $\lambda_{\min} = \min\{\lambda_1, \lambda_2\} = \Omega(1/n^2)$. We combine $\{\mathcal{M}_{\mathcal{T}'_{(i,i+1)}}\}$ to form $\mathcal{M}_{\cup_{i \in \{k-1,k\}} \mathcal{T}'_{(i,i+1)}}$ by allowing transitions between identical copies of $X \subseteq \mathcal{Z}$. The transition probability is then given by

$$P(X,Y) = \begin{cases} 1/8 & \text{if } X \text{ and } Y \text{ are identical copies, } X \neq Y, \\ P_{(i,i+1)}(X,Y) & \text{if } X, Y \in \mathcal{T}'_{(i,i+1)}, X \neq Y, \\ 1 - \sum_{Z \neq X} P(X,Z) & \text{if } X = Y. \end{cases}$$

After the chain mixes well, we output the identical copies (in $\mathcal{Z}$) of the resulting state (in

$\mathcal{T}'_{(k-1,k)} \cup \mathcal{T}'_{(k,k+1)}$) and then take the intersection with $\mathcal{V}$. This will give us an element in $\mathcal{C}$ with probability distribution $\pi_{\mathcal{C}}$.

The projection chain has transition probability

$$\overline{P}(k-1,k) = \frac{e_k(L)}{8(e_k(L)+2e_{k-1}(L))},$$
$$\overline{P}(k,k-1) = \frac{e_k(L)}{8(e_k(L)+2e_{k+1}(L))}.$$

This is a random walk on the states of $\{k-1,k\}$. Note that this is a lazy time-reversible chain with stationary distribution

$$\overline{\pi}(i) = \frac{[\![i \in \{k-1,k\}]\!]}{Z}\left(e_k(L) + 2[\![i=k-1]\!]e_{k-1}(L) + 2[\![i=k]\!]e_{k+1}(L)\right),$$

where $Z = 2\sum_{i=k-1}^{k+1} e_i(L)$. The resulting Poincaré constant is give by

$$\overline{\lambda} = \frac{e_k(e_{k-1}+e_k+e_{k+1})}{4(e_k+2e_{k-1})(e_k+2e_{k+1})}.$$

Since DPP is SR and by Lemma 54, we have

$$\begin{aligned} 20e_k(e_{k-1}+e_k+e_{k+1}) &\leq 8e_{k-1}e_k + 4e_k^2 + 8e_ke_{k+1} + 16e_k^2 \\ &\leq 8e_{k-1}e_k + 4e_k^2 + 8e_ke_{k+1} + 16e_{k-1}e_{k+1} = 4(e_k+2e_{k-1})(e_k+2e_{k+1}) \end{aligned}$$

It follows that $\overline{\lambda} \geq 1/20 = \Omega(1)$.

Finally we know that $\gamma = \frac{1}{8}$. Aggregating these results we have that the Poincaré constants for $\mathcal{M}'$ is

$$\lambda' \geq \min\left\{\Omega(1), \frac{\frac{1}{20} \times \Omega(1/n^2)}{\frac{3}{8}+\frac{1}{20}}\right\} = \Omega(1/n^2),$$

Thus it follows that the mixing time of the chain on constrained DPP is bounded as

$$\mathcal{O}(n^2(\log \pi(X_0)^{-1} + \log \varepsilon^{-1} + \log \binom{n}{|X_0|})).$$

□

Extension to interval constraints. The aforementioned result can be extended to the case where the constraint is an interval of sizes of the sampled subset. Specifically, consider the following *interval-constrained DPP*:

$$\Pr(X \mid \ell \leq |X| \leq u) \propto \det(L_X)[\![\ell \leq |X| \leq u]\!]. \tag{5.2.10}$$

By applying chain combination with a proper decomposition of the whole state space, we have the following bound on mixing time:

Theorem 55. *Assume $u \geq \ell \geq 0$, and let $C = \max_{\ell \leq i,j \leq u} \frac{e_i(L)}{e_j(L)}$. There is a Markov chain that samples a DPP with interval constraint, with mixing time of*

$$\mathcal{O}(C(u-\ell+1)^2 n^2 (\pi_{\mathcal{C}}(X_0)^{-1} + \log \varepsilon^{-1} + \log \binom{n}{|X_0|})).$$

The details of the proof are left to Appendix. The theorem indicates that the bound on the mixing time depends on the spectrum of the matrix L, and the cardinality interval. Note that by setting $\ell = 0$ and $u = k$, the distribution becomes a DPP with a uniform matroid constraint. The analysis for this case in [117] requires an intractable to compute constant in the bound on the mixing time. Our bound is also conditional, but the factor C is tractable and thus provides a way of directly computing the bound on mixing time for any given instance.

As before, the above result generalizes to general SR measures, by replacing the elementary symmetric polynomials e_i with the rank sequence of the measure of interest.

One-sided Constraint on Cardinality We consider a special case of cardinality constraint where the constraint is an upperbound on the sampled subsets: $\mathcal{C} = \{S : |S| \leq k\}$. We employ the lazy add-delete Markov chain in Algo. 14, where in each iteration, with probability 0.5 we uniformly randomly sample one element from $\mathcal{V}$ and either add it to or delete it from the current set, while respecting constraints. To show fast mixing, we consider using *path coupling*, which essentially says that if we have a contraction of two

(coupling) chains then we have fast mixing. We construct path coupling $(S,T) \to (S',T')$ on a carefully generated graph with edges E (from a proper metric). We end up with the following theorem:

Theorem 56. *Consider the chain shown in Algorithm 14. Let* $\alpha = \max_{(S,T)\in E}\{\alpha_1, \alpha_2\}$ *where* α_1 *and* α_2 *are functions of edges* $(S,T) \in E$ *and are defined as*

$$\begin{aligned}
\alpha_1 =& \sum_{i\in T} |p^-(T,i) - p^-(S,i)|_+ + [\![|S| < k]\!] \sum_{i\in[n]\setminus S} (p^+(S,i) - p^+(T,i))_+; \\
\alpha_2 =& 1 - (\min\{p^-(S,s), p^-(T,t)\} - \sum_{i\in R} |p^-(S,i) - p^-(T,i)| + \\
& [\![|S| < k]\!](\min\{p^+(S,t), p^+(T,s)\} - \sum_{i\in[n]\setminus(S\cup T)} |p^+(S,i) - p^+(T,i)|)),
\end{aligned}$$

where $(x)_+ = \max(0,x)$. *The summations over absolute differences quantify the sensitivity of transition probabilities to adding/deleting elements in neighboring* (S,T). *Assuming* $\alpha < 1$, *we get*

$$\tau(\varepsilon) \leq \frac{2n\log(n\varepsilon^{-1})}{1-\alpha}$$

Algorithm 14 Add-Delete Markov Chain for One-Sided Cardinality Constraint

Require: F the set function, β the inverse temperature, $\mathcal{V}$ the ground set, k the rank of $\mathcal{C}$
Ensure: S sampled from $\pi_{\mathcal{C}}$
  Initialize $S \in \mathcal{C}$
  while not mixed **do**
    Let $b = 1$ with probability 0.5
    if $b = 1$ **then**
      Draw $s \in \mathcal{V}$ uniformly randomly
      if $s \notin S$ and $|S \cup \{s\}| \leq k$ **then**
        $S \leftarrow S \cup \{s\}$ with probability $p^+(S,s) = \frac{\pi_{\mathcal{C}}(S\cup\{s\})}{\pi_{\mathcal{C}}(S)+\pi_{\mathcal{C}}(S\cup\{s\})}$
      else
        $S \leftarrow S\setminus\{s\}$ with probability $p^-(S,s) = \frac{\pi_{\mathcal{C}}(S\setminus\{s\})}{\pi_{\mathcal{C}}(S)+\pi_{\mathcal{C}}(S\setminus\{s\})}$
      end if
    end if
  end while

Remarks. If α is less than 1 and independent of n, then the mixing time is nearly linear in n. The condition is conceptually similar to those in [156, 113]. The fast mixing requires both

α_1 and α_2, specifically, the change in probability when adding or deleting single element to neighboring subsets, to be small. Such notion is closely related to the *curvature* of discrete set functions. However, a poly-time check of such condition remains open.

5.3 Sampling from DIPMs with Matroid Base Constraints

In this section we consider sampling from an explicitly-constrained distribution $\pi_{\mathcal{C}}$ where $\mathcal{C}$ specifies a matroid base. We consider the following special cases of matroid bases[1]:

- *Uniform matroid:* $\mathcal{C} = \{S \subseteq \mathcal{V} \mid |S| = k\}$,
- *Partition matroid:* Given a partition $\mathcal{V} = \bigcup_{i=1}^{k} \mathcal{P}_i$, we allow sets that contain exactly one element from each $\mathcal{P}_i$: $\mathcal{C} = \{S \subseteq \mathcal{V} \mid |S \cap \mathcal{P}_i| = 1 \text{ for all } 1 \leq i \leq k\}$.

An important special case of a distribution with a uniform matroid constraint is the k-DPP [105]. Partition matroids are used in multilabel problems [185], and also in probabilistic diversity models [95].

Algorithm 15 Exchange Markov Chain for Matroid Bases

Require: set function F, β, matroid $\mathcal{C} \subseteq 2^{\mathcal{V}}$
  Initialize $S \in \mathcal{C}$
  while not mixed **do**
    Let $b = 1$ with probability 0.5
    if $b = 1$ **then**
      Draw $s \in S$ and $t \in \mathcal{V} \backslash S$ ($t \in \mathcal{P}(s) \setminus \{s\}$) uniformly at random
      if $S \cup \{t\} \backslash \{s\} \in \mathcal{C}$ **then**
        $S \leftarrow S \cup \{t\} \backslash \{s\}$ with probability $\frac{\pi_{\mathcal{C}}(S \cup \{t\} \backslash \{s\})}{\pi_{\mathcal{C}}(S) + \pi_{\mathcal{C}}(S \cup \{t\} \backslash \{s\})}$
      end if
    end if
  end while

The sampler is shown in Algorithm 15. At each iteration, we randomly select an item $s \in S$ and $t \in \mathcal{V} \backslash S$ such that the new set $S \cup \{t\} \backslash \{s\}$ satisfies $\mathcal{C}$, and swap them with certain probability. For uniform matroids, this means $t \in \mathcal{V} \backslash S$; for partition matroids, $t \in \mathcal{P}(s) \setminus \{s\}$ where $\mathcal{P}(s)$ is the part that s resides in. The fact that the chain has stationary

[1] Drawing even a uniform sample from the bases of an arbitrary matroid can be hard. MCMC on a uniform distribution over matroid bases is proved to be fast mixing only very recently [9].

distribution $\pi_{\mathcal{C}}$ can be inferred via detailed balance. Similar to the analysis in [86] for *unconstrained* sampling, the mixing time depends on a quantity that measures how much F deviates from linearity: $\zeta_F = \max_{S,T \in \mathcal{C}} |F(S) + F(T) - F(S \cap T) - F(S \cup T)|$. Our proof, however, differs from that of [86]. While they use canonical paths [52], we use multicommodity flows, which are more effective in our constrained setting.

Theorem 57. *Consider the chain in Algorithm 15. For the uniform matroid, $\tau_{X_0}(\varepsilon)$ is bounded as*

$$\tau_{X_0}(\varepsilon) \leq 4k(n-k)\exp(\beta(2\zeta_F))(\log \pi_{\mathcal{C}}(X_0)^{-1} + \log \varepsilon^{-1}); \tag{5.3.1}$$

For the partition matroid, the mixing time is bounded as

$$\tau_{X_0}(\varepsilon) \leq 4k^2 \max_i |\mathcal{P}_i| \exp(\beta(2\zeta_F))(\log \pi_{\mathcal{C}}(X_0)^{-1} + \log \varepsilon^{-1}). \tag{5.3.2}$$

Observe that if $\mathcal{P}_i$'s form an equipartition, i.e., $|\mathcal{P}_i| = n/k$ for all i, then the second bound becomes $\widetilde{\mathcal{O}}(kn)$. For $k = \mathcal{O}(\log n)$, the mixing times depend as $\mathcal{O}(n\text{polylog}(n)) = \widetilde{\mathcal{O}}(n)$ on n. For uniform matroids, the time is equally small if k is close to n. Finally, the time depends on the initialization, $\pi_{\mathcal{C}}(X_0)$. If F is monotone increasing, one may run a simple greedy algorithm to ensure that $\pi_{\mathcal{C}}(X_0)$ is large. If F is monotone submodular, this ensures that $\log \pi_{\mathcal{C}}(X_0)^{-1} = \mathcal{O}(\log n)$.

Our proof uses a multicommodity flow to upper bound the largest eigenvalue of the transition matrix. Concretely, let $\mathcal{H}$ be the set of all simple paths between states in the state graph of Markov chain, we construct a flow $f : \mathcal{H} \to \mathbb{R}^+$ that assigns a nonnegative flow value to any simple path between any two states (sets) $X, Y \in \mathcal{C}$. Each edge $e = (S, T)$ in the graph has a capacity $Q(e) = \pi_{\mathcal{C}}(S)P(S,T)$ where $P(S,T)$ is the transition probability from S to T. The total flow sent from X to Y must be $\pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)$: if $\mathcal{H}_{XY}$ is the set of all simple paths from X to Y, then we need $\sum_{p \in \mathcal{H}_{XY}} f(p) = \pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)$. Intuitively, the mixing time relates to the congestion in any edge, and the length of the paths. If there are many short paths $X \rightsquigarrow Y$ across which flow can be distributed, then mixing is fast. This intuition is captured in a fundamental theorem:

Theorem 58 (Multicommodity Flow [166]). *Let E be the set of edges in the transition graph, and $P(X,Y)$ the transition probability. Define*

$$\overline{\rho}(f) = \max_{e \in E} \frac{1}{Q(e)} \sum\nolimits_{p \ni e} f(p)\mathrm{len}(p),$$

where $\mathrm{len}(p)$ *the length of the path p. Then* $\lambda_{\max} \leq 1 - 1/\overline{\rho}(f)$.

With this property of multicommodity flow, we are ready to prove Thm. 57.

Proof. (Theorem 57) We sketch the proof for partition matroids; the full proofs is in Appendix C.2. For any two sets $X, Y \in \mathcal{C}$, we distribute the flow equally across all shortest paths $X \rightsquigarrow Y$ in the transition graph and bound the amount of flow through any edge $e \in E$.

Consider two arbitrary sets $X, Y \in \mathcal{C}$ with symmetric difference $|X \oplus Y| = 2m \leq 2k$, i.e., m elements need to be exchanged to reach from X to Y. However, these m steps are a valid path in the transition graph only if every set S along the way is in $\mathcal{C}$. The exchange property of matroids implies that this requirement is indeed true, so any shortest path $X \rightsquigarrow Y$ has length m. Moreover, there are exactly $m!$ such paths, since we can exchange the elements in $X \setminus Y$ in any order to reach at Y. Note that once we choose $s \in X \setminus Y$ to swap out, there is only one choice $t \in Y \setminus X$ to swap in, where t lies in the same part as s in the partition matroid, otherwise the constraint will be violated. Since the total flow is $\pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)$, each path receives $\pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)/m!$ flow.

Next, let $e = (S,T)$ be any edge on some shortest path $X \rightsquigarrow Y$; so $S, T \in \mathcal{C}$ and $T = S \cup \{j\} \setminus \{i\}$ for some $i, j \in \mathcal{V}$. Let $2r = |X \oplus S| < 2m$ be the length of the shortest path $X \rightsquigarrow S$, i.e., r elements need to be exchanged to reach from X to S. Similarly, $m - r - 1$ elements are exchanged to reach from T to Y. Since there is a path for every permutation of those elements, the ratio of the total flow $w_e(X,Y)$ that edge e receives from pair X, Y, and $Q(e)$, becomes

$$\frac{w_e(X,Y)}{Q(e)} \leq \frac{2r!(m-1-r)!kL}{m!Z_{\mathcal{C}}} \exp(2\beta\zeta_F)(\exp(\beta F(\sigma_S(X,Y))) + \exp(\beta F(\sigma_T(X,Y)))), \tag{5.3.3}$$

where we define $\sigma_S(X,Y) = X \oplus Y \oplus S = (X \cap Y \cap S) \cup (X \setminus (Y \cup S)) \cup (Y \setminus (X \cup S))$. To bound the total flow, we must count the pairs X, Y such that e is on their shortest path(s),

and bound the flow they send. We do this in two steps, first summing over all (X, Y)'s that share the upper bound (5.3.3) since they have the same difference sets $U_S = \sigma_S(X, Y)$ and $U_T = \sigma_T(X, Y)$, and then we sum over all possible U_S and U_T. For fixed U_S, U_T, there are $\binom{m-1}{r}$ pairs that share those difference sets, since the only freedom we have is to assign r of the $m - 1$ elements in $S \setminus (X \cap Y \cap S)$ to Y, and the rest to X. Hence, for fixed U_S, U_T. Appropriate summing and canceling then yields

$$\sum_{\substack{(X,Y):\, \sigma_S(X,Y)=U_S,\\ \sigma_T(X,Y)=U_T}} \frac{w_e(X,Y)}{Q(e)} \leq \frac{2kL}{Z_{\mathcal{C}}} \exp(2\beta\zeta_F)(\exp(\beta F(U_S)) + \exp(\beta F(U_T))). \tag{5.3.4}$$

Finally, we sum over all valid U_S (U_T is determined by U_S). One can show that any valid $U_S \in \mathcal{C}$, and hence $\sum_{U_S} \exp(\beta F(U_S)) \leq Z_{\mathcal{C}}$, and likewise for U_T. Hence, summing the bound (5.3.4) over all possible choices of U_S yields

$$\overline{\rho}(f) \leq 4kL \exp(2\beta\zeta_F) \max_p \operatorname{len}(p) \leq 4k^2 L \exp(2\beta\zeta_F),$$

where we upper bound the length of any shortest path by k, since $m \leq k$. Hence

$$\tau_{X_0}(\varepsilon) \leq 4k^2 L \exp(2\beta\zeta_F)(\log \pi(X_0)^{-1} + \log \varepsilon^{-1}). \qquad \square$$

For more restrictive constraints, there are fewer paths, and the bounds can become larger. Appendix C.2 shows the general dependence on k (as $k!$). It is also interesting to compare the bound on uniform matroid in Eq. (5.3.1) to that shown in [8] for a sub-class of distributions that satisfy the property of being homogeneous strongly Rayleigh. If $\pi_{\mathcal{C}}$ is homogeneous strongly Rayleigh, we have $\tau_{X_0}(\varepsilon) \leq 2k(n-k)(\log \pi_{\mathcal{C}}(X_0)^{-1} + \log \varepsilon^{-1})$. In our analysis, without additional assumptions on $\pi_{\mathcal{C}}$, we pay a factor of $2\exp(2\beta\zeta_F))$ for generality. This factor is one for some strongly Rayleigh distributions (e.g., if F is modular), but not for all.

5.4 Experiments

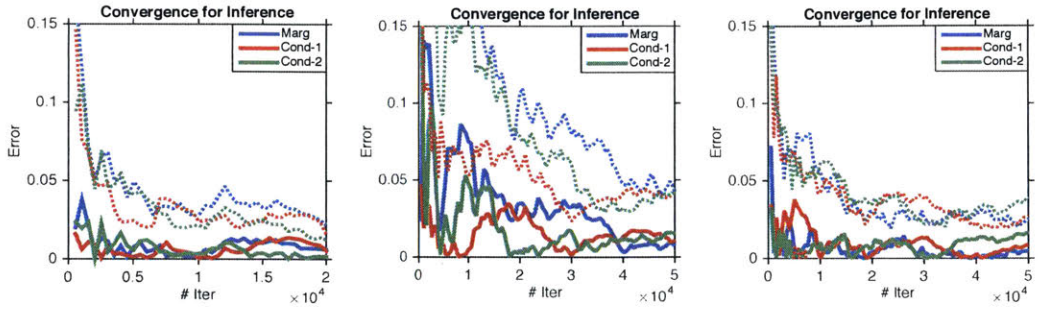
We next empirically study the dependence of sampling times on key factors that govern our theoretical bounds. In particular, we run Markov chains on chain-structured Ising models on a partition matroid base and DPPs on a uniform matroid, and consider estimating marginal and conditional probabilities of a single variable. To monitor the convergence of Markov chains, we use *potential scale reduction factor* (PSRF) [73, 38] that runs several chains in parallel and compares within-chain variances to between-chain variances. Typically, PSRF is greater than 1 and will converge to 1 in the limit; if it is close to 1 we empirically conclude that chains have mixed well. Throughout experiments we run 10 chains in parallel for estimations, and declare "convergence" at a PSRF of 1.05.

We first focus on small synthetic examples where we can compute exact marginal and conditional probabilities. We construct a 20-variable chain-structured Ising model as

$$\pi_{\mathcal{C}}(S) \propto \exp\left(\beta\left(\left(\delta\sum_{i=1}^{19} w_i(s_i \oplus s_{i+1})\right) + (1-\delta)|S|\right)\right)[\![S \in \mathcal{C}]\!],$$

where the s_i are 0-1 encodings of S, and the w_i are drawn uniformly randomly from $[0, 1]$. The parameters (β, δ) govern bounds on the mixing time via $\exp(2\beta\zeta_F)$; the smaller δ, the smaller ζ_F. $\mathcal{C}$ is a partition matroid of rank 5. We estimate conditional probabilities of one random variable conditioned on 0, 1 and 2 other variables and compare against the ground truth. We set (β, δ) to be $(1, 1)$, $(3, 1)$ and $(3, 0.5)$ and results are shown in Fig. 5-1. All marginals and conditionals converge to their true values, but with different speed. Comparing Fig. 5-1a against 5-1b, we observe that with fixed δ, increase in β slows down the convergence, as expected. Comparing Fig. 5-1b against 5-1c, we observe that with fixed β, decrease in δ speeds up the convergence, also as expected given our theoretical results. Appendix C.3.1 and C.3.2 illustrate the convergence of estimations under other (β, δ) settings.

We also check convergence on larger models. We use a DPP on a uniform matroid of rank 30 on the Ailerons data (`http://www.dcc.fc.up.pt/657~ltorgo/Regression/DataSets.html`) of size 200. Here, we do not have access to the ground truth, and hence plot the estimation mean with standard deviations among 10 chains in C-6.

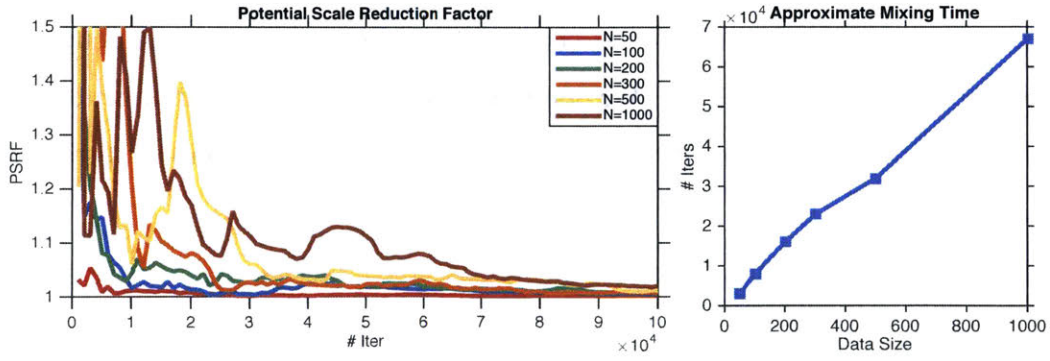


(a) $(\beta, \delta) = (1, 1)$ (b) $(\beta, \delta) = (3, 1)$ (c) $(\beta, \delta) = (3, 0.5)$

Figure 5-1: Convergence of marginal (`Marg`) and conditional (`Cond-1` and `Cond-2`, conditioned on 1 and 2 other variables) probabilities of a single variable in a 20-variable Ising model with different (β, δ). Full lines show the means and dotted lines the standard deviations of estimations.

We observe that the chains will eventually converge, i.e., the mean becomes stable and variance small. We also use PSRF to approximately judge the convergence. More results can be found in Appendix C.3.3.

Furthermore, the mixing time depends on the size n of the ground set. We use a DPP on Ailerons and vary n from 50 to 1000. Fig. 5-2a shows the PSRF from 10 chains for each setting. By thresholding PSRF at 1.05 in Fig. 5-2b we see a clearer dependence on n. At this scale, the mixing time grows almost linearly with n, indicating that this chain is efficient at least at small to medium scale.



(a) (b)

Figure 5-2: Empirical mixing time analysis when varying dataset sizes, (a) PSRF's for each set of chains, (b) Approximate mixing time obtained by thresholding PSRF at 1.05.

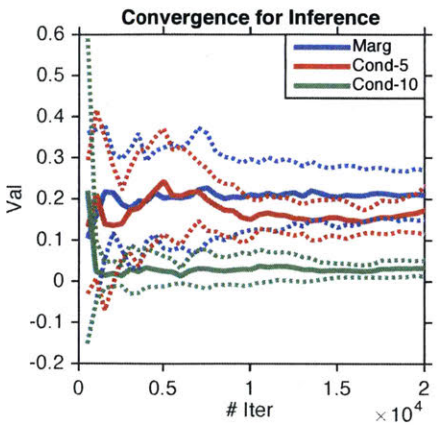


Figure 5-3: Convergence of marginal and conditional probabilities by DPP on uniform matroid

5.5 Summary

We presented theoretical results on Markov chain sampling for DIPMs subject to explicit constraints. For distributions with various explicit constraints we showed sufficient conditions for fast mixing. We show empirically that the dependencies of mixing times on various factors are consistent with our theoretical analysis.

There still exist many open problems in explicitly-constrained settings. Many bounds that we show depend on structural quantities (ζ_F or α) that may not always be easy to quantify in practice. It will be valuable to develop chains on special classes of distributions (like we did for SR) whose mixing time is independent of these factors. Moreover, we only considered cardinality or matroid bases as constraints, while several important settings such as knapsack constraints remain open. We defer the development of similar or better bounds, potentially with structural factors like $\exp(\beta\zeta_F)$, on specialized discrete probabilistic models as our future work.

Chapter 6

Conclusion and Open Problems

In this thesis, we study various diversity-inducing probability measures, including DPP, DVS, strongly Rayleigh measures and DIPMs with certain constraints. We show various efficient methods to sample from these distributions, and further show how we can apply them to core machine learning applications like Nyström method, kernel ridge regression and experimental design.

There still exist many open problems in this area. First, many poly-time mixing time bounds we have proved for either constrained or unconstrained DIPMs are conditional, namely, these mixing times are poly-time when DIPMs in consideration meets certain conditions. It would be interesting to further explore mathematical properties of certain classes of DIPMs (like we did for SR) to see if it is possible to come up with an unconditional mixing time bound. It will also be interesting to explore mathematical properties of other classes of DIPMs. Very recently in [9] the authors have shown that Markov chain on any *homogeneous Strong Log-Concave (SLC)* distribution is fast mixing. It is known that generating polynomials for both uniform distribution over matroid base or homogeneous SR is homogeneous SLC, thus the fast mixing MCMC follows in both cases. However, mixing times of MCMC on general SLC remains unknown. Further, whether adding a matroid base constraint or other forms of constraints on homogeneous SR/SLC will leave it in the class of homogeneous SR/SLC still remain open. Deeper mathematical properties, like whether homogeneous SLC is closed under operations like symmetric homogenization, is yet to be explored. We defer the further investigation to the future work.

Appendix A

Supplementary Experiments for Chapter 2

A.1 Kernel Approximation

Fig. A-1 shows the matrix norm relative error of various methods in kernel approximation on the remaining 7 datasets mentioned in the main text.

A.2 Approximated Kernel Ridge Regression

Fig. A-2 shows the training and test error of various methods for kernel ridge regression on the remaining 7 datasets.

A.3 Mixing of Markov Chain k-DPP

We first show the mixing of the Gibbs DPP-Nyström with 50 landmarks with different performance measures: relative spectral norm error, training error and test error of kernel ridge regression in Fig. A-3.

We also show corresponding results with respect to 100 and 200 landmarks in Fig. A-4 and Fig. A-5, so as to illustrate that for varying number of landmarks the chain is indeed fast mixing and will give reasonably good result within a small number of iterations.

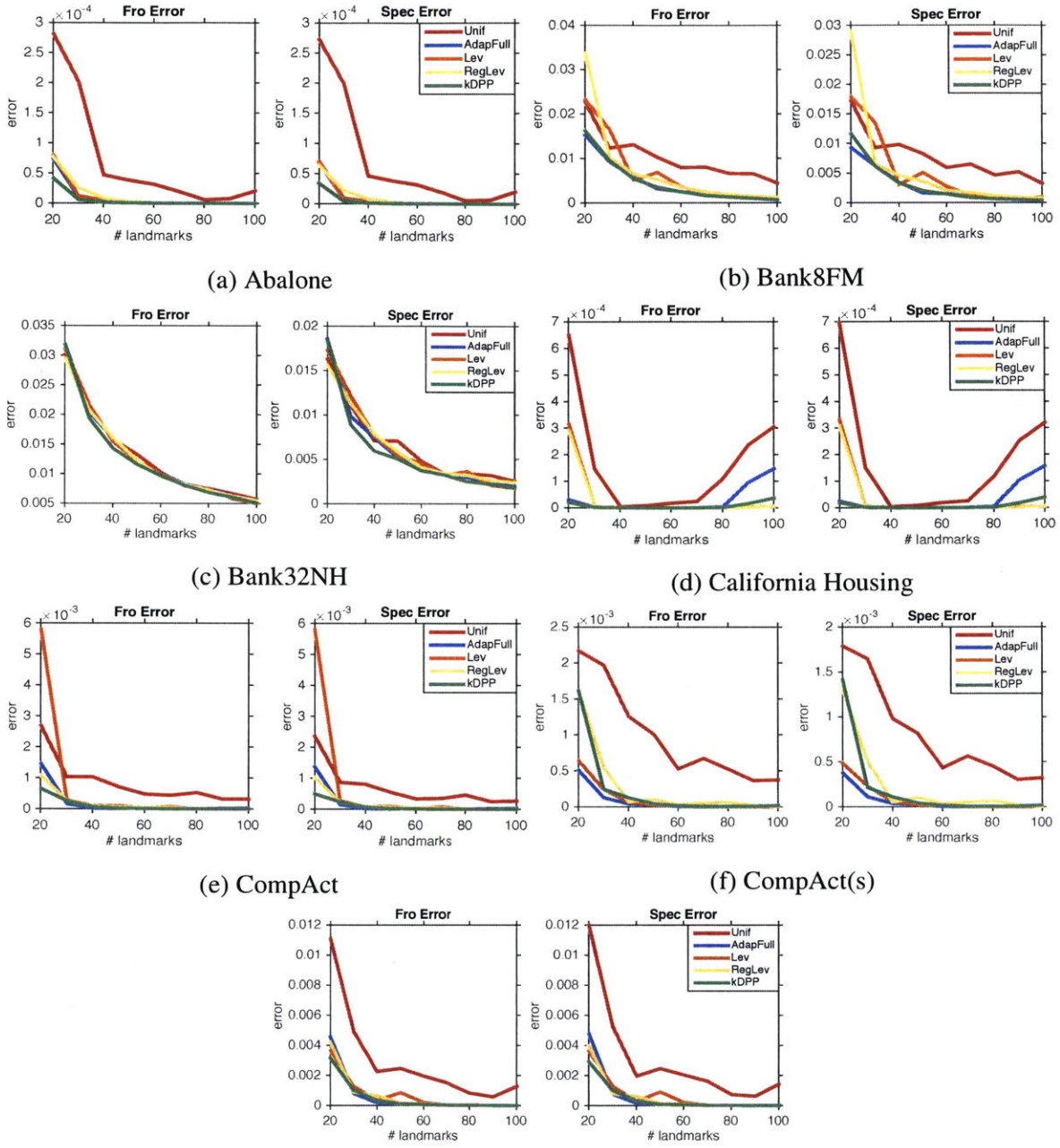
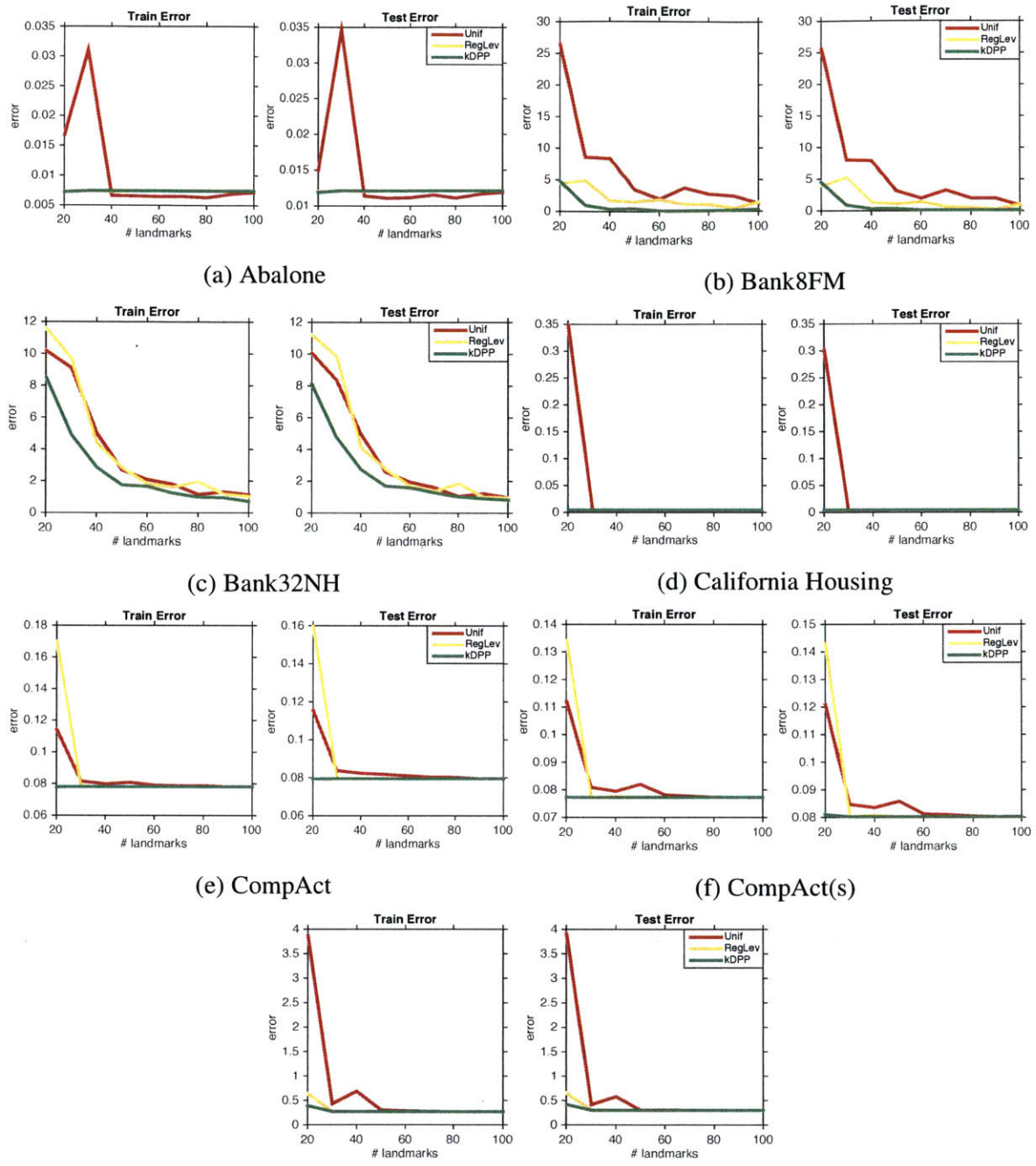


Figure A-1: Relative Frobenius norm and spectral norm error achieved by different kernel approximation algorithms on the remaining 7 data sets.

A.4 Running Time Analysis

We next show time-error trade-offs for various sampling methods on small and larger datasets with respect to Fnorm and 2norm errors. We sample 20 landmarks from Ailerons dataset of



(a) Abalone

(b) Bank8FM

(c) Bank32NH

(d) California Housing

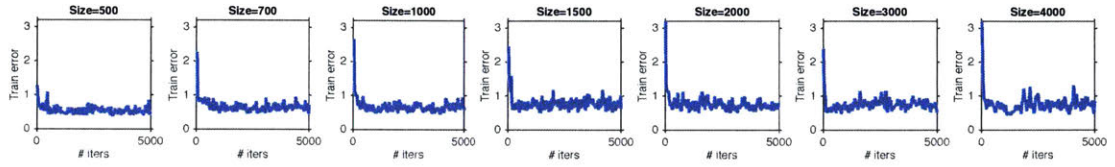
(e) CompAct

(f) CompAct(s)

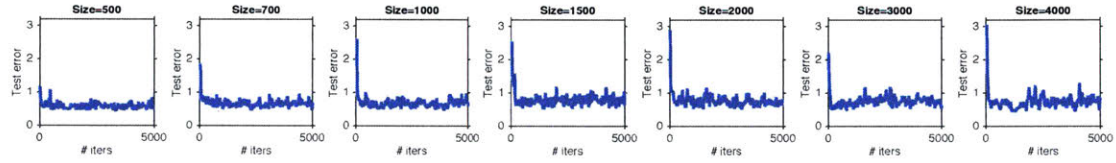
(g) Elevators

Figure A-2: Training and test error achieved by different Nyström kernel ridge regression algorithms on the remaining 7 regression datasets.

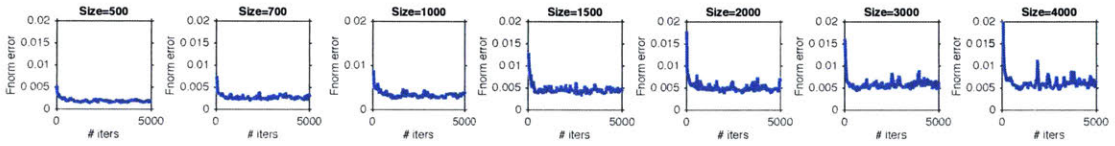
size 4,000 and California Housing of size 12,000. The result is shown in Figure A-6 and Figure A-7 and similar trends as the example results in the main text could be spotted: on small scale dataset (size 4,000) kDPP get very good time-error trade-off. It is more efficient



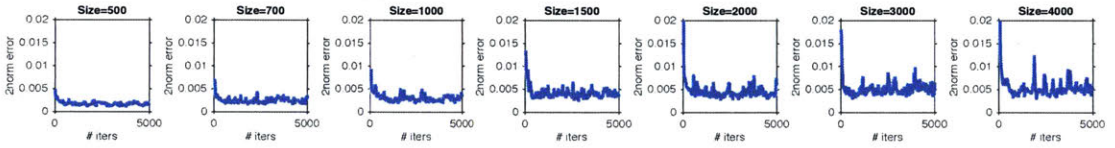
(a) Training error



(b) Test error

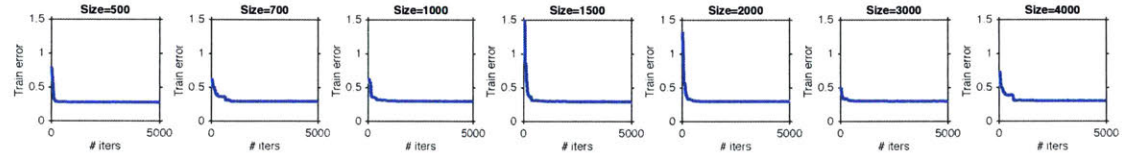


(c) Relative Frobenius norm error

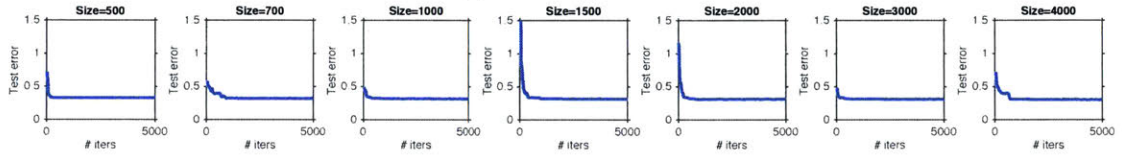


(d) Relative Spectral norm error

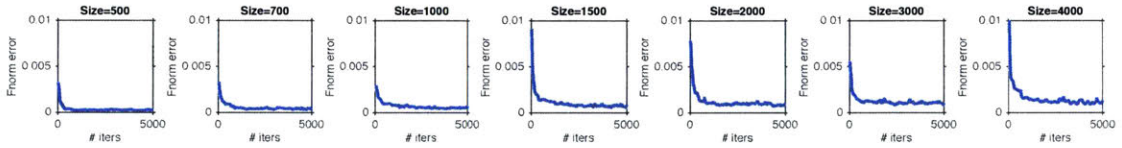
Figure A-3: Performance of Markov chain DPP-Nyström with 50 landmarks on Ailerons. Runs for 5,000 iterations.



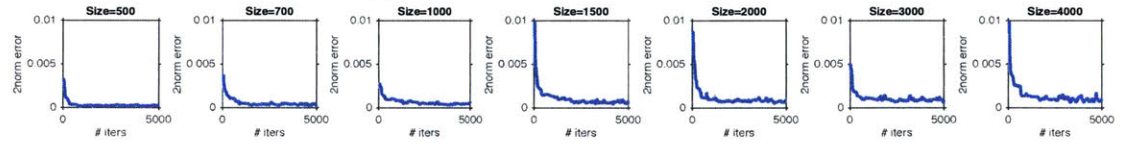
(a) Training error



(b) Test error

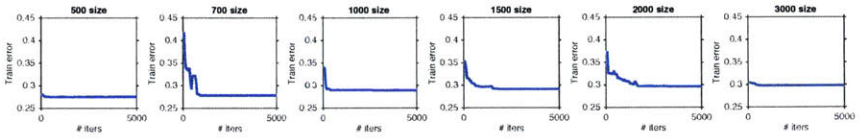


(c) Relative Frobenius norm error

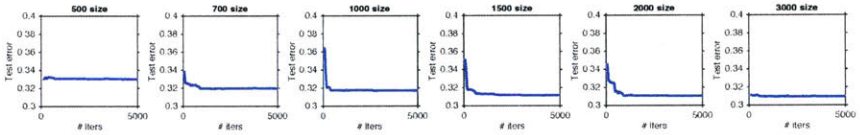


(d) Relative Spectral norm error

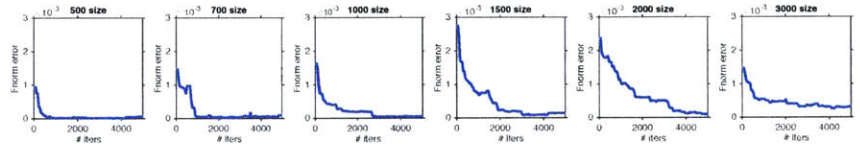
Figure A-4: Performance of Markov chain DPP-Nyström with 100 landmarks on Ailerons. Runs for 5,000 iterations.



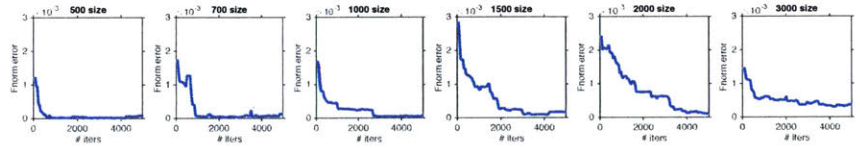
(a) Training error



(b) Test error

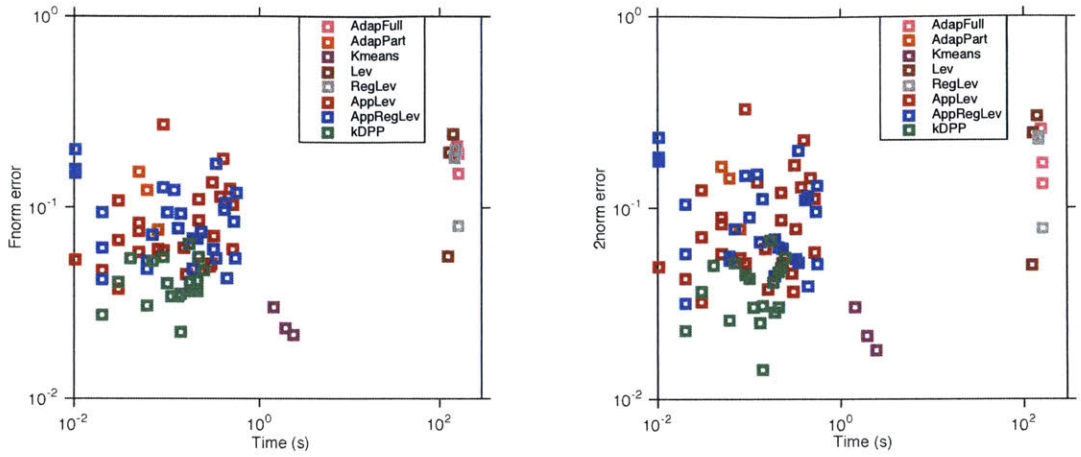


(c) Relative Frobenius norm error



(d) Relative Spectral norm error

Figure A-5: Performance of Markov chain DPP-Nyström with 200 landmarks on Ailerons. Runs for 5,000 iterations.

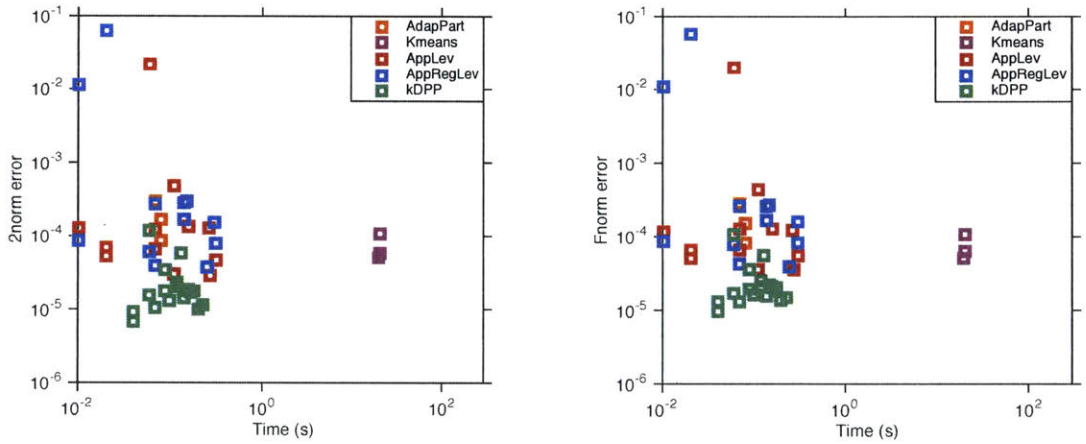


(a) Fnorm Error vs. Time

(b) 2norm Error vs. Time

Figure A-6: Time-Error tradeoff with 20 landmarks on Ailerons of size 4,000. Time and Errors shown in log-scale.

than `Kmeans`, though the error is a bit larger. While on larger dataset (size 12,000) the efficiency is further enhanced while the error is even lower than `Kmeans`. It also have lower variances in both cases compared to `AppLev` and `AppRegLev`. Overall, on larger dataset we obtain the best time-error trade-off with kDPP.



(a) 2norm Error vs. Time

(b) Training Error vs. Time

Figure A-7: Time-Error tradeoff with 20 landmarks on California Housing of size 12,000. Time and Errors shown in log-scale. We didn't include `AdapFull`, `Lev` and `RegLev` due to their inefficiency on larger datasets.

Appendix B

Supplementary Proofs and Experiments for Chapter 4

B.1 Partition Function

We recall two easily verified facts about determinants that will be useful in our analysis:

$$\det(K + uv^\top) = \det(K)(1 + u^\top K^{-1} v), \quad \text{for } K \in \mathrm{GL}_n(\mathbb{R}), \tag{B.1.1}$$

$$a^{m-n} \det(AA^\top + aI_n) = \det(A^\top A + aI_m), \quad \text{for } A \in \mathbb{R}^{n\times m} \ (n \le m), \text{ and } a > 0. \tag{B.1.2}$$

The first one is known as matrix determinant lemma.

The partition function of $P(\cdot; A)$, happens to have a pleasant closed-form formula. Although this formula is known [13], and follows immediately by an application of the Cauchy-Binet identity, we present an alternative proof based on the perturbation argument for its conceptual value and subsequent use.

Theorem 59 (Partition Function [13]). *Given $A \in \mathbb{R}^{n\times m}$ where $r(A) = n$ and $n \le |S| = k \le m$, we have*

$$\sum_{|S|=k, S\subseteq[m]} \det(A_S A_S^\top) = \begin{pmatrix} m-n \\ k-n \end{pmatrix} \det(AA^\top). \tag{B.1.3}$$

Proof. First note that for $n \leq |S| = k \leq m$ and any $\varepsilon > 0$, by (B.1.2) we have

$$\det(A_S A_S^\top + \varepsilon I_n) = \frac{1}{\varepsilon^{k-n}} \det(A_S^\top A_S + \varepsilon I_k)$$

Taking limits as $\varepsilon \to 0$ on both sides we have

$$\det(A_S A_S^\top) = \lim_{\varepsilon \to 0} \det(A_S A_S^\top + \varepsilon I_n) = \lim_{\varepsilon \to 0} \frac{1}{\varepsilon^{k-n}} \det(A_S^\top A_S + \varepsilon I_k).$$

Let us focus on $\det(A_S^\top A_S + \varepsilon I_k)$. We construct an identity matrix $I_m \in \mathbb{R}^{m \times m}$, then we have

$$\begin{aligned} \det(A_S^\top A_S + \varepsilon I_k) &= \det(A_S^\top A_S + \varepsilon I_S^\top I_S) = \det(A_S^\top A_S + (\sqrt{\varepsilon} I_S)^\top \sqrt{\varepsilon} I_S) \\ &= \det\left(\begin{bmatrix} A_S \\ \sqrt{\varepsilon}(I_m)_S \end{bmatrix}^\top \begin{bmatrix} A_S \\ \sqrt{\varepsilon}(I_m)_S \end{bmatrix} \right) \propto \widehat{P}\left(S; \begin{bmatrix} A \\ \sqrt{\varepsilon} I_m \end{bmatrix} \right). \end{aligned} \tag{B.1.4}$$

In other words, this value is proportional to the probability of sampling columns from $\begin{bmatrix} A \\ \sqrt{\varepsilon} I_m \end{bmatrix}$ using volume sampling. Therefore, using the definition of e_k we have

$$\begin{aligned} \frac{1}{\varepsilon^{k-n}} \sum_{|S|=k, S \subseteq [m]} \det(A_S^\top A_S + \varepsilon I_k) &= \frac{1}{\varepsilon^{k-n}} e_k(A^\top A + \varepsilon I_m) \\ &= \frac{1}{\varepsilon^{k-n}} e_k(Diag([(\sigma_1^2(A) + \varepsilon), (\sigma_2^2(A) + \varepsilon), \ldots, (\sigma_n^2(A) + \varepsilon), \varepsilon, \ldots, \varepsilon])) \\ &= \binom{m-n}{k-n} \prod_{i=1}^n (\sigma_i^2(A) + \varepsilon) + O(\varepsilon). \end{aligned}$$

Now taking the limit as $\varepsilon \to 0$ we obtain

$$\sum_{|S|=k, S \subseteq [m]} \det(A_S A_S^\top) = \lim_{\varepsilon \to 0} \binom{m-n}{k-n} \prod_{i=1}^n (\sigma_i^2(A) + \varepsilon) + O(\varepsilon) = \binom{m-n}{k-n} \det(AA^\top).$$

□

B.2 Marginal Probability

Proof. The marginal probability of a set $T \subseteq [m]$ for dual volume sampling is

$$P(T \subseteq S; A) = \frac{\sum_{S \supseteq T, |S|=k} \det(A_S A_S^\top)}{\sum_{|S'|=k} \det(A_{S'} A_{S'}^\top)}.$$

Theorem 59 shows how to compute the denominator, thus our main effort is devoted to the nominator. We have

$$\sum_{S \supseteq T, |S|=k} \det(A_S A_S^\top) = \sum_{R \cap T = \emptyset, |R|=k-|T|} \det(A_{T \cup R} A_{T \cup R}^\top)$$

Using the ε-trick we have

$$\begin{aligned}\sum_{R \cap T = \emptyset, |R|=k-|T|} \det(A_{T \cup R} A_{T \cup R}^\top) &= \lim_{\varepsilon \to 0} \sum_{R \cap T = \emptyset, |R|=k-|T|} \det(A_{T \cup R} A_{T \cup R}^\top + \varepsilon I_n) \\ &= \lim_{\varepsilon \to 0} \frac{1}{\varepsilon^{k-n}} \sum_{R \cap T = \emptyset, |R|=k-|T|} \det(A_{T \cup R}^\top A_{T \cup R} + \varepsilon I_k).\end{aligned}$$

By decomposing $\det(A_{T \cup R}^\top A_{T \cup R} + \varepsilon I_k)$ we have

$$\begin{aligned}&\det(A_{T \cup R}^\top A_{T \cup R} + \varepsilon I_k) \\ &\quad = \det(A_T^\top A_T + \varepsilon I_{|T|}) \det\left(A_R^\top A_R + \varepsilon I_{|R|} - A_R^\top A_T (A_T^\top A_T + \varepsilon I_{|T|})^{-1} A_T^\top A_R\right).\end{aligned}$$

Now we let $A_T = Q_T \Sigma_T V_T^\top$ be the singular value decomposition of A_T where $Q_T \in \mathbb{R}^{n \times r(A_T)}$, $\Sigma_T \in \mathbb{R}^{r(A_T) \times |T|}$ and $V_T \in \mathbb{R}^{|T| \times |T|}$. Plugging the decomposition in the equation

we obtain

$$
\begin{aligned}
A_R^\top A_T (A_T^\top A_T + \varepsilon I_{|T|})^{-1} A_T^\top A_R &= A_R^\top Q_T \Sigma_T V_T^\top (V_T \Sigma_T^\top \Sigma_T V_T^\top + \varepsilon I_{|T|})^{-1} V_T \Sigma_T^\top Q_T^\top A_R \\
&= A_R^\top Q_T \Sigma_T (\Sigma_T^\top \Sigma_T + \varepsilon I_{|T|})^{-1} \Sigma_T^\top Q_T^\top A_R \\
&= A_R^\top Q_T \begin{bmatrix} \frac{\sigma_1^2(A_T)}{\sigma_1^2(A_T)+\varepsilon} & 0 & \dots & 0 \\ 0 & \frac{\sigma_2^2(A_T)}{\sigma_2^2(A_T)+\varepsilon} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{\sigma_{r(A_T)}^2(A_T)}{\sigma_{r(A_T)}^2(A_T)+\varepsilon} \end{bmatrix} Q_T^\top A_R \\
&= A_R^\top Q_T Q_T^\top A_R - \varepsilon A_R^\top Q_T \begin{bmatrix} \frac{1}{\sigma_1^2(A_T)+\varepsilon} & 0 & \dots & 0 \\ 0 & \frac{1}{\sigma_2^2(A_T)+\varepsilon} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{1}{\sigma_{r(A_T)}^2(A_T)+\varepsilon} \end{bmatrix} Q_T^\top A_R.
\end{aligned}
$$

Thus it follows that

$$
\begin{aligned}
&A_R^\top A_R + \varepsilon I_{|R|} - A_R^\top A_T (A_T^\top A_T + \varepsilon I_{|T|})^{-1} A_T^\top A_R \\
&\quad = A_R^\top (I - Q_T Q_T^\top) A_R + \varepsilon A_R^\top Q_T \begin{bmatrix} \frac{1}{\sigma_1^2(A_T)+\varepsilon} & 0 & \dots \\ 0 & \frac{1}{\sigma_2^2(A_T)+\varepsilon} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} Q_T^\top A_R + \varepsilon I_{|R|} \\
&\quad = B_R^\top B_R + \varepsilon C_R^\top C_R + \varepsilon I_{|R|},
\end{aligned}
$$

where B_R is the projection of columns of A_R on the orthogonal space of columns of A_T. Let $Q_T^\perp \in \mathbb{R}^{n \times (n - r(A_T))}$ be the complement column space of Q_T, then we have $B_R = (Q_T^\perp)^\top A_R \in \mathbb{R}^{(n-r(A_T)) \times |R|}$. Moreover,

$$
C_R = \begin{bmatrix} \frac{1}{\sqrt{\sigma_1^2(A_T)+\varepsilon}} & 0 & \dots \\ 0 & \frac{1}{\sqrt{\sigma_2^2(A_T)+\varepsilon}} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} Q_T^\top A_R \in \mathbb{R}^{r(A_T) \times |R|}.
$$

We further let $B_{T_c} = (Q_T^{\perp})^\top A_{T_c} \in \mathbb{R}^{(n-r(A_T))\times(m-|T|)}$ and

$$C_{T_c} = \begin{bmatrix} \frac{1}{\sqrt{\sigma_1^2(A_T)+\varepsilon}} & 0 & \dots \\ 0 & \frac{1}{\sqrt{\sigma_2^2(A_T)+\varepsilon}} & \dots \\ \vdots & \vdots & \ddots \end{bmatrix} Q_T^\top A_{T_c} \in \mathbb{R}^{r(A_T)\times(m-|T|)}$$

where $T_c = [m]\backslash T$. Then we have

$$\begin{aligned} &\sum_{R\cap T=\emptyset, |R|=k-|T|} \det(A_{T\cup R}^\top A_{T\cup R} + \varepsilon I_k) \\ &\quad = \det(A_T^\top A_T + \varepsilon I_{|T|}) \sum_{R\cap T=\emptyset, |R|=k-|T|} \det(B_R^\top B_R + \varepsilon C_R^\top C_R + \varepsilon I_{|R|}) \\ &\quad = \det(A_T^\top A_T + \varepsilon I_{|T|}) \times e_{k-|T|}\left(\begin{bmatrix} B_{T_c} \\ \sqrt{\varepsilon}U_{T_c} \\ \sqrt{\varepsilon}C_{T_c} \end{bmatrix}\begin{bmatrix} B_{T_c} \\ \sqrt{\varepsilon}U_{T_c} \\ \sqrt{\varepsilon}C_{T_c} \end{bmatrix}^\top\right) \end{aligned}$$

where we construct an orthonormal matrix $U \in \mathbb{R}^{(m-|T|)\times(m-|T|)}$ whose columns are basis vectors. Since we are free to chose any orthonormal U, we simply let it be I. Let $W_{T_c} = \begin{bmatrix} I_{T_c} \\ C_{T_c} \end{bmatrix}$, we have

$$\begin{aligned} \left(\begin{bmatrix} B_{T_c} \\ \sqrt{\varepsilon}U_{T_c} \\ \sqrt{\varepsilon}C_{T_c} \end{bmatrix}\begin{bmatrix} B_{T_c} \\ \sqrt{\varepsilon}U_{T_c} \\ \sqrt{\varepsilon}C_{T_c} \end{bmatrix}^\top\right) &= \left(\begin{bmatrix} B_{T_c} \\ \sqrt{\varepsilon}W_{T_c} \end{bmatrix}\begin{bmatrix} B_{T_c} \\ \sqrt{\varepsilon}W_{T_c} \end{bmatrix}^\top\right) \\ &= F_{T_c} \in \mathbb{R}^{(m+n-|T|)\times(m+n-|T|)} \end{aligned}$$

The properties of characteristic polynomials imply that

$$\begin{aligned} e_{k-|T|}(F_{T_c}) &= \sum_{|S|=k-|T|} \det((F_{T_c})_{S,S}) \\ &= \sum_{S_1,S_2} \det((F_{T_c})_{S_1,S_1}) \det((F_{T_c})_{S_2,S_2} - (F_{T_c})_{S_2,S_1}(F_{T_c})_{S_1,S_1}^{-1}(F_{T_c})_{S_1,S_2}) \end{aligned}$$

where $S_1 = S \cap [r(B_{T_c})]$ and $S_2 = [m+n-|T|]\backslash S_1$. Further we have

$$\sum_{S_1,S_2} \det((F_{T_c})_{S_1,S_1}) \det((F_{T_c})_{S_2,S_2} - (F_{T_c})_{S_2,S_1}(F_{T_c})^{-1}_{S_1,S_1}(F_{T_c})_{S_1,S_2})$$
$$= \sum_{S_1,S_2} \varepsilon^{k-|T|-|S_1|} \det((B_{T_c})_{S_1}(B_{T_c})^\top_{S_1}) \times$$
$$\det((W_{T_c})_{S_2}(W_{T_c})^\top_{S_2} - (W_{T_c})_{S_2}(B_{T_c})^\top_{S_1}((B_{T_c})_{S_1}(B_{T_c})^\top_{S_1})^{-1}(B_{T_c})_{S_1}(W_{T_c})^\top_{S_2})$$

Hence it follows that

$$\lim_{\varepsilon\to 0}\frac{1}{\varepsilon^{k-n}} \sum_{R\cap T=\emptyset, |R|=k-|T|} \det(A^\top_{T\cup R}A_{T\cup R} + \varepsilon I_k) = \lim_{\varepsilon\to 0}\frac{1}{\varepsilon^{k-n}} \det(A^\top_T A_T + \varepsilon I_{|T|}) \times e_{k-|T|}(F_{T_c})$$
$$= \lim_{\varepsilon\to 0}\frac{1}{\varepsilon^{k-n}} \varepsilon^{|T|-r(A_T)} \left[\prod_{i=1}^{r(A_T)} (\sigma_i^2(A_T) + \varepsilon)\right] \times$$
$$\sum_{|S|=k-|T|} \varepsilon^{k-|T|-|S_1|} \det((B_{T_c})_{S_1}(B_{T_c})^\top_{S_1}) \times$$
$$\det((W_{T_c})_{S_2}(W_{T_c})^\top_{S_2} - (W_{T_c})_{S_2}(B_{T_c})^\top_{S_1}((B_{T_c})_{S_1}(B_{T_c})^\top_{S_1})^{-1}(B_{T_c})_{S_1}(W_{T_c})^\top_{S_2})$$

(Since $r(A_T) + r(B_{T_c}) = n$ and $|S_1| \le r(B_{T_c})$)

$$= \lim_{\varepsilon\to 0}\frac{1}{\varepsilon^{k-n}} \varepsilon^{|T|-r(A_T)} \left[\prod_{i=1}^{r(A_T)} (\sigma_i^2(A_T) + \varepsilon)\right] \times$$
$$\sum_{|S|=k-|T|} \varepsilon^{k-|T|-r(B_{T_c})} \det(B_{T_c}B^\top_{T_c}) \det((W_{T_c})_{S_2}(W_{T_c})^\top_{S_2} - (W_{T_c})_{S_2}B^\top_{T_c}(B_{T_c}B^\top_{T_c})^{-1}B_{T_c}(W_{T_c})^\top_{S_2}) + O(\varepsilon)$$
$$= \left[\prod_{i=1}^{r(A_T)} \sigma_i^2(A_T)\right] \times \left[\prod_{j=1}^{r(B_{T_c})} \sigma_j^2(B_{T_c})\right] \sum_{S_2} \det((W_{T_c})_{S_2}(W_{T_c})^\top_{S_2} - (W_{T_c})_{S_2}B^\top_{T_c}(B_{T_c}B^\top_{T_c})^{-1}B_{T_c}(W_{T_c})^\top_{S_2})$$

where $S_2 \subseteq [m+n-|T|]\backslash[r(B_{T_c})]$ and $|S_2| = k - |T| - r(B_{T_c})$.

Let $Q_{B_{T_c}} \mathrm{diag}(\sigma_i^2(B_{T_c}))Q^\top_{B_{T_c}}$ be the eigenvalue decomposition of $B^\top_{T_c}B_{T_c}$ where $Q_{B_{T_c}} \in$

$\mathbb{R}^{|T_c|\times r(B_{T_c})}$. Further, let $Q^{\perp}_{B_{T_c}}$ be the complement column space of $Q_{B_{T_c}}$, thus we have

$$\begin{bmatrix} Q^{\top}_{B_{T_c}} \\ (Q^{\perp}_{B_{T_c}})^{\top} \end{bmatrix} \begin{bmatrix} Q_{B_{T_c}} & Q^{\perp}_{B_{T_c}} \end{bmatrix} = I_{|T_c|} = I_{n-|T|}$$

Then for any $S_2 \subseteq [m+n-|T|]\backslash[r(B_{T_c})]$ we have

$$\begin{aligned}\det((W_{T_c})_{S_2}(W_{T_c})^{\top}_{S_2} - (W_{T_c})_{S_2}B^{\top}_{T_c}(B_{T_c}B^{\top}_{T_c})^{-1}B_{T_c}(W_{T_c})^{\top}_{S_2}) &= \det(W_{S_2}(I_{n-|T|} - Q_{B_{T_c}}Q^{\top}_{B_{T_c}})(W_{T_c})^{\top}_{S_2}) \\ &= \det((W_{T_c})_{S_2}(Q^{\perp}_{B_{T_c}}(Q^{\perp}_{B_{T_c}})^{\top})(W_{T_c})^{\top}_{S_2})\end{aligned}$$

It follows that

$$\begin{aligned}\sum_{S_2}\det(W_{S_2}(W_{T_c})^{\top}_{S_2} - (W_{T_c})_{S_2}B^{\top}_{T_c}(B_{T_c}B^{\top}_{T_c})^{-1}B_{T_c}(W_{T_c})^{\top}_{S_2}) &= e_{k-|T|-r(B_{T_c})}(W_{T_c}((Q^{\perp}_{B_{T_c}})^{\top}Q^{\perp}_{B_{T_c}})W^{\top}_{T_c}) \\ &= E_T\end{aligned}$$

Combining all the above derivations, we obtain that

$$\Pr(T \subseteq S | S \sim P(S;A)) = \frac{\left[\prod_{i=1}^{r(A_T)}\sigma_i^2(A_T)\right] \times \left[\prod_{j=1}^{r(B_{T_c})}\sigma_j^2(B_{T_c})\right] \times \Gamma_T}{\begin{pmatrix} n-m \\ k-m \end{pmatrix}\det(AA^{\top})}.$$

□

B.3 Approximate Sampling via Volume Sampling

Corollary 60 (Approximate DVS via Random Projection). *For any $\varepsilon > 0$ and $\delta_2 > 0$ there is an algorithm that, in time $\widetilde{\mathcal{O}}(\frac{k^2nm}{\delta_2^2} + \frac{k^7m}{\delta_2^6})$, samples a subset from an approximate distribution $\widetilde{P}(\cdot;A)$ with $\delta_1 = \max_{|S|=k}(1+\frac{\varepsilon}{\sigma^2_{\min}(A_S)})^n - 1 \approx \frac{n\varepsilon}{\sigma^2_{\min}(A_S)}$ and*

$$\frac{\widetilde{P}(S;A)}{(1+\delta_1)(1+\delta_2)} \leq P(S;A) \leq (1+\delta_1)(1+\delta_2)\widetilde{P}(S;A); \quad \forall S \subseteq [m].$$

It may happen in practice that $n \ll m$ but k is of the same order as n. In such case we can transform the dual volume sampling to slightly distorted volume sampling based on (B.1.2) and then take the advantage of determinant-preserving projections to accelerate the sampling procedure.

Concretely, instead of sampling column subset S with probability proportional to $\det(A_S A_S^\top)$, we sample with probability proportional to a distorted value $\det(A_S A_S^\top + \varepsilon I_n)$ for small $\varepsilon > 0$. Denoting this distorted distribution as $P_\varepsilon(S; A)$, we have

$$P_\varepsilon(S; A) = \frac{1}{\varepsilon^{k-n}} \det(A_S^\top A_S + \varepsilon I_k) = \frac{1}{\varepsilon^{k-n}} \prod_{i=1}^{n} (\sigma_i^2(A_S) + \varepsilon).$$

Letting $\sigma_{\min}(A_S) > 0$ be the minimum singular value, we have

$$1 \leq \frac{\prod_{i=1}^{n}(\sigma_i^2(A_S) + \varepsilon)}{\prod_{i=1}^{n}(\sigma_i^2(A_S))} \leq (1 + \frac{\varepsilon}{\sigma_{\min}^2(A_S)})^n.$$

We further let

$$\delta_1 = \max_{|S|=k}(1 + \frac{\varepsilon}{\sigma_{\min}^2(A_S)})^n - 1 \approx \frac{n\varepsilon}{\sigma_{\min}^2(A_S)},$$

when ε sufficiently small. Sampling from P_ε will yield $(1 + \delta_1)$-approximate dual volume sampling (in the sense of [49] and our Theorem 60). We can sample from P_ε via *volume sampling* with distribution $\widehat{P}(S; \begin{bmatrix} A \\ \sqrt{\varepsilon} I_m \end{bmatrix})$. With the volume sampling algorithm proposed in [49], the resulting running time would be $\widetilde{\mathcal{O}}(km^4)$.

To accelerate sampling procedure, we consider random projection techniques that preserve volumes. [130] showed that Gaussian random projections indeed preserve volumes as we need:

Theorem 61 (Random Projection [130]). *For any $X \in \mathbb{R}^{n \times m}$, $1 \leq k \leq m$ and $0 < \delta_2 \leq 1/2$, the random Gaussian projection of $\mathbb{R}^m \to \mathbb{R}^d$ where*

$$d = \mathcal{O}\left(\frac{k^2 \log n}{\delta_2^2}\right),$$

satisfies

$$\det(X_S^\top X_S) \leq \det(\widetilde{X}_S^\top \widetilde{X}_S) \leq (1+\delta_2)\det(X_S^\top X_S) \tag{B.3.1}$$

for all $S \subseteq [n]$ *and* $|S| \leq k$ *where* $\widetilde{X}$ *is the projected matrix.*

This theorem completes what we need to prove Corollary 60.

Proof. (Corollary 60) The idea is to project $\begin{bmatrix} A \\ \sqrt{\varepsilon} I_m \end{bmatrix}$ to a lower-dimensional space in a way that the values for submatrix determinants are preserved up to a small multiplicative factor. Then we perform volume sampling. We project columns of $\begin{bmatrix} A \\ \sqrt{\varepsilon} I_m \end{bmatrix}$, which is in $\mathbb{R}^{m+n}$, to vectors in $\mathbb{R}^d$ where $d = \mathcal{O}\left(\frac{k^2 \log m}{\delta_2^2}\right)$ so as to achieve a $(1+\delta_2)$ approximation by Theorem 61. Let G be a $d \times (m+n)$-dimensional i.i.d. Gaussian random matrix, then we have

$$G\begin{bmatrix} A \\ \sqrt{\varepsilon} I \end{bmatrix} = G_A A + \sqrt{\varepsilon} G'_A \tag{B.3.2}$$

where $G_A \in \mathbb{R}^{d\times n}$ and $G'_A \in \mathbb{R}^{d\times n}$ are two independent Gaussian random matrix. The projected matrix can be computed in $\mathcal{O}(dnm) = \widetilde{\mathcal{O}}(k^2 nmn/\delta_2^2)$ time. After that, if we use volume sampling algorithm proposed in [49] the resulting running time would be $\mathcal{O}(kd^3 m) = \widetilde{\mathcal{O}}(k^7 m/\delta_2^6)$. Thus the total running time would be $\widetilde{\mathcal{O}}(\frac{k^2 nm}{\delta_2^2} + \frac{k^7 m}{\delta_2^6})$. □

Remarks. An interesting observation is that the resulting running time is independent of δ_1, which means one can set ε arbitrarily small so as to make the approximation in the first step as accurate as possible, without affecting the running time. However, in practice, a very small ε can result in numerical problems. In addition, the dimensionality reduction is only efficient if $d < m + n$.

B.4 Conditional Expectation

Proof. We use A^j denote the matrix $A_{[n]\setminus\{j\},:}$, namely matrix A with row j deleted. We have

$$\begin{aligned}
&\mathbb{E}\left[\|A_S^\dagger\|_F^2 \mid s_1 = i_1, \ldots, s_{t-1} = i_{t-1}\right] \\
&= \sum_{(i_t,\ldots,i_k)\in[m]^{k-t+1}} \|A_S^\dagger\|_F^2 \overrightarrow{P}(s_1 = i_1, \ldots, s_k = i_k; A \mid s_1 = i_1, \ldots, s_{t-1} = i_{t-1}) \\
&= \sum_{(i_t,\ldots,i_k)\in[m]^{k-t+1}} \|A_S^\dagger\|_F^2 \frac{\overrightarrow{P}(s_1 = i_1, \ldots, s_k = i_k; A)}{\overrightarrow{P}(s_1 = i_1, \ldots, s_{t-1} = i_{t-1}; A)} \\
&= \frac{\sum_{(i_t,\ldots,i_k)\in[m]^{k-t+1}} \det(A_{\{i_1,\ldots,i_k\}} A_{\{i_1,\ldots,i_k\}}^\top) \|A_{\{i_1,\ldots,i_k\}}^\dagger\|_F^2}{\sum_{(i_t,\ldots,i_k)\in[m]^{k-t+1}} \det(A_{\{i_1,\ldots,i_k\}} A_{\{i_1,\ldots,i_k\}}^\top)} \\
&= \frac{\sum_{j=1}^n \sum_{(i_t,\ldots,i_k)\in[m]^{k-t+1}} \det(A^j_{\{i_1,\ldots,i_k\}} (A^j_{\{i_1,\ldots,i_k\}})^\top)}{\sum_{(i_t,\ldots,i_k)\in[m]^{k-t+1}} \det(A_{\{i_1,\ldots,i_k\}} A_{\{i_1,\ldots,i_k\}}^\top)}
\end{aligned}$$

While the denominator is the (unnormalized) marginal distribution $P(T \subseteq S \mid S \sim P(S; A))$, the numerator is the summation of (unnormalized) marginal distribution $P(T \subseteq S \mid S \sim P(S; A^j))$ for $j = 1, \ldots, n$. By Theorem 44 we can compute this expectation in $\mathcal{O}(nm^3)$ time. □

B.5 Greedy Derandomization

Theorem 62. *Algorithm 16 is a derandomization of dual volume sampling that selects a set S of columns satisfying*

$$\|A_S^\dagger\|_F^2 \leq \frac{m-n+1}{k-n+1}\|A^\dagger\|_F^2; \quad \|A_S^\dagger\|_2^2 \leq \frac{n(m-n+1)}{k-n+1}\|A^\dagger\|_2^2.$$

Algorithm 16 Derandomized Dual Volume Sampling for Column Subset Selection.

Input: Matrix $A \in \mathbb{R}^{n \times m}$ to sample columns from, $m \leq k \leq n$ the target size
Output: Set S such that $|S| = k$ with the guarantee

$$\|A_S^\dagger\|_F^2 \leq \frac{m-n+1}{k-n+1}\|A^\dagger\|_F^2; \quad \|A_S^\dagger\|_2^2 \leq \frac{n(m-n+1)}{k-n+1}\|A^\dagger\|_2^2$$

Initialize $\overrightarrow{S}$ as empty tuple
for $i = 1$ to k **do**
  for $j \notin \overrightarrow{S}$ **do**
    Compute conditional expectation $E_j = \mathbb{E}\left[\|A_T^\dagger\|_F^2 \mid t_1 = s_1, \ldots, t_{i-1} = s_{i-1}, t_i = j\right]$ with Corollary 46.
  end for
  Choose $j = \arg\min_{j \notin \overrightarrow{S}} E_j$
  $\overrightarrow{S} = \overrightarrow{S} \circ j$
end for
Output $\overrightarrow{S}$ as a set S

Proof. Observe that at each iteration t, we have

$$\mathbb{E}\left[\|A_T^\dagger\|_F^2 \mid t_1 = s_1, \ldots, t_{i-1} = s_{i-1}\right]$$
$$= \sum_{j \notin \overrightarrow{S}} \overrightarrow{P}(t_i = j | t_1 = s_1, \ldots, t_{i-1} = s_{i-1}) \mathbb{E}\left[\|A_T^\dagger\|_F^2 \mid t_1 = s_1, \ldots, t_{i-1} = s_{i-1}, t_i = j\right],$$

and we choose j such that $\mathbb{E}\left[\|A_T^\dagger\|_F^2 \mid t_1 = s_1, \ldots, t_{i-1} = s_{i-1}, t_i = j\right]$ is minimized. Since at the beginning we have

$$\mathbb{E}\left[\|A_T^\dagger\|_F^2\right] \leq \frac{m-n+1}{k-n+1}\|A^\dagger\|_F^2; \quad T \sim P(T; A),$$

it follows that the conditional expectation satisfies

$$\mathbb{E}\left[\|A_T^\dagger\|_F^2 \mid t_1 = s_1, \ldots, t_{i-1} = s_{i-1}, t_i = j\right] \leq \frac{m-n+1}{k-n+1}\|A^\dagger\|_F^2.$$

Hence we have

$$\|A_S^\dagger\|_F^2 = \mathbb{E}\left[\|A_T^\dagger\|_F^2 \mid t_1 = s_1, \ldots, t_{k-1} = s_{k-1}, t_k = s_k\right] \leq \frac{m-n+1}{k-n+1}\|A^\dagger\|_F^2.$$

Further, by using standard bounds relating the operator norm to the Frobenius norm, we

obtain

$$\|A_S^\dagger\|_2^2 \le \|A_S^\dagger\|_F^2 \le \frac{m-n+1}{k-n+1}\|A^\dagger\|_F^2 \le \frac{n(m-n+1)}{k-n+1}\|A^\dagger\|_2^2.$$

□

B.6 Initialization

Set $\varepsilon = \min_{|S|=k} \sigma_n^2(A_S) > 0$, whereby

$$\det(A_S A_S^\top + \varepsilon I_n) = \varepsilon^{n-k}\det(A_S^\top A_S + \varepsilon I_k) \propto VolSmpl\big(S; [A^\top \ \sqrt{\varepsilon} I_m]^\top\big).$$

The rhs is a distribution induced by volume sampling. Greedily choosing columns of A one by one gives a $k!$ approximation to the maximum volume submatrix [45]. This results in a set S such that

$$\begin{aligned}
\det(A_S A_S^\top) &\ge \frac{1}{2^n}\det(A_S A_S^\top + \varepsilon I_n) = \frac{1}{2^n \varepsilon^{k-n}}\det(A_S^\top A_S + \varepsilon I_k) \\
&\ge \max_{|S|=k} \frac{1}{2^n k! \varepsilon^{k-n}}\det(A_S^\top A_S + \varepsilon I_k) = \max_{|S|=k} \frac{1}{2^n k!}\det(A_S A_S^\top + \varepsilon I_n) \\
&\ge \frac{1}{2^n k!\binom{m}{k}} \sum_{|S|=k} \det(A_S A_S^\top + \varepsilon I_n) \ge \frac{1}{2^n k!\binom{m}{k}} \sum_{|S|=k} \det(A_S A_S^\top).
\end{aligned}$$

Thus, $\log P(S; A)^{-1} \ge \log(2^n k!\binom{m}{k}) = \mathcal{O}(k \log m)$. Note that in practice it is hard to set ε to be exactly $\min_{|S|=k} \sigma_n^2(A_S)$, but a small approximate value suffices.

B.7 Experiments

We show full results on CompAct(s), CompAct, Abalone and Bank32NH datasets in Figure B-1, B-2, B-3 and B-4 respectively. We also run DVS-*, which is $\frac{1}{*}$-generalized DVS algorithm. We observe that decreasing β sometimes helps but sometimes not. In Figure B-4 we observe that optimization- or greedy-based methods, while taking a huge amount of time to run, perform better than all sampling-based methods, thus for these selection methods,

one is not always superior than another.

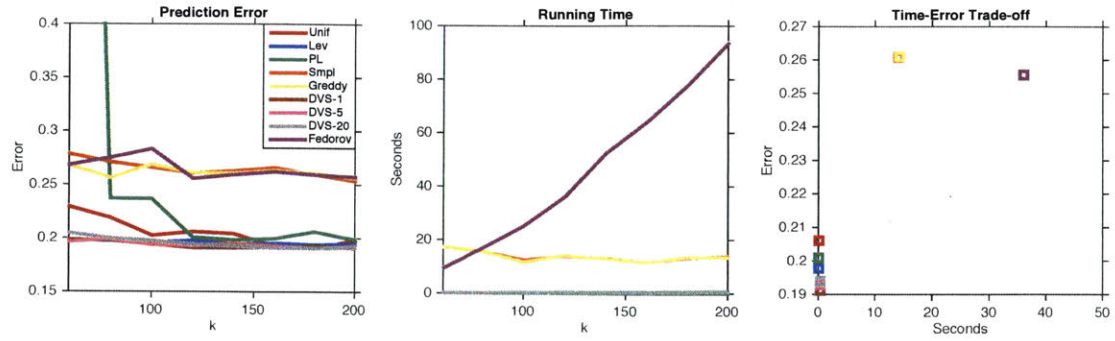


Figure B-1: Results on CompAct(s). Note that `Unif`, `Lev`, `PL` and `DVS` use less than 1 second to finish experiments.

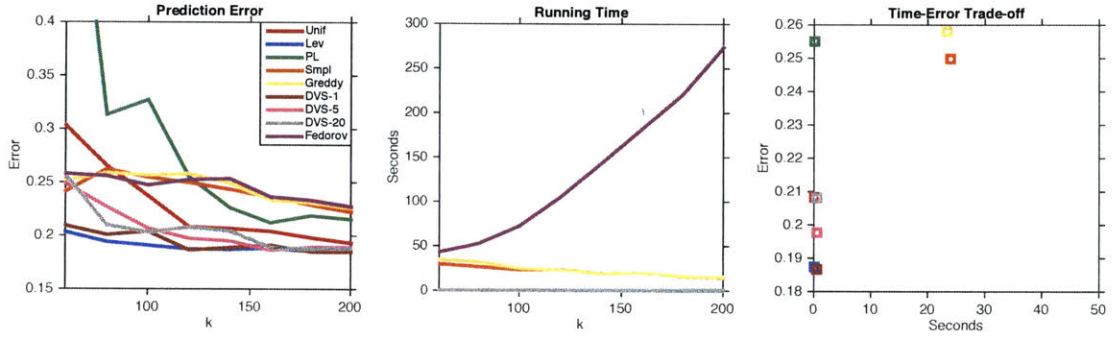


Figure B-2: Results on CompAct. Note that `Unif`, `Lev`, `PL` and `DVS` use less than 1 second to finish experiments.

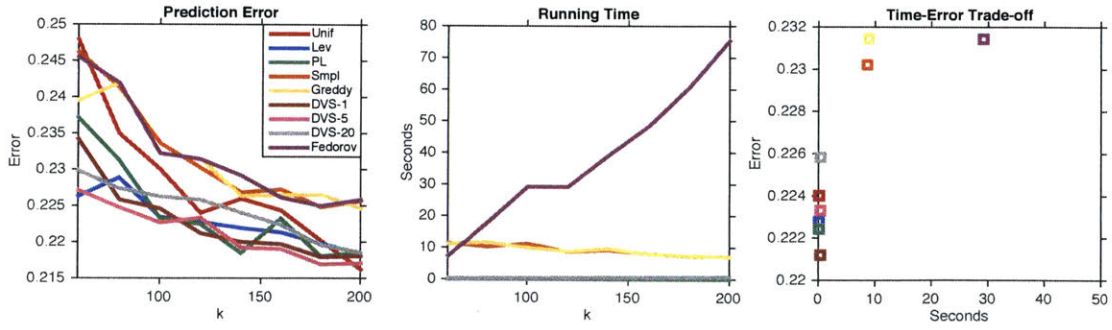


Figure B-3: Results on Abalone. Note that `Unif`, `Lev`, `PL` and `DVS` use less than 1 second to finish experiments.

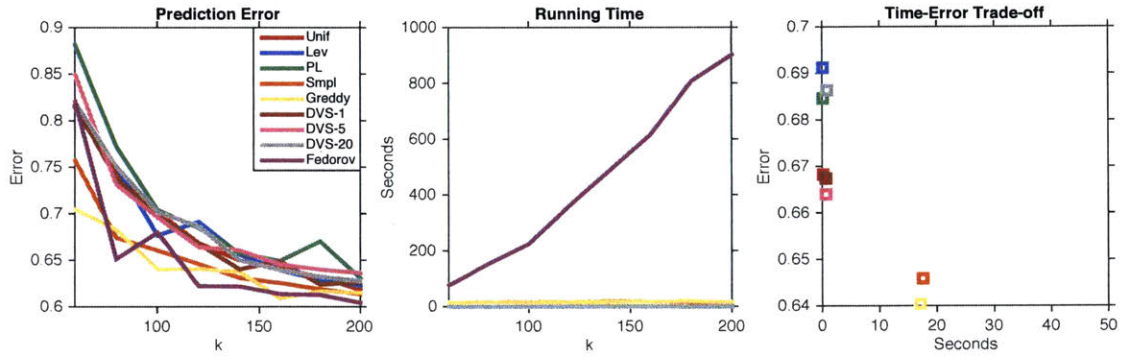


Figure B-4: Results on Bank32NH. Note that `Unif`, `Lev`, `PL` and `DVS` use less than 1 second to finish experiments.

Appendix C

Supplementary Proofs and Experiments for Chapter 5

C.1 Proof for One-sided Cardinality Constraint

Proof. Assume we have a chain (X_t) on state space V with transition matrix P, a *coupling* is a new chain (X_t, Y_t) on $V \times V$ such that both (X_t) and (Y_t), if considered marginally, are Markov chains with the same transition matrices P. The key point of coupling is to construct such a new chain to encourage X_t and Y_t to *coalesce* quickly. If, in the new chain, $\Pr(X_t \neq Y_t) \leq \varepsilon$ for some fixed t regardless of the starting state (X_0, Y_0), then $\tau(\varepsilon) \leq t$ [5]. To make the coupling construction easier, *Path coupling* [39] is then introduced so as to reduce the coupling to adjacent states in an appropriately constructed state graph. The coupling of arbitrary states follows by aggregation over a path between the two. Path coupling is formalized in the following lemma.

Lemma 63. *[39, 58] Let δ be an integer-valued metric on $V \times V$ where $\delta(\cdot, \cdot) \leq D$. Let E be a subset of $V \times V$ such that for all $(X_t, Y_t) \in V \times V$ there exists a path $X_t = Z^0, \ldots, Z^r = Y_t$ between X_t and Y_t where $(Z^i, Z^{i+1}) \in E$ for $i \in [r-1]$ and $\sum_i \delta(Z^i, Z^{i+1}) = \delta(X_t, Y_t)$. Suppose a coupling $(S, T) \to (S', T')$ of the Markov chain is defined on all pairs in E such that there exists an $\alpha < 1$ such that $\mathbb{E}[\delta(S', T')] \leq \alpha\delta(S, T)$ for all $(S, T) \in E$, then we have $\tau(\varepsilon) \leq \frac{\log(D\varepsilon^{-1})}{(1-\alpha)}$.*

We now are ready to state our proof.

We define $\delta(X,Y) = \frac{1}{2}(|X \oplus Y| + ||X| - |Y||)$. It is clear that $\delta(X,Y) \geq 1$ for $X \neq Y$. Let $E = \{(X,Y) : \delta(X,Y) = 1\}$ be the set of adjacent states (neighbors), and it follows that $\delta(\cdot,\cdot)$ is a metric satisfying conditions in Lemma 63. Also we have $\delta(X,Y) \leq k$.

We consider constructing a path coupling between any two states S and T with $\delta(S,T) = 1$, S' and T' be the two states after transition. We sample $c_S, c_T \in \{0,1\}$, if c_S is 0 then $S' = S$ and the same with c_T. $i_S, i_T \in V$ are drawn uniformly randomly. We consider two possible settings for S and T:

- If S or T is a subset of the other, we assume without of generality that $S = T \cup \{t\}$. In this setting we always let $i_S = i_T = i$. Then
  - If $i = t$, we let $c_S = 1 - c_T$;
    - If $c_S = 1$ then $\delta(S',T') = 0$ with probability $p^-(S,t)$;
    - If $c_S = 0$ then $\delta(S',T') = 0$ with probability $p^+(T,t)$;
  - If $i \in T$, we set $c_S = c_T$;
    - If $c_S = 1$ then $\delta(S',T') = 2$ with probability $(p^-(T,i) - p^-(S,i))_+$;
  - If $i \in V \backslash S$, we set $c_S = c_T$;
    - If $c_S = 1$ and $|S| < k$ then $\delta(S',T') = 2$ with probability $(p^+(S,i) - p^+(T,i))_+$.
- If S and T are of the same sizes, let $S = R \cup \{s\}$ and $T = R \cup \{t\}$. In this setting we always let $c_S = c_T = c$. We consider the case of $c = 1$:
  - If $i_S = s$, let $i_T = t$. Then $\delta(S',T') = 0$ with probability $\min\{p^-(S,s), p^-(T,t)\}$;
  - If $i_S = t$, let $i_T = s$. If $|S| < k$, Then $\delta(S',T') = 0$ with probability $\min\{p^+(S,t), p^+(T,s)\}$;
  - If $i_S \in R$, let $i_T = i_S$. Then $\delta(S',T') = 2$ with probability $|p^-(S,i_S) - p^-(T,i_T)|$;
  - If $i_S \in V \backslash (S \cup T)$, let $i_T = i_S$. If $|S| < k$, Then $\delta(S',T') = 2$ with probability $|p^+(S,i_S) - p^+(T,i_T)|$.

In all cases where we didn't specify $\delta(S', T')$, it will be $\delta(S', T') = 1$. In the first case of $S = T \cup \{t\}$ we have

$$\begin{aligned}\frac{\mathbb{E}[\delta(S',T')]}{\mathbb{E}[\delta(S,T)]} \leq & \frac{1}{2N}((1-p^-(S,t))+(1-p^+(T,t))+(2|T|+\sum_{i\in T}(p^-(T,i)-p^-(S,i))_+)+ \\ & (2(N-|S|)+[\![|S|<k]\!]\sum_{i\in[N]\setminus S}(p^+(S,i)-p^+(T,i))_+)) \\ = 1-\frac{1}{2N}(1-\sum_{i\in T}(p^-(T,i)-p^-(S,i))_+ & -[\![|S|<k]\!]\sum_{i\in[N]\setminus S}(p^+(S,i)-p^+(T,i))_+) = 1-\frac{1-\alpha_1}{2N},\end{aligned}$$

while in the second case of $|S| = R \cup \{s\}$ and $T = R \cup \{t\}$ we have

$$\begin{aligned}\frac{\mathbb{E}[\delta(S',T')]}{\mathbb{E}[\delta(S,T)]} \leq & \frac{1}{2N}((1-\min\{p^-(S,s),p^-(T,t)\})+(1-[\![|S|<k]\!]\min\{p^+(S,t),p^+(T,s)\})+ \\ & (2|R|+\sum_{i\in R}|p^-(S,i)-p^-(T,i)|)+ \\ & (2(N-|S|-1)+[\![|S|<k]\!]\sum_{i\in[N]\setminus(S\cup T)}|p^+(S,i)-p^+(T,i)|)) \\ = 1-\frac{1}{2N}(\min\{p^-(S,s),p^-(T,t)\} & -\sum_{i\in R}|p^-(S,i)-p^-(T,i)|+ \\ [\![|S|<k]\!](\min\{p^+(S,t),p^+(T,s)\} & -\sum_{i\in[N]\setminus(S\cup T)}|p^+(S,i)-p^+(T,i)|)) = 1-\frac{1-\alpha_2}{2N}.\end{aligned}$$

Let $\alpha = \max_{(S,T)\in E}\{\alpha_1, \alpha_2\}$. If $\alpha < 1$, with Lemma 63 we have

$$\tau(\varepsilon) \leq \frac{2N\log(k/\varepsilon)}{1-\alpha}. \qquad \square$$

C.2 Proof of Thm. 57

C.2.1 Proof for Uniform Matroid Base

Proof. We consider the case where $\mathcal{C}$ is uniform matroid base. For any two sets $X, Y \in \mathcal{C}$, we distribute the flow equally across all shortest paths $X \rightsquigarrow Y$ in the transition graph. Then, for arbitrary edge $e \in E$, we bound the number of paths (and flow) through e.

Consider two arbitrary sets $X, Y \in \mathcal{C}$ with symmetric difference $|X \oplus Y| = 2m \leq 2k$.

Any shortest path $X \rightsquigarrow Y$ has length m. Moreover, there are exactly $(m!)^2$ such paths, since we can exchange the elements in $X \setminus Y$ in any order with the elements in $Y \setminus X$ in any order to reach at Y. Since the total flow is $\pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)$, each path receives $\pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)/(m!)^2$ flow.

Next, let $e = (S, T)$ be any edge on some shortest path $X \rightsquigarrow Y$; so $S, T \in \mathcal{C}$ and $T = S \cup \{j\} \setminus \{i\}$ for some $i, j \in [N]$. Let $2r = |X \oplus S| < 2m$ be the length of the shortest path $X \rightsquigarrow S$, thus there are $(r!)^2$ ways to reach from X to S. Similarly, $m - r - 1$ elements are exchanged to reach from T to Y and there are in total $((m - r - 1)!)^2$ ways to do so. the total flow e receives from pair X, Y is

$$w_e(X, Y) = \frac{\pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)}{(m!)^2}(r!)^2((m - 1 - r)!)^2$$

Since in our chain,

$$Q(e) = \frac{2Z_{\mathcal{C}} \exp(\beta F(S)) \exp(\beta F(T))}{k(N - k)(\exp(\beta F(S)) + \exp(\beta F(T)))},$$

it follows that

$$\begin{aligned}\frac{w_e(X, Y)}{Q(e)} &= \frac{2(r!)^2((m - 1 - r)!)^2 k(N - k) \exp(\beta(F(X) + F(Y)))(\exp(\beta F(S)) + \exp(\beta F(T)))}{(m!)^2 Z_{\mathcal{C}} \exp(\beta(F(S) + F(T)))} \\ &\leq \frac{2(r!)^2((m - 1 - r)!)^2 k(N - k)}{(m!)^2 Z_{\mathcal{C}}} \exp(2\beta\zeta_F)(\exp(\beta F(\sigma_S(X, Y))) + \exp(\beta F(\sigma_T(X, Y)))),\end{aligned}$$

where we define $\sigma_S(X, Y) = X \oplus Y \oplus S$. The inequality draws from the fact that

$$\begin{aligned}&\frac{\exp(\beta(F(X) + F(Y) + F(S)))}{\exp(\beta(F(S) + F(T)))} = \exp(\beta(F(X) + F(Y) - F(T)) \\ &= \exp(\beta(F(X) + F(Y) - F(X \cap Y) - F(X \cup Y))) \\ &\exp(\beta(F(X \cap Y) + F(X \cup Y) - F(T) - F(\sigma_T(X, Y)))) \exp(\beta F(\sigma_T(X, Y)) \\ &\leq \exp(2\beta\zeta_F) \exp(\beta F(\sigma_T(X, Y)))\end{aligned}$$

and likewise for $\frac{\exp(\beta(F(X)+F(Y)+F(T)))}{\exp(\beta(F(S)+F(T)))}$. Similar trick has been used in [86].

Let $U_S = \sigma_S(X, Y)$ and $U_T = \sigma_T(X, Y)$, then for fixed U_S, U_T, the total flow that

passes e is

$$\begin{aligned}
&\sum_{\substack{(X,Y):\sigma_S(X,Y)=U_S,\\ \sigma_T(X,Y)=U_T}} \frac{w_e(X,Y)}{Q(e)} \\
&\leq 2\sum_{r=0}^{m-1} \binom{m-1}{r}^2 \frac{(r!)^2((m-1-r)!)^2 k(N-k)}{(m!)^2 Z} \\
&\qquad\qquad \times \exp(2\beta\zeta_F)(\exp(\beta F(U_S)) + \exp(\beta F(U_T))) \\
&= \frac{2k(N-k)}{mZ_{\mathcal{C}}} \exp(2\beta\zeta_F)(\exp(\beta F(U_S)) + \exp(\beta F(U_T))).
\end{aligned}$$

Finally, with the definition of $\overline{\rho}(f)$ we sum over all images of U_S and U_T. Recall that $Z = \sum_{U_S} \exp(\beta F(U_S))$. Since $|S \oplus X \oplus Y| = k$ we know that $U_S, U_T \in \mathcal{C}$, thus $Z \leq Z_{\mathcal{C}}$ and

$$\overline{\rho}(f) \leq 4k(N-k)\exp(2\beta\zeta_F).$$

Hence

$$\tau_{X_0}(\varepsilon) \leq 4k(N-k)\exp(2\beta\zeta_F)(\log \pi_{\mathcal{C}}(X_0)^{-1} + \log \varepsilon^{-1}).$$

C.2.2 Proof on Partition Matroid Base

Proof. Consider two arbitrary sets $X, Y \in \mathcal{C}$ with symmetric difference $|X \oplus Y| = 2m \leq 2k$, i.e., m elements need to be exchanged to reach from X to Y. However, these m steps are a valid path in the transition graph only if every set S along the way is in $\mathcal{C}$. The exchange property of matroids implies that this is indeed true, so any shortest path $X \rightsquigarrow Y$ has length m. Moreover, there are exactly $m!$ such paths, since we can exchange the elements in $X \setminus Y$ in any order to reach at Y. Note that once we choose $s \in X \setminus Y$ to swap out, there is only one choice $t \in Y \setminus X$ to swap in, where t lies in the same part as s in the partition matroid, otherwise the constraint will be violated. Since the total flow is $\pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)$, each path receives $\pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)/m!$ flow.

Next, let $e = (S,T)$ be any edge on some shortest path $X \rightsquigarrow Y$; so $S, T \in \mathcal{C}$ and $T = S \cup \{j\} \setminus \{i\}$ for some $i, j \in V$. Let $2r = |X \oplus S| < 2m$ be the length of the shortest

path $X \rightsquigarrow S$, i.e., r elements need to be exchanged to reach from X to S. Similarly, $m-r-1$ elements are exchanged to reach from T to Y. Since there is a path for every permutation of those elements, the total flow edge e receives from pair X, Y is

$$w_e(X,Y) = \frac{\pi_{\mathcal{C}}(X)\pi_{\mathcal{C}}(Y)}{m!} r!(m-1-r)!.$$

Since, in our chain, (using $L = \max_i |\mathcal{P}_i| - 1$)

$$Q(e) \geq \frac{\pi_{\mathcal{C}}(S)}{2kL} \frac{\pi_{\mathcal{C}}(T)}{\pi_{\mathcal{C}}(S) + \pi_{\mathcal{C}}(T)} = \frac{\exp(\beta F(S)) \exp(\beta F(T))}{2kLZ_{\mathcal{C}}(\exp(\beta F(S)) + \exp(\beta F(T)))},$$

it follows that

$$\begin{aligned} \frac{w_e(X,Y)}{Q(e)} &\leq \frac{2r!(m-1-r)!kL \exp(\beta(F(X)+F(Y)))(\exp(\beta F(S)) + \exp(\beta F(T)))}{m! Z_{\mathcal{C}} \exp(\beta(F(S)+F(T)))} \\ &\leq \frac{2r!(m-1-r)!kL}{m!Z_{\mathcal{C}}} \exp(2\beta\zeta_F)(\exp(\beta F(\sigma_S(X,Y))) + \exp(\beta F(\sigma_T(X,Y)))), \end{aligned} \tag{C.2.1}$$

where we define $\sigma_S(X,Y) = X \oplus Y \oplus S = (X \cap Y \cap S) \cup (X \setminus (Y \cup S)) \cup (Y \setminus (X \cup S))$. To bound the total flow, we must count the pairs X, Y such that e is on their shortest path(s), and bound the flow they send. We do this in two steps, first summing over all X, Y that share the upper bound (C.2.1) since they have the same difference sets $U_S = \sigma_S(X,Y)$ and $U_T = \sigma_T(X,Y)$, and then we sum over all possible U_S and U_T. For fixed U_S, U_T, there are $\binom{m-1}{r}$ pairs that share those difference sets, since the only freedom we have is to assign r of the $m-1$ elements in $S \setminus (X \cap Y \cap S)$ to Y, and the rest to X. Hence, for fixed U_S, U_T:

$$\begin{aligned} \sum_{\substack{(X,Y):\, \sigma_S(X,Y)=U_S, \\ \sigma_T(X,Y)=U_T}} \frac{w_e(X,Y)}{Q(e)} &\leq 2 \sum_{r=0}^{m-1} \binom{m-1}{r} \frac{r!(m-1-r)!kL}{m!Z_{\mathcal{C}}} \\ &\quad \times \exp(2\beta\zeta_F)(\exp(\beta F(U_S)) + \exp(\beta F(U_T))) \\ &= \frac{2kL}{Z_{\mathcal{C}}} \exp(2\beta\zeta_F)(\exp(\beta F(U_S)) + \exp(\beta F(U_T))). \end{aligned} \tag{C.2.2}$$

Finally, we sum over all valid U_S (U_T is determined by U_S), where by "valid" we mean there exists $X, Y \in \mathcal{C}$ and $S \in \mathcal{C}$ on one path from X to Y such that, $U_S = \sigma_S(X, Y)$. Any such U_S can be constructed by picking $k - m$ elements from S (including i), and by replacing the remaining elements $u \in S$ by another member of their partition: i.e., if $u \in \mathcal{P}_\ell$, then it is replaced by some other $v \in \mathcal{P}_\ell$, since both X and Y must be in $\mathcal{C}$. Hence, any U_S satisfies the partition constraint, i.e., $U_S \in \mathcal{C}$ and therefore $\sum_{U_S} \exp(\beta F(U_S)) \leq Z_\mathcal{C}$, and likewise for U_T. Hence, summing the bound (C.2.2) over all possible U_S yields

$$\overline{\rho}(f) \leq 4kL \exp(2\beta\zeta_F) \max_p \text{len}(p) \leq 4k^2 L \exp(2\beta\zeta_F),$$

where we upper bound the length of any shortest path by k, since $m \leq k$. Hence

$$\tau_{X_0}(\varepsilon) \leq 4k^2 L \exp(2\beta\zeta_F)(\log \pi_\mathcal{C}(X_0)^{-1} + \log \varepsilon^{-1}). \qquad \square$$

C.2.3 Proof for General Matroid Base

In the case where no structural assumption is made on $\mathcal{C}$, the proof needs to be more carefully handled. Because in this case, we know neither the number of legal paths between any two states, nor the number of $\sigma_S(X, Y)$ falls out of $\mathcal{C}$.

We again consider arbitrary sets $X, Y \in \mathcal{C}$ where $|X \oplus Y| = 2m \leq 2k$. The total number of shortest paths is *at least* $(m!)$ due to exchange property of matroids. Since the amount of flow from X to Y is $\pi_\mathcal{C}(X)\pi_\mathcal{C}(Y)$, each path receives *at most* $\pi_\mathcal{C}(x)\pi_\mathcal{C}(y)/m!$.

Next, let $e = (S, T)$ be any edge on some shortest path $X \rightsquigarrow Y$; so $S, T \in \mathcal{C}$ and $T = S \cup \{j\} \backslash \{i\}$ for some $i, j \in V$. Let $2r = |X \oplus S| < 2m$ be the length of the shortest path $X \rightsquigarrow S$, thus there are at most $(r!)^2$ ways to reach from X to S. Likewise there are at most $((m - r - 1)!)^2$ paths to reach from T to Y. The total flow edge e receives from pair X, Y is then upper-bounded as

$$w_e(X, Y) \leq \frac{\pi_\mathcal{C}(X)\pi_\mathcal{C}(Y)}{m!}(r!)^2((m - 1 - r)!)^2.$$

It follows that

$$\frac{w_e(X,Y)}{Q(e)} \leq \frac{2(r!)^2((m-1-r)!)^2k(N-k)}{m!Z_C}\exp(2\beta\zeta_F)(\exp(\beta F(U_S))+\exp(\beta F(U_T))).$$

The total pairs of (X,Y) that passes e with the same set of images is upper-bounded by $\binom{m-1}{r}^2$, thus the flow that passes e with the same set of images is bounded as

$$\begin{aligned}
&\sum_{\substack{(X,Y):\sigma_S(X,Y)=U_S,\\ \sigma_T(X,Y)=U_T}} \frac{w_e(X,Y)}{Q(e)} \\
&\leq 2\sum_{r=0}^{m-1}\binom{m-1}{r}^2\frac{(r!)^2((m-1-r)!)^2k(N-k)}{m!Z} \\
&\qquad\qquad \times \exp(2\beta\zeta_F)(\exp(\beta F(U_S))+\exp(\beta F(U_T))) \\
&= \frac{2(m-1)!k(N-k)}{Z_C}\exp(2\beta\zeta_F)(\exp(\beta F(U_S))+\exp(\beta F(U_T))).
\end{aligned}$$

Thus if we sum over all U_S, U_T, the result is upper-bounded as

$$\overline{\rho}(f) \leq \frac{4k!Z}{Z_C}k(N-k)\exp(2\beta\zeta_F).$$

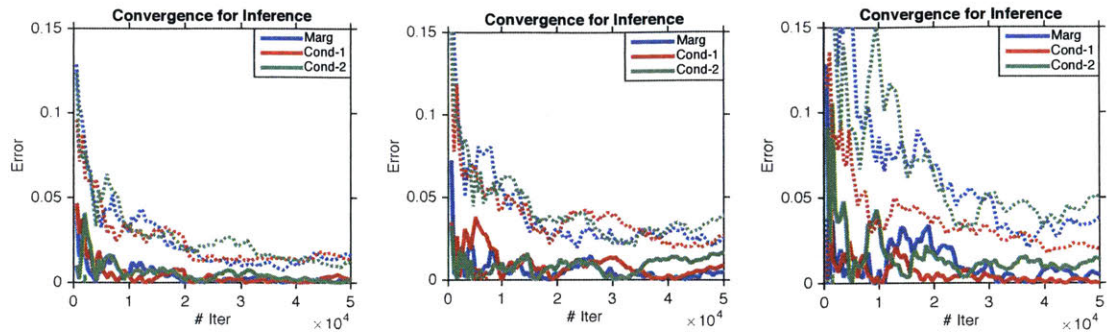
Note that here we upper-bounded m with k and Z could be larger than Z_C because it may happen that $U_S \notin \mathcal{C}$. It follows that

$$\tau_{X_0}(\varepsilon) \leq \frac{4k!Z}{Z_C}k(N-k)\exp(2\beta\zeta_F)(\log\pi_C(X_0)^{-1}+\log\varepsilon^{-1}). \qquad \square$$

C.3 Supplementary Experiments

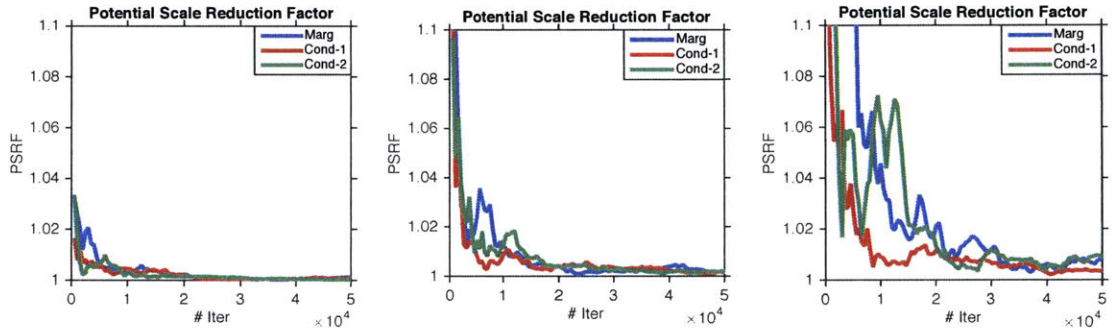
C.3.1 Varying δ

We run 20-variable chain-structured Ising model on partition matroid base of rank 5 with varying δ's. The results are shown in Fig. C-1 and Fig. C-2. We observe that the approximate mixing time grows with δ.



(a) (b) (c)

Figure C-1: Convergence of marginal (`Marg`) and conditional (`Cond-1` and `Cond-2`, conditioned on 1 and 2 other variables) probabilities of a single variable in a 20-variable Ising model. We fix $\beta = 3$ and vary δ as (a) $\delta = 0.2$, (b) $\delta = 0.5$ and (c) $\delta = 0.8$. Full lines show the means and dotted lines the standard deviations of estimations.



(a) (b) (c)

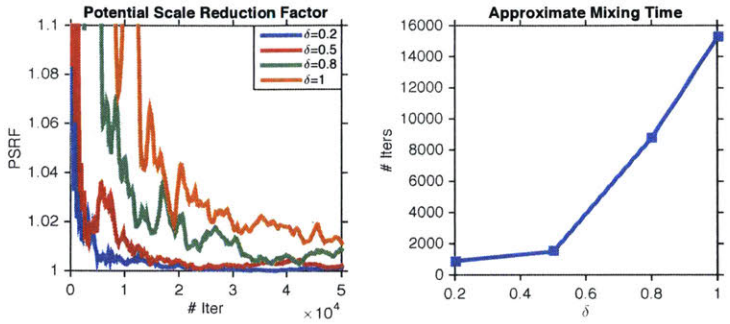
Figure C-2: PSRF of each set of chains in Fig. C-1 with $\beta = 3$ and (a) $\delta = 0.2$; (b) $\delta = 0.5$ and (c) $\delta = 0.8$.

C.3.2 Varying β

We run 20-variable chain-structured Ising model on partition matroid base of rank 5 with varying β's. The results are shown in Fig. C-4 and Fig. C-5. We observe that the approximate mixing time grows with β.

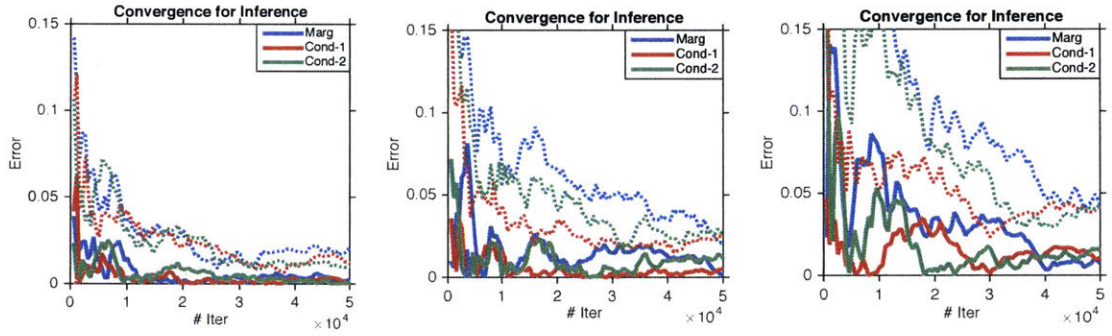
C.3.3 Varying Data Sizes

We run (k-)DPP that is constrained to sample subsets from 1) partition matroid base and 2) uniform matroid with different data sizes N.



(a) (b)

Figure C-3: Comparisons of PSRF's for marginal estimations with different δ's. (a) PSRF's with different δ's and (b) the approximate mixing time estimated by thresholding PSRF at 1.05.



(a) (b) (c)

Figure C-4: Convergence of marginal (`Marg`) and conditional (`Cond-1` and `Cond-2`, conditioned on 1 and 2 other variables) probabilities of a single variable in a 20-variable Ising model. We fix $\delta = 1$ and vary β as (a) $\beta = 0.5$; (b) $\beta = 2$ and (c) $\beta = 3$. Full lines show the means and dotted lines the standard deviations of estimations.

Partition Matroid Constraint

The estimations for marginal and conditional distributions are shown in Fig. C-7 and corresponding PSRF's are shown in Fig. C-8. We observe that the estimation becomes stable faster when N is small.

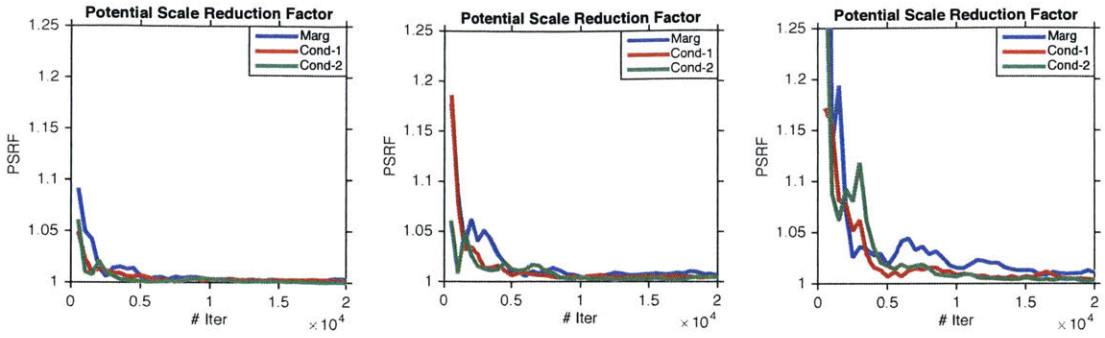


Figure C-5: PSRF of each set of chains in Fig. C-4 with $\delta = 1$ and (a) $\beta = 0.5$; (b) $\beta = 2$ and (c) $\beta = 3$.

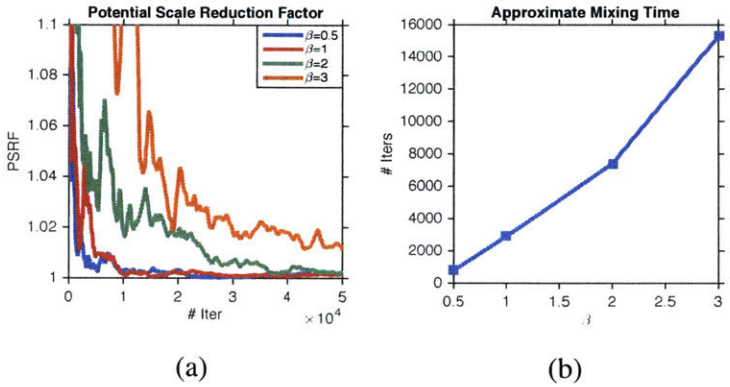


Figure C-6: Comparisons of PSRF's for marginal estimations with different β's. (a) PSRF's with different β's and (b) the approximate mixing time estimated by thresholding of 1.05 on PSRF's.

Uniform Matroid Constraint

The estimations for marginal and conditional distributions are shown in Fig. C-9 and corresponding PSRF's are shown in Fig. C-10. We observe the same thing as mentioned before, that the estimation becomes stable faster when N is small.

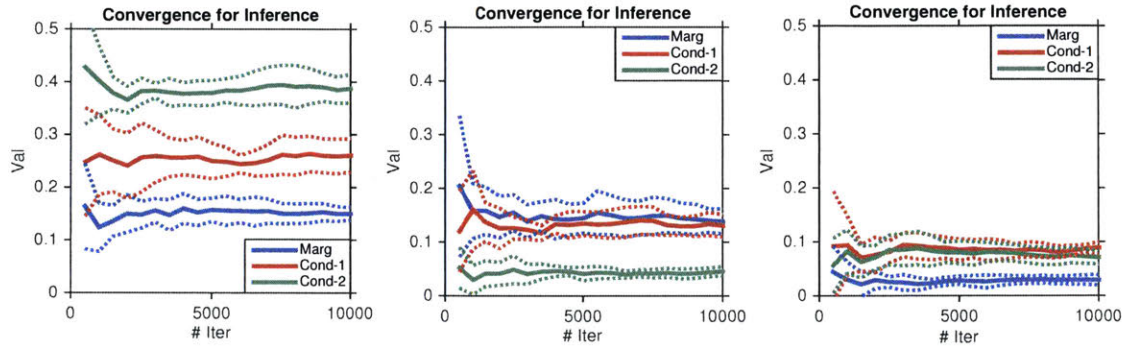


Figure C-7: Convergence of marginal (`Marg`) and conditional (`Cond-1` and `Cond-2`, conditioned on 1 and 2 other variables) probabilities of a single variable in a k-DPP on partition matroid base of rank 5, with (a) $N = 20$; (b) $N = 50$ and (c) $N = 100$. Full lines show the means and dotted lines the standard deviations of estimations.

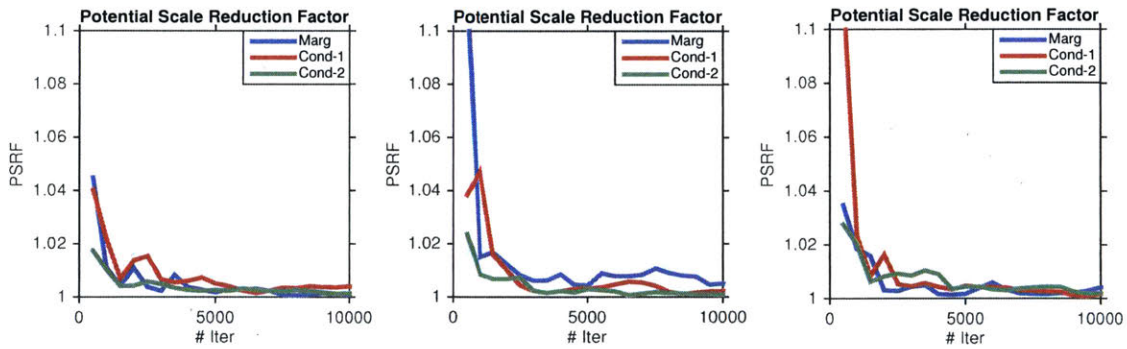


Figure C-8: PSRF of marginal (`Marg`) and conditional (`Cond-1` and `Cond-2`, conditioned on 5 and 10 other variables) probabilities of a single variable in a k-DPP on partition matroid base of rank 5, with (a) $N = 20$; (b) $N = 50$ and (c) $N = 100$.

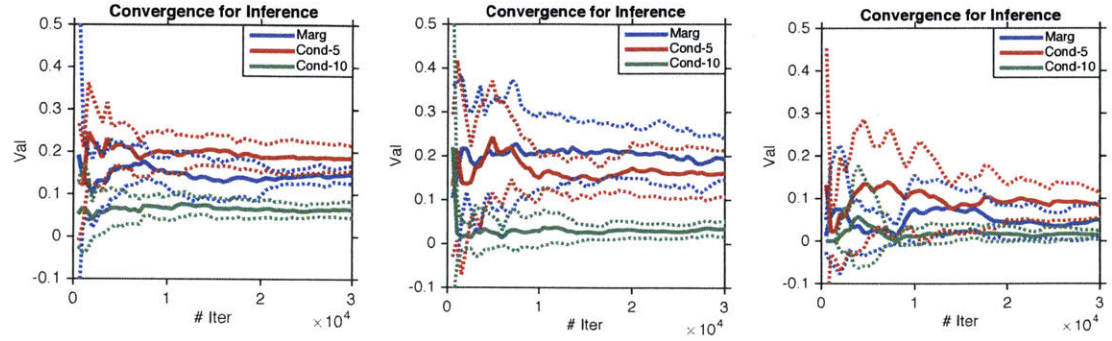


Figure C-9: Convergence of marginal (`Marg`) and conditional (`Cond-5` and `Cond-10`, conditioned on 5 and 10 other variables) probabilities of a single variable in a DPP on uniform matroid of rank 30, with (a) $N = 50$; (b) $N = 100$ and (c) $N = 200$. Full lines show the means and dotted lines the standard deviations of estimations.

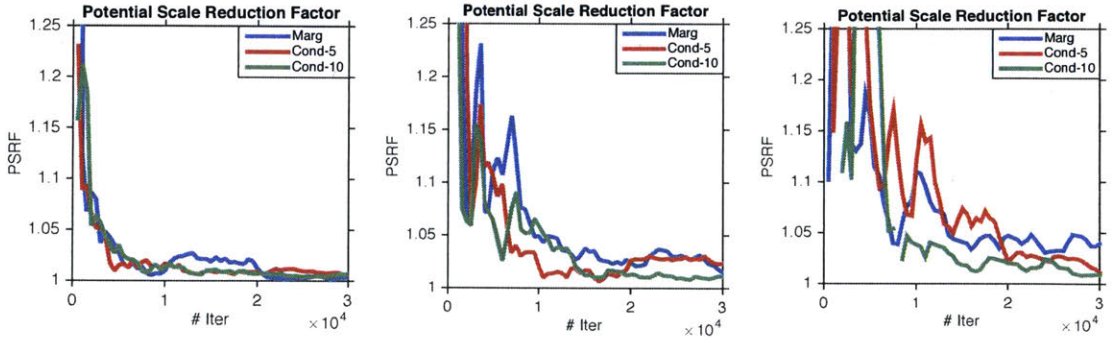


Figure C-10: PSRF of marginal (`Marg`) and conditional (`Cond-5` and `Cond-10`, conditioned on 5 and 10 other variables) probabilities of a single variable in a DPP on uniform matroid of rank 30, with (a) $N = 50$; (b) $N = 100$ and (c) $N = 200$.

Bibliography

[1] Raja Hafiz Affandi, Emily Fox, and Ben Taskar. Approximate inference in continuous determinantal processes. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1430–1438, 2013.

[2] Raja Hafiz Affandi, Alex Kulesza, and Emily Fox. Markov determinantal point processes. *Uncertainty in Artificial Intelligence (UAI)*, 2012.

[3] Raja Hafiz Affandi, Alex Kulesza, Emily Fox, and Ben Taskar. Nyström approximation for large-scale determinantal processes. In *Artificial Intelligence and Statistics*, pages 85–98, 2013.

[4] Ahmed Alaoui and Michael Mahoney. Fast randomized kernel methods with statistical guarantees. *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[5] David Aldous. Some inequalities for reversible Markov chains. *Journal of the London Mathematical Society*, pages 564–576, 1982.

[6] Nima Anari and Shayan Gharan. The Kadison-Singer problem for strongly Rayleigh measures and applications to asymmetric TSP. *arXiv:1412.1143*, 2014.

[7] Nima Anari and Shayan Gharan. Effective-resistance-reducing flows and asymmetric TSP. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2015.

[8] Nima Anari, Shayan Gharan, and Alireza Rezaei. Monte Carlo Markov chain algorithms for sampling strongly Rayleigh distributions and determinantal point processes. *Conference on Learning Theory (COLT)*, 2016.

[9] Nima Anari, Kuikui Liu, Shayan Gharan, and Cynthia Vinzant. Log-concave polynomials II: High-dimensional walks and an FPRAS for counting bases of a matroid. *arXiv preprint arXiv:1811.01816*, 2018.

[10] Mario Arioli and Iain Duff. Preconditioning of linear least-squares problems by identifying basic variables. *SIAM J. Sci. Comput.*, 2015.

[11] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *SIAM-ACM Symposium on Discrete Algorithms (SODA)*, pages 1027–1035, 2007.

[12] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer networks*, 54(15):2787–2805, 2010.

[13] Haim Avron and Christos Boutsidis. Faster subset selection for matrices and applications. *SIAM Journal on Matrix Analysis and Applications*, pages 1464–1499, 2013.

[14] Francis Bach. Sharp analysis of low-rank kernel matrix approximations. *Conference on Learning Theory (COLT)*, 2013.

[15] Francis Bach and Michael Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, pages 1–48, 2003.

[16] Francis Bach and Michael Jordan. Predictive low-rank decomposition for kernel methods. In *Int. Conference on Machine Learning (ICML)*, pages 33–40, 2005.

[17] Zhaojun Bai, Gark Fahey, and Gene Golub. Some large-scale matrix computation problems. *J. Comp. and Applied Math.*, pages 71–89, 1996.

[18] Christopher Baker. *The numerical treatment of integral equations*. Clarendon press Oxford, 1977.

[19] Nematollah Kayhan Batmanghelich, Gerald Quon, Alex Kulesza, Manolis Kellis, Polina Golland, and Luke Bornn. Diversifying sparsity using variational determinantal point processes. *arXiv preprint arXiv:1411.6307*, 2014.

[20] Constantine Bekas, Alessandro Curioni, and Irina Fedulova. Low cost high performance uncertainty quantification. In *Proc. the 2nd Workshop on High Performance Computational Finance*, 2009.

[21] Mohamed-Ali Belabbas and Patrick Wolfe. On landmark selection and sampling in high-dimensional data analysis. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, pages 4295–4312, 2009.

[22] Mohamed-Ali Belabbas and Patrick Wolfe. Spectral methods in machine learning and new strategies for very large datasets. *Proc. National Academy of Sciences of the United States of America (PNAS)*, 2009.

[23] Michele Benzi and Gene Golub. Bounds for the entries of matrix functions with applications to preconditioning. *BIT Numerical Mathematics*, pages 417–438, 1999.

[24] Michele Benzi and Christine Klymko. Total communicability as a centrality measure. *J. Complex Networks*, pages 124–149, 2013.

[25] Michele Benzi, Carl Meyer, and Miroslav Tuma. A sparse approximate inverse preconditioner for the conjugate gradient method. *SIAM J. Scientific Computing*, pages 1135–1149, 1996.

[26] Phillip Bonacich. Power and centrality: A family of measures. *American Journal of Sociology*, pages 1170–1182, 1987.

[27] Julius Borcea and Petter Brändén. Applications of stable polynomials to mixed determinants: Johnson's conjectures, unimodality, and symmetrized Fischer products. *Duke Mathematical Journal*, pages 205–223, 2008.

[28] Julius Borcea, Petter Brändén, and Thomas Liggett. Negative dependence and the geometry of polynomials. *Journal of the American Mathematical Society*, pages 521–567, 2009.

[29] Alexei Borodin. Determinantal point processes. *arXiv preprint arXiv:0911.1153*, 2009.

[30] Alexei Borodin and Vadim Gorin. Lectures on integrable probability, 2012.

[31] Alexei Borodin and Grigori Olshanski. Distributions on partitions, point processes, and the hypergeometric kernel. *Communications in Mathematical Physics*, 211(2):335–358, 2000.

[32] Alexandre Bouchard-Côté and Michael Jordan. Variational inference over combinatorial spaces. In *Advances in Neural Information Processing Systems (NIPS)*, 2010.

[33] Christos Boutsidis, Petros Drineas, and Malik Magdon-Ismail. Near-optimal column-based matrix reconstruction. *SIAM Journal on Computing*, pages 687–717, 2014.

[34] Christos Boutsidis, Petros Drineas, and Michael Mahoney. An improved approximation algorithm for the column subset selection problem. In *SIAM-ACM Symposium on Discrete Algorithms (SODA)*, 2009.

[35] Christos Boutsidis and Malik Magdon-Ismail. Deterministic feature selection for k-means clustering. *IEEE Transactions on Information Theory*, pages 6099–6110, 2013.

[36] Christos Boutsidis, Anastasios Zouzias, Michael Mahoney, and Petros Drineas. Stochastic dimensionality reduction for k-means clustering. *arXiv preprint arXiv:1110.2897*, 2011.

[37] Andrei Broder. Generating random spanning trees. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 442–447, 1989.

[38] Stephen Brooks and Andrew Gelman. General methods for monitoring convergence of iterative simulations. *Journal of computational and graphical statistics*, pages 434–455, 1998.

[39] Russ Bubley and Martin Dyer. Path coupling: A technique for proving rapid mixing in Markov chains. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 223–231, 1997.

[40] Niv Buchbinder, Moran Feldman, Joseph Seffi, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM Journal on Computing*, pages 1384–1402, 2015.

[41] Alexander Bufetov. Infinite determinantal measures. *arXiv:1207.6793*, 2012.

[42] Nicolo Cesa-Bianchi and Gabor Lugosi. Combinatorial bandits. In *Conference on Learning Theory (COLT)*, 2009.

[43] Siheng Chen, Rohan Varma, Aliaksei Sandryhaila, and Jelena Kovačević. Discrete signal processing on graphs: Sampling theory. *IEEE Transactions on Signal Processing*, 63(24):6510–6523, 2015.

[44] Dehua Cheng, Yu Cheng, Yan Liu, Richard Peng, and Shang-Hua Teng. Scalable parallel factorizations of SDD matrices and efficient sampling for Gaussian graphical models. *arXiv:1410.5392*, 2014.

[45] Ali Çivril and Malik Magdon-Ismail. On selecting a maximum volume sub-matrix of a matrix and related problems. *Theoretical Computer Science*, pages 4801–4811, 2009.

[46] Corinna Cortes, Mehryar Mohri, and Ameet Talwalkar. On the impact of kernel approximation on learning accuracy. In *Proc. Int. Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 113–120, 2010.

[47] Mary Cowles and Bradley Carlin. Markov chain Monte Carlo convergence diagnostics: a comparative review. *Journal of the American Statistical Association*, pages 883–904, 1996.

[48] Michal Derezinski and Manfred Warmuth. Unbiased estimates for linear regression via volume sampling. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3084–3093, 2017.

[49] Amit Deshpande and Luis Rademacher. Efficient volume sampling for row/column subset selection. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 329–338, 2010.

[50] Amit Deshpande, Luis Rademacher, Santosh Vempala, and Grant Wang. Matrix approximation and projective clustering via volume sampling. In *SIAM-ACM Symposium on Discrete Algorithms (SODA)*, pages 1117–1126, 2006.

[51] Persi Diaconis and Laurent Saloff-Coste. Comparison theorems for reversible Markov chains. *The Annals of Applied Probability*, 1993.

[52] Persi Diaconis and Daniel Stroock. Geometric bounds for eigenvalues of Markov chains. *The Annals of Applied Probability*, pages 36–61, 1991.

[53] Josip Djolonga and Andreas Krause. From MAP to marginals: Variational inference in Bayesian submodular models. In *Advances in Neural Information Processing Systems (NIPS)*, pages 244–252, 2014.

[54] Shao-Jing Dong and Keh-Fei Liu. Stochastic estimation with z_2 noise. *Physics Letters B*, pages 130–136, 1994.

[55] Petros Drineas, Ravi Kannan, and Michael Mahoney. Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing*, pages 158–183, 2006.

[56] Petros Drineas and Michael Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. *Journal of Machine Learning Research*, pages 2153–2175, 2005.

[57] Martin Dyer, Alan Frieze, and Mark Jerrum. On counting independent sets in sparse graphs. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 1999.

[58] Martin Dyer and Catherine Greenhill. A more rapidly mixing Markov chain for graph colorings. *Random Structures and Algorithms*, pages 285–317, 1998.

[59] Michael Elkin, Yuval Emek, Daniel Spielman, and Shang-Hua Teng. Lower-stretch spanning trees. *SIAM Journal on Computing*, 2008.

[60] Stefano Ermon, Carla Gomes, Ashish Sabharwal, and Bart Selman. Embed and project: Discrete sampling with universal hashing. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2085–2093, 2013.

[61] Ernesto Estrada and Desmond Higham. Network properties revealed through matrix functions. *SIAM Review*, pages 696–714, 2010.

[62] Tomás Feder and Milena Mihail. Balanced matroids. In *Symposium on Theory of Computing (STOC)*, pages 26–38, 1992.

[63] Valerii Vadimovich Fedorov. *Theory of optimal experiments*. Elsevier, 1972.

[64] Caterina Fenu, David Martin, Lothar Reichel, and Giuseppe Rodriguez. Network analysis via partial spectral factorization and Gauss quadrature. *SIAM Journal on Scientific Computing*, pages A2046–A2068, 2013.

[65] Shai Fine and Katya Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, pages 243–264, 2002.

[66] Charless Fowlkes, Serge Belongie, Fan Chung, and Jitendra Malik. Spectral grouping using the Nyström method. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, pages 214–225, 2004.

[67] James Freericks. Transport in multilayered nanostructures. *The Dynamical Mean-Field Theory Approach, Imperial College, London*, 2006.

[68] Alan Frieze, Navin Goyal, Luis Rademacher, and Santosh Vempala. Expanders via random spanning trees. *SIAM Journal on Computing*, 43(2):497–513, 2014.

[69] Andreas Frommer, Thomas Lippert, Björn Medeke, and Klaus Schilling. *Numerical Challenges in Lattice Quantum Chromodynamics: Joint Interdisciplinary Workshop of John Von Neumann Institute for Computing, Jülich, and Institute of Applied Computer*

Science, Wuppertal University, August 1999, volume 15. Springer Science & Business Media, 2012.

[70] Mike Gartrell, Ulrich Paquet, and Noam Koenigstein. Low-rank factorization of determinantal point processes for recommendation. *arXiv:1602.05436*, 2016.

[71] Carl Friedrich Gauss. *Methodus nova integralium valores per approximationem inveniendi.* apvd Henricvm Dieterich, 1815.

[72] Walter Gautschi. A survey of Gauss-Christoffel quadrature formulae. In *E.B. Christoffel*, pages 72–147. Springer, 1981.

[73] Andrew Gelman and Donald B Rubin. Inference from iterative simulation using multiple sequences. *Statistical science*, pages 457–472, 1992.

[74] Shayan Gharan and Alireza Rezaei. A polynomial time MCMC method for sampling from continuous DPPs. *arXiv preprint arXiv:1810.08867*, 2018.

[75] Shayan Gharan, Amin Saberi, and Mohit Singh. A randomized rounding approach to the Traveling Salesman Problem. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, 2011.

[76] Walter Gilks, Sylvia Richardson, and David Spiegelhalter. *Markov chain Monte Carlo in practice*. 1995.

[77] Jennifer Gillenwater, Alex Kulesza, and Ben Taskar. Discovering diverse and salient threads in document collections. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 710–720, 2012.

[78] Jennifer Gillenwater, Alex Kulesza, and Ben Taskar. Near-optimal MAP inference for determinantal point processes. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[79] Alex Gittens and Michael Mahoney. Revisiting the Nyström method for improved large-scale machine learning. *Int. Conference on Machine Learning (ICML)*, 2013.

[80] Gene Golub. Some modified matrix eigenvalue problems. *SIAM Review*, pages 318–334, 1973.

[81] Gene Golub and Gérard Meurant. Matrices, moments and quadrature II; how to compute the norm of the error in iterative methods. *BIT Numerical Mathematics*, pages 687–705, 1997.

[82] Gene Golub and Gérard Meurant. *Matrices, moments and quadrature with applications*. Princeton University Press, 2009.

[83] Gene Golub, Martin Stoll, and Andy Wathen. Approximation of the scattering amplitude and linear systems. *Elec. Tran. on Numerical Analysis*, pages 178–203, 2008.

[84] Gene Golub and John Welsch. Calculation of Gauss quadrature rules. *Mathematics of computation*, 23(106):221–230, 1969.

[85] Boqing Gong, Wei-Lun Chao, Kristen Grauman, and Fei Sha. Diverse sequential subset selection for supervised video summarization. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2069–2077, 2014.

[86] Alkis Gotovos, Hamed Hassani, and Andreas Krause. Sampling from probabilistic submodular models. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1936–1944, 2015.

[87] Dorothy Greig, Bruce Porteous, and Allan Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society*, 51(2), 1989.

[88] Carlos Guestrin, Andreas Krause, and Ajit Singh. Near-optimal sensor placements in Gaussian processes. In *Int. Conference on Machine Learning (ICML)*, pages 265–272, 2005.

[89] Venkatesan Guruswami and Ali Sinop. Optimal column-based low-rank matrix reconstruction. In *SIAM-ACM Symposium on Discrete Algorithms (SODA)*, 2012.

[90] Raja Hafiz Affandi, Emily Fox, and Ben Taskar. Approximate inference in continuous determinantal point processes. *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[91] Nathan Halko, Per-Gunnar Martinsson, and Joel Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.

[92] Willeen Hastings. Monte carlo sampling methods using Markov chains and their applications. 1970.

[93] Magnus Hestenes and Eduard Stiefel. Methods of conjugate gradients for solving linear systems. *J. Research of the National Bureau of Standards*, pages 409–436, 1952.

[94] John Hough, Manjunath Krishnapur, Yuval Peres, and Bálint Virág. Determinantal processes and independence. *Probability surveys*, pages 206–229, 2006.

[95] Rishabh Iyer and Jeff Bilmes. Submodular point processes. In *Proc. Int. Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

[96] Mark Jerrum and Alistair Sinclair. Polynomial-time approximation algorithms for the ising model. *SIAM J. Computing*, 22(5), 1993.

[97] Mark Jerrum, Alistair Sinclair, and Eric Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM*, 51(4), 2004.

[98] Mark Jerrum, Jung-Bae Son, Prasad Tetali, and Eric Vigoda. Elementary bounds on Poincaré and log-Sobolev constants for decomposable Markov chains. *Annals of Applied Probability*, pages 1741–1765, 2004.

[99] Siddharth Joshi and Stephen Boyd. Sensor selection via convex optimization. *IEEE Transactions on Signal Processing*, pages 451–462, 2009.

[100] Byungkon Kang. Fast determinantal point process sampling with application to clustering. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2319–2327, 2013.

[101] Tarun Kathuria and Amit Deshpande. On sampling and greedy map inference of constrained determinantal point processes. *arXiv preprint arXiv:1607.01551*, 2016.

[102] Mutsuki Kojima and Fumiyasu Komaki. Determinantal point process priors for Bayesian variable selection in linear regression. *Statistica Sinica*, pages 97–117, 2016.

[103] Andreas Krause, Ajit Singh, and Carlos Guestrin. Near-optimal sensor placements in Gaussian processes: Theory, efficient algorithms and empirical studies. *Journal of Machine Learning Research*, pages 235–284, 2008.

[104] Alex Kulesza and Ben Taskar. Structured determinantal point processes. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1171–1179, 2010.

[105] Alex Kulesza and Ben Taskar. k-DPPs: Fixed-size determinantal point processes. In *Int. Conference on Machine Learning (ICML)*, pages 1193–1200, 2011.

[106] Alex Kulesza and Ben Taskar. Learning determinantal point processes. *Uncertainty in Artificial Intelligence (UAI)*, 2011.

[107] Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends® in Machine Learning*, 5(2–3):123–286, 2012.

[108] Sanjiv Kumar, Mehryar Mohri, and Ameet Talwalkar. Ensemble Nyström method. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1060–1068, 2009.

[109] Cornelius Lanczos. *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*. United States Governm. Press Office Los Angeles, CA, 1950.

[110] Christina Lee, Asuman Ozdaglar, and Devavrat Shah. Solving systems of linear equations: Locally and asynchronously. *Computing Research Repository*, 2014.

[111] Jure Leskovec, Kevin Lang, Anirban Dasgupta, and Michael Mahoney. Statistical properties of community structure in large social and information networks. In *Proc. Int. World Wide Web Conference (WWW)*, pages 695–704, 2008.

[112] Chengtao Li, Stefanie Jegelka, and Suvrit Sra. Efficient sampling for k-determinantal point processes. *Proc. Int. Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.

[113] Chengtao Li, Stefanie Jegelka, and Suvrit Sra. Fast DPP sampling for Nyström with application to kernel methods. *Int. Conference on Machine Learning (ICML)*, 2016.

[114] Chengtao Li, Stefanie Jegelka, and Suvrit Sra. Fast sampling for strongly Rayleigh measures with application to determinantal point processes. *arXiv preprint arXiv:1607.03559*, 2016.

[115] Chengtao Li, Stefanie Jegelka, and Suvrit Sra. Polynomial time algorithms for dual volume sampling. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5038–5047, 2017.

[116] Chengtao Li, Stefanie Jegelka, and Suvrit Sra. Fast mixing Markov chains for strongly Rayleigh measures and variants. *In Preparation*, 2018.

[117] Chengtao Li, Suvrit Sra, and Stefanie Jegelka. Fast mixing Markov chains for strongly Rayleigh measures, DPPs, and constrained sampling. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[118] Chengtao Li, Suvrit Sra, and Stefanie Jegelka. Gaussian quadrature for matrix inverse forms with applications. In *Int. Conference on Machine Learning (ICML)*, pages 1766–1775, 2016.

[119] Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 510–520, 2011.

[120] Lin Lin, Chao Yang, Jianfeng Lu, and Lexing Ying. A fast parallel algorithm for selected inversion of structured sparse matrices with application to 2D electronic structure calculations. *SIAM Journal on Scientific Computing*, pages 1329–1351, 2011.

[121] Lin Lin, Chao Yang, Juan Meza, Jianfeng Lu, Lexing Ying, and Weinan E. Selinv–an algorithm for selected inversion of a sparse symmetric matrix. *ACM Transactions on Mathematical Software*, 2011.

[122] Rehuel Lobatto. *Lessen over de differentiaal-en integraal-rekening: Dl. 2 Integraal-rekening*, volume 1. Van Cleef, 1852.

[123] Russell Lyons. Determinantal probability measures. *Publications Mathématiques de l'Institut des Hautes Études Scientifiques*, 98:167–212, 2003.

[124] Russell Lyons. Determinantal probability: basic properties and conjectures. *arXiv:1406.2707*, 2014.

[125] Ping Ma, Michael Mahoney, and Bin Yu. A statistical perspective on algorithmic leveraging. In *Journal of Machine Learning Research*, 2015.

[126] Odile Macchi. The coincidence approach to stochastic point processes. *Advances in Applied Probability*, pages 83–122, 1975.

[127] Odile Macchi. The Fermion process—a model of stochastic point process with repulsive points. In *Transactions of the Seventh Prague Conference on Information Theory, Statistical Decision Functions, Random Processes and of the 1974 European Meeting of Statisticians*, pages 391–398. Springer, 1977.

[128] Chris Maddison, Daniel Tarlow, and Tom Minka. A* sampling. In *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[129] Neal Madras and Dana Randall. Markov chain decomposition for convergence rate analysis. *Annals of Applied Probability*, pages 581–606, 2002.

[130] Avner Magen and Anastasios Zouzias. Near optimal dimensionality reductions that preserve volumes. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, pages 523–534. 2008.

[131] Adam Marcus. Polynomial convolutions and (finite) free probability, 2016.

[132] Zelda Mariet and Suvrit Sra. Fixed-point algorithms for learning determinantal point processes. In *Int. Conference on Machine Learning (ICML)*, pages 2389–2397, 2015.

[133] Zelda Mariet and Suvrit Sra. Diversity networks. In *Int. Conference on Learning Representations (ICLR)*, 2016.

[134] Russell Martin and Dana Randall. Sampling adsorbing staircase walks using a new Markov chain decomposition method. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 492–502, 2000.

[135] Gérard Meurant. The computation of bounds for the norm of the error in the conjugate gradient algorithm. *Numerical Algorithms*, pages 77–87, 1997.

[136] Gérard Meurant. Numerical experiments in computing bounds for the norm of the error in the preconditioned conjugate gradient algorithm. *Numerical Algorithms*, pages 353–365, 1999.

[137] Alan Miller and Nam-Ky Nguyen. A Fedorov exchange algorithm for D-optimal design. *Journal of the royal statistical society*, 1994.

[138] Michel Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques*, pages 234–243. Springer, 1978.

[139] Baharan Mirzasoleiman, Ashwinkumar Badanidiyuru, Amin Karbasi, Jan Vondrák, and Andreas Krause. Lazier than lazy greedy. In *Proc. AAAI Conference on Artificial Intelligence*, pages 1812–1818, 2015.

[140] Max Morris and Toby Mitchell. Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference*, 43:381–402, 1995.

[141] George Nemhauser, Laurence Wolsey, and Marshall Fisher. An analysis of approximations for maximizing submodular set functions—I. *Mathematical programming*, pages 265–294, 1978.

[142] Nam-Ky Nguyen and Alan Miller. A review of some exchange algorithms for constructng discrete optimal designs. *Computational Statistics and Data Analysis*, 14:489–498, 1992.

[143] Aleksandar Nikolov and Mohit Singh. Maximizing determinants under partition constraints. In *Symposium on the Theory of Computing (STOC)*, 2016.

[144] Evert Nyström. Über die praktische Auflösung von Integralgleichungen mit Anwendungen auf Randwertaufgaben. *Acta Mathematica*, pages 185–204, 1930.

[145] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. *The PageRank citation ranking: bringing order to the Web.* Stanford InfoLab, 1999.

[146] Christopher Paige. *The computation of eigenvalues and eigenvectors of very large sparse matrices.* PhD thesis, University of London, 1971.

[147] Beresford Parlett and David Scott. The Lanczos algorithm with selective orthogonalization. *Mathematics of Computation*, pages 217–238, 1979.

[148] Robin Pemantle. Towards a theory of negative dependence. *Journal of Mathematical Physics*, pages 1371–1390, 2000.

[149] Robin Pemantle. Hyperbolicity and stable polynomials in combinatorics and probability. *arXiv preprint arXiv:1210.3231*, 2012.

[150] Robin Pemantle and Yuval Peres. Concentration of Lipschitz functionals of determinantal and other strong Rayleigh measures. *Combinatorics, Probability and Computing*, pages 140–160, 2014.

[151] Friedrich Pukelsheim. *Optimal design of experiments.* SIAM, 2006.

[152] Rodolphe Radau. Étude sur les formules d'approximation qui servent à calculer la valeur numérique d'une intégrale définie. *J. de Mathématiques Pures et Appliquées*, pages 283–336, 1880.

[153] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1177–1184, 2008.

[154] Gé rard Meurant. *The Lanczos and conjugate gradient algorithms: from theory to finite precision computations*, volume 19. SIAM, 2006.

[155] Carl Rasmussen. Gaussian processes in machine learning. In *Advanced lectures on machine learning*, pages 63–71. 2004.

[156] Patrick Rebeschini and Amin Karbasi. Fast mixing for discrete point processes. *Conference on Learning Theory (COLT)*, 2015.

[157] Alessandro Rudi, Raffaello Camoriano, and Lorenzo Rosasco. Less is more: Nyström computational regularization. *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[158] Laurent Schwartz, Laurent Schwartz, Laurent Schwartz, Laurent Schwartz, and France Mathematician. *Mathematics for the physical sciences*. Hermann Paris, 1966.

[159] John Scott. *Social network analysis*. Sage, 2012.

[160] Amar Shah and Zoubin Ghahramani. Determinantal clustering processes-a nonparametric Bayesian approach to kernel based semi-supervised clustering. *arXiv preprint arXiv:1309.6862*, 2013.

[161] Aidean Sharghi, Ali Borji, Chengtao Li, Tianbao Yang, and Boqing Gong. Improving sequential determinantal point processes for supervised video summarization. *Europ. Conference on Computer Vision (ECCV)*, 2018.

[162] Hao Shen, Stefanie Jegelka, and Arthur Gretton. Fast kernel-based independent component analysis. *IEEE Transactions on Signal Processing*, pages 3498–3511, 2009.

[163] Jack Sherman and Winifred Morrison. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *The Annals of Mathematical Statistics*, pages 124–127, 1950.

[164] Jonathan Shewchuk. An introduction to the conjugate gradient method without the agonizing pain, 1994.

[165] Roger Sidje and Yousef Saad. Rational approximation to the Fermi–Dirac function with applications in density functional theory. *Numerical Algorithms*, pages 455–479, 2011.

[166] Alistair Sinclair. Improved bounds for mixing rates of Markov chains and multicommodity flow. *Combinatorics, probability and Computing*, pages 351–370, 1992.

[167] David Smith and Jason Eisner. Dependency parsing by belief propagation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2008.

[168] Alex Smola and Bernhard Schölkopf. Sparse greedy matrix approximation for machine learning. *Int. Conference on Machine Learning (ICML)*, 2000.

[169] Jasper Snoek, Richard Zemel, and Ryan Adams. A determinantal point process latent variable model for inhibition in neural spiking data. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1932–1940, 2013.

[170] Alexander Soshnikov. Determinantal random point fields. *Russian Mathematical Surveys*, 55(5):923–975, 2000.

[171] Daniel Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. *SIAM Journal on Computing*, pages 1913–1926, 2011.

[172] Daniel Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Symposium on Theory of Computing (STOC)*, 2004.

[173] Josef Stoer and Roland Bulirsch. *Introduction to numerical analysis*, volume 12. Springer Science & Business Media, 2013.

[174] Damian Straszak and Nisheeth Vishnoi. Real stable polynomials and matroids: Optimization and counting. In *Symposium on Theory of Computing (STOC)*, pages 370–383, 2017.

[175] Shiliang Sun, Jing Zhao, and Jiang Zhu. A review of Nyström methods for large-scale machine learning. *Information Fusion*, pages 36–48, 2015.

[176] Maxim Sviridenko, Jan Vondrák, and Justin Ward. Optimal approximation for submodular and supermodular optimization with bounded curvature. In *SIAM-ACM Symposium on Discrete Algorithms (SODA)*, 2015.

[177] Ameet Talwalkar, Sanjiv Kumar, Mehryar Mohri, and Henry Rowley. Large-scale SVD and manifold learning. *Journal of Machine Learning Research*, pages 3129–3152, 2013.

[178] Ameet Talwalkar, Sanjiv Kumar, and Henry Rowley. Large-scale manifold learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[179] Mikhail Tsitsvero, Sergio Barbarossa, and Paolo Di Lorenzo. Signals on graphs: Uncertainty principle and sampling. *IEEE Transactions on Signal Processing*, 64(18):4845–4860, 2016.

[180] David Wagner. Multivariate stable polynomials: theory and applications. *Bulletin of the American Mathematical Society*, pages 53–84, 2011.

[181] Shusen Wang, Chao Zhang, Hui Qian, and Zhihua Zhang. Using the matrix ridge approximation to speedup determinantal point processes sampling algorithms. In *Proc. AAAI Conference on Artificial Intelligence*, pages 2121–2127, 2014.

[182] Yining Wang, Adams Wei Yu, and Aarti Singh. On computationally tractable selection of experiments in measurement-constrained regression models. *Journal of Machine Learning Research*, pages 5238–5278, 2017.

[183] WR Wasow. A note on the inversion of matrices by random walks. *Mathematical Tables and Other Aids to Computation*, pages 78–81, 1952.

[184] Christopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems (NIPS)*, pages 682–688, 2001.

[185] Jian Zhang, Josip Djolonga, and Andreas Krause. Higher-order inference for multi-class log-supermodular models. In *Int. Conference on Computer Vision (ICCV)*, pages 1859–1867, 2015.

[186] Kai Zhang, Ivor Tsang, and James Kwok. Improved Nyström low-rank approximation and error analysis. In *Int. Conference on Machine Learning (ICML)*, pages 1232–1239, 2008.

[187] Zhihua Zhang. The matrix ridge approximation: algorithms and applications. *Machine Learning*, pages 227–258, 2014.

[188] Yingbo Zhao, Fabio Pasqualetti, and Jorge Cortés. Scheduling of control nodes for improved network controllability. In *2016 IEEE Conference on Decision and Control (CDC)*, pages 1859–1864, 2016.

[189] Tao Zhou, Zoltán Kuscsik, Jian-Guo Liu, Matúš Medo, Joseph Wakeling, and Yi-Cheng Zhang. Solving the apparent diversity-accuracy dilemma of recommender systems. *Proc. National Academy of Sciences of the United States of America (PNAS)*, pages 4511–4515, 2010.

[190] Rong Zhu, Ping Ma, Michael Mahoney, and Bin Yu. Optimal subsampling approaches for large sample linear regression. *arXiv preprint arXiv:1509.05111*, 2015.