Towards efficient quantum algorithms for optimization and sampling

Dániel Szilágyi

Université Paris Cité

École Doctorale Sciences Mathématiques de Paris Centre (386)

Institut de Recherche en Informatique Fondamentale

Towards efficient quantum algorithms for optimization and sampling

par Dániel SZILÁGYI

Thèse de Doctorat en Informatique

dirigée par Iordanis KERENIDIS

Présentée et soutenue publiquement le 20 Octobre 2022, devant un jury composé de :

Iordanis KERENIDIS	Directeur de Recherche, CNRS, IRIF, Paris	*Directeur de thèse*
Alain DURMUS	Maître de Conférence, ENS Paris-Saclay	*Rapporteur*
Jérémie ROLAND	Professeur, Université Libre de Bruxelles	*Rapporteur*
Omar FAWZI	Directeur de Recherche, ENS de Lyon	*Examinateur*
Stacey JEFFERY	Senior Researcher, CWI, Amsterdam	*Examinatrice*
Sophie LAPLANTE	Directrice de Recherche, CNRS, IRIF, Paris	*Examinatrice*

Acknowledgments

Three years ago, that February Thursday was a day just like any other – until it wasn't. I was minding my own business at the office, thinking that I solved my internship problem (spoiler: I did not), when I got a short email from Iordanis: "Could you come to my office when you have five minutes?". My mind raced to remember if I did anything wrong, or if I managed to break something during the first month of my internship at IRIF. It turned out that I did neither, but there was a call for applications for a PhD position, closing the following day. We applied without much hesitation and got the funding several months later. After three years of work, I ended up with some Theorems that I present in the rest of this document. However, as the saying goes, maybe the *real* ~~treasure~~ thesis was the friends that we made along the way. So, now it is time to *acknowledge* and thank those people.

First of all, I would like to thank Iordanis for offering me the opportunity and guiding me through the highs and lows of this thesis. I could not have wished for a better supervisor. Together with Ale, Jonas and Amine, we spent many a productive meeting discussing the future of quantum computing. Next, there is Sander, who became my academic-adoptive-older-brother, he put up a truly valiant effort fielding my questions about research, cycling, and everything in between. Similarly, this thesis would not have existed without my coauthors Simon and Anupam, who taught me about scientific rigor, meticulousness, as well as a great deal of mathematics. Finally, I would like to thank all the members of the jury, who read this thesis with great care.

Of course, I did not spend all (or even most) of my time hunched over a piece of paper. An integral part of my experience at IRIF were the lunches, long coffee breaks, and after-work activities. This all started with Simon, Yassine, Mikaël, Zhou, Amaury and Yixin, who kept inviting me to join them for lunch every day for six months until I finally could. Then came my office-mates Robin and Avi, who provided support when the morale was low, and quality banter when the spirits were high. The atmosphere at IRIF was remarkably international, as I could speak Serbian with Simona, Hungarian with Dániel, Slovene with Klara, and English and French with everyone else. Many awesome people came and left in the subsequent years, and many a drink was drunk with Pierre, Anupa, Filippo, Enrique, Wael, Mouna, Gabriel, Geoffroy and Patrick. Thanks Frédéric for cultivating such a welcoming atmosphere in the lab, and thanks Eva, Etienne, Jemuel and Maximilien for helping me out when not everything was smooth sailing.

I can not complain about the life outside of the office, either. For that, I can thank my flatmate Ognjen, who kept me fed and supplied with the finest specialty coffee one can buy in Europe. Speaking of which, many have influenced my life these years from different corners of Europe. Tapping away on their phones from the Netherlands, Serbia and Finland, Aleksandar, Dušan and Marko provided me with a ~~steady stream~~ firehose of distractions, as well as occasional nuggets of wisdom about the research questions I had. I shared countless technical discussions with Joel in Sweden, while Marko and Slađa made me feel like at home in Belgium.

Still, there is no place like home (or, as I started calling it, "*home* home"). Whenever I visited Novi Sad, I knew that I could always relax and unwind with my high school friends Miloš, Vukašin[2], Nikola, Ilija, Zoran and Nemanja. No visit home would be complete without a visit to Belgrade, where I always enjoyed catching up with my friends Vladan, Nikola, Marija and Ema from Petnica. Finally, I would like to thank my family, who have stood by me for the last 27 years. Without the continued support of my parents Rozália and Csaba, and my sisters Krisztina and Éva, I would not be where I am now.

Résumé

Récemment, avec l'avènement du big data et de l'apprentissage machine à grande échelle, il y a eu une demande croissante pour des algorithmes quantiques qui seraient plus directement applicables à des problèmes pertinents en pratique. Dans cette thèse, nous présentons trois algorithmes qui visent à nous rapprocher de cet objectif.

Premièrement, nous développons un algorithme quantique pour l'optimisation conique du second ordre, une classe de problèmes d'optimisation qui se situe entre les programmes linéaires et semi-définis en termes d'expressivité et de facilité de résolution. Ces problèmes sont le plus souvent résolus avec l'aide d'une méthode de point intérieur, dont la complexité est principalement imposée par le coût de la résolution d'une série de systèmes linéaires. Dans notre algorithme, nous résolvons ces systèmes linéaires de manière approximative à l'aide d'un algorithme quantique, et nous prouvons que la méthode des points intérieurs qui en résulte converge vers la solution correcte en un même nombre d'itérations. Nous donnons des preuves numériques que l'algorithme fournit des accélérations de bout en bout dans certaines applications de faible précision telles que les machines à vecteurs de support et l'optimisation de portefeuille financier.

L'algorithme des systèmes linéaires quantiques que nous utilisons est le résultat d'une longue ligne de recherche engendrée par l'algorithme de système linéaire quantique de Harrow, Hassidim et Lloyd. Bien qu'il ait été prouvé qu'il est asymptotiquement optimal, le circuit correspondant compliqué et nécessite un prétraitement classique. La deuxième contribution de cette thèse est un algorithme quantique amélioré pour les systèmes linéaires, basé sur la méthode classique optimale de l'itération de Chebyshev.

Enfin, nous observons que les algorithmes susmentionnés ont la propriété commune d'approximer l'unique vraie solution du problème donné. En général, la conception de tels algorithmes quantiques est difficile, car souvent la seule façon de récupérer (une approximation de) la vraie solution est d'exécuter l'algorithme plusieurs fois (soit naïvement, soit par amplification d'amplitude) et de calculer certaines statistiques (par exemple la moyenne) des sorties mesurées. Si les ordinateurs quantiques donnent intrinsèquement accès à l'échantillonnage de leurs sorties, pourrions-nous les exploiter pour accélérer les problèmes d'échantillonnage classiques ? De manière surprenante, il s'avère que l'échantillonnage gaussien n'est pas un problème complètement résolu, même sur les ordinateurs classiques. Notre troisième contribution est un algorithme de Monte Carlo hamiltonien classique pour l'échantillonnage gaussien dans le modèle d'interrogation du premier ordre. Nous surmontons et contournons plusieurs bornes inférieures en prenant des pas longues et aléatoires pour intégrer le hamiltonien au cœur de l'algorithme.

Mot-clés : algorithmes quantiques, systèmes linéaires, optimisation conique du second ordre, machines à vecteurs de support, optimisation de portefeuille, échantillonnage, Monte Carlo hamiltonien.

Summary

Recently, with the advent of big data and large-scale machine learning, there has been an increasing demand for quantum algorithms that would be more directly applicable to practically-relevant problems. In this thesis we present three algorithms that aim to take us closer to this goal.

Firstly, we develop a quantum algorithm for second-order cone programming, a class of optimization problems that is between linear and semidefinite programs in terms of expressivity and ease of solving. These problems are most commonly solved using interior-point methods, the complexity of which is mainly dictated by the cost of solving a series of linear systems. In our algorithm we solve these linear systems approximately using a quantum linear system solver, and prove that the resulting interior-point method converges to the correct solution in the same number of iterations. We give numerical evidence that the algorithm provides end-to-end speedups in certain low-precision applications such as support-vector machines and portfolio optimization.

The quantum linear system solver that we use is the result of a long line of research spawned by the quantum linear system algorithm of Harrow, Hassidim and Lloyd. While it has been proven to be asymptotically optimal, its corresponding circuit is nontrivial and requires some classical preprocessing. The second contribution of this thesis is an improved quantum algorithm for linear systems, based on the optimal classical method of Chebyshev iteration.

Finally, we observe that the aforementioned algorithms have the common property of approximating the unique true solution of the input problem. In general, designing such quantum algorithms is nontrivial, as often the only way of recovering (an approximation to) the true solution is by running the algorithm many times (either naively or via amplitude amplification) and computing some statistics (e.g. the mean) of the measured outputs. If quantum computers intrinsically provide sampling access to their outputs, could we exploit them to speed up classically-relevant sampling problems? Surprisingly, it turns out that Gaussian sampling is not a completely solved problem, even on classical computers. Our third contribution is a classical Hamiltonian Monte Carlo algorithm for Gaussian sampling in first-order query model. We overcome and sidestep several lower bounds by taking long and random steps for integrating the Hamiltonian in the heart of the algorithm.

Keywords: quantum algorithms, linear systems, second-order optimization, support-vector machines, portfolio optimization, sampling, Hamiltonian Monte Carlo.

Résumé long

Introduction

Il y a environ 40 ans, lorsqu'il est devenu évident que l'ordinateur était devenu un succès retentissant, Richard Feynman a proposé un nouveau paradigme révolutionnaire pour s'attaquer aux problèmes de calcul du 21e siècle, qu'il a appelé l'*ordinateur quantique*. Il est rapidement apparu que les ordinateurs quantiques pouvaient fournir des accélérations dans des domaines sans rapport avec leur objectif initial de simulation de la mécanique quantique. Alors que le travail de fond de Deutsch et Penrose [DP85] et les premiers algorithmes de Bernstein et Vazirani [BV97] et Simon [Sim97] manquaient d'applications pratiques immédiates, elles ont ouvert la voie au résultat historique de Shor [Sho94], qui a exploité la transformation de Fourier quantique pour calculer les logarithmes discrets en temps polynomial. Cet algorithme a suscité un grand intérêt dans ce domaine naissant, ce qui a entraîné des avancées en cryptographie [Gis+02 ; BL17], théorie de la complexité [Wat09 ; Ji+21], théorie du codage [LB13], ainsi que de nouveaux algorithmes quantiques [Gro96 ; Bra+02 ; Sze04], pour n'en citer que quelques-uns. Nous renvoyons le lecteur à [Pre21] pour un aperçu récent du domaine, ainsi qu'à [NC12] pour une discussion approfondie des résultats classiques.

Récemment, avec l'avènement du big data et de l'apprentissage automatique à grande échelle, il y a eu une demande croissante d'algorithmes quantiques qui seraient plus directement applicables à des problèmes pertinents en pratique. Le travail de Harrow, Hassidim et Lloyd [HHL09] a fourni exactement cela : un algorithme quantique permettant de résoudre un système linéaire en un temps poly-logarithmique dans la taille. Bien sûr, un tel temps d'exécution ne peut être atteint qu'avec un modèle d'entrée et de sortie spécifique [Aar15], puisque lire naïvement l'entrée et écrire la sortie aurait une complexité qui est quadratique (resp. linéaire) dans la dimension du système. Plus précisément, l'algorithme de [HHL09] suppose que l'entrée est donnée comme un oracle quantique qui permet d'interroger les éléments de l'entrée en superposition, et fournit une sortie sous la forme de l'état quantique, qui peut être mesuré afin de récupérer un seul échantillon de la distribution induite par le vecteur solution normalisé (approximatif). A première vue, l'utilité d'un tel algorithme est discutable, avec un modèle d'entrée trop fort, et une sortie donnée sous une forme peu pratique. Néanmoins, il s'avère que ce modèle d'entrée est raisonnable pour les matrices structurées (sous le nom de *codage en bloc* [CGJ19]) tandis que la sortie du vecteur d'état peut être utilisée dans le cas où il est creux, ou lorsque le système linéaire lui-même découle d'un problème d'échantillonnage. Par exemple, l'algorithme du système de recommandation quantique de Kerenidis et Prakash [KP17] fonctionne avec un modèle d'entrée étroitement lié et est construit de manière à ce que l'échantillonnage soit la forme "naturelle" et souhaitée de la sortie.

Après [HHL09], la recherche a bifurqué dans deux directions différentes. D'une part, la communauté a travaillé à l'amélioration et à l'extension des éléments de base de l'*algèbre linéaire quantique*. Notamment, Low, Yoder et Chuang [LYC16] ainsi que Low et Chuang [LC17a ; LC17b ; LC19] ont introduit les techniques de *qubitisation* et de *traitement du signal quantique*, qui ont ensuite été améliorées et généralisées par le cadre de la *transformation quantique de la valeur singulière*

de Gilyén, Su, Low et Wiebe [Gil+19]. Ce cadre permet d'appliquer un polynôme arbitraire à une matrice codée en bloc, ce qui permet d'exprimer la simulation hamiltonienne, la résolution de systèmes linéaires, l'amplification d'amplitude et de nombreux autres algorithmes de manière unifiée [Mar+21]. D'autre part, de nombreux travaux ont été réalisés pour combiner ces éléments de base afin de résoudre des problèmes plus compliqués (plus "de haut niveau"). Une ligne de recherche notable a été lancée par Brandao et Svore [BS17b], qui ont proposé un algorithme de résolution de programmes semi-définis (PSD). Ce travail a ensuite été amélioré par Brandão, Kalev, Li, Lin, Svore et Wu [Bra+19], van Apeldoorn, Gilyén, Gribling et de Wolf [vApe+17] et Apeldoorn et Gilyén [AG19]. Le solveur PSD de [KP20a] utilise une approche légèrement différente, il est basé sur un analogue quantique robuste d'une méthode classique de point intérieur (l'algorithme classique habituellement utilisé pour résoudre les PSD).

Dans la première moitié de cette thèse, nous nous appuyons sur les deux branches du riche corpus de travaux mentionné ci-dessus. Nous présentons deux résultats principaux, dans l'ordre chronologique : Chapitre 2 contient un algorithme (quantique) pour la programmation conique du second ordre, et Chapitre 3 contient un algorithme quantique amélioré pour les systèmes linéaires. Le premier est basé sur une analyse robuste d'une méthode classique de point intérieur [MT00 ; AG03], et utilise un solveur de système linéaire quantique [HHL09 ; CKS17 ; CGJ19 ; Gil+19] pour accélérer sa boucle interne. Ce dernier contient une légère amélioration des algorithmes de systèmes linéaires susmentionnés, basée sur la méthode classique optimale d'itération de Tchebychev [Var00].

Les algorithmes de Chapitres 2 et 3 ont la propriété commune d'approcher l'unique vraie solution du problème d'entrée. En général, la conception de tels algorithmes quantiques n'est pas triviale, car souvent la seule façon de récupérer (une approximation de) la vraie solution est d'exécuter l'algorithme de nombreuses fois (soit naïvement, soit par amplification d'amplitude) et de calculer certaines statistiques (par exemple la moyenne) des sorties mesurées. Même si l'on peut borner précisément le nombre d'exécutions nécessaires ainsi que la complexité du post-traitement classique, une telle approche laisse un peu à désirer. Si les ordinateurs quantiques fournissent intrinsèquement un accès d'échantillonnage à leurs sorties, pourrions-nous les exploiter pour accélérer les problèmes d'échantillonnage classiques ? Une approche évidente pour ce faire serait de remplacer l'optimisation locale dans chaque itération de la méthode de point intérieur de Chapitre 2 par une procédure simple qui choisit l'itération suivante au hasard dans un voisinage du point courant. Il s'avère que cette marche aléatoire peut être utilisée pour échantillonner un point uniforme dans un polytope, et a déjà été analysée par Kannan et Narayanan [KN12], Narayanan [Nar16] et Sachdeva et Vishnoi [SV16] sous le nom de la *marche de Dikin*. En particulier, le principal élément constitutif de [Nar16] est une procédure d'échantillonnage à partir d'une loi gaussienne liée au hessien d'une certaine fonction barrière.

Étonnamment, selon le modèle d'entrée, l'échantillonnage gaussien n'est pas un problème complètement résolu, même sur les ordinateurs classiques. Bien sûr, dans le cas le plus simple où la matrice de covariance est complètement donnée, il semble que l'on ne puisse pas faire beaucoup mieux que l'algorithme traditionnel qui calcule (ou approche) l'inverse de la racine carrée de la matrice de covariance. Le modèle de requête du premier ordre, où l'on compte les requêtes sur le gradient de la log-densité, est plus intéressant. Ce modèle est étroitement lié aux différents modèles d'entrée quantique dont nous avons parlé, et est le modèle naturel pour certaines applications telles que l'échantillonnage de Thompson pour les bandits contextuels [Rus+18]. Les meilleurs algorithmes de ce modèle sont basés sur les méthodes de Monte Carlo hamiltonien [Dua+87] et Langevin [Bes94 ; RT96 ; RR98]. Ces algorithmes effectuent une marche aléatoire en intégrant de manière répétée la dynamique hamiltonienne avec des conditions initiales aléatoires. Leur performance est limitée par le nombre et la longueur des pas que l'on utilise pour approximer les intégrales, et les algorithmes actuels ont été conjecturés comme étant optimaux [CV22 ; LST21]. Dans la seconde partie de la

thèse (Chapitre 4), nous contournons ces bornes inférieures avec un algorithme qui prend des étapes d'intégration *longues* et *aléatoires*.

Nous décrivons nos contributions de manière plus détaillée.

Algorithmes quantiques

L'ensemble du domaine de l'algèbre linéaire quantique est rendu possible par une simple observation : nous pouvons identifier un vecteur unitaire $\mathbf{x} \in \mathbb{R}^{2^k}$ avec l'état $|\mathbf{x}\rangle := \sum_{i=1}^{2^k} x_i |i\rangle$ sur seulement k qubits. Dans ce cadre, les matrices doivent avoir une norme bornée, nous encodons donc une matrice $A \in \mathbb{R}^{2^k \times 2^k}$ dans le coin supérieur gauche d'un opérateur unitaire $(k+a)$-qubit U_A de sorte que $A = \zeta(\langle 0|^{\otimes a} \otimes I) U_A (|0\rangle^{\otimes a} \otimes I)$ pour un certain $\zeta \geq \|A\|_2$. Nous appelons U_A un ζ-codage en bloc de A. Maintenant, appliquer A à $\mathbf{x}$ revient à appliquer le codage en bloc U_A à $|0\rangle^{\otimes a} |\mathbf{x}\rangle$ et à post-sélectionner le résultat sur le premier registre restant $|0\rangle^{\otimes a}$. Il est important de noter que cette post-sélection réussit avec une probabilité de $\|A\mathbf{x}/\zeta\|^2$, qui peut être amplifiée jusqu'à une constante au coût de $O(\zeta/\|A\mathbf{x}\|)$ – le facteur de subnormalisation ζ joue un rôle crucial dans Chapitre 3.

Une fois que nous pouvons calculer le produit matrice-vecteur, le problème suivant évident est la résolution de systèmes linéaires. Pour une matrice A et un vecteur $\mathbf{b}$, le problème des systèmes linéaires quantiques demande de préparer l'état correspondant au vecteur $A^{-1}\mathbf{b}$. Sans perte de généralité, nous supposons que A est hermitienne et a des valeurs singulières dans l'intervalle $[1/\kappa, 1]$ (ce qui implique qu'elle a un *conditionnement* κ). En gardant cela à l'esprit, le "meilleur" (en termes de sous-normalisation) que nous puissions espérer est un κ-codage en bloc de A^{-1}. Cependant, on ne sait pas comment construire ce codage en bloc (*exact*) autrement qu'en inversant A sur un ordinateur classique et en appliquant les méthodes standard de codage en bloc des matrices denses génériques, ce qui annule toute amélioration de la complexité par rapport à un algorithme purement classique.

Heureusement, la TQVS (transformation quantique des valeurs singulières, [Gil+19]) et la CLU (combinaison linéaire des unitaires, [BCK15]) nous permettent de construire des codages en blocs de $p(A)$ pour les polynômes p, avec diverses sous-normalisations. Dans le cas de CLU, la sous-normalisation est $\|\mathbf{c}_p\|_1$, où $\mathbf{c}_p$ est le vecteur des coefficients dans le développement de Tchebychev $p(x) = \sum_{i=0}^{n} c_{p,i} \mathcal{T}_i(x)$. Dans le cas de TQVS, la sous-normalisation est simplement $\max_{x\in[-1,1]} |p(x)|$. Observons que la sous-normalisation CLU est au moins aussi grande que celle pour TQVS, puisque tous les polynômes avec $\|\mathbf{c}_p\|_1 \leq 1$ sont bornés par 1 sur $[-1, 1]$. Par conséquent, afin de construire un codage en bloc *approché* de A^{-1}, le polynôme p doit approximer l'inverse sur $D_\kappa = [-1, -1/\kappa] \cup [1/\kappa, 1]$, et ne pas être beaucoup plus grand que κ sur $[-1/\kappa, 1/\kappa]$. Un exemple simple d'un tel polynôme est $p_n(x) = \frac{1-(1-x^2)^n}{x}$, qui approche $1/x$ sur D_κ pour $n \geq \kappa^2 \log(\kappa/\varepsilon)$. Les coefficients de Tchebychev de ce polynôme peuvent être interprétés comme des probabilités binomiales, leur norme 1 est donc bornée et, par conséquent, p_n peut être évalué efficacement en utilisant à la fois TQVS et CLU. L'idée de Childs, Kothari et Somma [CKS17] était que l'expansion de Tchebychev de p_n peut être tronquée après $\widetilde{O}(\kappa)$ termes donnant une approximation polynomiale *asymptotiquement* optimale de l'inverse sur D_κ. Dans Chapitre 3, nous considérons *le* polynôme *optimal* de ce type,

$$q_n(x) := \frac{1 - \mathcal{T}_n(\frac{1+1/\kappa^2-2x^2}{1-1/\kappa^2}) / \mathcal{T}_n(\frac{1+\kappa^2}{1-\kappa^2})}{x}.$$

Nous montrons que la norme 1 des coefficients de Tchebychev de q_n admet une borne encore meilleure, impliquant des algorithmes optimaux de systèmes linéaires quantiques basés sur CLU et

TQVS. Étant donné la simple borne supérieure polynomiale (c'est-à-dire $O(\sqrt{\text{degré}})$) de l'écart entre TQVS et CLU pour des polynômes arbitraires, nous étudions également l'efficacité des algorithmes basés sur CLU pour l'approximation de certaines autres fonctions d'intérêt. Nous montrons que même les fonctions discontinues telles que la fonction signe ont une norme 1 des coefficients de Tchebychev bien bornée, et nous conjecturons que la simulation hamiltonienne (et en particulier $\cos(\kappa x)$ et $\sin(\kappa x)$ pour $\kappa > 1$) montre une séparation polynomiale entre CLU et TQVS. Nous concluons le Chapitre avec un algorithme simple basé sur la CLU imbriquée pour la simulation hamiltonienne.

Dans Chapitre 2, nous appliquons la machinerie des systèmes linéaires de Chapitre 3 au problème plus "compliqué" de la programmation conique du second ordre (PCSO), une classe de problèmes d'optimisation convexes qui se situe entre les programmes linéaires (PL) et semi-définis (PSD) en termes d'expressivité. Un algorithme efficace pour la programmation conique du second ordre produirait également des algorithmes efficaces pour de nombreux problèmes intéressants, tels que la programmation quadratique convexe (standard et sous contrainte quadratique), l'optimisation de portefeuille, et bien d'autres [AG03]. Dans Chapitre 2, nous présentons un algorithme quantique pour PCSO, ainsi que deux problèmes candidats pour des accélérations quantiques de bout en bout à l'aide de cet algorithme : les machines à vecteurs de support (MVS) et l'optimisation de portefeuille.

Dans PCSO, on doit optimiser un objectif linéaire $\mathbf{c}^\top \mathbf{x}$ soumis à des contraintes linéaires de la forme $A\mathbf{x} = \mathbf{b}$, ainsi qu'une contrainte de second ordre $\mathbf{x} \in \mathcal{L}^{n_1} \times \cdots \times \mathcal{L}^{n_r}$ sur les blocs de $\mathbf{x}$, où le cône de second ordre $\mathcal{L}_k \subseteq \mathbb{R}^k$ est défini comme $\mathcal{L}^k = \left\{ \mathbf{x} = (x_0; \tilde{\mathbf{x}}) \in \mathbb{R}^k \;\middle|\; \|\tilde{\mathbf{x}}\| \leq x_0 \right\}$. La méthode standard pour résoudre PCSO est la méthode de point intérieur (MPI), découverte pour la première fois par Karmarkar [Kar84], et généralisée à tous les cônes symétriques (une classe qui inclut le cône du second ordre) par Nesterov et Todd [NT97 ; NT98]. Une méthode classique de point intérieur résout un problème d'optimisation sur des cônes symétriques en commençant par une solution admissible et en trouvant itérativement des solutions avec un saut de dualité plus petit tout en maintenant l'admissibilité. Une seule étape itérative consiste à résoudre un système d'équations linéaires appelé système linéaire de Newton et à mettre à jour l'itération courante en utilisant les solutions du système de Newton. L'analyse d'une MPI classique montre qu'à chaque itération, les solutions mises à jour restent admissibles et que le saut de dualité est réduit d'un facteur de $(1 - \alpha/\sqrt{n})$ où n est la dimension du problème d'optimisation et $\alpha > 0$ est une constante. L'algorithme converge donc vers une solution admissible avec un saut de dualité ϵ en $O(\sqrt{n}\log(1/\epsilon))$ itérations.

Une méthode de point intérieur quantique [KP20a] utilise un solveur de système linéaire quantique au lieu d'un solveur classique dans chaque itération de la MPI. Cependant, il existe une différence importante entre les procédures d'algèbre linéaire classique et quantique pour résoudre le système linéaire $A\mathbf{x} = \mathbf{b}$. Contrairement aux solveurs de systèmes linéaires classiques qui renvoient une description exacte de $\mathbf{x}$, les procédures de tomographie quantique peuvent renvoyer une solution ε précise $\overline{\mathbf{x}}$ telle que $\|\overline{\mathbf{x}} - \mathbf{x}\| \leq \varepsilon\|\mathbf{x}\|$ avec $O(n/\varepsilon^2)$ d'exécutions du solveur de système linéaire quantique. De plus, ces solveurs de systèmes linéaires requièrent que A et $\mathbf{b}$ soient donnés sous forme de codage en bloc [CGJ19], ce modèle d'entrée est donc également utilisé par notre algorithme. L'un des principaux défis techniques dans le développement d'une méthode quantique de points intérieurs (MQPI) est d'établir la convergence de la MPI classique qui utilise des solutions ε-approchées du système linéaire de Newton (dans la norme ℓ_2) au lieu des solutions exactes dans la MPI classique.

La méthode de point intérieur quantique pour les programmes de cônes du second ordre nécessite des idées supplémentaires allant au-delà de celles utilisées pour les méthodes de points intérieurs quantiques pour les PSD [KP20a]. En particulier, les méthodes de points intérieurs pour PCSO peuvent être décrites en utilisant le cadre de l'algèbre de Jordan euclidienne [MT00]. Le cadre de

l'algèbre de Jordan euclidienne fournit des analogues de concepts tels que les valeurs propres, les normes spectrales et de Frobenius pour les matrices et les contraintes semi-définies positives pour le cas des PCSO. En utilisant ces idées conceptuelles du cadre de l'algèbre de Jordan euclidienne [MT00] et l'analyse de la méthode approximative de point intérieur PSD [KP20a] nous fournissons une MPI approximative pour PCSO qui converge en $O(\sqrt{n}\log(1/\epsilon))$ itérations. Les MPI approximatifs pour PCSO n'ont pas été étudiés auparavant dans la littérature d'optimisation classique ou quantique ; cette analyse est la principale contribution technique de Chapitre 2.

Algorithmes classiques

L'une des tâches les plus importantes en statistique et en apprentissage automatique consiste à échantillonner à partir de distributions hautement dimensionnelles et potentiellement compliquées. Les chaînes de Markov constituent un moyen efficace d'échantillonner de telles distributions, et il existe une grande variété d'algorithmes de chaînes de Markov conçus spécifiquement à cet objectif. En général, la principale difficulté de l'analyse de ces algorithmes est de lier le temps d'exécution précis ou le temps de mélange de la chaîne de Markov. Bien que de nombreux algorithmes soient très largement utilisés (heuristiques) depuis plusieurs décennies, les bornes rigoureuses de leurs performances sont souvent absentes. Un exemple clé est l'algorithme de *Monte Carlo Hamiltonien* (MCH) [Dua+87]. Il s'agit d'un élégant algorithme de chaîne de Markov qui utilise la dynamique hamiltonienne pour explorer efficacement l'espace d'état, sans trop s'éloigner de la région de haute probabilité. L'une de ses principales caractéristiques est qu'il surmonte le comportement lent et diffus qui est inhérent aux approches à "petits pas" telles que la marche par boules et l'algorithme de Langevin. Bien que cela soit effectivement observé dans l'utilisation et les études heuristiques de l'algorithme MCH [Nea11], les efforts récents sont principalement limités à des tailles de pas beaucoup plus courtes que les choix heuristiques [Che+20 ; CV22]. Dans ce travail, nous prouvons des bornes apparemment optimales sur l'algorithme MCH (avec intégrateur saute-mouton) pour le cas particulier des distributions gaussiennes. Il s'agit de la passerelle typique vers des distributions plus compliquées telles que les distributions logconcaves ou multimodales. Notre implémentation de MCH exploite des temps d'intégration longs et aléatoires. Cela surpasse les récents obstacles à l'échantillonnage de distributions gaussiennes à l'aide de la MCH avec des temps d'intégration courts [CV22] ou déterministes [LST21].

Nos bornes s'énoncent le plus facilement dans le "modèle de la boîte noire", où le but est d'échantillonner à partir une loi de la forme $e^{-f(\mathbf{x})}$ pour $\mathbf{x} \in \mathbb{R}^d$, et nous avons un accès par requête à la fois à f et à son gradient ∇f. Dans le cas d'une loi gaussienne, f doit être une forme quadratique $f(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})$, où $\boldsymbol{\mu}$ et Σ sont respectivement la moyenne (inconnue) et la matrice de covariance de la gaussienne. Le conditionnement κ de la distribution gaussienne est simplement le conditionnement de Σ^{-1}. Dans Chapitre 4, nous développons un algorithme MCH ajusté par Metropolis qui peut calculer un échantillon ε-approximatif à partir de la gaussienne donnée en utilisant $\widetilde{O}(\sqrt{\kappa}d^{1/4}\log(1/\varepsilon))$ d'évaluations du gradient.

Ces deux bornes semblent conformes aux attentes [Dua+87 ; Nea11 ; Bes+13], et nous pensons qu'elles sont serrées lorsque l'on utilise l'intégrateur saute-mouton habituel pour simuler la dynamique hamiltonienne. Notre algorithme dépasse la borne inférieure de $\widetilde{\Omega}(\kappa\sqrt{d})$ sur la complexité du MCH pour un échantillonnage gaussien de [LST21] en utilisant des temps d'intégration *aléatoires*. Cela permet d'éviter les problèmes de périodicité bien connus associés à un temps d'intégration déterministe.

Notre travail s'inscrit dans l'effort récent de prouver des bornes non asymptotiques (et souvent serrées) sur les algorithmes de chaînes de Markov pour les lois spécifiques telles que les lois gaussiennes et, plus généralement, les lois logconcaves (où f est supposé être convexe). La plupart de ces efforts

se sont concentrés sur des dynamiques à pas courts, comme la marche par boules, l'algorithme de Langevin et la MCH avec des temps d'intégration courts. L'utilisation de tels "pas locaux" permet de contrôler plus facilement la stabilité et la probabilité d'acceptation de l'algorithme. Cependant, la restriction aux dynamiques à pas courts est aussi ce qui ralentit ces algorithmes, et c'est ce que nous évitons dans notre algorithme MCH.

Enfin, la restriction à l'échantillonnage des distributions gaussiennes et logconcaves est précisément parallèle à la restriction aux fonctions quadratiques et convexes en optimisation. Néanmoins, un écart entre la complexité (en requete de premier ordre) de l'échantillonnage logconcave et la complexité $O(\min\{\sqrt{\kappa}, d\})$ de l'optimisation convexe est apparemment jugé plausible. Plus précisément, les auteurs de [LST20] suggèrent une borne inférieure de $\Omega(\kappa)$ pour l'échantillonnage logconcave. Notre travail montre qu'une dépendance sous-linéaire de κ est possible au moins pour le cas particulier des lois gaussiennes, et nous y voyons la preuve qu'une borne générale de $O(\sqrt{\kappa})$ pour l'échantillonnage logconcave pourrait être réalisable.

Contents

1 | Preliminaries

1.1 Introduction

Around 40 years ago, when it became clear that the computer had become a resounding success, Richard Feynman proposed a revolutionary new paradigm for tackling the computational problems of the 21st century – he called it the *quantum computer*. Soon it became apparent quantum computers could provide speedups in areas unrelated to their original purpose of simulating quantum mechanics. While the foundational work of Deutsch and Penrose [DP85] and the early algorithms of Bernstein and Vazirani [BV97] and Simon [Sim97] lacked immediate practical applications, they paved the way for the landmark result of Shor [Sho94], who leveraged the quantum Fourier transform to compute discrete logarithms in polynomial time. This algorithm has spawned a great deal of interest in the nascent field, which resulted in advances in cryptography [Gis+02; BL17], complexity theory [Wat09; Ji+21], coding theory [LB13], as well as new quantum algorithms [Gro96; Bra+02; Sze04], to name just a few. We refer the reader to [Pre21] for a recent overview of the field, as well as [NC12] for a thorough discussion of the classical results.

Recently, with the advent of big data and large-scale machine learning, there has been an increasing demand for quantum algorithms that would be more directly applicable to practically-relevant problems. The work of Harrow, Hassidim, and Lloyd [HHL09] provided just that: a quantum algorithm for solving a linear system in time that is poly-logarithmic in its size. Of course, such a running time can only be achieved with a specific input and output model [Aar15], since naively reading the input and writing the output would have a complexity that is quadratic (resp. linear) in the system dimension. More precisely, the algorithm of [HHL09] assumes that the input is given as a quantum oracle that allows querying the elements of the input in superposition, and provides an output in the form of the quantum state, that can be measured in order to recover a single sample from the distribution induced by the normalized (approximate) solution vector. At a first glance, the utility of such an algorithm is questionable, with an input model that is too strong, and the output given in an inconvenient form. Nevertheless, it turns out that this input model is reasonable for structured (e.g. sparse) matrices (under the name of *block encodings* [CGJ19]) while the state-vector output can be used in the case where it is sparse, or when the linear system itself arises from a sampling problem. For example, the quantum recommendation system algorithm of Kerenidis and Prakash [KP17] works with a closely related input model and is constructed so that sampling is the "natural" and desired form of output.

Following [HHL09], the research has branched in two different directions. On one hand, the community worked on improving an extending the basic building blocks of *quantum linear algebra*. Notably, Low, Yoder, and Chuang [LYC16] as well as Low and Chuang [LC17a; LC17b; LC19] introduced the techniques of *qubitization* and *quantum signal processing*, which have then been improved and generalized by the *quantum singular value transformation* framework of Gilyén, Su, Low, and Wiebe [Gil+19]. This framework allows one to apply an arbitrary polynomial to a block-encoded matrix, allowing one to express Hamiltonian simulation, linear system solving,

amplitude amplification, and many other algorithms in a unified way [Mar+21]. On the other hand, there has been a lot of work on combining these basic building blocks in order to solve more complicated (more "high level") problems. A notable line of research has been started by Brandao and Svore [BS17b], who proposed an algorithm for solving semidefinite programs (SDP). This work has later been improved by Brandão, Kalev, Li, Lin, Svore, and Wu [Bra+19], van Apeldoorn, Gilyén, Gribling, and de Wolf [vApe+17], and Apeldoorn and Gilyén [AG19]. The SDP solver of Kerenidis and Prakash [KP20a] uses a slightly different approach of directly developing a robust quantum analog of a classical interior-point method (the most commonly-used classical algorithm for solving SDP).

In the first half of this thesis we build upon both branches of the aforementioned rich body of work. We present two main results, in chronological order: Chapter 2 contains a (quantum) algorithm for second-order cone programming, and Chapter 3 contains an improved quantum linear systems algorithm. The former is based on a robust analysis of a classical interior-point method [MT00; AG03], and uses a quantum linear system solver [HHL09; CKS17; CGJ19; Gil+19] to speed up its inner loop. The latter contains a slight improvement of the aforementioned linear system algorithms, based on the optimal classical method of Chebyshev iteration [Var00].

The algorithms in Chapters 2 and 3 have the common property of approximating the unique true solution of the input problem. In general, designing such quantum algorithms is nontrivial, as often the only way of recovering (an approximation to) the true solution is by running the algorithm many times (either naively or via amplitude amplification) and computing some statistics (e.g. the mean) of the measured outputs. Even though one can precisely bound the number of required runs as well as the complexity of the classical postprocessing, such an approach leaves something to be desired. If quantum computers intrinsically provide sampling access to their outputs, could we exploit them to speed up classically-relevant sampling problems? An obvious approach for doing this would be by replacing the local optimization in each iteration of the interior-point method from Chapter 2 with a simple procedure that chooses the next iterate at random from a neighborhood of the current point. It turns out that this random walk can be used to sample a uniform point from a polytope, and has already been analyzed by Kannan and Narayanan [KN12], Narayanan [Nar16], and Sachdeva and Vishnoi [SV16] under the name of *Dikin walk*. In particular, the key building block of [Nar16] is a procedure for sampling from a Gaussian distribution related to the Hessian of a certain barrier function.

Surprisingly, depending on the input model, Gaussian sampling is not a completely solved problem, even on classical computers. Of course, in the simplest setting when the covariance matrix is completely given, it seems that one can not do much better than the folklore algorithm of computing (or approximating) the inverse of the square root of the covariance matrix. More interesting is the first-order query model, where one counts the queries to the gradient of the log-density. This model is closely related to the various quantum input models we discussed, and is the natural model for certain applications such as Thompson sampling for contextual bandits [Rus+18]. The best algorithms in this model are based on the Hamiltonian [Dua+87] and Langevin [Bes94; RT96; RR98] Monte Carlo methods. These algorithms perform a random walk by repeatedly integrating Hamiltonian dynamics with random initial conditions. Their performance is limited by the number and size of the steps one uses to approximate the integrals, and the current algorithms have been conjectured to be optimal [CV22; LST21]. In the second half of the thesis (Chapter 4) we circumvent these lower bounds with an algorithm that takes *long* and *random* integration steps.

In the remainder of this Chapter we describe our contributions in more detail.

1.2 Quantum algorithms

The entire area of quantum linear algebra is made possible by one simple observation: we can identify a unit vector $\mathbf{x} \in \mathbb{R}^{2^k}$ with the state $|\mathbf{x}\rangle := \sum_{i=1}^{2^k} x_i |i\rangle$ on only k qubits. In this framework, matrices need to have bounded norm, so we encode a matrix $A \in \mathbb{R}^{2^k \times 2^k}$ in the upper-left corner of a $(k+a)$-qubit unitary operator U_A so that $A = \zeta(\langle 0|^{\otimes a} \otimes I) U_A (|0\rangle^{\otimes a} \otimes I)$ for some $\zeta \geq \|A\|_2$. We call U_A a ζ-block-encoding of A. Now, applying A to $\mathbf{x}$ amounts to applying the block-encoding U_A to $|0\rangle^{\otimes a} |\mathbf{x}\rangle$ and post-selecting the result on the first register remaining $|0\rangle^{\otimes a}$. It is important to note that this post-selection succeeds with probability $\left\|A\mathbf{x}/\zeta\right\|^2$, which can be amplified up to a constant at cost $O(\zeta/\|A\mathbf{x}\|)$ – the subnormalization factor ζ plays a crucial role in Chapter 3.

Once we can compute matrix-vector product, the obvious next problem is solving linear systems. For a matrix A and a vector $\mathbf{b}$, the quantum linear systems problem asks one to prepare the state corresponding to the vector $A^{-1}\mathbf{b}/\|A^{-1}\mathbf{b}\|$. Without loss of generality, we assume that A is Hermitian and has singular values in the interval $[1/\kappa, 1]$ (implying that it has *condition number* κ). Having this in mind, the "best" (in terms of subnormalization) we can hope for is a κ-block-encoding of A^{-1}. However, it is unclear how to construct this (*exact*) block-encoding other than inverting A on a classical computer and applying standard methods for block-encoding generic dense matrices, thus negating any improvements in complexity over a purely classical algorithm.

Luckily, the QSVT (quantum singular value transformation, [Gil+19]) and LCU (linear combination of unitaries, [BCK15]) allow us to construct block-encodings of $p(A)$ for polynomials p, with various subnormalizations. In the case of LCU, the subnormalization is $\left\|\mathbf{c}_p\right\|_1$, where $\mathbf{c}_p$ is the vector of coefficients in the Chebyshev expansion $p(x) = \sum_{i=0}^{n} c_{p,i} \mathcal{T}_i(x)$. In the case of QSVT, the subnormalization is simply $\max_{x\in[-1,1]} |p(x)|$. Observe that the LCU subnormalization is at least as large as the one for QSVT, since all polynomials with $\left\|\mathbf{c}_p\right\|_1 \leq 1$ are bounded by 1 on $[-1, 1]$. Therefore, in order to construct an *approximate* block-encoding of A^{-1}, the polynomial p needs to approximate the inverse on $D_\kappa = [-1, -1/\kappa] \cup [1/\kappa, 1]$, and be not much larger than κ on $[-1/\kappa, 1/\kappa]$. A simple example of such a polynomial is $p_n(x) = \frac{1-(1-x^2)^n}{x}$, which approximates $1/x$ on D_κ for $n \geq \kappa^2 \log(\kappa/\varepsilon)$. The Chebyshev coefficients of this polynomial can be interpreted as binomial probabilities, so their 1-norm is bounded, and therefore p_n can be efficiently evaluated using both QSVT and LCU. The insight of Childs, Kothari, and Somma [CKS17] was that the Chebyshev expansion of p_n can be truncated after $\widetilde{O}(\kappa)$ terms yielding an *asymptotically* optimal polynomial approximation to the inverse on D_κ. In Chapter 3 we consider *the* optimal such polynomial,

$$q_n(x) := \frac{1 - \mathcal{T}_n(\frac{1+1/\kappa^2-2x^2}{1-1/\kappa^2})/\mathcal{T}_n(\frac{1+\kappa^2}{1-\kappa^2})}{x}.$$

We show that the Chebyshev coefficient 1-norm of q_n admits an even better bound, implying optimal LCU- and QSVT-based quantum linear system algorithms. Given the simple polynomial (i.e. $O(\sqrt{\text{degree}})$) upper bound on the gap between QSVT and LCU for arbitrary polynomials, we also investigate the efficiency of LCU-based algorithms for approximating some other functions of interest. We show that even discontinuous functions such as the sign function have a nicely-bounded Chebyshev coefficient 1-norm, and we conjecture that Hamiltonian simulation (and in particular $\cos(\kappa x)$ and $\sin(\kappa x)$ for $\kappa > 1$) provides a polynomial separation between LCU and QSVT. We conclude the Chapter with a simple nested LCU-based algorithm for Hamiltonian simulation.

In Chapter 2 we apply the linear systems machinery from Chapter 3 to the more "complicated" problem of second-order cone programming (SOCP), a class of convex optimization problems that lies between linear (LP) and semidefinite (SDP) programs in terms of expressiveness. An efficient algorithm for SOCP would also yield efficient algorithms for many interesting problems, such as

(standard and quadratically-constrained) convex quadratic programming, portfolio optimization, and many others [AG03]. In Chapter 2 we present a quantum algorithm for SOCP, as well as two candidate problems for end-to-end quantum speedups using this algorithm: support-vector machines (SVM) and portfolio optimization.

In SOCP one needs to optimize a linear objective $\mathbf{c}^\top \mathbf{x}$ subject to linear and second-order constraints on the blocks of $\mathbf{x}$. Traditionally, SOCPs are solved using the interior-point method (IPM), an algorithm that starts from a feasible solution, and iteratively improves it by reducing the duality gap. Each improvement is done by solving a linear system (called the *Newton system*), and updating the the iterate using the solution (called the *Newton step*) of this system. One can show [Kar84; NT97; NT98] that in each iteration the duality gap decreases multiplicatively by the same factor, reaching a duality gap of ϵ in $O(\sqrt{n}\log(1/\epsilon))$ iterations.

A quantum interior point method of Kerenidis and Prakash [KP20a] uses a quantum linear system solver instead of classical one in each iteration of the IPM. Several adjustments need to be made for this approach to work out. Firstly, the algorithm requires the input data to be provided in the form of appropriate block-encodings. Secondly (and crucially), the IPM analysis needs to be adapted to the case of inexact Newton steps. In particular, the algorithm works by solving the Newton system using a quantum linear systems solver, and recovering a classical approximation of the Newton step using *tomography*. Proving that this quantum algorithm still converges to (an approximation to) the correct solution is the main technical challenge of this his analysis is based on the quantum interior-point method for SDP [KP20a], but uses additional ideas from the classical SOCP analysis by [MT00].

1.3 Classical algorithms

One of the most important tasks in statistics and machine learning is to sample from high-dimensional and potentially complicated distributions. Markov chains are an efficient means for sampling from such distributions, and there is a wide variety of Markov chain algorithms designed specifically for this purpose. Typically, the main difficulty in analyzing these algorithms is to bound the precise running time or *mixing time* of the Markov chain. While many algorithms have been in very broad (heuristic) usage for several decades, rigorous bounds on their performance are often missing. A key example is the *Hamiltonian Monte Carlo* (HMC) algorithm [Dua+87]. This is an elegant Markov chain algorithm that utilizes Hamiltonian dynamics to efficiently explore the state space, without straying too far away from the high probability region. One of its key features is that it overcomes the slow, diffusive behavior that is inherent to "small step" approaches such as the ball walk and Langevin algorithm. While this is indeed observed in heuristic uses and studies of the HMC algorithm [Nea11], recent efforts are mostly restricted to step sizes much shorter than the heuristic choices [Che+20; CV22]. In this work, we prove seemingly optimal bounds on the HMC algorithm (with leapfrog integrator) for the special case of Gaussian distributions. This is the typical gateway to more complicated distributions such as logconcave or multimodal distributions. Our implementation of HMC exploits long and randomized integration times. This surpasses recent roadblocks on sampling Gaussian distributions using HMC with either short [CV22] or deterministic [LST21] integration times.

Our bounds are stated most easily in the "black box model", where the goal is to sample from a density of the form $e^{-f(\mathbf{x})}$ for $\mathbf{x} \in \mathbb{R}^d$, and we are given query access to both f and its gradient ∇f. The Gaussian case further restricts f to be a quadratic form $f(\mathbf{x}) = \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})$, where $\boldsymbol{\mu}$ and Σ are the (unknown) mean and covariance matrix of the Gaussian, respectively. The *condition number* κ of the Gaussian distribution is simply the condition number of Σ^{-1}. In Chapter 4 we

develop a Metropolis-adjusted HMC algorithm that can compute an ε-approximate sample from the given Gaussian using $\widetilde{O}(\sqrt{\kappa}d^{1/4}\log(1/\varepsilon))$ gradient evaluations.

Both bounds seem in line with expectation [Dua+87; Nea11; Bes+13], and we expect they are tight when using the usual leapfrog integrator for simulating the Hamiltonian dynamics. Our algorithm surpasses the $\widetilde{\Omega}(\kappa\sqrt{d})$ lower bound on the complexity of HMC for Gaussian sampling from [LST21] by using *randomized* integration times. This avoids the well-known periodicity issues associated to a deterministic integration time.

Our work fits within the recent effort of proving non-asymptotic (and often tight) bounds on Markov chain algorithms for constrained distributions such as Gaussian distributions and, more generally, logconcave distributions (where f is assumed to be convex). Most of these efforts have focused on short step dynamics such as the ball walk, the Langevin algorithm, and HMC with short integration times. The use of such "local steps" makes it easier to control the stability and acceptance probability of the algorithm. However, the restriction to short step dynamics is also what slows down these algorithms, and this is what we avoid in our HMC algorithm.

Finally, the restriction to sampling Gaussian and logconcave distributions precisely parallels the restriction to quadratic and convex functions in optimization. Nonetheless, a gap between the (first-order oracle) complexity for logconcave sampling and the $O(\min\{\sqrt{\kappa}, d\})$ complexity for convex optimization is apparently deemed plausible. More specifically, the authors in [LST20] suggest an $\Omega(\kappa)$ lower bound for logconcave sampling. Our work shows that a sublinear κ-dependency is possible at least for the special case of Gaussian distributions, and we see it as evidence that a general $O(\sqrt{\kappa})$ bound for logconcave sampling might be achievable.

Part I

Quantum algorithms

2 | Quantum interior-point methods

Joint work with Iordanis Kerenidis and Anupam Prakash

2.1 Introduction

It is well known that many interesting and relevant optimization problems in the domain of Machine Learning can be expressed in the framework of convex optimization [BV04; Bub15]. The landmark result in this area was the discovery of interior-point methods (IPM) by [Kar84], and their subsequent generalization to all "self-scaled" (i.e. symmetric) cones by [NT97; NT98]. Very recently, [CLS19] have shown that it is possible to solve linear programs (LP) in $\widetilde{O}(n^\omega)$, the time it takes to multiply two matrices (as long as $\omega \geq 2 + 1/6$, which is currently the case). This result has been further extended in [LSZ19] to a slightly more general class of cones, however, their techniques did not yield improved complexities for second-order (SOCP) and semidefinite programming (SDP). An efficient algorithm for SOCP would also yield efficient algorithms for many interesting problems, such as (standard and quadratically-constrained) convex quadratic programming, portfolio optimization, and many others [AG03].

Starting with the landmark results of [Gro96; Sho94], and, more recently, [HHL09], it has been demonstrated that quantum computers offer significant (sometimes even exponential) asymptotic speedups for a number of important problems. More recently, there has been substantial work in the area of convex optimization. Quantum speedups for gradient descent were investigated by [GAW19], whereas [BS17b; Bra+19; vApe+17; AG19] presented quantum algorithms for SDP based on the the multiplicative weights framework of [AHK12]. However, it has been difficult to establish asymptotic speedups for this family of quantum SDP solvers as their running time depends on problem-specific parameters, including a 5th-power dependence on the width of the SDP. Interestingly, the recent result of [BKF22] suggests that such a speedup might be obtained when applying an SDP algorithm of this type to some low-precision instances of quadratic binary optimization.

In an orthogonal approach, [KP20a] proposed a quantum algorithm for LPs and SDPs by quantizing a variant of the classical interior point method and using the state of the art quantum linear algebra tools [CGJ19; Gil+19] – in particular, the matrix multiplication and inversion algorithms whose running time is sub-linear in the input size. However, the complexity of this algorithm depends on the condition number of $O(n^2)$-sized matrices that is difficult to bound theoretically. It therefore remains an open question to find an end-to-end optimization problem for which quantum SDP solvers achieve an asymptotic speedup over state of the art classical algorithms. In this chapter, we propose two candidates for such end-to-end speedups: support-vector machines (SVM) and portfolio optimization.

2.1.1 Our results and techniques

In this section, we provide a high level sketch of our results and the techniques used for the quantum interior point method for SOCPs. We begin by discussing the differences between classical and quantum interior point methods.

A classical interior point method solves an optimization problem over symmetric cones by starting with a feasible solution and iteratively finding solutions with a smaller duality gap while maintaining feasibility. A single iterative step consists of solving a system of linear equations called the Newton linear system and updating the current iterate using the solutions of the Newton system. The analysis of the classical IPM shows that in each iteration, the updated solutions remain feasible and the duality gap is decreased by a factor of $(1 - \alpha/\sqrt{n})$ where n is the dimension of the optimization problem and $\alpha > 0$ is a constant. The algorithm therefore converges to a feasible solution with duality gap ϵ in $O(\sqrt{n}\log(1/\epsilon))$ iterations.

A quantum interior point method [KP20a] uses a quantum linear system solver instead of classical one in each iteration of the IPM. However, there is an important difference between classical and quantum linear algebra procedures for solving the linear system $A\mathbf{x} = \mathbf{b}$. Unlike classical linear system solvers which return an exact description of $\mathbf{x}$, quantum tomography procedures can return an ϵ-accurate solution $\overline{\mathbf{x}}$ such that $\|\overline{\mathbf{x}} - \mathbf{x}\| \leq \epsilon\|\mathbf{x}\|$ with $O(n/\epsilon^2)$ runs of the quantum linear system solver. Additionally, these linear system solvers require A and $\mathbf{b}$ to be given as block-encodings [CGJ19], so this input model is used by our algorithm as well. One of the main technical challenges in developing a quantum interior point method (QIPM) is to establish convergence of the classical IPM which uses ϵ-approximate solutions of the Newton linear system (in the ℓ_2 norm) instead of the exact solutions in the classical IPM.

The quantum interior point method for second-order cone programs requires additional ideas going beyond those used for the quantum interior point methods for SDP [KP20a]. Second order cone programs are optimization problems over the product of second-order or Lorentz cones (see section 2.2.1 for definitions), interior point methods for SOCP can be described using the Euclidean Jordan algebra framework [MT00]. The Euclidean Jordan algebra framework provides analogs of concepts like eigenvalues, spectral and Frobenius norms for matrices and positive semidefinite constraints for the case of SOCPs. Using these conceptual ideas from the Euclidean Jordan algebra framework [MT00] and the analysis of the approximate SDP interior point method [KP20a] we provide an approximate IPM for SOCP that converges in $O(\sqrt{n}\log(1/\epsilon))$ iterations. Approximate IPMs for SOCP have not been previously investigated in the classical or the quantum optimization literature, this analysis is one of the main technical contributions of this chapter.

From an algorithmic perspective, SOCPs are much closer to LPs (Linear Programs) than to SDPs, since for cones of dimension n, the Newton linear systems arising in LP and SOCP IPMs are of size $O(n)$, whereas in the SDP case they are of size $O(n^2)$. Namely, a second-order conic constraint of dimension n can be expressed as a single PSD constraint on a (sparse) $n \times n$ matrix [AG03] – this allows us to embed an SOCP with n variables and m constraints in an SDP with an $n \times n$ matrix and m constraints. The cost of solving that SDP would have a worse dependence on the error [AG19] or the input size [KP20a]. On the other hand, the quantum representations (block encodings) of the Newton linear systems for SOCP are also much simpler to construct than those for SDP. The smaller size of the SOCP linear system also makes it feasible to empirically estimate the condition number for these linear systems in a reasonable amount of time allowing us to carry out extensive numerical experiments to validate the running time of the quantum algorithm.

The theoretical analysis of the quantum algorithm for SOCP shows that its worst-case running time is

$$\widetilde{O}\left(n\sqrt{r}\frac{\zeta\kappa}{\delta^2}\log\left(\frac{1}{\epsilon}\right)\right), \tag{2.1}$$

where r is the rank and n the dimension of the SOCP, δ bounds the distance of intermediate solutions from the cone boundary, ζ is a parameter bounded by $\sqrt{n}$, κ is an upper bound on the condition number of matrices arising in the interior-point method for SOCPs, and ϵ is the target duality gap. The running time of the algorithm depends on problem dependent parameters like κ

and δ that are difficult to bound in terms of the problem dimension n – this is also the case with previous quantum SDP solvers [KP20a; AG19] and makes it important to validate the quantum optimization speedups empirically. Interestingly, since we require a classical solution of the Newton system, the linear system solver could also be replaced by a classical iterative solver [Saa03] which would yield a complexity of $O(n^2\sqrt{r}\kappa \log(n/\epsilon))$.

Let us make a remark about the complexity: as it is the case with all approximation algorithms, (2.1) depends on the inverse of the target duality gap ϵ, thus the running time of our algorithm grows to infinity as ϵ approaches zero, as in the case of classical IPM. Our running time also depends on κ, which in turn is empirically observed to grow inversely with the duality gap (in particular as $O(1/\epsilon)$) which again makes the running time go to infinity as ϵ approaches zero. The quantum IPM is a low precision method, unlike the classical IPM, and it can offer speedups for settings where the desired precision ϵ is moderate or low. Thus although at first glance it seems that the ϵ-dependence in (2.1) is logarithmic, experimental evidence suggests that the factor $\frac{\zeta\kappa}{\delta^2}$ depends polynomially on $1/\epsilon$.

2.1.2 Applications to SVM

Support Vector Machines (SVM) are an important application in machine learning, where even a modest value of $\epsilon = 0.1$ yields an almost optimal classifier. Since the SVM training problem can be reduced to SOCP, the quantum IPM for SOCP can be used to obtain an efficient quantum SVM algorithm. We perform extensive numerical experiments to evaluate our algorithm on random SVM instances and compare it against state of the art classical SOCP and SVM solvers.

The numerical experiments on random SVM instances indicate that the running time of the quantum algorithm scales as roughly $O(n^{2.591})$, where all the parameters in the running time are taken into account and the exponent is estimated using a least squares fit. We also benchmarked the exponent for classical SVM algorithms on the same instances and for a comparable accuracy, the scaling exponent was found to be 3.31 for general SOCP solvers and 3.11 for state-of-the-art SVM solvers. We note that this does not prove a worst-case asymptotic speedup, but the experiments on unstructured SVM instances provide strong evidence for a significant polynomial speedup of the quantum SVM algorithm over state-of-the-art classical SVM algorithms. We can therefore view SVMs as a candidate problem for which quantum optimization algorithms can achieve a polynomial speedup over state of the art classical algorithms for an end-to-end application.

Our IPM for SOCPs yields the first specialized quantum algorithm for training support-vector machines (SVM). While several quantum SVM algorithms have been proposed, they are unlikely to offer general speedups for the most widely used formulation of SVM – the soft-margin (ℓ_1-)SVM (defined in eq. (2.10)). On one hand, papers such as [SA22] formulate the SVM as a SDP, and solve that using existing quantum SDP solvers such as [BS17b; Bra+19; vApe+17; AG19] – with the conclusion being that a speedup is observed only for very specific sparse instances. On the other hand, [RML14] solves an easier related problem – the least-squares SVM (ℓ_2-SVM or LS-SVM, see eq. (2.11) for its formulation), thus losing the desirable sparsity properties of ℓ_1-SVM [Suy+02]. It turns out that applying our algorithm to this problem also yields the same complexity as in [RML14] for the ℓ_2-SVM. Very recently, a quantum algorithm for yet another variant of SVM (SVM-perf, [Joa06]) has been presented in [AH20].

2.1.3 Applications to portfolio optimization

Mathematical finance is an application area where quantum computers could potentially offer groundbreaking speedups. This is a very recent research area for quantum algorithms and is important in terms of applications as even modest speedups for computational financial problems

can have enormous real world impact. Of course, translating these theoretical advantages of quantum algorithms into real world applications necessitates both much more advanced hardware, which may take some more years to come, but also a close collaboration between the communities of quantum algorithms and of mathematical Finance in order to really understand where and how such quantum algorithms can become a new powerful tool to be used within the general framework of mathematical finance.

Rebentrost and Lloyd [RL18] proposed a quantum algorithm for the unconstrained portfolio optimization problem. Their algorithm uses quantum linear system solvers to obtain speedups for portfolio optimization problems that can be reduced to unconstrained quadratic programs, which in turn are reducible to a single linear system. The main limitation of their algorithm is that it can not incorporate positivity or budget constraints, thus restricting its applicability to real world problems that can have complex budget constraints. The reason for this limitation is algorithmic, the constrained portfolio optimization problem is known to be equivalent to quadratic programming (QP), a class of optimization problems that is more general than linear programming (LP).

In this chapter, we consider portfolio optimization with an arbitrary number of positivity as well as budget constraints. We express the given portfolio optimization problem as a SOCP and solve it using our quantum IPM. We show numerical evidence of polynomial speedups over the classical algorithms.

2.2 Preliminaries

2.2.1 Second-order cone programming

For the sake of completeness, in this section we present the most important results about classical SOCP IPMs, from [MT00; AG03]. We start by defining SOCP as the optimization problem over the product of second-order (or Lorentz) cones $\mathcal{L} = \mathcal{L}^{n_1} \times \cdots \times \mathcal{L}^{n_r}$, where $\mathcal{L}_k \subseteq \mathbb{R}^k$ is defined as $\mathcal{L}^k = \left\{ \mathbf{x} = (x_0; \tilde{\mathbf{x}}) \in \mathbb{R}^k \;\middle|\; \|\tilde{\mathbf{x}}\| \leq x_0 \right\}$. In this chapter we consider the problem (2.2) and its dual (2.3):

$$\begin{array}{lll} \min & \mathbf{c}^\top \mathbf{x} & \\ \text{s.t.} & A\mathbf{x} = \mathbf{b} & (2.2) \\ & \mathbf{x} \in \mathcal{L}, & \end{array} \qquad \begin{array}{lll} \max & \mathbf{b}^\top \mathbf{y} & \\ \text{s.t.} & A^\top \mathbf{y} + \mathbf{s} = \mathbf{c} & (2.3) \\ & \mathbf{s} \in \mathcal{L},\ \mathbf{y} \in \mathbb{R}^m. & \end{array}$$

We call $n := \sum_{i=1}^{r} n_i$ the *size* of the SOCP (2.2), and r is its *rank*.

A solution $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ satisfying the constraints of both (2.2) and (2.3) is *feasible*, and if in addition it satisfies $\mathbf{x} \in \operatorname{int} \mathcal{L}$ and $\mathbf{s} \in \operatorname{int} \mathcal{L}$, it is *strictly feasible*. If at least one constraint of (2.2) or (2.3), is violated, the solution is *infeasible*. The duality gap of a feasible solution $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ is defined as $\mu := \mu(\mathbf{x}, \mathbf{s}) := \frac{1}{r}\mathbf{x}^\top \mathbf{s}$. As opposed to LP, and similarly to SDP, strict feasibility is required for *strong duality* to hold [AG03]. From now on, we assume that our SOCP has a strictly feasible primal-dual solution (this assumption is valid, since the homogeneous self-dual embedding technique from [YTM94] allows us to embed (2.2) and (2.3) in a slightly larger SOCP where this condition is satisfied).

2.2.2 Euclidean Jordan algebras

The cone $\mathcal{L}^n$ has an algebraic structure similar to that of symmetric matrices under the matrix product. Here, we consider the Jordan product of $(x_0, \tilde{\mathbf{x}}) \in \mathbb{R}^n$ and $(y_0, \tilde{\mathbf{y}}) \in \mathbb{R}^n$, defined as

$$\mathbf{x} \circ \mathbf{y} := \begin{bmatrix} \mathbf{x}^\top \mathbf{y} \\ x_0 \tilde{\mathbf{y}} + y_0 \tilde{\mathbf{x}} \end{bmatrix}, \text{ and its identity element } \mathbf{e} := \begin{bmatrix} 1 \\ 0^{n-1} \end{bmatrix}.$$

This product is closely related to the (linear) matrix representation $\operatorname{Arw}(\mathbf{x}) := \begin{bmatrix} x_0 & \tilde{\mathbf{x}}^\top \\ \tilde{\mathbf{x}} & x_0 I_{n-1} \end{bmatrix}$, which in turn satisfies the following equality:

$$\mathbf{x} \circ \mathbf{y} = \operatorname{Arw}(\mathbf{x})\mathbf{y} = \operatorname{Arw}(\mathbf{x}) \operatorname{Arw}(\mathbf{y})\mathbf{e}.$$

The key observation is that this product induces a spectral decomposition of any vector $\mathbf{x}$, that has similar properties as its matrix relative. Namely, for any vector $\mathbf{x}$ we define

$$\begin{aligned} \lambda_1(\mathbf{x}) &:= x_0 + \|\tilde{\mathbf{x}}\|, \ \mathbf{c}_1(\mathbf{x}) := \frac{1}{2}\begin{bmatrix} 1 \\ \frac{\tilde{\mathbf{x}}}{\|\tilde{\mathbf{x}}\|} \end{bmatrix}, \\ \lambda_2(\mathbf{x}) &:= x_0 - \|\tilde{\mathbf{x}}\|, \ \mathbf{c}_2(\mathbf{x}) := \frac{1}{2}\begin{bmatrix} 1 \\ \frac{-\tilde{\mathbf{x}}}{\|\tilde{\mathbf{x}}\|} \end{bmatrix}. \end{aligned} \tag{2.4}$$

We use the shorthands $\lambda_1 := \lambda_1(\mathbf{x})$, $\lambda_2 := \lambda_2(\mathbf{x})$, $\mathbf{c}_1 := \mathbf{c}_1(\mathbf{x})$ and $\mathbf{c}_2 := \mathbf{c}_2(\mathbf{x})$ whenever $\mathbf{x}$ is clear from the context, so we observe that $\mathbf{x} = \lambda_1\mathbf{c}_1 + \lambda_2\mathbf{c}_2$. The set of eigenvectors $\{\mathbf{c}_1, \mathbf{c}_2\}$ is called the *Jordan frame* of $\mathbf{x}$, and satisfies several properties:

Proposition 2.1 (Properties of Jordan frames)**.** *Let $\mathbf{x} \in \mathbb{R}^n$ and let $\{\mathbf{c}_1, \mathbf{c}_2\}$ be its Jordan frame. Then, the following holds:*

1. $\mathbf{c}_1 \circ \mathbf{c}_2 = 0$ *(the eigenvectors are "orthogonal")*
2. $\mathbf{c}_1 \circ \mathbf{c}_1 = \mathbf{c}_1$ *and* $\mathbf{c}_2 \circ \mathbf{c}_2 = \mathbf{c}_2$
3. $\mathbf{c}_1$, $\mathbf{c}_2$ *are of the form* $\left(\frac{1}{2}; \pm\tilde{\mathbf{c}}\right)$ *with* $\|\tilde{\mathbf{c}}\| = \frac{1}{2}$

On the other hand, just like a given matrix is positive (semi)definite if and only if all of its eigenvalues are positive (nonnegative), a similar result holds for $\mathcal{L}^n$ and $\operatorname{int}\mathcal{L}^n$ (the Lorentz cone and its interior):

Proposition 2.2. *Let $\mathbf{x} \in \mathbb{R}^n$ have eigenvalues λ_1, λ_2. Then, the following holds:*

1. $\mathbf{x} \in \mathcal{L}^n$ *if and only if* $\lambda_1 \geq 0$ *and* $\lambda_2 \geq 0$.
2. $\mathbf{x} \in \operatorname{int}\mathcal{L}^n$ *if and only if* $\lambda_1 > 0$ *and* $\lambda_2 > 0$.

Now, using this decomposition, we can define arbitrary real powers $\mathbf{x}^p$ for $p \in \mathbb{R}$ as $\mathbf{x}^p := \lambda_1^p\mathbf{c}_1 + \lambda_2^p\mathbf{c}_2$, and in particular the "inverse" and the "square root"

$$\begin{aligned} \mathbf{x}^{-1} &= \frac{1}{\lambda_1}\mathbf{c}_1 + \frac{1}{\lambda_2}\mathbf{c}_2, \text{ if } \lambda_1\lambda_2 \neq 0, \\ \mathbf{x}^{1/2} &= \sqrt{\lambda_1}\mathbf{c}_1 + \sqrt{\lambda_2}\mathbf{c}_2, \text{ if } \mathbf{x} \in \mathcal{L}^n. \end{aligned}$$

Moreover, we can also define some operator norms, namely the Frobenius and the spectral one:

$$\begin{aligned} \|\mathbf{x}\|_F &= \sqrt{\lambda_1^2 + \lambda_2^2} = \sqrt{2}\|\mathbf{x}\|, \\ \|\mathbf{x}\|_2 &= \max\{|\lambda_1|, |\lambda_2|\} = |x_0| + \|\tilde{\mathbf{x}}\|. \end{aligned}$$

Finally, we define an analogue to the operation $Y \mapsto XYX$. It turns out that for this we need another matrix representation (*quadratic representation*) $Q_{\mathbf{x}}$, defined as

$$Q_{\mathbf{x}} := 2\operatorname{Arw}^2(\mathbf{x}) - \operatorname{Arw}(\mathbf{x}^2) = \begin{bmatrix} \|\mathbf{x}\|^2 & 2x_0\tilde{\mathbf{x}}^\top \\ 2x_0\tilde{\mathbf{x}} & \lambda_1\lambda_2 I_n + 2\tilde{\mathbf{x}}\tilde{\mathbf{x}}^\top \end{bmatrix}. \tag{2.5}$$

Now, the matrix-vector product $Q_{\mathbf{x}}\mathbf{y}$ will behave as the quantity XYX. To simplify the notation, we also define the matrix $T_{\mathbf{x}} := Q_{\mathbf{x}^{1/2}}$.

The definitions that we introduced so far are suitable for dealing with a single constraint $\mathbf{x} \in \mathcal{L}^n$. For dealing with multiple constraints $\mathbf{x}_1 \in \mathcal{L}^{n_1}, \ldots, \mathbf{x}_r \in \mathcal{L}^{n_r}$, we need to deal with block-vectors $\mathbf{x} = (\mathbf{x}_1; \mathbf{x}_2; \ldots; \mathbf{x}_r)$ and $\mathbf{y} = (\mathbf{y}_1; \mathbf{y}_2; \ldots; \mathbf{y}_r)$. We call the number of blocks r the *rank* of the vector (thus, up to now, we were only considering rank-1 vectors). Now, we extend all our definitions to rank-r vectors.

1. $\mathbf{x} \circ \mathbf{y} := (\mathbf{x}_1 \circ \mathbf{y}_1; \ldots; \mathbf{x}_r \circ \mathbf{y}_r)$

2. The matrix representations $\operatorname{Arw}(\mathbf{x})$ and $Q_{\mathbf{x}}$ are the block-diagonal matrices containing the representations of the blocks:

$$\operatorname{Arw}(\mathbf{x}) := \operatorname{Arw}(\mathbf{x}_1) \oplus \cdots \oplus \operatorname{Arw}(\mathbf{x}_r) \text{ and } Q_{\mathbf{x}} := Q_{\mathbf{x}_1} \oplus \cdots \oplus Q_{\mathbf{x}_r}$$

3. $\mathbf{x}$ has $2r$ eigenvalues (with multiplicities) – the union of the eigenvalues of the blocks $\mathbf{x}_i$. The eigenvectors of $\mathbf{x}$ corresponding to block i contain the eigenvectors of $\mathbf{x}_i$ as block i, and are zero everywhere else.

4. The identity element is $\mathbf{e} = (\mathbf{e}_1; \ldots; \mathbf{e}_r)$, where $\mathbf{e}_i$'s are the identity elements for the corresponding blocks.

Thus, all things defined using eigenvalues can also be defined for rank-r vectors:

1. The norms are extended as $\|\mathbf{x}\|_F^2 := \sum_{i=1}^r \|\mathbf{x}_i\|_F^2$ and $\|\mathbf{x}\|_2 := \max_i \|\mathbf{x}_i\|_2$, and

2. Powers are computed blockwise as $\mathbf{x}^p := (\mathbf{x}_1^p; \ldots; \mathbf{x}_r^p)$ whenever the corresponding blocks are defined.

Some further matrix-algebra inspired properties of block vectors are stated in the following two claims:

Claim 2.3 (Algebraic properties)**.** *Let $\mathbf{x}, \mathbf{y}$ be two arbitrary block-vectors. Then, the following holds:*

1. *The spectral norm is subadditive: $\|\mathbf{x} + \mathbf{y}\|_2 \leq \|\mathbf{x}\|_2 + \|\mathbf{y}\|_2$.*

2. *The spectral norm is less than the Frobenius norm: $\|\mathbf{x}\|_2 \leq \|\mathbf{x}\|_F$.*

3. *If A is a matrix with minimum and maximum singular values σ_{min} and σ_{max} respectively, then the norm $\|A\mathbf{x}\|$ is bounded as $\sigma_{min}\|\mathbf{x}\| \leq \|A\mathbf{x}\| \leq \sigma_{max}\|\mathbf{x}\|$.*

4. *The minimum eigenvalue of $\mathbf{x} + \mathbf{y}$ is bounded as $\lambda_{min}(\mathbf{x} + \mathbf{y}) \geq \lambda_{min}(\mathbf{x}) - \|\mathbf{y}\|_2$.*

5. *The following submultiplicativity property holds: $\|\mathbf{x} \circ \mathbf{y}\|_F \leq \|\mathbf{x}\|_2 \cdot \|\mathbf{y}\|_F$.*

In general, the proofs of these statements are analogous to the matrix case, with a few notable differences: First, the vector spectral norm $\|\cdot\|_2$ is not actually a norm, since there exist nonzero vectors outside $\mathcal{L}$ which have zero norm. It is, however, still bounded by the Frobenius norm (just like in the matrix case), which is in fact a proper norm. Secondly, the minimum eigenvalue bound also holds for matrix spectral norms, with the exact same statement. Finally, the last property is reminiscent of the matrix submultiplicativity property $\|A \cdot B\|_F \leq \|A\|_2 \|B\|_F$.

We finish with several well-known properties of the quadratic representation $Q_{\mathbf{x}}$ and $T_{\mathbf{x}}$.

Proposition 2.4 (Properties of $Q_{\mathbf{x}}$, from [AG03])**.** *Let $\mathbf{x} \in \operatorname{int} \mathcal{L}$. Then, the following holds:*

1. $Q_{\mathbf{x}}\mathbf{e} = \mathbf{x}^2$, *and thus* $T_{\mathbf{x}}\mathbf{e} = \mathbf{x}$.
2. $Q_{\mathbf{x}^{-1}} = Q_{\mathbf{x}}^{-1}$, *and more generally* $Q_{\mathbf{x}^p} = Q_{\mathbf{x}}^p$ *for all* $p \in \mathbb{R}$.
3. $\|Q_{\mathbf{x}}\|_2 = \|\mathbf{x}\|_2^2$, *and thus* $\|T_{\mathbf{x}}\|_2 = \|\mathbf{x}\|_2$.
4. $Q_{\mathbf{x}}$ *preserves* $\mathcal{L}$, *i.e.* $Q_{\mathbf{x}}(\mathcal{L}) = \mathcal{L}$ *and* $Q_{\mathbf{x}}(\operatorname{int} \mathcal{L}) = \operatorname{int} \mathcal{L}$.

2.2.3 Interior-point methods

Our algorithm follows the general IPM structure, i.e. it uses Newton's method to solve a sequence of increasingly strict relaxations of the Karush-Kuhn-Tucker (KKT) optimality conditions:

$$\begin{aligned} A\mathbf{x} = \mathbf{b},\ A^\top \mathbf{y} + \mathbf{s} = \mathbf{c} \\ \mathbf{x} \circ \mathbf{s} = \nu \mathbf{e},\ \mathbf{x} \in \mathcal{L}, \mathbf{s} \in \mathcal{L}, \end{aligned} \tag{2.6}$$

where the parameter $\nu > 0$ is decreased by a factor $\sigma < 1$ in each iteration. Since $\mathbf{x} \circ \mathbf{s} = \nu \mathbf{e}$ implies that the duality gap is $\mu = \nu$, by letting $\nu \to 0$ the IPM converges towards the optimal solution. The curve traced by (feasible) solutions $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ of (2.6) for $\nu > 0$ is called the *central path*, and we note that all points on it are strictly feasible.

More precisely, in each iteration we need to find $\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s}$ such that $\mathbf{x}_{\text{next}} := \mathbf{x} + \Delta\mathbf{x}$, $\mathbf{y}_{\text{next}} := \mathbf{y} + \Delta\mathbf{y}$ and $\mathbf{s}_{\text{next}} := \mathbf{s} + \Delta\mathbf{s}$ satisfy (2.6) for $\nu = \sigma\mu$. After linearizing the product $\mathbf{x}_{\text{next}} \circ \mathbf{s}_{\text{next}}$, we obtain the following linear system – the *Newton system*:

$$\begin{bmatrix} A & 0 & 0 \\ 0 & A^\top & I \\ \operatorname{Arw}(\mathbf{s}) & 0 & \operatorname{Arw}(\mathbf{x}) \end{bmatrix} \begin{bmatrix} \Delta\mathbf{x} \\ \Delta\mathbf{y} \\ \Delta\mathbf{s} \end{bmatrix} = \begin{bmatrix} \mathbf{b} - A\mathbf{x} \\ \mathbf{c} - \mathbf{s} - A^\top \mathbf{y} \\ \sigma\mu\mathbf{e} - \mathbf{x} \circ \mathbf{s} \end{bmatrix}. \tag{2.7}$$

As a final remark, it is not guaranteed that $(\mathbf{x}_{\text{next}}, \mathbf{y}_{\text{next}}, \mathbf{s}_{\text{next}})$ is on the central path (2.6), or even that it is still strictly feasible. Luckily, it can be shown that as long as $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ starts out in a neighborhood $\mathcal{N}$ of the central path, $(\mathbf{x}_{\text{next}}, \mathbf{y}_{\text{next}}, \mathbf{s}_{\text{next}})$ will remain both strictly feasible and in $\mathcal{N}$. It can be shown that this algorithm halves the duality gap every $O(\sqrt{r})$ iterations, so indeed, after $O(\sqrt{r} \log(\mu_0/\epsilon))$ it will converge to a (feasible) solution with duality gap at most ϵ (given that the initial duality gap was μ_0).

2.2.4 Quantum linear algebra

As it was touched upon earlier, the main speedup in our algorithms comes from the fact that we use the quantum linear algebra algorithms from [CGJ19; Gil+19] whose complexity is sublinear in the dimension. Of course, we need to change our computational model for this sentence to make

any sense: namely, we encode n-dimensional unit vectors as quantum states of a $\lceil \log_2(n) \rceil$-qubit system. In other words, for a vector $\mathbf{z} \in \mathbb{R}^{2^k}$ with $\|\mathbf{z}\| = 1$, we use the notation $|\mathbf{z}\rangle$ to refer to the k-qubit state $|\mathbf{z}\rangle := \sum_{i=0}^{2^k-1} z_i |i\rangle$, where $|i\rangle$ are the standard basis vectors of $\mathbb{C}^{2^k}$, the state space of a k-qubit system [NC12].

Given a quantum state $|\mathbf{z}\rangle$, we have very limited ways of interacting with it: we can either apply a unitary transformation $U : \mathbb{C}^{2^k} \to \mathbb{C}^{2^k}$, or we can *measure* it, which means that we discard the state and obtain a single random integer $0 \leq i \leq 2^k - 1$, with the probability of measuring i being z_i^2. In particular this means that we can neither observe the *amplitudes* z_i directly, nor can we create a copy of $|\mathbf{z}\rangle$ for an arbitrary $|\mathbf{z}\rangle$. In addition to this, it is *a priori* not clear how (and whether it is even possible) to implement the state $|\mathbf{z}\rangle$ or an arbitrary unitary U using the gates of a quantum computer. Luckily, there exists a quantum-classical framework using *QRAM* data structures described in [KP17] that provides a positive answer to both of these questions.

The QRAM can be thought of as the quantum analogue to RAM, i.e. an array $[\mathbf{b}^{(1)}, \ldots, \mathbf{b}^{(m)}]$ of w-bit bitstrings, whose elements we can access in poly-logarithmic time given their address (position in the array). More precisely, QRAM is just an efficient implementation of the unitary transformation

$$|i\rangle |0\rangle^{\otimes w} \mapsto |i\rangle |b_1^{(i)} \ldots b_w^{(i)}\rangle, \text{ for } i \in [m].$$

The usefulness of QRAM data structures becomes clear when we consider the *block encoding* framework:

Definition 2.5. Let $A \in \mathbb{R}^{n \times n}$ be a symmetric matrix. Then, the ℓ-qubit unitary matrix $U \in \mathbb{C}^{2^\ell \times 2^\ell}$ is a (ζ, ℓ) block encoding of A if $U = \begin{bmatrix} A/\zeta & \cdot \\ \cdot & \cdot \end{bmatrix}$. For an arbitrary matrix $B \in \mathbb{R}^{n \times m}$, a block encoding of B is any block encoding of its symmetrized version $\mathrm{sym}(B) := \begin{bmatrix} 0 & B \\ B^\top & 0 \end{bmatrix}$.

We want U to be implemented efficiently, i.e. using an ℓ-qubit quantum circuit of depth (poly-) logarithmic in n. Such a circuit would allow us to efficiently create states $|A_i\rangle$ corresponding to rows (or columns) of A. Moreover, we need to be able to construct such a data structure efficiently from the classical description of A. It turns out that we are able to fulfill both of these requirements using a data structure built on top of QRAM.

Theorem 2.6 (Block encodings using QRAM [KP17; KP20b])**.** *There exist QRAM data structures for storing vectors $\mathbf{v}_i \in \mathbb{R}^n$, $i \in [m]$ and matrices $A \in \mathbb{R}^{n \times n}$ such that with access to these data structures one can do the following:*

1. *Given $i \in [m]$, prepare the state $|\mathbf{v}_i\rangle$ in time $\widetilde{O}(1)$. In other words, the unitary $|i\rangle |0\rangle \mapsto |i\rangle |\mathbf{v}_i\rangle$ can be implemented efficiently.*
2. *A $(\zeta(A), 2\log n)$ unitary block encoding for A with $\zeta(A) = \|A\|_2^{-1} \min(\|A\|_F, s_1(A))$, where $s_1(A) = \max_i \sum_j |A_{i,j}|$ can be implemented in time $\widetilde{O}(\log n)$. Moreover, this block encoding can be constructed in a single pass over the matrix A, and it can be updated in $O(\log^2 n)$ time per entry.*

From now on, we will also refer to storing vectors and matrices in QRAM, meaning that we use the data structure from Theorem 2.6. Note that this is the same quantum oracle model that has been used to solve SDPs in [KP20a] and [AG19].

Once we have these block encodings, we may use them to perform linear algebra. In particular, we want to construct the quantum states $|A\mathbf{b}\rangle$ and $|A^{-1}\mathbf{b}\rangle$, corresponding to the matrix-vector product $A\mathbf{b}$ and the solution of the linear system $A\mathbf{x} = \mathbf{b}$:

Theorem 2.7 (Quantum linear algebra with block encodings [CGJ19; Gil+19])**.** *Let $A \in \mathbb{R}^{n\times n}$ be a matrix with non-zero eigenvalues in the interval $[-1, -1/\kappa] \cup [1/\kappa, 1]$, and let $\epsilon > 0$. Given an implementation of an $(\zeta, O(\log n))$ block encoding for A in time T_U and a procedure for preparing state $|b\rangle$ in time T_b,*

1. *A state ϵ-close to $|A^{-1}b\rangle$ can be generated in time $O((T_U\kappa\zeta + T_b\kappa)\,\mathrm{polylog}(\kappa\zeta/\epsilon))$.*
2. *A state ϵ-close to $|Ab\rangle$ can be generated in time $O((T_U\kappa\zeta + T_b\kappa)\,\mathrm{polylog}(\kappa\zeta/\epsilon))$.*
3. *For $\mathcal{A} \in \{A, A^{-1}\}$, an estimate Λ such that $\Lambda \in (1 \pm \epsilon)\|\mathcal{A}b\|$ can be generated in time $O((T_U + T_b)\frac{\kappa\zeta}{\epsilon}\,\mathrm{polylog}(\kappa\zeta/\epsilon))$.*

Finally, in order to recover classical information from the outputs of a linear system solver, we require an efficient procedure for *quantum state tomography*. The tomography procedure is linear in the dimension of the quantum state.

Theorem 2.8 (Efficient vector state tomography, [KP20a])**.** *There exists an algorithm that given a procedure for constructing $|\mathbf{x}\rangle$ (i.e. a unitary mapping $U : |0\rangle \mapsto |\mathbf{x}\rangle$ and its controlled version in time T_U) and precision $\delta > 0$ produces an estimate $\overline{\mathbf{x}} \in \mathbb{R}^n$ with $\|\overline{\mathbf{x}}\| = 1$ such that $\|\mathbf{x} - \overline{\mathbf{x}}\| \leq \sqrt{7}\delta$ with probability at least $(1 - 1/n^{0.83})$. The algorithm runs in time $O\left(T_U \frac{n \log n}{\delta^2}\right)$.*

Of course, repeating this algorithm $\widetilde{O}(1)$ times allows us to increase the success probability to at least $1 - 1/\,\mathrm{poly}(n)$. Putting Theorems 2.6, 2.7 and 2.8 together, assuming that A and $\mathbf{b}$ are already in QRAM, we obtain that the complexity of a completely self-contained algorithm for solving the system $A\mathbf{x} = \mathbf{b}$ with error δ is $\widetilde{O}\left(n \cdot \frac{\kappa\zeta}{\delta^2}\right)$. For well-conditioned matrices, this presents a significant improvement over $O(n^\omega)$ (or, in practice, $O(n^3)$) needed for solving linear systems classically, especially when n is large and the desired precision is not too high. This can be compared with classical iterative linear algebra algorithms [Saa03], whose per-iteration cost is equal to the number of non-zero elements of A, as well as the new quantum-inspired solvers [GLT18] that have high-degree polynomial dependence on the rank, error, and the condition number.

2.3 A quantum interior-point method

Having introduced the classical IPM for SOCP, we can finally introduce our quantum IPM (Algorithm 1). The main idea is to use quantum linear algebra as much as possible (including solving the Newton system – the most expensive part of each iteration), and falling back to classical computation when needed. Since quantum linear algebra introduces inexactness, we need to deal with it in the analysis. The inspiration for the "quantum part" of the analysis is [KP20a], whereas the "classical part" is based on [MT00]. Nevertheless, the SOCP analysis is unique in many aspects and a number of hurdles had to be overcome to make the analysis go through. In the rest of the section, we give a brief introduction of the most important quantum building blocks we use, as well as present a sketch of the analysis.

First, we note that the algorithms from the previous section allow us to "forget" that Algorithm 1 is quantum, and treat it as a small modification of the classical IPM, where the system (2.7) is solved up to an ℓ_2-error δ. Since Algorithm 1 is iterative, the main part of the analysis is proving that a single iteration preserves closeness to the central path, strict feasibility, and improves the

Algorithme 1 : A quantum IPM for SOCP

1 **Require:** Matrix A and vectors $\mathbf{b}, \mathbf{c}$ in QRAM, precision parameter ϵ

1. Find feasible initial point $(\mathbf{x}, \mathbf{y}, \mathbf{s}, \mu)$ and store it in QRAM.
2. Repeat the following steps for $O(\sqrt{r}\log(\mu_0/\epsilon))$ iterations:
   a) Compute the vector $\sigma\mu\mathbf{e} - \mathbf{x} \circ \mathbf{s}$ classically and store it in QRAM.
   b) Prepare and update the block encodings of the LHS and the RHS of the Newton system (2.7)
   c) Solve the Newton system to obtain $|(\Delta\mathbf{x}; \Delta\mathbf{y}; \Delta\mathbf{s})\rangle$, and obtain a classical approximate solution $\overline{(\Delta\mathbf{x}; \Delta\mathbf{y}; \Delta\mathbf{s})}$ using tomography.
   d) Update $\mathbf{x} \leftarrow \mathbf{x} + \overline{\Delta\mathbf{x}}$, $\mathbf{s} \leftarrow \mathbf{s} + \overline{\Delta\mathbf{s}}$ and store in QRAM.
   e) Update $\mu \leftarrow \frac{1}{r}\mathbf{x}^\top\mathbf{s}$.
3. Output $(\mathbf{x}, \mathbf{y}, \mathbf{s})$.

duality gap. In the remainder of this section, we state our main results informally, while the exact statements and proofs of all claims are presented later, as Theorems 2.10, 2.18 and 2.19, respectively.

Theorem (Per-iteration correctness, informal)**.** *Let $(\mathbf{x}, \mathbf{y}, \mathbf{s})$ be a strictly feasible primal-dual solution that is close to the central path, with duality gap μ, and at distance at least δ from the boundary of $\mathcal{L}$. Then, the Newton system (2.7) has a unique solution $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$. There exist positive constants ξ, α such that the following holds: If we let $\overline{\Delta\mathbf{x}}, \overline{\Delta\mathbf{s}}$ be approximate solutions of (2.7) that satisfy*

$$\left\|\Delta\mathbf{x} - \overline{\Delta\mathbf{x}}\right\|_F \leq \xi\delta \text{ and } \left\|\Delta\mathbf{s} - \overline{\Delta\mathbf{s}}\right\|_F \leq \xi\delta,$$

and let $\mathbf{x}_{next} := \mathbf{x} + \overline{\Delta\mathbf{x}}$ and $\mathbf{s}_{next} := \mathbf{s} + \overline{\Delta\mathbf{s}}$ be the updated solution, then:

1. *The updated solution is strictly feasible, i.e. $\mathbf{x}_{next} \in \operatorname{int}\mathcal{L}$ and $\mathbf{s}_{next} \in \operatorname{int}\mathcal{L}$.*
2. *The updated solution is close to the central path, and the new duality gap is less than $(1 - \alpha/\sqrt{r})\mu$.*

The proof of this theorem consists of 3 main parts:

1. Rescaling $\mathbf{x}$ and $\mathbf{s}$ so that they commute in the Jordan-algebraic sense [AG03]. This part can be reused from the classical analysis [MT00].
2. Bounding the norms of $\overline{\Delta\mathbf{x}}$ and $\overline{\Delta\mathbf{s}}$, and proving that $\mathbf{x} + \overline{\Delta\mathbf{x}}$ and $\mathbf{s} + \overline{\Delta\mathbf{s}}$ are still strictly feasible (in the sense of belonging to $\operatorname{int}\mathcal{L}$). This part of the analysis is also inspired by the classical analysis, but it has to take into account the inexactness of the Newton system solution.
3. Proving that the new solution $(\mathbf{x} + \overline{\Delta\mathbf{x}}, \mathbf{y} + \overline{\Delta\mathbf{y}}, \mathbf{s} + \overline{\Delta\mathbf{s}})$ is in the neighborhood of the central path, and the duality gap/central path parameter have decreased by a factor of $1 - \alpha/\sqrt{r}$, where α is constant. This part is the most technical, and while it is inspired by [KP20a], it required using many of the Jordan-algebraic tools from [AG03; MT00].

Theorem 2.10 formalizes the fact that the Algorithm 1 has the same iteration invariant as the classical IPM. Since the duality gap is reduced by the same factor in both algorithms, their iteration complexity is the same, and a simple calculation shows that they need $O(\sqrt{r})$ iterations to halve the duality gap. On the other hand, the cost of each iteration varies, since the complexity of the quantum linear system solver depends on the precision $\xi\delta$, the condition number κ of the Newton matrix, as well as its ζ-parameter. While exactly bounding these quantities is the subject of future research, it is worth noting that for ζ, we have the trivial bound $\zeta \leq \sqrt{n}$, and research on iterative linear algebra methods [Dol05] suggests that $\kappa = O(1/\mu) = O(1/\epsilon)$. The final complexity is summarized in the following theorem:

Theorem. *Let (2.2) be a SOCP with $A \in \mathbb{R}^{m\times n}$, $m \leq n$, and $\mathcal{L} = \mathcal{L}^{n_1} \times \cdots \times \mathcal{L}^{n_r}$. Then, Algorithm 1 achieves duality gap ϵ in time*

$$T = \widetilde{O}\left(\sqrt{r}\log\left(\mu_0/\epsilon\right) \cdot \frac{n\kappa\zeta}{\delta^2}\log\left(\frac{\kappa\zeta}{\delta}\right)\right),$$

where the $\widetilde{O}(\cdot)$ notation hides the factors that are poly-logarithmic in n and m.

Finally, the quality of the resulting (classical) solution is characterized by the following theorem:

Theorem. *Let (2.2) be a SOCP as in Theorem 2.18. Then, after T iterations, the (linear) infeasibility of the final iterate $\mathbf{x}, \mathbf{y}, \mathbf{s}$ is bounded as*

$$\|A\mathbf{x} - \mathbf{b}\| \leq \delta\|A\|,$$
$$\left\|A^\top\mathbf{y} + \mathbf{s} - \mathbf{c}\right\| \leq \delta\left(\|A\| + 1\right).$$

2.4 Technical results

In this section, we present our main technical results – the proofs of Theorems 2.10, 2.18 and 2.19.

2.4.1 Central path

In addition to the central path defined in (2.6), we define the distance from the central path as $d(\mathbf{x}, \mathbf{s}, \nu) = \|T_{\mathbf{x}}\mathbf{s} - \nu\mathbf{e}\|_F$, so the corresponding η-neighborhood is given by

$$\mathcal{N}_\eta(\nu) := \{(\mathbf{x}, \mathbf{y}, \mathbf{s}) \mid (\mathbf{x}, \mathbf{y}, \mathbf{s}) \text{ strictly feasible and } d(\mathbf{x}, \mathbf{s}, \nu) \leq \eta\nu\}.$$

Using this neighborhood definition, we can specify what exactly do we mean when we claim that the important properties of the central path are valid in its neighborhood as well.

Lemma 2.9 (Properties of the central path)**.** *Let $\nu > 0$ be arbitrary and let $\mathbf{x}, \mathbf{s} \in \operatorname{int}\mathcal{L}$. Then, $\mathbf{x}$ and $\mathbf{s}$ satisfy the following properties:*

1. *For all $\nu > 0$, the duality gap and distance from the central path are related as*

$$|\mathbf{x}^\top\mathbf{s} - r\nu| \leq \sqrt{\frac{r}{2}} \cdot d(\mathbf{x}, \mathbf{s}, \nu).$$

2. *The distance from the central path is symmetric in its arguments i.e. $d(\mathbf{x}, \mathbf{s}, \nu) = d(\mathbf{s}, \mathbf{x}, \nu)$.*

3. Let $\mu = \frac{1}{r}\mathbf{x}^\top \mathbf{s}$. If $d(\mathbf{x},\mathbf{s},\mu) \leq \eta\mu$, then $(1+\eta)\|\mathbf{s}^{-1}\|_2 \geq \|\mu^{-1}\mathbf{x}\|_2$.

Proof. For part 1, let $\{\lambda_i\}_{i=1}^{2r}$ be the eigenvalues of $T_{\mathbf{x}}\mathbf{s}$, note that T_x is invertible as $x \in \mathcal{L}$. Then using the properties of $T_{\mathbf{x}}$, we have

$$\mathbf{x}^\top \mathbf{s} = \mathbf{x}^\top T_{\mathbf{x}}^{-1} T_{\mathbf{x}} \mathbf{s} = (T_{\mathbf{x}^{-1}}\mathbf{x})^\top T_{\mathbf{x}}\mathbf{s} = \mathbf{e}^\top T_{\mathbf{x}}\mathbf{s} = \frac{1}{2}\sum_{i=1}^{2r}\lambda_i.$$

We can therefore bound the duality gap $x^\top s$ as follows,

$$\mathbf{x}^\top \mathbf{s} = \frac{1}{2}\sum_{i=1}^{2r}\lambda_i \leq r\nu + \frac{1}{2}\sum_{i=1}^{2r}|\lambda_i - \nu| \leq r\nu + \sqrt{\frac{r}{2}}\sqrt{\sum_{i=1}^{2r}(\lambda_i-\nu)^2} = r\nu + \sqrt{\frac{r}{2}}\cdot d(\mathbf{x},\mathbf{s},\nu).$$

The second step used the Cauchy-Schwarz inequality while the third follows from the definition $d(\mathbf{x},\mathbf{s},\nu)^2 = \sum_{i=1}^{2r}(\lambda_i-\nu)^2$. The proof of the lower bound is similar, but starts instead with the inequality

$$\frac{1}{2}\sum_{i=1}^{2r}\lambda_i \geq r\nu - \frac{1}{2}\sum_{i=1}^{2r}|\nu - \lambda_i|.$$

For part 2, it suffices to prove that $T_{\mathbf{x}}\mathbf{s}$ and $T_{\mathbf{s}}\mathbf{x}$ have the same eigenvalues. This follows from part 2 of Theorem 10 in [AG03]. Finally for part 3, as $d(\mathbf{s},\mathbf{x},\mu) \leq \eta\mu$ we have,

$$\begin{aligned}
\eta\mu &\geq \|T_{\mathbf{s}}\mathbf{x} - \mu\mathbf{e}\|_F \\
&= \|T_{\mathbf{s}}\mathbf{x} - \mu\left(T_{\mathbf{s}}T_{\mathbf{s}^{-1}}\right)\mathbf{e}\|_F \\
&= \|T_{\mathbf{s}}\left(\mathbf{x} - \mu T_{\mathbf{s}^{-1}}\mathbf{e}\right)\|_F \\
&\geq \lambda_{\min}(T_{\mathbf{s}})\|\mathbf{x} - \mu T_{\mathbf{s}^{-1}}\mathbf{e}\|_F \\
&\geq \lambda_{\min}(T_{\mathbf{s}})\|\mathbf{x} - \mu\mathbf{s}^{-1}\|_2 \\
&= \frac{1}{\|\mathbf{s}^{-1}\|_2}\cdot\mu\cdot\|\mu^{-1}\mathbf{x} - \mathbf{s}^{-1}\|_2
\end{aligned}$$

Therefore, $\eta\|\mathbf{s}^{-1}\|_2 \geq \|\mu^{-1}\mathbf{x} - \mathbf{s}^{-1}\|_2$. Finally, by the triangle inequality for the spectral norm,

$$\eta\|\mathbf{s}^{-1}\|_2 \geq \|\mu^{-1}\mathbf{x}\|_2 - \|\mathbf{s}^{-1}\|_2,$$

so we can conclude that $\|\mathbf{s}^{-1}\|_2 \geq \frac{1}{1+\eta}\|\mu^{-1}\mathbf{x}\|_2$. □

2.4.2 A single quantum IPM iteration

Recall that the essence of our quantum algorithm is repeated solution of the Newton system (2.7) using quantum linear algebra. As such, our goal is to prove the following theorem:

Theorem 2.10 (Per-iteration correctness, formal)**.** *Let $\chi = \eta = 0.01$ and $\xi = 0.001$ be positive constants and let $(\mathbf{x},\mathbf{y},\mathbf{s})$ be a feasible solution of (2.2) and (2.3) with $\mu = \frac{1}{r}\mathbf{x}^\top\mathbf{s}$ and $d(\mathbf{x},\mathbf{s},\mu) \leq \eta\mu$. Then, for $\sigma = 1 - \chi/\sqrt{n}$, the Newton system (2.7) has a unique solution $(\Delta\mathbf{x},\Delta\mathbf{y},\Delta\mathbf{s})$. Let $\overline{\Delta\mathbf{x}},\overline{\Delta\mathbf{s}}$ be approximate solutions of (2.7) that satisfy*

$$\|\Delta\mathbf{x} - \overline{\Delta\mathbf{x}}\|_F \leq \frac{\xi}{\|T_{\mathbf{x}^{-1}}\|}, \|\Delta\mathbf{s} - \overline{\Delta\mathbf{s}}\|_F \leq \frac{\xi}{2\|T_{\mathbf{s}^{-1}}\|},$$

where $T_{\mathbf{x}}$ and $T_{\mathbf{x}}$ are the square roots of the quadratic representation matrices in equation (2.5).

If we let $\mathbf{x}_{next} := \mathbf{x} + \overline{\Delta\mathbf{x}}$ *and* $\mathbf{s}_{next} := \mathbf{s} + \overline{\Delta\mathbf{s}}$*, the following holds:*

1. *The updated solution is strictly feasible, i.e.* $\mathbf{x}_{next} \in \operatorname{int}\mathcal{L}$ *and* $\mathbf{s}_{next} \in \operatorname{int}\mathcal{L}$.
2. *The updated solution satisfies* $d(\mathbf{x}_{next}, \mathbf{s}_{next}, \overline{\mu}) \leq \eta\overline{\mu}$ *and* $\frac{1}{r}\mathbf{x}_{next}^{\top}\mathbf{s}_{next} = \overline{\mu}$ *for* $\overline{\mu} = \overline{\sigma}\mu$, $\overline{\sigma} = 1 - \frac{\alpha}{\sqrt{r}}$ *and a constant* $0 < \alpha \leq \chi$.

Since the Newton system (2.7) is the same as in the classical case, we can reuse Theorem 1 from [MT00] for the uniqueness part of Theorem 2.10. Therefore, we just need to prove the two parts about strict feasibility and improving the duality gap. Our analysis is inspired by the general case analysis from [BN01], the derived SDP analysis from [KP20a], and uses some technical results from the SOCP analysis in [MT00]. The proof of Theorem 2.10 consists of three main steps:

1. Rescaling $\mathbf{x}$ and $\mathbf{s}$ so that they share the same Jordan frame.
2. Bounding the norms of $\Delta\mathbf{x}$ and $\Delta\mathbf{s}$, and proving that $\mathbf{x} + \Delta\mathbf{x}$ and $\mathbf{s} + \Delta\mathbf{s}$ are still strictly feasible (in the sense of belonging to $\operatorname{int}\mathcal{L}$).
3. Proving that the new solution $(\mathbf{x} + \Delta\mathbf{x}, \mathbf{y} + \Delta\mathbf{y}, \mathbf{s} + \Delta\mathbf{s})$ is in the η-neighborhood of the central path, and the duality gap/central path parameter have decreased by a factor of $1 - \alpha/\sqrt{n}$, where α is constant.

2.4.3 Rescaling x and s

As in the case of SDPs, the first step of the proof uses the symmetries of the Lorentz cone to perform a commutative scaling, that is to reduce the analysis to the case when $\mathbf{x}$ and $\mathbf{s}$ share the same Jordan frame. Although $\circ$ is commutative by definition, two vectors sharing a Jordan frame are akin to two matrices sharing a system of eigenvectors, and thus commuting (some authors [AG03] say that the vectors *operator commute* in this case). The easiest way to achieve this is to scale by $T_{\mathbf{x}} = Q_{\mathbf{x}^{1/2}}$ and μ^{-1}, i.e. to change our variables as

$$\mathbf{x} \mapsto \mathbf{x}' := T_{\mathbf{x}}^{-1}\mathbf{x} = \mathbf{e} \text{ and } \mathbf{s} \mapsto \mathbf{s}' := \mu^{-1}T_{\mathbf{x}}\mathbf{s}.$$

Note that for convenience, we have also rescaled the duality gap to 1. Recall also that in the matrix case, the equivalent of this scaling was $X \mapsto X^{-1/2}XX^{-1/2} = I$ and $S \mapsto \mu^{-1}X^{1/2}SX^{1/2}$. We use the notation $\mathbf{z}'$ to denote the appropriately-scaled vector $\mathbf{z}$, so that we have

$$\Delta\mathbf{x}' := T_{\mathbf{x}}^{-1}\Delta\mathbf{x}, \quad \Delta\mathbf{s}' := \mu^{-1}T_{\mathbf{x}}\Delta\mathbf{s}$$

For approximate quantities (e.g. the ones obtained using tomography, or any other approximate linear system solver), we use the notation $\overline{\cdot}$, so that the increments become $\overline{\Delta\mathbf{x}}$ and $\overline{\Delta\mathbf{s}}$, and their scaled counterparts are $\overline{\Delta\mathbf{x}}' := T_{\mathbf{x}}^{-1}\overline{\Delta\mathbf{x}}$ and $\overline{\Delta\mathbf{s}}' := \mu^{-1}T_{\mathbf{x}}\overline{\Delta\mathbf{s}}$. Finally, we denote the scaled version of the next iterate as $\mathbf{x}'_{\text{next}} := \mathbf{e} + \overline{\Delta\mathbf{x}}'$ and $\mathbf{s}'_{\text{next}} := \mathbf{s}' + \overline{\Delta\mathbf{s}}'$. Now, we see that the statement of

Theorem 2.10 implies the following bounds on $\left\|\Delta\mathbf{x}' - \overline{\Delta\mathbf{x}}'\right\|_F$ and $\left\|\Delta\mathbf{s}' - \overline{\Delta\mathbf{s}}'\right\|_F$:

$$\begin{aligned}
\left\|\Delta\mathbf{x}' - \overline{\Delta\mathbf{x}}'\right\|_F &= \left\|T_{\mathbf{x}^{-1}}\Delta\mathbf{x} - T_{\mathbf{x}^{-1}}\overline{\Delta\mathbf{x}}\right\|_F \\
&\leq \|T_{\mathbf{x}^{-1}}\| \cdot \left\|\Delta\mathbf{x} - \overline{\Delta\mathbf{x}}\right\|_F \leq \xi, \text{ and} \\
\left\|\Delta\mathbf{s}' - \overline{\Delta\mathbf{s}}'\right\|_F &= \mu^{-1}\left\|T_{\mathbf{x}}\Delta\mathbf{s} - T_{\mathbf{x}}\overline{\Delta\mathbf{s}}\right\|_F \\
&\leq \mu^{-1}\|T_{\mathbf{x}}\|\left\|\Delta\mathbf{s} - \overline{\Delta\mathbf{s}}\right\|_F \\
&= \mu^{-1}\|\mathbf{x}\|_2\left\|\Delta\mathbf{s} - \overline{\Delta\mathbf{s}}\right\|_F \\
&\leq (1+\eta)\|\mathbf{s}^{-1}\|_2\left\|\Delta\mathbf{s} - \overline{\Delta\mathbf{s}}\right\|_F \text{ by Lemma 2.9} \\
&\leq 2\|T_{\mathbf{s}^{-1}}\|\left\|\Delta\mathbf{s} - \overline{\Delta\mathbf{s}}\right\|_F \leq \xi.
\end{aligned}$$

Throughout the analysis, we will make use of several constants: $\eta > 0$ is the distance from the central path, i.e. we ensure that our iterates stay in the η-neighborhood $\mathcal{N}_\eta$ of the central path. The constant $\sigma = 1 - \chi/\sqrt{r}$ is the factor by which we aim to decrease our duality gap, for some constant $\chi > 0$. Finally constant $\xi > 0$ is the approximation error for the scaled increments $\overline{\Delta\mathbf{x}}', \overline{\Delta\mathbf{s}}'$. Having this notation in mind, we can state several facts about the relation between the duality gap and the central path distance for the original and scaled vectors.

Claim 2.11. *The following holds for the scaled vectors $\mathbf{x}'$ and $\mathbf{s}'$:*

1. *The scaled duality gap is $\frac{1}{r}\mathbf{x}'^\top\mathbf{s}' = 1$.*
2. *$d(\mathbf{x}, \mathbf{s}, \mu) \leq \eta\mu$ is equivalent to $\|\mathbf{s}' - \mathbf{e}\| \leq \eta$.*
3. *$d(\mathbf{x}, \mathbf{s}, \mu\sigma) = \mu \cdot d(\mathbf{x}', \mathbf{s}', \sigma)$, for all $\sigma > 0$.*

At this point, we claim that it suffices to prove the two parts of Theorem 2.10 in the scaled case. Namely, assuming that $\mathbf{x}'_{\text{next}} \in \operatorname{int}\mathcal{L}$ and $\mathbf{s}'_{\text{next}} \in \operatorname{int}\mathcal{L}$, by construction we get

$$\mathbf{x}_{\text{next}} = T_{\mathbf{x}}\mathbf{x}'_{\text{next}} \text{ and } \mathbf{s}_{\text{next}} = T_{\mathbf{x}^{-1}}\mathbf{s}'_{\text{next}}$$

and thus $\mathbf{x}_{\text{next}}, \mathbf{s}_{\text{next}} \in \operatorname{int}\mathcal{L}$.

On the other hand, if $\mu d(\mathbf{x}'_{\text{next}}, \mathbf{s}'_{\text{next}}, \overline{\sigma}) \leq \eta\overline{\mu}$, then $d(\mathbf{x}_{\text{next}}, \mathbf{s}_{\text{next}}, \overline{\mu}) \leq \eta\overline{\mu}$ follows by Claim 2.11. Similarly, from $\frac{1}{r}\mathbf{x}'^\top_{\text{next}}\mathbf{s}'_{\text{next}} = \overline{\sigma}$, we also get $\frac{1}{r}\mathbf{x}^\top_{\text{next}}\mathbf{s}_{\text{next}} = \overline{\mu}$. We conclude this part with two technical results from [MT00], that use the auxiliary matrix R_{xs} defined as $R_{xs} := T_{\mathbf{x}} \operatorname{Arw}(\mathbf{x})^{-1} \operatorname{Arw}(\mathbf{s}) T_{\mathbf{x}}$. These results are useful for the later parts of the proof of Theorem 2.10.

Claim 2.12 ([MT00], Lemma 3)**.** *Let η be the distance from the central path, and let $\nu > 0$ be arbitrary. Then, R_{xs} is bounded as*

$$\|R_{xs} - \nu I\| \leq 3\eta\nu.$$

Claim 2.13 ([MT00], Lemma 5, proof)**.** *Let μ be the duality gap. Then, the scaled increment $\Delta\mathbf{s}'$ is*

$$\Delta\mathbf{s}' = \sigma\mathbf{e} - \mathbf{s}' - \mu^{-1}R_{xs}\Delta\mathbf{x}'.$$

2.4.4 Maintaining strict feasibility

The main tool for showing that strict feasibility is conserved is the following bound on the increments $\Delta\mathbf{x}'$ and $\Delta\mathbf{s}'$:

Lemma 2.14 ([MT00], Lemma 6)**.** *Let η be the distance from the central path and let μ be the duality gap. Then, we have the following bounds for the scaled direction:*

$$\begin{array}{ll} \|\Delta\mathbf{x}'\|_F & \leq \frac{\Theta}{\sqrt{2}} \\ \|\Delta\mathbf{s}'\|_F & \leq \Theta\sqrt{2} \end{array}, \quad \text{where} \quad \Theta = \frac{2\sqrt{\eta^2/2 + (1-\sigma)^2 r}}{1-3\eta}$$

Moreover, if we substitute σ with its actual value $1-\chi/\sqrt{r}$, we get $\Theta = \frac{\sqrt{2\eta^2+4\chi^2}}{1-3\eta}$, which we can make arbitrarily small by tuning the constants. Now, we can immediately use this result to prove $\mathbf{x}'_{\text{next}}, \mathbf{s}'_{\text{next}} \in \text{int}\,\mathcal{L}$.

Lemma 2.15. *Let $\eta = \chi = 0.01$ and $\xi = 0.001$. Then, $\mathbf{x}'_{next}$ and $\mathbf{s}'_{next}$ are strictly feasible, i.e. $\mathbf{x}'_{next}, \mathbf{s}'_{next} \in \text{int}\,\mathcal{L}$.*

Proof. By Lemma 2.14, $\lambda_{\min}(\overline{\mathbf{x}}) \geq 1 - \left\|\overline{\Delta\mathbf{x}}'\right\|_F \geq 1 - \frac{\Theta}{\sqrt{2}} - \xi$. On the other hand, since $d(\mathbf{x},\mathbf{s},\mu) \leq \eta\mu$, we have $d(\mathbf{x}',\mathbf{s}',1) \leq \eta$, and thus

$$\eta^2 \geq \|\mathbf{s}' - e\|_F^2 = \sum_{i=1}^{2r} (\lambda_i(\mathbf{s}') - 1)^2$$

The above equation implies that $\lambda_i(\mathbf{s}') \in [1-\eta, 1+\eta]\,, \forall i \in [2r]$. Now, since $\|\mathbf{z}\|_2 \leq \|\mathbf{z}\|_F$,

$$\begin{aligned} \lambda_{\min}(\overline{\mathbf{s}}) &\geq \lambda_{\min}(\mathbf{s}' + \Delta\mathbf{s}') - \left\|\overline{\Delta\mathbf{s}}' - \Delta\mathbf{s}'\right\|_F \\ &\geq \lambda_{\min}(\mathbf{s}') - \|\Delta\mathbf{s}'\|_F - \left\|\overline{\Delta\mathbf{s}}' - \Delta\mathbf{s}'\right\|_F \\ &\geq 1 - \eta - \Theta\sqrt{2} - \xi, \end{aligned}$$

where we used Lemma 2.14 for the last inequality. Substituting $\eta = \chi = 0.01$ and $\xi = 0.001$, we get that $\lambda_{\min}(\overline{\mathbf{x}}) \geq 0.8$ and $\lambda_{\min}(\overline{\mathbf{s}}) \geq 0.8$. □

2.4.5 Maintaining closeness to central path

Finally, we move on to the most technical part of the proof of Theorem 2.10, where we prove that $\mathbf{x}'_{\text{next}}, \mathbf{s}'_{\text{next}}$ is still close to the central path, and the duality gap has decreased by a constant factor. We split this into two lemmas.

Lemma 2.16. *Let $\eta = \chi = 0.01$, $\xi = 0.001$, and let α be any value satisfying $0 < \alpha \leq \chi$. Then, for $\overline{\sigma} = 1 - \alpha/\sqrt{r}$, the distance to the central path is maintained, that is, $d(\mathbf{x}'_{next}, \mathbf{s}'_{next}, \overline{\sigma}) < \eta\overline{\sigma}$.*

Proof. By Claim 2.11, the distance of the next iterate from the central path is

$$d(\mathbf{x}'_{\text{next}}, \mathbf{s}'_{\text{next}}, \overline{\sigma}) = \left\|T_{\mathbf{x}'_{\text{next}}}\mathbf{s}'_{\text{next}} - \overline{\sigma}\mathbf{e}\right\|_F,$$

and we can bound it from above as

$$\begin{aligned} d(\mathbf{x}'_{\text{next}}, \mathbf{s}'_{\text{next}}, \overline{\sigma}) &= \left\|T_{\mathbf{x}'_{\text{next}}}\mathbf{s}'_{\text{next}} - \overline{\sigma}\mathbf{e}\right\|_F \\ &= \left\|T_{\mathbf{x}'_{\text{next}}}\mathbf{s}'_{\text{next}} - \overline{\sigma}T_{\mathbf{x}'_{\text{next}}}T_{\mathbf{x}'^{-1}_{\text{next}}}\mathbf{e}\right\|_F \\ &\leq \left\|T_{\mathbf{x}'_{\text{next}}}\right\| \cdot \left\|\mathbf{s}'_{\text{next}} - \overline{\sigma}\cdot(\mathbf{x}'_{\text{next}})^{-1}\right\|. \end{aligned}$$

So, it is enough to bound $\|\mathbf{z}\|_F := \|\mathbf{s}'_{\text{next}} - \overline{\sigma} \cdot \mathbf{x}'^{-1}_{\text{next}}\|_F$ from above, since

$$\left\|T_{\mathbf{x}'_{\text{next}}}\right\| = \|\mathbf{x}'_{\text{next}}\|_2 \leq 1 + \left\|\overline{\Delta\mathbf{x}}'\right\|_2 \leq 1 + \|\Delta\mathbf{x}'\|_2 + \xi \leq 1 + \frac{\Theta}{\sqrt{2}} + \xi.$$

We split $\mathbf{z}$ as

$$\mathbf{z} = \underbrace{\left(\mathbf{s}' + \overline{\Delta\mathbf{s}}' - \overline{\sigma}e + \overline{\Delta\mathbf{x}}'\right)}_{\mathbf{z}_1} + \underbrace{(\overline{\sigma} - 1)\overline{\Delta\mathbf{x}}'}_{\mathbf{z}_2} + \underbrace{\overline{\sigma}\left(e - \overline{\Delta\mathbf{x}}' - (e + \overline{\Delta\mathbf{x}}')^{-1}\right)}_{\mathbf{z}_3},$$

and we bound $\|\mathbf{z}_1\|_F$,$\|\mathbf{z}_2\|_F$, and $\|\mathbf{z}_3\|_F$ separately.

1. By the triangle inequality, $\|\mathbf{z}_1\|_F \leq \|\mathbf{s}' + \Delta\mathbf{s}' - \overline{\sigma}e + \Delta\mathbf{x}'\|_F + 2\xi$. Furthermore, after substituting $\Delta\mathbf{s}'$ from Claim 2.13, we get

$$\begin{aligned}\mathbf{s}' + \Delta\mathbf{s}' - \overline{\sigma}e + \Delta\mathbf{x}' &= \sigma e - \mu^{-1}R_{xs}\Delta\mathbf{x}' - \overline{\sigma}e + \Delta\mathbf{x}' \\ &= \frac{\alpha - \chi}{\sqrt{r}}e + \mu^{-1}(\mu I - R_{xs})\Delta\mathbf{x}'.\end{aligned}$$

   Using the bound for $\|\mu I - R_{xs}\|$ from Claim 2.12 as well as the bound for $\|\Delta\mathbf{x}'\|_F$ from Lemma 2.14, we obtain

$$\|\mathbf{z}_1\|_F \leq 2\xi + \frac{\chi}{\sqrt{r}} + \frac{3}{\sqrt{2}}\eta\Theta.$$

2. $\|\mathbf{z}_2\|_F \leq \frac{\chi}{\sqrt{r}}\left(\frac{\Theta}{\sqrt{2}} + \xi\right)$, where we used the bound from Lemma 2.14 again.

3. Here, we first need to bound $\left\|(e + \Delta\mathbf{x}')^{-1} - (e + \overline{\Delta\mathbf{x}}')^{-1}\right\|_F$. For this, we use the submultiplicativity of $\|\cdot\|_F$:

$$\begin{aligned}\|(e + \Delta\mathbf{x}')^{-1} - (e + \overline{\Delta\mathbf{x}}')^{-1}\|_F &= \left\|(e + \Delta\mathbf{x}')^{-1} \circ \left(e - (e + \Delta\mathbf{x}') \circ (e + \overline{\Delta\mathbf{x}}')^{-1}\right)\right\|_F \\ &\leq \|(e + \Delta\mathbf{x}')^{-1}\|_2 \cdot \left\|e - (e + \overline{\Delta\mathbf{x}}' + \Delta\mathbf{x}' - \overline{\Delta\mathbf{x}}') \circ (e + \overline{\Delta\mathbf{x}}')^{-1}\right\|_F \\ &= \|(e + \Delta\mathbf{x}')^{-1}\|_2 \cdot \left\|(\Delta\mathbf{x}' - \overline{\Delta\mathbf{x}}') \circ (e + \overline{\Delta\mathbf{x}}')^{-1}\right\|_F \\ &\leq \|(e + \Delta\mathbf{x}')^{-1}\|_2 \cdot \left\|\Delta\mathbf{x}' - \overline{\Delta\mathbf{x}}'\right\|_F \cdot \left\|(e + \overline{\Delta\mathbf{x}}')^{-1}\right\|_2 \\ &\leq \xi \cdot \|(e + \Delta\mathbf{x}')^{-1}\|_2 \cdot \left\|(e + \overline{\Delta\mathbf{x}}')^{-1}\right\|_2.\end{aligned}$$

   Now, we have the bound $\|(e + \Delta\mathbf{x}')^{-1}\|_2 \leq \frac{1}{1 - \|\Delta\mathbf{x}'\|_F}$ and similarly $\left\|(e + \overline{\Delta\mathbf{x}}')^{-1}\right\|_2 \leq \frac{1}{1 - \|\Delta\mathbf{x}'\|_F - \xi}$, so we get

$$\left\|(e + \Delta\mathbf{x}')^{-1} - (e + \overline{\Delta\mathbf{x}}')^{-1}\right\|_F \leq \frac{\xi}{(1 - \|\Delta\mathbf{x}'\|_F - \xi)^2}.$$

   Using this, we can bound $\|\mathbf{z}_3\|_F$:

$$\|\mathbf{z}_3\|_F \leq \overline{\sigma}\left(\|e - \Delta\mathbf{x}' - (e + \Delta\mathbf{x}')^{-1}\|_F + \xi + \frac{\xi}{(1 - \|\Delta\mathbf{x}'\|_F - \xi)^2}\right).$$

If we let λ_i be the eigenvalues of $\Delta\mathbf{x}'$, then by Lemma 2.14, we have

$$\begin{aligned}\left\|e-\Delta\mathbf{x}'-(e+\Delta\mathbf{x}')^{-1}\right\|_F &= \sqrt{\sum_{i=1}^{2r}\left((1-\lambda_i)-\frac{1}{1+\lambda_i}\right)^2}\\ &= \sqrt{\sum_{i=1}^{2r}\frac{\lambda_i^4}{(1+\lambda_i)^2}} \le \frac{\Theta}{\sqrt{2}-\Theta}\sqrt{\sum_{i=1}^{2r}\lambda_i^2}\\ &\le \frac{\Theta^2}{2-\sqrt{2}\Theta}.\end{aligned}$$

Combining all bound from above, we obtain

$$\begin{aligned}d(\mathbf{x}'_{\text{next}},\mathbf{s}'_{\text{next}},\overline{\sigma}) \le \left(1+\frac{\Theta}{\sqrt{2}}+\xi\right)\cdot\Bigg(&2\xi+\frac{\chi}{\sqrt{r}}+\frac{3}{\sqrt{2}}\eta\Theta\\ &+\frac{\chi}{\sqrt{r}}\left(\frac{\Theta}{\sqrt{2}}+\xi\right)\\ &+\overline{\sigma}\left(\frac{\Theta^2}{2-\sqrt{2}\Theta}+\xi+\frac{\xi}{(1-\Theta/\sqrt{2}-\xi)^2}\right)\Bigg).\end{aligned}$$

Finally, if we plug in $\chi=0.01$, $\eta=0.01$, $\xi=0.001$, we get $d(\overline{\mathbf{x}},\overline{\mathbf{s}},\overline{\sigma})\le 0.005\overline{\sigma}\le\eta\overline{\sigma}$. □

Now, we prove that the duality gap decreases.

Lemma 2.17. *For the same constants, the updated solution satisfies* $\frac{1}{r}\mathbf{x}'^{\top}_{next}\mathbf{s}'_{next}=\left(1-\frac{\alpha}{\sqrt{r}}\right)$ *for* $\alpha=0.005$.

Proof. Since $\mathbf{x}'_{\text{next}}$ and $\mathbf{s}'_{\text{next}}$ are scaled quantities, the duality gap between their unscaled counterparts is $\frac{\mu}{r}\mathbf{x}'^{\top}_{\text{next}}\mathbf{s}'_{\text{next}}$. Applying Lemma 2.9 (and Claim 2.11) with $\nu=\sigma\mu$ and $\mathbf{x}'_{\text{next}},\mathbf{s}'_{\text{next}}$, we obtain the upper bound

$$\mu\mathbf{x}'^{\top}_{\text{next}}\mathbf{s}'_{\text{next}} \le r\sigma\mu+\sqrt{\frac{r}{2}}\mu d(\mathbf{x}'_{\text{next}},\mathbf{s}'_{\text{next}},\sigma),$$

which in turn implies

$$\frac{1}{r}\mathbf{x}'^{\top}_{\text{next}}\mathbf{s}'_{\text{next}} \le \left(1-\frac{0.01}{\sqrt{r}}\right)\left(1+\frac{d(\mathbf{x}'_{\text{next}},\mathbf{s}'_{\text{next}},\sigma)}{\sigma\sqrt{2r}}\right).$$

By instantiating Lemma 2.9 for $\alpha=\chi$, from its proof, we obtain $d(\mathbf{x}'_{\text{next}},\mathbf{s}'_{\text{next}},\sigma)\le 0.005\sigma$, and thus

$$\frac{1}{r}\mathbf{x}'^{\top}_{\text{next}}\mathbf{s}'_{\text{next}} \le 1-\frac{0.005}{\sqrt{r}}$$

Therefore, the final α for this Lemma is 0.005. □

2.4.6 Final complexity and feasibility

In every iteration we need to solve the Newton system to a precision dependent on the norms of $T_{\mathbf{x}^{-1}}$ and $T_{\mathbf{s}^{-1}}$. Thus, to bound the running time of the algorithm (since the complexity of the vector

state tomography procedure depends on the desired precision), we need to bound $\|T_{\mathbf{x}^{-1}}\|$ and $\|T_{\mathbf{s}^{-1}}\|$. Indeed, by the properties of the quadratic representation, we get

$$\begin{aligned}\|T_{\mathbf{x}^{-1}}\| &= \|\mathbf{x}^{-1}\| = \lambda_{\min}(\mathbf{x})^{-1} \text{ and}\\ \|T_{\mathbf{s}^{-1}}\| &= \|\mathbf{s}^{-1}\| = \lambda_{\min}(\mathbf{s})^{-1}.\end{aligned}$$

If the tomography precision for iteration i is chosen to be at least (i.e. smaller than)

$$\delta_i := \frac{\xi}{4}\min\left\{\lambda_{\min}(\mathbf{x}_i), \lambda_{\min}(\mathbf{s}_i)\right\}, \tag{2.8}$$

then the premises of Theorem 2.10 are satisfied. The tomography precision for the entire algorithm can therefore be chosen to be $\delta := \min_i \delta_i$. Note that these minimum eigenvalues are related to how close the current iterate is to the boundary of $\mathcal{L}$ – as long as $\mathbf{x}_i, \mathbf{s}_i$ are not "too close" to the boundary of $\mathcal{L}$, their minimal eigenvalues should not be "too small".

There are two more parameters that impact the complexity of the quantum linear system solver: the condition number of the Newton matrix κ_i and the matrix parameter ζ_i of the QRAM encoding in iteration i. For both of these quantities we define their global versions as $\kappa = \max_i \kappa_i$ and $\zeta = \max_i \zeta_i$. Therefore, we arrive to the following statement about the complexity of Algorithm 1.

Theorem 2.18. *Let (2.2) be a SOCP with $A \in \mathbb{R}^{m\times n}$, $m \leq n$, and $\mathcal{L} = \mathcal{L}^{n_1} \times \cdots \times \mathcal{L}^{n_r}$. Then, our algorithm achieves duality gap ϵ in time*

$$T = \widetilde{O}\left(\sqrt{r}\log\left(\mu_0/\epsilon\right) \cdot \frac{n\kappa\zeta}{\delta^2}\log\left(\frac{\kappa\zeta}{\delta}\right)\right).$$

This complexity can be easily interpreted as product of the number of iterations and the cost of n-dimensional vector tomography with error δ. So, improving the complexity of the tomography algorithm would improve the running time of our algorithm as well.

Note that up to now, we cared mostly about strict (conic) feasibility of $\mathbf{x}$ and $\mathbf{s}$. Now, we address the fact that the linear constraints $A\mathbf{x} = \mathbf{b}$ and $A^\top\mathbf{y} + \mathbf{s} = \mathbf{c}$ are not exactly satisfied during the execution of the algorithm. Luckily, it turns out that this error is not accumulated, but is instead determined just by the final tomography precision:

Theorem 2.19. *Let (2.2) be a SOCP as in Theorem 2.18. Then, after T iterations, the (linear) infeasibility of the final iterate $\mathbf{x}, \mathbf{y}, \mathbf{s}$ is bounded as*

$$\begin{aligned}\|A\mathbf{x}_T - \mathbf{b}\| &\leq \delta\|A\|,\\ \left\|A^\top\mathbf{y}_T + \mathbf{s}_T - \mathbf{c}\right\| &\leq \delta\left(\|A\| + 1\right).\end{aligned}$$

Proof. Let $(\mathbf{x}_T, \mathbf{y}_T, \mathbf{s}_T)$ be the T-th iterate. Then, the following holds for $A\mathbf{x}_T - \mathbf{b}$:

$$A\mathbf{x}_T - \mathbf{b} = A\mathbf{x}_0 + A\sum_{t=1}^{\top}\overline{\Delta\mathbf{x}}_t - \mathbf{b} = A\sum_{t=1}^{\top}\overline{\Delta\mathbf{x}}_t. \tag{2.9}$$

On the other hand, the Newton system at iteration T has the constraint $A\Delta\mathbf{x}_T = \mathbf{b} - A\mathbf{x}_{T-1}$, which we can further recursively transform as,

$$\begin{aligned}A\Delta\mathbf{x}_T &= \mathbf{b} - A\mathbf{x}_{T-1} = \mathbf{b} - A\left(\mathbf{x}_{T-2} + \overline{\Delta\mathbf{x}}_{T-1}\right)\\ &= \mathbf{b} - A\mathbf{x}_0 - \sum_{t=1}^{T-1}\overline{\Delta\mathbf{x}}_t = -\sum_{t=1}^{T-1}\overline{\Delta\mathbf{x}}_t.\end{aligned}$$

Substituting this into equation (2.9), we get

$$A\mathbf{x}_T - \mathbf{b} = A\left(\overline{\Delta\mathbf{x}}_T - \Delta\mathbf{x}_T\right).$$

Similarly, using the constraint $A^\top \Delta\mathbf{y}_T + \Delta\mathbf{s}_T = \mathbf{c} - \mathbf{s}_{T-1} - A^\top \mathbf{y}_{T-1}$ we obtain that

$$A^\top \mathbf{y}_T + \mathbf{s}_T - \mathbf{c} = A^\top\left(\overline{\Delta\mathbf{y}}_T - \Delta\mathbf{y}_T\right) + \left(\overline{\Delta\mathbf{s}}_T - \Delta\mathbf{s}_T\right).$$

Finally, we can bound the norms of these two quantities,

$$\begin{aligned} \|A\mathbf{x}_T - \mathbf{b}\| &\leq \delta\|A\|, \\ \left\|A^\top \mathbf{y}_T + \mathbf{s}_T - \mathbf{c}\right\| &\leq \delta\left(\|A\| + 1\right). \end{aligned}$$

□

2.5 Quantum Support-Vector Machines

In this section we present our quantum support-vector machine (SVM) algorithm as an application of our SOCP solver. Given a set of vectors $\mathcal{X} = \{\mathbf{x}^{(i)} \in \mathbb{R}^n \mid i \in [m]\}$ (*training examples*) and their *labels* $y^{(i)} \in \{-1, 1\}$, the objective of the SVM training process is to find the "best" hyperplane that separates training examples with label 1 from those with label -1. In this section we focus on the (traditional) *soft-margin* (ℓ_1-)SVM, which can be expressed as the following optimization problem:

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,\boldsymbol{\xi}} & \|\mathbf{w}\|^2 + C\|\boldsymbol{\xi}\|_1 \\ \text{s.t.} & y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) \geq 1 - \xi_i, \ \forall i \in [m] \\ & \boldsymbol{\xi} \geq 0. \end{array} \tag{2.10}$$

Here, the variables $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ correspond to the hyperplane, $\boldsymbol{\xi} \in \mathbb{R}^m$ corresponds to the "linear inseparability" of each point, and the constant $C > 0$ is a hyperparameter that quantifies the tradeoff between maximizing the margin and minimizing the constraint violations.

As a slightly less traditional alternative, one might also consider the ℓ_2-SVM (or least-squares SVM, LS-SVM) [SV99], where the $\|\boldsymbol{\xi}\|_1$ regularization term is replaced by $\|\boldsymbol{\xi}\|^2$. This formulation arises from considering the least-squares regression problem with the constraints $y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) = 1$, which we solve by minimizing the squared 2-norm of the residuals:

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,\boldsymbol{\xi}} & \|\mathbf{w}\|^2 + C\|\boldsymbol{\xi}\|^2 \\ \text{s.t.} & y^{(i)}(\mathbf{w}^\top \mathbf{x}^{(i)} + b) = 1 - \xi_i, \ \forall i \in [m] \end{array} \tag{2.11}$$

Since this is a least-squares problem, the optimal $\mathbf{w}, b$ and $\boldsymbol{\xi}$ can be obtained by solving a linear system. In [RML14], a quantum algorithm for LS-SVM is presented, which uses a single quantum linear system solver. Unfortunately, replacing the ℓ_1-norm with ℓ_2 in the objective of (2.11) leads to the loss of a key property of (ℓ_1-)SVM – weight sparsity [Suy+02].

2.5.1 Reducing SVM to SOCP

Finally, we are going to reduce the SVM problem (2.10) to SOCP. In order to do that, we define an auxiliary vector $\mathbf{t} = (t+1; t; \mathbf{w})$, where $t \in \mathbb{R}$ – this allows us to "compute" $\|\mathbf{w}\|^2$ using the constraint $\mathbf{t} \in \mathcal{L}^{n+2}$ since

$$\mathbf{t} \in \mathcal{L}^{n+2} \Leftrightarrow (t+1)^2 \geq t^2 + \|\mathbf{w}\|^2 \Leftrightarrow 2t + 1 \geq \|\mathbf{w}\|^2.$$

Thus, minimizing $\|\mathbf{w}\|^2$ is equivalent to minimizing t. Note we can restrict our bias b to be nonnegative without any loss in generality, since the case $b < 0$ can be equivalently described by a bias $-b > 0$ and weights $-\mathbf{w}$. Using these transformations, we can restate (2.10) as the following SOCP:

$$\begin{aligned} \min_{\mathbf{t},b,\boldsymbol{\xi}} \quad & \begin{bmatrix} 0 & 1 & 0^n & 0 & C^m \end{bmatrix} \begin{bmatrix} \mathbf{t} & b & \boldsymbol{\xi} \end{bmatrix}^\top \\ \text{s.t.} \quad & \begin{bmatrix} 0 & 0 & & 1 & \\ \vdots & \vdots & X^\top & \vdots & \operatorname{diag}(\mathbf{y}) \\ 0 & 0 & & 1 & \\ 1 & -1 & 0^n & 0 & 0^m \end{bmatrix} \begin{bmatrix} \mathbf{t} \\ b \\ \boldsymbol{\xi} \end{bmatrix} = \begin{bmatrix} \mathbf{y} \\ 1 \end{bmatrix} \\ & \mathbf{t} \in \mathcal{L}^{n+2},\ b \in \mathcal{L}^1,\ \xi_i \in \mathcal{L}^1 \quad \forall i \in [m] \end{aligned} \tag{2.12}$$

Here, we use the notation $X \in \mathbb{R}^{n \times m}$ for the matrix whose columns are the training examples $\mathbf{x}^{(i)}$, and $\mathbf{y} \in \mathbb{R}^m$ for the vector of labels. This problem has $O(n+m)$ variables, and $O(m)$ conic constraints (i.e. its rank is $r = O(m)$). Therefore, in the interesting case of $m = \Theta(n)$, it can be solved in $\widetilde{O}(\sqrt{n})$ iterations. More precisely, if we consider both the primal and the dual, in total they have $3m + 2n + 7$ scalar variables and $2m + 4$ conic constraints.

In practice (as evidenced by the LIBSVM and LIBLINEAR libraries [CL11; Fan+08]), a small modification is made to the formulations (2.10) and (2.11): instead of treating the bias separately, all data points are extended with a constant unit coordinate. In this case, the SOCP formulation remains almost identical, with the only difference being that the constraints $\mathbf{t} \in \mathcal{L}^{n+2}$ and $b \in \mathcal{L}^1$ are replaced by a single conic constraint $(\mathbf{t}; b) \in \mathcal{L}^{n+3}$. This change allows us to come up with a simple feasible initial solution in our numerical experiments, without going through the homogeneous self-dual formalism of [YTM94].

Note also that we can solve the LS-SVM problem (2.11), by reducing it to a SOCP in a similar manner. In fact, this would have resulted in just $O(1)$ conic constraints, so an IPM would converge to a solution in $\widetilde{O}(1)$ iterations, which is comparable with the result from [RML14].

2.5.2 Experimental results

We next present some experimental results to assess the running time parameters and the performance of our algorithm for random instances of SVM. If an algorithm demonstrates a speedup on unstructured instances like these, it is reasonable to extrapolate that the speedup is generic, as it could not have used any special properties of the instance to derive an advantage. For a given dimension n and number of training points m, we denote our distribution of random SVMs with $\mathcal{SVM}(n, m, p)$, where p denotes the probability that a datapoint is misclassified by the optimal separating hyperplane. Additionally, for every training set sampled from $\mathcal{SVM}(n, m, p)$, a corresponding test set of size $\lfloor m/3 \rfloor$ was also sampled from the same distribution. These test sets are used to evaluate the generalization error of SVMs trained in various ways.

Our experiments consist of generating roughly 16000 instances of $\mathcal{SVM}(n, 2n, p)$, where n is chosen to be uniform between 2^2 and 2^9 and p is chosen uniformly from the discrete set $\{0, 0.1, \ldots, 0.9, 1\}$. The instances are then solved using a simulation of Algorithm 1 (with the target duality gap of $\epsilon = 0.1$) as well as using the ECOS SOCP solver [DCB13] (with the default target duality gap). We simulate the execution of Algorithm 1 by implementing the classical IPM and adding noise to the solution of the Newton system (2.7). The noise added to each coordinate is uniform, from an interval selected so that the noisy increment $(\Delta\mathbf{x}, \Delta\mathbf{y}, \Delta\mathbf{s})$ simulates the outputs of the tomography algorithm with precision determined by Theorem 2.10. The SVM parameter C is set to be equal to 1 in all experiments. Additionally, a separate, smaller experiment with roughly 1000 instances following the same distribution is performed for comparing Algorithm 1 with LIBSVM [CL11] using a linear kernel.

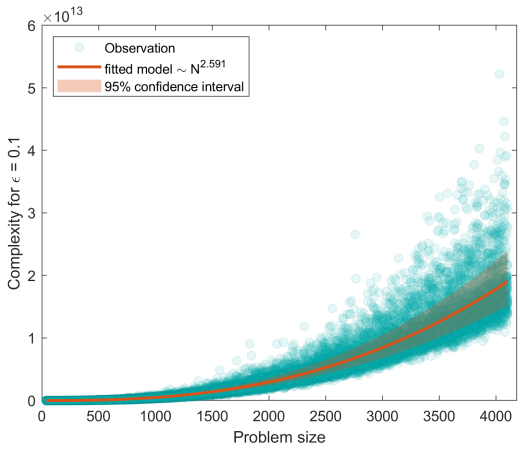


Figure 2.1: Observed complexity of Algorithm 1, its power law fit, and 95% confidence interval.

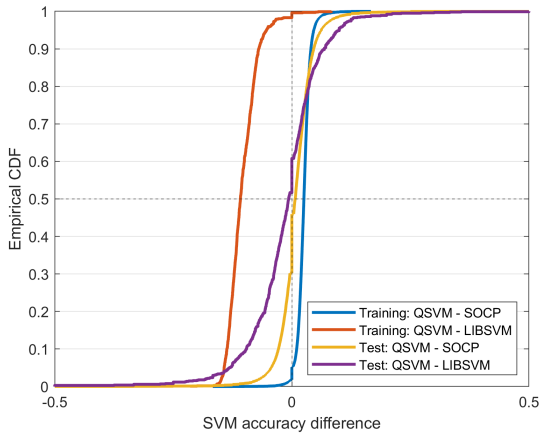


Figure 2.2: Empirical CDF of the difference in accuracy between SVMs trained in different ways.

The experiments are performed on a Dell Precision 7820T workstation with two Intel Xeon Silver 4110 CPUs and 64GB of RAM, and experiment logs are available at [KPS21b]. By finding the least-squares fit of the power law $y = ax^b$ through the observed values of the quantity $\frac{n^{1.5}\kappa\zeta}{\delta^2}$, we obtain the exponent $b = 2.591$, and its 95% confidence interval $[2.564, 2.619]$ (this interval is computed in the standard way using Student's t-distribution, as described in [Net+96]). These observations, the power law, and its confidence interval are show on Figure 2.1. Thus, we can say that for random $\mathcal{SVM}(n, 2n, p)$-instances, and fixed $\epsilon = 0.1$, the complexity of Algorithm 1 scales as $O(n^{2.591})$. This represents a polynomial improvement over general dense SOCP solvers with complexity $O(n^{\omega+0.5})$. In practice, the polynomial speedup is conserved when compared to ECOS [DCB13], that has a measured running time scaling of $O(n^{3.314})$, with a 95% confidence interval for the exponent of [3.297, 3.330] (this is consistent with the internal usage of a [Str69]-like matrix multiplication algorithm, with complexity $O(n^{2.807})$). Neglecting constant factors, this gives us a speedup of 10^4 for $n = 10^6$. The results from the LIBSVM solver indicate that the training time with a linear kernel has a complexity of $O(n^{3.112})$, with a 95% confidence interval for the exponent of [2.799, 3.425]. These results suggest Algorithm 1 retains its advantage even when compared to state-of-the-art specialized classical algorithms.

Additionally, we use the gathered data to verify that the accuracy of our quantum (or approximate) SVM is close to the optimum: Figure 2.2 shows that both at train- and at test-time the accuracies of all three classifiers are most of the time within a few percent of each other, with Algorithm 1 often outperforming the exact SOCP SVM classifier.

In conclusion, the performed numerical experiments indicate that Algorithm 1 provides a polynomial speedup for solving SOCPs with low- and medium precision requirements. In particular, for SVM, we achieve a polynomial speedup with no detriment to the quality of the trained classifier.

2.6 Quantum portfolio optimization

The theory of portfolio optimization in mathematical finance was developed by Markowitz [Mar52]. The theory describes how wealth can be optimally invested in assets which differ in expected return and risk.

The input for the portfolio optimization problem is data about the historical prices of certain financial assets, the goal is to assemble the optimal portfolio that maximizes the expected return for a given level of risk. Let us assume that there are m assets and we have data for historical returns on investment for T time epochs for each asset. Let $R(t) \in \mathbb{R}^m$ be the vector of returns for all assets for time epoch t. The expected reward and risk for the assets can be estimated from the data as follows,

$$\begin{aligned} \boldsymbol{\mu} &= \frac{1}{T} \sum_{t \in [T]} R(t) \\ \Sigma &= \frac{1}{T-1} \sum_{t \in [T]} (R(t) - \boldsymbol{\mu})(R(t) - \boldsymbol{\mu})^\top \end{aligned} \tag{2.13}$$

A portfolio is specified by the total investment x_j in asset j for all assets $j \in [m]$. The expected reward and risk for the portfolio x are respectively given by $\boldsymbol{\mu}^\top \mathbf{x}$ and $\mathbf{x}^\top \Sigma \mathbf{x}$.

The portfolio optimization can include positivity and budget constraints. Positivity constraints $x_j > 0$ corresponds to the situation where it is not possible to short sell asset j. Budget constraints limit the amount of money invested in a subset of the assets and can depend on the value of the assets and also on the size of the initial investment. Similarly, one can add constraints to ensure that the portfolio is diversified by adding constraints on the amount of investment in certain asset classes. The constrained portfolio optimization problem can therefore capture a variety of complex constraints that arise in real world portfolio selection problems.

In the quantum setting, we do not estimate the covariance matrix as in equation (2.13), we instead store the square root of the covariance Σ to which we have direct access from the data, i.e. the covariance matrix $\Sigma = MM^\top$ where M is the matrix with columns $\frac{1}{\sqrt{T-1}}(R(t) - \boldsymbol{\mu})$. Given the data it is easy to construct quantum data structures for operating with M [KP17], hence we formulate the portfolio optimization problem in terms of M.

The constrained portfolio optimization problem can be written as an optimization problem in one of several equivalent ways [CPT18]. We use the following formulation here:

$$\begin{aligned} \min \quad & \mathbf{x}^\top M^\top M \mathbf{x} \\ \text{s.t.} \quad & \boldsymbol{\mu}^\top \mathbf{x} = R \\ & A\mathbf{x} = \mathbf{b} \\ & \mathbf{x} \geq 0. \end{aligned} \tag{2.14}$$

Note that for the above formulation, $\mathbf{x}$ is the portfolio, $\boldsymbol{\mu}$ is the vector of mean asset prices, M is a matrix such that the covariance matrix of the asset prices is defined as $\Sigma = M^\top M$, and $A\mathbf{x} = \mathbf{b}$ and $\mathbf{x} \geq 0$ are the constraints. An inequality constraint $C\mathbf{x} \geq \mathbf{d}$ can be realized by introducing a slack variable $\mathbf{s} := C\mathbf{x} - \mathbf{d}$ and requiring $\mathbf{s} \geq 0$.

For the case when there are no inequality constraints, the problem becomes a linear least-squares problem, for which there is a closed-form solution [RL18]. However, there is no closed form solution for the general constrained portfolio optimization problem.

2.6.1 Reducing portfolio optimization to SOCP

The constrained portfolio optimization problem can be reduced to an SOCP by using the matrix M we defined earlier. First off, we see that $\mathbf{x}^\top \Sigma \mathbf{x} = \|M\mathbf{x}\|^2$, however, minimizing the squared norm is equivalent to minimizing the norm itself which turns out to be more naturally expressible using Lorentz constraints. We introduce an additional vector $\mathbf{t} := (t_0, \tilde{\mathbf{t}})$ such that $\tilde{\mathbf{t}} = M\mathbf{x}$. Using this

variable the portfolio optimization problem in equation (2.13) is easily seen to be equivalent to the following SOCP in the canonical form,

$$\begin{aligned} \min \quad & (1; 0^{n+m})^T (\mathbf{t}; \mathbf{x}) \\ \text{s.t.} \quad & \begin{bmatrix} 0^m & -I_m & M \\ 0 & (0^m)^\top & \boldsymbol{\mu}^\top \\ 0 & (0^m)^\top & A \end{bmatrix} \begin{bmatrix} \mathbf{t} \\ \mathbf{x} \end{bmatrix} = \begin{bmatrix} 0^m \\ R \\ \mathbf{b} \end{bmatrix} \\ & \mathbf{t} \in \mathcal{L}^{m+1}, x_i \in \mathcal{L}^1 \; \forall \; i \in [n], \end{aligned} \tag{2.15}$$

The optimal solution lies on the boundary of the cones and thus will have $t_0 = \left\|\tilde{\mathbf{t}}\right\| = \|P\mathbf{x}\|$, the SOCP objective function therefore also optimizes the objective function for the portfolio optimization problem (2.13). The remaining SOCP constraints respectively enforce the constraints $\tilde{\mathbf{t}} = M\mathbf{x}$, $\mu^\top \mathbf{x} = R$ and $A\mathbf{x} = \mathbf{b}$. The positivity constraints $\mathbf{x} \geq 0$ are ensured by the second order constraints $x_i \in \mathcal{L}^1$.

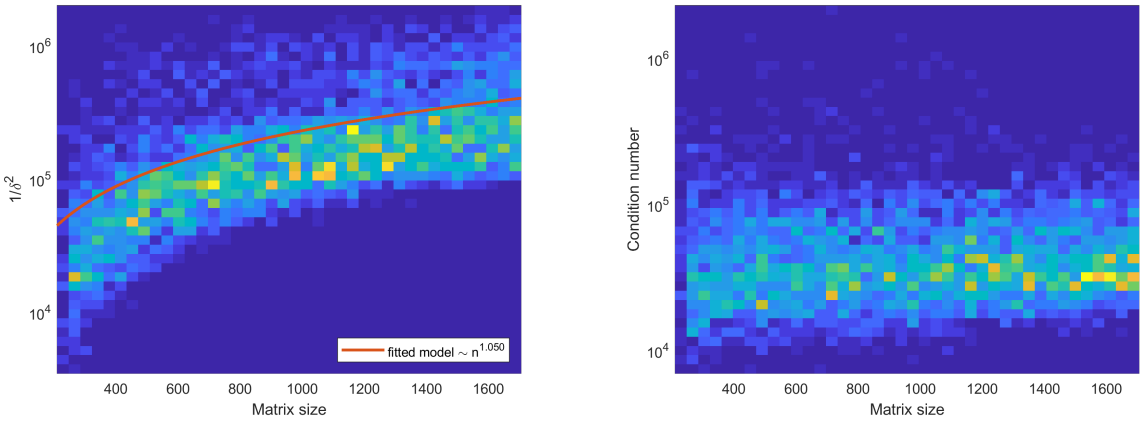
2.6.2 Experimental results

In the experiments, we solve the following portfolio problem

$$\begin{aligned} \min \quad & \mathbf{x}^T \Sigma \mathbf{x} \\ \text{s.t.} \quad & \boldsymbol{\mu}^T \mathbf{x} = R \\ & x \geq 0, \end{aligned} \tag{2.16}$$

i.e. just the problem (2.14) without the "complicated" linear constraints. We use the cvxportfolio dataset [Boy+17], that contains historical data about the stocks of the S&P-500 companies for each day over a period of 9 years (2007-2016). By subsampling the dataset for different numbers of companies and time periods, we obtain random (but structured) instances of different sizes. We perform the simulation as described in Section 2.5.2.

By repeatedly solving instances of varying sizes to a fixed precision $\varepsilon = 0.1$, we can understand the empirical distributions of the runtime parameters δ and κ (see Figure 2.3).



(a) Squared inverse tomography precision $1/\delta^2$ for instances of (2.16) of different sizes.

(b) Condition number κ for instances of (2.16) of different sizes.

Figure 2.3: Dependence of runtime parameters on the portfolio instance size

Finally, we estimate the average-case complexity of Algorithm 1 by substituting these observed values of $r, n, \epsilon, \delta, \kappa$ and ζ into the expression from Theorem 2.18. Figure 2.4 shows how this quantity

increases with the problem size n, after removing 1% of the biggest outliers. Again, by finding the least-squares fit of a power law through these points, one obtains a dependence of $O(n^{2.387})$, with a 95% confidence bound of $[2.184, 2.589]$. When compared to the classical complexity that scales as $O(n^{\omega+0.5})$ (which can be thought of as $n^{3.5}$ in practice), we see that for most instances the quantum algorithm achieves a speedup by a factor of almost $O(n)$.

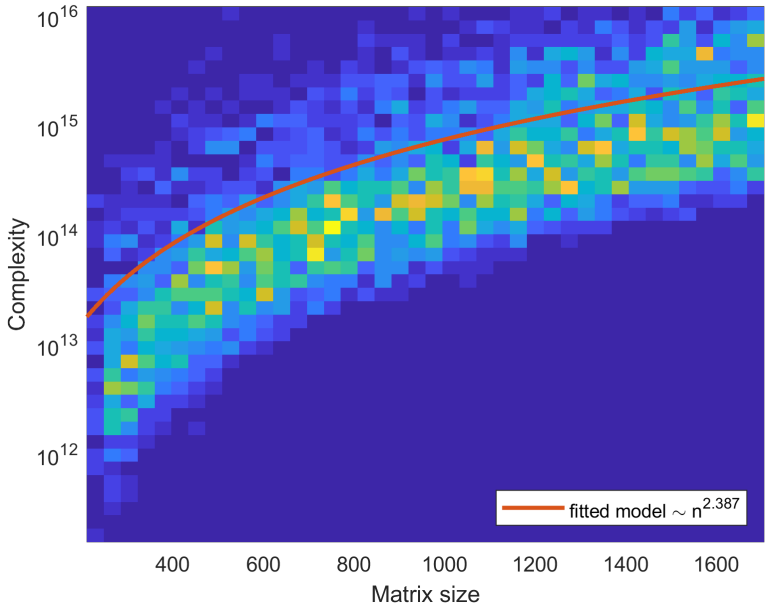


Figure 2.4: Observed complexities for $\epsilon = 0.1$ and the corresponding power-law fit. The x-axis is n, the size of the Newton system, and the y-axis is the observed complexity, as per Theorem 2.18.

3 | Optimal quantum linear system solvers

Joint work with Sander Gribling and Iordanis Kerenidis

3.1 Introduction

The quantum linear systems (QLS) problem asks for a state that encodes the solution of a linear system $A\mathbf{x} = \mathbf{b}$ where $A \in \mathbb{R}^{n\times n}$ and $\mathbf{b} \in \mathbb{R}^n$.[1] Solving linear systems of equations appears as a subproblem in many downstream applications in optimization and in machine learning. Some examples include least-squares regression [CGJ19], support-vector machines [RML14; KPS21a], as well as differential equations [LMS20; Ton+21]. Thus, optimizing the resources (depth in particular) required for solving QLS would bring us closer to running these algorithms on near-term quantum hardware.

In a seminal work, Harrow, Hassidim, and Lloyd [HHL09] showed how to solve the QLS-problem using only polylog(n) queries to the input. Their algorithm has a polynomial dependence on the condition number κ of A and the desired precision $\varepsilon > 0$. Subsequent work has improved the κ-dependence to near linear [Amb12],[2] and the error-dependence to polylog$(1/\varepsilon)$ [CKS17]. The algorithms in [HHL09; Amb12; CKS17] can all be viewed as implementing a polynomial transformation of A that approximates the inverse. They are based on various combinations of Hamiltonian simulation, quantum walks, linear combinations of unitaries, and most recently the quantum singular value transformation framework (QSVT) [LC19; Gil+19].

Very recently, the QLS-problem has been studied using an adiabatic approach [SSO19; AL22; LT20; Cos+21]. Interestingly, the state-of-the-art QLS-algorithm now comes from the adiabatic framework: in [Cos+21] the complexity of QLS is improved from $O(\kappa \log(\kappa/\varepsilon))$ (achieved using polynomial-based techniques) to $O(\kappa \log(1/\varepsilon))$. Their algorithm prepares a low-precision estimate of the $|A^{-1}\mathbf{b}\rangle$ using a gate-based implementation of the discrete adiabatic evolution and improves its quality using quantum eigenstate filtering.

In this work, we revisit the polynomial-based QLS solvers and show that the optimal approximation polynomial (arising from *Chebyshev iteration* [Var00; Pol87]) can be efficiently evaluated using a quantum algorithm. In particular, we recall the two algorithms for evaluating polynomials bounded by 1 (in absolute value) on the interval $[-1, 1]$: the linear combination of unitaries (LCU) lemma [BCK15], and the QSVT framework [Gil+19]. The latter is asymptotically optimal (see [Gil+19, Theorem 73]), but requires the computation of certain angles (see Section 3.3.1 for details), and doing so efficiently in a numerically stable way is the subject of ongoing research [Haa19; Cha+20; Don+21]. On the other hand, the LCU approach is simpler, but trades that simplicity for a logarithmic overhead (multiplicative in the depth and additive in the number of qubits), as well as a slowdown directly proportional to the 1-norm of the polynomial coefficients in the Chebyshev basis.

[1] Without loss of generality, one may assume that A is Hermitian.

[2] Here by a near linear runtime in terms of κ we mean a runtime that scales as $\kappa \,\mathrm{polylog}(\kappa)$.

Therefore, the motivating question is to determine which polynomials can be efficiently evaluated using LCU (at the cost of only a log-overhead).

The *asymptotically* best polynomial-based QLS-algorithm uses a polynomial introduced by Childs, Kothari, and Somma [CKS17] (abbreviated as CKS from now on), which yields a natural LCU-based algorithm. In a nutshell, the CKS polynomial is obtained by starting from the polynomial $p_t(x) := \frac{1-(1-x^2)^t}{x}$ for $t = \widetilde{O}(\kappa^2)$, expressing it in the Chebyshev basis, and truncating the sum after $\widetilde{O}(\kappa)$ terms. Since the Chebyshev coefficients can be interpreted as probabilities of a certain binomial distribution, their 1-norm is easily bounded by a constant.

Our main technical contribution is to show that the Chebyshev iteration polynomial also has a bounded Chebyshev coefficient 1-norm (see Theorem 3.17), and can thus be evaluated using LCU. Additionally, the same norm bound implies that the polynomial can be efficiently evaluated using QSVT, yielding an *optimal* (as opposed to *asymptotically* optimal) QLS algorithm.

In more detail, our approach is as follows. First recall that the Chebyshev iteration corresponds to the polynomial

$$q_t(x) := \frac{\mathcal{T}_t\left(\frac{1+1/\kappa^2-2x^2}{1-1/\kappa^2}\right) \Big/ \mathcal{T}_t\left(\frac{1+1/\kappa^2}{1-1/\kappa^2}\right)}{x},$$

where $\mathcal{T}_t$ is the t-th Chebyshev polynomial of the first kind. These Chebyshev polynomials are defined as $\mathcal{T}_0(x) = 1$, $\mathcal{T}_1(x) = x$, and $\mathcal{T}_{t+1}(x) = 2x\mathcal{T}_t(x) - \mathcal{T}_{t-1}(x)$ for $t \geq 1$. They have the property that $|\mathcal{T}_t(x)| \leq 1$ for all $x \in [-1, 1]$ and $t \geq 0$. One can show that the polynomial q_t is an ε-approximation of the inverse on the domain $x \in [-1, -1/\kappa] \cup [1/\kappa, 1]$, whenever $t \geq \frac{1}{2}\kappa \log(2\kappa^2/\varepsilon)$. To bound the maximum absolute value of q_t on $[-1, 1]$, we express $q_t(x)$ as $\sum_{i=0}^{t-1} c_i \mathcal{T}_{2i+1}(x)$ and bound the 1-norm of the vector $\mathbf{c}$. The vector of coefficients can be used to implement $q_t(A)/\|\mathbf{c}\|_1$ either directly via the linear combinations of unitaries approach, or via the quantum singular value transformation approach.

In Section 3.7 we show that this approach of bounding the 1-norm of the vector of coefficients in the Chebyshev basis more generally leads to near optimal quantum algorithms via the LCU framework for a variety of continuous functions (powers of monomials, exponentials, logarithms) and discontinuous functions (the error function and by extension the sign and rectangle functions). For these functions, the coefficient norm is only a logarithmic factor away from the maximum absolute value on the interval $[-1, 1]$, meaning that they can be approximately evaluated with LCU in addition to QSVT, with slightly deeper circuits (multiplicative logarithmic overhead) and slightly more qubits (additive logarithmic overhead).

The state of the art quantum linear systems solvers have a complexity that grows linearly in the condition number κ. In the small-κ regime ($\kappa = O(n)$), it has long been known that $\Omega(\kappa)$ queries to the entries of the matrix are also needed for general linear systems [HHL09] and recently this bound has (surprisingly) been extended to the case of positive definite systems [OD21]. For larger κ less is known. For example, we do not know if quantum algorithms can improve classical algorithms if κ is large (i.e., can we beat matrix multiplication time?). We do not even have a linear lower bound: are $\Omega(n^2)$ queries needed when $\kappa = \Omega(n^2)$? In [DT09] this question was answered positively when one wants to obtain a *classical description* of $A^{-1}\mathbf{b}$ and here we present a simplified proof of this result.

Organization. In Section 3.2 we recall the different approximation polynomials used for approximately solving linear systems. In Section 3.3 we recall the LCU and QSVT algorithms for evaluating matrix polynomials on a quantum computer. The main technical result of the work is contained in Section 3.4, where we show that the Chebyshev iteration polynomial can be efficiently evaluated using LCU and QSVT. We present some numerical evidence for the improved efficiency in

Section 3.5. Finally, in Section 3.6, we give an overview of known lower bounds on the complexity of quantum linear system solvers both in the small κ regime and in the large κ regime.

3.2 Preliminaries

3.2.1 Polynomials and approximations

Problem definition. We consider linear systems that are defined by a Hermitian n-by-n matrix $A \in \mathbb{C}^{n\times n}$ and a unit vector $\mathbf{b} \in \mathbb{C}^n$. We use κ to denote the condition number of A, that is, we assume that all non-zero eigenvalues of A lie in the set $D_\kappa := [-1, -1/\kappa] \cup [1/\kappa, 1]$. Our goal is to *approximately* solve the linear system

$$A\mathbf{x} = \mathbf{b}.$$

One can consider different notions of approximate solutions. Two natural ones are the following:

1) return $\tilde{\mathbf{x}}$ such that $\|\tilde{\mathbf{x}} - A^{-1}\mathbf{b}\| \le \varepsilon$.

2) return $\tilde{\mathbf{x}}$ such that $\|A\tilde{\mathbf{x}} - \mathbf{b}\| \le \varepsilon$.

Up to a change in ε, the two notions are equivalent. Indeed, we have the chain of inequalities

$$\|A\mathbf{x} - \mathbf{b}\| \le \|\mathbf{x} - A^{-1}\mathbf{b}\| \le \kappa\|A\mathbf{x} - \mathbf{b}\|. \tag{3.1}$$

We will focus on algorithms that achieve a polylogarithmic dependence in ε. In Lemma 3.10 we construct the optimal degree-t polynomial for approximation in the second notion, see Definition 3.13. Prior work [CKS17; CGJ19; Gil+19] focused on the first notion of approximation, which is equivalent up to polylogarithmic factors in the complexity. In Section 3.5 we show (numerically) that our polynomials also improve over prior work with respect to approximation in the first notion.

From matrices to scalars. Given a polynomial $p(x) = \sum_{t=1}^{T} c_t x^t$ with coefficients $c_t \in \mathbb{C}$, and a Hermitian matrix A, we define $p(A) = \sum_{t=1}^{T} c_t A^t$. If we let $A = \sum_{i=1}^{n} \lambda_i \mathbf{u}_i \mathbf{u}_i^*$ be the eigendecomposition of A, then $p(A) = \sum_{i=1}^{n} p(\lambda_i)\mathbf{u}_i\mathbf{u}_i^*$.

We focus on methods to obtain a vector $\tilde{\mathbf{x}}$ that approximates $A^{-1}\mathbf{b}$ that are based on polynomials that approximate the inverse function $\lambda \mapsto \lambda^{-1}$ on the domain $[1/\kappa, 1]$ (in the case of positive definite matrices) or D_κ (in the general case). For example, let A be a Hermitian matrix with eigenvalues in $[1/\kappa, 1]$ and let $p : \mathbb{R} \to \mathbb{R}$ be a polynomial such that $|p(\lambda) - \lambda^{-1}| \le \varepsilon$ for $\lambda \in [1/\kappa, 1]$. Then $\tilde{\mathbf{x}} := p(A)\mathbf{b}$ satisfies

$$\|\tilde{\mathbf{x}} - A^{-1}\mathbf{b}\| = \|\sum_i (p(\lambda_i) - \lambda_i^{-1})\mathbf{u}_i\mathbf{u}_i^*\mathbf{b}\| \le \|\sum_i (p(\lambda_i) - \lambda_i^{-1})\mathbf{u}_i\mathbf{u}_i^*\|\|\mathbf{b}\| \le \varepsilon\|\mathbf{b}\|.$$

Chebyshev decomposition. It is also useful to consider the *Chebyshev decomposition* of $p(x)$, i.e., the decomposition

$$p(x) = \sum_{i=0}^{t} c_i \mathcal{T}_i(x)$$

in the basis $\{\mathcal{T}_0(x), \mathcal{T}_1(x), \dots, \mathcal{T}_t(x)\}$, for some vector $\mathbf{c} = (c_i)_{i\in\{0,\dots,t\}}$ of coefficients. One can give an analytic expression for the coefficients c_i using the fact that the Chebyshev polynomials are orthogonal with respect to the *Chebyshev measure* which is defined in terms of the Lebesgue measure as $\mathrm{d}\mu(x) = (1-x^2)^{-1/2}\,\mathrm{d}x$. In other words, $c_i = \int_{-1}^{1} \frac{p(x)\mathcal{T}_i(x)}{\sqrt{1-x^2}}\,\mathrm{d}x$. Note that in practice this integral is rarely computed explicitly, as there exist efficient interpolation-based methods for computing the coefficient-vector $\mathbf{c}$ [Gen72].

3.2.2 Approximating the inverse on $[1/\kappa, 1]$

As mentioned above, approximating the solution of a linear system $A\mathbf{x} = \mathbf{b}$ amounts to approximating the function $1/x$ on a domain that contains the spectrum of A. We start by assuming that A is positive-definite, i.e. that all of its eigenvalues lie in the interval $[1/\kappa, 1]$. A "natural" polynomial we can consider is the degree-$(t-1)$ Taylor expansion of $1/x$ around the point $x = 1$:

$$p_t^+(x) := \sum_{k=0}^{t-1} (1-x)^k = \frac{1-(1-x)^t}{x}.$$

Its error on the interval $[1/\kappa, 1]$ is straightforward to analyze.

Lemma 3.1. *We have $|xp_t^+(x) - 1| \leq \varepsilon$ for all $x \in [1/\kappa, 1]$ whenever $t \geq \kappa \log(1/\varepsilon)$.*

Proof. For all $x \in [1/\kappa, 1]$ we have

$$|xp_t^+(x) - 1| = |1-x|^t \leq (1 - 1/\kappa)^t \leq e^{-\log(1/\varepsilon)} = \varepsilon. \qquad \square$$

Using the same reasoning as in Equation (3.1) we can analyze the error in the regime 1).

Corollary 3.2. *We have $|p_t^+(x) - 1/x| \leq \varepsilon$ for all $x \in [1/\kappa, 1]$ whenever $t \geq \kappa \log(\kappa/\varepsilon)$.*

Interestingly, it turns out (see e.g. [Pol87]) that this is exactly the polynomial that arises from the (standard, unaccelerated) gradient descent for minimizing the quadratic $\frac{1}{2}\mathbf{x}^\top A\mathbf{x} - \mathbf{b}^\top \mathbf{x}$ for a positive-definite matrix A, starting from the point $\mathbf{x}_1 = \mathbf{b}$. Given the existence of *accelerated* gradient descent methods that converge in $\widetilde{O}(\sqrt{\kappa})$ iterations, it is reasonable to expect that a corresponding polynomial with degree $\widetilde{O}(\sqrt{\kappa})$ exists as well.

Indeed, instead of p_t^+, one can ask what is the *best* degree-$(t-1)$ polynomial q_t^+? In other words, what is the degree-$(t-1)$ polynomial q_t^+ that minimizes

$$\max_{x \in [1/\kappa, 1]} |xq_t^+(x) - 1|. \tag{3.2}$$

First observe that all such polynomials can be expressed in the form $q_t^+(x) = \frac{1-r_t^+(x)}{x}$ where r_t^+ is a degree-t polynomial that satisfies $r_t^+(0) = 1$.[3] Thus, our goal is to find a degree-t polynomial $r_t^+(x)$ that has the smallest absolute value on the interval $[1/\kappa, 1]$ and satisfies the normalization constraint $r_t^+(0) = 1$. It turns out that we can use extremal properties of the Chebyshev polynomials $\mathcal{T}_t(x)$ to determine an optimal $r_t^+(x)$. We use the following well-known result (cf. [SV14, Prop. 2.4]).

Lemma 3.3. *For any degree-t polynomial $p(x)$ such that $|p(x)| \leq 1$ for all $x \in [-1, 1]$, and any y such that $|y| > 1$, we have $|p(y)| \leq |\mathcal{T}_t(y)|$.*

Using the affine transformation $x \mapsto \frac{1+1/\kappa-2x}{1-1/\kappa}$ this gives the following corollary:

Corollary 3.4. *Let $\kappa > 1$ be real, and let $t > 0$ be an integer. Then, the polynomial*

$$r_t^+(x) = \mathcal{T}_t\left(\frac{1+1/\kappa-2x}{1-1/\kappa}\right) \Big/ \mathcal{T}_t\left(\frac{1+1/\kappa}{1-1/\kappa}\right)$$

[3] For example, in the case of the Taylor expansion we have $r_t^+(x) = (1-x)^t$.

is a degree-t polynomial that satisfies $r_t^+(0) = 1$, and minimizes the quantity $\max_{x\in[1,1/\kappa]} |r_t^+(x)|$.

Note that the polynomials r_t^+ satisfy a Chebyshev-like 3-term recurrence. As a consequence, the polynomials $q_t^+(x) = (1 - r_t^+(x))/x$ also satisfy such a recurrence. The corresponding iterative method is known as the Chebyshev iteration.

Remark 3.5 (Chebyshev iteration)**.** The polynomial $q_t^+(x)$ satisfies the recurrence

$$q_{t+1}^+(x) = 2\frac{\mathcal{T}_t(\gamma)}{\mathcal{T}_{t+1}(\gamma)}\frac{\kappa+1-2\kappa x}{\kappa-1}q_t^+(x) - \frac{\mathcal{T}_{t-1}(\gamma)}{\mathcal{T}_{t+1}(\gamma)}q_{t-1}^+(x) - \frac{4\kappa}{\kappa-1}\frac{\mathcal{T}_t(\gamma)}{\mathcal{T}_{t+1}(\gamma)}, \tag{3.3}$$

where $\gamma = \frac{1+1/\kappa}{1-1/\kappa}$. This recurrence corresponds to the iterative method $\mathbf{x}_1 = \mathbf{b}$ and

$$\mathbf{x}_{t+1} = 2\frac{\mathcal{T}_t(\gamma)}{\mathcal{T}_{t+1}(\gamma)}\frac{(\kappa+1)I-2\kappa A}{\kappa-1}\mathbf{x}_t - \frac{\mathcal{T}_{t-1}(\gamma)}{\mathcal{T}_{t+1}(\gamma)}\mathbf{x}_{t-1} - \frac{4\kappa}{\kappa-1}\frac{\mathcal{T}_t(\gamma)}{\mathcal{T}_{t+1}(\gamma)}\mathbf{b}.$$

The convergence rate of this method is summarized by the following theorem:

Theorem 3.6. *Let $\kappa > 1$ and $\epsilon > 0$. Then, for $t \geq \frac{1}{2}\sqrt{\kappa}\log(2/\epsilon)$ we have*

$$\left|xq_t^+(x) - 1\right| \leq \epsilon \text{ for all } x \in [1/\kappa, 1].$$

Proof. First, we define $s(x) = \frac{1+1/\kappa-2x}{1-1/\kappa}$, so that we have $r_t^+(x) = \mathcal{T}_t(s(x))/\mathcal{T}_t(s(0))$. Thus, for all $x \in [1/\kappa, 1]$, we have

$$|xq_t^+(x) - 1| = |r_t^+(x)| = \left|\mathcal{T}_t(s(x))/\mathcal{T}_t(s(0))\right|.$$

Additionally, since $|s(x)| \leq 1$ on this interval, we also have $|\mathcal{T}_t(s(x))| \leq 1$. Thus, it suffices to find t for which $\mathcal{T}_t(s(0)) = \mathcal{T}_t(1 + \frac{2}{\kappa-1}) \geq \frac{1}{\epsilon}$. Since the Chebyshev polynomial $\mathcal{T}_t(\cdot)$ can be computed as

$$\mathcal{T}_t(x) = \frac{1}{2}\left(\left(x - \sqrt{x^2-1}\right)^t + \left(x + \sqrt{x^2-1}\right)^t\right) \text{ for } |x| \geq 1, \tag{3.4}$$

we can conclude that $\mathcal{T}_t(s(0)) = \frac{1}{2}\left(\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^t + \left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)^t\right) \geq \frac{1}{2}\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)^t$. Using the inequality $(1 + \frac{x}{n})^{n+x/2} \geq e^x$ for $x, n \geq 0$, after substituting $t = \frac{1}{2}\sqrt{\kappa}\log(2/\epsilon)$ we have

$$\begin{aligned}\mathcal{T}_t(s(0)) &\geq \frac{1}{2}\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)^t = \frac{1}{2}\left(1 + \frac{2}{\sqrt{\kappa}-1}\right)^{(\sqrt{\kappa}-1+2/2)\frac{\log(2/\epsilon)}{2}} \\ &\geq \frac{1}{2}\exp(\log(2/\epsilon)) = \frac{1}{\epsilon}.\end{aligned}$$

□

Just like in Corollary 3.2, we can bound the error $|q_t^+(x) - 1/x|$.

Corollary 3.7. *Let $\kappa > 1$ and $\epsilon > 0$. Then, for $t \geq \frac{1}{2}\sqrt{\kappa}\log(2\kappa/\epsilon)$ we have*

$$\left|q_t^+(x) - 1/x\right| \leq \epsilon \text{ for all } x \in [1/\kappa, 1].$$

3.2.3 The general case

We now return to the setting where A is a Hermitian matrix and has eigenvalues in the domain $D_\kappa = [-1, -1/\kappa] \cup [1/\kappa, 1]$. In this case, instead of the (indefinite) system $A\mathbf{x} = \mathbf{b}$, we consider the equivalent positive-definite linear system $A^2\mathbf{x} = A\mathbf{b}$. In other words, we approximate the inverse on D_κ using the positive case and a simple substitution:

Corollary 3.8. *Let $\varepsilon > 0$, $\kappa > 1$, and let P_t be any degree-$(t-1)$ polynomial such that $|yP_t(y) - 1| \leq \varepsilon$ for all $y \in [1/\kappa^2, 1]$. Then, $|x^2 P_t(x^2) - 1| \leq \epsilon$ for all $x \in D_\kappa$.*

We define the following two polynomials as the respective analogs of p_t^+ and q_t^+ for D_κ:

$$p_t(x) = xp_t^+(x^2) = \frac{1-(1-x^2)^t}{x}, \text{ and} \tag{3.5}$$

$$q_t(x) = xq_t^+(x^2) = \frac{1 - \mathcal{T}_t(\frac{1+1/\kappa^2-2x^2}{1-1/\kappa^2})/\mathcal{T}_t(\frac{1+\kappa^2}{1-\kappa^2})}{x}. \tag{3.6}$$

We call q_t the *Chebyshev iteration polynomial*. Both p_t and q_t are degree-$(2t-1)$ polynomials, but different values of t are required in order to achieve an ε-approximation of $1/x$ on D_κ. In particular, the following degrees are required:

Corollary 3.9. *Let $\kappa > 1$ and $\varepsilon > 0$. Then,*

1. *$|p_t(x) - 1/x| \leq \varepsilon$ for all $x \in D_\kappa$ whenever $t \geq \kappa^2 \log(\kappa/\varepsilon)$,*
2. *$|q_t(x) - 1/x| \leq \varepsilon$ for all $x \in D_\kappa$ whenever $t \geq \frac{1}{2}\kappa \log(2\kappa/\varepsilon)$.*

Lemma 3.10. *Let $t \in \mathbb{N}$ and $\kappa > 1$. The polynomial q_t is a degree-$(2t-1)$ polynomial that minimizes the quantity $\max_{x \in D_\kappa} |xP(x) - 1|$ among all degree-$(2t-1)$ polynomials $P \in \mathbb{R}[x]$.*

Proof. For a given t, we define

$$\varepsilon^+ := \min_{\substack{P^+ \in \mathbb{R}[y] \\ \deg P^+ = t-1}} \max_{y \in [1/\kappa^2, 1]} |yP^+(y) - 1|, \qquad \varepsilon := \min_{\substack{P \in \mathbb{R}[x] \\ \deg P = 2t-1}} \max_{x \in D_\kappa} |xP(x) - 1|.$$

We first show that q_t certifies that $\varepsilon \leq \varepsilon^+$, and then we show $\varepsilon = \varepsilon^+$. From Corollary 3.4, we know that ε^+ is achieved by the degree-$t-1$ polynomial $q_t^+(x) := \frac{1-\mathcal{T}_t(s(x))/\mathcal{T}_t(s(0))}{x}$, where $s(x) := \frac{1+1/\kappa^2-2x}{1-1/\kappa^2}$. Then, for $q_t(x) := \frac{1-\mathcal{T}_t(s(x^2))/\mathcal{T}_t(s(0))}{x}$ we have

$$\max_{x \in D_\kappa} |xq_t(x) - 1| = \max_{x \in D_\kappa} |x^2 q_t^+(x^2) - 1| = \max_{y \in [1/\kappa^2, 1]} |yq_t^+(y) - 1| = \varepsilon^+,$$

where in the first equality we use Equation (3.6). We now show that $\varepsilon = \varepsilon^+$. Let $P(x)$ be a degree-$(2t-1)$ polynomial that satisfies $\max_{x \in D_\kappa} |xP(x) - 1| = \varepsilon$. We first show that P is odd. To do this, decompose P as $P(x) = P_{\text{even}}(x) + P_{\text{odd}}(x)$ where P_{even} is even and P_{odd} is odd. Then

$$\begin{aligned}
\max_{x \in D_\kappa} |xP(x) - 1| &= \max_{x \in [1/\kappa, 1]} \max\{|xP(x) - 1|, |-xP(-x) - 1|\} \\
&= \max_{x \in [1/\kappa, 1]} \max\{|xP_{\text{odd}}(x) + xP_{\text{even}}(x) - 1|, |xP_{\text{odd}}(x) - xP_{\text{even}}(x) - 1|\} \\
&\geq \max_{x \in [1/\kappa, 1]} |xP_{\text{odd}}(x) - 1| = \max_{x \in D_\kappa} |xP_{\text{odd}}(x) - 1|.
\end{aligned}$$

Hence replacing P by P_{odd} decreases ε, so we may assume that $P(x)$ is odd. Then $P(x)/x$ is a degree-$(2t-2)$ even polynomial. Let $P^+(y)$ be the degree-$(t-1)$ polynomial for which $P(x)/x = P^+(x^2)$. Then we have

$$\max_{y \in [1/\kappa^2, 1]} |yP^+(y) - 1| = \max_{x \in [1/\kappa, 1]} |x^2 P^+(x^2) - 1| = \max_{x \in D_\kappa} |xP(x) - 1| = \varepsilon$$

This shows that $\varepsilon^+ \leq \varepsilon$ which concludes the proof: q_t is the degree-$(2t-1)$ polynomial that minimizes $\max_{x \in D_\kappa} |xP(x) - 1|$ over polynomials of degree $2t-1$. □

We conclude this section with a short discussion of the approach taken by [CKS17], the previous best polynomial-based QLS algorithm. The key insight of [CKS17] is that one can truncate the higher order terms of p_t in the Chebyshev basis without a significant impact on the approximation error. The polynomial p_t can be written in the Chebyshev basis as follows:

$$p_t(x) = 4\sum_{j=0}^{t-1}(-1)^j \left(\frac{\sum_{i=j+1}^{t} \binom{2t}{t+i}}{2^{2t}} \right) \mathcal{T}_{2j+1}(x). \tag{3.7}$$

This expansion can be truncated at $\tilde{O}(\kappa)$ terms, since the Chebyshev coefficients decay exponentially. This can be shown by relating the absolute value of the j-th coefficient (for $j = 0, 1, \dots$) to the probability of more than $t + j$ heads appearing in $2t$ tosses of a fair coin. This probability decreases as $e^{-j^2/t}$ which can be seen by applying the Chernoff bound. Thus, starting from p_t, an ε-approximation of the inverse, we obtain an ε-approximation of p_t by truncating the summation at $j = \sqrt{t \log(4t/\varepsilon)} = \tilde{O}(\kappa)$. For these parameters, we obtain a 2ε-approximation of the inverse on D_κ. We refer to this polynomial as the *CKS polynomial* and we state its main property below.

Theorem 3.11 ([CKS17])**.** *Let $\kappa > 1$, $\varepsilon > 0$ and let $\tilde{p}_t$ be the truncated degree-$(2t-1)$ CKS polynomial. Then, $|\tilde{p}_t(x) - 1/x| \leq \varepsilon$ for all $x \in D_\kappa$ whenever $t \geq 1 + \sqrt{t' \log(8t'/\varepsilon)}$, where $t' = \kappa^2 \log(2\kappa/\varepsilon)$.*

3.3 Quantum preliminaries

There exist different input models that one might consider when solving the linear system problem. In the standard case of a dense matrix A, one might assume that all entries of A are already stored in memory. Alternatively, if A is sparse, sometimes it is more efficient to consider oracle access to its nonzero entries. In the quantum setting, this *sparse-access* model is particularly amenable to speedups. In the sparse-access model we assume that access to A is provided through two oracles

$$\mathcal{O}_{\text{nz}} : |j, \ell\rangle \mapsto |j, \nu(j, \ell)\rangle \text{ and } \mathcal{O}_A : |j, k, z\rangle \mapsto |j, k, z \oplus A_{jk}\rangle\,,$$

where $\nu(j, \ell)$ is the row index of the ℓth nonzero entry of the jth column. Many quantum algorithms can be phrased naturally in terms of a different input model called the *block-encoding* model [LC19; CGJ19]. (One can efficiently construct a block-encoding, given sparse access.)

Definition 3.12 (Block encoding)**.** Let $A \in \mathbb{R}^{n\times n}$ be a Hermitian matrix, and let $N \in \mathbb{N}$ be such that $n = 2^N$, and let $\mu \geq 1$. The $(N + a)$-qubit operator U_A is a (μ, a)-block-encoding of A if it satisfies $A = \mu(\langle 0|^{\otimes a} \otimes I)U_A(|0\rangle^{\otimes a} \otimes I)$.

For convenience, if we are not interested in the number of ancillary qubits a, we simply call U_A a μ-block-encoding. In what follows, we assume that we have access to U_A, an (exact[4]) $(1, a)$-block-encoding of A. The case of μ-block-encodings with $\mu > 1$ can be reduced to the former by replacing our starting matrix with A/μ, that has eigenvalues in $D_{\mu\kappa}$. Furthermore, we assume that A is invertible, with eigenvalues in D_κ. Finally, we assume that we have access to $U_{\mathbf{b}}$, a unitary that (exactly) prepares the state $|\mathbf{b}\rangle = \mathbf{b}/\|\mathbf{b}\|$ on input $|\mathbf{0}\rangle$: $U_{\mathbf{b}}|\mathbf{0}\rangle = |\mathbf{b}\rangle$.

We define the quantum linear system problem (QLSP) as follows:

[4]Constructing exact block-encodings of arbitrary matrices A that are given in the sparse-access input model is a priori not possible with a finite gate set. Instead, one can construct a block-encoding of an approximation $\tilde{A}$, by allowing an overhead in the circuit depth that is proportional to $\log(\|A - \tilde{A}\|)$.

Definition 3.13 (Quantum linear systems)**.** Let $A \in \mathbb{R}^{n\times n}$ be a Hermitian matrix with eigenvalues in D_κ, let $\mathbf{b} \in \mathbb{R}^n$, and let $\varepsilon > 0$. Given a block-encoding U_A of A and a state preparation oracle $U_\mathbf{b}$, output a state

$$|\phi\rangle = \alpha\,|0\rangle\,|\mathbf{x}\rangle + \beta\,|1\rangle\,|\psi\rangle$$

where $\| \,|A\mathbf{x}\rangle - |b\rangle\,\| \leq \varepsilon$, $|\psi\rangle$ is an arbitrary state, and $\alpha, \beta \in \mathbb{C}$ are such that $|\alpha|^2 + |\beta|^2 = 1$ and $|\alpha|^2 \geq 2/3$.

As mentioned before, the widely-used definition from the the literature [CKS17; CGJ19; Gil+19] is equivalent to Definition 3.13 up to a change in ϵ. In this work we use Definition 3.13, as our algorithm is optimal in this sense. In Section 3.5 we (numerically) show that our algorithm also improves over prior work with respect to the more widely used definition.

Recent approaches for solving the QLS problem are based on applying a block-encoding of $p(A)$ to $|\mathbf{b}\rangle$. In the next two sections we describe two ways of computing a block-encoding of $p(A)$: through the QSVT framework, or by decomposing p in the Chebyshev basis, computing each term individually, and combining the results using the linear combination of unitaries lemma (the LCU approach).

3.3.1 QSVT approach

The most straightforward way for evaluating a polynomial quantumly is through the quantum singular value transformation framework [Gil+19]. Using QSVT, one can directly evaluate any polynomial p as long as its sup-norm is suitably bounded. Here the sup-norm of p is defined as

$$\|p\|_\infty := \max_{x\in[-1,1]} |p(x)|.$$

This is achieved by performing a series of rotations by angles $\mathbf{\Phi} = (\phi_1, \ldots, \phi_t)$ on a single qubit, that induces a degree-t polynomial transformation of the singular values of A. Determining these angles efficiently in a numerically stable way is the subject of ongoing research [Haa19; Cha+20; Don+21]. Below, we state a version of QSVT suitable for evaluating even and odd polynomials, since this is the case we are most interested in.

Theorem 3.14 ([Gil+19, Cor. 18], for block-encodings)**.** *Let $A \in \mathbb{R}^{n\times n}$ be Hermitian, and let U_A be a 1-block-encoding of A. Let $\Pi = (|0\rangle\langle 0|)^{\otimes a} \otimes I$, and suppose that $p \in \mathbb{R}[x]$ is a degree-t polynomial of parity-$(t \bmod 2)$ satisfying $\|p\|_\infty \leq 1$. Then there exists a $\mathbf{\Phi} \in \mathbb{R}^t$ such that*

$$p(A) = (\langle +| \otimes \Pi)\,(|0\rangle\langle 0| \otimes U_{\mathbf{\Phi}} + |1\rangle\langle 1| \otimes U_{-\mathbf{\Phi}})\,(|+\rangle \otimes \Pi)$$

where U_Φ is defined as the phased alternating sequence

$$U_{\mathbf{\Phi}} := \begin{cases} e^{\mathrm{i}\phi_1(2\Pi - I)} U_A \prod_{j=1}^{(t-1)/2} \left(e^{\mathrm{i}\phi_{2j}(2\Pi-I)} U_A^\dagger e^{\mathrm{i}\phi_{2j+1}(2\Pi-I)} U_A \right) & \text{if } n \text{ is odd, and} \\ \prod_{j=1}^{t/2} \left(e^{\mathrm{i}\phi_{2j-1}(2\Pi-I)} U_A^\dagger e^{\mathrm{i}\phi_{2j}(2\Pi-I)} U_A \right) & \text{if } n \text{ is even.} \end{cases}$$

Note that QSVT is fundamentally limited to evaluating polynomials that are bounded by 1 in absolute value on $[-1, 1]$ (since the output is a unitary matrix). Approximations p of x^{-1} on D_κ are inherently not bounded by 1 on the interval $[-1, 1]$: they are around κ for $x = 1/\kappa$. The QSVT framework allows us to evaluate $p(x)/M$ on A where M is an upper bound on $\|p\|_\infty$. This subnormalization reduces the success probability of for example a QSVT-based QLS-solver. It is thus important to obtain polynomial approximations p that moreover permit a good bound M.

3.3.2 LCU approach

An alternative approach is based on the Linear Combinations of Unitaries (LCU) lemma [BCK15]. It uses the fact that Chebyshev polynomials have a particularly nice vector of angles, which permits an efficient implementation of the LCU circuit.

Lemma 3.15 ([Gil+19, Lem. 9])**.** *Let $\mathbf{\Phi} \in \mathbb{R}^t$ be such that $\phi_1 = (1-t)\frac{\pi}{2}$ and $\phi_i = \frac{\pi}{2}$ for $2 \leq i \leq t$. For this choice of $\mathbf{\Phi}$, the polynomial p from Theorem 3.14 is $\mathcal{T}_t$, the t-th Chebyshev polynomial of the first kind.*

Computing a single Chebyshev polynomial. We consider in more detail the above circuit for computing $\mathcal{T}_{2t+1}(A)$ for a matrix A with a 1-block-encoding U_A. Let $\Pi = |0\rangle\langle 0| \otimes I$ be the same projector as in Theorem 3.14 (we drop the exponent $\otimes a$ for convenience, or equivalently, we assume that the block-encoding U_A has a single auxiliary qubit). By Lemma 3.15 the unitary

$$U_{2t+1} = e^{-\pi \mathbf{i} t(2\Pi - I)} U_A \prod_{j=1}^{t} \left(e^{\mathbf{i}\frac{\pi}{2}(2\Pi - I)} U_A^\dagger e^{\mathbf{i}\frac{\pi}{2}(2\Pi - I)} U_A \right)$$

satisfies $(\langle 0| \otimes I) U_{2t+1} (|0\rangle \otimes I) = \mathcal{T}_{2t+1}(A)$. We first simplify the above. Note that $2\Pi - I$ has eigenvalues ± 1 and therefore $e^{-\pi \mathbf{i} t(2\Pi - I)} = (-1)^t I$ and $e^{\mathbf{i}\frac{\pi}{2}(2\Pi - I)} = \mathbf{i}(2\Pi - I)$. This means that

$$\begin{aligned} U_{2t+1} &= (-1)^t U_A \prod_{j=1}^{t} \left(\mathbf{i}(2\Pi - I) U_A^\dagger \mathbf{i}(2\Pi - I) U_A \right) \\ &= (-1)^t (\mathbf{i})^{2t} U_A \prod_{j=1}^{t} \left((2\Pi - I) U_A^\dagger (2\Pi - I) U_A \right) \\ &= U_A \prod_{j=1}^{t} \Big(\underbrace{(2\Pi - I) U_A^\dagger (2\Pi - I) U_A}_{=:W} \Big) = U_A W^t. \end{aligned}$$

In other words, U_{2t+1} can be viewed as t applications of the unitary W, followed by a single application of U_A. The circuit for even Chebyshev polynomials U_{2t} is very similar, and can be obtained from U_{2t+1} by removing the final application of (left multiplication by) U_A – however, since we are ultimately interested in implementing the inverse, an odd function, we do not describe the circuit in more detail.

Computing a linear combination of Chebyshev polynomials. Given the above circuit that computes block-encodings of $\mathcal{T}_{2k+1}(A)$ for $k \geq 0$, the next step is to compute a block-encoding of linear combinations of the form

$$p(A) = \sum_{i=0}^{t-1} c_i \mathcal{T}_{2i+1}(A). \tag{3.8}$$

This can be achieved using a version of the LCU algorithm due to [CKS17]. In particular, the key to an efficient implementation of the linear combination $\sum_{i=0}^{t-1} c_i U_{2i+1}$ is the efficient implementation of the operator $\sum_{i=0}^{t-1} |i\rangle\langle i| \otimes U_{2i+1}$, which we achieve by introducing an $l = (\lceil \log_2 t \rceil + 1)$-qubit *counter* register, and successively applying $W, W^2, W^4, \ldots, W^{2^{l-1}}$ controlled on qubits $0, 1, \ldots, l-1$ of the counter, followed by a single application of U_A at the end. In [CKS17, Thm. 4] this circuit is analyzed for a specific polynomial-approximation of the inverse. The analysis naturally extends to arbitrary polynomials of the form (3.8).

Theorem 3.16 (based on [CKS17]). *Let A be a Hermitian matrix with eigenvalues in D_κ, let U_A be its block-encoding, and let $U_{\sqrt{\mathbf{c}}}$ be a unitary that prepares the state $\frac{1}{\sqrt{\|\mathbf{c}\|_1}}\sum_{i=1}^{n}\sqrt{c_i}\,|i\rangle$. Then, there exists an algorithm that computes a $\|\mathbf{c}\|_1$-block-encoding of $p(A)$ using $t+1$ calls to controlled versions of U_A and $U_A^\dagger$, and a single call to each of $U_{\sqrt{\mathbf{c}}}$ and $U_{\sqrt{\mathbf{c}}}^\dagger$. This circuit uses a logarithmic number of additional qubits, and has a gate complexity of $O(t\,\mathrm{polylog}(nt\kappa/\varepsilon))$.*

Compared to the QSVT approach, for this circuit we only need to compute the Chebyshev coefficients $\mathbf{c}$, as opposed to the vector of angles $\mathbf{\Phi}$ – this comes, however, at the cost of using $O(\log t)$ additional qubits. Moreover, the coefficient 1-norm $\|\mathbf{c}\|_1$ represents an upper bound for $\|p\|_\infty$, since

$$|p(x)| = \left|\sum_{i=0}^{t} c_i \mathcal{T}_i(x)\right| \leq \sum_{i=0}^{t} |c_i| \cdot |\mathcal{T}_i(x)| \leq \|\mathbf{c}\|_1, \quad \text{for } |x| \leq 1. \tag{3.9}$$

A natural question is how tight this bound is for general degree-t polynomials p with $\|p\|_\infty \leq 1$. By norm conversion (Equation (3.14) in particular), the ratio $\|\mathbf{c}\|_1 / \|p\|_\infty$ is provably upper bounded by $O(\sqrt{t})$ but in Section 3.7 we observe that for many "interesting" functions the ratio $\|\mathbf{c}\|_1 / \|p\|_\infty$ is in fact only $O(\log(t))$. A notable exception is the complex exponential $e^{\mathbf{i}\kappa x}$ (and thus $\sin(\kappa x)$ and $\cos(\kappa x)$) for which numerical experiments suggest that it attains the $O(\sqrt{t})$ upper bound. In Section 3.7.3 we show how to overcome this limitation by composing easily-implementable functions.

3.4 A QLS-algorithm based on q_t

As mentioned before, our main result is the following explicit upper bound on the Chebyshev coefficient 1-norm of q_t.

Theorem 3.17 (Main result). *For all $t \in \mathbb{N}$, the Chebyshev coefficient 1-norm of q_t is bounded as $\|\mathbf{c}_t\|_1 \leq 2(1 + \frac{1}{\mathcal{T}_t(s(0))})t$. In particular, for $t \geq \frac{1}{2}\kappa\log(2\kappa^2/\varepsilon)$ we have $\|\mathbf{c}_t\|_1 \leq 2(1+\varepsilon/\kappa^2)t$.*

The proof of this Theorem is split between Section 3.4.1 and Lemma 3.19. As a corollary, we get a QSVT-based QLS algorithm that can be described as applying the polynomial q_t to a 1-block-encoding of the input matrix A. This yields an $O(t)$-block-encoding of $q_t(A)$, which can then be applied to the input state $|\mathbf{b}\rangle$. Formally, we show the following.

Corollary 3.18 (QSVT-based algorithm). *Let A be a Hermitian matrix with eigenvalues in D_κ, let U_A be a 1-block-encoding of A, and let $\varepsilon > 0$. Then, for $t \geq \frac{1}{2}\kappa\log(2\kappa^2/\varepsilon)$, a $2(1+\varepsilon/\kappa^2)t$-block-encoding of $q_t(A)$ can be constructed using $2t-1$ calls to U_A and $U_A^\dagger$.*

Proof. The algorithm consists of applying QSVT (Theorem 3.14) to the polynomial $q_t(x)/\|q_t\|_\infty$. This allows us to construct a $\|q_t\|_\infty$-block-encoding of $q_t(A)$ with the desired complexity. It remains to upper bound $\|q_t\|_\infty$ by $2(1+\varepsilon/\kappa^2)t$. Motivated by Equation (3.9), it suffices to upper bound the 1-norm of the vector $\mathbf{c}$ of coefficients of q_t in the Chebyshev basis (again by $2(1+\varepsilon/\kappa^2)t$) – this is guaranteed by Theorem 3.17. □

The block-encoding of $q_t(A)$ can now be used as a black-box replacement for the block-encoding of the corresponding CKS polynomial evaluated at A. For example, using variable-time amplitude amplification, an $\widetilde{O}(\kappa)$-query (to U_A) complexity QLS algorithm can be derived. We refer the reader to [CKS17; Gil+19; Mar+21] for an overview of these techniques.

As an alternative approach, one could use the fact that $\|\mathbf{c}_t\|_1$ is bounded in order to evaluate q_t via LCU (Theorem 3.16). At the cost of using $O(\log t)$ additional qubits, an LCU-based approach would yield a more "natural" quantum algorithm, that does away with the classical angle computation preprocessing step required by QSVT – computing these angles efficiently in a numerically stable way is the subject of ongoing research [Cha+20; Don+21; Haa19].

3.4.1 Bounding the Chebyshev coefficients

As discussed above, in order to apply (a normalized version of) q_t to a block-encoding of a Hermitian matrix with eigenvalues in D_κ, we need a bound on the sup-norm of q_t on the interval $[-1, 1]$. In order to derive such a bound, we express q_t in the basis of Chebyshev polynomials. Each of the Chebyshev polynomials has sup-norm equal to 1 and therefore a bound on the 1-norm of the coefficient vector provides a bound on the sup-norm of q_t. Recall that since q_t is an odd polynomial, its expansion in the Chebyshev basis only involves the odd-degree Chebyshev polynomials. That is, we can write

$$q_t(x) = \sum_{i=0}^{t-1} c_{t,i} \mathcal{T}_{2i+1}(x) \tag{3.10}$$

for some vector $\mathbf{c}_t = (c_{t,i})_{i\in\{0,\dots,t-1\}}$ of coefficients. One can give an analytic expression for $c_{t,i}$ using the fact that the Chebyshev polynomials are orthogonal with respect to the *Chebyshev measure*. Here we take a different approach and use the following discrete orthogonality relations. Fix a degree $m \in \mathbb{N}$ and let $\{x_1, \dots, x_m\}$ be the roots of $\mathcal{T}_m(x)$. The x_k's are called the *Chebyshev nodes* and they admit an analytic formula:

$$x_k = \cos\left(\frac{(k-\frac{1}{2})\pi}{m}\right) \qquad \text{for } k = 1, \dots, m \tag{3.11}$$

The discrete orthogonality relation that we will use is the following. For $0 \le i, j < m$, we have

$$\sum_{k=1}^{m} \mathcal{T}_i(x_k)\mathcal{T}_j(x_k) = \begin{cases} m & \text{if } i = j = 0, \\ \frac{m}{2} & \text{if } i = j < m, \\ 0 & \text{if } i \neq j. \end{cases} \tag{3.12}$$

Since q_t is a polynomial of degree $2t-1$, we will use the discrete orthogonality conditions corresponding to $m = 2t$ to recover the coefficient of $\mathcal{T}_{2i+1}$ in q_t. We have

$$c_{t,i} = \frac{1}{t} \sum_{k=1}^{2t} q_t(x_k)\mathcal{T}_{2i+1}(x_k) \tag{3.13}$$

for all $i \in \{0, 1, \dots, t-1\}$. We can equivalently write this in matrix form, $\mathbf{c}_t = \frac{1}{t}\boldsymbol{\mathcal{T}}_t \mathbf{q}_t$, where

$$\boldsymbol{\mathcal{T}}_t = \begin{bmatrix} \mathcal{T}_1(x_1) & \mathcal{T}_1(x_2) & \dots & \mathcal{T}_1(x_{2t}) \\ \mathcal{T}_3(x_1) & \mathcal{T}_3(x_2) & \dots & \mathcal{T}_3(x_{2t}) \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{T}_{2t-1}(x_1) & \mathcal{T}_{2t-1}(x_2) & \dots & \mathcal{T}_{2t-1}(x_{2t}) \end{bmatrix} \text{ and } \mathbf{q}_t = \begin{bmatrix} q_t(x_1) \\ q_t(x_2) \\ \vdots \\ q_t(x_{2t}) \end{bmatrix}.$$

Our goal is to show that $\|\mathbf{c}_t\|_1 \le C \cdot t$ for a small constant C. To do so, we first use the Cauchy-Schwarz inequality to obtain

$$\|\mathbf{c}_t\|_1 \le \sqrt{t}\|\mathbf{c}_t\|_2 = \frac{1}{\sqrt{t}}\|\boldsymbol{\mathcal{T}}_t \mathbf{q}_t\|_2 \le \frac{\|\boldsymbol{\mathcal{T}}_t\|}{\sqrt{t}}\|\mathbf{q}_t\|_2 = \|\mathbf{q}_t\|_2 \tag{3.14}$$

where the last equality follows from the discrete orthogonality relations Equation (3.12): we see that $\boldsymbol{\mathcal{T}}_t \boldsymbol{\mathcal{T}}_t^* = tI_t$ and therefore $\|\boldsymbol{\mathcal{T}}_t\| = \sqrt{t}$. Thus, bounding $\|\mathbf{q}_t\|_2$ would conclude the proof of Theorem 3.17.

Lemma 3.19. *We have* $\|\mathbf{q}_t\|_2 \le 2(1+\frac{1}{\mathcal{T}_t(s(0))})t$ *for all* $t \in \mathbb{N}$. *In particular, for* $t \ge \frac{1}{2}\kappa\log(2\kappa^2/\varepsilon)$ *we have* $\|\mathbf{q}_t\|_2 \le 2(1+\varepsilon/\kappa^2)t$.

Proof. We start by bounding $|q_t(x)|$ on $[-1,1]$, and we recall that

$$q_t(x) = \frac{1-\mathcal{T}_t(s(x))/\mathcal{T}_t(s(0))}{x}, \quad \text{where} \quad s(x) = \frac{1+1/\kappa^2-2x^2}{1-1/\kappa^2}.$$

On one hand, when $x \in D_\kappa$ we have $s(x) \in [-1,1]$ and thus $|1-\mathcal{T}_t(s(x))/\mathcal{T}_t(s(0))| \le 1+1/\mathcal{T}_t(s(0))$. On the other hand, when $|x| \le 1/\kappa$ we have $1 \le s(x) \le s(0) = \frac{1+1/\kappa^2}{1-1/\kappa^2}$. Since $\mathcal{T}_t(x)$ is increasing for $x \ge 1$, it follows that $0 \le 1-\mathcal{T}_t(s(x))/\mathcal{T}_t(s(0)) \le 1$ for all $|x| \le 1/\kappa$. Together this shows that

$$|q_t(x)| = \left|\frac{1-\mathcal{T}_t(s(x))/\mathcal{T}_t(s(0))}{x}\right| \le \frac{1+1/\mathcal{T}_t(s(0))}{|x|} \qquad \text{for all } x \in [-1,1] \setminus \{0\}.$$

We now bound the norm of $\mathbf{q}_t$. We have

$$\|\mathbf{q}_t\|^2 = \sum_{k=1}^{2t} q_t(x_k)^2 \le \left(1+\frac{1}{\mathcal{T}_t(s(0))}\right)^2 \sum_{k=1}^{2t}\frac{1}{x_k^2} = \left(1+\frac{1}{\mathcal{T}_t(s(0))}\right)^2 \sum_{k=1}^{2t}\frac{1}{\cos^2\left(\frac{2k-1}{4t}\pi\right)},$$

where we substituted the exact expression for the Chebyshev nodes $x_k = \cos\left(\frac{2k-1}{4t}\pi\right)$. Moreover, we have

$$\cos^2\left(\frac{2(2t-k+1)-1}{4t}\pi\right) = \cos^2\left(\frac{2k-1}{4t}\pi\right) = \frac{1-\cos\left(\frac{2k-1}{2t}\pi\right)}{2} \quad \text{for all} \quad 1 \le k \le t,$$

where the first equality comes from $x_{2t-k+1} = -x_k$. Therefore, we have

$$\|\mathbf{q}_t\|^2 \le 4\left(1+\frac{1}{\mathcal{T}_t(s(0))}\right)^2 \sum_{k=1}^{t}\frac{1}{1-\cos(\frac{2k-1}{2t}\pi)}.$$

We note that the roots of $\mathcal{T}_t(x)$ are exactly $\cos(\frac{2k-1}{2t}\pi)$, for $k \in [t]$. For any polynomial $P(x) = C\prod_{k=1}^t(x-r_k)$ the following identity holds for all x for which $P(x) \ne 0$:

$$\sum_{k=1}^{t}\frac{1}{x-r_k} = \frac{P'(x)}{P(x)}.$$

Applying the above to $P(x) = \mathcal{T}_t(x)$ and $x = 1$ (which is not a root of $\mathcal{T}_t$), we get

$$\sum_{k=1}^{t}\frac{1}{1-\cos(\frac{2k-1}{2t}\pi)} = \frac{\mathcal{T}_t'(1)}{\mathcal{T}_t(1)} = \frac{t\cdot\mathcal{U}_{t-1}(1)}{1} = t^2.$$

This concludes the main part of the proof: we have shown that $\|\mathbf{q}_t\| \le 2(1+\frac{1}{\mathcal{T}_t(s(0))})t$.

Finally, for $t \ge \frac{1}{2}\kappa\log(2\kappa^2/\varepsilon)$, we bound $1/\mathcal{T}_t(s(0))$ as in the proof of Corollary 3.7. Namely, using the same inequalities, we have

$$\mathcal{T}_t(s(0)) \ge \frac{1}{2}\left(\frac{\kappa+1}{\kappa-1}\right)^t \ge \frac{1}{2}\left(1+\frac{2}{\kappa-1}\right)^{\frac{1}{2}\kappa\log(2\kappa^2/\varepsilon)} \ge \frac{\kappa^2}{\varepsilon}. \qquad \square$$

Combining this lemma with Equation (3.14), we derive the same bound for $\|\mathbf{c}_t\|_1$, thus completing the proof of Theorem 3.17.

3.4.2 Efficiently computing the coefficients

In the case of evaluating q_t via LCU, one question of practical relevance is how to compute the coefficients $\mathbf{c}_t$. Naively using the recurrence (3.3) to compute $\mathbf{c}_t$ gives rise to an algorithm with $O(t^2)$ arithmetic operations with real numbers. Alternatively, one can use FFT-based Chebyshev interpolation algorithms that can compute $\mathbf{c}_t$ with $O(t \log t)$ operations given the vector $\mathbf{q}_t$ of the values of $q_t(x)$ at the order-t Chebyshev nodes [Gen72]. Thus, in order to get an $O(t \log t)$-operation algorithm for computing $\mathbf{c}_t$, it suffices to show that $q_t(x)$ can be evaluated at a single Chebyshev node x_k with $O(\log t)$-operations. Given the form of q_t, this means that we need to compute $\mathcal{T}_t(s(x_k))$ with $O(\log t)$ operations. One way to do this is via the degree-halving identities

$$\mathcal{T}_{2t}(x) = 2\mathcal{T}_t(x)^2 - 1 \quad \text{and} \quad \mathcal{T}_{2t+1}(x) = 2\mathcal{T}_{t+1}(x)\mathcal{T}_t(x) - x.$$

3.4.3 A more natural quantum algorithm?

Given the reduction of the general linear system problem to the PD case (Corollary 3.8), one might be tempted to mirror this reduction when designing a quantum algorithm, with the goal of achieving $\widetilde{O}(\sqrt{\kappa})$ complexity for solving PD systems. The input of such an algorithm would be a (block-encoding of a) Hermitian matrix A with eigenvalues in $[1/\kappa, 1]$, and the output would be a block-encoding of $q_t^+(A)$. To evaluate this polynomial using QSVT, we first need to normalize it by dividing it by $\max_{x\in[-1,1]} |q_t^+(x)|$. It turns out that this maximum grows exponentially with t: one can lower bound it by $|q_t^+(-1)|$ and we have

$$\begin{aligned} |q_t^+(-1)| &\geq \frac{\mathcal{T}_t\left(\frac{1+1/\kappa+2}{1-1/\kappa}\right)}{\mathcal{T}_t\left(\frac{1+1/\kappa}{1-1/\kappa}\right)} - 1 \geq \frac{\mathcal{T}_t(3+4/(\kappa-1))}{\mathcal{T}_t(1+2/(\kappa-1))} \\ &\geq \frac{\mathcal{T}_t(3)}{\mathcal{T}_t(2)} \geq \frac{1}{2}\left(\frac{3+2\sqrt{2}}{2+\sqrt{3}}\right)^t \geq \frac{1}{2}\left(\frac{3}{2}\right)^t. \end{aligned}$$

Therefore, amplifying the output of QSVT would take exponential time. In the case of LCU, the coefficient 1-norm is lower bounded by $|q_t^+(-1)|$ (by Equation (3.9)), so the output of a LCU-based algorithm would also need to be amplified exponentially. Alternative approaches of multiplying $q_t^+(x)$ by a rectangle function that is close to 1 on $[1/\kappa, 1]$ and close to 0 elsewhere are similarly fruitless as the degree of the resulting approximation polynomial would become linear in κ. It should be noted, however, that these issues can be avoided if we assume that the mapping $x \mapsto \frac{1+1/\kappa-2x}{1-1/\kappa}$ has already been performed "ahead of time": in [OD21], Orsucci and Dunjko have shown that PD matrices can indeed be inverted in $\widetilde{O}(\sqrt{\kappa})$, provided that a block-encoding of $I - \alpha A$ is given as input (for suitable α).

Another natural alternative approach would be to quantize a method such as momentum gradient descent, which also converges in $\widetilde{O}(\sqrt{\kappa})$ for PD matrices [Pol87]. One way to achieve this would be using the approach of Kerenidis and Prakash [KP20b], who quantized the basic gradient descent algorithm by implementing the recurrence $\mathbf{r}_{t+1} = (I - \eta A)\mathbf{r}_t$ satisfied by the differences $\mathbf{r}_t := \mathbf{x}_t - \mathbf{x}_{t-1}$ of successive iterates. Applying this idea to momentum gradient descent, one gets a recurrence involving two successive differences:

$$\begin{bmatrix} \mathbf{r}_{t+1} \\ \mathbf{r}_t \end{bmatrix} = \underbrace{\begin{bmatrix} (1+\beta)I - \eta A & -\beta I \\ I & 0 \end{bmatrix}}_{M} \begin{bmatrix} \mathbf{r}_t \\ \mathbf{r}_{t-1} \end{bmatrix},$$

for suitable choices of η and β. For example, following [Pol87, Chapter 3], one can set $\eta = 4/(1+\sqrt{1/\kappa})^2$ and $\beta = \left(1-2/(1+\sqrt{\kappa})\right)^2$. Implementing a similar approach as in [KP20b] would require the construction of $O(1)$-block-encodings of powers of M. In particular, this would require M to have a small norm. Unfortunately, for large enough $\kappa \geq 9$ and the above choice of η, β, one has $\|M\| \geq \sqrt{2}$ which means that a block-encoding of M^t needs to have sub-normalization at least $2^{t/2}$.

3.5 Comparison with previous polynomial-based QLS-solvers

In Lemma 3.10 we saw that the Chebyshev iteration polynomial q_t is the degree-$(2t-1)$ polynomial that minimizes the error

$$\max_{x \in D_\kappa} |xP(x) - 1|$$

over all polynomials of degree $2t-1$. This implies that the CKS polynomial attains a larger error than the Chebyshev iteration polynomial, or conversely requires a higher degree to reach the same error on D_κ. In Table 3.1, we use Corollary 3.9 and Theorem 3.11 to compute the degree required to achieve error ε on D_κ, and observe that the degree of the CKS polynomial is roughly twice the degree of the corresponding Chebyshev iteration polynomial.

κ \ ε	0.5	10^{-2}	10^{-4}	10^{-6}
2	15	33	53	71
10	115	203	301	399
100	1819	2687	3669	4633
1000	24913	33515	43337	52989

(a) CKS polynomial

κ \ ε	0.5	10^{-2}	10^{-4}	10^{-6}
2	5	11	21	31
10	37	77	123	169
100	599	991	1451	1911
1000	8295	12207	16811	21417

(b) Chebyshev iteration

Table 3.1: Degrees of approximation polynomials for a given condition condition number κ and error ε, computed according to Corollary 3.9.

Another way of comparing these polynomials is to compute their errors according to

$$\max_{x \in D_\kappa} \left|P(x) - 1/x\right|$$

(corresponding to definition 1)). The optimal polynomial (also called the *minimax* polynomial) according to this definition has no explicit description, but it can still be computed relatively efficiently using the Remez exchange algorithm – see e.g. [PT09] for a practical implementation. In [Don+21] the authors compared the minimax and the CKS polynomials. In Figures 3.1 and 3.2 we add the Chebyshev iteration polynomials to this comparison. In particular, we show the errors $\max_{x \in D_\kappa} \left|P(x) - 1/x\right|$ where P is the minimax, CKS and Chebyshev iteration polynomial. In Figure 3.1 we see that that for a fixed condition number, the convergence is linear for all polynomials, with the CKS polynomial being the slowest to converge (i.e., for the same degree, the difference in errors is a few orders of magnitude). Conversely, in Figure 3.2 we see that with polynomials of a fixed degree, the error of the CKS is an order of magnitude higher, no matter the condition number.

We conclude this section with a remark on the (CPU) time needed to compute (the coefficients of) these polynomials. On the authors' hardware (an Intel Xeon Silver 4110 CPU), computing all the CKS and Chebyshev iteration polynomials from Figures 3.1 and 3.2 only took a few seconds, whereas computing the corresponding minimax polynomials took several hours. One reason for this is that while the former can be computed using simple and efficient machine-precision operations, the latter requires (at least in the implementation of [Don+21]) dealing with arbitrary-precision floating-point numbers.

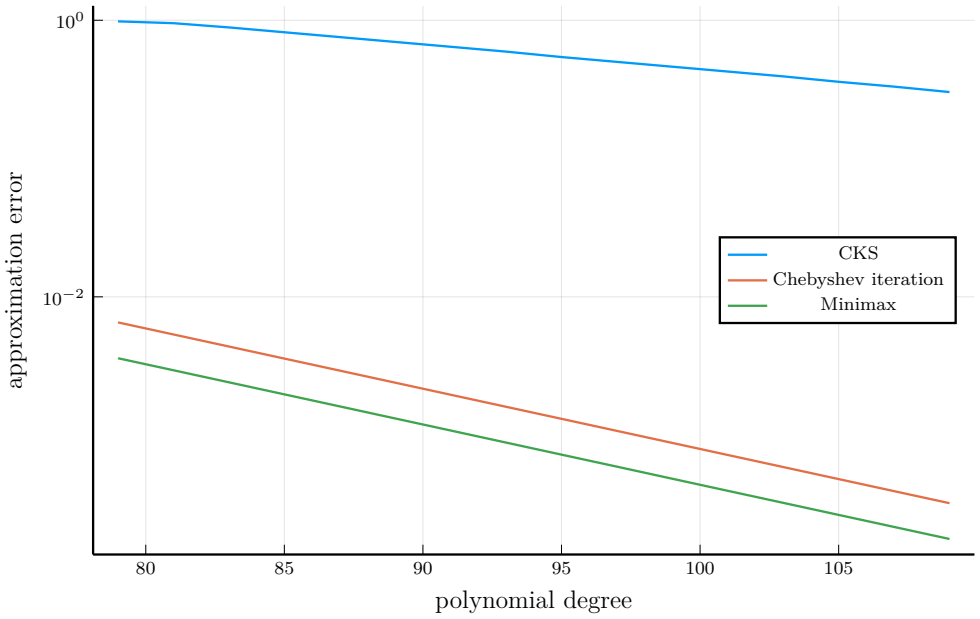


Figure 3.1: Approximation error for a fixed condition number $\kappa = 10$ and varying degrees.

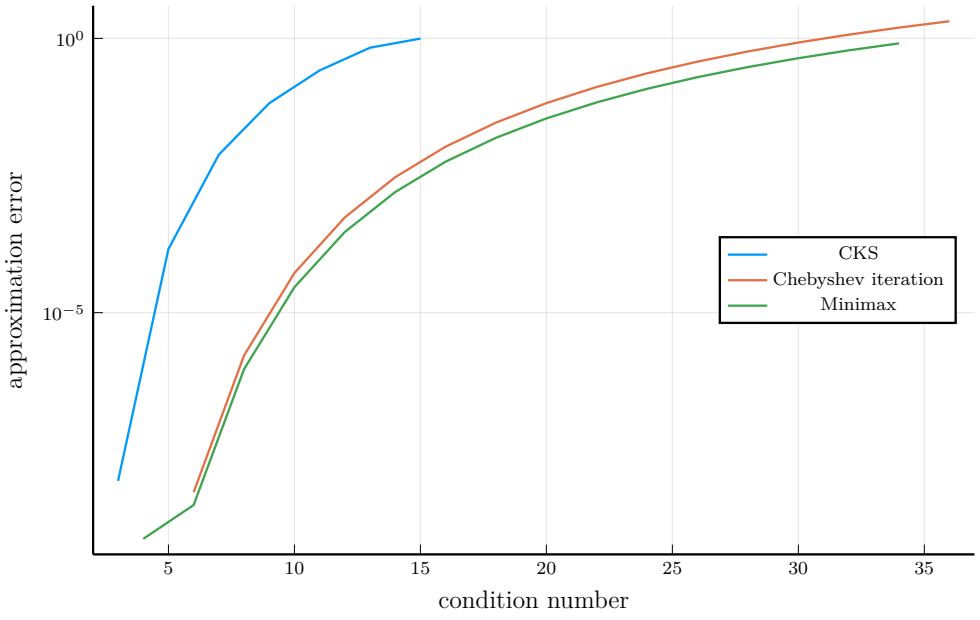


Figure 3.2: Approximation error for a fixed degree $2t - 1 = 127$ and varying condition numbers.

3.6 Query lower bounds

So far, we have been considering algorithms (i.e. upper bounds) for the QLS problem. The complexity of the best algorithm for the QLS problem depends linearly on κ (we ignore the polylogarithmic factors in this section), so a natural question is whether this dependence is optimal. In [HHL09] it has been shown that this is indeed the case: in the sparse access input model (the setting in which such lower bounds are usually proven), the complexity of QLS for general systems is $\Omega(\min(\kappa, n))$. Recently, it has been shown [OD21] that the same $\Omega(\min(\kappa, n))$ lower bound even holds for the

restriction of QLS to PD matrices – this is surprising since in the classical setting a $\sqrt{\kappa}$-separation exists between the general and the PD case. We note that both of these lower bounds apply when the output of the QLS solver is the quantum state $|A^{-1}\mathbf{b}\rangle$. As a consequence, one can show that computing a classical description of $A^{-1}\mathbf{b}$ is just as hard.

Both of the above results apply to the small-κ regime. In particular, they leave open the possibility of a $o(n^\omega)$-time quantum algorithm for solving linear systems (with classical output). The existence of such an algorithm would speed up many classical optimization algorithms (e.g., interior point methods) in a black-box way. In [DT09] it was shown that one cannot obtain a large quantum speedup when the output is required to be classical: $\Omega(n^2)$ quantum queries to the entries of A are needed to obtain a classical description of a single coordinate of $A^{-1}e_n$, where e_n is the n-th standard basis vector in $\mathbb{R}^n$. The statement is robust in the following sense: after normalizing $A^{-1}e_n$, it suffices to obtain a δ-additive approximation of the first coordinate for some $\delta = O(1/n^2)$. We present a simplified proof of this result of [DT09] at the end of this section. Note that this high precision prevents one from lifting the bound to the quantum-output setting: to obtain a δ-additive approximation of a single coordinate of $|A^{-1}b\rangle$ one can use roughly $1/\delta$ rounds of amplitude estimation on a QLS-solver $\mathcal{A}$. With $\delta = O(1/n^2)$ this only implies that $n^2 \cdot \mathrm{cost}(\mathcal{A}) = \Omega(n^2)$. A second type of quantum lower bound is described in [Gil+19]: roughly speaking, if a (smooth) function $f : I \to [-1, 1]$ has a derivative whose absolute value is d, then $\Omega(d)$ uses of a 1-block-encoding U_A of A are needed to create a block-encoding of $f(A)$. Here I is a subset of $[-1, 1]$ that contains the eigenvalues of the Hermitian matrix A. Applied to $f(x) = 1/(\kappa x)$, this shows that indeed $\Omega(\kappa)$ applications of U_A are needed to create a block-encoding of A^{-1}. As mentioned before, a block-encoding of A^{-1} can be combined with a state preparation oracle for $\mathbf{b}$ to solve the QLS problem. Such a strategy however naturally incurs a κ-dependence in the runtime, and it remains an interesting open question whether one could solve the QLS problem (with quantum output!) without such a dependence in κ and in time $o(n^\omega)$.

3.6.1 Lower bound for matrix inversion with classical output

We present a simplified proof of a matrix-inversion lower bound result of [DT09]. It is based on the quantum query complexity of the majority function $\mathrm{MAJ}_n : \{0,1\}^n \to \{0,1\}$ which takes value 1 on input $\mathbf{x}$ if and only if $\sum_{i\in[n]} x_i > n/2$. It is well known that the quantum query complexity of MAJ_n is $\Theta(n)$ [Bea+01].

Lemma 3.20. *Let $X \in \{0,1\}^{n\times n}$. Then, the matrix $A \in \{0,1\}^{(2n+2)\times(2n+2)}$ defined as*

$$A = \left[\begin{array}{cc|cc} 0 & 1_n^* & 0 & 0 \\ 1_n & 0 & X & 0 \\ \hline 0 & X^* & 0 & 1_n \\ 0 & 0 & 1_n^* & 0 \end{array}\right]$$

satisfies $(A^3)_{1,2n+2} = \sum_{i=1}^n \sum_{j=1}^n X_{i,j}$.

Proof. A is the adjacency matrix of an undirected graph that can be described as follows. We start with a bipartite graph between two sets of n vertices whose edge set is described by X, then we add two vertices labeled 1 and $2n + 2$ that we connect respectively to the first set of vertices and the second set of vertices. The entry $(1, 2n + 2)$ of A^3 counts the number of paths of length 3 from 1 to $2n + 2$ in this graph. This equals the number of edges between the sets $\{2, \dots, n + 1\}$ and $\{n + 2, \dots, 2n + 1\}$, that is, $(A^3)_{1,2n+2} = \sum_{i=1}^n \sum_{j=1}^n X_{i,j}$. □

Corollary 3.21. *Let $A \in \{0,1\}^{n\times n}$. Determining a single off-diagonal entry of A^3, with success probability $\geq 2/3$, takes $\Theta(n^2)$ quantum queries to A.*

Lemma 3.22. *Let $A \in \{0,1\}^{n\times n}$. Then, for $N = 4n$, the matrix $B \in \{0,1\}^{N\times N}$ defined by*

$$B = \begin{bmatrix} I & A & & \\ & I & A & \\ & & I & A \\ & & & I \end{bmatrix},$$

satisfies $(B^{-1})_{1,N} = -(A^3)_{1,n}$.

Proof. It is straightforward to verify that the inverse of B is

$$B^{-1} = \begin{bmatrix} I & -A & A^2 & -A^3 \\ & I & -A & A^2 \\ & & I & -A \\ & & & I \end{bmatrix}. \qquad \square$$

If A is the adjacency matrix of the directed version of the graph described in Lemma 3.20, we can also compute the norm of the last column as follows:

$$\left\|B^{-1}\mathbf{e}_{4n}\right\|^2 = \left(\sum_{i,j} X_{i,j}\right)^2 + \sum_i \left(\sum_j X_{i,j}\right)^2 + n + 1.$$

In particular, for the hard instances (where $|n/2 - \sum_{i,j} X_{i,j}| \leq 1$), we have that $\left\|B^{-1}\mathbf{e}_{4n}\right\| = \Theta(n^2)$.

Corollary 3.23. *Let $A \in \{0,1\}^{n\times n}$. Determining a single off-diagonal entry of A^{-1} up to precision $< 1/2$, with success probability $\geq 2/3$, takes $\Theta(n^2)$ quantum queries to A.*

3.7 Examples of functions with bounded Chebyshev coefficient norms

The inverse function is not the only function that can be efficiently evaluated using LCU of Chebyshev polynomials. Here we discuss several families of functions for which the 1-norm of the Chebyshev coefficients is of the order log(degree).

3.7.1 Simple examples

We first observe that the monomial x^n has the following Chebyshev expansion:

$$x^n = 2^{1-n} \sum_{j=0,\, n-j \text{ even}}^{n\,\prime} \binom{n}{\frac{n-j}{2}} \mathcal{T}_j(x),$$

where the prime at the sum symbol indicates that the contribution of $j = 0$ needs to be halved (if it appears). The sum of these coefficients is bounded by 1. This implies that for any polynomial the 1-norm of the coefficients in the Chebyshev basis is at most the 1-norm of the coefficients in

the monomial basis. This means, for example, that the Chebyshev coefficient 1-norm of the scaled exponential is at most 1. Similarly, for a degree n approximation of the (scaled) logarithm the 1-norm grows as $O(\log n)$. In particular, they have the following Taylor expansions for $\kappa \geq 1$

$$e^{\kappa(x-1)} = e^{-\kappa} \sum_{n=0}^{\infty} \frac{(\kappa x)^n}{n!},$$

$$\begin{aligned} \operatorname{slog}_\kappa(x) &:= \log(1/\kappa + ((x+1)/2)(1-1/\kappa)) = \log\left(\frac{\kappa+1}{2\kappa}\left(1 + \frac{\kappa-1}{\kappa+1}x\right)\right) \\ &= \log\left(\frac{\kappa+1}{2\kappa}\right) + \sum_{n=1}^{\infty} \frac{(-1)^{n+1}}{n}\left(\frac{\kappa-1}{\kappa+1}\right)^n x^n. \end{aligned}$$

3.7.2 Approximating discontinuities – the error function

Some more interesting examples are the sign and the rectangle functions, defined as

$$\operatorname{sign}(x) := \begin{cases} 1 & \text{if } x > 0, \\ 0 & \text{if } x = 0, \\ -1 & \text{if } x < 0, \end{cases} \quad \text{and} \quad \Pi(x) := \begin{cases} 1 & \text{if } |x| \leq 1/2, \\ 0 & \text{else.} \end{cases}$$

It is well-known [LC17a; Gil+19] that the error-function $\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-z^2}\, \mathrm{d}z$ is a fundamental building block for approximating discontinuous functions. For example, given $\epsilon, \delta > 0$, there exists a choice of $\kappa = O(\operatorname{polylog}(1/\epsilon)/\delta)$ such that $\operatorname{erf}(\kappa x)$ is ϵ-close to $\operatorname{sign}(x)$ on $[-1,1]$ $[-\delta,\delta]$. We show below that the 1-norm of the coefficients of the Chebyshev series of $\operatorname{erf}(\kappa x)$ is $O(\log \kappa)$. We start with the following expansion from [LC17a]:

$$\operatorname{erf}(\kappa x) = \frac{2\kappa e^{-\kappa^2/2}}{\sqrt{\pi}} \left(I_0(\kappa^2/2)x + \sum_{j=1}^{\infty} I_j(\kappa^2/2)(-1)^j \left(\frac{\mathcal{T}_{2j+1}(x)}{2j+1} - \frac{\mathcal{T}_{2j-1}(x)}{2j-1} \right) \right).$$

By regrouping the terms, we get the following explicit form of the Chebyshev series of $\operatorname{erf}(\kappa x)$:

$$\operatorname{erf}(\kappa x) = \frac{2\kappa e^{-\kappa^2/2}}{\sqrt{\pi}} \sum_{n=0}^{\infty} (-1)^n \frac{I_n(\kappa^2/2) + I_{n+1}(\kappa^2/2)}{2n+1} \mathcal{T}_{2n+1}(x). \tag{3.15}$$

Now, in order to bound the coefficient norm, we use the following inequality from [BP14]:

$$e^{-x} x^{-n} (I_n(x) + I_{n+1}(x)) \leq \sqrt{\frac{2}{\pi}} \left(x + \frac{n}{2} + \frac{1}{4} \right)^{-n-\frac{1}{2}}.$$

Note that $I_n(x) \geq 0$ for $x \geq 0$ and all $n \in \mathbb{N}$. So, the above in fact bounds the absolute value of the left hand side. We use this inequality to bound the (absolute value of the) coefficient of $\mathcal{T}_{2n+1}(x)$ in (3.15) as follows:

$$\frac{2\kappa e^{-\kappa^2/2}}{\sqrt{\pi}} \frac{I_n(\kappa^2/2) + I_{n+1}\kappa^2/2)}{2n+1} \leq \frac{4}{\pi} \frac{1}{2n+1} \left(\frac{\kappa^2}{\kappa^2 + n + 1/2} \right)^{n+1/2}. \tag{3.16}$$

Using this inequality, we can bound the coefficient norm of the truncated Chebyshev series:

Lemma 3.24. *Let $N > 0$ be an integer. Then,*

$$\frac{2\kappa e^{-\kappa^2/2}}{\sqrt{\pi}} \sum_{n=0}^{N} \frac{I_n(\kappa^2/2) + I_{n+1}(\kappa^2/2)}{2n+1} \leq \frac{6 + 2\log N}{\pi}.$$

Proof. Using (3.16), and the fact that $0 \leq \frac{\kappa^2}{\kappa^2+n+1/2} \leq 1$, we get

$$\begin{aligned}\frac{2\kappa e^{-\kappa^2/2}}{\sqrt{\pi}} \sum_{n=0}^{N} \frac{I_n(\kappa^2/2) + I_{n+1}(\kappa^2/2)}{2n+1} &\leq \frac{4}{\pi} \sum_{n=0}^{N} \frac{1}{2n+1} \left(\frac{\kappa^2}{\kappa^2+n+1/2}\right)^{n+1/2} \\ &\leq \frac{4}{\pi} \sum_{n=0}^{N} \frac{1}{2n+1}.\end{aligned}$$

It is well-known that the last sum is $O(\log N)$. To be more precise,

$$\frac{4}{\pi} \sum_{n=0}^{N} \frac{1}{2n+1} \leq \frac{4}{\pi} \left(1 + \sum_{n=1}^{N} \frac{1}{2n}\right) = \frac{4}{\pi} + \frac{2}{\pi} \sum_{n=1}^{N} \frac{1}{n} \leq \frac{4}{\pi} + \frac{2}{\pi}(1 + \log N) \leq \frac{6 + 2\log N}{\pi}.$$

□

Now, if we just want to bound the coefficients' 1-norm, it suffices to take $N = \lceil \kappa^2 \rceil$, and bound the rest of the coefficients using the following simple tail bound:

Lemma 3.25. *Let $N \geq \kappa^2$ be an integer. Then,*

$$\frac{2\kappa e^{-\kappa^2/2}}{\sqrt{\pi}} \sum_{n=N}^{\infty} \frac{I_n(\kappa^2/2) + I_{n+1}(\kappa^2/2)}{2n+1} \leq 2^{2-N}.$$

Proof. Again, we start by using (3.16), but now we note that for $n \geq \kappa^2$, $0 \leq \frac{\kappa^2}{\kappa^2+n+1/2} \leq \frac{1}{2}$, so

$$\frac{2\kappa e^{-\kappa^2/2}}{\sqrt{\pi}} \sum_{n=N}^{\infty} \frac{I_n(\kappa^2/2) + I_{n+1}(\kappa^2/2)}{2n+1} \leq \frac{4}{\pi} \sum_{n=N}^{\infty} \frac{2^{-n}}{2n+1} \leq \frac{2^{3-N}}{\pi} \leq 2^{2-N}.$$

□

Therefore, the coefficient norm of the entire series is bounded by $\frac{6+2\log\kappa^2}{\pi} + 2^{2-\kappa^2} \leq 4 + 2\log\kappa$. An easy consequence of this bound is that we can approximate $\operatorname{erf}(\kappa x)$ up to error $0 \leq \epsilon \leq 2^{2-\kappa^2}$ with a polynomial of degree $\log_2(4/\epsilon)$.

If the desired error ϵ is larger than $2^{2-\kappa^2}$, a more careful analysis of the tail bound for $\kappa \leq N \leq \kappa^2$ yields an ϵ-approximation polynomial of degree $O(k\sqrt{\log(\kappa/\epsilon)})$.

Lemma 3.26. *Let $1 \leq \alpha \leq \kappa$ be an integer. Then,*

$$\frac{4}{\pi} \sum_{n=\alpha\kappa}^{(\alpha+1)\kappa-1} \frac{1}{2n+1} \left(\frac{\kappa^2}{\kappa^2+n+1/2}\right)^{n+1/2} \leq \frac{4}{\pi} e^{\alpha^2/2}.$$

Proof. First, we note that $\left(\frac{\kappa^2}{\kappa^2+n+1/2}\right)^{n+1/2} \leq \left(\frac{\kappa^2}{\kappa^2+n}\right)^n$, so we get

$$\begin{aligned}
&\frac{4}{\pi}\sum_{n=\alpha\kappa}^{(\alpha+1)\kappa-1}\frac{1}{2n+1}\left(\frac{\kappa^2}{\kappa^2+n+1/2}\right)^{n+1/2} \leq \frac{4}{\pi}\sum_{n=\alpha\kappa}^{(\alpha+1)\kappa-1}\frac{1}{2n+1}\left(\frac{\kappa^2}{\kappa^2+n}\right)^n \\
&\leq \frac{4}{\pi}\sum_{n=\alpha\kappa}^{(\alpha+1)\kappa-1}\frac{1}{2(\alpha\kappa)+1}\left(\frac{\kappa^2}{\kappa^2+\alpha\kappa}\right)^{\alpha\kappa} = \frac{4}{\pi}\frac{\kappa}{2(\alpha\kappa)+1}\left(\frac{\kappa}{\kappa+\alpha}\right)^{\alpha\kappa} \\
&= \frac{4}{\pi}\frac{\kappa}{2(\alpha\kappa)+1}\left(\frac{1}{1+\alpha/\kappa}\right)^{\alpha\kappa} \leq \frac{4}{\pi}\frac{\kappa}{2(\alpha\kappa)+1}\left(e^{-\alpha/(2\kappa)}\right)^{\alpha\kappa} \\
&= \frac{4}{\pi}\frac{\kappa}{2(\alpha\kappa)+1}e^{-\alpha^2/2} \leq \frac{4}{\pi}e^{-\alpha^2/2}.
\end{aligned}$$

The second to last inequality requires $\alpha \leq \kappa$. □

So, to get an ϵ-approximation polynomial, we just need to an integer $1 \leq \alpha_0 \leq \kappa$ such that

$$2^{2-\kappa^2} + \frac{4}{\pi}\sum_{\alpha=\alpha_0}^{\kappa} e^{-\alpha^2/2} \leq \epsilon.$$

Indeed, if we let $\epsilon' = \epsilon - 2^{2-\kappa^2}$, it suffices to choose $\alpha_0 = \left\lceil \sqrt{2\log(\frac{4\kappa}{\pi\epsilon'})} \right\rceil$, so we get

$$2^{2-\kappa^2} + \frac{4}{\pi}\sum_{\alpha=\alpha_0}^{\kappa} e^{-\alpha^2/2} \leq 2^{2-\kappa^2} + \frac{4\kappa}{\pi}e^{-\alpha_0^2/2} \leq 2^{2-\kappa^2} + \epsilon' = \epsilon.$$

Thus, the degree of the ϵ-approximating polynomial is $\alpha_0\kappa - 1 = O(\kappa\sqrt{\log(\kappa/\epsilon)})$.

3.7.3 Hamiltonian simulation, $\sin(\kappa x)$ and $\cos(\kappa x)$

The Hamiltonian simulation problem requires one to evaluate the function $e^{\mathbf{i}\kappa x}$ for a possibly large value of κ. This function is closely related to $\sin(\kappa x)$ and $\cos(\kappa x)$, and in this section we focus ourselves to the case of $\cos(\kappa x)$ with $\kappa \geq 1$. We recall the Jacobi-Anger expansion [DLMF, Eq. 10.12.3] of $\cos(\kappa x)$:

$$\cos(\kappa x) = J_0(\kappa) + 2\sum_{j=1}^{\infty}(-1)^j J_{2j}(\kappa)\mathcal{T}_{2j}(x), \tag{3.17}$$

where $J_n(x)$ is the Bessel function of the first kind. One could attempt to show an $\Omega(\sqrt{\kappa})$ lower bound on the coefficient 1-norm by using the large-argument approximation [DLMF, Eq. 10.7.8]

$$J_n(x) \approx \sqrt{\frac{2}{\pi x}}\cos\left(x - \frac{n\pi}{2} - \frac{\pi}{4}\right).$$

Unfortunately this approximation is only valid for $n \ll \sqrt{x}$, or equivalently, $j \ll \sqrt{\kappa}$ (a simple proof would require the same equality up to $j = O(\kappa)$). Therefore, we turn to positive results, and show how to evaluate $\cos(\kappa x)$ as a composition of easily-implementable functions corresponding to simple quantum circuits in the matrix (block-encoding) case.

First, we show that $\cos(x)$ can be approximated using a polynomial with a coefficient norm that is less than 1. We do this by showing that the 1-norm of the entire Chebyshev series of $\cos(x)$ is 1.

By specializing the expansion Equation (3.17) for $\kappa = 1$ and observing that $\mathcal{T}_{2j}(0) = (-1)^j$, we get that

$$1 = \cos 0 = J_0(1) + 2\sum_{j=1}^{\infty} J_{2j}(1),$$

so in order to bound the coefficient norm, it suffices to show that $J_{2j}(1) \geq 0$ for all $j > 0$. This holds by [DLMF, Eq. 10.14.2] and [DLMF, Eq. 10.14.7], i.e.

$$0 < J_\nu(\nu) < \frac{2^{\frac{1}{3}}}{3^{\frac{2}{3}}\Gamma\left(\frac{2}{3}\right)\nu^{\frac{1}{3}}} \quad \text{and} \quad 1 \leq \frac{J_\nu(\nu x)}{x^\nu J_\nu(\nu)} \leq e^{\nu(1-x)}.$$

In order to approximate $\cos(x)$ to an error ε, it suffices to truncate the series Equation (3.17) after $O(\log(1/\varepsilon))$ terms, since [DLMF, Eq. 10.14.4] gives us the tail bound $|J_n(x)| \leq \frac{|x|^n}{2^n n!}$.

Finally, for $\kappa > 1$ define $k := \lceil \kappa \rceil$, and note that it suffices to evaluate $\cos(kx)$ due to the (trivial) identity $\cos(\kappa x) = \cos(k(\kappa x/k))$. We conclude by observing that $\cos(kx) = \mathcal{T}_k(\cos x)$. The key point is that the functions $\kappa x/k$, $\cos(x)$ and $\mathcal{T}_k(x)$ can all be approximated with polynomials of coefficient norm at most 1, so composing them "just works" without any costly subnormalization. In other words, by composing these functions, we can start from a 1-block-encoding of a matrix A, and construct 1-block-encodings of $\frac{\kappa}{k}A$, $\cos\left(\frac{\kappa}{k}A\right)$, and finally $\mathcal{T}_k\left(\cos\left(\frac{\kappa}{k}A\right)\right) = \cos(\kappa A)$.

We can apply a similar trick in order to evaluate $\sin(\kappa x)$. As before, we note that it suffices to implement $\sin(kx)$ for $k = \lceil \kappa \rceil$, and we use the following identity involving the $\mathcal{U}_k$, the Chebyshev polynomials of the second kind:

$$\sin(kx) = \mathcal{U}_{k-1}(\cos x)\sin(x).$$

An easy calculation shows that simply negating the ancilla qubit at the start of the circuit in Section 3.3.2 allows us to compute a 1-block-encoding of $\sin(\kappa A)$. As a consequence, we obtain a simple LCU-based algorithm for Hamiltonian simulation.

Part II

Classical algorithms

4 | Gaussian sampling

Joint work with Simon Apers and Sander Gribling

4.1 Introduction and main result

One of the most important tasks in statistics and machine learning is to sample from high-dimensional and potentially complicated distributions. Markov chains are an efficient means for sampling from such distributions, and there is a wide variety of Markov chain algorithms designed specifically for this purpose. Typically, the main difficulty in analyzing these algorithms is to bound the precise running time or *mixing time* of the Markov chain. While many algorithms have been in very broad (heuristic) usage for several decades, rigorous bounds on their performance are often missing. A key example is the *Hamiltonian Monte Carlo* (HMC) algorithm [Dua+87]. This is an elegant Markov chain algorithm that utilizes Hamiltonian dynamics to efficiently explore the state space, without straying too far away from the high probability region. One of its key features is that it overcomes the slow, diffusive behavior that is inherent to "small step" approaches such as the ball walk and Langevin algorithm. While this is indeed observed in heuristic uses and studies of the HMC algorithm [Nea11], recent efforts are mostly restricted to step sizes much shorter than the heuristic choices [Che+20; CV22]. In this work, we prove seemingly optimal bounds on the HMC algorithm (with leapfrog integrator) for the special case of Gaussian distributions. This is the typical gateway to more complicated distributions such as logconcave or multimodal distributions. Our implementation of HMC exploits long and randomized integration times. This surpasses recent roadblocks on sampling Gaussian distributions using HMC with either short [CV22] or deterministic [LST21] integration times.

Our bounds are stated most easily in the "black box model", where the goal is to sample from a density of the form $e^{-f(x)}$ for $x \in \mathbb{R}^d$, and we are given query access to both f and its gradient ∇f. The Gaussian case further restricts f to be a quadratic form $f(x) = \frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)$, where μ and Σ are the (unknown) mean and covariance matrix of the Gaussian, respectively. The *condition number* κ of the Gaussian distribution is simply the condition number of Σ^{-1}. We prove the following theorem.

Theorem (informal version of Theorem 4.17)**.** *The Metropolis-adjusted HMC algorithm with leapfrog integrator can sample from a distribution ε-close in total variation distance to a d-dimensional Gaussian distribution with condition number κ using a total number of gradient evaluations*

$$\widetilde{O}(\sqrt{\kappa} d^{1/4} \log(1/\varepsilon)).^{a}$$

[a] We use the $\widetilde{O}(\cdot)$-notation to hide polylogarithmic factors in the problem parameters d, κ and $\log(1/\varepsilon)$.

This theorem builds on an analysis of the *unadjusted* HMC algorithm, for which we get a bound of $\widetilde{O}(\sqrt{\kappa} d^{1/4}/\sqrt{\varepsilon})$ on the total number of gradient evaluations. Both bounds seem in line with expectation [Dua+87; Nea11; Bes+13], and we expect they are tight when using the usual leapfrog integrator for simulating the Hamiltonian dynamics. Our algorithm surpasses the $\widetilde{\Omega}(\kappa\sqrt{d})$

lower bound on the complexity of HMC for Gaussian sampling from [LST21] by using *randomized* integration times. This avoids the well-known periodicity issues associated to a deterministic integration time.

Our work fits within the recent effort of proving non-asymptotic (and often tight) bounds on Markov chain algorithms for constrained distributions such as Gaussian distributions and, more generally, logconcave distributions (where f is assumed to be convex). Most of these efforts have focused on short step dynamics such as the ball walk, the Langevin algorithm, and HMC with short integration times. The use of such "local steps" makes it easier to control the stability and acceptance probability of the algorithm. However, the restriction to short step dynamics is also what slows down these algorithms, and this is what we avoid in our HMC algorithm.

Finally, the restriction to sampling Gaussian and logconcave distributions precisely parallels the restriction to quadratic and convex functions in optimization. Nonetheless, a gap between the (first-order oracle) complexity for logconcave sampling and the $O(\min\{\sqrt{\kappa}, d\})$ complexity for convex optimization is apparently deemed plausible. More specifically, the authors in [LST20] suggest an $\Omega(\kappa)$ lower bound for logconcave sampling. Our work shows that a sublinear κ-dependency is possible at least for the special case of Gaussian distributions, and we see it as evidence that a general $O(\sqrt{\kappa})$ bound for logconcave sampling might be achievable.

4.1.1 Background and prior work

There is a vast body of work on the use of Markov chain algorithms for sampling from Gaussian and logconcave distributions. These works mostly consider the (Metropolized) random walk or ball walk (MRW), the Metropolis-adjusted Langevin algorithm (MALA), and HMC. We discuss those works most directly related to ours.

The earliest works focus on *asymptotic* bounds or scaling limits on the performance as $d \to \infty$. A $d^{1/4}$-scaling was suggested in [Dua+87; KP91; Bes+13] for the complexity of HMC with leapfrog integrator for Gaussians and logconcave product distributions. This improves over the expected d- and $d^{1/3}$-scalings of MRW and MALA, respectively. Indeed, in a recent work by Chewi et al. [Che+21] it was proven that the complexity of MALA for standard Gaussian distributions (with $\kappa = 1$) scales as $\widetilde{O}(d^{1/3})$. For leapfrog HMC, the only non-asymptotic bounds scaling with $d^{1/4}$ seem to have been proven recently in [MV18; Mou+21] for the *unadjusted* HMC chain, and under additional regularity assumptions. While these assumptions include Gaussians, the final complexities in these works scale at least with κ^2 and $1/\sqrt{\varepsilon}$, and so scale much worse in terms of both κ and ε compared to our bound.

An improved (linear) κ-dependency is obtained in recent works on MALA [Dwi+18; LST20; WSC21] and HMC [Che+20]. This seems optimal based on the $\widetilde{\Omega}(\kappa\sqrt{d})$ lower bounds on MALA and HMC from [WSC21; LST21], which even apply to the Gaussian case. Such lower bounds typically follow from either restricting to *short* integration times (as with MALA), which leads to diffusive behavior, or *fixed* integration times, which can lead to periodic behavior in the HMC algorithm. Either of these restrictions leads to an $\Omega(\kappa)$-dependency, and indeed we are not aware of any former non-asymptotic bounds on the mixing time achieving a *sublinear* κ-dependency (while using a numerical integrator). We sidestep these issues by using both *long* and *random* integration times. Analyzing the resulting algorithm can be significantly more involved, and for this we restrict our analysis to the Gaussian case. It however seems likely that this will form a gateway to proving $\sqrt{\kappa}$-scalings for general logconcave distributions.

The use of randomized integration times was also studied recently in the *randomized* HMC algorithm by Bou-Rabee and Sanz-Serna [BS17a]. Similarly to our work, they motivate their algorithm by looking at the Gaussian case, and obtain similar scalings to our work for properties

such as the autocorrelation time and mean displacement. In follow-up works [Del+21; LW22], bounds similar to ours are proven on the relaxation time. However, all of these results are proven only for the *idealized* case, and do not take into account the errors of numerical integration.

Finally, for completeness we also mention that there are algorithms for Gaussian sampling that are *not* based on Markov chains. While these are generally incomparable (e.g., require access to the precision or covariance matrix rather than gradient), we refer the interested reader to [VDC22].

4.2 Problem definition and preliminaries

4.2.1 Gaussian sampling

We consider a d-dimensional Gaussian distribution with unknown precision matrix B (equal to the inverse of the covariance matrix) and mean $\mu = 0$.[1] In such case, the Gaussian distribution is $\pi(x) \propto \exp(-f(x))$ with $f(x) = \frac{1}{2}x^\top Bx$ for $x \in \mathbb{R}^d$ and B a positive definite matrix. The algorithms we use (Hamiltonian Monte Carlo with a leapfrog integrator) are basis invariant, and so for ease of notation we will assume throughout that B is diagonal with $B_{ii} = \omega_i^2$ for each $i \in [d]$. As input, we are given bounds $0 < \alpha \le \beta$ such that $\alpha I \preceq B \preceq \beta I$, or, equivalently, $\alpha \le \omega_i^2 \le \beta$. The condition number of B is $\kappa = \beta/\alpha$ and we will also call this the condition number of π. We assume *first-order query access* to f, which means that a single query at a point $x \in \mathbb{R}^d$ provides both $f(x)$ and $\nabla f(x) = Bx$. The goal is to return a sample from a distribution that is ε-close to π in total variation distance, while making a minimal number of gradient queries to f.

4.2.2 Markov chains on $\mathbb{R}^d$

Throughout we work with Markov chains whose behaviour can be described as follows: when at $x \in \mathbb{R}^d$ move to $y \in \mathbb{R}^d$ with probability density $T(x, y) \ge 0$. We identify the Markov chain with the transition kernel (density) $T : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$. For a fixed $x \in \mathbb{R}^d$ we use T_x to denote the probability distribution on $\mathbb{R}^d$ with density $T(x, \cdot)$. Similarly (with some abuse of notation), we denote by T_μ the probability distribution on $\mathbb{R}^d$ with density $\int \mu(x)T(x, \cdot)\,\mathrm{d}x$. The K-step transition kernel T^K is defined recursively via $T^K(x, y) = \int_{\mathbb{R}^d} T^{K-1}(x, z)T(z, y)\,\mathrm{d}z$ for $K > 1$. We say that T satisfies the *detailed balance condition* with respect to the probability density $\pi : \mathbb{R}^d \to \mathbb{R}_+$ if

$$\pi(x)T(x, y) = \pi(y)T(y, x) \quad \text{for all } x, y \in \mathbb{R}^d.$$

The associated Markov chain is called *reversible*.

4.2.3 Hamiltonian dynamics and the harmonic oscillator

At its core, Hamiltonian Monte Carlo makes moves by integrating Hamiltonian dynamics. In general, these describe the evolution of a physical system parameterized by (generalized) *positions* and (generalized) *momenta*. For the purposes of this work, we denote the former with $x \in \mathbb{R}^d$ and the latter with $v \in \mathbb{R}^d$. We sometimes refer to v as the *velocity*, which in classical physics is equal to the momentum of a unit mass. The Hamiltonian evolution of a d-dimensional system is governed by its Hamiltonian $\mathcal{H} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$, which can be understood as the total energy of the system at position $x \in \mathbb{R}^d$ and with velocity $v \in \mathbb{R}^d$. The evolution of the system is described by the following equations:

$$\frac{\mathrm{d}x}{\mathrm{d}t} = \frac{\partial \mathcal{H}(x, v)}{\partial v}, \quad \frac{\mathrm{d}v}{\mathrm{d}t} = -\frac{\partial \mathcal{H}(x, v)}{\partial x}.$$

[1] This is without loss of generality. Using $\widetilde{O}(\sqrt{\kappa})$ gradient queries we can always determine the mean up to high precision and then translate the Gaussian to the origin.

The simplest example is the (one-dimensional) harmonic oscillator with Hamiltonian $\mathcal{H}(x,v) = \frac{1}{2}\omega^2 x^2 + \frac{1}{2}v^2$ for some given $\omega > 0$. Its evolution is described by $\frac{\mathrm{d}x}{\mathrm{d}t} = v$ and $\frac{\mathrm{d}v}{\mathrm{d}t} = -\omega^2 x$, which can be solved analytically to yield

$$\begin{bmatrix} x(t) \\ v(t) \end{bmatrix} = \begin{bmatrix} \cos(\omega t) & \frac{1}{\omega}\sin(\omega t) \\ -\omega \sin(\omega t) & \cos(\omega t) \end{bmatrix} \begin{bmatrix} x(0) \\ v(0) \end{bmatrix} \tag{4.1}$$

A more interesting example is the d-dimensional harmonic oscillator. For a given positive (semi)definite matrix $B \in \mathbb{R}^{d\times d}$, its Hamiltonian is $\mathcal{H}(x,v) = \frac{1}{2}x^\top Bx + \frac{1}{2}v^\top v$, and its evolution is described by

$$\frac{\mathrm{d}x}{\mathrm{d}t} = v, \quad \frac{\mathrm{d}v}{\mathrm{d}t} = -Bx. \tag{4.2}$$

If B has eigenvalues ω_i^2 then in the eigenbasis of B the system effectively decomposes into d independent harmonic oscillators with frequencies ω_i. When analyzing our algorithms, it is often useful to assume that B is diagonal, so we can treat each coordinate independently. Of course, the algorithms themselves remain basis-independent, and only require the aforementioned bounds α and β on the eigenvalues ω_i^2.

4.2.4 The leapfrog integrator

The leapfrog integrator, also known as the Störmer-Verlet method, is a well-known numerical integrator for Hamiltonian dynamics that uses two queries to $\frac{\partial \mathcal{H}(x,v)}{\partial x}$ in each integration step. In the Gaussian case $\mathcal{H}(x,v) = \frac{1}{2}x^\top Bx + \frac{1}{2}v^\top v$ and the propagator takes the following closed form:

$$\begin{bmatrix} x^{(n+1)} \\ v^{(n+1)} \end{bmatrix} = \begin{bmatrix} I - \frac{\delta^2}{2}B & \delta I \\ -\delta B(I - \frac{\delta}{4}B) & I - \frac{\delta^2}{2}B \end{bmatrix} \begin{bmatrix} x^{(n)} \\ v^{(n)} \end{bmatrix}, \tag{4.3}$$

where $\delta > 0$ is a parameter used to describe the integration time. See for example [LR05, Sec. 2.6] for details. We will exploit that, similarly as for the idealized Hamiltonian dynamics, the leapfrog dynamics also decouple in the diagonal basis of B. Hence, as before, we can assume without loss of generality that B is diagonal with entries $0 < \alpha \le \omega_i^2 \le \beta$, and the leapfrog integrator can be interpreted as integrating d independent harmonic oscillators. To understand the leapfrog integrator we can thus restrict to a single harmonic oscillator with parameter ω.

The propagator from Equation (4.3) has eigenvalues

$$\lambda^\pm = 1 - \frac{\delta^2\omega^2}{2} \pm \mathbf{i}\delta\omega\sqrt{1 - \frac{\delta^2\omega^2}{4}}.$$

If $\delta^2\omega^2 \le 4$, we can set $\lambda^\pm = e^{\pm \mathbf{i}\varphi}$, where $\varphi \in [0,\pi]$ is uniquely defined by $\cos(\varphi) = 1 - \frac{\delta^2\omega^2}{2}$ and $\sin(\varphi) = \delta\omega\sqrt{1 - \frac{\delta^2\omega^2}{4}}$. We can use φ to rewrite the propagator as a rotation with angle φ

$$\begin{bmatrix} \cos(\varphi) & \frac{1}{\hat\omega}\sin(\varphi) \\ -\hat\omega \sin(\varphi) & \cos(\varphi) \end{bmatrix}, \qquad \text{where } \hat\omega = \omega\sqrt{1 - \frac{\delta^2\omega^2}{4}}.$$

Comparing this with (4.1), we see that the leapfrog trajectory *exactly* follows the Hamiltonian dynamics for the modified Hamiltonian $\hat{\mathcal{H}}$ given by

$$\hat{\mathcal{H}}(x,v) = \frac{1}{2}\hat\omega^2 x^2 + \frac{1}{2}v^2.$$

Indeed, if $(\hat{x}(t), \hat{v}(t))$ is the solution of Hamilton's equations with Hamiltonian $\hat{\mathcal{H}}(x, v)$ and initial conditions $(\hat{x}(0) = x_0, \hat{v}(0) = v_0)$, then the nth point on the leapfrog trajectory equals

$$\begin{bmatrix} \hat{x}^{(n)} \\ \hat{v}^{(n)} \end{bmatrix} = \begin{bmatrix} \cos(n\varphi) & \frac{1}{\hat{\omega}}\sin(n\varphi) \\ -\hat{\omega}\sin(n\varphi) & \cos(n\varphi) \end{bmatrix} \begin{bmatrix} \hat{x}_0 \\ \hat{v}_0 \end{bmatrix} = \begin{bmatrix} \cos(\hat{\omega}t_n) & \frac{1}{\hat{\omega}}\sin(\hat{\omega}t_n) \\ -\hat{\omega}\sin(\hat{\omega}t_n) & \cos(\hat{\omega}t_n) \end{bmatrix} \begin{bmatrix} \hat{x}_0 \\ \hat{v}_0 \end{bmatrix} = \begin{bmatrix} \hat{x}(t_n) \\ \hat{v}(t_n)) \end{bmatrix},$$

where $t_n = n\varphi/\hat{\omega}$. We can now easily check that the difference between $\mathcal{H}$ and $\hat{\mathcal{H}}$ is

$$\mathcal{H}(x, v) - \hat{\mathcal{H}}(x, v) = \frac{\delta^2\omega^4 x^2}{8}.$$

By our former remark, this observation extends to general d-dimensional harmonic oscillators and the corresponding leapfrog integrator (4.3): we define $\hat{B}$ by replacing ω_i by $\hat{\omega}_i$ for each eigenvalue of B, where

$$\hat{\omega}_i := \omega_i\sqrt{1 - \frac{\delta^2\omega_i^2}{4}} \tag{4.4}$$

and we set $\hat{\mathcal{H}}(x, v) = \frac{1}{2}x^\top \hat{B}x + \frac{1}{2}v^\top v$, then the leapfrog integrator is an exact integrator for $\hat{\mathcal{H}}$ and we moreover have

$$\mathcal{H}(x, v) - \hat{\mathcal{H}}(x, v) = \frac{\delta^2}{8}\sum_{i\in[d]} \omega_i^4 x_i^2. \tag{4.5}$$

Finally we introduce the following notation: the tuple $(x', v') = \text{leapfrog}(x, v, t, \delta)$ is defined as the (position,momentum) vector after taking t/δ leapfrog integration steps for $\mathcal{H}$ with stepsize $0 \le \delta \le 1/\sqrt{\beta}$.[2]

4.3 Idealized and unadjusted HMC

We first analyze an idealized version of HMC, Algorithm 2, where we assume that we can exactly integrate the Hamiltonian dynamics. We use long and random integration times. In order to later apply the results from this section in the setting of a numerical integrator, we will use uniformly random integration times $t \sim U(\mathcal{T})$ from a *finite* set $\mathcal{T}$. We require only the following property of $\mathcal{T}$. For all $0 < \alpha \le \omega^2 \le \beta$ we have

$$\mathbb{P}_{t\sim U(\mathcal{T})}\big[|\cos(\omega t)| \le 0.9\big] \ge 1/2. \tag{4.6}$$

In Section 4.6.1 we show that $\mathcal{T} = \{k \cdot \delta \mid k \in \mathbb{N},\ k \cdot \delta < 10\pi/\sqrt{\alpha}\}$ satisfies this property for all $0 < \delta < \frac{c}{\sqrt{\beta}}$ for a sufficiently small constant $c > 0$.

Algorithme 2 : Markov kernel P (idealized HMC with random integration time)

Input : $x \in \mathbb{R}^d$, $\mathcal{T}$ as in Equation (4.6)
Output : $x' \in \mathbb{R}^d$
1 Draw $v \sim \mathcal{N}(0, I_d)$ and $t \sim U(\mathcal{T})$;
2 Define x' by following Hamiltonian dynamics for $\mathcal{H}$ for time t, starting from (x, v);

It is well known that idealized HMC with a fixed integration time has the desired stationary distribution π whose density at (x, v) is related to the Hamiltonian $\mathcal{H}(x, v) = \frac{1}{2}x^\top \operatorname{diag}(\boldsymbol{\omega})x + \frac{1}{2}v^\top v$, i.e., $\pi(x, y) \propto \exp(-\frac{1}{2}x^\top \operatorname{diag}(\boldsymbol{\omega})x + \frac{1}{2}v^\top v)$ (cf. [Dua+87; Nea96; Vis21]). From this it follows

[2] We will always apply this with $t/\delta \in \mathbb{N}$.

that also P has stationary distribution π. In Section 4.3.1 we show that P has a small mixing time. We then extend this result to the setting where we use a numerical integrator (leapfrog) instead of the idealized time evolution according to Hamiltonian dynamics. For this we use the fact (cf. Section 4.2.4) that the leapfrog integrator applied to $\mathcal{H}(x,v)$ can in fact be viewed as an exact integrator for the Hamiltonian dynamics of a modified Hamiltonian $\hat{\mathcal{H}}(x,v)$. By bounding the distance between π and $\hat{\pi} \propto \exp(-\hat{\mathcal{H}}(x,v))$, we obtain a sample from a distribution that is ε close to π in total variation distance using a number of gradient evaluations that scales as $\widetilde{O}(\sqrt{\kappa} d^{1/4}/\sqrt{\varepsilon})$, see Section 4.3.2.

4.3.1 Idealized HMC

Let P_x^t denote the proposal distribution from $x \in \mathbb{R}^d$, conditioned on having picked $t \in [0,T]$. Using the explicit expression Equation (4.1), we can expand it as

$$\begin{aligned} P_x^t(z) &= \mathbb{P}_{v\sim\mathcal{N}(0,1)}\left[\cos(\omega_i t)x_i + \frac{1}{\omega_i}\sin(\omega_i t)v_i = z_i \quad \forall i \in [d]\right] \\ &= (2\pi)^{-d/2}\prod_{i\in[d]}\frac{\omega_i}{|\sin(\omega_i t)|}\exp\left(-\frac{1}{2}\left(\frac{z_i - \cos(\omega_i t)x_i}{\frac{1}{\omega_i}\sin(\omega_i t)}\right)^2\right). \end{aligned} \tag{4.7}$$

The probability with which idealized HMC moves from x to z is then $P_x(z) = \frac{1}{|\mathcal{T}|}\sum_{t\in\mathcal{T}} P_x^t(z)\,\mathrm{d}t$.

We analyze the convergence in total variation distance by explicitly writing out the distribution P^K obtained by taking K steps of the idealized HMC method. If we condition on the choice of random integration times in step 2 of Algorithm 2, then the resulting distribution is again a normal distribution. Indeed, let $(v^{(1)},\dots,v^{(K)})$, $(t_1,\dots,t_K)$ and $(x^{(1)},\dots,x^{(K)})$ denote the velocities, integration times and positions, respectively, encountered during the first K steps. By repeatedly applying (4.7), we can express

$$\begin{aligned} x_i^{(K)} &= x_i^{(K-1)}\cos(\omega_i t_K) + \frac{1}{\omega_i}\sin(\omega_i t_K)v^{(K)} \\ &= x_i^{(0)}\left(\prod_{k=1}^K \cos(\omega_i t_k)\right) + \frac{1}{\omega_i}\sum_{k=1}^K v^{(k)}\sin(\omega_i t_k)\left(\prod_{j=k+1}^K \cos(\omega_i t_j)\right). \end{aligned}$$

For a fixed tuple $\mathbf{t} = (t_1,\dots,t_K) \in \mathcal{T}^K$ but *random* choices $(v^{(1)},\dots,v^{(K)}) \sim \mathcal{N}(0,I_d)^K$, we now argue that this describes a Gaussian distribution, which we denote by $P_x^{\mathbf{t}}$. First, note that $P_x^{\mathbf{t}}$ is a product distribution: $P_x^{\mathbf{t}}(z) = \prod_{i\in[d]} P_x^{\mathbf{t},i}(z_i)$ where we use $P_x^{\mathbf{t},i}$ for the marginal distribution of $P_x^{\mathbf{t}}$ with respect to the i-th coordinate. Then, note that $P_x^{\mathbf{t},i}$ describes a sum of Gaussians, and hence forms again a Gaussian. We formalize this in the next lemma.

Lemma 4.1. *Let $\mathbf{t} \in \mathcal{T}^K$, $\omega > 0$, $x \in \mathbb{R}$, and consider*

$$z = x\left(\prod_{k=1}^K \cos(\omega t_k)\right) + \frac{1}{\omega}\sum_{k=1}^K v^{(k)}\sin(\omega t_k)\left(\prod_{j=k+1}^K \cos(\omega t_j)\right)$$

where $v^{(k)} \sim \mathcal{N}(0,1)$ for each $k \in [K]$. Then $z \sim \mathcal{N}(x\prod_{k=1}^K \cos(\omega t_k), \frac{1}{\omega^2}(1 - \prod_{k=1}^K \cos(\omega t_j)^2))$.

If the term $\prod_{k=1}^K \cos(\omega t_k)$ is sufficiently small, then $P_x^{\mathbf{t}}$ is close to π. Equation (4.6) and Hoeffding's inequality show that for a random tuple $\mathbf{t} = (t_1,\dots,t_K) \sim U(\mathcal{T}^K)$ this term will indeed be small. Then we use this to prove convergence of the proposal distribution to π.

Lemma 4.2. *Let $0 < \alpha < \omega^2 < \beta$ and $\mathcal{T}$ as in Equation (4.6). Then there exists a constant $c > 0$ such that*

$$\mathbb{P}_{\mathbf{t}\sim U(\mathcal{T}^K)}\left[\left|\prod_{k=1}^{K}\cos(\omega t_k)\right| \geq 0.9^{K/4}\right] \leq \exp(-cK).$$

Using the above lemma, we show the proposal distributions $P_x^K(z) := P^K(x,z)$ and $P_y^K := P^K(y,z)$ are close provided that x and y are close.

Proposition 4.3. *There exists a constant $C > 0$ such that for every $x, y \in \mathbb{R}^d$, if*

$$K \geq C\max\{\log\left(\frac{d}{\varepsilon}\right), \log\left(\frac{d\|x-y\|_\infty}{\sqrt{\alpha}\varepsilon}\right)\},$$

then, with P the kernel of idealized HMC, we have

$$\|P_x^K - P_y^K\|_{\mathrm{TV}} \leq \varepsilon.$$

Proof. Recall that $P_x^K = \frac{1}{|\mathcal{T}|^K}\sum_{\mathbf{t}\in\mathcal{T}^K} P_x^{\mathbf{t}}$ and $P_x^{\mathbf{t}} = \prod_{i\in[d]} P_x^{\mathbf{t},i}$ is a product distribution. Hence, we can twice apply a triangle inequality to obtain

$$\begin{aligned}\|P_x^K - P_y^K\|_{\mathrm{TV}} &\leq \frac{1}{|\mathcal{T}|^K}\sum_{\mathbf{t}\in\mathcal{T}^K}\|P_x^{\mathbf{t}} - P_y^{\mathbf{t}}\|_{\mathrm{TV}} \\ &\leq \sum_{i\in[d]}\frac{1}{|\mathcal{T}|^K}\sum_{\mathbf{t}\in\mathcal{T}^K}\|P_x^{\mathbf{t},i} - P_y^{\mathbf{t},i}\|_{\mathrm{TV}} \qquad (4.8)\end{aligned}$$

Now let $\delta = \min\{\frac{1}{\sqrt{2}}, \frac{\sqrt{\alpha}\varepsilon}{d\|x-y\|_\infty}\}$. By Lemma 4.2, there exists a constant $C > 0$ such that for $K = C\max\{\log(\frac{d}{\varepsilon}), \log(\frac{1}{\delta})\}$ we have that

$$\mathbb{P}_{\mathbf{t}\sim U(\mathcal{T}^K)}\left[\left|\prod_{k=1}^{K}\cos(\omega_i t_j)\right| \geq \delta\right] \leq \frac{\varepsilon}{2d}$$

for each $i \in [d]$. Hence for each coordinate $i \in [d]$ we have

$$\begin{aligned}\frac{1}{|\mathcal{T}|^K}\sum_{\mathbf{t}\in\mathcal{T}^K}\|P_x^{\mathbf{t},i} - P_y^{\mathbf{t},i}\|_{\mathrm{TV}} &\leq \frac{\varepsilon}{2d} + \frac{1}{|\mathcal{T}|^K}\sum_{\mathbf{t}\in\mathcal{T}^K:|\prod_{k=1}^K\cos(\omega_i t_j)|\leq\delta}\|P_x^{\mathbf{t},i} - P_y^{\mathbf{t},i}\|_{\mathrm{TV}} \\ &\leq \frac{\varepsilon}{2d} + (1-\frac{\varepsilon}{2d})\frac{|x_i - y_i|\delta}{2\omega_i} \qquad (4.9)\end{aligned}$$

where we use that for $\mathbf{t} \in \mathcal{T}^K$ for which $\left|\prod_{k=1}^K\cos(\omega_i t_j)\right| \leq \delta \leq \frac{1}{\sqrt{2}}$, the proposal distributions $P_x^{\mathbf{t},i}$ and $P_y^{\mathbf{t},i}$ are univariate Gaussians with means μ_x, μ_y that satisfy $|\mu_x - \mu_y| \leq \delta|x_i - y_i|$, and both have variance $\sigma^2 \geq \frac{1-\delta^2}{\omega_i^2} \geq \frac{1}{2\omega_i^2}$. (For univariate Gaussians one has $\|\mathcal{N}(\mu_,\sigma^2) - \mathcal{N}(\mu_2,\sigma^2)\|_{\mathrm{TV}} < |\mu_1 - \mu_2|/\sigma$.) Combining Equations (4.8) and (4.9) we obtain $\|P_x^K - P_y^K\|_{\mathrm{TV}} \leq \varepsilon$. □

This bound then easily leads to a bound on the total variation distance between P_x^K and π for x that is sufficiently close to 0, and this is the main conclusion of this section.

Theorem 4.4 (Idealized HMC)**.** *There exists a constant $C > 0$ such that for every $x \in \mathbb{R}^d$, if*

$$K \geq C\log\left(\frac{d\kappa(1+\|x\|_\infty)}{\varepsilon}\right),$$

then, with $\pi \propto \exp(-\frac{1}{2}x^\top Bx)$ *and* P *the kernel of idealized HMC, we have*

$$\|P_x^K - \pi\|_{\mathrm{TV}} \le \varepsilon.$$

Proof. We write $\pi = \int_{\mathbb{R}^d} \delta_y \,\mathrm{d}\pi(y)$. Using that π is stationary for P (and hence P^K), we also have that $\pi = \int_{\mathbb{R}^d} P_y^K \,\mathrm{d}\pi(y)$. Now we apply Jensen's inequality:

$$\begin{aligned}\|P_x^K - \pi\|_{\mathrm{TV}} &\le \int_{y\in\mathbb{R}^d} \|P_x^K - P_y^K\|_{\mathrm{TV}} \,\mathrm{d}\pi(x) \\ &\le \pi(\{y : \|y\| > \gamma\}) + \int_{y\in\mathbb{R}^d : \|y\|\le\gamma} \|P_x^K - P_y^K\|_{\mathrm{TV}} \,\mathrm{d}\pi(x).\end{aligned}$$

Choosing γ sufficiently large ensures that $\pi(\{y : \|y\| > \gamma\}) \le \varepsilon/2$. Lemma 4.8 shows that $\gamma \in \widetilde{\Theta}(\sqrt{\beta d})$ suffices. We set $K = C\log(\frac{d\kappa(\gamma+\|x\|_\infty)}{\varepsilon})$ for the constant $C > 0$ for which Proposition 4.3 implies that $\|P_x^K - P_y^K\|_{\mathrm{TV}} \le \varepsilon/2$ for all $x, y \in \mathbb{R}^d$ with $\|x-y\|_\infty \le \gamma + \|x\|_\infty$. Combining these two bounds shows that $\|P_x^K - \pi\|_{\mathrm{TV}} \le \varepsilon$. □

4.3.2 Unadjusted HMC

The results from the previous section extend from the idealized setting where one can integrate exactly, to the setting where one uses a (suitably chosen) numerical integrator: the leapfrog integrator.

Algorithme 3 : Markov kernel $\hat{Q}$ (leapfrog HMC with random integration time)

Input : $x \in \mathbb{R}^d$, stepsize $\delta \le 1/\sqrt{\beta}$, $\mathcal{T} := \{k\cdot\delta \mid k \in \mathbb{N},\ k\cdot\delta < 10\pi/\sqrt{\alpha}\}$
Output : $x' \in \mathbb{R}^d$

1 Draw $v \sim \mathcal{N}(0, I_d)$ and move from x to (x, v) ;
2 Draw $t \sim U(\mathcal{T})$ and set $(x', v') = \mathrm{leapfrog}(x, v, t, \delta)$;

As discussed in Section 4.2.4, the leapfrog dynamics correspond to Hamiltonian dynamics for a slightly modified Hamiltonian $\widehat{\mathcal{H}}$. Bounding the distance between the stationary distribution $\hat{\pi}$ and π leads to the following poly$(1/\varepsilon)$-algorithm for sampling from a distribution ε-close to π.

Proposition 4.5 (Unadjusted HMC)**.** *There exist constants* $C, C' > 0$ *such that for every* $x \in \mathbb{R}^d$*, if*

$$K \ge C\log\left(\frac{d\kappa(1+\|x\|_\infty)}{\varepsilon}\right) \quad \text{and} \quad \delta \le C'\frac{\sqrt{\varepsilon}}{\sqrt{\beta}d^{1/4}},$$

then

$$\|\hat{Q}_x^K - \pi\|_{\mathrm{TV}} \le \varepsilon$$

where $\pi(x) \propto \exp(-\frac{1}{2}x^\top Bx)$ *and* $\hat{Q}$ *is the kernel of the unadjusted leapfrog HMC chain with step size* δ*. A sample from* $\hat{Q}_x^K$ *can be obtained using* $O(\frac{\sqrt{\kappa}d^{1/4}}{\sqrt{\varepsilon}}\log(\frac{d\kappa}{\varepsilon}))$ *gradient evaluations.*

Proof. By our discussion of the leapfrog integrator in Section 4.2.4, we know that $\hat{Q}$ corresponds to the idealized HMC algorithm for the modified Hamiltonian $\widehat{\mathcal{H}}$. Here we assume $\delta^2\omega_i^2 \le 4$ for all $i \in [d]$, i.e., $\delta \le \frac{1}{\sqrt{\beta}}$. It thus follows from Theorem 4.4 that if we start from $0 \in \mathbb{R}^d$ and take $K = O(\log(d\kappa/\varepsilon))$ steps of the chain $\hat{Q}$, then it returns a distribution that is $\varepsilon/2$-close to the modified stationary $\hat{\pi}$ defined as

$$\hat{\pi}(x) \propto \exp(-\frac{1}{2}x^\top \hat{B}x).$$

Using that $\hat{\pi}$ and π are both multivariate Gaussians, one can show (see Lemma 4.20 for completeness)

$$\|\pi - \hat{\pi}\|_{\mathrm{TV}} \leq \frac{3}{8}\delta^2 \sqrt{\sum_i \omega_i^4} \leq \frac{3}{8}\delta^2 \beta \sqrt{d}.$$

Hence by choosing a sufficiently small stepsize $\delta \in O(\sqrt{\varepsilon}/(\sqrt{\beta} d^{1/4}))$, we have that $\|\hat{\pi} - \pi\|_{\mathrm{TV}} \leq \varepsilon/2$. Together this shows that the resulting distribution after $K = O(\log(d\kappa/\varepsilon))$ steps will be ε-close to π.

It remains to bound the complexity of the algorithm. A single leapfrog step requires 2 gradient evaluations, and so a single step of the Markov chain $\hat{Q}$ requires $t/\delta \in O(\sqrt{\kappa} d^{1/4}/\sqrt{\varepsilon})$ gradient evaluations. Applying $O(\log(d\kappa/\varepsilon))$ steps of the Markov chain yields a total number of gradient evaluations

$$O\left(\frac{\sqrt{\kappa} d^{1/4}}{\sqrt{\varepsilon}} \log\left(\frac{d\kappa}{\varepsilon}\right)\right). \qquad \square$$

4.4 Metropolis-Adjusted HMC

Here we study the Metropolis-adjusted HMC algorithm. The algorithm applies a Metropolis filter to correct for the numerical errors of the integrator. This ensures that the algorithm has the correct stationary distribution, and leads to an overall improved error dependence.

Algorithme 4 : Markov kernel Q (Adjusted leapfrog HMC with random integration time)

Input : $x \in \mathbb{R}^d$, stepsize $\delta \in O(1/(\sqrt{\beta} d^{1/4}))$, $\mathcal{T} := \{k \cdot \delta \mid k \in \mathbb{N},\ k \cdot \delta < 10\pi/\sqrt{\alpha}\}$
Output : $x' \in \mathbb{R}^d$

1 Draw $v \sim \mathcal{N}(0, I_d)$ and move from x to (x, v) ;
2 Draw $t \sim U(\mathcal{T})$ and set $(x', v') = \mathrm{leapfrog}(x, v, t, \delta)$;
3 Accept with probability

$$\min\left\{1, \exp\left(-\mathcal{H}(x', -v') + \mathcal{H}(x, v)\right)\right\}$$

and return x'. Otherwise return $x' = x$;

We make a few observations about the adjusted HMC algorithm.

Lemma 4.6. *The Markov kernel Q defined in Algorithm 4 has the following properties:*

1. *Kernel Q is reversible with respect to its stationary distribution $\pi(x) \propto \exp(-\frac{1}{2}x^\top B x)$.*
2. *The acceptance probability in Line 3 is a function of only x and x':*

$$\min\left\{1, \exp\left(-\mathcal{H}(x', -v') + \mathcal{H}(x, v)\right)\right\} = \min\left\{1, \exp\left(\frac{\delta^2}{8}\sum_{i\in[d]} \omega_i^4 (x_i^2 - x_i'^2)\right)\right\} =: A(x, x'),$$

and we can rewrite $Q_x(x') = \hat{Q}_x(x') A(x, x')$ for $x \neq x'$.

Proof of Lemma 4.6, part 1. This fact is well known for fixed integration times. Here we prove that it also holds for *randomized* integration times.

We prove first that Q leaves the distribution $\pi(x) \propto \exp(-x^\top B x/2)$ invariant. To this end, we look at the larger *phase space*. Starting from $x \sim \pi$, the state (x, v) in step 2 is distributed according

to the distribution

$$\tilde{\pi}(x,v) \propto \exp(-x^\top Bx/2 - v^\top v/2) = \exp(-\mathcal{H}(x,v)).$$

It remains to prove that steps 2. and 3. leave $\tilde{\pi}$ invariant. Let T denote the kernel of the proposal generated in step 2. (i.e., proposal $(x', -v')$ has density $T((x,v),(x',v'))$). First we note that T is *symmetric*, i.e., $T((x,v),(x',v')) = T((x',v'),(x,v))$. To see this, recall that leapfrog integration is reversible in the sense that $\text{leapfrog}(x,v,t/\delta,\delta) = (x',v')$ implies that $\text{leapfrog}(x',-v',t/\delta,\delta) = (x,-v)$, and hence

$$\begin{aligned} T((x,v),(x',v')) &= \frac{1}{|U(\mathcal{T})|} \sum_{t \in U(\mathcal{T})} \mathbb{1}\left\{\text{leapfrog}(x,v,t) = (x',-v')\right\} \\ &= \frac{1}{|U(\mathcal{T})|} \sum_{t \in U(\mathcal{T})} \mathbb{1}\left\{\text{leapfrog}(x',v',t) = (x,-v)\right\} = T((x',v'),(x,v)). \end{aligned}$$

Then, note that step 3. effectively implements a Metropolis filter w.r.t. distribution $\tilde{\pi}$, which has acceptance probability

$$A((x,v),(x',v')) = \min\left\{1, \frac{\tilde{\pi}(x',v')}{\tilde{\pi}(x,v)}\right\} = \min\left\{1, \exp\Big(-\mathcal{H}(x',-v') + \mathcal{H}(x,v)\Big)\right\}.$$

It is then a direct consequence that steps 2. and 3. leave $\tilde{\pi}$ invariant as well.

Next, we show that Q is in fact *reversible* with respect to π, i.e.,

$$\pi(x)Q(x,x') = \pi(x')Q(x',x), \qquad \text{for all } x, x' \in \mathbb{R}^d.$$

To do this, we use the fact that for all $x, v \in \mathbb{R}^d$, the density $\tilde{\pi}(x,v)$ factorizes as $\tilde{\pi}(x,v) = \pi(x)\mu(v)$ with $\mu(v) \sim \exp(-v^\top v/2)$ a standard Gaussian. Using this, we get that

$$\begin{aligned} \pi(x)Q(x,x') &= \pi(x) \iint\limits_{v,v' \in \mathbb{R}^d} T((x,v),(x',v'))A((x,v),(x',v'))\,\mu(v)\,\mathrm{d}v\,\mathrm{d}v' \\ &= \pi(x) \iint\limits_{v,v' \in \mathbb{R}^d} T((x,v),(x',v'))\min\left\{1, \frac{\pi(x')\mu(v')}{\pi(x)\mu(v)}\right\}\mu(v)\,\mathrm{d}v\,\mathrm{d}v' \\ &= \iint\limits_{v,v' \in \mathbb{R}^d} T((x,v),(x',v'))\min\left\{\pi(x)\mu(v), \pi(x')\mu(v')\right\}\mathrm{d}v\,\mathrm{d}v'. \end{aligned}$$

Since each term in the last expression is symmetric under the exchange of (x,v) with (x',v'), we conclude that it is equal to $\pi(x')Q(x,x')$ for all x, x', and conclude that the chain is reversible. □

Proof of Lemma 4.6, part 2. First recall that $(x',v') = \text{leapfrog}(x,v,t/\delta,\delta)$. From Section 4.2.4 we know that the leapfrog integrator preserves the modified Hamiltonian and therefore we have

$$\hat{\mathcal{H}}(x,v) = \hat{\mathcal{H}}(x',v') = \hat{\mathcal{H}}(x',-v').$$

Moreover, by Equation (4.5) we have

$$\mathcal{H}(x,v) - \hat{\mathcal{H}}(x,v) = \frac{\delta^2}{8}\sum_{i \in [d]} \omega_i^4 x_i^2$$

for all $x, v \in \mathbb{R}^d$. Combining these two identities we find that

$$\begin{aligned}\mathcal{H}(x,v) - \mathcal{H}(x',-v') &= \left(\hat{\mathcal{H}}(x,v) + \frac{\delta^2}{8}\sum_{i\in[d]}\omega_i^4 x_i^2\right) - \left(\hat{\mathcal{H}}(x',-v') + \frac{\delta^2}{8}\sum_{i\in[d]}\omega_i^4 {x_i'}^2\right) \\ &= \frac{\delta^2}{8}\sum_{i\in[d]}\omega_i^4(x_i^2 - {x_i'}^2),\end{aligned}$$

and hence the acceptance probability takes the form $A(x,x')$ as claimed.

From this, it easily follows that Q_x takes the form $Q_x(x') = \hat{Q}_x(x')A(x,x')$ for $x \neq x'$:

$$\begin{aligned}Q_x(x') &= \iint_{v,v'\in\mathbb{R}^d} T((x,v),(x',v'))A((x,v),(x',v'))\,\mu(v)\,\mathrm{d}v\,\mathrm{d}v' \\ &= A(x,x')\iint_{v,v'\in\mathbb{R}^d} T((x,v),(x',v'))\,\mu(v)\,\mathrm{d}v\,\mathrm{d}v' = A(x,x')\hat{Q}_x(x').\end{aligned}$$ □

4.4.1 Concentration bounds on high-dimensional Gaussian random variables

Here we use concentration bounds on high-dimensional Gaussians to show that $\sum_{i\in[d]}\omega_i^4 x_i^2$ is, with high probability, close to $\sum_{i\in[d]}\omega_i^2$ both for $x \sim \pi$ and $x \sim \hat{\pi}$. We moreover show that in that case $\pi(x)$ and $\hat{\pi}(x)$ differ by at most a small multiplicative factor.

We will use the following version of the Hanson-Wright inequality [HW71] which gives a concentration inequality for quadratic forms of independent Gaussian random variables.

Theorem 4.7 (Hanson-Wright inequality [Ver18, Thrm 6.2.1])**.** *Let $X = (X_1, \dots, X_d) \in \mathbb{R}^d$ be a random vector with independent $\mathcal{N}(0,1)$ coordinates. Let A be a $d \times d$ matrix. Then, for every $t \geq 0$, we have*

$$\mathbb{P}\Big[|X^\top AX - \mathbb{E}[X^\top AX]| \geq t\Big] \leq 2\exp\left(-C\min\left\{\frac{t^2}{K^4\|A\|_F^2}, \frac{t}{K^2\|A\|}\right\}\right),$$

where $K, C > 0$ are constants.[a]

[a] The theorem holds more generally for independent mean zero *sub-gaussian* variables X_i. The constant K then upper bounds the *sub-gaussian norm* of all X_i.

Note that if $X \in \mathbb{R}^d$ is a random vector with independent $\mathcal{N}(0,1)$ coordinates, then so is $Y = UX$ for a rotation matrix U. This rotation-invariance allows us to again assume, for ease of notation, that $B = \mathrm{diag}(\boldsymbol{\omega})$. For convenience, recall that $\pi(x) = \frac{\prod_i \omega_i}{(2\pi)^{d/2}}\exp\left(-\frac{1}{2}\sum_i x_i^2\omega_i^2\right)$, and (cf. Equation (4.4)) that $\hat{\pi}$ is constructed similarly using $\hat{\boldsymbol{\omega}}$ which is defined, for each $i \in [d]$, as $\hat{\omega}_i = \omega_i\sqrt{1 - \frac{\delta^2\omega_i^2}{4}}$. We have $\omega_i^2 - \hat{\omega}_i^2 = \frac{1}{4}\delta^2\omega_i^4$. For $\gamma \geq 1$, we define the measurable set

$$E_\gamma := \left\{x \in \mathbb{R}^d \mid \left|x^\top \mathrm{diag}(\boldsymbol{\omega})^4 x - \sum_i \omega_i^2\right| \leq \gamma\sqrt{\sum_{i\in[d]}\omega_i^4}\right\}. \tag{4.10}$$

The Hanson-Wright inequality gives us the following concentration of measure for π and $\hat{\pi}$.

Lemma 4.8. *Let $\gamma \geq 1$ and consider E_γ as in Equation (4.10) then we have the following:*

1. *Let $\pi(x) \propto \exp(-\frac{1}{2}x^\top \operatorname{diag}(\boldsymbol{\omega})^2 x)$, then $\pi(E_\gamma) \geq 1 - 2\exp\big(-C\gamma\big)$ where $C > 0$ is a constant.*
2. *If $0 < \delta \leq \beta^{-1/2}d^{-1/4}$, then for $\hat{\pi}(x) \propto \exp(-\frac{1}{2}x^\top \operatorname{diag}(\hat{\boldsymbol{\omega}})^2 x)$ we have $\hat{\pi}(E_\gamma) \geq 1 - 2\exp(-C'\gamma)$ where $C' > 0$ is a constant.*

Proof. We first prove the concentration of measure for π. We have

$$\begin{aligned}\pi(E_\gamma) &= \mathbb{P}_{x\sim\pi}\left[\left|x^\top \operatorname{diag}(\boldsymbol{\omega})^4 x - \sum_i \omega_i^2\right| \leq \gamma\sqrt{\sum_{i\in[d]}\omega_i^4}\right] \\ &= \mathbb{P}_{z\sim\mathcal{N}(0,I_d)}\left[\left|z^\top \operatorname{diag}(\boldsymbol{\omega})^2 z - \sum_i \omega_i^2\right| \leq \gamma\sqrt{\sum_{i\in[d]}\omega_i^4}\right]\end{aligned}$$

where we set $z_i = \omega_i x_i$ for each $i \in [d]$ and observe that $z_i \sim \mathcal{N}(0,1)$. We apply Theorem 4.7 to the vector z, matrix $A = \operatorname{diag}(\boldsymbol{\omega})^2$, $t = \gamma\|A\|_F$, and note that $\|A\|_F \geq \|A\|$ implies the lower bound

$$\min\left\{\frac{(\gamma\|A\|_F)^2}{K^4\|A\|_F^2}, \frac{\gamma\|A\|_F}{K^2\|A\|}\right\} \geq \min\left\{\frac{\gamma^2}{K^4}, \frac{\gamma}{K^2}\right\} \geq \gamma\min\{K^{-2}, K^{-4}\}.$$

Therefore, for $C \leq \min\{K^{-2}, K^{-4}\}$ we obtain the desired bound for π.

We now use the same proof strategy to show concentration for $\hat{\pi}$. We have

$$\begin{aligned}\hat{\pi}(E_\gamma) &= \mathbb{P}_{x\sim\hat{\pi}}\left[\left|x^\top \operatorname{diag}(\boldsymbol{\omega})^4 x - \sum_i \omega_i^2\right| \leq \gamma\sqrt{\sum_{i\in[d]}\omega_i^4}\right] \\ &= \mathbb{P}_{z\sim\mathcal{N}(0,I_d)}\left[\left|z^\top \operatorname{diag}(\boldsymbol{\omega})^4 \operatorname{diag}(\hat{\boldsymbol{\omega}})^{-2} z - \sum_i \omega_i^2\right| \leq \gamma\sqrt{\sum_{i\in[d]}\omega_i^4}\right] \\ &\geq \mathbb{P}_{z\sim\mathcal{N}(0,I_d)}\left[\left|z^\top \operatorname{diag}(\boldsymbol{\omega})^4 \operatorname{diag}(\hat{\boldsymbol{\omega}})^{-2} z - \sum_i \omega_i^4/\hat{\omega}_i^2\right| \leq \gamma\sqrt{\sum_{i\in[d]}\omega_i^4} - |\sum_i \omega_i^2 - \omega_i^4/\hat{\omega}_i^2|\right]\end{aligned}$$

By definition $\omega_i^4/\hat{\omega}_i^2 = \omega_i^2/(1 - \delta^2\omega_i^2/4)$, and the upper bound on δ implies that $\delta^2\omega_i^2 \leq 2$. Using this bound, we get

$$\begin{aligned}\left|\sum_i \omega_i^2 - \omega_i^4/\hat{\omega}_i^2\right| &= \sum_i \omega_i^2\left(1 - \frac{1}{1-\delta^2\omega_i^2/4}\right) \leq \sum_i \omega_i^2(1 - 1 + \delta^2\omega_i^2/2) \\ &= \frac{1}{2}\sum_i \delta^2\omega_i^4 \leq \frac{1}{2\sqrt{d}}\sum_i \omega_i^2 \leq \frac{1}{2}\sqrt{\sum_i \omega_i^4}.\end{aligned}$$

Again, using the fact that $\omega_i^4/\hat{\omega}_i^2 \leq 2\omega_i^2$, we can further lower bound $\hat{\pi}(E_\gamma)$ as follows:

$$\hat{\pi}(E_\gamma) \geq \mathbb{P}_{z\sim\mathcal{N}(0,I_d)}\left[\left|z^\top \operatorname{diag}(\boldsymbol{\omega})^4 \operatorname{diag}(\hat{\boldsymbol{\omega}})^{-2} z - \sum_i \omega_i^4/\hat{\omega}_i^2\right| \leq \frac{\gamma}{4}\sqrt{\sum_{i\in[d]}(\omega_i^4/\hat{\omega}_i^2)^2}\right].$$

We can then again apply Theorem 4.7 to obtain $\hat{\pi}(E_\gamma) \geq 1 - 2\exp(-C'\gamma)$ for a suitable constant $C' > 0$. □

Next we give a bound on $\hat{\pi}(x)/\pi(x)$ for all $x \in E_\gamma$, which we will use later to show that $\hat{\pi}$ can be used as a warm start for π.

Lemma 4.9. *Let $\pi(x) \propto \exp(-\frac{1}{2}x^\top \operatorname{diag}(\boldsymbol{\omega})^2 x)$, let $\gamma \geq 1$ and consider E_γ as in Equation (4.10). Let $\delta = \frac{1}{10\sqrt{\gamma\beta}d^{1/4}}$, set $\hat{\omega}_i = \omega_i\sqrt{1-\frac{\delta^2\omega_i^2}{4}}$ for each $i \in [d]$, and let $\hat{\pi}(x) \propto \exp(-\frac{1}{2}x^\top \operatorname{diag}(\hat{\boldsymbol{\omega}})^2 x)$. Then for all $x \in E_\gamma$ we have*

$$0.9 \leq \frac{\hat{\pi}(x)}{\pi(x)} \leq 1.1.$$

Proof. For $x \in \mathbb{R}^d$ we have

$$\frac{\hat{\pi}(x)}{\pi(x)} = \left(\prod_i \left(1 - \frac{\delta^2\omega_i^2}{4}\right)\right)^{1/2} \exp\left(\frac{\delta^2}{8}\sum_i x_i^2\omega_i^4\right).$$

We first obtain an upper bound on $\frac{\hat{\pi}(x)}{\pi(x)}$ for $x \in E_\gamma$. Using the inequality $1 - z \leq \exp(-z)$ (which holds for all $z \in \mathbb{R}$), we obtain

$$\frac{\hat{\pi}(x)}{\pi(x)} \leq \exp\left(\frac{\delta^2}{8}\left(\sum_i x_i^2\omega_i^4 - \omega_i^2\right)\right) \leq \exp\left(\frac{1}{8}\delta^2\gamma\sqrt{\sum_{i\in[d]}\omega_i^4}\right) \leq \exp(\frac{1}{800}) \leq 1.1$$

where in the second inequality we use that $x \in E_\gamma$.

We can similarly bound $\frac{\hat{\pi}(x)}{\pi(x)}$ from below for $x \in E_\gamma$. For this we use the inequality $1 - z \geq \exp(-\eta z)$ which holds for $0 \leq z < 1$ and $\eta \geq \frac{1}{z}\ln(\frac{1}{1-z})$. For $z \leq 1/2$ one has $\frac{1}{z}\ln(\frac{1}{1-z}) \leq 1 + z$ and thus $\eta = 1 + z$ suffices. We apply this with $z = \frac{\delta^2\omega_i^2}{4} \leq \frac{1}{400\gamma\sqrt{d}} < 1/2$. This allows us to lower bound $\frac{\hat{\pi}(x)}{\pi(x)}$ as

$$\begin{aligned}
\frac{\hat{\pi}(x)}{\pi(x)} &\geq \exp\left(-\frac{1}{2}\left(1 + \frac{1}{400\gamma\sqrt{d}}\right)\frac{\delta^2}{4}\sum_i \omega_i^2\right)\exp\left(\frac{\delta^2}{8}\sum_i x_i^2\omega_i^4\right) \\
&\geq \exp\left(-\frac{1}{2}\frac{1}{400\gamma\sqrt{d}}\frac{\delta^2}{4}\sum_i \omega_i^2 - \frac{\delta^2}{8}\left|\sum_i x_i^2\omega_i^4 - \sum_i \omega_i^2\right|\right) \\
&\geq \exp\left(-\frac{1}{3200\cdot 100\gamma^2 d\beta}\sum_i \omega_i^2 - \frac{1}{800}\right) \geq \exp\left(-\frac{1}{400}\right) \geq 0.9
\end{aligned}$$

where in the third inequality we use that $\delta^2 = \frac{1}{100\gamma\beta\sqrt{d}}$ and $x \in E_\gamma$. □

Finally, we note that the acceptance probability is large on E_γ.

Lemma 4.10. *Let $A(x, x')$ be the acceptance probability of the adjusted leapfrog HMC with step size δ. If $x, x' \in E_\gamma$ then $A(x, x') \geq \exp\left(-\frac{\delta^2\gamma}{4}d^{1/2}\beta\right)$.*

Proof. If both $x, x' \in E_\gamma$ then we have that $\sum_{i\in[d]}\omega_i^4(x_i^2 - x_i'^2) \leq 2\gamma\sqrt{\sum_i \omega_i^4} \leq 2\gamma d^{1/2}\beta$. □

Lemmas 4.9 and 4.10 tell us that the stepsize δ should scale with γ, d and β as

$$\delta = \frac{1}{10\sqrt{\gamma\beta}d^{1/4}}. \tag{4.11}$$

This choice of δ ensures a high acceptance probability whenever $x, x' \in E_\gamma$ and a pointwise bound on the ratio $\hat{\pi}(x)/\pi(x)$ for $x \in E_\gamma$. In the next section we tune the choice of $\gamma \geq 1$ to apply an argument based on the s-conductance.

4.4.2 s-conductance and warm start

We will bound the mixing time of the Metropolis-adjusted chain using the so-called *s-conductance*. This is a generalization of the conductance that allows to ignore small subsets of measure $\pi(S) \leq s$.

Definition 4.11 (s-conductance)**.** Let $0 < s < 1/2$ and define the s-conductance C_s of a Markov chain with transition kernel T and stationary distribution π as

$$C_s := \inf\left\{C_s(S) \mid S \subseteq \mathbb{R}^d \text{ measurable},\ s < \pi(S) \leq \frac{1}{2}\right\}, \quad \text{with} \quad C_s(S) := \frac{\int_S T(x, S^c)\pi(\mathrm{d}x)}{\pi(S) - s}.$$

The s-conductance leads to a mixing time bound through the following theorem from Lovász and Simonovits (the formulation below is from [WSC21, Lem. 1]). It uses a *warmness* parameter $D_s^{\mu_0,\pi}$ between the initial distribution μ_0 and target distribution π, which for $0 < s < 1/2$ is defined by

$$D_s^{\mu_0,\pi} := \sup\{|\mu_0(A) - \pi(A)| \,:\, A \subseteq \mathbb{R}^d \text{ measurable},\ \pi(A) \leq s\}.$$

Lemma 4.12. *Consider a reversible, lazy*[a] *Markov chain with transition kernel R, stationary distribution π and initial distribution μ_0. Then for any $K \geq 0$ it holds that*

$$\|R_{\mu_0}^K - \pi\|_{\mathrm{TV}} \leq D_s^{\mu_0,\pi} + \frac{D_s^{\mu_0,\pi}}{s}\left(1 - \frac{C_s^2}{2}\right)^K.$$

[a] A lazy chain takes a step with probability $1/2$, and otherwise does nothing.

Using Lemma 4.9 we can prove that the stationary distribution $\hat{\pi}$ of the *unadjusted* chain $\hat{Q}$ for sufficiently small step size forms a warm start, if we take $\gamma \in \Theta(\log(1/s))$.

Lemma 4.13 (Unadjusted warm start)**.** *Let $\pi(x) \propto e^{-\frac{1}{2}x^\top \operatorname{diag}(\boldsymbol{\omega})^2 x}$ and let $\hat{\pi}(x) \propto e^{-\frac{1}{2}x^\top \operatorname{diag}(\hat{\boldsymbol{\omega}})^2 x}$ with $\hat{\omega}_i = \omega_i\sqrt{1 - \frac{\delta^2\omega_i^2}{4}}$. For any $0 < s < 1/2$, if $\delta \leq \frac{C}{\sqrt{\beta \log(1/s) d^{1/4}}}$ for a sufficiently small constant $C > 0$, then*

$$D_s^{\hat{\pi},\pi} \leq 3s.$$

Proof. Consider the set E_γ defined in (4.10) for a sufficiently large $\gamma \in O(\log(1/s))$. Then by Lemmas 4.8 and 4.9 both $\pi(E_\gamma) \geq 1 - s$ and $\hat{\pi}(E_\gamma) \geq 1 - s$, and $\hat{\pi}(x)/\pi(x) \leq 1.1$ for all $x \in E_\gamma$. Now let $A \subseteq \mathbb{R}^d$ with $\pi(A) \leq s$. Then we have

$$\begin{aligned}
|\hat{\pi}(A) - \pi(A)| &= |\hat{\pi}(A \cap E_\gamma) + \hat{\pi}(A \cap E_\gamma^c) - \pi(A \cap E_\gamma) - \pi(A \cap E_\gamma^c)| \\
&\leq |\hat{\pi}(A \cap E_\gamma) - \pi(A \cap E_\gamma)| + \hat{\pi}(A \cap E_\gamma^c) + \pi(A \cap E_\gamma^c) \\
&\leq \pi(A \cap E_\gamma) + \hat{\pi}(A \cap E_\gamma^c) + \pi(A \cap E_\gamma^c) \\
&\leq \pi(A) + s + s \leq 3s.
\end{aligned}$$

Here in the second inequality we use that $|\hat{\pi}(x) - \pi(x)| \leq \pi(x)$ for all $x \in E_\gamma$. □

4.4.3 Bounding the s-conductance of the adjusted HMC chain

To bound the s-conductance of the adjusted chain, we first bound the s-conductance of the *unadjusted* HMC chain $\hat{Q}$, and then relate both conductances. For the unadjusted chain, we can use our bounds on the mixing time of that chain to lower bound its conductance.

Lemma 4.14 (s-conductance unadjusted HMC)**.** *Let $0 < s < 1/2$ and let $\hat{C}_s$ be the s-conductance of the unadjusted HMC chain $\hat{Q}$ with step size $\delta \leq \frac{C}{\sqrt{\beta \log(1/s)} d^{1/4}}$ for a sufficiently small constant $C > 0$. Then*

$$\hat{C}_s \in \Omega(1/\log(d\kappa \log(1/s))).$$

Proof. First consider the s-conductance $\hat{C}_s^{(K)}$ of the K-*step* kernel $\hat{Q}^K$. From Proposition 4.5 we know that $\|\hat{Q}_x^K - \hat{\pi}\|_{\mathrm{TV}} \leq 1/10$ for $K \in \Omega(\log(d\kappa(1 + \|x\|_\infty)))$. In particular, if $x \in E_\gamma$ with $\gamma \geq 1$ then $\|x\|_\infty \leq (\gamma + 1)d\kappa$ and hence $\|\hat{Q}_x^K - \hat{\pi}\|_{\mathrm{TV}} \leq 1/10$ for all $x \in E_\gamma$ and $K \in \Omega(\log(\gamma d\kappa))$. By Lemma 4.8 we can ensure $\hat{\pi}(E_\gamma) \geq 1 - s$ by picking $\gamma \in O(\log(1/s))$ (recall that $\delta = \frac{1}{10\sqrt{\gamma\beta} d^{1/4}}$), in which case $K \in O(\log(d\kappa \log(1/s)))$. As a consequence, for any S for which $s < \hat{\pi}(S) \leq 1/2$ we have that

$$\begin{aligned}\hat{C}_s^{(K)}(S) = \frac{\int_S \hat{\pi}(x)\hat{Q}_x^K(S^c)}{\hat{\pi}(S) - s} &\geq \frac{\int_{S \cap E_\gamma} \hat{\pi}(x)\hat{Q}_x^K(S^c)}{\hat{\pi}(S) - s} \\ &\geq \frac{\hat{\pi}(S \cap E_\gamma)(\hat{\pi}(S^c) - 1/10)}{\hat{\pi}(S) - s} \geq \hat{\pi}(S^c) - \frac{1}{10} \geq \frac{2}{5},\end{aligned}$$

and hence $\hat{C}_s^{(K)} \geq 2/5$.

Now we can use the fact that $\hat{C}_s^{(K)} \leq K\hat{C}_s^{(1)} = K\hat{C}_s$ to conclude that $\hat{C}_s \geq 2/(5K)$, which is $\Omega(1/\log(d\kappa \log(1/s)))$ as claimed. To see that $\hat{C}_s^{(K)} \leq K\hat{C}_s^{(1)}$ (which is well-known, see e.g. [Lev+17, Eq. (7.10)]), define $\hat{\pi}_S$ by $\hat{\pi}_S(x) = \hat{\pi}(x)$ for $x \in S$ and $\hat{\pi}_S(x) = 0$ elsewhere. Then note that $\hat{C}_s^{(K)}(S) = \|Q_{\hat{\pi}_S}^K - \hat{\pi}_S\|_{\mathrm{TV}}/(\hat{\pi}(S) - s)$. Using a telescoping sum and a triangle inequality we can bound

$$\begin{aligned}\|Q_{\hat{\pi}_S}^K - \hat{\pi}_S\|_{\mathrm{TV}} &\leq \|Q_{\hat{\pi}_S}^K - Q_{\hat{\pi}_S}^{K-1}\|_{\mathrm{TV}} + \|Q_{\hat{\pi}_S}^{K-1} - Q_{\hat{\pi}_S}^{K-2}\|_{\mathrm{TV}} + \cdots + \|Q_{\hat{\pi}_S} - \hat{\pi}_S\|_{\mathrm{TV}} \\ &\leq K\|Q_{\hat{\pi}_S} - \hat{\pi}_S\|_{\mathrm{TV}},\end{aligned}$$

where the second inequality follows from submultiplicativity of the total variation distance. Dividing both sides by $(\hat{\pi}(S) - s)$ and taking the infimum over S proves that $\hat{C}_s^{(K)} \leq K\hat{C}_s^{(1)}$. □

To relate the s-conductance of the adjusted chain to the one of the unadjusted chain, we use the properties of π and $\hat{\pi}$ shown in Section 4.4.1: there is a set $E \subseteq \mathbb{R}^d$ of large measure on which π and $\hat{\pi}$ pointwise differ by at most a small multiplicative constant. Moreover, if both $x \in E$ and $x' \in E$, then the acceptance probability of the adjusted chain satisfies $A(x, x') \geq 99/100$.

Lemma 4.15 (s-conductance adjusted HMC)**.** *Let $0 < s < C/\log(d\kappa)$ for a sufficiently small constant $C > 0$, and let C_s and $\hat{C}_{s/2}$ be the s-conductance and the $s/2$-conductance of the adjusted and unadjusted chains Q and $\hat{Q}$ with step size $\delta \leq \frac{C'}{\sqrt{\beta \log(1/s)} d^{1/4}}$ for a sufficiently small constant $C' > 0$. Then*

$$C_s \geq \hat{C}_{s/2}/2.$$

Proof. Our goal is to lower bound $\frac{1}{\pi(S)-s}\int_S \pi(x)Q(x,S^c)\,\mathrm{d}x$ for all sets S such that $s < \pi(S) \le \frac{1}{2}$. To this end, we will use that by Lemmas 4.8, 4.8, 4.9 and 4.10 the set $E := E_\gamma \subset \mathbb{R}^d$ (defined in Equation (4.10)) for a suitable $\gamma \in \Theta(\log(1/s))$ and $\delta = \frac{1}{10\sqrt{\gamma\beta}d^{1/4}}$ (as in Equation (4.11)) satisfies

1. $\pi(E^c) \le s/10$,
2. $\hat{\pi}(E^c) \le s^2/10$,
3. $0.9 \le \frac{\hat{\pi}(x)}{\pi(x)} \le 1.1$ for all $x \in E$,
4. the acceptance probability $A(x,x') \ge 99/100$ for all $x, y \in E$.

Note that in Lemma 4.14 we have shown that $\hat{C}_{s/2} \in \Omega(1/\log(d\kappa\log(1/s)))$. Therefore, for $s < C/\log(d\kappa)$ for a small enough constant $C > 0$, we have $s \le \hat{C}_{s/2}$ and thus $\hat{\pi}(E^c) \le s\hat{C}_{s/2}/10$.

We can use this to lower bound the integral

$$\begin{aligned}
\int_S \pi(x)Q(x,S^c)\,\mathrm{d}x &\ge \int_{S\cap E} \pi(x)Q(x,S^c\cap E)\,\mathrm{d}x \\
&= \int_{S\cap E}\pi(x)\int_{S^c\cap E} Q(x,y)\,\mathrm{d}y\,\mathrm{d}x \\
&= \int_{S\cap E}\pi(x)\int_{S^c\cap E} \hat{Q}(x,y)A(x,y)\,\mathrm{d}y\,\mathrm{d}x \\
&\ge 0.85\int_{S\cap E}\hat{\pi}(x)\int_{S^c\cap E} \hat{Q}(x,y)\,\mathrm{d}y\,\mathrm{d}x \\
&= 0.85\int_{S\cap E}\hat{\pi}(x)\hat{Q}(x,S^c\cap E)\,\mathrm{d}x \\
&= 0.85\left(\int_{S\cap E}\hat{\pi}(x)\hat{Q}(x,S^c\cup E^c)\,\mathrm{d}x - \int_{S\cap E}\hat{\pi}(x)\hat{Q}(x,E^c)\,\mathrm{d}x\right) \\
&\ge 0.85\left(\int_{S\cap E}\hat{\pi}(x)\hat{Q}(x,S^c\cup E^c)\,\mathrm{d}x - \hat{\pi}(E^c)\right),
\end{aligned}$$

where the last inequality follows from detailed balance:

$$\int_{S\cap E}\hat{\pi}(x)\hat{Q}(x,E^c)\,\mathrm{d}x = \int_{E^c}\hat{\pi}(x)\hat{Q}(x,S\cap E)\,\mathrm{d}x \le \hat{\pi}(E^c).$$

We recognize the last integral as the ergodic flow from the set $S' := S\cap E$ to its complement, and so we can lower bound it in terms of the conductance of $\hat{Q}$, provided that S' has an appropriate measure according to $\hat{\pi}$. We bound $\hat{\pi}(S')$ from below

$$\hat{\pi}(S') \ge 0.9\pi(S') = 0.9(\pi(S) - \pi(S\cap E^c)) \ge 0.9s - \pi(E^c) \ge 0.8s,$$

and from above:

$$\hat{\pi}(S') \le 1.1\pi(S') \le 1.1\pi(S) \le 0.55.$$

We proceed in two different ways depending on the measure $\hat{\pi}(S')$.

1. If $0.8s \le \hat{\pi}(S') \le 1/2$, we have the lower bound

$$\begin{aligned} C_s = \frac{\int_S \pi(x)Q(x,S^c)\,\mathrm{d}x}{\pi(S)-s} &\ge 0.85\frac{\hat{C}_{s/2}(\hat{\pi}(S')-s/2)-\hat{\pi}(E^c)}{\pi(S)-s} \\ &\ge 0.85\frac{\hat{C}_{s/2}(\hat{\pi}(S')-0.6s)}{\pi(S)-s} \\ &\ge 0.85\frac{\hat{C}_{s/2}(0.9\pi(S')-0.6s)}{\pi(S)-s} \\ &\ge 0.85\frac{\hat{C}_{s/2}(0.9\pi(S)-\pi(E^c)-0.6s)}{\pi(S)-s} \\ &\ge 0.85\frac{\hat{C}_{s/2}(0.9\pi(S)-0.7s)}{\pi(S)-s} \\ &\ge 0.85\frac{\hat{C}_{s/2}(0.7\pi(S)-0.7s)}{\pi(S)-s} \ge \frac{\hat{C}_{s/2}}{2}. \end{aligned}$$

2. If $1/2 \le \hat{\pi}(S') \le 0.55$, we have $s \le \hat{\pi}(S'^c) \le 1/2$. Additionally, we know that $\hat{Q}$ satisfies detailed balance:

$$\int_{S'} \hat{\pi}(x)\hat{Q}(x,S'^c)\,\mathrm{d}x = \int_{S'^c} \hat{\pi}(x)\hat{Q}(x,S')\,\mathrm{d}x.$$

Therefore, we have the following lower bound

$$\begin{aligned} C_s = \frac{\int_S \pi(x)Q(x,S^c)\,\mathrm{d}x}{\pi(S)-s} &\ge 0.85\frac{\hat{C}_{s/2}(\hat{\pi}(S'^c)-s/2)-\hat{\pi}(E^c)}{\pi(S^c)-s} \\ &\ge 0.85\frac{\hat{C}_{s/2}(\hat{\pi}(S'^c)-0.6s)}{\pi(S^c)-s} \\ &= 0.85\frac{\hat{C}_{s/2}(1-\hat{\pi}(S')-0.6s)}{1-\pi(S)-s} \\ &\ge 0.85\frac{\hat{C}_{s/2}(1-1.1\pi(S)-0.6s)}{1-\pi(S)-s} \\ &\ge 0.85\frac{\hat{C}_{s/2}(1-1.1\pi(S)-0.6s)}{1-\pi(S)-0.6s} \ge \frac{\hat{C}_{s/2}}{2}. \end{aligned}$$

□

4.4.4 Mixing time of adjusted HMC

We can now plug our bounds on the s-conductance into Lemma 4.12 to get the following bound on the mixing time of the (lazy) Metropolis-adjusted HMC chain,[3] when starting from a warm start.

Theorem 4.16 (Metropolis-adjusted HMC with warm start)**.** *Let $0 < \varepsilon < C/\log(d\kappa)$ for a sufficiently small constant $C > 0$, and let μ_0 be an initial distribution with warmness $D_s^{\mu_0,\pi} \le \varepsilon/2$ for $s = \varepsilon/6$. There exist constants $C', C'' > 0$ such that for every $x \in \mathbb{R}^d$, if*

$$K \ge C' \log(d\kappa \log(1/\varepsilon))\log(1/\varepsilon) \quad \text{and} \quad \delta \le \frac{C''}{\sqrt{\beta \log(1/\varepsilon)}d^{1/4}},$$

[3] Making the chain lazy reduces the s-conductance only by a factor 2.

then

$$\|Q_{\mu_0}^K - \pi\|_{\mathrm{TV}} \le \varepsilon$$

where $\pi \propto \exp(-x^\top Bx/2)$ and Q is the kernel of the (lazy) Metropolis-adjusted leapfrog HMC chain with step size δ.

Proof. For $s = \varepsilon/6$ and our choice of δ we know from Lemmas 4.15 and 4.14 that Q has s-conductance $C_s \in \Omega(1/\log(d\kappa\log(1/s)))$. By invoking Lemma 4.12 we know that

$$\|Q_{\mu_0}^K - \pi\|_{\mathrm{TV}} \le D_s + \frac{D_s}{s}\left(1 - \frac{C_s^2}{2}\right)^K \le \frac{\varepsilon}{2} + 3\left(1 - \frac{C_s^2}{2}\right)^K \le \varepsilon$$

for $K \in \Omega(\log(1/\varepsilon)/C_s)$ and hence $K \in \Omega(\log(d\kappa\log(1/\varepsilon))\log(1/\varepsilon))$. □

Hence, starting from a warm start μ_0 we can sample from a distribution ε-close to π in TV-distance using $\widetilde{O}(\sqrt{\kappa}d^{1/4}\log(1/\varepsilon))$ gradient evaluations. To get around this warm start, recall from Lemma 4.13 that the stationary distribution of the *unadjusted* chain (with sufficiently small step size δ) provides a warm start for the adjusted chain. This gives the following, main theorem.

Theorem 4.17 (Metropolis-adjusted HMC)**.** *Let $0 < \varepsilon < C/\log(d\kappa)$ for a sufficiently small constant $C > 0$. There exists constants $C_0', C', C'' > 0$ such that for every $x \in \mathbb{R}^d$, if*

$$K \ge C'\log(d\kappa\log(1/\varepsilon))\log(1/\varepsilon), \quad K_0 \ge C_0'\log\left(d\kappa(1+\|x\|_\infty)/\varepsilon\right), \quad \delta \le \frac{C''}{\sqrt{\beta\log(1/s)}d^{1/4}},$$

then

$$\|(Q^K \circ \hat{Q}^{K_0})_x - \pi\|_{\mathrm{TV}} \le \varepsilon$$

where $\pi \propto \exp(-x^\top Bx/2)$ and Q (resp. $\hat{Q}$) is the kernel of the (lazy) Metropolis-adjusted (resp. unadjusted) leapfrog HMC chain with step size δ. We can thus obtain a sample from a distribution that is ε-close to π in TV-distance using $\widetilde{O}(\sqrt{\kappa}d^{1/4}\log(1/\varepsilon))$ gradient evaluations.

Proof. From Lemma 4.13 we know that there exists a constant $C'' > 0$ such that if $\delta \le \frac{C''}{\sqrt{\beta\log(1/s)}d^{1/4}}$, then $\hat{\pi}$ is such that $D_s^{\hat{\pi},\pi} \le \varepsilon/4$ for $s = \varepsilon/12$, i.e., $\hat{\pi}$ is warm for π. Theorem 4.16 shows that there exists a constant $C' > 0$ such that for all $K \ge C'\log(d\kappa\log(1/\varepsilon))\log(1/\varepsilon)$ we have $\|Q_{\hat{\pi}}^K - \pi\|_{\mathrm{TV}} \le \varepsilon/2$. On the other hand, for the unadjusted chain, by Theorem 4.4, there exists a constant $C_0' > 0$ such that for all $x \in \mathbb{R}^d$ and $K_0 \ge C_0'\log\left(\frac{d\kappa(1+\|x\|_\infty)}{\varepsilon}\right)$ we have $\|\hat{Q}_x^{K_0} - \hat{\pi}\|_{\mathrm{TV}} \le \varepsilon/2$. Combining these two estimates we obtain for such K and K_0 that

$$\begin{aligned}\|(Q^K \circ \hat{Q}^{K_0})_x - \pi\|_{\mathrm{TV}} &\le \|(Q^K \circ \hat{Q}^{K_0})_x - Q_{\hat{\pi}}^K\|_{\mathrm{TV}} + \|Q_{\hat{\pi}}^K - \pi\|_{\mathrm{TV}} \\ &\le \|\hat{Q}_x^{K_0} - \hat{\pi}\|_{\mathrm{TV}} + \|Q_{\hat{\pi}}^K - \pi\|_{\mathrm{TV}} \le \varepsilon,\end{aligned}$$

where we used submultiplicativity ($\|Q_\mu^K - Q_\nu^K\|_{\mathrm{TV}} \le \|\mu - \nu\|_{\mathrm{TV}}$) in the second inequality. □

4.5 Conclusions and open questions

To conclude, we studied the Hamiltonian Monte Carlo algorithm for sampling from high-dimensional Gaussian distributions, focusing on the dependency on both condition number κ and dimension d of the Gaussian. We showed that a HMC algorithm with the leapfrog integrator and long, randomized

integration times can be used to sample from a distribution ε-close to a Gaussian distribution by making only $\widetilde{O}(\sqrt{\kappa}d^{1/4}\log(1/\varepsilon))$ gradient queries. This scaling seems optimal for leapfrog HMC in both the dimension and the condition number (by well-known scaling limits [Dua+87; Nea11]).

The $\sqrt{\kappa}$-dependency also improves over similar, preceding work on leapfrog HMC that achieved at best a linear κ-dependency [MV18; Che+20]. While these works typically consider more general logconcave distributions, we feel that our work enhances the possibility of obtaining a similar $\sqrt{\kappa}$-dependency for such distributions as well. This would disprove the $\Omega(\kappa)$ versus $O(\sqrt{\kappa})$ gap that was suggested by Lee, Shen and Tian [LST20] between logconcave sampling and convex optimization, respectively.

4.6 Omitted proofs

4.6.1 Cosine contracts on average

Lemma 4.1. *Let $\mathbf{t} \in \mathcal{T}^K$, $\omega > 0$, $x \in \mathbb{R}$, and consider*

$$z = x\left(\prod_{k=1}^{K}\cos(\omega t_k)\right) + \frac{1}{\omega}\sum_{k=1}^{K} v^{(k)}\sin(\omega t_k)\left(\prod_{j=k+1}^{K}\cos(\omega t_j)\right)$$

where $v^{(k)} \sim \mathcal{N}(0,1)$ for each $k \in [K]$. Then $z \sim \mathcal{N}(x\prod_{k=1}^{K}\cos(\omega t_k), \frac{1}{\omega^2}(1-\prod_{k=1}^{K}\cos(\omega t_j)^2))$.

Proof. It is clear that $\mathbb{E}[z] = x\prod_{k=1}^{K}\cos(\omega t_k)$. The sum of Gaussian random variables is again distributed according to a Gaussian whose variance is the sum of the individual variances. That is,

$$\begin{aligned}
\mathbb{E}[(z-\mathbb{E}[z])^2] &= \frac{1}{\omega^2}\sum_{k=1}^{K}\sin(\omega t_k)^2\left(\prod_{j=k+1}^{K}\cos(\omega t_j)^2\right)\\
&= \frac{1}{\omega^2}\sum_{k=1}^{K}(1-\cos(\omega t_k)^2)\left(\prod_{j=k+1}^{K}\cos(\omega t_j)^2\right)\\
&= \frac{1-\prod_{j=1}^{K}\cos(\omega t_j)^2}{\omega^2}.
\end{aligned}$$

□

Lemma 4.18. *Let $0 < \sqrt{\alpha} < \sqrt{\beta}$. There exists a universal constant $c > 0$ such that for all $0 < \delta \le \frac{c}{\sqrt{\beta}}$ and*

$$\mathcal{T}_\delta = \{k\cdot\delta \mid k \in \mathbb{N},\ k\cdot\delta < \frac{10\pi}{\sqrt{\alpha}}\}$$

we have for all $\omega \in [\sqrt{\alpha}, \sqrt{\beta}]$ that

$$\mathbb{P}_{t\sim U(\mathcal{T}_\delta)}\left[|\cos(\omega t)| \le 0.9\right] \ge 1/2.$$

Proof. Let ω be such that $\sqrt{\alpha} \le \omega \le \sqrt{\beta}$. Note that $|\cos(\omega t)|$ is periodic with period $\frac{\pi}{2\omega}$. We write $\mathcal{T}_\delta$ as the disjoint union

$$\mathcal{T}_\delta = \bigcup_{n=1}^{N}\left(\mathcal{T}_\delta \cap [\frac{(n-1)\pi}{2\omega}, \frac{n\pi}{2\omega}]\right)$$

where N is the least integer such that $\frac{N\pi}{2\omega} > \frac{10\pi}{\sqrt{\alpha}}$, i.e., $N = \left\lfloor \frac{20\omega}{\sqrt{\alpha}} \right\rfloor$. Note that $N \geq 20$. For $\delta \leq c/\sqrt{\beta}$, the first $N-1$ such intervals contain at least

$$\left\lfloor \frac{\pi}{2\omega\delta} \right\rfloor \geq \left\lfloor \frac{\pi\sqrt{\beta}}{2\omega c} \right\rfloor \geq \left\lfloor \frac{\pi}{2c} \right\rfloor$$

equally spaced points. For small enough $c > 0$, Lemma 4.19 shows that for each of these $N-1$ intervals we have

$$\mathbb{P}_{t\sim U(\mathcal{T}_\delta \cap [\frac{(n-1)\pi}{2\omega}, \frac{n\pi}{2\omega}])} \left[|\cos(\omega t)| \leq 0.9 \right] \geq \frac{20}{19}\frac{1}{2}.$$

We thus have

$$\mathbb{P}_{t\sim U(\mathcal{T}_\delta)} \left[|\cos(\omega t)| \leq 0.9 \right] \geq \frac{19}{20}\frac{20}{19}\frac{1}{2} = \frac{1}{2}.$$

□

Lemma 4.19. *Let $\zeta > \eta \geq 0$, and $\mathcal{T} = \{\eta + n\zeta : n \in \mathbb{N}, \eta + n\zeta \leq \pi/2\}$ with $|\mathcal{T}| \geq 10$. Then for t chosen uniformly from $\mathcal{T}$ we have*

$$\mathbb{P}_{t\sim U(\mathcal{T})} \left\{ |\cos(t)| \leq 0.9 \right\} \geq 3/5.$$

Proof. Since $\zeta \leq \left\lfloor \frac{\pi}{2(|\mathcal{T}|-1)} \right\rfloor$, we have

$$\begin{aligned}
\mathbb{P}_{t\sim U(\mathcal{T})} \left\{ |\cos(t)| \leq 0.9 \right\} &= \mathbb{P}_{t\sim U(\mathcal{T})} \left\{ t \geq \arccos(0.9) \right\} \\
&\geq \frac{1}{|\mathcal{T}|} \left\lfloor \frac{\pi/2 - \arccos(0.9)}{\zeta} \right\rfloor \\
&\geq \frac{1}{|\mathcal{T}|} \left\lfloor (1 - 2\arccos(0.9)/\pi)\,(|\mathcal{T}| - 1) \right\rfloor.
\end{aligned}$$

The last quantity is at least $3/5$ for $|\mathcal{T}| \geq 10$. □

4.6.2 Distances between Gaussian distributions

For multivariate mean-zero Gaussians we use the following bound that can be found in [DMR22]:

$$\|\mathcal{N}(0, \Sigma_1) - \mathcal{N}(0, \Sigma_2)\|_{\mathrm{TV}} \leq \frac{3}{2} \min \left\{ 1, \left\| \Sigma_1^{-1}\Sigma_2 - I \right\|_F \right\}. \tag{4.12}$$

We use it to bound the distance between π and $\hat{\pi}$.

Lemma 4.20. *Let π and $\hat{\pi}$ be two Gaussians that satisfy $\pi(x) \propto \exp(-x^\top \operatorname{diag}(\boldsymbol{\omega})x/2)$ and $\hat{\pi}(x) \propto \exp(-x^\top \operatorname{diag}(\hat{\boldsymbol{\omega}})x/2)$ with $\hat{\omega}_i = \omega_i\sqrt{1 - \frac{\delta^2\omega_i^2}{4}}$. Then*

$$\|\pi - \hat{\pi}\|_{\mathrm{TV}} \leq \frac{3}{8}\delta^2\sqrt{\sum_i \omega_i^4} \leq \frac{3}{8}\delta^2\beta\sqrt{d}.$$

Proof. Applying the bound Equation (4.12) to $\Sigma_1 = \operatorname{diag}(\hat{\boldsymbol{\omega}})$ and $\Sigma_2 = \operatorname{diag}(\boldsymbol{\omega})$ we get

$$\|\pi - \tilde{\pi}\|_{\mathrm{TV}} \leq \frac{3}{2}\sqrt{\sum_i \left(\left(1 - \frac{\delta^2\omega_i^2}{4} \right) - 1 \right)^2} = \frac{3}{8}\delta^2\sqrt{\sum_i \omega_i^4} \leq \frac{3}{8}\delta^2\beta\sqrt{d}.$$

□

Bibliography

[Aar15] Scott Aaronson. "Read the Fine Print". In: *Nature Physics* 11.4 (Apr. 2015), pp. 291–293. ISSN: 1745-2473, 1745-2481. DOI: 10.1038/nphys3272.

[AG03] F. Alizadeh and D. Goldfarb. "Second-Order Cone Programming". In: *Mathematical Programming* 95.1 (Jan. 1, 2003), pp. 3–51. ISSN: 1436-4646. DOI: 10.1007/s10107-002-0339-5.

[AG19] Joran van Apeldoorn and András Gilyén. "Improvements in Quantum SDP-Solving with Applications". In: *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Ed. by Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi. Vol. 132. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019, 99:1–99:15. ISBN: 978-3-95977-109-2. DOI: 10.4230/LIPIcs.ICALP.2019.99.

[AH20] Jonathan Allcock and Chang-Yu Hsieh. "A Quantum Extension of SVM-perf for Training Nonlinear SVMs in Almost Linear Time". In: *Quantum* 4 (Oct. 15, 2020), p. 342. ISSN: 2521-327X. DOI: 10.22331/q-2020-10-15-342. arXiv: 2006.10299 [quant-ph].

[AHK12] Sanjeev Arora, Elad Hazan, and Satyen Kale. "The Multiplicative Weights Update Method: A Meta-Algorithm and Applications". In: *Theory of Computing* 8.6 (May 1, 2012), pp. 121–164. DOI: 10.4086/toc.2012.v008a006.

[AL22] Dong An and Lin Lin. "Quantum Linear System Solver Based on Time-optimal Adiabatic Quantum Computing and Quantum Approximate Optimization Algorithm". In: *ACM Transactions on Quantum Computing* 3.2 (June 30, 2022), pp. 1–28. ISSN: 2643-6809, 2643-6817. DOI: 10.1145/3498331.

[Amb12] Andris Ambainis. "Variable Time Amplitude Amplification and Quantum Algorithms for Linear Algebra Problems". In: *29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012)*. Ed. by Christoph Dürr and Thomas Wilke. Vol. 14. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2012, pp. 636–647. ISBN: 978-3-939897-35-4. DOI: 10.4230/LIPIcs.STACS.2012.636.

[BCK15] Dominic W. Berry, Andrew M. Childs, and Robin Kothari. "Hamiltonian Simulation with Nearly Optimal Dependence on All Parameters". In: *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*. 2015 IEEE 56th Annual Symposium on Foundations of Computer Science. Oct. 2015, pp. 792–809. DOI: 10.1109/FOCS.2015.54.

[Bea+01] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. "Quantum Lower Bounds by Polynomials". In: *Journal of the ACM* 48.4 (July 1, 2001), pp. 778–797. ISSN: 0004-5411. DOI: 10.1145/502090.502097.

[Bes+13] Alexandros Beskos, Natesh Pillai, Gareth Roberts, Jesus-Maria Sanz-Serna, and Andrew Stuart. "Optimal Tuning of the Hybrid Monte Carlo Algorithm". In: *Bernoulli* 19 (5A Nov. 2013), pp. 1501–1534. ISSN: 1350-7265. DOI: 10.3150/12-BEJ414.

[Bes94] Julian Besag. "Comments on "Representations of Knowledge in Complex Systems" by U. Grenander and MI Miller". In: *J. Roy. Statist. Soc. Ser. B* 56.591-592 (1994), p. 4.

[BKF22] Fernando G. S. L. Brandão, Richard Kueng, and Daniel Stilck França. "Faster Quantum and Classical SDP Approximations for Quadratic Binary Optimization". In: *Quantum* 6 (Jan. 20, 2022), p. 625. ISSN: 2521-327X. DOI: 10.22331/q-2022-01-20-625. arXiv: 1909.04613 [quant-ph].

[BL17] Daniel J. Bernstein and Tanja Lange. "Post-Quantum Cryptography". In: *Nature* 549.7671 (Sept. 2017), pp. 188–194. ISSN: 0028-0836, 1476-4687. DOI: 10.1038/nature23461.

[BN01] Aharon Ben-Tal and Arkadi Nemirovski. *Lectures on Modern Convex Optimization: Analysis, Algorithms, and Engineering Applications*. Society for Industrial and Applied Mathematics, Jan. 2001. ISBN: 978-0-89871-491-3. DOI: 10.1137/1.9780898718829.

[Boy+17] Stephen Boyd, Enzo Busseti, Steve Diamond, Ronald N. Kahn, Kwangmoo Koh, Peter Nystrup, and Jan Speth. "Multi-Period Trading via Convex Optimization". In: *Foundations and Trends® in Optimization* 3.1 (Aug. 7, 2017), pp. 1–76. ISSN: 2167-3888, 2167-3918. DOI: 10.1561/2400000023.

[BP14] Árpád Baricz and Tibor K. Pogány. "On a Sum of Modified Bessel Functions". In: *Mediterranean Journal of Mathematics* 11.2 (May 1, 2014), pp. 349–360. ISSN: 1660-5454. DOI: 10.1007/s00009-013-0365-y.

[Bra+02] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. "Quantum Amplitude Amplification and Estimation". In: *Contemporary Mathematics*. Ed. by Samuel J. Lomonaco and Howard E. Brandt. Vol. 305. Providence, Rhode Island: American Mathematical Society, 2002, pp. 53–74. ISBN: 978-0-8218-2140-4. DOI: 10.1090/conm/305/05215.

[Bra+19] Fernando G. S. L. Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta M. Svore, and Xiaodi Wu. "Quantum SDP Solvers: Large Speed-Ups, Optimality, and Applications to Quantum Learning". In: *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Ed. by Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi. Vol. 132. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019, 27:1–27:14. ISBN: 978-3-95977-109-2. DOI: 10.4230/LIPIcs.ICALP.2019.27.

[BS17a] Nawaf Bou-Rabee and Jesús María Sanz-Serna. "Randomized Hamiltonian Monte Carlo". In: *The Annals of Applied Probability* 27.4 (Aug. 2017), pp. 2159–2194. ISSN: 1050-5164, 2168-8737. DOI: 10.1214/16-AAP1255.

[BS17b] Fernando G.S.L. Brandao and Krysta M. Svore. "Quantum Speed-Ups for Solving Semidefinite Programs". In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS). Oct. 2017, pp. 415–426. DOI: 10.1109/FOCS.2017.45.

[Bub15] Sébastien Bubeck. "Convex Optimization: Algorithms and Complexity". In: *Foundations and Trends® in Machine Learning* 8.3-4 (2015), pp. 231–357. ISSN: 1935-8237, 1935-8245. DOI: 10.1561/2200000050.

[BV04] Stephen P. Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge, UK ; New York: Cambridge University Press, 2004. 716 pp. ISBN: 978-0-521-83378-3.

[BV97] Ethan Bernstein and Umesh Vazirani. "Quantum Complexity Theory". In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1411–1473. ISSN: 0097-5397. DOI: 10.1137/S0097539796300921.

[CGJ19] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. "The Power of Block-Encoded Matrix Powers: Improved Regression Techniques via Faster Hamiltonian Simulation". In: *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Ed. by Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi. Vol. 132. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019, 33:1–33:14. ISBN: 978-3-95977-109-2. DOI: 10.4230/LIPIcs.ICALP.2019.33.

[Cha+20] Rui Chao, Dawei Ding, Andras Gilyen, Cupjin Huang, and Mario Szegedy. *Finding Angles for Quantum Signal Processing with Machine Precision*. Mar. 8, 2020. DOI: 10.48550/arXiv.2003.02831. arXiv: 2003.02831 [quant-ph].

[Che+20] Yuansi Chen, Raaz Dwivedi, Martin J. Wainwright, and Bin Yu. "Fast Mixing of Metropolized Hamiltonian Monte Carlo: Benefits of Multi-Step Gradients". In: *Journal of Machine Learning Research* 21.92 (2020), pp. 1–72. ISSN: 1533-7928. URL: http://jmlr.org/papers/v21/19-441.html.

[Che+21] Sinho Chewi, Chen Lu, Kwangjun Ahn, Xiang Cheng, Thibaut Le Gouic, and Philippe Rigollet. "Optimal Dimension Dependence of the Metropolis-Adjusted Langevin Algorithm". In: *Proceedings of Thirty Fourth Conference on Learning Theory*. Conference on Learning Theory. PMLR, July 21, 2021, pp. 1260–1300. URL: https://proceedings.mlr.press/v134/chewi21a.html.

[CKS17] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. "Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision". In: *SIAM Journal on Computing* 46.6 (Jan. 2017), pp. 1920–1950. ISSN: 0097-5397. DOI: 10.1137/16M1087072.

[CL11] Chih-Chung Chang and Chih-Jen Lin. "LIBSVM: A Library for Support Vector Machines". In: *ACM Transactions on Intelligent Systems and Technology* 2.3 (Apr. 2011), pp. 1–27. ISSN: 2157-6904, 2157-6912. DOI: 10.1145/1961189.1961199.

[CLS19] Michael B. Cohen, Yin Tat Lee, and Zhao Song. "Solving Linear Programs in the Current Matrix Multiplication Time". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. STOC 2019. New York, NY, USA: Association for Computing Machinery, June 23, 2019, pp. 938–942. ISBN: 978-1-4503-6705-9. DOI: 10.1145/3313276.3316303.

[Cos+21] Pedro C. S. Costa, Dong An, Yuval R. Sanders, Yuan Su, Ryan Babbush, and Dominic W. Berry. *Optimal Scaling Quantum Linear Systems Solver via Discrete Adiabatic Theorem*. Nov. 15, 2021. DOI: 10.48550/arXiv.2111.08152. arXiv: 2111.08152 [quant-ph].

[CPT18] Gérard Cornuéjols, Javier Peña, and Reha Tütüncü. *Optimization Methods in Finance*. 2nd ed. Cambridge: Cambridge University Press, 2018. ISBN: 978-1-107-05674-9. DOI: 10.1017/9781107297340.

[CV22] Zongchen Chen and Santosh S. Vempala. "Optimal Convergence Rate of Hamiltonian Monte Carlo for Strongly Logconcave Distributions". In: *Theory of Computing* 18.9 (Apr. 30, 2022), pp. 1–18. DOI: 10.4086/toc.2022.v018a009.

[DCB13] Alexander Domahidi, Eric Chu, and Stephen Boyd. "ECOS: An SOCP Solver for Embedded Systems". In: *2013 European Control Conference (ECC)*. 2013 European Control Conference (ECC). July 2013, pp. 3071–3076. DOI: 10.23919/ECC.2013.6669541.

[Del+21] George Deligiannidis, Daniel Paulin, Alexandre Bouchard-Côté, and Arnaud Doucet. "Randomized Hamiltonian Monte Carlo as Scaling Limit of the Bouncy Particle Sampler and Dimension-Free Convergence Rates". In: *The Annals of Applied Probability* 31.6 (Dec. 2021), pp. 2612–2662. ISSN: 1050-5164, 2168-8737. DOI: 10.1214/20-AAP1659.

[DLMF] F. W. J. Olver, A. B. Olde Daalhuis, D. W. Lozier, B. I. Schneider, R. F. Boisvert, C. W. Clark, B. R. Miller, B. V. Saunders, H. S. Cohl, and M. A. McClain, eds. *NIST Digital Library of Mathematical Functions*. Release 1.1.5. Mar. 15, 2022. URL: http://dlmf.nist.gov/.

[DMR22] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. *The Total Variation Distance between High-Dimensional Gaussians with the Same Mean*. Feb. 26, 2022. arXiv: 1810.08693 [math, stat]. URL: http://arxiv.org/abs/1810.08693.

[Dol05] Hilary Dollar. "Iterative Linear Algebra for Constrained Optimization". PhD thesis. University of Oxford, 2005.

[Don+21] Yulong Dong, Xiang Meng, K. Birgitta Whaley, and Lin Lin. "Efficient Phase-Factor Evaluation in Quantum Signal Processing". In: *Physical Review A* 103.4 (Apr. 22, 2021), p. 042419. DOI: 10.1103/PhysRevA.103.042419.

[DP85] David Deutsch and Roger Penrose. "Quantum Theory, the Church–Turing Principle and the Universal Quantum Computer". In: *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* 400.1818 (July 8, 1985), pp. 97–117. DOI: 10.1098/rspa.1985.0070.

[DT09] Sebastian Dörn and Thomas Thierauf. "The Quantum Query Complexity of the Determinant". In: *Information Processing Letters* 109.6 (Feb. 28, 2009), pp. 325–328. ISSN: 0020-0190. DOI: 10.1016/j.ipl.2008.11.006.

[Dua+87] Simon Duane, A.D. Kennedy, Brian J. Pendleton, and Duncan Roweth. "Hybrid Monte Carlo". In: *Physics Letters B* 195.2 (Sept. 1987), pp. 216–222. ISSN: 03702693. DOI: 10.1016/0370-2693(87)91197-X.

[Dwi+18] Raaz Dwivedi, Yuansi Chen, Martin J. Wainwright, and Bin Yu. "Log-Concave Sampling: Metropolis-Hastings Algorithms Are Fast!" In: *Proceedings of the 31st Conference On Learning Theory*. Conference On Learning Theory. PMLR, July 3, 2018, pp. 793–797. URL: https://proceedings.mlr.press/v75/dwivedi18a.html.

[Fan+08] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. "LIBLINEAR: A Library for Large Linear Classification". In: *The Journal of Machine Learning Research* 9 (June 1, 2008), pp. 1871–1874. ISSN: 1532-4435.

[GAW19] András Gilyén, Srinivasan Arunachalam, and Nathan Wiebe. "Optimizing Quantum Optimization Algorithms via Faster Quantum Gradient Computation". In: *Proceedings of the 2019 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. Proceedings. Society for Industrial and Applied Mathematics, Jan. 2019, pp. 1425–1444. DOI: 10.1137/1.9781611975482.87.

[Gen72] W. Morven Gentleman. "Implementing Clenshaw-Curtis Quadrature, II Computing the Cosine Transformation". In: *Communications of the ACM* 15.5 (May 1972), pp. 343–346. ISSN: 0001-0782, 1557-7317. DOI: 10.1145/355602.361311.

[Gil+19] András Gilyén, Yuan Su, Guang Hao Low, and Nathan Wiebe. "Quantum Singular Value Transformation and beyond: Exponential Improvements for Quantum Matrix Arithmetics". In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. STOC '19: 51st Annual ACM SIGACT Symposium on the Theory of Computing. Phoenix AZ USA: ACM, June 23, 2019, pp. 193–204. ISBN: 978-1-4503-6705-9. DOI: 10.1145/3313276.3316366.

[Gis+02] Nicolas Gisin, Grégoire Ribordy, Wolfgang Tittel, and Hugo Zbinden. "Quantum Cryptography". In: *Reviews of Modern Physics* 74.1 (Mar. 8, 2002), pp. 145–195. DOI: 10.1103/RevModPhys.74.145.

[GLT18] András Gilyén, Seth Lloyd, and Ewin Tang. *Quantum-Inspired Low-Rank Stochastic Regression with Logarithmic Dependence on the Dimension*. Nov. 12, 2018. DOI: 10.48550/arXiv.1811.04909. arXiv: 1811.04909 [quant-ph].

[Gro96] Lov K. Grover. "A Fast Quantum Mechanical Algorithm for Database Search". In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. STOC '96. New York, NY, USA: Association for Computing Machinery, July 1, 1996, pp. 212–219. ISBN: 978-0-89791-785-8. DOI: 10.1145/237814.237866.

[Haa19] Jeongwan Haah. "Product Decomposition of Periodic Functions in Quantum Signal Processing". In: *Quantum* 3 (Oct. 7, 2019), p. 190. DOI: 10.22331/q-2019-10-07-190.

[HHL09] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. "Quantum Algorithm for Linear Systems of Equations". In: *Physical Review Letters* 103.15 (Oct. 7, 2009), p. 150502. DOI: 10.1103/PhysRevLett.103.150502.

[HW71] D. L. Hanson and F. T. Wright. "A Bound on Tail Probabilities for Quadratic Forms in Independent Random Variables". In: *The Annals of Mathematical Statistics* 42.3 (June 1971), pp. 1079–1083. ISSN: 0003-4851, 2168-8990. DOI: 10.1214/aoms/1177693335.

[Ji+21] Zhengfeng Ji, Anand Natarajan, Thomas Vidick, John Wright, and Henry Yuen. "MIP* = RE". In: *Communications of the ACM* 64.11 (Oct. 25, 2021), pp. 131–138. ISSN: 0001-0782. DOI: 10.1145/3485628.

[Joa06] Thorsten Joachims. "Training Linear SVMs in Linear Time". In: *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '06. New York, NY, USA: Association for Computing Machinery, Aug. 20, 2006, pp. 217–226. ISBN: 978-1-59593-339-3. DOI: 10.1145/1150402.1150429.

[Kar84] N. Karmarkar. "A New Polynomial-Time Algorithm for Linear Programming". In: *Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing*. STOC '84. New York, NY, USA: Association for Computing Machinery, Dec. 1, 1984, pp. 302–311. ISBN: 978-0-89791-133-7. DOI: 10.1145/800057.808695.

[KN12] Ravindran Kannan and Hariharan Narayanan. "Random Walks on Polytopes and an Affine Interior Point Method for Linear Programming". In: *Mathematics of Operations Research* 37.1 (Feb. 2012), pp. 1–20. ISSN: 0364-765X. DOI: 10.1287/moor.1110.0519.

[KP17] Iordanis Kerenidis and Anupam Prakash. "Quantum Recommendation Systems". In: *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Ed. by Christos H. Papadimitriou. Vol. 67. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017, 49:1–49:21. ISBN: 978-3-95977-029-3. DOI: 10.4230/LIPIcs.ITCS.2017.49.

[KP20a] Iordanis Kerenidis and Anupam Prakash. "A Quantum Interior Point Method for LPs and SDPs". In: *ACM Transactions on Quantum Computing* (Oct. 2, 2020). DOI: 10.1145/3406306.

[KP20b] Iordanis Kerenidis and Anupam Prakash. "Quantum Gradient Descent for Linear Systems and Least Squares". In: *Physical Review A* 101.2 (Feb. 14, 2020), p. 022316. DOI: 10.1103/PhysRevA.101.022316.

[KP91] A. D. Kennedy and Brian Pendleton. "Acceptances and Autocorrelations in Hybrid Monte Carlo". In: *Nuclear Physics B - Proceedings Supplements* 20 (May 20, 1991), pp. 118–121. ISSN: 0920-5632. DOI: 10.1016/0920-5632(91)90893-J.

[KPS21a] Iordanis Kerenidis, Anupam Prakash, and Dániel Szilágyi. "Quantum Algorithms for Second-Order Cone Programming and Support Vector Machines". In: *Quantum* 5 (Apr. 8, 2021), p. 427. ISSN: 2521-327X. DOI: 10.22331/q-2021-04-08-427. arXiv: 1908.06720 [quant-ph, stat].

[KPS21b] Iordanis Kerenidis, Anupam Prakash, and Dániel Szilágyi. *Quantum SVM via SOCP Experiment Logs*. figshare, Mar. 16, 2021. DOI: 10.6084/m9.figshare.11778189.v1.

[LB13] Daniel A. Lidar and Todd A. Brun. *Quantum Error Correction*. Cambridge University Press, Sept. 12, 2013. 689 pp. ISBN: 978-0-521-89787-7. Google Books: XV9sAAAAQBAJ.

[LC17a] Guang Hao Low and Isaac L. Chuang. *Hamiltonian Simulation by Uniform Spectral Amplification*. July 17, 2017. DOI: 10.48550/arXiv.1707.05391. arXiv: 1707.05391 [quant-ph].

[LC17b] Guang Hao Low and Isaac L. Chuang. "Optimal Hamiltonian Simulation by Quantum Signal Processing". In: *Physical Review Letters* 118.1 (Jan. 5, 2017), p. 010501. DOI: 10.1103/PhysRevLett.118.010501.

[LC19] Guang Hao Low and Isaac L. Chuang. "Hamiltonian Simulation by Qubitization". In: *Quantum* 3 (July 12, 2019), p. 163. ISSN: 2521-327X. DOI: 10.22331/q-2019-07-12-163. arXiv: 1610.06546 [quant-ph].

[Lev+17] David Asher Levin, Y. Peres, Elizabeth L. Wilmer, James Propp, and David B. Wilson. *Markov Chains and Mixing Times*. Second edition. Providence, Rhode Island: American Mathematical Society, 2017. 447 pp. ISBN: 978-1-4704-2962-1.

[LMS20] Noah Linden, Ashley Montanaro, and Changpeng Shao. *Quantum vs. Classical Algorithms for Solving the Heat Equation*. June 18, 2020. DOI: 10.48550/arXiv.2004.06516. arXiv: 2004.06516 [quant-ph].

[LR05] Benedict Leimkuhler and Sebastian Reich. *Simulating Hamiltonian Dynamics*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge: Cambridge University Press, 2005. ISBN: 978-0-521-77290-7. DOI: 10.1017/CBO9780511614118.

[LST20] Yin Tat Lee, Ruoqi Shen, and Kevin Tian. "Logsmooth Gradient Concentration and Tighter Runtimes for Metropolized Hamiltonian Monte Carlo". In: *Proceedings of Thirty Third Conference on Learning Theory*. Conference on Learning Theory. PMLR, July 15, 2020, pp. 2565–2597. URL: https://proceedings.mlr.press/v125/lee20b.html.

[LST21] Yin Tat Lee, Ruoqi Shen, and Kevin Tian. "Lower Bounds on Metropolized Sampling Methods for Well-Conditioned Distributions". In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., 2021, pp. 18812–18824. URL: https://papers.nips.cc/paper/2021/hash/9c4e6233c6d5ff637e7984152a3531d5-Abstract.html.

[LSZ19] Yin Tat Lee, Zhao Song, and Qiuyi Zhang. "Solving Empirical Risk Minimization in the Current Matrix Multiplication Time". In: *Proceedings of the Thirty-Second Conference on Learning Theory*. Conference on Learning Theory. PMLR, June 25, 2019, pp. 2140–2157. URL: https://proceedings.mlr.press/v99/lee19a.html.

[LT20] Lin Lin and Yu Tong. "Optimal Polynomial Based Quantum Eigenstate Filtering with Application to Solving Quantum Linear Systems". In: *Quantum* 4 (Nov. 11, 2020), p. 361. ISSN: 2521-327X. DOI: 10.22331/q-2020-11-11-361. arXiv: 1910.14596 [quant-ph].

[LW22] Jianfeng Lu and Lihan Wang. "On Explicit L2-convergence Rate Estimate for Piecewise Deterministic Markov Processes in MCMC Algorithms". In: *The Annals of Applied Probability* 32.2 (Apr. 2022), pp. 1333–1361. ISSN: 1050-5164, 2168-8737. DOI: 10.1214/21-AAP1710.

[LYC16] Guang Hao Low, Theodore J. Yoder, and Isaac L. Chuang. "Methodology of Resonant Equiangular Composite Quantum Gates". In: *Physical Review X* 6.4 (Dec. 28, 2016), p. 041067. DOI: 10.1103/PhysRevX.6.041067.

[Mar+21] John M. Martyn, Zane M. Rossi, Andrew K. Tan, and Isaac L. Chuang. "Grand Unification of Quantum Algorithms". In: *PRX Quantum* 2.4 (Dec. 3, 2021), p. 040203. DOI: 10.1103/PRXQuantum.2.040203.

[Mar52] Harry Markowitz. "Portfolio Selection". In: *The Journal of Finance* 7.1 (1952), pp. 77–91. ISSN: 1540-6261. DOI: 10.1111/j.1540-6261.1952.tb01525.x.

[Mou+21] Wenlong Mou, Yi-An Ma, Martin J. Wainwright, Peter L. Bartlett, and Michael I. Jordan. "High-Order Langevin Diffusion Yields an Accelerated MCMC Algorithm". In: *Journal of Machine Learning Research* 22.42 (2021), pp. 1–41. ISSN: 1533-7928. URL: http://jmlr.org/papers/v22/20-576.html.

[MT00] Renato D.C. Monteiro and Takashi Tsuchiya. "Polynomial Convergence of Primal-Dual Algorithms for the Second-Order Cone Program Based on the MZ-family of Directions". In: *Mathematical Programming* 88.1 (June 1, 2000), pp. 61–83. ISSN: 1436-4646. DOI: 10.1007/PL00011378.

[MV18] Oren Mangoubi and Nisheeth Vishnoi. "Dimensionally Tight Bounds for Second-Order Hamiltonian Monte Carlo". In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018. URL: https://papers.nips.cc/paper/2018/hash/e07bceab69529b0f0b43625953fbf2a0-Abstract.html.

[Nar16] Hariharan Narayanan. "Randomized Interior Point Methods for Sampling and Optimization". In: *The Annals of Applied Probability* 26.1 (Feb. 2016), pp. 597–641. ISSN: 1050-5164, 2168-8737. DOI: 10.1214/15-AAP1104.

[NC12] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. 1st ed. Cambridge University Press, June 5, 2012. ISBN: 978-1-107-00217-3. DOI: 10.1017/CBO9780511976667.

[Nea11] Radford M. Neal. *MCMC Using Hamiltonian Dynamics*. May 10, 2011. DOI: 10.1201/b10905. arXiv: 1206.1901 [physics, stat].

[Nea96] Radford M. Neal. *Bayesian Learning for Neural Networks*. Red. by P. Bickel, P. Diggle, S. Fienberg, K. Krickeberg, I. Olkin, N. Wermuth, and S. Zeger. Vol. 118. Lecture Notes in Statistics. New York, NY: Springer New York, 1996. ISBN: 978-0-387-94724-2. DOI: 10.1007/978-1-4612-0745-0.

[Net+96] John Neter, Michael H. Kutner, Christopher J. Nachtsheim, and William Wasserman. *Applied Linear Regression Models*. Irwin, 1996. 750 pp. ISBN: 978-0-256-08601-0. Google Books: 4CrvAAAAMAAJ.

[NT97] Yu. E. Nesterov and M. J. Todd. "Self-Scaled Barriers and Interior-Point Methods for Convex Programming". In: *Mathematics of Operations Research* 22.1 (Feb. 1997), pp. 1–42. ISSN: 0364-765X. DOI: 10.1287/moor.22.1.1.

[NT98] Yu. E. Nesterov and M. J. Todd. "Primal-Dual Interior-Point Methods for Self-Scaled Cones". In: *SIAM Journal on Optimization* 8.2 (May 1998), pp. 324–364. ISSN: 1052-6234. DOI: 10.1137/S1052623495290209.

[OD21] Davide Orsucci and Vedran Dunjko. "On Solving Classes of Positive-Definite Quantum Linear Systems with Quadratically Improved Runtime in the Condition Number". In: *Quantum* 5 (Nov. 8, 2021), p. 573. ISSN: 2521-327X. DOI: 10.22331/q-2021-11-08-573. arXiv: 2101.11868 [quant-ph].

[Pol87] Boris T. Polyak. *Introduction to optimization*. Translations series in mathematics and engineering. New York: Optimization Software, Publications Division, 1987. 438 pp. ISBN: 978-0-911575-14-9.

[Pre21] John Preskill. *Quantum Computing 40 Years Later*. June 25, 2021. DOI: 10.48550/arXiv.2106.10522. arXiv: 2106.10522 [quant-ph].

[PT09] Ricardo Pachón and Lloyd N. Trefethen. "Barycentric-Remez Algorithms for Best Polynomial Approximation in the Chebfun System". In: *BIT Numerical Mathematics* 49.4 (Oct. 10, 2009), p. 721. ISSN: 1572-9125. DOI: 10.1007/s10543-009-0240-1.

[RL18] Patrick Rebentrost and Seth Lloyd. *Quantum Computational Finance: Quantum Algorithm for Portfolio Optimization*. Nov. 9, 2018. DOI: 10.48550/arXiv.1811.03975. arXiv: 1811.03975 [quant-ph].

[RML14] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. "Quantum Support Vector Machine for Big Data Classification". In: *Physical Review Letters* 113.13 (Sept. 25, 2014), p. 130503. DOI: 10.1103/PhysRevLett.113.130503.

[RR98] Gareth O. Roberts and Jeffrey S. Rosenthal. "Optimal Scaling of Discrete Approximations to Langevin Diffusions". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 60.1 (Feb. 1998), pp. 255–268. ISSN: 1369-7412, 1467-9868. DOI: 10.1111/1467-9868.00123.

[RT96] Gareth O. Roberts and Richard L. Tweedie. "Exponential Convergence of Langevin Distributions and Their Discrete Approximations". In: *Bernoulli* 2.4 (Dec. 1996), p. 341. ISSN: 13507265. DOI: 10.2307/3318418. JSTOR: 3318418.

[Rus+18] Daniel J. Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. "A Tutorial on Thompson Sampling". In: *Foundations and Trends® in Machine Learning* 11.1 (July 11, 2018), pp. 1–96. ISSN: 1935-8237, 1935-8245. DOI: 10.1561/2200000070.

[SA22] Seyran Saeedi and Tom Arodz. *Quantum Sparse Support Vector Machines*. Apr. 22, 2022. DOI: 10.48550/arXiv.1902.01879. arXiv: 1902.01879 [quant-ph, stat].

[Saa03] Yousef Saad. *Iterative Methods for Sparse Linear Systems*. Second. Society for Industrial and Applied Mathematics, Jan. 2003. ISBN: 978-0-89871-534-7. DOI: 10.1137/1.9780898718003.

[Sho94] P.W. Shor. "Algorithms for Quantum Computation: Discrete Logarithms and Factoring". In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. Proceedings 35th Annual Symposium on Foundations of Computer Science. Nov. 1994, pp. 124–134. DOI: 10.1109/SFCS.1994.365700.

[Sim97] Daniel R. Simon. "On the Power of Quantum Computation". In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1474–1483. ISSN: 0097-5397. DOI: 10.1137/S0097539796298637.

[SSO19] Yiğit Subaşı, Rolando D. Somma, and Davide Orsucci. "Quantum Algorithms for Systems of Linear Equations Inspired by Adiabatic Quantum Computing". In: *Physical Review Letters* 122.6 (Feb. 14, 2019), p. 060504. DOI: 10.1103/PhysRevLett.122.060504.

[Str69] Volker Strassen. "Gaussian Elimination Is Not Optimal". In: *Numerische Mathematik* 13.4 (Aug. 1, 1969), pp. 354–356. ISSN: 0945-3245. DOI: 10.1007/BF02165411.

[Suy+02] J. A. K. Suykens, J. De Brabanter, L. Lukas, and J. Vandewalle. "Weighted Least Squares Support Vector Machines: Robustness and Sparse Approximation". In: *Neurocomputing* 48.1 (Oct. 1, 2002), pp. 85–105. ISSN: 0925-2312. DOI: 10.1016/S0925-2312(01)00644-0.

[SV14] Sushant Sachdeva and Nisheeth K. Vishnoi. "Faster Algorithms via Approximation Theory". In: *Foundations and Trends® in Theoretical Computer Science* 9.2 (Mar. 27, 2014), pp. 125–210. ISSN: 1551-305X, 1551-3068. DOI: 10.1561/0400000065.

[SV16] Sushant Sachdeva and Nisheeth K. Vishnoi. "The Mixing Time of the Dikin Walk in a Polytope—A Simple Proof". In: *Operations Research Letters* 44.5 (Sept. 1, 2016), pp. 630–634. ISSN: 0167-6377. DOI: 10.1016/j.orl.2016.07.005.

[SV99] J.A.K. Suykens and J. Vandewalle. "Least Squares Support Vector Machine Classifiers". In: *Neural Processing Letters* 9.3 (June 1, 1999), pp. 293–300. ISSN: 1573-773X. DOI: 10.1023/A:1018628609742.

[Sze04] Mario Szegedy. "Quantum Speed-up of Markov Chain Based Algorithms". In: *45th Annual IEEE Symposium on Foundations of Computer Science*. 45th Annual IEEE Symposium on Foundations of Computer Science. Oct. 2004, pp. 32–41. DOI: 10.1109/FOCS.2004.53.

[Ton+21] Yu Tong, Dong An, Nathan Wiebe, and Lin Lin. "Fast Inversion, Preconditioned Quantum Linear System Solvers, Fast Green's-Function Computation, and Fast Evaluation of Matrix Functions". In: *Physical Review A* 104.3 (Sept. 27, 2021), p. 032422. DOI: 10.1103/PhysRevA.104.032422.

[vApe+17] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. "Quantum SDP-Solvers: Better Upper and Lower Bounds". In: *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*. 2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS). Oct. 2017, pp. 403–414. DOI: 10.1109/FOCS.2017.44.

[Var00] Richard S. Varga. *Matrix Iterative Analysis*. Vol. 27. Springer Series in Computational Mathematics. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000. ISBN: 978-3-642-05154-8. DOI: 10.1007/978-3-642-05156-2.

[VDC22] Maxime Vono, Nicolas Dobigeon, and Pierre Chainais. "High-Dimensional Gaussian Sampling: A Review and a Unifying Approach Based on a Stochastic Proximal Point Algorithm". In: *SIAM Review* 64.1 (Feb. 3, 2022), pp. 3–56. ISSN: 0036-1445. DOI: 10.1137/20M1371026.

[Ver18] Roman Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge: Cambridge University Press, 2018. ISBN: 978-1-108-41519-4. DOI: 10.1017/9781108231596.

[Vis21] Nisheeth K. Vishnoi. *An Introduction to Hamiltonian Monte Carlo Method for Sampling*. Aug. 26, 2021. arXiv: 2108.12107 [cs, math, stat]. URL: http://arxiv.org/abs/2108.12107.

[Wat09] John Watrous. "Quantum Computational Complexity". In: *Encyclopedia of Complexity and Systems Science*. Ed. by Robert A. Meyers. New York, NY: Springer, 2009, pp. 7174–7201. ISBN: 978-0-387-30440-3. DOI: 10.1007/978-0-387-30440-3_428.

[WSC21] Keru Wu, Scott Schmidler, and Yuansi Chen. *Minimax Mixing Time of the Metropolis-Adjusted Langevin Algorithm for Log-Concave Sampling*. Sept. 27, 2021. DOI: 10.48550/arXiv.2109.13055. arXiv: 2109.13055 [cs, stat].

[YTM94] Yinyu Ye, Michael J. Todd, and Shinji Mizuno. "An O($\sqrt{n}$L)-Iteration Homogeneous and Self-Dual Linear Programming Algorithm". In: *Mathematics of Operations Research* 19.1 (Feb. 1994), pp. 53–67. ISSN: 0364-765X. DOI: 10.1287/moor.19.1.53.