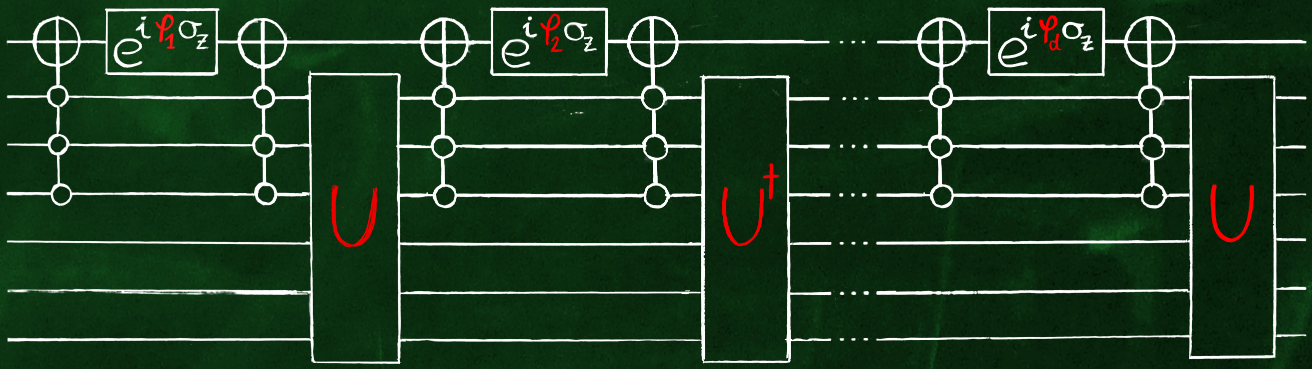
Quantum Singular Value Transformation

& Its Algorithmic Applications



András Gilyén

Quantum Singular Value Transformation & Its Algorithmic Applications

ILLC Dissertation Series DS-2019-03

INSTITUTE FOR LOGIC, LANGUAGE AND COMPUTATION

For further information about ILLC-publications, please contact

Institute for Logic, Language and Computation
Universiteit van Amsterdam
Science Park 107
1098 XG Amsterdam
phone: +31-20-525 6051
e-mail: `illc@uva.nl`
homepage: `http://www.illc.uva.nl/`

The investigations were supported by ERC Consolidator Grant QPROGRESS and partially supported by QuantERA project QuantAlgo 680-91-034.

Printed and bound by Ipskamp Printing.

ISBN: 978-94-028-1509-2

Quantum Singular Value Transformation & Its Algorithmic Applications

Academisch Proefschrift

ter verkrijging van de graad van doctor
aan de Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. K.I.J. Maex
ten overstaan van een door het College voor Promoties ingestelde
commissie, in het openbaar te verdedigen in de Agnietenkapel
op woensdag 29 mei 2019, te 14.00 uur

door

András Pál Gilyén

geboren te Boedapest

Ábrahám Mariannának és Gambár Katalinnak

akik bevezettek a magasabb matematika és fizika világába

This dissertation is based on the following papers. The dissertation's author is the main contributor of the papers where he is the first author. For the other papers where the authors are ordered alphabetically the co-authorship is shared equally.

[GSLW19] 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 ACM Symposium on Theory of Computing (STOC)*, 2019. (to appear) arXiv: `1806.01838`

[GAW19] András Gilyén, Srinivasan Arunachalam, and Nathan Wiebe. Optimizing quantum optimization algorithms via faster quantum gradient computation. In *Proceedings of the 30th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1425–1444, 2019. arXiv: `1711.00465`

[vAGGdW18] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Convex optimization using quantum oracles. arXiv: `1809.00643`, 2018

[vAGGdW17] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-solvers: Better upper and lower bounds. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 403–414, 2017. arXiv: `1705.01843`

[vAG19] Joran van Apeldoorn and András Gilyén. Improvements in quantum SDP-solving with applications. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, 2019. (to appear) arXiv: `1804.05058`

[GS17] András Gilyén and Or Sattath. On preparing ground states of gapped Hamiltonians: An efficient quantum Lovász local lemma. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 439–450, 2017. arXiv: `1611.08571`

[Gil16] András Gilyén. Bounds for probabilities in the variable setting for trees and cycles. Unpublished manuscript, June 2016

In the course of his PhD, the author has additionally (co-)authored the following articles that are not included in this dissertation.

[GKJ15] András Gilyén, Tamás Kiss, and Igor Jex. Exponential sensitivity and its cost in quantum physics. *Scientific Reports*, 6:20076, 2015. arXiv: `1508.03191`

[Gil16b] András Gilyén. Testing quantum state engineering protocols via LIQ$Ui|\rangle$ simulations. Technical report, 2nd prize winner entry at the Microsoft Quantum Challange, 2016.

[CGJ19] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, 2019. (to appear) arXiv: `1804.01973`

[GLT18] András Gilyén, Seth Lloyd, and Ewin Tang. Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension. arXiv: `1811.04909`, 2018.

[GL19] András Gilyén and Tongyang Li. Distributional property testing in a quantum world. arXiv: `1902.00814`, 2019.

[AGJK19] Andris Ambainis, András Gilyén, Stacey Jeffery, and Martins Kokainis. Quadratic speedup for finding marked vertices by quantum walks. arXiv: `1903.07493`, 2019

[BBGSz19] Tom Bannink, Harry Buhrman, András Gilyén, and Márió Szegedy. The power light cone of the Discrete Bak-Sneppen, Contact and other local processes. arXiv: `1903.12607`, 2019.

[vAG19] Joran van Apeldoorn and András Gilyén. Quantum algorithms for zero-sum games. arXiv: `1904.03180`, 2019.

Contents

Acknowledgments **xi**

Abstract **xv**

Samenvatting **xix**

Kivonat **xxiii**

1 Introduction and preliminaries **1**
- 1.1 Computation in the physical world . . . 1
- 1.2 Quantum mechanical principles . . . 2
- 1.3 The circuit model and quantum queries . . . 3
- 1.4 Mixed quantum states and distances . . . 8
- 1.5 Hamiltonians and Gibbs states . . . 10
- 1.6 Singular Value Decomposition (SVD) . . . 11
- 1.7 Some notation . . . 11

2 Quantum singular value transformation **15**
- 2.1 Introduction . . . 15
- 2.2 Quantum signal processing . . . 19
- 2.3 Quantum singular value transformation . . . 28
- 2.4 Robustness of singular value transformation . . . 36

3 Constructing quantum algorithms by QSVT **43**
- 3.1 Introduction . . . 43
- 3.2 Some direct applications of quantum singular value transformation 49
- 3.3 Matrix arithmetics using blocks of unitaries . . . 68
- 3.4 Implementing smooth functions of Hermitian matrices . . . 79
- 3.5 Limitations of smooth function techniques . . . 98
- 3.6 Conclusion . . . 100

3.A Generalized minimum-finding algorithm 101

4 Faster quantum gradient computation 111

4.1 Introduction 112
4.2 Organization of the chapter and preliminaries 116
4.3 A generic model of quantum optimization algorithms 118
4.4 The three natural input oracle models 122
4.5 Improved quantum gradient computation algorithm 125
4.6 Lower bounds on gradient computation 138
4.7 Applications 144
4.8 Conclusion and future research 152
4.A Error bounds on central-difference formulas 152

5 Convex optimization using quantum oracles 163

5.1 Introduction 163
5.2 Preliminaries 167
5.3 Computing approximate subgradients of convex Lipschitz functions 170
5.4 Algorithms for separation using membership queries 177
5.5 Lower bounds 181
5.6 Consequences of convex polarity 188
5.7 Future work 190
5.A Quantum gradient computation using relational oracles 190

6 Quantum algorithms for solving SDPs 195

6.1 Introduction 196
6.2 SDP-solving frameworks 197
6.3 High-level overview of quantum algorithms & conceptual improvements 204
6.4 Technical improvements 213
6.5 Application to shadow tomography 227
6.6 Lower bounds for the new input models 228

7 Quantum Lovász Local Lemma 233

7.1 Introduction 234
7.2 The ideal quantum algorithm 239
7.3 Implementation of the algorithm 248
7.4 Dependency structures: trees and cycles 250
7.A Optimized upper bound on $\binom{n}{k}$ 262

Acknowledgments

First and foremost I would like to thank Ronald de Wolf, who has been a very thorough supervisor, always available for providing advice and feedback. Ronald's amazing overview of the field was an invaluable resource, helping a lot in exploring various aspects of quantum computing. Our expansive brainstorms together with his two other students, Joran van Apeldoorn and Sander Gribling were very fruitful and led to nice collaborative efforts and many interesting quantum results about convex optimization. I learned a lot from our discussions, and his detailed comments on scientific writing greatly improved the quality of the papers covered in this dissertation. I clearly remember when we first met at the poster session of TQC in Brussels, and Ronald mentioned having an open PhD position; back then I did not imagine, that this meeting would lead to such a fruitful joint journey – thank you for the opportunity!

I would also like to thank my co-supervisor Harry Buhrman, who has a vibrant attitude to research and life, and has a unique sense of humor. Harry is an excellent group leader, and his grandiose vision about QuSoft started to manifest itself just around the time I joined the Algorithms & Complexity group. His dream about QuSoft brought together an amazing community of researchers, providing a lot of inspiration, and leading to many interesting discussions.

I am grateful to Stacey Jeffery, Maris Ozols, Kareljan Schoutens, Lex Schrijver and Márió Szegedy for accepting to be members of my PhD committee.

I am indebted to Márió Szegedy, who has had a huge impact on my research interest. My first successful project was inspired by his enlightening talks on the Lovász Local Lemma, and the Moser-Tardos algorithm. It has always been very inspirational talking to him, and he generously shared with me a lot of his insights into various problems. I would like to thank Márió for all his guidance and the many nice hikes we took together around Berkeley.

I am thankful to Nathan Wiebe, who was my mentor during a summer internship at Microsoft Research. We have worked on some very interesting research problems together, which have become a vital part of this dissertation. It has been a wonderful experience working with Nathan; his enthusiasm has always been a great source of motivation, and his fondness of geeky humor often made my day.

I am deeply grateful to Tamás Kiss who introduced me to quantum information science during my Master's studies. He gave me advice on many aspects of life. Tamás has always been very supportive and helped me a great deal in becoming a quantum researcher.

During my PhD I learned a lot from my wonderful coauthors, to whom I wish to express my gratitude. I wrote the most papers together with Joran van Apeldoorn, with whom I shared various offices and conference accommodations over the years, and had countless discussions about various aspects of life ranging from political activism to quantum research. He had many great insights into the world of LPs and SDPs and is a very persistent problem solver, always keen on cracking the next problem. When we got the feel for a problem there was no stopping us, and if Sander Gribling also joined the effort, it always became a bigger shot! I am very appreciative of the inclusive attitude of Srinivasan Arunachalam; amongst the group's students he was the most entangled with the larger quantum community, and was the most informed about new results, recent developments and gossips. He always had a great collection of interesting questions in mind that he was always willing to share – this is also how our paper grow out of a common discussion with Nathan. It was a pleasure to meet Yuan Su, who is the most humble, yet one of the most talented PhD students I had the fortune to work with. Without his insights into the delicate details of qubitization, this thesis would probably look quite different! I thank Guang Hao Low for his invaluable work on Hamiltonian simulation, quantum signal processing and qubitization, providing the ground for quantum singular value transformation – the central topic of this dissertation. I would also like to thank Or Sattath who became interested in my attempts to improve the Quantum Lovász Local Lemma just after a single Skype meeting – interestingly, we only met in person at the QIP where we presented our paper on the topic.

Thanks to all my not yet mentioned collaborators who worked with me on topics beyond the scope of this dissertation: Andris Ambainis, Simon Apers, Tom Bannink, Iva Bezděková, Shantanav Chakraborty, Rui Chao, Dawei Ding, Cupjin Huang, Stacey Jeffery, Igor Jex, Martins Kokainis, Bálint Kollár, Tongyang Li, Seth Lloyd, Martin Štefaňák, and Ewin Tang. Special thanks to Stacey: it was great working on interesting problems together; thanks for backing me up in the presence of headhunters and providing detailed feedback on my dissertation!

I would also like to thank, for many inspiring and insightful discussions, Dorit Aharonov, Johannes Bausch, Niel de Beaudrap, Alexander Belov, Ferenc Bencs, Fernando Brãndao, Sebastien Bubeck, Andrew Childs, Ignacio Cirac, Michael Cohen, Arjan Cornelissen, Daniel Dadush, Vedran Dunjko, Katalin Friedl, Yimin Ge, Alex Grilo, Thomas Häner, Aram Harrow, Robin Kothari, Yuanzhi Li, Ashley Montanaro, Ashwin Nayak, Maris Ozols, Guus Regts, Miklós Sántha, Maria Schuld, Martin Schwarz, Jamie Sikora, Csaba Szepesvári, Gábor Tardos, Barbara

Terhal, Thomas Vidick, Michael Walter, Xiaodi Wu, and Henry Yuen.

Special thanks go to my paranymphs Joran van Apeldoorn and Jan Czajkowski for helping me with the defense! I am grateful to Koen Groenland and János Asbóth, who helped me writing the Dutch and Hungarian versions of the abstract in this dissertation. I thank Yinan Li for taking pictures of the chalk drawings for the cover.

My life was greatly enriched by my fellow PhD students at CWI. I thank the members of the PhD activity committee for organizing many fun events, it was great arranging these events together or just enjoying being able to participate, and sometimes randomly hang out together. Special thanks to Esteban, Giulia, Jan, Jasmijn, Marco, Mark, and Mathé. I am glad that we could share some fun and casual activities, during various conference trips with Alvaro, Arjan, Johannes, Joran, and Srinivasan. It was great hanging out and sharing stuff with other PhD students in the "big green building" where I lived, thanks for all the support to Isabella, Jeroen, Krzysztof, and Teresa. Many thanks to Tom for setting up the Élő-system for the foosball table, and to Chris (all three of them), Farrokh, Florian, Freek, Joris, Koen, Lars, Pieter, Sander, Subhasree, and Yfke for many games!

I also thank the members of our group of young Catholics at the Onze Lieve Vrouwekerk for organizing our regular get-togethers, and for their support. I am especially thankful to father Rafael Ojeda, and Alex, Diana, Dominika, Eric, José, Julia, Piotr, Terézia, and Thijs.

I am grateful to my friends, especially Réka and Dudu, who always held onto me even when I was far away, and sometimes even came to visit me here in Amsterdam.

Finally, I am grateful to my parents for teaching me so much and guiding me throughout life! I am also thankful to my brother and other relatives for being always supportive.

Amsterdam
April, 2019.

András Gilyén

Abstract

In this dissertation we study how efficiently quantum computers can solve various problems, and how large speedups can be achieved compared to classical computers. In particular we develop a generic quantum algorithmic framework that we call "quantum singular value transformation", and show how it unifies a large number of prominent quantum algorithms. Then we show several problems where quantum singular value transformation leads to new quantum algorithms or improves various aspects of earlier approaches.

In **Chapter 2** we develop a new quantum singular value transformation algorithm capable of working with exponentially large matrices, that can apply polynomial transformations to the singular values of a block of a unitary. The proposed quantum circuits have a very simple structure, often give rise to optimal algorithms and have appealing constant factors, while typically only using a constant number of ancilla qubits. We prove several properties of quantum singular value transformation, including its robustness.

In **Chapter 3** we show that quantum singular value transformation leads to novel algorithms. We propose a new method for singular value estimation, and also show how to exponentially improve the complexity of implementing fractional queries to unitaries with a gapped spectrum. Finally, as a quantum machine learning application we show how to efficiently implement principal component regression. We also show that quantum singular value transformation leads to a unified framework of quantum algorithms incorporating a variety of quantum speed-ups. Using this framework we can describe many quantum algorithms on a high level, hopefully making them accessible to researchers even outside the quantum algorithm community. We illustrate this by showing how our meta-algorithm generalizes a number of prominent quantum algorithms, and quickly derive the following algorithms: optimal Hamiltonian simulation, implementing the Moore-Penrose pseudoinverse (i.e., the HHL algorithm) with exponential precision, fixed-point amplitude amplification, robust oblivious amplitude amplification, fast QMA amplification, fast quantum OR lemma, certain quantum walk

results and several quantum machine learning algorithms. In order to exploit the strengths of the presented method, it is useful to know its limitations too, therefore we also prove a bound on the efficiency of quantum singular value transformation, which often gives optimal lower bounds.

In **Chapter 4** we develop an improved quantum algorithm for computing the gradient of a multivariate real-valued function $f\colon \mathbb{R}^d \to \mathbb{R}$ by evaluating it at only a logarithmic number of points in superposition. Our algorithm is an improved version of Jordan's gradient computation algorithm [Jor05], providing an approximation of the gradient ∇f with quadratically better dependence on the evaluation accuracy of f, for an important class of smooth functions. Furthermore, we show that most objective functions arising from a class of quantum optimization procedures satisfy the necessary smoothness conditions, hence our algorithm improves the runtime of prior approaches for training quantum auto-encoders, variational quantum eigensolvers (VQE), and quantum approximate optimization algorithms (QAOA). Finally we prove that in a continuous phase-query model, our gradient computation algorithm has essentially optimal query complexity, for a class of smooth functions. For our lower bound we derive a continuous input version of the so-called hybrid method.

In **Chapter 5** we study to what extent quantum algorithms can speed up solving convex optimization problems. Following the classical literature we assume access to a convex set via various oracles, and we examine the efficiency of reductions between the different oracles. In particular, we show how a separation oracle can be implemented using $\widetilde{\mathcal{O}}(1)$ quantum queries to a membership oracle, which is an exponential quantum speed-up over the $\Omega(n)$ membership queries that are needed classically. We show that a quantum computer can very efficiently compute an approximate subgradient of a convex Lipschitz function. Combining this with a simplification of recent classical work of Lee, Sidford, and Vempala [LSV18] gives our efficient separation oracle. This in turn implies, via a known algorithm, that $\widetilde{\mathcal{O}}(n)$ quantum queries to a membership oracle suffice to implement an optimization oracle (the best known classical upper bound on the number of membership queries is quadratic). We also prove several lower bounds: $\Omega(\sqrt{n})$ quantum separation (or membership) queries are needed for optimization if the algorithm knows an interior point of the convex set, and $\Omega(n)$ quantum separation queries are needed if it does not.

In **Chapter 6** we take a new perspective on quantum SDP-solvers, introducing several new techniques, and improve on all prior quantum algorithms for SDP-solving. Our new input model generalizes all prior input models, and assumes that the input matrices are provided as block-encodings. In this model we give a $\widetilde{\mathcal{O}}((\sqrt{m}+\sqrt{n}\gamma)\alpha\gamma^4)$ algorithm, where n is the size of the matrices, m is the number of constraints, γ is the reciprocal of the scale-invariant relative precision parameter, and α is a normalization factor of the input matrices. In particular for the standard sparse-matrix access model, the above result gives a quantum algorithm where α equals the sparsity s. We also improve on recent results of Brandão et al. [BKL^{+}18], who consider the special case when the input matrices are proportional to mixed quantum states that one can query. For this model Brandão et al. [BKL^{+}18] showed that the dependence on n can be replaced by a polynomial dependence on both the rank and the trace of the input matrices. We remove the dependence on the rank and hence require only a dependence on the trace of the input matrices. We also show an application to the problem of shadow tomography, recently introduced by Aaronson [Aar18]. Finally we prove a new $\tilde{\Omega}(\sqrt{m}\alpha\gamma)$ lower bound for solving LPs and SDPs in the quantum operator model, which also implies a lower bound for the model of Brandão et al. [BKL^{+}18].

In **Chapter 7** we study constructive versions of the Lovász Local Lemma (LLL) and its quantum generalization. The quantum Lovász Local Lemma can be stated in terms of frustration-free local Hamiltonians: these Hamiltonians have the property that their ground state minimizes the energy of all local terms simultaneously. We improve on the previous constructive quantum results by designing an algorithm that works efficiently for non-commuting terms as well, assuming that the system is "uniformly" gapped, by which we mean that the system *and all its subsystems* have an energy gap that is at least inverse polynomially large. We generalize and simplify Moser's classical "compression argument", and derive a non-commutative quantum version of the famous Moser-Tardos resampling algorithm. Finally, in the so-called variable version of the classical LLL we find optimal bounds for the "guaranteed-to-be-feasible" probabilities on cyclic dependency graphs, and show that this region is always strictly larger than in the generic non-variable version, where Shearer's bound is optimal. This in turn shows a separation between the variable version of the classical and the quantum LLL.

Samenvatting

In dit proefschrift bestuderen we hoe efficiënt quantum computers verschillende problemen kunnen oplossen en de mate waarin dit sneller kan in vergelijking met klassieke computers. In het bijzonder ontwikkelen we een algemeen raamwerk voor quantum algoritmes dat we “quantum singulierewaardentransformatie” noemen, en we laten zien hoe dit een groot aantal prominente quantum algoritmes unificeert. Vervolgens laten we een aantal problemen zien waarbij dit raamwerk tot nieuwe quantum algoritmes of verbeteringen van bestaande algoritmes leidt.

In **Hoofdstuk 2** ontwikkelen we een nieuw quantum algoritme voor de singulierewaardentransformatie dat overweg kan met exponentieel grote matrices en dat polynomiale transformaties kan toepassen op de singuliere waarden van een blok van een unitaire matrix. De voorgestelde quantum circuits hebben een heel simpele structuur, geven vaak optimale algoritmes en hebben aantrekkelijke constante factoren, terwijl ze meestal maar een constant aantal ancilla qubits gebruiken. We bewijzen verschillende eigenschappen van de quantum singulierewaardentransformatie waaronder haar robuustheid.

In **Hoofdstuk 3** laten we zien hoe de quantum singulierewaardentransformatie tot nieuwe algoritmes leidt. We stellen een nieuwe methode voor om singuliere waarden te benaderen en we behalen exponentiële verbeteringen in de complexiteit van het implementeren van fractionele queries aan unitairen met een gapped spectrum. Als toepassing voor quantum machine learning geven we ten slotte een efficiënte implementatie van principale-componentenregressie. We laten ook zien dat de quantum singulierewaardentransformatie leidt tot een geünificeerd raamwerk van quantum algoritmes waar verscheidene quantum versnellingen bij zitten. Met dit raamwerk kunnen we veel quantum algoritmes op een hoog niveau beschrijven en daarmee zelfs toegankelijk maken voor onderzoekers buiten de quantum algoritmes gemeenschap. We illustreren dit door een aantal prominente quantum algoritmes te generaliseren in ons raamwerk en leiden kort de volgende algoritmes af: optimale simulatie van Hamiltonianen, impementatie van de Moore-Penrose pseudoinverse (d.w.z. het HHL algoritme) met exponentiële precisie, de zogeheten “fixed-point amplitude amplification”, robuuste “oblivious amplitude amplification”, snelle QMA amplificatie, het snelle quantum OR lemma, bepaalde quantum walk resultaten en een aantal quantum machine learning algoritmes. Om de methode volledig te benutten is het ook waardevol om de beperkingen te kennen en daarom bewijzen we ook een grens op de efficiëntie van de quantum singulierewaardentransformatie, die vaak optimale ondergrenzen geeft.

In **Hoofdstuk 4** ontwikkelen we een verbeterd quantum algoritme om de gradiënt van een multivariate reëelwaardige functie $f\colon \mathbb{R}^d \to \mathbb{R}$ te berekenen door deze hooguit op een logaritmisch aantal punten te evalueren in superpositie. Ons algoritme is een verbeterde versie van Jordans algoritme [Jor05] en geeft een benadering van de gradiënt ∇f met kwadratisch betere afhankelijkheid van de evaluatienauwkeurigheid van f, voor een belangrijke klasse van gladde functies. Verder laten we zien dat de meeste doelfuncties die voortkomen uit een klasse quantum optimalisatieprocedures voldoen aan de benodigde gladheidseigenschappen. Hierdoor verbetert ons algoritme de looptijd van bestaande algoritmes voor het trainen van zogeheten quantum auto-encoders, variationele quantum eigensolvers (VQE) en quantum approximate optimization algorithms (QAOA). Ten slotte bewijzen we dat in een continu fase-query model ons algoritme een optimale query complexiteit heeft voor een klasse van gladde functies. Voor onze ondergrens leiden we een continue invoer versie af van de zogeheten hybride methode.

In **Hoofdstuk 5** onderzoeken we in hoeverre quantum algoritmes het oplossen van convexe optimalisatie problemen kunnen versnellen. Net als in de klassieke literatuur beschouwen we verschillende manieren van toegang tot de convexe verzameling (zogeheten oracles), en we bestuderen de efficiëntie van de reducties tussen de verschillende oracles. In het bijzonder laten we zien hoe een “separatie oracle” kan worden geïmplementeerd met $\widetilde{\mathcal{O}}(1)$ quantum queries naar een “membership oracle”, wat een exponentiële verbetering is ten opzichte van de $\Omega(n)$ queries die klassiek nodig zijn. We laten zien dat een quantum computer heel efficiënt een benadering kan uitrekenen van een subgradiënt van een convexe Lipschitz functie. Dit combineren we met een versimpelde versie van het recente klassieke resultaat van Lee, Sidford, en Vempala [LSV18] om efficiënt een separatie oracle te krijgen. Vervolgens impliceert dit, via een bekend algoritme, dat $\widetilde{\mathcal{O}}(n)$ quantum queries aan een membership oracle voldoende zijn om een optimalisatie oracle te implementeren (de best bekende klassieke bovengrens op het aantal membership queries is kwadratisch). We bewijzen ook verschillende ondergrenzen: $\Omega(\sqrt{n})$ quantum separatie (of membership) queries zijn nodig voor optimalisatie als het algoritme een intern punt van de convexe verzameling weet, en $\Omega(n)$ quantum separatie queries zijn nodig als dat niet zo is.

In **Hoofdstuk 6** geven we een nieuwe kijk op quantum algoritmes voor het oplossen van SDPs. We introduceren verschillende nieuwe technieken en verbeteren voorgaande quantum algoritmes voor het oplossen van SDPs. Ons nieuwe invoermodel generaliseert alle voorgaande modellen en berust op de aanname dat de invoer matrices als blok-codering worden gegeven. In dit model geven we een $\widetilde{\mathcal{O}}((\sqrt{m}+\sqrt{n}\gamma)\alpha\gamma^4)$ algoritme, waarbij n de grootte van de matrices is, m het aantal randvoorwaarden, γ is de reciproke van de schaal-invariante precisieparameter, en α is een normalisatiefactor van de invoermatrices. In het bijzonder, voor het standaard sparse-matrixtoegangsmodel geeft het genoemde resultaat een quantum algoritme waar α gelijk is aan de zogeheten sparsity s. We geven ook een verbetering van een recent resultaat van Brandão et al. [BKL$^+$18], die een speciaal geval bekijken waar de invoermatrices proportioneel zijn aan quantum dichtheidsmatrices. Voor dit model laten Brandão et al. [BKL$^+$18] zien dat de afhankelijkheid van n vervangen kan worden door een polynomiale afhankelijkheid van de rang en het spoor van de invoermatrices. Ons resultaat haalt de afhankelijkheid van de rang weg en we houden alleen afhankelijkheid van het spoor over. We laten ook een toepassing zien voor het probleem van schaduwtomografie, recent geïntroduceerd door Aaronson [Aar18]. Ten slotte bewijzen we een nieuwe $\tilde{\Omega}(\sqrt{m}\alpha\gamma)$ ondergrens voor het oplossen van LPs en SDPs in het quantum operator model, wat ook een ondergrens impliceert voor het model van Brandão et al. [BKL$^+$18].

In **Hoofdstuk 7** bestuderen we constructieve versies van het Lovász Local Lemma (LLL) en de quantum generalisatie ervan. Het quantum Lovász Local Lemma kan worden geformuleerd in termen van zogeheten frustratievrije lokale Hamiltonianen. Deze Hamiltonianen hebben de eigenschap dat hun grondtoestand de energie van alle lokale termen tegelijk minimaliseert. We verbeteren de voorgaande constructieve quantum resultaten door een algoritme te ontwikkelen dat ook efficiënt met termen kan omgaan die niet commuteren, onder de aanname dat het systeem "uniformly gapped" is. Dit betekent dat het systeem *en alle deelsystemen* een energiekloof hebben die ten minste invers-polynomiaal groot is. We generaliseren en versimpelen Mosers klassieke "compressie-argument" en leiden een niet-commutatieve quantum versie af van het bekende Moser-Tardos resampling algoritme. Ten slotte, in de zogeheten variabele versie van het klassieke LLL verkrijgen we optimale grenzen voor de kansen op cyclische afhankelijkheidsgrafen die een oplossing garanderen, en we laten zien dat dit gebied altijd strikt groter is dan in het algemene niet-variabele geval, waar de grens van Shearer optimaal is. Dit laat vervolgens een scheiding zien tussen de variabele versie van het klassieke en quantum LLL.

Kivonat

E disszertáció azt járja körül, hogy a kvantumszámítógépek mennyire hatékonyan tudnak megoldani különböző problémákat, illetve mekkora „kvantumos gyorsítás” érhető el a klasszikus számítógépekhez viszonyítva. Ennek érdekében felépítjük a *„kvantumos szingulárisérték-transzformáció”* általános algoritmikus keretrendszerét, amivel számos nevezetes kvantumalgoritmus egységesen leírható. Több fontos számítási feladattal illusztráljuk, hogy a kvantumos szinguláriséték-transzformáció új algoritmusok kifejlesztését teszi lehetővé, illetve korábbi megoldások különböző aspektusain képes javítani.

A **2. fejezetben** felépítünk egy új szinguláriséték-transzformációs kvantumalgoritmust, ami képes exponenciálisan nagy mátrixokkal dolgozni, oly módon, hogy egy unitér mátrix bizonyos blokkjának szinguláris értékein hajt végre valamilyen polinomiális transzformációt. A felvázolt kvantumáramkörök nagyon egyszerű struktúrájúak, gyakran optimális algoritmusokhoz vezetnek, méretük kedvező konstans faktorokkal skálázódik, valamint tipikusan csak konstans sok ancilla kvantumbitet használnak. A kvantumos szinguláriséték-transzformáció számos tulajdonságát vizsgáljuk, többek között bebizonyítjuk hogy a bement apró pontatlansága mellett is megbízhatóan működik.

A **3. fejezetben** demonstráljuk, hogy a kvantumos szinguláriséték-transzformáció új algoritmusok kifejlesztését teszi lehetővé. Új módszert adunk kvantumos szinguláriséték-becslésre, és exponenciális mértékben javítunk spektrális réssel bíró unitér operátorok törthatványú alkalmazásának korábbi megvalósításain. Végül a főkomponens-regresszió hatékony implementálása által a kvantumos gépi tanulás területén is adunk egy alkalmazást. Megmutatjuk azt is, hogyan vezet a kvantumos szinguláriséték-transzformáció egy olyan algoritmikus keretrendszerhez, amiben különböző kvantumos gyorsítási módszerek egységesen kezelhetőek. Ez a keretrendszer többféle kvantumalgoritmus magas szintű leírását teszi lehetővé, ezáltal remélhetőleg a kvantumalgoritmusok szakértőin kívül mások számára is megközelíthetővé téve az eredményeket. Ezt illusztrálandó, megmutatjuk miként általánosítja metaalgoritmusunk számos korábbi nevezetes kvantumalgo-

ritmus működését, és röviden levezetjük a következő algoritmusokat: optimális Hamilton-szimuláció, a Moore-Penrose pszeudóinverz implementálása (vagyis az HHL algoritmus) exponenciális precizitással, fixpontú amplitúdó-amplifikáció, robusztus hanyag (oblivious) amplitúdó-amplifikáció, gyors QMA-amplifikáció, gyors kvantumos VAGY lemma, bizonyos kvantumbolyongásos eredmények, és különböző kvantumalgoritmusok gépi tanulásra. Módszerünk erősségeinek kiaknázásához hasznos ismerni annak határait is, ezért egy korlátot is bizonyítunk a kvantumos szinguláriSérték-transzformáció hatékonyságára, ami gyakran optimális eredményekre vezet.

A **4. fejezetben** konstruálunk egy optimalizált algoritmust többváltozós $f\colon \mathbb{R}^d \to \mathbb{R}$ függvények gradiensének számítására, amely a függvényt egymás után mindössze logaritmikusan sok pontban értékeli ki, de szuperpozícióban. Algoritmusunk Jordan gradiens-számító algoritmusát [Jor05] fejleszti tovább, és a ∇f gradiens approximációját kvadratikusan javítja f kiértékelési pontosságának függvényében, egy fontos sima függvényosztály esetében. Bebizonyítjuk, hogy a kvantumos variációs optimalizációban megjelenő célfüggvények kielégítik a simasági kritériumainkat, ezért algoritmusunk javítja a korábbi tanítási eljárások futásidejét, többek között kvantumos autoenkóderek, variációs kvantumos sajátérték-algoritmusok (variational quantum eigensolvers) és kvantumos közelítő optimalizációs algoritmusok (quantum approximate optimization algorithms) esetében. Végül bebizonyítjuk, hogy egy folytonos fázislekérdezési modellben a sima függvények egy bizonyos osztályára nézve gradiensszámító algoritmusunk lényegében optimális. Az alsó korlát bizonyításához levezetjük az úgynevezett hibrid módszer egy folytonos változójú verzióját.

Az **5. fejezetben** azt vizsgáljuk, hogy mekkora mértékű kvantumos gyorsítás érhető el konvex optimalizációs problémák esetében. A klasszikus szakirodalmat követve a konvex halmazokhoz való hozzáférést különböző orákulumokkal modellezzük, és az orákulumok közötti visszavezetések hatékonyságát vizsgáljuk. Bebizonyítjuk, hogy egy ún. szeparációs orákulum implementálható egy ún. tagsági orákulum mindössze $\widetilde{\mathcal{O}}(1)$ kvantumlekérdezése segítségével, ami exponenciális kvantumos gyorsítást eredményez a szükséges $\Omega(n)$ klasszikus lekérdezésszámhoz képest. Ezt a hatékony szeparációs orákulumot úgy kapjuk meg, hogy először megmutatjuk, hogy egy kvantumszámítógép nagyon hatékonyan tudja konvex Lipschitz függvények szubgradiensét közelíteni, majd ezt kombináljuk Lee, Sidford és Vempala [LSV18] friss eredményeinek egy egyszerűsített verziójával. Egy korábbi ismert algoritmus révén mindebből az is következik, hogy egy ún. optimalizációs orákulum implementálható egy tagsági orákulum $\widetilde{\mathcal{O}}(n)$ kvantumlekérdezése által; ez kvadratikus kvantumos gyorsítást eredményez a legjobb ismert klasszikus eredményhez képest. Több alsó korlátot is bizonyítunk: az optimalizációhoz $\Omega(\sqrt{n})$ szeparációs (vagy tagsági) kvantumlekérdezés szükséges a konvex halmaz egy belső pontjának ismeretében, nélküle pedig $\Omega(n)$.

A **6. fejezetben** új megvilágításban vizsgáljuk a szemidefinit programok (röviden SDP-k) megoldására szolgáló kvantumalgoritmusokat, és valamennyi korábbi kvantumalgoritmuson javítunk. Új bemeneti modellünk mindegyik korábbi bemeneti modellt általánosítja, és azt feltételezi, hogy a bemeneti mátrixok bizonyos unitér transzformációk blokk-beágyazásaként vannak megadva. Ebben a kvantumoperátoros modellben adunk egy $\widetilde{\mathcal{O}}((\sqrt{m}+\sqrt{n}\gamma)\alpha\gamma^4)$ komplexitású kvantumalgoritmust, ahol n a kényszerfeltételeket leíró mátrixok méretét és m a számukat jelöli, γ a skálainvariáns relatív precíziós paraméter, és α a bemeneti mátrixok normalizációs faktora. A ritka mátrixok bemeneti modelljére vonatkozó eredményünk úgy kapható meg ha a fenti formulában α-t lecseréljük s-re, a mátrix ritkasági korlátjára. Algoritmusunk javít Brandão és tsai. [BKL+18] eredményein is. Abban a speciális esetben, amikor a bemeneti mátrixok bizonyos kevert kvantumállapotok lineáris kombinációjaként adottak, Brandão és tsai. [BKL+18] megmutatták, hogy az n paramétertől való függés lecserélhető a bemeneti mátrixok rangjától és nyomától való polinomiális függésre. A mi megfelelő algoritmusunk azonban nem függ a rangtól, így az n-től való függés kiváltható a bemeneti mátrixok nyomától való polinomiális függéssel. Megmutatjuk azt is, hogy eredményeink miként alkalmazhatóak az Aaronson által nemrégiben felvetett árnyéktomográfia (shadow tomography) [Aar18] problémájára. Végül bizonyítunk egy új $\tilde{\Omega}(\sqrt{m}\alpha\gamma)$ alsó korlátot lineáris programok (LP-k) és szemidefinit programok (SDP-k) megoldásának komplexitására a kvantumoperátoros bemeneti modellben, amiből hasonló alsó korlátot kapunk Brandão és tsai. [BKL+18] bemeneti modellje esetében is.

A **7. fejezetben** a Lovász-féle lokális lemmának és kvantumos általánosításának konstruktív verzióit vizsgáljuk. A kvantumos Lovász-féle lokális lemma változókon alapuló verziója kimondható a frusztrációmentes lokális Hamilton-operátorok segítségével: ezen Hamilton-operátorok azzal a tulajdonsággal rendelkeznek, hogy alapállapotuk az összes lokális energiatagot külön-külön minimalizálja. A korábbi konstruktív eredményeken javítva egy olyan algoritmust adunk, amely hatékonyan működik nemkommutatív tagok esetében is, feltéve hogy a rendszer spektrális rése egyenletes (uniformly gapped), azaz a rendszernek *és minden részrendszerének* spektrális rése legalább inverz polinomiális. Miközben levezetjük a híres Moser-Tardos-féle újramintavételező algoritmus egy nemkommutatív kvantumos megfelelőjét, általánosítjuk és egyszerűsítjük Moser klasszikus „tömörítési érvelését" is. Végezetül a változókon alapuló (klasszikus) Lovász-féle lokális lemma esetében körgráfokon optimális korlátokat adunk az egyes változók valószínűségeire, amelyek mellett a rendszer garantáltan kielégíthető. A kapott korlát tágabb mint a Shearer-korlát, noha ez utóbbi az általános - nem változókon alapuló - esetben optimális. Ez egyben elkülöníti egymástól a változókon alapuló kvantumos és klasszikus Lovász-féle lokális lemmát.

Chapter 1

Introduction and preliminaries

1.1 Computation in the physical world

In the 1930s some influential mathematicians become interested in studying questions like "What problems are efficiently calculable?" and "What does it mean that something is computable?". They started formalizing models of the process of computation, intending to grasp the essence of systematic problem solution. There were three major angles of approach to this question. Kurt Gödel approached computability through the theory of *recursive functions*, while Alonzo Church [Chu36a, Chu36b] defined the λ-*calculus* of functions and Alan Turing [Tur37] defined his famous model of computational machines, today known as *Turing machines*. Church and Turing proved that these three seemingly different models of computation are all equivalent in the sense that if some function is "computable" according to one of the models, then it is "computable" in the other two models too. This result provided evidence that these three models are describing the "right model of computation" capturing all realistic computation method – this statement is called the Church–Turing thesis.

In the early 1900s some fundamental new physical discoveries dramatically changed our understanding of the physical world surrounding us. The discovery of relativity theory showed that space and time are intertwined in our universe, and with the discovery of quantum mechanics it become clear that the elementary particles look nothing like tiny balls, but rather like small waves. These discoveries showed that our world behaves in a very counter-intuitive way at many levels, and that our everyday experiences can paint a false image of how our world really behaves.

Still, Turing machines model our *classical* computational devices: they operate on some large data storage tape, and perform basic computational steps according to their internal state, and the data they read from the tape. However, if we want to consider an appropriate computational model for our physical world, then we should take into account its non-classical behavior. This is what

Richard Feynman suggested in his famous paper [Fey82], realizing that in order to understand and simulate various processes in our (quantum) physical world, we should use quantum computational devices. Following up on this idea David Deutsch defined *quantum Turing machines* [Deu85]. Although classical and quantum Turing machines have quite different capabilities, it turns out that the set of computable functions is the same for both of them.

There could still be a large difference in the efficiency of classical and quantum Turing machines. The class of decision problems that can be efficiently solved (in time upper bounded by a polynomial in the problem size) on a randomized classical Turing machine is called BPP, and the quantum analogue is called BQP. For example Peter Shor's famous quantum algorithm [Sho97] shows that integer factorization can be efficiently solved on a quantum computer, while no such efficient classical algorithm is known. Therefore, many researchers believe that $\mathsf{BPP} \subsetneq \mathsf{BQP}$, i.e., for some problems a quantum computer can provide very large speedups compared to classical computers.

In this dissertation we study how efficiently quantum computers can solve various problems, and how large speedups can be achieved compared to classical computers. In particular we develop a generic quantum algorithmic framework that we call quantum singular value transformation, and show how it unifies a large number of prominent quantum algorithms. Then we show several problems where quantum singular value transformation leads to new quantum algorithms or improves various aspects of earlier approaches.

1.2 Quantum mechanical principles

In this dissertation we will use Dirac's bra-ket notation, that is for a Hilbert space (complex Euclidean vector space) $\mathcal{H}$ and a vector $\psi \in \mathcal{H}$ we will also use the alternative notation $|\psi\rangle$, and will denote its adjoint by $\langle\psi|$. We will think about $|\psi\rangle$ as a column vector, and $\langle\psi|$ as a row vector. For $\psi, \phi \in \mathcal{H}$ we denote their inner product by $\langle\psi|\phi\rangle \in \mathbb{C}$, alternatively $|\psi\rangle\langle\phi| \in \mathcal{H} \otimes \mathcal{H}$ denotes a rank-1 matrix. In this dissertation we solely work with finite-dimensional Hilbert spaces.

Following the exposition of Jozsa [Jozs18] we list the four basic principles of quantum mechanics, describing our physical model behind quantum computers.

(i) A d-dimensional quantum system is a Hilbert space $\mathcal{H} \simeq \mathbb{C}^d$, and a *(pure) quantum state* $|\psi\rangle \in \mathcal{H}$ is a vector of unit length, i.e., $\langle\psi|\psi\rangle = 1$.

(ii) The *composite system* of two quantum systems $\mathcal{H}_1, \mathcal{H}_2$ is the tensor product $\mathcal{H}_1 \otimes \mathcal{H}_2$ of the subsystems.

(iii) The *coherent* operations of a quantum system $\mathcal{H}$ are its unitary operators.

(iv) Let $(|b_1\rangle, |b_2\rangle, \ldots, |b_d\rangle)$ be an orthonormal basis of $\mathcal{H}$. If we *measure* the pure quantum state $|\psi\rangle = \sum_{i=1}^{d} \beta_i |b_i\rangle$ in this basis, then we get outcome $i \in [d]$ with probability $|\beta_i|^2$, and the state of the quantum system then becomes $|b_i\rangle$.

The last principle **(iv)** is called the *Born rule*, while the *extended Born-rule* states that if $B = (|b_1\rangle, |b_2\rangle, \ldots, |b_d\rangle)$ is an orthonormal basis of $\mathcal{H}_1$ and $|\psi\rangle = \sum_{j=1}^{d} |b_j\rangle \otimes |\phi_j\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2$, then a measurement of the first system gives $j \in [d]$ with probability $\langle\phi_j|\phi_j\rangle$, and the *post-measurement state* of the quantum system becomes $|b_j\rangle \otimes |\phi_j\rangle / \sqrt{\langle\phi_j|\phi_j\rangle}$. Note that multiplying one of the components $|\phi_j\rangle$ by a *phase* $e^{i\theta}$ does not change the measurement statistics for measurements in the B basis. Moreover, the quantum states $|\psi\rangle$ and $e^{i\theta}|\psi\rangle$ are indistinguishable by any measurements, because they only differ by a *global phase.*

We call a two-dimensional quantum system $\mathbb{C}^2$ a *qubit*, and denote its *computational basis states* by $|0\rangle$ and $|1\rangle$, which form an orthonormal basis. Thereby a pure qubit state is $\alpha_0|0\rangle + \alpha_1|1\rangle$, such that $|\alpha_0|^2 + |\alpha_1|^2 = 1$; α_0 and α_1 are called the *amplitudes* of the $|0\rangle$ and $|1\rangle$ basis states. The composite system of n qubits is called an n-qubit system, and its computational basis states are denoted by $|b\rangle\colon b \in \{0,1\}^n$. In general a d-dimensional quantum system $\mathbb{C}^d$ with computational basis states $|0\rangle, |1\rangle, \ldots, |d-1\rangle$ might be called a *qudit.*

We say that the pure state $|\psi\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2$ is a *product state* if it can be written as $|\psi\rangle = |\psi_1\rangle \otimes |\psi_2\rangle$ for some $|\psi_1\rangle \in \mathcal{H}_1$, $|\psi_2\rangle \in \mathcal{H}_2$, and otherwise we say that $|\psi\rangle$ is an *entangled state.* Finally, if $|\psi\rangle = \alpha|\phi\rangle + \beta|\varphi\rangle$ with $\alpha \neq 0 \neq \beta$, then we say that $|\psi\rangle$ is a *superposition* of $|\phi\rangle$ and $|\varphi\rangle$. When forming a superposition, the amplitudes add up, which can lead to *interference.* Take for example the so-called plus $|+\rangle := \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and minus $|-\rangle := \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$ states. For these states a measurement in the computational basis leads to a uniformly random 0 / 1 result. However, measuring their superposition $\frac{1}{\sqrt{2}}(|+\rangle + |-\rangle) = |0\rangle$ we get outcome 0 with certainty, which is the result of *constructive interference* of the amplitudes of the $|0\rangle$ basis state.

1.3 The circuit model and quantum queries

In order to describe our quantum algorithms we will use the so-called *quantum circuit model.* In this model a quantum computation is described on a fixed number of qubits (or qudits), and a quantum circuit is simply a unitary operator followed by a (partial) measurement in the computational basis. Typically it is assumed that all qubits are set to the $|0\rangle$ state initially, except for the input qubits. The circuits are built from a sequence of *single-qubit gates* and *two-qubit gates*, that is unitary operators which act trivially except on a single- or two-qubit subsystem. If U is a single- or two-qubit (unitary) quantum gate, it is extended to the composite system by taking the tensor product with the identity operator on the rest of the system. The *gate complexity* of a quantum circuit is the number of single- and two-qubit gates it is built from. For simplicity we will allow measurements in the middle of the circuits as well, but in principle all such measurements can be deferred to the end of the circuit.

One of the most important single-qubit quantum gates is the *Hadamard gate*, denoted by

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

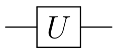
The *Pauli gates* are also important quantum operations, denoted by

$$\sigma_{\mathrm{x}} = X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \qquad \sigma_{\mathrm{y}} = Y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \qquad \sigma_{\mathrm{z}} = Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix},$$

and the corresponding *single-qubit rotations* of angle θ are

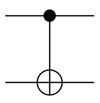
$$e^{i\theta\sigma_{\mathrm{x}}} = \begin{bmatrix} \cos(\theta) & i\sin(\theta) \\ i\sin(\theta) & \cos(\theta) \end{bmatrix} \quad e^{i\theta\sigma_{\mathrm{y}}} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \quad e^{i\theta\sigma_{\mathrm{z}}} = \begin{bmatrix} e^{i\theta} & 0 \\ 0 & e^{-i\theta} \end{bmatrix}.$$

The circuit representation of a single-qubit quantum gate U is



The most important two-qubit gate is the *controlled NOT gate*, denoted by

$$\mathrm{CNOT} = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} =$$



where the two *wires* correspond to the two qubits, with the first wire representing the *control qubit.* The symbol $\oplus$ refer to the modulo 2 addition of the first bit to the second bit.

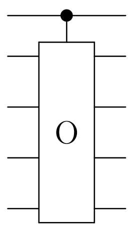
We note that instead of allowing arbitrary single- and two-qubit gates one could also build quantum circuits using a more restrictive *universal gate set*, which allows any n-qubit unitary to be arbitrarily well approximated using only these gates. A famous example of a universal gate set consists of H, $T = e^{i\pi/8}e^{-i\sigma_z\pi/8}$ and CNOT. This affects the gate complexity by at most a polylogarithmic factor, because any single- or two-qubit gate can be ε-approximated using polylog$(1/\varepsilon)$ gates from the universal gate set, as shown by the Solovay-Kitaev theorem; for more information see [NC00]. This restricted gate set is extremely useful when one needs to deal with errors, because it shows that it is enough to find *fault tolerant* implementations of these three quantum gates, and the measurement in the computational basis. The theory of *quantum error correction* and *fault-tolerant quantum computation* addresses this question, and shows that if error-rates are below some constant threshold, then under various error models, errors can be exponentially suppressed, hence a fault-tolerant implementation introduces only logarithmic overheads in the circuit complexity.

Classical computations. A classical Boolean circuit can be described as a sequence of logical gates. Since all logical gates can be built from NOT and AND gates using some ancilla bits initialized to 0, we can assume without loss of generality that our Boolean circuits are built solely from these two gates. We can define a reversible version of the AND gate that takes three input bits (b_0, b_1, b_2) and outputs the bits $(b_0, b_1, b_2 \oplus (b_0 \cdot b_1))$, where the operation $\oplus$, denotes[1] addition modulo 2. This reversible version of the AND gate is called the Toffoli gate, which also corresponds to a unitary operator, since it describes a permutation on 3-bit strings, and permutations are unitaries.

Since every Boolean circuit can be transformed into a reversible circuit, built from NOT and Toffoli gates, while introducing only constant overheads, every classical circuit can be simply transformed to a quantum circuit of roughly the same size. This shows that quantum circuits are at least as powerful as classical Boolean circuits. In particular, if we have a Boolean function f acting on n-bits $f\colon \{0,1\}^n \to \{0,1\}$, that has a circuit C built from s NOT and AND gates, we can efficiently convert it to a quantum circuit Q that acts as $|b_0, b_1, \ldots, b_n\rangle \mapsto |b_0 \oplus f(b_1, \ldots, b_n), b_1, \ldots, b_n\rangle$, with gate complexity of order s and using at most order s additional *ancilla qubits.* Ancilla qubits are assumed to have $|0\rangle$ state initially, and are required to be returned to $|0\rangle$ after the circuit is applied.

Quantum queries. It is often useful to think about quantum algorithms in a modular fashion, and study how efficiently a problem can be solved using a particular subroutine. Since we want to separate the implementation issues of the subroutine from the higher-level algorithms we often assume that we have access to a quantum subroutine in the form of a *quantum oracle* O, which is essentially a black-box unitary circuit. The only way the quantum circuit is allowed to extract information from such an oracle O is by using it on some of the circuit's qubits. We usually assume that both the oracle and its inverse can be applied in a controlled fashion, i.e., the circuit can use $|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes \mathrm{O}$ and $|0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes \mathrm{O}^\dagger$ gates as well.[2] If O is, say, a 4-qubit unitary, then the corresponding controlled quantum gate is denoted by

$$\text{controlled-O} = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes \mathrm{O} =$$



The *query complexity* of a particular quantum circuit that solves a problem using an oracle O is the number of uses of controlled-O and controlled-O† gates

[1]In the context of vector spaces we will also use the symbol $\oplus$ to denote the direct sum; however, the meaning of the symbol $\oplus$ should always be clear from the context.

[2]For a linear operator A we use $A^\dagger$ to denote its adjoint; A is unitary if and only if $A^\dagger = A^{-1}$.

in the circuit, whereas the gate complexity is the number of additional single- and two-qubit gates used. The *quantum query complexity* of some problem is the minimum query complexity over all quantum circuits that solve the problem (whatever that means in the given context).

In some cases, we also assume access to *quantum read-only memory (QROM)*, which is essentially an oracle that gives cheap access to a vector $x \in \{0,1\}^N$ and for all $i \in \{0,1,\ldots,N-1\}$ provides access to x_i by mapping $|i\rangle|b\rangle \mapsto |i\rangle|b \oplus x_i\rangle$. In some cases when we explicitly say it, we count QROM queries just as simple quantum gates and incorporate them as part of the gate complexity. However, in general building a QROM requires roughly N single- and two-qubit gates.

Grover search and amplitude amplification. In the black-box search problem, we are promised that there is a single good element among N elements, and we have a black box which can tell for each element whether it is good or not. It is not hard to show that classically one needs to make at least around N queries to the black box in order to find the marked element with high probability. However, if we can make quantum queries to the black-box, then roughly $\sqrt{N}$ queries suffice, as shown by Grover's search algorithm [Gro96]. This is essentially optimal, as shown by the hybrid-method argument of Bennett et al. [BBBV97],[3] see also Theorem 4.1.2. Thus the quantum query complexity of black-box search is about $\sqrt{N}$.

Grover's search algorithm follows from a more general technique called *amplitude amplification*. Suppose that we have a black-box unitary $U\colon |0\rangle^{\otimes k} \mapsto |\psi\rangle = \sqrt{p}|1\rangle|\psi_{\text{good}}\rangle + \sqrt{1-p}|0\rangle|\psi_{\text{bad}}\rangle$ that prepares some state $|\psi_{\text{good}}\rangle$ with success probability p, and marks success with a $|1\rangle$ flag qubit. For example, in the search problem one can first prepare a uniform superposition $\frac{1}{\sqrt{N}}\sum_{j=0}^{N-1}|j\rangle$ over all elements, and then make a query. If the marked element is $m \in \{0,1,\ldots,N-1\}$, then the resulting state is

$$\frac{1}{\sqrt{N}}|1\rangle|m\rangle + \frac{\sqrt{N-1}}{\sqrt{N}}|0\rangle\left(\frac{1}{\sqrt{N-1}}\sum_{j\neq m}|j\rangle\right),$$

so that $|\psi_{\text{good}}\rangle = |m\rangle$ is the basis state for the unique marked element. The classical approach for preparing the state with high probability would be to use the procedure roughly $1/p$ times, and stop once the flag qubit is measured in the $|1\rangle$ state. However, the problem can be solved with only roughly $\sqrt{1/p}$ uses of U and $U^\dagger$ with the help of the Grover operator

$$G_U := U\big(2|0\rangle\langle 0|^{\otimes k} - I_k\big)U^\dagger((2|0\rangle\langle 0| - I_1) \otimes I_{k-1}),$$

where I_ℓ denotes the identity operator on ℓ qubits.

[3]Note that this result provides evidence for the widely believed conjecture $\mathsf{NP} \not\subseteq \mathsf{BQP}$.

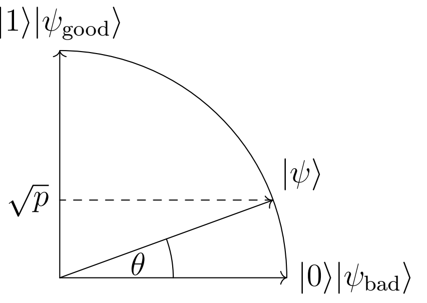

$|1\rangle|\psi_{\text{good}}\rangle$
$|\psi\rangle$
$\sqrt{p}$
θ
$|0\rangle|\psi_{\text{bad}}\rangle$

Figure 1.1. Geometric illustration of the Grover operator G_U.

Let $\theta := \arcsin(\sqrt{p})$, then we get that G_U acts as a rotation by angle 2θ on the subspace spanned by $|1\rangle|\psi_{\text{good}}\rangle$, and $|0\rangle|\psi_{\text{bad}}\rangle$. Indeed, within this subspace $((2|0\rangle\langle 0| - I_1) \otimes I_{k-1})$ is a reflection about $|0\rangle|\psi_{\text{bad}}\rangle$, and $U\big(2|0\rangle\langle 0|^{\otimes k} - I_k\big)U^\dagger$ is a reflection about $|\psi\rangle$. The product of these reflections is a rotation by angle 2θ, which can be seen for example considering the image of $|0\rangle|\psi_{\text{bad}}\rangle$ and $|\psi\rangle$, see Figure 1.1.

Thus $G_U^{\frac{\pi}{4\theta}-\frac{1}{2}}U|0\rangle^{\otimes k} = |1\rangle|\psi_{\text{good}}\rangle$. So applying G_U roughly $\frac{\pi}{4\theta} - \frac{1}{2}$ times solves the problem with high probability. However, if p is only a lower bound on the success probability then this operator might apply too much rotation, and the success probability might be small. This issue can be overcome by applying G_U a random number of times, and repeating the procedure a few times to boost the success probability. A more direct solution to this problem is to use the so-called *fixed-point amplitude amplification* algorithm of Theorem 3.2.4, which provides a single-shot solution without using multiple repetitions.

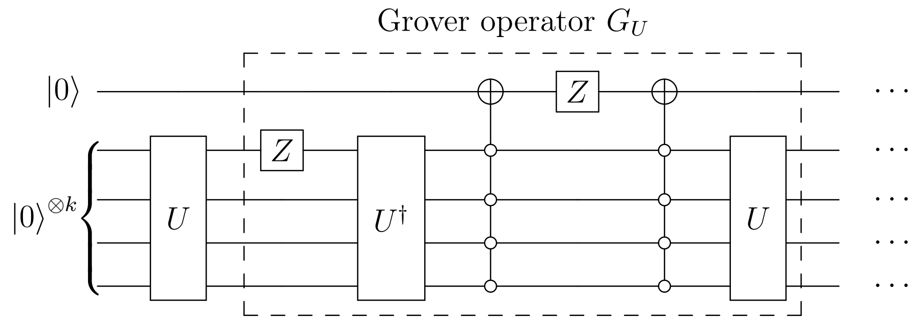


Figure 1.2. Amplitude amplification using the Grover operator. In quantum circuit diagrams the "time flows from left to right", i.e., in the above circuit we start with the quantum state $|0\rangle^{\otimes k}$ and append a single ancilla qubit, then we apply U. The following gates implement the Grover operator G_U with the help of the ancilla qubit. The empty dots denote control on the $|0\rangle$ state and so the $(k+1)$-qubit gates surrounding the second Z gate are reversible OR gates, but we can also think about them as (inverted) generalized Toffoli gates.

Quantum Fourier transform. The *Quantum Fourier Transform (QFT)* on N elements is the quantum analogue of the Discrete Fourier Transform (DFT) on $\mathbb{Z}_N$. If $N = 2^n$, then QFT becomes an n-qubit unitary defined as follows:

$$QFT_n : |j\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k\in\{0,\ldots,2^n-1\}} e^{2\pi i \frac{jk}{N}} |k\rangle.$$

This unitary can be built using about n^2 single- and two-qubit gates, which number can be reduced to about $n\log(n)$ if one only needs a $\frac{1}{\mathrm{poly}(n)}$-approximation. (In contrast the classical Fast Fourier Transform (FFT) algorithm has complexity about $2^n n$, which is exponentially larger.)

The fact that QFT has such an efficient implementation provides the basis of many important quantum algorithms including Shor's famous integer factoring algorithm.

1.4 Mixed quantum states and distances

Sometimes it is useful to work with probabilistic mixtures of pure states, called *mixed quantum states*. It turns out that the right way of representing such mixtures is to use *density operators*. If $(p_i)_{i=1}^k$ is a probability distribution over pure states $|\psi_j\rangle \in \mathbb{C}^n$, then the corresponding density operator $\rho \in \mathbb{C}^{n\times n}$ is

$$\sum_{j=1}^{k} p_j |\psi_j\rangle\langle\psi_j|.$$

Equivalently an n-dimensional mixed quantum state / density operator[4] is a positive-semidefinite matrix of trace 1. An alternative way to think about density operators is to consider a pure state on a larger composite system, and view a density operator as an object that describes the observable state on a subsystem. Mathematically it translates to the fact that for any density operator $\rho \in \mathbb{C}^{n\times n}$ and all $k \geq n$ there exists a $|\psi\rangle \in \mathbb{C}^k \otimes \mathbb{C}^n = \mathcal{H}_A$ such that $\mathrm{Tr}_{\mathcal{H}_A}[|\psi\rangle\langle\psi|] = \rho$, where $\mathrm{Tr}_{\mathcal{H}_A}$ denotes the partial trace over the first subsystem $\mathcal{H}_A = \mathbb{C}^k$. Such a pure state $|\psi\rangle$ is called a *purification* of ρ.

A related concept is *Schmidt decomposition.* A pure state $|\psi\rangle \in \mathcal{H}_1 \otimes \mathcal{H}_2$ can also be viewed as a linear operator $\Psi \in \mathcal{H}_1 \otimes \mathcal{H}_2^*$ via the natural isomorphism between $\mathcal{H}_2$ and its dual vector space $\mathcal{H}_2^*$. The rank of Ψ is called the *Schmidt rank* of $|\psi\rangle$. The singular values ς_j in the singular value decomposition (see Section 1.6) of $\Psi = \sum_{j=1}^k \varsigma_j |\varphi_j\rangle\langle\phi_j|$ are called the *Schmidt coefficients* of $|\psi\rangle$ and $|\psi\rangle = \sum_{j=1}^k \varsigma_j |\varphi_j\rangle|\phi_j\rangle$ is called the *Schmidt decomposition* of $|\psi\rangle$. Finally, the *reduced density operator* of $|\psi\rangle$ on $\mathcal{H}_2$ is $\rho = \mathrm{Tr}_{\mathcal{H}_1}[|\psi\rangle\langle\psi|] = \sum_{j=1}^k \varsigma_j^2 |\phi_j\rangle\langle\phi_j|$.

[4]Note that classical probability distributions correspond to diagonal density operators.

If we measure a mixed quantum state $\rho \in \mathbb{C}^{n\times n}$ in the orthonormal basis $(b_1,\dots,b_n)$, then the probability of measurement outcome j is $\mathrm{Tr}[\rho|b_j\rangle\langle b_j|]$. If $\rho = |\psi\rangle\langle\psi|$, we can see that this recovers the Born rule **(iv)**: $\mathrm{Tr}[|\psi\rangle\langle\psi||b_j\rangle\langle b_j|] = \mathrm{Tr}[\langle b_j|\psi\rangle\langle\psi|b_j\rangle] = |\langle b_j|\psi\rangle|^2$. In general a *projective measurement* is described by a list of k orthogonal projectors $(\Pi_1,\Pi_2,\dots,\Pi_k)$ which sum up to the identity operator in $\mathbb{C}^n$. The probability of measurement outcome j is $p_j = \mathrm{Tr}[\rho\Pi_j]$, and the corresponding post-measurement state is $\Pi_j\rho\Pi_j/p_j$.

An even more general notion of measurement is a *positive operator valued mesurement (POVM)*. Such a measurement is defined by a list of k positive-semidefinite operators $(M_j)_{j=1}^k$ satisfying $\sum_{j=1}^k M_j = I$. The probability of measurement outcome j is $\mathrm{Tr}[\rho M_j]$. We note that every POVM can be implemented by a projective measurement on a larger system.

Quantum channels. A *quantum channel* is a linear map that maps density operators to density operators. The concept of a quantum channel is a common generalization of all allowed physical operations, including coherent (unitary) operations and measurements. A unitary U implements the quantum channel $\rho \mapsto U\rho U^\dagger$, and a projective measurement according to the orthogonal projectors $(\Pi_j)_{j=1}^k$, implements the quantum channel $\rho \mapsto \sum_{j=1}^k \Pi_j\rho\Pi_j$.

All quantum channels can be viewed as the result of the following three-step procedure. First extend ρ to a composite system with a known auxiliary state $|\psi_0\rangle$, i.e., map $\rho \mapsto |\psi_0\rangle\langle\psi_0| \otimes \rho$. Then apply a unitary transformation U on the extended state, and finally take the reduced density operator on the original space by taking the partial trace over the auxiliary system. (A quantum channel can in principle also change the dimensionality of the state, in which case the last step should be slightly modified.)

Distances. For two pure states $|\psi\rangle$ and $|\phi\rangle$ we measure their distances by the ℓ^2-norm of their differences $\||\psi\rangle - |\phi\rangle\| = \sqrt{(\langle\psi| - \langle\phi|)(|\psi\rangle - |\phi\rangle)}$. For two density operators ρ and σ we measure their distance by the *trace distance*, which is defined as $\frac{1}{2}\|\rho-\sigma\|_1$, where for an operator A we define $\|A\|_p$ as the Schatten p-norm, i.e., the ℓ^p-norm of the singular values of A. Finally, for measuring the distance between unitary matrices U and V we use the operator norm $\|U-V\| = \|U-V\|_\infty$ which equals the Schatten ∞-norm.

Unless otherwise stated we use these distance notions for the above objects. In particular if we say that object A is *ε-close* to object B, or that A *ε-approximates* B, we mean that their respective distance is at most ε. Similarly, if we say that the unitary quantum circuit U is an *ε-precise implementation* of the unitary V we mean that $\|U-V\| \le \varepsilon$.

These distance notions have several remarkable properties. For example if $\left\|U-\widetilde{U}\right\| \le \varepsilon$ and $\left\|V-\widetilde{V}\right\| \le \delta$, then $\left\|UV-\widetilde{U}\widetilde{V}\right\| \le \varepsilon+\delta$ as can be seen from $\left\|UV-\widetilde{U}\widetilde{V}\right\| = \left\|UV-\widetilde{U}V+\widetilde{U}V-\widetilde{U}\widetilde{V}\right\| \le \left\|(U-\widetilde{U})V\right\| + \left\|\widetilde{U}(V-\widetilde{V})\right\| \le \varepsilon+\delta.$

Therefore, we also have

$$\left\|\prod_{i=1}^{k} U_i - \prod_{i=1}^{k} \widetilde{U}_i\right\| \leq \sum_{i=1}^{k}\left\|U_i - \widetilde{U}_i\right\|.$$

The trace distance extends the notion of total variation distance on probability distributions. Indeed, for classical distributions the corresponding density operators are diagonal, and it is clear that the trace distance corresponds to the total variation distance. Moreover, the trace distance between ρ and σ is the maximal total variation distance between the measurement statics corresponding to arbitrary POVM measurements. Indeed if $(M_1, M_2, \ldots, M_k)$ is a POVM, then

$$\frac{1}{2}\sum_{i=1}^{k}|\mathrm{Tr}[\rho M_i]-\mathrm{Tr}[\sigma M_i]|=\frac{1}{2}\sum_{i=1}^{k}|\mathrm{Tr}[(\rho-\sigma)M_i]| \leq \frac{1}{2}\sum_{i=1}^{k}\mathrm{Tr}[|\rho-\sigma|M_i]=\frac{1}{2}\|\rho-\sigma\|_1.$$

Let Π be the orthogonal projector to the subspace of positive eigenvalues of $\rho-\sigma$, then for the binary POVM $(\Pi, I-\Pi)$ the above inequality becomes an equality.

The trace distance between two pure states $|\psi\rangle$,$|\phi\rangle$ is always upper bounded by their ℓ^2-distance, because

$$\frac{1}{2}\||\psi\rangle\langle\psi| - |\phi\rangle\langle\phi|\|_1 = \sqrt{\mathrm{Tr}\left[(|\psi\rangle\langle\psi| - |\phi\rangle\langle\phi|)^2\right]/2} = \sqrt{1-|\langle\psi|\phi\rangle|^2} \leq \||\psi\rangle - |\phi\rangle\|,$$

where the first equality follows from the fact that $|\psi\rangle\langle\psi| - |\phi\rangle\langle\phi|$ has trace 0 and rank at most two, therefore it can have at most two non-zero eigenvalues, which must sum to 0.

These observations make it easy to argue about the approximate correctness of quantum algorithms. For example if a quantum algorithm consists of k steps $(U_i)_{i=1}^{k}$, and outputs a desired answer with probability at least $\frac{5}{6}$, then it is enough to implement each step U_i to precision $1/(6k)$. The resulting quantum circuit will still output a correct answer with probability at least $2/3$, since the final states (before measurement) will differ by at most $1/6$, and thus the output statistics (after the measurement) will differ by at most $1/6$ too.

1.5 Hamiltonians and Gibbs states

The Hamiltonian H of a d-dimensional quantum system is a Hermitian matrix $H \in \mathbb{C}^{d\times d}$ describing the quantum system's physical characteristics. If these characteristics do not change in time, then the Hamiltonian is time-independent, and the quantum states of the system evolve in time according to the unitary map e^{-itH}, as follows from Schrödinger's equation[5]

$$\frac{d}{dt}|\psi(t)\rangle = -iH|\psi(t)\rangle.$$

[5]For the sake of simplicity we ignore the factor $\hbar$, i.e., work with units where it becomes 1.

The eigenvalues $(\lambda_j)_{j=1}^d$ of H are the *energy levels* of the quantum system, and the corresponding eigenvectors are also called *energy eigenstates.* The smallest energy level is called the *ground-state energy* and the corresponding eigenvector(s) is(are) called *ground state(s).* The other energy eigenstates and their superpositions are sometimes called *excited states.*

In order to physically implement, say, a single-qubit rotation gate $e^{-i\theta Z}$, one could for example turn on some external interaction which changes the Hamiltonian from 0 to Z, and apply it for time θ, then disable the external interaction to reset the Hamiltonian to 0. However, in this dissertation we will not consider the question of physical implementation, but will still work with Hamiltonians due to their important connections to computer science, which we discuss in more detail in Chapter 7.

Finally, we note that a quantum system in thermal equilibrium at finite inverse temperature β has density operator $e^{-\beta H}/\mathcal{Z}$, where $\mathcal{Z} = \mathrm{Tr}\left[e^{-\beta H}\right]$ is the partition function. Such a mixed quantum state is called a *Gibbs state.* As temperature goes to 0, β goes to infinity and the Gibbs state converges to the ground state (or to a uniform mixture of the ground states in case there are multiple ground states). Apart from its physical relevance, it turns out that Gibbs states can play an important role in convex optimization, as we discuss in Chapter 6.

1.6 Singular Value Decomposition (SVD)

It is well known that for every matrix $A \in \mathbb{C}^{m\times n}$ there exists a pair of unitaries $W \in \mathbb{C}^{m\times m}$, $V \in \mathbb{C}^{n\times n}$ and $\Sigma \in \mathbb{R}^{m\times n}$ such that Σ is a diagonal matrix with non-negative non-increasing entries on the diagonal, and $A = W\Sigma V^\dagger$. Such a decomposition is called a *singular value decomposition* of A. Let $k := \min[m, n]$, then we use $\varsigma_i := \Sigma_{ii}$ for $i \in [k]$ to denote the *singular values* of A, which are the diagonal elements of Σ. The columns of V are called the *right singular vectors* of $|\psi\rangle$, and the columns of W are called the *left singular vectors.*

In the next chapter we often define the matrix A as the product of two orthogonal projectors $\widetilde{\Pi}, \Pi$ and unitary U such that $A = \widetilde{\Pi} U \Pi$. In such cases we will assume without loss of generality that the first $\mathrm{rank}(\widetilde{\Pi})$ left singular vectors span $\mathrm{img}(\widetilde{\Pi})$ and the first $\mathrm{rank}(\Pi)$ right singular vectors span $\mathrm{img}(\Pi)$.

1.7 Some notation

For a number $n \in \mathbb{N}$ we use the notation $[n] := \{1, 2, \ldots, n\}$. We write δ_{ij} for the Kronecker delta function, which is 1 if $i = j$, and 0 otherwise. We use e_j for the jth basis vector in the standard basis when the dimension of the space is clear from context. We denote by $\mathrm{End}(\mathcal{H})$ the set of linear maps $\mathcal{H} \to \mathcal{H}$.

We use the following definition for $\widetilde{\mathcal{O}}$:

$$\widetilde{\mathcal{O}}_{d,e}(f(a,b,c)) := \mathcal{O}(f(a,b,c) \cdot \mathrm{polylog}(f(a,b,c),d,e)),$$

and define $\tilde{\Omega}$ in a similar way for lower bounds and $\tilde{\Theta}$ as the intersection of the two.

Polynomials and functions. In this dissertation we will often work with polynomial approximations, and therefore we introduce some related notation. We denote the set of real polynomials in variable x with $\mathbb{R}[x]$, and similarly the complex polynomials with $\mathbb{C}[x]$. Let $P \in \mathbb{C}[x]$ be a complex polynomial $P(x) = \sum_{j=0}^{k} a_j x^j$, then we denote by $P^*(x) := \sum_{j=0}^{k} a_j^* x^j$ the polynomial with conjugated coefficients. The *real part* of P is denoted by $\Re[P](x) := \sum_{j=0}^{k} \Re[a_j] x^j$ which we get by taking the real part of the coefficients, similarly we define the *imaginary part* of P as $\Im[P](x) := \sum_{j=0}^{k} \Im[a_j] x^j$. We say that P *is even* if all coefficients corresponding to odd powers of x are 0, and similarly we say that P *is odd* if all coefficients corresponding to even powers of x are 0. For an integer number $z \in \mathbb{Z}$ we say that P has parity-(z mod 2) (or simply parity-z) if z is even and P is even or z is odd and P is odd. We will denote by $T_d \in \mathbb{R}[x]$ the d-th *Chebyshev polynomial* of the first kind, defined by $T_d(x) := \cos(d \arccos(x))$.

For a function $f : \mathbb{R} \to \mathbb{C}$ and a subset $I \subseteq \mathbb{R}$ we use the notation $\|f\|_I := \sup_{x \in I} |f(x)|$ to denote the sup-norm of the function f on the domain I. We say that a function $f\colon \mathbb{R} \to \mathbb{C}$ *is even* if for all $x \in \mathbb{R}$ we have $f(-x) = f(x)$, and that it *is odd* if for all $x \in \mathbb{R}$ we have $f(-x) = -f(x)$. The *even part* of a function is defined as $(f(x) + f(-x))/2$ and the *odd part* of a function is defined as $(f(x) - f(-x))/2$ which gives the unique decomposition of f to a sum of an even and an odd function.

We define the *sign function* $\mathrm{sign}(x)$ as

$$\mathrm{sign}(x) := \begin{cases} -1 & \text{for } x < 0 \\ 0 & \text{for } x = 0 \\ 1 & \text{for } x > 0. \end{cases}$$

Matrices and matrix functions. Whenever we present a matrix and put a . in some place we mean a matrix with arbitrary values of the elements in the unspecified block. For example [.] just denotes a matrix with completely arbitrary elements, similarly

$$U = \begin{bmatrix} A & . \\ . & . \end{bmatrix}$$

denotes an arbitrary matrix whose top-left block is A.

For a Hermitian matrix H we write $\operatorname{Spec}(H)$ for its spectrum (set of eigenvalues). For a function $f : \mathbb{R} \to \mathbb{R}$ we write $f(H)$ for the matrix we get by applying f to the eigenvalues of H, so that if U is a unitary diagonalizing H, then

$$f(H) = U \begin{bmatrix} f(\lambda_1) & & \\ & \ddots & \\ & & f(\lambda_n) \end{bmatrix} U^\dagger \text{ where } H = U \begin{bmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{bmatrix} U^\dagger.$$

Chapter 2

Quantum singular value transformation

Quantum computing is powerful because unitary operators describing the time-evolution of a quantum system have exponential size in terms of the number of qubits present in the system. In this chapter we develop a new “Quantum singular value transformation” algorithm capable of harnessing this exponential advantage, that can apply polynomial transformations to the singular values of a block of a unitary, generalizing the optimal Hamiltonian simulation results of Low and Chuang [LC17a]. This quantum “meta-algorithm” generalizes several quantum algorithmic constructions. Moreover, the proposed quantum circuits have a very simple structure, often give rise to optimal algorithms and have appealing constant factors, while typically only using a constant number of ancilla qubits.

2.1 Introduction

It is often said in quantum computing that there are only a few quantum algorithms [Sho03] that are known to give speed-ups over classical computers. While this is true, a remarkable number of applications stem from these primitives. The first class of quantum speedups is derived from quantum simulation, which was originally proposed by Feynman [Fey82]. Such algorithms yield exponential speedups over the best known classical methods for simulating quantum dynamics, as well as probing electronic structure problems in material science and chemistry. The two most influential quantum algorithms developed later in the ’90s are Shor’s algorithm [Sho97] (based on quantum Fourier transform) and Grover’s search [Gro96]. Other examples have emerged over the years, however, arguably the quantum walk algorithm of Ambainis [Amb04] and Szegedy [Szeg04] together with the quantum linear systems algorithm of Harrow, Hassidim and Lloyd [HHL09] are the most important other primitives that provide speed-ups

This chapter is based on [GSLW19].

relative to classical computing. An important question that remains is "are these primitives truly independent or can they be seen as examples of a deeper underlying concept?" The aim of this chapter, and the first part of the dissertation, is to provide an argument that a wide array of techniques from these disparate fields can all be seen as manifestations of a single quantum idea that we call "quantum singular value transformation" generalizing all the above-mentioned techniques except for quantum Fourier transform (and thus Shor's algorithm).

Of the aforementioned quantum algorithms, quantum simulation is arguably the most diverse and rapidly developing. Within the last few years a host of techniques have been developed that have led to ever more powerful methods [CMN$^+$17]. The problem in quantum simulation fundamentally is to take an efficient description of a Hamiltonian H, an evolution time t, and an error tolerance ε, and find a quantum operation V such that $\left\|e^{-iHt} - V\right\| \leq \varepsilon$ where the implementation of V should use as few resources as possible. The first methods introduced to solve this problem were Trotter formula decompositions [Llo96, BACS07] and subsequently methods based on linear combinations of unitaries were developed [CW12] to provide better asymptotic scaling of the cost of simulation.

An alternative strategy was also developed concurrently with these methods that used ideas from quantum walks. Asymptotically, this approach is perhaps the favored method for simulating time-independent Hamiltonians because it is capable of achieving near-optimal scaling with all relevant parameters. The main tool developed for this approach is a walk operator that has eigenvalues $e^{-i\arcsin(E_k/\alpha)}$ where E_k is the k^{th} eigenvalue of H and α is a normalizing parameter. While early work adjusted the spectrum to recover the desired eigenvalues e^{-iE_k} by using phase estimation to invert the arcsin, subsequent work achieved better scaling using linear combination of quantum walk steps [BCK15]. Recently another approach, called *qubitization* [LC16, LC17a], was introduced to transform the spectrum in a more efficient manner based on a technique called *quantum signal processing* [LYC16].

Quantum simulation is not the only field that uses such spectral transformations. Quantum linear systems algorithms [HHL09], as well as algorithms for semi-definite programming [BS17, vAGGdW17], use these ideas extensively. Earlier work on linear systems used a strategy similar to the quantum walk simulation method: use phase estimation to estimate the eigenvalues of a matrix λ_j and then use quantum rejection sampling to rescale the amplitude of each eigenvector $|\lambda_j\rangle$ via the map $|\lambda_j\rangle \mapsto \lambda_j^{-1}|\lambda_j\rangle$. This enacts the inverse of a matrix, and generalizations to the pseudoinverse are straightforward. More recent methods eschew the use of phase estimation in favor of linear combination of unitary methods [CKS17] which typically approximate the inversion using a Fourier series or Chebyshev series expansion. Similar ideas can be used to prepare Gibbs states efficiently [CS17, vAGGdW17].

These improvements typically result in exponentially improved scaling in terms of precision in various important subroutines. However, since these techniques work on quantum states, and usually one needs to learn certain properties of these states to a specified precision ε, a polynomial dependence on $\frac{1}{\varepsilon}$ is unavoidable. Therefore these improvements typically "only" result in polynomial savings in the complexity. Nevertheless, for complex algorithms this can make a huge difference. These techniques played a crucial role in improving the complexity of quantum semi-definite program solvers, where the scaling with accuracy was improved from the initial $\mathcal{O}(1/\epsilon^{32})$ of [BS17] to $\mathcal{O}(1/\epsilon^4)$, see Chapter 6.

A unifying perspective. We develop a new technique that we call *quantum singular value transformation*, which unifies qubitization and quantum signal processing. The central object for this result is *projected unitary encoding*, which is defined as follows. Suppose that $\widetilde{\Pi}$,Π are orthogonal projectors and U is a unitary, then we say that the unitary U and the projectors $\widetilde{\Pi}$,Π form a projected unitary encoding of $A := \widetilde{\Pi} U \Pi$. The encoding is called *symmetric* if $\widetilde{\Pi} = \Pi$.

Roughly speaking qubitization turns a symmetric projected unitary encoding $\Pi U \Pi = H$ of a Hermitian operator H into a unitary V which is the square-root of a (Szegedy-type) quantum walk operator $U(2\Pi - I)U^\dagger(2\Pi - I)$, so that each eigenvector $|\psi\rangle$ of H with eigenvalue λ becomes a superposition of two[1] eigenvectors $|\psi^\pm\rangle$ of V with eigenvalues $e^{\pm i \arccos \lambda}$. If U happens to be a Hermitian unitary (i.e., a *reflection operator*), then one can use $V := U(2\Pi - I)$; otherwise one can replace U by a suitable Hermitian unitary $\widetilde{U}$ using a controlled-U and controlled-$U^\dagger$ gate. Finally, using quantum signal processing one can transform the spectrum of the unitary V, resulting in a new circuit U', which is a projected unitary encoding of a transformed operator $P(H)$, see Figure 2.1.

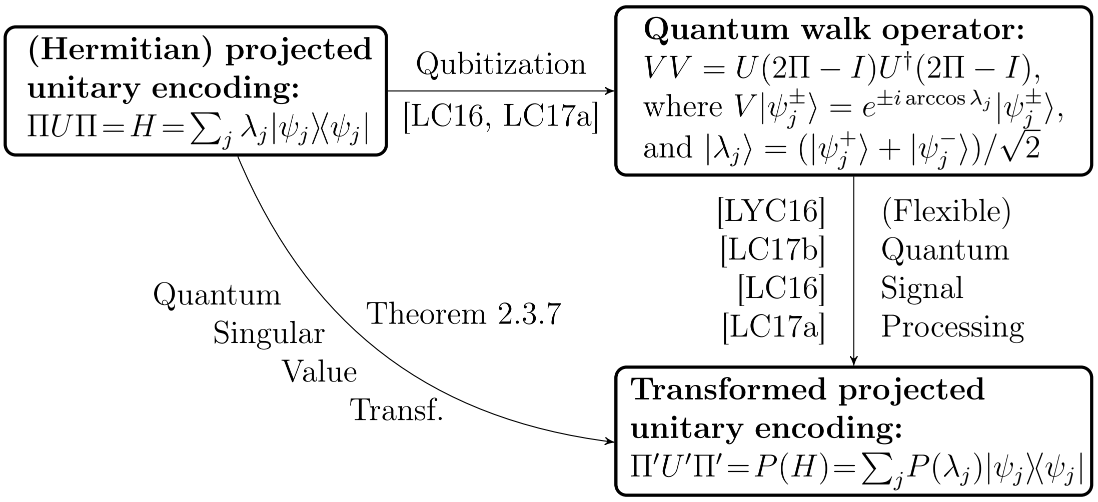

(Hermitian) projected unitary encoding:
$\Pi U \Pi = H = \sum_j \lambda_j |\psi_j\rangle\langle\psi_j|$
Qubitization
[LC16, LC17a]
Quantum walk operator:
$VV = U(2\Pi - I)U^\dagger(2\Pi - I)$,
where $V|\psi_j^\pm\rangle = e^{\pm i \arccos \lambda_j}|\psi_j^\pm\rangle$,
and $|\lambda_j\rangle = (|\psi_j^+\rangle + |\psi_j^-\rangle)/\sqrt{2}$
[LYC16]
[LC17b]
[LC16]
[LC17a]
(Flexible) Quantum Signal Processing
Quantum Singular Value Transf.
Theorem 2.3.7
Transformed projected unitary encoding:
$\Pi' U' \Pi' = P(H) = \sum_j P(\lambda_j)|\psi_j\rangle\langle\psi_j|$

Figure 2.1. Schematic comparison of QSVT with previous approaches.

[1]This justifies the name qubitization: each eigenvector of H splits into two eigenvectors of V, and so the resulting two-dimensional subspace is isomorphic to a qubit.

We generalize and unify the above two steps in quantum singular value transformation by directly implementing the transformation on the singular values of H. Our result works for arbitrary matrices and does not require symmetric encodings, (i.e., $\widetilde{\Pi} \neq \Pi$ is allowed). Moreover, we do not need U to be Hermitian, and in case P is an even or odd polynomial, we do not even require access to controlled versions of U or $U^\dagger$.

Quantum Singular Value Transformation (QSVT). We define singular value transformation by a polynomial $P \in \mathbb{C}[x]$ in the following way: if P is an odd polynomial and $A = W\Sigma V^\dagger$ is a singular value decomposition (SVD), then $P^{(SV)}(A) := WP(\Sigma)V^\dagger$, where the polynomial P is applied to the diagonal entries of Σ. Our main result is that for any degree-d odd polynomial $P \in \mathbb{R}[x]$ that is bounded by 1 in absolute value on $[-1,1]$, we can implement a unitary U_Φ with a simple circuit using U and its inverse a total number of d times such that

$$A = \widetilde{\Pi} U \Pi \Longrightarrow P^{(SV)}(A) = \widetilde{\Pi} U_\Phi \Pi. \tag{2.1}$$

We prove a similar result for even polynomials as well, but with replacing $\widetilde{\Pi}$ by Π in the above equation, and defining $P^{(SV)}(A) := VP(\Sigma)V^\dagger$ for even polynomials.

The result is based on quantum signal processing, which was introduced by Low, Yoder and Chuang [LYC16]. They asked the following question: suppose we can implement single-qubit gate sequences of the form

$$e^{i\phi_0\sigma_z}W(x)e^{i\phi_1\sigma_z}W(x)e^{i\phi_2\sigma_z}\cdot\ldots\cdot W(x)e^{i\phi_k\sigma_z},$$

where

$$W(x) := \begin{bmatrix} x & i\sqrt{1-x^2} \\ i\sqrt{1-x^2} & x \end{bmatrix} = e^{i\arccos(x)\sigma_x}$$

is a "signal unitary". Which single-qubit unitary operators can we build this way if we can choose the angles $\phi_0, \phi_1, \ldots, \phi_k$ arbitrarily, but $x \in [-1,1]$ is unknown to us? They give a characterization of the unitary operators that can be constructed this way, and find that the set of achievable unitary operators is quite rich, for example it includes single-qubit gates of the form

$$\begin{bmatrix} P(x) & . \\ . & . \end{bmatrix},$$

where the real part of P can be an arbitrary polynomial that maps $[-1,1]$ to $[-1,1]$ and has degree and parity matching the number of uses of $W(x)$. We provide a self-contained treatment of quantum signal processing, then show that the analysis can be extended to the case when x is replaced by a (possibly rectangular) matrix of norm at most 1, if the phase gates are properly extended and the signal unitary and its inverse are applied alternatingly, see Figure 2.2 and Figure 3.1. With a bit of additional work it yields the above described quantum singular value transformation result of Eq. (2.1). Based on this result we will be able to quickly recover well-optimized variants of several prominent quantum algorithms in Chapter 3, and derive new quantum algorithms in the later chapters.

Projector-controlled NOT gate. For an orthogonal projector Π we will frequently use the Π-controlled NOT gate, denoted by $\mathrm{C}_\Pi\mathrm{NOT}$, which flips the value of the first (single-qubit) register whenever the state of the second register is in the image of Π. For example if $\Pi = |1\rangle\langle 1|$ is the projector to the single-qubit basis state $|1\rangle$, then we just get back the CNOT gate controlled by the second qubit.

2.1.1. Definition ($\mathrm{C}_\Pi\mathrm{NOT}$ gate). For an orthogonal projector Π let us define the Π-controlled NOT gate as the unitary operator

$$\mathrm{C}_\Pi\mathrm{NOT} := X \otimes \Pi + I \otimes (I - \Pi).$$

This operator can be used for instance for implementing a coherent (unitary) analogue of a projective measurement. Also note that up to a conjugation by a Hadamard gate on the first qubit, this gate is the same as the controlled reflection $(I - 2\Pi)$ gate.

2.2 Quantum signal processing

The methods in this section are based on the so-called "Quantum Signal Processing" techniques introduced by Low, Yoder and Chuang [LYC16]. We present a self-contained treatment of these techniques, significantly streamline the formalism, and develop slightly improved versions of the results presented in [LYC16]. As a corollary of the results we also develop Corollary 2.2.6-2.2.8, which will be the only results that we need in the rest of the dissertation. We suggest the first-time reader to skip the proofs of this section, as they are not necessary for understanding the main ideas of Section 2.3 and the later chapters.

2.2.1 Parametrized SU(2) unitaries induced by Pauli rotations

In this section we review the results of Low, Yoder and Chuang [LYC16], who show how to build 2×2 unitary matrices whose entries are trigonometric polynomials by taking products of various rotation and phase gates. They consider essentially the following problem, which they call "Quantum Signal Processing": suppose one can apply a gate sequence

$$e^{i\phi_0\sigma_z}e^{i\theta\sigma_x}e^{i\phi_1\sigma_z}e^{i\theta\sigma_x}e^{i\phi_2\sigma_z}\cdot\ldots\cdot e^{i\theta\sigma_x}e^{i\phi_k\sigma_z}, \tag{2.2}$$

where θ is unknown (they call $e^{i\theta\sigma_x}$ the "signal unitary") but one has control over the angles $\phi_0, \phi_1, \ldots, \phi_k$; which unitary operators can we build this way? They give a characterization of the unitary operators that can be constructed this way, and find that the set of achievable unitary operators is quite rich.

We find it more useful to work with the above matrices using a slightly modified parametrization. For $x \in [-1,1]$ let us define

$$W(x) := \begin{bmatrix} x & i\sqrt{1-x^2} \\ i\sqrt{1-x^2} & x \end{bmatrix} = e^{i\arccos(x)\sigma_x}.$$

It is easy to see that if $\theta \in [0,\pi]$, then by setting $x := \cos(\theta)$ Eq. (2.2) can be rewritten as

$$e^{i\phi_0\sigma_z}W(x)e^{i\phi_1\sigma_z}W(x)e^{i\phi_2\sigma_z}\cdot\ldots\cdot W(x)e^{i\phi_k\sigma_z}. \tag{2.3}$$

Now we present the characterization of Low et al. [LYC16] using the above formalism. Our formulation makes the statement simpler and reduces the number of cases. We also present a succinct simplified proof which can be conveniently described using our formalism.

2.2.1. Theorem. *Let $k \in \mathbb{N}$ and $\Phi = (\phi_0, \phi_1, \ldots, \phi_k) \in \mathbb{R}^{k+1}$. Then there exist $P, Q \in \mathbb{C}[x]$ such that for all $x \in [-1,1]$ we have*

$$e^{i\phi_0\sigma_z}\prod_{j=1}^{k}\left(W(x)e^{i\phi_j\sigma_z}\right) = \begin{bmatrix} P(x) & iQ(x)\sqrt{1-x^2} \\ iQ^*(x)\sqrt{1-x^2} & P^*(x) \end{bmatrix}, \tag{2.4}$$

and

(i) $\deg(P) \leq k$, $\deg(Q) \leq k-1$ *and*

(ii) P has parity-(k mod 2), Q has parity-($k-1$ mod 2) and

(iii) $\forall x \in [-1,1]\colon |P(x)|^2 + (1-x^2)|Q(x)|^2 = 1.$

Moreover, if $P, Q \in \mathbb{C}[x]$ satisfy (i)-(iii), then we can find a $\Phi \in \mathbb{R}^{k+1}$ satisfying Eq. (2.4).

Proof:
"$\Longrightarrow$": For the $k = 0$ case the unitary on the left-hand side of (2.4) is $e^{i\phi_0\sigma_z}$, so that $P \equiv e^{i\phi_0}$ and $Q \equiv 0$ satisfy the properties (i)-(iii). Now we prove (i)-(ii) by induction. The induction step can be shown as follows: suppose we have proved for $k-1$ that

$$e^{i\phi_0\sigma_z}\prod_{j=1}^{k-1}\left(W(x)e^{i\phi_j\sigma_z}\right) = \begin{bmatrix} \tilde{P}(x) & i\tilde{Q}(x)\sqrt{1-x^2} \\ i\tilde{Q}^*(x)\sqrt{1-x^2} & \tilde{P}^*(x) \end{bmatrix},$$

where $\tilde{P}, \tilde{Q} \in \mathbb{C}[x]$ satisfy (i)-(ii). Then

$$e^{i\phi_0\sigma_z}\prod_{j=1}^{k}\left(W(x)e^{i\phi_j\sigma_z}\right) =$$
$$= \begin{bmatrix} \tilde{P}(x) & i\tilde{Q}(x)\sqrt{1-x^2} \\ i\tilde{Q}^*(x)\sqrt{1-x^2} & \tilde{P}^*(x) \end{bmatrix}\begin{bmatrix} e^{i\phi_k}x & ie^{-i\phi_k}\sqrt{1-x^2} \\ ie^{i\phi_k}\sqrt{1-x^2} & e^{-i\phi_k}x \end{bmatrix}$$
$$= \begin{bmatrix} \overbrace{e^{i\phi_k}\Big(x\tilde{P}(x)+(x^2-1)\tilde{Q}(x)\Big)}^{P(x):=} & ie^{-i\phi_k}\Big(x\tilde{Q}(x)+\tilde{P}(x)\Big)\sqrt{1-x^2} \\ \underbrace{ie^{i\phi_k}\Big(x\tilde{Q}^*(x)+\tilde{P}^*(x)\Big)}_{Q^*(x):=}\sqrt{1-x^2} & e^{-i\phi_k}\Big(x\tilde{P}^*(x)+(x^2-1)\tilde{Q}^*(x)\Big) \end{bmatrix}, \tag{2.5}$$

and it is easy to see that P, Q satisfy (i)-(ii). Finally note that the left-hand side of (2.4) is a product of unitaries, therefore the right-hand side is unitary too, which implies (iii).

"$\Longleftarrow$": Suppose P, Q satisfy (i)-(iii). First we handle a trivial case: suppose that $\deg(P) = 0$, then due to (iii) we must have that $|P(1)| = 1$ and thus $P \equiv e^{i\phi_0}$ for some $\phi_0 \in \mathbb{R}$. This again using (iii) implies that $Q \equiv 0$. Due to (ii) we must have that k is even, and thus $\Phi = (\phi_0, \frac{\pi}{2}, -\frac{\pi}{2}, \ldots, \frac{\pi}{2}, -\frac{\pi}{2}) \in \mathbb{R}^{k+1}$ is a solution, since

$$e^{i\phi_0\sigma_z}\prod_{j=1}^{k/2}\left(W(x)e^{i\frac{\pi}{2}\sigma_z}W(x)e^{-i\frac{\pi}{2}\sigma_z}\right) = e^{i\phi_0\sigma_z} = \begin{bmatrix} e^{i\phi_0} & 0 \\ 0 & e^{-i\phi_0} \end{bmatrix}.$$

This special case also covers the $k = 0$ case, providing the base of our induction.

Now we show the induction step, assuming that we proved the claim for $k-1$. Note that (iii) can be rewritten as

$$\forall x \in [-1,1]\colon P(x)P^*(x) + (1-x^2)Q(x)Q^*(x) = 1. \tag{2.6}$$

Since this equation holds for infinitely many points, the polynomial on the left-hand side of (2.6) must be the constant $\equiv 1$ polynomial. Assume without loss of generality that $1 \le \deg(P) = \ell \le k$, then we must have that $\deg(Q) = \ell - 1$, and the leading coefficients have equal magnitude $|p_\ell| = |q_{\ell-1}|$, since they cancel each other in (2.6). Let $\phi_k\in\mathbb{R}$ be such that $e^{2i\phi_k}=\frac{p_\ell}{q_{\ell-1}}$, and let us define $\tilde{P},\tilde{Q}$ via

$$\begin{bmatrix}\tilde{P}(x) & i\tilde{Q}(x)\sqrt{1-x^2}\\ i\tilde{Q}^*(x)\sqrt{1-x^2} & \tilde{P}^*(x)\end{bmatrix}:=\begin{bmatrix}P(x) & iQ(x)\sqrt{1-x^2}\\ iQ^*(x)\sqrt{1-x^2} & P^*(x)\end{bmatrix}e^{-i\phi_k\sigma_z}W^\dagger(x)=$$

$$=\begin{bmatrix}P(x) & iQ(x)\sqrt{1-x^2}\\ iQ^*(x)\sqrt{1-x^2} & P^*(x)\end{bmatrix}\begin{bmatrix}e^{-i\phi_k}x & -ie^{-i\phi_k}\sqrt{1-x^2}\\ -ie^{i\phi_k}\sqrt{1-x^2} & e^{i\phi_k}x\end{bmatrix}$$

$$=\begin{bmatrix}\overbrace{e^{-i\phi_k}xP(x)+e^{i\phi_k}(1-x^2)Q(x)}^{\tilde{P}(x)=} & i\tilde{Q}(x)\sqrt{1-x^2}\\ i\underbrace{\left(e^{-i\phi_k}xQ^*(x)-e^{i\phi_k}P^*(x)\right)}_{\tilde{Q}^*(x)=}\sqrt{1-x^2} & \tilde{P}^*(x)\end{bmatrix}\tag{2.7}$$

where

$$\tilde{P}(x)=e^{-i\phi_k}xP(x)+e^{i\phi_k}(1-x^2)Q(x)=e^{-i\phi_k}\left(xP(x)+\frac{p_\ell}{q_{\ell-1}}(1-x^2)Q(x)\right)\tag{2.8}$$

and

$$\tilde{Q}(x)=e^{i\phi_k}xQ(x)-e^{-i\phi_k}P(x)=e^{-i\phi_k}\left(\frac{p_\ell}{q_{\ell-1}}xQ(x)-P(x)\right).\tag{2.9}$$

It is easy to see that the highest-order terms in (2.8)-(2.9) cancel out, and therefore $\deg(\tilde{P})\leq\ell-1\leq k-1$, $\deg(\tilde{Q})\leq\ell-2\leq k-2$. Using (2.8)-(2.9) we can also verify that $\tilde{P},\tilde{Q}$ satisfy (i)-(ii) for $(k-1)$, moreover property (iii) is preserved due to the unitarity of $e^{-i\phi_k\sigma_z}W^\dagger(x)$. So by the induction hypothesis we get that (2.7) equals $e^{i\tilde{\phi}_0\sigma_z}\left(\prod_{j=1}^{k-1}W(x)e^{i\tilde{\phi}_j\sigma_z}\right)$ for some $\tilde{\Phi}\in\mathbb{R}^k$, therefore $\Phi:=(\tilde{\phi}_0,\tilde{\phi}_1,\tilde{\phi}_2,\ldots,\tilde{\phi}_{k-1},\phi_k)\in\mathbb{R}^{k+1}$ is a solution. □

Note that the above proof also gives an algorithm that finds Φ using $\mathcal{O}(k^2)$ arithmetic operations. The following two characterizations and their proofs also follow a constructive approach which can be translated to a polynomial-time algorithm. However, they have the drawback that they rely on finding roots of high-degree polynomials,[2] which makes it harder in practice to execute the resulting protocols.

In our algorithms we will mostly be concerned with the polynomial P, corresponding to the top-left corner of the matrices, and ignore Q. Therefore, it is useful to characterize the attainable polynomials P if one can choose Q arbitrarily in Theorem 2.2.1. These attainable polynomials can be characterized as follows.

[2]For a good bound on the complexity of approximate root finding see, e.g., the work of Neff and Reif [NR96].

2.2.2. THEOREM. *Let $k \in \mathbb{N}$ be fixed and let $P \in \mathbb{C}[x]$. There exists some $Q \in \mathbb{C}[x]$ such that P, Q satisfy properties (i)-(iii) of Theorem 2.2.1 if and only if P satisfies properties (i)-(ii) of Theorem 2.2.1 and*

(iv.a) $\forall x \in [-1,1]\colon |P(x)| \leq 1$

(iv.b) $\forall x \in (-\infty,-1] \cup [1,\infty)\colon |P(x)| \geq 1$

(iv.c) if k is even, then $\forall x \in \mathbb{R}\colon P(ix)P^*(ix) \geq 1$.

Similarly, let $Q \in \mathbb{C}[x]$. There exists some $P \in \mathbb{C}[x]$ such that P, Q satisfy properties (i)-(iii) of Theorem 2.2.1 if and only if Q satisfies properties (i)-(ii) of Theorem 2.2.1 and

(v.a) $\forall x \in [-1,1]\colon \sqrt{1-x^2}|Q(x)| \leq 1$

(v.b) if k is odd, then $\forall x \in \mathbb{R}\colon (1+x^2)Q(ix)Q^*(ix) \geq 1$.

Proof:
"$\Longrightarrow$": Trivially follows from (iii): $\forall x \in \mathbb{C}\colon P(x)P^*(x) + (1-x^2)Q(x)Q^*(x) = 1$.
"$\Longleftarrow$": First consider the case when k is odd, and consider the polynomial $A(x) := 1 - P(x)P^*(x)$. Note that $A \in \mathbb{R}[x]$ and A is even, therefore A is in fact a polynomial in x^2. Let $y = x^2$ and consider the real polynomial $\tilde{A}(y) := A(\sqrt{y})$. Observe that $\forall y \geq 1\colon \tilde{A}(y) \leq 0$ due to (iv.b), $\forall y \in [0,1]\colon \tilde{A}(y) \geq 0$ due to (iv.a) and $\forall y \leq 0\colon \tilde{A}(y) \geq 1$ since

$$\begin{aligned}\tilde{A}(y) &= A(i\sqrt{-y}) && (y \leq 0)\\ &= 1 - P(i\sqrt{-y})P^*(i\sqrt{-y}) = 1 + P(i\sqrt{-y})P^*(-i\sqrt{-y}) && (P \text{ is odd})\\ &= 1 + P(i\sqrt{-y})(P(i\sqrt{-y}))^* = 1 + |P(i\sqrt{-y})|^2 \geq 1.\end{aligned}$$

Therefore all real roots have even multiplicity except for $x = 1$, moreover all complex roots come in pairs. Thus $\tilde{A}(y) = (1-y)K^2\prod_{s\in S}(y-s)(y-s^*)$ for some $K \in \mathbb{R}$ and multiset $S \subseteq \mathbb{C}$ of roots. Let $W(y) := K\prod_{s\in S}(y-s) \in \mathbb{C}[y]$, then $\tilde{A}(y) = (1-y)W(y)W^*(y)$, and thus $A(x) = (1-x^2)W(x^2)W^*(x^2)$, i.e., $1 = P(x)P^*(x) + (1-x^2)W(x^2)W^*(x^2)$. Setting $Q(x) := W(x^2)$ concludes this case.

The other cases can be proven similarly, by examining the polynomial $1 - P(x)P^*(x)$ or $1 - (1-x^2)Q(x)Q^*(x)$ respectively. □

The above characterization is still somewhat complicated, but it turns out that there is a nice and simple characterization for the polynomials that we can get by taking the real (or imaginary) part of P. It turns out that we can work very smoothly with the real part of P in our quantum circuits, so we will mostly build our algorithmic results on this characterization, proven below.

The original proof of the next theorem in [LYC16] used the Weierstrass substitution, which made it difficult to understand, and made it hard to analyze the numerical stability of the induced algorithm. Also the theorem was stated in a slightly less general form requiring $\Re[\tilde{P}](1) = 1$. We roughly follow the approach of [LYC16], but improve all of the mentioned aspects of the theorem and its proof, while making the statement and the proof conceptually simpler.

2.2.3. Theorem. *Let $k \in \mathbb{N}$ be fixed and let $\tilde{P}, \tilde{Q} \in \mathbb{R}[x]$. There exists some $P, Q \in \mathbb{C}[x]$ satisfying properties (i)-(iii) of Theorem 2.2.1 such that $\tilde{P} = \Re[P]$ and $\tilde{Q} = \Re[Q]$, if and only if $\tilde{P}, \tilde{Q}$ satisfy properties (i)-(ii) of Theorem 2.2.1 and*

(vi) $\forall x \in [-1,1]\colon \tilde{P}(x)^2 + (1-x^2)\tilde{Q}(x)^2 \leq 1.$

(NB the same holds if we replace $\Re[P]$ by $\Im[P]$ and/or $\Re[Q]$ by $\Im[Q]$ in the statement. Also one might set $\tilde{Q}=0$ or $\tilde{P}=0$ if one is only interested in $\tilde{P}$ or $\tilde{Q}$.)

Proof:
"$\Longrightarrow$": Trivial.
"$\Longleftarrow$": Apply Lemma 2.2.4 to the polynomial $1 - \tilde{P}(x)^2 - (1-x^2)\tilde{Q}(x)^2$, and set $P := \tilde{P} + iB$, $Q := \tilde{Q} + iC$. □

2.2.4. Lemma. *Suppose that $A \in \mathbb{R}[x]$ is an even polynomial such that* $\deg(A) \leq 2k$ *and for all $x \in [-1,1]$ we have $A(x) \geq 0$. Then there exist polynomials $B, C \in \mathbb{R}[x]$ such that $A(x) = B^2(x) + (1-x^2)C^2(x)$, $\deg(B) \leq k$, $\deg(C) \leq k-1$, B has parity-(k mod 2) and C has parity-($k-1$ mod 2).*

Proof:
If $A = 0$ the statement is trivial, so we assume in the rest that $A \neq 0$. Let S be the multiset of roots, containing the roots of A with their algebraic multiplicity. Note that if $s \in S$ then also $-s \in S$ and $s^* \in S$ since A is an even real polynomial. (This statement holds considering multiplicities.) Let us introduce the following subsets of S (these are again multisets):

$$\begin{aligned}
S_0 &:= \{s \in S\colon s = 0\}\\
S_{(0,1)} &:= \{s \in S\colon s \in (0,1)\}\\
S_{[1,\infty)} &:= \{s \in S\colon s \in [1,\infty)\}\\
S_I &:= \{s \in S\colon \mathrm{Re}(s) = 0 \,\&\, \mathrm{Im}(s) > 0\}\\
S_C &:= \{s \in S\colon \mathrm{Re}(s) > 0 \,\&\, \mathrm{Im}(s) > 0\}.
\end{aligned}$$

Using the roots in S and some scaling factor $K \in \mathbb{R}_+$ we can write

$$A(x) = K^2 x^{|S_0|} \prod_{s \in S_{(0,1)}} (x^2 - s^2) \prod_{s \in S_{[1,\infty)}} (s^2 - x^2) \prod_{s \in S_I} (x^2 + |s|^2) \cdot$$
$$\cdot \prod_{(a+bi) \in S_C} \left(x^4 + 2x^2(b^2 - a^2) + (a^2 + b^2)^2\right). \qquad (2.10)$$

Consider the following rearrangement of the above terms corresponding to the roots in $S_{[1,\infty)}, S_I, S_C$:

$$s^2 - x^2 = (s^2-1)x^2 + s^2(1-x^2) = \underbrace{\left(\sqrt{(s^2-1)}x + is\sqrt{1-x^2}\right)}_{R_{(s)}(x):=} R^*_{(s)}(x) \tag{2.11}$$

$$x^2 + |s|^2 = (|s|^2+1)x^2 + |s|^2(1-x^2) = \underbrace{\left(\sqrt{(|s|^2+1)}x + i|s|\sqrt{1-x^2}\right)}_{P_{(s)}(x):=} P^*_{(s)}(x) \tag{2.12}$$

$$x^4 + 2x^2(b^2 - a^2) + (a^2+b^2)^2 = \underbrace{\left(\left(cx^2 - (a^2+b^2)\right) + i\sqrt{c^2-1}x\sqrt{1-x^2}\right)}_{Q_{(a,b)}(x):=} Q^*_{(a,b)}(x), \tag{2.13}$$

$$\text{where}^3\ c = a^2 + b^2 + \sqrt{2(a^2+1)b^2 + (a^2-1)^2 + b^4}.$$

Let us define

$$W(x) := Kx^{|S_0|/2} \prod_{s\in S_{(0,1)}} \sqrt{(x^2-s^2)} \prod_{s\in S_{[1,\infty)}} R_s(x) \prod_{s\in S_I} P_s(x) \prod_{(a+bi)\in S_C} Q_{(a,b)}(x).$$

Note that the factor $x^{|S_0|/2}\prod_{s\in S_{(0,1)}}\sqrt{(x^2-s^2)}$ is a polynomial, since every root in S_0 and $S_{(0,1)}$ has even multiplicity as $A(x)\geq 0$ for all $x\in(-1,1)$. Also note that $W(x)$ is a product of expressions of the form $B'(x) + i\sqrt{1-x^2}C'(x)$ where $B', C' \in \mathbb{R}[x]$ are polynomials having opposite parities (NB the zero polynomial is both even and odd, thus it has opposite parity to any even/odd polynomial). Since the product of expressions of such form can again be written in such a form, we have that $W(x) = B(x) + i\sqrt{1-x^2}C(x)$ for some $B, C \in \mathbb{R}[x]$ having opposite parities. Also note that $\deg(B) \leq |S|/2$ and $\deg(C) \leq |S|/2 - 1$.

Finally observe that by (2.10)-(2.13) we have that $A(x) = W(x)\cdot W^*(x)$, thus $A(x) = B(x)^2 + (1-x^2)C(x)^2$. Since $\deg(B) \leq |S|/2 \leq k$ and $\deg(C) \leq |S|/2 - 1 \leq k-1$, in case $\deg(A) = 2k$, we must have that B has parity-(k mod 2) and C has parity-($k-1$ mod 2). If $\deg(A) \leq 2k-2$ and B has parity-($k-1$ mod 2), then consider $\tilde{W}(x) := W(x)\cdot\left(x + i\sqrt{1-x^2}\right)$. Since $\left(x+i\sqrt{1-x^2}\right)\left(x+i\sqrt{1-x^2}\right)^* = 1$ we still have that $\tilde{W}(x)\tilde{W}^*(x) = A(x)$. Now let us denote $\tilde{W}(x) = \tilde{B}(x) + i\sqrt{1-x^2}\tilde{C}(x)$, then we get that $A(x) = \tilde{B}(x)^2 + (1-x^2)\tilde{C}(x)^2$, moreover $\deg(\tilde{B}) \leq k$, $\deg(\tilde{C}) \leq k-1$, $\tilde{B}$ has parity-(k mod 2) and $\tilde{C}$ has parity-($k-1$ mod 2). □

[3]Observe that $c \geq 1$ for all $a, b \geq 0$ and thus $\sqrt{c^2-1}\in\mathbb{R}$.

Note that the proofs of Theorems 2.2.1-2.2.3 are constructive, therefore they also give algorithms to find P, Q and Φ. The most difficult step in the proofs is to find the roots of a given degree-d univariate complex polynomial. This problem is fortunately well studied, and can be solved up to ε precision on a classical computer in time $\mathcal{O}(\mathrm{poly}(d, \log(1/\varepsilon)))$, when the roots have bounded magnitude. A rigorous analysis of numerical errors and an optimized algorithm for finding an angle sequence corresponding to a given polynomial was recently developed by Haah [Haa18]. This optimized algorithm runs in time[4] $\widetilde{\mathcal{O}}(d^3\mathrm{polylog}(1/\varepsilon))$ for a degree-d polynomial and returns the angle sequence for a uniform ε-approximating polynomial over the interval $[-1, 1]$.

Now we prove a corollary of the above results where we replace the $W(x)$ rotation operators with the following $R(x)$ reflection gates, better fitting our singular value formalism. We focus on the P polynomial because later the Q part will be projected out in the projected unitary encodings that we use.

2.2.5. Definition (Parametrized family of single-qubit reflections). We define a parametrized family of single-qubit reflection operators for all $x \in [-1, 1]$ as

$$R(x) := \begin{bmatrix} x & \sqrt{1-x^2} \\ \sqrt{1-x^2} & -x \end{bmatrix}. \tag{2.14}$$

2.2.6. Corollary (Quantum signal processing using reflections).
Let $P \in \mathbb{C}[x]$ be a degree-d polynomial, such that

- *P has parity-$(d \bmod 2)$,*
- *$\forall x \in [-1, 1]\colon |P(x)| \leq 1$,*
- *$\forall x \in (-\infty, -1] \cup [1, \infty)\colon |P(x)| \geq 1$,*
- *if d is even, then $\forall x \in \mathbb{R}\colon P(ix)P^*(ix) \geq 1$.*

Then there exists $\Phi \in \mathbb{R}^d$ such that

$$\prod_{j=1}^{d}\left(e^{i\phi_j\sigma_z}R(x)\right) = \begin{bmatrix} P(x) & . \\ . & . \end{bmatrix}. \tag{2.15}$$

Moreover for $x \in \{-1, 1\}$ we have that $P(x) = x^d \prod_{j=1}^{d} e^{i\phi_j}$, and for d even $P(0) = e^{-i\sum_{j=1}^{d}(-1)^j\phi_j}$.

[4]This result also accounts for the complexity of implementing finite-precision arithmetics, unlike our previously stated $\mathcal{O}(d^2)$ arithmetic complexity statement following Theorem 2.2.1.

Proof:
By Theorem 2.2.2 there exists $\Phi' \in \mathbb{R}^{d+1}$ for some $d \geq 1$ such that

$$e^{i\phi'_0\sigma_z}\left(\prod_{j=1}^{d} W(x)e^{i\phi'_j\sigma_z}\right) = \begin{bmatrix} P(x) & \cdot \\ \cdot & \cdot \end{bmatrix}. \tag{2.16}$$

Observe that

$$W(x) = ie^{-i\frac{\pi}{4}\sigma_z}R(x)e^{i\frac{\pi}{4}\sigma_z}, \tag{2.17}$$

thus the left-hand side of (2.15) equals

$$e^{i\phi'_0\sigma_z}\left(\prod_{j=2}^{d} e^{i\phi_j\sigma_z}ie^{-i\frac{\pi}{4}\sigma_z}R(x)e^{i\frac{\pi}{4}\sigma_z}\right)=i^d e^{i(\phi'_0-\frac{\pi}{4})\sigma_z}R(x)\left(\prod_{j=2}^{d} e^{i(\phi'_{j-1}-\frac{\pi}{2})\sigma_z}R(x)\right)e^{i(\phi'_d-\frac{\pi}{4})}.$$

Therefore

$$e^{i(\phi'_0+\phi'_d+(d-1)\frac{\pi}{2})}R(x)\left(\prod_{j=2}^{d} e^{i(\phi'_{j-1}-\frac{\pi}{2})\sigma_z}R(x)\right) = \begin{bmatrix} P(x) & \cdot \\ \cdot & \cdot \end{bmatrix}.$$

So choosing $\phi_1 := \phi'_0 + \phi'_d + (d-1)\frac{\pi}{2}$ and for all $j \in \{2, 3, \ldots, d\}$ setting $\phi_j := \phi'_{j-1} - \frac{\pi}{2}$, results in a $\Phi \in \mathbb{R}^d$ that clearly satisfies (2.15). The additional result for $x \in \{-1, 1\}$ follows from the fact that for $x \in \{-1, 1\}$ every matrix in (2.15) becomes diagonal.

The claim about $P(0)$ follows from the observation that

$$e^{i\phi_1\sigma_z}R(0)e^{i\phi_2\sigma_z}R(0) = e^{i(\phi_1-\phi_2)\sigma_z}. \qquad \square$$

One could directly show that Chebyshev polynomials satisfy the requirements of Corollary 2.2.6, but instead we explicitly describe the corresponding Φ; by Theorem 2.2.2 it actually proves that the conditions of Corollary 2.2.6 hold.

2.2.7. LEMMA (Chebyshev polynomials in quantum signal processing).
Let $T_d \in \mathbb{R}[x]$ be the d-th Chebyshev polynomial of the first kind. Let $\Phi \in \mathbb{R}^d$ be such that $\phi_1 = (1-d)\frac{\pi}{2}$, and for all $i \in [d] \setminus \{1\}$ let $\phi_i := \frac{\pi}{2}$. Using this Φ in equation (2.15) we get that $P = T_d$.

Proof:
One can prove this, e.g., by induction using the substitution $x := \cos(\theta)$. $\square$

In Corollary 2.2.6 the requirements on P are not very intuitive, but fortunately we have a good understanding of the polynomials that can emerge by taking the real part of these polynomials. This results in the following useful corollary:

2.2.8. COROLLARY (Real quantum signal processing).
Let $P_{\Re}(x) \in \mathbb{R}[x]$ be a degree-d polynomial for some $d \geq 1$, such that

- *$P_{\Re}$ has parity-(d mod 2), and*
- *for all $x \in [-1,1]$: $|P_{\Re}(x)| \leq 1$.*

Then there is a $P \in \mathbb{C}[x]$ of degree d satisfying the requirements of Corollary 2.2.6.

Moreover, given $P_{\Re}(x)$ and $\delta \geq 0$ we can find a P and a corresponding Φ, such that $|\Re[P] - P_{\Re}| \leq \delta$ for all $x \in [-1,1]$, using a classical computer in time $\mathcal{O}(\mathrm{poly}(d, \log(1/\delta)))$.

Proof:
The existence of such P follows directly from Theorem 2.2.1-2.2.3.

The complexity statement follows from the fact that we can find P and Φ' using the procedures of Theorems 2.2.1-2.2.3 on a classical computer in time $\mathcal{O}(\mathrm{poly}(d, \log(1/\varepsilon)))$ as noted above. Computing Φ from Φ' as in the proof of Corollary 2.2.6 only yields a small overhead. □

2.3 Quantum singular value transformation

In this section we show how to leverage the results of the previous section to perform quantum singular value transformation of projected unitary matrices, with ideas coming from qubitization [LC16]. Qubitization was introduced by Low and Chuang [LC16] in order to apply polynomial transformations to the spectrum of a Hermitian (or normal) operator, which is represented as the top-left block of a unitary matrix. Low and Chuang [LC17a] also showed how to use their techniques in order to develop advanced amplitude transformation results. We generalize their results, and develop the technique of singular value transformation, which applies to any operator as opposed to only normal operators.

It turns out that by applying a unitary U back and forth interleaved with some simple phase operators one can induce polynomial transformations to the singular values of a particular (not necessarily rectangular) block-matrix of the unitary U. The main idea behind our approach is to lift the quantum signal processing results presented in the previous section by studying two-dimensional invariant subspaces coming from Camille Jordan's Lemma [Jor75]. Then by understanding how the two-dimensional subspaces behave, one can infer the higher-dimensional behavior.

The original qubitization approach can be understood along the lines of Jordan's Lemma [Jor75] about the common invariant subspaces of two reflections. Jordan's result is most often presented stating that the product of two reflections decomposes into one- and two-dimensional invariant subspaces, such that the operator has eigenvalue ± 1 on the one-dimensional subspaces, and the operator acts as a rotation on the two-dimensional subspaces. This higher-dimensional insight lies at the heart of Szegedy's quantum walk results [Szeg04] as well as Marriott and Watrous's QMA amplification scheme [MW05].

Motivated by a series of prior work on quantum search algorithms [Gro05, Hø00, YLC14], Low and Chuang [LC16] replaced one of the reflections in Jordan's Lemma by a phase-gate, such as in Figure 2.2b. They examined the operators arising by iterative application of the reflection- and phase-operator with applying possibly different phases in each step. In this chapter we go one step further and replace the other reflection[5] by an arbitrary unitary operator U, and analyze the procedure with carefully chosen one- and two-dimensional subspaces coming from singular value decomposition similar to Jordan's Lemma.

2.3.1. Definition (Singular value decomposition of a projected unitary).
Let $\mathcal{H}_U$ be a finite-dimensional Hilbert space and let $U, \Pi, \widetilde{\Pi} \in \mathrm{End}(\mathcal{H}_U)$ be linear operators on $\mathcal{H}_U$ such that U is a unitary, and $\Pi, \widetilde{\Pi}$ are orthogonal projectors, and let $A = \widetilde{\Pi} U \Pi$. Let $d := \mathrm{rank}(\Pi)$, $\tilde{d} := \mathrm{rank}(\widetilde{\Pi})$, $d_{\min} := \min(d, \tilde{d})$. By the singular value decomposition of A we know that there exist orthonormal bases $(|\psi_i\rangle \colon i \in [d])$, $(|\tilde{\psi}_i\rangle \colon i \in [\tilde{d}])$ of the subspaces $\mathrm{img}(\Pi)$ and $\mathrm{img}(\widetilde{\Pi})$ respectively, such that

$$A = \sum_{i=1}^{d_{\min}} \varsigma_i |\tilde{\psi}_i\rangle\langle\psi_i|, \tag{2.18}$$

and[6] for all $i \in [d_{\min}] \colon \varsigma_i \in \mathbb{R}_0^+$. Moreover, $\varsigma_i \geq \varsigma_j$ for all $i \leq j \in [d_{\min}]$.

2.3.2. Definition (Invariant subspaces associated to an SVD).
Assume the notation of Definition 2.3.1. Let $r = \mathrm{rank}(A)$; if $\varsigma_1 < 1$, then let $k = 0$, else let $k \in [d_{\min}]$ be the largest index for which $\varsigma_k = 1$. We define

$$|\psi_i^\perp\rangle := \frac{(I-\Pi)U^\dagger|\tilde{\psi}_i\rangle}{\left\|(I-\Pi)U^\dagger|\tilde{\psi}_i\rangle\right\|} = \frac{(I-\Pi)U^\dagger|\tilde{\psi}_i\rangle}{\sqrt{1-\varsigma_i^2}}, \text{ and}$$

$$|\tilde{\psi}_i^\perp\rangle := \frac{(I-\widetilde{\Pi})U|\psi_i\rangle}{\left\|(I-\widetilde{\Pi})U|\psi_i\rangle\right\|} = \frac{(I-\widetilde{\Pi})U|\psi_i\rangle}{\sqrt{1-\varsigma_i^2}}.$$

$$\begin{array}{llll}
\text{For} \;\; i \in [k] & \text{let} \;\; \mathcal{H}_i := \mathrm{Span}(|\psi_i\rangle) & \text{and} & \tilde{\mathcal{H}}_i := \mathrm{Span}\Big(|\tilde{\psi}_i\rangle\Big), \\
i \in [r] \setminus [k] & \text{let} \;\; \mathcal{H}_i := \mathrm{Span}\big(|\psi_i\rangle, |\psi_i^\perp\rangle\big) & \text{and} & \tilde{\mathcal{H}}_i := \mathrm{Span}\Big(|\tilde{\psi}_i\rangle, |\tilde{\psi}_i^\perp\rangle\Big) \\
i \in [d] \setminus [r] & \text{let} \;\; \mathcal{H}_i^R := \mathrm{Span}(|\psi_i\rangle) & \text{and} & \tilde{\mathcal{H}}_i^R := \mathrm{Span}(U|\psi_i\rangle), \\
i \in [\tilde{d}] \setminus [r] & \text{let} \;\; \mathcal{H}_i^L := \mathrm{Span}\Big(U^\dagger|\tilde{\psi}_i\rangle\Big) & \text{and} & \tilde{\mathcal{H}}_i^L := \mathrm{Span}\Big(|\tilde{\psi}_i\rangle\Big).
\end{array}$$

[5]One could in principle merge U and $U^\dagger$ into one of the projectors, leading to a product of reflections as in Jordan's Lemma, however we take a slightly different perspective.

[6]In singular value decomposition one usually requires that the diagonal elements of Σ are non-negative. Here we could also allow negative reals, all the proofs of this section would go through with minor modifications, e.g., defining the ordering of the singular values with decreasing absolute value. This would enable one to treat spectral decompositions of Hermitian matrices also as singular value decompositions.

Finally let $\mathcal{H}_\perp := \left(\bigoplus_{i\in[r]}\mathcal{H}_i \oplus \bigoplus_{i\in[d]\setminus[r]}\mathcal{H}_i^R \oplus \bigoplus_{i\in[\tilde{d}]\setminus[r]}\mathcal{H}_i^L\right)^\perp$, and define $\widetilde{\mathcal{H}}_\perp$ the same way, just putting a $\tilde{}$ on top of every $\mathcal{H}$ symbol.

Now we show that the subspaces $\mathcal{H}_i\colon i\in[k]$, $\mathcal{H}_i\colon i\in[r]\setminus[k]$, $\mathcal{H}_i^R\colon i\in[d]\setminus[r]$ and $\mathcal{H}_i^L\colon i\in[\tilde{d}]\setminus[r]$, are indeed pairwise orthogonal, by proving that their spanning bases described in Definition 2.3.2 form an orthonormal system of vectors. (By symmetry it also implies that the spanning bases of the $\widetilde{\mathcal{H}}$ subspaces also form an orthonormal system of vectors.) The proof is summarized in Table 2.1, relying on the following observations:

$$\forall i,j\in[d] \qquad \langle\psi_i|\psi_j\rangle=\delta_{ij} \tag{2.19}$$

$$\forall i\in[d], j\in[r]\setminus[k] \quad \langle\psi_i|\psi_j^\perp\rangle \propto \langle\psi_i|(I-\Pi)U^\dagger|\tilde{\psi}_j\rangle \propto \langle\psi_i|(I-\Pi)=0 \tag{2.20}$$

$$\forall i\in[d], j\in[\tilde{d}]\setminus[r] \quad \langle\psi_i|U^\dagger|\tilde{\psi}_j\rangle=\langle\psi_i|\Pi U^\dagger\widetilde{\Pi}|\tilde{\psi}_j\rangle=\langle\psi_i|A^\dagger|\tilde{\psi}_j\rangle\propto A^\dagger|\tilde{\psi}_j\rangle=0 \tag{2.21}$$

$$\forall i,j\in[r]\setminus[k] \qquad \langle\psi_i^\perp|\psi_j^\perp\rangle=\frac{\langle\tilde{\psi}_i|U(I-\Pi)U^\dagger|\tilde{\psi}_j\rangle}{\sqrt{(1-\varsigma_i^2)(1-\varsigma_j^2)}}=\frac{\delta_{ij}-\langle\tilde{\psi}_i|AA^\dagger|\tilde{\psi}_j\rangle}{\sqrt{(1-\varsigma_i^2)(1-\varsigma_j^2)}}=\delta_{ij} \tag{2.22}$$

$\forall i\in[r]\setminus[k], j\in[\tilde{d}]\setminus[r]$

$$\langle\psi_i^\perp|U^\dagger|\tilde{\psi}_j\rangle=\frac{\langle\tilde{\psi}_i|U(I-\Pi)U^\dagger|\tilde{\psi}_j\rangle}{\sqrt{(1-\varsigma_i^2)}}=\frac{\delta_{ij}-\langle\tilde{\psi}_i|AA^\dagger|\tilde{\psi}_j\rangle}{\sqrt{(1-\varsigma_i^2)}}=0 \tag{2.23}$$

$$\forall i,j\in[\tilde{d}] \qquad \langle\tilde{\psi}_i|\tilde{\psi}_j\rangle=\delta_{ij} \tag{2.24}$$

$\mathcal{H}_i\perp\mathcal{H}_j$	$\lvert\psi_j\rangle\in\mathcal{H}_j$ $j\in[k]$	$\lvert\psi_j\rangle\in\mathcal{H}_j$ $j\in[r]\setminus[k]$	$\lvert\psi_j^\perp\rangle\in\mathcal{H}_j$ $j\in[r]\setminus[k]$	$\lvert\psi_j\rangle\in\mathcal{H}_j^R$ $j\in[d]\setminus[r]$	$U^\dagger\lvert\tilde{\psi}_j\rangle\in\mathcal{H}_j^L$ $j\in[\tilde{d}]\setminus[r]$
$\lvert\psi_i\rangle\in\mathcal{H}_i$ $i\in[k]$	by (2.19) $\langle\psi_i\lvert\psi_j\rangle=\delta_{ij}$	by (2.19) $\langle\psi_i\lvert\psi_j\rangle=0$	by (2.20) $\langle\psi_i\lvert\psi_j^\perp\rangle=0$	by (2.19) $\langle\psi_i\lvert\psi_j\rangle=0$	by (2.21) $\langle\psi_i\lvert U^\dagger\lvert\tilde{\psi}_j\rangle=0$
$\lvert\psi_i\rangle\in\mathcal{H}_i$ $i\in[r]\setminus[k]$		by (2.19) $\langle\psi_i\lvert\psi_j\rangle=\delta_{ij}$	by (2.20) $\langle\psi_i\lvert\psi_j^\perp\rangle=0$	by (2.19) $\langle\psi_i\lvert\psi_j\rangle=0$	by (2.21) $\langle\psi_i\lvert U^\dagger\lvert\tilde{\psi}_j\rangle=0$
$\lvert\psi_i^\perp\rangle\in\mathcal{H}_i$ $i\in[r]\setminus[k]$			by (2.22) $\langle\psi_i^\perp\lvert\psi_j^\perp\rangle=\delta_{ij}$	by (2.20) $\langle\psi_i^\perp\lvert\psi_j\rangle=0$	by (2.23) $\langle\psi_i^\perp\lvert U^\dagger\lvert\tilde{\psi}_j\rangle=0$
$\lvert\psi_i\rangle\in\mathcal{H}_i^R$ $i\in[d]\setminus[r]$				by (2.19) $\langle\psi_i\lvert\psi_j\rangle=\delta_{ij}$	by (2.21) $\langle\psi_i\lvert U^\dagger\lvert\tilde{\psi}_j\rangle=0$
$U^\dagger\lvert\tilde{\psi}_i\rangle\in\mathcal{H}_i^L$ $i\in[\tilde{d}]\setminus[r]$					by (2.24) $\langle\tilde{\psi}_i\lvert UU^\dagger\lvert\tilde{\psi}_j\rangle=\delta_{ij}$

Table 2.1. Orthonormality of the spanning bases described in Definition 2.3.2.

Now we introduce some notation for matrices that represent linear maps acting between different subspaces. This will enable us to conveniently express matrices in a block-diagonal form. We will use the subspaces of Definition 2.3.2, because they enable us to block-diagonalize the unitaries used for implementing singular value transformation.

2.3.3. Definition (Notation for linear maps between different vector spaces). For two vector (sub)spaces $\mathcal{H},\mathcal{H}'$ let us denote by $[\,\cdot\,]^{\mathcal{H}}_{\mathcal{H}'}$ the matrix of a linear map that maps $\mathcal{H}\mapsto\mathcal{H}'$. Moreover, if the subspaces are as in Definition 2.3.2 and we explicitly write down matrix elements, they are meant to be interpreted in the spanning bases we used for defining $\mathcal{H},\mathcal{H}'$ in Definition 2.3.2.

2.3.4. Lemma (Invariant subspace decomposition of a projected unitary). *Let $\mathcal{H}_U$ be a finite-dimensional Hilbert-space and $U,\Pi,\widetilde{\Pi}\in\mathrm{End}(\mathcal{H}_U)$ be as in Definition 2.3.1. Then using the singular value decomposition of Definition 2.3.2 we have that U equals*

$$\bigoplus_{i\in[k]}[1]^{\mathcal{H}_i}_{\tilde{\mathcal{H}}_i}\oplus\bigoplus_{i\in[r]\setminus[k]}\begin{bmatrix}\varsigma_i & \sqrt{1-\varsigma_i^2}\\ \sqrt{1-\varsigma_i^2} & -\varsigma_i\end{bmatrix}^{\mathcal{H}_i}_{\tilde{\mathcal{H}}_i}\oplus\bigoplus_{i\in[d]\setminus[r]}[1]^{\mathcal{H}_i^R}_{\tilde{\mathcal{H}}_i^R}\oplus\bigoplus_{i\in[\tilde{d}]\setminus[r]}[1]^{\mathcal{H}_i^L}_{\tilde{\mathcal{H}}_i^L}\oplus[\,\cdot\,]^{\mathcal{H}_\perp}_{\tilde{\mathcal{H}}_\perp}. \tag{2.25}$$

Moreover, $2\Pi-I$ and $e^{i\phi(2\Pi-I)}$ respectively can be written as

$$\bigoplus_{i\in[k]}[1]^{\mathcal{H}_i}_{\mathcal{H}_i}\oplus\bigoplus_{i\in[r]\setminus[k]}\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}^{\mathcal{H}_i}_{\mathcal{H}_i}\oplus\bigoplus_{i\in[d]\setminus[r]}[1]^{\mathcal{H}_i^R}_{\mathcal{H}_i^R}\oplus\bigoplus_{i\in[d]\setminus[r]}[-1]^{\mathcal{H}_i^L}_{\mathcal{H}_i^L}\oplus[\,\cdot\,]^{\mathcal{H}_\perp}_{\mathcal{H}_\perp}, \tag{2.26}$$

$$\bigoplus_{i\in[k]}[e^{i\phi}]^{\mathcal{H}_i}_{\mathcal{H}_i}\oplus\bigoplus_{i\in[r]\setminus[k]}\begin{bmatrix}e^{i\phi} & 0\\ 0 & e^{-i\phi}\end{bmatrix}^{\mathcal{H}_i}_{\mathcal{H}_i}\oplus\bigoplus_{i\in[d]\setminus[r]}[e^{i\phi}]^{\mathcal{H}_i^R}_{\mathcal{H}_i^R}\oplus\bigoplus_{i\in[d]\setminus[r]}[e^{-i\phi}]^{\mathcal{H}_i^L}_{\mathcal{H}_i^L}\oplus[\,\cdot\,]^{\mathcal{H}_\perp}_{\mathcal{H}_\perp}, \tag{2.27}$$

similarly $2\widetilde{\Pi}-I$ and $e^{i\phi(2\widetilde{\Pi}-I)}$ respectively can be written as

$$\bigoplus_{i\in[k]}[1]^{\tilde{\mathcal{H}}_i}_{\tilde{\mathcal{H}}_i}\oplus\bigoplus_{i\in[r]\setminus[k]}\begin{bmatrix}1 & 0\\ 0 & -1\end{bmatrix}^{\tilde{\mathcal{H}}_i}_{\tilde{\mathcal{H}}_i}\oplus\bigoplus_{i\in[d]\setminus[r]}[-1]^{\tilde{\mathcal{H}}_i^R}_{\tilde{\mathcal{H}}_i^R}\oplus\bigoplus_{i\in[d]\setminus[r]}[1]^{\tilde{\mathcal{H}}_i^L}_{\tilde{\mathcal{H}}_i^L}\oplus[\,\cdot\,]^{\tilde{\mathcal{H}}_\perp}_{\tilde{\mathcal{H}}_\perp}, \tag{2.28}$$

$$\bigoplus_{i\in[k]}[e^{i\phi}]^{\tilde{\mathcal{H}}_i}_{\tilde{\mathcal{H}}_i}\oplus\bigoplus_{i\in[r]\setminus[k]}\begin{bmatrix}e^{i\phi} & 0\\ 0 & e^{-i\phi}\end{bmatrix}^{\tilde{\mathcal{H}}_i}_{\tilde{\mathcal{H}}_i}\oplus\bigoplus_{i\in[d]\setminus[r]}[e^{-i\phi}]^{\tilde{\mathcal{H}}_i^R}_{\tilde{\mathcal{H}}_i^R}\oplus\bigoplus_{i\in[d]\setminus[r]}[e^{i\phi}]^{\tilde{\mathcal{H}}_i^L}_{\tilde{\mathcal{H}}_i^L}\oplus[\,\cdot\,]^{\tilde{\mathcal{H}}_\perp}_{\tilde{\mathcal{H}}_\perp}. \tag{2.29}$$

Proof:
For all $i\in[r]\setminus[k]$ we can verify that

$$U|\psi_i\rangle=\widetilde{\Pi}U|\psi_i\rangle+(I-\widetilde{\Pi})U|\psi_i\rangle=\underbrace{\widetilde{\Pi}U\Pi}_{A}|\psi_i\rangle+(I-\widetilde{\Pi})U|\psi_i\rangle=\varsigma_i|\tilde{\psi}_i\rangle+\sqrt{1-\varsigma_i^2}|\tilde{\psi}_i^\perp\rangle, \tag{2.30}$$

and

$$\sqrt{1-\varsigma_i^2}U|\psi_i^\perp\rangle = U(I-\Pi)U^\dagger|\tilde{\psi}_i\rangle = |\tilde{\psi}_i\rangle - U\Pi U^\dagger|\tilde{\psi}_i\rangle = |\tilde{\psi}_i\rangle - U\underbrace{\Pi U^\dagger\widetilde{\Pi}}_{A^\dagger}|\tilde{\psi}_i\rangle$$

$$= |\tilde{\psi}_i\rangle - U\varsigma_i|\psi_i\rangle = (1-\varsigma_i^2)|\tilde{\psi}_i\rangle - \varsigma_i\sqrt{1-\varsigma_i^2}|\tilde{\psi}_i^\perp\rangle, \qquad (2.31)$$

where in the last equality we used (2.30). Since U is unitary, it preserves the inner product and therefore maps $\mathcal{H}_\perp$ onto $\widetilde{\mathcal{H}}_\perp$. Now equation (2.25) directly follows from (2.30)-(2.31). The other statements trivially follow from Definition 2.3.1. □

2.3.5. Definition (Alternating phase modulation sequences).
Let $\mathcal{H}_U$ be a finite-dimensional Hilbert space and let $U, \Pi, \widetilde{\Pi} \in \mathrm{End}(\mathcal{H}_U)$ be linear operators on $\mathcal{H}_U$ such that U is a unitary, and $\Pi, \widetilde{\Pi}$ are orthogonal projectors. Let $\Phi \in \mathbb{R}^n$; we define the *alternating phase modulation sequences* U_Φ as follows

$$U_\Phi := \begin{cases} e^{i\phi_1(2\widetilde{\Pi}-I)}U\prod_{j=1}^{(n-1)/2}\left(e^{i\phi_{2j}(2\Pi-I)}U^\dagger e^{i\phi_{2j+1}(2\widetilde{\Pi}-I)}U\right) & \text{if } n \text{ is odd, and} \\ \prod_{j=1}^{n/2}\left(e^{i\phi_{2j-1}(2\Pi-I)}U^\dagger e^{i\phi_{2j}(2\widetilde{\Pi}-I)}U\right) & \text{if } n \text{ is even.} \end{cases} \qquad (2.32)$$

2.3.6. Definition (Singular value transformation by even/odd functions).
Let $f:\mathbb{R}\to\mathbb{C}$ be an even or odd function. Let $A \in \mathbb{C}^{\tilde{d}\times d}$, let $d_{\min} := \min(d,\tilde{d})$ and let

$$A = \sum_{i=1}^{d_{\min}} \varsigma_i|\tilde{\psi}_i\rangle\langle\psi_i|$$

be a singular value decomposition of A.

We define the *singular value transform* of A, for an odd function f as

$$f^{(SV)}(A) := \sum_{i=1}^{d_{\min}} f(\varsigma_i)|\tilde{\psi}_i\rangle\langle\psi_i|,$$

and for an even f as

$$f^{(SV)}(A) := \sum_{i=1}^{d} f(\varsigma_i)|\psi_i\rangle\langle\psi_i|,$$

where for $i \in [d]\setminus[d_{\min}]$ we define $\varsigma_i := 0$.

The following theorem is a generalized and improved version of the "Flexible quantum signal processing" result of Low and Chuang [LC17a, Theorem 4]. Our result is more general because it works for arbitrary matrices as opposed to only Hermitian (or normal) matrices. Another improvement is that we remove the constraint $P_\Re(0) = 0$ for even d, thanks to our improved treatment of Theorem 2.2.3 and Corollary 2.2.6. Also we note that the following theorem can be viewed as a generalization of the quantum walk techniques introduced by Szegedy [Szeg04].

2.3.7. Theorem (QSVT by alternating phase modulation).
Let $\mathcal{H}_U$ be a finite-dimensional Hilbert space and let $U, \Pi, \widetilde{\Pi} \in \mathrm{End}(\mathcal{H}_U)$ be linear operators on $\mathcal{H}_U$ such that U is a unitary, and $\Pi, \widetilde{\Pi}$ are orthogonal projectors. Let $P \in \mathbb{C}[x]$ and $\Phi \in \mathbb{R}^n$ be as in Corollary 2.2.6. Then

$$P^{(SV)}(\widetilde{\Pi} U \Pi) = \begin{cases} \widetilde{\Pi} U_\Phi \Pi & \text{if } n \text{ is odd, and} \\ \Pi U_\Phi \Pi & \text{if } n \text{ is even.} \end{cases} \tag{2.33}$$

Proof:
We first prove the odd case. Observe that $P(1) = \prod_{j=1}^n e^{i\phi_j}$, and let $e^{i\phi_0} := e^{i\sum_{j=1}^n (-1)^n \phi_j}$, then $U_\Phi = e^{i\phi_1(2\widetilde{\Pi}-I)} U \prod_{j=1}^{n/2}\Big(e^{i\phi_{2j}(2\Pi-I)} U^\dagger e^{i\phi_{2j+1}(2\widetilde{\Pi}-I)} U\Big)$, which further equals

$$\bigoplus_{i\in[k]} [\varsigma_k^n P(1)]_{\tilde{\mathcal{H}}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[r]\setminus[k]} \left[\prod_{j=1}^n \big(e^{i\phi_j\sigma_z} R(\varsigma_\ell)\big)\right]_{\tilde{\mathcal{H}}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[d]\setminus[r]} [e^{i\phi_0}]_{\tilde{\mathcal{H}}_i^R}^{\mathcal{H}_i^R} \oplus \bigoplus_{i\in[\tilde{d}]\setminus[r]} [e^{-i\phi_0}]_{\tilde{\mathcal{H}}_i^L}^{\mathcal{H}_i^L} \oplus [\,\cdot\,]_{\tilde{\mathcal{H}}_\perp}^{\mathcal{H}_\perp}$$

(by Lemma 2.3.4)

$$= \bigoplus_{i\in[k]} [P(\varsigma_i)]_{\tilde{\mathcal{H}}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[r]\setminus[k]} \begin{bmatrix} P(\varsigma_i) & \cdot \\ \cdot & \cdot \end{bmatrix}_{\tilde{\mathcal{H}}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[d]\setminus[r]} [e^{i\phi_0}]_{\tilde{\mathcal{H}}_i^R}^{\mathcal{H}_i^R} \oplus \bigoplus_{i\in[\tilde{d}]\setminus[r]} [e^{-i\phi_0}]_{\tilde{\mathcal{H}}_i^L}^{\mathcal{H}_i^L} \oplus [\,\cdot\,]_{\tilde{\mathcal{H}}_\perp}^{\mathcal{H}_\perp}.$$

(by Corollary 2.2.6)

Finally equation (2.33) follows from the fact that $\Pi = \sum_{i=1}^d |\psi_i\rangle\langle\psi_i|$ and $\widetilde{\Pi} = \sum_{i=1}^{\tilde{d}} |\tilde{\psi}_i\rangle\langle\tilde{\psi}_i|$, therefore

$$\begin{aligned}\widetilde{\Pi} U_\Phi \Pi &= \bigoplus_{i\in[k]} [P(\varsigma_i)]_{\tilde{\mathcal{H}}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[r]\setminus[k]} \begin{bmatrix} P(\varsigma_i) & 0 \\ 0 & 0 \end{bmatrix}_{\tilde{\mathcal{H}}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[d]\setminus[r]} [0]_{\tilde{\mathcal{H}}_i^R}^{\mathcal{H}_i^R} \oplus \bigoplus_{i\in[\tilde{d}]\setminus[r]} [0]_{\tilde{\mathcal{H}}_i^L}^{\mathcal{H}_i^L} \oplus [0]_{\tilde{\mathcal{H}}_\perp}^{\mathcal{H}_\perp} \\ &= \sum_{i=1}^{d_{\min}} P(\varsigma_i) |\tilde{\psi}_i\rangle\langle\psi_i|.\end{aligned}$$

The last equality follows from the observation that for odd n the polynomial P is also odd, therefore $P(0) = 0$.

For the even case we can similarly derive that U_Φ equals

$$\bigoplus_{i\in[k]} [P(\varsigma_i)]_{\mathcal{H}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[r]\setminus[k]} \begin{bmatrix} P(\varsigma_i) & \cdot \\ \cdot & \cdot \end{bmatrix}_{\mathcal{H}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[d]\setminus[r]} [e^{-i\phi_0}]_{\mathcal{H}_i^R}^{\mathcal{H}_i^R} \oplus \bigoplus_{i\in[\tilde{d}]\setminus[r]} [e^{i\phi_0}]_{\mathcal{H}_i^L}^{\mathcal{H}_i^L} \oplus [\,\cdot\,]_{\mathcal{H}_\perp}^{\mathcal{H}_\perp}.$$

(by Corollary 2.2.6)

Finally equation (2.33) follows from the fact that $\Pi = \sum_{i=1}^{d} |\psi_i\rangle\langle\psi_i|$, and therefore

$$\begin{aligned}\Pi U_\Phi \Pi &= \bigoplus_{i\in[k]}[P(\varsigma_i)]_{\mathcal{H}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[r]\setminus[k]} \begin{bmatrix} P(\varsigma_i) & 0 \\ 0 & 0 \end{bmatrix}_{\mathcal{H}_i}^{\mathcal{H}_i} \oplus \bigoplus_{i\in[d]\setminus[r]}[e^{-i\phi_0}]_{\mathcal{H}_i^R}^{\mathcal{H}_i^R} \oplus \bigoplus_{i\in[\tilde{d}]\setminus[r]}[0]_{\mathcal{H}_i^L}^{\mathcal{H}_i^L} \oplus [0]_{\mathcal{H}_\perp}^{\mathcal{H}_\perp}\\ &= \sum_{i=1}^{d} P(\varsigma_i)|\psi_i\rangle\langle\psi_i|.\end{aligned}$$

The last equality uses the observation that for n even $P(0) = e^{-i\phi_0}$, as shown by Corollary 2.2.6. □

2.3.8. Corollary (Singular value transformation by real polynomials).
Let $U, \Pi, \widetilde{\Pi}$ be as in Theorem 2.3.7. Suppose that $P_\Re \in \mathbb{R}[x]$ is a degree-n polynomial satisfying that

- *$P_\Re$ has parity-(n mod 2) and*
- *for all $x \in [-1,1]$: $|P_\Re(x)| \leq 1$.*

Then there exist $\Phi \in \mathbb{R}^n$, such that

$$P_\Re^{(SV)}\left(\widetilde{\Pi}U\Pi\right) = \begin{cases} \left(\langle+|\otimes\widetilde{\Pi}\right)\left(|0\rangle\langle 0|\otimes U_\Phi + |1\rangle\langle 1|\otimes U_{-\Phi}\right)\left(|+\rangle\otimes\Pi\right) & \textit{if } n \textit{ is odd,}\\ \left(\langle+|\otimes\Pi\right)\left(|0\rangle\langle 0|\otimes U_\Phi + |1\rangle\langle 1|\otimes U_{-\Phi}\right)\left(|+\rangle\otimes\Pi\right) & \textit{if } n \textit{ is even.}\end{cases} \tag{2.34}$$

Proof:
By Corollary 2.2.8 we can find a $\Phi \in \mathbb{R}^n$ such that $\Re[P] = P_\Re$. Observe that $-\Phi$ gives rise to P^* in Corollary 2.2.6 as can be seen from equation (2.15). Let $\Pi' = \widetilde{\Pi}$ for n odd and let $\Pi' = \Pi$ for n even. Then by Theorem 2.3.7 we get that $P^{(SV)}\left(\widetilde{\Pi}U\Pi\right) = \Pi' U_\Phi \Pi$, and $P^{*(SV)}\left(\widetilde{\Pi}U\Pi\right) = \Pi' U_{-\Phi}\Pi$. Therefore

$$(\langle+|\otimes\Pi')(|0\rangle\langle 0|\otimes U_\Phi)(|+\rangle\otimes\Pi) = P^{(SV)}\left(\widetilde{\Pi}U\Pi\right)/2$$
$$(\langle+|\otimes\Pi')(|1\rangle\langle 1|\otimes U_{-\Phi})(|+\rangle\otimes\Pi) = P^{*(SV)}\left(\widetilde{\Pi}U\Pi\right)/2.$$

We can conclude by observing that $P_\Re = (P + P^*)/2$, and therefore

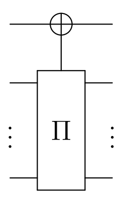
$$P_\Re^{(SV)}\left(\widetilde{\Pi}U\Pi\right) = \left(P^{(SV)}\left(\widetilde{\Pi}U\Pi\right) + P^{*(SV)}\left(\widetilde{\Pi}U\Pi\right)\right)/2.$$

□

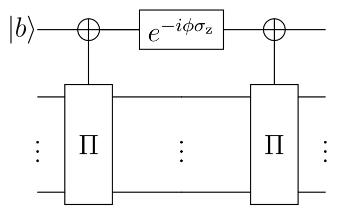
Note that the above result is essentially optimal in the sense that each requirement is necessary. It is obvious that the polynomial needs to be bounded within $[-1,1]$ since the matrix must have norm at most 1 as it is a projected unitary. Also one cannot implement a degree-d Chebyshev polynomial with $d-1$ uses of the unitary U, as shown by our lower bound in Section 3.5, because $T_d(x)$ evaluates to 1 at $x=1$ and has derivative d^2. Indeed, by setting $x:=1$ and $y:=1-\delta$ in equation (3.32) for some small enough $\delta>0$, Theorem 3.5.1 shows that precisely implementing T_d requires at least d uses of U.

Finally, regarding the parity constraint, note that every result in this subsection would stay valid if we would extend the concept of singular values by allowing negative values as well. Then, changing a singular vector/singular value term from $\varsigma|\phi\rangle\langle\psi|$ to $-\varsigma(-|\phi\rangle)\langle\psi|$ would result in a valid alternative decomposition, and singular value transformation by a polynomial P could be evaluated using both decompositions. For consistency it would require that $P(\varsigma)|\psi\rangle\langle\psi| = P(-\varsigma)|\psi\rangle\langle\psi|$, and $P(\varsigma)|\phi\rangle\langle\psi| = P(-\varsigma)(-|\phi\rangle)\langle\psi|$, showing the necessity of the even/odd constraint. Equations (2.35)-(2.36) in the proof of Corollary 2.4.2 also show that the even/odd case separation is quite natural.

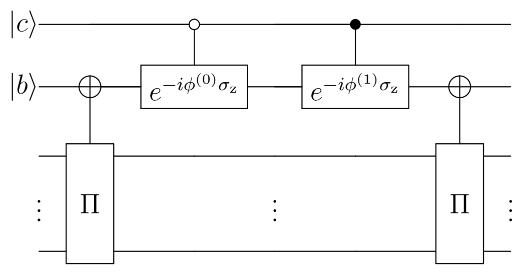
What remains is to discuss how to efficiently implement alternating phase modulation sequences. The operator $e^{i\phi(2\Pi-I)}=\mathrm{C}_\Pi\mathrm{NOT}\big(I\otimes e^{-i\phi\sigma_z}\big)\mathrm{C}_\Pi\mathrm{NOT}$ can be implemented using a single ancilla qubit, two uses of $\mathrm{C}_\Pi\mathrm{NOT}$, and a single-qubit phase gate $e^{-i\phi\sigma_z}$, leading to an efficient implementation of U_Φ, see Figure 2.2b.



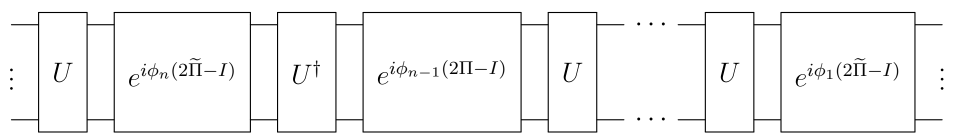
(a) $\mathrm{C}_\Pi\mathrm{NOT}$



(b) $|b\rangle\langle b|\otimes e^{(-1)^b i\phi(2\Pi-I)}$



(c) $|cb\rangle\langle cb|\otimes e^{(-1)^b i\phi^{(c)}(2\Pi-I)}$


U
$e^{i\phi_n(2\widetilde{\Pi}-I)}$
$U^\dagger$
$e^{i\phi_{n-1}(2\Pi-I)}$
U
U
$e^{i\phi_1(2\widetilde{\Pi}-I)}$

(d) $U_\Phi = e^{i\phi_1(2\widetilde{\Pi}-I)}U\prod_{j=1}^{(n-1)/2}\Big(e^{i\phi_{2j}(2\Pi-I)}U^\dagger e^{i\phi_{2j+1}(2\widetilde{\Pi}-I)}U\Big)$ (for odd n)

Figure 2.2. Gates and gate sequences used for singular value transformation in Theorem 2.3.7. Figure 2.2a shows how to implement a $\mathrm{C}_\Pi\mathrm{NOT}$ gate, and Figure 2.2b shows how to implement $e^{i\phi(2\Pi-I)}$ using a single ancilla qubit, two $\mathrm{C}_\Pi\mathrm{NOT}$ gates and an $e^{-i\phi\sigma_z}$ gate. Figure 2.2c demonstrates how to implement a controlled version of the gate $e^{i\phi^{(c)}(2\Pi-I)}$, by only controlling the single-qubit gate $e^{-i\phi^{(c)}\sigma_z}$. Finally, Figure 2.2d summarizes the complete circuit used in Theorem 2.3.7.

2.3.9. LEMMA (Implementating of alternating phase modulation sequences).
Let $\Phi \in \mathbb{R}^n$; the alternating phase modulation sequence U_Φ of Definition 2.3.5 can be implemented using a single ancilla qubit with n uses of U and $U^\dagger$, n uses of $C_\Pi NOT$ and n uses of $C_{\widetilde{\Pi}} NOT$ gates and n single-qubit gates. A controlled version of U_Φ can be built similarly just replacing the n single-qubit gates by controlled gates, and in case n is odd replacing one U gate with a controlled U gate. For a set of vectors $\{\Phi^{(k)} \in \mathbb{R}^n \colon k \in \{0,1\}^m\}$ a multi-controlled alternating phase modulation sequence $\sum_{k\in\{0,1\}^m} |k\rangle\langle k| \otimes U_{\Phi^{(k)}}$ can be implemented similarly by replacing the single-qubit gates with multiply controlled single-qubit gates of the form $\sum_{k\in\{0,1\}^m} |k\rangle\langle k| \otimes e^{i\phi^{(k)}}$.

Proof:
See the constructions of Figure 2.2. □

2.4 Robustness of singular value transformation

In this section we prove results about the robustness of singular value transformation. More precisely we prove bounds on the difference $\left\|P^{(SV)}(A) - P^{(SV)}(\tilde{A})\right\|$ in terms of the magnitude of the initial "perturbation" $\left\|A - \tilde{A}\right\|$.

First consider the generalization of ordinary $\mathbb{R} \to \mathbb{C}$ functions to Hermitian matrices. One is tempted to think that if such a function is Lipschitz-continuous, then the induced operator function is also Lipschitz-continuous, however this turns out to be false. For a recent survey on the topic see the work of Aleksandrov and Peller [AP16].

Although the Lipschitz property cannot be saved directly, one need not lose more than some logarithmic factors in the modulus of continuity. We invoke a nice result form the theory of operator functions, quantifying this claim. The following theorem is due to Farforovskaya and Nikolskaya [FN09, Theorem 10].

2.4.1. THEOREM (Robustness of eigenvalue transformation).
Suppose that $f\colon [-1,1] \to \mathbb{C}$ is a function such that $\omega\colon [0,2] \to [0,\infty]$ is a modulus of continuity, i.e., for all $x, x' \in [-1,1]$

$$|f(x) - f(x')| \leq \omega(|x - x'|).$$

Then for all Hermitian matrices A, B such that $\|A\|, \|B\| \leq 1$, we have that

$$\|f(A) - f(B)\| \leq 4\left[\ln\left(\frac{2}{\|A - B\|} + 1\right) + 1\right]^2 \omega(\|A - B\|).$$

Now we show how this general theorem implies a general robustness result for singular value transformation.

2.4.2. Corollary (Robustness of singular value transformation 1).
If $f\colon [-1,1] \to \mathbb{C}$ is an even or odd function such that $\omega\colon [0,2] \to [0,\infty]$ is a modulus of continuity, and $A, \tilde{A} \in \mathbb{C}^{\tilde{d}\times d}$ are matrices of operator norm at most 1, then we have that

$$\left\| f^{(SV)}(A) - f^{(SV)}(\tilde{A}) \right\| \le 4\left[\ln\left(\frac{2}{\left\|A-\tilde{A}\right\|}+1\right)+1\right]^2 \omega\left(\left\|A-\tilde{A}\right\|\right).$$

Proof:
Let us assume that f is an even function and that $\tilde{d} \le d$. Then, using singular value decomposition, we can rewrite A as

$$A = W\begin{bmatrix}\Sigma & 0\end{bmatrix}V^\dagger,$$

where $W \in \mathbb{C}^{\tilde{d}\times\tilde{d}}$, $V \in \mathbb{C}^{d\times d}$ are unitaries and $\Sigma \in \mathbb{R}^{\tilde{d}\times\tilde{d}}$ is a diagonal matrix with nonnegative diagonal entries. Let $\overline{A} := \begin{bmatrix}0 & A\\ A^\dagger & 0\end{bmatrix} \in \mathbb{C}^{(\tilde{d}+d)\times(\tilde{d}+d)}$ be the Hermitian matrix obtained from A. We claim that

$$f(\overline{A}) = \begin{bmatrix} f^{(SV)}(A^\dagger) & 0\\ 0 & f^{(SV)}(A)\end{bmatrix}. \tag{2.35}$$

To prove this claim, first note that

$$\overline{A} = \begin{bmatrix}0 & A\\ A^\dagger & 0\end{bmatrix} = \begin{bmatrix} 0 & W\begin{bmatrix}\Sigma & 0\end{bmatrix}V^\dagger \\ V\begin{bmatrix}\Sigma\\0\end{bmatrix}W^\dagger & \begin{bmatrix}0&0\\0&0\end{bmatrix}\end{bmatrix} = \begin{bmatrix}W & 0\\ 0 & V\end{bmatrix}\begin{bmatrix}0&\Sigma&0\\ \Sigma&0&0\\0&0&0\end{bmatrix}\begin{bmatrix}W^\dagger & 0\\ 0 & V^\dagger\end{bmatrix}$$

and that

$$\begin{bmatrix}0 & \Sigma\\ \Sigma & 0\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}I & I\\ I & -I\end{bmatrix}\begin{bmatrix}\Sigma & 0\\ 0 & -\Sigma\end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix}I & I\\ I & -I\end{bmatrix}.$$

Therefore, if we denote

$$U = \begin{bmatrix}W & 0\\ 0 & V\end{bmatrix}\begin{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix}I & I\\ I & -I\end{bmatrix} & \begin{matrix}0\\0\end{matrix}\\ \begin{matrix}0 & 0\end{matrix} & I\end{bmatrix},$$

we get

$$\overline{A} = U\begin{bmatrix}\Sigma & 0 & 0\\ 0 & -\Sigma & 0\\ 0 & 0 & 0\end{bmatrix}U^\dagger,$$

which implies that

$$\begin{aligned} f(\overline{A}) =& U\begin{bmatrix}f(\Sigma) & 0 & 0\\ 0 & f(-\Sigma) & 0\\ 0 & 0 & f(0)I\end{bmatrix}U^\dagger = U\begin{bmatrix}f(\Sigma) & 0 & 0\\ 0 & f(\Sigma) & 0\\ 0 & 0 & f(0)I\end{bmatrix}U^\dagger\\ =& \begin{bmatrix}Wf(\Sigma)W^\dagger & \begin{matrix}0 & 0\end{matrix}\\ \begin{matrix}0\\0\end{matrix} & V\begin{bmatrix}f(\Sigma) & 0\\ 0 & f(0)I\end{bmatrix}V^\dagger\end{bmatrix} = \begin{bmatrix}f^{(SV)}(A^\dagger) & 0\\ 0 & f^{(SV)}(A)\end{bmatrix}.\end{aligned}$$

Thus, using Theorem 2.4.1 we get that

$$\begin{aligned}\left\|f^{(SV)}(A)-f^{(SV)}(\tilde{A})\right\| &\leq \left\|f(\overline{A})-f(\overline{\tilde{A}})\right\|\\ &\leq 4\left[\ln\left(\frac{2}{\left\|\overline{A}-\overline{\tilde{A}}\right\|}+1\right)+1\right]^2\omega\left(\left\|\overline{A}-\overline{\tilde{A}}\right\|\right)\\ &= 4\left[\ln\left(\frac{2}{\left\|A-\tilde{A}\right\|}+1\right)+1\right]^2\omega\left(\left\|A-\tilde{A}\right\|\right),\end{aligned}$$

which completes the proof for the case where f is an even function and $\tilde{d} \leq d$. The case $\tilde{d} \geq d$ can be handled by symmetry. Finally, the remaining case where f is odd can be handled similarly by observing that

$$f(\overline{A}) = \begin{bmatrix} 0 & f^{(SV)}(A) \\ f^{(SV)}(A^\dagger) & 0 \end{bmatrix}. \tag{2.36}$$

$\square$

We can also prove robustness results by bootstrapping our exact (non-robust) results, enabling us to remove the log factor from the above corollary under certain circumstances. We study two cases. First we make no extra assumptions, and establish error bounds that scale with the square root of the initial error $\|A-\tilde{A}\|$. Then we improve the dependence on the initial error to linear under the assumption that the matrices have norm significantly less than 1.

2.4.3. Lemma (Robustness of singular value transformation 2).
If $P \in \mathbb{C}[x]$ is a degree-n polynomial satisfying the requirements of Corollary 2.2.6, and $A, \tilde{A} \in \mathbb{C}^{\tilde{d}\times d}$ are matrices of operator norm at most 1, then we have[7] *that*

$$\left\|P^{(SV)}(A)-P^{(SV)}(\tilde{A})\right\| \leq 4n\sqrt{\left\|A-\tilde{A}\right\|}.$$

Proof:
Let $\varepsilon = \left\|\tilde{A}-A\right\|$, and let $B, \tilde{B} \in \mathbb{C}^{(d+\tilde{d})\times(d+\tilde{d})}$ be the matrices

$$B := \left[\begin{array}{cc} A & 0 \\ 0 & 0 \end{array}\right],\quad \tilde{B} := \left[\begin{array}{cc} \frac{\tilde{A}-A}{\varepsilon} & 0 \\ 0 & 0 \end{array}\right],$$

[7]Let us do a sanity check for $d = \tilde{d} = 1$. Let $A = 1$ and $\tilde{A} = 1 - \frac{1}{2n^2}$. For large n we have that $T_n(A) - T_n(\tilde{A}) \approx 1 - \cos(1) \approx 0.46$, whereas our upper bound gives $2\sqrt{2}$, showing that the upper bound is tight up to a constant factor, for arbitrary large n and for arbitrary small ε. (However, the joint dependence on n and ε might not be optimal.)

and let $U \in \mathbb{C}^{4(d+\tilde{d})\times 4(d+\tilde{d})}$ be a unitary such that[8]

$$U = \begin{bmatrix} B & 0 & \cdot & \cdot \\ 0 & \tilde{B} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}.$$

Such U must exist because $\|B\| \leq 1$ and $\left\|\tilde{B}\right\| \leq 1$. Let Π be the orthogonal projector projecting to the first d coordinates, and let $\widetilde{\Pi}$ be the orthogonal projector projecting to the first $\tilde{d}$ coordinates. Observe that $\widetilde{\Pi}U\Pi = A$. Let $W \in \mathbb{C}^{4(d+\tilde{d})\times 4(d+\tilde{d})}$ be the unitary

$$W := \begin{bmatrix} \sqrt{\frac{1}{1+\varepsilon}}I & -\sqrt{\frac{\varepsilon}{1+\varepsilon}}I & 0 & 0 \\ \sqrt{\frac{\varepsilon}{1+\varepsilon}}I & \sqrt{\frac{1}{1+\varepsilon}}I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix}.$$

Let $\bar{U} := W^\dagger U W$, and observe that $\widetilde{\Pi}\bar{U}\Pi = \tilde{A}/(1+\varepsilon)$. Also observe that

$$\|W - I\| = \sqrt{2 - 2/\sqrt{1+\varepsilon}} \leq \sqrt{\varepsilon},$$

therefore $\left\|U - \bar{U}\right\| \leq 2\sqrt{\varepsilon}$. Let $\Pi' = \widetilde{\Pi}$ if n is odd, and let $\Pi' = \Pi$ for n even. Let Φ be as in Corollary 2.2.6, then Theorem 2.3.7 implies that

$$\begin{aligned} \left\|P^{(SV)}(A) - P^{(SV)}(\tilde{A}/(1+\varepsilon))\right\| &= \left\|\Pi' U_\Phi \Pi - \Pi' \bar{U}_\Phi \Pi\right\| \leq \left\|U_\Phi - \bar{U}_\Phi\right\| \\ &\leq n\left\|U - \bar{U}\right\| \leq 2n\sqrt{\left\|A - \tilde{A}\right\|}. \end{aligned}$$

Let $B' \in \mathbb{C}^{(d+\tilde{d})\times(d+\tilde{d})}$ be the matrix

$$B' := \begin{bmatrix} \tilde{A} & 0 \\ 0 & 0 \end{bmatrix},$$

and let $U' \in \mathbb{C}^{4(d+\tilde{d})\times 4(d+\tilde{d})}$ be a unitary such that

$$U' = \begin{bmatrix} B' & 0 & \cdot & \cdot \\ 0 & 0 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}.$$

[8]We denote by $\cdot$ arbitrary matrix blocks and elements that are irrelevant for our presentation.

Observe that $\widetilde{\Pi}U'\Pi = \tilde{A}$, and $\bar{U}' := W^\dagger \tilde{V} W$ is such that $\widetilde{\Pi}\bar{U}'\Pi = \tilde{A}/(1+\varepsilon)$. By the same argument as before we get that

$$\left\| P^{(SV)}(\tilde{A}) - P^{(SV)}(\tilde{A}/(1+\varepsilon)) \right\| \leq 2n\sqrt{\left\| A - \tilde{A} \right\|}.$$

We can conclude using the triangle inequality. □

Now we establish another lemma which improves on the previous results in some cases, especially when the matrices have norm significantly less than 1.

2.4.4. Lemma (Robustness of singular value transformation 3).
If $P \in \mathbb{C}[x]$ is a degree-n polynomial satisfying the requirements of Corollary 2.2.6, and $A, \tilde{A} \in \mathbb{C}^{\tilde{d}\times d}$ are matrices of operator norm at most 1, such that

$$\left\| A - \tilde{A} \right\| + \left\| \frac{A+\tilde{A}}{2} \right\|^2 \leq 1,$$

then we have that

$$\left\| P^{(SV)}(A) - P^{(SV)}(\tilde{A}) \right\| \leq n\sqrt{\frac{2}{1 - \left\| \frac{A+\tilde{A}}{2} \right\|^2}} \left\| A - \tilde{A} \right\|.$$

Proof:
Let $B, \tilde{B} \in \mathbb{C}^{(d+\tilde{d})\times(d+\tilde{d})}$ be the matrices

$$B := \begin{bmatrix} \frac{A+\tilde{A}}{\|A+\tilde{A}\|} & 0 \\ 0 & 0 \end{bmatrix}, \quad \tilde{B} := \begin{bmatrix} \frac{A-\tilde{A}}{\|A-\tilde{A}\|} & 0 \\ 0 & 0 \end{bmatrix}.$$

Let $x > 1$ and let $U \in \mathbb{C}^{4(d+\tilde{d})\times 4(d+\tilde{d})}$ be a unitary such that

$$U = \begin{bmatrix} \sqrt{\frac{x-1}{x}}B & \sqrt{\frac{1}{x}}\tilde{B} & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}.$$

Let $C := \sqrt{\frac{x-1}{x}}B \oplus \sqrt{\frac{1}{x}}\tilde{B}$ be the top-left block of U. It is easy to see that

$$\|C\|^2 \leq \frac{x-1}{x}\|B\|^2 + \frac{1}{x}\left\|\tilde{B}\right\|^2 = \frac{x-1}{x} + \frac{1}{x} = 1,$$

therefore a unitary U must exist with C being the top-left block. Suppose that

$$\frac{x}{x-1}\frac{\left\|A+\tilde{A}\right\|^2}{4} + x\frac{\left\|A-\tilde{A}\right\|^2}{4} = 1. \tag{2.37}$$

Let $W_\pm \in \mathbb{C}^{4(d+\tilde{d})\times 4(d+\tilde{d})}$ be the unitary

$$W_\pm := \begin{bmatrix} \sqrt{\frac{x}{x-1}}\frac{\|A+\tilde{A}\|}{2}I & \mp\sqrt{x}\frac{\|A-\tilde{A}\|}{2}I & 0 & 0 \\ \pm\sqrt{x}\frac{\|A-\tilde{A}\|}{2}I & \sqrt{\frac{x}{x-1}}\frac{\|A+\tilde{A}\|}{2}I & 0 & 0 \\ 0 & 0 & I & 0 \\ 0 & 0 & 0 & I \end{bmatrix}.$$

Let Π be the orthogonal projector projecting to the first d coordinates, and let $\widetilde{\Pi}$ be the orthogonal projector projecting to the first $\tilde{d}$ coordinates. Observe that $\widetilde{\Pi}UW_+\Pi = A$ and $\widetilde{\Pi}UW_-\Pi = \tilde{A}$. Also observe that $\|W_+ - W_-\| = \sqrt{x}\left\|A-\tilde{A}\right\|$, thus $\|UW_+ - UW_-\| = \sqrt{x}\left\|A-\tilde{A}\right\|$.

Let $\varepsilon := \left\|A-\tilde{A}\right\|^2$ and let $\delta := 4 - \left\|A+\tilde{A}\right\|^2$. We can rewrite (2.37) as

$$\frac{x}{x-1}\frac{4-\delta}{4} + x\frac{\varepsilon}{4} = 1, \tag{2.38}$$

which has a solution

$$x = \frac{4}{\delta+\varepsilon}\left(1 + \frac{\left(1-\frac{8\varepsilon}{(\delta+\varepsilon)^2}\right) - \sqrt{1-\frac{16\varepsilon}{(\delta+\varepsilon)^2}}}{\frac{8\varepsilon}{(\delta+\varepsilon)^2}}\right). \tag{2.39}$$

Now let $y := 8\varepsilon/(\delta+\varepsilon)^2$. If $\varepsilon \leq \delta^2/16$, then $y \leq \frac{1}{2}$, and so $\frac{(1-y)-\sqrt{1-2y}}{y} \leq 1$. Hence for $\varepsilon \leq \delta^2/16$ by Eq. (2.39) we get that $x \leq 8/(\delta+\varepsilon)$, and therefore $\|UW_+ - UW_-\| = \sqrt{8/(\delta+\varepsilon)}\left\|A-\tilde{A}\right\| \leq \sqrt{8/\delta}\left\|A-\tilde{A}\right\|$.

Now we proceed similarly to the proof of Lemma 2.4.3. Let $\Pi' = \widetilde{\Pi}$ if n is odd, and let $\Pi' = \Pi$ for n even. Let Φ be as in Corollary 2.2.6 and let $U^{(\pm)} := UW_\pm$, then Theorem 2.3.7 implies that

$$\begin{aligned}\left\|P^{(SV)}(A) - P^{(SV)}(\tilde{A})\right\| &= \left\|\Pi' U_\Phi^+ \Pi - \Pi' U_\Phi^- \Pi\right\| \leq \left\|U_\Phi^+ - U_\Phi^-\right\| \leq n\left\|U^+ - U^-\right\| \\ &= n\sqrt{\frac{8}{\delta}}\left\|A-\tilde{A}\right\|.\end{aligned}$$

Finally note that $\varepsilon \leq \delta^2/16$ is equivalent to $4\sqrt{\varepsilon} \leq \delta$, which by definition is equivalent to

$$\left\|A-\tilde{A}\right\| + \left\|\frac{A+\tilde{A}}{2}\right\|^2 \leq 1.$$

□

Chapter 3

Constructing quantum algorithms by QSVT

In this chapter we show that quantum singular value transformation leads to novel algorithms. We propose a new method for singular value estimation, and show how to exponentially improve the complexity of implementing fractional queries to unitaries with a gapped spectrum. Finally, as a quantum machine learning application we show how to efficiently implement principal component regression.

We also show that quantum singular value transformation leads to a conceptually simple unified framework of quantum algorithms incorporating a variety of quantum speed-ups. Our framework allows us to describe quantum algorithms on a high level, hopefully making them accessible to researchers even outside the quantum algorithms community. We illustrate this by showing how our meta-algorithm generalizes a number of prominent quantum algorithms, and quickly (re)derive the following algorithms: optimal Hamiltonian simulation, implementing the Moore-Penrose pseudoinverse (i.e., the HHL algorithm) with exponential precision, fixed-point amplitude amplification, robust oblivious amplitude amplification, fast QMA amplification, fast quantum OR lemma, certain quantum walk results and several quantum machine learning algorithms.

In order to exploit the strengths of the presented method, it is useful to know its limitations too, therefore we also prove a bound on the efficiency of quantum singular value transformation, which often gives optimal lower bounds.

3.1 Introduction

We show that many prominent quantum algorithms can be viewed as an instantiation of our quantum singular value transformation meta-algorithm, when applied with an appropriately chosen polynomial. In order to illustrate the power of this

This chapter is based on [GSLW19], with extensions from [vAGGdW17, Appendix B,C].

technique we briefly explain some corollaries, by showing natural examples of projected unitary encodings. For example suppose that U is a quantum algorithm that, starting from the initial state $|0\rangle^{\otimes n}$, prepares a desired state with success probability at least p, and indicates success by setting the first qubit to $|1\rangle$. Then we can take $\widetilde{\Pi} := |1\rangle\langle 1| \otimes I_{n-1}$ and $\Pi := |0\rangle\langle 0|^{\otimes n}$. Observe that $A = \widetilde{\Pi}U\Pi$ is a rank-1 matrix having a single non-trivial singular value which is the square root of the success probability. If P is an odd polynomial bounded by 1 in absolute value such that P is $\frac{\varepsilon}{2}$-close to 1 on the interval $[\sqrt{p}, 1]$, then by applying singular value transformation we get an algorithm U_Φ that succeeds with probability at least $1-\varepsilon$. Such a polynomial can be constructed with degree $\mathcal{O}\left(\frac{1}{\sqrt{p}}\log\left(\frac{1}{\varepsilon}\right)\right)$ providing a conceptually simple and efficient implementation of *fixed-point* amplitude amplification, which prepares the desired state in one shot. This improves on ordinary amplitude amplification that potentially requires multiple repetitions to achieve high success probability.

It also becomes straightforward to implement the Moore-Penrose pseudoinverse directly (i.e., the HHL algorithm). Suppose that $A = W\Sigma V^\dagger$ is an SVD, then the pseudoinverse is simply $A^+ = V\Sigma^{-1}W^\dagger$, where we take the inverse of each non-zero diagonal element of Σ. If we have A represented as a projected unitary encoding, then simply finding an appropriately scaled approximation polynomial of $\frac{1}{x}$ and applying singular value transformation to it implements an approximation of the Moore-Penrose pseudoinverse directly. As an application in quantum machine learning, we design a quantum algorithm for *principal component regression*, and argue that singular value transformation could become a central tool in designing quantum machine learning algorithms.

Based on singular value transformation we develop some *new* algorithms as well, including the *singular vector transformation* algorithm, which maps right singular vectors to left singular vectors in a single-shot manner. This is a common generalization and extension of fixed-point amplitude amplification [YLC14] and oblivious amplitude amplification [BCC+14], described in Section 3.2.1.

3.1.1. Theorem (Informal version of Theorem 3.2.3). *Call an application of the unitary U, or the controlled reflection operators $(2\Pi - I)$, $(2\widetilde{\Pi} - I)$ a "query". If*

$$\widetilde{\Pi}U\Pi = \sum_{i=1}^{k} \varsigma_i |\phi_i\rangle\langle\psi_i|$$

is a singular value decomposition, then we can transform an arbitrary input state

$$|\psi\rangle = \sum_{i=i}^{k} \alpha_i |\psi_i\rangle \quad \textit{to} \quad |\phi\rangle = \sum_{i=i}^{k} \alpha_i |\phi_i\rangle,$$

with precision ε, in query and gate complexity $\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{1}{\varepsilon}\right)\right)$, under the assumption that $\varsigma_i \geq \delta$ for all $\alpha_i \neq 0$.

This algorithm also gives an efficient solution to a form of "non-commutative measurement" problem used for efficient ground-state preparation of certain local Hamiltonians [GS17], and quadratically improves the gap dependence of this algorithm. It can also be used for constructing a new method for singular value estimation [KP17b, CGJ19].

As another application we develop *singular value threshold projectors*, which project out singular vectors with singular value below a certain threshold. These threshold projectors play a major role in quantum algorithms recently proposed by Kerenidis et al. [KP17b, KL18], and our work fills a minor gap that was present in earlier implementation proposals. Our implementation is also simpler and applies in greater generality than the algorithm of Kerenidis and Prakash [KP17b]. As a useful application of singular value threshold projectors we develop *singular value discrimination*, which decides whether a given quantum state has singular value below or above a certain threshold.

Other algorithms can also be cast in the singular value transformation framework, including optimal Hamiltonian simulation, robust oblivious amplitude amplification, fast QMA amplification, fast quantum OR lemma and certain quantum walk results. Based on these techniques we also show how to exponentially improve the complexity of implementing fractional queries to unitaries with a gapped spectrum. We summarize in Table 3.1 the various types of quantum speed-ups that are inherently incorporated in our singular value transformation framework.

Speed-up	Source of speed-up	Examples of algorithms
Exponential	Dimensionality of the Hilbert space	Hamiltonian simulation [Llo96]
	Precise polynomial approximations	Improved HHL algorithm [CKS17]
Quadratic	Singular value = square root of probability	Grover search [Gro96]
	Distinguishability of singular values	Amplitude estimation [BHMT02]
	Singular values close to 1 are more useful	Quantum walks [Szeg04]

Table 3.1. This table gives an intuitive summary of the different types of speed-ups that our singular value transformation framework inherently incorporates. The explanations, examples and the cited papers are far from complete or representative, the table only intends to give some intuition and to illustrate the different sources of speed-ups.

3.1.1 Block-encoding

In order to harness the power of quantum singular value transformation one needs to construct projected unitary encodings. A special case of projected unitary encoding is called *block-encoding*, when $\widetilde{\Pi} = \Pi = |0\rangle\langle 0|^{\otimes a} \otimes I$. In this case A is literally[1] the top-left block of the unitary U:

$$A = \big(\langle 0|^{\otimes a} \otimes I\big) U \big(|0\rangle^{\otimes a} \otimes I\big) \iff U = \begin{bmatrix} A & \cdot \\ \cdot & \cdot \end{bmatrix}.$$

We call such a unitary U an a-qubit block-encoding of A. One can think of U as a probabilistic implementation of A: given an input state $|\psi\rangle$, applying the unitary U to the state $|0\rangle^{\otimes a}|\psi\rangle$, measuring the first a-qubit register and post-selecting on the $|0\rangle^{\otimes a}$ outcome, we get a state $\propto A|\psi\rangle$ in the second register.

In Section 3.3 we provide a versatile toolbox for efficiently constructing block-encodings, summarizing recent developments in the field. In particular we demonstrate how to construct block-encodings of unitary matrices, density operators, POVM operators, sparse-access matrices and matrices stored in a QROM[2]. Furthermore, we show how to form linear combinations and products of block-encodings, enabling the efficient implementation of quantum matrix arithmetics. Quantum matrix arithmetics carries out all calculations in an operational way, meaning that the matrices are represented by block-encodings, enabling exponential speed-ups in terms of the dimension of the matrices.

When we work with block-encodings, the circuits for quantum singular value transformation become especially nice and easy to describe. Our main theorem applied to block-encodings gives the following result:

3.1.2. Theorem (Special case of Theorem 2.3.7). *Let $P\colon [-1,1] \mapsto [-1,1]$ be a degree-d even/odd polynomial map. Suppose that U is a block-encoding of A that has singular value decomposition $A = \sum_i \varsigma_i |w_i\rangle\langle v_i|$. Let H denote the Hadamard matrix, then $(H \otimes I) U_\Phi (H \otimes I)$ is a block-encoding of*

$$\sum_i P(\varsigma_i)|w_i\rangle\langle v_i| \quad \textit{if } d \textit{ is odd, and}$$
$$\sum_i P(\varsigma_i)|v_i\rangle\langle v_i| \quad \textit{if } d \textit{ is even,}$$

where $\Phi \in \mathbb{R}^d$ is (classically) efficiently computable given the coefficients of P, and U_Φ is the following circuit:

[1] When we talk about block-encodings we omit the dimensions projected out by $|0\rangle\langle 0|^{\otimes a}$ for convenience, i.e., A and U have different sizes, unlike in projected unitary encodings.

[2] By QROM we mean quantum read-only memory, which stores classical data that can be accessed in superposition.

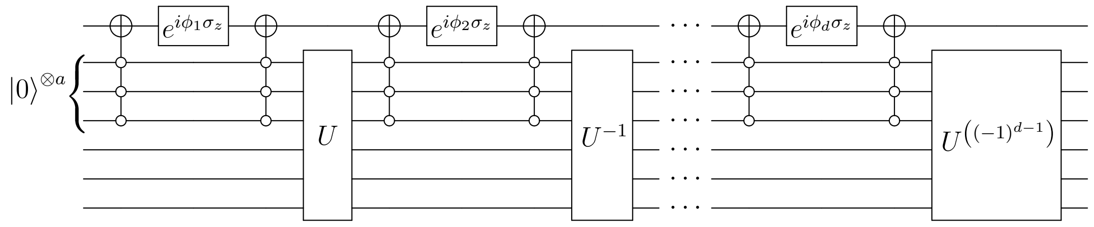


Figure 3.1. Circuits used for quantum singular value transformation. The empty dots denote control by the $|0\rangle$ state, so that the corresponding gate is an "inverted" Toffoli gate, where each qubit is conjugated by an X gate compared to the usual Toffoli gate. The other gates are single-qubit rotations or applications of U or $U^\dagger$. The structure is very similar to the amplitude amplification circuit of Figure 1.2.

In fact it turns out that some earlier quantum algorithms also constructed and used block-encodings under the hood. For example we show that the update operator of a quantum walk corresponding to a reversible Markov chain M [MNRS11] essentially implements a block-encoding of the "discriminant matrix" $D(M)$ of the Markov chain. Moreover, applying k steps of the quantum walk operator applies the Chebyshev polynomial T_{2k} to the discriminant matrix. In fact our quantum singular value transformation circuit corresponds to a Szegedy [Szeg04] quantum walk in the special case when all phases are set to $\phi_j := \frac{\pi}{2}$.

Using the above theorem we can reprove a recent result about quantum fast-forwarding [AS18] of Markov chains in a few lines, while substantially improving its complexity. As we discussed above, quantum walks provide block-encodings of the discriminant matrix $D(M)$. On the other hand we know [SV14], cf. Theorem 3.4.9, that the monomial x^t can be ε-approximated by a polynomial P of degree $\sqrt{2t\log(2/\varepsilon)}$ on the interval $[-1,1]$. Since the discriminant matrix is symmetric, by using this polynomial P in Theorem 3.1.2 we get an ε-approximate block-encoding of $D(M)^t$. This solves the quantum fast-forwarding problem, since in some sense it emulates t steps of the walk using only $\propto \sqrt{t}$ quantum operations. Due to Theorem 3.1.2 the (query) complexity of this implementation is $\mathcal{O}\big(\sqrt{t\log(1/\varepsilon)}\big)$, which is a significant improvement in terms of precision over the complexity $\mathcal{O}\big(\sqrt{t/\varepsilon}\log\left(1/\varepsilon\right)\big)$ of the original[3] result of Apers and Sarlette [AS18].

In the special case when the block-encoded matrix is Hermitian we can remove the parity constraint from Theorem 3.1.2 by combining the even and odd parts of the polynomials. This, e.g., enables us to give a simple proof of recent optimal block-Hamiltonian simulation results [LC17b].

Since in our framework, designing quantum algorithms mostly boils down to finding low-degree polynomial approximations to various functions we also develop some general tools for finding such approximations. We build on existing results from the theory of approximation polynomials, and develop some new results that are adapted to the boundedness requirement of Theorem 3.1.2. We also

[3]In the second arXiv version of their paper Apers and Sarlette also recovered this improved result using an alternative method, but they also cite our approach [GSLW19].

show that in general for smooth functions the error dependence is asymptotically logarithmic.

To actually construct the quantum circuit corresponding to a given polynomial one should first find the corresponding angle sequence. Our proofs are constructive, so in principle they also provide an algorithm for finding the angles. However, the described method requires finding roots of high-degree polynomials, which can be computationally expensive. Fortunately, as we mentioned in Chapter 2, recently Haah [Haa18] developed an optimized classical algorithm for finding the angles, that has cubic dependence on the degree, and polylogarithmic dependence on the error, which shows that the classical preprocessing can indeed be efficiently performed.

3.1.2 A lower bound

We also prove a bound on the efficiency of singular value transformation. Our lower bound suggests that the spectrum of a Hermitian block-encoded matrix H lying close to ± 1 is more "flexible" than the spectrum lying below, say, $\frac{1}{2}$ in absolute value. It also gives a lower bound on singular value transformation, as Hermitian eigenvalue transformation is a special case of singular value transformation.

3.1.3. Theorem (Informal version of Theorem 3.5.1). *Let $I \subseteq [-1,1]$ and suppose a unitary U block-encodes an unknown Hermitian matrix H with the only promise that the spectrum of H lies in I. Let $f : I \to \mathbb{R}$, and suppose that we have a quantum circuit V that block-encodes $f(H)$ with accuracy ε using T applications of U or $U^\dagger$. Then for all $x \neq y \in I \cap [-\frac{1}{2}, \frac{1}{2}]$ we have that $T = \Omega\Big(\frac{|f(x)-f(y)|-2\varepsilon}{|x-y|}\Big)$.*

This lower bound shows for example the optimality of our pseudoinverse implementation. For simplicity let us assume that A is Hermitian, and suppose that $\|A^{-1}\| \leq \kappa$, then we can find a polynomial ε-approximation of $\frac{1}{2\kappa x}$ on the interval $[-1,1] \setminus (-\frac{1}{\kappa}, \frac{1}{\kappa})$ of degree $\mathcal{O}(\kappa \log(1/\varepsilon))$, which also gives the complexity of our ε-approximate implementation of $\frac{A^{-1}}{2\kappa}$, the subnormalized inverse. On the other hand the absolute value of the derivative of the function $\frac{1}{2\kappa x}$ at $\frac{1}{\kappa}$ is $\frac{\kappa}{2}$, which shows that our implementation is essentially optimal up to the $\log(1/\varepsilon)$ factor.

Our lower bound also shows that our singular/eigenvalue projector implementations are essentially optimal. Suppose we have a block-encoding of a Hermitian matrix. If we want to approximately implement a projector projecting out every eigenvalue below say a and keeping every eigenvalue above b, then we need to use the block-encoding $\Omega\big(\frac{1}{b-a}\big)$, unless a or b is very close to ± 1. However, if $a = -1$ or $b = 1$, then we can get a quadratic advantage [LC17a], which has implications to ground state preparation [GTC17].

3.1.3 Structure of the chapter

In Section 3.2 we show some direct applications of quantum singular value transformation. In particular we introduce singular vector transformation and singular value amplification in Subsection 3.2.1. Using these results we give simple derivations of fixed-point amplitude amplification and robust oblivious amplitude amplification. We then extend these ideas in Subsection 3.2.2 to solve the problem of singular value threshold projection and singular value discrimination. This then allows us to detect and find marked elements in a reversible Markov chain. Then in Subsection 3.2.3 we provide an easy derivation of the quantum linear-systems algorithm, and more generally the quantum least-squares fitting algorithm. In Subsection 3.2.4, we design a quantum algorithm for principal component regression, and show how various other machine learning problems can be solved within our framework. Finally, in Subsection 3.2.5 we propose a new method for quantum singular value estimation.

Section 3.3 shows how to efficiently construct block-encodings and contains a discussion of how these techniques can be employed to perform matrix arithmetic on a quantum computer. In particular we show how to perform basic linear algebra operations on Hamiltonians using block-encodings; we discuss matrix addition and multiplication in Subsections 3.3.3 and 3.3.4. We follow this up with a discussion of how arbitrary smooth functions of Hermitian matrices can be performed in Section 3.4. We give an elementary proof of the complexity of block-Hamiltonian simulation in Subsection 3.4.1, discuss approximating piecewise smooth functions of Hamiltonians in Subsection 3.4.2, and present the special cases of Gibbs-state preparation and fractional queries in Subsection 3.4.3. We then conclude by proving lower bounds for implementing functions of Hermitian matrices in Section 3.5, which in turn implies lower bounds on singular value transformation. Finally in Appendix 3.A we derive a generalized version of the quantum minimum finding algorithm of [DH96].

3.2 Some direct applications of quantum singular value transformation

3.2.1 Singular vector transformation and singular value amplification

In this subsection we derive some corollaries of singular value transformation. We call the first corollary *projected singular vector transformation*, because it implements a unitary that transforms the right singular vectors to the left singular vectors above some singular value threshold. Then we show how to quickly derive advanced amplitude amplification results using this general technique. Finally, we develop a corollary called *singular value amplification*, showing how to uniformly amplify the singular values of a matrix represented as a projected unitary.

First we define singular value threshold projectors. We note that the singular value decomposition is not necessarily unique, however these projectors are well defined.

3.2.1. DEFINITION (Singular value threshold projectors). Let $A=\widetilde{\Pi}U\Pi=W\Sigma V^\dagger$ be a singular value decomposition of a projected unitary. For $S\subseteq\mathbb{R}$ we define $\Pi_S := \Pi V\Sigma_S V^\dagger\Pi$, and similarly $\widetilde{\Pi}_S := \widetilde{\Pi}W\Sigma_S W^\dagger\widetilde{\Pi}$. For $\delta\in\mathbb{R}$ we define $\Pi_{\geq\delta} := \Pi_{[\delta,\infty)}$, also we define $\Pi_{>\delta},\Pi_{\leq\delta},\Pi_{<\delta},\Pi_{=\delta}$ and $\widetilde{\Pi}_{>\delta},\widetilde{\Pi}_{\leq\delta},\widetilde{\Pi}_{<\delta},\widetilde{\Pi}_{=\delta}$ analogously.

Then we invoke a result of Low and Chuang [LC17a, Corollary 6] about constructive polynomial approximations of the sign function. The error of the optimal approximator, studied by Eremenko and Yuditskii [EY07], has similar scaling but the result is non-constructive.

3.2.2. LEMMA (Polynomial approximations of the sign function). *For all $\delta>0$, $\varepsilon\in(0,1/2)$ there exists an efficiently computable odd polynomial $P\in\mathbb{R}[x]$ of degree $n=\mathcal{O}\left(\frac{\log(1/\varepsilon)}{\delta}\right)$, such that*

- *for all $x\in[-2,2]$: $|P(x)|\leq 1$, and*
- *for all $x\in[-2,2]\setminus(-\delta,\delta)$: $|P(x)-\mathrm{sign}(x)|\leq\varepsilon$.*

Now we are ready to prove our result about singular value transformation. Our singular vector transformation implements a unitary which maps a right singular vector having singular value at least δ to the corresponding left singular vector.

3.2.3. THEOREM (Singular vector transformation). *Let $U,\Pi,\widetilde{\Pi}$ be as in Theorem 2.3.7 and let $\delta>0$. Suppose that $\widetilde{\Pi}U\Pi=W\Sigma V^\dagger$ is a singular value decomposition. Then there is an $m=\mathcal{O}\left(\frac{\log(1/\varepsilon)}{\delta}\right)$ and a $\Phi\in\mathbb{R}^m$ such that $\left\|\widetilde{\Pi}_{\geq\delta}U_\Phi\Pi_{\geq\delta}-\widetilde{\Pi}_{\geq\delta}(WV^\dagger)\Pi_{\geq\delta}\right\|\leq\varepsilon$. Moreover, U_Φ can be implemented using a single ancilla qubit, with m uses of U and $U^\dagger$, m uses of $C_\Pi NOT$ and m uses of $C_{\widetilde{\Pi}}NOT$ gates and m single-qubit gates.*

Proof:
By Lemma 3.2.2 we can construct an odd polynomial $P_\Re\in\mathbb{R}[x]$ of degree $m=\mathcal{O}\left(\frac{\log(1/\varepsilon^2)}{\delta}\right)$ that approximates the sign function with $\varepsilon^2/2$ precision on the domain $[-1,1]\setminus(-\delta,\delta)$. By Corollary 2.2.8 we know that there exists a polynomial P of the same degree as $P_\Re$ such that $\Re[P]=P_\Re$, moreover P satisfies the conditions of Corollary 2.2.6. Use singular value transformation Theorem 2.3.7 to construct a $\Phi\in\mathbb{R}^m$ such that $\widetilde{\Pi}U_\Phi\Pi=P^{(SV)}\left(\widetilde{\Pi}U\Pi\right)$ up to error ε, and observe

that $\left\|\widetilde{\Pi}_{\geq\delta}P^{(SV)}\left(\widetilde{\Pi}U\Pi\right)\Pi_{\geq\delta}-\widetilde{\Pi}_{\geq\delta}(WV^\dagger)\Pi_{\geq\delta}\right\|\leq\varepsilon$. Conclude the gate complexity using Lemma 2.3.9. $\square$

As an easy corollary we recover and improve upon fixed-point amplitude amplification results [Hø00, Gro05, AC12, YLC14]. In amplitude amplification we are given a unitary U that maps $|\psi_0\rangle\mapsto\sqrt{p}|0\rangle|\psi_G\rangle+\sqrt{1-p}|1\rangle|\psi_B\rangle$, and the goal is to prepare the state $|\psi_G\rangle$. Since ordinary amplitude amplification may "overamplify" we need multiple repetitions in order to succeed, and if we want to perform amplification coherently it creates a large garbage state attached to $|\psi_G\rangle$. Fixed-point amplitude amplification provides a way to perform the $|\psi_0\rangle\mapsto|\psi_G\rangle$ mapping without creating a garbage state attached to $|\psi_G\rangle$.

Our solution combines the advantages of prior art. On the one hand, the query complexity of $\mathcal{O}(\frac{1}{\delta}\operatorname{poly}(1/\varepsilon))$ by [AC12] is optimal with respect to target state overlap δ, but converges slowly with respect to error ε. On the other hand, the query complexity of $\mathcal{O}(\frac{1}{\delta}\log(1/\varepsilon))$ by [YLC14] is optimal and exhibits exponentially fast convergence with respect to the error, but it introduces an unknown phase on the amplified state, which can be problematic when used as a quantum subroutine in superposition. Our presented approach has the same optimal asymptotic scaling and also ensures that this phase error is ϵ-close to 0.

3.2.4. Theorem (Fixed-point amplitude amplification). *For every $\delta>0$ there is a unitary circuit Q, which uses a single ancilla qubit and consists of $\mathcal{O}\left(\frac{\log(1/\varepsilon)}{\delta}\right)$ $U,U^\dagger$, $C_\Pi NOT$, $C_{|\psi_0\rangle\langle\psi_0|}NOT$ and $e^{i\phi\sigma_z}$ gates, and has the following property: If $\Pi U|\psi_0\rangle=a|\psi_G\rangle$ for some $a\geq\delta$, then $\||\psi_G\rangle-Q|\psi_0\rangle\|\leq\varepsilon$. (The structure of Q is independent of U and $|\psi_G\rangle$.)*

Proof:
Set $\widetilde{\Pi}:=\Pi$ and $\Pi':=|\psi_0\rangle\langle\psi_0|$ and observe that

$$\widetilde{\Pi}U\Pi'=a|\psi_G\rangle\langle\psi_0|.$$

Now use Theorem 3.2.3 in order to get an algorithm Q that satisfies

$$\||\psi_G\rangle\langle\psi_G|Q|\psi_0\rangle\langle\psi_0|-|\psi_G\rangle\langle\psi_0|\|\leq\varepsilon.$$ $\square$

Another easy-to-derive corollary of our machinery is robust oblivious amplitude amplification [BCC+15], which is a technique originally developed [BCC+14] for Hamiltonians simulation.[4] This algorithm solves the following problem: given one copy of a quantum state $|\psi\rangle$, apply the unitary U to this state, having access only to a subnormalized implementation of U in the form of a block-encoding $(\langle 0|\otimes I)V(|0\rangle\otimes I)=U/100$.

[4]Note that we could also easily derive a fixed-point version of oblivious amplitude amplification based on Theorem 3.2.3, but we state the usual version instead for readability.

If we would simply apply V to the quantum state $|0\rangle|\psi\rangle$, then by measuring the first register, and only accepting the $|0\rangle$ outcome, we get a probabilistic implementation. In order to boost the success probability we could use multiple copies, or apply amplitude amplification. But we only have a single copy, and ordinary amplitude amplification uses a reflection about the initial state $|0\rangle|\psi\rangle$, which might be expensive to implement. Fortunately it turns out that the amplitude amplification circuit still works if replace the reflection about $|0\rangle|\psi\rangle$ by a reflection about $|0\rangle$, acting non-trivially only on the ancilla register, resulting in a circuit that is oblivious to the input state $|\psi\rangle$. The fact that this procedure works crucially relies on the assumption that $(\langle 0| \otimes I)V(|0\rangle \otimes I) = U/100$ is a subnormalized *unitary*. If the subnormalization factor is exactly $\sin(\pi/(2k+1))$, then k application of the oblivious amplification circuit solves the problem exactly. If the subnormalization is known, we can always apply a bit of extra subnormalization resulting in a subnormalization factor of this form. Finally, if we do not exactly have a subnormalized unitary, just an operator close to it, we can still bound the accumulating errors, giving a robust version of oblivious amplitude amplification.

3.2.5. THEOREM (Robust oblivious amplitude amplification). *Let $n \in \mathbb{N}_+$ be odd, let $\varepsilon \in \mathbb{R}_+$, let U be a unitary, let $\widetilde{\Pi}, \Pi$ be orthogonal projectors, and let $W : \mathrm{img}(\Pi) \mapsto \mathrm{img}\left(\widetilde{\Pi}\right)$ be an isometry, such that*

$$\left\|\sin\left(\frac{\pi}{2n}\right)W|\psi\rangle - \widetilde{\Pi}U|\psi\rangle\right\| \leq \varepsilon \tag{3.1}$$

for all $|\psi\rangle \in \mathrm{img}(\Pi)$. Then we can construct a unitary $\tilde{U}$ such that for all $|\psi\rangle \in \mathrm{img}(\Pi)$

$$\left\|W|\psi\rangle - \widetilde{\Pi}\tilde{U}|\psi\rangle\right\| \leq 2n\varepsilon,$$

which uses a single ancilla qubit, with n uses of U and $U^\dagger$, n uses of $C_\Pi NOT$ and n uses of $C_{\widetilde{\Pi}}NOT$ gates, and n single-qubit gates.

Proof:
First we prove the $\varepsilon = 0$ case, by reproducing the polynomials stemming from ordinary amplitude amplification. Let $T_n \in \mathbb{R}[x]$ be the degree-n Chebyshev polynomial of the first kind. As discussed after Corollary 2.2.6 there is an easy to describe $\Phi \in \mathbb{R}^n$ which corresponds to T_n in equation (2.15).

Now observe that by (3.1) we have that $\widetilde{\Pi}U\Pi = \sin\left(\frac{\pi}{2n}\right)W$. We can apply singular value transformation using T_n to obtain U_Φ such that

$$\widetilde{\Pi}U_\Phi\Pi = T_n\left(\sin\left(\frac{\pi}{2n}\right)\right)W = T_n\left(\cos\left(\frac{\pi}{2} - \frac{\pi}{2n}\right)\right)W = \cos\left(\frac{n-1}{2}\pi\right)W = \pm W.$$

After correcting the global phase ± 1 (which depends on the parity of $(n-1)/2$) we get $\tilde{U} := \pm U_\Phi$ such that for all $|\psi\rangle \in \mathrm{img}(\Pi)$ we have $\tilde{U}|\psi\rangle = W|\psi\rangle$. The complexity statement follows from Lemma 2.3.9.

In the $\varepsilon \neq 0$ case we first handle some trivial cases. If $n = 1$ or $\varepsilon > \frac{1}{3}$ we simply take $\tilde{U} := U$. Otherwise if $n \geq 3$ and $\varepsilon \in [0, \frac{1}{3}]$ the error bounds follow from Lemma 2.4.4, in the following way: Let $A := \sin\!\left(\frac{\pi}{2n}\right)W$ and let $\tilde{A} := \widetilde{\Pi}U\Pi$, by (3.1) we have that $\left\|A - \tilde{A}\right\| \leq \varepsilon$. Then

$$\left\|A + \tilde{A}\right\| \leq \|A\| + \|A\| + \left\|\tilde{A} - A\right\| = 2\sin\!\left(\frac{\pi}{2n}\right) + \varepsilon \leq 2\sin\!\left(\frac{\pi}{6}\right) + \frac{1}{3} = \frac{4}{3},$$

thus $\left\|\frac{A+\tilde{A}}{2}\right\|^2 \leq \frac{4}{9}$ and $\left\|A - \tilde{A}\right\| + \left\|\frac{A+\tilde{A}}{2}\right\|^2 \leq \frac{7}{9} < 1$. This also implies that $\sqrt{\frac{2}{1-\left\|\frac{A+\tilde{A}}{2}\right\|^2}} \leq \sqrt{\frac{2}{1-\frac{4}{9}}} = \sqrt{\frac{18}{5}} < 2$, and therefore by Lemma 2.4.4 we get that $\left\|W - \widetilde{\Pi}\tilde{U}\Pi\right\| \leq 2n\varepsilon$. $\square$

Now we turn to solving the linear singular value amplification problem. That is, given a matrix in a projected encoding form, construct a projected encoding of a matrix which has singular values that are γ times larger than the original singular values.

In order to proceed we first construct some polynomials similarly that can be used in combination with our singular value transformation results.

3.2.6. Lemma (Polynomial approximations of the rectangle function). *Let $\delta', \varepsilon' \in (0, \frac{1}{2})$ and $t \in [-1, 1]$. There exists an even polynomial $P' \in \mathbb{R}[x]$ of degree $\mathcal{O}\!\left(\log(\frac{1}{\varepsilon'})/\delta'\right)$, such that $|P'(x)| \leq 1$ for all $x \in [-1, 1]$, and*

$$\begin{cases} P'(x) \in \quad [0, \varepsilon'] & \text{for all } x \in [-1, -t-\delta'] \cup [t+\delta', 1], \text{ and} \\ P'(x) \in [1-\varepsilon', 1] & \text{for all } x \in [-t+\delta', t-\delta']. \end{cases} \tag{3.2}$$

Proof:
First let us take a real polynomial P which $\frac{\varepsilon'}{2}$-approximates the sign function on the interval $[-2, 2] \setminus (-\delta', \delta')$, moreover for all $x \in [-2, 2]$: $|P(x)| \leq 1$. Such a polynomial of degree $\mathcal{O}\!\left(\frac{1}{\delta'}\log\!\left(\frac{1}{\varepsilon'}\right)\right)$ can be efficiently constructed by Lemma 3.2.2. Now take the polynomial

$$P'(x) := (1-\varepsilon')\frac{P(x+t) + P(-x+t)}{2} + \varepsilon'.$$

It is easy to see that by construction $P'(x)$ is an even polynomial of degree $\mathcal{O}\!\left(\frac{1}{\delta'}\log\!\left(\frac{1}{\varepsilon'}\right)\right)$. Moreover $|P'(x)| \leq 1$ for all $x \in [-1, 1]$ and (3.2) also holds. $\square$

Now we prove our result about uniform singular value amplification, which is a common generalization of the results of Low and Chuang [LC17a, Theorems 2,8].

3.2.7. THEOREM (Uniform singular value amplification). *Let $U, \Pi, \widetilde{\Pi}$ be as in Theorem 2.3.7, let $\gamma > 1$ and let $\delta, \varepsilon \in (0, \frac{1}{2})$. Suppose that $\widetilde{\Pi}U\Pi = W\Sigma V^\dagger = \sum_i \varsigma_i |w_i\rangle\langle v_i|$ is a singular value decomposition. Then there is an $m = \mathcal{O}\big(\frac{\gamma}{\delta}\log\big(\frac{\gamma}{\varepsilon}\big)\big)$ and an efficiently computable $\Phi \in \mathbb{R}^m$ such that*[5]

$$\Big(\langle +| \otimes \widetilde{\Pi}_{\leq \frac{1-\delta}{\gamma}}\Big) U_\Phi \Big(|+\rangle \otimes \Pi_{\leq \frac{1-\delta}{\gamma}}\Big) = \sum_{i\colon \varsigma_i \leq \frac{1-\delta}{\gamma}} \tilde{\varsigma}_i |w_i\rangle\langle v_i|, \text{ where } \left\| \frac{\tilde{\varsigma}_i}{\gamma\varsigma_i} - 1 \right\| \leq \varepsilon. \tag{3.3}$$

Moreover, U_Φ can be implemented using a single ancilla qubit with m uses of U and $U^\dagger$, m uses of $C_\Pi NOT$ and m uses of $C_{\widetilde{\Pi}} NOT$ gates and m single-qubit gates.

Proof:
Let us set in Lemma 3.2.6 $t := \frac{1-\delta/2}{\gamma}$, $\delta' := \frac{\delta}{2\gamma}$ and $\varepsilon' := \frac{\varepsilon}{\gamma}$ in order to get an even polynomial P of degree $\mathcal{O}\big(\frac{\gamma}{\delta}\log\big(\frac{\gamma}{\varepsilon}\big)\big)$ that is an $\frac{\varepsilon}{\gamma}$-approximation of the rectangle function. Let $P_\Re(x) := \gamma \cdot x \cdot P(x)$, which is an odd polynomial of degree $m = \mathcal{O}\big(\frac{\gamma}{\delta}\log\big(\frac{\gamma}{\varepsilon}\big)\big)$. It is easy to see that $P_\Re$ approximates the linear function $\gamma \cdot x$ with ε-multiplicative precision on the domain $\Big[\frac{-1+\delta}{\gamma}, \frac{1-\delta}{\gamma}\Big]$, and observe that $|P_\Re(x)| \leq 1$ for all $x \in [-1, 1]$, thereby it satisfies the requirements of Corollary 2.3.8. We use singular value transformation Corollary 2.3.8 to construct a $\Phi \in \mathbb{R}^m$ such that

$$(\langle +| \otimes \widetilde{\Pi}) U_\Phi (|+\rangle \otimes \Pi) = P_\Re^{(SV)}\Big(\widetilde{\Pi}U\Pi\Big) = \sum_i P_\Re(\sigma_i)|w_i\rangle\langle v_i|$$

which shows that equation (3.3) is satisfied because $\frac{P_\Re(x)}{\gamma \cdot x}$ is ε-close to 1 on the domain $\Big[\frac{-1+\delta}{\gamma}, \frac{1-\delta}{\gamma}\Big]$. We conclude the gate complexity using Lemma 2.3.9. □

Finally, note that if $\|\Sigma\| \leq \frac{1-\delta}{\gamma}$ in the above theorem then we get that

$$\Big\| \gamma\widetilde{\Pi}U\Pi - (\langle +| \otimes \widetilde{\Pi}) U_\Phi (|+\rangle \otimes \Pi) \Big\| \leq \varepsilon,$$

thereby this procedure gives an efficient way to magnify a projected unitary encoding.

3.2.2 Singular value discrimination, quantum walks and the fast OR lemma

First we show how to efficiently implement approximate singular value threshold projectors, which will be the main tool of this section. Unfortunately, singular

[5]Here we implicitly assumed that U_Φ is implemented as in Figure 2.2, with the phase gates as in Figure 2.2b and the $|+\rangle$ ancilla state corresponds to the ancilla qubit in Figure 2.2b.

values that are close to the threshold are hard to handle. Let V be the subspace spanned by singular vectors which have singular value δ-close to the threshold t. Then we can implement an operator for which V is an invariant subspace, and which is ε-close to the desired projector on the orthogonal complement of V.

3.2.8. Theorem (Implementing singular value threshold projectors). *Let $U, \Pi, \widetilde{\Pi}$ be as in Theorem 2.3.7 and let $t, \delta > 0$. Suppose that $\widetilde{\Pi}U\Pi = W\Sigma V^\dagger$ is a singular value decomposition. Then there is an $m = \mathcal{O}\Big(\frac{\log(1/\varepsilon)}{\delta}\Big)$ and a $\Phi \in \mathbb{R}^m$ such that $\|\Pi_{\geq t+\delta}U_\Phi\Pi_{\geq t+\delta} - I \otimes \Pi_{\geq t+\delta}\| \leq \varepsilon$, and*[5] *$\|(\langle +| \otimes \Pi_{\leq t-\delta})U_\Phi(|+\rangle \otimes \Pi_{\leq t-\delta})\| \leq \varepsilon$. Moreover, U_Φ can be implemented using a single ancilla qubit with m uses of U and $U^\dagger$, m uses of $C_\Pi NOT$ and m uses of $C_{\widetilde{\Pi}}NOT$ gates and m single-qubit gates.*

Proof:
By Lemma 3.2.6 we can construct an even polynomial $P_\Re \in \mathbb{R}[x]$ of degree $m = \mathcal{O}\Big(\frac{\log(1/\varepsilon^2)}{\delta}\Big)$ that approximates the rectangle function with $\varepsilon^2/4$ precision on the domain $[-1,1] \setminus (-t-\delta, -t+\delta) \cup (t-\delta, t+\delta)$. By Corollary 2.2.8 we know that there exists a polynomial P of the same degree as $P_\Re$ such that $\Re[P] = P_\Re$, moreover P satisfies the conditions of Corollary 2.2.6. Use singular value transformation Theorem 2.3.7 to construct a $\Phi \in \mathbb{R}^m$ such that $\widetilde{\Pi}U_\Phi\Pi = P^{(SV)}\big(\widetilde{\Pi}U\Pi\big)$ up to error ε and observe that $\|\Pi_{\geq t+\delta}U_\Phi\Pi_{\geq t+\delta} - I \otimes \Pi_{\geq t+\delta}\| \leq \varepsilon$, and $\|(\langle +| \otimes \Pi_{\leq t-\delta})U_\Phi(|+\rangle \otimes \Pi_{\leq t-\delta})\| \leq \varepsilon$. Conclude the gate complexity using Lemma 2.3.9. $\square$

We note that the above complexity can be improved up to quadratically in terms of scaling with δ, when the threshold t is close to 1, see, e.g., Lemma 3.2.12. For the error of the optimal polynomial approximation of the step function see the results of Eremenko and Yuditskii [EY11].

It is also possible to implement eigenvalue threshold projectors with essentially the same complexity, using Theorem 3.4.1 and Corollary 3.2.5. This result has implications to ground state preparation, and directly improves the results of Ge et al. [GTC17].

The singular value discrimination problem is the following: find out whether a given quantum state has singular value at most a or at least b, under the promise that the singular value is not in (a, b). As we indicated above, whenever a and b are $\mathcal{O}(|a-b|)$-close to 1 we can get a quadratic improvement. A simple way to achieve this quadratic improvement is to perform singular value projection on the complementary singular values rather than on the original ones, by replacing the matrix $\widetilde{\Pi}U\Pi$ by the complementary projection $(I - \widetilde{\Pi})U\Pi$.

3.2.9. Theorem (Efficient singular value discrimination). *Let $0 \leq a < b \leq 1$, and let $A = \widetilde{\Pi}U\Pi$ be a projected unitary encoding. Let $|\psi\rangle$ be a given unknown quantum state, with the promise that $|\psi\rangle$ is a right singular vector of A with* *singular value at most a or at least b. Then we can distinguish the two cases with error probability at most ε using singular value transformation of degree*

$$\mathcal{O}\left(\frac{1}{\max[b-a,\sqrt{1-a^2}-\sqrt{1-b^2}]}\log\left(\frac{1}{\varepsilon}\right)\right).$$

Moreover, if $a=0$ or $b=1$, then we can make the error one sided.

Proof:
Let us assume that $b-a\geq\sqrt{1-a^2}-\sqrt{1-b^2}$. First we apply an $\sqrt{\varepsilon}$-approximate singular value projector on $|\psi\rangle$ using Theorem 3.2.8, with $t:=\frac{a+b}{2}$ and $\delta:=\frac{b-a}{2}$, at the end measuring the projector $|+\rangle\langle+|\otimes\Pi$. If we find the state in the image of $|+\rangle\langle+|\otimes\Pi$ we conclude that the singular value is at least b, otherwise we conclude that it is at most a. The correctness and the complexity follow from Theorem 3.2.8. If $a=0$ then we make the error one-sided by using singular vector transformation Theorem 3.2.3 with setting $\delta:=b$, and measuring $\widetilde{\Pi}$ at the end. Similarly as before, if we find the state in the image of $\widetilde{\Pi}$ we conclude that the singular value is at least b, otherwise we conclude that it is 0. The error becomes one-sided because Theorem 3.2.3 uses an odd-degree singular value transformation which always preserves 0 singular values.

The proof of the $b-a<\sqrt{1-a^2}-\sqrt{1-b^2}$ case works analogously by changing $\widetilde{\Pi}$ to $\Pi':=I-\widetilde{\Pi}$ in the proof, which leads to $A':=\Pi'U\Pi$. It is easy to see by Lemma 2.3.2 that $|\psi\rangle$ is a singular vector of A' with singular value at least $\sqrt{1-a^2}$ in the first case or with singular value at most $\sqrt{1-b^2}$ in the second case. Also if $b=1$ we can make the error one-sided since then the corresponding singular value of $|\psi\rangle$ with respect to A' is 0.

Finally note that if $a=0$, then $b-a=b\geq 1-\sqrt{1-b^2}=\sqrt{1-a^2}-\sqrt{1-b^2}$, and if $b=1$, then $b-a=1-a\leq\sqrt{1-a^2}=\sqrt{1-a^2}-\sqrt{1-b^2}$, therefore we covered all cases. $\square$

The above result can also be used when the input state is promised to be a superposition of singular values, with the promised bounds. Also in order to distinguish the two cases with constant success probability it is enough if most of the amplitude is on singular vectors with singular vectors satisfying the promise.

Relationship to quantum walks

Now we show how to quickly derive the quadratic speed-ups of Markov chain based search algorithms using our singular value transformation and discrimination results. Before doing so we introduce some definitions and notation for Markov Chains.

Let $\mathcal{P}\in\mathbb{R}^{n\times n}$ be a time-independent Markov chain on discrete state space X with $|X|=n$, which sends an element $x\in X$ to $y\in X$ with probability p_{xy}. Thus $\mathcal{P}$ is a row-stochastic matrix. We say that $\mathcal{P}$ is *ergodic* if for a large enough $t\in\mathbb{N}$

all elements of $\mathcal{P}^t$ are non-zero. For an ergodic $\mathcal{P}$ there exists a unique stationary distribution π such that $\pi\mathcal{P} = \pi$, and we define the *time-reversed* Markov chain as $\mathcal{P}^* := \mathrm{diag}(\pi)^{-1} \cdot \mathcal{P}^T \cdot \mathrm{diag}(\pi)$. We say that $\mathcal{P}$ is *reversible* if it is ergodic and $\mathcal{P}^* = \mathcal{P}$. For an ergodic Markov chain $\mathcal{P}$ we define the discriminant matrix $D(\mathcal{P})$ such that its xy entry is $\sqrt{p_{xy}p^*_{yx}}$, where p^*_{yx} stands for entries of the time-reversed chain. It is easy to see that

$$D(\mathcal{P}) = \mathrm{diag}(\pi)^{\frac{1}{2}} \cdot \mathcal{P} \cdot \mathrm{diag}(\pi)^{-\frac{1}{2}}.$$

This form has several important consequences. First of all the spectra of $\mathcal{P}$ and $D(\mathcal{P})$ coincide, moreover the vector $\sqrt{\pi}$, that we get from π by taking the square root element-wise, is a left eigenvector of $D(\mathcal{P})$ with eigenvalue 1. Also from the definition $\sqrt{p_{xy}p^*_{yx}}$ of the xy entry it follows that for reversible Markov chains, $D(\mathcal{P})$ is a symmetric matrix, therefore its singular values and eigenvalues coincide after taking their absolute value.

In the literature [Szeg04, MNRS11, KMOR16] quantum walk based search methods are usually analyzed with the help of this discriminant matrix. Here we directly use the discriminant matrix as opposed to the associated quantum walk, significantly simplifying the analysis. Before deriving our versions of the Markov chain speed-up results we introduce some definitions regarding sets of marked elements.

For a set of marked elements $M \subseteq X$, we denote by $D_M(\mathcal{P})$ the matrix that we get after setting to zero the rows and columns of $D(\mathcal{P})$ corresponding to the marked elements. For an ergodic Markov chain $\mathcal{P}$ we define the *hitting time* $\mathrm{HT}(\mathcal{P}, M)$ as the expected number of steps of the Markov chain before reaching the first marked element, if started from the stationary[6] distribution π. We denote the probability that an element is marked in the stationary distribution by $p_M := \sum_{x\in M} \pi_x$. Now we invoke some results about the connection between the hitting time and the discriminant matrix, which are proven for example in [KMOR16, Proposition 2] and [Gil14, Lemma 10].

3.2.10. Lemma (Hitting times and discriminant matrix). *Let $\mathcal{P}$ be a reversible Markov chain and M a set of marked elements. Let (v_i, λ_i) be the eigenvector-eigenvalue pairs of $D_M(\mathcal{P})$, then*

$$\mathrm{HT}(\mathcal{P}, M) = \sum_{i=1}^{n} \frac{|\langle v_i, \sqrt{\pi}\rangle|^2}{1-\lambda_i} - p_M, \quad \textit{and} \quad \sum_{i=1}^{n} \frac{|\langle v_i, \sqrt{\pi}\rangle|^2}{1-|\lambda_i|} \leq 2\sum_{i=1}^{n} \frac{|\langle v_i, \sqrt{\pi}\rangle|^2}{1-\lambda_i} \tag{3.4}$$

The following result shows how the presence of marked elements can be detected quadratically faster using singular value discrimination compared to using the corresponding classical Markov chain. A slightly less general version of this result was proven by Szegedy [Szeg04].

[6]Here we follow the convention of Szegedy [Szeg04, Equation (15)], and define hitting time without conditioning on the stationary distribution to unmarked vertices.

3.2.11. Corollary (Detecting marked elements in a reversible Markov chain). *Let $\mathcal{P}$ be a reversible Markov chain, and $M \subseteq X$ a set of marked elements. Let U be a unitary and $\tilde{\Pi}, \Pi$ orthogonal projectors in* $\mathrm{End}(\mathcal{H})$*. Suppose that $B, \tilde{B}$ are orthonormal bases of $\mathcal{H}$, such that representing the matrix of $\tilde{\Pi}U\Pi$ in the bases $B \to \tilde{B}$ we have that*

$$\tilde{\Pi}U\Pi = \begin{bmatrix} D_M(\mathcal{P}) & 0 \\ 0 & . \end{bmatrix}.$$

Suppose that we are given a copy of $|\sqrt{\pi}\rangle := \sum_{x\in X}\sqrt{\pi_x}|x\rangle$, where $(|x\rangle\colon x \in X)$ are the first n basis elements in B. Then we can distinguish with constant one-sided error the cases $\mathrm{HT}(\mathcal{P}, M) \leq K$ *and $M = \emptyset$ (i.e.,* $\mathrm{HT}(\mathcal{P}, M) = \infty$*) with singular value transformation of degree $\mathcal{O}\big(\sqrt{K+1}\big)$.*

Proof:
Suppose that $M \neq \emptyset$. Let $\{(|v_i\rangle, \lambda_i)\colon i \in [n]\}$ be the eigenvector and eigenvalue pairs of the $D_M(\mathcal{P})$ block of $\tilde{\Pi}U\Pi$. By Lemma 3.2.10 we have that

$$\sum_{i=1}^{n} \frac{|\langle v_i|\sqrt{\pi}\rangle|^2}{1-|\lambda_i|} \leq 2\mathrm{HT}(\mathcal{P}, M) + 2p_M \leq 2(K+1).$$

By Markov's inequality we have that

$$\sum_{i\colon |\lambda_i| \geq 1-\frac{1}{12(K+1)}} |\langle v_i|\sqrt{\pi}\rangle|^2 \leq \frac{1}{6},$$

and so $\left\|\Pi_{\leq 1-\frac{1}{12(K+1)}}|\sqrt{\pi}\rangle\right\|^2 \geq \frac{5}{6}$. On the other hand if $M = \emptyset$, then $D_M(\mathcal{P}) = D(\mathcal{P})$, and $\|D(\mathcal{P})|\sqrt{\pi}\rangle\| = 1$. Therefore we can apply our singular value discrimination result Theorem 3.2.9 to distinguish the two cases $M = \emptyset$ and $\mathrm{HT}(\mathcal{P}, M) \leq K$ using singular value transformation of degree $\mathcal{O}\big(\sqrt{K+1}\big)$. □

The above result shows how to detect the presence of marked elements quadratically faster than the classical hitting time. In practice one usually also wants to *find* a marked element, and quantum walks are also good at solving this problem. In order to show the connection to the literature of quantum walk based search algorithms we define some additional notation.

First we define the standard implementation procedures for Markov chains together with their associated costs following Magniez et al. [MNRS11]. We slightly generalize the usual approach fitting our singular value transformation framework. Let us fix an orthogonal basis B, such that the first n elements of B are labeled by $|x\rangle_d\colon x \in X$. The states $|x\rangle_d$ need not correspond to the first $|X|$ computational basis states, but can have some data structure attached helping the updates, as suggested by the subscript d. We define the main operations with their matrices represented in the basis B, and the associated costs as follows:

U : Update cost U. The cost of implementing $\mathrm{C}_\Pi\mathrm{NOT}$ and U gates such that

$$\Pi U \Pi = \begin{bmatrix} D(\mathcal{P}) & 0 \\ 0 & . \end{bmatrix}. \tag{3.5}$$

C : Checking cost C. The cost of implementing a $\mathrm{C}_{\Pi_M}\mathrm{NOT}$ gate such that for all $x \in M$: $\Pi_M|x\rangle_d = |x\rangle_d$ and for all $x \in X \setminus M$: $\Pi_M|x\rangle_d = 0$. This implies that

$$(I - \Pi_M)\Pi U \Pi(I - \Pi_M) = \begin{bmatrix} D_M(\mathcal{P}) & 0 \\ 0 & . \end{bmatrix}.$$

S : Setup cost S. The cost of preparing the stationary state in basis B: $|\sqrt{\pi}\rangle := \sum_{x\in X}\sqrt{\pi_x}|x\rangle_d$.

First we would like to describe how these operators are usually implemented in the literature. Usually $\mathcal{P}$ is represented using basis elements $|x\rangle_d = |0\rangle|x\rangle|d_x\rangle$, where the $|d_x\rangle$ register stores some data associated with the vertex x, which enables efficient implementation of the update procedure. The unitary U is usually implemented using a product of state-preparation unitaries $U = U_L^\dagger U_R$:

$$U_R : |0\rangle|x\rangle|d_x\rangle \to \sum_{y\in[n]} \sqrt{p_{xy}}|x\rangle|y\rangle|d_{xy}\rangle \qquad \forall x \in X \tag{3.6}$$

$$U_L : |0\rangle|y\rangle|d_y\rangle \to \sum_{x\in[n]} \sqrt{p_{yx}^*}|x\rangle|y\rangle|d_{xy}\rangle \qquad \forall y \in X \tag{3.7}$$

We assume for simplicity that $0 \notin X$, resulting in a helpful "free" label, and also let us assume that the first register is $(n+1)$-dimensional and the second register is n-dimensional. If the third register is one-dimensional (i.e., we can just trivially omit it), then by equations (3.6)-(3.7) we get that

$$(|0\rangle\langle 0| \otimes I)U(|0\rangle\langle 0| \otimes I) = \begin{bmatrix} D(\mathcal{P}) & 0 \\ 0 & 0 \end{bmatrix}.$$

If the data structure register is non-trivial, we can still conclude that

$$(|0\rangle\langle 0| \otimes I)U(|0\rangle\langle 0| \otimes I) = \begin{bmatrix} D(\mathcal{P}) & . \\ . & . \end{bmatrix}.$$

However, we need a slightly stronger assumption about U. We assume that U_L, U_R are implemented such that the block-matrices next to $D(\mathcal{P})$ are 0 as in (3.5). This is implicitly assumed[7] in [MNRS11], and a sufficient condition is presented in the work of Childs et al. [CJKM13, Footnote 1].

Before solving the above problem, we invoke a useful polynomial approximation result due to Dolph [Dol46]. This gives an explicit construction of the optimal polynomial of degree n, that stays bounded by ε in absolute value on the largest possible interval of the form $[-\lambda, \lambda]$, while taking value $(\pm 1)^n$ at $x = \pm 1$.

[7]This assumption is necessary for the correctness of the analysis in [MNRS11], however it is not explicitly stated.

3.2.12. Lemma. *For all $\varepsilon \in (0,1]$ and $n \in \mathbb{N}$ we have that*[8]

$$\underset{P\in\mathbb{R}[x]}{\operatorname{argmax}}\Big(\max\left\{\lambda\colon \|P(x)\|_{[-\lambda,\lambda]} \leq \varepsilon\right\}\Big) = \varepsilon T_n(xT_{1/n}(1/\varepsilon)),$$

where argmax *is over all real degree-n polynomials satisfying $\|P\|_{[-1,1]} \leq 1$, and $P(\pm 1) = (\pm 1)^n$. Moreover, for any $\delta \in (0,1)$ for some $n = \mathcal{O}\Big(\frac{1}{\sqrt{\delta}}\log(\frac{1}{\varepsilon})\Big)$ we have that*

$$\left\|\varepsilon T_n(xT_{1/n}(1/\epsilon))\right\|_{[-1+\delta,1-\delta]} \leq \varepsilon.$$

Remarkably, the phase sequence required to implement this polynomial using quantum signal processing is expressed in closed form in the work of Yoder et al. [YLC14].

3.2.13. Theorem (Speed-up for finding marked elements of a Markov chain). *Let $\mathcal{P}$ be a reversible Markov chain, such that the singular value gap*[9] *of $D(\mathcal{P})$ is at least δ, and the set of marked elements M is such that $p_M \geq \varepsilon$. Then we can find a marked element with high probability in complexity of order $\mathsf{S} + \frac{1}{\sqrt{\varepsilon}}\Big(\mathsf{C} + \sqrt{\frac{1}{\delta}}\log(\frac{1}{\varepsilon})\mathsf{U}\Big)$.*

Proof:
First we apply singular value transform on $\Pi U \Pi$ using an ε-approximation of the zero-function given by Lemma 3.2.12 in order to get U_Φ with all $\neq 1$ singular values of $D(\mathcal{P})$ shrunk below a level of $\mathcal{O}(\varepsilon)$. Then the top-left block of $\Pi U_\Phi \Pi$ is $\mathcal{O}(\varepsilon)$-close to $|\sqrt{\pi}\rangle\langle\sqrt{\pi}|$, and the implementation of U_Φ has complexity $\mathcal{O}\Big(\sqrt{\frac{1}{\delta}}\log(\frac{1}{\varepsilon})\mathsf{U}\Big)$. We pretend that the top-left block is $|\sqrt{\pi}\rangle\langle\sqrt{\pi}|$, in which case we would have that $\Pi_M \Pi U_\Phi \Pi = \sqrt{p_M}|\sqrt{\pi_M}\rangle\langle\sqrt{\pi}|$, where $|\sqrt{\pi_M}\rangle := \frac{\sum_{x\in M}\sqrt{\pi_x}|x\rangle_d}{\sqrt{p_M}}$. Then we apply singular vector transform to get a constant approximation of $|\sqrt{\pi_M}\rangle\langle\sqrt{\pi}|$ in the top-left block, and apply it to the state $|\sqrt{\pi}\rangle$ in order to find a marked element with high probability. Finally, we use the robustness of singular value transformation (Lemma 2.4.3) to show that we can indeed dismiss the $\mathcal{O}(\varepsilon)$ discrepancy between $\Pi U_\Phi \Pi$ and $|\sqrt{\pi}\rangle\langle\sqrt{\pi}|$. $\square$

Note that the above algorithm is simpler and more efficient than the phase estimation based algorithm of Magniez et al. [MNRS11]. However, Magniez et al. showed how to remove the $\log(\frac{1}{\varepsilon})$ factor completely using a more involved

[8]The standard generalization of Chebyshev polynomials to non-integer degree y is $T_y(x) \equiv \cosh(y\operatorname{arccosh}(x)) \equiv \cos(y\arccos(x))$.

[9]We define the singular value gap as the difference between the two largest singular values. For a reversible Markov chain the singular values of $D(\mathcal{P})$ are the same as the absolute values of the eigenvalues of $\mathcal{P}$. Note however, that this is not strictly necessary, we could work with the eigenvalues too using eigenvalue transformation results, such as Theorem 3.4.1.

procedure, which fine-tunes the precision-level of the utilized approximate reflections. Their approach can also be directly applied here, plugging our approximate implementation of $|\sqrt{\pi}\rangle\langle\sqrt{\pi}|$ into their Lemma 2 [MNRS11].

Finally, we note that Krovi et al. [KMOR16] showed that a unique marked element can be found quadratically faster than the classical hitting time[10] by using an interpolated quantum walk. However, for a long time it was an open question whether in the presence of multiple marked elements the quadratic advantage can be retained. Very recently Ambainis et al. [AGJK19] gave a positive answer to this question; their algorithm combines the interpolation idea of Krovi et al. [KMOR16] with the fast-forwarding technique of Apers and Sarlette [AS18], described in the introduction of this chapter. In some cases this result gives a better complexity than Theorem 3.2.13

Fast QMA amplification and fast quantum OR lemma

The improved version of the Marriott-Watrous [MW05] QMA amplification procedure due Nagaj et al. [NWZ09] can be seen as a direct corollary of our singular value discrimination results. In order to state the result we give the definition of the complexity class QMA.

3.2.14. Definition (The complexity class QMA). Let $L \subseteq \{0,1\}^*$ be a language. The language L belongs to the class QMA if there exists a uniform family of quantum verifier circuits $V(|x|)$ working on[11] $|x| + n = \mathrm{poly}(|x|)$ qubits using $m = \mathrm{poly}(|x|)$ ancillae, and numbers $0 \leq b_{|x|} < a_{|x|} \leq 1$ satisfying $\frac{1}{a_{|x|}-b_{|x|}} = \mathcal{O}(\mathrm{poly}(|x|))$, such that for all

$x \in L$: there exists an n-qubit witness $|\psi\rangle$ such that upon measuring the state $V|x\rangle|0\rangle^{\otimes m}|\psi\rangle$ the probability of finding the $(|x|+1)$st qubit in state $|1\rangle$ has probability at least $a_{|x|}$,

$x \notin L$: for any n-qubit state $|\phi\rangle$ upon measuring the state $V|x\rangle|0\rangle^{\otimes m}|\phi\rangle$ the probability of finding the $(|x|+1)$st qubit in state $|1\rangle$ has probability at most $b_{|x|}$.

Note that unlike in the classical setting, where a witness can be used arbitrarily many times, in the quantum case the verification procedure might destroy the witness. If one would collect classical statistics, and the witness would be destroyed at each trial, then in order to boost the success probability to $1-\varepsilon$ one would require about $\log(1/\varepsilon)/(a-b)^2$ copies of the witness, and a proportionally large number of computational steps.

[10]One can show that $p_M = \Omega\left(\frac{1}{\mathrm{HT}(\mathcal{P},M)}\right)$, therefore using amplitude amplification one can find a marked element quadratically faster, however with a large S cost. The appeal of the quantum walk algorithms is that they use the setup procedure only very few times.

[11]By $|x|$ here we denote the length of the string x, rather than its Hamming weight.

However, Marriott and Watrous [MW05] showed that a single witness suffices, and Nagaj et al. [NWZ09] additionally showed how to a improve the computational complexity of the gap-amplification. Now we are ready to reprove this. (For readability we removed the $|x|$ subscript from the success probabilities.)

3.2.15. Theorem (Fast QMA amplification). *Suppose that $L \in \mathsf{QMA}$ as in Definition 3.2.14. We can modify the verifier circuit V such that the acceptance probability thresholds become $a' := 1-\varepsilon$ and $b' := \varepsilon$ using singular value transformation of degree $\mathcal{O}\Big(\frac{1}{\max[\sqrt{a}-\sqrt{b},\sqrt{1-b}-\sqrt{1-a}]}\log\big(\frac{1}{\varepsilon}\big)\Big)$.*

Proof:
First note that we can assume without loss of generality that the first register containing $|x\rangle$ does not get modified by V. Otherwise we can just copy $|x\rangle$ first to an ancilla register and then run V using this copy of $|x\rangle$. Observe that by Definition 3.2.14 for all $x \in L$ we have that

$$\left\|(\langle x| \otimes |1\rangle\langle 1| \otimes I_{n+m-1})V\big(|x\rangle \otimes |0\rangle\langle 0|^{\otimes m} \otimes I_n\big)\right\| \geq \sqrt{a},$$

and for all $x \notin L$ we have that

$$\left\|(\langle x| \otimes |1\rangle\langle 1| \otimes I_{n+m-1})V\big(|x\rangle \otimes |0\rangle\langle 0|^{\otimes m} \otimes I_n\big)\right\| \leq \sqrt{b}.$$

After applying a singular value discrimination circuit for discriminating singular values below $\sqrt{b}$ and above $\sqrt{a}$ we get a circuit that in the former case accepts some witness $|\psi\rangle$ with probability at least $1-\varepsilon$ and in the latter case rejects every state $|\phi\rangle$ with probability at least $1-\varepsilon$. $\square$

Finally, we turn to discussing the quantum OR lemma, which has a somewhat similar flavor to the above QMA amplification problem. The problem is the following: we are given a collection of m projective measurements, and a mixed quantum state ρ, and would like to compute the OR function of the measurement outcomes, under the promise that for each projective measurement the measurement probability is either very close to 1 or very close to 0. The first question is whether we can solve this problem using only a constant number of copies of ρ. The quantum OR lemma of Harrow et al. [HLM17] shows that it is indeed possible, and the fast quantum OR lemma of Brandão et al. [BKL+17b] improves the procedure's computational complexity.

Now we show how to quickly derive a slightly improved version of the fast quantum OR lemma. We use the main ideas of the proof of the original quantum OR lemma in a way similar to the approach of Brandão et al.

3.2.16. Theorem (Fast quantum OR lemma). *Let $m \in \mathbb{N}$, let $\Pi_i\colon i \in [m]$ be orthogonal projectors and let $\eta, \nu \in (0, \frac{1}{2}]$. Suppose we are given one copy of a quantum state ρ with the promise that either* *(i) there exists some $i \in [m]$ such $\mathrm{Tr}[\rho\Pi_i] \geq 1-\eta$, or*

(ii) $\frac{1}{m}\sum_{j=1}^{m} \mathrm{Tr}[\rho\Pi_j] \leq \nu$.

Suppose that we have access[12] to an operator V such that $(\langle i| \otimes I)V(|i\rangle \otimes I) = \mathrm{C}_{\Pi_i}NOT$ for all $i \in [m]$. Then for all $\varepsilon \in (0, \frac{1}{2}]$ we can construct an algorithm which, in case (i) accepts ρ with probability at least $\frac{(1-\eta)^2}{4} - \varepsilon$, and in case (ii) it accepts ρ with probability at most $5m\nu + \varepsilon$. Moreover, the algorithm uses V and its inverse a total number of $\mathcal{O}\big(\sqrt{m}\log\big(\frac{1}{\varepsilon}\big)\big)$ times and uses $\mathcal{O}\big(\sqrt{m}\log(m)\log\big(\frac{1}{\varepsilon}\big)\big)$ other gates and $\mathcal{O}(\log(m))$ ancilla qubits.

Proof:
Let us define $A := \frac{1}{m}\sum_{i=1}^{m}(I - \Pi_i)$. First observe that

$$I - \Pi_i = (\langle 0| \otimes I)\mathrm{C}_{\Pi_i}\mathrm{NOT}(|0\rangle \otimes I).$$

Let $a := \lceil \log_2(m+1) \rceil + 1$ and let U be a unitary that implements the map $|0\rangle^{\otimes(a-1)} \to \frac{1}{\sqrt{m}}\sum_{i=1}^{m}|i\rangle$, and let us define $\tilde{V} := \big(U^\dagger \otimes I\big)V(U \otimes I)$ and $\Pi := |0\rangle\langle 0|^{\otimes a} \otimes I$. Then it is easy to see that $A = \Pi\tilde{V}\Pi$.

Now let $\lambda := \frac{1-\eta}{2m}$, in case (i) Harrow et al. [HLM17, Corollary 11] proved that

$$\mathrm{Tr}[\rho\Pi_{\leq 1-\lambda}] \geq (1-\eta)^2/4. \tag{3.8}$$

On the other hand in case (ii) we have that $\mathrm{Tr}[\rho A] \geq 1 - \nu$. Using Markov's inequality we get

$$\mathrm{Tr}\Big[\rho\Pi_{\leq 1-\frac{4}{5}\lambda}\Big] \leq \frac{\nu}{\frac{4}{5}\lambda} = \frac{5m\nu}{2(1-\eta)} \leq 5m\nu. \tag{3.9}$$

Finally, we apply ε-precise singular value discrimination on ρ with $a := 1-\lambda$ and $b := 1-\frac{4}{5}\lambda$. The correctness follows from (3.8)-(3.9) and Theorem 3.2.9. The complexity statement follows from Theorem 3.2.9, Lemma 2.3.9 and the fact that U can be implemented using $\mathcal{O}(\log(m))$ one- and two-qubit gates. □

3.2.3 Improving the HHL algorithm via a direct implementation of the Moore-Penrose pseudoinverse

The famous quantum algorithm of Harrow, Hassidim, and Lloyd [HHL09] solves the system of linear equations $Ax = b$ in a very quantum sense. Given suitable access to the matrix A, and the ability to prepare a quantum state $|b\rangle$, it prepares a

[12] Brandão et al. [BKL+17b] assume access to controlled reflection operators instead of $\mathrm{C}_{\Pi_i}\mathrm{NOT}$ gates, but it is easy to see that these gates are the same up to conjugation by a Hadamard gate on the control qubit, as we noted after Definition 2.1.1.

quantum state proportional to $|x\rangle$. The most common assumption for the matrix A is that it is s-sparse and that the entries of the matrix are efficiently computable, as well as the locations of the non-zero entries. Then the HHL algorithm runs in time that depends polynomially on the sparsity and the condition number of A, and polylogarithmically on the dimension and the desired precision. When A is not invertible, then one can still prepare a state proportional to the least-square solution $A^+|b\rangle$, with appropriately defining a substitute for the condition number. Now we derive a variant of this latter version, under the assumption that A is given by a projected unitary encoding.

Suppose $\widetilde{\Pi}U\Pi = A$ and $A = W\Sigma V^\dagger$ is a singular value decomposition. Then the pseudoinverse of A is $A^+ = V\Sigma^+W^\dagger$, where Σ^+ contains the inverses of the non-zero elements of Σ^T.

It is pretty straightforward to implement the pseudoinverse using singular value transformation. Suppose that all non-zero singular values are at least δ. Let $P_\Re$ be an odd real polynomial that ε-approximates the function $\delta/(2x)$ on the domain $[-1,1]\setminus(-\delta,\delta)$, then $P_\Re^{(SV)}(A^\dagger) = \Pi U_\Phi^\dagger \widetilde{\Pi}$ ε-approximates $\frac{\delta}{2}A^+$. The only thing remaining is to construct a relatively low-degree odd polynomial $P_\Re$ that achieves the desired approximation, and which is bounded by 1 in absolute value on $[-1,1]$, in order to be able to apply Corollary 2.3.8. Childs et al. [CKS17, Lemmas 17-19] constructed a useful polynomial for improving the HHL algorithm, which we can use after some adjustments.

3.2.17. Lemma. (Polynomial approximations of $1/x$, [CKS17, Lemmas 17-19]) *Let $\kappa > 1$ and $\varepsilon \in (0,\frac{1}{2})$. For $b = \lceil \kappa^2 \log(\kappa/\varepsilon)\rceil$ the odd function*

$$f(x) := \frac{1-(1-x^2)^b}{x}$$

is ε-close to $1/x$ on the domain $[-1,1]\setminus(-\frac{1}{\kappa},\frac{1}{\kappa})$. Let $J := \left\lceil \sqrt{b\log(4b/\varepsilon)}\right\rceil$, then the $\mathcal{O}\big(\kappa\log(\frac{\kappa}{\varepsilon})\big)$-degree odd real polynomial

$$P(x) := 4\sum_{j=0}^{J}(-1)^j\left[\frac{\sum_{i=j+1}^{b}\binom{2b}{b+i}}{2^{2b}}\right]T_{2j+1}(x)$$

is ε-close to $f(x)$ on the interval $[-1,1]$, moreover $|P(x)| \leq 4J = \mathcal{O}\big(\kappa\log(\frac{\kappa}{\varepsilon})\big)$ on this interval.

3.2.18. Theorem (Implementing the Moore-Penrose pseudoinverse). *Let U,Π, $\widetilde{\Pi}$ be as in Theorem 2.3.7 and let $0 < \varepsilon \leq \delta \leq \frac{1}{2}$. Suppose that $A = \widetilde{\Pi}U\Pi = W\Sigma V^\dagger$ is a singular value decomposition. Let $\Pi_{0,\geq\delta} := \Pi_{=0} + \Pi_{\geq\delta}$ and similarly $\widetilde{\Pi}_{0,\geq\delta} := \widetilde{\Pi}_{=0} + \widetilde{\Pi}_{\geq\delta}$. Then there is an $m = \mathcal{O}\big(\frac{1}{\delta}\log(\frac{1}{\varepsilon})\big)$ and an efficiently computable $\Phi \in \mathbb{R}^m$ such that*[5]

$$\left\|\Big(\langle+|\otimes\Pi_{0,\geq\delta}\Big)U_\Phi\Big(|+\rangle\otimes\widetilde{\Pi}_{0,\geq\delta}\Big) - \Pi_{0,\geq\delta}\Big(\frac{\delta}{2}\cdot A^+\Big)\widetilde{\Pi}_{0,\geq\delta}\right\| \leq \varepsilon. \tag{3.10}$$

Moreover, U_Φ can be implemented using a single ancilla qubit with m uses of U and $U^\dagger$, m uses of $C_\Pi NOT$ and m uses of $C_{\widetilde{\Pi}}NOT$ gates and m single-qubit gates.

Proof:
Using Lemma 3.2.17 we can construct an odd polynomial $P(x)$ having degree $\mathcal{O}(\log(1/\varepsilon)/\delta)$ that $\frac{\varepsilon}{3}$-approximates the function $\frac{\delta}{2x}$ on the domain $[-1,1]\setminus(-\frac{\delta}{2},\frac{\delta}{2})$, and is less than 1 on this domain. Let us define $P_{\max} := \max_{x\in[-1,1]}|P(x)|$ and observe that $P_{\max} = \mathcal{O}\big(\log(\frac{1}{\varepsilon})\big)$. Let us also construct an even polynomial P' of degree $\mathcal{O}(\log(1/\varepsilon)/\delta)$ using Lemma 3.2.6 setting $t := \frac{3}{4}\delta$, $\delta' := \frac{\delta}{4}$ and $\varepsilon' := \min\left(\frac{\varepsilon}{3},\frac{1}{P_{\max}}\right)$ that ε'-approximates the rectangle function. Finally let $P_\Re := P\cdot(1-P')$, which is an odd real polynomial of degree $m = \mathcal{O}(\log(1/\varepsilon)/\delta)$. It is easy to see that $P_\Re$ ε-approximates $\frac{\delta}{2x}$ on the domain $[-1,1]\setminus(-\frac{\delta}{2},\frac{\delta}{2})$, moreover $P_\Re$ is bounded by 1 in absolute value on $[-1,1]$. Finally, we apply real singular value transformation on $A^\dagger = \Pi U^\dagger\widetilde{\Pi}$ using the polynomial $P_\Re$ by Corollary 2.3.8, and conclude the gate complexity using Lemma 2.3.9. $\square$

The $\varepsilon \leq \delta$ assumption in the above statement is quite natural, but it is not necessary, and can be removed by using our general polynomial approximation results, see Corollary 3.4.16.

3.2.4 Applications in quantum machine learning

The ability to transform singular values is central to the operation of many popular machine learning methods. Quantum machine learning methods such as quantum support vector machines [RML14], principal component analysis [LMR14, WK17], regression [HHL09, WBL12, CKS17, CGJ19], slow feature analysis [KL18], and Gibbs-sampling [CS17, vAGGdW17] all hinge upon this idea. These results are among the most celebrated in quantum machine learning, showing that singular value transformation has substantial impact on this field of quantum computing.

Many quantum algorithms for basic machine learning problems, such as ordinary least squares, weighted least squares, generalized least squares, were studied in a series of works [HHL09, WBL12, CKS17, CGJ19]. We do not examine these problems case-by-case, but point out that they can all be reduced to implementing the Moore-Penrose pseudoinverse and matrix multiplication, therefore they can be straightforwardly implemented by Theorem 3.2.18 and Lemma 3.3.10 (to be discussed in Subsection 3.3.4) within our framework.

We demonstrate how to apply our singular value transformation techniques to quantum machine learning by developing a new quantum algorithm for principal component regression. This machine learning algorithm is closely related to principal component analysis (PCA), which is a tool that is commonly used to reduce the effective dimension of a model by excising irrelevant features from it. The PCA method is quite intuitive, it simply involves computing the covariance matrix for a data set and then diagonalizing it. The eigenvectors of the covariance matrix then represent the independent directions of least or greatest variation in the data. Dimension reduction can be achieved by culling any components that have negligibly small variation over them. This technique has many applications ranging from anomaly detection to quantitative finance. Quantum algorithms are known for this task and can lead to substantial speed-ups under appropriate assumptions about the data and the oracles used to provide it to the algorithm [LMR14, KLL$^+$17].

Principal component regression is identical in spirit to principal component analysis. Rather than simply truncating small eigenvalues of the covariance matrix for a data set, principal component regression aims to approximately reconstruct a target vector on a domain/range that is spanned by the right or left singular vectors of the data set with large singular values. A least-squares type estimation of the target vector within this subspace of the data can be found by performing a singular vector transformation. This can provide a more flexible and powerful approach to dimensionality reduction than ordinary principal component analysis permits.

The problem of principal component regression can be formally stated as follows [FMMS16]: given a matrix $A \in \mathbb{R}^{n\times d}$, a vector $b \in \mathbb{R}^n$ and a threshold value $0 < \varsigma$, find $x \in \mathbb{R}^d$ such that

$$x = \mathrm{argmin}_{x\in\mathbb{R}^d}\left\|\widetilde{\Pi}_{\geq\varsigma}A\Pi_{\geq\varsigma}x - b\right\|, \tag{3.11}$$

where $\widetilde{\Pi}_{\geq\varsigma}, \Pi_{\geq\varsigma}$ denote left and right singular value threshold projectors. A closed-form expression for the optimal solution of (3.11) is given by $x = \Pi_{\geq\varsigma}A^+\widetilde{\Pi}_{\geq\varsigma}b = A^+\widetilde{\Pi}_{\geq\varsigma}b$.

As the following corollary shows, our singular value transformation techniques give rise to an efficient quantum algorithm for implementing $\Pi_{\geq\varsigma}A^+\widetilde{\Pi}_{\geq\varsigma}$, and thus principal component regression.

3.2.19. Corollary (Implementing the threshold pseudoinverse). *Let U, Π, $\widetilde{\Pi}$ form a projected unitary encoding of the matrix A, and let $\varepsilon, \delta \in (0, \frac{1}{2}]$ and $0 < \varsigma < 1$. Suppose that $A = \widetilde{\Pi}U\Pi = W\Sigma V^\dagger$ is a singular value decomposition of the projected unitary encoding of A. Then there is an $m = \mathcal{O}\left(\frac{1}{\delta}\log(\frac{1}{\varepsilon})\right)$ and an efficiently computable $\Phi \in \mathbb{R}^m$ such that*[5]

$$\left\|\Big(\langle+|\otimes\big(\Pi - \Pi_{[\varsigma-\delta,\varsigma+\delta]}\big)\Big)U_\Phi\Big(|+\rangle\otimes\big(\widetilde{\Pi} - \widetilde{\Pi}_{[\varsigma-\delta,\varsigma+\delta]}\big)\Big) - \Pi_{\geq\varsigma}\Big(\frac{\varsigma}{2}A^+\Big)\widetilde{\Pi}_{\geq\varsigma}\right\| \leq \varepsilon. \tag{3.12}$$

Moreover, U_Φ can be implemented using a single ancilla qubit with m uses of U and $U^\dagger$, m uses of $C_\Pi NOT$ and m uses of $C_{\widetilde{\Pi}}NOT$ gates and m single-qubit gates.

Proof:
We can implement this operator by first applying a singular value threshold projector $\widetilde{\Pi}_{\geq\varsigma}$ according to Theorem 3.2.8, followed by performing the Moore-Penrose pseudoinverse A^+ as in Theorem 3.2.18.

Implementing these two operations separately is actually suboptimal. In order to get the stated result we simply take the polynomials used for singular value transformation in Theorem 3.2.8 and Theorem 3.2.18, then take their product and implement singular value transformation according to the product polynomial. The complexity statement can be proven similarly to the proofs of Theorem 3.2.8 and Theorem 3.2.18. $\square$

Given a unitary preparing a quantum state $|b\rangle$ we can approximately solve principal component regression by applying an approximation of $\Pi_{\geq\varsigma}\left(\frac{\varsigma}{2}A^+\right)\widetilde{\Pi}_{\geq\varsigma}$ to $|b\rangle$, and then applying amplitude amplification to get $|x\rangle$. Strictly speaking, in order for this to work as required by (3.11), we would need to have the promise that $|b\rangle$ does not have an overlap with left singular vectors that have eigenvalues in $[\varsigma-\delta,\varsigma+\delta]$, while it does have a non-negligible overlap with left singular vectors having singular value $>\varsigma+\delta$. In fact, due to the nature of singular value transformation, for a left singular vector $|w_j\rangle$ with singular value $\varsigma_j\in[\varsigma-\delta,\varsigma+\delta]$ the procedure still performs a meaningful operation: it maps $|w_j\rangle\rightarrow f(\varsigma_j)|v_j\rangle$, such that $f(\varsigma_j)\in[-1,1]$. It is plausible that the transition behavior on $[\varsigma-\delta,\varsigma+\delta]$ would in practice not significantly degrade the performance of typical machine learning applications, therefore the promise of not having singular values in $[\varsigma-\delta,\varsigma+\delta]$ is probably not crucial. Also note that an essentially quadratic improvement to the runtime of the above procedure can be achieved using variable-time amplitude amplification techniques [Amb12, CKS17, CGJ19].

Finally, we briefly discuss a recently developed quantum machine learning algorithm which is significantly more complex then the previous algorithm, but can still be easily fitted to our framework. Kerenidis and Luongo recently proposed a quantum algorithm for slow feature analysis [KL18]. The main ingredient of their algorithm is to apply a threshold projection on some input state, i.e., to project onto the subspace spanned by the singular vectors of a matrix with singular values smaller than a certain threshold. Their algorithm is based on singular value estimation, whereas our Theorem 3.2.8 approaches the same problem in a more direct way, by transforming the singular values according to a threshold function.

In the first step of the quantum algorithm of Kerenidis and Luongo, the task is to implement $Y:=V\Sigma^{-1}V^\dagger$ for a given input matrix $X=W\Sigma V^\dagger$. In our framework this can be performed analogously to Theorem 3.2.18 using singular value transformation; the only difference is that one needs to use an even polynomial approximation of $\frac{1}{x}$, for example given by Corollary 3.4.13. In the next step, one needs to implement singular value threshold projection using the matrix $\dot{X}Y$

for a given “derivative” matrix $\dot{X}$. Taking the product[13] of the two matrices can be implemented using Lemma 3.3.10, after which we can use our Theorem 3.2.8 to implement singular value threshold projection.

3.2.5 Singular value estimation

Finally, we turn to the singular value estimation results of Kerenidis and Prakash [KP17b]. Kerenidis and Prakash mostly use singular value estimation in order to implement singular vector projectors, with similar complexity to that of Theorem 3.2.8. However, there is a subtle issue stemming from the ambiguity of the phase labels produced by phase estimation, which is not explicitly handled in their implementation [KP17b, Algorithm (5.)1, step 4]. Using our techniques combined with ideas of Chakraborty et al. [CGJ19], we show an alternative approach to singular value estimation.

Suppose that $A = \widetilde{\Pi} U \Pi$, and we would like to perform singular value estimation of A. The main idea is to first implement an operator V such that $(I \otimes \Pi) V (I \otimes \Pi) = \sum_{t=0}^{2^n-1} |t\rangle\langle t| \otimes T_{2t}^{(SV)}(A)$. This can be done by using controlled quantum walk steps, i.e., using controlled alternating phase modulation sequences with phases as in Lemma 2.2.7. Suppose that $|\psi_j\rangle \in \mathrm{img}(\Pi)$ is a right singular vector of A with singular value $\cos(\theta_j)$, then

$$\begin{aligned}(I \otimes \Pi) V \big(H^{\otimes n} \otimes I\big) |0\rangle^n |\psi_j\rangle &= (I \otimes \Pi) V \frac{1}{\sqrt{2^n}} \sum_{t=0}^{2^n-1} |t\rangle |\psi_j\rangle \\ &= \frac{1}{\sqrt{2^n}} \sum_{t=0}^{2^n-1} \cos(2t\theta_j) |t\rangle |\psi_j\rangle.\end{aligned}$$

One can show that the norm of this state N_j is always bigger than some constant. However, the problem is that the norm N_j depends on the singular value $\cos(\theta_j)$. Fortunately we can use singular vector transformation using the projected unitary encoding $(I \otimes \Pi) V (H^{\otimes n} \otimes I)(|0\rangle\langle 0|^{\otimes n} \otimes \Pi)$ by Theorem 3.2.3 in order to ε-approximate the map $|0\rangle^n |\psi_j\rangle \to \frac{1}{\sqrt{2^n}} \sum_{j=0}^{2^n-1} \frac{\cos(2t\theta_j)}{N_j} |t\rangle |\psi_j\rangle$. Applying a Fourier transform on the first (time) register, and taking half of the absolute value of the resulting estimation solves the singular value estimation problem. The correctness can be seen using the usual analysis of quantum phase estimation [CEMM98] by utilizing the identity $\cos(x) = \frac{e^{ix} + e^{-ix}}{2}$. For more details see [CGJ19, version 2].

3.3 Matrix arithmetics using blocks of unitaries

In this section we describe a generic toolbox for implementing matrix calculations on a quantum computer, representing matrices by unitary circuits and vectors as

[13] In case we would have a subnormalized version of $\dot{X}$, in order to get maximal efficiency, it is usually worth amplifying $\dot{X}$ using Theorem 3.2.7 before taking the product $\dot{X}Y$.

quantum states. The matrix arithmetics methodology we propose carries out all calculations in an operational way, meaning that the matrices are represented by blocks of unitary operators of the quantum system, thus can in principle result in exponential speed-ups in terms of the dimension of the matrices. The methodology we describe is a distilled version of the results of a series of works on quantum algorithms [HHL09, BCC+15, CKS17, LC16, vAGGdW17, CGJ19].

We present the results in an intuitively structured way. First we define how to represent arbitrary matrices as blocks of unitaries, and show how to efficiently encode various matrices this way. Then we show how to implement addition and subtraction of these matrices, and finally show how to efficiently obtain products of block-encoded matrices. In order to make the results maximally reusable we also give bounds on the propagation of errors arising from inaccurate encodings.

3.3.1 Block-encoding

We introduce a definition of block-encoding which we are going to work with in the rest of the dissertation. As we already mentioned in the introduction, the main idea is to represent a (subnormalized) matrix as the upper-left block of a unitary.

$$U = \begin{bmatrix} A/\alpha & . \\ . & . \end{bmatrix} \qquad \Longrightarrow \qquad A = \alpha(\langle 0| \otimes I)U(|0\rangle \otimes I)$$

3.3.1. Definition (Block-encoding). Suppose that A is an s-qubit operator, $\alpha, \varepsilon \in \mathbb{R}_+$ and $a \in \mathbb{N}$, then we say that the $(s+a)$-qubit unitary U is an (α, a, ε)-block-encoding of A, if

$$\left\| A - \alpha(\langle 0|^{\otimes a} \otimes I)U(|0\rangle^{\otimes a} \otimes I)\right\| \leq \varepsilon.$$

Note that since $\|U\| = 1$ we necessarily have $\|A\| \leq \alpha + \varepsilon$. Also note that using the above definition it seems that we can only represent square matrices of size $2^s \times 2^s$. However, this is not really a restriction. Suppose that $A \in \mathbb{C}^{n\times m}$, where $n, m \leq 2^s$. Then we can define an embedding matrix denoted by $A_e \in \mathbb{C}^{2^s\times 2^s}$ such that the top-left block of A_e is A and all other elements are 0. This embedding is a faithful representation of the matrix A. Suppose that $A, B \in \mathbb{C}^{n\times m}$ are matrices, then $A_e + B_e = (A + B)_e$. Moreover, suppose $C \in \mathbb{C}^{m\times k}$ for some $k \leq 2^s$, then $A_e \cdot C_e = (A \cdot C)_e$.

The block-encoding defined above is a special case of the projected-encoding of Definition 2.3.1, therefore we can later apply our singular value transformation results for block-encoded matrices. In this manner the advantage of block-encoding is that the C_ΠNOT gate which is required in order to implement the gates of Figure 2.2 is just a Toffoli gate on $a + 1$ qubits, which can be implemented by $\mathcal{O}(a + 1)$ two-qubit gates and using a single additional ancilla qubit [HLZ+17].

3.3.2 Constructing block-encodings

A unitary matrix is a $(1,0,0)$-block-encoding of itself, which we call a *trivial block-encoding.* If we ε-approximately implement a unitary U using a ancilla qubits via a unitary $\tilde{U}$ acting jointly on the system and the ancilla qubits, then $\tilde{U}$ is an $(1,a,\varepsilon)$-block-encoding of U. This is also a rather trivial encoding. Note that we make a slight distinction between ancilla qubits that are exactly returned to their original state after the computation and the ones that might pick up some error. The latter qubits we will treat as part of the encoding, and the former qubits we usually treat separately as purely ancillary qubits.

Now we present some non-trivial ways for constructing block-encodings, which will serve as a toolbox for efficiently inputting and representing matrices for arithmetic computations on a quantum computer. We will denote by I_w a w-qubit identity operator, and let SWAP_w denote the swap operation of two w-qubit registers. When clear from the context we use the simple notation $|0\rangle$ to denote $|0\rangle^{\otimes w}$.

First we show, following Low and Chuang [LC16], how to create a block-encoding of a purified density operator. This technique can be used in combination with the optimal block-Hamiltonian simulation result Theorem 3.4.3, in order to get much better simulation performance, compared to density matrix exponentiation techniques [LMR14, KLL$^+$17] which does not use purification. This result can be generalized for subnormalized density operators too, for more details see [vAG19].

3.3.2. Lemma (Block-encoding of density operators). *Suppose that ρ is an s-qubit density operator and G is an $(a+s)$-qubit unitary that on the $|0\rangle|0\rangle$ input state prepares a purification $|0\rangle|0\rangle \to |\rho\rangle$, s.t. $\mathrm{Tr}_a|\rho\rangle\langle\rho| = \rho$. Then $(G^\dagger \otimes I_s)(I_a \otimes \mathrm{SWAP}_s)(G \otimes I_s)$ is a $(1, a+s, 0)$-block-encoding of ρ.*

Proof:
Let r be the Schmidt rank of ρ, let $\{|\psi_k\rangle : k \in [2^s]\}$ be an orthonormal basis, let $\{|\phi_k\rangle : k \in [r]\}$ be an orthonormal system and let $p \in [0,1]^{2^s}$ be such that $|\rho\rangle = \sum_{k=1}^{r} \sqrt{p_k}|\phi_k\rangle|\psi_k\rangle$ and $p_\ell = 0$ for all $\ell \in [2^s] \setminus [r]$. Then for all $i,j \in [2^s]$ we

have that

$$\begin{aligned}
\langle 0|^{\otimes a+s}\langle\psi_i|(G^\dagger\otimes I_s)(I_a\otimes \mathrm{SWAP}_s)(G^\dagger\otimes I_s)|0\rangle^{\otimes a+s}|\psi_j\rangle &= \\
&= \langle\rho|\langle\psi_i|(I_a\otimes\mathrm{SWAP}_s)|\rho\rangle|\psi_j\rangle\\
&= \left(\sum_{k=1}^{r}\sqrt{p_k}\langle\phi_k|\langle\psi_k|\right)\langle\psi_i|(I_a\otimes\mathrm{SWAP}_s)\left(\sum_{\ell=1}^{r}\sqrt{p_\ell}|\phi_\ell\rangle|\psi_\ell\rangle\right)|\psi_j\rangle\\
&= \left(\sum_{k=1}^{r}\sqrt{p_k}\langle\phi_k|\langle\psi_k|\langle\psi_i|\right)\left(\sum_{\ell=1}^{r}\sqrt{p_\ell}|\phi_\ell\rangle|\psi_j\rangle|\psi_\ell\rangle\right)\\
&= \sqrt{p_jp_i}\delta_{ij}\\
&= \langle\psi_i|\rho|\psi_j\rangle.
\end{aligned}$$

□

Van Apeldoorn and Gilyén [vAG19] recently also showed that an implementation scheme for a binary POVM measurement $\{M, I-M\}$ can also be easily transformed to block-encodings of the POVM operators. By an implementation scheme we mean a quantum circuit U that given an arbitrary input state ρ, sets a flag qubit to 0 with probability $\mathrm{Tr}[\rho M]$. (In the following lemma the CNOT gate is controlled on the second qubit.)

3.3.3. Lemma (Block-encoding of POVM operators). *Suppose that U is an $(a+s)$-qubit unitary, which implements a POVM operator M with ε-precision such that for all s-qubit density operators ρ*

$$\left|\mathrm{Tr}[\rho M]-\mathrm{Tr}\left[U\left(|0\rangle\langle 0|^{\otimes a}\otimes\rho\right)U^\dagger\left(|0\rangle\langle 0|^{\otimes 1}\otimes I_{a+s-1}\right)\right]\right|\leq\varepsilon. \tag{3.13}$$

Then $(I_1\otimes U^\dagger)(\mathrm{CNOT}\otimes I_{a+s-1})(I_1\otimes U)$ is a $(1,1+a,\varepsilon)$-block-encoding of the matrix M.

Proof:
First observe that by the cyclicity of trace we have that

$$\begin{aligned}
\mathrm{Tr}\left[U\left(|0\rangle\langle 0|^{\otimes a}\otimes\rho\right)U^\dagger(|0\rangle\langle 0|\otimes I_{a+s-1})\right]&\\
&= \mathrm{Tr}\left[U\left(|0\rangle^{\otimes a}\otimes I\right)\rho\left(\langle 0|^{\otimes a}\otimes I\right)U^\dagger(|0\rangle\langle 0|\otimes I_{a+s-1})\right]\\
&= \mathrm{Tr}\left[\rho\left(\langle 0|^{\otimes a}\otimes I\right)U^\dagger(|0\rangle\langle 0|\otimes I_{a+s-1})U\left(|0\rangle^{\otimes a}\otimes I\right)\right].
\end{aligned}$$

Together with (3.13) this implies that for all density operators ρ

$$\left|\mathrm{Tr}\left[\rho\left(M-\left(\langle 0|^{\otimes a}\otimes I\right)U^\dagger(|0\rangle\langle 0|\otimes I_{a+s-1})U\left(|0\rangle^{\otimes a}\otimes I\right)\right)\right]\right|\leq\varepsilon,$$

which is equivalent to saying that

$$\left\|M-\left(\langle 0|^{\otimes a}\otimes I\right)U^\dagger(|0\rangle\langle 0|\otimes I_{a+s-1})U\left(|0\rangle^{\otimes a}\otimes I\right)\right\|\leq\varepsilon.$$

We can conclude by observing that

$$\big(\langle 0|^{\otimes a}\otimes I\big)U^\dagger(|0\rangle\langle 0|\otimes I_{a+s-1})U\big(|0\rangle^{\otimes a}\otimes I\big) =$$
$$= \big(\langle 0|^{\otimes 1+a}\otimes I\big)\big(I_1\otimes U^\dagger\big)(\mathrm{CNOT}\otimes I_{a+s-1})\Big(I_1\otimes U\Big)\big(|0\rangle^{\otimes 1+a}\otimes I\big). \qquad \square$$

Now we turn to a more traditional way of constructing block-encodings via state preparation. This is a common technique for example to implement quantum walks. Now we introduce the notation $[n]-1$ to denote the set $\{0,1,\ldots,n-1\}$.

3.3.4. Lemma (Block-encoding of Gram matrices by state preparation unitaries). *Let U_L and U_R be "state preparation" unitaries acting on $a+s$ qubits preparing the vectors $\{|\psi_i\rangle : i\in[2^s]-1\}$, $\{|\phi_j\rangle : j\in[2^s]-1\}$, s.t.*

$$U_L\colon |0\rangle|i\rangle \to |\psi_i\rangle$$
$$U_R\colon |0\rangle|j\rangle \to |\phi_j\rangle.$$

Then $U = U_L^\dagger U_R$ is an $(1,a,0)$-block-encoding of the Gram matrix A such that $A_{ij} = \langle\psi_i|\phi_j\rangle$.

Based on the above idea one can efficiently implement block-encodings of sparse-access matrices.

3.3.5. Lemma (Block-encoding of sparse-access matrices). *Let $A\in\mathbb{C}^{2^w\times 2^w}$ be a matrix that is s_r-row-sparse and s_c-column-sparse, and each element of A has absolute value at most 1. Suppose that we have access to the following sparse-access oracles acting on two $(w+1)$ qubit registers*

$$\mathrm{O}_r\colon |i\rangle|k\rangle \to |i\rangle|r_{ik}\rangle \qquad \forall i\in[2^w]-1, k\in[s_r], \text{ and}$$
$$\mathrm{O}_c\colon |\ell\rangle|j\rangle \to |c_{\ell j}\rangle|j\rangle \qquad \forall \ell\in[s_c], j\in[2^w]-1, \text{ where}$$

r_{ij} is the index for the j-th non-zero entry of the i-th row of A, or if there are less than i non-zero entries, then it is $j+2^w$, and similarly c_{ij} is the index for the i-th non-zero entry of the j-th column of A, or if there are less than j non-zero entries, then it is $i+2^w$. Additionally assume that we have access to an oracle O_A that returns the entries of A in a binary description

$$\mathrm{O}_A\colon |i\rangle|j\rangle|0\rangle^{\otimes b} \to |i\rangle|j\rangle|a_{ij}\rangle \qquad \forall i,j\in[2^w]-1, \text{ where}$$

a_{ij} is a b-bit binary description[14] of the ij-matrix element of A. Then we can implement a $(\sqrt{s_r s_c}, w+3, \varepsilon)$-block-encoding of A with a single use of $\mathrm{O}_r, \mathrm{O}_c$, two uses of O_A and additionally using $\mathcal{O}\big(w+\log^{2.5}(\frac{s_r s_c}{\varepsilon})\big)$ one and two-qubit gates while using $\mathcal{O}\big(b+\log^{2.5}(\frac{s_r s_c}{\varepsilon})\big)$ ancilla qubits.

[14]For simplicity we assume here that the binary representation is exact.

Proof:
We proceed by constructing state preparation unitaries like in Lemma 3.3.4. We will work with 3 registers the first of which is a single-qubit register, and the other two registers have $(w+1)$ qubits. Let D_s be a $(w+1)$-qubit unitary that implements the map $|0\rangle \to \sum_{k=1}^{s} \frac{|k\rangle}{\sqrt{s}}$. It is known that this operator D_s can be implemented with $\mathcal{O}(w)$ quantum gates using $\mathcal{O}(1)$ ancilla qubits. Then we define the $2(w+1)$-qubit unitary $V_L := \mathrm{O}_r(I_{w+2} \otimes D_{s_r})\mathrm{SWAP}_{w+1}$ such that

$$V_L\colon |0\rangle^{w+2}|i\rangle \to \sum_{k=1}^{s_r} \frac{|i\rangle|r_{ik}\rangle}{\sqrt{s_r}} \qquad \forall i \in [2^w]-1.$$

We implement the operator $V_R := \mathrm{O}_c(D_{s_c} \otimes I_{w+1})$ in a similar way acting as

$$V_R\colon |0\rangle^{w+2}|j\rangle \to \sum_{\ell=1}^{s_c} \frac{|c_{\ell j}\rangle|j\rangle}{\sqrt{s_c}} \qquad \forall j \in [2^w]-1.$$

It is easy to see that the above unitaries are such that

$$\langle 0|^{w+2}\langle i|V_L^\dagger V_R|0\rangle^{w+2}|j\rangle = \frac{1}{\sqrt{s_r s_c}} \text{ if } a_{ij} \neq 0 \text{ and } 0 \text{ otherwise.}$$

Now we define $U_L := I_1 \otimes V_L$ and define U_R as performing the unitary $I_1 \otimes V_R$ followed by some extra computation. After performing V_R we get a superposition of index pairs $|i\rangle|j\rangle$. Given an index pair $|i\rangle|j\rangle$ we query the matrix element $|a_{ij}\rangle$ using the oracle O_A. Then we do some elementary computations in order to implement a single-qubit gate $|0\rangle \to a_{ij}|0\rangle + \sqrt{1-|a_{ij}|^2}|1\rangle$ on the first qubit, with precision $\mathcal{O}\Big(\mathrm{poly}\Big(\frac{\varepsilon}{s_r s_c}\Big)\Big)$. This can be executed with the stated complexity, for more details see, e.g., the work of Berry et al. [BCK15]. Finally we also need to uncompute everything, which requires one more use of O_A. This way we get a good approximation of

$$U_R\colon |0\rangle^{w+3}|j\rangle \to \sum_{\ell=1}^{s_c} \frac{\Big(a_{c_{\ell j}j}|0\rangle + \sqrt{1-|a_{c_{\ell j}j}|^2}|1\rangle\Big)|c_{\ell j}\rangle|j\rangle}{\sqrt{s_c}} \qquad \forall j \in [2^w]-1. \qquad \square$$

Note that in the above method the matrix gets subnormalized by a factor of $\frac{1}{\sqrt{s_r s_c}}$. If we would know that for example $\|A\| \leq \frac{1}{2}$, then we could amplify the block-encoding in order to remove this unwanted subnormalization using singular value amplification Theorem 3.2.7 using the block-encoding roughly $\sqrt{s_r s_c}$ times. However, under some circumstances one can defeat the subnormalization more efficiently by doing an amplification at the level of the state preparation unitaries. The idea comes from Low and Chuang [LC17a], who called this technique “Uniform spectral gap amplification”. We generalize their results combining with ideas of Kerenidis and Prakash [KP17a] and Chakraborty et al. [CGJ19], who used similar ideas but assumed QROM-access to matrices rather than sparse-access.

3.3.6. Lemma (Preamplified block-encoding of sparse-access matrices). *Let $A \in \mathbb{C}^{2^w \times 2^w}$ be a matrix that is s_r-row-sparse and s_c-column-sparse, and is given using the input oracles defined in Lemma 3.3.5. Let $a_{i.}$ denote the i-th row of A and similarly $a_{.j}$ the j-th column. Let $q \in [0,2]$ and suppose that $n_r \in [1, s_r]$ is an upper bound on $\|a_{i.}\|_q^q$ and $n_c \in [1, s_c]$ is an upper bound on $\|a_{.j}\|_{2-q}^{2-q}$.*

Let $m = \max[\frac{s_r}{n_r}, \frac{s_c}{n_c}]$. Then we can implement a $\left(\sqrt{2n_r n_c}, w+6, \varepsilon\right)$-block-encoding of A with $\mathcal{O}\left(\sqrt{\frac{s_r}{n_r}}\log(\frac{s_r s_c}{\varepsilon})\right)$ uses of O_r, $\mathcal{O}\left(\sqrt{\frac{s_c}{n_c}}\log(\frac{s_r s_c}{\varepsilon})\right)$ uses of O_c, $\mathcal{O}\left(\sqrt{m}\log(\frac{s_r s_c}{\varepsilon})\right)$ uses of O_A, while using $\mathcal{O}\left(\sqrt{m}\left(w\log(\frac{s_r s_c}{\varepsilon}) + \log^{3.5}(\frac{s_r s_c}{\varepsilon})\right)\right)$ additional one and two-qubit gates and $\mathcal{O}\left(b + \log^{2.5}(\frac{s_r s_c}{\varepsilon})\right)$ ancilla qubits.

Proof:
The idea is very similar to the proof of Lemma 3.3.5, we implement the unitaries V_L, V_R the same way. However, we define U_R, U_L slightly differently. Using a similar method as in Lemma 3.3.5, we implement $\mathcal{O}\left(\mathrm{poly}\left(\frac{\varepsilon}{s_r s_c}\right)\right)$-approximations of the maps U_L, U_R which for all $i \in [2^w]-1$ and $j \in [2^w]-1$ map

$$U_L\colon |0\rangle^{w+4}|i\rangle \to \sum_{k=1}^{s_r} \frac{\left(|a_{ir_{ik}}|^{\frac{q}{2}}|0\rangle + \sqrt{1-|a_{ir_{ik}}|^q}|1\rangle\right)|0\rangle|i\rangle|r_{ik}\rangle}{\sqrt{s_r}}$$

$$U_R\colon |0\rangle^{w+4}|j\rangle \to \sum_{\ell=1}^{s_c} \frac{\frac{a_{c_{\ell j}j}}{|a_{c_{\ell j}j}|}|0\rangle\left(|a_{c_{\ell j}j}|^{1-\frac{q}{2}}|0\rangle + \sqrt{1-|a_{c_{\ell j}j}|^{2-q}}|1\rangle\right)|c_{\ell j}\rangle|j\rangle}{\sqrt{s_c}}.$$

It is easy to see that the above unitaries are such that

$$\langle 0|^{w+4}\langle i|U_L^\dagger U_R|0\rangle^{w+4}|j\rangle = \frac{a_{ij}}{\sqrt{s_r s_c}} \qquad \forall i,j \in [2^w]-1.$$

We can see that for all $i \in [2^w]-1$ the modified row vector $\sum_{k=1}^{s_r} \frac{|a_{ir_{ik}}|^{\frac{q}{2}}|0\rangle|0\rangle|i\rangle|r_{ik}\rangle}{\sqrt{s_r}}$ has squared norm at most $\frac{n_r}{s_r}$, and a similar $\frac{n_c}{s_c}$ upper bound holds for the squared norm of the modified column vector. Also observe that

$$(|0\rangle\langle 0| \otimes I_{2w+3})U_L(|0\rangle\langle 0|^{w+4} \otimes I_w) = \sum_{j=0}^{2^w-1}\left(\sum_{k=1}^{s_r} \frac{|a_{ir_{ik}}|^{\frac{q}{2}}|0\rangle|0\rangle|i\rangle|r_{ik}\rangle}{\sqrt{s_r}}\right)\langle 0|^{w+4}\langle j|,$$

which is a singular value decomposition with the singular values being the modified row norms. Therefore we can apply singular value amplification (Theorem 3.2.7) with amplification $\gamma_r = \sqrt{\frac{s_r}{\sqrt{2}n_r}}$ and precision $\mathcal{O}\left(\mathrm{poly}\left(\frac{\varepsilon}{s_r s_c}\right)\right)$ resulting in an $\mathcal{O}\left(\mathrm{poly}\left(\frac{\varepsilon}{s_r s_c}\right)\right)$ approximation of $\tilde{U}_L$ such that

$$\begin{aligned}(\langle +| \otimes |0\rangle\langle 0| \otimes I_{2w+3})\tilde{U}_L(|+\rangle \otimes |0\rangle\langle 0|^{w+4} \otimes I_w)&\\ = \gamma_r \sum_{j=0}^{2^w-1}\left(\sum_{k=1}^{s_r} \frac{|a_{ir_{ik}}|^{\frac{q}{2}}|0\rangle|0\rangle|i\rangle|r_{ik}\rangle}{\sqrt{s_r}}\right)\langle 0|^{w+4}\langle j|.&\end{aligned}$$

Similarly we apply singular value amplification with amplification $\gamma_c = \sqrt{\frac{s_c}{\sqrt{2}n_c}}$ and precision $\mathcal{O}\Big(\mathrm{poly}\Big(\frac{\varepsilon}{s_r s_c}\Big)\Big)$ resulting in a $\mathcal{O}\Big(\mathrm{poly}\Big(\frac{\varepsilon}{s_r s_c}\Big)\Big)$ approximation of $\tilde{U}_R$ such that

$$\langle{+}{+}|\langle 0|^{w+4}\langle i|\tilde{U}_L^\dagger \tilde{U}_R|{+}{+}\rangle|0\rangle^{w+4}|j\rangle = \gamma_r\gamma_c\frac{a_{ij}}{\sqrt{s_r s_c}} = \frac{a_{ij}}{\sqrt{2n_r n_c}} \qquad \forall i,j\in[2^w]-1.$$

Finally adding 4 Hadamard gates we can change the $|+\rangle$ states above to $|0\rangle$ states, resulting in the $\big(\sqrt{\frac{1}{2n_r n_c}}, w+6, \varepsilon\big)$-block-encoding of A. The complexity statement follows similarly as in the proof of Lemma 3.3.5, with the extra observation that the singular value amplifications of U_L and U_R can be performed using degree $\mathcal{O}\big(\gamma_r \log(\frac{s_r s_c}{\varepsilon})\big)$ and $\mathcal{O}\big(\gamma_c \log(\frac{s_r s_c}{\varepsilon})\big)$ singular value transformations respectively. $\square$

Finally, for completeness we invoke results of Kerenidis and Prakash [KP17a] and Chakraborty et al. [CGJ19], who showed how to efficiently implement block-encodings of matrices that are stored in a clever quantum data structure in QROM. The QROM and the data structure for the matrix A enables performing several useful tasks in time that is polylogarithmic in the size of the matrix and the required precision. These tasks include: for any given row of A compute its norm or prepare a quantum state that is proportional to it; compute the maximal row norm and the Frobenius norm of the matrix, and prepare a quantum state that has amplitudes proportional to the norms of the rows of the matrix A.

For $q\in[0,2]$ let us define $\mu_q(A)=\sqrt{n_q(A)n_{(2-q)}(A^\dagger)}$, where $n_q(A) := \max_i \|a_{i.}\|_q^q$ is the q-th power of the maximum q-norm of the rows of A. Let $A^{(q)}$ denote the matrix of the same dimensions as A, with[15] $A^{(q)}_{ij}=\sqrt{a^q_{ij}}$. The following was proven in [KP17a], although not in the language of block-encodings, and was stated in this form by Chakraborty et al. [CGJ19]. The result can be proven along the lines of Lemma 3.3.4.

3.3.7. LEMMA (Block-encodings of matrices stored in quantum data structures). *Let $A\in\mathbb{C}^{2^w\times 2^w}$.*

1. *Fix $q\in[0,2]$. If $A^{(q)}$ and $(A^{(2-q)})^\dagger$ are both stored in quantum-accessible data structures[16], then there exist unitaries U_R and U_L that can be implemented in time $\mathcal{O}(\mathrm{poly}(w\log(1/\varepsilon)))$ and $U_R^\dagger U_L$ is a $(\mu_q(A), w+2, \varepsilon)$-block-encoding of A.*

2. *On the other hand, if A is stored in a quantum-accessible data structure[16], then there exist unitaries U_R and U_L that can be implemented in time $\mathcal{O}(\mathrm{poly}(w\log(1/\varepsilon)))$ and $U_R^\dagger U_L$ is an $(\|A\|_F, w+2, \varepsilon)$-block-encoding of A.*

[15]For complex values we define these non-integer powers using the principal value of the complex logarithm function.

[16]Here we assume that the data-structure stores the matrices with sufficient precision, cf. [CGJ19].

3.3.3 Addition and subtraction: Linear combination of block-encoded matrices

We use a simple but powerful method for implementing linear combinations of unitary operators on a quantum computer. This technique was introduced by Berry et al. [BCC+15] for exponentially improving the precision of Hamiltonian simulation. Later it was adapted by Childs et al. [CKS17] for exponentially improving the precision of quantum linear equation solving. Here we present this method from the perspective of block-encoded matrices.

First we define state-preparation unitaries in order to conveniently state our result in the following lemma.

3.3.8. Definition (State preparation pair). Let $y \in \mathbb{C}^m$ and $\|y\|_1 \leq \beta$. The pair of unitaries (P_L, P_R) is called a (β, b, ε)-state-preparation-pair if $P_L|0\rangle^{\otimes b} = \sum_{j=0}^{2^b-1} c_j|j\rangle$ and $P_R|0\rangle^{\otimes b} = \sum_{j=1}^{2^b-1} d_j|j\rangle$ such that $\sum_{j=0}^{m-1} |\beta(c_j^* d_j) - y_j| \leq \varepsilon$ and for all $j \in m, \ldots, 2^b - 1$ we have $c_j^* d_j = 0$.

Now we show how to implement a block-encoding of a linear combination of block-encoded operators.

3.3.9. Lemma (Linear combination of block-encoded matrices).
Let $A = \sum_{j=1}^{m} y_j A_j$ be an s-qubit operator and $\varepsilon \in \mathbb{R}_+$. Suppose that (P_L, P_R) is a $(\beta, b, \varepsilon_1)$-state-preparation-pair for y, $W = \sum_{j=0}^{m-1} |j\rangle\langle j| \otimes U_j + ((I - \sum_{j=0}^{m-1} |j\rangle\langle j|) \otimes I_a \otimes I_s)$ is an $(s+a+b)$-qubit unitary such that for all $j \in 0, \ldots, m$ we have that U_j is an $(\alpha, a, \varepsilon_2)$-block-encoding of A_j. Then we can implement a $(\alpha\beta, a+b, \alpha\varepsilon_1 + \alpha\beta\varepsilon_2)$-block-encoding of A, with a single use of W, P_R and $P_L^\dagger$.

Proof:
Observe that $\widetilde{W} = (P_L^\dagger \otimes I_a \otimes I_s) W (P_R \otimes I_a \otimes I_s)$ is a $(\alpha\beta, a+b, \alpha\varepsilon_1 + \alpha\beta\varepsilon_2)$-block-encoding of A:

$$\begin{aligned}
&\left\| A - \alpha\beta(\langle 0|^{\otimes b} \otimes \langle 0|^{\otimes a} \otimes I)\widetilde{W}(|0\rangle^{\otimes b} \otimes |0\rangle^{\otimes a} \otimes I) \right\| \\
&\qquad = \left\| A - \alpha \sum_{j=0}^{m-1} \beta(c_j^* d_j)(\langle 0|^{\otimes a} \otimes I) U_j (|0\rangle^{\otimes a} \otimes I) \right\| \\
&\qquad \leq \alpha\varepsilon_1 + \left\| A - \alpha \sum_{j=0}^{m-1} y_j (\langle 0|^{\otimes a} \otimes I) U_j (|0\rangle^{\otimes a} \otimes I) \right\| \\
&\qquad \leq \alpha\varepsilon_1 + \alpha \sum_{j=0}^{m-1} y_j \left\| A_j - (\langle 0|^{\otimes a} \otimes I) U_j (|0\rangle^{\otimes a} \otimes I) \right\| \\
&\qquad \leq \alpha\varepsilon_1 + \alpha \sum_{j=0}^{m-1} y_j \varepsilon_2 \\
&\qquad \leq \alpha\varepsilon_1 + \alpha\beta\varepsilon_2. \qquad \square
\end{aligned}$$

3.3.4 Multiplication: Product of block-encoded matrices

In general if we want to take the product of two block-encoded matrices we need to treat their ancilla qubits separately. In this case, as the following lemma shows, the errors simply add up and the block-encoding does not introduce any additional errors.

3.3.10. LEMMA (Product of block-encoded matrices). *If U is an (α, a, δ)-block-encoding of an s-qubit operator A, and V is an (β, b, ε)-block-encoding of an s-qubit operator B then*[17] $(I_b \otimes U)(I_a \otimes V)$ *is an* $(\alpha\beta, a+b, \alpha\varepsilon + \beta\delta)$*-block-encoding of* AB.

Proof:

$$\begin{aligned}
&\left\|AB - \alpha\beta(\langle 0|^{\otimes a+b} \otimes I)(I_b \otimes U)(I_a \otimes V)(|0\rangle^{\otimes a+b} \otimes I)\right\| \\
=&\left\|AB - \underbrace{\alpha(\langle 0|^{\otimes a} \otimes I)U(|0\rangle^{\otimes a} \otimes I)}_{\tilde{A}}\underbrace{\beta(\langle 0|^{\otimes b} \otimes I)V(|0\rangle^{\otimes b} \otimes I)}_{\tilde{B}}\right\| \\
=&\left\|AB - \tilde{A}B + \tilde{A}B - \tilde{A}\tilde{B}\right\| \\
=&\left\|(A - \tilde{A})B + \tilde{A}(B - \tilde{B})\right\| \\
\leq&\left\|A - \tilde{A}\right\|\beta + \alpha\left\|B - \tilde{B}\right\| \\
\leq&\alpha\varepsilon + \beta\delta.
\end{aligned}$$

□

In the special case when the encoded matrices are unitaries and their block-encoding does not use any extra scaling factor, then we might reuse the ancilla qubits, however it introduces an extra error term, which can be bounded by the geometrical mean of the two input error bounds.

3.3.11. LEMMA (Product of two block-encoded unitaries). *If U is an $(1, a, \delta)$-block-encoding of an s-qubit unitary operator A, and V is a $(1, a, \varepsilon)$-block-encoding of an s-qubit unitary operator B then UV is a $(1, a, \delta + \varepsilon + 2\sqrt{\delta\varepsilon})$-block-encoding of the unitary operator AB.*

Proof:
It is enough to show that for all s-qubit pure states $|\phi\rangle, |\psi\rangle$ we have that

$$\left|\langle\phi|AB|\psi\rangle - \langle\phi|(\langle 0|^{\otimes a} \otimes I)UV(|0\rangle^{\otimes a} \otimes I)|\psi\rangle\right| \leq \delta + \varepsilon + 2\sqrt{\delta\varepsilon}.$$

[17]The identity operators act on each other's ancilla qubits, which is hard to express properly using simple tensor notation, but the reader should read this tensor product this way.

Observe that

$$\begin{aligned}
&\langle\phi|(\langle 0|^{\otimes a}\otimes I)UV(|0\rangle^{\otimes a}\otimes I)|\psi\rangle\\
=&\langle\phi|(\langle 0|^{\otimes a}\otimes I)U\big((|0\rangle\langle 0|^{\otimes a}\otimes I)+\big((I-|0\rangle\langle 0|^{\otimes a})\otimes I\big)\big)V(|0\rangle^{\otimes a}\otimes I)|\psi\rangle\\
=&\langle\phi|(\langle 0|^{\otimes a}\otimes I)U(|0\rangle^{\otimes a}\otimes I)(\langle 0|^{\otimes a}\otimes I)V(|0\rangle^{\otimes a}\otimes I)|\psi\rangle\\
&+\langle\phi|(\langle 0|^{\otimes a}\otimes I)U\big((I-|0\rangle\langle 0|^{\otimes a})\otimes I\big)V(|0\rangle^{\otimes a}\otimes I)|\psi\rangle
\end{aligned}$$

Now we can see that similarly to the proof of Lemma 3.3.10 we have

$$\begin{aligned}
&\big|\langle\phi|AB|\psi\rangle-\langle\phi|(\langle 0|^{\otimes a}\otimes I)U(|0\rangle^{\otimes a}\otimes I)(\langle 0|^{\otimes a}\otimes I)V(|0\rangle^{\otimes a}\otimes I)|\psi\rangle\big|\\
=&\big|\langle\phi|\big(AB-(\langle 0|^{\otimes a}\otimes I)U(|0\rangle^{\otimes a}\otimes I)(\langle 0|^{\otimes a}\otimes I)V(|0\rangle^{\otimes a}\otimes I)\big)|\psi\rangle\big|\\
\leq&\big\|AB-(\langle 0|^{\otimes a}\otimes I)U(|0\rangle^{\otimes a}\otimes I)(\langle 0|^{\otimes a}\otimes I)V(|0\rangle^{\otimes a}\otimes I)\big\|\\
\leq&\delta+\varepsilon.
\end{aligned}$$

Finally note that

$$\begin{aligned}
&\big|\langle\phi|(\langle 0|^{\otimes a}\otimes I)U\big((I-|0\rangle\langle 0|^{\otimes a})\otimes I\big)V(|0\rangle^{\otimes a}\otimes I)|\psi\rangle\big|\\
=&\Big|\langle\phi|(\langle 0|^{\otimes a}\otimes I)U\big((I-|0\rangle\langle 0|^{\otimes a})\otimes I\big)^2V(|0\rangle^{\otimes a}\otimes I)|\psi\rangle\Big|\\
\leq&\big\|\big((I-|0\rangle\langle 0|^{\otimes a})\otimes I\big)U(|0\rangle^{\otimes a}\otimes I)|\phi\rangle\big\|\cdot\big\|\big((I-|0\rangle\langle 0|^{\otimes a})\otimes I\big)V(|0\rangle^{\otimes a}\otimes I)|\psi\rangle\big\|\\
=&\sqrt{1-\ \big\|(|0\rangle\langle 0|^{\otimes a}\otimes I)U(|0\rangle^{\otimes a}\otimes I)|\phi\rangle\big\|^2}\cdot\sqrt{1-\ \big\|(|0\rangle\langle 0|^{\otimes a}\otimes I)V(|0\rangle^{\otimes a}\otimes I)|\psi\rangle\big\|^2}\\
\leq&\sqrt{1-(1-\delta)^2}\cdot\sqrt{1-(1-\varepsilon)^2}\\
\leq&2\sqrt{\delta\varepsilon}.
\end{aligned}$$

□

The following corollary suggests that if we multiply together multiple block-encoded unitaries, the error may grow super-linearly, but it increases at most quadratically with the number of factors in the product.

3.3.12. Corollary (Product of multiple block-encoded unitaries). *Suppose that U_j is a $(1,a,\varepsilon)$-block-encoding of an s-qubit unitary operator W_j for all $j\in[K]$. Then $\prod_{j=1}^K U_j$ is a $(1,a,4K^2\varepsilon)$-block-encoding of $\prod_{j=1}^K W_j$.*

Proof:
First observe that for the product of two matrices we get the precision bound 4ε by the above lemma. If $K=2^k$ for some $k\in\mathbb{N}$. Then we can apply the above observation in a recursive fashion in a binary tree structure, to get the upper bound $4^k\varepsilon$ on the precision, and observe that $4^k=K^2$.

If $2^{k-1}\leq K<2^k$, then we can just add identity operators so that we have 2^k matrices to multiply, giving the precision bound $4^k\varepsilon\leq 4^{1+\log_2 K}\varepsilon=4K^2\varepsilon$. □

3.4 Implementing smooth functions of Hermitian matrices

In the previous section we developed an efficient methodology to perform basic matrix arithmetics, such as addition and multiplication. In principle all smooth functions of matrices can be approximated arbitrarily precisely using such basic arithmetic operations. In this section we show a more efficient way to transform Hermitian matrices according to smooth functions using singular value transformation techniques. The key observation is that for a Hermitian matrix A we have that $P^{(SV)}(A) = P(A)$, i.e., singular value transformation and eigenvalue transformation coincide.

The following theorem is our improvement of Corollary 2.3.8 removing the counter-intuitive parity constraint at the expense of a subnormalization factor 1/2, which is not a problem in most applications.

3.4.1. Theorem (Polynomial eigenvalue transformation of arbitrary parity). *Suppose that U is an (α, a, ε)-block-encoding of a Hermitian matrix A. If $\delta \geq 0$ and $P_{\Re} \in \mathbb{R}[x]$ is a degree-d polynomial satisfying that*

- *for all $x \in [-1,1]$: $|P_{\Re}(x)| \leq \frac{1}{2}$.*

Then there is a quantum circuit $\tilde{U}$, which is an $(1, a+2, 4d\sqrt{\varepsilon/\alpha}+\delta)$-encoding of $P_{\Re}(A/\alpha)$, and which consists of d applications of U and $U^\dagger$ gates, a single application of controlled-U and $\mathcal{O}((a+1)d)$ other one- and two-qubit gates. Moreover we can compute a description of such a circuit with a classical computer in time $\mathcal{O}(\mathrm{poly}(d, \log(1/\delta)))$.

Proof:
First note that for a Hermitian matrix A and for any even/odd polynomial $P \in \mathbb{C}[x]$ we have that $P^{(SV)}(A) = P(A)$. Now let $P_{\Re}^{(\mathrm{even})}(x) := P_{\Re}(x) + P_{\Re}(-x) \leq 1$, and let $P_{\Re}^{(\mathrm{odd})}(x) := P_{\Re}(x) - P_{\Re}(-x) \leq 1$. Then we can implement both $P_{\Re}^{(\mathrm{even})}(x)$ and $P_{\Re}^{(\mathrm{odd})}(x)$ using Corollary 2.3.8 and we can take an equal $\frac{1}{2}/\frac{1}{2}$ linear combination of them by Lemma 2.4.3. Using the notation of Figure 2.2 the final circuit is simply $(H \otimes H \otimes I)U_{\Phi^{(c)}}(H \otimes H \otimes I)$, where H denotes the Hadamard gate. $\square$

Note that a similar statement can be proven for arbitrary $P \in \mathbb{C}[x]$ that satisfy $|P(x)| \leq \frac{1}{4}$ for all $x \in [-1,1]$. The only difference is that for implementing the complex part one needs to add a (controlled) phase $e^{i\frac{\pi}{2}}$. The $\leq \frac{1}{4}$ constraint comes from the fact that the implementation is a sum of 4 different terms (even/odd component of the real/imaginary part).

3.4.1 Optimal Hamiltonian simulation

Let us define for $t \in \mathbb{R}_+$ and $\varepsilon \in (0,1)$ the number $r(t,\varepsilon) \geq t$ as the unique solution (in r) to the equation

$$\varepsilon = \left(\frac{t}{r}\right)^r : r \in (t,\infty). \tag{3.14}$$

This equation is closely related to the Lambert-W function, and unfortunately we cannot give the solution in terms of elementary functions. However, one can see that the function $\left(\frac{t}{r}\right)^r$ is strictly monotone decreasing for $r \in [t,\infty)$ and in the limit $r \to \infty$ it tends to 0. Since for $r = t$ the function value is 1, the equation (3.14) has a unique solution. In particular for any $r, R \in [t,\infty)$ such that $\left(\frac{t}{r}\right)^r \geq \varepsilon \geq \left(\frac{t}{R}\right)^R$ we have that $r \leq r(t,\varepsilon) \leq R$. This is an important expression for this section, since Low and Chuang proved [LC17b, LC16] that the complexity of Hamiltonian simulation for time t with precision ε is $\Theta(r(|t|,\varepsilon))$.

Low and Chuang also claimed [LC17b] that for all $t \geq 1$ one gets $r(t,\varepsilon) = \Theta\left(t + \frac{\log(1/\varepsilon)}{\log(\log(1/\varepsilon))}\right)$, which led to their complexity statement. However, there is a subtle issue in their calculations, and it turns out that this formula is invalid for some range of values of t. We show in Lemma 3.4.4 how to correct the formula by a slight modification of the $\log(\log(1/\varepsilon))$ term. This is the reason why we need to give more complicated expressions for the complexity of block-Hamiltonian simulation. First we show what is the connection between equation (3.14) and the complexity of Hamiltonian simulation by constructing polynomials[18] of degree $\mathcal{O}(r(|t|,\varepsilon))$, which ε-approximate trigonometric functions with t-times rescaled argument.

3.4.2. Lemma (Polynomial approximations of the exponential function).
Let $t \in \mathbb{R} \setminus \{0\}$, $\varepsilon \in (0,\frac{2}{3})$, and let $R := \left\lfloor r\left(e|t|,\frac{3}{2}\varepsilon\right)\right\rfloor$, then the following degree-R polynomial satisfies

$$\left\| e^{itx} - \sum_{k=0}^{R} \frac{(itx)^k}{k!} \right\|_{[-1,1]} \leq \varepsilon.$$

Proof:
For all $x \in [-1,1]$ and positive integer $q \geq |et|$, by the Taylor series of e^{itx} we get

$$\left| e^{itx} - \sum_{k=0}^{q-1} \frac{(itx)^k}{k!} \right| = \left| \sum_{k=q}^{\infty} \frac{(itx)^k}{k!} \right| \leq \sum_{k=q}^{\infty} \frac{|t|^k}{k!} \leq \frac{|t|^q}{q!} \sum_{j=0}^{\infty} \left(\frac{1}{e}\right)^j < \frac{8}{5}\frac{|t|^q}{q!}. \tag{3.15}$$

[18]By using the Jacobi-Anger expansion, one can achieve further constant improvements, as shown in [GSLW19].

By Stirling's approximation for $q \geq 1$ we have $q! \geq \sqrt{2\pi q}\left(\frac{q}{e}\right)^q \geq \frac{5}{2}\left(\frac{q}{e}\right)^q$, so

$$\frac{8}{5}\frac{|t|^q}{q!} \leq \frac{16}{25}\left(\frac{e|t|}{q}\right)^q < \frac{2}{3}\left(\frac{e|t|}{q}\right)^q. \tag{3.16}$$

If we choose $q = R + 1 \geq r\big(e|t|, \frac{3}{2}\varepsilon\big)$, then $q \geq e|t|$, so Eqs. (3.15)-(3.16) hold. Moreover, we can upper bound the right-hand side of Eq. (3.16) by ε. □

Now we are ready to reprove the optimal block-Hamiltonian simulation result of Low and Chuang. The optimality is discussed in an earlier work of the same authors [LC17b].

3.4.3. Theorem. (Optimal block-Hamiltonian simulation [LC16]) *Let $t \in \mathbb{R} \setminus \{0\}$, $\varepsilon \in (0,1)$ and let U be an $(\alpha, a, 0)$-block-encoding of the Hamiltonian H. Then we can implement an ε-precise Hamiltonian simulation unitary V which is a $(1, a+2, \varepsilon)$-block-encoding of e^{itH}, with $3r\big(e\alpha|t|, \frac{\varepsilon}{8}\big)$ uses of U or its inverse, 3 uses of controlled-U or its inverse and with $\mathcal{O}\big(ar\big(e\alpha|t|, \frac{\varepsilon}{8}\big)\big)$ two-qubit gates and using $\mathcal{O}(1)$ ancilla qubits.*

Proof:
Use the polynomial of Lemma 3.4.2 to $\frac{\varepsilon}{12}$-approximate e^{itx}, and subnormalize it by a factor $(1 - \frac{\varepsilon}{12})$. Take the even (real) and the odd (imaginary) parts of the resulting $\frac{\varepsilon}{6}$-approximating polynomial, and combine them using the same method as in Theorem 3.4.1 in order to get a $(1, a+2, \frac{\varepsilon}{6})$-block-encoding of $e^{itH}/2$. Then use robust oblivious amplitude amplification (Corollary 3.2.5) in order to get a $(1, a+2, \varepsilon)$-block-encoding of e^{itH}. □

Now we prove some bounds on $r(t,\varepsilon)$, in order to make the above result more accessible.

3.4.4. Lemma (Bounds on $r(t,\varepsilon)$). *For $t \in \mathbb{R}_+$ and $\varepsilon \in (0,1)$*

$$r(t,\varepsilon) = \Theta\left(t + \frac{\ln(1/\varepsilon)}{\ln(e + \ln(1/\varepsilon)/t)}\right).$$

Moreover, for all $q \in \mathbb{R}_+$ we have that

$$r(t,\varepsilon) < e^q t + \frac{\ln(1/\varepsilon)}{q}.$$

Proof:
First consider the case $t \geq \frac{\ln(1/\varepsilon)}{e}$ and set $r := et$, then we get that

$$\forall t \geq \frac{\ln(1/\varepsilon)}{e} \colon \left(\frac{t}{et}\right)^{et} = \left(\frac{1}{e}\right)^{et} \leq \varepsilon \qquad \Longrightarrow \qquad \forall t \geq \frac{\ln(1/\varepsilon)}{e} \colon r(t,\varepsilon) \leq et. \tag{3.17}$$

Now we turn to the case $t \leq \frac{\ln(1/\varepsilon)}{e}$, and try to find $r = r(t,\varepsilon)$.

$$\left(\frac{t}{r}\right)^r = \varepsilon \qquad \Longleftrightarrow \qquad \left(\frac{r}{t}\right)^{\frac{r}{t}} = \left(\frac{1}{\varepsilon}\right)^{\frac{1}{t}} \qquad \Longleftrightarrow \qquad \frac{r}{t}\ln\left(\frac{r}{t}\right) = \ln\left(\frac{1}{\varepsilon}\right)\frac{1}{t}. \tag{3.18}$$

Let us define $x := \frac{r}{t} \geq 1$ and $c := \ln\left(\frac{1}{\varepsilon}\right)\frac{1}{t} \geq e$. We will examine the solution of the equation $x\ln(x) = c$ for $c \geq e$. The function $x\ln(x)$ is monotone increasing on $[1,\infty)$, takes value 0 at 1, and in the $x \to \infty$ limit it tends to infinity, therefore the equation $x\ln(x) = c$ has a unique solution for all $c \in \mathbb{R}_+$. Moreover, if $b, B \in [1,\infty)$ are such that $b\ln(b) \leq c \leq B\ln(B)$, then $b \leq x \leq B$. Therefore we can see that $\frac{c}{\ln(c)} \leq x$ since

$$\frac{c}{\ln(c)}\ln\left(\frac{c}{\ln(c)}\right) = \frac{c}{\ln(c)}(\ln(c) - \ln(\ln(c))) = c\left(1 - \frac{\ln(\ln(c))}{\ln(c)}\right) \leq c.$$

By a similar argument we can see that $x \leq \frac{5}{3}\frac{e+c}{\log(e+c)} \leq \frac{4c}{\log(e+c)}$, since

$$\begin{aligned}
\frac{5}{3}\frac{e+c}{\ln(e+c)}\ln\left(\frac{5}{3}\frac{e+c}{\ln(e+c)}\right) &> \frac{5}{3}\frac{e+c}{\ln(e+c)}\ln\left(\frac{e+c}{\ln(e+c)}\right)\\
&= \frac{5}{3}\frac{e+c}{\ln(e+c)}(\ln(e+c) - \ln(\ln(e+c)))\\
&= \frac{5}{3}(e+c)\left(1 - \frac{\ln(\ln(e+c))}{\ln(e+c)}\right)\\
&\geq \frac{5}{3}(e+c)\left(1 - \frac{1}{e}\right) \qquad \left(\forall y \in \mathbb{R}_+ \colon \tfrac{\ln(y)}{y} \leq \tfrac{1}{e}\right)\\
&> e + c\\
&> c.
\end{aligned}$$

Thus for $x \geq 1$, $c \geq e$ we get that the solution of the equation $x\log(x) = c$ satisfies

$$\frac{c}{\log(e+c)} \leq \frac{c}{\ln(c)} \leq x \leq \frac{4c}{\log(e+c)}. \tag{3.19}$$

Using $x = \frac{r}{t} \Rightarrow r = tx$ and $c = \ln\left(\frac{1}{\varepsilon}\right)\frac{1}{t}$ from (3.18)-(3.19) we get that

$$\forall t \leq \frac{\ln(1/\varepsilon)}{e} \colon \frac{\ln(1/\varepsilon)}{\ln(e + \ln(1/\varepsilon)/t)} \leq r(t,\varepsilon) \leq \frac{4\ln(1/\varepsilon)}{\ln(e + \ln(1/\varepsilon)/t)}. \tag{3.20}$$

Combining (3.17) and (3.20) while observing $t \leq r(t,\varepsilon)$ and $\frac{\ln(1/\varepsilon)}{\ln(e+\ln(1/\varepsilon)/t)} \leq \ln(1/\varepsilon)$, we get that

$$\forall \varepsilon \in (0,1)\forall t \in \mathbb{R}_+ \colon r(t,\varepsilon) = \Theta\left(t + \frac{\ln(1/\varepsilon)}{\ln(e + \ln(1/\varepsilon)/t)}\right). \tag{3.21}$$

Finally note that for $r_q := e^q t + \ln(1/\varepsilon)/q$, then we get

$$\left(\frac{t}{r_1}\right)^{r_q} \leq \left(e^{-q}\right)^{r_q} \leq e^{-\ln(1/\varepsilon)} = \varepsilon \qquad \Longrightarrow \qquad r(t,\varepsilon) \leq r_q.$$

□

This enables us to conclude the complexity of block-Hamiltonian simulation. Note that for $t \leq \varepsilon$ Hamiltonian simulation with ε-precision is trivial if $\|H\| \leq 1$, therefore we should assume that $t = \Omega(\varepsilon)$ in order to avoid this trivial situation. Apart from this we can conclude the complexity of block-Hamiltonian simulation for the entire range of interesting parameters.

3.4.5. Corollary (Complexity of block-Hamiltonian simulation). *Let $\varepsilon \in (0,\frac{1}{2})$, $t \in \mathbb{R}$ and $\alpha \in \mathbb{R}_+$. Let U be an $(\alpha, a, 0)$-block-encoding of the unknown Hamiltonian H. In order to implement an ε-precise Hamiltonian simulation unitary V which is an $(1, a+2, \varepsilon)$-block-encoding of e^{itH}, it is necessary and sufficient to use the unitary U a total number of times*

$$\Theta\left(\alpha|t| + \frac{\log(1/\varepsilon)}{\log(e + \log(1/\varepsilon)/(\alpha|t|))}\right).$$

Proof:
The upper bound follows from Theorem 3.4.3 and Lemma 3.4.4. The lower bound follows from the argument laid out in [LC17b] using Lemma 3.4.4. □

Note that the above corollary also covers the range $t \ll 1$, unlike the result of Low and Chuang [LC16] who assumed $t = \Omega(1)$. Also note that this result does not entirely match the complexity stated by Low and Chuang [LC17b, LC16]. For example in the case $t = \frac{\log(1/\varepsilon)}{\log(\log(1/\varepsilon))}$ the above corollary shows that the complexity is $\Theta\left(\frac{\log(1/\varepsilon)}{\log(\log(\log(1/\varepsilon)))}\right)$, whereas the expression of [LC17b, LC16] claims complexity $\mathcal{O}\left(\frac{\log(1/\varepsilon)}{\log(\log(1/\varepsilon))}\right)$.

The following lemma of Chakraborty et al. [CGJ19, Appendix A] helps us to understand error accumulation in Hamiltonian simulation, which enables us to present a slightly improved claim in Theorem 3.4.7.

3.4.6. Lemma. *Let $H, H' \in \mathbb{C}^{n\times n}$ be Hermitian operators, then*

$$\left\| e^{iH} - e^{iH'} \right\| \leq \|H - H'\|.$$

Now we prove a robust version of Theorem 3.4.3 using Lemma 3.4.6, and also substitute a simple expression of Lemma 3.4.4 bounding $r(t,\varepsilon)$ in order to get explicit constants.

3.4.7. Corollary. (Robust block-Hamiltonian simulation) *Let $t \in \mathbb{R}$, $\varepsilon \in (0,1)$ and let U be an $(\alpha, a, \varepsilon/|2t|)$-block-encoding of the Hamiltonian H. Then we can implement an ε-precise Hamiltonian simulation unitary V which is an $(1, a+2, \varepsilon)$-block-encoding of e^{itH}, with $10\alpha|t| + 15\log(16/\varepsilon)$ uses of U or its inverse, 3 uses of controlled-U or its inverse, using $\mathcal{O}(a(\alpha|t| + \log(2/\varepsilon)))$ two-qubit gates and using $\mathcal{O}(1)$ ancilla qubits.*

Proof:
Let $H' = \alpha(\langle 0|^{\otimes a} \otimes I)U(|0\rangle^{\otimes a} \otimes I)$, then $\|H' - H\| \leq \varepsilon/|2t|$. By Theorem 3.4.3 we can implement V, a $(1, a+2, \varepsilon/2)$-block-encoding of $e^{itH'}$, with $3r\big(e\alpha|t|, \frac{\varepsilon}{16}\big)$ uses of U or its inverse, 3 uses of controlled-U or its inverse and with $\mathcal{O}\big(ar\big(e\alpha|t|, \frac{\varepsilon}{16}\big)\big)$ two-qubit gates and using $\mathcal{O}(1)$ ancilla qubits. By Lemma 3.4.6 we get that V is an $(1, a+2, \varepsilon)$-block-encoding of e^{itH}. Finally by Lemma 3.4.4 choosing $q := \frac{1}{5}$ we get that

$$r\Big(e\alpha|t|, \frac{\varepsilon}{16}\Big) \leq e^q e\alpha|t| + \frac{\ln(16/\varepsilon)}{q} \leq \frac{10}{3}\alpha|t| + 5\ln(16/\varepsilon).$$

□

Finally, we prove a corollary about the complexity of controlled Hamiltonian simulation which can be useful, for example if one wants to use phase estimation.

3.4.8. Corollary. *Let $T \geq 2$ be a power of 2, $\varepsilon \in (0,1)$ and $\alpha = \Omega(1)$. Suppose that U is an $(\alpha, a, \varepsilon/(8T\log_2^2(T)))$-block-encoding of the Hamiltonian H. Then we can implement a unitary V which is an $(1, a+\mathcal{O}(1), \varepsilon)$-block-encoding of*

$$\sum_{t=0}^{T-1} |t\rangle\langle t| \otimes e^{itH},$$

with $\mathcal{O}\big(\alpha T + \log(T)\log(\frac{1}{\varepsilon})\big)$ uses of (controlled) U, $U^\dagger$ and $\mathcal{O}(a)$ times more two-qubit gates. (Here t is assumed to be represented as a binary number.)

Proof:
Let $T = 2^\theta$, and observe that due to the binary representation[19] $t = t_{\theta-1}t_{\theta-2}\ldots t_0$

$$\sum_{t=0}^{T-1} |t\rangle\langle t| \otimes e^{itH} = \prod_{\tau=0}^{\theta-1}\big(|0\rangle\langle 0|_\tau \otimes I + |1\rangle\langle 1|_\tau \otimes e^{i2^\tau H}\big).$$

We get the stated complexity by summing the complexities of implementing each term with precision $\varepsilon/(4\log_2^2(T))$ in the above product, using Corollary 3.4.7, and then combining the implementations using Corollary 3.3.12. □

[19]In the equation below for simplicity we omit the tensor factors of all but the relevant qubits in the binary expansion of t. Thereby there are "hidden" identity factors in the formula.

3.4.2 Bounded polynomial approximations of piecewise smooth functions

We begin by invoking a slightly surprising result showing how to efficiently approximate monomials on the interval $[-1,1]$ with essentially quadratically smaller degree polynomials than the monomial itself. The following theorem can be found in the survey of Sachdeva and Vishnoi [SV14, Theorem 3.3].

3.4.9. Theorem (Efficient approximation of monomials on [–1,1]). *For any positive integers s and d, there exists an efficiently computable degree-d polynomial $P_{s,d} \in \mathbb{R}[x]$ that satisfies*

$$\|P_{s,d}(x) - x^s\|_{[-1,1]} \leq 2e^{-d^2/(2s)}.$$

If one wants to approximate smooth functions on the entire $[-1,1]$ interval this result gives essentially quadratic savings. For example one can easily derive Corollary 3.4.10 using the above result, as shown in [SV14].

3.4.10. Corollary (Polynomial approximations of the exponential function). *Let $\beta \in \mathbb{R}_+$ and $\varepsilon \in (0, \frac{1}{2}]$. There exists an efficiently constructable polynomial $P \in \mathbb{R}[x]$ such that*

$$\left\|e^{-\beta(x+1)} - P(x)\right\|_{[-1,1]} \leq \varepsilon,$$

and the degree of P is $\mathcal{O}\left(\sqrt{\max\left[\beta, \log(\frac{1}{\varepsilon})\right] \log(\frac{1}{\varepsilon})}\right)$.

However, we often want to implement functions that are smooth only on some compact subset of $C \subseteq [-1,1]$, which requires different techniques. The main difficulty is to achieve a good approximation on C, while keeping the norm of the approximating polynomial bounded on the whole $[-1,1]$ interval. We overcome this difficulty by using Fourier approximations on C, which give rise to bounded functions naturally. Later we convert these Fourier series to a polynomial using Lemma 3.4.2 from the previous subsection.

Now we prove a useful technical result of [vAGGdW17, Lemma 37] about approximating smooth functions by low-weight Fourier series. By low weight we mean that the 1-norm of the coefficients is small. A notable property of the following result is that the bound on the sum of Fourier coefficients does not depend on the degrees of the polynomial terms. This can however be expected since the terms that have large degree make negligible contribution due to the restricted domain $x \in [-1+\delta, 1-\delta]$, and therefore we can drop them without loss of generality.

3.4.11. Lemma (Low-weight approximation by Fourier series). *Let $\delta, \varepsilon \in (0,1)$ and $f : \mathbb{R} \to \mathbb{C}$ s.t. $\left|f(x) - \sum_{k=0}^{K} a_k x^k\right| \leq \varepsilon/4$ for all $x \in [-1+\delta, 1-\delta]$. Then*

$\exists\, c \in \mathbb{C}^{2M+1}$ such that

$$\left| f(x) - \sum_{m=-M}^{M} c_m e^{\frac{i\pi m}{2}x} \right| \le \varepsilon$$

for all $x \in [-1+\delta, 1-\delta]$, where $M = \max\left(2\left\lceil \ln\left(\frac{4\|a\|_1}{\varepsilon}\right)\frac{1}{\delta}\right\rceil, 0\right)$ and $\|c\|_1 \le \|a\|_1$. Moreover c can be efficiently calculated on a classical computer in time poly$(K, M, \log(1/\varepsilon))$.

Proof:
Let us introduce the notation $\|f\|_\infty := \sup\{|f(x)| : x \in [-1+\delta, 1-\delta]\}$. First we consider the case when $\|a\|_1 < \varepsilon/2$. Then $\|f\|_\infty \le \left\| f(x) - \sum_{k=0}^{K} a_k x^k \right\|_\infty + \left\| \sum_{k=0}^{K} a_k x^k \right\|_\infty < \varepsilon/4 + \varepsilon/2 < \varepsilon$. So in this case the statement holds with $M = 0$ and $c = 0$, i.e., even with an empty sum.

From now on we assume $\|a\|_1 \ge \varepsilon/2$. We are going to build up our approximation gradually. Our first approximate function $\tilde{f}_1(x) := \sum_{k=0}^{K} a_k x^k$ satisfies $\left\| f - \tilde{f}_1 \right\|_\infty \le \varepsilon/4$ by assumption. In order to construct a Fourier series, we will work towards a linear combination of sines. To that end, note that $\forall x \in [-1,1]$: $\tilde{f}_1(x) = \sum_{k=0}^{K} a_k \left(\frac{\arcsin(\sin(x\pi/2))}{\pi/2}\right)^k$. Let $b^{(k)}$ denote the series of coefficients such that $\left(\frac{\arcsin(y)}{\pi/2}\right)^k = \sum_{\ell=0}^{\infty} b_\ell^{(k)} y^\ell$ for all $y \in [-1,1]$. For $k = 1$ the coefficients are just $\frac{2}{\pi}$ times the coefficients of the Taylor series of arcsin so we know that $b_{2\ell}^{(1)} = 0$ while $b_{2\ell+1}^{(1)} = \binom{2\ell}{\ell}\frac{2^{-2\ell}}{2\ell+1}\frac{2}{\pi}$. Since $\left(\frac{\arcsin(y)}{\pi/2}\right)^{k+1} = \left(\frac{\arcsin(y)}{\pi/2}\right)^k \left(\sum_{\ell=0}^{\infty} b_\ell^{(1)} y^\ell\right)$, we obtain the formula $b_\ell^{(k+1)} = \sum_{\ell'=0}^{\ell} b_{\ell'}^{(k)} b_{\ell-\ell'}^{(1)}$, so one can recursively calculate each $b^{(k)}$. As $b^{(1)} \ge 0$ one can use the above identity inductively to show that $b^{(k)} \ge 0$. Therefore $\left\| b^{(k)} \right\|_1 = \sum_{\ell=0}^{\infty} b_\ell^{(k)} 1^\ell = \left(\frac{\arcsin(1)}{\pi/2}\right)^k = 1$. Using the above definitions and observations we can rewrite

$$\forall x \in [-1,1] : \tilde{f}_1(x) = \sum_{k=0}^{K} a_k \sum_{\ell=0}^{\infty} b_\ell^{(k)} \sin^\ell(x\pi/2).$$

To obtain the second approximation function, we want to truncate the summation over ℓ at $L = \ln\left(\frac{4\|a\|_1}{\varepsilon}\right)\frac{1}{\delta^2}$ in the above formula. We first estimate the tail of the sum. We are going to use that for all $\delta \in [0,1]$: $\sin((1-\delta)\pi/2) \le 1 - \delta^2$. For all

$k \in \mathbb{N}$ and $x \in [-1+\delta, 1-\delta]$ we have:

$$\begin{aligned}
\left|\sum_{\ell=\lceil L\rceil}^{\infty} b_\ell^{(k)} \sin^\ell(x\pi/2)\right| &\leq \sum_{\ell=\lceil L\rceil}^{\infty} b_\ell^{(k)} \left|\sin^\ell(x\pi/2)\right| \\
&\leq \sum_{\ell=\lceil L\rceil}^{\infty} b_\ell^{(k)} \left|1-\delta^2\right|^\ell \\
&\leq \left(1-\delta^2\right)^L \sum_{\ell=\lceil L\rceil}^{\infty} b_\ell^{(k)} \\
&\leq \left(1-\delta^2\right)^L \\
&\leq e^{-\delta^2 L} \\
&= \frac{\varepsilon}{4\|a\|_1}.
\end{aligned}$$

Thus we have $\left\|\tilde{f}_1 - \tilde{f}_2\right\|_\infty \leq \varepsilon/4$ for

$$\tilde{f}_2(x) := \sum_{k=0}^{K} a_k \sum_{\ell=0}^{\lfloor L\rfloor} b_\ell^{(k)} \sin^\ell(x\pi/2).$$

To obtain our third approximation function, we will approximate $\sin^\ell(x\pi/2)$. First observe that

$$\sin^\ell(z) = \left(\frac{e^{-iz} - e^{iz}}{-2i}\right)^\ell = \left(\frac{i}{2}\right)^\ell \sum_{m=0}^{\ell} (-1)^m \binom{\ell}{m} e^{iz(2m-\ell)} \tag{3.22}$$

which, as we will show (for M' much larger than $\sqrt{\ell}$) is very well approximated by

$$\left(\frac{i}{2}\right)^\ell \sum_{m=\lceil \ell/2\rceil - M'}^{\lfloor \ell/2\rfloor + M'} (-1)^m \binom{\ell}{m} e^{iz(2m-\ell)}.$$

Truncating the summation in (3.22) based on this approximation reduces the maximal time evolution parameter (i.e., the maximal value of the parameter t in the $\exp(izt)$ terms) quadratically. To make this approximation precise, we use Chernoff's inequality [AS08, A.1.7] for the binomial distribution, or more precisely its corollary for sums of binomial coefficients, stating

$$\sum_{m=\lceil \ell/2 + M'\rceil}^{\ell} 2^{-\ell} \binom{\ell}{m} \leq e^{-\frac{2(M')^2}{\ell}}.$$

Let $M' = \left\lceil \ln\left(\frac{4\|a\|_1}{\varepsilon}\right)\frac{1}{\delta}\right\rceil$ and suppose $\ell \leq L$, then this bound implies that

$$\sum_{m=0}^{\lfloor \ell/2\rfloor - M'} 2^{-\ell}\binom{\ell}{m} = \sum_{m=\lceil \ell/2\rceil + M'}^{\ell} 2^{-\ell}\binom{\ell}{m} \leq e^{-\frac{2(M')^2}{\ell}} \leq e^{-\frac{2(M')^2}{L}} \leq \left(\frac{\varepsilon}{4\|a\|_1}\right)^2 \leq \frac{\varepsilon}{4\|a\|_1}, \tag{3.23}$$

where for the last inequality we use the assumption $\varepsilon \leq 2\|a\|_1$. By combining (3.22) and (3.23) we get that for all $\ell \leq L$

$$\left\| \sin^\ell(z) - \left(\frac{i}{2}\right)^\ell \sum_{m=\lceil \ell/2\rceil - M'}^{\lfloor \ell/2\rfloor + M'} (-1)^m \binom{\ell}{m} e^{iz(2m-\ell)} \right\|_\infty \leq \frac{\varepsilon}{2\|a\|_1}.$$

Substituting $z = x\pi/2$ into this bound we can see that $\left\|\tilde{f}_2 - \tilde{f}_3\right\|_\infty \leq \varepsilon/2$, for

$$\tilde{f}_3(x) := \sum_{k=0}^{K} a_k \sum_{\ell=0}^{\lfloor L\rfloor} b_\ell^{(k)} \left(\frac{i}{2}\right)^\ell \sum_{m=\lceil \ell/2\rceil - M'}^{\lfloor \ell/2\rfloor + M'} (-1)^m \binom{\ell}{m} e^{\frac{i\pi x}{2}(2m-\ell)}, \tag{3.24}$$

using $\sum_{k=0}^{K} |a_k| \sum_{\ell=0}^{\lfloor L\rfloor} \left|b_\ell^{(k)}\right| \leq \sum_{k=0}^{K} |a_k| = \|a\|_1$. Therefore we can conclude that $\tilde{f}_3$ is an ε-approximation to f:

$$\left\|f - \tilde{f}_3\right\|_\infty \leq \left\|f - \tilde{f}_1\right\|_\infty + \left\|\tilde{f}_1 - \tilde{f}_2\right\|_\infty + \left\|\tilde{f}_2 - \tilde{f}_3\right\|_\infty \leq \varepsilon.$$

Observe that in (3.24) the largest value of $|m - \ell|$ in the exponent is upper bounded by $2M' = M$. So by rearranging the terms in $\tilde{f}_3$ we can write $\tilde{f}_3(x) = \sum_{m=-M}^{M} c_m e^{\frac{i\pi m}{2}x}$. Now let us fix a value k in the first summation of (3.24). Observe that after taking the absolute value of each term, the last two summations still yield a value ≤ 1, since $\left\|b^{(k)}\right\|_1 = 1$ and $\sum_{m=0}^{\ell}\binom{\ell}{m} = 2^\ell$. It follows that $\|c\|_1 \leq \|a\|_1$. From the construction of the proof, it is easy to see that (an ε-approximation of) c can be calculated in time $\mathrm{poly}(K, M, \log(1/\varepsilon))$. □

Note that the above result can also be used to implement smooth functions of a Hamiltonian H, based on Fourier series decompositions via the Linear Combinations of Unitaries (LCU) Lemma [BCK15] as shown by van Apeldoorn et al. [vAGGdW17, Appendix B]. Accordingly the techniques developed in [vAGGdW17, Appendix B] access H only through controlled-Hamiltonian simulation. If we are given a block-encoding of H, then the simulation step can be omitted, and using a polynomial approximation of the function we can use singular value transformation techniques to directly implement the required transformation. However, a very recent result by Low [Low18] suggests that in some cases the controlled Hamiltonian simulation subroutine might be more efficient than converting the entire Hamiltonian to a block-encoding.

Our main idea here is to combine the above result with the polynomial approximation of the exponential function as in Lemma 3.4.2. The low weights are useful because they let us reduce the precision required for approximating the Fourier terms.

3.4.12. Corollary (Taylor series based bounded polynomial approximations). *Let $x_0 \in [-1,1]$, $r \in (0,2]$, $\delta \in (0,r]$ and let $f\colon [-x_0-r-\delta, x_0+r+\delta] \to \mathbb{C}$ and be such that $f(x_0+x)=\sum_{\ell=0}^{\infty} a_\ell x^\ell$ for all $x \in [-r-\delta, r+\delta]$. Suppose $B>0$ is such that $\sum_{\ell=0}^{\infty}(r+\delta)^\ell |a_\ell| \leq B$. Let $\varepsilon \in \left(0,\frac{1}{2B}\right]$, then there is an efficiently computable polynomial $P \in \mathbb{C}[x]$ of degree $\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{B}{\varepsilon}\right)\right)$ such that*

$$\|f(x)-P(x)\|_{[x_0-r,x_0+r]} \leq \varepsilon \tag{3.25}$$

$$\|P(x)\|_{[-1,1]} \leq \varepsilon + \|f(x)\|_{[x_0-r-\delta/2,x_0+r+\delta/2]} \leq \varepsilon + B \tag{3.26}$$

$$\|P(x)\|_{[-1,1]\setminus[x_0-r-\delta/2,x_0+r+\delta/2]} \leq \varepsilon. \tag{3.27}$$

Proof:
We proceed similarly to the proof of [vAGGdW17, Theorem 40]. Let $L(x) := \frac{x-x_0}{r+\delta}$ be the linear transformation taking $[x_0-r-\delta, x_0+r+\delta]$ to $[-1,1]$. Let $g(y) := f(L^{-1}(y))$, and $b_\ell := a_\ell (r+\delta)^\ell$ such that $g(y)=\sum_{\ell=0}^{\infty} b_\ell y^\ell$. Let $\delta' := \frac{\delta}{2(r+\delta)}$ and let $J = \left\lceil \frac{1}{\delta'}\log(\frac{12B}{\varepsilon})\right\rceil$, then for all $y \in [-1,1]$ we have that

$$\left|g(y) - \sum_{j=0}^{J-1} b_j y^j\right| = \left|\sum_{j=J}^{\infty} b_j y^j\right| \leq \sum_{j=J}^{\infty} |b_j (1-\delta')^j| \leq (1-\delta')^J \sum_{j=J}^{\infty} |b_j|$$
$$\leq (1-\delta')^J B \leq e^{-\delta' J} B \leq \frac{\varepsilon}{12}.$$

Now we construct a Fourier-approximation of g for all $y \in [-1+\delta', 1-\delta']$, with precision $\frac{\varepsilon}{3}$. Let $b' := (b_0, b_1, \ldots, b_{J-1})$ and observe that $\|b'\|_1 \leq \|b\|_1 \leq B$. We apply Lemma 3.4.11 to the function g, using the polynomial approximation corresponding to the truncation to the first J terms, i.e., using the coefficients in b'. Then we obtain a Fourier $\frac{\varepsilon}{3}$-approximation $\tilde{g}(y) := \sum_{m=-M}^{M} \tilde{c}_m e^{\frac{i\pi m}{2} y}$ of g, with

$$M = \mathcal{O}\left(\frac{1}{\delta'}\log\left(\frac{\|b'\|_1}{\varepsilon'}\right)\right) = \mathcal{O}\left(\frac{r}{\delta}\log\left(\frac{B}{\varepsilon}\right)\right),$$

such that the vector of coefficients $\tilde{c} \in \mathbb{C}^{2M+1}$ satisfies $\|\tilde{c}\|_1 \leq \|b'\|_1 \leq \|b\|_1 \leq B$. Let

$$\tilde{f}(x) := \tilde{g}(L(x)) = \tilde{g}\left(\frac{x-x_0}{r+\delta}\right) = \sum_{m=-M}^{M} \tilde{c}_m e^{\frac{i\pi m}{2(r+\delta)}(x-x_0)} = \sum_{m=-M}^{M} \tilde{c}_m e^{-\frac{i\pi m}{2(r+\delta)}x_0} e^{\frac{i\pi m}{2(r+\delta)}x}.$$

Since $g(y) = f(L^{-1}(y))$ we have that $f(x) = g(L(x))$ thus we can see that $\tilde{f}$ is an $\frac{\varepsilon}{3}$-precise Fourier approximation of f on the interval $[x_0-r-\frac{\delta}{2}, x_0+r+\frac{\delta}{2}]$.

Now we define $\tilde{P}$ as the polynomial that we get by replacing each of the Fourier terms $e^{\frac{i\pi m}{2(r+\delta)}x}$ by $\frac{\varepsilon}{3B}$-approximating polynomials given by Lemma 3.4.2. Using a tiny rescaling we can assure that the polynomial approximations of $e^{\frac{i\pi m}{2(r+\delta)}x}$ have absolute value at most 1 on $[-1,1]$. Moreover by Lemma 3.4.4 we know that the degree of these polynomials are $\mathcal{O}\big(\frac{M}{r+\delta}+\log\big(\frac{B}{\varepsilon}\big)\big)=\mathcal{O}\big(\frac{1}{\delta}\log\big(\frac{B}{\varepsilon}\big)\big)$. Since $\|\tilde{c}\|\leq B$, we get that the absolute value of the polynomial $\tilde{P}$ is bounded by B on the interval $[-1,1]$. Finally we define P as the product of $\tilde{P}$ and an approximation polynomial of the rectangle function that is $\frac{\varepsilon}{3B}$-close to 1 on the interval $[x_0-r,x_0+r]$, and is $\frac{\varepsilon}{3B}$-close to 0 on the interval $[-1,1]\setminus[x_0-r-\frac{\delta}{2},x_0+r+\frac{\delta}{2}]$, finally which is bounded by 1 on the interval $[-1,1]$ in absolute value. By Lemma 3.2.6 we can construct such a polynomial of degree $\mathcal{O}\big(\frac{1}{\delta}\log\big(\frac{B}{\varepsilon}\big)\big)$. As we can see P has degree $\mathcal{O}\big(\frac{1}{\delta}\log\big(\frac{B}{\varepsilon}\big)\big)$, and by construction satisfies the properties (3.25)-(3.27). □

Combining this polynomial approximation result with Theorem 3.4.1 we can efficiently implement smooth functions of Hermitian matrices. As an application, motivated by the work of Chakraborty et al. [CGJ19] we show how to construct low-degree polynomial approximations of power functions.

3.4.13. Corollary (Polynomial approximations of negative power functions). *Let $\delta,\varepsilon\in(0,\frac{1}{2}]$, $c>0$ and let $f(x):=\frac{\delta^c}{2}x^{-c}$, then there exist even/odd polynomials $P,P'\in\mathbb{R}[x]$, such that $\|P-f\|_{[\delta,1]}\leq\varepsilon$, $\|P\|_{[-1,1]}\leq 1$ and similarly $\|P'-f\|_{[\delta,1]}\leq\varepsilon$, $\|P'\|_{[-1,1]}\leq 1$, moreover the degrees of the polynomials are $\mathcal{O}\Big(\frac{\max[1,c]}{\delta}\log\big(\frac{1}{\varepsilon}\big)\Big)$.*

Proof:
First note that for all $y\in(-1,1)$ we have that $(1+y)^{-c}=\sum_{k=0}^{\infty}\binom{-c}{k}y^k$. We first find a polynomial $\tilde{P}\in\mathbb{C}[x]$ such that $\Big\|\tilde{P}-f\Big\|_{[\delta,1]}\leq\frac{\varepsilon}{2}$, $\Big\|\tilde{P}\Big\|_{[-1,0]}\leq\frac{\varepsilon}{2}$ and $\Big\|\tilde{P}\Big\|_{[-1,1]}\leq 1$. We construct such a polynomial of degree $\mathcal{O}\Big(\frac{\max[1,c]}{\delta}\log\big(\frac{1}{\varepsilon}\big)\Big)$ using Corollary 3.4.12, with $x_0:=1$, $r:=1-\delta$, $\delta':=\frac{\delta}{2\max[1,c]}$ and $B:=1$. The choice of B is justified by the observation that

$$\begin{aligned}\frac{\delta^c}{2}\sum_{k=0}^{\infty}\left|\binom{-c}{k}\right|(r+\delta')^k=\frac{\delta^c}{2}\sum_{k=0}^{\infty}\binom{-c}{k}(-r-\delta')^k=\frac{\delta^c}{2}(1-r-\delta')^{-c}\\ =\frac{\delta^c}{2}(\delta-\delta')^{-c}=\frac{1}{2}\left(1-\frac{\delta'}{\delta}\right)^{-c}=\frac{1}{2}\left(1-\frac{1}{2\max[1,c]}\right)^{-c}\leq 1.\end{aligned}$$

Finally, we define P as the even real part of $\tilde{P}$, and define P' as the odd real part of $\tilde{P}$. □

Given a $(1,a,0)$-block-encoding of A, with the promise that the spectrum of A lies in $[\delta,1]$, using the above polynomials and Theorem 3.4.1 we can implement a $(1, a+2, \varepsilon)$-block-encoding of $\frac{\delta^c}{2}A^{-c}$ with $\mathcal{O}\left(\frac{\max[1,c]}{\delta}\log\left(\frac{1}{\varepsilon}\right)\right)$ uses of the block-encoding of A. Since the derivative of the function $\frac{\delta^c}{2}x^{-c}$ at $x=\delta$ is $-\frac{c}{2\delta}$, we get by Theorem 3.5.1 that the δ and c dependence of the complexity of this procedure is optimal.

Similarly to the case of negative powers we can also construct polynomials approximating positive power functions. We only show how to do this for exponents which are at most 1, since if we have some exponent c we can always take the fractional part $c - \lfloor c \rfloor$, approximate the corresponding power function, and multiply with the monomial $x^{\lfloor c \rfloor}$ to get a polynomial approximation of the original power function x^c.

3.4.14. Corollary (Polynomial approximations of positive power functions). *Let $\delta, \varepsilon \in (0, \frac{1}{2}]$, $c \in (0,1]$ and let $f(x) := \frac{1}{2}x^c$, then there exist even/odd polynomials $P, P' \in \mathbb{R}[x]$, such that $\|P - f\|_{[\delta,1]} \leq \varepsilon$, $\|P\|_{[-1,1]} \leq 1$ and similarly $\|P' - f\|_{[\delta,1]} \leq \varepsilon$, $\|P'\|_{[-1,1]} \leq 1$, moreover the degree of the polynomials are $\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{1}{\varepsilon}\right)\right)$.*

Proof:
First note that for all $y \in [-1,1]$ we have that $(1+y)^c = \sum_{k=0}^{\infty}\binom{c}{k}y^k$. We first find a polynomial $\tilde{P} \in \mathbb{C}[x]$ such that $\left\|\tilde{P} - f\right\|_{[\delta,1]} \leq \frac{\varepsilon}{2}$, $\left\|\tilde{P}\right\|_{[-1,0]} \leq \frac{\varepsilon}{2}$ and $\left\|\tilde{P}\right\|_{[-1,1]} \leq 1$. We construct such a polynomial of degree $\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{1}{\varepsilon}\right)\right)$ using Corollary 3.4.12, with choosing $x_0 := 1$, $r := 1-\delta$, $\delta' := \frac{\delta}{2}$ and $B := 1$. The choice of B is justified by the observation that

$$\frac{1}{2}\sum_{k=0}^{\infty}\left|\binom{c}{k}\right|(r+\delta)^k = \frac{1}{2}\sum_{k=0}^{\infty}\left|\binom{c}{k}\right| = \frac{1}{2} - \frac{1}{2}\sum_{k=1}^{\infty}\binom{c}{k}(-1)^k$$
$$= 1 - \sum_{k=0}^{\infty}\binom{c}{k}(-1)^k = 1 - f(1-1) = 1.$$

Finally, we define P as the even real part of $\tilde{P}$, and define P' as the odd real part of $\tilde{P}$. □

Finally we develop a theorem that is analogous to [vAGGdW17, Corollary 42], and shows that any function that has quickly converging local Taylor series, can be approximated with polynomials whose degree scales logarithmically with the required precision. This result is useful when the function is piecewise smooth, but does not have a single Taylor series covering the whole region of interest.

3.4.15. Theorem (Bounded polynomial approximation: multiple Taylor series). *Let $J \in \mathbb{N}$, $(x_j, r_j, \delta_j) \in [-1,1] \times (0,2] \times (0,1]$, such that $(x_j : j \in [J])$ is monotone increasing, and $\delta_j \leq r_j$ for all $j \in [J]$. Let $I := \bigcup_{j\in[J]}[x_j - r_j, x_j + r_j]$ be the* *union of the intervals $[x_j - r_j, x_j + r_j]$, and suppose that for all $i < j \in [J]$ such that $j - i \geq 2$ we have that $r_j + r_j < x_j - x_j$. Let*

$$\delta = \min\left[\min_{j\in[J]} \delta_j, \min_{j\in[J-1]} |x_{j+1} - x_j - (r_{j+1} + r_j)|\right].$$

Let $f : I + [-\frac{\delta}{2}, \frac{\delta}{2}] \to \mathbb{C}$, $B \in \mathbb{R}_+$ be such that for all $j \in [J]$ we have $f(x_j + x) = \sum_{k=0}^{\infty} a_k^{(j)} x^k$ for all $x \in [x_j - r_j - \frac{\delta_j}{2}, x_j + r_j + \frac{\delta_j}{2}]$ and $\sum_{k=0}^{\infty}(r_j + \delta_j)^k |a_k^{(j)}| \leq B$. Let $\varepsilon \in \left(0, \frac{1}{2BJ}\right]$, then there is an efficiently computable polynomial $P \in \mathbb{C}[x]$ of degree $\mathcal{O}\left(\frac{J}{\delta}\log\left(\frac{BJ}{\varepsilon}\right)\right)$ such that

$$\begin{aligned} \|f(x) - P(x)\|_I &\leq \varepsilon \\ \|P(x)\|_{[-1,1]} &\leq \|f(x)\|_{I+[-\delta/2,\delta/2]} \\ \|P(x)\|_{[-1,1]\setminus(I+[-\delta/2,\delta/2])} &\leq \varepsilon. \end{aligned}$$

Proof:
Use Corollary 3.4.12 to construct polynomials $f_j \colon j \in [J]$ of degree $\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{BJ}{\varepsilon}\right)\right)$ such that

$$\begin{aligned} \|f(x) - f_j(x)\|_{[x_j-r_j,x_j+r_j]} &\leq \frac{\varepsilon}{4J} \\ \|f_j(x)\|_{[-1,1]} &\leq \|f(x)\|_{I+[-\delta/2,\delta/2]} \\ \|f_j(x)\|_{[-1,1]\setminus([x_j-r_j,x_j+r_j]+[-\delta/2,\delta/2])} &\leq \varepsilon. \end{aligned}$$

Let us introduce a notation for the union of the intervals $[x_i - r_i, x_i + r_i]$ as

$$I_{[j,k]} := \bigcup_{i\in\{j,j+1,\ldots,k\}} [x_i - r_i, x_i + r_i].$$

We show inductively how to construct polynomials $f_{[j,k]}$ of degree $\mathcal{O}\left(\frac{k-j+1}{\delta}\log\left(\frac{BJ}{\varepsilon}\right)\right)$ such that

$$\left\|f(x) - f_{[j,k]}(x)\right\|_{I_{[j,k]}} \leq \frac{2(k-j+1)\varepsilon}{2J} \tag{3.28}$$

$$\left\|f_{[j,k]}(x)\right\|_{[-1,1]} \leq \|f(x)\|_{I+[-\delta/2,\delta/2]} \tag{3.29}$$

$$\left\|f_{[j,k]}(x)\right\|_{[-1,1]\setminus\left(I_{[j,k]}+[-\delta/2,\delta/2]\right)} \leq \varepsilon. \tag{3.30}$$

We already showed how to construct $f_{[j,j]} := f_j \colon j \in [J]$. Suppose that we already constructed $f_{[1,j]}$, then we construct $f_{[1,j+1]}$ as follows. We take a polynomial $S(x)$ of degree $\mathcal{O}\left(\frac{1}{\delta}\log\left(\frac{BJ}{\varepsilon}\right)\right)$ that approximates the shifted sign function s.t. $\left\|S(x) - \mathrm{sign}\left(x - \frac{x_i+x_j}{2}\right)\right\|_{[-1,1]\setminus\left[\frac{x_i+x_j-\delta}{2},\frac{x_i+x_j+\delta}{2}\right]} \leq \frac{\varepsilon}{8BJ}$, moreover $\|S\|_{[-1,1]} \leq 1$. Then we define $f_{[1,j+1]} := \frac{1-S(x)}{2} f_{[1,j]} + \frac{1+S(x)}{2} f_{[j+1,j+1]}$. It satisfies (3.29)-(3.30), since $f_{[1,j+1]}$ is a point-wise convex combination of $f_{[1,j]}$ and $f_{[j+1,j+1]}$. Similarly

(3.28) is also easy to verify. Therefore by induction we can finally construct $P := f_{[1,J]}$, which satisfies (3.28)-(3.30) and therefore also the requirements of the theorem.[20] $\square$

A direct corollary of this theorem is for example that for all $\varepsilon \in (0, \frac{1}{2}]$ the function $\frac{\delta}{x}$ can be ε-approximated on the domain $[-1,1] \setminus [-\delta, \delta]$ with a polynomial of degree $\mathcal{O}\big(\frac{1}{\delta}\log\big(\frac{1}{\varepsilon}\big)\big)$. Although this also follows from Corollary 3.4.13, we prove it directly using Theorem 3.4.15, in order to illustrate the versatility of this Theorem 3.4.15.

3.4.16. Corollary (Polynomial approximations of $\frac{1}{x}$). *Let $\varepsilon, \delta \in (0, \frac{1}{2}]$, then there is an odd polynomial $P \in \mathbb{R}[x]$ of degree $\mathcal{O}\big(\frac{1}{\delta}\log\big(\frac{1}{\varepsilon}\big)\big)$ that is ε-approximating $f(x) = \frac{3}{4}\frac{\delta}{x}$ on the domain $I = [-1,1] \setminus [-\delta, \delta]$, moreover it is bounded by 1 in absolute value.*

Proof:
Take $J := 2$, $(x_1 := -1, r_1 := 1-\delta, \delta_1 := \frac{\delta}{2})$, $(x_2 := 1, r_2 := 1-\delta, \delta_2 := \frac{\delta}{2})$ and $B = 1$ in Theorem 3.4.15, observing that $f(1+x) = \frac{3\delta}{4}\sum_{k=0}^{\infty} -(1)^k x^k = -f(-1+x)$. Define P as the odd real part of the polynomial given by Theorem 3.4.15. $\square$

3.4.3 Applications: fractional queries and Gibbs-sampling

Scott Aaronson listed as one of "The ten most annoying questions in quantum computing" [Aar06] the following problem: given a unitary U, can we implement $\sqrt{U}$? This was positively answered by Sheridan et al. [SMM09]. We show how to improve the complexity of the result of Sheridan et al. exponentially in terms of the error dependence. We proceed following ideas of Low and Chuang [LC17a].

Suppose that we have access to a unitary $U = e^{iH}$, where H is a Hamiltonian of norm at most $\frac{1}{2}$. Low and Chuang [LC17a] showed how to get a $(1, 2, \varepsilon)$-block-encoding of H with $\mathcal{O}\big(\log\big(\frac{1}{\varepsilon}\big)\big)$ uses of U. We reprove this result; our proof becomes quite simple thanks to Corollary 3.4.12.

3.4.17. Lemma (Central polynomial approximations of the function $\arcsin(x)$). *For all $\delta, \varepsilon \in (0, \frac{1}{2}]$ there is an efficiently computable odd real polynomial $P \in \mathbb{R}[x]$ of degree at most $\big\lceil \frac{1}{\delta}\ln\big(\frac{1}{\varepsilon}\big)\big\rceil$ such that $\|P\|_{[-1,1]} \leq 1$ and*

$$\left\| P(x) - \frac{2}{\pi}\arcsin(x) \right\|_{[-1+\delta, 1-\delta]} \leq \varepsilon.$$

[20]Note that this approach could be further improved to produce an approximating polynomial of degree $\mathcal{O}\Big(\frac{\log(J)}{\delta}\log\Big(\frac{B\log(J)}{\varepsilon}\Big)\Big)$ by combining the polynomial approximations on the different intervals in a binary tree structure. Since $J = \mathcal{O}\big(\frac{1}{\delta}\big)$, we have $\log(J) = \log\big(\frac{1}{\delta}\big)$ and then this gives at most a logarithmic overhead.

Proof:
Observe that $\frac{2}{\pi}\arcsin(x) = \sum_{\ell=0}^{\infty}\binom{2\ell}{\ell}\frac{2^{-2\ell}}{2\ell+1}\frac{2}{\pi}x^{2\ell+1}$ for all $x \in [-1,1]$. Therefore we also have $\sum_{\ell=0}^{\infty}\left|\binom{2\ell}{\ell}\frac{2^{-2\ell}}{2\ell+1}\frac{2}{\pi}\right| = \frac{2}{\pi}\arcsin(1) = 1$. Thus for $K = \left\lceil\frac{1}{\delta}\ln\left(\frac{1}{\varepsilon}\right)\right\rceil$ the truncated series $P(x) := \sum_{\ell=0}^{K}\binom{2\ell}{\ell}\frac{2^{-2\ell}}{2\ell+1}\frac{2}{\pi}x^{2\ell+1}$ is an approximating polynomial satisfying all the requirements. $\square$

3.4.18. COROLLARY (Implementing the logarithm of unitaries). *Suppose that $U = e^{iH}$, where H is a Hamiltonian of norm at most $\frac{\pi}{2}-\delta$ for some $\delta \in (0,1)$. Let $\varepsilon \in (0,\frac{1}{2}]$, then we can implement a $(\frac{\pi}{2},2,\varepsilon)$-block-encoding of H with $\left\lceil\frac{8}{\delta^2}\ln\left(\frac{1}{\varepsilon}\right)\right\rceil$ uses of controlled-U and its inverse, using $\mathcal{O}\left(\frac{1}{\delta^2}\log\left(\frac{1}{\varepsilon}\right)\right)$ two-qubit gates and using a single ancilla qubit.*

Proof:
Let cU denote the controlled version of U controlled by the first qubit. Then

$$\sin(H) = -i(\langle +| \otimes I)cU^\dagger(ZX \otimes I)cU(|+\rangle \otimes I).$$

Now we apply singular value transformation (Corollary 2.3.8) using an ε-approximating polynomial of $\frac{2}{\pi}\arcsin(x)$ on the domain $[-1+\frac{\delta^2}{4}, 1-\frac{\delta^2}{4}]$. $\square$

Combining the above result with block-Hamiltonian simulation Corollary 3.4.7 we can implement fractional queries of unitaries with complexity $\mathcal{O}\left(\log^2\left(\frac{1}{\varepsilon}\right)\right)$. As we show in the following corollary this complexity can be reduced to $\mathcal{O}\left(\log\left(\frac{1}{\varepsilon}\right)\right)$ by directly implementing[21] Hamiltonian simulation using a block-encoding of $\sin(H)$ rather than H.

3.4.19. COROLLARY (Implementing fractional queries). *Suppose that $U = e^{iH}$, where H is a Hamiltonian of norm at most 1. Let $\varepsilon \in (0,\frac{1}{2}]$ and $t \in [-1,1]$, then we can implement an ε-approximation of $U^t = e^{itH}$ with $\mathcal{O}\left(\log\left(\frac{1}{\varepsilon}\right)\right)$ uses of controlled-U and its inverse, using $\mathcal{O}\left(\log\left(\frac{1}{\varepsilon}\right)\right)$ two-qubit gates and using $\mathcal{O}(1)$ ancilla qubits.*

Proof:
As we have shown in the proof of Corollary 3.4.18, one can implement a block-encoding of $\sin(H)$ with a constant number of queries to U. Let us look at the Taylor series of $e^{it\arcsin(x)}$. One can see that the 1-norm of the coefficients of the Taylor series of $t\arcsin(x)$ is $|t|\arcsin(1) = |t|\frac{\pi}{2}$. Therefore, for $t \in [-\frac{2}{\pi},\frac{2}{\pi}]$ we get that the 1-norm of the Taylor series of $e^{it\arcsin(x)}$ is at most $e^1 = e$. Thereby, using Theorem 3.4.15 we can construct polynomial $\mathcal{O}(\varepsilon)$-approximations of $\sin(t\arcsin(x))$

[21]The method we describe uses the block-encoding formalism, but in fact one could implement it more directly using a Fourier series based approach similarly to the one used for Hamiltonian simulation by Low and Chuang [LC17b].

and $\cos(t\arcsin(x))$ on the interval $[\sin(-1),\sin(1)]$ of degree $\mathcal{O}\big(\log\big(\frac{1}{\varepsilon}\big)\big)$, which are bounded by 1 in absolute value on the interval $[-1,1]$. We can combine these polynomials in a similar way as in the proof of Theorem 3.4.3. This way we can implement an ε-approximation of U^t for all $t\in[-\frac{2}{\pi},\frac{2}{\pi}]$ with complexity $\mathcal{O}\big(\log\big(\frac{1}{\varepsilon}\big)\big)$. Implementing U^t for all $t\in[-1,1]$ can be achieved by implementing $U^{\frac{t}{2}}$ twice and taking their product. □

Note that in the above corollary it is not essential that $\|H\|\leq 1$. If we would be promised that all eigenvalues lie in say the interval $[1,3]$, that would be fine as well. We could just use the procedure of the corollary but applied to the unitary $e^{-2i}U$, and multiply the resulting unitary with e^{2it}. Pushing this observation even further, the above technique can be combined with an initial phase estimation in order to implement fractional queries under the weaker promise $\|H\|\leq\pi-\delta$. It suffices to perform a δ-precise phase estimation with success probability $1-\mathrm{poly}(\varepsilon)$, then use the approximate knowledge of the eigenvalues of H to implement fractional queries using Corollary 3.4.19 and finally undo the initial phase estimation. This leads to complexity $\mathcal{O}\big(\frac{1}{\delta}\log\big(\frac{1}{\varepsilon}\big)\big)$, which exponentially improves the complexity $\mathcal{O}\big(\max[\frac{1}{\delta},\frac{1}{\varepsilon}]\log\big(\frac{1}{\varepsilon}\big)\big)$ of Sheridan et al. [SMM09] in the case of $\delta=\Theta(1)$. We note that Sheridan et al. [SMM09] also proved a lower bound on this problem, which shows that the δ-dependence of this algorithm is actually optimal. We believe that the $\log(\frac{1}{\varepsilon})$ dependence in the runtime is also necessary, therefore this algorithm is probably fairly close to optimal.

After implementing a fractional query, assuming that $\|H\|\leq 1$ is satisfied, one can use Corollary 3.4.19 to implement the logarithm of the unitary. Also note the gap promise on the spectrum of U is necessary for implementing the fractional queries, but it is not important that the gap is exactly at $e^{i\pi}$, one can for example just add a phase gate to U in order to rotate the spectrum.

Gibbs-sampling. Here we describe how these techniques can be used for Gibbs-sampling. If one first prepares a maximally entangled state on two registers and applies the map $e^{-\frac{\beta}{2}(H+I)}$ on the first register, then one gets a subnormalized Gibbs state $e^{-\beta(H+I)}$ on the first register. Then, using (fixed-point) amplitude amplification one gets a purification of a Gibbs state. Each of these steps can be compactly performed using singular value transformation techniques, providing an efficient implementation.

An ε-approximation of the map $e^{-\frac{\beta}{2}(H+I)}$ can be implemented using Theorem 3.4.1 and Corollary 3.4.10 with query complexity $\mathcal{O}\big(\sqrt{\beta}\log(1/\epsilon)\big)$, and it suffices to use $\mathcal{O}\Big(\sqrt{\frac{n}{\mathcal{Z}}}\Big)$ amplitude amplification steps in order to prepare the Gibbs state with constant success probability, where n is the dimension of H and $\mathcal{Z}:=\mathrm{Tr}\big[e^{-\beta(H+I)}\big]$ is the partition function. In the case when H does not have an eigenvalue close to -1, but say $\lambda_{\min}$ is the smallest eigenvalue, then one should implement an approximation of $e^{-\beta(H-\lambda_{\min}I)}$ on the domain $[\lambda_{\min},1]$ in order to avoid unnecessary subnormalization, ensuring that $\mathcal{Z}' := \mathrm{Tr}\left[e^{-\beta(H-\lambda_{\min}I)}\right] = \Omega(1)$. However note, that this restricted domain in general increases the complexity and yields a linear dependence on β.

Now we derive our Gibbs-sampler with worst-case guarantees, having cost proportional to $\sqrt{n}$. The following lemma is our first step, showing how to efficiently implement the function $e^{-\beta H}$ for a block-encoding of $0 \preceq H \preceq I$.

3.4.20. Lemma. *Let $\beta > 0$, and $\varepsilon \in (0, 1/40)$. There is polynomial P of degree $\mathcal{O}((1+\beta)\log(1/\varepsilon))$ such that $\|P(x)\|_{[-1,1]} \leq 1/2$ and*

$$\left\|P(x) - e^{-\beta x}/e\right\|_{[0,1]} \leq \varepsilon.$$

Proof:
We apply Corollary 3.4.12 to the function $f(x) = e^{-\beta x}/e$, with setting $x_0 = 1$, $r = 1$, $\delta = \min\{1, \frac{1}{4\beta}\}$ and $B = \frac{19}{40}$. Since

$$f(1+x) = e^{-\beta(1+x)}/e = e^{-\beta-1}e^{-\beta x} = e^{-\beta-1}\sum_{k=0}^{\infty}\frac{(\beta x)^n}{n!}$$

the choice of B is justified by

$$B = \frac{19}{40} \geq e^{-3/4} = e^{-\beta-1}e^{\beta+1/4} = e^{-\beta-1}\sum_{k=0}^{\infty}\frac{(\beta(1+1/(4\beta)))^n}{n!}.$$

□

Now we show how to use the above polynomial in quantum singular value transformation followed by (fixed-point) amplitude amplification to prepare a Gibbs state with cost depending logarithmically on the precision parameter. To our knowledge no prior Gibbs-sampler achieved logarithmic dependence on the precision parameter without assuming access to the entries of $\sqrt{H}$ as in [CS17]. This can yield a significant reduction in the complexity for SDP-solving; for more details see Chapter 6.

3.4.21. Theorem. *Let $\varepsilon \in (0, 1/2)$, $\beta = \Omega(1)$ and suppose that U is an a-qubit block-encoding of the Hamiltonian $H \in \mathbb{C}^{n\times n}$. Then we can prepare a pure state on two registers, so that tracing out the first register yields a density operator ε-close to the Gibbs state $e^{-\beta H}/\mathrm{Tr}\left[e^{-\beta H}\right]$ in trace distance, with*

$$\mathcal{O}\left(\beta\sqrt{n}\log^3\left(\frac{n}{\varepsilon}\right)\right)$$

uses of (controlled) U, $U^\dagger$ and $\mathcal{O}(a)$ times more two-qubit gates.

Proof:
First we show how to prepare a purified sub-normalized Gibbs state. Then we use (fixed-point) amplitude amplification to "postselect" on this sub-normalized state in a similar fashion as in Algorithm 3.1. There is a possible caveat here: if we postselect on a state with norm q, then it gets rescaled by $1/q$ and its preparation error is rescaled by $1/q$ as well, therefore we need to work with increased precision.

Let $H=\sum_{j=1}^{n} E_j|\phi_j\rangle\langle\phi_j|$, where $\{|\phi_j\rangle : j \in [n]\}$ is an orthonormal eigenbasis of H and $E_i \leq E_j$ for $i \leq j$. We begin with computing $\tilde{E}_1$, a $\frac{1}{\beta}$-precise approximation of E_1 using Lemma 3.A.4 and boost the success probability to $1-\mathcal{O}(\varepsilon)$ by $\mathcal{O}(\log(1/\varepsilon))$ repetitions. Using Lemma 3.3.9 this enables us to construct a block-encoding of $H' := \frac{H-(\tilde{E}_1-1/\beta)I}{2}$. Since the estimation procedure succeeds with probability $1-\mathcal{O}(\varepsilon)$, from now on we assume without loss of generality that $0 \preceq H' \not\preceq \frac{1}{\beta}I$.

Now we describe how to approximately prepare a purified sub-normalized Gibbs state $e^{-2\beta H'-2I}/n$. Due to the invariance of maximally entangled states under transformations of the form $Q\otimes Q^*$ for unitary Q, there is an orthonormal basis $\{|\upsilon_j\rangle : j \in [n]\}$ such that

$$\frac{1}{\sqrt{n}}\sum_{j=0}^{n-1}|j\rangle_A|j\rangle_B = \frac{1}{\sqrt{n}}\sum_{j=1}^{n}|\upsilon_j\rangle_A|\phi_j\rangle_B. \tag{3.31}$$

Suppose we can implement a unitary U such that $(\langle 0|\otimes I)U(|0\rangle\otimes I) = e^{-\beta H'-I}$. If we apply U to the B-register of the state (3.31), then we get a state $|\gamma\rangle$ such that $\mathrm{Tr}_A((\langle 0|\otimes I)|\gamma\rangle\langle\gamma|(|0\rangle\otimes I)) = e^{-2\beta H'-2I}/n$.

If we implement $e^{-\beta H'-2I}$ with precision $\mathcal{O}(\varepsilon/n^2)$, then the subnormalized Gibbs state will have error $\mathcal{O}(\varepsilon/n)$ in trace distance. Since $\mathrm{Tr}\big[e^{-2\beta H'-2I}/n\big] = \Omega(1/n)$, after postselection the normalized approximate Gibbs state will be have error at most $\mathcal{O}(\varepsilon)$ in trace distance. Moreover, the postselection can be performed with success probability $1-\mathcal{O}(\varepsilon)$ using $\mathcal{O}(\sqrt{n}\log(1/\varepsilon))$ steps of fixed-point amplitude amplification (Theorem 3.2.4).

The query complexity of boosted ground state energy estimation is upper bounded by $\mathcal{O}\big(\beta\sqrt{n}\log(n)\log(1/\varepsilon)+\log(\beta)\sqrt{n}\log^2(n)\log(1/\varepsilon)\big)$, as shown by Lemma 3.A.4. The query complexity of implementing the block-encoding of H' is 1, and the gate complexity is $\mathcal{O}(\log(1/\varepsilon))$. Due to Lemma 3.4.20 and Theorem 3.4.1, an $(1, a+\mathcal{O}(1), \mathcal{O}(\varepsilon/n^2))$-block-encoding of $e^{-\beta H'-I}$ can be implemented using $\mathcal{O}(\beta\log(n/\varepsilon))$ queries and at most $\mathcal{O}(a\log(1/\varepsilon))$-times more two-qubit gates. Finally fixed-pint amplitude amplification multiplies the cost with at most $\mathcal{O}(\sqrt{n}\log(1/\varepsilon))$. Therefore we can upper bound the query complexity by $\mathcal{O}\big(\beta\sqrt{n}\log(n/\varepsilon)\log(1/\varepsilon)+\log(\beta)\sqrt{n}\log^2(n)\log(1/\varepsilon)\big)$, while the gate complexity can be upper bounded by $\mathcal{O}\big(a\beta\sqrt{n}\log^2(n/\varepsilon)\log(1/\varepsilon)\big)$. □

Finally, note that in the special case when H has an eigenvalue $\mathcal{O}(1/\beta)$-close to -1, then a quadratic improvement can be achieved in β using Corollary 3.4.10.

Similarly, if one has access to the square root of H, and H has an eigenvalue close to 0, then one can achieve quadratically improved scaling with β as observed by Chowdhury and Somma [CS17]. This can be also easily shown using our techniques by noting that $e^{-\beta H} = e^{-\beta\left(\sqrt{H}\right)^2}$, and that the function $e^{-\beta x^2}$ can be ε-approximated on the interval $[-1,1]$ using a polynomial of degree $\mathcal{O}\big(\sqrt{\beta}\log\big(\frac{1}{\varepsilon}\big)\big)$ as follows from[22] Corollary 3.4.10.

3.5 Limitations of smooth function techniques

In the classical literature there are many good techniques for lower bounding the degrees of approximation polynomials [SV14]. There is an intimate relationship between the degrees of approximation polynomials and quantum query complexity [BBC+01]. In a recent result Arunachalam et al. [ABP17] showed that for discrete problems certain polynomial approximations characterize the quantum query complexity. There are also some results about lower bounds for continuous problems [Aar09, Bel15, GAW19], however the literature on this is much more sparse.

To advance the knowledge on lower bounds in the continuous regime, we prove a conceptually simple lower bound on eigenvalue transformations, which guides our intuition about what sort of transformations are possible. Intuitively speaking if a function has derivative d on the domain of interest then we need to use the block-encoding $\Omega(d)$-times in order to implement the eigenvalue transformation corresponding to f. This suggests that Theorem 3.4.15 applied together with Theorem 3.4.1 often gives optimal results, since the δ parameter usually turns out to be $\propto \frac{1}{d}$, where d is the maximal derivative of the function on the domain of interest.

3.5.1. Theorem (Lower bound for eigenvalue transformation). *Let $I \subseteq [-1,1]$, $a \geq 1$ and suppose U is a $(1,a,0)$-block-encoding of an unknown Hermitian matrix H with the only promise that the spectrum of H lies in I. Let $f : I \to \mathbb{R}$, and suppose that we have a quantum circuit V that implements a $(1,b,\varepsilon)$-block-encoding of $f(H)$ using T applications of U or $U^\dagger$, for all U fulfilling the promise. Then for all $x \neq y \in I \cap [-\frac{1}{2},\frac{1}{2}]$ we have that*

$$T = \Omega\left(\frac{|f(x)-f(y)| - 2\varepsilon}{|x-y|}\right).$$

[22]First approximate the function $e^{-\beta y}$ on $[0,1]$ using Corollary 3.4.10, then substitute $y \leftarrow x^2$.

More precisely for all $x, y \in I$ we have that

$$T \geq \frac{\max\Big[f(x)-f(y)-2\varepsilon, \sqrt{1-(f(y)-\varepsilon)^2}-\sqrt{1-(f(x)+\varepsilon)^2}\Big]}{\sqrt{2}\sqrt{1-xy-\sqrt{(1-x^2)(1-y^2)}}} \tag{3.32}$$

$$\geq \frac{\max\Big[f(x)-f(y)-2\varepsilon, \sqrt{1-(f(y)-\varepsilon)^2}-\sqrt{1-(f(x)+\varepsilon)^2}\Big]}{\sqrt{2}\max\Big[|x-y|, \Big|\sqrt{1-x^2}-\sqrt{1-y^2}\Big|\Big]}. \tag{3.33}$$

Proof:
First let us examine the case when H is a $d \times d$ matrix, $a = 1$ and U is of size $2d \times 2d$. Recall that in (2.14) we defined the two-dimensional reflection operator

$$R(x) = \begin{bmatrix} x & \sqrt{1-x^2} \\ \sqrt{1-x^2} & -x \end{bmatrix},$$

and note, that for all $x, y \in [0,1]$ we have that

$$\begin{aligned}\|R(x)-R(y)\| &= \sqrt{2}\sqrt{1-xy-\sqrt{(1-x^2)(1-y^2)}} \\ &\leq \sqrt{2}\max\Big[|x-y|, \Big|\sqrt{1-x^2}-\sqrt{1-y^2}\Big|\Big].\end{aligned} \tag{3.34}$$

For all $z \in [0,1]$ let $U_z := \bigoplus_{i=1}^d R(z)$, where the direct sum structure is arranged in such a way that U_z is a $(1,1,0)$-block-encoding of $H_z := zI$. Let $V[U_z]$ denote the circuit V when using the input unitary U_z. Since $V[U_z]$ uses U_z a total number of T times we have that

$$\|V[U_x]-V[U_y]\| \leq T\|U_x-U_y\| = T\|R(x)-R(y)\|. \tag{3.35}$$

By the promise on V we get that $V[U_z]$ is a $(1,b,\varepsilon)$-block-encoding of $f(H_z) = f(z)I$. Let $\varsigma_{\max/\min}$ denote the maximal/minimal singular value of a matrix. Using this notation we get that

$$\varsigma_{\max}\Big[(\langle 0|^{\otimes b}\otimes I)V[U_y](|0\rangle^{\otimes b}\otimes I)\Big] \leq f(y)+\varepsilon, \tag{3.36}$$

$$\varsigma_{\min}\Big[(\langle 0|^{\otimes b}\otimes I)V[U_x](|0\rangle^{\otimes b}\otimes I)\Big] \geq f(x)-\varepsilon. \tag{3.37}$$

Let us introduce the notation $\Pi_{\overline{|0\rangle\langle 0|}} := \big(\big(I_b - |0\rangle\langle 0|^{\otimes b}\big)\otimes I\big)$, then by (3.36)-(3.37) we have that $\|V[U_x]-V[U_y]\|$ can be lower bounded by

$$\begin{aligned}
&\geq \left\|(|0\rangle\langle 0|^{\otimes b}\otimes I)V[U_x](|0\rangle\langle 0|^{\otimes b}\otimes I)-(|0\rangle\langle 0|^{\otimes b}\otimes I)V[U_y](|0\rangle\langle 0|^{\otimes b}\otimes I)\right\|\\
&\geq \varsigma_{\min}\Big[(\langle 0|^{\otimes b}\otimes I)V[U_x](|0\rangle^{\otimes b}\otimes I)\Big]-\varsigma_{\max}\Big[(\langle 0|^{\otimes b}\otimes I)V[U_y](|0\rangle^{\otimes b}\otimes I)\Big]\\
&\geq f(x)-f(y)-2\varepsilon, \qquad (3.38)
\end{aligned}$$

and also by

$$\begin{aligned}
&\geq \left\|\Pi_{\overline{|0\rangle\langle 0|}}V[U_y](|0\rangle\langle 0|^{\otimes b}\otimes I)-\Pi_{\overline{|0\rangle\langle 0|}}V[U_x](|0\rangle\langle 0|^{\otimes b}\otimes I)\right\|\\
&\geq \varsigma_{\min}\Big[\Pi_{\overline{|0\rangle\langle 0|}}V[U_y](|0\rangle^{\otimes b}\otimes I)\Big]-\varsigma_{\max}\Big[\Pi_{\overline{|0\rangle\langle 0|}}V[U_x](|0\rangle^{\otimes b}\otimes I)\Big]\\
&= \sqrt{1-\varsigma^2_{\max}\Big[(\langle 0|^{\otimes b}\otimes I)V[U_y](|0\rangle^{\otimes b}\otimes I)\Big]}-\sqrt{1-\varsigma^2_{\min}\Big[(\langle 0|^{\otimes b}\otimes I)V[U_x](|0\rangle^{\otimes b}\otimes I)\Big]}\\
&\geq \sqrt{1-(f(y)-\varepsilon)^2}-\sqrt{1-(f(x)+\varepsilon)^2}. \qquad (3.39)
\end{aligned}$$

By combining (3.35) and (3.38)-(3.39) we get that

$$T\geq \frac{\max\Big[f(x)-f(y)-2\varepsilon,\sqrt{1-(f(y)-\varepsilon)^2}-\sqrt{1-(f(x)+\varepsilon)^2}\Big]}{\|R(x)-R(y)\|}.$$

Combining this inequality with (3.34) proves (3.32)-(3.33). Finally note that if $a>1$, then essentially the same argument can be used to prove (3.32)-(3.33), one just needs to define U_z with additional tensor products of identity matrices acting on the new ancilla qubits. □

The above lower bound suggests that the spectrum of H lying close to 1 is more "flexible" than the spectrum lying below say $\frac{1}{2}$ in absolute value. Indeed it turns out that the spectrum of H lying close to 1 is quadratically more useful than the spectrum $I\subseteq[-\frac{1}{2},\frac{1}{2}]$, cf. Corollary 3.4.10 and Lemma 2.4.3. This lower bound also explains why is it so difficult to amplify the spectrum close to 1, cf. Theorem 3.2.7. Finally note that since eigenvalue transformation is a special case of singular value transformation, it also gives a lower bound in singular value transformation.

3.6 Conclusion

Our main contribution in this chapter is to provide a paradigm, based on efficient transformations of block-encoded matrices, that unifies a host of quantum algorithms ranging from singular value estimation, linear equation solving, and quantum simulation to quantum walks. Prior to our contribution each of these algorithms would have to be understood independently, which makes mastering all of them a challenge. By presenting them all within the framework of quantum singular value transformation, many of the most popular techniques in these fields follow as a direct consequence. This greatly simplifies the learning process while also revealing algorithms that were hitherto unknown.

The chapter describes several novel applications fitting this general paradigm, including an exponentially improved algorithm for simulating fractional queries to an unknown unitary oracle, a new method for singular value estimation, and an improved algorithm for principal component regression.

We also give a novel view on quantum matrix arithmetics by summarizing known results about block-encoded matrices, showing that they enable performing matrix arithmetic on quantum computers in a simple and efficient manner. The described method in principle can give exponential savings in terms of the dimension of the matrices, and perfectly fits into our framework.

An interesting question for future work involves the recent work by Dohotaru and Høyer which shows that a wide range of quantum walk algorithms can be unified within a single paradigm called controlled quantum amplification [DH17]. While the structure of the quantum circuits introduced by them bears a strong resemblance to those used in qubitization, it is difficult to place their work within the framework we present here. The question of how to unify their approach with our techniques therefore remains open.

3.A Generalized minimum-finding algorithm

In this appendix we describe our generalized quantum minimum-finding algorithm, which we are going to apply to finding an approximation of the ground state energy of a Hamiltonian. This algorithm generalizes the results of Dürr and Høyer [DH96] in a manner similar to the way amplitude amplification [BHMT02] generalizes Grover search: we do not need to assume the ability to query individual elements of the search space, we just need to be able to generate a superposition over the search space. The algorithm also has the benefit over binary search that it removes a logarithmic factor from the complexity.

The backbone of our analysis will be the meta-algorithm below from [DH96]. The meta-algorithm finds the minimal element in the range of the random variable X by sampling, where by "range" we mean the values which occur with non-zero probability. We assume X has finite range.

Meta-Algorithm 1 Minimum-finding

```
Input  A discrete random variable $X$ with finite range.
Output  The minimal value $x_{\min}$ in the range of $X$.
  Init $t \leftarrow 0$; $s_0 \leftarrow \infty$
  Repeat until $s_t$ is minimal in the range of $X$
    1. $t \leftarrow t+1$
    2. Sample a value $s_t$ according to the distribution $\Pr(X = s_t \mid X < s_{t-1})$.
```

Note that the above algorithm will always find the minimum, since the obtained samples are strictly decreasing.

3.A.1. Lemma. *Let X be a finite discrete random variable whose range of values is $x_1 < x_2 < \ldots < x_N$. Let $S(X) = \{s_1, s_2, \ldots\}$ denote the random set of values obtained via sampling during a run of Meta-Algorithm 1 with input random variable X. If $k \in [N]$, then*

$$\Pr(x_k \in S(X)) = \frac{\Pr(X = x_k)}{\Pr(X \leq x_k)}.$$

Proof:
The intuition of the proof is to show that

$$\Pr(s_t = x_k \mid t \in [N] \text{ is the first time such that } s_t \leq x_k) = \frac{\Pr(X = x_k)}{\Pr(X \leq x_k)}. \tag{3.40}$$

To formulate the statement more precisely[23] we consider a fixed value $t \in [N]$. For notational convenience let $x := x_k$, $x_{N+1} := \infty$, we prove (3.40) by:

$$\begin{aligned}
&\Pr(s_t = x \mid s_t \leq x \wedge s_{t-1} > x) = \\
&\qquad = \frac{\Pr(s_t = x)}{\Pr(s_t \leq x \wedge s_{t-1} > x)} \\
&\qquad = \sum_{x_\ell > x} \frac{\Pr(s_t = x \wedge s_{t-1} = x_\ell)}{\Pr(s_t \leq x \wedge s_{t-1} > x)} \\
&\qquad = \sum_{x_\ell > x} \frac{\Pr(s_t = x \wedge s_{t-1} = x_\ell)}{\Pr(s_t \leq x \wedge s_{t-1} = x_\ell)} \frac{\Pr(s_t \leq x \wedge s_{t-1} = x_\ell)}{\Pr(s_t \leq x \wedge s_{t-1} > x)} \\
&\qquad = \sum_{x_\ell > x} \frac{\Pr(s_t = x \mid s_{t-1} = x_\ell)\Pr(s_{t-1} = x_\ell)}{\Pr(s_t \leq x \mid s_{t-1} = x_\ell)\Pr(s_{t-1} = x_\ell)} \frac{\Pr(s_t \leq x \wedge s_{t-1} = x_\ell)}{\Pr(s_t \leq x \wedge s_{t-1} > x)} \\
&\qquad = \frac{\Pr(X = x)}{\Pr(X \leq x)} \sum_{x_\ell > x} \frac{\Pr(s_t \leq x \wedge s_{t-1} = x_\ell)}{\Pr(s_t \leq x \wedge s_{t-1} > x)} \\
&\qquad = \frac{\Pr(X = x)}{\Pr(X \leq x)}.
\end{aligned}$$

This is enough to conclude the proof, since there is always a smallest $t \in [N]$ such that $s_t \leq x$, as the algorithm always finds the minimum in at most N steps. So

[23]Throughout this proof, whenever a fraction is 0/0, we simply interpret it as 0. Therefore we also interpret conditional probabilities, conditioned on events that happen with probability 0, as 0.

we can finish the proof by

$$\begin{aligned}
\Pr(x \in S(X)) &= \sum_{t=1}^{N} \Pr(s_t = x) \\
&= \sum_{t=1}^{N} \Pr(s_t = x \mid s_t \le x \wedge s_{t-1} > x)\Pr(s_t \le x \wedge s_{t-1} > x) \\
&= \frac{\Pr(X = x)}{\Pr(X \le x)} \sum_{t=1}^{N} \Pr(s_t \le x \wedge s_{t-1} > x) \\
&= \frac{\Pr(X = x)}{\Pr(X \le x)} \Pr(\exists t \in [N]\colon\ s_t \le x \wedge s_{t-1} > x) \\
&= \frac{\Pr(X = x)}{\Pr(X \le x)}.
\end{aligned}$$

□

Now we describe our generalized minimum-finding algorithm which is based on Meta-Algorithm 1. We take some unitary U, and replace X by the distribution obtained if we measured the second register of $U|0\rangle$. We implement conditional sampling via amplitude amplification and the use of the exponential search algorithm of Boyer et al. [BBHT98]. If a unitary prepares the state $|0\rangle|\phi\rangle + |1\rangle|\psi\rangle$ where $\|\phi\|^2 + \|\psi\|^2 = 1$, then this exponential search algorithm built on top of amplitude amplification prepares the state $|1\rangle|\psi\rangle$ probabilistically using an expected number of $\mathcal{O}(1/\|\psi\|)$ applications of U and U^{-1} (we will skip the details here, which are straightforward modifications of [BBHT98]).

Algorithm 3.1 Generalized minimum-finding

Input A number M and a unitary U, acting on q qubits, such that $U|0\rangle = \sum_{k=1}^{N}|\psi_k\rangle|x_k\rangle$, where x_k is a binary string representing some number and $|\psi_k\rangle$ is an unnormalized quantum state on the first register. Let $x_1 < x_2 < \ldots < x_N$ and define X to be the random variable with $\Pr(X = x_k) = \|\psi_k\|^2$.

Output Some $|\psi_k\rangle|x_k\rangle$ for a (hopefully) small k.

```
Init $t \leftarrow 0$; $s_0 \leftarrow \infty$
While the total number of applications of $U$ and $U^{-1}$ does not exceed $M$:
  1. $t \leftarrow t + 1$
  2. Use the exponential search algorithm with amplitude amplification on
     states such that $x_k < s_{t-1}$ to obtain a sample $|\psi_k\rangle|x_k\rangle$.
  3. $s_t \leftarrow x_k$
```

3.A.2. Lemma. *There exists $C \in \mathbb{R}_+$, such that if we run Algorithm 3.1 indefinitely (setting $M = \infty$), then for every U and x_k the expected number of uses of U and U^{-1} before obtaining a sample $x \leq x_k$ is at most $\frac{C}{\sqrt{\Pr(X \leq x_k)}}$.*

Proof:
Let $X_{<x_\ell}$ denote the random variable for which $\Pr(X_{<x_\ell} = x) = \Pr(X = x \mid X < x_\ell)$. The expected number of uses of U and U^{-1} in Algorithm 3.1 before obtaining any value $x \leq x_k$ is

$$
\begin{aligned}
&\mathbb{E}[\#\text{uses of } U \text{ before finding } x \leq x_k] = \\
&\quad = \sum_{\ell=k+1}^{N} \Pr(x_\ell \in S(X))\, \mathbb{E}[\#\text{uses of } U \text{ for sampling from } X_{<x_\ell}] \\
&\quad = \sum_{\ell=k+1}^{N} \frac{\Pr(X = x_\ell)}{\Pr(X \leq x_\ell)} \mathcal{O}\left(\frac{1}{\sqrt{\Pr(X < x_\ell)}}\right) \\
&\quad = \mathcal{O}\left(\sum_{\ell=k+1}^{N} \frac{\Pr(X = x_\ell)}{\Pr(X \leq x_\ell)} \frac{1}{\sqrt{\Pr(X < x_\ell)}}\right) \\
&\quad = \mathcal{O}\left(\frac{1}{\sqrt{\Pr(X \leq x_k)}}\right),
\end{aligned}
$$

where the last equality follows from Equation (3.46) below. The constant C from the lemma is the constant hidden by the $\mathcal{O}$. The remainder of this proof consists of proving (3.46) using elementary calculus. Let us introduce the notation $p_0 := \Pr(X \leq x_k)$ and for all $j \in [N-k]$ let $p_j := \Pr(X = x_{k+j})$. Then the expression inside the $\mathcal{O}$ on the second-to-last line above becomes

$$
\sum_{\ell=k+1}^{N} \frac{\Pr(X = x_\ell)}{\Pr(X \leq x_\ell)} \sqrt{\frac{1}{\Pr(X < x_\ell)}} = \sum_{j=1}^{N-k} \frac{p_j}{\sum_{i=0}^{j} p_i} \sqrt{\frac{1}{\sum_{i=0}^{j-1} p_i}}. \tag{3.41}
$$

The basic idea is that we treat the expression on the right-hand side of (3.41) as an integral approximation sum for the integral $\int_{p_0}^{1} z^{-3/2} dz$, and show that it is actually always less than the value of this integral. We proceed by showing that replacing a p_j with two distinct probabilities $p_j/2$, as in Eq. (3.42), always increases the sum.

Let us fix some $j \in [N-k]$ and define

$$
p'_i = \begin{cases} p_i & \text{for } i \in \{0, 1, \ldots, j-1\} \\ p_i/2 & \text{for } i \in \{j, j+1\} \\ p_{i-1} & \text{for } i \in \{j+2, \ldots, N-k+1\} \end{cases} \tag{3.42}
$$

and observe that

$$\begin{aligned}\sum_{j=1}^{N-k+1} \frac{p'_j}{\sum_{i=0}^{j} p'_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} p'_i}} - \sum_{j=1}^{N-k} \frac{p_j}{\sum_{i=0}^{j} p_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} p_i}} = \\ =&\frac{p'_j}{\sum_{i=0}^{j} p'_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} p'_i}} + \frac{p'_{j+1}}{\sum_{i=0}^{j+1} p'_i}\sqrt{\frac{1}{\sum_{i=0}^{j} p'_i}} - \frac{p_j}{\sum_{i=0}^{j} p_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} p_i}} \\ =&\frac{p_j/2}{p_j/2+\sum_{i=0}^{j-1} p_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} p_i}} + \frac{p_j/2}{p_j+\sum_{i=0}^{j-1} p_i}\sqrt{\frac{1}{p_j/2+\sum_{i=0}^{j-1} p_i}} \\ &- \frac{p_j}{p_j+\sum_{i=0}^{j-1} p_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} p_i}}. \end{aligned} \tag{3.43}$$

We show that (3.43) is ≥ 0 after simplifying the expression by substituting $a := \sum_{i=0}^{j-1} p_i$ and $b := p_j/2$:

$$\frac{b}{a+b}\sqrt{\frac{1}{a}} + \frac{b}{a+2b}\sqrt{\frac{1}{a+b}} - \frac{2b}{a+2b}\sqrt{\frac{1}{a}} = \frac{\big(a+b-\sqrt{a}\sqrt{a+b}\big)b}{(a+b)^{3/2}(a+2b)} \geq 0. \tag{3.44}$$

Let us fix some parameter $\delta > 0$. Recursively applying the halving procedure of Eq. (3.42) for different indices, we can find some $J \in \mathbb{N}$ and $\tilde{p} \in \mathbb{R}_+^{J+1}$ such that $\sum_{j=0}^{J} \tilde{p}_j = 1$, $\tilde{p}_0 = p_0$ and $\tilde{p}_j \leq \delta$ for all $j \in [J]$, moreover

$$\sum_{j=1}^{N-k} \frac{p_j}{\sum_{i=0}^{j} p_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} p_i}} \leq \sum_{j=1}^{J} \frac{\tilde{p}_j}{\sum_{i=0}^{j} \tilde{p}_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} \tilde{p}_i}}.$$

Observe that for all $j \in [J]$

$$\begin{aligned}\sqrt{\frac{1}{\sum_{i=0}^{j-1} \tilde{p}_i}} = \sqrt{\frac{1}{\sum_{i=0}^{j} \tilde{p}_i}}\sqrt{\frac{\sum_{i=0}^{j} \tilde{p}_i}{\sum_{i=0}^{j-1} \tilde{p}_i}} =& \sqrt{\frac{1}{\sum_{i=0}^{j} \tilde{p}_i}}\sqrt{1+\frac{\tilde{p}_j}{\sum_{i=0}^{j-1} \tilde{p}_i}} \\ \leq& \sqrt{\frac{1}{\sum_{i=0}^{j} \tilde{p}_i}}\sqrt{1+\frac{\delta}{\tilde{p}_0}} \leq \sqrt{\frac{1}{\sum_{i=0}^{j} \tilde{p}_i}}\left(1+\frac{\delta}{p_0}\right). \end{aligned} \tag{3.45}$$

Therefore

$$\begin{aligned}
\sum_{j=1}^{N-k} \frac{p_j}{\sum_{i=0}^{j} p_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} p_i}} &\leq \sum_{j=1}^{J} \frac{\tilde{p}_j}{\sum_{i=0}^{j} \tilde{p}_i}\sqrt{\frac{1}{\sum_{i=0}^{j-1} \tilde{p}_i}} \\
\text{(by (3.45))} \quad &\leq \sum_{j=1}^{J} \frac{\tilde{p}_j}{\sum_{i=0}^{j} \tilde{p}_i}\sqrt{\frac{1}{\sum_{i=0}^{j} \tilde{p}_i}}\left(1+\frac{\delta}{p_0}\right) \\
&= \left(1+\frac{\delta}{p_0}\right)\sum_{j=1}^{J} \frac{\tilde{p}_j}{\left(\sum_{i=0}^{j} \tilde{p}_i\right)^{3/2}} \\
&\leq \left(1+\frac{\delta}{p_0}\right)\sum_{j=1}^{J} \int_{\sum_{i=0}^{j-1} \tilde{p}_i}^{\sum_{i=0}^{j} \tilde{p}_i} z^{-\frac{3}{2}} dz \\
&= \left(1+\frac{\delta}{p_0}\right)\int_{p_0}^{1} z^{-\frac{3}{2}} dz \\
&= \left(1+\frac{\delta}{p_0}\right)\left[-2z^{-\frac{1}{2}}\right]_{p_0}^{1} \\
&\leq \left(1+\frac{\delta}{p_0}\right)\left(\frac{2}{\sqrt{p_0}}\right).
\end{aligned}$$

Since this inequality holds for every $\delta > 0$, we can conclude using (3.41) that

$$\sum_{\ell=k+1}^{N} \frac{\Pr(X = x_\ell)}{\Pr(X \leq x_\ell)}\sqrt{\frac{1}{\Pr(X < x_\ell)}} \leq \frac{2}{\sqrt{p_0}} = \frac{2}{\sqrt{\Pr(X \leq x_k)}}. \tag{3.46}$$

□

It is not too hard to work out the constant by following the proof of [BBHT98] providing something like $C \approx 25$. The following theorem works with any C satisfying Lemma 3.A.2.

3.A.3. Theorem (Generalized Minimum-Finding). *If we run Algorithm 3.1 with input satisfying $M \geq 4C/\sqrt{\Pr(X \leq x)}$ for C as in Lemma 3.A.2 and a unitary U that acts on q qubits, then at termination we obtain an x_i from the range of X that satisfies $x_i \leq x$ with probability at least $\frac{3}{4}$. This will require at most M applications of U and U^{-1} and $\mathcal{O}(qM)$ other gates. Moreover the success probability can be boosted to at least $1-\delta$ with $\mathcal{O}(\log(1/\delta))$ repetitions.*

Proof:
Let x_k be the largest value in the range of X such that $x_k \leq x$. Then Lemma 3.A.2 says that the expected number of applications of U and U^{-1} before finding a value $x_i \leq x_k$ is at most $C/\sqrt{\Pr(X \leq x_k)} = C/\sqrt{\Pr(X \leq x)}$, therefore by the Markov inequality we know that the probability that we need to use U and U^{-1} at least

$4C/\sqrt{\Pr(X \leq x)}$ times is at most 1/4. The boosting of the success probability can be done using standard techniques, e.g., by repeating the whole procedure $\mathcal{O}(\log(1/\delta))$ times and taking the minimum of the outputs.

The number of applications of U and U^{-1} follows directly form the algorithms description. Then, for the number of other gates, each amplitude amplification step needs to implement a binary comparison and a reflection through the $|0\rangle$ state, both of which can be constructed using $\mathcal{O}(q)$ elementary gates, giving a total of $\mathcal{O}(qM)$ gates. □

Note that this result is a generalization of Dürr and Høyer [DH96]: if we can create a uniform superposition over N values $x_1 < x_2 < \ldots < x_N$, then $\Pr(X \leq x_1) = 1/N$ and therefore Theorem 3.A.3 guarantees that we can find the minimum with high probability with $\mathcal{O}(\sqrt{N})$ steps.

Now we describe an application of this generalized search algorithm that we use in the paper. This final lemma in this appendix describes how to estimate the smallest eigenvalue of a Hamiltonian. A similar result was shown by Poulin and Wocjan [PW09], but we improve on the analysis to fit our framework better. We assume block-encoding access to the Hamiltonian H, and will count queries to the unitary implementing the block-encoding. Due to our generalized minimum-finding techniques the complexity has no $\frac{1}{\varepsilon}\log\left(\frac{1}{\varepsilon}\right)$ term.

3.A.4. Lemma. *Let $\varepsilon \in (0, 1/2)$, and suppose that U is an a-qubit block-encoding of the Hamiltonian $H \in \mathbb{C}^{n\times n}$. Then we can find an ε-precise estimate of the ground-state energy of H with success probability at least $2/3$ with*

$$\mathcal{O}\left(\frac{\sqrt{n}}{\varepsilon}\log(n) + \sqrt{n}\log^2(n)\log(1/\varepsilon)\right)$$

uses of (controlled) U, $U^\dagger$ and $\mathcal{O}(a)$ times more two-qubit gates.

Proof:
The general idea is as follows: we prepare a maximally entangled state on two registers, and apply phase estimation [NC00, Section 5.2][CEMM98] to the first register with respect to the unitary e^{iH}. We then use Theorem 3.A.3 to find the minimal phase. In order to guarantee correctness we need to account for all the approximation errors coming from approximate implementations. This causes some technical difficulty, since the approximation errors can introduce phase estimates that are much less than the true minimum. We need to make sure that the minimum-finding algorithm finds these faulty estimates only with a tiny probability.

Let $H = \sum_{j=1}^{n} E_j |\phi_j\rangle\langle\phi_j|$, be an eigendecomposition with eigenvalues $E_1 \leq E_2 \leq \ldots \leq E_n$. We first initialize two $\log(n)$-qubit registers in a maximally entangled state $\frac{1}{\sqrt{n}}\sum_{j=0}^{n-1}|j\rangle|j\rangle$. This can be done using $\log(n)$ Hadamard and

CNOT gates.[24] Due to the invariance of maximally entangled states under transformations of the form $Q \otimes Q^*$ for unitary Q, there is an orthonormal basis $\{|\upsilon_j\rangle : j \in [n]\}$ such that

$$\frac{1}{\sqrt{n}}\sum_{j=0}^{n-1}|j\rangle|j\rangle = \frac{1}{\sqrt{n}}\sum_{j=1}^{n}|\phi_j\rangle|\upsilon_j\rangle.$$

Let $T := \Theta(1/\varepsilon)$, and first assume that we have access to a perfect unitary V which implements $V = \sum_{t=0}^{T-1}|t\rangle\langle t| \otimes e^{itH}$. If we apply phase estimation to the quantum state $|\phi_j\rangle$ using V, then we get some phase estimate $|E\rangle$ such that $|E-E_j| \leq \varepsilon$ with high probability. If we repeat phase estimation $\mathcal{O}(\log(n))$ times, and take the median of the estimates, then we obtain an E such that $|E-E_j| \leq \varepsilon$ with probability at least $1-b/n$, for some $b = \Theta(1)$.

Since in our maximally entangled state $|\phi_j\rangle$ is entangled with $|\upsilon_j\rangle$ on the second register, applying phase estimation to the first register in superposition does not cause interference on the phase register. Denote the above preparation-estimation-boost circuit by W. Define Π to be the projector which projects to the subspace of estimation values E such that there is a $j \in [n]$ with $|E-E_j| \leq \varepsilon$. By the non-interference argument we can see that, after applying W, the probability that we get an estimation E such that $|E-E_j| > \varepsilon$ for all $j \in [n]$, is at most b/n. Therefore $\|(I-\Pi)W|0\rangle\|^2 \leq b/n$. Also let Π_1 denote the projector which projects to phase estimates that yield E such that $|E-E_1| \leq \varepsilon$. It is easy to see that $\|\Pi_1 W|0\rangle\|^2 \geq 1/n - b/n^2$.

Now let us replace V by $\tilde{V}$ implemented via Corollary 3.4.8, with precision such that $\|V-\tilde{V}\| \leq c'/(n\log(n))$ for some $c' = \Theta(1)$. Let $\tilde{W}$ denote the circuit that we obtain from W by replacing V with $\tilde{V}$. Since in the repeated phase-estimation procedure we use V in total $\mathcal{O}(\log(n))$ times, by using the triangle inequality we see that $\|W-\tilde{W}\| \leq c/(2n)$, where $c = \Theta(1)$. We use the well-known fact that if two unitaries are δ-close in operator norm, and they are applied to the same quantum state, then the measurement statistics of the resulting states are 2δ-close. Therefore we can upper bound the difference in probability of getting outcome $(I-\Pi)$:

$$\|(I-\Pi)\tilde{W}|0\rangle\|^2 - \|(I-\Pi)W|0\rangle\|^2 \leq 2\|W|0\rangle - \tilde{W}|0\rangle\| \leq c/n,$$

hence $\|(I-\Pi)\tilde{W}|0\rangle\|^2 \leq (b+c)/n$, and we can prove similarly that $\|\Pi_1\tilde{W}|0\rangle\|^2 \geq 1/n - (b+c)/n$.

Now let $|\psi\rangle := (I-\Pi)\tilde{W}|0\rangle/\|(I-\Pi)\tilde{W}|0\rangle\|$ be the state that we would get after post-selecting on the $(I-\Pi)$-outcome of the projective measurement Π. For small enough b, c we have that $\||\psi\rangle - \tilde{W}|0\rangle\| = \mathcal{O}(\sqrt{(b+c)/n})$ by the triangle inequality. Thus there exists an idealized unitary W' such that

[24] If n is not a power of 2, then one might need to use a constant times more two-qubit gates, involving a constant number of amplitude amplification rounds.

$|\psi\rangle = W'|0\rangle$, and $\|\tilde{W} - W'\| = \mathcal{O}(\sqrt{(b+c)/n})$. Observe that $\|\Pi_1 W'|0\rangle\|^2 = \||\psi\rangle\|^2 \geq \|\Pi_1 \tilde{W}|0\rangle\|^2 \geq 1/n - (b+c)/n$.

Now suppose $(b+c) \leq 1/2$ and we run the generalized minimum-finding algorithm of Theorem 3.A.3 using W' with $M = 6C\sqrt{n}$. Since

$$\Pr(E \leq E_1 + \varepsilon) \geq \|\Pi_1 U'|0\rangle\|^2 \geq (1-b-c)/n \geq 1/(2n) > 4/(9n)$$

we will obtain an estimate E such that $E \leq E_1 + \varepsilon$, with probability at least 3/4. But since $\Pi|\psi\rangle = |\psi\rangle$, we find that any estimate that we might obtain satisfies $E \geq E_1 - \varepsilon$. So an estimate $E \leq E_1 + \varepsilon$ always satisfies $|E - E_1| \leq \varepsilon$.

The problem is that we only have access to $\tilde{U}$ as a quantum circuit. Let $C_{MF}(\tilde{U})$ denote the circuit that we get from Theorem 3.A.3 when using it with $\tilde{U}$ and define similarly $C_{MF}(U')$ for U'. Since we use $\tilde{U}$ a total of $\mathcal{O}(\sqrt{n})$ times in $C_{MF}(\tilde{U})$ and

$$\left\|\tilde{U} - U'\right\| = \mathcal{O}\Big(\sqrt{(b+c)/n}\Big), \text{ we get that } \left\|C_{MF}(\tilde{U}) - C_{MF}(U')\right\| = \mathcal{O}\Big(\sqrt{b+c}\Big).$$

Therefore the measurement statistics of the two circuits differ by at most $\mathcal{O}(\sqrt{b+c})$. Choosing b, c small enough constants ensures that $C_{MF}(\tilde{U})$ outputs a proper estimate E such that $|E - E_1| \leq \varepsilon$ with probability at least 2/3.

The query complexity has an $\mathcal{O}(1/\varepsilon + \log(1/\varepsilon)\log(n))$ factor coming from the implementation of $\tilde{V}$ by Corollary 3.4.8. This gets multiplied with $\mathcal{O}(\log(n))$ by the boosting of phase estimation, and by $\mathcal{O}(\sqrt{n})$ due to the minimum-finding algorithm. The gate complexity coming from Corollary 3.4.8 is at most $\mathcal{O}(a)$-times query complexity. The phase estimation (Fourier transform) can be performed with $\mathcal{O}\big(\log^2(1/\varepsilon)\big)$ gates, and the comparisons in minimum finding take $\mathcal{O}(\log(1/\varepsilon))$ gates, so both costs are dominated by the cost of implementing $\tilde{V}$. $\square$

Note that the minimum-finding algorithm of Theorem 3.A.3 can also be used for state preparation. If we choose 2ε less than the energy-gap of the Hamiltonian, then upon finding the approximation of the ground-state energy we also prepare an approximate ground state. The precision of this state preparation can be improved with logarithmic cost, as can be seen from the proof of Lemma 3.A.4.

Chapter 4

Faster quantum gradient computation

Gradient descent is a simple iterative method for optimization which makes discrete steps in the direction of steepest descent, determined by the gradient, until it finds an approximate local minimum. In this chapter we consider a generic framework of optimization algorithms based on gradient descent. We develop a quantum algorithm that computes the gradient of a multi-variate real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ by evaluating it at only a logarithmic number of points in superposition. Our algorithm is an improved version of Jordan's gradient computation algorithm [Jor05], providing an approximation of the gradient ∇f with quadratically better dependence on the evaluation accuracy of f, for an important class of smooth functions. Furthermore, we show that most objective functions arising from quantum optimization procedures satisfy the necessary smoothness conditions, hence our algorithm provides a quadratic improvement in the complexity of computing their gradient. We also show that in a continuous phase-query model, our gradient computation algorithm has optimal query complexity up to poly-logarithmic factors, for a particular class of smooth functions. Moreover, we show that for low-degree multivariate polynomials our algorithm can provide exponential speedups compared to Jordan's algorithm in terms of the dimension d.

One of the technical challenges in applying our gradient computation procedure for quantum optimization problems is the need to convert between a probability oracle (which is common in quantum optimization procedures) and a phase oracle (which is common in quantum algorithms) of the objective function f. We provide *efficient* subroutines to perform this delicate interconversion between the two types of oracles, incurring only a logarithmic overhead, which might be of independent interest. Finally, using these tools we improve the runtime of prior approaches for training quantum auto-encoders, variational quantum eigensolvers (VQE), and quantum approximate optimization algorithms (QAOA).

This chapter is based on [GAW19] © 2019 SIAM.

4.1 Introduction

Quantum optimization. Optimization is a fundamentally important task that touches on virtually every area of science. Recently, there have been many quantum algorithms that provide substantial improvements for several optimization problems [Gro96, DHHM06, Jor05, HHL09, CKS17, BS17, vAGGdW17, BKL$^+$17b, vAG19, KP18, vAGGdW18, CCLW18]. However, applying non-Grover techniques to real-word optimization problems has proven challenging, because generic problems often fail to satisfy the delicate requirements of advanced quantum techniques.

A different paradigm of quantum optimization is based on variational quantum circuits. These circuits are usually based on heuristics, nevertheless they are promising candidates for providing quantum advantage in real-word problems, including quantum simulation [PMS$^+$14, WHT15], optimization [FGG14], quantum neural networks [FN18] and machine learning [SBSW18]. These variational circuits usually try to optimize some objective function, which could correspond to, e.g., the energy of a quantum state in a molecule or the prediction loss in a learning model. These quantum circuits have the appealing feature that they often have low depth, and therefore can potentially be implemented on NISQ (noisy intermediate-scale quantum) hardware. The training is usually performed by running the circuit several times and using classical gradient descent on the parameter space.

In the long term it is expected that training could become a bottleneck as the size of the variational quantum circuits grow, similarly to for example the training of *classical* deep neural networks. However, the ultimate limitations of variational training algorithms are not well understood. In particular it has been unclear whether it is possible to achieve improvements beyond the simple quantum speedups provided by amplitude estimation techniques. This underscores the importance of understanding the performance of training algorithms as we begin to push beyond NISQ-era devices. Our main contribution is showing that non-trivial speedups can be achieved via quantum gradient computation of the objective function. We remark that in this variational setting the prior works on quantum gradient descent [RSPL16, KP17a] are not applicable.

It is usually difficult to reduce the number of iterations using quantum computers in iterative algorithms such as gradient descent. We therefore focus on the cost per iteration, and speed up gradient computation. Since in general very little is known about the landscape of objective functions arising from variational quantum circuits, we study the problem in a black-box setting where we can only learn about the objective function by evaluating it at arbitrary points. This setting simplifies the problem and allows us to reason about upper and lower bounds on the cost of gradient computation.

Classical gradient-based optimization algorithms. We first give a high-level description of a basic classical gradient-based optimization technique. The problem is, given $p : \mathbb{R}^d \to \mathbb{R}$, compute

$$\mathrm{OPT} = \min\{p(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^d\}. \tag{4.1}$$

A heuristic solution of the optimization problem (4.1) can be obtained by computing the *gradient* of p:

$$\nabla p = \left(\frac{\partial p}{\partial x_1}, \frac{\partial p}{\partial x_2}, \ldots, \frac{\partial p}{\partial x_d}\right) \tag{4.2}$$

It is a well-known fact in calculus that p decreases the *fastest* in the direction of $-(\nabla p(\boldsymbol{x}))$. This simple observation is the basis of gradient-based optimization algorithms. Given the generality of the optimization problem (4.1) and the simplicity of the algorithm, gradient-based techniques are widely used in mathematics, physics and engineering.

Probability oracles. Since quantum algorithms such as QAOA or VQE (see Section 4.7) work with an objective function that needs to be learned by sampling, we assume the function is given by a *probability oracle*, which for every $\boldsymbol{x} \in \mathbb{R}^d$ acts as:

$$U_p \colon |\vec{0}\rangle|\boldsymbol{x}\rangle \mapsto \left(\sqrt{p(\boldsymbol{x})}|1\rangle|\psi^{(1)}_{\boldsymbol{x}}\rangle + \sqrt{1-p(\boldsymbol{x})}|0\rangle|\psi^{(0)}_{\boldsymbol{x}}\rangle\right)|\boldsymbol{x}\rangle \tag{4.3}$$

where the continuous variable $\boldsymbol{x}$ is represented binarily with some finite precision.[1]

The classical analogue of this model is when we get samples from the distribution $(p(\boldsymbol{x}), 1 - p(\boldsymbol{x}))$. Using empirical estimation, $\mathcal{O}(1/\varepsilon^2)$ samples suffice for estimating $p(\boldsymbol{x})$ with precision $\mathcal{O}(\varepsilon)$. If the function is sufficiently smooth, using standard techniques we can compute an ε-approximation of $\nabla_i p(\boldsymbol{x})$ using a logarithmic number of such function estimations. Computing the gradient this way uses $\widetilde{\mathcal{O}}(d/\varepsilon^2)$ samples.

Our main result shows that such an ε-approximate gradient estimate (in the ℓ_∞-norm) can be computed with quadratically fewer *quantum* queries to a probability oracle. We remark that our quadratic speedup is not yet-another Grover speedup, instead it comes from applying the Fourier transformation together with some optimized interpolation techniques.

Our improved gradient computation algorithm. Jordan [Jor05] assumed access to $f \colon \mathbb{R}^d \to \mathbb{R}$ by an oracle, which on input $\boldsymbol{x}$, outputs $f(\boldsymbol{x})$ binarily with some finite accuracy, and constructed a quantum algorithm that outputs an ε-coordinate-wise approximation of the gradient ∇f with a single use of this binary

[1]Note that this is a much weaker input model than the oracle model used by Jordan [Jor05].

oracle. This demonstrates a striking quantum advantage, since classically one needs $\Omega(d)$ queries. His algorithm prepares a uniform superposition of evaluation points over a finite grid, then approximately implements a phase unitary

$$\mathrm{O}_f : |\boldsymbol{x}\rangle \mapsto e^{2\pi i \frac{\sqrt{d}}{\varepsilon^2} f(\boldsymbol{x})}|\boldsymbol{x}\rangle,$$

using a *single* $\mathcal{O}\big(\varepsilon^2/\sqrt{d}\big)$-accurate evaluation of f and then applies an inverse Fourier transformation to obtain an approximation of the gradient. Although this algorithm only uses a single query, the required precision of the function evaluation can be prohibitive. Moreover, the original analysis of Jordan [Jor05] implicitly assumes that the function is essentially quadratic, by neglecting third and higher-order contributions. Using Jordan's algorithm in our framework, where we assume access to a probability oracle, evaluating the function with the prescribed precision using amplitude amplification would require $\Omega(\sqrt{d}/\varepsilon^2)$ queries.

Our improvements are two-fold: our quantum algorithm requires only $\widetilde{\mathcal{O}}(\varepsilon/\sqrt{d})$-accurate evaluations of the function; on the other hand it also works for functions with non-negligible higher-order terms, such as the objective functions arising from variational circuits. The main new ingredient is the use of higher-degree central-difference formulas, a technique borrowed from calculus. The heart of our proof is showing that if f is sufficiently smooth, then by using central-difference formulas we can approximately linearize the function on a diameter-1 hypercube by performing only $\log(\sqrt{d}/\varepsilon)$ function evaluations. We prove this by bounding the "second moment" of higher-order bounded tensors using Lemma 4.5.8, which might be of independent interest.

Our algorithm works by evaluating the approximately linearized function over a uniform superposition of grid points in the hypercube, followed by a d-dimensional quantum Fourier transform, providing a classical description of an approximate gradient, similarly to Jordan's algorithm.

4.1.1. Theorem (See Theorem 4.5.10 & Theorem 4.6.3). *Let $\varepsilon > 0$, $d \in \mathbb{N}$, and $c = \mathcal{O}(1)$. Suppose we are given probability (or phase) oracle access to a function*[2] *$f\colon \mathbb{R}^d \to \mathbb{R}$, such that $|\partial_{i_1,i_2,\ldots,i_k} f(\mathbf{0})| \leq c^k\sqrt{k!}$ for every $k \in \mathbb{N}$ and $(i_1, i_2, \ldots, i_k) \in [d]^k$. The quantum query complexity of computing (with high probability) an ε-coordinatewise-approximation of $\nabla f(\mathbf{0})$ is*

$$\widetilde{\Theta}\Big(\sqrt{d}/\varepsilon\Big).$$

Our algorithm is also gate-efficient; the gate complexity is $\widetilde{\mathcal{O}}(Q + d)$, where Q is the query complexity.

[2]Such a function belongs to the Gevrey [Gev18] class $G^{\frac{1}{2}}$.

Our lower bound techniques. In order to prove the optimality of our algorithm we prove an extended version of the so-called hybrid method [BBBV97].

4.1.2. Theorem. (Hybrid method for arbitrary phase oracles) *Let G be a (finite) set of labels and let $\mathcal{H} := \mathrm{Span}(|x\rangle : x \in G)$ be a Hilbert space. For a function $\tilde{f} : G \to \mathbb{R}$ let $\mathrm{O}_{\tilde{f}}$ be the phase oracle acting on $\mathcal{H}$ such that*

$$\mathrm{O}_{\tilde{f}} : |x\rangle \to e^{i\tilde{f}(x)}|x\rangle \quad \textit{for every } x \in G.$$

Suppose that $\mathcal{F}$ is a finite set of functions $G \to \mathbb{R}$, and the function $f_* : G \to \mathbb{R}$ is not in F. If a quantum algorithm makes T queries to a (controlled) phase oracle $\mathrm{O}_{\tilde{f}}$ (or its inverse) and for all $f \in \mathcal{F}$ can distinguish with probability at least 2/3 *the case $\tilde{f} = f$ from the case $\tilde{f} = f_*$, then*

$$T \geq \frac{\sqrt{|\mathcal{F}|}}{3} \Bigg/ \sqrt{\max_{x \in G} \sum_{f \in \mathcal{F}} \min\left(|f(x) - f_*(x)|^2, 4\right)}.$$

In order to prove the lower bound in Theorem 4.1.1, we exhibit a family of functions $\mathcal{F}$ for which the functions can be well distinguished by calculating their gradients with accuracy ε in the ℓ_∞-norm. Then, with the help of Theorem 4.1.2 we show that this requires $\Omega(\sqrt{d}/\varepsilon)$ queries.

Applications. We consider three problems to which we apply our quantum gradient descent algorithm. We briefly describe below the problem of quantum variational eigensolvers (VQE) [PMS+14, WHT15], quantum approximate optimization algorithms (QAOA) [FGG14], and the quantum auto-encoding problem [WKGK16, ROAG17]. In each case we show how our gradient computation algorithm can provide a quadratic speedup in terms of the dimension d of the associated problem.

VQE is widely used to estimate the eigenvalue corresponding to some eigenstate of a Hamiltonian. The main idea in VQE is to begin with an efficiently parameterizable ansatz to the eigenstate. For the example of ground state energy estimation, the ansatz state is often taken to be a unitary coupled cluster expansion, coming from quantum chemistry insights, cf. [MEAG+18]. The terms in that unitary coupled cluster expansion are varied to provide the lowest energy for the groundstate, and the expected energy of the quantum state is mapped to the probability of some measurement outcome, making it accessible to our methods.

QAOA has a similar approach, the basic idea of the algorithm is to consider a parametrized family of states such as $|\psi(\boldsymbol{x})\rangle = \prod_{j=1}^{d} e^{-ix_j H_j}|0\rangle$. The aim is to tune the parameters of $|\psi(\boldsymbol{x})\rangle$ in order to minimize some objective function, which can, e.g., represent some combinatorial optimization problem. In particular, if H is a Hermitian operator corresponding to the objective function then we wish to find $\boldsymbol{x}$ such that $\langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle$ is minimized. For example, in order to minimize the number of violated constraints of a constraint satisfaction problem, we can choose $H = \sum_{m=1}^{M} C_m$ to represent the number of violations: C_m is 1 if and only if the m^{th} constraint is violated, else $C_m = 0$ [FGG14]. After normalization and using some standard techniques we can map this expectation value to the measurement probability of a single qubit. Thus, from the perspective of our algorithm, QAOA looks exactly like VQE.

The classical auto-encoder paradigm [Azo94] is an important technique in machine learning, which is widely used for data compression. An auto-encoder is essentially a neural network architecture which is tuned for the following task: given a set of high-dimensional vectors, we would like to learn a low-dimensional representation of the vectors, so that computations on the original data set can be "approximately" carried out by working only with the low-dimensional representations. What makes auto-encoding powerful is that it does not assume any prior knowledge about the data set. This makes it a viable technique in machine learning, with various applications in natural language processing, training neural networks, object classification, prediction or extrapolation of information, etc. In this chapter, we consider a natural quantum analogue (which was also considered before in the works of [WKGK16, ROAG17]) of the auto-encoder paradigm, and show how to use our quantum gradient computation algorithm to quadratically speed up the training of quantum autoencoders.

4.2 Organization of the chapter and preliminaries

In Section 4.3, we give a generic model of quantum optimization algorithms and a detailed description of the classical gradient descent algorithm. In Section 4.4, we describe how to convert a probability oracle to a phase oracle. In Section 4.5 we present our quantum gradient computation algorithm and prove our main Theorem 4.5.10 regarding its complexity. In Section 4.6, we present query lower bounds for algorithms that (approximately) compute the gradient of a function. In Section 4.7 we describe some applications. We conclude with some directions for future research in Section 4.8.

Notation. Now we describe some notation that we use throughout this chapter. Let $\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_d \in \mathbb{R}^d$ denote the standard basis vectors. We use bold letters for vectors $\boldsymbol{x} \in \mathbb{R}^d$, in particular we use the notation $\mathbf{0}$ for the 0 vector, and $\mathbf{1}$ for the all-1 vector $(\boldsymbol{e}_1 + \boldsymbol{e}_2 + \cdots + \boldsymbol{e}_d)$. By writing $\boldsymbol{y} + rS$ we mean $\{\boldsymbol{y} + r\boldsymbol{v} : \boldsymbol{v} \in S\}$ for vectors $S \subseteq \mathbb{R}^d$, and use the same notation for sets of numbers. For $\boldsymbol{x} \in \mathbb{R}^d$, let $\|\boldsymbol{x}\|_\infty = \max_{i\in[d]} |x_i|$ and $\|\boldsymbol{x}\| = (\sum_{i=1}^{d} x_i^2)^{1/2}$. For $M \in \mathbb{R}^{d\times d}$, let $\|M\|$ denote the operator norm of M.

In general, we use $\mathcal{H}$ to denote a finite-dimensional Hilbert space. For the n-qubit all-0 basis state we use the notation $|0\rangle^{\otimes n}$, or simply write $|\vec{0}\rangle$ when we do not want to focus on the value of n.

We use the convention $0^0 = 1$ throughout, and use the notation $\mathbb{N}_0 = \mathbb{N} \cup \{0\}$.

Higher-order calculus. Many technical lemmas in this chapter will revolve around the use of higher-order calculus. We briefly introduce some notation here and give some basic definitions.

4.2.1. Definition (Index-sequences). For $d \in \mathbb{N}$ and $k \in \mathbb{N}_0$ we call $\alpha \in [d]^k$ a d-dimensional length-k index-sequence. For a vector $\boldsymbol{r} \in \mathbb{R}^d$ we define $\boldsymbol{r}^\alpha := \prod_{j\in[k]} r_{\alpha_j}$, and for a k-times differentiable function, we define $\partial_\alpha f := \partial_{\alpha_1}\partial_{\alpha_2}\cdots\partial_{\alpha_k} f$. Finally, we use the notation $|\alpha| = k$ for denoting the length of the index-sequence.

4.2.2. Definition (Analytic function). We say that the function $f : \mathbb{R}^d \to \mathbb{R}$ is analytic if for all $\boldsymbol{x} \in \mathbb{R}^d$

$$f(\boldsymbol{x}) = \sum_{k=0}^{\infty} \sum_{\alpha\in[d]^k} \boldsymbol{x}^\alpha \frac{\partial_\alpha f(\boldsymbol{0})}{k!}. \tag{4.4}$$

4.2.3. Definition (Directional derivative). Suppose $f : \mathbb{R}^d \to \mathbb{R}$ is k-times differentiable at $\boldsymbol{x} \in \mathbb{R}^d$. We define the k-th order directional derivative in the direction $\boldsymbol{r} \in \mathbb{R}^d$ using the derivative of a one-parameter function parametrized by $\tau \in \mathbb{R}$ along the ray in the direction of $\boldsymbol{r}$:

$$\partial_{\boldsymbol{r}}^k f(\boldsymbol{x}) = \frac{d^k}{(d\tau)^k} f(\boldsymbol{x} + \tau\boldsymbol{r}).$$

Observe that, using the definitions above, one has

$$\partial_{\boldsymbol{r}}^k f = \sum_{\alpha\in[d]^k} \boldsymbol{r}^\alpha \cdot \partial_\alpha f. \tag{4.5}$$

In particular for every $i \in [d]$, we have that $\partial_{\boldsymbol{e}_i}^k f = \partial_i^k f$.

Central difference formulas (see, e.g. [Li05]) are often used to give precise approximations of derivatives of a function $h : \mathbb{R} \to \mathbb{R}$. These formulas are coming from polynomial interpolation, and yield precise approximations of directional derivatives too. Thus, we can use them to approximate the gradient of a high-dimensional function as shown in the following definition.

4.2.4. Definition. The degree-$2m$ *approximate central-difference linearization* of a function $f : \mathbb{R}^d \to \mathbb{R}$ is:

$$f_{(2m)}(\boldsymbol{x}) := \sum_{\substack{\ell=-m\\ \ell\neq 0}}^{m} \frac{(-1)^{\ell-1}}{\ell} \frac{\binom{m}{|\ell|}}{\binom{m+|\ell|}{|\ell|}} f(\ell\boldsymbol{x}) \approx \nabla f(\boldsymbol{0}) \cdot \boldsymbol{x}. \tag{4.6}$$

We denote the corresponding *central-difference coefficients* for $\ell \in \{-m, \ldots, m\}$ by

$$a_\ell^{(2m)} := \frac{(-1)^{\ell-1}}{\ell} \frac{\binom{m}{|\ell|}}{\binom{m+|\ell|}{|\ell|}} \quad \text{except for } a_0^{(2m)} := 0$$

In Appendix 4.A we prove some bounds on the approximation error of the above formulas[3] for generic m. Usually such error bounds are only derived for some finite values of m, because that is sufficient in practice, but in order to prove our asymptotic results we need to derive more general results.

4.3 A generic model of quantum optimization algorithms

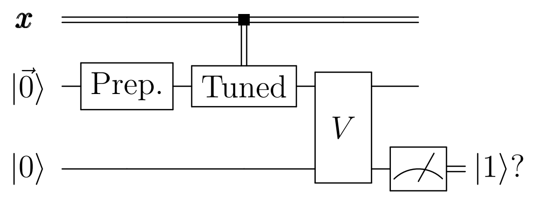
Variational quantum algorithms designed for quantum optimization and machine learning procedures have the following core idea: they approximate an optimal solution to a problem by tuning some parameters in a quantum circuit. The circuit usually consists of several simple gates, some of which have tunable real parameters, e.g., the angle of single qubit (controlled) rotation gates. Often, if there are enough tunable gates arranged in a nice topology, then there exist parameters that induce a unitary capable of achieving a close to optimal solution.

In such variational approaches, one can decompose the circuit into three parts each having a different role (see Figure 4.1). The circuit starts with a state preparation part which prepares the initial quantum state relevant for the problem. We call this part 'Prep.' in Figure 4.1. The middle part consists of tunable parameters x and fixed gates, which are together referred to as 'Tuned' in Figure 4.1. Finally, there is a verification circuit that evaluates the output state, and marks success if the auxiliary qubit is $|1\rangle$. We denote the verification process by V in Figure 4.1. The quality of the circuit (for parameter $\boldsymbol{x}$) is assessed by the probability of measuring the auxiliary qubit and obtaining 1.

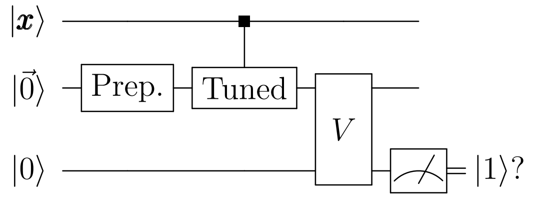
One can think of the tunable circuit as being tuned in a classical way as shown in Figure 4.1a or a quantum way as in Figure 4.1b. In the classical case, the parameters can be thought of as being manually set. Alternatively, the parameters can be quantum variables represented by qubits. The advantage of the latter is that it allows us to use quantum techniques to speedup optimization algorithms. However the drawback is that it requires more qubits to represent the parameters and requires implementation of additional controlled-gates, see for example, Fig. 4.1c.

Let us denote by $U(\boldsymbol{x})$ the circuit in Figure 4.1a and the corresponding circuit in Figure 4.1b as $U := \sum_{\boldsymbol{x}} |\boldsymbol{x}\rangle\langle\boldsymbol{x}| \otimes U(\boldsymbol{x})$. The goal in these optimization problems

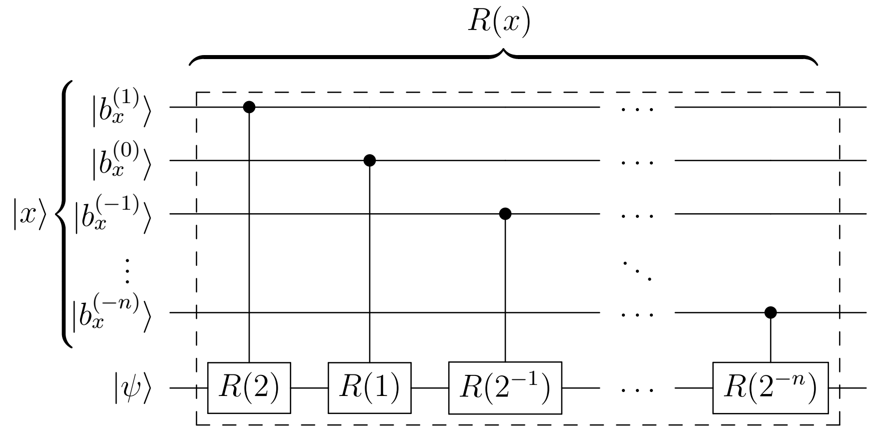
[3]One can read out the coefficients described in Definition 4.2.4 from the second row of the inverse of the Vandermonde matrix, as Arjan Cornelissen pointed out to us.



(a) A classically tunable circuit



(b) A quantumly tunable circuit



(c) A 2^{-n} precisely tunable rotation gate $R(x)$ for the fixed-point binary parameter $x = b_1 b_0.b_{-1} \cdots b_{-n}$.

Figure 4.1. Two different approaches to tunable quantum optimization. The circuit on the top left has classically set parameters $|\boldsymbol{x}\rangle$ (represented as a vector of fixed point binary numbers), whereas the circuit on the top right has parameters $\boldsymbol{x}$ described by an array of qubits $|\boldsymbol{x}\rangle$. The black squares connected to the 'Tuned' circuit indicate non-trivial control structure for which an example is presented on the bottom figure, showing how to implement a quantumly tunable rotation gate built from simple controlled rotation gates.

is to find the optimal parameters (i.e., $\boldsymbol{x}$) which maximize the probability of obtaining 1 after the final measurement, thereby solving the problem

$$\operatorname*{argmax}_{\boldsymbol{x}}\, p(\boldsymbol{x}), \text{ where } p(\boldsymbol{x}) = \left\|(|1\rangle\langle 1| \otimes I)U(\boldsymbol{x})|\vec{0}\rangle\right\|^2. \tag{4.7}$$

A well-known technique to solve continuous-variable optimization problems like the one above is gradient-ascent. In practice, gradient-based methods represent one of the most commonly used paradigms for solving continuous optimization problems.

4.3.1 Classical gradient ascent algorithm

As we discussed earlier, finding globally optimal parameters for optimization problems (4.7) is often hard. Therefore in practice one usually relies on heuristics to find an approximate solution $\boldsymbol{x}_a$ such that $p(\boldsymbol{x}_a)$ is close to optimal. There are several heuristic optimization techniques that are often applied to handle such problems. One of the most common techniques is gradient ascent, which follows a greedy strategy to obtain the optimal solution. It simply follows the path of steepest ascent on the landscape of the objective function to find a solution that is, at least up to small local perturbations, optimal. Such solutions are called locally optimal.

A naïve gradient-based algorithm[4] can be described as follows: pick N random points $\{\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_N^{(0)}\}$. For each $i \in [N]$, compute $\nabla p(\mathbf{x}_i^{(0)})$, and take a δ-step in the direction of $\nabla p(\mathbf{x}_i^{(0)})$ leading to $\mathbf{x}_i^{(1)} = \mathbf{x}_i^{(0)} + \delta \nabla p(\mathbf{x}_i^{(0)})$ (for some step size $\delta > 0$). Repeat this procedure for T steps, obtaining $\mathbf{x}_i^{(T)}$ which has hopefully approached some local maxima of (4.1). Finally, take the maximum of $\{p(\mathbf{x}_1^{(T)}), \ldots, p(\mathbf{x}_N^{(T)})\}$ as an approximation to (4.1).

By using empirical estimation to evaluate the function to precision ε, under some mild smoothness assumption, we can compute the gradient to ε-precision in the ℓ_∞ norm using $\widetilde{\mathcal{O}}\left(\frac{d}{\varepsilon^2}\right)$ samples. This way the algorithm uses the quantum circuit $U(\boldsymbol{x})$ a total number $\widetilde{\mathcal{O}}(NTd/\varepsilon^2)$ times.

4.3.2 Quantum speedups to the classical algorithm

Now, let us consider the possible quantum speedups to this naïve gradient ascent algorithm discussed in the previous section. The most basic improvement which works even for classically controlled circuits (Figure 4.1a) is to estimate the probability $p(\boldsymbol{x})$ in Step 5 using quantum amplitude estimation rather than doing repeated measurements and taking the average. If one wants to determine the value $p(\boldsymbol{x})$ up to error ε for some fixed $\boldsymbol{x}$, the quantum approach uses the circuit $\mathcal{O}(1/\varepsilon)$ times, whereas the classical statistical method would require $\Omega(1/\varepsilon^2)$ repetitions, due to the additive property of variances of uncorrelated random variables. Although this is a natural improvement, which does not require much additional quantum resources, many papers that describe a similar procedure do not mention it.

Another quantum improvement can be achieved [Bul05a, LPL14] by using Grover search, which requires a quantumly controlled circuit like in Figure 4.1b. Let $P(\boldsymbol{z})$ denote the probability that for a randomly chosen starting point $\boldsymbol{x}_0$ we get $\boldsymbol{x}_T = \boldsymbol{z}$, i.e., we end up with $\boldsymbol{z}$ after performing T gradient steps. Let $\tilde{p}$ be a value such that $P(p(\boldsymbol{z}) \geq \tilde{p}) \geq 1/N$. If we use N randomly chosen initial

[4]There are more sophisticated versions of gradient ascent, but for simplicity here we discuss a very basic version.

points, then with high probability at least one initial point will yield a point $\boldsymbol{x}_T$ with $p(\boldsymbol{x}_T) \geq \tilde{p}$.[5] If we use the quantum maximum-finding algorithm [DH96] or more precisely one if its generalizations [NW99, vAGGdW17], we can reduce the number of repetitions to $O(\sqrt{N})$ and still find a point $\boldsymbol{x}_T$ having $p(\boldsymbol{x}_T) \geq \tilde{p}$ with high probability. Due to reversability, we need to maintain all points visited during the gradient ascent algorithm, thereby possibly introducing a significant overhead in the number of qubits used.

However, there is a drawback using Grover search-based techniques. The disadvantage of quantum maximum finding approach over classical methods is, that the amount of time it takes to reach a local maximum using the gradient ascent might vary a lot. The reason is that classically, once we reached a local maximum we can start examining the next starting point, whereas if we use Grover search we do the gradient updates in superposition so we need to run the procedure for the largest possible number of gradient steps. To reduce this disadvantage one could use variable time amplitude amplification techniques introduced by Ambainis [Amb12], however, we leave such investigations for future work.

Method:	Sampling alg.	+Amp. est.	+Grover search	+**This work**
Complexity:	$\widetilde{O}(TNd/\varepsilon^2)$	$\widetilde{O}(TNd/\varepsilon)$	$\widetilde{O}\left(T\sqrt{N}d/\varepsilon\right)$	$\widetilde{O}\left(T\sqrt{Nd}/\varepsilon\right)$

Table 4.1. Quantum speedups for the naïve gradient-ascent algorithm

Our contribution. We show a quadratic speedup in d – the number of control parameters. For this we also need to use a quantumly controlled circuit, but the overhead in the number of qubits is much smaller than in the previous Grover-type speedup. The underlying quantum technique crucially relies on the quantum Fourier transform as it is based on an improved version of Jordan's gradient computation [Jor05] algorithm. We can optionally combine this speedup with the above-mentioned maximum finding improvement, which then gives a quantum algorithm that uses the quantumly controlled circuit (Figure 4.1b) $\widetilde{O}\left(T\sqrt{Nd}/\varepsilon\right)$ times, and achieves essentially the same guarantees as the classical algorithm. Therefore we can achieve a quadratic speedup in terms of all parameters except in T and obtain an overall complexity of $\widetilde{O}\left(T\sqrt{Nd/\varepsilon}\right)$. For a summary of the speedups see Table 4.1.

[5]I.e., with high probability, we will find a point from the top $1/N$ percentile of the points regarding the objective function $p(\boldsymbol{z})$.

4.4 The three natural input oracle models

As we discussed in the previous section, the optimization problem associated with Figure 4.1 was to maximize

$$\max_{\boldsymbol{x}} p(\boldsymbol{x}) = \max_{\boldsymbol{x}} \left\| (|1\rangle\langle 1| \otimes I) U(\boldsymbol{x}) |\vec{0}\rangle \right\|^2 .$$

Typically, one should think of U as the unitary corresponding to some variational quantum circuit or parametrized quantum algorithm. In more abstract terms, we can view U as a *probability oracle* that maps $|\vec{0}\rangle|\boldsymbol{x}\rangle$ to some state, such that the probability of obtaining 1 on measuring the last qubit is $p(\boldsymbol{x})$. This measurement probability serves as a "benchmark score" for the corresponding unitary U with respect to the vector of parameters $\boldsymbol{x}$.

4.4.1. Definition (Probability oracle). We say that $U_p : \mathcal{H}_{\mathrm{aux}} \otimes \mathcal{H} \to \mathcal{H}_{\mathrm{aux}} \otimes \mathcal{H}$ is a probability oracle for the function $p : X \to [0, 1]$, if $\{|x\rangle : x \in X\}$ is an orthonormal basis of the Hilbert space $\mathcal{H}$, and for all $x \in X$ it acts as

$$U_p \colon |\vec{0}\rangle|x\rangle \mapsto \left(\sqrt{p(x)}|1\rangle|\psi_x^{(1)}\rangle + \sqrt{1 - p(x)}|0\rangle|\psi_x^{(0)}\rangle \right)|x\rangle,$$

where $|\psi_x^{(1)}\rangle$ and $|\psi_x^{(1)}\rangle$ are arbitrary (normalized) quantum states.

This oracle model is not commonly used in quantum algorithms, therefore we need to convert it to a different format, for example to a *phase oracle*, in order to use standard quantum techniques.

4.4.2. Definition (Phase oracle). We say that $\mathrm{O}_f : \mathcal{H}_{\mathrm{aux}} \otimes \mathcal{H} \to \mathcal{H}_{\mathrm{aux}} \otimes \mathcal{H}$ is a phase oracle for $f : X \to [-1, 1]$, if $\{|x\rangle : x \in X\}$ is an orthonormal basis of the Hilbert space $\mathcal{H}$, and for all $x \in X$ it acts as

$$\mathrm{O}_f : |\vec{0}\rangle|x\rangle \mapsto e^{if(x)}|\vec{0}\rangle|x\rangle.$$

There is a third type of oracle that is commonly used in (quantum) algorithms, namely the *binary oracle* that outputs a finite-precision binary representation of the probability [WKS15]. This is the input model used in the work of Jordan [Jor05], but in fact the first step of his algorithm converts this oracle to a phase oracle.

4.4.3. Definition (Binary oracle). For $\eta \in \mathbb{R}_+$, we say $B_f^\eta : \mathcal{H} \otimes \mathcal{H}_{\mathrm{aux}} \to \mathcal{H} \otimes \mathcal{H}_{\mathrm{aux}}$ is an η-accurate binary oracle for $f : X \to \mathbb{R}$, if $\{|x\rangle : x \in X\}$ is an orthonormal basis of the Hilbert space $\mathcal{H}$, and for all $x \in X$ it acts as

$$B_f^\eta : |x\rangle|\vec{0}\rangle \mapsto |x\rangle|\tilde{f}(x)\rangle,$$

where $|\tilde{f}(x)\rangle$ is a fixed-point binary number satisfying $|\tilde{f}(x) - f(x)| \leq \eta$. We denote the cost of one query to B_f^η by $C(\eta)$.[6,7]

Note that using amplitude estimation[8] it is possible to turn a probability oracle into a binary oracle and then convert a binary oracle to a phase oracle. However, it is more efficient to avoid this analogue-digital-analogue conversion, and directly convert a probability oracle to a phase oracle.

4.4.1 Converting a probability oracle to a phase oracle

We use block-encoding and Hamiltonian simulation techniques introduced in Chapter 3 to implement the conversion efficiently. We implement (fractional) phase oracles by first turning a probability oracle to a block-encoding of the diagonal matrix containing the probabilities, then use the Hamiltonian simulation result described in Section 3.4.1.

Let U_p be a probability oracle, observe that

$$(\langle\vec{0}| \otimes I)\big(U_p^\dagger(Z \otimes I)U_p\big)(|\vec{0}\rangle \otimes I) = \mathrm{diag}(1 - 2p(x)).$$

Therefore $U_p^\dagger(Z\otimes I)U_p$ is a block-encoding of a diagonal matrix with diagonal entries $(1-2p(x))$. Moreover, we can implement a controlled version of this unitary by simply replacing Z with a controlled-Z. By Theorem 3.4.3 and Corollary 3.4.5 we get the following result:

4.4.4. Corollary (Probability to phase oracle). *Suppose that we are given a probability oracle for $p(x)\colon X \to [0,1]$. Let $t \in \mathbb{R}$ and $f(x) = tp(x)$ for all $x \in X$. We can implement an ε-approximate phase oracle O_f with query complexity $\mathcal{O}(|t| + \log(1/\varepsilon))$, i.e., that many uses of U_p and its inverse. Moreover, if $|\vec{0}\rangle = |0\rangle^{\otimes a}$ is an a-qubit state, then this query complexity bound times $\mathcal{O}(a)$ upper bounds the gate complexity.*

Form this we see that a probability oracle can be efficiently converted to a (fractional) phase oracle. As we have shown in Corollary 3.4.19, if $|f| \leq 1$, then with a similar logarithmic overhead a phase oracle can be converted to fractional phase oracle. For these reasons, and for simplicity, throughout this chapter we assume that a fractional phase query has the same cost as a non-fractional/full phase query.

[6]One could also consider a more general definition allowing the oracle to output a superposition of η-accurate answers.

[7]The cost function would typically be polylog$(1/\eta)$ for functions that can be calculated using a classical circuit. However, when the binary oracle is obtained via quantum phase estimation this cost is typically $1/\eta$.

[8]In some cases people use sampling and classical statistics to learn this probability. However amplitude estimation is quadratically more efficient. Typically one can improve sampling procedures quadratically using quantum techniques [Mon15, HM18].

4.4.5. DEFINITION (Fractional phase oracle). For $r \in [-1,1]$ we say that $\mathrm{O}_{rf} : \mathcal{H}_{\mathrm{aux}} \otimes \mathcal{H} \to \mathcal{H}_{\mathrm{aux}} \otimes \mathcal{H}$ is a *fractional query*[9] phase oracle for $f : X \to [-1,1]$, if $\{|x\rangle : x \in X\}$ is an orthonormal basis of the Hilbert space $\mathcal{H}$, and for all $x \in X$ it acts as

$$\mathrm{O}_{rf} : |\vec{0}\rangle|x\rangle \mapsto e^{irf(x)}|\vec{0}\rangle|x\rangle.$$

4.4.2 Converting a phase oracle to a probability oracle

Using the techniques of Chapter 3 we can also implement the inverse conversion. Starting from a phase oracle for $\frac{1}{3} \leq p(x) \leq \frac{2}{3}$ we can implement its logarithm by Corollary 3.4.18, which results in the block-encoding of a diagonal matrix with $p(x)$ on the diagonal. We can then take the square-root of this operator by Corollary 3.4.14, which then results in a probability oracle: apply the block-encoding of $\mathrm{diag}(\sqrt{p(x)})$ and check if the auxiliary qubits remained zero; if not set the flag qubit to 1. This procedure can be slightly improved as follows.

4.4.6. PROPOSITION. *Let $\varepsilon, \delta \in (0, 1/2)$, and $p : X \to [\delta, 1-\delta]$. Then with $\mathcal{O}(\log(1/\varepsilon)/\delta)$ uses of the (controlled) O_p, $\mathrm{O}_p^\dagger$ oracles and other two-qubit gates, we can implement a probability oracle*

$$U_p : |x\rangle|0\rangle^{\otimes a}|0\rangle \mapsto |x\rangle \otimes \left(\sqrt{\tilde{p}(x)}|0\rangle^{\otimes a}|0\rangle + \sqrt{1-\tilde{p}(x)}|\Phi^\perp\rangle|1\rangle\right),$$

where $\left|\sqrt{\tilde{p}(x)} - \sqrt{p(x)}\right| \leq \varepsilon$ *for every* $x \in X$, $\langle 0^{\otimes a}|\Phi^\perp\rangle = 0$ *and* $a = \mathcal{O}(1)$.

Proof: Let cU denote the controlled version of the unitary $U := e^i\mathrm{O}_p^\dagger$ controlled by the first qubit. We use the trick of Corollary 3.4.18 and work with the operator

$$-i(\langle +| \otimes I)cU^\dagger(ZX \otimes I)cU(|+\rangle \otimes I) = \mathrm{diag}(\sin(1-p(x))).$$

Applying singular value transformation (Corollary 2.3.8) using a bounded even polynomial $\widetilde{P}(z)$, ε-approximating $\sqrt{1-\arcsin(z)}$ on the domain $[\frac{\delta}{2}, \sin(1) - \frac{\delta}{2}]$ would solve our problem, since for $p \in [\delta, 1-\delta]$ we have $\sin(1-p) \in [\frac{\delta}{2}, \sin(1) - \frac{\delta}{2}]$.

We finish the proof by constructing an even approximating polynomial of degree $\mathcal{O}(\log(1/\varepsilon)/\delta)$. For this we first analyze the Taylor series $\sqrt{1-\arcsin(z)} = \sum_{j=0}^{\infty} a_j z^j$, by observing that $\arcsin(z) = \sum_{\ell=0}^{\infty} b_\ell z^\ell = \sum_{i=0}^{\infty} \binom{2i}{i} \frac{2^{-2i}}{2i+1} z^{2i+1}$ for all $z \in [-1,1]$, and $\sqrt{1-y} = \sum_{k=0}^{\infty} c_k y^k = \sum_{k=0}^{\infty} \binom{1/2}{k}(-y)^k$ for all $y \in [-1,1]$. Since $b_\ell \geq 0$ for all $\ell \in \mathbb{N}_0$ and $c_k \leq 0$ for all $k \in \mathbb{N}$ we get $a_j \leq 0$ for all $j \in \mathbb{N}$. It is also easy to see that for all $z \in [\sin(-1), \sin(1)]$ we have $\sum_{j=0}^{\infty} a_j z^j = \sqrt{1-\arcsin(z)}$, so

$$\sum_{j=0}^{\infty} |a_j| \sin^j(1) = |a_0| - (\sum_{j=0}^{\infty} a_j \sin^j(1) - a_0) = 1 - (\sqrt{1-\arcsin(\sin(1))} - 1) = 2.$$

[9] Note that this fractional query is more general than the fractional query introduced by Cleve et al. [CGM+09], because we have a continuous phase rather than discrete. Thus, the results of [CGM+09] do not give a way to implement a generic fractional query to our phase oracle O_f. However, we can use Corollary 3.4.19 instead.

By Corollary 3.4.12 (choosing $x_0 \leftarrow 0$, $r \leftarrow \sin(1) - \delta/8$, $\delta \leftarrow \delta/8$ and $B \leftarrow 2$) we can construct an approximating polynomial $P(z)$ of degree $\mathcal{O}(\log(1/\varepsilon)/\delta)$, which is bounded by 2 in absolute value on the domain $z \in [-1,1]$, and for all $z \in [\frac{\delta}{4}, \sin(1) - \frac{\delta}{4}]$ it is $(\varepsilon/16)$-close to $\sqrt{1-\arcsin(z)}$. By Lemma 3.2.2 we can construct a polynomial $S(z)$ of degree $\mathcal{O}(\log(1/\varepsilon)/\delta)$, such that $S(z) \in [0, 1-\varepsilon/16]$ for all $z \in [-2,2]$, $S(z) \leq \varepsilon/64$ for all $z \in [-2, -\delta/8]$ and $S(z) \geq 1 - \varepsilon/8$ for all $z \in [\delta/8, 2]$. Taking the even part of $2S(z - \frac{3\delta}{8})P(z)S(1 - \frac{3\delta}{8} - z)$ gives our desired bounded polynomial approximation $\widetilde{P}(z)$, which is ε-close to $\sqrt{1-\arcsin(z)}$ on the domain $[\frac{\delta}{2}, \sin(1) - \frac{\delta}{2}]$, and is bounded by 1 in absolute value on $[-1,1]$. $\square$

4.5 Improved quantum gradient computation algorithm

4.5.1 Overview of Jordan's algorithm

Stephen Jordan constructed a surprisingly simple quantum algorithm [Jor05, Bul05b] that can approximately calculate the d-dimensional gradient of a function $f : \mathbb{R}^d \to \mathbb{R}$ with a *single* evaluation of f. In contrast, using standard classical techniques, one would use $d+1$ function evaluations to calculate the gradient at a point $\boldsymbol{x} \in \mathbb{R}^d$: one can first evaluate $f(\boldsymbol{x})$ and then, for every $i \in [d]$, evaluate $f(\boldsymbol{x} + \delta \boldsymbol{e}_i)$ (for some $\delta > 0$) to get an approximation of the gradient in direction i using the standard formula

$$\nabla_i f(\boldsymbol{x}) \approx \frac{f(\boldsymbol{x} + \delta \boldsymbol{e}_i) - f(\boldsymbol{x})}{\delta}.$$

The basic idea of Jordan's quantum algorithm [Jor05] is simple. First make two observations. Observe that if f is twice differentiable at $\boldsymbol{x}$, then $f(\boldsymbol{x} + \boldsymbol{\delta}) = f(\boldsymbol{x}) + \nabla f \cdot \boldsymbol{\delta} + \mathcal{O}(\|\boldsymbol{\delta}\|^2)$, which in particular implies that for small $\|\boldsymbol{\delta}\|$, the function f is very close to being affine linear. The second observation is that, using the value of $f(\boldsymbol{x} + \boldsymbol{\delta})$, one can implement a phase oracle:

$$\mathrm{O}_{2\pi S f} : |\boldsymbol{\delta}\rangle \mapsto e^{2\pi i S f(\boldsymbol{x}+\boldsymbol{\delta})} |\boldsymbol{\delta}\rangle \approx e^{2\pi i S f(\boldsymbol{x})} e^{2\pi i S \nabla f \cdot \boldsymbol{\delta}} |\boldsymbol{\delta}\rangle \tag{4.8}$$

for a scaling factor $S > 0$, where the approximation uses $f(\boldsymbol{x} + \boldsymbol{\delta}) \approx f(\boldsymbol{x}) + \nabla f \cdot \boldsymbol{\delta}$ for small $\|\boldsymbol{\delta}\|$. The role of S is to make the phases appropriate for the final quantum Fourier transform.

Sketch of the algorithm. Assume that all real vectors are expressed up to some finite amount of precision. In order to compute the gradient at $\boldsymbol{x}$, the algorithm starts with a uniform superposition $|\psi\rangle = \frac{1}{\sqrt{|G_{\boldsymbol{x}}^d|}} \sum_{\boldsymbol{\delta} \in G_{\boldsymbol{x}}^d} |\boldsymbol{\delta}\rangle$ over the points of a sufficiently small discretized d-dimensional grid $G_{\boldsymbol{x}}^d$ around $\boldsymbol{x}$, and applies the phase oracle $\mathrm{O}_{2\pi Sf}$ (in Eq. 4.8) to $|\psi\rangle$. Next, the inverse quantum Fourier transform is applied to the resulting state and each register is measured to obtain the gradient of f at $\boldsymbol{x}$ approximately. Due to approximate linearity of the phase, as described in Eq. (4.8), applying the inverse Fourier transform will approximately give us the gradient. This algorithm uses $\mathrm{O}_{2\pi Sf}$ once and Jordan showed how to implement $\mathrm{O}_{2\pi Sf}$ using one sufficiently precise function evaluation.

In order to improve the accuracy of the simple algorithm above, one could use some natural tricks. If f is twice continuously differentiable, it is easy to see that the smaller the grid $G_{\boldsymbol{x}}^{d}$ becomes, the closer the function gets to being linear. This gives control over the precision of the algorithm. However, if we "zoom-in" to the function using a smaller grid, the difference between nearby function values becomes smaller, making it harder the distinguish them and thus increasing the complexity of the algorithm proportionally.

Also, it is well known that if the derivative is calculated based on the differences between the points $(f(x-\delta/2),\ f(x+\delta/2))$ rather than $(f(x),\ f(x+\delta))$, then one gets a better approximation since the quadratic correction term cancels. To mimic this trick, Jordan chose a symmetric grid $G_{\boldsymbol{x}}^{d}$ around $\mathbf{0}$.

Complexity of the algorithm For Jordan's algorithm, it remains to pick the parameters of the grid and the constant S in Eq. 4.8. For simplicity, assume that $\|\nabla f(\boldsymbol{x})\|_{\infty} \leq 1$, and suppose we want to approximate $\nabla f(\boldsymbol{x})$ coordinate-wise up to precision ε, with high success probability. Under the assumption that "the 2nd partial derivatives of f have a magnitude of approximately D_2", Jordan argues[10] that choosing $G_{\boldsymbol{x}}^{d}$ to be a d-dimensional hypercube with edge length $\ell \approx \frac{\varepsilon}{D_2\sqrt{d}}$ and with $N \approx \frac{1}{\varepsilon}$ equally spaced grid points in each dimension, the quantum algorithm yields an ε-approximate gradient by setting $S = \frac{N}{\ell} \approx \frac{D_2\sqrt{d}}{\varepsilon^2}$. Moreover, since the Fourier transform is relatively insensitive to local phase errors it is sufficient to implement the phase $Sf(\boldsymbol{x}+\boldsymbol{\delta})$ up to some constant, say 1% accuracy.

During the derivation of the above parameters Jordan makes the assumption, that the third and higher-order terms of the Taylor expansion of f around $\boldsymbol{x}$ are negligible. However, it is not clear from his work [Jor05], how to actually handle the case when they are non-negligible. This could be a cause of concern for the runtime analysis, since these higher-order terms potentially introduce a dependence on the dimension d.

Finally, in order to assess the complexity of his algorithm, Jordan considers the Binary oracle input model of Definition 4.4.3. This input model captures functions that are evaluated numerically using, say, an arithmetic circuit. Typically, the number of one and two-qubit gates needed to evaluate such functions up to n digits of precision is polynomial in n and d. However, this input model does not

[10]We specifically refer to equation (4) in [Jor05] (equation (3) in the arXiv version), and the discussion afterwards. Note that our precision parameter ε corresponds to the uncertainty parameter σ in [Jor05].

fit the quantum optimization framework that we introduced in Section 4.3.

Our improvements We improve on the results of Jordan [Jor05] in a number of ways. Jordan [Jor05] argued that evaluating the function on a superposition of grid-points symmetrically arranged around $\mathbf{0}$ is analogous to using a simple central-difference formula. We also place the grid symmetrically, but we realized that it is possible to directly use central-difference formulas, which is the main idea behind our modified algorithm.

As discussed in the introduction, we realized that in applications of the gradient descent algorithm for optimization problems, it is natural to assume access to a phase oracle $\mathrm{O}_f : |\boldsymbol{x}\rangle \mapsto e^{if(\boldsymbol{x})}|\boldsymbol{x}\rangle$ (allowing fractional queries as well – see Definition 4.4.5) instead of the Binary access oracle B_f^η. If we wanted to use Jordan's original algorithm in order to obtain the gradient with accuracy ε, we need to implement the query oracle O_f^S by setting $S \approx D_2\sqrt{d}/\varepsilon^2$, which can be achieved using $\lceil S \rceil$ consecutive (fractional) queries. Although it gives a square-root dependence on d it scales as $\mathcal{O}(1/\varepsilon^2)$ with the precision. In this work, we employ the phase oracle model and improve the quadratic dependence on $1/\varepsilon$ to essentially linear. Additionally, we rigorously prove the square-root scaling with d under reasonable assumptions on the derivatives of f. We describe our algorithm in Section 4.5.3 and later also show that, for a class of smooth functions, the $\widetilde{\mathcal{O}}\left(\sqrt{d}/\varepsilon\right)$-query complexity is optimal up to poly-logarithmic factors.

4.5.2 Analysis of Jordan's algorithm

In this section we describe Jordan's algorithm and provide a generic analysis of its behavior. In the next subsection we combine these results with our finite-difference methods. Before describing the algorithm, we introduce appropriate representation of our qubit strings suitable for fixed-point arithmetics.

4.5.1. Definition. For every $b \in \{0,1\}^n$, let $j^{(b)} \in \{0,\ldots,2^n-1\}$ be the integer corresponding to the binary string $b = (b_1,\ldots,b_n)$. We label the n-qubit basis state $|b_1\rangle|b_2\rangle\cdots|b_n\rangle$ by $|x^{(b)}\rangle$, where

$$x^{(b)} = \frac{j^{(b)}}{2^n} - \frac{1}{2} + 2^{-n-1}.$$

We denote the set of corresponding labels as

$$G_n := \left\{\frac{j^{(b)}}{2^n} - \frac{1}{2} + 2^{-n-1} : j^{(b)} \in \{0,\ldots,2^n-1\}\right\}.$$

Note that there is a bijection between $\{j^{(b)}\}_{b\in\{0,1\}^n}$ and $\{x^{(b)}\}_{b\in\{0,1\}^n}$, so we will use $|x^{(b)}\rangle$ and $|j^{(b)}\rangle$ interchangeably in Claim 4.5.3. In the rest of this section we always label n-qubit basis states by elements of G_n.

4.5.2. Definition. For $x \in G_n$ we define the Fourier transform of a state $|x\rangle$ as

$$QFT_{G_n} : |x\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k \in G_n} e^{2\pi i 2^n x k} |k\rangle.$$

4.5.3. Proposition. *This unitary is the same as the usual quantum Fourier transform (QFT_n) up to conjugation with a tensor product of n single-qubit gates.*

Proof:
For bitstrings $b, c \in \{0,1\}^n$, let $x^{(b)} \in G_n$ and $j^{(b)} \in \{0, \ldots, 2^n - 1\}$ be as defined in Definition 4.5.1. Then QFT_{G_n} acts on $|j^{(b)}\rangle \equiv |x^{(b)}\rangle$ as

$$\begin{aligned} |x^{(b)}\rangle \mapsto & \frac{1}{\sqrt{2^n}} \sum_{x^{(c)} \in G_n} e^{2\pi i 2^n x^{(b)} x^{(c)}} |x^{(c)}\rangle \\ \equiv & \frac{1}{\sqrt{2^n}} \sum_{j^{(c)} \in \{0,\ldots,2^n-1\}} e^{2\pi i 2^n \left(\frac{j^{(b)}}{2^n} - \frac{1}{2} + 2^{-n-1}\right)\left(\frac{j^{(c)}}{2^n} - \frac{1}{2} + 2^{-n-1}\right)} |j^{(c)}\rangle \\ \equiv & \frac{1}{\sqrt{2^n}} \sum_{j^{(c)} \in \{0,\ldots,2^n-1\}} e^{2\pi i \left(\frac{j^{(b)} j^{(c)}}{2^n} - \left(j^{(b)} + j^{(c)}\right)\left(\frac{1}{2} + 2^{-n-1}\right) + \left(2^{n-2} - \frac{1}{2} + 2^{-n-2}\right)\right)} |j^{(c)}\rangle. \end{aligned}$$

Using the usual quantum Fourier transform

$$QFT_n : |j^{(b)}\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{j^{(c)} \in \{0,\ldots,2^n-1\}} e^{2\pi i 2^{-n} j^{(b)} j^{(c)}} |j^{(c)}\rangle$$

and the phase unitary U acting for every $j^{(b)} \in \{0, \ldots, 2^n - 1\}$ as

$$U : |j^{(b)}\rangle \mapsto e^{2\pi i \left(-j^{(b)}\left(\frac{1}{2} + 2^{-n-1}\right) + \left(2^{n-2} - \frac{1}{2} + 2^{-n-2}\right)/2\right)} |j^{(b)}\rangle,$$

it is easy to see that

$$QFT_{G_n} = U \cdot QFT_n \cdot U.$$

By writing $j^{(b)}$ in binary it is easy to see that U is a tensor product of n phase gates. $\square$

Now we are ready to precisely describe Jordan's quantum gradient computation algorithm.

Algorithm 4.1 Jordan's quantum gradient computation algorithm

```
    **Registers:** Use $n$-qubit input registers $|x_1\rangle|x_2\rangle\cdots|x_d\rangle$ with each qubit initialized to $|0\rangle$.
    **Labels:** Label the $n$-qubit states of each register with elements of $G_n$ as in Definition 4.5.1.
    **Input:** A function $f : G_n^d \to \mathbb{R}$ with phase-oracle $\mathrm{O}_f$ access such that
        $\mathrm{O}_f^{\pi 2^{n+1}}|x_1\rangle|x_2\rangle\cdots|x_d\rangle = e^{2\pi i 2^n f(x_1,x_2,\ldots,x_d)}|x_1\rangle|x_2\rangle\cdots|x_d\rangle.$
1: **Init** Apply a Hadamard transform to each qubit of the input registers.
2: **Oracle call** Apply the modified phase oracle $\mathrm{O}_f^{\pi 2^{n+1}}$ on the input registers.
3: **QFT**$_{G_n}^{-1}$ Fourier transform each register individually:
        $|x\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k\in G_n} e^{-2\pi i 2^n xk}|k\rangle.$
4: **Measure** each input register $j$ and denote the measurement outcome by $k_j$.
5: **Output** $(k_1, k_2, \ldots, k_d)$ as the estimation for the gradient.
```

4.5.4. Lemma. *Let $N = 2^n$, $c \in \mathbb{R}$ and $\boldsymbol{g} \in \mathbb{R}^d$ such that $\|\boldsymbol{g}\|_\infty \leq 1/3$. If $f : G_n^d \to \mathbb{R}$ is such that*

$$|f(\boldsymbol{x}) - \boldsymbol{g}\cdot\boldsymbol{x} - c| \leq \frac{1}{42\pi N}, \tag{4.9}$$

for all but a $1/1000$ fraction of the points $\boldsymbol{x} \in G_n^d$, then the output of Algorithm 4.1 satisfies:

$$\Pr[|k_i - g_i| > 4/N] \leq 1/3 \quad \textit{for every } i \in [d].$$

Proof:
First, note $|G_n| = N$ from Definition 4.5.1. Consider the following quantum states

$$|\phi\rangle := \frac{1}{\sqrt{N^d}} \sum_{\boldsymbol{x}\in G_n^d} e^{2\pi i N f(\boldsymbol{x})}|\boldsymbol{x}\rangle \qquad \text{and} \qquad |\psi\rangle := \frac{1}{\sqrt{N^d}} \sum_{\boldsymbol{x}\in G_n^d} e^{2\pi i N(\boldsymbol{g}\cdot\boldsymbol{x}+c)}|\boldsymbol{x}\rangle.$$

Note that $|\phi\rangle$ is the state we obtain in Algorithm 4.1 after line 2 and $|\psi\rangle$ is its "ideal version" that we try to approximate with $|\phi\rangle$. Observe that the "ideal" $|\psi\rangle$ is actually a product state:

$$|\psi\rangle = e^{2\pi i N c} \cdot \Big(\frac{1}{\sqrt{N}} \sum_{x_1\in G_n} e^{2\pi i N g_1\cdot x_1}|x_1\rangle\Big) \otimes \cdots \otimes \Big(\frac{1}{\sqrt{N}} \sum_{x_d\in G_n} e^{2\pi i N g_d\cdot x_d}|x_d\rangle\Big).$$

It is easy to see that after applying the inverse Fourier transform to each register separately (as in line 3) to $|\psi\rangle$, we obtain the state

$$e^{2\pi i N c} \cdot \Big(\frac{1}{N} \sum_{x_1,k_1\in G_n^2} e^{2\pi i N x_1(g_1-k_1)}|k_1\rangle\Big) \otimes \cdots \otimes \Big(\frac{1}{N} \sum_{x_d,k_d\in G_n^2} e^{2\pi i N x_d(g_d-k_d)}|k_d\rangle\Big).$$

Suppose we make a measurement and observe $(k_1,\dots,k_d)$. As shown in the analysis of phase estimation [NC00, Section 5.2.1], we have the following[11]: for every $i\in[d]$ (for a fixed accuracy parameter $\kappa>1$), the following holds:

$$\Pr\left[|k_i-g_i|>\frac{\kappa}{N}\right]\leq\frac{1}{2(\kappa-1)}\qquad\text{for every } i\in[d].$$

By fixing $\kappa=4$, we obtain the desired conclusion of the theorem, i.e., if we had access to $|\psi\rangle$ (instead of $|\phi\rangle$), then we would get a $4/N$-approximation of each coordinate of the gradient with probability at least $5/6$. It remains to show that this probability does not change more than $1/3-1/6=1/6$ if we apply the Fourier transform to $|\phi\rangle$ instead of $|\psi\rangle$. As shown in Chapter 1, the difference in the probability of any measurement outcome on these states is bounded by twice the trace distance between $|\psi\rangle$ and $|\phi\rangle$ which is

$$\frac{1}{2}\left\||\psi\rangle\langle\psi|-|\phi\rangle\langle\phi|\right\|_1=\sqrt{1-|\langle\psi|\phi\rangle|^2}\leq\left\||\psi\rangle-|\phi\rangle\right\|.\tag{4.10}$$

Since the Fourier transform is unitary and does not change the Euclidean distance, it is sufficient to show that $\||\psi\rangle-|\phi\rangle\|\leq 1/12$ in order to conclude the theorem. Let $S\subseteq G_n^d$ denote the set of points satisfying Eq. (4.9). We conclude the proof of the theorem by showing $\||\psi\rangle-|\phi\rangle\|^2\leq(1/12)^2$:

$$\begin{aligned}
\||\phi\rangle-|\psi\rangle\|^2&=\frac{1}{N^d}\sum_{\boldsymbol{x}\in G_n^d}\left|e^{2\pi iNf(\boldsymbol{x})}-e^{2\pi iN(\boldsymbol{g}\cdot\boldsymbol{x}+c)}\right|^2\\
&=\frac{1}{N^d}\sum_{\boldsymbol{x}\in S}\left|e^{2\pi iNf(\boldsymbol{x})}-e^{2\pi iN(\boldsymbol{g}\cdot\boldsymbol{x}+c)}\right|^2+\frac{1}{N^d}\sum_{\boldsymbol{x}\in G_n^d\setminus S}\left|e^{2\pi iNf(\boldsymbol{x})}-e^{2\pi iN(\boldsymbol{g}\cdot\boldsymbol{x}+c)}\right|^2\\
&\leq\frac{1}{N^d}\sum_{\boldsymbol{x}\in S}|2\pi Nf(\boldsymbol{x})-2\pi N(\boldsymbol{g}\cdot\boldsymbol{x}+c)|^2+\frac{1}{N^d}\sum_{\boldsymbol{x}\in G_n^d\setminus S}4\\
&\qquad\qquad\qquad\qquad(\text{since } |e^{iz}-e^{iy}|\leq|z-y|)\\
&=\frac{1}{N^d}\sum_{\boldsymbol{x}\in S}(2\pi N)^2|f(\boldsymbol{x})-(\boldsymbol{g}\cdot\boldsymbol{x}+c)|^2+4\frac{|G_n^d\setminus S|}{N^d}\\
&\leq\frac{1}{N^d}\sum_{\boldsymbol{x}\in S}\left(\frac{1}{21}\right)^2+\frac{4}{1000}\qquad(\text{by the assumptions of the theorem})\\
&\leq\frac{1}{441}+\frac{1}{250}<\frac{1}{144}=\left(\frac{1}{12}\right)^2.
\end{aligned}$$

□

[11]Note that our Fourier transform is slightly altered, but the same proof applies as in [NC00, (5.34)]. In fact this result can be directly translated to our case by considering the unitary conjugations proven in Remark 4.5.3.

In the following theorem we assume that we have access to (a high power of) a phase oracle of a function f that is very well approximated by an affine linear function $\boldsymbol{g}\cdot\boldsymbol{z}+c$ on a hypergrid with edge-length $r\in\mathbb{R}$ around some $\boldsymbol{y}\in\mathbb{R}^d$. We show that if the approximation error is sufficiently low relative to the grid size r, then Algorithm 4.1 can compute an approximation of $\boldsymbol{g}$ (the “steepness”, similar to a gradient) with small query and gate complexity.

4.5.5. Theorem. *Let $c\in\mathbb{R}$, $\varepsilon<M\in\mathbb{R}_+$, $r,\rho\in\mathbb{R}_+$, and $\boldsymbol{y},\boldsymbol{g}\in\mathbb{R}^d$ such that $\|\boldsymbol{g}\|_\infty\leq M$. Let $n_\varepsilon:=\lceil\log_2(4/(r\varepsilon))\rceil$, $n_M:=\lceil\log_2(3rM)\rceil$ and $n:=n_\varepsilon+n_M$. Suppose $f:\left(\boldsymbol{y}+rG_n^d\right)\to\mathbb{R}$ is such that*

$$|f(\boldsymbol{y}+r\boldsymbol{x})-\boldsymbol{g}\cdot r\boldsymbol{x}-c|\leq\frac{\varepsilon r}{8\cdot 42\pi}$$

for all but a $1/1000$ fraction of the points $\boldsymbol{x}\in G_n^d$. If we have access to a phase oracle $\mathrm{O}:|\boldsymbol{x}\rangle\to e^{2\pi i 2^{n_\varepsilon}f(\boldsymbol{y}+r\boldsymbol{x})}|\boldsymbol{x}\rangle$ acting on $\mathcal{H}=\mathrm{Span}\{|\boldsymbol{x}\rangle:\boldsymbol{x}\in G_n^d\}$, then we can calculate a vector $\tilde{\boldsymbol{g}}\in\mathbb{R}^d$, simultaneously approximating all coordinates such that

$$\Pr[\,\|\tilde{\boldsymbol{g}}-\boldsymbol{g}\|_\infty>\varepsilon]\leq\rho,$$

with $\mathcal{O}\Big(\log\left(\frac{d}{\rho}\right)\Big)$ queries to O and with gate complexity

$$\mathcal{O}\left(d\log\left(\frac{d}{\rho}\right)\log\left(\frac{M}{\varepsilon}\right)\log\log\left(\frac{d}{\rho}\right)\log\log\left(\frac{M}{\varepsilon}\right)\right).$$

Proof:
Let $N_M:=2^{n_M}$, $N:=2^n$, and $h(\boldsymbol{x}):=\frac{f(\boldsymbol{y}+r\boldsymbol{x})}{N_M}$, then $\left|h(x)-\boldsymbol{g}\frac{r}{N_M}\boldsymbol{x}-\frac{c}{N_M}\right|\leq\frac{\varepsilon r}{8\cdot 42\pi N_M}\leq\frac{1}{42\pi N}$. Note that $\mathrm{O}=\mathrm{O}_h^{2\pi N}$, therefore Algorithm 4.1 yields an output $\tilde{\boldsymbol{g}}$, which, as shown by Lemma 4.5.4, is such that that for each $i\in[d]$ with probability at least $2/3$ we have $\left|\tilde{g}_i-\frac{r}{N_M}g_i\right|\leq\frac{4}{N}$. Thus also $\left|\frac{N_M}{r}\tilde{g}_i-g_i\right|\leq\frac{4N_M}{rN}\leq\varepsilon$. By repeating the procedure $\mathcal{O}(\log(d/\rho))$ times and taking the median coordinate-wise we get a vector $\tilde{\boldsymbol{g}}_{\mathrm{med}}$, such that $\|\tilde{\boldsymbol{g}}_{\mathrm{med}}-\boldsymbol{g}\|_\infty\leq\varepsilon$ with probability at least $(1-\rho)$.

The gate complexity statement follows from the fact that the complexity of Algorithm 4.1 is dominated by that of the d independent quantum Fourier transforms, each of which can be approximately implemented using $\mathcal{O}(n\log n)$ gates. We repeat the procedure $\mathcal{O}(\log(d/\rho))$ times, which amounts to $\mathcal{O}(d\log(d/\rho)n\log n)$ gates. At the end we get d groups of numbers each containing $\mathcal{O}(\log(d/\rho))$ numbers with n bits of precision. We can sort each group with a circuit having $\mathcal{O}(\log(d/\rho)\log\log(d/\rho)n\log n)$ gates.[12] So the final gate complexity is

$$\mathcal{O}(d\log(d/\rho)\log\log(d/\rho)n\log n),$$

which gives the stated gate complexity by observing that $n=\mathcal{O}(\log(M/\varepsilon))$. $\square$

[12]Note that using the median of medians algorithm [BFP+73] we could do this step with $\mathcal{O}(\log(d/\rho)n)$ time complexity, but this result probably does not apply to the circuit model, which is somewhat weaker than, e.g., a Turing machine or the RAM model.

4.5.3 Improved quantum gradient algorithm using higher-degree methods

Theorem 4.5.5 shows Jordan's algorithm works well if the function is very close to linear function over a large hyprecube. However, in general even highly regular functions tend to quickly diverge from their linear approximations. To tackle this problem we borrow ideas from numerical analysis and use higher-degree finite-difference formulas to extend the range of approximate linearity.

We will apply Jordan's algorithm to the approximate finite-difference linearization of the function rather than the function itself. We illustrate the main idea on a simple example: suppose we want to calculate the gradient at $\mathbf{0}$, then we could use the 2-point approximation $(f(\boldsymbol{x}) - f(-\boldsymbol{x}))/2$ instead of f, which has the advantage that it cancels out even-order contributions. The corresponding phase oracle $|\boldsymbol{x}\rangle \to e^{2^n\pi i(f(\boldsymbol{x})-f(-\boldsymbol{x}))}|\boldsymbol{x}\rangle$ is also easy to implement as the product of the oracles:

$$+ \text{ phase oracle: } |\boldsymbol{x}\rangle = e^{2^n\pi i f(\boldsymbol{x})}|\boldsymbol{x}\rangle \quad \text{and} \quad - \text{ phase oracle: } |\boldsymbol{x}\rangle = e^{-2^n\pi i f(-\boldsymbol{x})}|\boldsymbol{x}\rangle.$$

We use high-order central-difference approximation formulas. There is a variety of other related formulas [Li05], but we stick to the central difference because the absolute values of the coefficients in this formula scale favorably with the approximation degree. Since we only consider central differences, all our approximations have even degree, which is sufficient for our purposes as we are interested in the asymptotic scaling. Nevertheless, it is not difficult to generalize our approach using other formulas [Li05] that can provide odd-degree approximations as well.

Algorithm 4.2 Improved quantum gradient computation

Input: A point $\boldsymbol{x} \in \mathbb{R}^d$ and a function $f : \mathbb{R}^d \to \mathbb{R}$ with probability or (fractional) phase oracle access
Parameters: m : interpolation order; R : rescaling
Function transformation: $g(\boldsymbol{y}) := R \cdot f(\boldsymbol{x} + \boldsymbol{y}/R)$
Oracle implementation:

$$\mathrm{O}^t_{g_{(2m)}} := \prod_{\ell=-m}^{m} \left(I \otimes S^\dagger_\ell\right) \cdot O_f^{tRa_\ell^{(2m)}} \cdot \left(I \otimes S_\ell\right),$$

where $O_f^{tRa_\ell^{(2m)}}$ is implemented by Corollary 4.4.4 or as a product of (fractional) phase oracles; I acts on the ancilla qubits used by $O_f^{tRa_\ell^{(2m)}}$, and $S_\ell \colon |\boldsymbol{y}\rangle \mapsto |\boldsymbol{x} + \ell\boldsymbol{y}/R\rangle$ for all $\boldsymbol{y} \in G$ evaluation points used by Algorithm 4.1.
Use Jordan's Algorithm 4.1: compute $\nabla g_{(2m)}(\mathbf{0})$
Output: Approximation of $\nabla g_{(2m)}(\mathbf{0}) = \nabla f(\boldsymbol{x})$

In the following lemma, we show that if $f:\mathbb{R}\to\mathbb{R}$ is $(2m+1)$-times continuously differentiable, then the central-difference formula in Eq. (4.6) of Definition 4.2.4 is a good approximation to the derivative $f'(0)$. Eventually we will generalize this to the setting where $f:\mathbb{R}^d\to\mathbb{R}$.

4.5.6. Lemma. *Let $\delta\in\mathbb{R}_+$, $m\in\mathbb{N}$ and suppose $f:[-m\delta,m\delta]\to\mathbb{R}$ is $(2m+1)$-times differentiable. Then*

$$\left|f'(0)\delta - f_{(2m)}(\delta)\right| = \left|f'(0)\delta - \sum_{\ell=-m}^{m} a_\ell^{(2m)} f(\ell\delta)\right| \leq e^{-\frac{m}{2}}\left\|f^{(2m+1)}\right\|_\infty |\delta|^{2m+1}, \tag{4.11}$$

where $\left\|f^{(2m+1)}\right\|_\infty := \sup_{\xi\in[-m\delta,m\delta]}|f^{(2m+1)}(\xi)|$. Moreover,

$$\sum_{\ell=0}^{m}\left|a_\ell^{(2m)}\right| < \sum_{\ell=1}^{m}\frac{1}{\ell} \leq \ln(m)+1. \tag{4.12}$$

This lemma shows that for small enough δ the approximation error in (4.6) is upper bounded by a factor proportional to δ^{2m+1}. If $\left\|f^{(2m+1)}\right\|_\infty \leq c^m$ for all m and we choose $\delta\leq 1/c$, then the approximation error becomes exponentially small in m, motivating the use of higher-degree methods in our modified gradient computation algorithm. We generalize this statement to higher dimensions in Appendix 4.A, which leads to our first result regarding our improved algorithm (for the definition of the directional partial derivative $\partial_{\boldsymbol{r}}^{2m+1}f$ see Definition 4.2.3):

4.5.7. Theorem. *Let $m\in\mathbb{Z}_+$, $D\in\mathbb{R}_+$ and $B\geq 0$. Suppose $f:[-D,D]^d\to\mathbb{R}$ is given with (fractional) phase oracle access. If f is $(2m+1)$-times differentiable and for all $\boldsymbol{x}\in[-D,D]^d$ we have that*

$$|\partial_{\boldsymbol{r}}^{2m+1}f(\boldsymbol{x})| \leq B \quad \text{for } \boldsymbol{r}=\boldsymbol{x}/\|\boldsymbol{x}\|,$$

then using Algorithm 4.2 with setting

$$R = \Theta\left(\max\left(\sqrt{d}\sqrt[2m]{\frac{B\sqrt{d}}{\varepsilon}}, \frac{m}{D}\right)\right)$$

we can compute an approximate gradient $\boldsymbol{g}$ such that $\|\boldsymbol{g}-\nabla f(\boldsymbol{0})\|_\infty \leq \varepsilon$ with probability at least $(1-\rho)$, using $\mathcal{O}\left(\left(\frac{R}{\varepsilon}\log(2m)+m\right)\log\left(\frac{d}{\rho}\right)\right)$ phase queries.

Suppose that $D=\Theta(1)$, and f is a multi-variate polynomial of degree k. Then for $m=\lceil k/2\rceil$ we get that $B=0$, as can be seen by using (4.5), therefore the above result gives an $\widetilde{\mathcal{O}}\left(\frac{k}{\varepsilon}\log\left(\frac{d}{\rho}\right)\right)$ query algorithm. If $2\leq k=\mathcal{O}(\log(d))$, then this result gives an exponential speedup in terms of the dimension d compared to

Jordan's algorithm. For comparison we note that other recent results concerning quantum gradient descent also work under the assumption that the objective function is a polynomial [RSPL16, KP17a].

However, as we argue in Appendix 4.A, for non-polynomial functions we can have $B \approx d^m$ even under strong regularity conditions. This then results in an $\widetilde{\mathcal{O}}\left(\frac{d}{\varepsilon}\right)$ query algorithm, achieving the desired scaling in ε but failing to capture the sought $\sqrt{d}$ scaling. In order to tackle the non-polynomial case we need to introduce some smoothness conditions.

4.5.4 Approximation-error bounds for smooth functions

In this section we show how to improve the result of Theorem 4.5.7, assuming some smoothness condition. The calculus gets a bit involved, because we need to handle higher-dimensional analysis. In order to focus on the main results, we keep this section concise and move the proof of some technical results to Appendix 4.A. We show that under reasonable smoothness assumptions, the complexity of our quantum algorithm is $\widetilde{O}(\sqrt{d}/\varepsilon)$ and in the next section show that for a specific class of smooth functions this is in fact optimal up to polylog factors.

A key ingredient of our proof is the following lemma, bounding the "second moment" of higher-order bounded tensors, proven in Appendix 4.A.

4.5.8. Lemma. *Let $d, k \in \mathbb{N}_+$, and suppose $H \in \left(\mathbb{R}^d\right)^{\otimes k}$ is an order-k tensor of dimension d, having all elements bounded by 1 in absolute value, i.e., $\|H\|_\infty \leq 1$. Suppose $(x_1, \ldots, x_d)$ is a vector of i.i.d. symmetric random variables bounded in $[-1/2, 1/2]$. Then for every $t > 0$*

$$\Pr\left(\left|\sum_{\alpha\in[d]^k} H_\alpha x^\alpha\right| \geq \sqrt{2}\left(t\sqrt{\frac{dk}{2}}\right)^k\right) \leq \frac{1}{t^{2k}}.$$

Using this lemma, we can prove the following result about analytic functions:

4.5.9. Theorem. *If $r \in \mathbb{R}_+$, $f : \mathbb{R}^d \to \mathbb{R}$ is analytic and for all $k \in \mathbb{N}, \alpha \in [d]^k$ we have*

$$|\partial_\alpha f(\mathbf{0})| \leq c^k k^{\frac{k}{2}},$$

then

$$|\nabla f(\mathbf{0})\boldsymbol{y} - f_{(2m)}(\boldsymbol{y})| \leq \sum_{k=2m+1}^{\infty} \left(8rcm\sqrt{d}\right)^k,$$

for all but a $1/1000$ fraction of points $\boldsymbol{y} \in r \cdot G_n^d$.

We can use this result to analyze the complexity of Algorithm 4.1 when applied to functions evaluated using a central-difference formula. In particular it makes it easy to prove the following theorem, which is one of our main results.

4.5.10. THEOREM. *Let $\boldsymbol{x} \in \mathbb{R}^d$, $\varepsilon \leq c \in \mathbb{R}_+$ be fixed constants and suppose $f\colon \mathbb{R}^d \to \mathbb{R}$ is analytic*[13] *and satisfies the following: for every $k \in \mathbb{N}$ and $\alpha \in [d]^k$*

$$|\partial_\alpha f(\boldsymbol{x})| \leq c^k k^{\frac{k}{2}}.$$

Using Algorithm 4.2 with setting $m = \log(c\sqrt{d}/\varepsilon)$ and

$$R = \Theta\Big(cm\sqrt{d}\Big)$$

we can compute an ε-approximate gradient $\tilde{\nabla} f(\boldsymbol{x}) \in \mathbb{R}^d$ such that

$$\left\|\nabla f(\boldsymbol{x}) - \tilde{\nabla} f(\boldsymbol{x})\right\|_\infty \leq \varepsilon,$$

with probability at least $1-\delta$, using $\widetilde{\mathcal{O}}\Big(\frac{c\sqrt{d}}{\varepsilon}\log\big(\frac{d}{\delta}\big)\Big)$ queries to a probability or (fractional) phase oracle of f.

Proof:
Let $g(\boldsymbol{y}) := f(\boldsymbol{x}+\boldsymbol{y})$. By Theorem 4.5.9 we know that for a uniformly random $\boldsymbol{y} \in r \cdot G_n^d$ we have $|\nabla g(\boldsymbol{0})\boldsymbol{y} - g_{(2m)}(\boldsymbol{y})| \leq \sum_{k=2m+1}^{\infty}\Big(8rcm\sqrt{d}\Big)^k$ with probability at least 999/1000. Now we choose r such that this becomes smaller than $\frac{\varepsilon r}{8 \cdot 42\pi}$. Now let us define $R := r^{-1} := 9cm\sqrt{d}\Big(81 \cdot 8 \cdot 42\pi cm\sqrt{d}/\varepsilon\Big)^{1/(2m)}$, then we get $8rcm\sqrt{d} = \frac{8}{9}\Big(81 \cdot 8 \cdot 42\pi cm\sqrt{d}/\varepsilon\Big)^{-1/(2m)}$ and so

$$\begin{aligned}
\sum_{k=2m+1}^{\infty}\Big(8rcm\sqrt{d}\Big)^k &= \Big(8rcm\sqrt{d}\Big)^{2m+1}\sum_{k=0}^{\infty}\Big(8rcm\sqrt{d}\Big)^k \\
&\leq \frac{\varepsilon}{81 \cdot 8 \cdot 42\pi cm\sqrt{d}}\Big(81 \cdot 8 \cdot 42\pi cm\sqrt{d}/\varepsilon\Big)^{\frac{-1}{2m}}\sum_{k=0}^{\infty}\Big(\frac{8}{9}\Big)^k \\
&\qquad\qquad \text{(by our choice of } r\text{)} \\
&= \frac{\varepsilon}{9cm\sqrt{d} \cdot 8 \cdot 42\pi}\Big(81 \cdot 8 \cdot 42\pi cm\sqrt{d}/\varepsilon\Big)^{\frac{-1}{2m}} \\
&\qquad\qquad \Big(\text{since } \textstyle\sum_{k=0}^{\infty}\big(\frac{8}{9}\big)^k = 9\Big) \\
&= \frac{\varepsilon r}{8 \cdot 42\pi}.
\end{aligned}$$

[13] For convenience we assume in the statement that f can be evaluated at any point of $\mathbb{R}^d$, but in fact we only evaluate it inside a finite ball around $\boldsymbol{x}$. It is straightforward to translate the result to case where the function is only accessible on an open ball around $\boldsymbol{x}$. However, a finite domain imposes restrictions to the evaluation points of the function. If $\boldsymbol{x}$ lies too close to the boundary, this might impose additional scaling requirements and thus potentially increases the complexity of the derived algorithm. Fortunately in our applications it is natural to assume that f can be evaluated at distant points too, so we don't need to worry about this detail.

By Theorem 4.5.5 we can compute an ε-approximation of $\nabla g_{(2m)}(\boldsymbol{x})$ with $\mathcal{O}\big(\log\frac{d}{\delta}\big)$ queries to $\mathrm{O}^S_{g_{(2m)}}$, where $S=\mathcal{O}\big(\frac{1}{\varepsilon r}\big)$. Observe that

$$\mathrm{O}^S_{g_{(2m)}}|\boldsymbol{y}\rangle = e^{iSg_{(2m)}(\boldsymbol{y})}|\boldsymbol{y}\rangle = e^{iS\sum_{\ell=-m}^{m} a_\ell^{(2m)} g(\ell\boldsymbol{y})}|\boldsymbol{y}\rangle.$$

Using the relation between f and g, it is easy to see that the number of (fractional) phase queries to O_f we need in order to implement a modified oracle call $\mathrm{O}^S_{g_{(2m)}}$ is

$$\sum_{\ell=-m}^{m}\left\lceil\left|a_\ell^{(2m)}\right|S\right\rceil \le 2m + S\sum_{\ell=-m}^{m} a_\ell^{(2m)} \overset{(4.12)}{\le} 2m + S(2\log(m)+2). \tag{4.13}$$

Thus $\mathrm{O}^S_{g_{(2m)}}$ can be implemented using $\mathcal{O}\Big(\frac{\log(m)}{\varepsilon r}+m\Big)$ (fractional) queries to O_f. Picking $m=\log(c\sqrt{d}/\varepsilon)$ the query complexity becomes[14]

$$\mathcal{O}\left(\frac{c\sqrt{d}}{\varepsilon}m\log(m)\right) = \mathcal{O}\left(\frac{c\sqrt{d}}{\varepsilon}\log\left(\frac{c\sqrt{d}}{\varepsilon}\right)\log\log\left(\frac{c\sqrt{d}}{\varepsilon}\right)\right). \tag{4.14}$$

□

The above achieves, up to logarithmic factors, the desired $1/\varepsilon$ scaling in the precision parameter and also the $\sqrt{d}$ scaling with the dimension. This improves the results of [Jor05] both quantitatively and qualitatively.

We also show that the query complexity for this problem is almost optimal, by proving a lower bound in Section 4.6 which matches the above upper bound up to log factors.

4.5.5 Most quantum optimization problems are "smooth"

We now show that the condition on the derivatives in Theorem 4.5.10 is fairly reasonable, i.e., a wide range of probability oracles that arise from quantum optimization problems satisfy this condition. In particular, consider the function $p:\mathbb{R}^d\to\mathbb{R}$ that we looked at (see Eq. (4.7)) during the discussion of a generic model of quantum optimization algorithms:

$$p(\boldsymbol{x}) = \langle\mathbf{0}|U(\boldsymbol{x})^\dagger(|1\rangle\langle 1|\otimes I)U(\boldsymbol{x})|\mathbf{0}\rangle.$$

We will now show that for every $k\in\mathbb{N}$ and index-sequence $\alpha\in[d]^k$, we have[15] $|\partial_\alpha p(\boldsymbol{x})|\le 2^k$ when $U(\boldsymbol{x})$ is a product of d (controlled) rotations

$$\mathrm{Rot}(x_j) = \begin{pmatrix}\cos(x_j) & \sin(x_j)\\ -\sin(x_j) & \cos(x_j)\end{pmatrix} = \exp\left[ix_j\begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}\right] = e^{ix_j\sigma_y}$$

[14]If we strengthen the $c^k k^{\frac{k}{2}}$ upper bound assumption on the derivatives to c^k, then we could improve the bound of Theorem 4.5.9 by a factor of $k^{-k/2}$. Therefore in the definition of R^{-1} we could replace m by $\sqrt{m}$ which would quadratically improve the log factor in (4.14).

[15]This essentially means that the function $p(\boldsymbol{x})$ in the probability oracle is in the Gevrey [Gev18] class G^0.

and other fixed unitaries. In order to prove this, we use Lemma 4.5.11 to show that $\|\partial_\alpha U(\boldsymbol{x})\| \leq 1$ for every index-sequence α, which by Lemma 4.5.12 implies

$$\left\|\partial_\alpha\big(U(\boldsymbol{x})^\dagger(|1\rangle\langle 1| \otimes I)U(\boldsymbol{x})\big)\right\| \leq 2^k,$$

hence proving the claim. In fact, we prove slightly stronger statements, so that these lemmas can be used later in greater generality.

4.5.11. LEMMA. *Suppose $\gamma \geq 0$ and*

$$U(\boldsymbol{x}) = U_0 \prod_{j=1}^{d} \big(\Pi_j \otimes e^{ix_j H_j} + (I - \Pi_j) \otimes I\big)U_j,$$

where $\|U_0\| \leq 1$ and for every $j \in [d]$ we a have that $\|U_j\| \leq 1$, Π_j is an orthogonal projector and H_j is Hermitian with $\|H_j\| \leq \gamma$. Then for every $k \in \mathbb{N}$ and $\alpha \in [d]^k$, we have that $\|\partial_\alpha U(\boldsymbol{x})\| \leq \gamma^k$.

Proof:
We have that

$$\partial_\alpha U(\boldsymbol{x}) = U_0 \prod_{j=1}^{d} \Big(\Pi_j \otimes (iH_j)^{|\{\ell\in[k]:\alpha_\ell=j\}|} e^{ix_j H_j} + 0^{|\{\ell\in[k]:\alpha_\ell=j\}|}(I - \Pi_j) \otimes I\Big)U_j,$$

therefore

$$\begin{aligned}
\|\partial_\alpha U(\boldsymbol{x})\| &= \left\|U_0 \prod_{j=1}^{d} \Big(\Pi_j \otimes (iH_j)^{|\{\ell\in[k]:\alpha_\ell=j\}|} e^{ix_j H_j} + 0^{|\{\ell\in[k]:\alpha_\ell=j\}|}(I - \Pi_j) \otimes I\Big)U_j\right\| \\
&\leq \prod_{j=1}^{d} \left\|\Big(\Pi_j \otimes (iH_j)^{|\{\ell\in[k]:\alpha_\ell=j\}|} e^{ix_j H_j} + 0^{|\{\ell\in[k]:\alpha_\ell=j\}|}(I - \Pi_j) \otimes I\Big)\right\| \\
&\leq \prod_{j=1}^{d} \gamma^{|\{\ell\in[k]:\alpha_\ell=j\}|} = \gamma^k.
\end{aligned}$$

□

4.5.12. LEMMA. *Suppose that $A(\boldsymbol{x}), B(\boldsymbol{x})$ are linear operators parametrized by $\boldsymbol{x} \in \mathbb{R}^d$. If for all $k \in \mathbb{N}_0$ and $\alpha \in [d]^k$ we have that $\|\partial_\alpha A\| \leq \gamma^k$ and $\|\partial_\alpha B\| \leq \gamma^k$, then for all $k \in \mathbb{N}_0$ and $\alpha \in [d]^k$ we get that $\|\partial_\alpha(AB)\| \leq (2\gamma)^k$.*

Proof:
For an index-sequence $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_k) \in [d]^k$ and a set $S = \{i_1 < i_2 < \ldots < i_\ell\} \subseteq [k]$ consisting of positions of the index-sequence, we define $\alpha_S :=$

$(\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_\ell}) \in [d]^{|S|}$ to be the index-sequence where we only keep indexes corresponding to positions in S; also let $\overline{S} := [k] \setminus S$. It can be seen that

$$\partial_\alpha(AB) = \sum_{S \subseteq [k]} \partial_{\alpha_S} A \partial_{\alpha_{\overline{S}}} B,$$

therefore

$$\|\partial_\alpha(AB)\| \leq \sum_{S \subseteq [k]} \left\|\partial_{\alpha_S} A \partial_{\alpha_{\overline{S}}} B\right\| \leq \sum_{S \subseteq [k]} \gamma^{|S|} \gamma^{k-|S|} = (2\gamma)^k.$$

□

4.5.13. Remark. Finally note that the unitary in Lemma 4.5.11 is an analytic function of its parameters, therefore the probability that we get by taking products of such unitaries and some fixed matrices/vectors is also an analytic function.

4.6 Lower bounds on gradient computation

In this section we prove that the number of phase oracle queries required to compute the gradient for some of the smooth functions satisfying the requirement of Theorem 4.5.10 is $\Omega\left(\sqrt{d}/\varepsilon\right)$, showing that Theorem 4.5.10 is optimal up to log factors. As Proposition 4.4.6 shows, a probability oracle can be simulated using a logarithmic number of phase oracle queries, therefore this lower bound also translates to probability oracles.[16]

We first prove a useful theorem providing a lower bound on the number of queries needed in order to distinguish a particular phase oracle from a family of other phase oracles. This result is of independent interest, and has already found an application for proving lower bounds on quantum SDP-solving [vAG19].

Then we prove our lower bound on gradient computation by constructing a family of functions which can be distinguished from the trivial phase oracle I by ε-precise gradient computation, but for which our hybrid-method based lower bound shows that distinguishing the phase oracles from I requires $\Omega\left(\sqrt{d}/\varepsilon\right)$ queries.

4.6.1 Hybrid method for arbitrary phase orcales

Now we turn to proving our general lower bound result based on the hybrid method, which was originally introduced for proving a lower bound for quantum

[16]A probability oracle can only represent functions which map to $[0, 1]$, whereas the range of the function f we use in the lower bound proof is a subinterval of $[-1, 1]$. However, by using the transformed function $g := (2 + f)/4$ we get a function which has a range contained in $[1/2, 3/4]$ so it can in principle be represented by a probability oracle. Moreover for a function whose range is contained in $[1/2, 3/4]$ we can efficiently convert between phase and probability oracles as shown by Corollary 4.4.4 and Proposition 4.4.6.

search[17] by Bennett et al. [BBBV97], and can be viewed as a special case of the adversary method [Amb00, HŠ05, LMR+11, Bel15]. Our proof closely follows the presentation of the hybrid method in Montanaro's lecture notes [Mon11, Section 1].

4.1.2. Theorem. (Hybrid method for arbitrary phase oracles) *Let G be a (finite) set of labels and let $\mathcal{H} := \mathrm{Span}(|x\rangle\colon x \in G)$ be a Hilbert space. For a function $\tilde{f}: G \to \mathbb{R}$ let $\mathrm{O}_{\tilde{f}}$ be the phase oracle acting on $\mathcal{H}$ such that*

$$\mathrm{O}_{\tilde{f}} : |x\rangle \to e^{i\tilde{f}(x)}|x\rangle \quad \text{for every } x \in G.$$

Suppose that $\mathcal{F}$ is a finite set of functions $G \to \mathbb{R}$, and the function $f_*\colon G \to \mathbb{R}$ is not in F. If a quantum algorithm makes T queries to a (controlled) phase oracle $\mathrm{O}_{\tilde{f}}$ (or its inverse) and for all $f \in \mathcal{F}$ can distinguish with probability at least $2/3$ the case $\tilde{f} = f$ from the case $\tilde{f} = f_*$, then

$$T \geq \frac{\sqrt{|\mathcal{F}|}}{3} \Bigg/ \sqrt{\max_{x\in G} \sum_{f\in\mathcal{F}} \min\left(|f(x) - f_*(x)|^2, 4\right)}.$$

Proof:
Suppose that $\mathcal{F} = \{f_j\colon j \in \{1, \ldots, d\}\}$, and let $f_0 := f_*$. Let $\mathcal{A}_j$ denote the algorithm which uses phase oracle O_{f_j} and let $|\psi_j\rangle := \mathcal{A}_j|\vec{0}\rangle$ denote the state of the algorithm before the final measurement. Since we can distinguish the states $|\psi_0\rangle$ and $|\psi_j\rangle$ with probability at least 2/3, by the Holevo-Helstrom theorem [Wat18, Chapter 3.1.1], it follows that

$$\frac{2}{3} \leq \frac{1}{2} + \frac{1}{4}\| |\psi_0\rangle\langle\psi_0| - |\psi_j\rangle\langle\psi_j| \|_1.$$

Since $\| |\psi_0\rangle\langle\psi_0| - |\psi_j\rangle\langle\psi_j| \|_1 \leq 2\| |\psi_0\rangle - |\psi_j\rangle \|$, we have that $1/3 \leq \| |\psi_0\rangle - |\psi_j\rangle \|$.

In general the quantum algorithm might use some workspace $\mathcal{W} = \underset{w\in W}{\mathrm{Span}}\{|w\rangle : w \in W\}$ along with the Hilbert space $\mathcal{H}$. To emphasize this we introduce the notation $\mathrm{O}'_f := \mathrm{O}_f \otimes I_{\mathcal{W}}$, and $G' := G \times W$, so that the elements of $\mathcal{H} \otimes \mathcal{W}$ can be labeled by the elements of G'. It is well known that in a quantum algorithm all measurements can be postponed to the end of the quantum circuit, so we can assume without loss of generality that between the queries the algorithm acts in a unitary fashion. Thus we can write $\mathcal{A} = U_T\mathrm{O}'_f U_{T-1}\mathrm{O}'_f U_{T-1} \cdots U_1\mathrm{O}'_f U_0$.

[17] One might wonder why we do not make a reduction to a search problem, e.g., by considering a function which has non-zero gradient only at some marked coordinates. We expect that this approach is not going to lead to a good lower bound, because the phase oracle is too strong, and by calculating the gradient one can actually solve exact counting, which problem has a linear lower bound for usual search oracles.

Let us define for $t \in \{0, 1, \ldots, T\}$

$$|\psi_j^{(t)}\rangle := \left(\prod_{\tau=1}^{t} U_\tau \mathrm{O}'_{f_j}\right) U_0 |\vec{0}\rangle,$$

the state of algorithm $\mathcal{A}_j$ after making t queries. We now prove by induction that for all $t \in \{0, 1, \ldots, T\}$

$$\left\| |\psi_j^{(t)}\rangle - |\psi_0^{(t)}\rangle \right\| \leq \sum_{\tau=0}^{t-1} \left\| (\mathrm{O}'_{f_j} - \mathrm{O}'_{f_0}) |\psi_0^{(\tau)}\rangle \right\|. \tag{4.15}$$

For $t = 0$ the left-hand side is 0, so the base case holds. Let us assume that (4.15) holds for $t - 1$, we prove the inductive step as follows:

$$\begin{aligned}
\left\| |\psi_j^{(t)}\rangle - |\psi_0^{(t)}\rangle \right\| &= \left\| U_t \mathrm{O}'_{f_j} |\psi_j^{(t-1)}\rangle - U_t \mathrm{O}'_{f_0} |\psi_0^{(t-1)}\rangle \right\| \\
&= \left\| \mathrm{O}'_{f_j} |\psi_j^{(t-1)}\rangle - \mathrm{O}'_{f_0} |\psi_0^{(t-1)}\rangle \right\| \\
&\qquad\qquad \text{(since norms are unitarily invariant)} \\
&= \left\| \mathrm{O}'_{f_j} \left(|\psi_j^{(t-1)}\rangle - |\psi_0^{(t-1)}\rangle + |\psi_0^{(t-1)}\rangle \right) - \mathrm{O}'_{f_0} |\psi_0^{(t-1)}\rangle \right\| \\
&\leq \left\| \mathrm{O}'_{f_j} \left(|\psi_j^{(t-1)}\rangle - |\psi_0^{(t-1)}\rangle \right) \right\| + \left\| (\mathrm{O}'_{f_j} - \mathrm{O}'_{f_0}) |\psi_0^{(t-1)}\rangle \right\| \\
&\qquad\qquad \text{(triangle inequality)} \\
&= \left\| |\psi_j^{(t-1)}\rangle - |\psi_0^{(t-1)}\rangle \right\| + \left\| (\mathrm{O}'_{f_j} - \mathrm{O}'_{f_0}) |\psi_0^{(t-1)}\rangle \right\| \\
&\leq \sum_{\tau=0}^{t-1} \left\| (\mathrm{O}'_{f_j} - \mathrm{O}'_{f_0}) |\psi_0^{(\tau)}\rangle \right\|. \qquad \text{(by the induction hypothesis)}
\end{aligned}$$

Since $|\psi_j\rangle = |\psi_j^{(T)}\rangle$, we additionally have that

$$1/9 \leq \| |\psi_0\rangle - |\psi_j\rangle \|^2 \leq \left(\sum_{\tau=0}^{T-1} \left\| (\mathrm{O}'_{f_j} - \mathrm{O}'_{f_0}) |\psi_0^{(\tau)}\rangle \right\| \right)^2 \leq T \sum_{\tau=0}^{T-1} \left\| (\mathrm{O}'_{f_j} - \mathrm{O}'_{f_0}) |\psi_0^{(\tau)}\rangle \right\|^2,$$

where the last inequality uses the Cauchy-Schwarz inequality. Averaging the above inequality over the different oracles, we have

$$1/9 \leq \frac{T}{d} \sum_{\tau=0}^{T-1} \sum_{j \in [d]} \left\| (\mathrm{O}'_{f_j} - \mathrm{O}'_{f_0}) |\psi_0^{(\tau)}\rangle \right\|^2 \leq \frac{T^2}{d} \max_{\tau} \sum_{j \in [d]} \left\| (\mathrm{O}'_{f_j} - \mathrm{O}'_{f_0}) |\psi_0^{(\tau)}\rangle \right\|^2. \tag{4.16}$$

We now upper bound the right-hand side of Eq. (4.16) for an arbitrary pure state

$|\psi\rangle$ to conclude the proof of the theorem.

$$\begin{aligned}
\sum_{j\in[d]}\left\|(\mathrm{O}'_{f_j}-\mathrm{O}'_{f_0})|\psi\rangle\right\|^2 &= \sum_{j\in[d]}\left\|\left(\sum_{x\in G'}|x\rangle\langle x|\right)(\mathrm{O}'_{f_j}-\mathrm{O}'_{f_0})\left(\sum_{x'\in G'}|x'\rangle\langle x'|\right)|\psi\rangle\right\|^2\\
&= \sum_{j\in[d]}\left\|\sum_{x\in G'}|x\rangle\langle x|(\mathrm{O}'_{f_j}-\mathrm{O}'_{f_0})|x\rangle\langle x||\psi\rangle\right\|^2\\
&\qquad\qquad\text{(since } \langle x|\mathrm{O}'_{f_j}|x'\rangle = 0 \text{ for } x\neq x'\text{)}\\
&= \sum_{x\in G'}|\langle x|\psi\rangle|^2\sum_{j\in[d]}\left|\langle x|(\mathrm{O}'_{f_j}-\mathrm{O}'_{f_0})|x\rangle\right|^2\\
&\le \max_{x\in G'}\sum_{j\in[d]}\left|\langle x|(\mathrm{O}'_{f_j}-\mathrm{O}'_{f_0})|x\rangle\right|^2\\
&= \max_{x\in G}\sum_{j\in[d]}\left|e^{if_j(x)}-e^{if_0(x)}\right|^2 \quad \text{(note the } G'\to G \text{ change)}\\
&\le \max_{x\in G}\sum_{j\in[d]}\min\big(|f_j(x)-f_0(x)|^2,4\big)\\
&\qquad\qquad\text{(since } |e^{iz}-e^{iy}|\le\min(|z-y|,2)\text{)}
\end{aligned}$$

Combining this upper bound with Eq. (4.16), we have

$$\frac{1}{9}\le\frac{T^2}{d}\max_{x\in G}\sum_{j\in[d]}\min\big(|f_j(x)-f_0(x)|^2,4\big),$$

which in turn gives us the desired lower bound

$$T\ge\frac{\sqrt{d}}{3}\Bigg/\sqrt{\max_{x\in G}\sum_{j\in[d]}\min\big(|f_j(x)-f_0(x)|^2,4\big)}\,.$$

Finally note that a controlled phase oracle is also a phase oracle, and the inverse oracles have the same operator distance as the non-inverted versions, therefore the above lower bound holds even if we allow controlled phase oracles or inverse oracle calls. □

4.6.2 A family of functions for proving the gradient computation lower bound

Now we prove our lower bound on gradient computation by constructing a family of functions $\mathcal{F}$ for which the corresponding phase oracles $\{\mathrm{O}_f : f\in\mathcal{F}\}$ require $\Omega(\sqrt{d}/\varepsilon)$ queries to distinguish them from the constant 0 function (as shown by

Theorem 4.1.2), but the functions in $\mathcal{F}$ can be uniquely identified by calculating their gradient at $\mathbf{0}$ with accuracy ε. In particular, this implies that calculating an approximation of the gradient vector for these functions must be at least as hard as distinguishing the phase oracles corresponding to functions in $\mathcal{F}$.

4.6.1. Lemma. *Let $d \in \mathbb{N}$, $\varepsilon, c \in \mathbb{R}_+$ and let us define the following $\mathbb{R}^d \to \mathbb{R}$ functions: $f_*(\boldsymbol{x}) := 0$ and $f_j(\boldsymbol{x}) := 2\varepsilon x_j e^{-c^2\|\boldsymbol{x}\|^2/2}$ for all $j \in [d]$. Consider the family of functions $\mathcal{F} := \bigcup_{j\in[d]}\{f_j(\boldsymbol{x})\}$, then for all $\boldsymbol{x} \in \mathbb{R}^d$ we have that*

$$\sum_{j\in[d]} |f_j(\boldsymbol{x}) - f_*(\boldsymbol{x})|^2 \leq \frac{4\varepsilon^2}{ec^2}.$$

Proof:

$$\begin{aligned}
\sum_{j\in[d]} |f_j(\boldsymbol{x}) - f_*(\boldsymbol{x})|^2 &= \sum_{j\in[d]} \left|2\varepsilon x_j e^{-c^2\|\boldsymbol{x}\|^2/2}\right|^2 \\
&= 4\varepsilon^2\|\boldsymbol{x}\|^2 e^{-c^2\|\boldsymbol{x}\|^2} \\
&\leq \frac{4\varepsilon^2}{ec^2}. \qquad \text{(using } ze^{-z} \leq 1/e \text{ with } z := c^2\|\boldsymbol{x}\|^2\text{)}
\end{aligned}$$

□

Now we prove bounds on the partial derivatives of the above functions to determine their smoothness.

4.6.2. Lemma. *Let d, k be positive integers, $c \in \mathbb{R}_+$ and $\boldsymbol{x} \in \mathbb{R}^d$. Then, the function $f_j(\boldsymbol{x}) := cx_j e^{-\frac{c^2\|\boldsymbol{x}\|^2}{2}}$ satisfies the following: for every index-sequence $\alpha \in [d]^k$, the derivative of f is bounded by $|\partial_\alpha f(\mathbf{0})| \leq c^k k^{\frac{k}{2}}$. Moreover $\nabla f_j(\mathbf{0}) = c\boldsymbol{e}_j$.*

Proof:
Observe that

$$f(\boldsymbol{x}) = cx_j e^{-\frac{c^2x_j^2}{2}} \prod_{i\neq j}^{d} e^{-\frac{c^2x_i^2}{2}}, \tag{4.17}$$

and as one can see from the Taylor series $e^{-\frac{(cx)^2}{2}} = \sum_{\ell=0}^{\infty}\left(-\frac{1}{2}\right)^\ell \frac{(cx)^{2\ell}}{\ell!}$ we have for $k \geq 0$

$$\partial_i^k e^{-\frac{c^2x_i^2}{2}}\bigg|_{x_i=0} = \begin{cases} (-\frac{1}{2})^\ell c^{2\ell}\frac{(2\ell)!}{\ell!} & \text{for } k = 2\ell \\ 0 & \text{for } k = 2\ell+1 \end{cases}, \tag{4.18}$$

$$\partial_j^k cx_j e^{-\frac{c^2x_j^2}{2}}\bigg|_{x_j=0} = \begin{cases} 0 & \text{for } k = 2\ell \\ (-\frac{1}{2})^\ell c^{2\ell+1}\frac{(2\ell+1)!}{\ell!} & \text{for } k = 2\ell+1 \end{cases}. \tag{4.19}$$

Also observe that, for $\ell \geq 0$

$$\frac{(2\ell)!}{\ell!} \leq (2\ell)^\ell \qquad \text{and} \qquad \left(\frac{1}{2}\right)^\ell \frac{(2\ell+1)!}{\ell!} \leq (2\ell+1)^{\ell+1/2}. \tag{4.20}$$

The statements of the lemma follow by combining the results (4.17)-(4.20). □

Now we use the above lemmas combined with the hybrid method (Theorem 4.1.2) to prove our general lower bound result.

4.6.3. Theorem. *Let $\varepsilon, c, d > 0$ such that $2\varepsilon \leq c$ and for an arbitrary finite set $G \subseteq \mathbb{R}^d$ let*

$$\mathcal{H} = \operatorname*{Span}_{\boldsymbol{x} \in G}\{|\boldsymbol{x}\rangle : \boldsymbol{x} \in G\}.$$

Suppose $\mathcal{A}$ is a T-query quantum algorithm (assuming query access to phase oracle $\mathrm{O}_f \colon |\boldsymbol{x}\rangle \mapsto e^{if(\boldsymbol{x})}|\boldsymbol{x}\rangle$, acting on $\mathcal{H}$) for analytic functions $f : \mathbb{R}^d \to \mathbb{R}$ satisfying

$$|\partial_\alpha f(\mathbf{0})| \leq c^k k^{\frac{k}{2}} \qquad \textit{for all } k \in \mathbb{N}, \alpha \in [d]^k,$$

such that $\mathcal{A}$ computes an ε-approximation $\boldsymbol{g}$ of the gradient at $\mathbf{0}$ such that

$$\|\boldsymbol{g} - \nabla f(\mathbf{0})\|_\infty < \varepsilon,$$

succeeding with probability at least $2/3$. Then $T > \frac{c\sqrt{d}}{4\varepsilon}$.

Proof:
Inspired by Lemma 4.6.1, we first define a set of "hard-to-distinguish" functions, which we will use to prove our lower bound Let $f_* := f_0 := 0$ and $f_j(\boldsymbol{x}) := 2\varepsilon x_j e^{-c^2\|\boldsymbol{x}\|^2/2}$ for all $j \in [d]$. Consider the family of functions $\mathcal{F} := \bigcup_{j\in[d]}\{f_j(\boldsymbol{x})\}$. By Lemma 4.6.2, every $f \in \mathcal{F}$ satisfies $|\partial_\alpha f(\mathbf{0})| \leq c^k k^{\frac{k}{2}}$ for all $k \in \mathbb{N}$ and $\alpha \in [d]^k$.

Suppose we are given a phase oracle O_f acting on $\mathcal{H}$, such that $\mathrm{O}_f = \mathrm{O}_{f_j} : |\boldsymbol{x}\rangle \mapsto e^{if_j(\boldsymbol{x})}|\boldsymbol{x}\rangle$ for some unknown $j \in \{0, \ldots, d\}$. Since $\nabla f_0(\mathbf{0}) = \mathbf{0}$ and $\nabla f_j(\mathbf{0}) = 2\varepsilon \boldsymbol{e}_j$, using the T-query algorithm $\mathcal{A}$ in the theorem statement, one can determine the $j \in \{0, \ldots, d\}$ for which $f_j = f$ with success probability at least $2/3$. In particular we can distinguish the case $f = f_*$ from $f \in \mathcal{F}$, and thus by Theorem 4.1.2 and Lemma 4.6.1 we get that

$$T \geq \sqrt{d}\frac{c}{\varepsilon}\sqrt{\frac{e}{36}} > \frac{c\sqrt{d}}{4\varepsilon}.$$

□

4.6.3 Lower bound for more regular functions

Note that the functions for which we apply our results in this chapter tend to satisfy a stronger condition than our lower bound example in Theorem 4.6.3. They usually satisfy[18] $|\partial_\alpha f(\boldsymbol{x}_0)| \leq c^k$. We conjectured that the same lower bound holds for this subclass of functions as well.

Very recently, Cornelissen [Cor18] managed to prove this conjecture, also building on Theorem 4.1.2. Moreover, he showed an $\Omega\left(d^{\frac{1}{2}+\frac{1}{p}}/\varepsilon\right)$ lower bound for ε-precise gradient computation in p-norm *for every* $p \in [1,\infty]$. Note that these results shows that our algorithm is essentially optimal for a large class of gradient computation problems, because our algorithm provides an almost matching upper bound via reducing ℓ^p-approximation to ℓ^∞-approximation.

Now we argue heuristically why Jordan's algorithm should not be able to calculate the gradient with significantly fewer queries for the above mentioned class of functions. Algorithm 4.1 performs a Fourier transform, after applying a phase oracle that puts a phase $\sim \tilde{f}(\boldsymbol{x})/\varepsilon$ to the state $\boldsymbol{x} \in G_n^d$. We can prove that for $n \geq \log(3c/\varepsilon)$ the Fourier transform will provide the coordinates of $\nabla f(\boldsymbol{0})$ up to error $\mathcal{O}(\varepsilon)$ with high probability, given that for most of the points $\boldsymbol{x} \in G_n^d$ we have $|\tilde{f}(\boldsymbol{x})/\varepsilon - \nabla f(\boldsymbol{0}) \cdot \boldsymbol{x}/\varepsilon| \ll 1$. Using a fractional phase oracle we can prepare phases of the form $\tilde{f}(\boldsymbol{x})/\varepsilon = \sum_{\boldsymbol{y}\in S} \lambda_{\boldsymbol{y}} f(\boldsymbol{x})$ for some $S \subseteq G_n^d$, and $(\lambda_{\boldsymbol{y}}) \in [-1,1]^{|S|}$, where S (and possibly $(\lambda_{\boldsymbol{y}})$) might depend on $\boldsymbol{x}$. The query complexity is thus driven by $|S|$.

Let us assume that $\nabla f(\boldsymbol{0}) = c \cdot \boldsymbol{1}$, and observe that since G_n^d is symmetric around $\boldsymbol{0}$ we get that the typical value of $c \cdot \boldsymbol{1} \cdot \boldsymbol{x} = c\sum_{i=1}^d x_i$ is $\Theta(c\sqrt{d})$, as can be seen for example by the central limit theorem. If we also assume that $|f| \leq 1$, then by the triangle inequality we see that $|S| = \Omega(\frac{c\sqrt{d}}{\varepsilon})$.

To conclude we still need to show that there exists an analytic function $f : \mathbb{R}^d \to \mathbb{R}$, which has $|f| \leq 1$ and $\nabla f(\boldsymbol{0}) = c \cdot \boldsymbol{1}$ while it also satisfies for all $k \in \mathbb{N}$ and $\alpha \in [d]^k$ that $|\partial_\alpha f| \leq c^k$. At first sight this set of requirements seem slightly contradicting, but we found a very simple example of such a function:

$$f(\boldsymbol{x}) := \sin(cx_1 + cx_2 + \ldots + cx_d) \text{ for which } \partial_\alpha f(\boldsymbol{x}) = c^{|\alpha|} \sin^{(|\alpha|)}(cx_1 + cx_2 + \ldots + cx_d).$$

The above argument also shows that placing the grid G_n^d symmetrically around $\boldsymbol{0}$ is of crucial importance. For example if the midpoint would be shifted by say $\delta\boldsymbol{1}$, then the typical magnitude of $\boldsymbol{1} \cdot \boldsymbol{x} = \boldsymbol{1} \cdot \left(\delta\boldsymbol{1} + \boldsymbol{x}^{(\text{symm.})}\right)$ would be $\Theta(\delta d + \sqrt{d})$, which would give rise to an elevated lower bound when $\delta \gg 1/\sqrt{d}$.

4.7 Applications

In this section, we first consider variational quantum eigensolvers and QAOA algorithms, which can be treated essentially identically using our techniques. Then

[18]Without the $k^{\frac{k}{2}}$ factor – i.e., they are of Gevrey class G^0 instead of $G^{\frac{1}{2}}$.

we consider the training of quantum autoencoders, which requires a slightly different formalism. We show that our gradient descent algorithms can be applied to these problems by reducing such problems to a probability maximization problem. For each application our quantum gradient computation algorithm yields a quadratic speedup in terms of the dimension.

4.7.1 Variational quantum eigensolvers

In recent years, variational quantum eigensolvers and QAOA [PMS$^+$14, WHT15, FGG14] have been favored methods for providing low-depth quantum algorithms for solving important problems in quantum simulation and optimization. Current quantum computers are limited by decoherence, hence the option to solve optimization problems using very short circuits can be enticing even if such algorithms are polynomially more expensive than alternative strategies that could possibly require long gate sequences. Since these methods are typically envisioned as being appropriate only for low-depth applications, comparably less attention is paid to the question of what their complexity would be, if they were executed on a fault-tolerant quantum computer. In this section, we consider the case that these algorithms are in fact implemented on a fault-tolerant quantum computer and show that the gradient computation step in these algorithms can be performed quadratically faster compared to the earlier approaches that were tailored for pre-fault-tolerant applications.

Variational quantum eigensolvers (VQEs) are widely used to estimate the eigenvalue corresponding to some eigenstate of a Hamiltonian. The idea behind these approaches is to begin with an efficiently parameterizable ansatz to the eigenstate. For the example of ground state energy estimation, the ansatz state is often taken to be a unitary coupled cluster expansion. The terms in that unitary coupled cluster expansion are then varied to provide the lowest energy for the groundstate. For excited states a similar argument can be applied, but minimizing a free energy rather than ground state energy is the most natural approach.

For simplicity, let us focus on the simplest (and most common) example of groundstate estimation. Consider a Hamiltonian of the form $H = \sum_j a_j U_j$ where U_j is a unitary matrix, $a_j > 0$ and $\sum_j a_j = 1$. This assumption can be made without loss of generality by renormalizing the Hamiltonian and absorbing signs into the unitary matrix. Let the state $|\psi(\boldsymbol{x})\rangle$ for $\boldsymbol{x} \in \mathbb{R}^d$ be the variational state prepared by the Prep. and Tuned circuits in Fig. 4.1b. Our objective function is then to estimate

$$\boldsymbol{x}_{\text{opt}} = \operatorname*{argmin}_{\boldsymbol{x}} \left(\langle \psi(\boldsymbol{x})| \sum_j a_j U_j |\psi(\boldsymbol{x})\rangle \right), \tag{4.21}$$

which is real-valued because H is Hermitian.

In order to translate this problem to one that we can handle using our gradient descent algorithm, we construct a verifier circuit that given $|\psi(\boldsymbol{x})\rangle$ sets an ancilla qubit to 1 with probability $p = (1 + \langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle)/2$. This is possible since $\|H\| \leq 1$ due to the assumption that $\sum_j a_j = 1$. This motivates the definition of new input oracles used for implementing the Hamiltonian.

$$\text{prepareW}\colon |0\rangle \mapsto \sum_j \sqrt{a_j}|j\rangle, \tag{4.22}$$

$$\text{selectH} := \sum_j |j\rangle\langle j| \otimes U_j. \tag{4.23}$$

We can then use the unitaries (4.22)-(4.23) to define and compute the query complexity of performing a single variational step in a VQE algorithm.

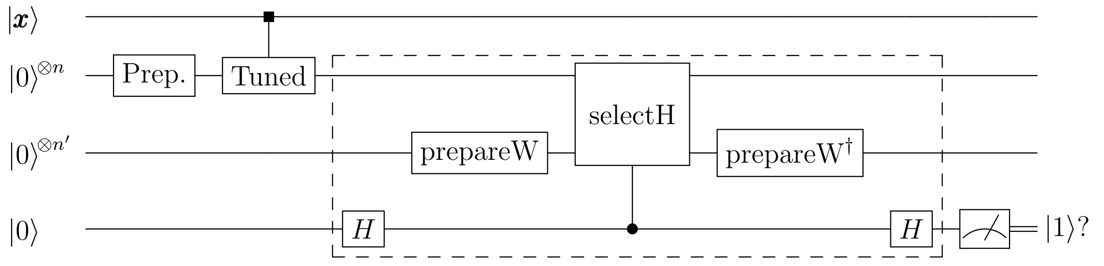


Figure 4.2. Circuit for converting groundstate energy to a probability for VQE. The dashed box denotes the verifier circuit, V, in Fig. 4.1b which corresponds here to the Hadamard test circuit. Probability of measuring 1 is $1/2 - \langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle/2$. Note here Prep. circuit is the identity gate, which is kept in the circuit only for clarity. Note that in this context, the final prepareW† can be omitted.

4.7.1. Corollary. *Let* Tuned$(\boldsymbol{x}) = \prod_{j=1}^d e^{-iH_jx_j}$ *for* $\|H_j\| = 1$ *and* $\boldsymbol{x} \in \mathbb{R}^d$ *and let* Prep $= I$. *If* $H = \sum_{j=1}^M a_jH_j$ *for unitary* H_j *with* $a_j \geq 0$ *and* $\sum_j a_j = 1$ *then the number of queries to prepareW, selectH and Tuned needed to output a qubit string* $|\boldsymbol{y}\rangle$ *such that* $|\nabla\langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle - \boldsymbol{y}| \leq \varepsilon$ *with probability at least* $2/3$ *is* $\widetilde{\mathcal{O}}\left(\sqrt{d}/\varepsilon\right)$.

Proof:
First we argue that the circuit in Fig. 4.2 outputs the claimed probability. We then convert this into a phase oracle, use our improvement over Jordan's algorithm (Theorem 4.5.10) and prove that $c \in \mathcal{O}(1)$ for this problem to show the claimed complexity.

First, if we examine the gates in Fig. 4.2 we note that the Prep. and Tuned gates by definition prepare the state $|\psi(\boldsymbol{x})\rangle$. In this context the prep circuit is the identity. While this could be trivially removed from the circuit, we retain it to match the formal description of the model that we gave earlier. Under the assumption that $\sum_j a_j = 1$ note that

$$\begin{aligned}\langle 0|\langle\psi(\boldsymbol{x})|\text{prepareW}^\dagger(\text{selectH})\text{prepareW}|0\rangle|\psi(\boldsymbol{x})\rangle &= \\ &= \sum_j\sum_k \sqrt{a_j a_k}\langle k|j\rangle \otimes \langle\psi(\boldsymbol{x})|U_j|\psi(\boldsymbol{x})\rangle. \\ &= \langle\psi(\boldsymbol{x})|\sum_j a_j U_j|\psi(\boldsymbol{x})\rangle = \langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle. \quad (4.24)\end{aligned}$$

Then because prepareW is unitary it follows that controlling the selectH operation enacts the controlled prepareW†(selectH)prepareW operation.

The claim regarding the probability then follows directly from the Hadamard test, which we describe below for completeness. Let $\Lambda(U)$ be a controlled unitary operation. Then

$$\begin{aligned}H\Lambda(U)H|0\rangle|\psi(\boldsymbol{x})\rangle &= H(|0\rangle|\psi(\boldsymbol{x})\rangle + |1\rangle U|\psi(\boldsymbol{x})\rangle)/\sqrt{2} \\ &= |0\rangle\left(\frac{(1+U)|\psi(\boldsymbol{x})\rangle}{2}\right) + |1\rangle\left(\frac{(1-U)|\psi(\boldsymbol{x})\rangle}{2}\right). \quad (4.25)\end{aligned}$$

Thus it follows from Born's rule that the probability of measuring the first register to be 1 is $(1 - \mathrm{Re}(\langle\psi|U|\psi\rangle))/2$. Combining this result with (4.24) and recalling that H is Hermitian gives us that the probability of measuring 1 in the output of the circuit in Fig. 4.2 is $1/2 - \langle\psi|H|\psi\rangle/2$ as claimed. Thus we have an appropriate probability oracle for the approximate groundstate energy expectation.

Each query to the circuit of Fig. 4.2 requires $\mathcal{O}(1)$ queries to prepareW and selectH. Thus the probability oracle can be simulated at cost $\mathcal{O}(1)$ fundamental queries. Now if we remove the measurement from the circuit we see that we can view the transformation as a circuit of the form

$$U|0\rangle = \sqrt{1/2 - \langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle/2}|\psi_{\text{good}}\rangle|1\rangle + \sqrt{1/2 + \langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle/2}|\psi_{\text{bad}}\rangle|0\rangle. \quad (4.26)$$

The above unitary (4.26) is exactly of the form described by Definition 4.4.1. By Corollary 4.4.4 for any $\delta \in (0, 1/3)$ we can simulate a δ-approximate query to the phase oracle analogue of U using $\mathcal{O}(\log(1/\delta))$ applications of U. Therefore, this phase oracle can be implemented by $\mathcal{O}(\log(1/\delta))$ uses of selectH, prepareW, Tuned and Prep. and their (controlled) inverses.

From Theorem 4.5.10 it then follows that we can compute

$$\nabla(1/2 - \langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle/2) = -\nabla\langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle/2, \quad (4.27)$$

within error $\varepsilon/2$ and error probability at most $1/3$ using $\widetilde{\mathcal{O}}\left(c\sqrt{d}/\varepsilon\right)$ applications of selectH and prepareW. Our result then immediately follows if $c \in \widetilde{\mathcal{O}}(1)$. This is equivalent to proving that for some $c \in \widetilde{\mathcal{O}}(1)$ we have that $|\partial_{\alpha_1} \cdots \partial_{\alpha_k} \langle\psi|H|\psi\rangle| \leq c^k$ holds for all $k \in \mathbb{N}$ and $\alpha \in [d]^k$.

We prove that for this application $c \leq 2$. To see this, note that for any index sequence $\alpha \in [d]^k$

$$\begin{aligned}|\partial_\alpha \langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle| &\leq \left\| \partial_\alpha \left(\prod_{j=d}^{1} e^{iH_j x_j} H \prod_{j=1}^{d} e^{-iH_j x_j} \right) \right\| \\ &\leq \sum_{\ell=1}^{M} |a_\ell| \left\| \partial_\alpha \left(\prod_{j=d}^{1} e^{iH_j x_j} H_\ell \prod_{j=1}^{d} e^{-iH_j x_j} \right) \right\|. \end{aligned} \tag{4.28}$$

Lemma 4.5.11 directly implies that

$$\left\| \partial_\alpha \prod_{j=d}^{1} e^{iH_j x_j} \right\| = 1, \tag{4.29}$$

and similarly because H_ℓ is unitary and Hermitian for each ℓ Lemma 4.5.11 also implies,

$$\left\| H_\ell \partial_\alpha \prod_{j=1}^{d} e^{-iH_j x_j} \right\| = 1. \tag{4.30}$$

Finally, Lemma 4.5.12 in concert with Eq. (4.29) and (4.30) then implies

$$\sum_{\ell=1}^{M} |a_\ell| \left\| \partial_\alpha \left(\prod_{j=d}^{1} e^{iH_j x_j} H_\ell \prod_{j=1}^{d} e^{-iH_j x_j} \right) \right\| \leq \sum_{\ell=1}^{M} |a_\ell| 2^k = 2^k. \tag{4.31}$$

□

The application of this method to QAOA directly follows from the analysis given above. There are many flavors of the quantum approximate optimization algorithm (QAOA) [FGG14]. The core idea of the algorithm is to consider a parametrized family of states such as $|\psi(\boldsymbol{x})\rangle = \prod_{j=1}^{d} e^{-ix_j H_j}|0\rangle$. The aim is to modify the state in such a way as to maximize an objective function. In particular, if we let O be a Hermitian operator corresponding to the objective function then we wish to find $\boldsymbol{x}$ such that $\langle\psi(\boldsymbol{x})|H|\psi(\boldsymbol{x})\rangle$ is maximized. For example, in the case of combinatorial optimization problems the objective function is usually expressed as the number of satisfied clauses: $O = \sum_{\alpha=1}^{m} C_\alpha$ where C_α is 1 if and only if the α^{th} clause is satisfied and 0 otherwise [FGG14]. Such clauses can be expressed as sums of tensor products of Pauli operators, which allows us to express them as Hermitian operators. Thus, from the perspective of our algorithm, QAOA looks exactly like variational quantum eigensolvers except that the parameterization chosen for the state may be significantly different from that chosen for variational quantum eigensolvers.

4.7.2 Quantum auto-encoders

Classically, one application of neural networks is *auto-encoders*, which are networks that encode information about a data set into a low-dimensional representation. Auto-encoding was first introduced by Rumelhart et al. [RHW86]. Informally, the goal of an auto-encoding circuit is the following: suppose we are given a set of high-dimensional vectors, we would like to learn a representation of the vectors hopefully of low dimension, so that computations on the original data set can be "approximately" carried out by working only with the low-dimensional vectors. More precisely the problem in auto-encoding is: Given $K < N$ and m data vectors $\{v_1, \ldots, v_m\} \subseteq \mathbb{R}^N$, find an encoding map $\mathcal{E} : \mathbb{R}^N \to \mathbb{R}^K$ and decoding map $\mathcal{D} : \mathbb{R}^K \to \mathbb{R}^N$ such that the average squared distortion $\|v_i - (\mathcal{D} \circ \mathcal{E})(v_i)\|^2$ is minimized:[19]

$$\min_{\mathcal{E},\mathcal{D}} \sum_{i\in[m]} \frac{\|v_i - (\mathcal{D} \circ \mathcal{E})(v_i)\|^2}{m}. \tag{4.32}$$

What makes auto-encoding interesting is that it does not assume any prior knowledge about the data set. This makes it a viable technique in machine learning, with various applications in natural language processing, training neural networks, object classification, prediction or extrapolation of information, etc.

Given that classical auto-encoders are 'work-horses' of classical machine learning [Azo94], it is also natural to consider a quantum variant of this paradigm. Very recently such quantum auto-encoding schemes have been proposed by Wan Kwak et al. [WKGK16] and independently by Romero et al. [ROAG17]. Inspired by their work we provide a slightly generalized description of quantum auto-encoders by 'quantizing' auto-encoders the following way: we replace the data vectors v_i by quantum states ρ_i and define the maps $\mathcal{E}, \mathcal{D}$ as quantum channels transforming states back and forth between the Hilbert spaces $\mathcal{H}$ and $\mathcal{H}'$. A natural generalization of squared distortion for quantum states ρ, σ that we consider is $1 - F^2(\rho, \sigma)$,[20] giving us the following minimization problem

$$\min_{\mathcal{E},\mathcal{D}} \sum_{i\in[m]} \frac{1 - F^2(\rho_i, (\mathcal{D} \circ \mathcal{E})(\rho_i))}{m}. \tag{4.33}$$

Since $F^2(|\psi\rangle\langle\psi|, \sigma) = \langle\psi|\sigma|\psi\rangle$ in the special case when the input states are pure states $\rho_i = |\psi_i\rangle\langle\psi_i|$, the above minimization problem is equivalent to the maximization problem

$$\max_{\mathcal{E},\mathcal{D}} \sum_{i\in[N]} \frac{\langle\psi_i|[(\mathcal{D} \circ \mathcal{E})(|\psi_i\rangle\langle\psi_i|)]|\psi_i\rangle}{m}. \tag{4.34}$$

[19] There are other natural choices of dissimilarity functions that one might want to minimize, for a comprehensive overview of the classical literature see [Bal12].

[20] Note that some authors (including [ROAG17]) call $F' = F^2$ the fidelity. The distortion measure we use here is $P(\rho, \sigma) = \sqrt{1 - F^2(\rho, \sigma)}$, which is called the purified (trace) distance [TCR10].

Observe that $\langle\psi|[(\mathcal{D}\circ\mathcal{E})(|\psi\rangle\langle\psi|)]|\psi\rangle$ is the probability of finding the output state $(\mathcal{D}\circ\mathcal{E})(|\psi\rangle\langle\psi|)$ in state $|\psi\rangle$ after performing the projective measurement

$$\{|\psi\rangle\langle\psi|, I - |\psi\rangle\langle\psi|\}.$$

Thus we can think about this as maximizing the probability of recovering the initial pure state after encoding and decoding, which is a natural measure of the quality of the quantum auto-encoding procedure.

Training quantum auto-encoders

Similarly to [WKGK16, ROAG17] we describe a way to perform this optimization problem in the special case when the input states are n-qubit pure states and they are mapped to k-qubit states, i.e., $\mathcal{H}$ is the Hilbert space of n qubits and $\mathcal{H}'$ is the Hilbert space of $k < n$ qubits. We also show how our gradient computation algorithm can speed up solving the described optimization problem.

We observe that by adding a linear amount of ancilla qubits we can represent the encoding and decoding channels by unitaries, which makes the minimization conceptually simpler. Indeed by Stinespring's dilation theorem [Wat18, Corollary 2.27], [Key02] we know that any quantum channel $\mathcal{E}$ that maps n qubit states to k qubit states can be constructed by adding $2k$ qubits initialized in $|\vec{0}\rangle$ state, then acting with a unitary $U_{\mathcal{E}}$ on the extended space and then tracing out $k+n$ qubits. Applying this result to both $\mathcal{E}$ and $\mathcal{D}$ results in a unitary circuit representing the generic encoding/decoding procedure, see Figure 4.3. (This upper bound on the required number of ancilla qubits for $\mathcal{D}$ becomes $2n$.)

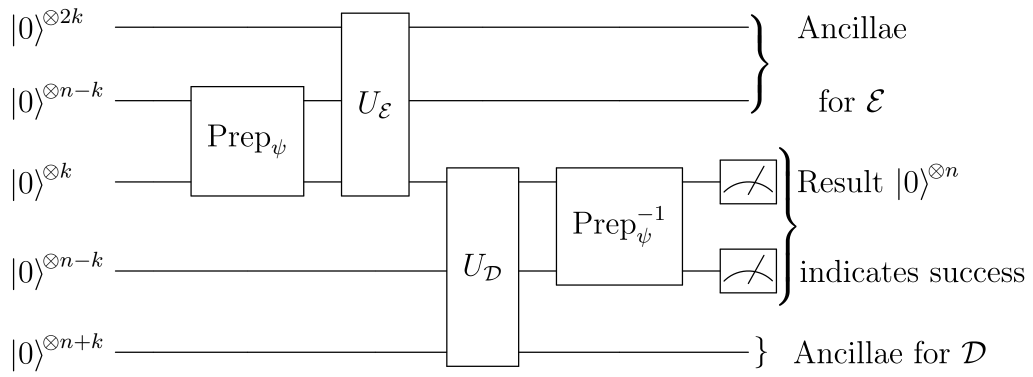

$|0\rangle^{\otimes 2k}$
$|0\rangle^{\otimes n-k}$
$|0\rangle^{\otimes k}$
$|0\rangle^{\otimes n-k}$
$|0\rangle^{\otimes n+k}$
Prep_ψ
$U_{\mathcal{E}}$
$U_{\mathcal{D}}$
Prep_ψ^{-1}
Ancillae for $\mathcal{E}$
Result $|0\rangle^{\otimes n}$ indicates success
Ancillae for $\mathcal{D}$

Figure 4.3. A unitary quantum auto-encoding circuit: For the input $|\psi\rangle$, the circuit prepares $|\psi\rangle$, applies a purified version of the channels $\mathcal{E}, \mathcal{D}$ and finally checks by a measurement whether the decoded state is $|\psi\rangle$.

In order to solve the maximization problem (4.34) we could just introduce a parametrization of the unitaries $U_{\mathcal{E}}, U_{\mathcal{D}}$ and search for the optimal parameters using gradient descent. Unfortunately a complete parametrization of the unitaries requires exponentially many parameters, which is prohibitive. However, analogously to, e.g., classical machine learning practices, one could hope that a well-structured circuit can achieve close to optimal performance using only a polynomial number of parameters. If the circuits $U_{\mathcal{E}}, U_{\mathcal{D}}$ are parametrized nicely, so that Lemma 4.5.11 can be applied, then we can use our gradient computation algorithm to speed up optimization.

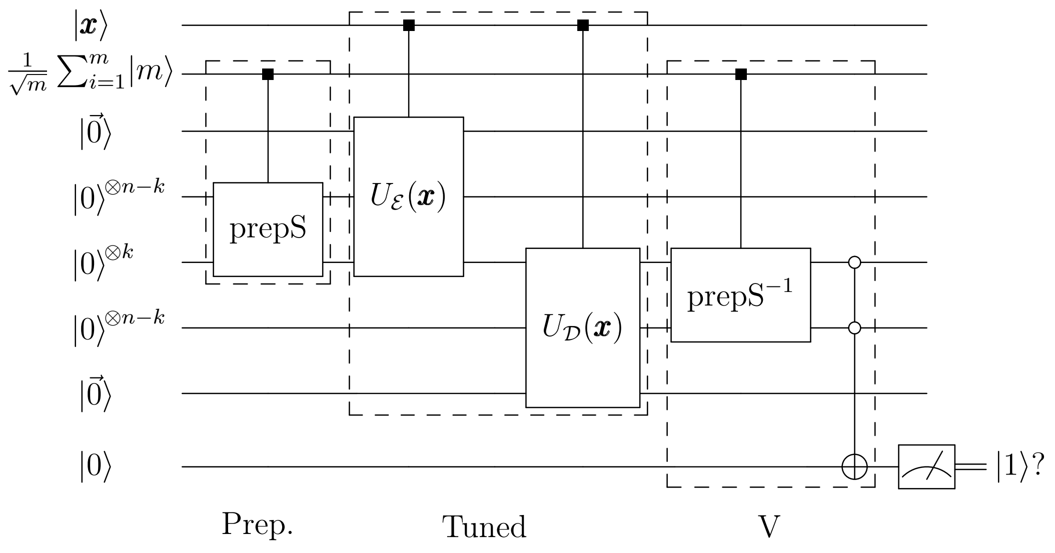

$|\boldsymbol{x}\rangle$
$\frac{1}{\sqrt{m}}\sum_{i=1}^{m}|m\rangle$
$|\vec{0}\rangle$
$|0\rangle^{\otimes n-k}$
$|0\rangle^{\otimes k}$
$|0\rangle^{\otimes n-k}$
$|\vec{0}\rangle$
$|0\rangle$
prepS
$U_{\mathcal{E}}(\boldsymbol{x})$
$U_{\mathcal{D}}(\boldsymbol{x})$
prepS^{-1}
$|1\rangle$?
Prep.
Tuned
V

Figure 4.4. Quantum circuit which outputs 1 with probability equal to the objective function (4.34). The structure of the circuit fits the generic model of quantum optimization circuits (Figure 4.1), therefore we can use our gradient computation methods to speedup its optimization.

We can do the whole optimization using stochastic gradient descent [JKK$^+$17], so that in each step we only need to consider the effect of the circuit on a single pure state. Or if we have more quantum resources available we can directly evaluate the full gradient by preparing a uniform superposition over all input vectors. In this case the state preparation unitary $\text{Prep} = \sum_{i=1}^{m} |i\rangle\langle i| \otimes \text{Prep}_{\psi_i}$ is a controlled unitary, which controlled on index i would prepare $|\psi_i\rangle$. Graphically we represent this type of control by a small black square in contrast to the small black circle used for denoting simple controlled unitaries. See the full quantum circuit in Figure 4.4.

Finally, note that in some application it might be desirable to ask for a coherent encoding/decoding procedure, where all the ancilla qubits are returned to the $|\vec{0}\rangle$ state. In this case similarly to [WKGK16, ROAG17] one could define $U_{\mathcal{D}} = U_{\mathcal{E}}^{-1}$ and optimize the probability of measuring $|\vec{0}\rangle$ on the ancilla qubits after applying $U_{\mathcal{E}}$.

4.8 Conclusion and future research

We gave a new approach to quantum gradient computation that is asymptotically optimal (up to logarithmic factors) for a class of smooth functions, in terms of the number of queries needed to estimate the gradient within fixed error with respect to the max-norm. This is based on several new ideas including the use of differentiation formulæ originating from high-degree interpolatory polynomials. These high-degree methods quadratically improve the scaling of the query complexity with respect to the approximation quality compared to what one would see if the results from Jordan's work were used. In the case of low-degree multivariate polynomials we showed that our algorithm can yield an exponential speedup compared to Jordan's algorithm or classical algorithms. We also provided lower bounds on the query complexity of the problem for certain smooth functions revealing that our algorithm is essentially optimal for a class of functions.

While it has proven difficult to find natural applications for Jordan's original algorithm, we provide in this chapter several applications of our gradient descent algorithm to areas ranging from machine learning to quantum chemistry simulation. These applications are built upon a method we provide for interconverting between phase and probability oracles. The polynomial speedups that we see for these applications is made possible by our improved quantum gradient algorithm via the use of this interconversion process. It would be interesting to find applications where we can apply the results for low-degree multivariate polynomials providing an exponential speedup.

A major open problem is to understand whether it is possible to reduce the number of required iterations in gradient descent using quantum techniques. It would be also interesting to see how quantum computers can speed up more sophisticated higher-level methods such as stochastic gradient descent. Another interesting question is whether quantum algorithms can provide further speedups for calculating higher-order derivatives, such as the Hessian, using ideas related to Jordan's algorithm, see e.g. [Jor08, Appendix D]. Such improvements might open the door for improved quantum analogues of Newton's method and in turn substantially improve the scaling of the number of epochs needed to converge to a local optima in quantum methods.

4.A Error bounds on central-difference formulas

In this appendix we develop error bounds on finite-difference formulas using some higher-dimensional calculus. The goal is to give upper bounds on the query complexity of gradient computation of f under some smoothness conditions that f satisfies. The main idea is to use Algorithm 4.1 in combination with central-difference formulas and analyze the query complexity using some technical lemmas involving higher-dimensional calculus. We first prove Theorem 4.5.7, which gives rise to a quantum algorithm that yields potentially exponential speedups for low-degree polynomial functions. The query complexity bound that we can derive for smooth functions using Theorem 4.5.7 scales as $\widetilde{O}(d/\varepsilon)$ (which is already an improvement in $1/\varepsilon$, but worse in d compared to Jordan's algorithm), which we later improve to $\widetilde{O}(\sqrt{d}/\varepsilon)$ via Theorem 4.5.9.

In the proof of the following lemma we will use Stirling's approximation of the factorial:

4.A.1. Fact (Stirling's approximation). For every $j \in \mathbb{N}_+$, we have

$$\sqrt{2\pi j}\left(\frac{j}{e}\right)^j \leq j! \leq e\sqrt{j}\left(\frac{j}{e}\right)^j. \tag{4.35}$$

As a first step towards proving Theorem 4.5.7 and Theorem 4.5.9 we derive a bound on the following sum of coefficients which appears in central-difference formulas (recall Definition 4.2.4):

4.A.2. Lemma. *For all $m \in \mathbb{N}_+$ and $k \geq 2m$ we have*[21]

$$\sum_{\ell=-m}^{m}\left|a_\ell^{(2m)}\ell^{k+1}\right| \leq 6e^{-\frac{7m}{6}}m^{k+3/2}, \tag{4.36}$$

where $a_0^{(2m)} = 0$ and for $\ell \neq 0$

$$a_\ell^{(2m)} = \frac{(-1)^{\ell-1}}{\ell}\frac{\binom{m}{|\ell|}}{\binom{m+|\ell|}{|\ell|}}.$$

Proof:
First we bound the left-hand side of (4.36) as follows,

$$\sum_{\ell=-m}^{m}\left|a_\ell^{(2m)}\ell^{k+1}\right| = 2\sum_{\ell=1}^{m}\frac{\binom{m}{\ell}}{\binom{m+\ell}{\ell}}\ell^k \leq 2m \cdot \max_{\ell\in[m]}\frac{\binom{m}{\ell}}{\binom{m+\ell}{\ell}}\ell^k. \tag{4.37}$$

We now upper bound the binomial quantity on the right as follows. For every

[21] Sometimes we will be interested in bounding $\left|\sum_{\ell=-m}^{m}a_\ell^{(2m)}\ell^{k+1}\right|$ rather than the left-hand side of (4.36). One could ask how good this bound is, do not we lose too much by dismissing the $(-1)^\ell$ cancellations? It turns out that the most important case for us is when $k = 2m$. In this case our numerical experiments showed that the quantity $\left|\sum_{\ell=-m}^{m}a_\ell^{(2m)}\ell^{k+1}\right|$ is lower bounded by $(m/e)^{2m}$, providing evidence for showing that the proven upper bound is qualitatively right.

$\ell \in [m]$, we have

$$\begin{aligned}
\frac{\binom{m}{\ell}}{\binom{m+\ell}{\ell}}\ell^k &= \frac{(m!)^2}{(m+\ell)!(m-\ell)!}\ell^k \\
&\leq \frac{e^2 m\left(\frac{m}{e}\right)^{2m}}{\sqrt{2\pi(m+\ell)}\left(\frac{m+\ell}{e}\right)^{m+\ell}\left(\frac{m-\ell}{e}\right)^{m-\ell}}\ell^k && \text{(using}^{22}\text{Fact 4.A.1)} \\
&\leq 3\sqrt{m}\frac{\left(\frac{m}{e}\right)^{2m}}{\left(\frac{m+\ell}{e}\right)^{m+\ell}\left(\frac{m-\ell}{e}\right)^{m-\ell}}\ell^k && (\text{since } e^2/\sqrt{2\pi} \leq 3) \\
&= 3\sqrt{m}\frac{\left(\frac{m}{e}\right)^{2m}}{\left(\frac{(1+y)m}{e}\right)^{(1+y)m}\left(\frac{(1-y)m}{e}\right)^{(1-y)m}}(ym)^k && (\text{substitute } y := \ell/m) \\
&= 3\sqrt{m}\left(\frac{1}{(1+y)^{1+y}(1-y)^{1-y}}\right)^m (ym)^k \\
&= 3\sqrt{m}\left(\frac{y^z}{(1+y)^{1+y}(1-y)^{1-y}}\right)^m m^k && (\text{substitute } z := k/m) \\
&\leq 3\sqrt{m}\left(\frac{y^2}{(1+y)^{1+y}(1-y)^{1-y}}\right)^m m^k && (y \leq 1 \text{ and } z \geq 2) \\
&\leq 3\sqrt{m}\left(e^{-\frac{7}{6}}\right)^m m^k. && \text{(by elementary calculus)}
\end{aligned}$$

□

Now we are ready to prove Lemma 4.5.6 from Section 4.5.3, which we restate here.

4.A.3. Lemma. *Let $\delta \in \mathbb{R}_+$, $m \in \mathbb{N}$ and suppose $f : [-m\delta, m\delta] \to \mathbb{R}$ is $(2m+1)$-times differentiable. Then*

$$\left|f'(0)\delta - f_{(2m)}(\delta)\right| = \left|f'(0)\delta - \sum_{\ell=-m}^{m} a_\ell^{(2m)} f(\ell\delta)\right| \leq e^{-\frac{m}{2}} \left\|f^{(2m+1)}\right\|_\infty |\delta|^{2m+1}, \tag{4.11}$$

where $\left\|f^{(2m+1)}\right\|_\infty := \sup_{\xi\in[-m\delta,m\delta]} |f^{(2m+1)}(\xi)|$. Moreover,

$$\sum_{\ell=0}^{m}\left|a_\ell^{(2m)}\right| < \sum_{\ell=1}^{m}\frac{1}{\ell} \leq \ln(m) + 1. \tag{4.12}$$

Proof:
We prove this lemma as follows: first we approximate $f(x)$ using order-$(2m)$

[22] Additionally to Stirling's approximation we also used that $\left(\frac{m-\ell}{e}\right)^{m-\ell} \leq (m-\ell)!$, which is true even for $m - \ell = 0$.

Taylor expansion, and bound the error using Lagrange remainder term. Then we use Lagrange interpolation polynomials to re-express the Taylor polynomial, and use this interpolation formula to approximately compute the derivative of f at 0 yielding the $(2m)$-th central-difference formula. Finally, we use Lemma 4.A.2 to upper bound the difference between $f'(0)\delta$ and the $(2m)$-th central-difference formula $f_{(2m)}(\delta)$.

Recall that Taylor's theorem with Lagrange remainder term says that for all $y \in \mathbb{R}$,

$$f(y) = \underbrace{\sum_{j=0}^{2m} \frac{f^{(j)}(0)}{j!} y^j}_{:=p(y/\delta)} + \frac{f^{(2m+1)}(\xi)}{(2m+1)!} y^{2m+1} \tag{4.38}$$

for some $\xi \in [0, y]$ (in case $y < 0$, then $\xi \in [y, 0]$). Now let $z := y/\delta$. We introduce a δ-scaled version of the $(2m)$-th order Taylor polynomial at 0, which we will use for re-expressing $f'(0)$:

$$p(z) := -\sum_{j=0}^{2m} \frac{f^{(j)}(0)}{j!} (z\delta)^j = f(z\delta) - \frac{f^{(2m+1)}(\xi)}{(2m+1)!} (z\delta)^{2m+1}. \tag{4.39}$$

Because $\deg(p) \leq 2m$ we can use the following Lagrange interpolation formula to represent it as:

$$p(z) = \sum_{\ell=-m}^{m} p(\ell) \prod_{\substack{i=-m \\ i\neq\ell}}^{m} \frac{z-i}{\ell-i}.$$

Using the above identity, it is not hard to see that $p'(0)$ equals

$$\sum_{\substack{\ell=-m \\ \ell\neq 0}}^{m} p(\ell) \frac{(m!)^2}{-\ell} \frac{(-1)^\ell}{(m+|\ell|)!(m-|\ell|)!} = \sum_{\substack{\ell=-m \\ \ell\neq 0}}^{m} (-1)^{\ell-1} \frac{\binom{m}{|\ell|}}{\binom{m+|\ell|}{|\ell|}} \frac{p(\ell)}{\ell} = \sum_{\ell=-m}^{m} a_\ell^{(2m)} p(\ell). \tag{4.40}$$

Observe that by definition $p'(0) = f'(0)\delta$, therefore

$$\begin{aligned} f'(0)\delta &= p'(0) \\ &\overset{(4.40)}{=} \sum_{\ell=-m}^{m} a_\ell^{(2m)} p(\ell) \\ &\overset{(4.39)}{=} \sum_{\ell=-m}^{m} a_\ell^{(2m)} \left(f(\ell\delta) - \frac{f^{(2m+1)}(\xi_\ell)}{(2m+1)!} \ell^{2m+1} \delta^{2m+1} \right). \end{aligned}$$

Now we bound the left-hand side of (4.11) using the above equality:

$$\begin{aligned}
\left|\sum_{\ell=-m}^{m} a_\ell^{(2m)} \frac{f^{(2m+1)}(\xi_\ell)}{(2m+1)!}\ell^{2m+1}\delta^{2m+1}\right| &\leq \sum_{\ell=-m}^{m}\left|a_\ell^{(2m)}\ell^{2m+1}\right| \frac{\left\|f^{(2m+1)}\right\|_\infty}{(2m+1)!}|\delta|^{2m+1}\\
&\leq 6e^{-\frac{7m}{6}} m^{2m+3/2}\frac{\left\|f^{(2m+1)}\right\|_\infty}{(2m+1)!}|\delta|^{2m+1}\\
&\qquad\text{(Lemma 4.A.2 with } k{:=}2m)\\
&\leq 3e^{-\frac{7m}{6}} m^{2m+1/2}\frac{\left\|f^{(2m+1)}\right\|_\infty}{(2m)!}|\delta|^{2m+1}\\
&\leq 3e^{-\frac{7m}{6}} m^{2m+1/2}\frac{\left\|f^{(2m+1)}\right\|_\infty}{\sqrt{4\pi m}(2m/e)^{2m}}|\delta|^{2m+1}\\
&\qquad\text{(using Fact 4.A.1)}\\
&\leq e^{-\frac{7m}{6}}\left(\frac{e}{2}\right)^{2m}\left\|f^{(2m+1)}\right\|_\infty|\delta|^{2m+1}\\
&\qquad\left(\text{since } \tfrac{3}{\sqrt{4\pi}}\leq 1\right)\\
&\leq e^{-\frac{m}{2}}\left\|f^{(2m+1)}\right\|_\infty|\delta|^{2m+1}.\\
&\qquad\left(\text{since } e^{-\frac{7m}{6}}\left(\tfrac{e}{2}\right)^{2m}\leq e^{-m/2}\right)
\end{aligned}$$

Finally, the first inequality[23] in (4.12) holds element-wise and the second inequality is standard from elementary calculus, and can be proven using the integral of $1/x$. □

We now prove a version of Lemma 4.5.6 but for higher-dimensional functions, by making the assumption that all the higher derivatives are bounded.

4.A.4. COROLLARY. *Let $m \in \mathbb{N}$, $B > 0$, $\boldsymbol{x} \in \mathbb{R}^d$ and $\boldsymbol{r} := \boldsymbol{x}/\|\boldsymbol{x}\|$. Suppose $f : [-m\|\boldsymbol{x}\|_\infty, m\|\boldsymbol{x}\|_\infty]^d \to \mathbb{R}$ is $(2m+1)$-times differentiable and*

$$|\partial_{\boldsymbol{r}}^{2m+1} f(\tau\boldsymbol{x})| \leq B \quad \textit{for all } \tau \in [-m, m],$$

then

$$\left|f_{(2m)}(\boldsymbol{x}) - \nabla f(\mathbf{0})\cdot\boldsymbol{x}\right| \leq Be^{-\frac{m}{2}}\|\boldsymbol{x}\|^{2m+1}.$$

Proof:

[23] We conjecture that the first inequality of (4.12) becomes an equality if we take half of the middle term.

Consider the function $h(\tau) := f(\tau\boldsymbol{x})$, then

$$\begin{aligned}
\left|f_{(2m)}(\boldsymbol{x}) - \nabla f(\mathbf{0}) \cdot \boldsymbol{x}\right| &= \left|h_{(2m)}(1) - h'(0)\right| \\
&\leq e^{-\frac{m}{2}} \sup_{\tau\in[-m,m]} \left|h^{(2m+1)}(\tau)\right| \qquad \text{(by Lemma 4.5.6)} \\
&= e^{-\frac{m}{2}} \sup_{\tau\in[-m,m]} \left|\partial_{\boldsymbol{x}}^{2m+1} f(\tau\boldsymbol{x})\right| \\
&= e^{-\frac{m}{2}} \sup_{\tau\in[-m,m]} \left|\partial_{\boldsymbol{r}}^{2m+1} f(\tau\boldsymbol{x})\right| \|\boldsymbol{x}\|^{2m+1} \\
&\leq B e^{-\frac{m}{2}} \|\boldsymbol{x}\|^{2m+1}.
\end{aligned}$$

□

With this corollary in hand, we now show how to calculate the gradient of $f : \mathbb{R}^d \to \mathbb{R}$ under a bounded higher derivative condition.

4.5.7. THEOREM. *Let $m \in \mathbb{Z}_+$, $D \in \mathbb{R}_+$ and $B \geq 0$. Suppose $f : [-D, D]^d \to \mathbb{R}$ is given with (fractional) phase oracle access. If f is $(2m+1)$-times differentiable and for all $\boldsymbol{x} \in [-D, D]^d$ we have that*

$$|\partial_{\boldsymbol{r}}^{2m+1} f(\boldsymbol{x})| \leq B \quad \text{for } \boldsymbol{r} = \boldsymbol{x}/\|\boldsymbol{x}\|,$$

then using Algorithm 4.2 with setting

$$R = \Theta\left(\max\left(\sqrt{d}\sqrt[2m]{\frac{B\sqrt{d}}{\varepsilon}}, \frac{m}{D}\right)\right)$$

we can compute an approximate gradient $\boldsymbol{g}$ such that $\|\boldsymbol{g} - \nabla f(\mathbf{0})\|_\infty \leq \varepsilon$ with probability at least $(1-\rho)$, using $\mathcal{O}\Big(\big(\frac{R}{\varepsilon}\log(2m) + m\big)\log\Big(\frac{d}{\rho}\Big)\Big)$ phase queries.

Proof:
Let $r_{\text{opt}} := \frac{2}{\sqrt{d}}\sqrt[2m]{\frac{\varepsilon e^{\frac{m}{2}}}{B\sqrt{d}\cdot 4\cdot 42\cdot\pi}}$, and let $r := \min\big(r_{\text{opt}}, \frac{2D}{m}\big)$. By Corollary 4.A.4 we get that whenever $\|\boldsymbol{x}\|_\infty \leq r/2$ we have

$$\begin{aligned}
\left|f_{(2m)}(\boldsymbol{x}) - \nabla f(\mathbf{0}) \cdot \boldsymbol{x}\right| &\leq B e^{-\frac{m}{2}} \|\boldsymbol{x}\|^{2m+1} \\
&\leq B e^{-\frac{m}{2}} \left(r\frac{\sqrt{d}}{2}\right)^{2m+1} \\
&= B e^{-\frac{m}{2}} \left(r_{\text{opt}}\frac{\sqrt{d}}{2}\right)^{2m+1} \left(\frac{r}{r_{\text{opt}}}\right)^{2m+1} \\
&= \frac{\varepsilon r_{\text{opt}}}{8\cdot 42\cdot\pi}\left(\frac{r}{r_{\text{opt}}}\right)^{2m+1} \\
&\leq \frac{\varepsilon r}{8\cdot 42\cdot\pi}.
\end{aligned}$$

Assume without loss of generality that $\frac{1}{\varepsilon r} = 2^n$ for some $n \in \mathbb{N}$. In Theorem 4.5.5, we showed that $\mathcal{O}\Big(\log\Big(\frac{d}{\rho}\Big)\Big)$ queries to the phase oracle $\mathrm{O}_{f_{(2m)}}^{2^{n+1}\pi}$ suffice to compute an ε-precise approximation of the gradient with probability $\geq 1-\rho$. Now observe that the phase oracle

$$\mathrm{O}_{f_{(2m)}}^{2^{n+1}\pi}(\boldsymbol{x}) = \prod_{\ell=-m}^{m} \mathrm{O}_f^{2^{n+1}\pi a_\ell^{(2m)}}(\ell \boldsymbol{x}) = \prod_{\ell=-m}^{m} \mathrm{O}_f^{a_\ell^{(2m)}\frac{2\pi}{\varepsilon r}}(\ell \boldsymbol{x}),$$

can be implemented using

$$\begin{aligned}\sum_{\ell=-m}^{m}\left\lceil a_\ell^{(2m)}\frac{2\pi}{\varepsilon r}\right\rceil &\leq 2m + \frac{2\pi}{\varepsilon r}\sum_{\ell=-m}^{m} a_\ell^{(2m)} \\ &\overset{(4.12)}{\leq} 2m + \frac{2\pi}{\varepsilon r}(2\log(m)+2) \\ &= \mathcal{O}\bigg(m + \frac{R}{\varepsilon}\log(2m)\bigg)\end{aligned}$$

fractional phase queries to O_f. □

Let us elaborate on the above cost by making some strong regularity assumptions on f. Suppose that for every $k \in [2m+1]$, index-sequence $\alpha \in [d]^k$ and $\boldsymbol{x} \in \mathbb{R}^d$, we have $|\partial_\alpha f(\boldsymbol{x})| \leq 1$ (implying also $B = 1$). What can we say by using the above corollary?

Well, it could happen[24] that for every $\beta \in [d]^{2m+1}$, we have $\partial_\beta f(\mathbf{0}) = 1$. Then by Eq. (4.5), by picking $\boldsymbol{r} := \mathbf{1}/\sqrt{d}$ we have $\partial_{\boldsymbol{r}}^{(2m+1)} f(\mathbf{0}) = d^{\frac{2m+1}{2}}$. This is actually the worst possible case under our assumptions, it is easy to show that whenever $\|\boldsymbol{r}\| \leq 1$, we must have $|\partial_{\boldsymbol{r}}^{(2m+1)} f(\boldsymbol{x})| \leq B = d^{\frac{2m+1}{2}}$ for all $\boldsymbol{x} \in \mathbb{R}^d$. In this case the best complexity we can get from Theorem 4.5.7 is by choosing $m = \log(d/\varepsilon)$ which yields an overall query complexity upper bound of $\mathcal{O}\big(\frac{d}{\varepsilon}\log(d/\rho)\log\log(d/\varepsilon)\big)$.

This bound achieves the desired $O(1/\varepsilon)$-scaling precision parameter ε, but fails to grasp the $\sqrt{d}$ scaling. This failure is mainly due to the loose upper bound on B. Also as we discussed in Section 4.6.3, we cannot really hope to achieve a $\sqrt{d}$ scaling with an algorithm that implements a phase oracle for an approximate affine function that uniformly approximates an affine function for all points of the hypergrid. But fortunately, as we showed in Theorem 4.5.5, it is sufficient if the approximation works for most of the evaluation points.

In order to rigorously prove $\sqrt{d}$ scaling with the dimension we assume that the function is analytic. We will use the following lemma for answering the question: Given a (complex) analytic function with its multi-dimensional Taylor series as in (4.4), where do we need to truncate its Taylor series if we want to get a good approximation on the d-dimensional hypercube $[-1,1]^d$?

[24] An example for such a function is $f(\boldsymbol{x}) := \sin(x_1 + x_2 + \ldots + x_d)$.

4.A.5. LEMMA. *Let $d, k \in \mathbb{N}_+$, and suppose $H \in \left(\mathbb{R}^d\right)^{\otimes k}$ is an order-k tensor of dimension d, having all elements bounded by 1 in absolute value, i.e., $\|H\|_\infty \leq 1$. Suppose $(x_1, \ldots, x_d)$ is a vector of i.i.d. symmetric random variables bounded in $[-1/2, 1/2]$. Then for every $t > 0$*

$$\Pr\left(\left|\sum_{\alpha \in [d]^k} H_\alpha x^\alpha\right| \geq \sqrt{2}\left(t\sqrt{\frac{dk}{2}}\right)^k\right) \leq \frac{1}{t^{2k}}.$$

Proof:

$$\begin{aligned}
\mathbb{E}\left[\left(\sum_{\alpha\in[d]^k} H_\alpha x^\alpha\right)^2\right] &= \mathbb{E}\left[\sum_{(\alpha,\beta)\in[d]^{2k}} H_\alpha H_\beta x^{(\alpha,\beta)}\right] \\
&\leq \mathbb{E}\left[\sum_{(\alpha,\beta)\in[d]^{2k}} x^{(\alpha,\beta)}\right] \\
&\qquad\qquad (x_i \text{ is symmetric i.i.d. and } \|H\|_\infty \leq 1) \\
&= \mathbb{E}\Big[(x_1 + x_2 + \ldots + x_d)^{2k}\Big] \\
&= \int_0^\infty \mathbb{P}\Big((x_1 + x_2 + \ldots + x_d)^{2k} \geq t\Big)dt \\
&= \int_0^\infty \mathbb{P}\big(|x_1 + x_2 + \ldots + x_d| \geq t^{1/2k}\big)dt \\
&\leq \int_0^\infty 2e^{-\left(\frac{2}{d}t^{\frac{1}{k}}\right)}dt \qquad \text{(by the Hoeffding bound)} \\
&= \int_0^\infty 2\left(\frac{d}{2}\right)^k k y^{k-1} e^{-y} dy \\
&\qquad\qquad \left(\text{by change of variables } y := \left(\left(\tfrac{2}{d}\right)^k t\right)^{\frac{1}{k}}\right) \\
&= 2\left(\frac{d}{2}\right)^k k\Gamma(k) \qquad \text{(by definition of } \Gamma(x)) \\
&= 2\left(\frac{d}{2}\right)^k k! \qquad \text{(main property of } \Gamma(x)) \\
&\leq 2e\sqrt{k}\left(\frac{dk}{2e}\right)^k \qquad \text{(Stirling's approximation)} \\
&< 2\left(\frac{dk}{2}\right)^k. \qquad \left(\text{for all } k \geq 1 : \sqrt{k}e^{1-k} \leq ke^{1-k} \leq 1\right)
\end{aligned}$$

Now use Chebyshev's inequality to conclude. □

4.A.6. Remark. The dependence on d in Lemma 4.5.8 cannot be improved, as can be seen using the central limit theorem: by choosing $H \equiv 1$ (the all-1 tensor) it is not hard to see, that for a fixed k the typical value of $\left|\sum_{\alpha\in[d]^k} H_\alpha x^\alpha\right| = \left|(x_1 + x_2 + \ldots + x_d)^k\right| = \Theta(\sqrt{d}^k)$. A natural follow-up question is whether we can improve the k-dependence, in particular the $k^{k/2}$ factor? While it is possible that one can improve the above result we show in the next paragraph that the typical value eventually becomes much larger than $\sim d^{\frac{k}{2}}$. (An interesting regime, where our discussion does not imply a lower bound, is when $k \sim \sqrt{d}$.)

Here is a counterexample to the $\sim d^{\frac{k}{2}}$ scaling: suppose $\mathbb{N} \ni a \geq 5$, $d \geq 2^a$ and $k = ad$, then let H be the tensor which is 1 for index-sequences containing each index with even multiplicity, and 0 otherwise. There are at least $d^{(a-1)d}$ such index sequences since there are $d^{(a-1)d}$ index-sequences of length $(a-1)d$ and each such index-sequence can be extended to an even-multiplicity index-sequence of length ad. Also suppose that $\Pr(|X_i| \geq 1/4) \geq 1/2$, then this tensor evaluated at every possible value of the random vector will yield at least $d^{(a-1)d}2^{-k} = d^{k-d}2^{-a(k/a)} \geq d^{\left(1-\frac{1}{a}\right)k}d^{-\frac{k}{a}} = d^{\left(1-\frac{2}{a}\right)k} \gg d^{\frac{k}{2}}$.

Now we are ready to prove Theorem 4.5.9. We restate the theorem for convenience.

4.5.9. Theorem. *If $r \in \mathbb{R}_+$, $f : \mathbb{R}^d \to \mathbb{R}$ is analytic and for all $k \in \mathbb{N}, \alpha \in [d]^k$ we have*

$$|\partial_\alpha f(\mathbf{0})| \leq c^k k^{\frac{k}{2}},$$

then

$$|\nabla f(\mathbf{0})\boldsymbol{y} - f_{(2m)}(\boldsymbol{y})| \leq \sum_{k=2m+1}^{\infty} \left(8rcm\sqrt{d}\right)^k,$$

for all but a $1/1000$ fraction of points $\boldsymbol{y} \in r \cdot G_n^d$.

Proof:
Because f is analytic it coincides with its d-dimensional Taylor series:

$$f(\boldsymbol{y}) = \sum_{k=0}^{\infty} \sum_{\alpha\in[d]^k} \frac{\boldsymbol{y}^\alpha \cdot \partial_\alpha f(\mathbf{0})}{k!}. \tag{4.41}$$

We are now going to use the central-difference formula defined earlier in Definition 4.2.4:

$$\begin{aligned}
f_{(2m)}(\boldsymbol{y}) &= \sum_{\ell=-m}^{m} a_\ell^{(2m)} f(\ell \boldsymbol{y}) && \text{(using Definition 4.2.4)}\\
&= \sum_{\ell=-m}^{m} a_\ell^{(2m)} \sum_{k=0}^{\infty} \frac{1}{k!} \sum_{\alpha\in[d]^k} (\ell\boldsymbol{y})^\alpha \cdot \partial_\alpha f(\mathbf{0}) && \text{(using Eq. 4.41)}\\
&= \sum_{k=0}^{\infty} \frac{1}{k!} \sum_{\alpha\in[d]^k} \boldsymbol{y}^\alpha \cdot \partial_\alpha f(\mathbf{0}) \underbrace{\sum_{\ell=-m}^{m} a_\ell^{(2m)} \ell^k}_{*}.
\end{aligned}$$

Now, we apply Lemma 4.5.6 to the function x^k with the choice $\delta := 1$, to conclude that $(*)$ is 0 if $k \leq 2m$ except for $k = 1$, in which case it is 1. This implies that

$$\begin{aligned}
\left|\nabla f(\mathbf{0})\boldsymbol{y} - f_{(2m)}(\boldsymbol{y})\right| &= \left|\sum_{k=2m+1}^{\infty} \frac{1}{k!} \sum_{\alpha\in[d]^k} \boldsymbol{y}^\alpha \cdot \partial_\alpha f(\mathbf{0}) \sum_{\ell=-m}^{m} a_\ell^{(2m)} \ell^k\right|\\
&\leq \sum_{k=2m+1}^{\infty} \left(\frac{e}{k}\right)^k \frac{1}{\sqrt{4\pi m}} \left|\sum_{\alpha\in[d]^k} \boldsymbol{y}^\alpha \cdot \partial_\alpha f(\mathbf{0})\right| \left|\sum_{\ell=-m}^{m} a_\ell^{(2m)} \ell^k\right|\\
&&& \text{(Stirling bound (4.35))}\\
&\leq \sum_{k=2m+1}^{\infty} \left|\sum_{\alpha\in[d]^k} \boldsymbol{y}^\alpha \cdot \partial_\alpha f(\mathbf{0})\right| \left(\frac{e}{k}\right)^k \frac{3e^{-\frac{7m}{6}} m^{k+\frac{1}{2}}}{\sqrt{\pi m}}\\
&&& \text{(Lemma 4.A.2 with } k' := k-1)\\
&\leq \sum_{k=2m+1}^{\infty} \left|\sum_{\alpha\in[d]^k} \boldsymbol{y}^\alpha \cdot \partial_\alpha f(\mathbf{0})\right| \frac{1}{\sqrt{2}} \left(\frac{em}{k}\right)^k.\\
&&& \text{(using } 3\sqrt{2/\pi}e^{-\frac{7m}{6}} \leq 1)
\end{aligned}$$

If we take a uniformly random $\boldsymbol{y} \in r \cdot G_d^{(n)}$, then $\boldsymbol{y}$ has coordinates symmetrically distributed around zero, therefore by Lemma 4.5.8 (choosing $t := 4$) we know that for all $k \in \mathbb{N}_+$ the fraction of $\boldsymbol{y}$ vectors such that

$$\left|\sum_{\alpha\in[d]^k} \frac{\boldsymbol{y}^\alpha}{r^k} \cdot \frac{\partial_\alpha f(\mathbf{0})}{c^k k^{\frac{k}{2}}}\right| \geq \sqrt{2}\left(4\sqrt{\frac{dk}{2}}\right)^k \tag{4.42}$$

is at most 4^{-2k}. Since $\sum_{k=2m+1}^{\infty} 4^{-2k} \leq \sum_{k=3}^{\infty} 4^{-2k} < 1/1000$, it follows that apart from a 1/1000-th fraction of the $\boldsymbol{y}$ vectors, the other $\boldsymbol{y}$s satisfy the following:

$$\begin{aligned}
\left|\nabla f(\mathbf{0})\boldsymbol{y} - f_{(2m)}(\boldsymbol{y})\right| &\leq \sum_{k=2m+1}^{\infty}\left|\sum_{\alpha\in[d]^k}\boldsymbol{y}^{\alpha}\cdot\partial_{\alpha}f(\mathbf{0})\right|\frac{1}{\sqrt{2}}\left(\frac{em}{k}\right)^k\\
&\leq \sum_{k=2m+1}^{\infty}\sqrt{2}\left(4\sqrt{\frac{dk}{2}}\right)^k r^k c^k k^{\frac{k}{2}}\frac{1}{\sqrt{2}}\left(\frac{em}{k}\right)^k \qquad \text{(using Eq. (4.42))}\\
&= \sum_{k=2m+1}^{\infty}\left(\frac{4\sqrt{d}rcem}{\sqrt{2}}\right)^k\\
&< \sum_{k=2m+1}^{\infty}\left(8rcm\sqrt{d}\right)^k. \qquad \text{(using } 4e < 8\sqrt{2}\text{)}
\end{aligned}$$

□

Chapter 5

Convex optimization using quantum oracles

In this chapter we study to what extent quantum algorithms can speed up solving convex optimization problems. Following the classical literature we assume access to a convex set via various oracles, and we examine the efficiency of reductions between the different oracles. In particular, we show how a separation oracle can be implemented using $\widetilde{\mathcal{O}}(1)$ quantum queries to a membership oracle, which is an exponential quantum speed-up over the $\Omega(n)$ membership queries that are needed classically. We show that a quantum computer can very efficiently compute an approximate subgradient of a convex Lipschitz function. Combining this with a simplification of recent classical work of Lee, Sidford, and Vempala [LSV18] gives our efficient separation oracle. This in turn implies, via a known algorithm, that $\widetilde{\mathcal{O}}(n)$ quantum queries to a membership oracle suffice to implement an optimization oracle (the best known classical upper bound on the number of membership queries is quadratic). We also prove several lower bounds: $\Omega(\sqrt{n})$ quantum separation (or membership) queries are needed for optimization if the algorithm knows an interior point of the convex set, and $\Omega(n)$ quantum separation queries are needed if it does not.

5.1 Introduction

Optimization is a fundamental problem in mathematics and computer science, with many real-world applications. As people try to solve larger and larger optimization problems, the efficiency of optimization becomes more and more important, motivating us to find the best possible algorithms. Recent experimental progress on building quantum computers draws attention to new approaches to the problem: can we solve optimization problems more efficiently by exploiting quantum effects such as superposition, interference, and entanglement? For many

This chapter is based on [vAGGdW18].

discrete optimization problems [Gro96, DH96, Szeg04, DHHM06, AŠ06] significant speed-ups have been shown, but less is known about *continuous* optimization problems.

One of the most successful continuous optimization paradigms is *convex* optimization, which optimizes a convex function over a convex set that is given explicitly (by a set of constraints) or implicitly (by an oracle). See Bubeck [Bub15] for a recent survey. Quantum algorithms for convex optimization have been considered before. In 2008, Jordan [Jor08] described a faster quantum algorithm for minimizing quadratic functions. Recently, for an important class of convex optimization problems (semidefinite optimization) quantum speed-ups were achieved using algorithms whose runtime scales polynomially with the desired precision and some geometric parameters [BS17, vAGGdW17, BKL$^+$17b, vAG19]. However, many convex optimization problems can be solved classically using algorithms whose runtime scales *logarithmically* with the desired precision and the relevant geometric parameters. We are aware of only one quantum speed-up which is partially in this regime, namely the very recent quantum interior point method of Kerenidis and Prakash [KP18]. In this chapter we look at general convex optimization problems, considering algorithms that have such favorable logarithmic scaling with the precision.

The generic problem in convex optimization is minimizing a convex function $f : K \to \mathbb{R} \cup \{\infty\}$, where $K \subseteq \mathbb{R}^n$ is a convex set. We consider the setting where an interior point $x_0 \in \text{int}(K)$ is given and radii $r, R > 0$ are known such that $B(x_0, r) \subseteq K \subseteq B(x_0, R)$, where $B(x_0, r)$ is the Euclidean ball of radius r centered at x_0.

It is well-known that if the convex function is bounded on K, then we can equivalently consider the problem of minimizing a *linear* function over a different convex set $K' \subseteq \mathbb{R}^{n+1}$, namely the epigraph $K' = \{(x, \mu) : x \in K, f(x) \geq \mu\}$ of f. Accessing K' is easy given access to K and f, and the parameters involved will be similar. Conversely, for any linear optimization problem over an unknown convex set K, there is an equivalent optimization problem over a known convex set (say, the ball), with an unknown bounded convex objective function f that can be evaluated easily given access to K. From now on we therefore focus on optimizing a known linear function over an unknown convex set.

We consider the setting where access to the convex set is given only in a black-box manner, through an oracle. The five basic problems (oracles) in convex optimization identified by Grötschel, Lovász, and Schrijver [GLS88] are: membership, separation, optimization, violation, and validity (see Section 5.2 for the definitions). They showed that all five basic problems are polynomial-time equivalent. That is, given an oracle O for one of these problems, one can implement an oracle for any of the other problems using a polynomial number of calls to O and polynomially many other elementary operations. Subsequent work made these polynomial-time reductions more efficient, reducing the degree of the polynomials. Recently Lee et al. [LSV18], in the classical setting, showed that with $\widetilde{\mathcal{O}}(n^2)$

calls[1] to a membership oracle (and $\widetilde{\mathcal{O}}(n^3)$ other elementary arithmetic operations) one can solve an optimization problem. They did so by showing that $\widetilde{\mathcal{O}}(n)$ calls to a membership oracle suffice to do separation, and then composing this with the known fact [LSW15] (see also [LSV18, Theorem 15]) that $\widetilde{\mathcal{O}}(n)$ calls to a separation oracle suffice for optimization.

Our main result (Section 5.4) shows that on a quantum computer, $\widetilde{\mathcal{O}}(1)$ calls to a membership oracle suffice to implement a separation oracle, and hence (by the known classical reduction from optimization to separation) $\widetilde{\mathcal{O}}(n)$ calls to a membership oracle suffice for optimization.[2] Lee et al. [LSV18] use a geometric idea to reduce separation to finding an approximate subgradient of a convex Lipschitz function. They then show that $\widetilde{\mathcal{O}}(n)$ evaluations of a convex Lipschitz function suffice to get an approximate subgradient. Our contributions here are twofold (Section 5.3 and 5.4). We slightly simplify the reduction of Lee et al. [LSV18] from separation to finding an approximate subgradient of a convex Lipschitz function: in contrast to theirs, our argument directly analyzes convex Lipschitz functions without smoothing the function. More importantly, we give a simplified algorithm for computing such an approximate subgradient that recovers the result of [LSV18], and that is suitable for a quantum speed-up using known quantum algorithms for computing approximate (sub)gradients [Jor05], see Chapter 4.

As a second set of results, in Section 5.5 we provide lower bounds on the number of membership or separation queries needed to implement several other oracles. We show that our quantum reduction from separation to membership indeed improves over the best possible classical reduction: $\Omega(n)$ classical membership queries are needed to do separation.[3] We only have partial results regarding the optimality of the reduction from optimization to separation. In the setting where we are not given an interior point of the set K, we can prove an essentially optimal $\Omega(n)$ lower bound on the number of quantum queries to a separation oracle needed to do optimization. However, for the case of quantum algorithms that *do* know an interior point, we are only able to prove an $\Omega(\sqrt{n})$ lower bound. In the classical setting, regardless of whether or not we know an interior point, the reduction uses $\widetilde{\Theta}(n)$ queries. This raises the interesting question of whether knowing an interior point can lead to a better quantum algorithm. We therefore view closing the gap between upper and lower bound as an important direction for future work.

[1]Here, and in the rest of the chapter, the notation $\widetilde{\mathcal{O}}(\cdot)$ is used to hide polylogarithmic factors in n, r, R, ϵ.

[2]Although not stated explicitly in our results, we also use $\widetilde{\mathcal{O}}(n^3)$ additional operations for optimization using membership, like [LSV18]. This is because our quantum algorithm for separation uses only $\widetilde{\mathcal{O}}(n)$ gates in addition to the $\widetilde{\mathcal{O}}(1)$ membership queries, and we use the same reduction from optimization to separation as [LSV18].

[3]We are not aware of an existing proof of this classical lower bound, but it may well be somewhere in the vast literature on convex optimization.

Finally, we briefly mention (Section 5.6) how to obtain upper and lower bounds for some of the other oracle reductions, using a convex polarity argument. As we show, in the setting where we are given an interior point, the relation between membership and separation is analogous to the relation between validity and optimization. In particular, our better quantum algorithm for separation using membership queries implies that on a quantum computer $\widetilde{\mathcal{O}}(1)$ queries to a validity oracle suffice to implement an optimization oracle. That is, on a quantum computer, finding the optimal value is equivalent to finding an optimizer. Also, the same polarity argument shows that algorithms for optimization using separation are essentially equivalent to algorithms for separation using optimization. In particular, this turns our lower bound on the number of separation queries needed to implement an optimization oracle into a lower bound on the reverse direction.

Classical:

$$\begin{tikzcd}
\boxed{\mathrm{MEM}(K)} \arrow[r, shift right, "\widetilde{\Theta}(n)"'] & \boxed{\mathrm{SEP}(K)} \arrow[l, shift right, "\Theta(1)"'] \arrow[r, shift right, "\widetilde{\Theta}(n)"'] & \boxed{\mathrm{OPT}(K)} \arrow[l, shift right, "\widetilde{\Theta}(n)"'] \arrow[r, shift right, "\Theta(1)"'] & \boxed{\mathrm{VAL}(K)} \arrow[l, shift right, "\widetilde{\Theta}(n)"']
\end{tikzcd}$$

Quantum:

$$\begin{tikzcd}
\boxed{\mathrm{MEM}(K)} \arrow[r, shift right, "\boldsymbol{\widetilde{\Theta}(1)}"'] & \boxed{\mathrm{SEP}(K)} \arrow[l, shift right, "\Theta(1)"'] \arrow[r, shift right, "\widetilde{\mathcal{O}}(n)\ \boldsymbol{\Omega(n)^*}"'] & \boxed{\mathrm{OPT}(K)} \arrow[l, shift right, "\widetilde{\mathcal{O}}(n)\ \boldsymbol{\Omega(\sqrt{n})}"'] \arrow[r, shift right, "\Theta(1)"'] & \boxed{\mathrm{VAL}(K)} \arrow[l, shift right, "\boldsymbol{\widetilde{\Theta}(1)}"']
\end{tikzcd}$$

Figure 5.1. The top and bottom diagram illustrate the relations between the basic (weak) oracles for respectively classical and quantum queries, with boldface entries marking our new results. All upper and lower bounds hold in the setting where we know an interior point of K, except the $*$-marked $\Omega(n)$ lower bound on the number of separation queries needed for optimization. Notice the central symmetry of the diagrams, which is a consequence of polarity.

Figure 5.1 gives an informal presentation of our results; the upper bounds arise from oracle reductions, the (change in) accuracy is ignored here for simplicity. The above-mentioned polarity manifests itself in the central symmetry of the figure.

Related independent work. In independent simultaneous work, Chakrabarti, Childs, Li, and Wu [CCLW18] discovered a similar upper bound as ours: combining the recent classical work of Lee et al. [LSV18] with a quantum algorithm for computing gradients, they show how to implement an optimization oracle via $\widetilde{\mathcal{O}}(n)$ quantum queries to a membership oracle and to an oracle for the objective function. Their proof stays quite close to [LSV18] while ours first simplifies some of the technical lemmas of [LSV18], giving us a slightly simpler presentation and a better error-dependence of the resulting algorithm.

5.2 Preliminaries

For $p \geq 1$, $\epsilon \geq 0$, and a set $C \subseteq \mathbb{R}^n$ we let

$$B_p(C, \epsilon) = \{x \in \mathbb{R}^n : \exists y \in C \text{ such that } ||x - y||_p \leq \epsilon\}$$

be the set of points of distance at most ϵ from C in the ℓ_p-norm. When $C = \{x\}$ is a singleton set we abuse notation and write $B_p(x, \epsilon)$. We overload notation by setting

$$B_p(C, -\epsilon) = \{x \in \mathbb{R}^n : B_p(x, \epsilon) \subseteq C\}.$$

Whenever p is omitted it is assumed that $p = 2$.

Recall that a function $f : C \to \mathbb{R}$ is *Lipschitz* if there exists a constant $L > 0$ such that

$$|f(y') - f(y)| \leq L\|y' - y\|_2 \text{ for all } y, y' \in C.$$

We write that f is *L-Lipschitz*. The inner product between vectors $v, w \in \mathbb{R}^n$ is $\langle v, w\rangle = v^T w$.

5.2.1. Definition (Subgradient). Let $C \subseteq \mathbb{R}^n$ be convex and let x be an element of the interior of C. For a convex function $f : C \to \mathbb{R}$ we denote by $\underline{\partial} f(x)$ the set of subgradients of f at x, i.e., those vectors g satisfying

$$f(y) \geq f(x) + \langle g, y - x\rangle \text{ for all } y \in C.$$

Note that in the above definition $\underline{\partial} f(x) \neq \emptyset$ due to convexity.

If $f : C \to \mathbb{R}$ is L-Lipschitz, then for any x in the interior of C and any $g \in \underline{\partial} f(x)$ we have $\|g\| \leq L$, as follows. Consider a $y \in C$ such that $y - x = \alpha g$ for some $\alpha > 0$. Then since g is a subgradient of f at x we have

$$\alpha\|g\|^2 = \langle g, y - x\rangle \leq f(y) - f(x) \leq L\|y - x\| = \alpha L\|g\|, \tag{5.1}$$

and therefore $\|g\| \leq L$.

Recall that a standard quantum oracle corresponds to a unitary transformation that acts on two registers, where the first register contains the query and the answer is added to the second register. For example, a function evaluation oracle for $f\colon X \to Y$ would map $|x, 0\rangle$ to $|x, f(x)\rangle$, where $|x\rangle$ and $|f(x)\rangle$ are basis states corresponding to binary representations of x and $f(x)$ respectively. Unlike classical algorithms, quantum computers can apply such an oracle to a *superposition* of different y's. They are also allowed to apply the inverse of a unitary oracle.

The standard quantum oracle described above models problems where there is a single correct answer to a query. When there are multiple good answers (for instance, different good approximations to the correct value) and the oracle is only required to give a correct answer with high probability, then we will work with the more liberal notion of *relational* quantum oracles.

5.2.2. DEFINITION (Relational quantum oracle). Let $\mathcal{F}\colon X \to \mathcal{P}(Y)$ be a function, such that for each $x \in X$ the subset $\mathcal{F}(x) \subseteq Y$ is the set of valid answers to an x query. A relational quantum oracle for $\mathcal{F}$ which answers queries with success probability $\geq 1-\rho$, is a unitary that for all $x \in X$ maps

$$U\colon |x,0,0\rangle \mapsto \sum_{y\in Y} \alpha_{x,y}|x,y,\psi_{x,y}\rangle,$$

where $|\psi_{x,y}\rangle$ denotes some normalized quantum state and $\sum_{y\in\mathcal{F}(x)} |\alpha_{x,y}|^2 \geq 1-\rho$. Thus measuring the second register of $U|x,0,0\rangle$ gives a valid answer to the x query with probability at least $1-\rho$.

This definition is very natural for cases where the oracle is implemented by a quantum algorithm that produces a valid answer with probability $\geq 1-\rho$.

5.2.1 Oracles for convex sets

We consider the following five basic oracles for a convex set K (cf. [GLS88]).

5.2.3. DEFINITION (Membership oracle $\mathrm{MEM}_{\epsilon,\rho}(K)$). Queried with a vector $y \in \mathbb{R}^n$, the oracle, with success probability $\geq 1-\rho$, correctly asserts one of the following

- $y \in B(K,\epsilon)$, or
- $y \notin B(K,-\epsilon)$.

5.2.4. DEFINITION (Separation oracle $\mathrm{SEP}_{\epsilon,\rho}(K)$). Queried with a vector $y \in \mathbb{R}^n$, the oracle, with success probability at least $\geq 1-\rho$, correctly asserts one of the following

- $y \in B(K,\epsilon)$, or
- $y \notin B(K,-\epsilon)$,

and in the second case it returns a unit vector $g \in \mathbb{R}^n$ such that $\langle g,x\rangle \leq \langle g,y\rangle + \epsilon$ for all $x \in B(K,-\epsilon)$.

5.2.5. DEFINITION (Optimization oracle $\mathrm{OPT}_{\epsilon,\rho}(K)$). Queried with a unit vector $c \in \mathbb{R}^n$, the oracle, with probability $\geq 1-\rho$, does one of the following:

- it returns a vector $y \in \mathbb{R}^n$ such that $y \in B(K,\epsilon)$ and $\langle c,x\rangle \leq \langle c,y\rangle + \epsilon$ for all $x \in B(K,-\epsilon)$,

- or it correctly asserts that $B(K, -\epsilon)$ is empty.

Note that the above optimization oracle corresponds to *maximizing* a linear function over a convex set; we could equally well state it for minimization.

5.2.6. Definition (Violation oracle $\mathrm{VIOL}_{\epsilon,\rho}(K)$). Queried with a unit vector $c \in \mathbb{R}^n$ and a real number γ, the oracle, with probability $\geq 1 - \rho$, does one of the following:

- it asserts that $\langle c, x\rangle \leq \gamma + \epsilon$ for all $x \in B(K, -\epsilon)$,
- or it finds a vector $y \in B(K, \epsilon)$ such that $\langle c, y\rangle \geq \gamma - \epsilon$.

5.2.7. Definition (Validity oracle $\mathrm{VAL}_{\epsilon,\rho}(K)$). Queried with a unit vector $c \in \mathbb{R}^n$ and a real number γ, the oracle, with probability $\geq 1 - \rho$, does one of the following:

- it asserts that $\langle c, x\rangle \leq \gamma + \epsilon$ for all $x \in B(K, -\epsilon)$,
- or it asserts that $\langle c, y\rangle \geq \gamma - \epsilon$ for some $y \in B(K, \epsilon)$.

If in the above definitions both ε and ρ are equal to 0, then we call the oracle *strong*. If either is non-zero then we sometimes call it *weak*.

When we discuss membership queries, we will always assume that we are given a small ball which lies inside the convex set. It is easy to see that without such a small ball one cannot obtain an optimization oracle using only poly(n) classical queries to a membership oracle (see, e.g., [GLS88, Sec. 4.1] or the example below). As the following example shows, the same holds for quantum queries. We will use a reduction from a version of the well-studied *search* problem:

Given $z \in \{0,1\}^N$ such that $|z| = 1$, find $b \in [N]$ such that $z_b = 1$.

It is not hard to see that if the access to z is given via classical queries $i \mapsto z_i$, then $\Omega(N)$ queries are needed. It is well known [BBBV97] that if we allow quantum queries, i.e., applications of the unitary $|i\rangle|b\rangle \mapsto |i\rangle|z_i \oplus b\rangle$, then $\Omega(\sqrt{N})$ queries are needed. Now let $N = 2^n$ and consider an input $z \in \{0,1\}^N$ to the search problem. Let $b \in \{0,1\}^n$ be the index such that $z_b = 1$. Consider maximizing the linear function $\langle e, z\rangle$ (where e is the all-1 vector) over the set $K_z = \bigtimes_{i=1}^n [b_i - 1/2, b_i]$. Clearly the optimal solution to this convex optimization problem, even with a small constant additive error in the answer, gives the solution to the search problem. However, a membership query is essentially equivalent to querying a bit of z and therefore $\Omega(\sqrt{N}) = \Omega(2^{n/2})$ quantum queries to the membership oracle are needed for optimization.

5.3 Computing approximate subgradients of convex Lipschitz functions

Here we show how to compute an approximate subgradient (at 0) of a convex Lipschitz function. That is, given a convex set C such that $0 \in \mathrm{int}(C)$ and a convex function $f : C \to \mathbb{R}$, we show how to compute a vector $\tilde{g} \in \mathbb{R}^n$ such that $f(y) \geq f(0) + \langle \tilde{g}, y \rangle - a\|y\| - b$ for some real numbers $a, b > 0$ that will be defined later (see Lemma 5.3.5 and Lemma 5.3.9). The idea of the classical algorithm given in the next section is to pick a point $z \in B_\infty(0, r_1)$ uniformly at random and use the finite difference $\nabla^{(r_2)} f(z)$ (defined below) as an approximate subgradient of f at 0; the radii r_1 and r_2 need to be chosen small to make the approximation good. This results in a slightly simplified version of the algorithm of Lee et al. [LSV18]. In Section 5.3.2 we show how to improve on this classical algorithm on a quantum computer.

5.3.1 Classical approach

5.3.1. Definition (Finite difference gradient approximations). For a function $f\colon C \to \mathbb{R}$, and a point $x \in \mathbb{R}^n$ such that $B_1(x, r) \subseteq C$, and $i \in [n]$, we define $\nabla_i^{(r)} f(x) := \frac{f(x+re_i)-f(x-re_i)}{2r}$, where $e_i \in \{0,1\}^n$ is the vector that has a 1 only in its ith coordinate. Similarly we define

$$\nabla^{(r)} f(x) := \Big(\nabla_1^{(r)} f(x), \nabla_2^{(r)} f(x), \ldots, \nabla_n^{(r)} f(x)\Big).$$

5.3.2. Definition (Finite difference Laplace approximation). For a function $f : C \to \mathbb{R}$, and a point $x \in \mathbb{R}^n$ such that $B_1(x, r) \subseteq C$, and $i \in [n]$, we define $\Delta_i^{(r)} f(x) := \frac{f(x+re_i)-2f(x)+f(x-re_i)}{r^2}$. Similarly

$$\Delta^{(r)} f(x) := \sum_{i=1}^{n} \Delta_i^{(r)} f(x).$$

Note that for a convex function we have $\Delta_i^{(r)} f(x) \geq 0$ for all x such that $B_1(x, r) \subseteq C$.

The next two lemmas will be needed in the proof of the main result of this section, Lemma 5.3.5. In Lemma 5.3.3 we give an upper bound on the deviation $\left\|g - \nabla^{(r_2)} f(z)\right\|_1$ of a finite difference gradient approximation $\nabla^{(r_2)} f(z)$ from an actual subgradient g at the point z, in terms of the finite difference Laplace approximation $\Delta^{(r_2)} f(z)$. Then, in Lemma 5.3.4 we show that in expectation, the finite difference Laplace approximation is small. Together with Markov's inequality this gives us good control over the quality of a finite difference gradient approximation.

5.3.3. LEMMA. *If* $r_2 > 0$, $z \in \mathbb{R}^n$, *and* $f : B_1(z, r_2) \to \mathbb{R}$ *is convex, then*

$$\sup_{g \in \underline{\partial} f(z)} \left\| g - \nabla^{(r_2)} f(z) \right\|_1 \le \frac{r_2 \Delta^{(r_2)} f(z)}{2}.$$

Proof:
Fix a $g \in \underline{\partial} f(z)$. For every $i \in [n]$, we have $f(z + r_2 e_i) \ge f(z) + \langle g, r_2 e_i \rangle = f(z) + r_2 g_i$, and, similarly, $f(z - r_2 e_i) \ge f(z) - r_2 g_i$. Rearranging gives

$$\underbrace{\frac{f(z) - f(z - r_2 e_i)}{r_2}}_{:=A} \le g_i \le \underbrace{\frac{f(z + r_2 e_i) - f(z)}{r_2}}_{:=B}.$$

Note that $|g_i - \frac{A+B}{2}| \le \frac{B-A}{2}$ for any three real numbers $A \le g_i \le B$. Moreover, $\frac{A+B}{2} = \nabla_i^{(r_2)} f(z)$ and $B - A = r_2 \Delta_i^{(r_2)} f(z)$, thus $\left| g_i - \nabla_i^{(r_2)} f(z) \right| \le \frac{r_2 \Delta_i^{(r_2)} f(z)}{2}$. Now we can finish the proof by summing this inequality over all $i \in [n]$. □

5.3.4. LEMMA. *If* $0 < r_2 \le r_1$, *and* $f : B_\infty(x, r_1 + r_2) \to \mathbb{R}$ *is convex and L-Lipschitz, then*

$$\mathop{\mathbb{E}}_{z \in B_\infty(x, r_1)} \Delta^{(r_2)} f(z) \le \frac{nL}{r_1}.$$

Proof:
Below we show that $\mathop{\mathbb{E}}_{z \in B_\infty(x, r_1)} \Delta_i^{(r_2)} f(z) \le \frac{L}{r_1}$ for all $i \in [n]$, and then sum over i.

$$\begin{aligned}
&\mathop{\mathbb{E}}_{z \in B_\infty(x, r_1)} \Delta_i^{(r_2)} f(z) = \\
&= \frac{1}{(2r_1)^n} \int_{z \in B_\infty(x, r_1)} \frac{f(z + r_2 e_i) - 2f(z) + f(z - r_2 e_i)}{r_2^2} \, dz \\
&= \frac{1}{(2r_1)^n} \int_{\substack{z_j \in [x_j - r_1, x_j + r_1], \\ j \in [n], j \ne i}} \int_{z_i \in [x_i - r_1, x_i + r_1]} \frac{f(z + r_2 e_i) - 2f(z) + f(z - r_2 e_i)}{r_2^2} \, dz \\
&= \frac{1}{(2r_1)^n} \int_{\substack{z_j \in [x_j - r_1, x_j + r_1], \\ j \in [n], j \ne i}} \Big(\int_{z_i \in [x_i - r_1, x_i - r_1 + r_2]} \frac{f(z + r_2 e_i) - f(z)}{r_2^2} \, dz \\
&\qquad + \int_{z_i \in [x_i + r_1 - r_2, x_i + r_1]} \frac{-f(z) + f(z - r_2 e_i)}{r_2^2} \, dz \Big) \\
&\le \frac{1}{(2r_1)^n} \int_{\substack{z_j \in [x_j - r_1, x_j + r_1], \\ j \in [n], j \ne i}} 2L \, dz \quad = \frac{L}{r_1}.
\end{aligned}$$

□

Note that the above lemma is stated and proved for continuous random variables, but the same proof holds if we have a uniform hypergrid over the same hypercube, providing a discrete version of the above result. In the discrete case, in order to get the same cancellations we need to assume that both r_1 and r_2 are integer multiples of the grid spacing.

We are now ready to prove the main result of this section. Informally, the next lemma proves that an approximate subgradient of a convex Lipschitz function f at 0 can be obtained by an algorithm that outputs $\nabla^{(r_2)}\tilde{f}(z)$ for a random z close enough to 0, where $\tilde{f}$ is an approximate version of f. In other words, this lemma gives us a classical algorithm to compute an approximate subgradient of f using $2n$ classical queries to an approximate version of f.

5.3.5. LEMMA. *Let $r_1 > 0$, $L > 0$, $\rho \in (0,1/3]$, $\delta \in (0, r_1\sqrt{n}L/\rho]$, then $r_2 := \sqrt{\frac{\delta r_1 \rho}{\sqrt{n}L}} \leq r_1$. Suppose $f : C \to \mathbb{R}$ is a convex function that is L-Lipschitz on $B_\infty(0, 2r_1)$, and $\tilde{f} : B_\infty(0, 2r_1) \to \mathbb{R}$ is such that $\left\|\tilde{f} - f\right\|_\infty \leq \delta$. Then for a uniformly random $z \in B_\infty(0, r_1)$, with probability at least $1 - \rho$*

$$f(y) \geq f(0) + \langle \nabla^{(r_2)}\tilde{f}(z), y\rangle - \frac{3n^{\frac{3}{4}}}{2}\sqrt{\frac{\delta L}{\rho r_1}}\|y\| - 2L\sqrt{n}r_1 \qquad \text{for all } y \in C.$$

Proof:
Let $z \in B_\infty(0, r_1)$ and $g \in \underline{\partial} f(z)$. Recall $\|g\| \leq L$ by Equation (5.1). Then for all $y \in C$

$$\begin{aligned}
f(y) &\geq f(z) + \langle g, y - z\rangle \\
&= f(z) + \langle g, y - z\rangle + \left(\langle \nabla^{(r_2)} f(z), y\rangle - \langle \nabla^{(r_2)} f(z), y\rangle\right) + (f(0) - f(0)) \\
&= f(0) + \langle \nabla^{(r_2)} f(z), y\rangle + \langle g - \nabla^{(r_2)} f(z), y\rangle + (f(z) - f(0)) + \langle g, -z\rangle \\
&\geq f(0) + \langle \nabla^{(r_2)} f(z), y\rangle - \left\|g - \nabla^{(r_2)} f(z)\right\|_1 \|y\|_\infty - L\|z\| - \|g\|\|z\| \\
&\geq f(0) + \langle \nabla^{(r_2)} f(z), y\rangle - \left\|g - \nabla^{(r_2)} f(z)\right\|_1 \|y\|_\infty - L\sqrt{n}r_1 - L\sqrt{n}r_1 \\
&\geq f(0) + \langle \nabla^{(r_2)} \tilde{f}(z), y\rangle - \frac{\delta\sqrt{n}}{r_2}\|y\| - \left\|g - \nabla^{(r_2)} f(z)\right\|_1 \|y\|_\infty - 2L\sqrt{n}r_1.
\end{aligned}$$

Note that in the last line we switched from f to $\tilde{f}$, using that $\nabla^{(r_2)} f(z)$ and $\nabla^{(r_2)}\tilde{f}(z)$ differ by at most δ/r_2 in each coordinate. Our choice of r_2 gives $\frac{\delta\sqrt{n}}{r_2} = n^{\frac{3}{4}}\sqrt{\frac{\delta L}{\rho r_1}}$ and by Lemma 5.3.3–5.3.4 we have

$$\mathop{\mathbb{E}}_{z \in B_\infty(x, r_1)} \left\|g - \nabla^{(r_2)} f(z)\right\|_1 \leq \frac{nLr_2}{2r_1} = \frac{n^{\frac{3}{4}}}{2}\sqrt{\frac{\delta L \rho}{r_1}}.$$

By Markov's inequality we get that $\left\|g - \nabla^{(r_2)} f(z)\right\|_1 \leq \frac{n^{\frac{3}{4}}}{2}\sqrt{\frac{\delta L}{\rho r_1}}$ with probability $\geq 1 - \rho$ over the choice of z. Plugging this bound on $\left\|g - \nabla^{(r_2)} f(z)\right\|_1$ into the above lower bound on $f(y)$ concludes the proof of the lemma. □

5.3.2 Quantum improvements

In this section we show how to improve subgradient computation of convex functions via Jordan's quantum algorithm for gradient computation [Jor05]. We use the results of the previous chapter, in particular Lemma 4.5.4.

We show that Jordan's Algorithm 4.1 allows us to compute an approximate subgradient of a function f, even if we are only given standard oracle access to a function $\tilde{f}$ which is sufficiently close to f. In particular, we will assume we are given access to a standard unitary oracle of a function $\tilde{f}\colon G_m^n \to \mathbb{R}$ which satisfies $|\tilde{f}(x) - f(x)| \leq \delta$ for all $x \in G_m^n$. That is, we assume we are given access to a unitary U acting as

$$U : |x\rangle|0\rangle \mapsto |x\rangle|\tilde{f}(x)\rangle \tag{5.2}$$

Note that if we can classically efficiently evaluate $\tilde{f}$, then it is well known that we can construct such a unitary as a small quantum circuit (see [NC00, Sec. 1.4.1]).

The main idea is that, using one application of U, a phase gate corresponding to the output register, and another application of $U^\dagger$ to uncompute the function value, we can implement a phase oracle for $\tilde{f}$. Moreover, Equation (5.3) below will also hold for $\tilde{f}$, with a slightly worse right-hand side, since f is close to $\tilde{f}$. The following corollary is analogous to Theorem 4.5.5; we present it together with a proof sketch, in order get a statement better fitting the setting of this chapter.

5.3.6. Corollary (Gradient computation by approximate function evaluation). *Let $\delta, B, r, c \in \mathbb{R}$, $\rho \in (0, 1/3]$. Let $x_0, g \in \mathbb{R}^n$ with $\|g\|_\infty \leq \frac{B}{r}$. Let $m := \lceil \log_2\big(\frac{B}{28\pi\delta}\big)\rceil$ and suppose $f : (x_0 + rG_m^n) \to \mathbb{R}$ is such that*

$$|f(x_0 + rx) - \langle g, rx\rangle - c| \leq \delta \tag{5.3}$$

for 99.9% of the points $x \in G_m^n$, and we have access to a standard unitary oracle U, providing $\mathcal{O}\big(\log\big(\frac{B}{\delta}\big)\big)$-bit fixed-point binary approximations $\tilde{f}(z)$ s.t. $|\tilde{f}(z) - f(z)| \leq \delta$ for all $z \in (x_0 + rG_m^n)$. Then we can compute a vector $\tilde{g} \in \mathbb{R}^n$ such that

$$\Pr\left[\ \|\tilde{g} - g\|_\infty > \frac{8 \cdot 42\pi\delta}{r}\right] \leq \rho,$$

with $\mathcal{O}\Big(\log\big(\frac{n}{\rho}\big)\Big)$ queries to U and $U^\dagger$ and $\mathcal{O}\Big(n\log\big(\frac{n}{\rho}\big)\log\big(\frac{B}{\delta}\big)\operatorname{loglog}\big(\frac{n}{\rho}\big)\operatorname{loglog}\big(\frac{B}{\delta}\big)\Big)$ additional gate complexity.

Proof:
As described above the corollary, we first implement a phase oracle for $\tilde{f}$ and then we apply Jordan's gradient computation algorithm (Lemma 4.5.4).

With a single query to U and its inverse we can implement a phase oracle O that acts as $\mathrm{O} : |x\rangle \mapsto e^{2\pi i \frac{M}{3B}\tilde{f}(x_0+rx)}|x\rangle$, where $M := \frac{3B}{84\pi\delta}$, and[4] $m := \log_2(M)$.

[4]We can assume without loss of generality that the upper bound B is such that M is a power of two.

Let $h(x) := \frac{\tilde{f}(x_0+rx)}{3B}$, then by (5.3) 99.9% of the points $x \in G_m^n$ satisfy $\left|h(x) - \langle \frac{r}{3B}g, x\rangle - \frac{c}{3B}\right| \leq \frac{2\delta}{3B} = \frac{1}{42\pi M}$. Since $\left\|\frac{r}{3B}g\right\|_\infty \leq \frac{1}{3}$, by Lemma 4.5.4 we can compute a vector $v \in \mathbb{R}^n$ which is a coordinatewise $\frac{4}{M}$-approximator of $\frac{r}{3B}g$: for each $i \in [n]$ we have $\left|g_i - \frac{3B}{r}v_i\right| \leq \frac{12B}{rM} = \frac{8\cdot 42\pi\delta}{r}$ with probability at least $\frac{2}{3}$.

Note that the above success probability is per coordinate of g. However, repeating the whole procedure $\mathcal{O}\big(\log(\frac{n}{\rho})\big)$ times and taking the median of the resulting vectors coordinatewise gives a gradient approximator $\tilde{g}$ with the desired approximation quality with probability at least $1-\rho$. For more details on the proof of the gate complexity see[5] Theorem 4.5.5. □

Remark. With essentially the same approach, the above corollary of Jordan's quantum gradient computation algorithm can also be proven in the setting where our access to an approximation of f is not given by a standard quantum oracle but by a *relational* quantum oracle, see Appendix 5.A for both the definition of this type of approximation to f and a proof of this corollary.

In terms of applications, we want to point out that if the membership oracle used in Section 5.4 comes from a deterministic algorithm, then we get a standard quantum oracle. Only when the membership oracle itself is relational (for example, when it is itself computed by a bounded-error quantum algorithm) do we need the more general setting of Appendix 5.A.

In order to apply the above corollary, we need to find some function which is sufficiently close to linear. Fortunately, convex Lipschitz functions can be very well approximated by linear functions over most small-enough regions. Similarly to the classical case (Lemma 5.3.5) we make this claim quantitative using Lemma 5.3.4. In order to apply the more efficient quantum gradient computation of Corollary 5.3.6 we also need the following two lemmas to ensure that Equation (5.3) holds.

5.3.7. Lemma. *Let $S \subseteq \mathbb{R}^n$ be such that $S = -S$, and let* $\mathrm{conv}(S)$ *denote the convex hull of S. If $f : \mathrm{conv}(S) \to \mathbb{R}$ is a convex function, $f(0) = 0$, and $|f(s)| \leq \delta$ for all $s \in S$, then*

$$|f(s')| \leq \delta \textit{ for all } s' \in \mathrm{conv}(S).$$

Proof:
Since f is convex and $f(s) \leq \delta$ for all $s \in S$ we immediately get that $f(s') \leq \delta$ for all $s' \in \mathrm{conv}(S)$. Because $f(0) = 0$ and $S = -S$, due to convexity we get that $f(s') \geq -f(-s') \geq -\delta$. □

[5]The correspondence with the parametrization of Theorem 4.5.5 is $\varepsilon \leftrightarrow \frac{8\cdot 42\pi\delta}{r}$, $M \leftrightarrow \frac{B}{r}$.

5.3.8. LEMMA. *If $r_2 > 0$, $z \in \mathbb{R}^n$ and $f : B_1(z, r_2) \to \mathbb{R}$ is convex, then*

$$\sup_{y\in B_1(0,r_2)} \left|f(z+y) - f(z) - \langle y, \nabla^{(r_2)} f(z)\rangle\right| \leq \frac{r_2^2 \Delta^{(r_2)} f(z)}{2}.$$

Proof:
Let $d(y) := f(z+y) - f(z) - \langle y, \nabla^{(r_2)} f(z)\rangle$ be the difference between $f(z+y)$ and its linear approximator. Let $S := \{\pm r_2 e_i : i \in [n]\}$. It is easy to see that $d(0) = 0$, $S = -S$, and $\mathrm{conv}(S) = B_1(0, r_2)$. Also, for all $s \in S$ we have $|d(s)| \leq r_2^2 \Delta^{(r_2)} f(z)/2$:

$$\begin{aligned} d(\pm r_2 e_i) &= f(z \pm r_2 e_i) - f(z) - \langle \pm r_2 e_i, \nabla^{(r_2)} f(z)\rangle \\ &= f(z \pm r_2 e_i) - f(z) \mp r_2 \nabla_i^{(r_2)} f(z) \\ &= f(z \pm r_2 e_i) - f(z) \mp \frac{f(z + r_2 e_i) - f(z - r_2 e_i)}{2} \\ &= \frac{f(z + r_2 e_i) - 2f(z) + f(z - r_2 e_i)}{2} \\ &= r_2^2 \Delta_i^{(r_2)} f(z)/2 \quad \leq r_2^2 \Delta^{(r_2)} f(z)/2. \end{aligned}$$

Therefore Lemma 5.3.7 implies that $\sup_{y\in B_1(0,r_2)} |d(y)| \leq r_2^2 \Delta^{(r_2)} f(z)/2$. □

We can now state the main result of this section, the quantum analogue of Lemma 5.3.5.

5.3.9. LEMMA. *Let $r_1 > 0$, $L > 0$, $\rho \in (0, 1/3]$, and suppose $\delta \in (0, r_1 n L/\rho]$, then $r_2 := \sqrt{\frac{\delta r_1 \rho}{nL}} \leq r_1$. Suppose $f : C \to \mathbb{R}$ is a convex function that is L-Lipschitz on $B_\infty(0, 2r_1)$, and we have quantum query access[6] to $\tilde{f}$, which is a δ-approximate version of f, via a unitary U over a (fine-enough) hypergrid of $B_\infty(0, 2r_1)$. Then we can compute a $\tilde{g} \in \mathbb{R}^n$ using $\mathcal{O}(\log(n/\rho))$ queries to U, such that with probability $\geq 1 - \rho$, we have*

$$f(y) \geq f(0) + \langle \tilde{g}, y\rangle - (23n)^2 \sqrt{\frac{\delta L}{\rho r_1}} \|y\| - 2L\sqrt{n} r_1 \qquad \textit{for all } y \in C.$$

Proof:
The quantum algorithm works roughly as follows. It first picks a uniformly[7] random $z \in B_\infty(0, r_1)$. Then it uses Jordan's quantum algorithm to compute an approximate gradient at z by approximately evaluating f in superposition over a

[6]Using Corollary 5.A.2 instead of Corollary 5.3.6 shows that a relational quantum oracle also suffices as input.

[7]A discrete quantum computer strictly speaking cannot do this, but (as noted after Lemma 5.3.4) a uniformly random point from a fine enough hypergrid suffices.

discrete hypergrid of $B_\infty(z, r_2/n)$. This then yields an approximate subgradient of f at 0.

We now work out this rough idea. Since $B_\infty(z, \frac{r_2}{n}) \subseteq B_1(z, r_2)$, Lemma 5.3.8 implies

$$\sup_{y \in B_\infty(0, r_2/n)} \left| f(z+y) - f(z) - \langle y, \nabla^{(r_2)} f(z) \rangle \right| \leq \frac{r_2^2 \Delta^{(r_2)} f(z)}{2}. \tag{5.4}$$

Also as shown by Lemma 5.3.4 and Markov's inequality we have

$$\Delta^{(r_2)} f(z) \leq \frac{2nL}{\rho r_1} \tag{5.5}$$

with probability $\geq 1 - \rho/2$ over the choice of z. If z is such that Equation (5.5) holds, then we get

$$\sup_{y \in B_\infty(0, r_2/n)} \left| f(z+y) - f(z) - \langle y, \nabla^{(r_2)} f(z) \rangle \right| \leq \frac{nLr_2^2}{\rho r_1} = \delta.$$

Now apply the quantum algorithm of Corollary 5.3.6 with $r = 2r_2/n$, $c = f(z)$, $g = \nabla^{(r_2)} f(z)$, and $B = Lr$. This uses $\mathcal{O}(\log(n/\rho))$ queries to U, and with probability $\geq 1 - \rho/2$ computes an approximate gradient $\tilde{g}$ such that

$$\left\| \nabla^{(r_2)} f(z) - \tilde{g} \right\|_\infty \leq \frac{8 \cdot 42 \pi n}{2r_2} \cdot \delta = 4 \cdot 42 \cdot \pi \sqrt{\frac{\delta n^3 L}{\rho r_1}}. \tag{5.6}$$

Also, if z is such that Equation (5.5) holds, then by Lemma 5.3.3 we get that

$$\sup_{g \in \underline{\partial} f(z)} \left\| \nabla^{(r_2)} f(z) - g \right\|_1 \leq \frac{r_2 \Delta^{(r_2)} f(z)}{2} \leq \frac{nLr_2}{\rho r_1} = \sqrt{\frac{\delta n L}{\rho r_1}},$$

and therefore by the triangle inequality and Equation (5.6) we get that

$$\begin{aligned}
\sup_{g \in \underline{\partial} f(z)} \|g - \tilde{g}\|_\infty &\leq \sup_{g \in \underline{\partial} f(z)} \left\| g - \nabla^{(r_2)} f(z) \right\|_\infty + \left\| \nabla^{(r_2)} f(z) - \tilde{g} \right\|_\infty \\
&\leq \sup_{g \in \underline{\partial} f(z)} \left\| g - \nabla^{(r_2)} f(z) \right\|_1 + \left\| \nabla^{(r_2)} f(z) - \tilde{g} \right\|_\infty \\
&\leq \sqrt{\frac{\delta n L}{\rho r_1}} + 4 \cdot 42 \cdot \pi \sqrt{\frac{\delta n^3 L}{\rho r_1}} \quad < 23^2 \sqrt{\frac{\delta n^3 L}{\rho r_1}}.
\end{aligned}$$

Thus with probability at least $1 - \rho$, for all $y \in C$ and for all $g \in \underline{\partial} f(z)$ we have that

$$\begin{aligned}
f(y) &\geq f(z) + \langle g, y - z\rangle \\
&= f(0) + \langle \tilde{g}, y\rangle + \langle g - \tilde{g}, y\rangle + (f(z) - f(0)) + \langle g, -z\rangle \\
&\geq f(0) + \langle \tilde{g}, y\rangle - |\langle g - \tilde{g}, y\rangle| - L\|z\| - \|g\|\|z\| \\
&\geq f(0) + \langle \tilde{g}, y\rangle - \|g - \tilde{g}\|_\infty \|y\|_1 - L\sqrt{n}r_1 - L\sqrt{n}r_1 \qquad \text{(by (5.1))} \\
&\geq f(0) + \langle \tilde{g}, y\rangle - 23^2\sqrt{\frac{\delta n^3 L}{\rho r_1}}\|y\|_1 - 2L\sqrt{n}r_1 \\
&\geq f(0) + \langle \tilde{g}, y\rangle - (23n)^2\sqrt{\frac{\delta L}{\rho r_1}}\|y\| - 2L\sqrt{n}r_1.
\end{aligned}$$

□

5.4 Algorithms for separation using membership queries

Let $K \subseteq \mathbb{R}^n$ be a convex set such that $B(0,r) \subseteq K \subseteq B(0,R)$. Given a membership oracle[8] $\mathrm{MEM}_{\epsilon,0}(K)$ as in Definition 5.2.3, we construct a separation oracle $\mathrm{SEP}_{\eta,\rho}(K)$ as in Definition 5.2.4. Let x be the point we want to separate from K. We first make a membership query to x itself, receiving answer $x \in B(K,\varepsilon)$ or $x \notin B(K,-\varepsilon)$. Suppose $x \notin B(K,-\varepsilon)$, then we need to find a hyperplane that approximately separates x from K. Due to the rotational symmetry of the separation problem, for ease of notation we assume that $x = -\|x\|e_n$.[9] For this x define $h : \mathbb{R}^{n-1} \to \mathbb{R} \cup \{\infty\}$ as

$$h(y) := \inf_{(y,y_n)\in K} y_n.$$

[8]For simplicity we assume throughout this section that the membership oracle succeeds with certainty (i.e., its error probability is 0). This is easy to justify: suppose we have a classical T-query algorithm, which uses $\mathrm{MEM}_{\epsilon,0}(K)$ queries and succeeds with probability at least $1-\rho$. If we are given access to a $\mathrm{MEM}_{\epsilon,\frac{1}{3}}(K)$ oracle instead, then we can create a $\mathrm{MEM}_{\epsilon,\frac{\rho}{T}}(K)$ oracle by $\mathcal{O}(\log(T/\rho))$ queries to $\mathrm{MEM}_{\epsilon,\frac{1}{3}}(K)$ and taking the majority of the answers. Then running the original algorithm with $\mathrm{MEM}_{\epsilon,\frac{\rho}{T}}(K)$ will fail with probability at most 2ρ. Therefore the assumption of a membership oracle with error probability 0 can be removed at the expense of only a small logarithmic overhead in the number of queries. A similar argument works for the quantum case.

[9]For the query complexity this is without loss of generality, since we can always apply a rotation to all the points such that this holds. If we instead consider the computational cost of our algorithm, then we have to take into account the cost of this rotation and its inverse. Note, however, that this rotation can always be written as the product of n rotations on only 2 coordinates, and hence can be applied in $\widetilde{\mathcal{O}}(n)$ additional steps.

Our h is a bit different from the one used in [LSV18], but we can show that it has many of the same properties. Since K is a convex set, h is a convex function over $\mathbb{R}^{n-1}$. As we show below, the function h is also Lipschitz (Lemma 5.4.1) and we can approximately compute its value using binary search with $\widetilde{\mathcal{O}}(1)$ classical queries to a membership oracle (Lemma 5.4.2). Furthermore, an approximate subgradient of h at 0 allows to construct a hyperplane approximately separating x from K (Lemma 5.4.3). Combined with the results of Section 5.3 this leads to the main results of this section, Theorems 5.4.4 and 5.4.5, which show how to efficiently construct a separation oracle using classical (resp. quantum) queries to a membership oracle.

Analogously to [LSV18, Lemma 12] we first show that our h is Lipschitz.

5.4.1. Lemma. *For every $\delta \in (0, r)$, h is $\frac{R}{r-\delta}$-Lipschitz on $B(0,\delta) \subseteq \mathbb{R}^{n-1}$, that is, we have*

$$|h(y') - h(y)| \leq \frac{R}{r-\delta}\|y' - y\| \quad \textit{for all } y, y' \in B(0,\delta).$$

Proof:
Observe that for all $y \in B(0,r)$ we have $-R \leq h(y) \leq 0$, because $B(0,r) \subseteq K \subseteq B(0,R)$. Let $y, y' \in B(0,\delta)$ be arbitrary, and let $z = \frac{y'-y}{\|y'-y\|}$. Observe that $y + (\|y'-y\| + (r-\delta))z = y' + (r-\delta)z \in B(0,r)$, and that

$$y' = \frac{\|y'-y\|}{\|y'-y\| + (r-\delta)}(y' + (r-\delta)z) + \frac{r-\delta}{\|y'-y\| + (r-\delta)}y,$$

and therefore due to convexity

$$h(y') - h(y) \leq [h(y' + (r-\delta)z) - h(y)]\frac{\|y'-y\|}{\|y'-y\| + (r-\delta)} \leq \frac{R}{r-\delta}\|y'-y\|.$$

□

Now we show how to compute the value of h using membership queries to K.

5.4.2. Lemma. *For all $y \in B\big(0, \frac{r}{2}\big) \subset \mathbb{R}^{n-1}$ we can compute a δ-approximation of $h(y)$ with $\mathcal{O}\big(\log\big(\frac{R}{\delta}\big)\big)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle, where $\varepsilon \leq \frac{r}{3R}\delta$.*

Proof:
Let $y \in B(0, \frac{r}{2})$, then $(y, h(y))$ is a boundary point of K by the definition of h. Note that $h(y) \in [-R, -r/2]$, our goal is to perform binary search over this interval to find a good approximation of $h(y)$. Suppose $y_n \leq -\frac{r}{2}$ is our current guess for $h(y)$. We first show that

(a) if $(y, y_n) \in B(K, \varepsilon)$, then $y_n \geq h(y) - \delta$, and

(b) if $(y, y_n) \notin B(K, -\varepsilon)$, then $y_n \leq h(y) + \frac{2}{3}\delta$.

For the proof of (a) consider a $g \in \underline{\partial} h(y)$. Since g is a subgradient we have that $h(z) \geq h(y) + \langle g, z-y \rangle$ for all $z \in \mathbb{R}^{n-1}$. Hence, for all $z \in \mathbb{R}^{n-1}$ and z_n such that $(z, z_n) \in K$ we have

$$\left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} y \\ h(y) \end{pmatrix} \right\rangle \leq \left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} z \\ h(z) \end{pmatrix} \right\rangle \leq \left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} z \\ z_n \end{pmatrix} \right\rangle$$

where the first inequality is a rewriting of the subgradient inequality and the second inequality uses that $z_n \geq h(z)$ since $(z, z_n) \in K$. Since $(y, y_n) \in B(K, \varepsilon)$ it follows from the above inequality that

$$\left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} y \\ y_n \end{pmatrix} \right\rangle \geq \left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} y \\ h(y) \end{pmatrix} \right\rangle - \varepsilon \left\| \begin{pmatrix} -g \\ 1 \end{pmatrix} \right\| \geq \left\langle \begin{pmatrix} -g \\ 1 \end{pmatrix}, \begin{pmatrix} y \\ h(y) \end{pmatrix} \right\rangle - \varepsilon(\|g\|+1).$$

Lemma 5.4.1 together with the argument of Equation (5.1) implies that $\|g\| \leq \frac{2R}{r}$. Since

$$\varepsilon(\|g\|+1) \leq \varepsilon\left(\frac{2R}{r}+1\right) \leq \varepsilon\frac{3R}{r} \leq \delta,$$

we obtain the inequality of (a).

For (b), consider the convex set C which is the convex hull of $B((y,0), r/2)$ and $(y, h(y))$. Note that $B(C, -\varepsilon)$ is the convex hull of $B((y,0), r/2-\varepsilon)$ and $\big(y, h(y)\big(1-\frac{2\varepsilon}{r}\big)\big)$. Since $C \subseteq K$, we have $B(C,-\varepsilon) \subseteq B(K,-\varepsilon)$. Therefore $(y, y_n) \notin B(K, -\varepsilon)$ implies $(y, y_n) \notin B(C, -\varepsilon)$, and

$$y_n \leq h(y)\left(1-\frac{2\varepsilon}{r}\right) = h(y) - \varepsilon\frac{2h(y)}{r} \leq h(y) + \varepsilon\frac{2R}{r} \leq h(y) + \frac{2}{3}\delta.$$

Now we can analyze the binary search algorithm. By making $\mathcal{O}\big(\log\big(\frac{R}{\delta}\big)\big)$ $\mathrm{MEM}_{\varepsilon,0}(K)$ queries to points of the form (y,z), we can find a value $y_n \in [-R, -\frac{r}{2}]$ such that $(y, y_n) \in B(K, \varepsilon)$ but $(y, y_n - \frac{\delta}{3}) \notin B(K, -\varepsilon)$. By (a)-(b) we get that $|h(y) - y_n| \leq \delta$. □

The following lemma shows how to convert an approximate subgradient of h to a hyperplane that approximately separates x from K.

5.4.3. LEMMA. *Suppose $-\|x\|e_n = x \notin B(K, -\varepsilon)$, and $\tilde{g} \in \mathbb{R}^{n-1}$ is an approximate subgradient of h at 0, meaning that for some $a, b \in \mathbb{R}$ and for all $y \in \mathbb{R}^{n-1}$*

$$h(y) \geq h(0) + \langle \tilde{g}, y \rangle - a\|y\| - b,$$

then $s := \frac{(-\tilde{g},1)}{\|(-\tilde{g},1)\|}$ satisfies $\langle s, z \rangle \geq \langle s, x \rangle - \frac{aR+b}{\|(-\tilde{g},1)\|} - \frac{2R}{r}\frac{\varepsilon}{\|(-\tilde{g},1)\|}$ for all $z \in K$.

Proof:
Let us introduce the notation $z = (y, z_n)$ and $s' := (-\tilde{g}, 1) = \|(-\tilde{g},1)\|s$, then

$$\begin{aligned}\langle s', z \rangle &= z_n - \langle \tilde{g}, y \rangle \geq h(y) - \langle \tilde{g}, y \rangle \geq h(0) - a\|y\| - b \\ &\geq -\|x\| - \frac{2R}{r}\varepsilon - aR - b = \langle s', x \rangle - aR - b - \frac{2R}{r}\varepsilon,\end{aligned}$$

where the last inequality used claim (b) from the proof of Lemma 5.4.2. □

We now construct a separation oracle using $\widetilde{\mathcal{O}}(n)$ classical queries to a membership oracle. In particular, for an η-precise separation oracle, we require an ε-precise membership oracle with

$$\varepsilon = \frac{\eta}{676} n^{-2} \left(\frac{r}{R}\right)^3 \left(\frac{\eta}{R}\right)^2 \rho$$

The analogous result in [LSV18, Theorem 14] uses the stronger assumption[10]

$$\varepsilon \approx \frac{\eta}{8 \cdot 10^6} n^{-\frac{7}{2}} \left(\frac{r}{R}\right)^6 \left(\frac{\eta}{R}\right)^2 \rho^3.$$

Compared to this, our result scales better in terms of n, $\frac{r}{R}$ and ρ.

5.4.4. THEOREM. *Let K be a convex set satisfying $B(0,r) \subseteq K \subseteq B(0,R)$. For any $\eta \in (0,R]$ and $\rho \in (0,1/3]$, we can implement the oracle* $\mathrm{SEP}_{\eta,\rho}(K)$ *using* $\mathcal{O}\Big(n \log\Big(\frac{n}{\rho}\frac{R}{\eta}\frac{R}{r}\Big)\Big)$ *classical queries to a* $\mathrm{MEM}_{\varepsilon,0}(K)$ *oracle, where*

$$\varepsilon \leq \eta(26n)^{-2} \left(\frac{r}{R}\right)^3 \left(\frac{\eta}{R}\right)^2 \rho.$$

Proof:
Let $x \notin B(K, -\varepsilon)$ be the point we want to separate from K. Let us fix $\delta := \eta\frac{n^{-2}}{9 \cdot 24}\big(\frac{r}{R} \cdot \frac{\eta}{R}\big)^2 \rho$, then $\varepsilon \leq \frac{r}{3R}\delta$. By Lemma 5.4.2 we can evaluate h to within error δ using $\mathcal{O}\big(\log\big(\frac{R}{\delta}\big)\big)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle. By Lemma 5.4.1 we know that h is $\frac{2R}{r}$-Lipschitz on $B(0,r/2)$. Let us choose $r_1 := \frac{r}{12\sqrt{n}}\frac{\eta}{R}$, then $r_1\sqrt{n} \leq \frac{r}{4}$, therefore $B_\infty(0,2r_1) \subseteq B(0,r/2)$. Also note that $\delta \leq \frac{\eta}{6\rho} = \frac{2r_1\sqrt{n}R}{\rho r}$. Hence by Lemma 5.3.5, using $\mathcal{O}\big(n\log\big(\frac{R}{\delta}\big)\big)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle, we can compute an approximate subgradient $\tilde{g}$ such that with probability at least $1-\rho$ we have

$$h(y) \geq h(0) + \langle \tilde{g}, y\rangle - \frac{3n^{\frac{3}{4}}}{2}\sqrt{\frac{\delta 2R}{\rho r_1 r}}\|y\| - \frac{4R}{r}\sqrt{n}r_1 \qquad \text{for all } y \in \mathbb{R}^{n-1}.$$

Substituting the value of r_1 and δ we get $h(y) \geq h(0) + \langle \tilde{g}, y\rangle - \frac{\eta}{2R}\|y\| - \frac{\eta}{3}$, which by Lemma 5.4.3 gives an s such that $\langle s, z\rangle \geq \langle s, x\rangle - \frac{5}{6}\eta - \frac{2R}{r}\varepsilon \geq \langle s, x\rangle - \eta$ for all $z \in K$ □

Finally, we give a proof of our main result: we construct a separation oracle using $\widetilde{\mathcal{O}}(1)$ quantum queries to a membership oracle.

[10]It seems that Lee et al. [LSV18, Algorithm 1] did not take into account the change in precision analogous to our Lemma 5.4.2, therefore one would probably need to worsen their exponent of $\frac{r}{R}$ from 6 to 7.

5.4.5. THEOREM. *Let K be a convex set satisfying $B(0,r) \subseteq K \subseteq B(0,R)$. For any $\eta \in (0,R]$ and $\rho \in (0,1/3]$, we can implement the oracle* $\mathrm{SEP}_{\eta,\rho}(K)$ *using* $\mathcal{O}\Big(\log\left(\frac{n}{\rho}\right)\log\Big(\frac{n}{\rho}\frac{R}{\eta}\frac{R}{r}\Big)\Big)$ *quantum queries to a* $\mathrm{MEM}_{\varepsilon,0}(K)$ *oracle, where* $\varepsilon \leq \eta(58n)^{-\frac{9}{2}}\left(\frac{r}{R}\right)^3\left(\frac{\eta}{R}\right)^2\rho$.

Proof:
Let $x \notin B(K,-\varepsilon)$ be the point we want to separate from K. Let us fix $\delta := \eta\frac{23^{-4}}{4\cdot 24}n^{-\frac{9}{2}}\left(\frac{r}{R}\cdot\frac{\eta}{R}\right)^2\rho$, then $\varepsilon \leq \frac{r}{3R}\delta$. By Lemma 5.4.2 we can evaluate h to within error δ using $\mathcal{O}\big(\log\big(\frac{R}{\delta}\big)\big)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle. By Lemma 5.4.1 we know that h is $\frac{2R}{r}$-Lipschitz on $B(0,r/2)$. Let us choose $r_1 := \frac{r}{12\sqrt{n}}\frac{\eta}{R}$, then $r_1\sqrt{n} \leq \frac{r}{4}$, therefore $B_\infty(0,2r_1) \subseteq B(0,r/2)$. Also note that $\delta \leq \frac{\eta}{6\rho} = \frac{2r_1nR}{\rho r}$. Hence by Lemma 5.3.9, using $\mathcal{O}\Big(\log\big(\frac{n}{\rho}\big)\log\big(\frac{R}{\delta}\big)\Big)$ queries to a $\mathrm{MEM}_{\varepsilon,0}(K)$ oracle, we can compute an approximate subgradient $\tilde{g}$ such that with probability at least $1-\rho$ we have

$$h(y) \geq h(0) + \langle \tilde{g}, y\rangle - (23n)^2\sqrt{\frac{2\delta R}{\rho r_1 r}}\|y\| - \frac{4R}{r}\sqrt{n}r_1 \qquad \text{for all } y \in \mathbb{R}^{n-1}.$$

Substituting the value of r_1 and δ we get $h(y) \geq h(0) + \langle \tilde{g}, y\rangle - \frac{\eta}{2R}\|y\| - \frac{\eta}{3}$, which by Lemma 5.4.3 gives an s such that $\langle s, z\rangle \geq \langle s, x\rangle - \frac{5}{6}\eta - \frac{2R}{r}\varepsilon \geq \langle s, x\rangle - \eta$ for all $z \in K$. $\square$

5.5 Lower bounds

For a convex set K satisfying $B(0,r) \subseteq K \subseteq B(0,R)$, we have shown in Theorem 5.4.5 that one can implement a $\mathrm{SEP}(K)$ oracle with $\widetilde{\mathcal{O}}(1)$ quantum queries to a $\mathrm{MEM}(K)$ oracle if the membership oracle is sufficiently precise. In this section we first show that this is exponentially better than what can be achieved using classical access to a membership oracle. We also investigate how many queries to a membership/separation oracle are needed in order to implement an optimization oracle. Our results are as follows.

- We show that $\Omega(n)$ classical queries to a membership oracle are needed to implement a weak separation oracle.
- We show that $\Omega(n)$ classical (resp. $\Omega(\sqrt{n})$ quantum) queries to a separation oracle are needed to implement a weak optimization oracle; even when we *know an interior point* in the set.
- We show an $\Omega(n)$ lower bound on the number of classical and/or quantum queries to a separation oracle needed to optimize over the set when we *do not know an interior point.*

In this section we will always assume that the input oracle is a strong oracle but the output oracle is allowed to be a weak oracle with error ε. Furthermore, we will make sure that R, $1/r$, and $1/\varepsilon$ are all upper bounded by a polynomial in n. This guarantees that the lower bound is based on the dimension of the problem, not the required precision.

5.5.1 Classical lower bound on the number of MEM queries needed for SEP

Here we show that a separation query can provide $\Omega(n)$ bits of information about the underlying convex set K; since a classical membership query returns a 0 or a 1 and hence can give at most 1 bit of information[11], this theorem immediately implies a lower bound of $\Omega(n)$ on the number of classical membership queries needed to implement one separation query.

5.5.1. Theorem. *Let $\epsilon \leq \frac{39}{1600}$. There exist a set of $m = 2^{\Omega(n)}$ convex sets $K_1, \ldots, K_m$ and points $y, x_0 \in \mathbb{R}^n$ such that $B(x_0, 1/3) \subseteq K_i \subseteq B(x_0, 2\sqrt{n})$ for all $i \in [m]$, and such that the result of a classical query to $\mathrm{SEP}_{\varepsilon,0}(K_i)$ with the point y correctly identifies i.*

Proof:
Let $h_1, \ldots, h_m \in \mathbb{R}^n$ be a set of $m = 2^{\Omega(n)}$ entrywise non-negative unit vectors such that $\langle h_i, h_j \rangle \leq 0.51$ for all distinct $i, j \in [m]$. Such a set of m vectors can for instance be constructed from a good error-correcting code that encodes $\Omega(n)$-bit words into n-bit codewords with pairwise Hamming distance close to $n/2$.

Now pick an $i \in [m]$ and define $\hat{K}_i := \{x : \langle h_i, x \rangle \leq 0\} \cap B(0, \sqrt{n})$ and $K_i := B(\hat{K}_i, \epsilon)$. Then $\hat{K}_i = B(K_i, -\epsilon)$. We claim that a query to $\mathrm{SEP}_{\varepsilon,0}(K_i)$ with the point $y = 3\varepsilon e \in \mathbb{R}^n$ will identify h_i. First note that $y \notin B(K_i, \epsilon)$, since $\hat{K}_i$ does not contain any entrywise positive vectors and y has distance at least 3ε from all vectors that have at least one non-positive entry. Hence a separation query with y will return a unit vector g such that for all $x \in \hat{K}_i$

$$\langle g, x \rangle \leq \langle g, y \rangle + \varepsilon \leq \|g\| \cdot \|y\| + \varepsilon \leq (3\sqrt{n} + 1)\varepsilon \leq 4\sqrt{n}\varepsilon. \tag{5.7}$$

Now consider the specific point x that is the projection of g onto $h_i^\perp$ (the hyperplane orthogonal to h_i) scaled by a factor $\sqrt{n}$, i.e., $x = \sqrt{n}(g - \langle g, h_i \rangle h_i)$. Since $\langle h_i, x \rangle = 0$ and $\|x\| \leq \sqrt{n}$, we have $x \in \hat{K}_i$. Therefore (5.7) gives the following inequality

$$\sqrt{n}(1 - \langle g, h_i \rangle^2) = \langle g, x \rangle \leq 4\sqrt{n}\varepsilon.$$

Hence $|\langle g, h_i \rangle| \geq \sqrt{1 - 4\varepsilon} \geq \frac{19}{20}$. This implies that $g - h_i$ or $g + h_i$ has length at most $\sqrt{2 - 2|\langle g, h_i \rangle|} \leq \sqrt{\frac{1}{10}}$; assume the former for simplicity. Now for all $j \neq i$

[11]This is not true for *quantum* membership queries!

we have

$$|\langle g, h_j\rangle| \leq |\langle g - h_i, h_j\rangle| + |\langle h_i, h_j\rangle| \leq \sqrt{\frac{1}{10}} + 0.51 < \frac{9}{10}.$$

Hence g uniquely identifies h_i. Finally, for $x_0 = -e/3$ we have $B(x_0, 1/3) \subseteq K_i \subseteq B(x_0, 2\sqrt{n})$. $\square$

5.5.2 Lower bound on number of SEP queries for OPT (given an interior point)

We now consider lower bounding the number of quantum queries to a separation oracle needed to do optimization. In fact, we prove a lower bound on the number of separation queries needed for validity, which implies the same bound on optimization. We will use a reduction from a version[12] of the well-studied *search* problem:

Given $z \in \{0,1\}^n$ such that either $|z|=0$ or $|z|=1$, decide which of the two holds.

It is not hard to see that if the access to z is given via classical queries, then $\Omega(n)$ queries are needed. It is well known [BBBV97] that if we allow quantum queries, then $\Omega(\sqrt{n})$ queries are needed (i.e., Grover's quantum search algorithm [Gro96] is optimal). This was first proven using the hybrid-method, see Theorem 4.1.2. We use this problem to show that there exist convex sets for which it is hard to construct a weak validity oracle, given a strong separation oracle. Since a separation oracle can be used as a membership oracle, this gives the same hardness result for constructing a weak validity oracle from a strong membership oracle.

5.5.2. Theorem. *Let $0 < \rho \leq 1/3$. Let $\mathcal{A}$ be an algorithm that can implement a $\mathrm{VAL}_{(4n)^{-1},\rho}(K)$ oracle for every convex set K (with $B(x_0, r) \subseteq K \subseteq B(x_0, R)$) using only queries to a $\mathrm{SEP}_{0,0}(K)$ oracle, and unitaries that are independent of K. Then the following statements are true, even when we restrict to convex sets K with $r = 1/3$ and $R = 2\sqrt{n}$:*

- *if the queries to $\mathrm{SEP}_{0,0}(K)$ are classical, then the algorithm uses $\Omega(n)$ queries.*
- *if the queries to $\mathrm{SEP}_{0,0}(K)$ are quantum, then the algorithm uses $\Omega(\sqrt{n})$ queries.*

Proof:
Let $z \in \{0,1\}^n$ have Hamming weight $|z| = 0$ or $|z| = 1$. We construct a set K_z

[12]Note that this is a slightly different version from the one used in Section 5.2.1.

in such a way that solving the weak validity problem solves the search problem for z, while separation queries for K_z can be answered using a single query to z. The known classical and quantum lower bounds on the search problem then imply the two claims of the theorem, respectively.

Define $K_z := \times_{i=1}^{n}[-1, z_i]$. We first show how to implement a strong separation oracle using a single query to z. Suppose the input is the point y. The strong separation oracle works as follows:

1. If $y \in [-1, 0]^n$, then return the statement that $y \in B(K_z, 0) = K_z$.

2. If $y \notin [-1, 1]^n$, then return a hyperplane that separates y from $[-1, 1]^n$ (and hence from K_z).

3. Let i be such that $y_i > 0$. Query z_i.

   (a) If $z_i = 1$ and i is the only index such that $y_i > 0$, then return that $y \in B(K_z, 0) = K_z$.

   (b) If $z_i = 1$ and there is a $j \neq i$ such that $y_j > 0$, return separating hyperplane $x_j \leq y_j$.

   (c) If $z_i = 0$, then return the separating hyperplane $x_i \leq y_i$.

It remains to show that a query to a weak validity oracle with accuracy $\epsilon = \frac{1}{4n}$ can solve the search problem on z. We show that a validity query over K_z with the direction $c = \frac{1}{\sqrt{n}}(1, \ldots, 1) \in \mathbb{R}^n$ and value $\gamma = \frac{1}{2\sqrt{n}}$ solves the search problem:

- If $|z| = 0$, then we claim validity will return that $\langle c, x\rangle \leq \gamma + \epsilon$ holds for all $x \in B(K_0, -\epsilon)$.

  Indeed, we show there is no $x \in B(K_0, \epsilon)$ with $\langle c, x\rangle \geq \gamma - \epsilon$. For all points $x \in K_0$ we have $\langle c, x\rangle \leq 0$. Thus, for all points $x \in B(K_0, \epsilon)$ we have $\langle c, x\rangle \leq \epsilon < \gamma - \epsilon$.

- If $|z| = 1$, then we claim validity will return that $\langle c, x\rangle \geq \gamma - \epsilon$ holds for some $x \in B(K_z, \epsilon)$.

  Indeed, we show there is an $x \in B(K_z, -\epsilon)$ for which $\langle c, x\rangle > \gamma + \epsilon$. The point $z \in K_z$ satisfies $\langle z, c\rangle = \frac{1}{\sqrt{n}}$ and therefore $x = z - \epsilon e \in B(K_z, -\epsilon)$ satisfies $\langle c, x\rangle = \frac{1}{\sqrt{n}} - \sqrt{n}\epsilon > \gamma + \epsilon$.

Finally, we observe that if we set $x_0 = (-1/2, \ldots, -1/2)$, then $B(x_0, \frac{1}{3}) \subseteq K_z \subseteq B(x_0, 2\sqrt{n})$. $\square$

5.5.3 Lower bound on number of SEP queries for OPT (without interior point)

We now lower bound the number of quantum queries to a separation oracle needed to solve the optimization problem, if our algorithm does not already know an interior point of K. In fact we prove a lower bound on finding a point in K using separation queries, which implies the lower bound on the number of separation queries needed for optimization.

We prove our lower bound by a reduction to the problem of learning z with *first-difference queries*. Here one needs to find an initially unknown n-bit binary string z via a guessing game. For a given guess $g \in \{0,1\}^n$ a query returns the first index in $[n]$ for which the binary strings z and g differ (or it returns $n+1$ if $z = g$). The goal is to recover z with as few guesses as possible. First we prove an $\Omega(n)$ quantum query lower bound for this problem.[13]

5.5.3. Theorem.(Quantum lower bound for learning z by first-difference queries) *Let $z \in \{0,1\}^n$ be an unknown string accessible by an oracle acting as $O_z|g,b\rangle = |g, b \oplus f(g,z)\rangle$, where $f(g,z)$ is the first index for which z and g differ, more precisely $f(g,z) = \min\{i \in [n] : g_i \neq z_i\}$ if $g \neq z$ and $f(g,z) = n+1$ otherwise. Then every quantum algorithm that outputs z with high probability uses at least $\Omega(n)$ queries to O_z.*

Proof:
We will use the general adversary bound [HLŠ07]. For this problem, we call $\Gamma \in \mathbb{R}^{2^n \times 2^n}$ an *adversary matrix* if it is a non-zero matrix with zero diagonal whose rows and columns are indexed by all $z \in \{0,1\}^n$. For $g \in \{0,1\}^n$ let us define $\Delta_g \in \{0,1\}^{2^n \times 2^n}$ such that the $[z,z']$ entry of Δ_g is 0 if and only if $f(g,z) = f(g,z')$. The general adversary bound tells us that for any adversary matrix Γ, the quantum query complexity of our problem is

$$\Omega\left(\frac{\|\Gamma\|}{\max_{g\in\{0,1\}^n}\|\Gamma \circ \Delta_g\|}\right), \tag{5.8}$$

where "$\circ$" denotes the Hadamard product and $\|\cdot\|$ the operator norm.

We claim that Equation (5.8) gives a lower bound of $\Omega(n)$ for the adversary matrix Γ defined as

$$\Gamma[z,z'] = \begin{cases} 2^{f(z,z')} & \text{if } z \neq z' \\ 0 & \text{if } z = z' \end{cases}$$

It is easy to see that Γ is indeed an adversary matrix since it is zero on the diagonal and non-zero everywhere else. Furthermore, the all-one vector e is an

[13]Note that this is a strengthening of the $\Omega(n)$ quantum query lower bound for binary search on a space of size 2^n by Ambainis [Amb99], since first-difference queries are at least as strong as the queries one makes in binary search.

eigenvector of Γ with eigenvalue $n2^n$:

$$(\Gamma e)_z = \sum_{z'\in\{0,1\}^n} \Gamma[z,z'] = \sum_{d=1}^{n} 2^d \cdot |\{z' \in \{0,1\}^n \,:\, f(z,z') = d\}| = \sum_{d=1}^{n} 2^d 2^{n-d} = n2^n.$$

So $\Gamma e = n2^n e$ and hence $\|\Gamma\| \geq n2^n$.

From the definition of Δ_g it follows that

$$(\Gamma \circ \Delta_g)[z,z'] = 2^{f(z,z')}\chi_{[f(g,z)\neq f(g,z')]},$$

where $\chi_{[f(g,z)\neq f(g,z')]}$ stands for the indicator function of the condition $f(g,z) \neq f(g,z')$. Let $\Gamma_g := \Gamma \circ \Delta_g$. We will show an upper bound on $\|\Gamma_g\|$. We decompose Γ_g in an "upper-triangular" and a "lower-triangular" part:

$$\begin{aligned} \Gamma_g^U[z,z'] &:= 2^{f(z,z')}\chi_{[f(g,z)<f(g,z')]} = 2^{f(g,z)}\chi_{[f(g,z)<f(g,z')]}, \\ \Gamma_g^L[z,z'] &:= 2^{f(z,z')}\chi_{[f(g,z')<f(g,z)]} = 2^{f(g,z')}\chi_{[f(g,z')<f(g,z)]}. \end{aligned} \tag{5.9}$$

So $\Gamma_g = \Gamma_g^U + \Gamma_g^L$ and $\Gamma_g^U = (\Gamma_g^L)^T$. Hence by the triangle inequality we have

$$\|\Gamma_g\| \leq \left\|\Gamma_g^U\right\| + \left\|\Gamma_g^L\right\| = 2\left\|\Gamma_g^U\right\|. \tag{5.10}$$

It thus suffices to upper bound $\left\|\Gamma_g^U\right\|$. Notice that as (5.9) shows, $\Gamma_g^U[z,z']$ only depends on the values $f(g,z)$, $f(g,z')$. Since the range of $f(g,\,\cdot\,)$ is $[n+1]$, we can think of Γ_g^U as an $(n+1)\times(n+1)$ block-matrix, where the blocks are determined by the values of $f(g,z)$ and $f(g,z')$, and within a block all matrix elements are the same. Also observe that for all $k \in [n]$ there are 2^{n-k} bitstrings $y \in \{0,1\}^n$ such that $f(g,y) = k$, which tells us the sizes of the blocks. Motivated by these observations we define an orthonormal set of vectors in $\mathbb{R}^{2^n}$ by $v_{n+1} := e_g$, and for all $k \in [n]$

$$v_k := \sum_{y:f(g,y)=k} \frac{e_y}{\sqrt{2^{n-k}}}.$$

Since the row and column spaces of Γ_g^U are spanned by $\{v_k : k \in [n+1]\}$, we can reduce Γ_g^U to a $(n+1)\times(n+1)$-dimensional matrix G:

$$\begin{aligned} \Gamma_g^U &= \left(\sum_{k=1}^{n+1} v_k v_k^T\right)\Gamma_g^U\left(\sum_{\ell=1}^{n+1} v_\ell v_\ell^T\right) \\ &= \left(\sum_{k=1}^{n+1} v_k e_k^T\right)\underbrace{\left(\sum_{k=1}^{n+1} e_k v_k^T\right)\Gamma_g^U\left(\sum_{\ell=1}^{n+1} v_\ell e_\ell^T\right)}_{G:=}\left(\sum_{\ell=1}^{n+1} e_\ell v_\ell^T\right). \end{aligned}$$

It follows from the above identity, together with the orthonormality of the vectors $\{v_1, \ldots, v_n, v_{n+1}\}$, that

$$\|\Gamma_g^U\| = \left\| \left(\sum_{k=1}^{n+1} e_k v_k^T \right) \Gamma_g^U \left(\sum_{\ell=1}^{n+1} v_\ell e_\ell^T \right) \right\| = \|G\|. \tag{5.11}$$

$G \in \mathbb{R}^{(n+1)\times(n+1)}$ is a strictly upper-triangular matrix, with the following entries for $k, \ell \in [n]$:

$$\begin{aligned}
G[k,\ell] &= v_k^T \Gamma_g^U v_\ell \\
&= \left(\sum_{z: f(g,z)=k} \frac{e_z^T}{\sqrt{2^{n-k}}} \right) \Gamma_g^U \left(\sum_{z': f(g,z')=\ell} \frac{e_{z'}}{\sqrt{2^{n-\ell}}} \right) \\
&= \frac{2^{\frac{k+\ell}{2}}}{2^n} \left(\sum_{z: f(g,z)=k} e_z^T \right) \Gamma_g^U \left(\sum_{z': f(g,z')=\ell} e_{z'} \right) \\
&= \frac{2^{\frac{k+\ell}{2}}}{2^n} \sum_{z: f(g,z)=k} \sum_{z': f(g,z')=\ell} \Gamma_g^U[z,z'] \\
&= \frac{2^{\frac{k+\ell}{2}}}{2^n} \sum_{z: f(g,z)=k} \sum_{z': f(g,z')=\ell} 2^k \chi_{[k<\ell]} \qquad \text{(by (5.9))} \\
&= \frac{2^{\frac{k+\ell}{2}}}{2^n} 2^{n-k} 2^{n-\ell} 2^k \chi_{[k<\ell]} \\
&= 2^{n-\frac{\ell-k}{2}} \chi_{[k<\ell]}.
\end{aligned}$$

Similarly for $\ell = n+1$ we get that $G[k,\ell] = \sqrt{2}\, 2^{n-\frac{\ell-k}{2}} \chi_{[k<\ell]}$ for all $k \in [n+1]$. For each $d \in [n]$ define $G_d \in \mathbb{R}^{(n+1)\times(n+1)}$ such that $G_d[k,\ell] = G[k,\ell]\chi_{[d=\ell-k]}$. This G_d is only non-zero on a non-main diagonal (namely the (k,ℓ)-entries where $d = \ell - k$), and its non-zero entries are all upper bounded by $\sqrt{2}\, 2^n 2^{-\frac{d}{2}}$. We have $G = \sum_{d=1}^n G_d$ and therefore

$$\|G\| \le \sum_{d=1}^n \|G_d\| = \sum_{d=1}^n \sqrt{2}\, 2^n 2^{-\frac{d}{2}} = 2^n \sum_{d=0}^{n-1} (\sqrt{2})^{-d} \le \frac{2^n}{1 - 1/\sqrt{2}} \le 2^{n+2}. \tag{5.12}$$

Inequalities (5.10)-(5.12) give that $\|\Gamma_g\| \le 2^{n+3}$ and hence (5.8) yields a lower bound of $\Omega\left(\frac{n2^n}{2^{n+3}}\right) = \Omega(n)$ on the number of quantum queries to O_z needed to learn z. $\square$

5.5.4. THEOREM. *Finding a point in $B_\infty(K, 1/7)$ for an unknown convex set K such that $K \subseteq B_\infty(0,2) \subseteq \mathbb{R}^n$ requires $\Omega(n)$ quantum queries to a separation oracle $\mathrm{SEP}_{0,0}(K)$, even if we are promised there exists some unknown $x \in \mathbb{R}^n$ such that $B_\infty(x, 1/3) \subseteq K$.*

Proof:
We will prove an $\Omega(n)$ quantum query lower bound for this problem by a reduction from learning with first-difference queries. Let $z \in \{0,1\}^n$ be an unknown binary string, and let us define $K_z := B_\infty(z, 1/3) \subset \mathbb{R}^n$ as a small box around the corner of the hypercube corresponding to z. Then clearly $K_z \subset B_\infty(0,2)$, and finding a point close enough to K_z is enough to recover z.

We can also easily reduce a separation oracle query to a first-difference query to z, as follows. Suppose y is the vector we query:

1. If y is outside $[-1/3, 4/3]^n$, then output a hyperplane separating y from $[-1/3, 4/3]^n$.
2. If y is in $[-1/3, 4/3]^n$, then let g be the nearest corner of the hypercube.
3. Let i be the result of a first-difference query to z with g.
   (a) If $z = g$, then we know K_z exactly, so we can find a separating hyperplane or conclude that $y \in K_z$.
   (b) If $z \neq g$, then return e_i if $g_i = 1$, and $-e_i$ if $g_i = 0$.

Hence our $\Omega(n)$ quantum lower bound on learning z with first-difference queries implies an $\Omega(n)$ lower bound on the number of quantum queries to a separation oracle needed for finding a point in a convex set. □

Since optimization over a set K gives a point in the set K, this also implies a lower bound on the number of separation queries needed for optimization. This theorem is tight up to logarithmic factors, since it is known that $\widetilde{\mathcal{O}}(n)$ classical separation queries suffice for optimization, even without knowing a point in the convex set. Finally we remark that, due to our improved algorithm for optimization using validity queries, this also gives an $\widetilde{\Omega}(n)$ lower bound on the number of separation queries needed to implement validity.[14]

5.6 Consequences of convex polarity

Here we justify the central symmetry of Figure 5.1 using the results of Grötschel, Lovász, and Schrijver [GLS88, Section 4.4]. We first need to recall the definition and some basic properties of the polar K^* of a set $K \subseteq \mathbb{R}^n$. This is the closed convex set defined as follows:

$$K^* = \{y \in \mathbb{R}^n : \langle y, x\rangle \leq 1 \text{ for all } x \in K\}.$$

[14] It is easy to modify Theorem 5.5.3 to prove a lower bound on computing the majority of z, which would imply an $\Omega(n)$ lower bound on the number of separation queries needed to implement a validity oracle, without the log factors.

It is straightforward to verify that if $B(0,r) \subseteq K \subseteq B(0,R)$, then $B(0,1/R) \subseteq K^* \subseteq B(0,1/r)$, moreover $(K^*)^* = K$ for closed convex sets.[15] For the remainder of this section we assume that K is a closed convex set such that $B(0,r) \subseteq K \subseteq B(0,R)$.

We will observe that for the polar K^* of a set K the following holds:

$$\mathrm{MEM}(K^*) \leftrightarrow \mathrm{VAL}(K), \qquad \mathrm{SEP}(K^*) \leftrightarrow \mathrm{VIOL}(K), \tag{5.13}$$

where $\mathrm{MEM}(K^*) \leftrightarrow \mathrm{VAL}(K)$ means we can implement a weak validity oracle for K using a single query to a weak membership oracle for K^*, and vice versa. Since $\mathrm{VIOL}(K)$ and $\mathrm{OPT}(K)$ are equivalent up to $\widetilde{\Theta}(1)$ reductions (via binary search), this justifies the central symmetry of Figure 5.1, because it shows that algorithms that implement $\mathrm{VIOL}(K)$ given $\mathrm{VAL}(K)$ are equivalent to algorithms that implement $\mathrm{SEP}(K^*)$ given $\mathrm{MEM}(K^*)$, and similarly algorithms that implement $\mathrm{SEP}(K)$ given $\mathrm{VIOL}(K)$ are equivalent to algorithms that implement $\mathrm{VIOL}(K^*)$ given $\mathrm{SEP}(K^*)$.

Grötschel, Lovász, and Schrijver [GLS88, Section 4.4] showed that the weak membership problem for K^* can be solved using a single query to a weak validity oracle for K, and that the weak separation problem for K^* can be solved using a single query to a weak violation oracle for K. Using similar arguments one can show the reverse directions as well, which justifies (5.13). Here we only motivate the equivalences between the above-mentioned weak oracles by showing the equivalence of the strong oracles (i.e., where ρ and ε are 0).

Strong membership on K^* is equivalent to strong validity on K. First, for a given vector $c \in \mathbb{R}^n$ and a $\gamma > 0$ observe the following:

$$\frac{c}{\gamma} \notin \mathrm{int}(K^*) \iff \exists y \in K \text{ s.t. } \langle c/\gamma, y\rangle \geq 1 \iff \exists y \in K \text{ s.t. } \langle c, y\rangle \geq \gamma.$$

Hence, a strong membership query to K^* with a point c can be implemented by querying a strong validity oracle for K with the vector c and the value 1. Likewise, a strong validity query to K with a point c and value[16] $\gamma > 0$ can be implemented using a strong membership query to K^* with c/γ.

Strong separation on K^* is equivalent to strong violation on K. To implement a strong separation query on K^* for a vector $y \in \mathbb{R}^n$ we do the following. Query the strong violation oracle for K with y and the value 1. If the answer is that $\langle y, x\rangle \leq 1$ for all $x \in K$, then $y \in K^*$. If instead we are given a

[15]Note that K^* is a dual representation of the convex set K. Each point in K^* corresponds to a (normalized) valid inequality for K. This duality is not to be confused with Lagrangian duality.

[16]Observe that queries with value $\gamma \leq 0$ can be answered trivially, since $0 \in K$.

vector $x \in K$ with $\langle y, x\rangle \geq 1$, then x separates y from K^* (indeed, for all $z \in K^*$, we have $\langle z, x\rangle \leq 1 \leq \langle y, x\rangle$).

For the reverse direction, to implement a strong violation oracle for K on the vector c and value[16] $\gamma > 0$ we do the following. Query the strong separation oracle for K^* with the point c/γ. If the answer is that $c/\gamma \in K^*$ then $\langle c, x\rangle \leq \gamma$ for all $x \in K$. If instead we are given a non-zero vector $y \in \mathbb{R}^n$ that satisfies $\langle c/\gamma, y\rangle \geq \langle z, y\rangle$ for all $z \in K^*$, then $\tilde{y} = y/\langle c/\gamma, y\rangle$ will be a valid answer for the strong violation oracle for K. Indeed, we have $\tilde{y} \in K$ because $\langle z, \tilde{y}\rangle \leq 1$ for all $z \in K^*$ and $K = (K^*)^*$, and by construction $\langle c, \tilde{y}\rangle = \gamma$.

5.7 Future work

We mention several open problems for future work:

- Can we improve our $\Omega(\sqrt{n})$ lower bound on the number of separation queries needed to implement an optimization oracle when our algorithm knows a point in K? We conjecture that the correct bound is $\widetilde{\Theta}(n)$, in which case knowing a point in K does not confer much benefit for query complexity.

- Are there interesting convex optimization problems where separation is much harder than membership for classical computers? Such problems would be good candidates for quantum speed-up in optimization in the real, non-oracle setting. It is known that given a deterministic algorithm for a function, an algorithm with roughly the same complexity can be constructed to compute the gradient of that function [GW08], so for deterministic oracles separation is not much harder than membership queries. This, however, still leaves randomized and quantum membership oracles to be considered.

- The algorithms that give an $\widetilde{\mathcal{O}}(n)$ upper bound on the number of separation queries for optimization (for example [LSW15, Theorem 42]) give the best theoretical results for many convex optimization problems. However, due to the large constants in these algorithms they are rarely used in a practical setting. A natural question is whether the algorithms used in practice lend themselves to quantum speed-ups as well. Recent work by Kerenidis and Prakash [KP18] on quantum interior point methods is a first step in this direction.

5.A Quantum gradient computation using relational oracles

In this appendix we extend the result of Corollary 5.3.6 to functions given by a relational input oracle. As a direct consequence this shows that the algorithm from Theorem 5.4.5 also works when the input is given as a relational membership oracle instead of a standard oracle.

5.A.1. Definition (Unitary δ-approximator). Let X be a finite set and let Y denote a set of fixed-point b-bit numbers. Let $f\colon X \to Y$ be a function. We say that a relational quantum oracle U on X is a *b-bit unitary δ-approximator of f* if the valid answers for each $x \in X$ differ at most δ from $f(x)$ (i.e., $\mathcal{F}(x) = \{y \in Y\colon |f(x) - y| \leq \delta\}$), and the success probability is at least $\frac{2}{3}$.

5.A.2. Corollary (Gradient computation using a unitary δ-approximator). *Let $\delta, B, r, c \in \mathbb{R}$, $\rho \in (0, 1/3]$. Let $x_0, g \in \mathbb{R}^n$ with $\|g\|_\infty \leq \frac{B}{r}$, and let $m := \left\lceil \log_2\left(\frac{B}{28\pi\delta}\right)\right\rceil$. Suppose $f : (x_0 + rG^n_m) \to \mathbb{R}$ is such that*

$$|f(x_0 + rx) - \langle g, rx\rangle - c| \leq \delta$$

for 99.9% of the points $x \in G^n_m$, and we have access to U, an $\mathcal{O}\left(\log\left(\frac{B}{\delta}\right)\right)$-bit unitary δ-approximator of f over the domain $(x_0 + rG^n_m)$. Then we can compute a vector $\tilde{g} \in \mathbb{R}^n$ such that

$$\Pr\left[\, \|\tilde{g} - g\|_\infty > \frac{8 \cdot 42\pi\delta}{r}\right] \leq \rho,$$

with $\mathcal{O}\Big(\log\left(\frac{n}{\rho}\right)\Big)$ queries to U and $U^\dagger$ and $\mathcal{O}\Big(n \log\left(\frac{n}{\rho}\right)\log\left(\frac{B}{\delta}\right)\mathrm{loglog}\left(\frac{n}{\rho}\right)\mathrm{loglog}\left(\frac{B}{\delta}\right)\Big)$ additional gate complexity.

Proof:
The algorithm is the same as in the less general Corollary 5.3.6 presented in Section 5.3.2, we just need to analyze it a bit more carefully. The main idea is still to implement an approximate version of the phase oracle $\mathrm{O} : |x, 0, 0\rangle \mapsto e^{2\pi i \frac{M}{3B} f(x_0+rx)}|x, 0, 0\rangle$, and then use Jordan's gradient computation algorithm. We approximate O by first approximately computing f using U, then applying[17] a controlled phase operation cP acting as $\mathrm{cP}\colon |y\rangle \mapsto e^{2\pi i \frac{M}{3B} y}|y\rangle$ (where $M = \frac{3B}{84\pi\delta}$ as in the proof of Corollary 5.3.6), and finally applying $U^\dagger$ to approximately uncompute f.

We can assume without loss of generality that our unitary δ-approximator is such that the probability of $|f(x) - y| > \delta$ is at most $\frac{1}{1200}$. If this is not the case, we can improve the success probability by querying U a few times and taking the median of the results.

[17]If y is a b-bit fixed-point binary number, then this can be implemented using b single-qubit phase gates as follows: we can assume without loss of generality that $y = a_0 + a \cdot \sum_{j=1}^{b} y_j 2^j$ for some fixed $a_0, a \in \mathbb{R}$. Then $e^{2\pi i \frac{M}{3B} y} = e^{2\pi i \frac{M}{3B} a_0} \prod_{j=1}^{b} e^{2\pi i \frac{M}{3B} a y_j 2^j}$. The global phase is irrelevant, and the other phase factors can be implemented by using b single-qubit phase gates, each acting as $|y_j\rangle \mapsto e^{2\pi i \frac{M}{3B} a y_j 2^j}|y_j\rangle$.

Let us define $\mathcal{F}(x) := \{y \in Y \colon |f(x)-y| \leq \delta\}$ as in Definition 5.A.1. Observe that

$$\begin{aligned}
\left\|\mathrm{O}|x,0,0\rangle - U^\dagger(I\otimes \mathrm{cP}\otimes I)U|x,0,0\rangle\right\|^2 &= \\
&= \left\|\left(I\otimes (e^{2\pi i \frac{M}{3B}f(x_0+rx)}I-\mathrm{cP})\otimes I\right)U|x,0,0\rangle\right\|^2 \\
&= \left\|\sum_{y\in Y}\left(e^{2\pi i \frac{M}{3B}f(x_0+rx)}-e^{2\pi i \frac{M}{3B}y}\right)\alpha_{x,y}|x,y,\psi_{x,y}\rangle\right\|^2.
\end{aligned}$$

We bound the above quantity in two parts using the triangle inequality as follows:

$$\left\|\sum_{y\in Y\setminus\mathcal{F}(x)}\left(e^{2\pi i \frac{M}{3B}f(x_0+rx)}-e^{2\pi i \frac{M}{3B}y}\right)\alpha_{x,y}|x,y,\psi_{x,y}\rangle\right\|^2 \leq \sum_{y\in Y\setminus\mathcal{F}(x)}|2\alpha_{x,y}|^2 \leq \frac{1}{300};$$

$$\begin{aligned}
\left\|\sum_{y\in \mathcal{F}(x)}\left(e^{2\pi i \frac{M}{3B}f(x_0+rx)}-e^{2\pi i \frac{M}{3B}y}\right)\alpha_{x,y}|x,y,\psi_{x,y}\rangle\right\|^2 &\leq \sum_{y\in\mathcal{F}(x)}\left|\frac{2\pi i M}{3B}(f(x_0+rx)-y)\alpha_{x,y}\right|^2 \\
&\leq \sum_{y\in Y_x}\left|2\pi i\frac{M}{3B}\delta\right|^2|\alpha_{x,y}|^2 \\
&\leq \left|2\pi i\frac{M}{3B}\delta\right|^2 = \frac{1}{42^2}.
\end{aligned}$$

Thus for all $x\in G_m^n$ we have that

$$\left\|\mathrm{O}|x,0,0\rangle - U^\dagger(I\otimes \mathrm{cP}\otimes I)U|x,0,0\rangle\right\| \leq \sqrt{\frac{1}{300}+\frac{1}{42^2}} < \frac{1}{16}. \tag{5.14}$$

We can assume without loss of generality that our approximate phase oracle does not change the value of the input register. Otherwise we can just copy $|x\rangle$ to another register, then apply our approximate phase oracle on the second copy, then (approximately) erase the second copy of $|x\rangle$ using mod 2 bitwise addition with the first copy. Under this assumption by (5.14) we get that

$$\left\|\mathrm{O}|\psi\rangle - U^\dagger(I\otimes \mathrm{cP}\otimes I)U|\psi\rangle\right\| < \frac{1}{16}, \text{ for any quantum state } |\psi\rangle = \sum_{x\in G_m^n}\alpha_x|x,0,0\rangle. \tag{5.15}$$

From now on the proof is the same as the proof of Corollary 5.3.6. In that proof we showed that if we use the phase oracle O in Jordan's gradient computation algorithm, then we would get a gradient estimate where each individual coordinate has the required approximation quality with probability at least $\frac{2}{3}$. Equation (5.15) implies that if instead we use our approximate implementation of the phase oracle, $U^\dagger(I \otimes cP \otimes I)U$, then the outcome probability distribution changes by at most $\frac{1}{16}$ in total variation distance. So one run of Jordan's algorithm using this approximate phase oracle still outputs a vector $v \in \mathbb{R}^n$ such that

$$\Pr\left[\left|g_i - \frac{3B}{r}v_i\right| > \frac{8 \cdot 42\pi\delta}{r}\right] \leq \frac{1}{3} + \frac{1}{16} < \frac{2}{5} \text{ for every } i \in [n].$$

As in the proof of Corollary 5.3.6, repeating the whole procedure $\mathcal{O}\Big(\log(\frac{n}{\rho})\Big)$ times, and taking the median of the resulting vectors coordinatewise, gives a gradient approximator $\tilde{g}$ of the desired quality. The gate complexity analysis follows similarly to the proof of Theorem 4.5.5, noting that each controlled phase operation cP can be implemented using $\mathcal{O}\big(\log(\frac{B}{\delta})\big)$ single-qubit phase gates. $\square$

Chapter 6

Quantum algorithms for solving SDPs

Semidefinite Programs (SDPs) generalize Linear Programs (LPs), and have important applications in optimization, computer science and quantum information. They can be solved in polynomial time, but usually less efficiently than LPs. Following the first paper on quantum algorithms for SDP-solving by Brandão and Svore [BS17] in 2016, rapid developments have been made on quantum optimization algorithms. In this chapter we improve and generalize all prior quantum algorithms for SDP-solving and give a simpler and unified framework.

We take a new perspective on quantum SDP-solvers and introduce several new techniques. One of these is the quantum operator input model, which generalizes the different input models used in previous work, and arguably even any other reasonable input model. This new model assumes that the input matrices are provided as block-encodings. In this model we give an $\widetilde{\mathcal{O}}((\sqrt{m}+\sqrt{n}\gamma)\alpha\gamma^4)$ algorithm, where n is the size of the matrices, m is the number of constraints, γ is the reciprocal of the scale-invariant relative precision parameter (which we explain later), and α is a normalization factor of the input matrices. In particular for the standard sparse-matrix access, the above result gives a quantum algorithm where $\alpha = s$. We also improve on recent results of Brandão et al. [BKL$^+$18], who consider the special case when the input matrices are proportional to mixed quantum states that one can query. For this model Brandão et al. [BKL$^+$18] showed that the dependence on n can be replaced by a polynomial dependence on both the rank and the trace of the input matrices. We remove the dependence on the rank and hence require only a dependence on the trace of the input matrices.

After we obtain these results we show an application to the problem of shadow tomography, recently introduced by Aaronson [Aar18]. Here we simultaneously improve both the sample and computational complexity of the previous best results. Finally we prove a new $\tilde{\Omega}(\sqrt{m}\alpha\gamma)$ lower bound for solving LPs and SDPs in the quantum operator model, which also implies a lower bound for the model of Brandão et al. [BKL$^+$18].

This chapter is based on [vAG19] with additions from its precursor [vAGGdW17] © 2017 IEEE.

6.1 Introduction

In this chapter we consider *Semidefinite programs* (SDPs). SDPs have many applications in optimization, notable examples include approximation of NP-hard problems like MAXCUT [GW95] and polynomial optimization through the Sum-Of-Squares hierarchy [Las01, Par00]. SDPs have also found applications in quantum information theory. Examples include POVM measurement design [Eld03] and finding the maximal winning probability of non-local games [CHTW04], where the players can share entanglement.

We consider the basic (primal) form of an SDP which maximizes the objective function $\mathrm{Tr}[CX]$ over a matrix valued variable $X \in \mathbb{C}^{n\times n}$ as follows:

$$\begin{aligned} \mathrm{OPT} = \max \quad & \mathrm{Tr}[CX] \\ \text{s.t.} \quad & \mathrm{Tr}[A_j X] \leq b_j \quad \text{for all } j \in [m], \\ & X \succeq 0, \end{aligned} \tag{6.1}$$

where $[m] := \{1,\ldots,m\}$. The input to the problem consists of $n \times n$ Hermitian constraint matrices $A_1,\ldots,A_m$, an objective matrix C, and reals $b_1,\ldots,b_m$. For normalization purposes we assume $\|C\|, \|A_j\| \leq 1$. The number of constraints is m (we do not count the standard $X \succeq 0$ constraint for this). The variable X of this SDP is an $n \times n$ positive semidefinite (psd) matrix. We assume that $A_1 = I$ and $b_1 = R$, giving a known bound on the trace of a solution: $\mathrm{Tr}[X] \leq R$. A primal SDP of the above form (6.1) also has a *dual*, optimizing over $y \in \mathbb{R}^m$, which is the following SDP:

$$\begin{aligned} \mathrm{OPT} = \min \quad & b^T y \\ \text{s.t.} \quad & \sum_{j=1}^{m} y_j A_j - C \succeq 0, \\ & y \geq 0. \end{aligned} \tag{6.2}$$

We assume that the dual optimum is attained and that an explicit $r \geq 1$ is known such that at least one optimal dual solution y exists with $\|y\|_1 \leq r$. These assumptions imply that *strong duality* holds, i.e., the optimal value of the primal (6.1) and dual 6.2 SDPs equal. Finally not that Linear programs (LPs) correspond to the case where all constraint matrices are diagonal.

In this chapter we build on the observation that a normalized psd matrix can be naturally represented as a quantum state. Since operations on $\log(n)$-qubit quantum states can sometimes be cheaper to perform on a quantum computer than operations on classical descriptions of $n \times n$ matrices, this can give rise to faster algorithms for solving SDPs on a quantum computer [BS17].

We say an algorithm is an ε-approximate *quantum SDP-solver* if for all input numbers $g \in \mathbb{R}$ and $\zeta \in (0,1)$, with success probability $1-\zeta$, all of the following hold:

(i) The algorithm finds a vector $y' \in \mathbb{R}^{m+1}$ and a number $z \in \mathbb{R}$ defining its output

$$X := z \frac{e^{-\sum_{j=1}^{m} y'_j A_j + y'_0 C}}{\mathrm{Tr}\left[e^{-\sum_{j=1}^{m} y'_j A_j + y'_0 C}\right]}. \tag{6.3}$$

The output X is an *ε-feasible primal solution* with objective value at least $g-\varepsilon$, i.e.,

$$\forall j \in [m] \colon \mathrm{Tr}[XA_j] \leq b_j + \varepsilon,$$

and $\mathrm{Tr}[XC] \geq g - \varepsilon$. If the algorithm cannot find such an output X, then it correctly concludes that no feasible solution exist (if we set $\varepsilon = 0$).

(ii) The algorithm finds a $y \in \mathbb{R}^{m+1}$ that is an *ε-feasible solution* to the dual problem with objective value at most $g + \varepsilon$, i.e.,

$$\sum_{j=1}^{m} y_j A_j - C \succeq -\varepsilon I, \tag{6.4}$$

and $b^T y \leq g + \varepsilon$, or it correctly concludes that no feasible y exists (if we set $\varepsilon = 0$).

(iii) The algorithm determines whether $\mathrm{OPT} \leq g - \varepsilon$ or $\mathrm{OPT} \geq g + \varepsilon$. If $\mathrm{OPT} \in (g-\varepsilon, g+\varepsilon)$ then it may output either. (Note that this essentially follows from (i)-(ii).)

Notice that this solves a decision version of the problem. However, we can easily find an approximation of OPT using binary search on g if we have an ε-approximate SDP-solver. Since ε is scale depended we actually care about the dependence on the scale invariant parameter $\gamma := Rr/\varepsilon$. An algorithm that only satisfies (i) will be called an *ε-approximate SDP primal oracle.* For such an algorithm the relevant scale invariant parameter is $\gamma := R/\varepsilon$. Due to the form of the objective value constraint in the first point, and to simplify statements like (6.3), we write $A_0 := -C$ and $b_0 := -g$.

6.2 SDP-solving frameworks

In this section we present two frameworks for SDP-solving. First we present an algorithm to implement a primal oracle, and then the Arora-Kale framework, which is used for finding a good approximation of the optimal value and an almost feasible solution to the dual. These together implement a full SDP-solver.

Both frameworks have a very similar iterative structure, with iteration count $\widetilde{\mathcal{O}}_n(\gamma^2)$, where $1/\gamma$ is the relevant scale-invariant precision parameter (as we will see the value of γ differs by a factor r in the two cases). The main difference is that the iterative step of the primal oracle framework requires only a simple search, whereas in the Arora-Kale framework one needs to solve a slightly more complex task. Both algorithms start with $y = 0$, and in each step only a constant number of indices of y are incremented, thus in both cases we will work with a y vector that is non-negative and $\widetilde{\mathcal{O}}_n(1/\theta^2)$-sparse.

6.2.1 An SDP primal oracle

For the primal oracle we use the same algorithm as Brandão et al. [BKL+18] following the proof of Lemma 4.6 of Lee, Raghavendra and Steurer [LRS15]. A few small reductions are required to apply this technique. To be able to work with density operators instead of X, the b_js in the constraints $1 \dots m$ are scaled down by a factor R, such that every solution X' to the new SDP has trace at most 1. Then, we add one new variable denoted by ω such that

$$\rho := \begin{bmatrix} X' & 0 \\ 0 & \omega \end{bmatrix}.$$

Now $\mathrm{Tr}[\rho] = 1$ and $\rho \succeq 0$ imply that $\mathrm{Tr}[X'] \leq 1$, and we get a new SDP that is equivalent to the previous one. It can be shown that in our input models this reduction does not introduce more than a constant factor overhead in the complexity; note that this reduction also illustrates that for an SDP primal oracle $\frac{\varepsilon}{R}$ is the relevant scale-invariant precision parameter.

The following meta-algorithm for an SDP primal oracle assumes access to (some form of) a description of the SDP after the above-discussed reduction, and provides an output as in Eq. (6.3) of (i)

1. Let $y = 0 \in \mathbb{R}^{m+1}$ and $\theta = \frac{\varepsilon}{2R}$.
2. Repeat $\frac{\ln(n)}{\theta^2}$ times the following:
    (a) Define $\rho := e^{-\sum_{j=0}^m y_j A_j} / \mathrm{Tr}\left[e^{-\sum_{j=0}^m y_j A_j}\right]$.
    (b) Find an index j such that $\mathrm{Tr}[A_j \rho] \geq b_j$ or conclude correctly that for all j, $\mathrm{Tr}[A_j \rho] \leq b_j + \theta$.
    (c) If no j is found, then we are done and output y and $z = R\mathrm{Tr}[X']$, where $\mathrm{Tr}[X']$ is the probability[1] of measuring ρ to be in the subspace corresponding to the variable X'.
    (d) Otherwise update $y \leftarrow y + \theta e_j$.
3. Conclude that there is no solution for $\theta = 0$.

[1]Note that a θ-approximation of $\mathrm{Tr}[X']$ is easy to compute by means of amplitude estimation if ρ can be efficiently prepared as a quantum state – which is the case in our algorithms.

6.2.2 The Arora-Kale framework

Similarly to previous work [BS17] we build our results on the Arora-Kale framework. For a detailed description see the original paper by Arora and Kale [AK16]. For our application, the following broad overview suffices.[2]

Now we describe the Arora-Kale meta-algorithm. This algorithm assumes access to (some form of) a description of the SDP, such that the first constraint is $\mathrm{Tr}[X] \leq R$, i.e., $A_1 = I$ and $b_1 = R$. It provides an output as in Eq. (6.4) of (ii). (Remember that we set $A_0 = -C$ and $b_0 = -g$.)

1. Let $y = 0 \in \mathbb{R}^{m+1}$ and set $\theta = \frac{\varepsilon}{6Rr}$.
2. Repeat $\frac{\ln(n)}{\theta^2}$ times the following:
   (a) Define $\rho := e^{-\sum_{j=0}^{m} y_j A_j} / \mathrm{Tr}\left[e^{-\sum_{j=0}^{m} y_j A_j}\right]$.
   (b) Find a $\tilde{y}$ in the polytope
   $$\begin{aligned}\mathcal{P}_\delta(\rho) := \Big\{\tilde{y} \in \mathbb{R}^{m+1} :\ & b^T \tilde{y} \leq 0,\\ & \sum_{j=0}^{m} \tilde{y}_j \mathrm{Tr}[A_j \rho] \geq -\delta,\\ & \tilde{y} \geq 0,\ \tilde{y}_0 = \frac{1}{2r},\ \|\tilde{y}\|_1 \leq 1\Big\}.\end{aligned}$$
   for $\delta = \theta$; if cannot find such a $\tilde{y}$ then conclude that none exists for $\delta = 0$.
   (c) If no such $\tilde{y}$ exists, then conclude that OPT $> g$ and stop.
   (d) If such a $\tilde{y}$ exists, then update $y \leftarrow y + \theta \tilde{y}$.
3. Conclude OPT $\leq g + \varepsilon$ and output $\frac{2r\theta}{\ln(n)} y + \frac{\varepsilon}{R} e_1 - e_0$ as a dual solution.

Note that in the above meta-algorithm, up to a constant factor, the θ parameter is essentially $\gamma^{-1} = \frac{rR}{\varepsilon}$, illustrating that γ^{-1} is the relevant scale-invariant precision parameter for SDP-solving, for a more detailed discussion see [vAGGdW17].

Brandão and Svore [BS17] observed that $\rho := e^{-\sum_j y_j A_j} / \mathrm{Tr}\left[e^{-\sum_j y_j A_j}\right]$ is a quantum Gibbs state and this state can be prepared efficiently on a quantum computer, allowing fast trace estimation, in particular resulting in a quadratic speedup in n compared to classical methods.

A procedure that solves step (b) is called a *θ-oracle*, not to be confused with the input oracles. In the rest of this chapter we will assume that the cost of updating the y vector is no more than the cost of a θ-oracle call.

Below we give an oracle implementation that always outputs a 3-sparse $\tilde{y}$. The oracle is constructed using a geometric argument that boils down to minimizing over m angles, one for each constraint. Each angle is easily computed from the corresponding b_j and $\mathrm{Tr}[A_j \rho]$, where a $\frac{\theta}{2}$-additive error is allowed in the ap-

[2]For more discussion on general (quantum) SDP-solvers along this line, see [vAGGdW17].

proximation of $\mathrm{Tr}[A_j\rho]$. We perform this minimization using quantum minimum finding [DH96], or more precisely its generalization (Theorem 3.A.3), allowing for a quadratic speedup in m. Previously Brandão and Svore [BS17] applied other techniques to similarly get a quadratic speedup in m but this introduced a much worse dependence on Rr/ε.

6.2.3 An efficient 3-sparse oracle

For all $j \in \{0,1,\dots,m\}$ let $a_j := \mathrm{Tr}[A_j\rho]$ and let $\tilde{a}_j \in \mathbb{R}$ such that $|\tilde{a}_j - a_j| \leq \theta/2$, for some fixed $\theta \geq 0$. For convenience in this subsection we will only work with m-dimensional vectors, and will think about the vectors $b, y \in \mathbb{R}^m$ with the 0-th coordinate of the corresponding $(m+1)$-dimensional vectors omitted. Let $\tilde{a} := (\tilde{a}_1,\dots,\tilde{a}_m)$ and $c' := \frac{\tilde{a}_0}{2r} - \frac{\theta}{2}$, we define an approximate version of the polytope from the Arora-Kale framework:

$$\tilde{\mathcal{P}}(\tilde{a},c') := \Big\{ y \in \mathbb{R}^m : \quad \begin{aligned} b^T y &\leq \frac{g}{2r} =: \alpha, \\ \sum_{j=1}^m \tilde{a}_j y_j &\geq \frac{\tilde{a}_0}{2r} - \frac{\theta}{2} = c', \\ y &\geq 0, \\ \sum_{j=1}^m y_j &\leq 1 - \frac{1}{2r} \end{aligned} \Big\}.$$

Observe that
$$(y_0, y_1, \dots, y_m) \in \mathcal{P}_0(\rho) \Longrightarrow (y_1,\dots,y_m) \in \tilde{\mathcal{P}}(\tilde{a},c'), \tag{6.5}$$
moreover
$$(y_1,\dots,y_m) \in \tilde{\mathcal{P}}(\tilde{a},c') \Longrightarrow (1/(2r), y_1,\dots,y_m) \in \mathcal{P}_\theta(\rho). \tag{6.6}$$

Therefore constructing a 3-sparse oracle for the Arora-Kale framework boils down to finding a 2-sparse vector in $\tilde{\mathcal{P}}(\tilde{a},c')$.

We first describe our quantum algorithm for finding a 2-sparse vector in $\tilde{\mathcal{P}}(\tilde{a},c')$ assuming access to a unitary which acts as $|j\rangle|0\rangle|0\rangle \mapsto |j\rangle|\tilde{a}_j\rangle|\psi_j\rangle$, where $|\psi_j\rangle$ is some workspace state depending on j. We then briefly discuss how to modify the analysis when we are given an oracle which acts as $|j\rangle|0\rangle|0\rangle \mapsto |j\rangle\sum_i \beta_j^i|\tilde{a}_j^i\rangle|\psi_j^i\rangle$ (where each $\tilde{a}_j^i$ is an additive θ-approximation to $\mathrm{Tr}[A_j\rho]$), since this is the output of the trace-estimation procedure of the previous section.

If $\alpha \geq 0$ and $c' \leq 0$, then $y = 0$ is a solution and our oracle can return it. If not, then we may write $y = Nq$ with $N = \|y\|_1 > 0$ and hence $\|q\|_1 = 1$. So we are looking for an N and a q such that

$$\begin{aligned} b^T q &\leq \alpha/N \\ \tilde{a}^T q &\geq c'/N \\ \|q\|_1 &= 1 \\ q &\geq 0 \\ 0 < N &\leq 1 - \frac{1}{2r} \end{aligned} \tag{6.7}$$

We can now view $q \in \mathbb{R}_+^m$ as the coefficients of a convex combination of the points $p_i = (b_i, \tilde{a}_i)$ in the plane. We want such a combination that lies to the upper left of $g_N = (\alpha/N, c'/N)$ for some $0 < N \leq 1 - \frac{1}{2r}$. Let $\mathcal{G}_N$ denote the upper-left quadrant of the plane starting at g_N.

6.2.1. LEMMA. *If there is a $y \in \tilde{\mathcal{P}}(\tilde{a}, c')$, then[3] there is a 2-sparse $y' \in \tilde{\mathcal{P}}(\tilde{a}, c')$ such that $\|y\|_1 = \|y'\|_1$.*

Proof:
Consider $p_i = (b_i, \tilde{a}_i)$ and $g = (\alpha/N, c'/N)$ as before, and write $y = Nq$ where $\sum_{j=1}^m q_j = 1$, $q \geq 0$. The vector q certifies that a convex combination of the points p_i lies in $\mathcal{G}_N$. But then there exist $j, k \in [m]$ such that the line segment $\overline{p_j p_k}$ intersects $\mathcal{G}_N$. All points on this line segment are convex combinations of p_j and p_k, hence there is a convex combination of p_j and p_k that lies in $\mathcal{G}_N$. This gives a 2-sparse q', and $y' = Nq' \in \tilde{\mathcal{P}}(\tilde{a}, c')$. □
We can now restrict our search to 2-sparse y. Let $\mathcal{G} = \bigcup_{N \in (0, 1 - \frac{1}{2r}]} \mathcal{G}_N$. Then we want to find two points p_j, p_k that have a convex combination in $\mathcal{G}$, since this implies that a scaled version of their convex combination gives a $y \in \tilde{\mathcal{P}}(\tilde{a}, c')$ and $\|y\|_1 \leq r$.

6.2.2. LEMMA. *There is an oracle that returns a 2-sparse vector $y \in \tilde{\mathcal{P}}(\tilde{a}, c')$, if one exists, using one search and two minimizations over the m points $p_j = (b_j, \tilde{a}_j)$. This gives a classical algorithm that uses $\mathcal{O}(m)$ calls to the subroutine that gives the entries of $\tilde{a}$ and $\mathcal{O}(m)$ other operations, and a quantum algorithm that (in order to solve the problems with high probability) uses $\mathcal{O}(\sqrt{m})$ calls to the subroutine that gives the entries of $\tilde{a}$ and $\widetilde{\mathcal{O}}(\sqrt{m})$ other gates.*

Proof:
The algorithm can be summarized as follows:

1. Check if $\alpha \geq 0$ and $c' \leq 0$. If so, output $y = 0$.
2. Check if there is a $p_i \in \mathcal{G}$. If so, let $q = e_i$ and $N = \frac{c'}{\tilde{a}_i}$.
3. Find p_j, p_k so that the line segment $\overline{p_j p_k}$ goes through $\mathcal{G}$. This gives coefficients q of a convex combination that can be scaled by $N = \frac{c'}{\tilde{a}^T q}$ to give y. The main realization is that we can search separately for p_j and p_k.

First we will need a better understanding of the shape of $\mathcal{G}$ (see Figure 6.1 for illustration). This depends on the sign of α and c'. If we define $\mathrm{sign}(0) := 1$, and $v := (\alpha/(1 - \frac{1}{2r}), c'/(1 - \frac{1}{2r}))$, we get the following intuitive characterizations:

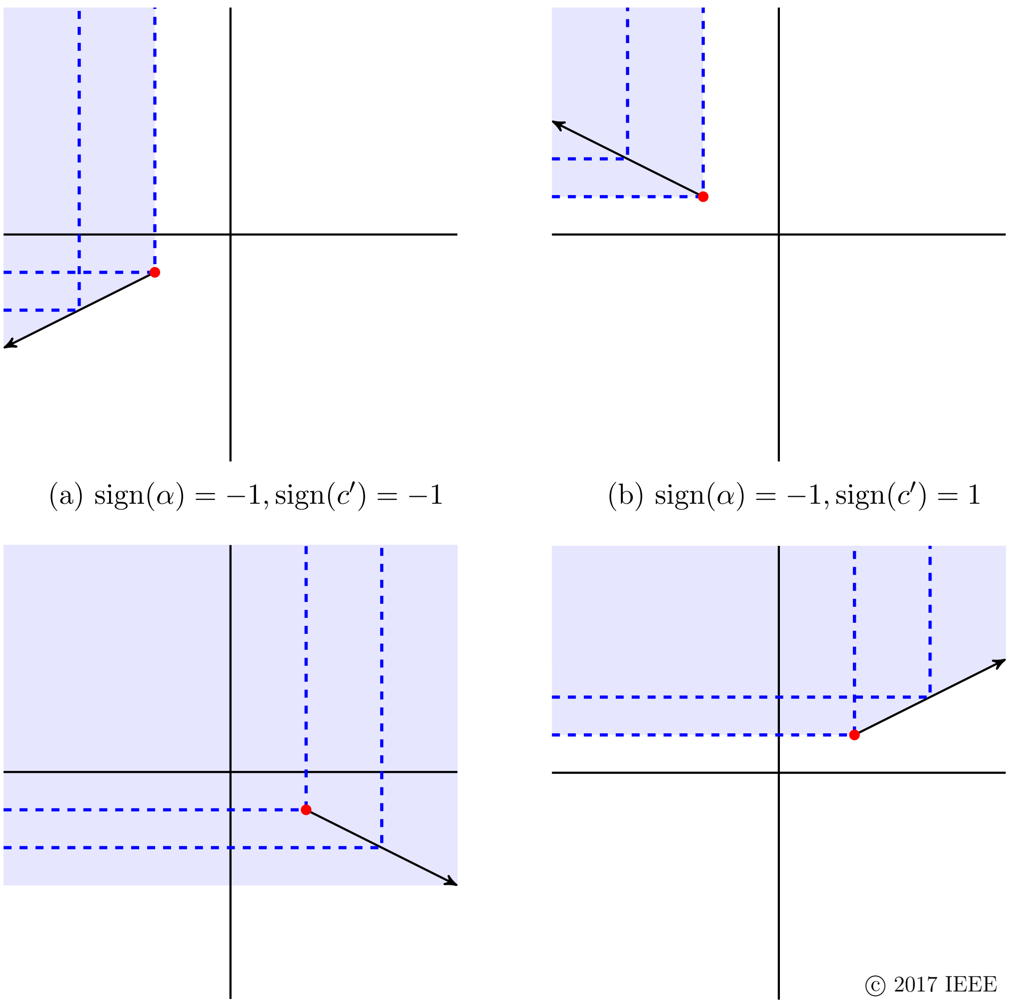
[3]This is statement is essentially a corollary of Carathéodory's theorem in convex geometry.

(a) If $\text{sign}(\alpha) = -1, \text{sign}(c') = -1$. The corner point of $\mathcal{G}$ is v. One edge goes up vertically and another follows the line segment $\lambda \cdot v$ for $\lambda \in [1, \infty)$ starting at the corner.

(b) If $\text{sign}(\alpha) = -1, \text{sign}(c') = 1$. Here $\mathcal{G}_N \subseteq \mathcal{G}_{1-\frac{1}{2r}}$ for $N \leq 1 - \frac{1}{2r}$. So $\mathcal{G} = \mathcal{G}_{1-\frac{1}{2r}}$. The corner point is again v, but now one edge goes up vertically and one goes to the left horizontally.

(c) If $\text{sign}(\alpha) = 1, \text{sign}(c') = -1$. This is the case where $y = 0$ is a solution, $\mathcal{G}$ is the whole plane and has no corner.

(d) If $\text{sign}(\alpha) = 1, \text{sign}(c') = 1$. The corner point of $\mathcal{G}$ is again v. From there one edge goes to the left horizontally and one edge follows the line segment $\lambda \cdot v$ for $\lambda \in [1, \infty)$.



(a) $\text{sign}(\alpha) = -1, \text{sign}(c') = -1$

(b) $\text{sign}(\alpha) = -1, \text{sign}(c') = 1$

(c) $\text{sign}(\alpha) = 1, \text{sign}(c') = -1$

(d) $\text{sign}(\alpha) = 1, \text{sign}(c') = 1$

Figure 6.1. The region $\mathcal{G}$ in light blue. The borders of two quadrants $\mathcal{G}_N$ have been drawn by thick dashed blue lines. The red dot at the beginning of the arrow is the point $v = (\alpha/(1 - \frac{1}{2r}), c'/(1 - \frac{1}{2r}))$.

Since $\mathcal{G}$ is always an intersection of at most 2 halfspaces, steps 1-2 of the algorithm are easy to perform. In step 1 we handle case (c) by simply returning $y = 0$. For the other cases v is the corner point of $\mathcal{G}$ and the two edges are simple lines. Hence in step 2 we can easily search through all the points to find out if there is one lying in $\mathcal{G}$; since $\mathcal{G}$ is a very simple region, this only amounts to checking on which side of the two lines a point lies.

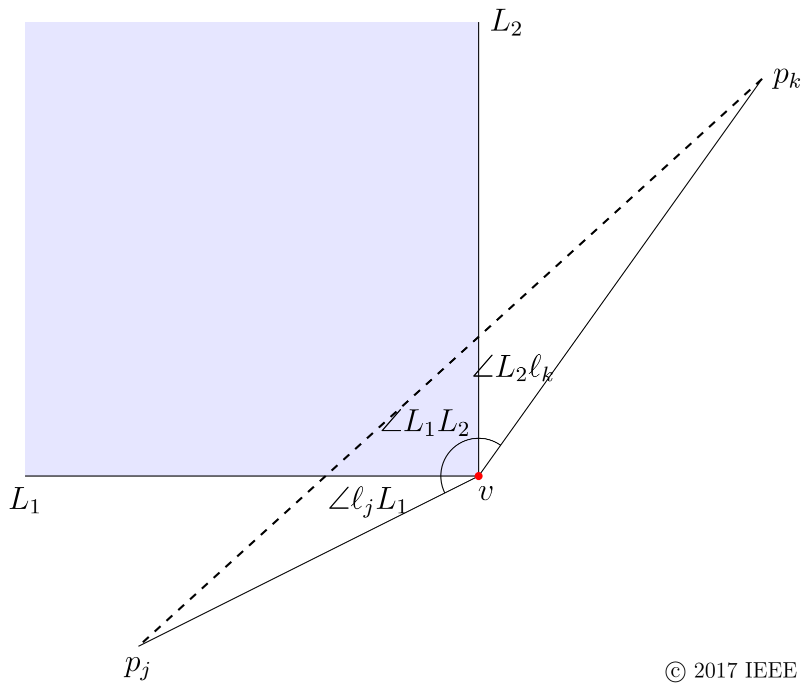


Figure 6.2. Illustration of $\mathcal{G}$ with the points p_j, p_k and the angles $\angle\ell_jL_1, \angle L_1L_2, \angle L_2\ell_k$ drawn in. Clearly the line $\overline{p_jp_k}$ only crosses $\mathcal{G}$ when the total angle is less than π.

Now, if we cannot find a single point in $\mathcal{G}$, we need a combination of two points in step 3. Let L_1, L_2 be the edges of $\mathcal{G}$ and let ℓ_j and ℓ_k be the line segments from v to p_j and p_k, respectively. Then, as can be seen in Figure 6.2, the line segment $\overline{p_jp_k}$ goes through $\mathcal{G}$ if and only if (up to relabeling p_j and p_k) $\angle\ell_jL_1 + \angle L_1L_2 + \angle L_2\ell_k \leq \pi$. Since $\angle L_1L_2$ is fixed, we can simply look for a j such that $\angle\ell_jL_1$ is minimized and a k such that $\angle L_2\ell_k$ is minimized. If $\overline{p_jp_k}$ does not pass through $\mathcal{G}$ for this pair of points, then it does not for any of the pairs of points.

Notice that these minimizations can be done separately and hence can be done in the stated complexity. Given the minimizing points p_j and p_k, it is easy to check if they give a solution by calculating the angle between ℓ_j and ℓ_k. The coefficients of the convex combination q are then easy to compute. It remains to compute the scaling N. This is done by rewriting the constraints of (6.7):

$$\frac{c'}{q^T\tilde{a}} \leq N \leq \frac{\alpha}{q^T b}$$

So we can pick any value in this range for N. $\square$

The analysis above applies if we are given an oracle which acts as $|j\rangle|0\rangle|0\rangle \mapsto |j\rangle|\tilde{a}_j\rangle|\psi_j\rangle$. However, our trace estimation procedure acts as

$$|j\rangle|0\rangle|0\rangle \mapsto |j\rangle \sum_i \beta_j^i |\tilde{a}_j^i\rangle|\psi_j^i\rangle$$

where each $|\tilde{a}_j^i\rangle$ is an approximation of a_j and the amplitudes β_j^i are such that measuring the second register with high probability returns an $\tilde{a}_j^i$ which is $\frac{\theta}{2}$-close to a_j. Since we can exponentially reduce the probability that we obtain an $\tilde{a}_j^i$ which is further than $\frac{\theta}{2}$ away from a_j, we will for simplicity assume that for all i, j we have $|\tilde{a}_j^i - a_j| \leq \frac{\theta}{2}$; the neglected exponentially small probabilities will only affect the analysis in negligible ways. Note that while we do not allow our oracle enough time to obtain classical descriptions of all $\tilde{a}_j$s (we aim for a runtime of $\widetilde{\mathcal{O}}(\sqrt{m})$), we do have enough time to compute $\tilde{a}_0$ and thus c' once initially. Knowing c', we can compute the angles defined by the points $(b_j, \tilde{a}_j^i)$ with respect to the corner point of $(\alpha/(1-\frac{1}{2r}), c'/(1-\frac{1}{2r}))$ and the lines L_1, L_2 (see Figure 6.2). We now apply our generalized minimum-finding algorithm with runtime $\widetilde{\mathcal{O}}(\sqrt{m})$ (see Theorem 3.A.3) starting with a uniform superposition over the js to find $\ell, k \in [m]$ with points $(\tilde{a}_\ell, b_\ell)$ and $(\tilde{a}_k, b_k)$ approximately minimizing the respective angles to lines L_1, L_2. It follows from Eq. (6.5) and Lemma 6.2.1 that if $\mathcal{P}_0(\rho)$ is non-empty, then some convex combination of $(\tilde{a}_\ell, b_\ell)$ and $(\tilde{a}_k, b_k)$ lies in $\mathcal{G}$. On the other hand, if $\mathcal{P}_\theta(\rho)$ is empty, then by Eq. (6.6) and Lemma 6.2.1 the respective angles will be such that we correctly conclude that $\mathcal{P}_0(\rho)$ is empty.

6.3 High-level overview of the quantum algorithms with some conceptual improvements

6.3.1 Input models & subroutines

We will consider three input models: the *sparse matrix model*, the *quantum state model*, and the *quantum operator model*. The first two models were already studied in previous work. The quantum operator model is a common generalization of the other two models, and in fact any other reasonable model for SDPs, as we will show later.

In all models we assume quantum oracle access to the numbers b_j via the input oracle O_b satisfying[4] for all $j \in [m]$:

$$O_b|j\rangle|0\rangle = |j\rangle|b_j\rangle.$$

For all input oracles, we assume that we can apply both the oracle and its inverse[5] in a controlled fashion.

Sparse matrix model. In the *sparse matrix model* the input matrices are assumed to be s-row sparse for a known bound $s \in [n]$, meaning that there are at most s non-zero elements per row. The model is close to the classical model for sparse matrices. Access to the A_j matrices is provided by two oracles, similar to previous work on Hamiltonian simulation in [BCK15]. The first of the two oracles is a unitary O_{sparse}, which serves the purpose of sparse access. This oracle calculates the $\mathbf{index} : [m] \times [n] \times [s] \to [n]$ function, which for input (j, k, ℓ) gives the column index of the ℓth non-zero element in the kth row of A_j. We assume this oracle computes the index “in place":

$$O_{\text{sparse}}|j, k, \ell\rangle = |j, k, \mathbf{index}(j, k, \ell)\rangle. \tag{6.8}$$

(In the degenerate case where the kth row has fewer than ℓ non-zero entries, $\mathbf{index}(j, k, \ell)$ is defined to be ℓ together with some special symbol indicating this case.)

We also need another oracle O_A, returning a bitstring[4] representation of $(A_j)_{ki}$ for every $j \in [m]$ and $k, i \in [n]$:

$$O_A|j, k, i, z\rangle = |j, k, i, z \oplus (A_j)_{ki}\rangle. \tag{6.9}$$

This model corresponds directly to a classical way of accessing sparse matrices. In contrast, the following *quantum state model* is inherently quantum and has no classical counterpart for SDPs. In order to introduce this other input model we need the following definition:

6.3.1. Definition (Subnormalized density operators & Purification). A *subnormalized density operator* ϱ is a psd matrix of trace at most 1.

A *purification* of a subnormalized density operator ϱ is a 3-register pure state such that tracing out the third register[6] and projecting on the subspace where the second register is $|0\rangle$ yields ϱ.

We write “ϱ” and “ς” for subnormalized density operators to distinguish them from normalized density operators, for which we write “ρ” and “σ”.

[4]For simplicity we assume the bitstring representation has at most $\mathcal{O}(\log(nmRr/\varepsilon))$ bits.

[5]When we talk about samples, e.g. in Section 6.5, then we do not assume we can apply the inverse operation.

[6]For simplicity we assume that for a d-dimensional density operator a purification has at most polylog(d) qubits.

Quantum state model. In this model we assume that each A_j has a fixed decomposition of the form

$$A_j = \mu_j^+ \varrho_j^+ - \mu_j^- \varrho_j^- + \mu_j^I I$$

for (subnormalized) density operators $\varrho_j^\pm$, non-negative reals $\mu_j^\pm$ and real number $\mu_j^I \in \mathbb{R}$. We assume access to an oracle O_μ that takes as input an index j and outputs binary representations[4] of μ_j^+, μ_j^- and μ_j^I.

Furthermore we assume access to a state-preparing oracle $O_{|\cdot\rangle}$ that prepares purifications[6] $|\psi_j^\pm\rangle$ of $\varrho_j^\pm$:

$$O_{|\cdot\rangle}|j\rangle|\pm\rangle|0\rangle = |j\rangle|\pm\rangle|\psi_j^\pm\rangle.$$

Finally we assume that a bound $B \in \mathbb{R}_+$ is known such that

$$\forall j : \mu_j^+ + \mu_j^- \leq B.$$

Note that a tight upper bound B can easily be found w.h.p. using $\mathcal{O}(\sqrt{m})$ quantum queries to O_μ by means of maximum-finding [DH96], see also Appendix 3.A.

Quantum operator model. Motivated by recent work [LC16] and Chapter 3 we propose a new input model that we call the *quantum operator model.* In this model the input matrices are given by a unitary that implements block-encodings of the input matrices.

In the quantum operator model we assume access to an oracle O_U that acts as follows:

$$O_U|j\rangle|\psi\rangle = |j\rangle(U_j|\psi\rangle).$$

Where U_j is an a-qubit block-encoding[7] of A_j/α, for some fixed[8] $\alpha \in \mathbb{R}$ and $a = \mathcal{O}(\log(nmRr/\varepsilon))$.

As we have seen in Chapter 3, the sparse input model can be reduced to the quantum operator model with $\alpha = s$ (Lemma 3.3.5) and we will also show that the quantum state model can be reduced to the quantum operator model with $\alpha = B$. In Chapter 3, we have shown several other ways to construct block-encodings, e.g., based on Hamiltonian simulation [LC17a] (Lemma 3.4.18), or for matrices stored in a QROM data structure [KP17b, CGJ19] (Lemma 3.3.7). In Lemma 3.3.3 we have also shown that if one can perform a POVM measurement on a quantum computer with a measurement operator M, then one can also implement a block-encoding of M, and use it as an input matrix in our operator model. Thus if we can perform a measurement corresponding to $A_j \succeq 0$ using a ancilla qubits, i.e.,

[7]If $n \in (2^{w-1}, 2^w)$ is not a power of 2, then we can simply extend A_j to a w-qubit operator, by defining it to be zero on the additional $2^w - n$ dimensions.

[8]Having a single normalization parameter α is not a serious restriction as it is easy to make a block-encoding more subnormalized so that every A_j gets the same normalization, cf. Lemma 3.3.9.

accept a state ρ with probability $\mathrm{Tr}[A_j\rho]$, then we can implement a $(1, a+1, 0)$-block-encoding of A_j. Hence, the quantum operator input model is a common generalization of all reasonable input models for SDPs, since at the very least an input model should allow one to approximately compute $\mathrm{Tr}[A_jX]$.

Computational cost. We analyze the query complexity of algorithms and subroutines, i.e., the number of queries to controlled versions of the input oracles and their inverses. We will denote the optimal quantum query complexity of an ε-approximate *quantum SDP-solver* with success probability 2/3 by $T_{SDP}(\varepsilon)$ (this is a "meta quantity", which becomes concrete once the input model is specified). We only consider success probability 2/3 to simplify the notation and proofs. However in all cases an ε-approximate SDP-solver with success probability $1-\zeta$ can easily be constructed using $\mathcal{O}(\log(1/\zeta)T_{SDP}(\varepsilon))$ queries.

In our algorithms we will assume access to a quantum-read/classical-write RAM (QCRAM), and assume one read/write operation has a constant gate complexity[9]; the size of the QCRAM will typically be $\widetilde{\mathcal{O}}_{n,m}\left(\left(\frac{Rr}{\varepsilon}\right)^2\right)$ bits. Most often in our results the number of non-query elementary operations, i.e., two-qubit gates and QCRAM calls, matches the query complexity up to polylog factors. In particular, if not otherwise stated, in our results a T-query quantum algorithm uses at most $\widetilde{\mathcal{O}}_{n,m}(T)$ elementary operations.

Subroutines. We will work with two major subroutines which need to be implemented according to the specific input model. First, the algorithms will require an implementation of a Gibbs-sampler.

6.3.2. DEFINITION (Gibbs-sampler). A θ-precise *Gibbs-sampler* on bounded input vectors $y \in \mathbb{R}^{m+1}_{\geq 0}$ is a quantum circuit that works under the promise that the support of y has size at most d, and $\|y\|_1 \leq K$. It takes as input a data structure storing the vector y, and for any input satisfying the promise, it creates as output a purification of a θ-approximation of the Gibbs state

$$e^{-\sum_{j=0}^m y_jA_j}/\mathrm{Tr}\left[e^{-\sum_{j=0}^m y_jA_j}\right].$$

The minimum cost of such a circuit is denoted by $T_{Gibbs}(K, d, 4\theta)$ (this is again a "meta quantity", which becomes concrete once the input model is specified).

For technical reasons we also allow Gibbs-samplers that require a random classical input seed $S \in \{0,1\}^a$ for some $a = \mathcal{O}(\log(1/\theta))$. In this case the output should be a θ-approximation of the Gibbs state with high probability ($\geq 4/5$) over a uniformly random input seed S.

[9]Note that read/write operations of a QRAM or QCRAM of size S can be implemented using $\widetilde{\mathcal{O}}(S)$ two-qubit gates, so this assumption could hide a factor in the gate complexity which is at most $\widetilde{\mathcal{O}}(S)$.

We will use the approximate Gibbs states in order to estimate the quantity $\mathrm{Tr}[A_j\rho]$.

6.3.3. DEFINITION (Trace estimator). A (θ,σ)-*trace estimator* is a quantum circuit that as input takes a quantum state ρ and index j. It outputs a sample from a random variable $Z_j \in \mathbb{R}$ which is an estimator of $\mathrm{Tr}[A_j\rho]$ with bias at most $\theta/4$:

$$|\mathrm{Tr}[A_j\rho] - \mathbb{E}[Z_j]| \leq \theta/4,$$

and the standard deviation of Z_j is at most σ. We write $T^{\sigma}_{Tr}(\theta)$ for the minimum cost of such a circuit (this is again a "meta quantity" as in the above definition).

6.3.2 Prior work

Classical SDP-solvers roughly fall into two categories: those with logarithmic dependence on R, r and $1/\varepsilon$, and those with polynomial dependence on these parameters but better dependence on m and n. In the first category the best known algorithm [LSW15] at the time of writing has complexity

$$\widetilde{\mathcal{O}}_{Rr/\varepsilon}\big(m(m^2 + n^{\omega} + mns)\big).$$

where $\omega \in [2, 2.38]$ is the yet unknown exponent of matrix multiplication.

In the second category Arora and Kale [AK16] gave an alternative framework for solving SDPs, using a matrix version of the "multiplicative weights update" method. Their framework can be tuned for specific types of SDPs, allowing for nearly linear-time algorithms, as shown by the example of the Goemans-Williamson SDP for the approximation of the maximum cut in a graph [GW95].

In 2016 Brandão and Svore [BS17] used the Arora-Kale framework to implement a general quantum SDP-solver in the sparse matrix model. They observed that the matrix

$$\rho := \frac{e^{-\sum_{j=0}^{m} y_j A_j}}{\mathrm{Tr}\left[e^{-\sum_{j=0}^{m} y_j A_j}\right]},$$

that is used for calculations in the Arora-Kale framework is in fact a $\log(n)$-qubit Gibbs state and can be efficiently prepared as a quantum state on a quantum computer. Using this they achieved a quantum speedup in terms of n. Combining this with a Grover-like speedup allowed for a speedup in terms of m as well, leading to an ε-approximate quantum SDP-solver with complexity

$$\widetilde{\mathcal{O}}\left(\sqrt{mn}s^2\left(\frac{Rr}{\varepsilon}\right)^{32}\right).$$

They also showed an $\Omega(\sqrt{m}+\sqrt{n})$ quantum query lower bound for solving SDPs when all other parameters are constant. This left as open question whether a better lower bound, matching the $\sqrt{mn}$ upper bound, could be found. The upper bound for the sparse input model was subsequently improved by van Apeldoorn et al. [vAGGdW17] to

$$\widetilde{\mathcal{O}}\left(\sqrt{mn}s^2\left(\frac{Rr}{\varepsilon}\right)^8\right).$$

Van Apeldoorn et al. also gave an $\Omega(\sqrt{\max(n,m)}\min(n,m)^{3/2})$ lower bound, albeit for non-constant parameters R and r. This bound implies that there is no general quantum SDP-solver that has a $o(nm)$ dependence on n and m and logarithmic dependence on R, r and $1/\varepsilon$. They also showed that every SDP-solver whose efficiency relies on outputting sparse dual solutions (including their algorithm and that of Brandão and Svore [BS17]) is limited, since problems with a lot of symmetry (like maxflow-mincut) in general require non-sparse dual solutions. Furthermore, they showed that for many combinatorial problems (like MAXCUT) R and r increase linearly with n and m.

Recently Brandão et al. [BKL⁺17a] gave an improved SDP-solver for the quantum state input model[10] that has a complexity bound with logarithmic dependence on n:

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}_n\left(\sqrt{m}\ \mathrm{poly}\left(\frac{Rr}{\varepsilon}, B, \max_{j\in\{0,\ldots,m\}}[\mathrm{rank}(A_j)]\right)\right).$$

Brandão et al. also applied their algorithm to the problem of *shadow tomography*, giving the first non-trivial application of a quantum SDP-solver.

Subsequently these results where further improved by the introduction of the Fast Quantum OR lemma by the same authors [BKL⁺18]. Approaches prior to [BKL⁺18] searched for a violated constraint in the SDP using Grover-like techniques, thereby multiplying the complexities of Gibbs-sampling and searching. The Fast Quantum OR lemma can be used to separate the search phase from the initial Gibbs-state preparation phase. This led to the improved complexity bound [BKL⁺18] of

$$\widetilde{\mathcal{O}}_n\left(\left(\sqrt{m} + \mathrm{poly}(\max_{j\in\{0\ldots m\}}[\mathrm{rank}(A_j)])\right)\ \mathrm{poly}\left(\frac{Rr}{\varepsilon}, B\right)\right).$$

We thank the authors of [BKL⁺18] for sending us an early draft of their paper, that introduced the fast quantum OR lemma and applied it to the quantum state model. This enabled us to work on some of the related improvements; during the correspondence, the application of the fast OR lemma to the sparse matrix model was independently suggested by Brandão et al. [Wu17] and by us.

[10]This model was already introduced in the first version of [BS17] together with a similar complexity statement, but there were some unresolved issues in the proof, that were only fixed by the contributions of [BKL⁺17a].

6.3.3 Our results

In this chapter we present multiple results. The main contribution consists of multiple improvements to the algorithms for SDP-solving, based on combining various recent quantum algorithmic developments. Although some of these improvements require quite technical proofs, they come from simple new perspectives and ideas, often combining previous works in new ways. We also apply the resulting algorithms to a few problems in convex optimization. Finally, we prove a new lower bound that fits the novel input models that we work with.

Improvements to earlier quantum SDP-solvers

In this chapter we build on the Arora-Kale framework for SDP-solving in a similar fashion as [BS17] and also use results from [BKL$^+$17a, LRS15] to construct a primal oracle. We improve on the previous results about quantum SDP-solving in three different ways:

Two-phase quantum search and minimum finding. We give a computationally more efficient version of the "Gentle Quantum Search Lemma" [Aar18] using the fast quantum OR lemma (Lemma 3.2.16). We also extend this to minimum finding to get our *two-phase quantum minimum finding* (Lemma 6.4.2). As independently observed by the authors of [BKL$^+$18] the fast quantum OR lemma gives a speed-up for SDP primal oracles in general. Moreover, using two-phase quantum minimum finding, we show how to modify the upper bound on the complexity of general SDP-solving from

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}_n\big(\sqrt{m} T^{\sigma}_{Tr}(\gamma) T_{Gibbs}(\gamma, \gamma^2, \gamma^{-1})\gamma^3\sigma\big) \tag{6.10}$$

as implied in previous work [BS17, vAGGdW17] to

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}_n\big(\big(\sqrt{m} T^{\sigma}_{Tr}(\gamma) + T_{Gibbs}(\gamma, \gamma^2, \gamma^{-1})\big)\gamma^4\sigma^2\big), \tag{6.11}$$

where $\gamma = \Theta(Rr/\varepsilon)$. For the complexity of SDP primal oracles, the same upper bounds holds. Note that this is usually an improvement unless γ is very large.

Quantum operator model and efficient data structures. We introduce the quantum operator input model unifying prior approaches. We show that both the sparse model and the quantum state model can be reduced to the quantum operator model with a constant overhead and with the choices of $\alpha = s$ and $\alpha = B$ respectively. Moreover, we show that for $\sigma = \Theta(1)$, we have that

$$T^{\sigma}_{Tr}(\gamma) = \widetilde{\mathcal{O}}_\gamma(\alpha),$$

in the quantum operator model.

The complexity bound on Gibbs-sampling in the sparse matrix input model previously obtained in [vAGGdW17] was:

$$T_{Gibbs}(K,d,\theta)=\widetilde{\mathcal{O}}_{\theta}\big(\sqrt{n}Ks^2d^2\big).$$

By considering the operator model, we derive the new upper bound

$$T_{Gibbs}(K,d,\theta)=\widetilde{\mathcal{O}}_{\theta,d}\big(\sqrt{n}K\alpha\big),$$

which for the sparse input model (with $\alpha=s$) gives a significant improvement. This result is based on the idea of gradually building up an efficient data structure for state preparation, following ideas of [KP17b]. This demonstrates that these data structures can be used efficiently even if one does not assume preprocessed data. Moreover, it shows that working in the operator model does not only unify prior approaches but also inspires more efficient quantum algorithms due to its conceptual clarity.

Gibbs-sampling for the quantum state model. We develop a new method for Gibbs-sampling in the quantum state model. As noted in [BKL+18] this model has the nice property that it is relatively easy to find the important eigenspaces of the input matrices. We introduce a new technique for finding these important eigenspaces that, in contrast to the approach in [BKL+18], does not introduce a dependence on the rank of the input matrices in the complexity. In particular we improve the complexity bound of [BKL+18]

$$T_{Gibbs}(K,d,\theta)=\mathcal{O}\left(\mathrm{poly}(K,B,d,1/\theta,\max_{j\in\{0,\dots,m\}}[\mathrm{rank}(A_j)])\right),$$

to

$$T_{Gibbs}(K,d,\theta)=\widetilde{\mathcal{O}}_{d,\theta,n}\big((KB)^{3.5}\big),$$

both making the polynomial dependence explicit and improving it.

An important consequence of this improvement is that we do not get a dependence on the rank of the input matrices in the complexity of SDP-solving, unlike Brandão et al. [BKL+18]. Since the most natural use for the quantum state model is when the A_j matrices naturally correspond to quantum states, B is often just 1. However, when the states are highly mixed, then the rank will be n, eliminating the speedup of previous works over the sparse input model where a rank dependence is present. Finally note that this Gibbs-sampling method is only beneficial if $\sqrt{n}\leq(KB)^{2.5}$, otherwise the reduction to the quantum operator model with $\alpha=B$ gives a better algorithm.

For the quantum operator input model the above results lead to the complexity bound

$$T_{SDP}(\varepsilon)=\widetilde{\mathcal{O}}\big((\sqrt{m}+\sqrt{n}\gamma)\alpha\gamma^4\big), \tag{6.12}$$

where $\gamma = \frac{Rr}{\varepsilon}$, providing a significant improvement for the sparse input model (with $\alpha = s$). The $\Omega(\sqrt{n}+\sqrt{m})$ lower bound of [BS17] also applies to the quantum operator model due to our reductions, matching the above upper bound (6.12) up to polylog factors in n and m when γ and α are constant. For the quantum state input model our improved Gibbs-sampler yields the complexity bound

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}_n\big(\big(\sqrt{m} + B^{2.5}\gamma^{3.5}\big)B\gamma^4\big).$$

In both cases, the same bound holds for an SDP primal oracle but with $\gamma := R/\varepsilon$.

With OR lemma / Two-Phase Search		Without OR lemma / Two-Phase Search	
Sparse input	Quantum state input	Sparse input	Quantum state input
$\widetilde{\mathcal{O}}((\sqrt{m}+\sqrt{n}s\gamma^5)s\gamma^4)$ Theorem[11] 6.4.3 + [vAGGdW17]	$\widetilde{\mathcal{O}}_n((\sqrt{m}+\mathrm{poly}(\mathrm{rk}))\mathrm{poly}(\gamma, B))$ [BKL+18]	$\widetilde{\mathcal{O}}(\sqrt{mn}s^2\gamma^8)$ [vAGGdW17]	$\widetilde{\mathcal{O}}_n(\sqrt{m}\mathrm{poly}(\gamma, B, \mathrm{rk}))$ [BKL+17a]
$\widetilde{\mathcal{O}}((\sqrt{m}+\sqrt{n}\gamma)s\gamma^4)$ Theorem 6.4.9	$\widetilde{\mathcal{O}}_n((\sqrt{m}+B^{2.5}\gamma^{3.5})B\gamma^4)$ Theorem 6.4.16	$\widetilde{\mathcal{O}}(\sqrt{mn}s\gamma^4)$ Corollary 6.4.10	$\widetilde{\mathcal{O}}_n(\sqrt{m}B^{3.5}\gamma^{6.5})$ Corollary 6.4.17

Table 6.1. Summary of the role of our various improvements. The first row of complexity bounds use previous Gibbs-samplers, while the second row of complexities follow from our new Gibbs-sampling techniques. The main new results are on the bottom left, the other complexity statements represent partial results following from only applying some of the improvements. We present the results for the sparse matrix and quantum state input models for comparison to prior work. However, note that our results presented for sparse input hold more generally for the quantum operator input model; to get the corresponding results one should just replace s by α in the table. Hence similar bounds hold in the quantum state input model too, after replacing s by B, which can be beneficial when $B^{2.5}\gamma^{2.5} \geq \sqrt{n}$. Notation: $\mathrm{rk} = \max_{j\in\{0,\ldots,m\}} \mathrm{rank}(A_j)$ and $\gamma = \frac{Rr}{\varepsilon}$.

Application to Shadow tomography

We extend the idea of applying SDP-solving to the problem of shadow tomography: given samples of an unknown, n-dimensional quantum state ρ, find ε-additive approximations of the expectation values $\mathrm{Tr}[E_1\rho], \ldots, \mathrm{Tr}[E_m\rho]$ of several binary measurement operators. This problem was introduced by Aaronson in [Aar18], where he gave an efficient algorithm in terms of the number of samples from ρ. In particular he proved that $\widetilde{\mathcal{O}}\big(\log^4(m)\log(n)/\varepsilon^5\big)$ samples suffice. Brandão et al. [BKL+18] applied their SDP-solver to get a more efficient algorithm in terms of computation time when the measurements E_i are given in the quantum state model, while keeping the sample complexity as low as $\mathrm{poly}(\log(m), \log(n), 1/\varepsilon, B)$. We simultaneously improve on both results,

[11]A similar result was independently proved by Brandão et al. [BKL+18].

giving a sample bound of $\widetilde{\mathcal{O}}\big(\log^4(m)\log(n)/\varepsilon^4\big)$ while also improving the best known time complexity [Aar18, BKL$^+$18] of the implementation for all input models. Finally we show that if we can efficiently implement the measurements $\mathrm{Tr}[E_1\rho],\ldots,\mathrm{Tr}[E_m\rho]$ on a quantum computer, then we can also efficiently represent $E_1,\ldots,E_m$ using the quantum operator input model, hence the computational complexity can be stated in terms of the number of measurements needed.

Lower bounds

We end the chapter by proving some new lower bounds. Lower bounds on the quantum query complexity of SDP-solving for the sparse input model were presented in previous works [BS17, vAGGdW17]. We add to this by giving $\Omega(\sqrt{m}B/\varepsilon)$ and $\Omega(\sqrt{m}\alpha/\varepsilon)$ bounds for the quantum state model and quantum operator model respectively. These lower bounds show that the $\sqrt{m}$ factor and the polynomial dependence on the parameters B,α, and $1/\varepsilon$ are necessary.

Compared to problems with a discrete input, proving lower bounds on continuous-input quantum problems gives rise to extra challenges and often requires more involved techniques, see for example the work of Belovs [Bel15] on generalizations of the adversary method. Due to these difficulties, fewer results are known in this regime. Examples of known continuous-input lower-bound results include phase-estimation related problems (cf. Bessen [Bes05]) and the complexity-theoretic version of the no-cloning theorem due to Aaronson [Aar09]. We use the hybrid-method based approach presented in Chapter 4 in order to handle continuous-input oracles, combined with efficient reductions between input models stemming from the smooth functions of Chapter 3.

6.3.4 Subsequent work

A few months after the first version of our paper [vAG19] (providing the basis of this chapter) was posted online, Kerenidis and Prakash [KP18] gave a quantum interior point algorithm for solving LPs and SDPs. They work in an input model where the input matrices are stored in QROM, which input model is also covered by our quantum operator input model. However, it is hard to compare their complexities to ours, because their final complexity statement depends polynomially on the condition number of the matrices that the interior point method encounters, and they do not give explicit bounds for these condition numbers. Also they have two accuracy parameters; while one accuracy parameter only appears as a logarithmic factor, their complexity depends polynomially on the other.

6.4 Technical improvements

Here we present multiple improvements to quantum SDP-solvers. In Section 6.4.1 we show how recent work on the quantum OR lemma [HLM17, BKL$^+$18] (see also

Lemma 3.2.16) can be used to speed up minimum finding and search in certain scenarios. In Section 6.4.2 we show how a combination of recent ideas, such as using block-encodings [LC16, GSLW19], Linear combination of unitaries [CW12, CKS17] and quantum data structures [KP17b] can be used to improve Gibbs-sampling. Finally, in Section 6.4.3 we focus on the quantum state model[BKL$^+$17a]. Similarly to quantum principal component analysis [LMR14], we use the input states both operationally and as a quantum distribution. Utilizing the fact that we can access purifications of the quantum states, we exponentially improve the precision-dependence of some subroutines, and also remove the rank-dependence that was previously present [BKL$^+$18].

6.4.1 Two-Phase Minimum Finding

To speed up the SDP-solvers derived from the frameworks of Section 6.2 we use a fast version of the quantum OR lemma (Lemma 3.2.16), and derive a slightly extended version of it that we call two-phase quantum search.

In the setting of the two-phase quantum search we have m algorithms for decision problems and we ask whether one of them evaluates to 1, and if so, we want to find one. We also know that all algorithms start with preparing some state ρ, followed by some procedure U_j that depends on the index of the decision problem $j \in [m]$. In classical deterministic processes it is quite natural that only one preparation of ρ is needed since the result can be stored. For bounded-error classical processes $\mathcal{O}(\log(m))$ preparations of ρ suffice to get the error probability of one decision problem much below $1/m$. By the classical union bound this is low enough that we can find a marked element with high probability. However, if ρ is a quantum state and the U_j are quantum algorithms, then such a bound is not so straightforward, since progress made in constructing ρ might be destroyed when running one of the U_j. Nevertheless, using the Fast Quantum OR Lemma it can be shown that $\widetilde{\mathcal{O}}\big(\log^4(m)\big)$ samples from ρ suffice.

6.4.1. Lemma (two-phase quantum search). *Let $\xi \in (0,1)$. Let ρ be a quantum state and $U_1,\dots,U_m$ be unitaries with the U_j accessible through a unitary U that acts as $U|j\rangle|\psi\rangle = |j\rangle U_j|\psi\rangle$. Then there is a quantum algorithm, that using $\widetilde{\mathcal{O}}\big(\log^4(m)\log(\xi)\big)$ samples of ρ and $\widetilde{\mathcal{O}}(\sqrt{m}\log(\xi))$ applications of U and its inverse, outputs with success probability at least $1-\xi$ either*

- *a j such that $\mathrm{Tr}\Big[(I\otimes|1\rangle\!\langle 1|)U_j\rho U_j^\dagger\Big] \geq 1/3$, i.e., a j such that U_j outputs 1 with probability at least $1/3$ on input ρ,*
- *or concludes correctly that $\mathrm{Tr}\Big[(I\otimes|1\rangle\!\langle 1|)U_j\rho U_j^\dagger\Big] < 2/3$ for all j, i.e., no unitary outputs 1 with probability at least $2/3$ on input ρ.*

Proof:
This follows from the proof of "Gentle Quantum Search" in [Aar18, Lemma 15]

using the fast quantum OR lemma (Lemma 3.2.16) instead of the normal quantum OR lemma.[12]

□

Using the above lemma we construct the *two-phase quantum minimum finding* algorithm. It turns out that we need to use this algorithm in a situation where different values have different error-scales, therefore the statement gets slightly complicated. In typical use-cases one can probably just choose each error-margin η equal to, say, δ resulting in a simpler statement.

6.4.2. LEMMA (two-phase quantum minimum finding). *Let $\delta, \nu' \in (0,1)$. Let ρ be a quantum state and $U_1, \ldots, U_m$ be unitaries, with the U_j accessible through a unitary U that acts as $U|j\rangle|\psi\rangle = |j\rangle U_j|\psi\rangle$. Let $a_1, \ldots, a_m$, $\eta_1, \ldots, \eta_m$ be real numbers such that $\min_j |a_j| + |\eta_j| \leq M$. Assume that for all j with probability at least $2/3$, U_j computes a binary representation of a_j up to additive error η_j using one copy of ρ. Then, with probability at least $1-\nu'$, we can find a j such that $a_j - \eta_j \leq \min_i(a_i + \eta_i) + \delta$ using $\widetilde{\mathcal{O}}\big(\log^4(m)\log(M/\delta)\log(\nu')\big)$ samples of ρ and $\widetilde{\mathcal{O}}(\sqrt{m}\log(\nu')\log(M/\delta))$ applications of U and its inverse.*

Proof:
Do a binary search on the value v to precision δ by checking whether there is still an element with $a_i + \eta_i \leq v$ using Lemma 6.4.1 in each round with $\nu = \Theta(\nu'/\log(M/\delta))$. This binary search will result in a value $v \leq \min_i(a_i+\eta_i)+\delta$ with probability at least $1-\nu/2$, and it is not hard to see that the last j found by Lemma 6.4.1 during the binary search will be such that $a_j - \eta_j \leq v$ with probability at least $1-\nu/2$. Therefore this j satisfies the required inequality with probability at least $1-\nu$. □

This leads to the following general bound on SDP-solving.

6.4.3. THEOREM. *Assume that updating an entry of $y \in \mathbb{R}^{m+1}$ in the data structure requires at most $\widetilde{\mathcal{O}}(T_{Gibbs}(\gamma, \gamma^2, \gamma^{-1}))$ elementary operations, where $\gamma := 6Rr/\varepsilon$. Then there is a quantum SDP-solver for which*

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}_n\big(\big(\sqrt{m}T^{\sigma}_{Tr}(\gamma^{-1}) + T_{Gibbs}(\gamma, \gamma^2, \gamma^{-1})\big)\gamma^4\sigma^2\big),$$

similarly there is also a quantum algorithm with the same complexity, but with $\gamma := 6R/\varepsilon$, that implements an SDP primal oracle.

[12]One might be wondering why the fourth power on the logarithm, is it not just a simple binary search argument using the quantum OR lemma? The difficulty arises from the fact that at each level of the binary search we can only answer a question like "is there a j with success probability $\geq \delta$ or every success probability is below $\delta + \varepsilon$?" It turns out that one needs to choose $\varepsilon \approx 1/\log(m)$, and δ decreasing at each step with ε. This then requires $\approx \log^3(m)$ copies for every level of the binary search.

Proof:
To construct an SDP-solver use both frameworks in succession, otherwise use only the primal oracle. The frameworks run for $\widetilde{\mathcal{O}}_n(\gamma^2)$ iterations. In each iteration we need to update at most three entries of the y vector, which takes at most $\widetilde{\mathcal{O}}(T_{Gibbs}(\gamma,\gamma^2,\gamma^{-1}))$ elementary operations by assumption. To search for a violated constraint when using the primal oracle framework, we use the Two-Phase Quantum Search, and we use Two-Phase Minimum Finding to implement the minimum finding needed in the Oracle for the Arora-Kale framework[13], following the geometric approach of Lemma 6.2.2 for implementing a γ^{-1}-oracle.

Let $\rho := \rho_S^{\otimes k}$ where $k = 6(4\sigma\gamma)^2$, S is a uniform random seed and $\tilde{\rho}_S$ is a density operator, that is a $\gamma^{-1}/4$-approximation in trace distance of the Gibbs state $\rho_{Gibbs(y)}$ corresponding to the current y vector (for at least a 4/5 fraction of the possible input seeds). Let U_j be the operator that applies a $(\gamma^{-1}/4,\sigma)$-trace estimator to each copy of $\tilde{\rho}_S$ and takes the average of the outcomes. I.e., it obtains estimates of $\mathrm{Tr}[A_j\tilde{\rho}_S]$ with bias at most $\gamma^{-1}/4$ and standard deviation at most σ independently k times, taking the average at the end. By Chebyshev's inequality we can see that this way U_j computes a $2\gamma^{-1}/4$-precise estimate of $\mathrm{Tr}[A_j\tilde{\rho}_S]$ with probability at least 5/6. Also with probability at least 4/5 we have that $\left\|\tilde{\rho}_S - \rho_{Gibbs(y)}\right\| \leq \gamma^{-1}/4$ and thus we get a $3\gamma^{-1}/4$-precise estimate of $\mathrm{Tr}\left[A_j\rho_{Gibbs(y)}\right]$ with probability at least $4/5 \cdot 5/6 = 2/3$. The preparation of ρ can be performed using $kT_{Gibbs}(\gamma,\gamma^2,\gamma^{-1})$ queries by definition, whereas U_j can be implemented with query complexity $kT^{\sigma}_{Tr}(\gamma^{-1})$.

By using Two-Phase Quantum Search and Two-Phase Quantum Minimum Finding with $\nu = \widetilde{\mathcal{O}}_n(\gamma^{-2})$ we get that it takes

$$\widetilde{\mathcal{O}}_n\big(\big((\sqrt{m}T^{\sigma}_{Tr}(\gamma^{-1}) + T_{Gibbs}(\gamma,\gamma^2,\gamma^{-1})\big)\gamma^2\sigma^2\big)$$

queries to implement an iteration. The stated final complexity follows considering the number of iterations is $\widetilde{\mathcal{O}}_n(\gamma^2)$. □

In the rest of this section we give upper bounds on T_{Gibbs} and T^{σ}_{Tr} for different input models. In particular, due to the results from the next section, $T_{Gibbs}(K,d,\theta)$ will always depend only logarithmically on d and θ, and $T^{\sigma}_{Tr}(\theta)$ will only depend logarithmically on θ. Nevertheless we left these parameters in Theorem 6.4.3 for completeness and to allow for comparison with previous results.

6.4.2 SDP-solving using the quantum operator input model

In this subsection we first show how to convert the quantum state input model to the quantum operator input model, then we show how to implement efficient

[13]In the Oracle implementation of Lemma 6.2.2 the minimum finding is not done over the computed traces, but rather the angles calculated using these traces. The trace → angle conversion suffices with precision $\delta = 1/\mathrm{poly}(\gamma)$, and since the magnitude M of angles is bounded by π, we get $\log(M/\delta) = \mathcal{O}(\log(\gamma))$ in Lemma 6.4.2.

trace estimation and Gibbs-sampling in the quantum operator model.

Subnormalized density operators as block-encodings

We present a lemma based on ideas of [LC16, Corollary 9] showing how to implement a block-encoding of a (subnormalized) density operator using purified access to the density operator, slightly generalizing our earlier Lemma 3.3.2.

6.4.4. LEMMA (Block-encoding of a (subnormalized) density operator [LC16]). *Let G be a $(w+a)$-qubit unitary which on the input state $|0\rangle^{\otimes w}|0\rangle^{\otimes a}$ prepares a purification $|\varrho\rangle$ of the subnormalized w-qubit density operator ϱ. Then we can implement a $(1, w+a, 0)$-block-encoding of ϱ, with a single use of G and its inverse and with $w+1$ two-qubit gates.*

Proof:
Let us write $|\varrho\rangle = \alpha|\rho_0\rangle + \beta|\rho_1\rangle$, where $\alpha, \beta \in \mathbb{R}$, $(I_{2^w} \otimes |0\rangle\langle 0| \otimes I_{2^{a-1}})|\rho_0\rangle = |\rho_0\rangle$ and $(I_{2^w} \otimes |1\rangle\langle 1| \otimes I_{2^{a-1}})|\rho_1\rangle = |\rho_1\rangle$. Moreover, without loss of generality we can assume that $|\varrho_0\rangle = \sum_{j=1}^{D} \sqrt{p_j}|\psi_j\rangle|0\rangle|\tilde{\psi}_j\rangle$ such that $\langle\tilde{\psi}_i|\tilde{\psi}_j\rangle = \delta_{ij}$ and $\varrho = \alpha^2 \sum_{j=1}^{D} p_j |\psi_j\rangle\langle\psi_j|$. Consider the $(2w+a+1)$-qubit unitary $V = (I_{2^{w+1}} \otimes G^\dagger)(\mathrm{SWAP}_{w+1} \otimes I_{2^a})(I_{2^{w+1}} \otimes G)$, where SWAP_{w+1} denotes the unitary which swaps the first $(w+1)$-qubit register with the second $(w+1)$-qubit register. Observe that

$$\begin{aligned}
(I_{2^w} \otimes \langle 0|^{\otimes 1+w+a}) & V(I_{2^w} \otimes |0\rangle^{\otimes 1+w+a}) = \\
&= (I_{2^w} \otimes \langle 0|\langle\varrho|)(\mathrm{SWAP}_{w+1} \otimes I_{2^a})(I_{2^w} \otimes |0\rangle|\varrho\rangle) \\
&= \alpha^2 (I_{2^w} \otimes \langle 0|\langle\rho_0|)(\mathrm{SWAP}_{w+1} \otimes I_{2^a})(I_{2^w} \otimes |0\rangle|\rho_0\rangle) \\
&= \alpha^2 \sum_{j=1}^{D} p_j (I_{2^w} \otimes \langle 0|\langle\psi_j|\langle 0|)(\mathrm{SWAP}_{w+1} \otimes I_{2^a})(I_{2^w} \otimes |0\rangle|\psi_j\rangle|0\rangle) \\
&= \alpha^2 \sum_{j=1}^{D} p_j (I_{2^w} \otimes \langle 0| \otimes \langle\psi_j|\langle 0|)(|\psi_j\rangle|0\rangle \otimes I_{2^w} \otimes |0\rangle) \\
&= \alpha^2 \sum_{j=1}^{D} p_j (|\psi_j\rangle \otimes \langle\psi_j|) \\
&= \alpha^2 \sum_{j=1}^{D} p_j |\psi_j\rangle\langle\psi_j| \\
&= \varrho.
\end{aligned}$$

□

The above corollary essentially shows that the quantum operator input model generalizes the quantum state input model too, by choosing $\alpha = B$ (since $\mu_j^+ - \mu_j^- + |\mu_j^I| \leq B$). What is left is to show how to implement a linear combination of block-encodings, e.g., $A_j = \mu_j^+ \varrho_j^+ - \mu_j^- \varrho_j^- + \mu_j^I I$. We show how to efficiently implement such a block-encoding in the next subsection.

Efficient trace estimation

Now we show how to use a block-encoding of a Hermitian matrix A for efficient trace estimation.

6.4.5. Corollary. *Suppose $-I \preceq A \preceq I$, $0 < \theta < 1$ and $\alpha \geq 1$. We can implement a trace estimator for A with standard deviation $\sigma \leq 6$ and bias $\leq \theta$ with $\widetilde{\mathcal{O}}_\theta(\alpha)$ uses of a block-encoding of A/α.*

Proof:
Suppose that we are given a unitary U block-encoding A/α. By Theorem 3.2.7 we can convert it to a unitary U' that block-encodes an operator $\mathcal{O}(\theta)$-close to $A/2$ with $\mathcal{O}\big(\alpha\log\big(\frac{\alpha}{\theta}\big)\big)$ uses of U. Then by linear combination of block-encodings (Lemma 3.3.9) we can construct a unitary U'' that block-encodes an operator $\mathcal{O}(\theta)$-close to $I/2 + A/4$ with a single use of U'. Finally by Corollary 3.4.14 (setting $\kappa \geq 4$) and Corollary 2.3.8 we can implement a unitary $\widetilde{U}$ block-encoding of an operator $\mathcal{O}(\theta)$-close to $\sqrt{I/2 + A/4}$ with $\mathcal{O}(\log(1/\theta))$ uses of U'', and apply the block-encoding $\widetilde{U}$ to ρ. Suppose $\widetilde{U}$ is an a-qubit block-encoding, then the probability of finding the ancilla qubits to be $|0\rangle^{\otimes a} := |\bar{0}\rangle$ upon measurement is

$$\begin{aligned}\mathrm{Tr}\Big[(\langle\bar{0}|\otimes I)\tilde{U}^\dagger(|\bar{0}\rangle\langle\bar{0}|\otimes\rho)\tilde{U}(|\bar{0}\rangle\otimes I)\Big] &=\\ &= \mathrm{Tr}\Big(\underbrace{(\langle\bar{0}|\otimes I)\tilde{U}(|\bar{0}\rangle\otimes I)}_{\approx\frac{\sqrt{I/2+A/4}}{2}}\underbrace{(\langle\bar{0}|\otimes I)\tilde{U}^\dagger(|\bar{0}\rangle\otimes I)}_{\approx\frac{\sqrt{I/2+A/4}}{2}}\rho\Big)\\ &= \frac{1}{8} + \frac{\mathrm{Tr}[A\rho]}{16} + \mathcal{O}(\theta).\end{aligned}$$

Upon measuring the ancilla qubit and getting outcome $|\bar{0}\rangle$ we output $16-2=14$. In case of any other measurement outcome we output -2. By choosing the right constants so that $\tilde{U}$ is a precise enough block-encoding we can ensure that the bias is less than $\theta/2$, and the standard deviation $\sigma \leq 6$. □

6.4.6. Corollary. *For $\sigma = 6$, we have that $T^\sigma_{Tr}(\theta) = \widetilde{\mathcal{O}}_{\frac{n}{\varepsilon},\theta}(\alpha)$ in the quantum operator input model.*

Implementing a linear combination of block-encodings

In the SDP-solving frameworks we need to prepare Gibbs states for Hamiltonians of the form $\sum_{j=0}^m y_j A_j$. Our innovation is to first convert the matrices to block-encodings and then directly construct linear combinations of block-encodings.

Using linear combinations of block-encodings bypasses the entrywise summation of the input matrices which was a major bottleneck in previous SDP-solvers for the sparse input model [BS17, vAGGdW17].

This is done by using an efficient data structure as the intermediate storage of y. This data structure has recently been used by many quantum algorithms as part of their input model. However, as far as we are aware this is the first use of it as an intermediate storage in a quantum algorithm. Combining these techniques for block-encodings with the meta-algorithm of Theorem 6.4.3 leads to an efficient quantum SDP-solver, see Theorem 6.4.9.

Now we describe how to use a quantum-access classical RAM (QCRAM) to efficiently implement a state-preparation-pair unitary (Definition 3.3.8) that can be used to construct the linear combinations of the block-encodings (Lemma 3.3.9). In the SDP-solver we use this data structure for the summation of constraint matrices needed for Gibbs-sampling.

6.4.7. Lemma. *There is a data structure that can store an m-dimensional d-sparse vector $y \in \mathbb{R}^m$ with θ-precision (in ℓ^1-norm) using a RAM (or QCRAM) of size $\widetilde{\mathcal{O}}_{\frac{m}{\theta}}(d)$. Initialising the data structure costs $\widetilde{\mathcal{O}}_{\frac{m}{\theta}}(1)$ binary operations, furthermore:*

- *Given a classical s-sparse vector, adding[14] it to the stored vector has classical cost $\widetilde{\mathcal{O}}_{\frac{m}{\theta}}(s)$.*
- *Querying the i-th element of the vector costs $\widetilde{\mathcal{O}}_{\frac{m}{\theta}}(1)$ binary operations and queries to the QCRAM.*
- *Given that $\beta \geq \|y\|_1$ we can implement a (symmetric) $(\beta, \widetilde{\mathcal{O}}_{\frac{m}{\theta}}(1), \theta)$-state-preparation pair for y with $\widetilde{\mathcal{O}}_{\frac{m}{\theta}}(1)$ queries to the QCRAM.*

Proof:
We use the data structure of [KP17b, Appendix A]. □

6.4.8. Corollary. *Having access to the above data structure for y, we have $T_{Gibbs}(K, d, \theta) = \widetilde{\mathcal{O}}_\theta(\alpha K \sqrt{n})$ in the quantum operator input model.*

Proof:
This can be proven using the Gibbs-sampler of Theorem 3.4.21, combined with the above lemma and Lemma 3.3.9 for constructing a block-encoding of the linear combination of the operators $\sum_{j=0}^{m} y_j A_j$. □

This directly gives the following result for SDP-solving:

[14]In order to avoid error accumulation from repeated rounding, we assume for simplicity that there can be at most $\mathcal{O}(\mathrm{poly}(m/\theta))$ such calls to the data structure in total.

6.4.9. Theorem. *In the quantum operator input model*

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}\big((\sqrt{m} + \sqrt{n}\gamma)\alpha\gamma^4\big),$$

where $\gamma = Rr/\varepsilon$ is. For a primal oracle the same complexity can be accomplished with $\gamma = R/\varepsilon$. The input oracle of the quantum operator model can be constructed using $\mathcal{O}(1)$ queries and $\widetilde{\mathcal{O}}_{mn\alpha\gamma}(1)$ elementary operations in the sparse matrix model and also in the quantum state model with $\alpha = s$ or $\alpha = B$ respectively. Therefore the above bound applies to these input models too.

Proof:
The complexity statement follows from Theorem 6.4.3 using Corollary 6.4.6 and 6.4.8. The reductions follow from Lemma 3.3.5 and Lemma 6.4.4. □

If we do not apply two-phase minimum finding but use standard quantum minimum finding [DH96] instead, then we get a result with a slightly better dependence on γ, cf. (6.10):

6.4.10. Corollary. *In the quantum operator input model*

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}\big(\sqrt{mn}\alpha\gamma^4\big),$$

where $\gamma = Rr/\varepsilon$ for a full SDP-solver and $\gamma = R/\varepsilon$ for a SDP primal oracle.

6.4.3 Exponentially improved Gibbs-sampling in the quantum state input model

In this subsection we show how to harness the special structure of the quantum state input model, to improve the complexity of Corollary 6.4.8. As shown in [BKL⁺18] this allows for an SDP-solver with a polylog dependence on n. We improve on the results of [BKL⁺18] by constructing a Gibbs-sampler with no explicit dependence on the rank of the input matrices. Moreover, we also improve the dependence on precision from polynomial to logarithmic.

We will use the following lemma about projectors.

6.4.11. Lemma. *Let $0 < q < 1$, Π be a projector and ϱ a subnormalized density operator. Suppose that $q\Pi \preceq \varrho \preceq I/2$, $(I-\Pi)\varrho(I-\Pi) = 0$ and we have access to an a-qubit unitary $U_{\tilde{\varrho}}$ preparing a purification of a subnormalized density operator $\tilde{\varrho}$ such that $\|\varrho - \tilde{\varrho}\|_1 \leq 4\nu$. Then we can a prepare a purification of a subnormalized density operator $\tilde{\varrho}_{\mathrm{unif}}$ such that $\left\|\frac{q}{4}\Pi - \tilde{\varrho}_{\mathrm{unif}}\right\|_1 \leq \widetilde{\mathcal{O}}(\nu/q)$, with $\widetilde{\mathcal{O}}_\nu(1/q)$ queries to $U_{\tilde{\varrho}}$ and its inverse and using $\widetilde{\mathcal{O}}_\nu(a/q)$ two-qubit gates.*

Proof:
First let us assume that we have access to U_ϱ instead of $U_{\tilde{\varrho}}$. Then, using Corollary 3.4.13 we could implement a unitary W which is a $(1, a+2, 0)$-block-encoding

of V such that $\left\|\left(V - \frac{\sqrt{q}}{2\sqrt{\varrho}}\right)\Pi\right\| \leq \nu$, with $\widetilde{\mathcal{O}}_\nu(1/q)$ uses of U_ϱ. Note that since $q\Pi \preceq \varrho$ and $(I-\Pi)\varrho(I-\Pi) = 0$ we know that ϱ is supported on the image of Π, in particular $\Pi\varrho\Pi = \varrho$. Using Hölder's inequality it is easy to see that

$$\left\| V\varrho V^\dagger - \frac{\sqrt{q}}{2\sqrt{\varrho}}\varrho\frac{\sqrt{q}}{2\sqrt{\varrho}} \right\|_1 \leq 2\nu,$$

which is equivalent to

$$\left\| V\varrho V^\dagger - \frac{q}{4}\Pi \right\|_1 \leq 2\nu.$$

Considering that $\|\varrho - \tilde{\varrho}\| \leq \|\varrho - \tilde{\varrho}\|_1 \leq 4\nu$, if in the implementation of the controlled Hamiltonian simulation we replace U_ϱ by $U_{\tilde{\varrho}}$, then we make error $\widetilde{\mathcal{O}}(\nu/q)$, as shown by Lemma 2.4.4. The resulting new unitary $\tilde{W}$ is therefore an $(1, a+2, \widetilde{\mathcal{O}}(\nu/q))$-block-encoding of V. Thus we can prepare a purification of the a subnormalized density operator $\tilde{\varrho}_{\text{unif}}$ such that

$$\left\| \frac{q}{4}\Pi - \tilde{\varrho}_{\text{unif}} \right\|_1 \leq \widetilde{\mathcal{O}}(\nu/q).$$

□

In the proof of the next lemma we will mostly be looking at *Eigenvalue threshold projectors.*

6.4.12. Definition (Eigenvalue threshold projector). Suppose H is a Hermitian matrix and $q \in \mathbb{R}$. Let $\Pi_{H>q}$ denote the orthogonal projector corresponding to the subspace spanned by the eigenvectors of H that have eigenvalue larger than q. We define $\Pi_{H\leq q} := I - \Pi_{H>q}$.

We are now ready to prove the main lemma of this section, an improved Gibbs-sampler in the quantum state model. In the proof we will use some specific conventions and notation to simplify the proof. We say that two subnormalized density operators are *δ-close* when their trace distance is at most δ, and that two unitaries are δ-close when their operator norm distance is at most δ. We will always work with purifications of subnormalized density operators, so when for example we say that we apply an operator to a subnormalized density operator, we mean that we apply the operator to its purification.

6.4.13. Lemma (Gibbs-sampling of the difference of density operators). *Suppose that we have unitaries $U_{\varrho^\pm}$ preparing a purification of the subnormalized density operators $\varrho^\pm$ using $a = \mathcal{O}(\text{poly}\log(n))$ qubits. Let*[15] *$H := (\varrho^+ - \varrho^-)/2$, $\beta \in [1, n/2]$ and $\delta, \eta \in (0,1]$. Assume we are given a point $q \in [2/n, 1/\beta]$ that is η-far from the spectrum of H, i.e.*

$$|\lambda - q| \geq \eta \textit{ for all } \lambda \in \text{Spec}(H).$$

[15]In case n is not a power of 2 we still represent H on $\lceil\log_2(n)\rceil$ qubits, but think about it as an $\mathbb{C}^{n\times n}$ operator.

Then we can prepare a purification of an approximate Gibbs state $\tilde{\rho}_{Gibbs}$ such that

$$\left\| \frac{e^{\beta H}}{\mathrm{Tr}[e^{\beta H}]} - \tilde{\rho}_{Gibbs} \right\|_1 \leq \delta$$

with[16] $\widetilde{\mathcal{O}}_{\frac{n}{\delta}}(q^{-1.5}/\eta)$ *queries to controlled-*$U_{\varrho^{\pm}}$ *and their inverses.*[17]

Proof:
The main idea of the proof is that we prepare (slightly subnormalized) density operators corresponding to $\Pi_{H>q}$ and $\Pi_{H\leq q}$, i.e., uniform distributions over a partition of eigenspaces of H. Utilising these states we prepare subnormalized Gibbs states on the corresponding subspaces, then merge and amplify the states in order to obtain the final Gibbs state. This is beneficial since on the subspace corresponding to $\Pi_{H\leq q}$ the map $e^{\beta H}$ is nicely bounded. However, on the image of $\Pi_{H>q}$ the map $e^{\beta H}$ might behave wildly, and in the extreme case this map might magnify the amplitude of some eigenvectors tremendously. This implies that we need to "find" such magnified elements, as the Gibbs state is concentrated around them. Fortunately the rank of $\Pi_{H>q}$ is at most $1/q$, which makes it easier to "find" the extreme vectors then if we would apply the same procedure to the uniform distribution I/n.

We start with implementing the unitary $\tilde{V}_{H>q}$ that labels eigenstates of H according to which component of $\mathbb{R} \setminus \{q\}$ their eigenvalue lies in. More precisely, we set $\delta' := \widetilde{\Theta}(\delta q^2)$, and we want to implement a unitary $\tilde{V}_{H>q}$ which is a $(1, \widetilde{\mathcal{O}}_{\frac{n}{\eta\delta'}}(1), \delta')$-block-encoding of $(\Pi_{H>q} \otimes I + \Pi_{H\leq q} \otimes X)$. Due to the assumption that q lies at least η-far from $\mathrm{Spec}(H)$ we can implement these unitaries using $\Theta(\eta)$ precise phase estimation of the operator e^{iH}, repeated $\mathcal{O}(\log(1/\delta'))$ times. This can be implemented with $\widetilde{\mathcal{O}}_{\delta'}(1/\eta)$ queries as shown by Lemmas 6.4.4 and 3.3.9.

Now let us consider Gibbs-sampling on the image of $\Pi_{H>q}$. Let $\varsigma^+ := (\varrho^+ + \varrho^-)/2$ be a subnormalized density operator, which we can prepare in a purified manner using $\mathcal{O}(1)$ queries. Also let

$$\varsigma^+_{H>q} := \Pi_{H>q}\varsigma^+\Pi_{H>q},$$

and observe that we can prepare $\tilde{\varsigma}^+_{H>q}$, such that

$$\left\|\tilde{\varsigma}^+_{H>q} - \varsigma^+_{H>q}\right\|_1 \leq 2\delta'(\leq q/4), \tag{6.13}$$

by applying $\tilde{V}_{H>q}$ to ς^+ (the second inequality can be assumed w.l.o.g. since $\delta' = \widetilde{\Theta}(\delta q^2)$).

[16]We think that it should be possible to improve the complexity to $\widetilde{\mathcal{O}}_{\frac{n}{\delta}}(q^{-1}/\eta)$ using recent results about variable-time amplitude amplification and estimation [CGJ19].

[17]If $U_{\varrho^{\pm}}$ are not controlled, then it is easy to construct a controlled version using $\mathcal{O}(a)$ extra ancilla qubits.

Observe that

$$\begin{aligned} q\Pi_{H>q} &= \Pi_{H>q}(q\Pi_{H>q})\Pi_{H>q} \\ &\preceq \Pi_{H>q}(H)\Pi_{H>q} \\ &\preceq \Pi_{H>q}(H+2\varrho^{-})\Pi_{H>q} \\ &= \varsigma^{+}_{H>q}. \end{aligned} \tag{6.14}$$

This allows us to apply Lemma 6.4.11 to $\varrho := \varsigma^{+}_{H>q}$, $\tilde{\varrho} := \tilde{\varsigma}^{+}_{H>q}$ and $\Pi := \Pi_{H>q}$ with $\nu := \delta'$ so we get that we can prepare a state $\tilde{\varrho}_{\mathrm{unif}}$ such that

$$\left\| \frac{q}{4}\Pi_{H>q} - \tilde{\varrho}_{\mathrm{unif}} \right\|_1 \leq \widetilde{\mathcal{O}}(\delta'/q)$$

using $\widetilde{\mathcal{O}}_{\delta'}(q^{-1}/\eta)$ queries.

Now we can check if $\Pi_{H>q} = 0$ or not as follows. If it is not 0 then $\mathrm{Tr}[\Pi_{H>q}] \geq 1$ and hence by (6.13)-(6.14) we have $\mathrm{Tr}\big[\tilde{\varsigma}^{+}_{H>q}\big] \geq \mathrm{Tr}\big[\varsigma^{+}_{H>q}\big] - q/4 \geq q - q/4 = 3q/4$. Since $\varsigma^{+}_{H>q} = \mathrm{Tr}[\Pi_{H>q}]\varsigma^{+}_{H>q}\mathrm{Tr}[\Pi_{H>q}]$, when $\Pi_{H>q} = 0$ it similarly follows that $\mathrm{Tr}\big[\tilde{\varsigma}^{+}_{H>q}\big] \leq \mathrm{Tr}\big[\varsigma^{+}_{H>q}\big] + q/4 = q/4$. Thus we can check whether $\Pi_{H>q} = 0$ by checking whether $\mathrm{Tr}\big[\tilde{\varsigma}^{+}_{H>q}\big] \leq q/4$ or $\mathrm{Tr}\big[\tilde{\varsigma}^{+}_{H>q}\big] \geq 3q/4$. This can be done with success probability at least $1-\delta'$ by using amplitude estimation with $\widetilde{\mathcal{O}}_{\delta'}(q^{-0.5})$ calls to the procedure preparing $\tilde{\varsigma}^{+}_{H>q}$, costing $\widetilde{\mathcal{O}}_{\delta'}(q^{-0.5}/\eta)$ queries in total. For the final Gibbs-sampling we will consider the Gibbs state on the image of $\Pi_{H>q}$ and $\Pi_{H\leq q}$ separately. Therefore if $\Pi_{H>q} = 0$ we only need to consider the Gibbs state on the image of $\Pi_{H\leq q}$, which we do later in this proof. For now we assume $\Pi_{H>q} \neq 0$ and consider the Gibbs state on its image.

Now we use binary search in order to find $\lambda_{\max}$ the maximal eigenvalue of H, with precision $\beta^{-1}/2$ and success probability $1-\delta'$. By our assumption $\lambda_{\max} \in (q,1]$. We start each iteration of the binary search by performing phase estimation on $\tilde{\varsigma}^{+}_{H>q}$ using the unitary e^{iH} with precision $\beta^{-1}/4$ and success probability $1-q/4$. By (6.13)-(6.14) we know that the eigenvector corresponding to $\lambda_{\max}$ is present with probability at least $3q/4$ in $\tilde{\varsigma}^{+}_{H>q}$, and the other eigenvalues are present with a probability at most 1 in total. Therefore the probability of obtaining a phase estimate $\tilde{\lambda}$ such that $\tilde{\lambda} \geq \lambda_{\max} - \beta^{-1}/4$ is at least $q/2$, whereas the total probability of obtaining a phase estimate λ' such that $\lambda' \geq \lambda_{\max} + \beta^{-1}/4$ is at most $q/4$. Therefore we can perform each iteration of the binary search with precision $\beta^{-1}/2$ and success probability $1-\delta'/\log(q^{-1})$ by applying amplitude estimation to the probability of getting an eigenvalue estimate from the current search interval, using $\widetilde{\mathcal{O}}_{\beta}(q^{-0.5})$ repetitions of the initial state preparation and phase estimation procedure. Thus each iteration has query complexity $\widetilde{\mathcal{O}}_{\beta}(q^{-0.5}(\beta + 1/\eta)) = \widetilde{\mathcal{O}}(q^{-1.5} + q^{-0.5}/\eta)$, giving the same total query complexity bound $\widetilde{\mathcal{O}}(q^{-1.5} + q^{-0.5}/\eta)$ for the complete binary search.

After finding the minimum up to precision $1/(2\beta)$ we can compute a number $\tilde{\lambda}_{\max}$ such that

$$\tilde{\lambda}_{\max} I \succeq H \text{ but } (\tilde{\lambda}_{\max} - 1/\beta)I \not\succeq H. \tag{6.15}$$

Using Lemmas 6.4.4,3.3.9, and 3.4.20 we can implement an $(1/2, \widetilde{\mathcal{O}}_{\frac{\beta}{\delta'}}(1), \delta')$-block encoding of $e^{\beta\frac{H-\tilde{\lambda}_{\max}I}{2}}$ using $\widetilde{\mathcal{O}}_{\delta'}(\beta)$ queries. Applying this map to $\tilde{\varrho}_{\text{unif}}$ gives an $\widetilde{\mathcal{O}}(\delta'/q)$-approximation of the subnormalized density operator $\frac{qe^{-\beta\tilde{\lambda}_{\max}}}{16}\Pi_{H>q}e^{\beta H}$. Observe that since we assumed $\mathrm{Tr}[\Pi_{H>q}] \geq 1$, by (6.15) we get that

$$\mathrm{Tr}\left[\frac{qe^{-\beta\tilde{\lambda}_{\max}}}{16}\Pi_{H>q}e^{\beta H}\right] \geq q/(16e). \tag{6.16}$$

Thus we can prepare a subnormalized $\widetilde{\mathcal{O}}(\delta'/q)$-approximation of the Gibbs state on the image of $\Pi_{H>q}$ having trace at least $q/(16e)$.

Now we consider the Gibbs state on $\Pi_{H\leq q}$. First observe that we can prepare the density operator I/n by preparing the maximal entangled state $\frac{1}{\sqrt{n}}\sum_{j=1}^{n}|j\rangle|j\rangle$ using $\widetilde{\mathcal{O}}(\log(n))$ two-qubit quantum gates. With a single use of the unitary $\tilde{V}_{H>q}$ we can prepare an $\mathcal{O}(\delta')$-approximation of $\frac{1}{n}\Pi_{H\leq q}$ by simply marking the appropriate eigenstates of I/n, which takes $\widetilde{\mathcal{O}}(1/\eta)$ queries. Then we apply the map $e^{\beta\frac{H}{2}}/(2\sqrt{e})$ on the subspace $\Pi_{H\leq q}$ with δ' accuracy. Since $q \leq 1/\beta$ all eigenvalues of $\beta H/2$ that we are concerned with are smaller in absolute value than $1/2$. As shown by Lemma 3.4.20 this implies that implementing the map $e^{\beta\frac{H}{2}}/(2\sqrt{e})$ with δ' precision can be done using $\widetilde{\mathcal{O}}_{\delta'}(\beta)$ queries. Therefore we can prepare an $\mathcal{O}(\delta')$ approximation of the state $\frac{1}{4en}\Pi_{H\leq q}e^{\beta H}$ with $\widetilde{\mathcal{O}}_{\delta'}(\beta + 1/\eta) \leq \widetilde{\mathcal{O}}_{\delta'}(q^{-1}/\eta)$ queries.

Like before, we would like to lower bound the trace of the created subnormalized density operator. First note that $\|H\|_1 \leq 1$, and so the number[18] of eigenvalues that are larger than q in absolute value is at most $1/q \leq n/2$, thus $\mathrm{Tr}\left[\Pi_{|H|\leq q}\right] \geq n/2$. Also note that $\Pi_{H\leq q}e^{\beta H} \succeq \Pi_{|H|\leq q}e^{\beta H}$, and for an eigenvalue λ such that $|\lambda| \leq q$ we have $e^{\beta\lambda} \geq e^{-\beta q} \geq 1/e$. It follows that

$$\mathrm{Tr}\left[\frac{1}{4en}\Pi_{H\leq q}e^{\beta H}\right] \geq \mathrm{Tr}\left[\frac{1}{4en}\Pi_{|H|\leq q}e^{\beta H}\right] \geq \mathrm{Tr}\left[\frac{1}{4e^2 n}\Pi_{|H|\leq q}\right] \geq \frac{1}{8e^2}. \tag{6.17}$$

As we can now Gibbs-sample on both parts of the spectrum, we are ready to combine the two. Let[19] $\xi := \min\left(\frac{qe^{-\beta\tilde{\lambda}_{\max}}}{16}, \frac{1}{4en}\right)$, then we can prepare a purification of $\tilde{\varrho}_G$ which is an $\widetilde{\mathcal{O}}(\delta'/q)$-approximation of

$$\varrho_G := \frac{\xi}{2}e^{\beta H} = \underbrace{\left(\frac{\xi}{2}\frac{16}{qe^{-\beta\tilde{\lambda}_{\max}}}\right)}_{\leq 1/2}\frac{qe^{-\beta\tilde{\lambda}_{\max}}}{16}\Pi_{H>q}e^{\beta H} + \underbrace{\left(\frac{\xi}{2}\frac{4en}{1}\right)}_{\leq 1/2}\frac{1}{4en}\Pi_{H\leq q}e^{\beta H},$$

[18] We count eigenvalues with algebraic multiplicity.

[19] In the special case when $\Pi_{H>q} = 0$ we simply set $\xi := \frac{1}{4en}$.

by mixing the two subnormalized Gibbs states on the corresponding subspaces with appropriate ($\leq 1/2$) coefficients. This subnormalized Gibbs state $\tilde{\varrho}_G$ can be prepared at the same cost as the two partial Gibbs-state preparation, that is $\widetilde{\mathcal{O}}(q^{-1}/\eta)$ queries[20] as we have already shown.

Note that $\mathrm{Tr}[\varrho_G] = \Omega(q)$ as shown by (6.16)-(6.17), therefore we can use $\widetilde{\mathcal{O}}\left(\sqrt{1/q}\right)$ amplitude amplification steps to prepare an $\widetilde{\mathcal{O}}(\delta'/q^2)$ approximation of $\frac{\varrho_G}{\mathrm{Tr}[\varrho_G]}$, which is clearly ρ_{Gibbs}. In total this yields an $\widetilde{\mathcal{O}}(q^{-1.5}/\eta)$ query algorithm. Since $\delta' = \tilde{\Theta}(\delta q^2)$ this concludes the proof. $\square$

The following corollary expands our new Gibbs-sampling result, giving an exponential improvement in terms of the precision over the previous approach for this input model by Brandão et al. [BKL+18]. The dependence on the success probability is worse, but in our application to SDP-solving we only require success for a constant fraction of random seeds. Furthermore, there is no longer a dependence on the rank of the input matrices.

6.4.14. THEOREM. *Suppose we have query access to the $a = \mathrm{polylog}(n)$-qubit unitaries $U_{\varrho^\pm}$ preparing a purification of the (subnormalized) density operators $\varrho^\pm \in \mathbb{C}^{n\times n}$, such that $H = (\varrho^+ - \varrho^-)/2$, $\beta \geq 1$, $\theta, \delta \in (0,1]$. Then there is a quantum algorithm, that using[21] $\widetilde{\mathcal{O}}_\theta(\beta^{3.5}/\delta)$ queries to controlled-$U_{\varrho^\pm}$ or their inverses, prepares a purification of a quantum state ρ_S such that*

$$\left\| \rho_S - \frac{e^{-\beta H}}{\mathrm{Tr}[e^{-\beta H}]} \right\|_1 \leq \theta,$$

where S is an $\mathcal{O}(\log(\beta/\delta))$-bit random seed, and the above holds for at least a $(1-\delta)$-fraction of the seeds.

Proof:
If $\beta \geq n/2$, then we simply use the Gibbs-sampler from Theorem 6.4.9. Otherwise, using the random seed S we generate a uniform random number q_S from the interval $[1/(2\beta), 1/\beta]$. Note that since $\mathrm{Tr}[|H|] \leq 1$ we have that

$$|\mathrm{Spec}(|H|) \cap [1/(2\beta), 1/\beta]| \leq 2\beta.$$

Also the length of the interval is $1/(2\beta)$ therefore a random point in the interval falls $\delta/(8\beta^2)$-close to $\mathrm{Spec}(|H|)$ with probability at most δ. Therefore the random seed can be used in such a way that q_S will be $\eta = \delta/(8\beta^2)$-far from any point

[20]Note that we do the maximum finding to find $\tilde{\lambda}_{\max}$ only once, and we do not count its complexity in the state preparation.

[21]Similarly to Lemma 6.4.13 we think that it should be possible to improve the complexity to $\widetilde{\mathcal{O}}_\theta(\beta^3/\delta)$ using recent results about variable time amplitude amplification and estimation [CGJ19] (maybe at the expense of worse dependence on the error).

of Spec($|H|$) with probability at least $1-\delta$. If this is the case the procedure of Lemma 6.4.13 prepares the sought Gibbs state with the stated complexity. □

6.4.15. Corollary. *Having access to the data structure of Lemma 6.4.7 storing the vectors $\nu^{\pm} \in \mathbb{R}^{m+1}$ such that $\nu_j^{\pm} = y_j \mu_j^{\pm}$ for all $j \in \{0,\dots,m\}$, we have that $T_{Gibbs}(K,d,\theta) = \widetilde{\mathcal{O}}_\theta((BK)^{3.5})$ using the quantum state input model.*

Proof:
To start, let us define

$$\begin{aligned} H := \sum_{j=0}^{m} \frac{y_j A_j}{KB} &= \sum_{j=0}^{m} \frac{y_j}{KB}\left(\mu_j^+ \varrho_j^+ - \mu_j^- \varrho_j^- + \mu^I I\right) \\ &= \underbrace{\sum_{j=0}^{m} \frac{y_j \mu_j^+}{KB}\varrho_j^+}_{\varrho^+ :=} - \underbrace{\sum_{j=0}^{m} \frac{y_j \mu_j^-}{KB}\varrho_j^-}_{\varrho^- :=} + I \sum_{j=0}^{m} \frac{y_j \mu_j^I}{KB}. \end{aligned}$$

Notice that we can ignore the identity terms since adding identities in the exponent does not change a Gibbs state. Also let

$$\beta := KB \geq \sum_{j=0}^{m} y_j(\mu_j^+ + \mu_j^-).$$

Using Lemma 6.4.7 we can see that a $\mathcal{O}(\theta/\beta)$ approximation of $\varrho^{\pm}$ can be prepared with $\mathcal{O}(1)$ queries and using $\mathcal{O}(\operatorname{poly}\log(m\beta/\theta))$ elementary operation. By setting $\delta := 1/5$ and using Theorem 6.4.14 the statement follows. □

This directly gives the following result for SDP-solving in the quantum state model

6.4.16. Theorem. *In the quantum state input model*

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}\big(\big(\sqrt{m} + B^{2.5}\gamma^{3.5}\big)B\gamma^4\big),$$

where $\gamma = Rr/\varepsilon$. The same bound holds for a primal oracle with $\gamma = R/\varepsilon$.

Proof:
This follows directly from Theorem 6.4.3 using Corollaries 6.4.6, 6.4.8, 6.4.15. □

Also we could simply not apply two-phase minimum finding and use standard quantum minimum finding [DH96] instead to get a slightly better dependence on γ, cf. (6.10):

6.4.17. Corollary. *In the quantum state input model*

$$T_{SDP}(\varepsilon) = \widetilde{\mathcal{O}}\big(\sqrt{m}B^{3.5}\gamma^{6.5}\big),$$

where $\gamma = Rr/\varepsilon$. The same bound holds for a primal oracle with $\gamma = R/\varepsilon$.

6.5 Application to shadow tomography

We apply the idea from Brandão et al. [BKL+18] to use an SDP primal oracle to the problem of *shadow tomography* proposed by Aaronson [Aar18]. In *shadow tomography* we are given the ability to sample from an n-dimensional quantum state τ and we have a description of some measurement operators $E_1, \ldots, E_m$; the goal is to find ε-approximations of the corresponding expectation values $\mathrm{Tr}[E_j\tau]$ for all $j \in [m]$. Aaronson showed that this can be done with only

$$\widetilde{\mathcal{O}}\left(\frac{\log^4(m)\log(n)}{\varepsilon^5}\right)$$

samples from τ, but his method has high computational costs.

Brandão et al. [BKL+18] showed that a slightly relaxed problem can be efficiently solved using an SDP primal oracle. The problem they solved is to find a $y \in \mathbb{R}^m$ for which $\rho := e^{-\sum_{j=1}^m y_j E_j}/\mathrm{Tr}\left[e^{-\sum_{j=1}^m y_j E_j}\right]$ is such that $|\mathrm{Tr}[E_j(\tau-\rho)]| \leq \varepsilon/2 \ \forall j \in [m]$, i.e., ρ is in

$$\begin{aligned}\mathcal{P}_\varepsilon = \{\rho\colon\ & \rho \succeq 0\\ & \mathrm{Tr}[\rho] = 1\\ & \mathrm{Tr}[\rho E_j] \leq \mathrm{Tr}[\tau E_j] + \varepsilon/2 \quad \forall j \in [m]\\ & \mathrm{Tr}[-\rho E_j] \leq \mathrm{Tr}[-\tau E_j] - \varepsilon/2 \quad \forall j \in [m]\}.\end{aligned}$$

We call the problem of finding a classical description of τ that suffices to solve the shadow tomography problem without any more samples from τ the *descriptive shadow tomography problem.* In particular if we get a vector y as above, then for a given $j \in [m]$ using $\widetilde{\mathcal{O}}_m(1/\varepsilon^2)$ invocations of a Gibbs-sampler for y, followed by the measurement E_j, suffices to find an ε-approximation of $\mathrm{Tr}[\tau E_j]$ with success probability at least $1 - \mathcal{O}(1/m)$. If we can coherently apply E_j, then using amplitude estimation techniques the number of (coherent) Gibbs-sampler calls can be reduced to $\widetilde{\mathcal{O}}_m(1/\varepsilon)$.

Due to the output size of the shadow tomography problem, a trivial lower bound of $\Omega(m\log(1/\varepsilon))$ can be given on the computational complexity. However, this limitation does not exist for the descriptive shadow tomography problem. Both problems clearly have the same sample complexity, furthermore the best known lower bound on the sample complexity is $\Omega(\log(m)/\varepsilon^2)$ [Aar18].

6.5.1. THEOREM. *The descriptive shadow tomography problem can be solved using*

$$\widetilde{\mathcal{O}}\left(\frac{\log^4(m)\log(n)}{\varepsilon^4}\right)$$

samples from τ. Furthermore, when the E_j matrices are accessible in the quantum operator model this can be done using

$$\widetilde{\mathcal{O}}\left(\left(\sqrt{m} + \frac{\sqrt{n}}{\varepsilon}\right)\frac{\alpha}{\varepsilon^4}\right)$$

queries. It follows that the same bound holds with $\alpha = s$ for the sparse model and with $\alpha = B$ for the quantum state model. When the measurements are given in the quantum state model the query complexity can also be bounded by

$$\widetilde{\mathcal{O}}_n\left(\left(\sqrt{m} + \min\left(\frac{\sqrt{n}}{\varepsilon}, \frac{B^{2.5}}{\varepsilon^{3.5}}\right)\right)\frac{B}{\varepsilon^4}\right).$$

Proof:
The samples from τ are only used for calculating the values b_j, i.e., $(\varepsilon/4)$-approximations of $\mathrm{Tr}[\tau E_j]$, when checking the constraints in the SDP primal oracle. Like in [BKL+18] we make a small adjustment to our SDP primal oracle: when Gibbs-sampling the Gibbs state ρ, we also sample τ to create the state $\rho \otimes \tau$. Then, when checking the constraint, we measure $(E_j \otimes I - I \otimes E_j)/2$ to obtain an approximation of $(\mathrm{Tr}[E_j\rho] - \mathrm{Tr}[E_j\tau])/2$. Notice that our SDP primal oracle uses $\widetilde{\mathcal{O}}\left(\frac{\log^4(m)\log(n)}{\varepsilon^4}\right)$ Gibbs states ($\widetilde{\mathcal{O}}_{\log(n)}\left(\log^4(m)/\varepsilon^2\right)$ in each of the $\mathcal{O}(\log(n)/\varepsilon^2)$ iterations) and hence the modified version uses that many samples from τ too.

The statement about the computational complexity follows directly from Theorem 6.4.9 and 6.4.16. □

Finally note that Lemma 3.3.3 shows that if we can implement the measurements E_j in a controlled fashion on a quantum computer, then we can also implement the corresponding block-encoding with essentially the same cost. Hence the descriptive shadow tomography problem can be solved with the same cost as $(\sqrt{m}+\sqrt{n}/\varepsilon)/\varepsilon^4$ controlled measurements of E_j, if the measurement is performed on a quantum computer as we described above.

6.6 Lower bounds for the new input models

In [vAGGdW17] we showed an $\Omega\left(\sqrt{\max\{n, m\}}\min\{n, m\}^{3/2}\right)$ lower bound for the quantum query complexity of SDP-solving in the sparse input model. For this bound $s = 1$, $\varepsilon = 1/3$ and $R = r = \min\{n, m\}^2$. By letting either n or m be constant, the $\Omega(\sqrt{n} + \sqrt{m})$ lower bound from [BS17] can be recovered. The improved upper bounds of this chapter show that the dependence on n and m is tight up to logarithmic factors. It remains an open question whether a lower bound with an interesting dependence on s and Rr/ε can be proven.

In this section we prove lower bounds for the new input models: the quantum state model and the quantum operator model. To do so, we first prove a lower bound in the Hamiltonian input model, where we can time-evolve under the matrices A_j, see Definition 6.6.1. In all cases the goal is to show that the term $\sqrt{m}/\varepsilon$ times the relevant normalization parameter (for example B in the quantum state model) is necessary in the complexity upper bounds.

6.6.1. Definition (Hamiltonian input model). In the *Hamiltonian* input model for SDPs, we have access to two oracles for the A_j matrices. The first oracle, O_t, gives a classical description of a real vector $t \in \mathbb{R}^j$ in the usual way

$$O_t|j\rangle|0\rangle = |j\rangle|t_j\rangle.$$

The second oracle, O_H, performs the Hamiltonian simulation with A_j for time $1/t_j$:

$$O_H|j\rangle|\psi\rangle = |j\rangle e^{iA_j/t_j}|\psi\rangle$$

Alongside the oracles we also require an upper bound $\tau \geq \max_j t_j$ as part of the input for an SDP. As in the other input models, we assume that we can also apply the inverse of the oracles.

We will use the hybrid-method-based lower bound Theorem 4.1.2 of Chapter 4, in order to lower bound on the number of queries needed for distinguishing different phase oracles. In the spirit of Definition 6.6.1, we will view this as the task of distinguishing a set of diagonal Hamiltonians. The following lower bound follows naturally by reducing this "Hamiltonian discrimination problem" to solving an SDP in the Hamiltonian input model.

6.6.2. Lemma. *Let $\varepsilon \in (0, 1/2]$, $2 \leq m$ and $1 \leq \tau$. Then there is a family of LPs (and hence SDPs) (with $R, r, n = 2$) for which an ε-approximation of the optimal value requires $\Omega(\sqrt{m}\frac{\tau}{\varepsilon})$ queries to O_H in the Hamiltonian input model.*

Proof:
Let $H_1, \ldots, H_m \in \mathbb{R}^{2\times 2}$ be such that

(a) either all H_j are $I/(2\tau)$,

(b) or all but one of the matrices are $I/(2\tau)$, and there is one H_j that is equal to
$$\begin{bmatrix} 1/(2\tau) + \varepsilon/\tau & 0 \\ 0 & 1/(2\tau) - \varepsilon/\tau \end{bmatrix} = \frac{1}{2\tau}I + \frac{\varepsilon}{\tau}Z.$$

Let us assume that we have access to the phase oracle O : $|j\rangle|b\rangle \to e^{i(H_j)_{bb}}|j\rangle|b\rangle$. In case (b) there are m possible different choices for this oracle. It is easy to see by Theorem 4.1.2 that distinguishing case (a) form (b) requires $\Omega\big(\sqrt{m}\frac{\tau}{\varepsilon}\big)$ queries to O.

Now we show that using the above phase oracles we can define an SDP in the Hamiltonian input model. Solving this SDP to ε-precision distinguishes case (a) form (b), proving the sought lower bound.

Let us define $A_j := \tau H_j$ (and $t = (\tau, \ldots, \tau)$), so all the A_j's are either $I/2$ or $I/2 + \varepsilon Z$, furthermore let

$$C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix},$$

and $b = (1, \dots, 1)^T$ the all-one vector. Note that since at least one of the A_j matrices is $I/2$, we know that $R = 2$ suffices as an upper bound on the traces. Furthermore, it is easy to verify from the dual that $r = 2$ suffices as well.

Now we analyze the optimal value. If all A_j matrices are $I/2$ then all constraints are the same:

$$X_{11}/2 + X_{22}/2 \leq 1$$

from which it clearly follows that OPT $= 2$.

If one A_j matrix is not $I/2$, then the constraint

$$(1/2+\varepsilon)X_{11} + (1/2-\varepsilon)X_{22} \leq 1$$

is present. It follows that

$$(1/2+\varepsilon)X_{11} \leq 1 \Rightarrow X_{11} \leq \frac{1}{1/2+\varepsilon}$$

which will clearly be tight in the optimum. Using that

$$2 - 4\varepsilon \leq \frac{1}{1/2+\varepsilon} \leq 2 - 2\varepsilon$$

we conclude that $2 - 4\varepsilon \leq$ OPT $\leq 2 - 2\varepsilon$.

Hence solving this SDP up to ε-precision will distinguishing case (a) form (b) and hence requires $\Omega\big(\sqrt{m}\frac{\tau}{\varepsilon}\big)$ queries. □

To prove the lower bounds for the quantum state model and the quantum operator model we reduce the Hamiltonian input model to them.

6.6.3. Lemma. *Let $\varepsilon \in (0,1]$. Given an SDP in the Hamiltonian input model with parameter $\tau \geq 2$ (for technical reasons also assume that $t_j \geq 2$ for all j), an ε-approximate oracle call in the quantum operator model with $\alpha = 2\tau$ can be simulated using $\widetilde{\mathcal{O}}_\varepsilon(1)$ queries.*

Proof:
For simplicity let us drop the index j. By applying Corollary 3.4.18 we get a $(\frac{\pi}{2}t, 2, \varepsilon)$-block encoding of H with $\mathcal{O}(\log(1/\varepsilon))$ controlled oracle calls. We can then turn this into a $(2\tau, 2, \varepsilon)$ block-encoding of H by Lemma 3.3.9. □

6.6.4. Corollary. *Let $\varepsilon \in (0,1/2]$, $2 \leq m$ and $2 \leq \alpha$. Then there is a family of LPs (and hence SDPs) (with $R, r, n = 2$) for which an ε-approximation of the optimal value requires $\widetilde{\Omega}(\sqrt{m}\frac{\alpha}{\varepsilon})$ queries to O_H in the quantum operator model.*

For the quantum state input model we only give a reduction for LPs, i.e., the case where all input matrices are diagonal.

6.6.5. LEMMA. *Let $\varepsilon \in (0,1]$. Given an LP in the Hamiltonian input model with parameter $\tau \geq 4$ (for technical reasons also assume that $t_j \geq 4$ for all j), then an ε-approximate oracle call in the quantum state model with $B = 2n\tau$ can be simulated using $\widetilde{\mathcal{O}}_\varepsilon(1)$ queries.*

Proof:
For simplicity let us drop the index j. By applying Corollary 3.4.18 we get a $(\frac{\pi}{2}t, 2, \mathcal{O}(\varepsilon))$-block encoding of H with $\mathcal{O}(\log(1/\varepsilon))$ controlled oracle calls. We can then turn this to a $(1, 2, \mathcal{O}(\varepsilon))$ block-encoding of $I/2 \pm H/(4\tau)$ by Lemma 3.3.9. By Corollary 3.4.14 and Corollary 2.3.8 we can turn this into a $(1, 3, \mathcal{O}(\varepsilon))$ block-encoding of $\sqrt{(I + H/(2\tau))/2}$.

The state input oracles can be implemented as follows: controlled on the state $|\pm\rangle$ we apply $\sqrt{(I \pm H/(2\tau))/2}$ to the first half of the state $\sum_{i=1}^n |i\rangle|i\rangle/\sqrt{n}$, resulting in subnormalized density operators $\varrho_\pm = (I \pm H/(2\tau))/(2n)$, so that $\varrho_+ - \varrho_- = H/(2n\tau)$. $\square$

6.6.6. COROLLARY. *Let $\varepsilon \in (0, 1/2]$, $2 \leq m$ and $1 \leq B$. Then there is a family of LPs (and hence SDPs) (with $R, r, n = 2$) for which an ε-approximation of the optimal value requires $\widetilde{\Omega}(\sqrt{m}\frac{B}{\varepsilon})$ queries to O_H in the quantum state model.*

Chapter 7

Quantum Lovász Local Lemma

The Lovász Local Lemma (LLL) is an important and powerful tool in combinatorics and probability theory. It gives a sufficient condition under which the union of some “sparsely dependent” events is guaranteed to have probability less than 1. While early versions of the LLL were non-constructive, Moser [Mos09] gave an algorithmic version in the so-called variable version. The variable version of the Quantum Lovász Local Lemma (QLLL) can be stated in terms of frustration-free local Hamiltonians: these Hamiltonians have the property that their ground state minimizes the energy of all local terms simultaneously. In general, deciding whether a Hamiltonian is frustration-free is a hard task, as it is closely related to the quantum satisfiability problem (QSAT) – the quantum generalization of SAT, the archetypal NP-complete problem in classical computer science. The QLLL provides a sufficient condition for frustration-freeness.

Is there an efficient way to prepare a frustration-free state under the conditions of the QLLL? Previous results showed that the answer is positive if all local terms commute. These works were based on Moser’s “compression argument” which was the original analysis technique of his celebrated classical resampling algorithm. We generalize and simplify the “compression argument”, so that it provides a simplified version of the previous quantum results, and improves on some classical results as well. More importantly, we improve on the previous results by designing a quantum algorithm that works efficiently for non-commuting terms as well, assuming that the system is “uniformly” gapped, by which we mean that the system *and all its subsystems* have an energy gap that is at least inverse polynomially large. Also, our analysis works under the most general condition for the QLLL, known as the Shearer bound, which we describe later in the chapter.

Finally, in the variable version of the LLL we find optimal bounds for the “guaranteed to be feasible” probabilities on any cyclic dependency graph, and show that this region is always strictly larger than in the generic non-variable version, where Shearer’s bound is optimal. This in turn shows a separation between the variable version of the classical and the quantum LLL.

This chapter is based on [GS17] © 2017 IEEE, and [Gil16].

7.1 Introduction

Frustration-free Hamiltonians and quantum satisfiability. Many physical systems and models are described by a local Hamiltonian $H = \sum_i H_i$ where each term H_i is k-local, meaning that it acts non-trivially only on at most k of its subsystems. Such a Hamiltonian is called *frustration-free* if its ground state is also the ground state of each of the local terms H_i. Frustration-free Hamiltonians appear in various areas, for example: quantum error correcting codes [Got96], parent Hamiltonians for PEPS (a 2-D generalization of matrix-product-states) [PVWC08], and various models in many-body quantum physics.

An equivalent way to ask whether a Hamiltonian H is frustration-free is whether $H' = \sum_i \Pi_i$ is frustration-free, where Π_i is the projector on the excited states of H_i. The quantum satisfiability problem[1] (QSAT) is to determine whether H' in the above form is frustration-free or not. QSAT is QMA_1-complete [Bra11],[2] and therefore intractable in general even for quantum computers (unless $\mathsf{BQP} = \mathsf{QMA}_1$). In this work we tackle the search problem – finding a ground state of a frustration-free Hamiltonian – which is, in general, an even harder task than deciding frustration-freeness.[3]

The (spectral) gap of a Hamiltonian H, denoted by $\Delta(H)$, is the energy difference between its two lowest distinct energy levels. The *uniform gap* of a local Hamiltonian $H = \sum_{i=1}^m H_i$ is defined by

$$\gamma(H) := \min_{S \subseteq [m]} \Delta\left(\sum_{i \in S} H_i\right). \tag{7.1}$$

The spectral gap of Hamiltonians plays an important role both in physics and computer science, particularly in Hamiltonian complexity theory, see for example [Has07, FGGS00, AvDK+08, CPGW15].

The Classical and Quantum Lovász Local Lemma. We would like to understand the QSAT problem, so it is natural to first look at the classical SAT and the techniques that were useful in studying it. A "local" version of SAT is called k-SAT. It asks whether a Boolean formula of the following form can be satisfied: $\bigwedge_{i \in [m]} c_i$, where each c_i is a *clause* containing the OR ($\bigvee$) of exactly k distinct Boolean variables or their negation.

A natural question is, when can we be sure that a satisfying assignment exists? Since each k-SAT constraint excludes a $p = 2^{-k}$ fraction of assignments, $pm < 1$

[1] For technical reasons, one might require a promise that if H' is not frustration-free, the minimal energy of H' is at least inverse-polynomial in the number of qubits.

[2] QMA_1 is similar to QMA (Definition 3.2.14), except that it requires perfect completness.

[3] SAT (as well as any other NP-complete problem) has a search-to-decision reduction [BG94]. No such reduction is known for QSAT.

is a sufficient condition (by the union bound). If we have the additional information that none of the constraints share variables, then the formula is clearly satisfiable. What can we say in the intermediate regime, where each constraint shares variables with at most d constraints (including itself)? The (symmetric) Lovász Local Lemma [EL75, AS92, KST93, Sze13], applied to this setting, implies that the (symmetric) Lovász condition

$$pde \leq 1 \tag{7.2}$$

is a sufficient condition for satisfiability.[4] Shearer generalized the Lovász Local Lemma and found the optimal sufficient condition in this framework [She85], see Section 7.4.

How hard is it to *find* such a satisfying assignment? A series of works [Bec91, Mos09, MT10, KSz11] have culminated in an efficient constructive algorithm.

It is natural to ask the analogous questions in the quantum setting, where the Boolean variables are replaced by qubits and the clauses by rank-1 k-local projectors. The connection between a k-SAT clause and a rank-1 projector is the following: a k-SAT clause excludes one out of the 2^k possible configurations of the relevant variables, while a rank-1 k-local projector excludes one dimension out of the 2^k relevant dimensions. So given a set of k-local rank-1 projectors acting on n qubits[5], under what conditions can we guarantee that the system is frustration-free? A "dimension-counting" argument can be used to show that the Lovász condition ($pde \leq 1$) [AKS12] is indeed sufficient, as is Shearer's condition [SMLM16].

Is there an algorithm which efficiently prepares a ground state under these conditions? In the past, such constructions have been achieved only for commuting Hamiltonians, where $[\Pi_i, \Pi_j] = 0$ for all i, j. Commuting Hamiltonians are somewhat "half-way" between classical and quantum. For example, the commuting 2-local Hamiltonian problem is in (the purely classical class) NP for qudits of all dimensions [BV05], whereas 2-local QSAT is QMA_1-complete if the dimension of the qudits is large enough [AGIK09]. Yet commuting Hamiltonians, such as the toric code, can have the striking quantum property of topological order [Kit03]. In this work we extend the previous results to non-commuting projectors, thereby entering the fully quantum regime.

Moser-Tardos type resampling algorithm. Following the seminal work of Moser and Tardos [Mos09, MT10], a variety of algorithms and analysis techniques were introduced for proving efficient versions of the Lovász Local Lemma based on their resampling algorithm. The resampling algorithm starts with a random state, and repeatedly checks the constraints that we want to satisfy. If

[4]The constant e in Eq. (7.2) is $2.71\ldots$, the base of the natural logarithm.

[5]The uniform k-locality and rank-1 constraints are only for convenience, for a more general treatment see our paper [GS16].

a constraint c is violated, then it performs a "resampling", which is some random "local" change to the current state only affecting c and a few other constraints, hopefully fixing c. Once all constraints are fixed, the algorithm returns a satisfying state. The main challenge is the analysis of the algorithm: proving a bound on the expected number of resamplings needed. In Algorithm 7.1 we present a meta-algorithm sketching this procedure, which captures the basic structure of most related algorithms.

The algorithm can be interpreted both as a classical and as a quantum algorithm. For example, in the case of SAT, the initial state is n uniformly random Boolean variables, and a constraint is simply a clause, which is simple to check by looking at the corresponding Boolean variables. In the quantum setting of QSAT, the initial state is similarly n uniformly randomly initialized 0/1 qubits (which is the maximally mixed state). A constraint c corresponds to an orthogonal projector Π_c. We say that $|\psi\rangle$ *satisfies* the constraint c if the quantum state is in the kernel of Π_c, and that c is *unsatisfied* if $|\psi\rangle$ is not in the kernel. Finally we say that $|\psi\rangle$ *violates* c if it is in the image of Π_c.

Algorithm 7.1 Moser-Tardos resampling meta-algorithm

```
    input: set of constraints $C$
 1: initialize system to a uniformly random starting state
 2: $F \leftarrow \varnothing$ ($F$ is the set of fixed (i.e., satisfied) clauses)
 3: while $F \neq C$ do
 4:     pick $c \in C \setminus F$ and check if constraint $c$ is satisfied
 5:     if "Satisfied"
 6:         update $F \leftarrow F \cup \{c\}$
 7:     else if "Violated"
 8:         resample $c$ (and thereby hopefully fix it)
 9:         update $F \leftarrow F \setminus \Gamma^+(c)$    ($\star$ $\Gamma^+(c)$ denotes the constraints possibly
    affected by resampling $c$, including $c$ itself $\star$)
10: end while
```

In Algorithm 7.1, **pick** and **check** in line 4 and **resample** in line 8 need to be specified in order to get a well-defined algorithm. In this chapter, all the results apply for any deterministic strategy for executing **pick**, see Definition 7.2.2. In order to get improved bounds, up to the optimal Shearer bound [She85], one might need to be more careful regarding **pick**, for more details see the paper on which this chapter is based [GS16]. In this article we mostly work in the so-called variable framework [Mos09, MT10, KSz11], which is sufficient for the SAT and QSAT applications. In this setting each constraint depends on some (qu)bits of the system. (For simplicity we will only consider systems of n qubits, but all the results generalize trivially to qudits.) In this binary variable framework we simply define **resample** as reinitializing the specific constraint's (qu)bits to uniformly random true-false (0-1) values. We define $\Gamma^+(c)$ as the set of constraints c' such that c and c' both act non-trivially on some shared (qu)bit, and $d = \max_{c\in C}|\Gamma^+(c)|$. In general one could also work with other models, as described in [HV15].

The **check**ing step in line 4 should be performed using some measurement operator, corresponding to Π_c. The algorithm implicitly assumes that all the constraints in F are fixed (satisfied). This loop invariant is easy to maintain in the classical and commuting quantum case by implementing **check** using the two-outcome measurement $\{\Pi_c, \mathrm{Id} - \Pi_c\}$. But, in the non-commuting setting, using this two-outcome measurement can break the loop invariant: suppose that all the constraints in F are fixed, and then another constraint Π_c is checked (i.e., measured) and is found to be satisfied. A constraint which was fixed before, and shares a qubit with Π_c, may become unsatisfied because of the collapse caused by the measurement. Because of this caveat the analysis of the previous quantum algorithms [SCV13, SA15] worked only in the commuting case. Next, we explain how to maintain this loop invariant also in the non-commuting case.

The progressive measurement channel. We first need one more notation. We denote by Π^F the projection onto $\ker(\sum_{c\in F}\Pi_c)$. Note that for $c \in C$ the states in the image of Π_c violate c, whereas for $F \subseteq C$ the states in the image of Π^F satisfy all $c' \in F$, e.g., $\Pi^{\{c\}} = I - \Pi_c$. We changed from sub- to superscript to help avoid confusion caused by this difference.

Suppose that $|\psi\rangle$ satisfies all the constraints in F, i.e., $|\psi\rangle = \Pi^F|\psi\rangle$, and $\{\Pi^{F\cup\{c\}}, \mathrm{Id} - \Pi^{F\cup\{c\}}\}$ is measured. The unnormalized post-measurement state associated with outcome $\Pi^{F\cup\{c\}}$ is $\Pi^{F\cup\{c\}}|\psi\rangle$, which we obtain with probability $\langle\psi|\Pi^{F\cup\{c\}}|\psi\rangle$. If instead $\{\Pi_c, \mathrm{Id} - \Pi_c\}$ is measured, the post-measurement state $|\varphi\rangle$ associated with outcome Π_c has the property that $|\varphi\rangle = \Pi_c|\varphi\rangle$, and due to locality also $|\varphi\rangle = \Pi^{F\setminus\Gamma^+(c)}|\varphi\rangle$. One of our key observations is that the outcomes $\Pi^{F\cup\{c\}}$ and Π_c are in some sense complementary to each other.

We call a quantum channel a *progressive measurement channel*, if it combines these two properties:[6] for an input state $|\psi\rangle \in \mathrm{im}(\Pi^F)$, it has two classically labeled outputs (corresponding to measurement labels): the "Satisfied" output is $\Pi^{F\cup\{c\}}|\psi\rangle$, and the "Violated" output is ρ such that $\rho = \Pi_c\rho = \Pi^{F\setminus\Gamma^+(c)}\rho$ and $\mathrm{Tr}[\rho] = 1 - \left\|\Pi^{F\cup\{c\}}|\psi\rangle\right\|^2$. Here, the name *progressive* is used to emphasize that for a state which satisfies F, the channel either adds c to the set of fixed constraints, or provides a state in which c is violated (but the state ρ keeps at least $F \setminus \Gamma^+(c)$ satisfied).

We devised two different, but closely related constructions, which satisfy the requirements of a progressive measurement channel. In Section 7.2.3 we show

[6]The formal definition is slightly different, and is adapted for our needs, see Definition 7.2.7. The progressive measurement channel that we discuss in Section 7.2.3 satisfies these two properties.

an explicit procedure and prove that it is a progressive measurement channel. The construction itself and its analysis are fairly simple. In Section 7.3 we show how to efficiently implement this progressive measurement channel using singular value transformation techniques under a promise on the uniform gap. In the paper [GS17], on which this chapter is based, we have another construction which can be implemented using only (weak) measurements, but which has a quadratically worse gap-dependence.

The main idea of the latter variant is to use weak measurements coupled with a quantum Zeno effect[7]. This variant uses only Π_c and Π^F measurements, and the number of measurements it performs depends on the spectral gap of $\sum_{c'\in F\cup\{c\}}\Pi_{c'}$. It repeats the following T times: (strongly) measure Π^F, followed by a weak measurement of Π_c. If Π_c is found to be violated, we immediately return with the classical "Violated" label. If we ever get measurement outcome $I-\Pi^F$, then we immediately abort, otherwise we return "Satisfied". We show that by choosing the weak measurement parameter to be weak enough, the probability of "abort" becomes proportionally small. Also if we choose T to be large enough, then the procedure closely approximates a progressive measurement channel. Finally we show how to appropriately approximate a Π^F measurement by repeated $\Pi_{c'}$ measurements for $c'\in F$.

We think that the definition and efficient construction of a progressive measurement channel could be of independent interest, and might find applications in other quantum algorithms.

New existential proof. Our work does not require any of the previous existential proofs, and therefore provides an alternative proof for the main results in [AKS12] and [SMLM16].

Our contributions. We present three main results in this chapter.

Our first contribution is the adaptation of the "forward-looking" analysis technique of [HV15] to the quantum setting, which enables the generalization for the non-commuting case, and makes it possible to extend the previous commuting results up to Shearer's bound (see our paper [GS16]). This is done via our Key Lemma 7.2.8, which borrows ideas from [HV15, Kol16]. It is proved using semidefinite inequalities which introduce quantum analogues of uniform probability bounds.

Our second contribution is the generalization and simplification of Moser's "entropy compression argument" [Mos09] that was originally used for proving efficiency of the classical resampling algorithm. This generalization simplifies the proof of the previous commuting quantum results from [SCV13, SA15]. On top of the quantum implications, it also improves the runtime analysis of some

[7]The quantum Zeno effect is a quantum technique which uses frequently repeated measurements to prevent unwanted changes in the quantum state of some quantum system [MS77].

classical algorithms, see the discussion in Section 7.2.1. Last, but not least, it gives valuable insight through the "Log compression" Lemma 7.2.3 showing that the core of the "entropy compression argument" can be distilled to a straightforward counting argument.

Our third and most important contribution is that we prove a constructive Quantum Lovász Local Lemma for *non-commuting* projectors. We construct the appropriate progressive measurement channel which can handle the non-commuting case in a way suggested by our Key Lemma 7.2.8.

The algorithm's running time is polynomial in the number of the constraints and qubits, but also depends inverse-polynomially on the uniform gap, see Eq. (7.1) and the discussion there. The main open question left is whether this dependency on the uniform gap is necessary. Specifically, given a Hamiltonian H which satisfies the Lovász condition, and an energy bound ϵ, is there a quantum algorithm which can output a state with energy at most ϵ in time $poly(n, |C|, 1/\epsilon)$? (The running time should not depend on any gap promise for the Hamiltonian.)

7.2 The ideal quantum algorithm

7.2.1 Generalized compression argument

The generalized compression argument that we present makes the proof significantly simpler compared to the original work of Moser [Mos09], probably providing the simplest known proof of any Moser-Tardos type algorithm. It works for any deterministic constraint-selection rule, and can be applied beyond the variable framework [HV15].

7.2.1. Definition (Logs). The log of the first T steps of Algorithm 7.1 is a string $L \in \{S, V\}^T$ containing the first T outcomes of **check** (where S stands for "Satisfied", and V for "Violated").

Let $\mathcal{L}^{(r)}$ denote the set of all valid logs which contain exactly r V's, and end with a V.

7.2.2. Definition. (Constraint-selection Strategy) A deterministic constraint-selection strategy is a function s, which given the current log L, determines which next constraint to **pick** at line 4 of Algorithm 7.1.

7.2.3. Lemma. *(Log compression) Suppose we run Algorithm 7.1 using a deterministic constraint-selection strategy. Then the log uniquely encodes the sequence of resamples that happened during the algorithm. Moreover, if $\Gamma^+(c) \leq d$ for all $c \in C$, then for all $r \in \mathbb{N}$ we have $|\mathcal{L}^{(r)}| \leq \binom{|C|+rd}{r}$.*

Proof:
Since the constraint-selection strategy is deterministic and initially $F = \emptyset$, we can

recover the content of the set F after each execution of the main loop at line 3 by only looking at the binary log telling us whether a resampling happened or not. Therefore the log compresses the whole resample history into a binary string.

Now observe that any log $L \in \mathcal{L}^{(r)}$ contains at most $|C| + rd$ entries: Suppose the algorithm performed $k - 1$ steps before the r-th resampling. At this step $0 < |C \setminus F|$ since a resampling is performed at the k-th step. On the other hand F starts with 0 elements, and gains one element with the $k - r$ successful checks, and loses at most $d - 1$ elements after each resampling. Therefore $|C \setminus F| \leq |C| - (k - r) + (r - 1)(d - 1) \leq |C| + rd - k$ and so $k < |C| + rd$.

Finally we map each $L \in \mathcal{L}^{(r)}$ to a binary string of length $|C| + rd$ by extending it with "S"s. Note that this mapping is injective, and observe that the number of length $(|C| + dr)$ binary sequences containing r "V"s is $\binom{|C|+dr}{r}$, which by injectivity proves the desired upper bound on $|\mathcal{L}^{(r)}|$. □

7.2.4. Theorem. *Let* $d = \max_{c \in C} |\Gamma^{+}(c)|$. *Suppose we run Algorithm 7.1 using a deterministic constraint-selection strategy, and in each step we log the constraint that we checked and whether it was satisfied or not. Let* $L_k = \ell_1, \ell_2, \ldots, \ell_k$ *denote the log obtained during the first* k *steps. Let* $r = 4|C|$. *If*

(i) $pde \leq 1$, *and*

(ii) $\Pr($*seeing a specific log* $L_k \in \mathcal{L}^{(r)}$ *during a run*$) \leq p^r$

then Algorithm 7.1 terminates with constant probability making less than $4|C|$ *resamplings. If also*

(iii) during the algorithm the constraints in F *remain fixed*[8] *(i.e., satisfied),*

then upon termination Algorithm 7.1 provides a satisfying state.

Proof:
Suppose we set a bound $r = 4|C|$ on the number of resamplings, such that we terminate with "timeout" upon the r-th resampling. The "Log compression" Lemma 7.2.3 shows that the number of logs that we might obtain at "timeout" is at most $\binom{|C|+rd}{r}$. Using the bound

$$\forall k, n \in \mathbb{N} \colon k < n \Longrightarrow \binom{n}{k} \leq \left(\frac{en}{k} - \frac{e}{2}\right)^{k} \tag{7.3}$$

from Appendix 7.A, we upper bound $\binom{|C|+rd}{r}$ by $(d - 1/4)^r e^r$. Combining this with (ii) using the union bound, we can see that the probability of termination

[8]This requirement is mostly trivial in the classical case, since constraints can only appear after resamplings, which is handled by Algorithm 7.1. But in the non-commutative quantum case it becomes problematic, as was discussed in the introduction.

with "timeout" is at most

$$p^r\left(d-\frac{1}{4}\right)^r e^r \overset{(i)}{\leq} \left(\frac{p(d-1/4)e}{pde}\right)^r = \left(1-\frac{1}{4d}\right)^r \leq e^{-\frac{r}{4d}} \leq \frac{1}{e}.$$

Finally, note that if Algorithm 7.1 terminates normally (without "timeout") then $F = C$, and by (iii) it means that the final state is a satisfying state. $\square$

7.2.5. Remark. Since every randomized strategy is a convex combination of deterministic strategies, the above theorem implies that for any randomized constraint-selection strategy the probability of performing at least $4|C|$ resamplings is also at most $1/e$.

Theorem 7.2.4 gives a fast algorithm whenever the conditions (i)-(iii) are met. It is easy to show that properties (ii)-(iii) hold for the classical variable setting for p which is the maximal probability of encountering a constraint in the uniformly random distribution (so, for example, in a k-SAT formula, $p = 2^{-k}$), and even for more general settings if an appropriate resampling procedure is used, e.g., as in [HV15]. This improvement partially answers an open question posed in [HV15], by providing an improved upper bound on the number of resamplings for the case of the symmetric Lovász condition.

In the quantum case, we can choose p to be the maximal probability, over all constraints $c \in C$, that c is violated in the maximally mixed state (so, for example, in a k-QSAT formula, $p = 2^{-k}$). In the commuting case, if **check** is performed using standard projective measurements of the constraint projectors, then (ii) and (iii) hold (see Proposition 7.2.13), and therefore Theorem 7.2.4 implies the results of [SCV13, SA15]. Our proof is not only simpler, but due to the use of our optimized bound (7.3), our result does not require a slack in the condition $pde \leq 1$. (As shown above, we require slack in the condition $p(d-1/4)e \leq 1$, which can actually be pushed to be a slack in $p(d-1/2)e \leq 1$.) Since property (ii) holds even in the non-commuting case, the algorithm is guaranteed to terminate under the Lovász condition (i.e., when property (i) is satisfied), but the problem is that the output may not be satisfying for all constraints.

7.2.2 The progressive measurement channel and the key lemma

To adapt the algorithm to the quantum setting we introduce a quantum channel $\mathcal{M}_c^F$, which performs some quantum operation on the n-qubit quantum register determined by the classical input (F, c), where F is the set of already "fixed" constraints, and c is the next constraint to address. In the case of commuting projectors $\mathcal{M}_c^F$ will be simply the application of a projective measurement $(\Pi_c, \mathrm{Id} - \Pi_c)$ where the classical measurement outcomes are labeled with (V, S) standing for ("Violated", "Satisfied") respectively.

7.2.6. Definition. (Quantum-classical states) For the description of quantum-classical states consisting of an N-dimensional quantum system and a k-dimensional classical system we are going to use elements of $\mathbb{C}^{N\times N}\otimes\mathbb{R}^k$. We can interpret these as quantum states of restricted form via defining an embedding of $\mathbb{R}^k$ into $\mathbb{C}^{k\times k}$ using diagonal matrices.

For $c\in C$ let $b(c)\subseteq[n]$ be the set of qubits on which Π_c acts non-trivially. Let Π_c^{loc} denote Π_c restricted to $b(c)$, so that we can write $\Pi_c=\Pi_c^{loc}\otimes\mathrm{Id}_{[n]\setminus b(c)}$ (up to ordering of the qubits).

7.2.7. Definition. We say that $\mathcal{M}$ is a *progressive measurement channel* if the following holds: Conditional on receiving classical information $F\subseteq C$ and $c\in C$, the quantum channel $\mathcal{M}_c^F$ performs the quantum operation $\mathcal{M}_c^F:\mathbb{C}^{N\times N}\to\mathbb{C}^{N\times N}\otimes\mathbb{R}^2$, satisfying the following properties:

(i) The quantum channel labels its output with the classical labels (S,V) corresponding to (“Satisfied”, “Violated”) outcomes, so that for input ρ the output state is written as:
$\mathcal{M}_c^F(\rho)=\mathcal{M}_{c,S}^F(\rho)\otimes S+\mathcal{M}_{c,V}^F(\rho)\otimes V$.

(ii) For the (unnormalized) input state Π^F, the output state labeled as “Satisfied” is upper bounded by $\Pi^{F\cup\{c\}}$:
$\mathcal{M}_{c,S}^F(\Pi^F)\preceq\Pi^{F\cup\{c\}}$.

(iii) For the input state Π^F, the output state labeled as “Violated” is upper bounded by a state of tensor product form:
$\mathcal{M}_{c,V}^F(\Pi^F)\preceq\Pi_c^{loc}\otimes\tilde{\Pi}^{F\setminus\Gamma^+(c)}$, where $\Pi^{F\setminus\Gamma^+(c)}=\mathrm{Id}_{b(c)}\otimes\tilde{\Pi}^{F\setminus\Gamma^+(c)}$.

One might be puzzled why it is important to transform states to the “Violated” image of Π_c. (The weaker alternative to property **(iii)** would be $\mathcal{M}_{c,V}^F(\Pi^F)\preceq\Pi^{F\setminus\Gamma^+(c)}$. Since the qubits in c are resampled after c is found to be unsatisfied, it might not be immediately clear why we set any conditions on these qubits.) The reason is that it ensures that the resampling operation uniformly mixes quantum states, for more details see the proof of Lemma 7.2.8. The **resampling operation** on ρ in line 8 can be formally described as

$$R_c(\rho)=\mathrm{Tr}_{b(c)}[\rho]\otimes\frac{\mathrm{Id}_{b(c)}}{2^k}. \tag{7.4}$$

In order to state and prove the Key Lemma, we need to define several concepts. For a log L, let ρ_L denote the unnormalized quantum state after having seen and processed all measurement results in L, i.e., including the resampling step in line 8 if the last result was “V”. Let F_L denote the inner variables F of Algorithm 7.1 after it has seen and processed all the measurement results described by L. Moreover, for $X\in\{S,V\}$ let $(L,X)\in\{S,V\}^{T+1}$ be the log obtained by appending X to the end of log L. If the algorithm did not terminate after L, then let c_L denote the next constraint Algorithm 7.1 will address.

7.2.8. LEMMA. *(Key lemma) If we run Algorithm 7.1 using a progressive measurement channel $\mathcal{M}$, then for every log L which contains r occurrences of V,*

$$\rho_L \preceq p^r \cdot \frac{\Pi^{F_L}}{N}, \tag{7.5}$$

where $N = 2^n$.

Proof:
We prove (7.5) for a log $L \in \{S, V\}^T$ by induction on T. For $T = 0$ we have $\rho_L = \rho_0 = \mathrm{Id}/N$, $\Pi^{F_L} = \mathrm{Id}$ and $p^r = p^0 = 1$, so the relation holds with equality. Now suppose that (7.5) holds for all logs $L \in \{S, V\}^T$. For the induction step it is enough to show that (7.5) also holds for (L, S) and (L, V), whenever (L, S) and (L, V) are valid logs. Let us denote by r the number of "V"s in L, $F = F_L$, $F_S = F_L \cup \{c_L\}$, $F_V = F_L \setminus \Gamma^+(c_L)$ and $c = c_L$. Observe $F_{(L,S)} = F_S$ and $F_{(L,V)} = F_V$. First we show the inductive step for (L, S):

$$\begin{aligned}
\rho_{(L,S)} =& \mathcal{M}^F_{c,S}(\rho_L) && \text{(by definition)}\\
\preceq& \mathcal{M}^F_{c,S}\left(p^r \cdot \frac{\Pi^F}{N}\right) && \text{(by the inductive hypothesis)}\\
\preceq& p^r \cdot \frac{\Pi^{F_S}}{N} && \text{(by property } \textbf{(ii)}\text{)}\\
=& p^r \cdot \frac{\Pi^{F_{(L,S)}}}{N} && (F_{(L,S)} = F_L)
\end{aligned}$$

Indeed, the number of violations in (L, S) remains r.

Now we show the inductive step for (L, V):

$$\begin{aligned}
\rho_{(L,V)} =& R_c\big(\mathcal{M}^F_{c,V}(\rho_L)\big) && \text{(by definition)}\\
\preceq& R_c\left(\mathcal{M}^F_{c,V}\left(p^r \cdot \frac{\Pi^F}{N}\right)\right) && \text{(induction hypothesis)}\\
\preceq& \frac{p^r}{N} \cdot R_c\left(\Pi^{loc}_c \otimes \tilde{\Pi}^{F_V}\right) && \text{(by property } \textbf{(iii)}\text{)}\\
=& \frac{p^r}{N}\mathrm{Tr}[\Pi^{loc}_c] \cdot \frac{\mathrm{Id}_{b(c)}}{2^k} \otimes \tilde{\Pi}^{F_V} && \text{(Eq. (7.4))}\\
=& \frac{p^{r+1}}{N} \cdot \mathrm{Id}_{b(c)} \otimes \tilde{\Pi}^{F_V} && (\mathrm{Tr}(\Pi^{loc}_c) = 1,\ p = \frac{1}{2^k})\\
=& \frac{p^{r+1}}{N} \cdot \Pi^{F_V} && \text{(by property } \textbf{(iii)}\text{)}\\
=& \frac{p^{r+1}}{N} \cdot \Pi^{F_{(L,V)}} && (F_{(L,V)} = F_V)
\end{aligned}$$

Note that the number of violations in (L, V) is $r + 1$, as required. $\square$

Taking the trace of Eq. (7.5) shows property (ii) in Theorem 7.2.4 for a progressive measurement channel, and property (iii) in Theorem 7.2.4 follows from Definition 7.2.7-(**iii**). Therefore, the only missing ingredient for an efficient algorithm is to efficiently implement a progressive measurement channel. This is done in two steps: we next define a progressive measurement channel (Definition 7.2.9), and later, in Section 7.3 show how to implement it.

7.2.3 The exact measurement channel

We are now ready to provide the first explicit construction of a progressive measurement channel, which we call the *exact measurement channel*. We argue that this is probably the most faithful generalization of the commuting algorithm for the non-commuting case. The proposed quantum operation applies a measurement conditionally followed by a unitary operation. The combined procedure respects the loop-invariant, and handles new constraints in a way which seems essential for the resampling algorithm.

7.2.9. Definition. We define the *exact measurement channel*, denoted here by $\mathcal{M}$, in the following way: conditional on receiving classical information $F \subseteq C$ and $c \in C$, the quantum channel $\mathcal{M}_c^F : \mathbb{C}^{N\times N} \to \mathbb{C}^{N\times N} \otimes \mathbb{R}^2$ performs the projective measurement $\big(\Pi^{F\cup\{c\}}, \mathrm{Id} - \Pi^{F\cup\{c\}}\big)$. If the outcome is $\Pi^{F\cup\{c\}}$, then it labels its output with S ("Satisfied"). If the outcome is $\mathrm{Id} - \Pi^{F\cup\{c\}}$, then it labels its output with V ("Violated"), and applies the unitary operation $WU^\dagger$, where $W\Sigma U^\dagger$ is a singular value decomposition of $\Pi_c\Pi^F$.[9] For the output state corresponding to pure input state $|\psi\rangle$ we use notation $\mathcal{M}_c^F(|\psi\rangle) = |\psi_S\rangle \otimes S + |\psi_V\rangle \otimes V$, where $|\psi_S\rangle = \Pi^{F\cup\{c\}}|\psi\rangle$ and $|\psi_V\rangle = WU^\dagger\big(\mathrm{Id} - \Pi^{F\cup\{c\}}\big)|\psi\rangle$.

To keep things conceptually simple, in the following we present an example where we calculate some of the important maps explicitly, although it diverges from some of the conditions and assumptions we had before, namely the Lovász condition does not hold, one of the projectors is not rank-1, and it uses a qudit (not a qubit).

7.2.10. Example. Consider a qudit of of dimension 4, and the following 2 projectors: $\Pi_1 = |0\rangle\langle 0|$, $\Pi_2 = \frac{1}{2}(|0\rangle + |1\rangle)(\langle 0| + \langle 1|) + |2\rangle\langle 2|$. Let $F = \{1\}$, $c = 2$. In this case, $\Pi_c\Pi^F = \Pi_2\Pi^{\{1\}} = \frac{1}{2}(|0\rangle + |1\rangle)\langle 1| + |2\rangle\langle 2|$. A possible choice for W and $U^\dagger$ yields $WU^\dagger = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\langle 1| + \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)\langle 0| + |2\rangle\langle 2| + |3\rangle\langle 3|$. $\Pi^{F\cup\{c\}} = \Pi^{\{1,2\}} = |3\rangle\langle 3|$. The state $|\psi\rangle = \frac{1}{\sqrt{3}}(|1\rangle + |2\rangle + |3\rangle)$ satisfies F: $\Pi^F|\psi\rangle = |\psi\rangle$. The "Satisfied" post-measurement state is $|\psi_S\rangle = \frac{1}{\sqrt{3}}|3\rangle$, which

[9] There is a choice of W and $U^\dagger$ in the SVD decomposition, such that for $R = WU^\dagger$ we get $\Pi^{F\cup\{c\}} = R\Pi^{F\cup\{c\}}R^\dagger$. In this case, the unitary R can be applied after both "Satisfied" and "Violated" outcomes, but here we apply it only after "Violated" outcomes for convenience.

occurs with probability $\frac{1}{3}$. The "Violated" post-measurement state is

$$
\begin{aligned}
|\psi_V\rangle = WU^\dagger(\mathrm{Id} - \Pi^{F\cup\{c\}}|\psi\rangle) &= WU^\dagger\frac{1}{\sqrt{3}}(|1\rangle + |2\rangle) \\
&= \frac{1}{\sqrt{3}}\left(\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) + |2\rangle\right) \qquad (7.6)
\end{aligned}
$$

which occurs with probability $\frac{2}{3}$.

In the rest of this subsection, we show in Lemma 7.2.12 that the exact measurement channel is a progressive measurement channel (and therefore respects the loop invariants (**ii**)-(**iii**) in Definition 7.2.7). Then in Proposition 7.2.13 we show that in the commuting case, the exact measurement channel can be implemented by simply measuring $\{\mathrm{Id}-\Pi_c, \Pi_c\}$. This shows that the exact measurement channel is a generalization of the simple projective measurement that is performed in the classical and commuting algorithms [MT10, SCV13, SA15]. Before we prove these results, we need some identities of the relevant subspaces.

7.2.11. Proposition. *Suppose $W\Sigma U^\dagger$ is a singular value decomposition of $\Pi_c\Pi^F$ (i.e., $\Pi_c\Pi^F = W\Sigma U^\dagger$ with $W^\dagger = W^{-1}$, $U^\dagger = U^{-1}$ and Σ non-negative real and diagonal), then the following identities hold:*

$$
\begin{aligned}
\Pi_{\mathrm{im}(\Pi_c\Pi^F)} &= W\mathrm{sign}(\Sigma)W^\dagger && (7.7)\\
\Pi^F - \Pi^{F\cup\{c\}} &= U\mathrm{sign}(\Sigma)U^\dagger && (7.8)\\
\Pi_{\mathrm{im}(\Pi_c\Pi^F)} &\preceq \Pi_c\Pi^{F\setminus\Gamma^+(c)} && (7.9)
\end{aligned}
$$

In the above, $\mathrm{sign}(\Sigma)$ uses the natural extension of the sign function to diagonal (and in this case, non-negative) matrices, moreover for a subspace S we denote by Π_S the orthogonal projector to S.

Proof:

We prove the three properties (7.7)-(7.9) one-by-one:

(7.7): Note $\big(W\mathrm{sign}(\Sigma)W^\dagger\big)\Pi_c\Pi^F = \big(W\mathrm{sign}(\Sigma)W^\dagger\big)W\Sigma U^\dagger = W\Sigma U^\dagger = \Pi_c\Pi^F$. As $W\mathrm{sign}(\Sigma)W^\dagger$ is an orthogonal projector it implies $\Pi_{\mathrm{im}(\Pi_c\Pi^F)} \preceq W\mathrm{sign}(\Sigma)W^\dagger$. But also $\mathrm{rank}\big(\Pi_{\mathrm{im}(\Pi_c\Pi^F)}\big) = \mathrm{rank}\big(\Pi_c\Pi^F\big) = \mathrm{rank}\big(W\mathrm{sign}(\Sigma)W^\dagger\big)$, thus $\Pi_{\mathrm{im}(\Pi_c\Pi^F)} = W\mathrm{sign}(\Sigma)W^\dagger$, which concludes the proof.

(7.8): Similarly to (7.7) $\Pi_{\mathrm{im}(\Pi^F\Pi_c)} = U\mathrm{sign}(\Sigma)U^\dagger$, so it is enough to show that $\Pi^F - \Pi^{F\cup\{c\}} = \Pi_{\mathrm{im}(\Pi^F\Pi_c)}$. Again $\Pi_{\mathrm{im}(\Pi^F\Pi_c)} \preceq \Pi^F - \Pi^{F\cup\{c\}}$, since

$$\big(\Pi^F - \Pi^{F\cup\{c\}}\big)\Pi^F\Pi_c = \Pi^F\Pi_c - \Pi^{F\cup\{c\}}\Pi_c = \Pi^F\Pi_c.$$

But

$$\begin{aligned}
\operatorname{rank}\big(\Pi_{\mathrm{im}(\Pi^F\Pi_c)}\big) &= \operatorname{rank}\big(\Pi^F\Pi_c\big)\\
&= \operatorname{rank}\big(\Pi_c\Pi^F\big)\\
&= \operatorname{rank}\big(\Pi^F\big) - \dim(\ker(\Pi_c)\cap\mathrm{im}(\Pi^F))\\
&= \operatorname{rank}\big(\Pi^F\big) - \operatorname{rank}\big(\Pi^{F\cup\{c\}}\big)\\
&= \operatorname{rank}\big(\Pi^F - \Pi^{F\cup\{c\}}\big).
\end{aligned}$$

Here, the second equality is justified by $\operatorname{rank}(A) = \operatorname{rank}\big(A^\dagger\big)$, and the third equality by $\operatorname{rank}(AB) = \operatorname{rank}(B) - \dim(\ker(A)\cap\mathrm{im}(B))$ (see, e.g. [Mey00, p. 210]).

So $\Pi^F - \Pi^{F\cup\{c\}} = \Pi_{\mathrm{im}(\Pi^F\Pi_c)}$ and thus $\Pi^F - \Pi^{F\cup\{c\}} = U\mathrm{sign}(\Sigma)U^\dagger$.

(7.9): The proof follows from the following line of (in)equalities which are justified below:

$$\begin{aligned}
\Pi_{\mathrm{im}(\Pi_c\Pi^F)} &= \Pi_c\Pi_{\mathrm{im}(\Pi_c\Pi^F)}\Pi_c\\
&\preceq \Pi_c\Pi^{F\setminus\Gamma^+(c)}\Pi_c\\
&= \Pi_c^2\Pi^{F\setminus\Gamma^+(c)}\\
&= \Pi_c\Pi^{F\setminus\Gamma^+(c)}.
\end{aligned}$$

First observe that $\Pi_c\big(\Pi_c\Pi^F\big) = \Pi_c\Pi^F$ so $\Pi_c\Pi_{\mathrm{im}(\Pi_c\Pi^F)} = \Pi_{\mathrm{im}(\Pi_c\Pi^F)}$, implying the first equality. The penultimate equality is due to $\Pi_c\Pi^{F\setminus\Gamma^+(c)} = \Pi^{F\setminus\Gamma^+(c)}\Pi_c$, which follows from the fact that these operators act on disjoint qubits. Finally note that $\Pi^{F\setminus\Gamma^+(c)}\Pi^F = \Pi^F$. Therefore, $\Pi^{F\setminus\Gamma^+(c)}\big(\Pi_c\Pi^F\big) = \Pi_c\Pi^{F\setminus\Gamma^+(c)}\Pi^F = \Pi_c\Pi^F$ so $\Pi_{\mathrm{im}(\Pi_c\Pi^F)} \preceq \Pi^{F\setminus\Gamma^+(c)}$, which justifies the inequality. □

Using the above proposition we can easily show in the following lemma that the exact measurement channel is indeed progressive (see Definition 7.2.7).

7.2.12. LEMMA. *Suppose $|\psi\rangle = \Pi^F|\psi\rangle$. If we apply the exact measurement channel $\mathcal{M}_c^F$ on $|\psi\rangle$, then*

(i) $|\psi_S\rangle = \Pi^{F\cup\{c\}}|\psi\rangle$*, and the outcome "S" has probability* $\mathrm{Tr}(\Pi^{F\cup\{c\}}|\psi\rangle\langle\psi|)$*.*

(ii) $|\psi_V\rangle = \Pi_c^{loc}\otimes\tilde{\Pi}^{F\setminus\Gamma^+(c)}|\psi_V\rangle$
(where $\Pi^{F\setminus\Gamma^+(c)} = \mathrm{Id}_{b(c)}\otimes\tilde{\Pi}^{F\setminus\Gamma^+(c)}$*).*

(iii) $\mathcal{M}_{c,V}^F(\Pi^F) \preceq \Pi_c^{loc}\otimes\tilde{\Pi}^{F\setminus\Gamma^+(c)}$.

Proof:
Property (**i**) is trivial by Definition 7.2.9.

Assume the notation of Proposition 7.2.11. By Definition 7.2.9 we have that $|\psi_V\rangle = WU^\dagger\big(\mathrm{Id} - \Pi^{F\cup\{c\}}\big)|\psi\rangle$. Due to $|\psi\rangle = \Pi^F|\psi\rangle$ we get

$$\big(\mathrm{Id} - \Pi^{F\cup\{c\}}\big)|\psi\rangle = \big(\Pi^F - \Pi^{F\cup\{c\}}\big)|\psi\rangle. \tag{7.10}$$

Using (7.8) we can see $WU^\dagger\big(\Pi^F - \Pi^{F\cup\{c\}}\big) = WU^\dagger U\mathrm{sign}(\Sigma)U^\dagger = W\mathrm{sign}(\Sigma)U^\dagger$. Considering $(\mathrm{sign}(\Sigma))^2 = \mathrm{sign}(\Sigma)$ and $U^\dagger U = \mathrm{Id}$ we get

$$W\mathrm{sign}(\Sigma)U^\dagger = W\mathrm{sign}(\Sigma)W^\dagger WU^\dagger U\mathrm{sign}(\Sigma)U^\dagger$$

and by (7.7)-(7.8) we get

$$W\mathrm{sign}(\Sigma)W^\dagger WU^\dagger U\mathrm{sign}(\Sigma)U^\dagger = \Pi_{\mathrm{im}(\Pi_c\Pi^F)}WU^\dagger\big(\Pi^F - \Pi^{F\cup\{c\}}\big).$$

Therefore, we proved $WU^\dagger\big(\Pi^F - \Pi^{F\cup\{c\}}\big) = \Pi_{\mathrm{im}(\Pi_c\Pi^F)}WU^\dagger\big(\Pi^F - \Pi^{F\cup\{c\}}\big)$. Also by (7.9) we have $\Pi_{\mathrm{im}(\Pi_c\Pi^F)} \preceq \Pi_c\Pi^{F\setminus\Gamma^+(c)} = \Pi_c^{loc}\otimes\tilde{\Pi}^{F\setminus\Gamma^+(c)}$ which implies that

$$WU^\dagger\big(\Pi^F - \Pi^{F\cup\{c\}}\big) = \Big(\Pi_c^{loc}\otimes\tilde{\Pi}^{F\setminus\Gamma^+(c)}\Big)WU^\dagger\big(\Pi^F - \Pi^{F\cup\{c\}}\big)$$

proving $|\psi_V\rangle = \Pi_c^{loc}\otimes\tilde{\Pi}^{F\setminus\Gamma^+(c)}|\psi_V\rangle$ via (7.10).

For the proof of property (**iii**) note that $\mathcal{M}^F_{c,V}(I) \preceq I$, which implies that $\mathcal{M}^F_{c,V}(\Pi^F) \preceq \mathcal{M}^F_{c,V}(I) \preceq I$. This together with property (**ii**) finally implies that $\mathcal{M}^F_{c,V}(\Pi^F) \preceq \Pi_c^{loc}\otimes\tilde{\Pi}^{F\setminus\Gamma^+(c)}$. □

For completeness we show that Definition 7.2.9 is indeed a generalization of the commuting case.

7.2.13. PROPOSITION. *Suppose all local projectors commute in Definition 7.2.9, and the input state $|\psi\rangle$ is such that $|\psi\rangle = \Pi^F|\psi\rangle$, then the output of the exact quantum channel $\mathcal{M}^F_c$ coincides with the output of the projective measurement $(\mathrm{Id} - \Pi_c, \Pi_c)$, i.e., $|\psi_S\rangle = (\mathrm{Id} - \Pi_c)|\psi\rangle$ and $|\psi_V\rangle = \Pi_c|\psi\rangle$.*

Proof:
Since all local projectors commute we have $\Pi^F = \prod_{c'\in F}(\mathrm{Id} - \Pi_{c'})$. By Definition 7.2.9 $|\psi_S\rangle = \Pi^{F\cup\{c\}}|\psi\rangle$ and due to commutation we have $\Pi^{F\cup\{c\}} = (\mathrm{Id} - \Pi_c)\Pi^F$, so $|\psi_S\rangle = (\mathrm{Id} - \Pi_c)\Pi^F|\psi\rangle = (\mathrm{Id} - \Pi_c)|\psi\rangle$.

By Definition 7.2.9 $|\psi_V\rangle = WU^\dagger\big(\mathrm{Id} - \Pi^{F\cup\{c\}}\big)|\psi\rangle$, furthermore similarly to the proof of Lemma 7.2.12 $\big(\mathrm{Id} - \Pi^{F\cup\{c\}}\big)|\psi\rangle = \big(\mathrm{Id} - \Pi^{F\cup\{c\}}\big)\Pi^F|\psi\rangle = \big(\Pi^F - \Pi^{F\cup\{c\}}\big)|\psi\rangle$ by our assumption on $|\psi\rangle$. Using (7.8) we get that $|\psi_V\rangle = WU^\dagger U\mathrm{sign}(\Sigma)U^\dagger|\psi\rangle = W\mathrm{sign}(\Sigma)U^\dagger|\psi\rangle$. By commutation we have that $\Pi_c\Pi^F = \Pi^F\Pi_c$ is an orthogonal projector and thus $\Sigma = \mathrm{sign}(\Sigma)$. Therefore, $W\mathrm{sign}(\Sigma)U^\dagger = W\Sigma U^\dagger = \Pi_c\Pi^F$ and thus $|\psi_V\rangle = \Pi_c\Pi^F|\psi\rangle = \Pi_c|\psi\rangle$. □

An efficient implementation of the exact measurement channel, by Lemma 7.2.12, Lemma 7.2.8 and Theorem 7.2.4, gives an efficient algorithm for preparing frustration-free states, even in the non-commuting case.

7.3 Implementation of the algorithm

In this section we describe and analyze our final algorithm. We will assume access to a (controlled) unitary V which provides access to the projectors $\Pi_i\colon i \in [m]$ that correspond to the constraints C; for notational simplicity we set $C = [m]$. Similarly to the fast quantum OR lemma (Lemma 3.2.16) we assume access to a unitary V such that $(\langle i| \otimes I)V(|i\rangle \otimes I) = \mathrm{C}_{\Pi_i}\mathrm{NOT}$ for all $i \in [m]$. In this section we define a query as a (controlled) application of V or $V^\dagger$, and also assume that the system is frustration-free. Therefore the gap of the sum of some constraints projectors is equal to the smallest non-zero eigenvalue.

First we show how to implement the measurement Π^S for some $S \subseteq [m]$.

7.3.1. Lemma. *Let $S \subseteq [m]$, and suppose that $\Delta\big(\sum_{i\in S}\Pi_i\big) \geq \delta$. For every $\varepsilon \in (0, 1/2)$ we can ε-precisely implement a $C_{\Pi^S}NOT$ gate with query complexity $\mathcal{O}\big(\sqrt{\frac{m}{\delta}}\log\big(\frac{1}{\varepsilon}\big)\big)$ and gate complexity at most $\mathcal{O}(m)$ times higher.*

Proof:
The main idea of the proof is very similar to our proof of the fast quantum OR lemma (Lemma 3.2.16). Let us define $A := \frac{1}{|S|}\sum_{i\in S}(I - \Pi_i)$, and observe that

$$I - \Pi_i = (\langle 0| \otimes I)\mathrm{C}_{\Pi_i}\mathrm{NOT}(|0\rangle \otimes I).$$

Let $q := \lceil \log_2(|S| + 1)\rceil + 1$ and let U be a unitary that implements the map $|0\rangle^{q-1} \mapsto \frac{1}{\sqrt{|S|}}\sum_{i\in S}|i\rangle$, and let us define $\tilde{V} := \big(U^\dagger \otimes I\big)V(U \otimes I)$ and $\Pi := |0\rangle\langle 0|^q \otimes I$. Then it is easy to see that $A = \Pi\tilde{V}\Pi$.

Finally, observe that singular value discrimination (Theorem 3.2.9) with $a := 1 - \frac{\delta}{|S|}$ and $b := 1$ implements the required operation. The complexity statement follows from Theorem 3.2.9, and the fact that U can be implemented using $\mathcal{O}\Big(m \cdot \sqrt{m/|S|}\Big)$ one- and two-qubit gates. □

Now we show how to implement the second step of an exact measurement channel. Let $W\Sigma U^\dagger$ be a singular value decomposition of $\Pi_c\Pi^F$; the goal is to implement the map $= WU^\dagger$. However, due to the structure of the algorithm we need to do this only on the image[10] of $(\Pi^F - \Pi^S)$. So we will implement the map $WU^\dagger(\Pi^F - \Pi^S) \overset{(7.8)}{=} W\mathrm{sign}(\Sigma)U^\dagger$.

7.3.2. Lemma. *Let $c \in [m], F \subseteq [m]$, $S := F\cup\{c\}$ and $\varepsilon \in (0, 1/2)$. Suppose that $\Delta\big(\sum_{i\in S}\Pi_i\big) \geq \delta$ and $W\Sigma U^\dagger$ is a singular value decomposition of $\Pi_c\Pi^F$. Then we can ε-precisely implement the map $W\mathrm{sign}(\Sigma)U^\dagger$ with $\mathcal{O}\Big(\sqrt{\frac{1}{\delta}}\log\big(\frac{1}{\varepsilon}\big)\Big)$ $C_{\Pi^F}NOT$, $C_{\Pi_c}NOT$ and other two-qubit gates.*

[10]Alternatively we could measure $\{\Pi^F, I - \Pi^F\}$, and only implement the map in case of outcome Π^F, and do nothing in the other case.

Proof:
First we lower bound the non-zero singular values. Let ς be the smallest non-zero singular value, we show that $\varsigma^2 \geq \min\{\delta, 1\}$. First observe that

$$\min\{\delta, 1\} \cdot I \preceq \Pi^S + \sum_{i\in S} \Pi_i. \tag{7.11}$$

Then we conclude by

$$\begin{aligned}
U \cdot \min\{\delta, 1\} \cdot \mathrm{sign}(\Sigma) U^\dagger =& f(\Pi^F - \Pi^S) \cdot \min\{\delta, 1\} \cdot I(\Pi^F - \Pi^S) && \text{(by (7.8))}\\
\preceq& (\Pi^F - \Pi^S)\left(\Pi^S + \sum_{i\in S}\Pi_i\right)(\Pi^F - \Pi^S) && \text{(by (7.11))}\\
=& (\Pi^F - \Pi^S)\left(\sum_{i\in S}\Pi_i\right)(\Pi^F - \Pi^S)\\
=& \Pi^F\left(\sum_{i\in S}\Pi_i\right)\Pi^F\\
=& \Pi^F \Pi_c \Pi^F\\
=& U\Sigma^2 U^\dagger.
\end{aligned}$$

Finally we apply singular vector transformation (Theorem 3.2.3) on $\Pi_c\Pi^F = W\Sigma U^\dagger$, which gives the claimed complexity. □

Let γ be the uniform gap of the constraint system, then we get the following corollary.

7.3.3. Corollary. *For every $\varepsilon \in (0, 1/2)$ we can ε-precisely implement an exact measurement channel with query complexity $\mathcal{O}\left(\frac{\sqrt{|C|}}{\gamma}\log\left(\frac{1}{\varepsilon}\right)\right)$ and with $\mathcal{O}(|C|)$ times higher gate complexity.*

7.3.1 The final algorithm and its complexity analysis

To obtain a working quantum algorithm we just need to run Algorithm 7.1 performing the **check**ing step using the approximate exact measurement channel of Corollary 7.3.3. We need to set the precision parameter small enough, so that the overall error during the algorithm remains bounded. Since the implementation of Corollary 7.3.3 depends logarithmically on the precision, this will give only logarithmic overhead.

As Theorem 7.2.4 shows under the Lovász condition ($pde \leq 1$), the expected number of resamplings is $\mathcal{O}(|C|)$, and therefore as the proof of Lemma 7.2.3 shows, the expected number of **check** operations performed by Algorithm 7.1 is $\mathcal{O}(|C|d)$.

Thus we use the progressive measurement channel at most $\mathcal{O}(|C|d)$ times in expectation. Using Corollary 7.3.3 and some standard boosting techniques we get a final algorithm that (in the non-commuting case) uses a total number of

$$\tilde{\mathcal{O}}\left(\frac{|C|^{1.5}\cdot d}{\gamma}\cdot\log^2\left(\frac{1}{\delta}\right)\right),$$

queries and $\mathcal{O}(|C|)$ times more two-qubit gates, where γ is the uniform gap, and δ is an upper bound on the trace distance of the output state from a density operator which is supported on the ground space.

In the paper [GS16] on which this chapter is based, we also analyze our algorithm's runtime under Shearer's condition. The exact formula for the runtime bound we prove is more complicated, but it is easy to compare to classical results. Let R_c be the upper bound of [KSz11] on the expected number of resamplings of the classical Moser-Tardos algorithm. The number of queries performed by our quantum algorithm is

$$\tilde{\mathcal{O}}\left(\frac{R_c|C|^{1.5}n}{\gamma}\log^2\left(\frac{1}{\delta}\right)\right),$$

where n is the number of qubits and the other parameters are as before.

7.4 Dependency structures: trees and cycles

The classical LLL gives a sufficient condition under which the union of some "sparsely dependent" events is guaranteed to have probability strictly less than 1. Shearer [She85] found the optimal bound on the probabilities of the bad events regarding arbitrary dependency graph, i.e., the weakest possible condition under which the general LLL holds. The bound is tight if we only force dependency structure on the events.

However, the LLL is most often applied to problems where there is additional restriction on the structure of the events, namely they are built on top of some independent random variables; this is called the variable setting. In the variable setting the Shearer bound may no longer be tight, as shown by Kolipaka and Szegedy [KSz11]. We generalize their result and formulate the optimal bound in the variable setting for tree and cyclic graphs. These results were also independently discovered by He et al. [HLL$^+$17].

State of the art (quantum) algorithms work up to the Shearer bound, but there are only a few examples of algorithms potentially working beyond that, see for example Harris [Har16]. Understanding the bounds under which the existence of a solution is guaranteed in the variable setting could help to find efficient algorithms that better exploit the variable structure.

7.4.1 Shearer bound

Now we describe a general version of the Lovász Local Lemma which gives optimal bounds in a generic (non-variable) setting.

7.4.1. Definition. We say that $G = (V, E)$ is a *dependency graph* for events $\{A_v : v \in V\}$ if for all $v \in V$, A_v is (totally) independent from the collection of events $\{A_w : w \text{ is not adjacent to } v\}$.

7.4.2. Definition. Let $G = (V, E)$, $\vec{x} = (x_v : v \in V)$ and let $\mathrm{Indep}(G)$ denote the independent (vertex) sets of G. Then we call

$$I(G, \vec{x}) = \sum_{S \subseteq \mathrm{Indep}(G)} (-1)^{|S|} \prod_{v \in S} x_v$$

the *multivariate independence polynomial* of G.

7.4.3. Definition. We say that the vector of probabilities $(p_v : v \in V) = \vec{p} \in \mathbb{R}^{|V|}$ is *above the Shearer bound* for a graph G if there is a vertex set $V' \subseteq V$ such that for the corresponding induced subgraph $G' = (V', E')$: $I(G', (p_v : v \in V')) \leq 0$. Otherwise we say $\vec{p}$ is *below the Shearer bound.*

The following is a simple reformulation of Shearer's result [She85]:

7.4.4. Theorem. *For a graph $G = (V, E)$ and probabilities $\vec{p} \in \mathbb{R}^{|V|}$ the following are equivalent:*

(i) $\vec{p}$ is below the Shearer bound for G.

(ii) there exists a probability space Ω and a set of events $\{A_v \subseteq \Omega : v \in V\}$ with G as dependency graph, satisfying the extremal conditions $\forall v \in V : \Pr(A_v) = p_v$, $\forall \{v, w\} \in E : A_v \cap A_w = \emptyset$ and $\Pr(\overline{\bigcup_{v \in V} A_v}) > 0$.

(iii) for any probability space Ω' and events $\{A'_v \subseteq \Omega' : v \in V\}$ having G as dependency graph and satisfying $\Pr(A_v) \leq p_v$, we have $\Pr(\overline{\bigcup_{v \in V} A'_v}) \geq I(G, \vec{p}) > 0$.

To gain some intuition on this Theorem, note that in (ii) the inclusion-exclusion formula applied to $\Pr(\overline{\bigcup_{v \in V} A_v})$ gives exactly $I(G, \vec{p}) > 0$, due to the extremal property of the events.

7.4.2 The variable version: tree and cycle graphs

The variable version of the Lovász Local Lemma is defined on probability spaces which are products of independent random variables. Each "bad" event can depend on a few of the variables, and two such events are considered dependent if they share a variable. If one thinks about events as (hyper)graphs on the variables, then the dependency graph is simply the *line graph* of the event (hyper)graph. The line graph of an undirected hypergraph $G = (V, E)$ is denoted by $L(G)$, has vertex set E, and a pair of distinct $a, b \in E$ forms an edge in $L(G)$ if and only if a and b share some vertex in G, formally speaking

$$G = (V, E) \Longrightarrow L(G) = (E, \{\{a, b\} \subseteq E \colon a \cap b \neq \emptyset \wedge a \neq b\})$$

In this subsection we determine the optimal bounds on the variable version of the Lovász Local Lemma for tree and cycle event (hyper)graphs, which can go beyond the Shearer bound.

We analyze what are the vectors of probabilities for which we can always avoid some "bad" events when they are coming from the variable setting. It turns out that the trivial lower bound we get by applying Shearer's theorem to the line graph of the event (hyper)graph (see Proposition 7.4.6), is always tight for tree graphs (see Theorem 7.4.7). However, our result (Theorem 7.4.17) states that Shearer's bound for the line graph $L(C_n) = C_n$ is not tight in the variable setting; the true bound equals the smallest of the Shearer bounds we can get after deleting an edge of $L(C_n)$. This is surprising because it tells that in the variable setting the events cannot really make use of the cyclic structure.

For ease of notation in the following we denote both a probability space (Ω_X, S_X, P_X) and its base set Ω_X by the same letter X. We also assume (without loss of generality) that the base sets are disjoint ($\Omega_X \cap \Omega_Y = \emptyset$) for distinct probability spaces $X \neq Y$. This enables us to interpret their products in a "commutative" way in the following sense: Suppose we have disjoint base sets named X, Y, Z. Then the product of events should be evaluated in some fixed (e.g. alphabetical) order of the corresponding base sets, e.g. let $A_{XZ} \subseteq X \times Z$ when we write $Y \times A_{XZ}$ it should be interpreted as a subset of $X \times Y \times Z$. Also as standard in probability theory, we will often not emphasize trivial extension of events and write just A_{XZ} instead of $Y \times A_{XZ}$ when this is clear from the context.

In the rest of this section all graphs will refer to event (hyper)graphs, unless otherwise stated.

7.4.5. Definition. We say that for the (hyper)graph $G = (V, E)$ the probabilities $\vec{p} = (p_e \colon e \in E) \in [0, 1]^{|E|}$ are *above the variable bound* if there exist probability spaces $W_v \colon v \in V$, and events $A_e \subseteq \bigtimes_{v \in e} W_v \colon e \in E$ such that $\Pr(A_e) \leq p_e$ and $\Pr(\bigcup_{e \in E} A_e) = 1$. Otherwise we say $\vec{p}$ is *below the variable bound* with respect to G.

7.4.6. PROPOSITION. *If the probabilities $\vec{p} = (p_e \colon e \in E) \in [0,1]^{|E|}$ are above the variable bound for the graph $G = (V, E)$ then $\vec{p}$ is above the Shearer bound with respect to the line graph $L(G)$.*

Proof:
Observe that $L(G)$ is always a dependency graph for events $A_e \colon e \in E$ coming from the variable setting, and so we can apply Theorem 7.4.4. □

The following theorem states that for tree graphs the above bound is tight.

7.4.7. THEOREM. *The probabilities $\vec{p} = (p_e \colon e \in E) \in [0,1]^{|E|}$ are above the variable bound for the tree graph $T = (V, E)$ if and only if $\vec{p}$ is above the Shearer bound with respect to the line graph $L(T)$.*

Proof:
Due to Proposition 7.4.6 it is enough to show that if $\vec{p}$ is above the Shearer bound with respect to $L(T)$, then $\vec{p}$ is above the variable bound. First we assume without loss of generality that $V = [n]$. We choose $W_i = [0,1)_i := \{\{i\} \times [0,1)\}$ a disjoint copy of $[0,1)$ with the Lebesgue measure. Let us define an ordering of the edges in the following way: we choose 1 as root and do a breadth-first search on T. Then let e_i be the i-th edge traversed. Also without loss of generality we assume $i \in V$ is the i-th vertex reached by this breadth-first search. We are going to construct the events A_e, one by one, in a recursive way in reversed order staring with e_{n-1}, such that after A_{e_k} is defined we are going maintain the following:

- The pairs of already defined events which share a vertex are pairwise disjoint.
- There is a set of numbers $b_i^k \in [0,1]$ such that the union of the already defined events $\bigcup_{\ell=k}^{n-1} A_{e_\ell} = \bigcup_{i=1}^{n} [b_i^k, 1)_i$.
- For $\ell \geq k$ and $e_\ell = \{i, j\} \colon i < j$, we have $A_{e_\ell} \subseteq [b_i^k, 1)_i$.

Note that for $k = n$ these are trivially satisfied with the choice $b_i^n = 1$, so we can start our recursion. Suppose we already defined $A_{e_{n-1}}, \ldots, A_{e_{k+1}}$ and we are about to define A_{e_k} with $e_k = \{i, j\} \colon i < j$. Then we set $b_i^k = \max\bigl(0, b_i^{k+1} - p_{i,j}/b_j^{k+1}\bigr)$ and for $\ell \neq i$ we set $b_\ell^k = b_\ell^{k+1}$. We define $A_{e_k} := [b_i^k, b_i^{k+1})_i \times [0, b_j^{k+1})_j$. Then clearly $\Pr(A_{e_k}) \leq p_{e_k}$ and $A_{e_k} \subseteq [b_i^k, 1)_i$. Also $A_{e_k} \subseteq [0, b_i^{k+1})_i$ and $A_{e_k} \subseteq [0, b_j^{k+1})_j$ so it is disjoint from all already defined events since if $\ell > k$ and $e_\ell = \{i, m\}$ then $i < m$, and if $e_\ell = \{j, m\}$ then $j < m$ due to the breadth-first structure.

Finally

$$\begin{aligned}\bigcup_{\ell=k}^{n-1} A_{e_\ell} &= \bigcup_{\ell=k+1}^{n-1} A_{e_\ell} \cup A_{e_k}\\ &= \bigcup_{\ell=1}^{n}[b_\ell^{k+1},1)_\ell \cup [b_i^k, b_i^{k+1})_i \times [0, b_j^{k+1})_j\\ &= \bigcup_{\ell\notin\{i,j\}} [b_\ell^{k+1},1)_\ell \cup [b_i^{k+1},1) \cup [b_j^{k+1},1) \cup [b_i^k, b_i^{k+1})_i \times [0, b_j^{k+1})_j\\ &= \bigcup_{\ell\notin\{i,j\}} [b_\ell^{k+1},1)_\ell \cup [b_i^k,1)_i \cup [b_j^k,1)_j = \bigcup_{\ell=1}^{n}[b_\ell^k,1)_\ell.\end{aligned}$$

After the recursion ends we get $\overline{A} := \overline{\bigcup_{\ell=1}^{n-1} A_{e_\ell}} = \overline{\bigcup_{i=1}^{n}[b_i^1,1)_i} = \bigtimes_{i=1}^{n}[0,b_i^1)_i$.

- If for some $i \in [n]\colon\ b_i^1 = 0$, then $\overline{A} = \emptyset$ so $\vec{p}$ is above the variable bound.
- Otherwise we have $\Pr(\overline{A}) = \prod_{i=i}^{n} b_i^1 > 0$, $\Pr(A_e) = p_e$ and all events that share a variable are disjoint, so we are below the Shearer bound by Theorem 7.4.4.

□

Now we start proving our result about cycle graphs; for this we introduce a few definitions and lemmas.

7.4.8. Definition. Let X, V be probability spaces. Then for every event $A \subseteq X \times V$ and $x \in X$ we can define the *cross section event* $A^x \subseteq V$ satisfying the identity $\{x\} \times A^x = A \cap \{x\} \times V$. Note that this is indeed an event in V, i.e., a measurable set, see e.g. [Rie70, Fol99].

The following lemma slightly generalizes the key discretization idea introduced in [KSz11], showing that it is sufficient to work with probability spaces on finite sets.

7.4.9. Lemma. *Let $X, W_1, W_2, \ldots, W_n$ be probability spaces, and $A_i \subseteq X \times W_i$, $A_W \subseteq \bigtimes_{i=1}^{n} W_i$ events, with the property* $\Pr(\bigcup_{i=1}^{n} A_i \cup A_W) = 1$. *Then there exists $m \in [n]$ and $x^k \in X$, $\mu_k \in [0,1]$ for all $k \in [m]$ such that $\sum_{k=1}^{m} \mu_k = 1$ and $\forall i \in [n]\colon \sum_{k=1}^{m} \mu_k \Pr(A_i^{x^k}) \leq \Pr(A_i)$ while $\forall k \in [m]\colon \Pr(\bigcup_{k=1}^{m} A_i^{x^k} \cup A_W) = 1$.*

Proof:
Let $X_{\text{Bad}} = \{x \in X \colon \Pr(\bigcup_{i=1}^{n} A_i^x \cup A_W) < 1\}$. Since $\Pr(\bigcup_{i=1}^{n} A_i \cup A_W) = 1$ we must have $\Pr(X_{\text{Bad}}) = 0$ and we can assume without loss of generality that

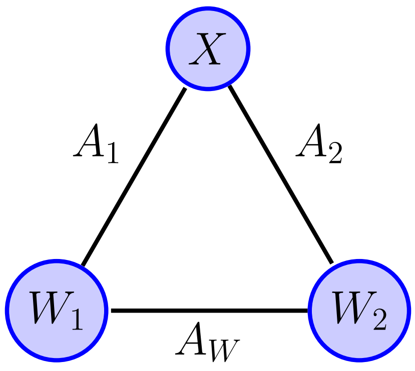


Figure 7.1. The graph of events from Lemma 7.4.9 for the case when $n = 2$.

$X_{\text{Bad}} = \emptyset$ otherwise we can just remove a zero-probability set from X. Thus we ensured that for any $x \in X$ $\Pr(\bigcup_{k=1}^{m} A_i^x \cup A_W) = 1$.

Let $f_i(x) = \Pr(A_i^x)$, which is a measurable function with $\mathbb{E}_X[f_i(x)] = \Pr(A_i)$ as implied by the Fubini-Tonelli Theorem [Rie70, Fol99]. Then $f = (f_1, f_2, \ldots, f_n)$ is a random variable over $\mathbb{R}^n$. Also $\vec{p}\colon = (p_i\colon i \in [n]) = \mathbb{E}_X[f] \in \operatorname{conv}(\{f(x)\colon x \in X\})$ (see e.g. [GT12]), and due to Carathéodory's Theorem [Roc70] $\vec{p}$ is a convex combination of some $n+1$ points $f(x_0^i)\colon i \in [n+1]$. If $\vec{p}$ lies in an affine subspace spanned by at most n points of $\{f(x_0^i)\}$ then we can find a convex combination of at most n points by Carathéodory's Theorem. Otherwise there is an n-dimensional simplex $S_n := \operatorname{conv}(\{f(x_0^i)\colon i \in [n+1]\})$ such that $\vec{p}$ lies in its interior. In this case we choose the largest $t \in \mathbb{R}_+$ such that $\vec{r} := \vec{p} - t \cdot \mathbb{1} \in S_n$. Then $\vec{r}$ lies on a facet of S_n of dimension at most $n-1$ and again by Carathéodory's Theorem there are $x^k, \mu_k\colon k \in [n]$ such that they form a convex combination $\vec{r} = \sum_{k=1}^{n} \mu_k f(x^k) = \vec{p} - t \cdot \mathbb{1} \le \vec{p}$. □

Now we refine the above statement for the most relevant case for cycles: $n = 2$.

7.4.10. Lemma. *If in addition to the conditions of Lemma 7.4.9 we also have $n = 2$ then we can find $\forall i \in \{1,2\}\colon$ $A_1^i \subseteq W_1$, $A_2^i \subseteq W_2$, $\nu_i \in [0,1]$ such that $A_1^1 \subseteq A_1^2$, $A_2^1 \supseteq A_2^2$, $\nu_1 + \nu_2 = 1$, $\forall j \in \{1,2\}\colon \sum_{i=1}^{2} \nu_i \Pr(A_j^i) \le \Pr(A_j)$ and $\forall i \in \{1,2\}\colon \Pr(A_1^i \cup A_2^i \cup A_W) = 1$.*

Proof:
Lemma 7.4.9 guarantees the existence of $x^1, x^2 \in X$ such that $\overline{A_1^{x^1}} \times \overline{A_2^{x^1}} \subseteq A_W$ and $\overline{A_1^{x^2}} \times \overline{A_2^{x^2}} \subseteq A_W$ almost surely. Now assume without loss of generality that $\mu_1 \le \mu_2$.
Let us define $A_1^a = A_1^{x^1} \cap A_1^{x^2}$, $A_2^a = A_2^{x^1} \cup A_2^{x^2}$, $A_1^b = A_1^{x^2}$, $A_2^b = A_2^{x^2}$, $A_1^c = A_1^{x^1} \cup A_1^{x^2}$,

$A_2^c = A_2^{x^1} \cap A_2^{x^2}$. Observe that

$$\begin{aligned}\overline{A_1^a} \times \overline{A_2^a} &= \overline{A_1^{x^1} \cap A_1^{x^2}} \times \overline{A_2^{x^1} \cup A_2^{x^2}}\\ &= \left(\overline{A_1^{x^1}} \cup \overline{A_1^{x^2}}\right) \times \left(\overline{A_2^{x^1}} \cap \overline{A_2^{x^2}}\right)\\ &= \underbrace{\left(\overline{A_1^{x^1}} \times \left(\overline{A_2^{x^1}} \cap \overline{A_2^{x^2}}\right)\right)}_{\subseteq \overline{A_1^{x^1}} \times \overline{A_2^{x^1}}} \cup \underbrace{\left(\overline{A_1^{x^2}} \times \left(\overline{A_2^{x^1}} \cap \overline{A_2^{x^2}}\right)\right)}_{\subseteq \overline{A_1^{x^2}} \times \overline{A_2^{x^2}}} \quad \subseteq \quad A_W\end{aligned}$$

almost surely. Due to symmetry we have $\overline{A_1^c} \times \overline{A_2^c} \subseteq A_W$ almost surely as well, whereas $\overline{A_1^b} \times \overline{A_2^b} \subseteq A_W$ trivially holds (almost surely). Also

$$\begin{aligned}&\mu_1 \Pr(A_1^a) + (\mu_2 - \mu_1)\Pr(A_1^b) + \mu_1 \Pr(A_1^c) =\\ &\qquad = \mu_1 \Pr(A_1^{x^1} \cap A_1^{x^2}) + (\mu_2 - \mu_1)\Pr(A_1^{x^2}) + \mu_1 \Pr(A_1^{x^1} \cup A_1^{x^2})\\ &\qquad = \mu_1 \Pr(A_1^{x^1}) + \mu_1 \Pr(A_1^{x^2}) + (\mu_2 - \mu_1)\Pr(A_1^{x^2})\\ &\qquad = \mu_1 \Pr(A_1^{x^1}) + \mu_2 \Pr(A_1^{x^2}) \le \Pr(A_1).\end{aligned}$$

Due to symmetry we also have $\mu_1 \Pr(A_2^a) + (\mu_2 - \mu_1)\Pr(A_2^b) + \mu_1 \Pr(A_2^c) \le \Pr(A_2)$. A convexity argument such as in the proof of Lemma 7.4.9 shows that $\exists k, l \in \{a, b, c\}, \nu_1 \in [0, 1]$ such that $\forall i \in \{1, 2\}\colon\ \nu_1 \Pr(A_i^k) + (1 - \nu_1)\Pr(A_i^l) \le \Pr(A_i)$. Without loss of generality assume $A_1^k \subseteq A_1^l$, then since $A_1^a \subseteq A_1^b \subseteq A_1^c$ and $A_2^a \supseteq A_2^b \supseteq A_2^c$ and $k, l \in \{a, b, c\}$ we must have $A_2^k \subseteq A_2^l$. Now setting $A_i^1 = A_i^k$, $A_i^2 = A_i^l$ for all $i \in \{1, 2\}$ and $\nu_2 = 1 - \nu_1$ satisfies all required conditions. □

We introduce a monotonicity concept that plays a key role in our proof, helps to avoid large case separations, and provides an alternative approach to [HLL+17].

7.4.11. DEFINITION. We say that an event $A \subseteq X \times Y$ is X-monotone if for each $x, x' \in X$ at least one of $A^x \subseteq A^{x'}$, $A^x \supseteq A^{x'}$ holds.

7.4.12. PROPOSITION. *Let $X' = \{1, 2, 3\}$. If $A \subseteq X \times Y \times W$ is Y-monotone then $\forall x^1, x^2 \in X$ the event*

$$A' := \left(\{1\} \times \left(A^{x^1} \cap A^{x^2}\right)\right) \cup \left(\{2\} \times \left(A^{x^2}\right)\right) \cup \left(\{3\} \times \left(A^{x^1} \cup A^{x^2}\right)\right) \subseteq X' \times Y \times W$$

is X'-monotone and also Y-monotone.

Proof:
X'-monotonicity is trivial since $\left(A^{x^1} \cap A^{x^2}\right) \subseteq A^{x^2} \subseteq \left(A^{x^1} \cup A^{x^2}\right)$. Let $y^1, y^2 \in Y$, then without loss of generality $A^{y^1} \subseteq A^{y^2}$. Then

$$\begin{aligned}A'^{y^i} &:= \left(\{1\} \times \left(A^{x^1} \cap A^{x^2}\right)^{y^i}\right) \cup \left(\{2\} \times \left(A^{x^2}\right)^{y^i}\right) \cup \left(\{3\} \times \left(A^{x^1} \cup A^{x^2}\right)^{y^i}\right)\\ &= \left(\{1\} \times \left(A^{(y^i, x^1)} \cap A^{(y^i, x^2)}\right)\right) \cup \left(\{2\} \times \left(A^{(y^i, x^2)}\right)\right) \cup \left(\{3\} \times \left(A^{(y^i, x^1)} \cup A^{(y^i, x^2)}\right)\right)\end{aligned}$$

so clearly $A'^{y^1} \subseteq A'^{y^2}$. □

7.4.13. Corollary. *If $\vec{p}$ is above the variable bound for $G = (V, E)$, then it is above the variable bound even if W_v is restricted to $|W_v| \leq \deg(v)$ for each $v \in V$ and we require for each $v \in V$ with $\deg(v) = 2$ that $\forall e \in E \colon v \in e$ the event A_e is W_v-monotone.*

Proof:
Start with any probability spaces $W_v \colon v \in V$, and events $A_e \subseteq \bigtimes_{v \in e} W_v \colon e \in E$ showing $\vec{p}$ is above the variable bound. Then reduce each probability space W_v using Lemma 7.4.9 above by choosing the new discrete probability space $W_v = \{x^1, \ldots, x^m\}$ (where $m \leq \deg(v)$) with the elements having probabilities $\mu_i \colon i \in m$, and choose modified events $A_e := \bigcup_{i=1}^m \{x^i\} \times A_e^{x^i}$. Except in case $\deg(v) = 2$ use the events coming from the Lemma 7.4.10. Note that due to Proposition 7.4.12 already established monotonicity is preserved during the iterative application of Lemma 7.4.10 while reduction via Lemma 7.4.9, trivially preserves monotonicity. □

Note that the construction of Theorem 7.4.7 effectively also achieves this, i.e. divides each node into d parts, and the root maybe into $d+1$ parts, but only if below the Shearer bound. Also we note that Lemma 7.4.9, 7.4.10 and Corollary 7.4.13 hold for hypergraphs as well, the same proofs work.

Now we prove a simple corollary showing that the set of points above the variable bound is closed.

7.4.14. Corollary. *For a (hyper)graph $G = (V, E)$ let us define the set $VB_G = \{\vec{p} = (p_e \colon e \in E) \in [0,1]^{|E|} \colon \vec{p}$ is above the variable bound corresponding to $G\}$. The set VB_G is closed.*

Proof:
First observe that if $\vec{p}' \in VB_G$, and $\vec{p}' \leq \vec{p}''$, then by definition $\vec{p}'' \in VB_G$ so it is a monotone property. Note that there are finitely many configurations of possible events of the form $C^\ell = (A_e \subseteq \bigtimes_{v \in e} \{(v, i) \colon i \in [\deg(v)]\} \colon e \in E)$. Also for any such configuration C^ℓ the set of probability vectors allowed by this configuration $VB_G^{C^\ell} = \{\vec{p} = (\Pr(A_e) \colon e \in E) \colon \Pr(\{(v,i)\}) = p_{(v,i)}, \sum_{i=1}^{\deg(v)} p_{(v,i)} = 1, (p_{(v,i)} \colon (v,i) \in \bigcup_{v \in V} \{v\} \times [\deg(v)]) \in [0,1]^{\sum_{v \in V} \deg(v)}\}$ is the image of a closed set by a continuous function, so clearly closed. Due to Corollary 7.4.13, $VB_G = \{\vec{p} \colon \exists C^\ell$ configuration and $\vec{p}' \in C^\ell$ such that $\vec{p}' \leq \vec{p}\}$ which is then a finite union of closed sets, so closed itself. □

We are almost ready to prove our main result about cycles, but first we need to understand paths a bit better.

7.4.15. Definition. Let $P_{k\dots\ell}$ denote the path graph on vertex set $V=\{k,k+1,\dots,\ell\}$ with edges $E=\{\{i,i+1\}\colon i,i+1\in V\}$. For $k>\ell$ let us interpret $P_{k\dots\ell}$ as the empty graph. Also let us introduce the notation

$$I_{k\dots\ell}(x_v\colon v\in V)=\begin{cases}0 & \text{if } k\geq \ell+3\\ I(P_{k\dots\ell},(x_v\colon v\in V)) & \text{otherwise.}\end{cases}$$

When clear from context we may just write $I_{k\dots\ell}(x_v)$ or even simpler just $I_{k\dots\ell}$.

Later we are going to use the following identity considering $I_{k\dots\ell}$ for $k\leq \ell+1$:

$$I_{k\dots\ell}(x_i\colon i\in\{k,\dots,\ell\})=I_{k\dots\ell-1}-x_\ell\cdot I_{k\dots\ell-2} \tag{7.12}$$

To prove this, observe that for $k=\ell+1$ the above is just $1=\ell-x_\ell\cdot 0$, and for $k\leq\ell$ we have

$$\begin{aligned}I_{k\dots\ell}&=\sum_{S\subseteq \mathrm{Indep}(P_{k\dots\ell})}(-1)^{|S|}\prod_{v\in S}x_v\\ &=\sum_{S\subseteq \mathrm{Indep}(P_{k\dots\ell-1})}(-1)^{|S|}\prod_{v\in S}x_v+\sum_{\substack{S=S'\cup\{\ell\}\\ S'\subseteq \mathrm{Indep}(P_{k\dots\ell-2})}}(-1)^{|S|}\prod_{v\in S}x_v=I_{k\dots\ell-1}-x_\ell I_{k\dots\ell-2}.\end{aligned}$$

7.4.16. Proposition. *Let $n\geq 3$, $\vec{p}=(p_i\colon i\in[n-1])\in[0,1]^{n-1}$ and $x,y\in(0,1]$. Then there exist probability spaces $W_i\colon i\in[n]$, events $A_i\subseteq W_i\times W_{i+1}$ with $\Pr(A_i)=p_i$ and $X\subseteq W_1, Y\subseteq W_n$ events with $\Pr(X)=x,\Pr(Y)=y$ satisfying $\Pr(X\times Y\setminus\bigcup_{i=1}^{n-1}A_i)=0$ if and only if the probabilities*

$$\vec{p}'=(p_1/x,p_2,p_3,\dots,p_{n-2},p_{n-1}/y) \tag{7.13}$$

are above the Shearer bound corresponding to the dependency graph $P_{1\dots n-1}$.

Moreover, if $\vec{p}'$ is below the Shearer bound with respect to all induced proper subgraphs of $P_{1\dots n-1}$, then (7.13) is above the Shearer bound corresponding to the dependency graph $P_{1\dots n-1}$ if and only if

$$f_{\vec{p}}(x):=p_{n-1}\cdot\frac{x\cdot I_{2\dots n-3}(p_i)-p_1\cdot I_{3\dots n-3}(p_i)}{x\cdot I_{2\dots n-2}(p_i)-p_1\cdot I_{3\dots n-2}(p_i)}\geq y,$$

and the derivative $f'_{\vec{p}}(x)=\dfrac{-p_1\cdot p_2\cdots p_{n-1}}{[x\cdot I_{2\dots n-2}(p_i)-p_1\cdot I_{3\dots n-2}(p_i)]^2}$.

Proof:
$\Longrightarrow$: Suppose we have the probability spaces and events as required. Then

$$\begin{aligned}\Pr(A_1|X\times Y)&=\frac{\Pr(A_1\cap X\times Y)}{\Pr(X\times Y)}=\frac{\Pr(A_1\cap X\cap Y)}{\Pr(X\cap Y)}\\ &=\frac{\Pr(A_1\cap X)\Pr(Y)}{\Pr(X)\Pr(Y)}=\Pr(A_1|X)\leq p_1/x\end{aligned}$$

and similarly $\Pr(A_{n-1}|X\times Y)\le p_{n-1}/y$. Also $\Pr(A_i)=p_i$ for $i\in\{2,3,\dots,n-2\}$. Finally $\Pr(\bigcup_{i=1}^{n-1}A_i|X\times Y)=1$ so by Theorem 7.4.7 the (conditional) probabilities $(p_1/x,p_2,p_3,\dots,p_{n-2},p_{n-1}/y)$ are above the Shearer bound.
$\Longleftarrow$: Suppose the probabilities $(p'_1,p'_2,\dots,p'_{n-1})=(p_1/x,p_2,p_3,\dots,p_{n-2},p_{n-1}/y)$ are above the Shearer bound. Then there exist probability spaces W'_i and events A_i with the right structure and satisfying $P'(A_i)\le p'_i$. Then we define $X:=W'_1,Y:=W'_n,W_1:=\{x_0\}\dot\cup X,W_n:=\{y\}\dot\cup Y$ and $P_{W_1}(\{x_0\}):=1-x,\forall A\subseteq X:P_{W_1}(A|X):=x\cdot P'_{W'_1}(A)$ similarly. Also $P_{W_n}(\{y_0\}):=1-y$, $\forall A\subseteq Y:P_{W_n}(A|Y):=y\cdot P'_{W'_n}(A)$ finally $W_i:=W'_i$ for $i\in\{2,3,\dots,n-2\}$. Then clearly $\Pr(A_i)\le p_i$ and the other conditions are also satisfied.

If $\vec{p}'$ is below the Shearer bound with respect to any induced proper subgraphs of $P_{1\dots n-1}$, then being above the Shearer bound is equivalent to $I_{1\dots n-1}(p'_i\colon i\in[n-1])\le 0$. Due to (7.12)

$$\begin{aligned}
I_{1\dots n-1}(p'_i)\le 0 &\iff I_{1\dots n-2}(p'_i)-\frac{p_{n-1}}{y}\cdot I_{1\dots n-3}(p'_i)\le 0\\
&\iff y\le p_{n-1}\frac{I_{1\dots n-3}(p'_i)}{I_{1\dots n-2}(p'_i)}\\
&\iff y\le p_{n-1}\frac{I_{2\dots n-3}(p'_i)-p_1/x\cdot I_{3\dots n-3}(p'_i)}{I_{2\dots n-2}(p'_i)-p_1/x\cdot I_{3\dots n-2}(p'_i)}\\
&\iff y\le p_{n-1}\frac{x\cdot I_{2\dots n-3}(p_i)-p_1\cdot I_{3\dots n-3}(p_i)}{x\cdot I_{2\dots n-2}(p_i)-p_1\cdot I_{3\dots n-2}(p_i)} \qquad (7.14)
\end{aligned}$$

where in the first step we used (7.12) and in the third step we again used (7.12) combined with the symmetry $I_{k\dots\ell}(r_i)=I_{k\dots\ell}(s_i=r_{k+\ell-i})$.

For the derivative consider $\left(p_{n-1}\frac{x\cdot a-p_1\cdot b}{x\cdot c-p_1\cdot d}\right)'=p_{n-1}\frac{a\cdot(x\cdot c-p_1\cdot d)-c(x\cdot a-p_1\cdot b)}{(x\cdot c-p_1\cdot d)^2}=-p_1\cdot p_{n-1}\cdot\frac{ad-bc}{(x\cdot c-p_1\cdot d)^2}$. Applying this identity to (7.14) we see that the only thing we need to show is that $p_2\cdot p_3\cdots p_{n-2}=I_{2\dots n-3}\cdot I_{3\dots n-2}-I_{2\dots n-2}\cdot I_{3\dots n-3}$. This we prove by induction on n. For the base case $n=3$ observe that both the left and the right-hand side is 1.

Induction step: Suppose we showed this for $n=k$, then for $k+1$

$$\begin{aligned}
&I_{2\dots k-2}\cdot I_{3\dots k-1}-I_{2\dots k-1}\cdot I_{3\dots k-2}=\\
&=I_{2\dots k-2}\cdot[I_{3\dots k-2}-p_{k-1}\cdot I_{3\dots k-3}]-[I_{2\dots k-2}-p_{k-1}\cdot I_{2\dots k-3}]\cdot I_{3\dots k-2}\\
&=p_{k-1}\cdot[I_{2\dots k-3}I_{3\dots k-2}-I_{2\dots k-2}\cdot I_{3\dots k-3}]=\\
&=p_2\cdot p_3\cdots p_{k-2}\cdot p_{k-1}
\end{aligned}$$

where in the first equality we again used (7.12) whereas in the last equality we used our inductive hypothesis. □

The following is the main theorem in this section, giving the optimal value of the variable bound for cyclic dependency graphs and showing the separation from the bound we get from Proposition 7.4.6.

7.4.17. THEOREM. *Let us consider the cyclic graph $C_n = (V, E)$ where $n \geq 3$, $V = [n] - 1$, $E = \{\{i, i+1\} \colon i \in [n] - 1\}$ (the indices should be interpreted* mod *n). Then the probabilities $\vec{p} = (p_{i,i+1} \colon i \in [n] - 1) \in [0,1]^n$ are above the variable bound with respect to C_n if and only if $\exists k \in [n] - 1$ such that $\vec{p}' = (p'_i = p_{i+k,i+k+1} \colon i \in [n] - 1) \in [0,1]^n$ is above the Shearer bound with respect to the path graph $P_{0\ldots n-1}$.*

Proof:
$\Longleftarrow$: Suppose $\exists k \in [n] - 1$ such that $\vec{p}' = (p'_i = p_{i+k,i+k+1} \colon i \in [n] - 1) \in [0,1]^n$ is above the Shearer bound with respect to the path graph $P_{0\ldots n-1}$. Then due to Theorem 7.4.7 $\forall i \in [n+1] - 1$ there are probability spaces W'_i and $\forall i \in [n] - 1$ there are events $A'_{i,i+1} \subseteq W'_i \times W'_{i+1}$ such that $\Pr(A'_i) \leq p'_i$ and $\Pr(\bigcup_{i=0}^{n-1} A'_i) = 1$. Now let us interpret indices mod n, and let $W_k := W'_0 \times W'_n$ and let $W_\ell = W'_{\ell-k} \colon \ell \in ([n] - 1 \setminus \{k\})$, and let $A_{k,k+1} := A'_0 \times W'_n$, $A_{k-1,k} := W'_0 \times A'_{n-1}$ and $A_{\ell,\ell+1} := A'_{\ell-k}$ for $\ell \not\equiv k, \ell + 1 \not\equiv k$. Then clearly $A_{\ell,\ell+1} \subseteq W_\ell \times W_{\ell+1}$, $\Pr(A_{\ell,\ell+1}) = \Pr(A'_{\ell-k}) \leq p'_{\ell-k} = p_{\ell,\ell+1}$ and $\Pr(\bigcup_{i=0}^{n-1} A_i) = 1$ showing that $\vec{p}$ is above the variable bound with respect to C_n.

$\Longrightarrow$: It is enough to show the statement for the set of minimal elements $\min(VB_{C_n}) = \{\vec{p} \in VB_{C_n} \colon \nexists \vec{p}' \in VB_{C_n} \setminus \{\vec{p}\} \colon \vec{p}' \leq \vec{p}\}$ since VB_{C_n} is a closed set due to Corollary 7.4.14, and thus $\forall \vec{p} \in VB_{C_n} \exists \vec{p}' \in \min(VB_{C_n})$ such that $\vec{p}' \leq \vec{p}$. Our strategy is the following: we show that for $\vec{p} \in \min(VB_{C_n})$ there are probability spaces $W_v \colon v \in V$ and events $A_e \colon e \in E$ with $\Pr(A_e) \leq p_e$ and $\Pr(\bigcup_{e \in E}) = 1$ such that $\exists e \in E$ and $v \in e$ such that A_e is independent of W_v (e.g. $A_e = W_v \times A'_e$). This is enough because then A_e is independent from $A_{e'}$, where e' is the other edge adjacent to v: $e \cap e' = \{v\}$. So the path graph we get after deleting the edge $\{v-1, v\}$ from $L(C_n \setminus \{v\})$ is still a dependency graph, thus due to Theorem 7.4.4 the conclusion holds with the choice $k = v$.

Suppose $\vec{p} \in \min(VB_{C_n})$, then due to Corollary 7.4.13 we can assume without loss of generality that $|W_v| \leq 2$ and $A_{i,i+1} \subseteq W_i \times W_{i+1}$ is monotone, while $\Pr(A_e) = p_e$ due to minimality and $\Pr(\bigcup_{e \in E} A_e) = 1$ as usual. Now we proceed using case analysis:

(i) If $\exists v \in V$ such that $|W_v| < 1$ or $\exists w \in W_v \colon \Pr(w) = 0$, then no event can be dependent on W_v so we are done.

(ii) If $\exists e \in E$ such that $|A_e| = 0$ or $|A_e| = 4$, then clearly $\Pr(A_e) \in \{0,1\}$ and thus A_e is independent of any other event so we are done.

(iii) If $\exists \{v, v'\} = e \in E$ such that $|A_e| = 2$, then due to monotonicity A_e is independent of either W_v or $W_{v'}$ so we are done.

Assume (i),(ii) does not hold. We show that $\nexists \{v, v+1\} = e \in E \colon\ |A_e| = 3$. First note that $\forall e \in E\ p_e \in (0,1)$, because of (i),(ii). For simplicity assume $v = 0$, due to cyclic symmetry the same argument works $\forall v \in [n] - 1 (= \{0, 1, \ldots, n -$

$1\}$). Let us denote $W_0 = \{w_0^1, w_0^2\}$, $W_1 = \{w_1^1, w_1^2\}$. Assume without loss of generality $\overline{A_e} = \{w_0^1\} \times \{w_1^1\}$ and let us denote $\Pr(\{w_1^1\}) = x$, $\Pr(\{w_0^1\}) = y$. Because of (i) we should have $x, y \in (0,1)$. Due to minimality $y = \frac{1-p_0}{x}$, also due to minimality, (ii) and Proposition 7.4.16 $y = f_{\vec{p}}(x)$. Here we used the observation that if $(p_1/x, p_2, p_3, \dots, p_{n-2}, p_{n-1}/y)$ is above the Shearer bound for a proper induced subgraph of $P_{1\dots n-1}$ not containing some $v' \in [n-1]$ then due to Theorem 7.4.7 and Proposition 7.4.16 we can find events $A_{v,v+1} \in W_v \times W_{v+1} \colon v \in [n-1] \setminus \{v'\}$ such that $\Pr(A_0 \cup \bigcup_{v \in [n-1]\setminus v'} A_v) = 1$ implying $p_{v',v'+1} = 0$ due to minimality. But since we assumed neither (i) nor (ii) holds, we can conclude $I_{1\dots n-1}(p_1/x, p_2, p_3, \dots, p_{n-2}, p_{n-1}/y) > 0$ for proper subgraphs. Due to minimality and because $x, y \in (0,1)$ we must also have $\left(\frac{1-p_0}{x}\right)' = f'_{\vec{p}}(x)$, where $\left(\frac{1-p_0}{x}\right)' = \frac{p_0-1}{x^2} < 0$ since $p_0 < 1$. Observe that $\left(\frac{1-p_0}{x}\right)'' = \left(\frac{1-p_0}{x}\right)' \cdot \frac{-2}{x}$ and $f''_{\vec{p}}(x) = f'_{\vec{p}}(x) \cdot \frac{-2}{x - p_1 \cdot \frac{I_{2\dots n-2}(p_i)}{I_{3\dots n-2}(p_i)}}$ where $0 < p_1 \cdot \frac{I_{2\dots n-2}(p_i)}{I_{3\dots n-2}(p_i)} < x$ because we assumed $I_{1\dots n-1}(p_1/x, p_2, p_3, \dots, p_{n-2}, p_{n-1}/y) > 0$ for proper subgraphs. Thus $\left(\frac{1-p_0}{x}\right)'' < f''_{\vec{p}}(x)$, contradicting minimality.

If none of (i)-(iii) holds then the only remaining case to check is $\forall e \in E\colon$ $|A_e| = 1$. Let us denote $W_v = \{w_v^1, w_v^2\}$, then we can assume without loss of generality that $A_{v,v+1} = \{w_v^1\} \times \{w'_{v+1}\}$ for some $w'_{v+1} \in W_{v+1}$. Let $A = \bigtimes_{v \in V} \{w_v^2\}$, then clearly $A \cap \bigcup_{e \in E} A_e = \emptyset$. Also since (i) does not hold, we have $\Pr(A) > 0$, contradicting $\Pr(\bigcup_{e \in E} A_e) = 1$. $\square$

We note that it seems a small miracle that the case $\overline{A_e} = X \times Y$ can be eliminated using this second derivative argument. It is surprising that the corresponding event structure is always suboptimal, as this initially might seem the optimal configuration of the events. The statement of the theorem itself is also surprising because it says there is always a useless link or connection in the case of cyclic structure.

Finally note that Theorem 7.4.17 in the case $n = 3$ (the triangle) can be seen as a geometric statement about the 3-dimensional unit cube if we choose the probability spaces $W_j = [0,1] \cdot e_j$. We can interpret the result as stating that the geometric problem can always be reduced to an essentially 2-dimensional problem.

7.4.3 Shearer bound for subspaces

The Lovász Local Lemma was generalized for subspaces of vector spaces by Ambainis et al. [AKS12] and later a tighter version was proved by Sattath et al. [Sat15] recovering the bound of Shearer [She85] for subspaces. It is not difficult to show that this bound is tight for general vector states.

The quantum analogue of the variable version Lovász Local Lemma is when we have a vector space with tensor product structure, and the "bad projectors" act non-trivially only on a few of the tensor factors. Very recently it was shown by He et al. [HLSZ19] (who also explicitly use some of the insight of this section [Gil16]), that in the quantum variable case Shearer's bound remains tight. However, if the projectors need to commute then Shearer's bound can become loose, similarly to the classical case.

The results of this paper about cyclic dependency graphs therefore provide explicit examples of separation between the classical and quantum Lovász Local Lemma in the variable setting. It seems that the largest separation for cyclic graphs between the two bounds happens when we consider the uniform case for the triangle graph. There the Shearer bound gives 1/3 whereas the tight bound we proved is $\frac{3-\sqrt{5}}{2} = 0.38...$ for the probabilities of the three events.

7.A Optimized upper bound on $\binom{n}{k}$

7.A.1. Lemma. *If $0 < k < n$ are positive integers, then*

$$\binom{n}{k} < \left(\frac{en}{k} - \frac{e}{2}\right)^k. \tag{7.15}$$

Note that the above is an improvement of the more standard $\binom{n}{k} < \left(\frac{en}{k}\right)^k$ bound.
Proof:
We use the following upper bound [Juk11, Corollary 22.9] on binomial coefficients

$$\begin{aligned}\forall\, 0 < k < n: \ \binom{n}{k} &\leq 2^{n\cdot H(k/n)}\\ &= 2^{n\left(-\frac{k}{n}\log_2\left(\frac{k}{n}\right)-\frac{n-k}{n}\log_2\left(\frac{n-k}{n}\right)\right)}\\ &= e^{\left(k\ln\left(\frac{n}{k}\right)+(n-k)\ln\left(\frac{n}{n-k}\right)\right)}.\end{aligned}$$

(In the statement above, $H(p) = -p\log_2(p) - (1-p)\log_2(1-p)$ denotes the binary entropy.) We use this inequality to prove (7.15). It remains to show that

$$e^{\left(k\ln\left(\frac{n}{k}\right)+(n-k)\ln\left(\frac{n}{n-k}\right)\right)} < \left(\frac{en}{k} - \frac{e}{2}\right)^k$$

$$\Updownarrow$$

$$\ln\left(\frac{n}{k}\right) + \left(\frac{n}{k} - 1\right)\ln\left(\frac{n}{n-k}\right) < 1 + \ln\left(\frac{n}{k} - \frac{1}{2}\right)$$

$$\Updownarrow$$

$$0 < 1 + \ln\left(1 - \frac{1}{2}\frac{k}{n}\right) + \left(\frac{n}{k} - 1\right)\ln\left(1 - \frac{k}{n}\right). \tag{7.16}$$

For $x = k/n$ let $f(x) := 1 + \ln(1-x/2) + (1/x - 1)\ln(1-x)$ denote the right-hand side of (7.16). In order to prove that $f(x) > 0$ for all $x \in (0,1)$, we first observe that

$$\lim_{x\to 0} f(x) = 1 + \lim_{x\to 0}\frac{\ln(1-x)}{x} = 0.$$

Finally we prove $f(x) > 0$ by showing that $f'(x) > 0$ for all $x \in (0,1)$:

$$\begin{aligned} f'(x) &= -\frac{1}{2-x} - \frac{1}{x^2}\ln\left(\frac{1}{1-x}\right) - (1/x-1)\frac{1}{1-x} \\ &= -\frac{1}{2-x} + \frac{1}{x^2}\ln\left(\frac{1}{1-x}\right) + \frac{1}{x} \\ &= \frac{1}{x} - \frac{1}{2-x} + \frac{1}{x^2}\ln\left(\frac{1+x/(2-x)}{1-x/(2-x)}\right) \\ &\overset{(7.17)}{>} \frac{1}{x} - \frac{1}{2-x} + \frac{1}{x^2}\frac{2x}{2-x} = 0. \end{aligned}$$

The last inequality can be deduced using the Taylor series $\forall y \in (-1,1)\ \ln(1+y) = \sum_{\ell=1}^{\infty} \frac{(-y)^\ell}{-\ell}$:

$$\begin{aligned} \forall z \in (0,1):\ \ln\left(\frac{1+z}{1-z}\right) &= \ln(1+z) - \ln(1-z) \\ &= 2z\sum_{k=0}^{\infty}\frac{z^{2k}}{2k+1} > 2z. \end{aligned} \tag{7.17}$$

□

Note that by using one more term in (7.17) one can strengthen (7.15) to $\binom{n}{k} < \left(\frac{en}{k} - \frac{e}{2} - \frac{ek}{42n}\right)^k$.

Bibliography

[Aar06] Scott Aaronson. The ten most annoying questions in quantum computing, 2006. `https://www.scottaaronson.com/blog/?p=112`.

[Aar09] Scott Aaronson. Quantum copy-protection and quantum money. In *Proceedings of the 24th IEEE Conference on Computational Complexity (CCC)*, pages 229–242, 2009. arXiv: `1110.5353`

[Aar18] Scott Aaronson. Shadow Tomography of Quantum States. In *Proceedings of the 50th ACM Symposium on Theory of Computing (STOC)*, pages 325–338, 2018. arXiv: `1711.01053`

[ABP17] Srinivasan Arunachalam, Jop Briët, and Carlos Palazuelos. Quantum query algorithms are completely bounded forms. arXiv: `1711.07285`, 2017.

[AC12] Scott Aaronson and Paul Christiano. Quantum money from hidden subspaces. In *Proceedings of the 44th ACM Symposium on Theory of Computing (STOC)*, pages 41–60, 2012. arXiv: `1203.4740`

[AGIK09] Dorit Aharonov, Daniel Gottesman, Sandy Irani, and Julia Kempe. The power of quantum systems on a line. *Communications in Mathematical Physics*, 287(1):41–65, 2009, arXiv: `0705.4077`.

[AGJK19] Andris Ambainis, András Gilyén, Stacey Jeffery, and Martins Kokainis. Quadratic speedup for finding marked vertices by quantum walks. arXiv: `1903.07493`, 2019.

[AK16] Sanjeev Arora and Satyen Kale. A combinatorial, primal-dual approach to semidefinite programs. *Journal of the ACM*, 63(2):12:1–12:35, 2016. Earlier version in STOC'07.

[AKS12] Andris Ambainis, Julia Kempe, and Or Sattath. A quantum Lovász local lemma. *Journal of the ACM*, 59(5):24, 2012. arXiv: `0911.1696`

[Amb99] Andris Ambainis. A better lower bound for quantum algorithms searching an ordered list. In *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 352–357, 1999. arXiv: `quant-ph/9902053`

[Amb00] Andris Ambainis. Quantum lower bounds by quantum arguments. In *Proceedings of the 32nd ACM Symposium on Theory of Computing (STOC)*, pages 636–643, 2000. arXiv: `quant-ph/0002066`

[Amb04] Andris Ambainis. Quantum walk algorithm for element distinctness. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 22–31, 2004. arXiv: `quant-ph/0311001`

[Amb12] Andris Ambainis. Variable time amplitude amplification and quantum algorithms for linear algebra problems. In *Proceedings of the 29th Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 636–647, 2012. arXiv: `1010.4458`

[AP16] Alexei B. Aleksandrov and Vladimir V. Peller. Operator Lipschitz functions. *Russian Mathematical Surveys*, 71(4):605–702, 2016. arXiv: `1611.01593`

[vAG19] Joran van Apeldoorn and András Gilyén. Improvements in quantum SDP-solving with applications. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, 2019. (to appear) arXiv: `1804.05058`

[vAGGdW17] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Quantum SDP-solvers: Better upper and lower bounds. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 403–414, 2017. arXiv: `1705.01843`

[vAGGdW18] Joran van Apeldoorn, András Gilyén, Sander Gribling, and Ronald de Wolf. Convex optimization using quantum oracles. arXiv: `1809.00643`, 2018.

[AS92] Noga Alon and Joel Spencer. *The Probabilistic Method.* John Wiley, 1992.

[AŠ06] Andris Ambainis and Robert Špalek. Quantum algorithms for matching and network flows. In *Proceedings of the 23rd Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 172–183, 2006. arXiv: `quant-ph/0508205`

[AS08] Noga Alon and Joel H. Spencer. *The Probabilistic Method.* Wiley-Interscience, third edition, 2008.

[AS18] Simon Apers and Alain Sarlette. Quantum fast-forwarding Markov chains. arXiv: `1804.02321`, 2018.

[AvDK$^+$08] Dorit Aharonov, Wim van Dam, Julia Kempe, Zeph Landau, Seth Lloyd, and Oded Regev. Adiabatic quantum computation is equivalent to standard quantum computation. *SIAM Review*, 50(4):755–787, 2008, arXiv: `quant-ph/0405098`.

[Azo94] E. Michael Azoff. *Neural Network Time Series Forecasting of Financial Markets.* John Wiley & Sons, New York, NY, USA, 1st edition, 1994.

[BACS07] Dominic W. Berry, Graeme Ahokas, Richard Cleve, and Barry C. Sanders. Efficient quantum algorithms for simulating sparse Hamiltonians. *Communications in Mathematical Physics*, 270(2):359–371, 2007. arXiv: `quant-ph/0508139`

[Bal12] Pierre Baldi. Autoencoders, unsupervised learning, and deep architectures. In *Unsupervised and Transfer Learning - Workshop held at ICML 2011*, pages 37–50, 2012.

[BBBV97] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani. Strengths and weaknesses of quantum computing. *SIAM Journal on Computing*, 26(5):1510–1523, 1997. arXiv: `quant-ph/9701001`

[BBC$^+$01] Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald de Wolf. Quantum lower bounds by polynomials. *Journal of the ACM*, 48(4):778–797, 2001. Earlier version in FOCS'98. arXiv: `quant-ph/9802049`

[BBHT98] Michel Boyer, Gilles Brassard, Peter Høyer, and Alain Tapp. Tight bounds on quantum searching. *Fortschritte der Physik*, 46(4–5):493–505, 1998. arXiv: `quant-ph/9605034`

[BCC$^+$14] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma. Exponential improvement in precision for simulating sparse Hamiltonians. In *Proceedings of* the 46th ACM Symposium on Theory of Computing (STOC)*, pages 283–292, 2014. arXiv: `1312.1414`

[BCC$^+$15] Dominic W. Berry, Andrew M. Childs, Richard Cleve, Robin Kothari, and Rolando D. Somma. Simulating Hamiltonian dynamics with a truncated Taylor series. *Physical Review Letters*, 114(9):090502, 2015. arXiv: `1412.4687`

[BCK15] Dominic W. Berry, Andrew M. Childs, and Robin Kothari. Hamiltonian simulation with nearly optimal dependence on all parameters. In *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 792–809, 2015. arXiv: `1501.01715`

[Bec91] József Beck. An algorithmic approach to the Lovász local lemma. I. *Random Structures & Algorithms*, 2(4):343–366, 1991.

[Bel15] Aleksandrs Belovs. Variations on quantum adversary. arXiv: `1504.06943`, 2015.

[Bes05] Arvid J. Bessen. Lower bound for quantum phase estimation. *Physical Review A*, 71(4):042313, 2005. arXiv: `quant-ph/0412008`

[BFP$^+$73] Manuel Blum, Robert W. Floyd, Vaughan R. Pratt, Ronald L. Rivest, and Robert Endre Tarjan. Time bounds for selection. *Journal of Computer and System Sciences*, 7(4):448–461, 1973.

[BG94] Mihir Bellare and Shafi Goldwasser. The complexity of decision versus search. *SIAM Journal on Computing*, 23(1):97–119, 1994.

[BHMT02] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. In *Quantum Computation and Quantum Information: A Millennium Volume*, volume 305 of *Contemporary Mathematics Series*, pages 53–74. AMS, 2002. arXiv: `quant-ph/0005055`

[BKL$^+$17a] Fernando G. S. L. Brandão, Amir Kalev, Tongyang Li, Cedric Yen-Yu Lin, Krysta M. Svore, and Xiaodi Wu. Exponential quantum speed-ups for semidefinite programming with applications to quantum learning, 2017, arXiv: `1710.02581v1`. First arXiv version.

[BKL$^+$17b] 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. arXiv: `1710.02581`, 2017.

[BKL$^+$18] 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, 2018, arXiv: `1710.02581v2`.

[Bra11] S. Bravyi. Efficient algorithm for a quantum analogue of 2-SAT. In Kazem Mahdavi, Deborah Koslover, and Leonard L. Brown, editors, *Contemporary Mathematics*, volume 536. American Mathematical Society, 2011, arXiv: `quant-ph/0602108`.

[BS17] Fernando G. S. L. Brandão and Krysta M. Svore. Quantum speed-ups for solving semidefinite programs. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 415–426, 2017. arXiv: `1609.05537`

[Bub15] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Foundations and Trends in Machine Learning*, 8(3–4):231–357, 2015. arXiv: `1405.4980`

[Bul05a] David Bulger. Quantum basin hopping with gradient-based local optimisation. Unpublished. arXiv: `quant-ph/0507193`, 2005.

[Bul05b] David Bulger. Quantum computational gradient estimation. Unpublished. arXiv: `quant-ph/0507109`, 2005.

[BV05] Sergey Bravyi and Mikhail Vyalyi. Commutative version of the local Hamiltonian problem and common eigenspace problem. *Quantum Information and Computation*, 5(3):187–215, 2005, arXiv: `quant-ph/0308021`.

[CCLW18] Shouvanik Chakrabarti, Andrew M. Childs, Tongyang Li, and Xiaodi Wu. Quantum algorithms and lower bounds for convex optimization. arXiv: `1809.01731`, 2018.

[CEMM98] Richard Cleve, Artur Ekert, Chiara Macchiavello, and Michele Mosca. Quantum algorithms revisited. *Proceedings of the Royal Society A*, 454(1969):339–354, 1998. arXiv: `quant-ph/9708016`

[CGJ19] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation. In *Proceedings of the 46th International Colloquium on Automata, Languages, and Programming (ICALP)*, 2019. (to appear) arXiv: `1804.01973`

[CGM+09] Richard Cleve, Daniel Gottesman, Michele Mosca, Rolando D. Somma, and David L. Yonge-Mallo. Efficient discrete-time simulations of continuous-time quantum query algorithms. In *Proceedings of the 41st ACM Symposium on Theory of Computing (STOC)*, pages 409–416, 2009. arXiv: `0811.4428`

[CHTW04] Richard Cleve, Peter Høyer, Benjamin Toner, and John Watrous. Consequences and limits of nonlocal strategies. In *Proceedings of the 19th IEEE Conference on Computational Complexity (CCC)*, pages 236–249, 2004. arXiv: `quant-ph/0404076`

[Chu36a] Alonzo Church. A note on the entscheidungsproblem. *Journal of Symbolic Logic*, 1(1):40–41, June 1936.

[Chu36b] Alonzo Church. An unsolvable problem of elementary number theory. *American Journal of Mathematics*, 58(2):345–363, April 1936.

[CJKM13] Andrew M. Childs, Stacey Jeffery, Robin Kothari, and Frédéric Magniez. A time-efficient quantum walk for 3-distinctness using nested updates. arXiv: `1302.7316`, 2013.

[CKS17] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. Quantum algorithm for systems of linear equations with exponentially improved dependence on precision. *SIAM Journal on Computing*, 46(6):1920–1950, 2017. arXiv: `1511.02306`

[CMN+17] Andrew M. Childs, Dmitri Maslov, Yunseong Nam, Neil J. Ross, and Yuan Su. Toward the first quantum simulation with quantum speedup. arXiv: `1711.10980`, 2017.

[Cor18] Arjan Cornelissen. Quantum gradient estimation and its application to quantum reinforcement learning. Master's thesis, Technische Universiteit Delft, 2018.

[CPGW15] Toby S. Cubitt, David Perez-Garcia, and Michael M. Wolf. Undecidability of the spectral gap. *Nature*, 528(7581):207–211, 2015, arXiv: `1502.04573`.

[CS17] Anirban Narayan Chowdhury and Rolando D. Somma. Quantum algorithms for Gibbs sampling and hitting-time estimation. *Quantum Information and Computation*, 17(1&2):41–64, 2017. arXiv: `1603.02940`

[CW12] Andrew M. Childs and Nathan Wiebe. Hamiltonian simulation using linear combinations of unitary operations. *Quantum Information and Computation*, 12(11&12):901–924, 2012. arXiv: `1202.5822`

[Deu85] David Deutsch. Quantum theory, the church–turing principle and the universal quantum computer. *Proceedings of the Royal Society A*, 400(1818):97–117, 1985.

[DH96] Christoph Dürr and Peter Høyer. A quantum algorithm for finding the minimum. arXiv: `quant-ph/9607014`, 1996.

[DH17] Cătălin Dohotaru and Peter Høyer. Controlled quantum amplification. In *Proceedings of the 44th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 18:1–18:13, 2017.

[DHHM06] Christoph Dürr, Mark Heiligman, Peter Høyer, and Mehdi Mhalla. Quantum query complexity of some graph problems. *SIAM Journal on Computing*, 35(6):1310–1328, 2006. arXiv: `quant-ph/0401091`

[Dol46] Charles L. Dolph. A current distribution for broadside arrays which optimizes the relationship between beam width and side-lobe level. *Proceedings of the IRE*, 34(6):335–348, 1946.

[EL75] Pál Erdős and László Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. In *Infinite and finite sets (Colloq., Keszthely, 1973; dedicated to P. Erdős on his 60th birthday), Vol. II*, pages 609–627. Colloq. Math. Soc. János Bolyai, Vol. 10. North-Holland, Amsterdam, 1975.

[Eld03] Yonina C. Eldar. A semidefinite programming approach to optimal unambiguous discrimination of quantum states. *IEEE Transactions on Information Theory*, 49:446–456, 2003. arXiv: `quant-ph/0206093`

[EY07] Alexandre Eremenko and Peter Yuditskii. Uniform approximation of sgn(x) by polynomials and entire functions. *Journal d'Analyse Mathématique*, 101(1):313–324, 2007. arXiv: `math/0604324`

[EY11] Alexandre Eremenko and Peter Yuditskii. Polynomials of the best uniform approximation to sgn(x) on two intervals. *Journal d'Analyse Mathématique*, 114(1):285, 2011. arXiv: `1008.3765`

[Fey82] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6-7):467–488, 1982.

[FGG14] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. arXiv: `1411.4028`, 2014.

[FGGS00] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. Quantum computation by adiabatic evolution, 2000, arXiv: `quant-ph/0001106`.

[FMMS16] Roy Frostig, Cameron Musco, Christopher Musco, and Aaron Sidford. Principal component projection without principal component analysis. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 2349–2357, 2016. arXiv: `1602.06872`

[FN09] Yuliya B. Farforovskaya and Ludmila N. Nikolskaya. Modulus of continuity of operator functions. *St. Petersburg Math. J. – Algebra i Analiz*, 20(3):493–506, 2009.

[FN18] Edward Farhi and Hartmut Neven. Classification with quantum neural networks on near term processors. arXiv: `1802.06002`, 2018.

[Fol99] Gerald B. Folland. *Real Analysis: Modern Techniques and Their Applications, 2nd Edition.* Wiley, 1999.

[GAW19] András Gilyén, Srinivasan Arunachalam, and Nathan Wiebe. Optimizing quantum optimization algorithms via faster quantum gradient computation. In *Proceedings of the 30th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1425–1444, 2019. arXiv: `1711.00465`

[Gev18] Maurice Gevrey. Sur la nature analytique des solutions des équations aux dérivées partielles. Premier mémoire. *Annales Scientifiques de l'École Normale Supérieure*, 3(35):129–190, 1918.

[Gil14] András Gilyén. Quantum walk based search methods and algorithmic applications. Master's thesis, Eötvös Loránd University, 2014.

[Gil16] András Gilyén. Bounds for probabilities in the variable setting for trees and cycles. Unpublished manuscript, June 2016.

[GLS88] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer, 1988.

[Got96] Daniel Gottesman. Class of quantum error-correcting codes saturating the quantum Hamming bound. *Physical Review A*, 54(3):1862–1868, 1996. arXiv: `quant-ph/9604038`

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th ACM Symposium on Theory of Computing (STOC)*, pages 212–219, 1996. arXiv: `quant-ph/9605043`

[Gro05] Lov K. Grover. Fixed-point quantum search. *Physical Review Letters*, 95(15):150501, 2005. arXiv: `quant-ph/0503205`

[GS16] András Gilyén and Or Sattath. On preparing ground states of gapped hamiltonians: An efficient quantum Lovász local lemma [full version], 2016. arXiv: `1611.08571`

[GS17] András Gilyén and Or Sattath. On preparing ground states of gapped Hamiltonians: An efficient quantum Lovász local lemma. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 439–450, 2017. arXiv: `1611.08571`

[GSLW19] 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 ACM Symposium on Theory of Computing (STOC)*, 2019. (to appear) arXiv: `1806.01838`

[GT12] Geoff Gordon and Ryan Tibshirani. Optimization lecture notes, lecture 3. 2012.

[GTC17] Yimin Ge, Jordi Tura, and J. Ignacio Cirac. Faster ground state preparation and high-precision ground energy estimation with fewer qubits. arXiv: `1712.03193`, 2017.

[GW95] Michel X. Goemans and David P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM*, 42(6):1115–1145, 1995. Earlier version in STOC'94.

[GW08] Andreas Griewank and Andrea Walther. *Evaluating derivatives - principles and techniques of algorithmic differentiation.* SIAM, 2. edition, 2008.

[Haa18] Jeongwan Haah. Product decomposition of periodic functions in quantum signal processing. arXiv: `1806.10236`, 2018.

[Har16] David G. Harris. Lopsidependency in the Moser-Tardos framework: Beyond the lopsided Lovász local lemma. *ACM Transactions on Algorithms*, 13(1):17:1–17:26, 2016. arXiv: `1610.02420`

[Has07] Matthew B. Hastings. An area law for one-dimensional quantum systems. *Journal of Statistical Mechanics: Theory and Experiment*, 2007(08):P08024, 2007, arXiv: `0705.2024`.

[HHL09] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15):150502, 2009. arXiv: `0811.3171`

[HLL$^+$17] Kun He, Liang Li, Xingwu Liu, Yuyi Wang, and Mingji Xia. Variable-version Lovász local lemma: Beyond Shearer's bound. In *Proceedings of the 58th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 451–462, 2017. arXiv: `1709.05143`

[HLM17] Aram W. Harrow, Cedric Yen-Yu Lin, and Ashley Montanaro. Sequential measurements, disturbance and property testing. In *Proceedings of the 28th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1598–1611, 2017. arXiv: `1607.03236`

[HLŠ07] Peter Høyer, Troy Lee, and Robert Špalek. Negative weights make adversaries stronger. In *Proceedings of the 39th ACM Symposium on Theory of Computing (STOC)*, pages 526–535, 2007. arXiv: `quant-ph/0611054`

[HLSZ19] Kun He, Qian Li, Xiaoming Sun, and Jiapeng Zhang. Quantum Lovász local lemma: Shearer's bound is tight. In *Proceedings of the 51st ACM Symposium on Theory of Computing (STOC)*, 2019. (to appear) arXiv: `1804.07055`

[HLZ$^+$17] Yong He, Ming-Xing Luo, E. Zhang, Hong-Ke Wang, and Xiao-Feng Wang. Decompositions of n-qubit Toffoli gates with linear circuit complexity. *International Journal of Theoretical Physics*, 56(7):2350–2361, 2017.

[HM18] Yassine Hamoudi and Frédéric Magniez. Quantum Chebyshev's inequality and applications. arXiv: `1807.06456`, 2018.

[Hø00] Peter Høyer. Arbitrary phases in quantum amplitude amplification. *Physical Review A*, 62(5):052304, 2000. arXiv: `quant-ph/0006031`

[HŠ05] Peter Høyer and Robert Špalek. Lower bounds on quantum query complexity. *Bulletin of the EATCS*, 87:78–103, 2005. arXiv: `quant-ph/0509153`

[HV15] Nicholas Harvey and Jan Vondrák. An algorithmic proof of the Lovász local lemma via resampling oracles. In *Proceedings of* *the 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, 2015.

[JKK$^+$17] Prateek Jain, Sham M. Kakade, Rahul Kidambi, Praneeth Netrapalli, and Aaron Sidford. Accelerating stochastic gradient descent. arXiv: `1704.08227`, 2017.

[Jor75] Camille Jordan. Essai sur la géométrie à n dimensions. *Bulletin de la Société Mathématique de France*, 3:103–174, 1875.

[Jor05] Stephen P. Jordan. Fast quantum algorithm for numerical gradient estimation. *Physical Review Letters*, 95(5):050501, 2005. arXiv: `quant-ph/0405146`

[Jor08] Stephen P. Jordan. *Quantum Computation Beyond the Circuit Model*. PhD thesis, Massachusetts Institute of Technology, 2008. arXiv: `0809.2307`

[Jozs18] Richard Jozsa. Quantum information and computation, 2018. Lecture notes.

[Juk11] Stasys Jukna. *Extremal Combinatorics - With Applications in Computer Science (2nd ed.)*. Texts in Theoretical Computer Science. Springer, 2011.

[Key02] Michael Keyl. Fundamentals of quantum information theory. *Physics Reports*, 369(5):431 – 548, 2002. arXiv: `quant-ph/0202122`

[Kit03] Alexei Yu. Kitaev. Fault-tolerant quantum computation by anyons. *Annals of Physics*, 303(1):2–30, 2003, arXiv: `quant-ph/9707021`.

[KL18] Iordanis Kerenidis and Alessandro Luongo. Quantum classification of the MNIST dataset via slow feature analysis. arXiv: `1805.08837`, 2018.

[KLL$^+$17] Shelby Kimmel, Cedric Yen-Yu Lin, Guang Hao Low, Maris Ozols, and Theodore J. Yoder. Hamiltonian simulation with optimal sample complexity. *npj Quantum Information*, 3(1):13, 2017. arXiv: `1608.00281`

[KMOR16] Hari Krovi, Frédéric Magniez, Maris Ozols, and Jérémie Roland. Quantum walks can find a marked element on any graph. *Algorithmica*, 74(2):851–907, 2016. arXiv: `1002.2419`

[Kol16] Vladimir Kolmogorov. Commutativity in the algorithmic Lovász local lemma. In *Proceedings of the 57th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 780–787, 2016, arXiv: `1506.08547`.

[KP17a] Iordanis Kerenidis and Anupam Prakash. Quantum gradient descent for linear systems and least squares. arXiv: `1704.04992`, 2017.

[KP17b] Iordanis Kerenidis and Anupam Prakash. Quantum recommendation systems. In *Proceedings of the 8th Innovations in Theoretical Computer Science Conference (ITCS)*, pages 49:1–49:21, 2017. arXiv: `1603.08675`

[KP18] Iordanis Kerenidis and Anupam Prakash. A quantum interior point method for LPs and SDPs. arXiv: `1808.09266`, 2018.

[KST93] Jan Kratochvíl, Petr Savický, and Zsolt Tuza. One more occurrence of variables makes satisfiability jump from trivial to np-complete. *SIAM Journal on Computing*, 22(1):203–210, 1993.

[KSz11] Kashyap Babu Rao Kolipaka and Márió Szegedy. Moser and Tardos meet Lovász. In *Proceedings of the 43th ACM Symposium on Theory of Computing (STOC)*, pages 235–244, 2011.

[Las01] Jean Lasserre. Global optimization with polynomials and the problem of moments. *SIAM Journal on Optimization*, 11(3):796–817, 2001.

[LC16] Guang Hao Low and Isaac L. Chuang. Hamiltonian simulation by qubitization. arXiv: `1610.06546`, 2016.

[LC17a] Guang Hao Low and Isaac L. Chuang. Hamiltonian simulation by uniform spectral amplification. arXiv: `1707.05391`, 2017.

[LC17b] Guang Hao Low and Isaac L. Chuang. Optimal Hamiltonian simulation by quantum signal processing. *Physical Review Letters*, 118(1):010501, 2017. arXiv: `1606.02685`

[Li05] Jianping Li. General explicit difference formulas for numerical differentiation. *Journal of Computational and Applied Mathematics*, 183(1):29 – 52, 2005.

[Llo96] Seth Lloyd. Universal quantum simulators. *Science*, 273(5278):1073–1078, 1996.

[LMR$^+$11] Troy Lee, Rajat Mittal, Ben W. Reichardt, Robert Špalek, and Márió Szegedy. Quantum query complexity of state conversion. In *Proceedings of the 52nd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 344–353, 2011. arXiv: `1011.3020`

[LMR14] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum principal component analysis. *Nature Physics*, 10:631–633, 2014. arXiv: `1307.0401`

[Low18] Guang Hao Low. Hamiltonian simulation with nearly optimal dependence on spectral norm. arXiv: `1807.03967`, 2018.

[LPL14] Pedro C. S. Lara, Renato Portugal, and Carlile Lavor. A new hybrid classical-quantum algorithm for continuous global optimization problems. *J. Global Optimization*, 60(2):317–331, 2014. arXiv: `1301.4667`

[LRS15] James R. Lee, Prasad Raghavendra, and David Steurer. Lower bounds on the size of semidefinite programming relaxations. In *Proceedings of the 47th ACM Symposium on Theory of Computing (STOC)*, pages 567–576, 2015. arXiv: `1411.6317`

[LSV18] Yin Tat Lee, Aaron Sidford, and Santosh S. Vempala. Efficient convex optimization with membership oracles. In *Proceedings of the 31st Conference On Learning Theory (COLT)*, pages 1292–1294, 2018. arXiv: `1706.07357`

[LSW15] Yin Tat Lee, Aaron Sidford, and Sam Chiu-wai Wong. A faster cutting plane method and its implications for combinatorial and convex optimization. In *Proceedings of the 56th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 1049–1065, 2015. arXiv: `1508.04874`

[LYC16] Guang Hao Low, Theodore J. Yoder, and Isaac L. Chuang. Methodology of resonant equiangular composite quantum gates. *Physical Review X*, 6(4):041067, 2016. arXiv: `1603.03996`

[MEAG$^+$18] Sam McArdle, Suguru Endo, Alan Aspuru-Guzik, Simon Benjamin, and Xiao Yuan. Quantum computational chemistry. arXiv: `1808.10402`, 2018.

[Mey00] Carl Meyer. *Matrix analysis and applied linear algebra.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

[MNRS11] Frédéric Magniez, Ashwin Nayak, Jérémie Roland, and Miklos Santha. Search via quantum walk. *SIAM Journal on Computing*, 40(1):142–164, 2011. arXiv: `quant-ph/0608026`

[Mon11] Ashley Montanaro. Quantum computation, 2011. Lecture notes.

[Mon15] Ashley Montanaro. Quantum speedup of Monte Carlo methods. *Proceedings of the Royal Society A*, 471(2181), 2015. arXiv: `1504.06987`

[Mos09] Robin A. Moser. A constructive proof of the Lovász local lemma. In *Proceedings of the 41st ACM Symposium on Theory of Computing (STOC)*, pages 343–350, 2009. arXiv: `0810.4812`

[MS77] Baidyanath Misra and Ennackal C. G. Sudarshan. The zeno's paradox in quantum theory. *Journal of Mathematical Physics*, 18(4):756–763, 1977.

[MT10] Robin A. Moser and Gábor Tardos. A constructive proof of the general Lovász local lemma. *Journal of the ACM*, 57(2):11, 2010. arXiv: `0903.0544`

[MW05] Chris Marriott and John Watrous. Quantum Arthur–Merlin games. *Computational Complexity*, 14(2):122–152, 2005. arXiv: `cs/0506068`

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum computation and quantum information.* Cambridge University Press, 2000.

[NR96] C. Andrew Neff and John H. Reif. An efficient algorithm for the complex roots problem. *Journal of Complexity*, 12(2):81 – 115, 1996.

[NW99] Ashwin Nayak and Felix Wu. The quantum query complexity of approximating the median and related statistics. In *Proceedings of the 31th ACM Symposium on Theory of Computing (STOC)*, pages 384–393, 1999. arXiv: `quant-ph/9804066`

[NWZ09] Daniel Nagaj, Pawel Wocjan, and Yong Zhang. Fast amplification of QMA. *Quantum Information and Computation*, 9(11&12):1053–1068, 2009. arXiv: `0904.1549`

[Par00] Pablo A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization.* PhD thesis, California Institute of Technology, 2000.

[PMS$^+$14] Alberto Peruzzo, Jarrod McClean, Peter Shadbolt, Man-Hong Yung, Xiao-Qi Zhou, Peter J Love, Alán Aspuru-Guzik, and Jeremy L. O'brien. A variational eigenvalue solver on a photonic quantum processor. *Nature Communications*, 5, 2014. arXiv: `1304.3061`

[PVWC08] David Pérez-García, Frank Verstraete, Michael M. Wolf, and J. Ignacio Cirac. PEPS as unique ground states of local Hamiltonians. *Quantum Information and Computation*, 8(6):650–663, 2008, arXiv: `0707.2260`.

[PW09] David Poulin and Pawel Wocjan. Preparing ground states of quantum many-body systems on a quantum computer. *Physical Review Letters*, 102(13):130503, 2009. arXiv: `0809.2705`

[RHW86] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. In David E. Rumelhart, James L. McClelland, and PDP Research Group, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1*, pages 318–362. MIT Press, Cambridge, MA, USA, 1986.

[Rie70] Marc A. Rieffel. Lectures on bochner lebesgue integration, university of california, berkeley. 1970.

[RML14] Patrick Rebentrost, Masoud Mohseni, and Seth Lloyd. Quantum support vector machine for big data classification. *Physical Review Letters*, 113(13):130503, 2014. arXiv: `1307.0471`

[ROAG17] Jonathan Romero, Jonathan P. Olson, and Alan Aspuru-Guzik. Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology*, 2(4):045001, 2017. arXiv: `1612.02806`

[Roc70] Ralph Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, NJ, USA, 1970.

[RSPL16] Patrick Rebentrost, Maria Schuld, Francesco Petruccione, and Seth Lloyd. Quantum gradient descent and Newton's method for constrained polynomial optimization. arXiv: `1612.01789`, 2016.

[SA15] Or Sattath and Itai Arad. A constructive quantum Lovász local lemma for commuting projectors. *Quantum Information and Computation*, 15(11&12):987–996, 2015. arXiv: `1310.7766`

[Sat15] Or Sattath. On the restrictiveness of quantum satisfiability. Unpublished Manuscript, 2015.

[SBSW18] Maria Schuld, Alex Bocharov, Krysta M. Svore, and Nathan Wiebe. Circuit-centric quantum classifiers. arXiv: `1804.00633`, 2018.

[SCV13] Martin Schwarz, Toby S. Cubitt, and Frank Verstraete. Quantum information-theoretic proof of the commutative quantum Lovász local lemma. arXiv: `1311.6474`, 2013.

[She85] James B. Shearer. On a problem of Spencer. *Combinatorica*, 5(3):241–245, 1985.

[Sho97] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on Computing*, 26(5):1484–1509, 1997. Earlier version in FOCS'94. arXiv: `quant-ph/9508027`

[Sho03] Peter W. Shor. Why haven't more quantum algorithms been found? *Journal of the ACM*, 50(1):87–90, 2003.

[SMLM16] Or Sattath, Siddhardh C. Morampudi, Chris R. Laumann, and Roderich Moessner. When a local Hamiltonian must be frustration-free. *PNAS*, 113(23):6433–6437, 2016, arXiv: `1509.07766`.

[SMM09] Lana Sheridan, Dmitri Maslov, and Michele Mosca. Approximating fractional time quantum evolution. *Journal of Physics A: Mathematical and Theoretical*, 42(18):185302, 2009. arXiv: `0810.3843`

[SV14] Sushant Sachdeva and Nisheeth K. Vishnoi. Faster algorithms via approximation theory. *Found. Trends Theor. Comput. Sci.*, 9(2):125–210, 2014.

[Sze13] Márió Szegedy. The Lovász local lemma - A survey. In *Computer Science - Theory and Applications - 8th International Computer Science Symposium in Russia, CSR 2013, Ekaterinburg, Russia, June 25-29, 2013. Proceedings*, pages 1–11, 2013.

[Szeg04] Márió Szegedy. Quantum speed-up of Markov chain based algorithms. In *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 32–41, 2004. arXiv: `quant-ph/0401053`

[TCR10] Marco Tomamichel, Roger Colbeck, and Renato Renner. Duality between smooth min- and max-entropies. *IEEE Transactions on Information Theory*, 56(9):4674–4681, 2010. arXiv: `0907.5238`

[Tur37] Alan M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, s2-42(1):230–265, 1937.

[Wat18] John Watrous. *The Theory of Quantum Information.* Cambridge University Press, 2018.

[WBL12] Nathan Wiebe, Daniel Braun, and Seth Lloyd. Quantum algorithm for data fitting. *Physical Review Letters*, 109(5):050505, 2012. arXiv: `1204.5242`

[WHT15] Dave Wecker, Matthew B. Hastings, and Matthias Troyer. Progress towards practical quantum variational algorithms. *Physical Review A*, 92(4):042303, 2015. arXiv: `1507.08969`

[WK17] Nathan Wiebe and Ram Shankar Siva Kumar. Hardening quantum machine learning against adversaries. arXiv: `1711.06652`, 2017.

[WKGK16] Oscar Dahlsten Wan, Kwok Ho, Hlér Kristjánsson, Robert Gardner, and M. S. Kim. Quantum generalisation of feedforward neural networks. arXiv: `1612.01045`, 2016.

[WKS15] Nathan Wiebe, Ashish Kapoor, and Krysta M. Svore. Quantum nearest-neighbor algorithms for machine learning. *Quantum Information and Computation*, 15(3&4):318–358, 2015. arXiv: `1401.2142`

[Wu17] Xiaodi Wu. Personal communication. Email, November 2017.

[YLC14] Theodore J. Yoder, Guang Hao Low, and Isaac L. Chuang. Fixed-point quantum search with an optimal number of queries. *Physical Review Letters*, 113(21):210501, 2014. arXiv: `1409.3305`

András Gilyén Quantum Singular Value Transformation & Its Algorithmic Applications