



AFRL-RI-RS-TR-2021-121

QUANTUM COMPUTING ALGORITHMS FOR DECISION MAKING UNDER UNCERTAINTY

INTERNATIONAL BUSINESS MACHINES (IBM) CORP.

JULY 2021

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

**AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the 88th ABW, Wright-Patterson AFB Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (<http://www.dtic.mil>).

AFRL-RI-RS-TR-2021-121 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /

KRISTI T. MEZZANO
Work Unit Manager

/ S /

GREGORY J. HADYNSKI
Assistant Technical Advisor,
Computing & Communications Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.					
1. REPORT DATE (DD-MM-YYYY) JULY 2021		2. REPORT TYPE FINAL TECHNICAL REPORT		3. DATES COVERED (From - To) MAY 2018 – DEC 2020	
4. TITLE AND SUBTITLE QUANTUM COMPUTING ALGORITHMS FOR DECISION MAKING UNDER UNCERTAINTY				5a. CONTRACT NUMBER FA8750-18-C-0098	
				5b. GRANT NUMBER N/A	
				5c. PROGRAM ELEMENT NUMBER 625319	
6. AUTHOR(S) Lior Horesh, Ken Clarkson, Vasileios Kalantzis, Mark Squillante, Shashanka Ubaru, Amir Abboud				5d. PROJECT NUMBER CYDT	
				5e. TASK NUMBER IB	
				5f. WORK UNIT NUMBER M1	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) IBM Thomas J. Watson Research Center 1101 Kitchawan Rd Yorktown Heights NY 10598				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Research Laboratory/RITQ 525 Brooks Road Rome NY 13441-4505				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/RI	
				11. SPONSOR/MONITOR'S REPORT NUMBER AFRL-RI-RS-TR-2021-121	
12. DISTRIBUTION AVAILABILITY STATEMENT Approved for Public Release; Distribution Unlimited. PA# AFRL-2021-2198 Date Cleared: 12 Jul 2021					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT The Quantum Computing Algorithms fundamental research project focuses on the challenge of design of quantum computing algorithms viable for near-term noisy quantum systems. An emphasis was set on design of algorithms that avoid taking unrealistic assumptions while offering theoretical rigor regarding the expected computational advantage. The activity addressed three major development initiatives, namely: algorithmic design, implementation and testing, quantum computing application scoping, and data porting.					
15. SUBJECT TERMS Quantum computing algorithms, Quantum advantage, Decision making under uncertainty, Data porting bottleneck, Variational expectations, Phase estimation, Amplitude estimation, Quantum walks, Topological data analysis, Approximate and sparse quantum Fourier transform					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 109	19a. NAME OF RESPONSIBLE PERSON KRISTI T. MEZZANO
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U			19b. TELEPHONE NUMBER (Include area code) N/A

TABLE OF CONTENTS

Section	Page
LIST OF FIGURES	iii
1.0 SUMMARY	1
2.0 INTRODUCTION	5
3.0 METHODS, ASSUMPTIONS, AND PROCEDURES.....	7
3.1 Algorithms - Variational Quantum k-Eigensolver and Monte Carlo Minimizer.....	7
3.2 Algorithms - Improving Variational Quantum Optimization Using CVaR.....	15
3.3 Algorithms - Robust Optimization.....	16
3.4 Algorithms - Structured Markov Processes	18
3.5 Algorithms - Structured Random Walks.....	33
3.6 Algorithms - Quantum Walks.....	40
3.7 Algorithms - Active Learning-Quantum Amplitude Estimation	49
3.8 Algorithms - Topological Data Analysis	55
3.9 Algorithms - Betti Number by Chebyshev Approximation.....	58
3.10 Applications - Topological Classifier	60
3.11 Applications - Pricing & Risk Analysis	60
3.12 Applications - Variational Heuristics for Combinatorial Optimization	62
3.13 Applications - Additional Applications.....	63
3.14 Data-Porting Overview	64
3.15 Data-Porting - Narrow Band and Sparse Quantum Fourier Transforms.....	67
4.0 RESULTS AND DISCUSSION.....	69
4.1 Algorithms - Variational Quantum k-Eigensolver and Monte Carlo Minimizer....	69
4.2 Algorithms - Improving Variational Quantum Optimization Using CVaR.....	70
4.3 Algorithms - Robust Optimization.....	71
4.4 Algorithms – Structured Random Walks and Quantum Walks	72
4.5 Algorithms - Active Learning-Quantum Amplitude Estimation	76
4.6 Algorithms - Topological Data Analysis	80
4.7 Algorithms - Betti Number by Chebyshev Approximation.....	86
4.8 Applications - Topological Classifier	87
4.9 Applications - Pricing & Risk Analysis.....	89
4.10 Applications - Variational Heuristics for Combinatorial Optimization	90

4.11	Data-porting - Approximate Quantum Fourier Transform	91
4.12	Data porting - Narrow Band and Sparse Quantum Fourier Transforms	94
5.0	CONCLUSIONS.....	96
5.1	Recommendations.....	97
6.0	REFERENCES	99
	ABBREVIATIONS AND ACRONYMS	103

LIST OF FIGURES

Figure 1. 3 rd degree tensor unique symmetries	10
Figure 2. Variational Quantum k-Eigensolver by meta-level optimization.....	11
Figure 3. Quantum Variational High Order Moment algorithm flow chart	12
Figure 4. The circuit defining the quantum states explored.....	13
Figure 5. A circuit yielding an eigenstate of $\mathcal{H}$	13
Figure 6. CVaR as a function of θ	16
Figure 7. Probability distribution for a biased quantum walker.	43
Figure 8. Probability distribution for an unbiased quantum (blue) and classical (red) random walker.....	43
Figure 9. Illustrative example of clique detection using classical random walk	46
Figure 10. Illustrative example of clique detection using quantum walk	46
Figure 11. Rejection Filter Quantum Phase Estimation circuit	49
Figure 12. Boundary map	56
Figure 13. Various shapes and their respective Betti numbers	57
Figure 14. Barcode representing persistent homology across scales.....	58
Figure 15. Amplitude estimation circuit.	61
Figure 16. Illustrative example of a given demand distribution and the safety stock value that leads to a service level of 95%, i.e. a stock-out probability of 5%.	64
Figure 17. a) QFT circuit description b) Application of the A gate (Hadamard) c) Application of the entangling phase gates θ_{ij}	65
Figure 18. Top: the standard (full) QFT circuit. Bottom: the Mean Field decimated (Approximated QFT) circuit with approximation level of $m=2$	67
Figure 19. Eigenvalue error as a function of the Hamiltonian density assessment of the Variational Quantum k-Eigensolver with periodical penalty parameter increase (green) and without (purple)	70
Figure 20. CVaR optimization on a portfolio optimization problem, executed on IBM's quantum hardware.....	71
Figure 21. A sample of a graph.....	75
Figure 22. Sparsity plot.....	76
Figure 23. Convergence of the leading eigenvalue.....	76
Figure 24. Mean eigenvalue error vs iteration; One curve for each of the two eigenvalues.	79
Figure 25: (left) Exact probabilities of different estimated values, where the maximum probability corresponds to the result of the amplitude estimation algorithm. (right) Error in probability if inverse QFT is replaced by inverse Approximate QFT.	90
Figure 26. Approximated QFT with varying degree of approximation m	91
Figure 27. Error of approximate QFT, low frequency signal	92
Figure 28. Error of approximate QFT, medium frequency signal	93
Figure 29. Error of approximate QFT, high frequency signal	93

1.0 SUMMARY

Quantum computing holds great promise to provide asymptotic computational advantage. A necessary (yet not sufficient) condition for a quantum algorithm to be efficient is to make purposeful use of unique quantum properties (superposition, entanglement, uncertainty). Real-world problems introduce unprecedented levels of complexity. The primary goal of this fundamental research project was to conceive new advantageous quantum algorithms while avoiding the making of unrealistic assumptions.

From a core algorithmic perspective, we have conceived and developed the following algorithms.

Variational Quantum k-Eigensolver - The mathematical formulation of the algorithm was established, and an implementation of which in IBM Quantum Computing development environment has been concluded. The algorithm has been tested numerically. A key innovation of the algorithm is the search of k eigenpairs, by reformulating the variational optimization problem, as a meta-level optimization problem, where both the eigenvalue and its variational form are searched simultaneously.

Variational Quantum Monte Carlo Minimizer - Mathematical formulation of the algorithm was established, followed by implementation using the IBM Quantum Computing development environment and experimentation on IBM Q quantum computers. The central idea revolves around formulation of hybrid quantum-classical variational framework for computation of high order moments. Naïve design of such framework, computation of the p th moment entails polynomial increase $\mathcal{O}(d^p)$ in the number of variational terms to be computed by the quantum device, where d is the number of tensor product terms in the Hamiltonian representation. A formulation that builds upon matrix function construct, avoid such computational overhead.

Robust Optimization – Robust Optimization seems like a natural candidate for quantum computation. Given the decision, the probability distribution characterizing a utility function conveys the worst possible distribution among a possibly infinite set of distributions. We assume that both the vector of decisions x and the probability distribution $\xi \in D(\bar{\xi})$ can be obtained by parametrized unitaries. This formulation entails a saddle point problem, for which classical algorithms exist. The vector of decisions $x = U(\theta)|0\rangle$ can be encoded variationally. Vectors in which each element is 0 or 1 with given probability can be encoded with a single layer of Y rotations; changing the rotation angle directly determines the probability of a 0 or 1. Correlation between vector elements can be introduced using two-qubit entangling gates.

Structured Markov Processes - The computational complexity of many different forms of decision making under uncertainty problems limit the degree to which these fundamental problems can be solved in classical computing environments. This is particularly true for general forms of decision making under uncertainty problems that involve Markov processes as the stochastic modeling framework for the system under uncertainty within which decisions must be made. We have identified and investigated a broad spectrum of

such problem formulations that we believe are fertile for developing different types of quantum algorithms that have the potential to provide significant improvements in computational complexity. One reason for this is the wide usage of fast Fourier transforms as the basis of the most advanced and efficient classical computing algorithms. Another reason for this is the strong probabilistic properties and structures that we believe may enable us to develop even more efficient quantum algorithms from first principles. In addition to identifying and researching this class of computational problems, we have started to pursue both directions for developing quantum algorithms that drive significant gaps between classical and quantum computing environments.

Quantum Walk - The quantum walk is defined in the Hilbert space constituting a discrete position state $\mathcal{H}_s$ and a coin flip space, $\mathcal{H}_c$, of size two, in representing transition actions $\mathcal{H}_s \otimes \mathcal{H}_c$. Thus, at every one-time step a coin and a shift operator perform the evolution of the quantum random walk $U = S(I \otimes C)$. The unique virtue of quantum walks, in contrast with classical random walks, is the possibility of prescription of a super-positioned coin, which effectively enables the exploration of multiple paths simultaneously. The natural choice of a Hadamard gate as a super-positing coin, one can consider various other options (balanced or imbalanced) which affects the walker's dynamics. In section §3.6, we describe our preliminary design of a quantum walk algorithm and the implications of different choices of coin operators.

Quantum Walk-Based Clique Detection – As part of our continuing research in the area of random walks within the context of quantum algorithms, we extended our analysis and our development of algorithms related to quantum walks and in particular quantum walks for community clique detection. We expand our original definition of the space of the quantum walk on a line to an arbitrary undirected graph by using different position and coin spaces, considering various coins including those related to the Hadamard coin, Fourier coin, and Grover coin. For clique detection, the classical graph is encoded into a quantum object by a preparation circuit. The quantum random walk circuit performs the clique analysis by repeated applications of the walk operator (combination of the coin operator and step operator). The measurement circuit passes the quantum information from the qubits to the classical world for analysis. This quantum computational approach can lead to polynomial computational speedup over the classical analogs.

Structured Random Walks - In addition to our continuing research in the area of quantum walks (above), there is another general and important class of random walks that are of interest, namely those with certain structural properties. The best-known classical algorithm for computing the equilibrium distribution of such structured random walks is called cyclic reduction, whose computational complexity is dominated by the need to solve multiple systems of linear equations in each iteration of the cyclic reduction algorithm adapted to structured random walks. In section §3.5, we describe our initial design of a quantum algorithm for computing the equilibrium distribution of structured random walks that provides an exponential speedup in the size of the linear systems with respect to these computational complexity bottlenecks in each iteration of the cyclic reduction algorithm. In section 4.4, we further show that our quantum algorithms for structured random walks provide significant speedup over the corresponding algorithms for quantum walks. We also

note that cyclic reduction is at the core of a number of important numerical methods that go far beyond our interests here on structured random walks, and thus the quantum algorithm devised herein can be exploited to address a much larger class of numerical problems.

Active Learning-Quantum Phase Estimation – A variational Quantum Phase Estimation (QPE) approach that combines ideas associated with Rejection Filtering and neural network inference was considered in this study. The approach allows almost continuous steering between Variational Quantum Eigensolver (VQE) and QPE, based upon the computational resources available. The merit of using a hybrid quantum classical approach is that it utilizes the strength of each domain to obtain the best tradeoff between the error ϵ and the computational complexity. A mathematical formulation of the approach has been developed and a prototype algorithm was implemented.

Quantum Expectation Estimation - In keeping with the Active Learning mission of combining VQE with more sophisticated versions of amplitude estimation, we report a novel no-collapse approach to Quantum Amplitude Estimation, which together with Knill's real and complex extraction techniques, constitutes a novel Quantum Expectation Estimation Algorithm.

Topological Data Analysis - Among the appealing application areas of Quantum Computing, for which substantial (up to exponential) acceleration can be considered, we find Topological Data Analysis, and in particular, the field of Persistent Homology. Tracking how Betti numbers change as function of the scale ϵ reveals how topological features come into existence and disappear as the data is analyzed at different ϵ . A topological feature that persists over many length scales can be identified with a 'true' feature of the structure: Persistent Homology. In section §3.8 we provide the theoretical foundations and cover topics such as boundary maps, chain complex, homology groups and complex filtrations and BarCodes. Previously, we reported significant implementation details, specifically the design and implementation of the controlled-Permutation operation. This period, we developed a novel fast quantum algorithm to compute the Betti numbers based on Chebyshev polynomial approximation. The method only requires computing the matrix moments, and does not require computing the eigen-decomposition (using QPE) of the matrix. We present many numerical results that show that the new method is very efficient in estimating the Betti numbers.

Another thread of research work revolved around identification of application areas where quantum computing can offer viable advantage. Specifically, we have explored the applications listed below.

Topological Classifiers - One of the applications of Topological data analysis and persistent homology, is in machine learning, for classification problems. We can use the topological features of a given data for training classifiers, which we call the topological classifiers. The idea is that the Betti number of the points at different scale form the persistence barcode. We can use these Betti numbers at various scales ϵ as the (topological) input features for the classifiers. We show how these ideas can be used for digit

classification. In particular, we compute the Betti numbers for the 10 digits (0-9) at different scale ϵ , and show how we can distinguish the digits using just the topological features.

Pricing and Risk Analysis - Pricing and risk management play a central role in the financial system and represent some of the computationally most demanding problems in this area. This is due to the relatively slow convergence of classical Monte Carlo simulations, which often lead to massively parallelized overnight simulations. Using quantum amplitude estimation, it is possible to build quantum algorithms that lead to a quadratic speedup over classical Monte Carlo simulation. This could reduce millions of classical samples to thousands of quantum samples, potentially reducing overnight simulations to near real-time applications. This might have a significant impact to the financial system, where even marginal speed-ups can lead to huge benefits. The studied quantum algorithms are not only applicable to financial applications, for instance, it is straight-forward to apply them to problems inventory management as well.

Variational Heuristic for combinatorial optimization -. Variational approaches have been applied to a variety of fields, e.g., chemistry, machine learning, optimization. Typically, the utility function is a Hamiltonian encoding the total energy of the system, to be minimized. This directly relates to an optimization context: the same idea can be applied to classical combinatorial optimization problems, provided that we can construct a Hamiltonian encoding the objective function of the optimization problem. In our research we employ classical derivative-free optimization methods to seek for good solutions for these problems and sheds some lights on limitations that should be overcome to increase the effectiveness of the variational approach.

Lastly, we have dedicated our attention towards devising methods to address the fundamental data porting problem:

Sparse and Narrow Banded Quantum Fourier Transform – We extend beyond approximated QFT and propose a new sparse QFT (SQFT) algorithm. The construct of the algorithm is of appeal for understanding and implementing QFT on near term devices. We report a novel reformulation and application of the AQFT for sparse signals. We extended the algorithm to narrow-band signals and improve the complexity by appealing to newer parallelized versions.

Approximated Quantum Fourier Transform - Quantum Fourier transforms (QFT) are the key ingredient in a multitude of quantum algorithms. For most potential applications the QFT complexity is high and such, potential applications of approximate schemes that decrease the complexity can have tremendous impact.

We have implemented and thoroughly tested a mean field approximate quantum Fourier transform (AQFT) in python on several different popular frameworks: qiskit, qutip and pyquil. This led to a set of preliminaries benchmarking results presented in the main text. We have shown that the error in the measured wave function is negligible compared with an almost linear complexity (contrary to a cubic for the full QFT).

2.0 INTRODUCTION

The IBM Quantum Computing Algorithms fundamental research project focused on the challenge of design of quantum computing algorithms viable for near-term noisy quantum systems. An emphasis was set on design of algorithms that avoid taking unrealistic assumptions while offering theoretical rigor regarding the expected computational advantage. There were three major development initiatives listed below:

- Algorithmic design, implementation and testing
- Applications
- Data Porting

Each of the threads above, involved several tasks. The algorithmic design activity involved:

- Task 1.1 - Derivation and design of an integrated quantum variational expectation estimator-based eigenvalue and Monte Carlo minimization algorithm
- Task 1.2 - Algorithm development using IBM QISKit
- Task 1.3 - Algorithm testing
- Task 1.4 - Explore performance of variational algorithms for combinatorial optimization
- Task 1.5 - Null-space variational eigen solver formulation and assessment
- Task 1.6 - Develop an active learning β -QPE algorithm that combines the strengths of Variational Quantum Eigensolver and Quantum Phase Estimation
- Task 1.7 - Conduct an assessment of the active learning β -QPE algorithm
- Task 1.8 - Refine active learning β -QPE algorithm
- Task 1.9 - Conduct an analysis of the Approximate Quantum Fourier Transform algorithm
- Task 1.10 - Refine the Approximate Quantum Fourier Transform algorithm

The application scoping activity included exploration of the following domain problems:

- Task 2.1 - Scoping realistic problems of relevance to the AFRL, representable in the integrated quantum Monte Carlo framework
- Task 2.2 - Performance assessment conducted upon an actual (non-fault tolerant) quantum computer
- Task 2.3 - Design Quantum Walk-based clique detection algorithm
- Task 2.4 - Test Quantum Walk-based clique detection algorithms

Lastly, as part of the data porting thrust, the following activities were pursued:

- Task 3.1 - Exploration of mathematical formulations for sub-linear (in particular sketching based) classical (implicit or explicit) to quantum data mapping
- Task 3.2 - Technical report covering theoretical and implementation details
- Task 3.3 - Algorithm demonstration on an actual Quantum Computer
- Task 3.4 - Initial characterization of topological features (Betti numbers), based on the β -QPE algorithm
- Task 3.5 - Refine characterization of topological features (Betti numbers)
- Task 3.6 - Initial topological classifier, based upon the topological characterization

- Task 3.7 - Refine topological classifier algorithm

A summary of the activities each thread involved are described in detail in the following sections.

3.0 METHODS, ASSUMPTIONS, AND PROCEDURES

3.1 Algorithms - Variational Quantum k-Eigsolver and Monte Carlo Minimizer

3.1.1 Problem Definition.

There are several related problems that we can define with similar setup. Given an Hermitian operator $\mathcal{H} \in \mathbb{C}^{2^n \times 2^n}$, find:

- Eigen value minimization $\lambda_{\min} \mathcal{H} = \min_{|\psi\rangle} \frac{\langle \psi | \mathcal{H} | \psi \rangle}{\langle \psi | \psi \rangle}$
- Eigen value k-decomposition $\{\lambda_1, \lambda_2, \dots, \lambda_k\}$ for $\{\psi_1, \psi_2, \dots, \psi_k\}$
- Monte Carlo variance minimizer $V_{\min}(\mathcal{H}) = \min_{|\psi\rangle} \frac{\langle \psi | \mathcal{H}^2 | \psi \rangle}{\langle \psi | \psi \rangle} - \frac{\langle \psi | \mathcal{H} | \psi \rangle^2}{\langle \psi | \psi \rangle}$

Hereafter we shall assume that the state is normalized $\langle \psi | \psi \rangle = 1$.

3.1.2 The Variational Quantum Eigsolver.

The eigenvalue problem can be casted as a variational problem on Rayleigh–Ritz quotient

$$\lambda_{\min} = \min_{|\psi(\theta)\rangle} \min_{\psi} \frac{\langle \psi(\theta) | \mathcal{H} | \psi(\theta) \rangle}{\langle \psi(\theta) | \psi(\theta) \rangle} \quad (1)$$

such that the eigenvector $|\psi\rangle$ corresponds to the lowest eigenvalue. Through variation of the experimental parameters, θ , in the preparation of $|\psi(\theta)\rangle$ and computation of the Rayleigh-Ritz quotient, the minimum can be searched. At the termination of the algorithm, a simple prescription for the reconstruction of the eigenvector is stored in the final set of experimental parameters, θ , that define $|\psi(\theta)\rangle$.

The Quantum Variational Eigsolver (QVE) [1] algorithm hybridizes quantum (expectation) and classical (variational optimization) architectures. It lends itself natively to a parallel quantum computation implementation and relieves considerably the requirement for coherent evolution of the quantum state. To its disadvantage, scaling in the total number of operations (sampling in this case), as a function of the desired precision is linearly worse compared to QPE ($\mathcal{O}(\epsilon^{-2})$ vs. $\mathcal{O}(\epsilon^{-1})$) [2]. QVE will add a polynomial number of repetitions with respect to QPE. However, the bottom-line is a dramatic reduction in the coherence-time requirement.

3.1.3 Prior Art.

The term Quantum Monte Carlo has been used in several contexts, including classical computing algorithms inspired by quantum physics notions. In our context of §3.1.1, one can readily see that the Monte Carlo and the eigenvalue minimization problems are closely associated. Quantum approaches to finding eigenvalues have previously relied on the Quantum Phase Estimation (QPE) algorithm. The QPE algorithm requires a $\mathcal{O}(\epsilon^{-1})$ scaling in quantum resources to obtain an estimate with precision ϵ . In QPE, one starts from the

preparation of a quantum state $|\psi\rangle$, that possess a large (i.e., not exponentially small in the size of the problem) component of the ground state of a Hermitian operator $\mathcal{H}$ (and respectively $\mathcal{H}^2$ for variance computation). After the preparation of the state $|\psi\rangle$, the algorithm determines the ground eigenvalue λ through implementation of the operator $\mathcal{H}$ controlled by the state of some ancillary qubits. This last step of the algorithm can be very costly, involving a sheer amount of quantum gates, especially affecting small-size quantum computers prone to errors.

3.1.4 1st Moment Expectation.

Assume a Hermitian operator (Hamiltonian), $\mathcal{H} \in \mathbb{C}^N$, is decomposable into a sum of tensor products, $q_i = \sigma_{\pi_i(1)}^p \otimes \dots \otimes \sigma_{\pi_i(n)}^p$, of Pauli and identity operators σ^p

$$\mathcal{H} = \sum_{i=1}^d w_i q_i \quad (2)$$

where $w_i \in \mathbb{R}$, $p \in \{I, X, Y, Z\}$ and d is the number of terms in the Hamiltonian (polynomial with respect to the system size).

Assume that we have prepared a state $|\psi(\theta)\rangle$ whose variational form depends upon the parameters θ . Due to linearity of quantum measurement, the expectation of $\mathcal{H}$ can be expressed as:

$$\langle \psi(\theta) | \mathcal{H} | \psi(\theta) \rangle = \langle \psi(\theta) | \sum_{i=1}^d w_i q_i | \psi(\theta) \rangle = \sum_{i=1}^d w_i \langle \psi(\theta) | q_i | \psi(\theta) \rangle \quad (3)$$

By the variational theorem, for all θ

$$\langle \psi(\theta) | \mathcal{H} | \psi(\theta) \rangle \geq \lambda_{min} \quad (4)$$

Thus, the optimal choice of θ to approximate the ground state is the choice that minimizes the expectation term. The expectation value of a tensor product of an arbitrary number of Pauli operators can be estimated by local measurement of each qubit. Assuming the measurements are performed in the Z basis, diagonalized basis measurements can be obtained by projections, T_p , to the Z space $T_p^\dagger \sigma^p T_p = \sigma^Z$.

$$T_X = H, T_Y = HS^{-1}, T_Z = I \quad (5)$$

Assuming the state $|\psi(\theta)\rangle$ can be articulated in the quantum computer for a prescribed circuit depth, the additional overhead to compute its expectation for each component of the Hamiltonian requires at most one additional layer of Pauli gates. Following projections to the Z basis, quantum measurements, $\{m_i^1, \dots, m_i^n\}$ are processed on a classical computer to form samples, s_i , for the expectation of the i^{th} component

$$s_i = (-1)^{\sum_j m_i^j} = (-1)^{m_i^1 \oplus m_i^2 \oplus \dots \oplus m_i^n}. \quad (6)$$

The sampling process is repeated per the required precision specification, ϵ , and the expectation components are taken to be the sample means

$$\langle \psi(\theta) | \mathcal{H} | \psi(\theta) \rangle = \langle \psi(\theta) | \sum_{i=1}^d w_i q_i | \psi(\theta) \rangle = \sum_{i=1}^d w_i \langle \psi(\theta) | q_i | \psi(\theta) \rangle = \sum_{i=1}^d w_i \mathbb{E}\{s_i\}. \quad (7)$$

Since the operators are normalized and finite-dimensional, their spectra are bounded. Each expectation component $w_i \langle \psi(\theta) | q_i | \psi(\theta) \rangle$ can be estimated to ϵ precision, at the cost of $\mathcal{O}(|w_{\max}|^2 d \epsilon^{-2})$ repetitions, where $w_{\max}$ stands for the maximum norm coefficient in the decomposition of the Hamiltonian. The coherence time to make a single measurement after preparing the state is $\mathcal{O}(1)$. Such independent measurements can be performed in parallel, incurring a constant cost in time.

Scaling in the total number of operations for eigenvalue estimation, as a function of the desired precision is linearly worse compared to QPE ($\mathcal{O}(\epsilon^{-2})$ vs. $\mathcal{O}(\epsilon^{-1})$). Yet, overall, dramatic reduction in coherence time requirement while maintaining an exponential speedup over the classical case.

If a quantum state is characterized by an exponentially large number of parameters, it cannot be prepared with a polynomial number of terms. The set of efficiently preparable states are therefore characterized by polynomially many parameters.

3.1.5 2nd Moment Expectation.

Similarly, the 2^{nd} moment (expectation of the square of the Hamiltonian) can be computed as a sum of expectations of products of the terms of the Hamiltonian

$$\langle \psi(\theta) | \mathcal{H}^2 | \psi(\theta) \rangle = \langle \psi(\theta) | \sum_{i,j} w_i w_j q_i q_j | \psi(\theta) \rangle = \sum_{i,j} w_i w_j \langle \psi(\theta) | q_i q_j | \psi(\theta) \rangle \quad (8)$$

This calculation requires expectation computation of cross-terms involving two operators from the set p , comprising the Pauli operators and identity, e.g. $\langle \psi(\theta) | \sigma^X \sigma^Y | \psi(\theta) \rangle$, followed by weighted summation according to the coefficients $w_i w_j$. Symmetric Pauli products are trivially the identity I_2 . Since $\sigma^X \sigma^Y = i\sigma^Z$, $\sigma^Y \sigma^Z = i\sigma^X$, $\sigma^X \sigma^Z = -i\sigma^Y$, each non-symmetric Pauli product is replaced by its respective single Pauli operator, and an appropriate Z-space projector, T_p is incorporated.

Equally, expectations of higher order moments can be computed. The incurred cost is associated with the dimension of the non-symmetric terms (grows factorially with the moment's order). For instance, a 3^{rd} degree moment, is a super-symmetric tensor, whose cardinality of non-symmetric moment factors is

$$\frac{n(n+1)(n+2)}{3!} \quad (9)$$

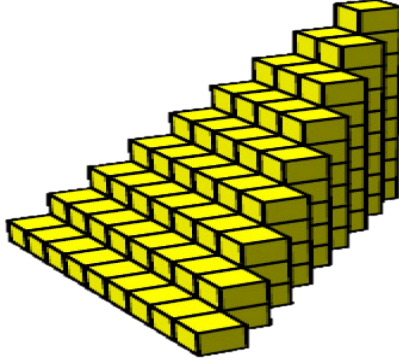


Figure 1. 3rd degree tensor unique symmetries

3.1.6 Meta-Level Search over Spectral Decompositions.

To estimate additional eigen-pairs (in addition to the one corresponding to the smallest eigenvalue), meta-level search is performed. The eigen-value condition can be articulated as:

$$\mathcal{H}|\psi(\theta)\rangle = \lambda|\psi(\theta)\rangle \Leftrightarrow \mathcal{H}|\psi(\theta)\rangle - \lambda|\psi(\theta)\rangle = 0 \quad (10)$$

In case we have already extracted several eigen-pairs $\{\hat{\lambda}_i\}$, the optimization reads

$$\min_{\lambda, \theta} \|(\mathcal{H} - \lambda I)|\psi(\theta)\rangle\|^2 \quad (11)$$

$$\text{s. t. } \lambda \neq \{\hat{\lambda}_i\}$$

which eludes to

$$\min_{\lambda, \theta} \langle \psi(\theta) | (\mathcal{H} - \lambda I)^\dagger (\mathcal{H} - \lambda I) | \psi(\theta) \rangle \quad (12)$$

$$\text{s. t. } \lambda \neq \{\hat{\lambda}_i\}$$

To extract the smallest eigen-pairs orderly, the following meta-level optimization problem is performed iteratively

$$\min_{\lambda, \theta} \langle \psi(\theta) | \mathcal{H}^2 - 2\lambda\mathcal{H} + \lambda^2 I | \psi(\theta) \rangle + \alpha f(\lambda) \quad (13)$$

$$\text{s. t. } \lambda \geq \{\hat{\lambda}_i\}, \alpha \geq 0$$

where f is a monotonically increasing function w.r.t. λ [3].

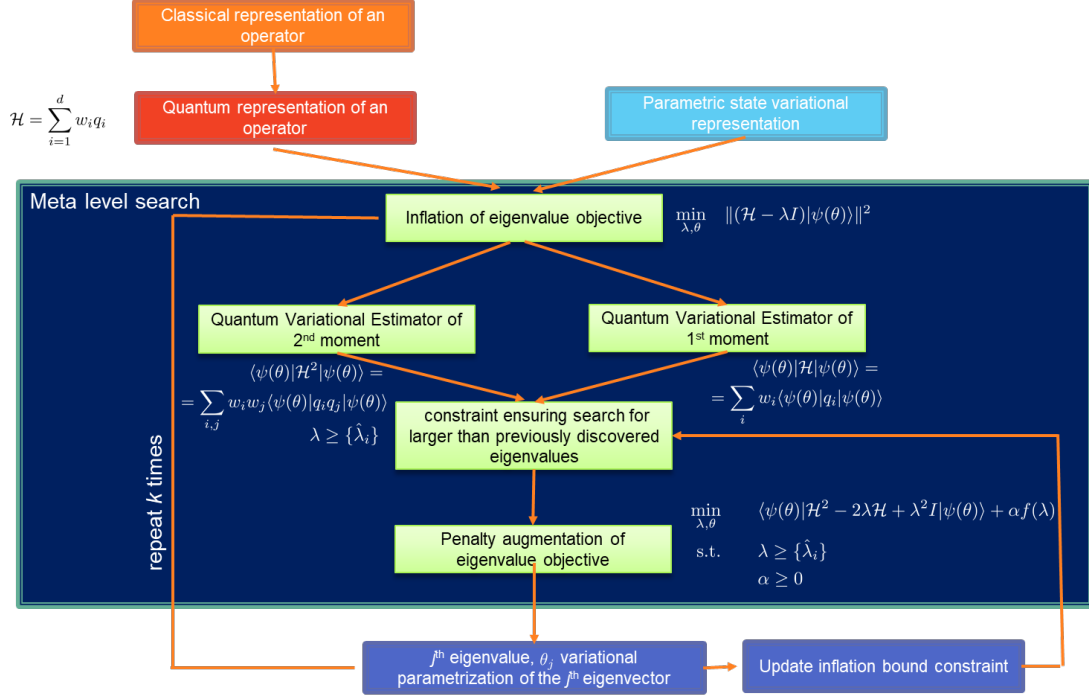


Figure 2. Variational Quantum k-Eigsolver by meta-level optimization

3.1.7 2nd Moment Speedup.

Let us define as $S = \sqrt{\mathcal{H} - \lambda I}$, a matrix square root (as a matrix function)

$$SS = \mathcal{H} - \lambda I \quad (14)$$

For $(\mathcal{H} - \lambda) = U^\dagger \Sigma U$, with eigenvalues matrix $\Sigma = \text{dia}(\lambda_1, \dots, \lambda_n)$, we have $S = U^\dagger \sqrt{\Sigma} U$, where $\sqrt{\Sigma}$ is a diagonal matrix with the square root of the eigenvalues on the diagonal $(\sqrt{\lambda_1}, \dots, \sqrt{\lambda_n})$. While eigenvalues of $\mathcal{H} - \lambda I$ are real for $\lambda \geq \lambda_0$, the eigenvalues of S can be complex in general.

$$\min_{\gamma, \theta} \langle \psi(\theta) | S^\dagger S^\dagger S S | \psi(\theta) \rangle + \alpha f(\gamma) \quad (15)$$

The decomposition problem is equivalent to $\mathcal{H} \geq 0$. Assuming $S = \sqrt{\mathcal{H} - \lambda I}$ (positive semi-definite), i.e. all its eigenvalues are real and non-negative, the squares root of which are all real and non-negative. The eigen space spanned by the root operator, S , is identical to the eigen space spanned by $\mathcal{H} - \lambda I$, where the eigenvalues themselves are the square of the ones of S . Compute all negative eigenvalues using $\mathcal{H}^2$ formulation. For all positive eigenvalues, directly resort to optimization of the problem

$$\min_{\gamma, \theta} \langle \psi(\theta) | \mathcal{H} - \gamma I | \psi(\theta) \rangle^2 + \alpha f(\gamma^2) \quad (16)$$

$$s.t. \gamma^2 \geq \gamma_i^2 \geq 0$$

where $\lambda = \gamma^2$. Otherwise, inflate (i.e. shift the eigen spectrum by adding identity to the term) and compensate for the shift in a retrospect. An $\mathcal{O}(d)$ speed-up eliminates the need to perform $\mathcal{O}(d^2)$ expectation for k-eigenpairs [4].

3.1.8 Hybrid Quantum-Classical Framework.

Either the 1st or 2nd or combination of both moments can be attained by a hybrid Quantum-Classical computational framework. The guiding principle in such collaborative computation, is to resort to quantum computation only for computations that cannot be performed classically efficiently. This way, we minimize the chances of being affected by the fidelity of the quantum device. In fact, other than state representation (which is mandatory for any quantum computation), the framework only requires application of a single additional gate for each expectation realization. Since d terms are required for each expectation sample, and for each, multiple realizations are required per the desired level of accuracy, it is beneficial (yet, not mandatory) to consider a parallel quantum processing computer. The classical computer aggregates the measured quantum realizations, performs modulo 2 summation, and sums up all sample contributions to estimate the expectation for a given variational parametrization θ^k . Further, optimization of the variational form is performed on a classical computer, to provide the quantum device(s) with the next trial function parametrization $\theta^{(k+1)}$. A high-level illustration of the proposed symbiotic architecture is given in Figure 2.

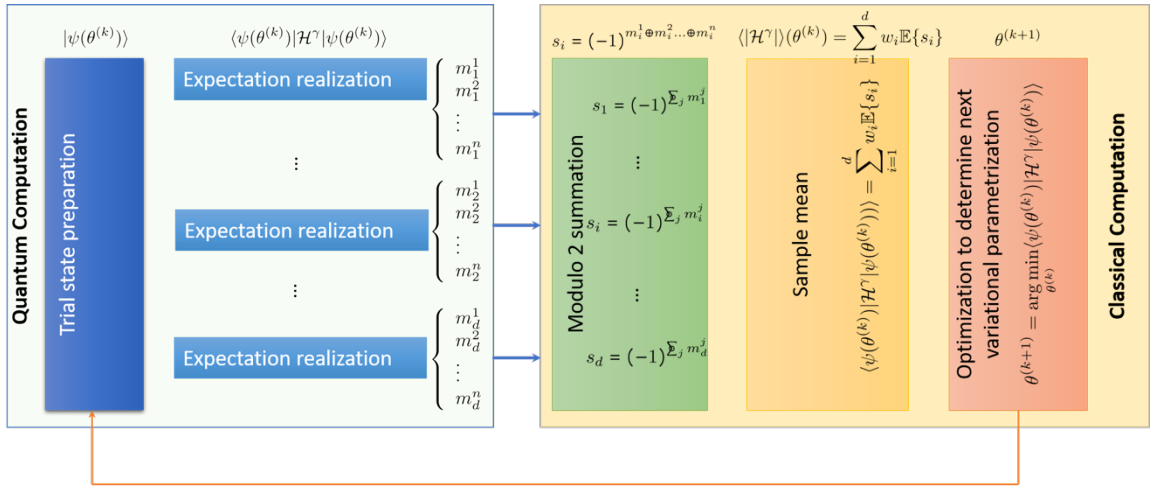


Figure 3. Quantum Variational High Order Moment algorithm flow chart

3.1.9 Proof of Concept.

We provide a brief illustration of the main idea for QVE. Suppose we have a 8×8 matrix:

$$\mathcal{H} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -2 & -2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & -2 & 0 & 0 & 0 & 0 & -2 & 0 \\ 0 & -2 & 0 & 0 & 0 & 0 & -2 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -2 & -2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \quad (17)$$

Since $\log_2 8 = 3$, we can attempt to determine the smallest eigenvalue of this matrix with only 3 qubits. $\mathcal{H}$ can be expressed as a weighted summation of products of Pauli operators as follows:

$$\mathcal{H} = \frac{1}{2}I \otimes I \otimes I - 1I \otimes X \otimes I + \frac{1}{2}Z \otimes I \otimes Z + 1Z \otimes X \otimes Z - 1X \otimes I \otimes X + \frac{1}{2}X \otimes X \otimes X - 1Y \otimes I \otimes Y - \frac{1}{2}Y \otimes X \otimes Y. \quad (18)$$

The circuit defining the quantum states explored is shown below. R_z and R_y represents rotations: $R_z(\theta) = e^{(-i \theta/2 Z)}$, $R_y(\theta) = e^{(-i \theta/2 Y)}$

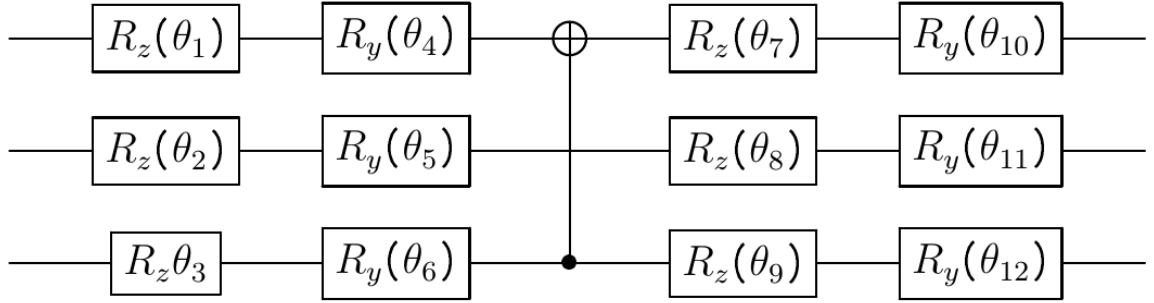


Figure 4. The circuit defining the quantum states explored.

Notice that this circuit has a single two-qubit gate, which is important in view of the coherence times of current-generation devices. All gates represented here are available on IBM's quantum devices, and the angles can be programmed via the IBM Quantum Experience. Let $|\psi\rangle$ be the state produced by the quantum circuit. A simplified version of the QVE, in which we consider only minimization of the first order moment, can be written as:

$$\min_{\theta_1, \dots, \theta_{12}} \langle \psi | \mathcal{H} | \psi \rangle. \quad (19)$$

It can be easily verified that an optimal solution to this problem is $\theta_1 = \theta_2 = \theta_9 = \theta_{12} = \pi$, $\theta_4 = \theta_5 = \frac{\pi}{2}$ with the remaining angles at zero. Such a solution is equivalent to the following circuit:

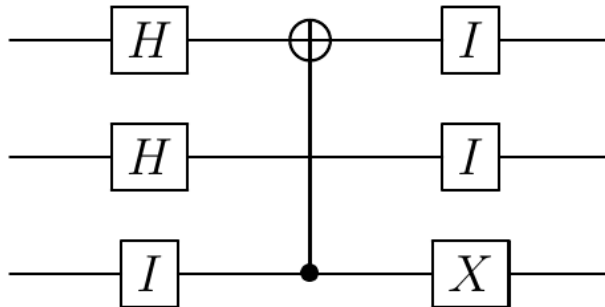


Figure 5. A circuit yielding an eigenstate of $\mathcal{H}$

This is the vector $(0, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, 0)^\top$, which is an eigenstate of $\mathcal{H}$ with eigenvalue -4 . Furthermore, -4 is the smallest eigenvalue of $\mathcal{H}$, as desired.

3.1.10 Orthogonal (null) subspace (variational deflation).

So far, we have attempted an inflation approach in which every stationary point was mapped to a local minimum. A complementary approach is to consider a deflated formulation that effectively attempts to search in the orthogonal sub-space to the space spanned by the eigenstates already found. Such an approach is similar in spirit to the one employed by sequential classical eigensolvers. That is, at every iteration one seeks for the smallest (or largest) eigenpair λ_k and $|\psi_k\rangle$ of a modified operator $\mathcal{H}_k$. Once the eigenpair is found, the rank 1 subspace spanned by the k th eigenstate is subtracted from the operator; this can be articulated recursively as:

$$\mathcal{H}_{k+1} = \mathcal{H}_k - \lambda_k |\psi_k\rangle\langle\psi_k| \quad (20)$$

Despite its intuitive appeal, the subtraction of rank 1 operators from the operator cannot be performed straightforwardly in quantum circuit terms. Instead, we can attempt to use this methodology by seeking eigenstates that are (approximately) orthogonal to previously found eigenstates. This requirement can be imposed by either penalty or constraint involving a term $\langle\psi_i|\psi_j\rangle$.

3.1.11 Formulation – Variational Deflation

Let $\lambda_i, \psi_i, \theta_i$ for $i = 1, \dots, k$ be the previously found eigenvalues, eigenstates and respective variational parameterizations. We solve for the $k + 1$ eigenpair as follows:

$$\min_{\theta} \langle 0 | U(\theta)^* \mathcal{H} U(\theta) | 0 \rangle + \alpha \sum_{i=1, \dots, k} \langle 0 | U(\theta_i)^* U(\theta) | 0 \rangle \quad (21)$$

Notice that this is an unconstrained formulation that attempts to penalize quantum states that are not orthogonal to previously found eigenstates. It is easy to verify that the penalty term does exactly that. However, the penalty term gives rise to a very nonconvex objective function, hence the resulting optimization problem is difficult to solve with existing optimization algorithms.

3.1.12 Formulation – Ensuring orthogonality at the input

An alternative formulation may be obtained by using the property that a unitary transformation maps orthogonal states into orthogonal states. Thus, orthogonality can be ensured at the input. Suppose we are looking to determine the first k eigenpairs. Then we can choose k orthogonal states (for example, k basis states) and determine a transformation that hopefully maps them to eigenstates. More specifically, let $|\phi_i\rangle$ be k orthogonal states

as mentioned above, k distinct computational basis states will do. Then we can solve the following problem:

$$\min_{\theta} \sum_{i=1, \dots, k} \alpha_i \langle \phi_i | U(\theta)^* \mathcal{H} U(\theta) | \phi_i \rangle \quad (22)$$

Under mild conditions (e.g., the α_i have to be nonincreasing) it can be shown that the global minimum for this problem is attained when $U(\theta)$ maps the initial states $|\phi_i\rangle$ into the first k eigenstates. However, the difficulty lies in the fact that we are attempting to find a unitary that implements a potentially complicated map, because it essentially encodes all the eigenstates at once. Thus, the variational form must have high expressive power, and the corresponding classical optimization problem may be very difficult to solve.

3.2 Algorithms - Improving Variational Quantum Optimization Using CVaR

It is widely believed that quantum computers cannot solve NP-hard combinatorial optimization problems in polynomial time (i.e., NP is not contained in BQP), but there is significant effort toward designing quantum algorithms that could be practically useful by quickly finding near-optimal solutions to these hard problems. Among these algorithms, two candidates are more likely to be efficiently implementable on Noisy Intermediate-Scale Quantum (NISQ) devices, as already discussed in this report: the Variational Quantum Eigensolver (VQE), and the Quantum Approximate Optimization Algorithm (QAOA). Both VQE and QAOA use a parametrized quantum circuit $U(\theta)$ (also called variational form) to generate trial wave functions $|\psi(\theta)\rangle = U(\theta)|0\rangle$, guided by a classical optimization algorithm that aims to solve:

$$\langle \psi(\theta) | H | \psi(\theta) \rangle. \quad (23)$$

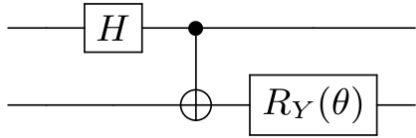
This expression encodes the total energy of a system through its Hamiltonian H , and more specifically its expected value. There is a well-known transformation to map Combinatorial Optimization (CO) problems into a Hamiltonian. Empirical evaluations (see the section “Performance of Hybrid Quantum/Classical Variational Heuristics for Combinatorial Optimization”) indicate that VQE's performance suffers from some key weaknesses in the context of CO problems, and there is significant room for improvement. At the same time, the literature on QAOA paints a mixed picture: while some work concludes that QAOA has promise, other papers indicate that it may not perform better than classical algorithms.

We propose a modification to the problem that is given to the classical optimization algorithm: rather than minimizing the expected value $\langle \psi(\theta) | H | \psi(\theta) \rangle$, we minimize its Conditional Value-at-Risk (CVaR) -- a measure that takes into account only the tail of the probability distribution and is widely used in finance. We argue that CVaR more closely matches the practical goal of heuristic solution methods for CO problems. We provide an analytical and numerical study, showing that the proposed methodology significantly improves performance and robustness of VQE and QAOA in the context of CO. As a byproduct of this analysis, we find that for low-depth circuits it is important to use variational forms with a sufficient number of parameters, e.g., at least linear in the number of qubits. QAOA for fixed (small) depth does not satisfy this requirement, and produces “flat” quantum states that yield low probability of sampling the optimal solution. Thus, not only does our work introduce a simple way to improve the performance of the two most

prominent candidates for CO on NISQ devices, but it also sheds some light on their behavior. While we do not claim that this brings their performance on par with state-of-the-art classical heuristics, our results indicate a marked improvement as compared to what is proposed in the literature. The numerical experiments discussed in our work are implemented in the open-source library QISKit, and executed on classical quantum simulators as well as IBM's quantum hardware.

3.2.1 Motivation of CVaR optimization.

We provide a simple example to justify the benefits of CVaR. Suppose we are optimizing the Hamiltonian $H = \text{diag}(0, 1, 1, 2)$, using the variational form indicated here on the left. It is not difficult to carry out the calculations and verify that the expected value of the Hamiltonian is constant, and does not depend on θ . On the other hand, the objective function of CVaR optimization is given in the second figure on the left, and it can be seen that the objective function depends on θ and on the value of α , that determines the level of risk for which the CVaR is computed. Notice that for certain values of α the objective function clearly indicates that smaller values of θ are better. Indeed, the probability of



sampling the optimal solution (which is 00) for this specific Hamiltonian and variational form is maximized when $\theta = 0$. Thus, in this example CVaR optimization helps determine the optimal solution to the problem.

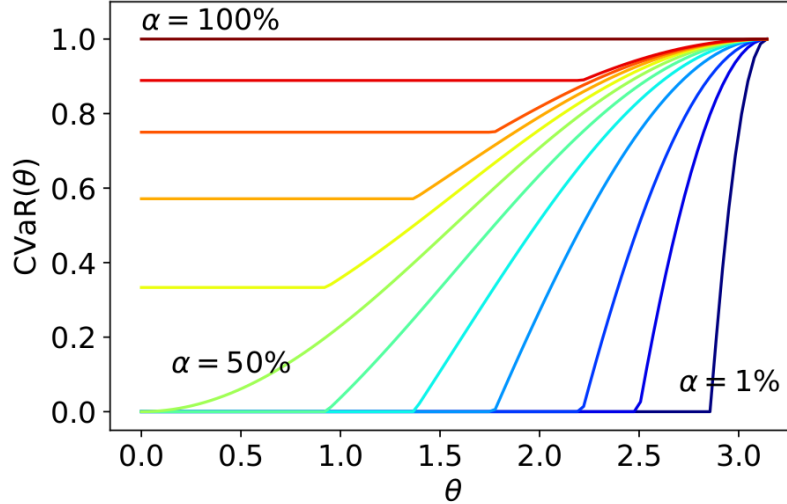


Figure 6. CVaR as a function of θ .

3.3 Algorithms - Robust Optimization

3.3.1 Problem Definition.

Robust optimization is a cornerstone of optimization under uncertainty. In robust optimization, one tries to determine decisions that yield an optimum of a stochastic utility

function under a worst-case scenario. More specifically, it is assumed that the probability distribution characterizing the utility function is chosen to be the worst possible distribution (given the decisions) among a possibly infinite set of distributions. The problem is formalized as follows. Let $f(x, \xi)$ be the utility function, which depends on a vector $x \in R^{2^q}$ of decisions and a probability mass function ξ . Let P be a set of probability mass functions over the outcome space $\{0, \dots, 2^p - 1\}$. Our goal is to determine:

$$\min_x \max_{\{\xi \in P\}} f(x, \xi) \quad (24)$$

Problems of the form (24) arise in many contexts, such as distributionally robust learning and optimization. The difficulty of solving problem (24) depends on the set of probability distributions P , among other things. In the context of quantum computing, it is natural to assume that a distribution $\xi \in P$ is encoded via the amplitudes of a quantum state $|\xi\rangle = \sum_{j=1}^{2^p-1} \alpha_j |j\rangle$, so that $\Pr(\xi = j) = |\alpha_j|^2$. If no restrictions on the form of $\xi \in P$ is given, then (24) becomes:

$$\min_x \max_{|\xi\rangle} f(x, |\xi\rangle), \quad (25)$$

whose difficulty depends on the form of f and may not be amenable to a systematic treatment in general. Instead, we study a version of the problem in which P is defined as the set of distributions within a given distance (according to some suitable metric) from a given *base distribution*. From a practical point of view, this is motivated by the commonly used approach of defining an empirical distribution, and optimizing while ensuring that the solution performs well for all distributions that differ “not too much” from the empirical distribution (e.g., because the data used for the empirical distributions is not fully accurate). Calling $\bar{\xi}$ the base distribution, we therefore define:

$$D(\bar{\xi}) := \{ \xi \in P : d(\xi, \bar{\xi}) \leq \rho \}, \quad (26)$$

where $d(\cdot, \cdot)$ is some suitable measure. The optimization problem (24) can thus be rephrased as:

$$\min_x \max_{\xi \in D(\bar{\xi})} f(x, \xi). \quad (27)$$

Given the framework (27), the crucial research question is to understand which measures d are practically and theoretically interesting, while leading to tractable problems. A possible approach to investigate this question is to assume that both the vector of decisions x and the probability distribution $\xi \in D(\bar{\xi})$ can be obtained by parametrized unitaries $U(\theta), V(\sigma)$ respectively, and in particular they satisfy:

$$|x\rangle = U(\theta)|0\rangle \quad (28)$$

$$|\xi\rangle = V(\sigma)|\bar{\xi}\rangle. \quad (29)$$

In the above expression, θ and σ are vectors of continuous parameters of suitable dimension. Under this assumption, (27) becomes:

$$\min_{\theta} \max_{\sigma} f(U(\theta)|0\rangle, V(\sigma)|\bar{\xi}\rangle). \quad (30)$$

This is now a saddle point problem, for which classical algorithms exist. In other words, assuming a certain form for x and ξ we can formulate problem (27) into a saddle point problem that we know how to solve. Of course, we still have to define the unitaries $U(\theta)$ and $V(\sigma)$.

To encode the vector of decisions $x = U(\theta)|0\rangle$ we can use commonly used variational approaches, which have been popularized by recent literature. Vectors in which each element is 0 or 1 with given probability can be encoded with a single layer of Y rotations; changing the rotation angle directly determines the probability of a 0 or 1. Correlation between vector elements can be introduced using two-qubit entangling gates.

3.4 Algorithms - Structured Markov Processes

At the core of many decision making under uncertainty problems are multidimensional Markov processes playing the role of the stochastic model of the system under uncertainty for which the decisions need to be made. Such multidimensional Markov processes, as well as such decision making under uncertainty problems involving specific parameters embedded in the multidimensional Markov process, arise in many fields of science, engineering and business. Of particular theoretical and practical importance are cases in which the Markov process is defined on the nonnegative quarter-plane; this two-dimensional case often occurs directly or through decomposition of higher dimensional Markov processes. Of additional theoretical and practical importance are general classes of Markov processes whose structural properties enable efficient computational solutions. Although both discrete-time and continuous-time Markov processes are of interest, we shall focus in this section on the former noting that the latter can be addressed in an analogous manner.

Consider a discrete-time (time-homogeneous) Markov process $\{X(n); n \in \mathbb{Z}_+\}$ defined on the state space $\mathbb{S} := \{(i, j): i \in \mathbb{Z}_+, j \in [M]\}$, with $\mathbb{Z}_+$ denoting the nonnegative integers and $[n] := \{0, 1, \dots, n\}$. The variable M can be infinite for many of the structural properties and results that follow. We define the set $\ell(i) := \{(i, j): j \in [M]\}$ to be *level* $i \in \mathbb{Z}_+$. Let $x_{i,j} \subset \mathbb{S}$ be a lexicographic ordering of the elements of $\ell(i), i \in \mathbb{Z}_+, j \in [M]$. We then define

$$\pi \equiv (\pi_0, \pi_1, \pi_2, \dots), \quad (31)$$

where

$$\pi_i \equiv (\pi(x_{i,0}), \pi(x_{i,1}), \dots, \pi(x_{i,M})), \quad (32)$$

and

$$\pi(x_{i,j}) \equiv \lim_{n \rightarrow \infty} \mathbb{P}[X(n) = x_{i,j}]. \quad (33)$$

The limiting probability vector π is the stationary distribution for the Markov process. Assuming the process to be ergodic, this invariant probability vector π is uniquely determined by solving the global balance equations

$$\pi P = P \quad (34)$$

together with the normalizing constraint

$$\pi e = 1, \quad (35)$$

where P is the transition probability matrix for the process and e is the column vector of appropriate dimension containing all ones. Various functions of the stationary distribution π can then be computed to evaluate the system of interest under a given set of decision variables.

In general, the computational complexity of solving for the stationary distribution can be prohibitive, and thus one must exploit the structural properties of the Markov process $\{X(n); n \in \mathbb{Z}_+\}$ to realize efficient solution approaches. Specifically, we consider the structural properties of the transition probability matrix P organized in the same order as the elements of the stationary vector π . Of particular interest herein are the general classes of structured Markov processes consisting of so-called $M/G/1$ -type processes, $G/M/1$ -type processes, and quasi-birth-death (QBD) processes, which respectively have the properties of skip-free to the left, skip-free to the right, and skip-free to the left and right. We therefore present in turn some background information and an analysis for each of these types of structured Markov processes. Throughout we shall assume that the Markov process is ergodic, and define $M_+ := M + 1$.

We note that random walks represent specific instances of QBD processes, as illustrated in §3.4.4; such instances of structured Markov processes are of interest and are being investigated from a quantum perspective. We further note that, while much of the results which follow concern stationary behaviors and properties of structured Markov processes, the transient behaviors and properties of these Markov processes also are of interest and are being investigated from a quantum perspective, where some of the subsequent results can be exploited to this end.

3.4.1 $M/G/1$ -Type Markov Processes.

Consider the discrete-time Markov process $\{X(n); n \in \mathbb{Z}_+\}$ whose transition probability matrix P has the structure given by

$$P = \begin{bmatrix} B_0 & B_1 & B_2 & B_3 & B_4 & \cdots \\ A_{-1} & A_0 & A_1 & A_2 & A_3 & \cdots \\ 0 & A_{-1} & A_0 & A_1 & A_2 & \ddots \\ 0 & 0 & A_{-1} & A_0 & A_1 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (36)$$

where B_i , for $i \geq 0$, and A_i , for $i \geq -1$, are nonnegative matrices in $\mathbb{R}^{M_+ \times M_+}$ such that $\sum_{i=0}^{\infty} B_i$ and $\sum_{i=-1}^{\infty} A_i$ are stochastic. Observe that the transition probability matrix P is block upper Hessenberg and, with the exception of the first block row, is block Toeplitz.

The elements of the B_i and A_i matrices over the two-dimensional state space $\mathbb{S}$ have the following probabilistic interpretation:

$$\begin{aligned} (B_k)_{j,j'} &= \mathbb{P}[X(n+1) = (k, j') | X(n) = (0, j)], \quad \forall k \geq 0, j, j' \in [M], \\ (A_k)_{j,j'} &= \mathbb{P}[X(n+1) = (i+k, j') | X(n) = (i, j)], \quad \forall k \geq -1, i \geq 1, j, j' \in [M]. \end{aligned} \quad (37)$$

Namely, B_0 provides the probabilities of transitions within the boundary $\ell(0)$, $B_{k \geq 1}$ provides the probabilities of transitions to the right by k levels from the boundary $\ell(0)$, A_0 provides the probabilities of transitions within any $\ell(i)$, $i \geq 1$, $A_{k \geq 1}$ provides the probabilities of transitions to the right by k levels from any $\ell(i)$, $i \geq 1$, and A_{-1} provides the probabilities of transitions to the left by 1 level from any $\ell(i)$, $i \geq 1$. The key to solving for the stationary distribution is the matrix equation

$$Z = \sum_{i=-1}^{\infty} A_i Z^{i+1}, \quad (38)$$

where the $M_+ \times M_+$ matrix Z is unknown. A solution Z of (38) is called a *minimal nonnegative solution* if $0 \leq Z \leq Y$ for any other nonnegative solution Y , where the inequalities hold component-wise.

An analysis to obtain the stationary distribution starts with consideration of the probabilities of first passage times to lower levels. In particular, we define $\theta := \min\{n \geq 0: X(n) = (0, j)\}$ to be the first return time to $\ell(0)$ (in any state j of level 0), and define the matrices $G^{(i)}$ such that

$$G_{j,j'}^{(i)} := \mathbb{P}[\theta < \infty, X(\theta) = (0, j') | X(0) = (i, j)]; \quad (39)$$

thus, $G_{j,j'}^{(i)}$ provides the probability that the process reaches $\ell(0)$ in finite time with $(0, j')$ being the first state visited in $\ell(0)$, given that the process starts in state (i, j) of $\ell(i)$. Since any sequence of transitions that the process follows from $\ell(i)$ to $\ell(0)$ can also be followed by the process from $\ell(i+1)$ to $\ell(1)$, from $\ell(i+2)$ to $\ell(2)$, and so on, we further have

$$G_{j,j'}^{(i)} = \mathbb{P}[\theta(k) < \infty, X(\theta(k)) = (k, j') | X(0) = (i+k, j)], \quad \forall k \geq 0, \quad (40)$$

where $\theta(k) := \min\{n \geq 0: X(n) = (k, j)\}$ is defined to be the first return time to $\ell(k)$ (in any state j of level k). Define $G_{\min}$ to be the matrix of first passage time probabilities from

$\ell(i+1)$ to $\ell(i)$ independent of $i \geq 0$. Then the stationary distribution for $M/G/1$ -type processes can be obtained through the combination of the following results.

Lemma 3.4.1. We have

$$H \begin{bmatrix} G^{(1)} \\ G^{(2)} \\ G^{(3)} \\ \vdots \end{bmatrix} = \begin{bmatrix} A_{-1} \\ 0 \\ 0 \\ \vdots \end{bmatrix}, \quad (41)$$

where

$$H = \begin{bmatrix} I - A_0 & -A_1 & -A_2 & -A_3 & \cdots \\ -A_{-1} & I - A_0 & -A_1 & -A_2 & \cdots \\ 0 & -A_{-1} & I - A_0 & -A_1 & \ddots \\ 0 & 0 & -A_{-1} & I - A_0 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (42)$$

and where the matrices $G^{(1)}, G^{(2)}, G^{(3)}, \dots$ form the minimal nonnegative element-wise solution of (41)

Theorem 3.4.1a. The matrix $G_{\min}$ is the minimal nonnegative solution of (38), where we also have $G^{(i)} = G_{\min}^i$ for all $i \geq 0$ and $G_{\min}$ is stochastic.

Theorem 3.4.1b. For $i \geq 0$, let

$$A_i^* = \sum_{k=i}^{\infty} A_k G_{\min}^{k-i}, \quad B_i^* = \sum_{k=i}^{\infty} B_k G_{\min}^{k-i}. \quad (43)$$

Then $I - A_0^*$ is nonsingular and the stationary distribution of the $M/G/1$ -type Markov process is given by

$$\pi_i = (\pi_0 B_i^* + \sum_{k=1}^{i-1} \pi_k A_{i-k}^*) (I - A_0^*)^{-1}, \quad \text{for } i \geq 1, \quad (44)$$

where π_0 is obtained up to a multiplicative constant by

$$\pi_0 B_0^* = \pi_0. \quad (45)$$

Note that the above results hold when $M = \infty$, in which case $(I - A_0^*)^{-1}$ in (44) is simply replaced with $\sum_{i=0}^{\infty} (A_0^*)^i$. We associate with the transition probability matrix 36 the following matrix functions

$$S(z) = z^{-1} A_{-1} + A_0 + z A_1 + z^2 A_2 + \cdots = \sum_{i=-1}^{\infty} z^i A_i, \quad A(z) = z S(z), \quad (46)$$

which can play an important role in what follows. The function $A(z)$ is a matrix power series whose elements are analytic in the open unit disk and convergent in the closed unit disk. This series is often called the generating function of the Markov process.

We then have the following additional result.

Theorem 3.4.1c. Under certain technical assumptions, the matrix Laurent power series $I - S(z)$ has the weak canonical factorization

$$\begin{aligned} I - S(z) &= U(z)L(z), & |z| &= 1, \\ U(z) &= I - \sum_{i=0}^{\infty} z^i A_i^*, & L(z) &= I - z^{-1}G_{\min}, \end{aligned} \quad (47)$$

where the matrices A_i^* are given in (43).

We can express the above weak factorization in matrix form by $H = UL$ where

$$U = \begin{bmatrix} I - A_0^* & -A_1^* & -A_2^* & -A_3^* & \cdots \\ 0 & I - A_0^* & -A_1^* & -A_2^* & \cdots \\ 0 & 0 & I - A_0^* & -A_1^* & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix}, \quad L = \begin{bmatrix} I & 0 & 0 & 0 & \cdots \\ -G_{\min} & I & 0 & 0 & \cdots \\ 0 & -G_{\min} & I & 0 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (48)$$

and H is given in 42. Then $\pi P = \pi$ can be rewritten as

$$(\pi_1, \pi_2, \pi_3, \dots)H = \pi_0(B_1, B_2, B_3, \dots), \quad (49)$$

which is consistent with (44).

Additional theoretical results can be established on the close connections among characteristics of the solutions G and characteristics of the matrix $A(z)$. However, we conclude the analysis for the $M/G/1$ -type Markov process at this point for now.

3.4.2 G/M/1-Type Markov Processes.

Consider the discrete-time Markov process $\{X(n); n \in \mathbb{Z}_+\}$ whose transition probability matrix P has the structure given by

$$P = \begin{bmatrix} B_0 & A_1 & 0 & 0 & 0 & \cdots \\ B_{-1} & A_0 & A_1 & 0 & 0 & \cdots \\ B_{-2} & A_{-1} & A_0 & A_1 & 0 & \cdots \\ B_{-3} & A_{-2} & A_{-1} & A_0 & A_1 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (50)$$

where B_{-i} , for $i \geq 0$, and A_{-i} , for $i \geq -1$, are nonnegative matrices in $\mathbb{R}^{M_+ \times M_+}$ such that $\sum_{k=-1}^{i-1} A_{-k} + B_{-i}$ is stochastic for all $i \geq 0$. Observe that the transition probability matrix P is block lower Hessenberg and, with the exception of the first block column, is block Toeplitz.

The elements of the B_{-i} and A_{-i} matrices over the two-dimensional state space $\mathcal{S}$ have the following probabilistic interpretation:

$$\begin{aligned} (B_{-k})_{j,j'} &= \mathbb{P}[X(n+1) = (0, j') | X(n) = (k, j)], \quad \forall k \geq 0, j, j' \in [M], \\ (A_{-k})_{j,j'} &= \mathbb{P}[X(n+1) = (i-k, j') | X(n) = (i, j)], \quad \forall k \geq -1, i \geq 1, j, j' \in [M]. \end{aligned} \quad (51)$$

Namely, B_0 provides the probabilities of transitions within the boundary $\ell(0)$, $B_{-k, k \geq 1}$ provides the probabilities of transitions to the left by k levels to the boundary $\ell(0)$, A_0 provides the probabilities of transitions within any $\ell(i)$, $i \geq 1$, $A_{-k, k \geq 1}$ provides the probabilities of transitions to the left by k levels from any $\ell(i)$, $i \geq 1$, and A_1 provides the probabilities of transitions to the right by 1 level from any $\ell(i)$, $i \geq 0$. The key to solving for the stationary distribution is the matrix equation

$$Z = \sum_{i=-1}^{\infty} Z^{i+1} A_{-i}, \quad (52)$$

where the $M_+ \times M_+$ matrix Z is unknown. A solution Z of (52) is called a *minimal nonnegative solution* if $0 \leq Z \leq Y$ for any other nonnegative solution Y , where the inequalities hold component-wise.

An analysis to obtain the stationary distribution is summarized in the following results for $G/M/1$ processes.

Theorem 3.4.2a. The stationary distribution of the $G/M/1$ -type Markov process is of a matrix-geometric form given by

$$\pi_i = \pi_0 R_{\min}^i, \quad \text{for } i \geq 0, \quad (53)$$

where $R_{\min}$ is the minimal nonnegative solution of (52). The boundary vector π_0 is characterized by

$$\pi_0 \sum_{i=0}^{\infty} R_{\min}^i B_{-i} = \pi_0, \quad \pi_0 (I - R_{\min})^{-1} e = 1. \quad (54)$$

Note that the above results hold when $M = \infty$, in which case $(I - R_{\min})^{-1}$ in (54) is simply replaced with $\sum_{i=0}^{\infty} R_{\min}^i$. A probabilistic interpretation of the elements of the matrix $R_{\min}$ is such that $(R_{\min})_{j,j'}$ represents the expected number of visits to state $(i+1, j')$, given that the process starts in state (i, j) and before the first return to $\ell(i)$.

Analogous to the $M/G/1$ -type processes, we associate with the transition probability matrix (50) the following matrix functions

$$S(z) = zA_1 + A_0 + z^{-1}A_{-1} + z^{-2}A_{-2} + \cdots = \sum_{i=-1}^{\infty} z^{-i} A_{-i}, \quad A(z) = z^{-1}S(z). \quad (55)$$

The function $A(z)$ is a matrix power series in z^{-1} and $A(z^{-1})$ is analytic in the open unit disk and convergent in the closed unit disk. This series is often called the generating function of the Markov process.

There are additional important connections between $M/G/1$ -type processes and $G/M/1$ -type processes. Let $A = \sum_{i=-1}^{\infty} A_{-i}$, which we will often assume to be irreducible and stochastic in what follows. Define $D := \text{diag}(\pi_A)$, where π_A is the strictly positive stationary probability vector of A , and define $\tilde{A}_i := D^{-1}A_{-i}^T D$ for $i = -1, 0, 1, \dots$. The matrix $\tilde{A}_i = \sum_{j=-1}^{\infty} \tilde{A}_{ij}$ is stochastic and has the same stationary probability vector π_A as A . Taking any solution R of (52) and defining $\tilde{G} := D^{-1}R^T D$, it is readily verified that $\tilde{G}$ is a solution of

$$Z = \sum_{i=-1}^{\infty} \tilde{A}_i Z^{i+1}, \quad (56)$$

which is the characteristic equation of the $M/G/1$ -type process with transition probability matrix

$$\tilde{P} = \begin{bmatrix} \tilde{B}_0 & \tilde{B}_1 & \tilde{B}_2 & \tilde{B}_3 & \tilde{B}_4 & \cdots \\ \tilde{A}_{-1} & \tilde{A}_0 & \tilde{A}_1 & \tilde{A}_2 & \tilde{A}_3 & \cdots \\ 0 & \tilde{A}_{-1} & \tilde{A}_0 & \tilde{A}_1 & \tilde{A}_2 & \ddots \\ 0 & 0 & \tilde{A}_{-1} & \tilde{A}_0 & \tilde{A}_1 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (57)$$

where $\tilde{B}_i = D^{-1}B_{-i}^T D$ for $i = 0, 1, 2, \dots$. It is also readily verified that $D^{-1}R_{\min}^T D$ is the minimal nonnegative solution of (56). From these and related results, we then have the following theorem connecting $G/M/1$ -type and $M/G/1$ -type processes.

Theorem 3.4.2b. Determining $R_{\min}$ for a positive recurrent $G/M/1$ -type Markov process is equivalent to determining $G_{\min}$ for a transient $M/G/1$ -type Markov process. Conversely, determining $R_{\min}$ for a null recurrent or transient $G/M/1$ -type Markov process is equivalent to determining $G_{\min}$ for a recurrent $M/G/1$ -type Markov process.

Based on additional related properties, the canonical factorization associated with the $M/G/1$ -type Markov process of Theorem 3.4.1.c gives rise to a canonical factorization of the matrix function $I - S(z)$ in (55) as follows.

Theorem 3.4.2c. Under certain technical assumptions, the matrix Laurent power series $I - S(z)$ has the factorization

$$I - S(z) = (I - zR_{\min})(I - \sum_{i=0}^{\infty} z^{-i} A_{-i}^*), \quad (58)$$

where

$$A_{-i}^* = \sum_{j=i}^{\infty} R_{\min}^{j-i} A_{-j}, \quad i \geq 0. \quad (59)$$

This factorization is canonical if A is substochastic and is weak canonical if A is stochastic.

We can then express the above factorization in matrix form by rewriting (58) to obtain the factorization $H = UL$ where

$$H = \begin{bmatrix} I - A_0 & -A_1 & 0 & 0 & \cdots \\ -A_{-1} & I - A_0 & -A_1 & 0 & \cdots \\ -A_{-2} & -A_{-1} & I - A_0 & -A_1 & \ddots \\ -A_{-3} & -A_{-2} & -A_{-1} & I - A_0 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix} \quad (60)$$

and

$$U = \begin{bmatrix} I & -R_{\min} & 0 & 0 & \cdots \\ 0 & I & -R_{\min} & 0 & \cdots \\ 0 & 0 & I & -R_{\min} & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix}, \quad L = \begin{bmatrix} I - A_0^* & 0 & 0 & 0 & \cdots \\ -A_1^* & I - A_0^* & 0 & 0 & \cdots \\ -A_2^* & -A_1^* & I - A_0^* & 0 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix}. \quad (61)$$

Now $\pi P = \pi$ can be rewritten as

$$(\pi_1, \pi_2, \pi_3, \dots)H = \pi_0(A_1, 0, 0, \dots), \quad (62)$$

and from the above factorization of H we have

$$(\pi_1, \pi_2, \pi_3, \dots)U = \pi_0(A_1(I - A_0^*)^{-1}, 0, 0, \dots). \quad (63)$$

Solving this block triangular system renders (53). These results help to explain why the vast majority of classical computational algorithms have focused on solving equation (38) instead of equation (52).

3.4.3 Quasi-Birth-Death Processes.

The quasi-birth-death (QBD) process falls at the intersection of the $G/M/1$ -type and $M/G/1$ -type processes. Consider the discrete-time Markov process $\{X(n); n \in \mathbb{Z}_+\}$ whose transition probability matrix P has the structure given by

$$P = \begin{bmatrix} B_0 & A_1 & 0 & 0 & 0 & \cdots \\ A_{-1} & A_0 & A_1 & 0 & 0 & \cdots \\ 0 & A_{-1} & A_0 & A_1 & 0 & \cdots \\ 0 & 0 & A_{-1} & A_0 & A_1 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (64)$$

where B_0, B_1 and A_{-1}, A_0, A_1 are nonnegative matrices in $\mathbb{R}^{M_+ \times M_+}$ such that $A_{-1} + A_0 + A_1$ and $B_0 + B_1$ are stochastic.

Since a QBD process can be viewed as either a $G/M/1$ -type process or an $M/G/1$ -type process, Theorem 3.4.2.a applies and therefore the stationary distribution is of the matrix-geometric form (53). More precisely, we have the following result for the stationary distribution.

Theorem 3.4.3. The stationary distribution of the QBD process is of a matrix-geometric form given by

$$\begin{aligned} \pi_i &= \pi_0 R_{\min}^i, & \text{for } i \geq 0, \\ \pi_0(B_0 + B_1 G_{\min}) &= \pi_0, & \pi_0(I - R_{\min})^{-1}e = 1, \end{aligned} \quad (65)$$

where $R_{\min}$ is the minimal nonnegative solution of

$$Z = Z^2 A_{-1} + Z A_0 + A_1, \quad (66)$$

and $G_{\min}$ is the minimal nonnegative solution of

$$Z = A_{-1} + A_0 Z + A_1 Z^2. \quad (67)$$

Because it lies at the intersection of $G/M/1$ -type and $M/G/1$ -type processes, there is a direct connection between the matrix $R_{\min}$ and the matrix $\tilde{G}_{\min}$ of the QBD process. This connection is established via the matrix A_0^* which can play an important role and is typically denoted by U in the QBD case. More specifically, we have

$$G_{\min} = (I - U)^{-1} A_{-1}, \quad R_{\min} = A_1 (I - U)^{-1}, \quad (68)$$

where $U = A_0 + A_1 G_{\min}$. Furthermore, it can be shown that U is the minimal nonnegative solution of

$$Z = A_0 + A_1 (I - Z)^{-1} A_{-1}. \quad (69)$$

The matrix function $S(z)$ associated with the transition probability matrix (64) of a QBD process is the matrix Laurent polynomial

$$S(z) = z^{-1} A_{-1} + A_0 + z A_1, \quad (70)$$

where the corresponding generating function of the Markov process is the polynomial $A(z) = A_{-1} + z A_0 + z^2 A_1$ from the perspective of the QBD process as an $M/G/1$ -type process and is the polynomial $\hat{A}(z) = A_1 + z A_0 + z^2 A_{-1}$ from the perspective of the QBD process as a $G/M/1$ -type process. By reversing the order of the matrix coefficients of the polynomial $A(z)$ through the transformation $A(z) \Rightarrow z^2 A(z^{-1}) = \hat{A}(z)$, one can exploit whichever of the two perspectives is most convenient as well as switch between the two perspectives.

3.4.4 Simple Examples.

Two-Queue Tandem Network. Consider two queues in tandem to which arrivals come from an exogeneous Poisson process with rate λ . The service times at the first queue are independent and identically distributed (i.i.d.) according to an exponential distribution with rate μ_1 , and the service times at the second queue are i.i.d. according to an exponential

distribution with rate μ_2 . This simple example is a continuous-time QBD process, but it illustrates the structural properties just the same; in particular, one can apply uniformization to obtain the corresponding discrete-time QBD process with transition probability matrix $\mathbf{P}$ as in (64).

The transition rates are as follows:

1. state $(i, j) \rightarrow$ state $(i + 1, j)$ with rate λ : arrival;
2. state $(i, j) \rightarrow$ state $(i - 1, j + 1)$ with rate μ_1 : departure from queue 1, going to queue 2;
3. state $(i, j) \rightarrow$ state $(i, j - 1)$ with rate μ_2 : departure from queue 2.

We therefore have $A_1 = \lambda I$,

$$A_{-1} = \begin{bmatrix} 0 & \mu_1 & 0 & 0 & 0 & \cdots \\ 0 & 0 & \mu_1 & 0 & 0 & \cdots \\ 0 & 0 & 0 & \mu_1 & 0 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (71)$$

$$A_0 = \begin{bmatrix} -(\lambda + \mu_1 + \mu_2) & 0 & 0 & 0 & 0 & \cdots \\ \mu_2 & -(\lambda + \mu_1 + \mu_2) & 0 & 0 & 0 & \cdots \\ 0 & \mu_2 & -(\lambda + \mu_1 + \mu_2) & 0 & 0 & \ddots \\ 0 & 0 & \mu_2 & -(\lambda + \mu_1 + \mu_2) & 0 & \ddots \\ \vdots & \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (72)$$

$$B_0 = \begin{bmatrix} -(\lambda + \mu_2) & 0 & 0 & 0 & 0 & \cdots \\ \mu_2 & -(\lambda + \mu_2) & 0 & 0 & 0 & \cdots \\ 0 & \mu_2 & -(\lambda + \mu_2) & 0 & 0 & \ddots \\ 0 & 0 & \mu_2 & -(\lambda + \mu_2) & 0 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}. \quad (73)$$

Random Walk. Consider a discrete-time random walk defined on the nonnegative quarter-plane with transitions upward having probability p_U , transitions downward having probability p_D , transitions to the left having probability p_L , and transitions to the right having probability p_R such that $p_U + p_D + p_L + p_R = \mathbf{1}$. This simple example is a discrete-time QBD process with transition probability matrix $\mathbf{P}$ as in (64) and having appropriate boundary conditions for the states $(i, \mathbf{0})$ and $(\mathbf{0}, j)$.

The transition probabilities are as follows:

1. state $(i, j) \rightarrow$ state $(i, j + 1)$ with probability p_U : upward;
2. state $(i, j) \rightarrow$ state $(i, j - 1)$ with probability p_D : downward;
3. state $(i, j) \rightarrow$ state $(i - 1, j)$ with probability p_L : left;
4. state $(i, j) \rightarrow$ state $(i + 1, j)$ with probability p_R : right.

We therefore have $A_{-1} = p_D I$, $A_1 = p_U I$,

$$A_0 = \begin{bmatrix} p_L & p_R & 0 & 0 & 0 & \cdots \\ p_L & 0 & p_R & 0 & 0 & \cdots \\ 0 & p_L & 0 & p_R & 0 & \ddots \\ 0 & 0 & p_L & 0 & p_R & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (74)$$

$$B_0 = \begin{bmatrix} p_L + p_D & p_R & 0 & 0 & 0 & \cdots \\ p_L & p_D & p_R & 0 & 0 & \cdots \\ 0 & p_L & p_D & p_R & 0 & \ddots \\ 0 & 0 & p_L & p_D & p_R & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}. \quad (75)$$

3.4.5 Computational Algorithms.

As previously noted, the computational complexity of solving for the stationary distribution of the Markov process $\{X(n); n \in \mathbb{Z}_+\}$ can be prohibitive. In some cases, one can exploit probabilistic interpretations of the G and R matrices of the Markov process to derive explicit solutions; refer to [5]. More generally, however, one must exploit the structural properties of the transition probability matrix P to realize more efficient solution methods. The primary difficulty concerns computing the solution of equation (38) that corresponds to the $M/G/1$ -type process, $G/M/1$ -type process or QBD processes of interest. This is true both in the context of classical computing environments, where a considerable amount of algorithmic research has been conducted, and in the context of quantum computing environments, where little to no research has been performed. In fact, to the best of our knowledge, quantum computational algorithms for solving the general class of decision making under uncertainty problems involving structured Markov processes of interest in our study have heretofore received no attention in the research literature.

Given the focus on equation (38) together with its properties and corresponding results, we first study the best-known algorithms within the context of classical computing environments from the perspective of developing more efficient quantum algorithms. At the heart of the most advanced and efficient classical computational algorithms are the fast Fourier transform (FFT), fast power series arithmetic, Toeplitz matrix computations, and/or displacement representations. The analysis and theoretical results presented above, including factorizations and matrix equations, establish important connections across the fields of applied probability, numerical analysis and operator theory. This in turn provides a framework in which computational algorithms can be developed from a quantum perspective, where one of our goals is to devise quantum algorithms based on the most advanced and efficient existing classical computational algorithms, and another of our goals is to exploit this framework to devise even more efficient quantum computational algorithms from first principles. As a specific example of the former, equation (44) is the canonical factorization of a matrix power series and thus FFT-based algorithms are often exploited to efficiently compute the solution of this formula.

We are investigating corresponding quantum FFT-based algorithms to improve the efficiency of solving for the stationary distribution of various $M/G/1$ -type, $G/M/1$ -type and

QBD processes. Moreover, as we increase our understanding of quantum algorithmic versions of the most advanced and efficient classical computational algorithms, we are exploiting the framework set forth above to investigate and devise quantum computational algorithms from first principles that significantly improve the computational complexity of solving for the stationary distribution of various $M/G/1$ -type, $G/M/1$ -type and QBD processes. Both lines of research are expected to provide significant improvements in the efficiency of solving decision making under uncertainty problems that involve specific decision parameters embedded in the corresponding $M/G/1$ -type, $G/M/1$ -type and QBD processes.

3.4.6 Classical Computing Algorithms.

We consider classical computing algorithms and start with the simple case that converge linearly toward the desired solution. To this end, we note that any solution G of (38) is a fixed point of the matrix valued function $V(G)$, i.e., $G = V(G)$, where $V(G) = \sum_{i=-1}^{\infty} A_i G^{i+1}$. The fixed-point iteration associated with $V(Z)$ is given by the set of all matrix sequences $\{Z_n\}$, $n \in \mathbb{Z}_+$, where $Z_{n+1} = V(Z_n)$, $n \in \mathbb{Z}_+$, and any given matrix Z_0 . We seek to obtain matrix sequences that converge to the minimal nonnegative solution G_{min} of (38). Since the fixed points of $V(Z) = \sum_{i=-1}^{\infty} A_i Z^{i+1}$ satisfy (38), this is one option for the desired fixed-point iteration. Alternative fixed-point iterations can be identified by working with (38), which in particular renders the matrix valued functions $V(Z) = (I - A_0)^{-1}(A_{-1} + \sum_{i=1}^{\infty} A_i Z^{i+1})$ and $V(Z) = (I - \sum_{i=0}^{\infty} A_i Z^i)^{-1}A_{-1}$. Based on these functions, we consider the splitting $A(z) = zV^{(1)}(z) + V^{(2)}(z)$ of the matrix power series (55) where $V^{(1)}(z) = \sum_{i=0}^{\infty} z^i V_i^{(1)}$ and $V^{(2)}(z) = \sum_{i=0}^{\infty} z^i V_i^{(2)}$ with $V_i^{(1)} \geq 0$ and $V_i^{(2)} \geq 0$ for all $i \geq 0$. This splitting then yields $A_{-1} = V_0^{(2)}$, $A_i = V_i^{(1)} + V_{i+1}^{(2)}$, $i \geq 0$, the matrix function

$$V(Z) = \left(I - V^{(1)}(Z)\right)^{-1} V^{(2)}(Z), \quad (76)$$

and in turn the matrix equation $Z = (I - V^{(1)}(Z))^{-1}V^{(2)}(Z)$, where $I - V^{(1)}(Z)$ is nonsingular for each matrix Z , $V^{(1)}(Z) = \sum_{i=0}^{\infty} V_i^{(1)} Z^i$ and $V^{(2)}(Z) = \sum_{i=0}^{\infty} V_i^{(2)} Z^i$. This leads to the following linearly convergent classical computing algorithm, which takes as input a matrix Z_0 , a tolerance $\epsilon > 0$, the $j + 2$ matrices $A_{-1}, A_0, A_i, i = 1, \dots, j \geq 1$, of the corresponding finite version of the matrix polynomial (55), and the matrix polynomials $V^{(1)}(Z)$ and $V^{(2)}(Z)$ of the matrix function $V(Z)$ in (76), and produces as output an approximation G of the minimal nonnegative solution G_{min} of $Z = A(Z)$ such that $\|G - V(G)\| < \epsilon$.

1. Initialize $Z = Z_0$.
2. Compute $V^{(1)}(Z) = \sum_{i=0}^j V_i^{(1)} Z^i$.
3. Compute $V^{(2)}(Z) = \sum_{i=0}^j V_i^{(2)} Z^i$.
4. Compute $G = (I - V^{(1)}(Z))^{-1}V^{(2)}(Z)$.

5. If $\|G - Z\| \geq \epsilon$, then set $Z = G$ and go to step 2.
6. Return G .

The convergence of the above algorithm is linear in the sense that, given the sequence $\{G_n\}$, $n \in \mathbb{Z}_+$, with $\lim_{n \rightarrow \infty} G_n = G_{min}$, then the limit $\lim_{n \rightarrow \infty} \frac{\|G_{min} - G_{n+1}\|_\infty}{\|G_{min} - G_n\|_\infty}$ is strictly between 0 and 1. On the other hand, if this limit is equal to 0, then the convergence is superlinear, in which case the convergence is quadratic if there exists a finite and nonzero limit for $\lim_{n \rightarrow \infty} \frac{\|G_{min} - G_{n+1}\|_\infty}{\|G_{min} - G_n\|_\infty^2}$.

An alternative algorithm that leads to quadratic convergence, within the context of classical computing environments, is based on applying Newton's method to solve (38). This renders the following algorithm, which takes as input a tolerance $\epsilon > 0$, and the $j + 2$ matrices A_{-1} , A_0 , A_i , $i = 1, \dots, j \geq 1$, of the corresponding finite version of the matrix polynomial (55), and produces as output an approximation G of the minimal nonnegative solution G_{min} of $Z = A(Z)$ such that $\|G - G_{min}\|_\infty < \epsilon$. Let $z = \text{vec}(Z)$ denote the vector formed by column-wise arranging of the matrix Z , $Z = \text{vec}^{-1}(z)$ the matrix such that $z = \text{vec}(Z)$, Z^T the transpose of the matrix Z , and e the vector of all ones.

1. Initialize $G = (I - A_0)^{-1}A_{-1}$.
2. Set $W = A_j$.
3. For $i = 1, \dots, j + 1$, compute $W = A_{j-i} + WG$.
4. Set $D_G = 0$, $Y = A_{-1}^T(I - W^T)^{-1}$.
5. For $i = 1, \dots, j$:
 - a. Set $X = A_j$;
 - b. For $k = j - i, \dots, 1$, compute $X = A_{j-k} + XG$;
 - c. Compute $X = (I - W)^{-1}X$, $D_G = D_G + Y \otimes X$, $Y = YG^T$.
6. Compute $z = (I - D_G)^{-1}\text{vec}((I - W)^{-1}A_{-1} - G)$.
7. Compute $G = G + \text{vec}^{-1}(z)$.
8. If $\|Ge - e\|_\infty \geq \epsilon$, then go to step 2.
9. Return G .

We next consider a class of algorithms that retains quadratic convergence but, in contrast to the above Newton-based method, has a one-step computational effort which is comparable to that of the linear convergent algorithm above. To this end, we rewrite (38) as the semi-infinite linear system

$$\begin{bmatrix} I - A_0 & -A_1 & -A_2 & -A_3 & \cdots \\ -A_{-1} & I - A_0 & -A_1 & -A_2 & \ddots \\ 0 & -A_{-1} & I - A_0 & -A_1 & \ddots \\ 0 & 0 & -A_{-1} & I - A_0 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots \end{bmatrix} \begin{bmatrix} G_{min} \\ G_{min}^2 \\ G_{min}^3 \\ G_{min}^4 \\ \vdots \end{bmatrix} = \begin{bmatrix} A_{-1} \\ 0 \\ 0 \\ 0 \\ \vdots \end{bmatrix}. \quad (77)$$

An even-odd permutation of the block rows and block columns of (77) then renders

$$\begin{bmatrix} I - U_{11} & -U_{12} \\ -U_{21} & I - U_{22} \end{bmatrix} \begin{bmatrix} x_+ \\ x_- \end{bmatrix} = \begin{bmatrix} 0 \\ y \end{bmatrix} \quad (78)$$

where

$$\begin{aligned} U_{11} &= \begin{bmatrix} -A_0 & -A_2 & -A_4 & \cdots \\ 0 & -A_0 & -A_2 & \ddots \\ 0 & 0 & -A_0 & \ddots \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix}, & U_{12} &= \begin{bmatrix} -A_{-1} & -A_1 & -A_3 & \cdots \\ 0 & -A_{-1} & -A_1 & \ddots \\ 0 & 0 & -A_{-1} & \ddots \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix}, \\ U_{21} &= \begin{bmatrix} -A_1 & -A_3 & -A_5 & \cdots \\ -A_{-1} & -A_1 & -A_3 & \ddots \\ 0 & -A_{-1} & -A_1 & \ddots \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix}, & U_{22} &= \begin{bmatrix} -A_0 & -A_2 & -A_4 & \cdots \\ 0 & -A_0 & -A_2 & \ddots \\ 0 & 0 & -A_0 & \ddots \\ \vdots & \ddots & \ddots & \ddots \end{bmatrix}, \end{aligned} \quad (79)$$

x_+ is the block vector of components G_{min}^{2i} , x_- is the block vector of components G_{min}^{2i-1} , $i \in \mathbb{Z}^+$, and y is the block vector with first component A_{-1} and all other components 0. We then have the following so-called cyclic reduction algorithm, which takes as input a tolerance $\epsilon > 0$ and the $j + 1$ matrices $A_{-1}, A_0, A_i, i = 1, \dots, j \geq 1$, of the block Toeplitz block Hessenberg matrix in (76), and produces as output an approximation G of the minimal nonnegative solution G_{min} of (38) within an ϵ error tolerance.

1. Initialize $n = 0$, $A^{(n)}(z) = \sum_{i=-1}^{j-1} z^{i+1} A_i$, $\hat{A}^{(n)}(z) = \sum_{i=0}^{j-1} z^i A_i$.
2. Compute approximation $Y^{(n+1)}(z)$ of $A^{(n+1)}(z)$ and approximation $\hat{Y}^{(n+1)}(z)$ of $\hat{A}^{(n+1)}(z)$ using a functional version of the cyclic reduction algorithm.
3. If $\|e - (A_{-1} + \hat{Y}_0^{(n)})e\|_\infty \geq \epsilon$, set $n = n + 1$, $A^{(n)}(z) = Y^{(n)}(z)$, $\hat{A}^{(n)}(z) = \hat{Y}^{(n)}(z)$, go to step 2.
4. Compute and return $G = (I - \hat{Y}_0^{(n)})^{-1} A_{-1}$.

Let us define the block Hessenberg matrix $H^{(n)}$ of the n th iteration as

$$H^{(n)} = \begin{bmatrix} I - \hat{A}_0^{(n)} & -\hat{A}_1^{(n)} & -\hat{A}_2^{(n)} & -\hat{A}_3^{(n)} & \cdots \\ -A_{-1}^{(n)} & I - A_0^{(n)} & -A_1^{(n)} & -A_2^{(n)} & \ddots \\ 0 - A_{-1}^{(n)} & & I - A_0^{(n)} & -A_1^{(n)} & \ddots \\ 0 & 0 & -A_{-1}^{(n)} & I - A_0^{(n)} & \ddots \\ \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}. \quad (80)$$

We also define the vectors $a^{(n)} = \sum_{i=0}^{\infty} (i+1)A_i^{(n)}e$ and $\hat{a}^{(n)} = \sum_{i=1}^{\infty} i\hat{A}_i^{(n)}e$, which can be recursively computed as $a^{(n+1)} = (e + a^{(n)} - A_{even}^{(n)}(1)(I - A_{odd}^{(n)}(1))^{-1}(e - a^{(n)}))/2$ and $\hat{a}^{(n+1)} = (\hat{a}^{(n)} - \hat{A}_{odd}^{(n)}(1)(I - A_{odd}^{(n)}(1))^{-1}(e - a^{(n)}))/2$, respectively.

Let DFT_n denote the discrete Fourier transform of order n and $IDFT_n$ the inverse discrete Fourier transform of order n . The functional version of the cyclic reduction algorithm then proceeds as follows, taking as input a tolerance $\epsilon > 0$, the $d = 2^c$ matrices $A_{-1}, A_0, A_i, i = 1, \dots, d-2$ and the d matrices $\hat{A}_i, i = 0, \dots, d-1$ of the first two block rows of (80), and the two vectors $a^{(n)}, \hat{a}^{(n)}$ defined above, and produces as output the $2d' > 0$ approximations $S'_{-1}, S'_0, \dots, S'_{d'-2}$ and $\hat{S}'_0, \dots, \hat{S}'_{d'-1}$ of the block elements $A'_{-1}, A'_0, \dots, A'_{d'-2}$ and $\hat{A}'_0, \dots, \hat{A}'_{d'-1}$ of the first two block rows of (80), and the vectors $a' = \sum_{i=0}^{d'-2} (i+1)A'_i e$ and $\hat{a}' = \sum_{i=1}^{d'-1} i\hat{A}'_i e$.

1. Initialize $a' = (e + a - A_{even}(1)(I - A_{odd}(1))^{-1}(e - a))/2$, $\hat{a}' = (\hat{a} - \hat{A}_{odd}(1)(I - A_{odd}(1))^{-1}(e - a))/2$, and $M = d/2$.
2. Compute $W_{1j} = A_{odd}(\omega_M^j)$, $W_{2j} = A_{even}(\omega_M^j)$, $\hat{W}_{1j} = \hat{A}_{odd}(\omega_M^j)$, $\hat{W}_{2j} = \hat{A}_{even}(\omega_M^j)$, for $j = 0, \dots, M-1$, as follows:
 - a. Set $x = (X_i)_{i=1, \dots, M}$ where $X_i = A_{2i-2}$ for $i = 1, \dots, d/2$ and $X_i = 0$ for $i > d/2$; compute $y = IDFT_M(x)$ and set $W_{1j-1} = Y_j$ for $j = 1, \dots, M$.
 - b. Set $x = (X_i)_{i=1, \dots, M}$ where $X_i = A_{2i-3}$ for $i = 1, \dots, d/2$ and $X_i = 0$ for $i > d/2$; compute $y = IDFT_M(x)$ and set $W_{2j-1} = Y_j$ for $j = 1, \dots, M$.
 - c. Set $x = (X_i)_{i=1, \dots, M}$ where $X_i = A_{2i-1}$ for $i = 1, \dots, d/2$ and $X_i = 0$ for $i > d/2$; compute $y = IDFT_M(x)$ and set $W_{1j-1} = Y_j$ for $j = 1, \dots, M$.
 - d. Set $x = (X_i)_{i=1, \dots, M}$ where $X_i = A_{2i-2}$ for $i = 1, \dots, d/2$ and $X_i = 0$ for $i > d/2$; compute $y = IDFT_M(x)$ and set $W_{2j-1} = Y_j$ for $j = 1, \dots, M$.
3. Compute $V_j = \omega_M^j W_{1j} + W_{2j}(I - W_{1j})^{-1}W_{2j}$ and $\hat{V}_j = \hat{W}_{2j} + \hat{W}_{1j}(I - W_{1j})^{-1}W_{2j}$, for $j = 0, \dots, M-1$.
4. Compute the coefficients P_i and $\hat{P}_i, i = 0, \dots, M-1$, of the matrix polynomials $P(z)$ and $\hat{P}(z)$, respectively, such that $P(\omega_M^j) = V_j$ and $\hat{P}(\omega_M^j) = \hat{V}_j, j = 0, \dots, M-1$, as follows:
 - a. Set $y = (Y_i)_{i=1, \dots, M}$ where $Y_i = V_{i-1}$ for $i = 1, \dots, M$; compute $x = DFT_M(y)$ and set $P_{j-1} = X_j$, for $j = 1, \dots, M$.
 - b. Set $y = (Y_i)_{i=1, \dots, M}$ where $Y_i = \hat{V}_{i-1}$ for $i = 1, \dots, M$; compute $x = DFT_M(y)$ and set $\hat{P}_{j-1} = X_j$, for $j = 1, \dots, M$.

5. If $a' - \sum_{i=1}^{M-1} i P_i e \geq M/2\epsilon e$ or $\hat{a}' - \sum_{i=1}^{M-1} i \hat{P}_i e \geq M/2\epsilon e$, set $M = 2M$ and go to step 3.
6. Return $d' = M$, $S'_{j-1} = P_j$ and $\hat{S}'_j = \hat{P}_j$, $j = 0, \dots, d' - 1$.

3.5 Algorithms - Structured Random Walks

Random walks represent a very large class of Markov processes that are of particular interest from a quantum perspective. One important subset of this class of Markov processes concerns random walks on a finite graph, which have been considered extensively from a quantum computing perspective in the computer science theory community (see, e.g., [6], [7], [8], [9]). We are continuing to pursue research in this area of so-called quantum walks, with some of our progress discussed in section §3.6.

Another important subset of the very large class of random walk Markov processes concerns random walks with certain structural properties. One simple specific example is the one-dimensional random walk defined on the nonnegative quarter-plane, as briefly discussed in section §3.4.4. More generally, quasi-birth-death (QBD) processes represent an important class of structured random walks whose transition probability matrix P (for the discrete-time case) has the structure given by

$$P = \begin{bmatrix} B_0 & A_1 & 0 & 0 & 0 & \dots \\ A_{-1} & A_0 & A_1 & 0 & 0 & \dots \\ 0 & A_{-1} & A_0 & A_1 & 0 & \dots \\ 0 & 0 & A_{-1} & A_0 & A_1 & \ddots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots \end{bmatrix}, \quad (81)$$

where B_0, B_1 and A_{-1}, A_0, A_1 are nonnegative matrices in $\mathbb{R}^{M_+ \times M_+}$ such that $A_{-1} + A_0 + A_1$ and $B_0 + B_1$ are stochastic.

The stationary distribution of the QBD process renders the equilibrium distribution of the corresponding structured random walk. We know that this desired stationary distribution for any irreducible and ergodic QBD process is given by Theorem 3.4.3, which establishes that the equilibrium distribution has a matrix-geometric form given by

$$\begin{aligned} \pi_i &= \pi_0 R_{\min}^i, & \text{for } i \geq 0, \\ \pi_0(B_0 + B_1 G_{\min}) &= \pi_0, & \pi_0(I - R_{\min})^{-1}e = 1, \end{aligned} \quad (82)$$

where $R_{\min}$ is the minimal nonnegative solution of

$$Z = Z^2 A_{-1} + Z A_0 + A_1, \quad (83)$$

and $G_{\min}$ is the minimal nonnegative solution of

$$Z = A_{-1} + A_0 Z + A_1 Z^2. \quad (84)$$

The existence of close connections between the matrix $R_{\min}$ and the matrix $\tilde{G}_{\min}$ of the dual process was established in section §4.2.2. Because it lies at the intersection of $G/M/1$ -type and $M/G/1$ -type processes, there is a direct connection between the matrix $R_{\min}$ and the matrix $\tilde{G}_{\min}$ of the QBD process. This connection is established via the matrix A_0^* which can play an important role and is typically denoted by U in the QBD case. More specifically, we have

$$G_{\min} = (I - U)^{-1}A_{-1}, \quad R_{\min} = A_1(I - U)^{-1}, \quad (85)$$

where $U = A_0 + A_1G_{\min}$. Furthermore, it can be shown that U is the minimal nonnegative solution of

$$Z = A_0 + A_1(I - Z)^{-1}A_{-1}. \quad (86)$$

The matrix function $S(z)$ associated with the above transition probability matrix P of a QBD process is the matrix Laurent polynomial

$$S(z) = z^{-1}A_{-1} + A_0 + zA_1, \quad (87)$$

where the corresponding generating function of the Markov process is the polynomial $A(z) = A_{-1} + zA_0 + z^2A_1$ from the perspective of the QBD process as an $M/G/1$ -type process and is the polynomial $\hat{A}(z) = A_1 + zA_0 + z^2A_{-1}$ from the perspective of the QBD process as a $G/M/1$ -type process. By reversing the order of the matrix coefficients of the polynomial $A(z)$ through the transformation $A(z) \Rightarrow z^2A(z^{-1}) = \hat{A}(z)$, one can exploit whichever of the two perspectives is most convenient as well as switch between the two.

The equilibrium distribution $\pi = (\pi_0, \pi_1, \pi_2, \dots)$ of the random walk is leveraged as a core component of a number of important algorithms, such as those related to search and clique detection. The best-known classical algorithm for computing the equilibrium distribution of structured random walks is called cyclic reduction. We next present this classical computing algorithm to solve for the equilibrium distribution of any QBD process, and then we present a quantum algorithm approach to significantly improve the computational complexity of such computations on a quantum computer. Lastly, we summarize the benefits of this proposed hybrid classical-quantum computational approach.

3.5.1 Classical Computing Algorithm: Cyclic Reduction

Applying cyclic reduction to QBD processes [10], our goal is to compute the minimal nonnegative solution $G_{\min}$ from which we can obtain $R_{\min} = A_1(I - U)^{-1}$ in Theorem

3.4.3 where $U = A_0 + A_1 G_{\min}$. For the computation of the minimal nonnegative solution $G_{\min}$, we have

$$\begin{bmatrix} I - A_0 & -A_1 & & \\ -A_{-1} & I - A_0 & -A_1 & \\ & -A_{-1} & I - A_0 & \ddots \\ & & \ddots & \ddots \end{bmatrix} \begin{bmatrix} G_{\min} \\ G_{\min}^2 \\ G_{\min}^3 \\ \vdots \end{bmatrix} = \begin{bmatrix} A_{-1} \\ 0 \\ 0 \\ \vdots \end{bmatrix}. \quad (88)$$

Applying an even-odd permutation to the block rows and to the block columns of the above system yields

$$\begin{bmatrix} I - A_0 & & -A_{-1} & -A_1 & & \\ & I - A_0 & & -A_{-1} & \ddots & \\ & & \ddots & & \ddots & \\ -A_1 & & & I - A_0 & & \\ -A_{-1} & -A_1 & & & I - A_0 & \\ & \ddots & \ddots & & & \ddots \end{bmatrix} \begin{bmatrix} G_{\min}^2 \\ G_{\min}^4 \\ \vdots \\ G_{\min} \\ G_{\min}^3 \\ \vdots \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ A_{-1} \\ 0 \\ \vdots \end{bmatrix}, \quad (89)$$

$$\begin{bmatrix} I - U_{1,1} & -U_{1,2} \\ -U_{2,1} & I - U_{2,2} \end{bmatrix} \begin{bmatrix} x_+ \\ x_- \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix},$$

$$\begin{bmatrix} I - U_{1,1} & -U_{1,2} \\ & H \end{bmatrix} \begin{bmatrix} x_+ \\ x_- \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix},$$

where $H = I - U_{2,2} - U_{2,1}(I - U_{1,1})^{-1}U_{1,2}$.

It is important to observe in the above equation that H is a block-tridiagonal matrix which is Toeplitz except for the block position (1,1), namely

$$H = \begin{bmatrix} I - \hat{A}_0^{(1)} & -A_1^{(1)} & & \\ -A_{-1}^{(1)} & I - A_0^{(1)} & -A_1^{(1)} & \\ & -A_{-1}^{(1)} & I - A_0^{(1)} & \ddots \\ & & \ddots & \ddots \end{bmatrix} \quad (90)$$

where

$$\begin{aligned} A_{-1}^{(1)} &= A_{-1}(I - A_0)^{-1}A_{-1}, \\ A_0^{(1)} &= A_0 + A_{-1}(I - A_0)^{-1}A_{-1} + A_1(I - A_0)^{-1}A_{-1}, \\ A_1^{(1)} &= A_1(I - A_0)^{-1}A_1, \\ \hat{A}_0^{(1)} &= A_0 + A_1(I - A_0)^{-1}A_{-1}. \end{aligned} \quad (91)$$

Exploiting the above structure of H allows us to rewrite the equation $Hx_- = b$ above as

$$\begin{bmatrix} I - \hat{A}_0^{(1)} & -A_1^{(1)} & & \\ -A_{-1}^{(1)} & I - A_0^{(1)} & -A_1^{(1)} & \\ & -A_{-1}^{(1)} & I - A_0^{(1)} & \ddots \\ & & \ddots & \ddots \end{bmatrix} \begin{bmatrix} G_{\min} \\ G_{\min}^3 \\ G_{\min}^5 \\ \vdots \end{bmatrix} = \begin{bmatrix} A_{-1} \\ 0 \\ 0 \\ \vdots \end{bmatrix}. \quad (92)$$

The cyclic reduction algorithm then arises from repeated application of the foregoing mechanism in a recursive manner. In particular, after n reductions, the linear system of interest becomes

$$H = \begin{bmatrix} I - \hat{A}_0^{(n)} & -A_1^{(n)} & & \\ -A_{-1}^{(n)} & I - A_0^{(n)} & -A_1^{(n)} & \\ & -A_{-1}^{(n)} & I - A_0^{(n)} & \ddots \\ & & \ddots & \ddots \end{bmatrix} \begin{bmatrix} G_{\min} \\ G_{\min}^{2^{n+1}} \\ G_{\min}^{2 \cdot 2^{n+1}} \\ \vdots \end{bmatrix} = \begin{bmatrix} A_{-1}^{(n)} \\ 0 \\ 0 \\ \vdots \end{bmatrix} \quad (93)$$

where

$$\begin{aligned} A_{-1}^{(n+1)} &= A_{-1}^{(n)} (I - A_0^{(n)})^{-1} A_{-1}^{(n)}, \\ A_0^{(n+1)} &= A_0^{(n)} + A_{-1}^{(n)} (I - A_0^{(n)})^{-1} A_{-1}^{(n)} + A_1^{(n)} (I - A_0^{(n)})^{-1} A_{-1}^{(n)}, \\ A_1^{(n+1)} &= A_1^{(n)} (I - A_0^{(n)})^{-1} A_{-1}^{(n)}, \\ \hat{A}_0^{(n+1)} &= A_0^{(n)} + A_1^{(n)} (I - A_0^{(n)})^{-1} A_{-1}^{(n)}, \end{aligned} \quad (94)$$

$A_0^{(0)} = A_0$, and $A_i^{(0)} = A_i$ for $i = -1, 0, 1$. Under the assumptions that $\det(I - A_0^{(n)}) \neq 0$ and $\det(I - \hat{A}_0^{(n)}) \neq 0$ for any $n = 0, 1, \dots$, the cyclic reduction algorithm computes

$$G_{\min} = (I - \hat{A}_0^{(n)})^{-1} (A_{-1} + A_1^{(n)} G_{\min}^{2^{n+1}}), \quad (95)$$

which is a valid approximate solution as long as the sequence $\{A_1^{(n)} G_{\min}^{2^{n+1}}\}$ converges quadratically to zero and, at the same time, $\{I - \hat{A}_0^{(n)}\}$ converges quadratically to a nonsingular matrix.

To summarize, the classical cyclic reduction algorithm to compute the matrix $G_{\min}$ consists of the following steps:

1. Input: the $M_+ \times M_+$ matrices A_{-1} , A_0 , A_1 , and a real $\xi > 0$.
2. Set $V_i = A_i$, $i = -1, 0, 1$, and $\hat{V} = A_0$.
3. Compute $H = (I - V_0)^{-1} V_1$ and $K = (I - V_0)^{-1} V_{-1}$.
4. Compute $V'_1 = V_1 H$, $V'_{-1} = V_{-1} H$, $W = V_1 K$, $\hat{V}' = \hat{V} + W$, $V'_0 = V_0 + W + V_{-1} H$.
5. If $\|V_1\|_\infty < \xi$, then output $Y = (I - \hat{V})^{-1} A_{-1}$ and stop, otherwise repeat step 2.

Convergence and Complexity Analysis. We note that each iteration of the above cyclic reduction algorithm involves one matrix inversion and six matrix multiplications. Hence, the computational cost per iteration will be on the order $O(M_+^3 + M_+^2)$. The algorithm has been shown to have a quadratic convergence rate and the approximation error is given by (Bini, Latouche, and Meini 2005 Theorem 7.6)

$$\|G_{\min} - (I - \widehat{A}_0^{(n)})^{-1}A_{-1}\| \leq O\left(\left(\frac{1}{\sigma}\right)^{2^n}\right), \quad (96)$$

where $\sigma = \min\{|\lambda|: |\lambda| > 1, \phi(\lambda) = 0\}$ and $\phi(\lambda) = \det(-A_0 + (I - A_1)\lambda - A_2\lambda^2)$. Therefore, the algorithm converges rapidly, with a very few number of iterations, and the error quickly approaches zero.

The most computationally expensive steps in the above cyclic reduction algorithm consists of solving the system of linear equations for H and K in step 3 and the system of linear equations for Y in step 5, i.e., computing the inverse of the matrix $(I - V_0)$ at each iteration of step 3 and computing the inverse of the matrix $(I - \widehat{V})$ in step 5. Upon replacing the above exact computations with an iterative solver, the best-known computational complexity for classical algorithms to iteratively solve such linear systems is $O(N\kappa s \log(1/\epsilon))$, where $N = M_+$ is the number of equations, s and κ are respectively the number of nonzeros and the condition number of the matrix, and $\epsilon > 0$ is the desired level of accuracy (or error tolerance). To reduce the computational complexity of these expensive steps, we consider developing quantum algorithms to solve these systems of linear equations on a quantum computer in a more efficient manner. One such approach is based on exploiting and tailoring quantum algorithms that can solve such linear systems with a computational complexity of $O(s\kappa \text{polylog}(s\kappa/\epsilon))$. We next present several of these approaches that realize significant reductions in the overall computation complexity (including exponential speedup) of obtaining the equilibrium distribution π of structured random walks.

Throughout what follows, we shall use ϵ to represent the desired level of accuracy irrespective of whether the computations are performed on a digital computer or a quantum computer.

3.5.2 Quantum Computing Algorithm: Linear System Solver

An important result in recent years in the quantum computing literature has been the Quantum Linear System algorithm (QLSA) [11] [12], also known as the Harrow-Hassidim-Lloyd (HHL) algorithm. The HHL algorithm reduces the problem of solving a system of linear equations into the problem of computing matrix inversion, and in doing so offers an exponential speedup for solving the linear system problem. In particular, given a linear system of N equations $Ax = b$, where $A \in \mathbb{C}^{N \times N}$ is a coefficient matrix, $b \in \mathbb{C}^N$ a solution vector and x the unknown vector to be obtained, the HHL algorithm solves the corresponding quantum version of this problem known as QLSP [12]. If the matrix A is invertible, we know that the solution for this linear system is given by $x = A^{-1}b$. In QLSP, given a quantum state on $\lceil \log N \rceil$ qubit $|b\rangle$ corresponding to b , namely

$$|b\rangle := \frac{\sum_i b_i |i\rangle}{\|\sum_i b_i |i\rangle\|_2} \quad (97)$$

where b_i is the i th component of $\mathbf{b}$, then the problem is to output a state $|x\rangle$ corresponding to the solution $\mathbf{x}$ such that

$$|x\rangle := \frac{\sum_i x_i |i\rangle}{\|\sum_i x_i |i\rangle\|_2}, \quad (98)$$

with x_i the i th component of $\mathbf{x}$. The HHL algorithm outputs a state $|\tilde{x}\rangle$ such that $\| |\tilde{x}\rangle - |x\rangle \|_2 \leq \epsilon$ with high probability, for a predefined tolerance ϵ .

The HHL algorithm depends on a number of quantum subroutines and a quantum RAM (qRAM) [13] memory structure. This includes a Hamiltonian simulation technique [7] in order to encode the Hermitian matrix A as a Hamiltonian operator on a given state $|\psi\rangle$, i.e., $|\psi\rangle \rightarrow \exp(-iAt)|\psi\rangle$ for some $t > 0$, where the simulator in turn exploits the qRAM. The HHL algorithm also includes Quantum Phase estimation (QPE), Quantum Fourier Transform (QFT) and Amplitude amplification, each of which are discussed in other sections of the present report. This algorithm involves three major steps, namely phase estimation, controlled rotation, and uncomputation.

Suppose $A = \sum_j \lambda_j |u_j\rangle\langle u_j|$, where λ_j are the eigenvalues of A and u_j are the corresponding eigenvectors of A . Further suppose the initial state is an eigenvector $|b\rangle = |u_j\rangle$. Then, the first step is to use the Hamiltonian simulator and Phase estimation to implement the mapping

$$|0\rangle|u_j\rangle \rightarrow |\tilde{\lambda}_j\rangle|u_j\rangle, \quad (99)$$

where $\tilde{\lambda}_j$ is the binary representation of λ_j . The second step is to use a controlled rotation [12] conditioned on $|\tilde{\lambda}_j\rangle$ in order to produce a normalized state

$$\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} |\tilde{\lambda}_j\rangle|u_j\rangle|0\rangle + \frac{C^2}{\tilde{\lambda}_j^2} |\tilde{\lambda}_j\rangle|u_j\rangle|1\rangle, \quad (100)$$

where C is a constant of normalization.

Now suppose we express the state $|b\rangle$ in terms of the eigenbasis $\{|u_j\rangle\}$ as $|b\rangle = \sum_j \beta_j |u_j\rangle$. Then using $|b\rangle$ as the starting state and applying the first two steps yields

$$\sum_{j=1}^N \beta_j |u_j\rangle \left(\sqrt{1 - \frac{C^2}{\tilde{\lambda}_j^2}} |\tilde{\lambda}_j\rangle|u_j\rangle|0\rangle + \frac{C^2}{\tilde{\lambda}_j^2} |\tilde{\lambda}_j\rangle|u_j\rangle|1\rangle \right). \quad (101)$$

Since $A^{-1}|b\rangle = \sum_{j=1}^N \frac{\beta_j}{\lambda_j} |u_j\rangle$, we can obtain a state close to $|x\rangle$ using an uncompute operation [12] and Amplitude amplification on the outcome ‘1’.

Computational Complexity and Improvements. It has been shown that the HHL algorithm has a computational complexity of $O(\log(N)s^2\kappa^2/\epsilon)$, and therefore our

exploitation of this algorithm to compute the linear system solving problems at each iteration of step 3, as well as in step 5, of the cyclic reduction algorithm will be exponentially faster in the size of the linear system N in comparison to the classical computation of these steps in the cyclic reduction algorithm. This speedup can be improved further by exploiting a number of the recent improvements to the runtime of the HHL algorithm proposed in the literature [12]. For example, [14] improved the runtime of QLSP to have a computational complexity of $O(sk\text{polylog}(sk/\epsilon))$.

We also note that the cyclic reduction algorithm is at the core of a number of important numerical methods that go well beyond our interests here in structured random walks, and thus the quantum algorithm proposed herein can be exploited to address a much larger class of numerical problems.

3.5.3 Quantum Phase Algorithm and Eigensolver

A primary drawback of (naively) using the HHL solver in the cyclic reduction algorithm as described above is the requirement that we have to apply the HHL algorithm N times either in series or in parallel, thus either increasing the computational cost by a factor of N (though still linear) or increasing the number of qubits required for the computation by a factor of N . To address this issue, we next consider an alternate approach based on a particular use of a quantum phase algorithm.

It is well known that QPE methods can be used to compute the eigenvalues of a given matrix. Indeed, many Quantum Eigensolvers exist that effectively compute the eigenvalues of a given matrix; see, e.g., [15] and some of the related contributions in this report. Quantum singular value estimation procedures also exist [16] to rapidly estimate singular values of matrices. Hence, one alternative approach consists of exploiting a quantum eigensolver, say a variational quantum eigensolver, to compute the eigendecomposition of the matrix, and then computing the matrix inverse either using the rotation idea of HHL or doing so classically. This approach, however, will still be linear in N since the matrix is full rank (non-singular), and thus a variational quantum eigensolver will have to be applied N times to compute all eigenvalues.

We instead devise an alternate approach to compute step 3 of the cyclic reduction algorithm, namely solving the matrix inversion and multiple right-hand-side equation. The main idea is to exploit a quantum phase algorithm to decompose the right-hand-side matrix V_1 into the eigenstates of $(I - V_0)$, and then leverage the controlled rotation of HHL to compute its inverse. More specifically, we first employ qRAM loading to prepare the right-hand-side matrix V_1 as a mixture state using an ancilla register; see [17] for details. The computational cost of this step is $\log(N) * s$, where s is the row sparsity of the matrix. We next exploit the quantum phase algorithm of HHL to decompose the prepared mixture state into the eigenstates of $(I - V_0)$, and then leverage the controlled rotation step of the HHL algorithm to compute the inverse. The diagonalization and rotation have computational

cost on the order $O(\log^5(N)/\epsilon)$ [17] where ϵ is the predefined HHL approximation error tolerance. This approach now will be linear in the row sparsity s , which renders a lower computational cost per iteration than the previous two approaches when the matrix V_1 is sparse, as is typically the case for structured random walks.

The above quantum algorithm to compute the expensive matrix inversion step of the cyclic reduction algorithm yields a significant improvement in the computational complexity of each iteration. Meanwhile, we note that this quantum algorithm will introduce an additional ϵ approximation error that needs to be taken into account. On the other hand, given that the error in the cyclic reduction algorithm reduces exponentially, the convergence rate of the cyclic reduction algorithm will not be affected as long as the additional approximation error ϵ of the quantum algorithm is selected to be sufficiently small. More specifically, we can express the impact of this approximation error as

$$\begin{aligned} \|G_{\min} - (I - \hat{A}_0^{(n)})^{-1}A_{-1}\| &= \| (I - \hat{A}_0^{(n)})^{-1}A_1^{(n)}G_{\min}^{2^n+1} \| \\ &\leq \| (I - \hat{A}_0^{(n)})^{-1}A_1^{(n)} \| \|G_{\min}^{2^n+1}\| \text{ where } \epsilon \text{ is a} \\ &\leq (\| (I - \hat{A}_0^{(n)})^{-1} \| + \epsilon) \|A_1^{(n)}\| \|G_{\min}^{2^n+1}\|, \end{aligned}$$

(small) constant that reflects the quantum error of computing the matrix inversion step of the cyclic reduction algorithm. We know that $\|A_1^{(n)}\| \leq \xi$ and $\|G_{\min}^{2^n+1}\| \leq \gamma \left(\frac{1}{\sigma}\right)^{2^n}$, for some $\gamma > 0$ [10]. Hence, with ϵ chosen to be an appropriately small constant such that $\epsilon + \sigma^{-1} < 1$ and with $\xi + \sigma^{-1} < 1$, the convergence rate is not affected and the approximation error continues to be given by the expression in Section 3.5.1, leading to convergence of the algorithm in a very few number of iterations with the error quickly approaching zero.

3.6 Algorithms - Quantum Walks

Consider any classical system undergoing random transitions between a discrete set of possibilities. Let a column vector p with each term nonnegative and summing to one define a probability distribution over these states. For each possible state, there is a corresponding column vector that provides the probability of transition to every other state. Accordingly, transitions to another state via a random process can be described by a Markov transition matrix which we will denote here by P , where the ij th element of P is given by $P_{ij} = P(i|j)$.

Following the dynamics of the stochastic process, we can write the state as

$$P\pi(t) = \pi(t+1) \tag{102}$$

for example, if the process is a simple random walk on a one-dimensional lattice, then P will naturally be the discrete adjacency matrix

$$P = \begin{pmatrix} \frac{1}{2} & 0 & \frac{1}{2} & & \\ & \frac{1}{2} & 0 & \frac{1}{2} & \\ & & \ddots & \ddots & \ddots \end{pmatrix} \quad (103)$$

and a particle at position n will transition to positions $n - 1$ and $n + 1$ each with probability $\frac{1}{2}$. Recall that this one-dimensional random walk is equivalent to the example in Section 4.4.4 in which $p_U = p_D = 0$ and $p_L = p_R = \frac{1}{2}$.

The dynamics generated by such a Markovian process are actually very similar to the well-known quantum dynamics generated by the Schrödinger evolution operator

$$|\psi(t + 1)\rangle = U|\psi(t)\rangle \quad (104)$$

where ψ is a square integrable vector in Hilbert space $\langle\psi(t)|\psi(t)\rangle = 1$ and U takes the role of the Markov propagation. In fact, one can attain the classical limit by performing projective measurement (in the steady state regime) onto basis state after each propagation step, thus turning U into a generator of a classical stochastic process with the limit

$$P_{ij} = U_{ij}^* U_{ij} = |U_{ij}|^2. \quad (105)$$

In fact, this “classical limit” is the natural way in which a quantum operation suggests a random walk. We call such a quantum system, when thought of in the context of the random walk, a *quantum walk*.

As an illustrative example let us look at how a quantum walk is defined on a one dimensional lattice, and focus on just one lattice point. The probability of finding the particle at position n at time T , $p(n, T)$, is zero if $n + T$ is odd and otherwise given by

$$p(n, T) = \frac{1}{2^T} \binom{T}{\frac{1}{2}(n + T)}, \text{ if } n + T \text{ even} \quad (106)$$

with the following moments $\langle n \rangle = 0$, $\sigma^2 = \langle (n - \langle n \rangle)^2 \rangle = T$.

The quantum walk is defined in the Hilbert space $\mathcal{H}_s \otimes \mathcal{H}_c$. The space $\mathcal{H}_s$ is a discrete position states $|n\rangle$ of dimension n while $\mathcal{H}_c$ is a “coin” space of size two, in charge of the transition actions. To symbolize the actions, we label a set of orthonormal basis vectors in $\mathcal{H}_c$ as $|+\rangle$ and $|-\rangle$ with the associated “coin” action C for the walk “decision” given by

$$\begin{cases} C|+\rangle = a|-\rangle + b|+\rangle \\ C|-\rangle = a|-\rangle - b|+\rangle \end{cases} \quad (107)$$

and the shift operator S in $\mathcal{H}_c$ associated with the walk “step” given by

$$\begin{cases} S|n, +\rangle = |n-2, -\rangle \\ S|n, -\rangle = |n+2, +\rangle \end{cases} \quad (108)$$

In fact, the evolution of the quantum walk-in one-time step is therefore given by $U = S(I \otimes C)$.

One instance of a coin operator is the Hadamard gate

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad (109)$$

or for the balanced coin

$$Y = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & i \\ i & 1 \end{pmatrix} \quad (110)$$

while the shift operator embodies the topological structure of the graph, so for a one-dimensional lattice topology (line), we have

$$S = \sum_{n=-\infty}^{\infty} (|n+1\rangle\langle n| \otimes |+\rangle\langle +| + |n-1\rangle\langle n| \otimes |-\rangle\langle -|) \quad (111)$$

So, for example using the Hadamard coin operator will result, for an initial state $|\psi\rangle = |0, +\rangle$ in the following dynamics

$$\begin{aligned} |\psi\rangle &\xrightarrow{U} \frac{1}{\sqrt{2}}| -1, -\rangle - \frac{1}{\sqrt{2}}|1, +\rangle \\ &\xrightarrow{U} \frac{1}{2}| -2, -\rangle - \frac{1}{2}|0, -\rangle + \frac{1}{2}|0, +\rangle - \frac{1}{2}|2, -\rangle \\ &\xrightarrow{U} \frac{1}{2\sqrt{2}}| -3, -\rangle + \frac{1}{2\sqrt{2}}| -1, +\rangle + \frac{1}{\sqrt{2}}| -1, -\rangle - \frac{1}{2\sqrt{2}}|1, -\rangle + \frac{1}{2\sqrt{2}}|3, +\rangle \end{aligned} \quad (112)$$

With this example we already see the onset of quantum effects. After two steps of the operator U , the probability of measuring the particle at any position remains the same as for a classical random walk, but on the third step, the probabilities change.

Positive moving amplitudes at $n = 1$ interfere destructively simultaneously with constructive interference at $n = -1$, this leads to a probability $\frac{5}{8}$ of the particle being measured in the state $| -1 \rangle$ but only a probability $\frac{1}{8}$ at $|1\rangle$.

More strikingly the difference in the choice of the coin operator is shown in Figure 7 and Figure 8 where the former shows a probability distribution for a biased quantum random walker and the latter shows a probability distribution for an unbiased quantum (solid curve) and classical (dashed curve) random walker. One can see the probability flux generated by the bias in the Hadamard gate (Figure 7) and the symmetric probability measure in the fair coin in which amplitude is distributed symmetrically between different sections of the lattice (Figure 8).

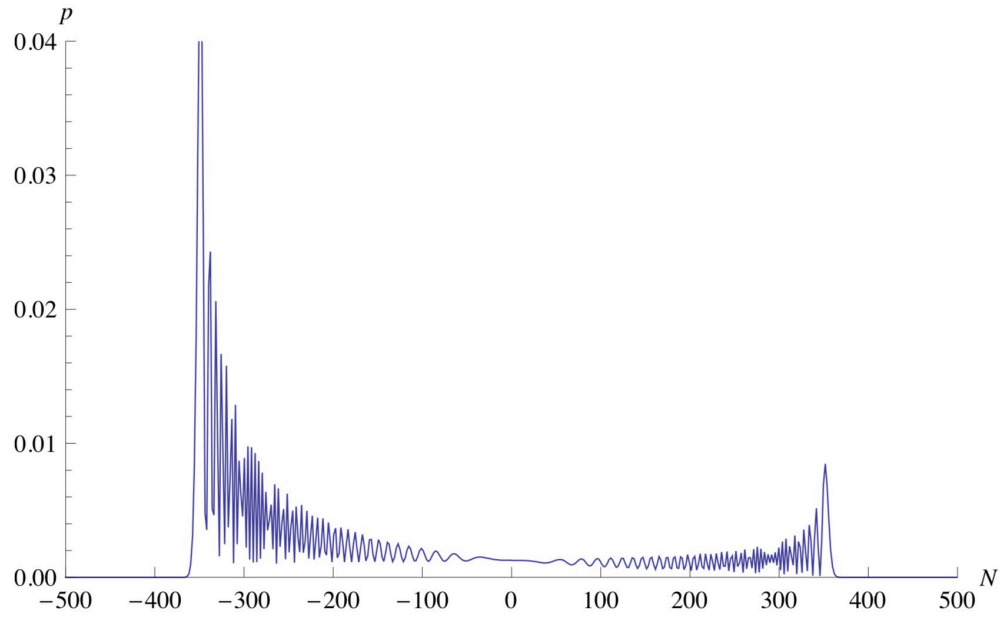


Figure 7. Probability distribution for a biased quantum walker.

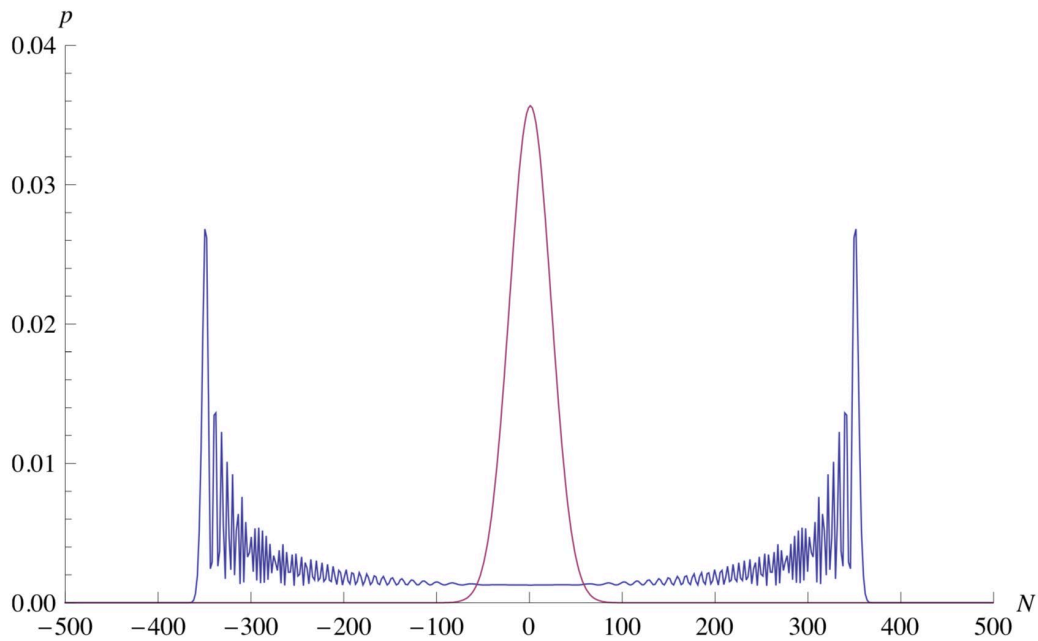


Figure 8. Probability distribution for an unbiased quantum (blue) and classical (red) random walker.

We note that the steady state distribution of the one-dimensional random walk can also be obtained through the use of the algorithms described in section 3.4. As previously pointed out, the above example is equivalent to the random walk in Section 3.4.4 with $p_U = p_D =$

0 and $p_L = p_R = \frac{1}{2}$. The equilibrium distribution p is then obtained by solving for the stationary distribution π of Section 3.4 using the computational algorithms described therein. An interesting research question that we are pursuing is to compare the efficiency and efficacy of these two different ways of obtaining p for the random walk, both for the one-dimensional case and in general.

3.6.1 Quantum Walk-Based Clique Detection Algorithms

Network theory is a field of research that is spreading to many disciplines, such as physics, engineering, biology, sociology and economics, where it plays a fundamental role. Most interaction systems in nature can be constructed or thought of in terms of a graph. A graph includes a finite set of items, referred to as “nodes” or “vertices,” which are connected by edges. An edge connects two adjacent nodes. A clique is a subset of the nodes wherein every two distinct nodes are adjacent. Most versions of the clique problem are known to be computationally hard and the clique decision problem is known to be NP-complete.

Finding cliques on a graph has widespread applications across the many disciplines in which network theory can be applied, including social media, logistical problems, networks, and communication. For example, efficient logistics rely on an optimal way to distribute products and goods. Inferring cliques and moving distribution to localized regions can have great economic value. Any computational advantage in finding cliques can have great repercussions and can be broadly applied. We consider here algorithms based on quantum walks for community clique detection. Quantum random walk expands much more quickly than the classical random walk, enabling large information areas to be explored, and cliques to be detected.

In the same way that we have defined above the quantum walk on a line, we can expand the space to an arbitrary undirected graph by using different position and coin spaces. Similar to the quantum walk on a line, we can define the process on a composite Hilbert space

$$\mathcal{H}_s \otimes \mathcal{H}_c \quad (113)$$

where, as before, the position space has discrete values and is the dimension of the number of nodes, but the coin space is now of size n where n is the number of paths connected to a node

$$\text{state} = |v, i\rangle \quad (114)$$

$$S|v, i\rangle \geq |v', i'\rangle \quad (115)$$

where v' is the other vertex attached to edge marked by i , and i' is the labeling for the index associated with the same edge marked by i .

In general, the coin operator C may also be chosen arbitrarily, for example the discrete Fourier transform (DFT) (which is natural extension of the Hadamard coin). Another possible operator is the extension of the fair coin operator Y

$$\begin{pmatrix} -1 + \frac{2}{n} & \frac{2}{n} & \dots & \frac{2}{n} \\ \frac{2}{n} & -1 + \frac{2}{n} & & \vdots \\ & \frac{2}{n} & -1 + \frac{2}{n} & \\ \vdots & \vdots & & \ddots \\ \frac{2}{n} & & \frac{2}{n} & -1 + \frac{2}{n} \end{pmatrix} \quad (116)$$

which is the diffusion operator G from Grover's search algorithm,

$$G = -I + 2|\psi\rangle\langle\psi| \quad (117)$$

where ψ is the basis of the coin space

$$|\psi\rangle = \frac{1}{\sqrt{n}} \sum_{i=1}^n |i\rangle. \quad (118)$$

Quantum walks are at the heart of modern quantum technologies. They allow us to deal with quantum transport phenomena and are an advanced tool for constructing novel quantum algorithms. Quantum walks on graphs are fundamentally different from their classical random walk analogs. In particular, they walk faster than classical random walks on certain graphs, enabling in these cases quantum algorithmic application. In recent years there has been an increase in the use of quantum and classical walkers as novel algorithms for search and satisfiability problems. This stems from the fact that, depending on certain symmetries and local structures, a quantum walker can achieve a polynomial or even exponential speed up. The fact that speedups are dependent on local structure means quantum random walkers can be used as a probing mechanism to such local structures by measuring expansion coefficients or other related observables. Such local structures are deeply interconnected with the formation and characterization of cliques.

Formally, given an undirected graph $G = (V, E)$, a clique C is a subset of the vertices, $C \subseteq V$, such that every two distinct vertices are adjacent. The computational problem of finding cliques in a graph can have several different formulations depending on which cliques, and what information about the cliques should be found. Most versions of the clique problem are known to be computationally hard, for example the clique decision problem is known to be NP-complete. Since cliques arise in a multitude of settings, such as social networks, logistics, biological communities, and can hold the key to real-world large-scale applications, there is a clear interest in more efficient algorithms to detect them.

One method for exploring such structures is by looking at the different powers of the adjacency matrix. For an adjacency matrix A , the a_{ij} term is an indicator function of the path between i and j . The n^{th} power of the adjacency matrix A^n has a special meaning where the a_{ij} term is the number of paths of stride n between i and j . Once the adjacency matrix is raised to the different powers, one can gain diagonal terms in the new adjacency matrix. Such terms a_{ii} are the numbers of paths from i back to i and in fact are an indication of loops. By taking different powers and seeing how the spectrum or the trace of the adjacency matrix grows, we can gain the indication of the local structure.

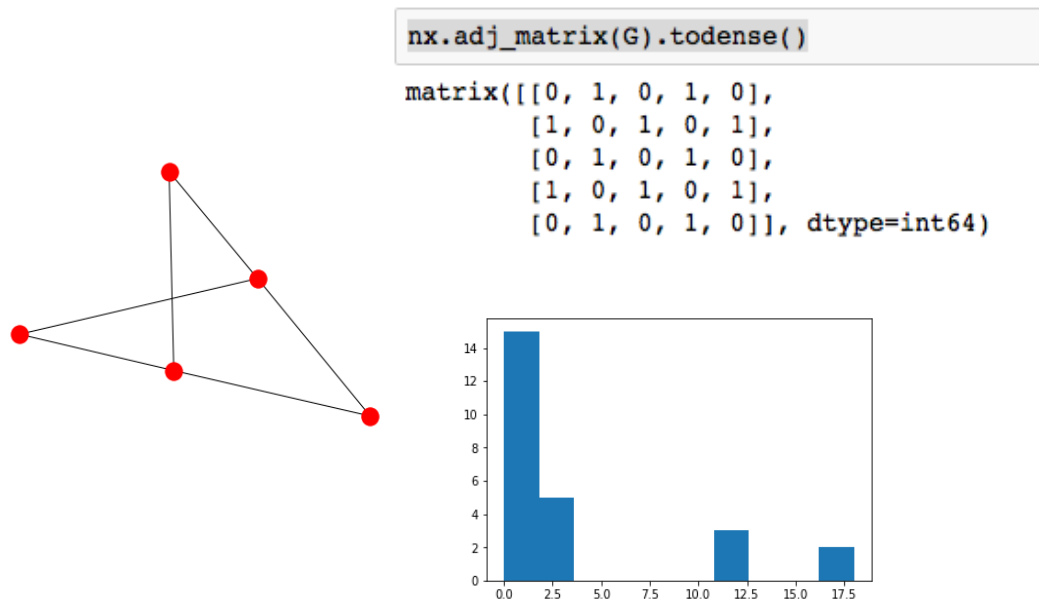


Figure 9. Illustrative example of clique detection using classical random walk

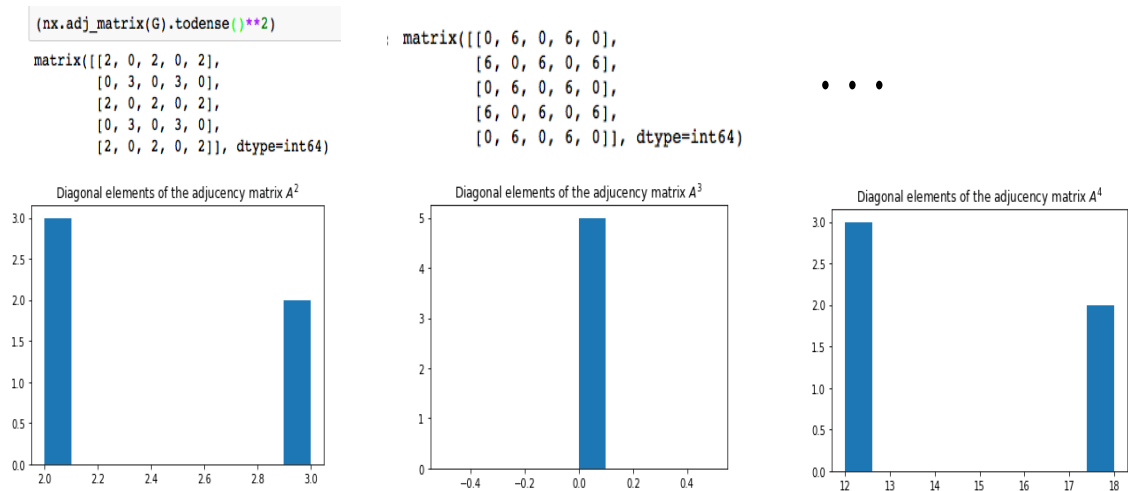


Figure 10. Illustrative example of clique detection using quantum walk

We present and discuss an illustrative example with the use of Figure 9 and Figure 10. Figure 9 shows an example graph (left) and the corresponding adjacency matrix (right). By looking at the trace of the adjacency matrix, one can obtain probabilities for different cliques. The bar graph in Figure 9 is a plot of the diagonal elements of an example adjacency matrix (not shown).

Figure 10 illustrates how element a_{ij} of the n^{th} power of the adjacency matrix yields the number of paths of length n between i and j . The figure shows the square of the adjacency matrix (A^2) (left) and the cube of the adjacency matrix (A^3) (middle). The diagonal elements of the square of the adjacency matrix are plotted in the histogram below the matrix (left). The diagonal elements of the cube of the adjacency matrix are plotted in the histogram below the corresponding matrix (middle), and the diagonal elements of the fourth power of the adjacency matrix are plotted in rightmost histogram.

The action of taking the powers of the adjacency matrix can be mapped towards consecutive operations of the quantum shift operator in the random walk. Coupling this to a spectrum inference mechanism such as standard or learning-based quantum phase estimation can give an exponential speed up with respect to a local cycle detection leveraging the classical quantum hybrid approach. The classical graph is encoded into a quantum object by a preparation circuit. The quantum random walk circuit performs the clique analysis by repeated applications of the walk operator (combination of the coin operator and step operator). The measurement circuit passes the quantum information from the qubits to the classical world for analysis.

More specifically, this approach provides a method for the detection of cliques on a graph by quantum computation in which a quantum processor takes as input the graph structure of interest from a user and outputs realizations associated with the probability of a clique. The quantum computation involves the design of a quantum circuit which first encodes the structure of the graph and then performs parallel evaluations of the different paths on it. The paths are the different routes that can be traversed on the graph. The parallel path traverse is done by means of a circuit embodying a quantum random walk. The clique analysis can be performed by repeated application of the quantum random walk circuit. The quantum walk encodes in its spectrum the clique analysis. Eigenvalues (of the spectrum of the random walk) of higher powers of the shift followed by the quantum coin operator provide a measure of self-cliques. Consequent to the quantum walk sequence, a measurement stage follows, where the information is passed to the classical world for analysis. This quantum computational approach can lead to polynomial computational speedup over the classical analogs.

The n^{th} power of the adjacency matrix can be used to gain information about cliques in the graph. For example, the n^{th} power of the adjacency matrix can be used to find an upper bound on the clique number of the graph. The n^{th} root (computed classically) of the trace of the n^{th} power of the adjacency matrix (computed using the random walk circuit) is one

way to estimate the leading eigenvalue. The leading eigenvalue can be used to calculate an upper bound for the clique number. To detect longer cliques, one can compute the trace of various powers of the adjacency matrix. The random walk circuit and measurement circuit may be run multiple times, and an average may be taken over the measurement outputs. The number of times that the random walk circuit and measurement circuit are executed may depend on the desired precision level ϵ . The number of samples would generally grow like $O(1/\epsilon^2)$.

Another approach to clique detection, as an alternative to the above quantum walk approach, can be based on utilizing Grover's search algorithm by first performing a pre-processing step using a quantum circuit (i.e., an "oracle") built to detect maximal cliques. This circuit is given a set of nodes and determines whether they form a maximal clique or not by computing the bit-wise multiplication of their respective columns in the adjacency matrix associated with the input graph. If the output vector is the same as the input vector, then the given set of nodes indeed forms a maximal clique. Nonetheless, the running time of the overall Grover search scheme is $O(\sqrt{n^2 2^n})$ which is not a big improvement over the $O(n^{n/3})$ running time of the worst-case complexity introduced by a well-known classical approach.

3.6.2 Computational Complexity

We now consider the computational complexity of the foregoing Quantum Walk algorithms to compute the equilibrium distribution of the corresponding Markov process. Applying the above Quantum Walk algorithms to compute the equilibrium distribution consists of repeated applications of the coin and shift operators, where various coins are available to us (e.g., Hadamard, Fourier, Grover). This, however, renders a quasi-periodic pattern in the time evolution of the Quantum Walk that prevents convergence to the equilibrium distribution (analogous to that of an irreducible periodic Markov process). Instead, one obtains the *average probability distribution* that approaches the equilibrium distribution according to the mixing times associated with the Quantum Walk.

More precisely, given that the quasi-periodic pattern in the time evolution of a finite Quantum Walk prevents convergence to the equilibrium distribution, we seek to obtain the average probability distribution that approaches the equilibrium distribution over time. Let $\tilde{\pi}_n(t)$ denote the probability of the Quantum Walk being in node $n = 1, \dots, N$ at time-step t , dependent upon the initial state of the Quantum Walk, where N denotes the number of nodes in the Quantum Walk. Then the average probability distribution is essentially a time-average of the behavior of the Quantum Walk, defined for each node n at each time-step T as $\tilde{\pi}_n(T) = \frac{1}{T} \sum_{t=0}^{T-1} \tilde{\pi}_n(t)$ where $\sum_{n=1}^N \tilde{\pi}_n(t) = 1$ for all $t \geq 1$. We know that this average probability distribution asymptotically approaches the equilibrium distribution over the time steps, or more formally we have $\pi_n = \lim_{T \rightarrow \infty} \tilde{\pi}_n(T)$. The mixing time, denoted by $T_{mix}(\epsilon)$, captures how quickly the average probability distribution approaches the equilibrium distribution, or more formally we have $T_{min}(\epsilon) = \min\{T \mid D(\tilde{\pi}_n(t), \pi_n) \leq$

$\epsilon, \forall t \geq T, \forall n\}$ where $D(\cdot, \cdot)$ is a distance metric (such as total variation) and ϵ is the predefined error tolerance.

Once the Quantum Walk has run sufficiently long to achieve the desired level of accuracy, then the equilibrium distribution obtained through the average probability distribution can be used directly for various algorithms of interest and for other purposes. Examples include applications in computer vision and applications based on generating random elements in large and complicated sets (such as the set of lattice points in a convex body).

3.7 Algorithms - Active Learning-Quantum Amplitude Estimation

Quantum Phase Estimation [2] and Variational Quantum Eigen (VQE) solvers [1] are instrumental for the computation of eigenvalues. Quantum Phase Estimation may offer an exponential advantage over classical computation, in simulation of the evolution of the state of n qubits by matrix multiplication. The algorithm has been used extensively in applications such as the solution of linear systems of equations [18], factorization, eigenvalue decomposition, and more. A key issue for realizing the potential of QPE is the requirement that the circuit depth scales like $\mathcal{O}(\frac{1}{\epsilon})$ in order to attain ϵ precision. A popular alternative is the VQE algorithm that replaces the long coherence requirement of QPE with multiple shorter coherence time expectation realizations. The drawbacks of this algorithm are the requirement to form a proficient variational form for the representation of the eigenstate, and the cost of computation in terms of sample counts (which grow like $\mathcal{O}(\frac{1}{\epsilon^2})$ for ϵ error, while each realization requires preparation of the state over again).

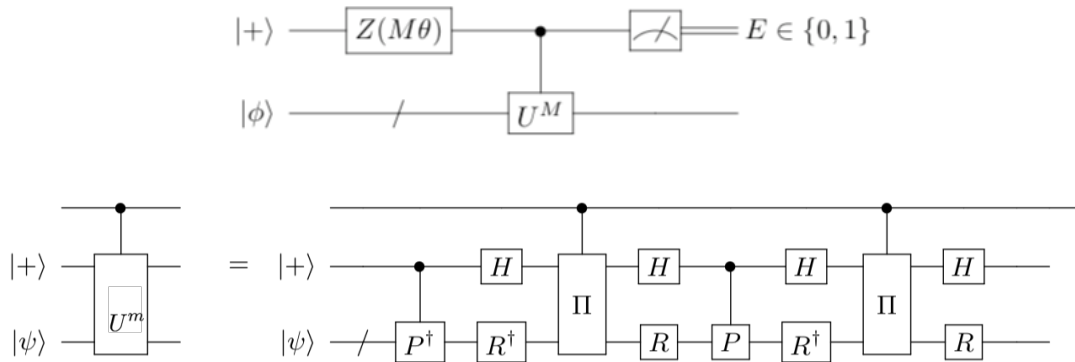


Figure 11. Rejection Filter Quantum Phase Estimation circuit

Although this proposition can provide benefit from a quantum system fidelity standpoint, as the performance of quantum devices improves over time, this may result in underutilization.

β -QPE proposes to continuously interpolate between the full QPE and VQE. The first step towards realizing this goal is an improvement of the Quantum Amplitude Estimation component of VQE. Existing techniques for Quantum Amplitude Estimation do indeed perform much better than the naïve VQE component. This is the reason for the worsening performance of VQE mentioned above in terms of precision. In contrast, the more

sophisticated asymptotic approaches are able to obtain the so called Heisenburg limit by either using entanglement or tantalizingly equivalently by using feed-forward iterative prepare and measure schemes. Unfortunately, these more sophisticated schemes still suffer from the problem of preparing an eigenstate on which to perform amplitude estimation.

We propose a novel method to circumvent this limitation while hopefully still maintaining the desired Heisenburg limit performance. This novel method begins with Knill's Grover Operator, but circumvents the need to collapse into an eigenstate, by only performing operations that are invariant to the precise eigenvector.

We combine Knill et al.'s Quantum Expectation Estimation algorithm with Wiebe et al.'s Bayesian Phase Estimation Algorithm to perform Quantum Bayesian Amplitude Estimation. Furthermore, we extend this novel combination to make use of multiple noisy qubits as they become available on the road to fault tolerant computation. Even if fault tolerant quantum computers become available, a Bayesian approach is still expected to be useful, for multiple reasons including: incorporating prior information (e.g. discrete Betti numbers) to speed up calculations and extracting the most out of the available qubits).

A second method of overcoming the need to prepare an exact eigenstate is to use an advanced Bayesian scheme proposed by an internal IBM Research collaborator, Ewout van den Berg. Here we track multiple eigenvalues at the same time and with a sophisticated posterior expansion.

With the above novel reframing and the use of van-den Berg's framework, we not only no longer need to worry about preparing a single eigenstate, we also need not restrict ourselves to choices that are invariant to the eigenvalue.

We expect this second scheme to perform even better than the first. The remaining step is to experiment with the code provided by van den Berg.

3.7.1 Introduction

The calculation of the expectation of a matrix A with respect to a quantum state $|\psi\rangle$, where A is either a unitary or Hermitian matrix, is a very common and important calculation in quantum physics and quantum information theory. For example, the average energy of a quantum state undergoing evolution governed by a Hamiltonian H is $\langle\psi|H|\psi\rangle$ or the fraction of a state in the subspace of the image of a projection operator, P is $\langle\psi|P|\psi\rangle$.

Current methods of calculating the expectation of A involve eigenvector projections or sampling algorithms.

Knill et al. introduce a sampling method with optimal statistical performance. Furthermore, their approach allows for a clever iterative version that calculates the expectation one bit at a time. This reduces the required number of qubits but does not handle noisy qubits in a principled way.

Wiebe et al. introduce a Bayesian framework for eigenvalue estimation that handles a low number of noisy qubits. However, this approach is not applied to expectation estimation and requires knowledge of the eigenvectors of an operator. Furthermore, there is no scheme

for moving between a few number of qubits to increasingly many. We propose to combine the above two methods, quantum expectation estimation and Bayesian phase estimation to achieve a novel, iterative, Bayesian expectation estimation algorithm that can successfully use multiple noisy qubits as they become available.

3.7.2 Summary of Knill et al.'s Quantum Expectation Estimation

Goal

The goal of Quantum Expectation Estimation is to calculate the expectation of a matrix A (whether unitary or Hermitian) with respect to a state $|\psi\rangle$:

$$\langle\psi|A|\psi\rangle \quad (119)$$

When A is a unitary U , this is just the *complex* inner product between two states:

$$\begin{aligned} |\psi_1\rangle &= U|\psi_0\rangle \\ \langle\psi_0|U|\psi_0\rangle &= \langle\psi_0|\psi_1\rangle \end{aligned} \quad (120)$$

When A is a Hermitian matrix, H , the expectation is a real number, and can be cast into a unitary form by evolving for a very short time with a unitary $U = e^{iHt}$. Therefore, it suffices to only consider the unitary case $A = U$.

Furthermore, if in some setup, all we can calculate is the magnitude of the expectation of an arbitrary U with respect to an arbitrary $|\psi\rangle$ (including augmenting with ancillas), then the restriction of only being able to measure the real magnitude does not hinder us from extracting the full complex expectation. All we have to do is run three separate experiments, making use of the freedom to adjust U and $|\psi\rangle$ and stitch the results together. Therefore, in order to calculate the full expectation (119), it suffices to only know how to calculate the, so called, *amplitude* of a unitary expectation, corresponding to the magnitude of the inner product between two states:

$$|\langle\psi_0|U|\psi_0\rangle| = |\langle\psi_0|\psi_1\rangle| \quad (121)$$

Statistical Performance

The standard quantum statistical limit (also called shot-noise, classically) states that when performing repeated independent measurements, the precision of the sample mean (in approximating the true mean) improves as $1/\sqrt{N}$, where N is the number of measurements. This result is a simple classical statistics result. If we somehow go beyond the independence assumption, this can be improved to $1/N$, called the *Heisenberg* limit. Examples of techniques that work beyond the independence assumption is *sketching* or Bayesian Variational Trace Methods. In the quantum setting this can be achieved by

making use of entanglement between measurements or as in Knill et al.'s case by feeding-forward the results of past measurements into future measurements!

Amplitude Estimation

A trick to doing Amplitude Estimation that saturates the Heisenberg limit is to use Quantum Phase Estimation. Quantum Phase Estimation employs entanglement (via the inverse Quantum Fourier Transform) between repeated applications of U to extract the phase. An iterative version of Quantum Phase Estimation exists that uses feed-forward measurements instead of entanglement.

In order to use Phase Estimation to do Amplitude Estimation, we need two seemingly difficult things: firstly, we need to get the amplitude encoded into the phase and secondly, we need to define an operator for which we know the eigenvectors of (since Phase Estimation relies on its preparation). The second is made even more difficult when we realize that the U from our task of Amplitude Estimation is arbitrary and we do not know its eigenvalues nor eigenvectors a-priori.

The solution to the second challenge is to build a new *rotation* operator, S (related to U), that by design leaves a certain subspace invariant. Since S is a two-dimensional rotation we know the form of the two eigenvalues: $e^{\pm i\phi}$. Furthermore, since there are only two eigenvalues which differ only with respect to the **signs** of their phases, as long as we prepare our initial state within the subspace, we can learn “both” phases in **one** run of full Quantum Phase Estimation.

Knill et al. try to go further and design an iterative phase algorithm to replace the full phase algorithm. The authors try to leverage the invariance of the sub-space and the similar eigenvalue forms to propose an iterative phase algorithm that does not require knowledge of the eigenvectors of S , but merely that the initial state starts in the subspace. Unfortunately, this does not seem to be borne out in the calculations because the feed-forward phase compensation would have a different effect depending on the eigenvector component. Thus, Knill's iterative algorithm seems to require projection onto one of the eigenvectors.

Nevertheless, we think that we can use this dual eigenvalue property in a Bayesian iterative context.

Therefore, continuing with Knill et al.'s approach, the following operator has both the above desired properties, a rotation by $\phi = |\langle\psi|U|\psi\rangle|$ that operates only in a two-dimensional subspace (spanned by $|\psi_0\rangle$ and $|\psi_1\rangle$):

$$\begin{aligned} S &= S_0 S_1 \\ S_0 &= \mathbb{I} - 2|\psi_0\rangle\langle\psi_0| \\ S_1 &= \mathbb{I} - 2|\psi_1\rangle\langle\psi_1| = \mathbb{I} - 2U|\psi_0\rangle\langle\psi_0|U^\dagger \end{aligned} \tag{122}$$

Where $S_x|\psi\rangle$, $x \in \{0,1\}$ is a sign flip of the component of $|\psi\rangle$ in the direction of $|\psi_x\rangle$. Each S_x operates in a two-dimensional subspace spanned by the state being acted upon and $|\psi_x\rangle$. Therefore, if we start in the sub-space spanned by $|\psi_0\rangle$ and $|\psi_1\rangle$, we remain there.

A sign flip of the component in the direction of a vector is a reflection around the line perpendicular to the vector. This allows us to think in terms of reflections about axes. Two reflections around the same axes returns a vector to where it began. Two reflections about two non-coinciding axes rotates the vector by exactly twice the angle between the two axes (to see this, note how reflection about the second axis can be seen as rotation to the first axis, followed by reflection about the second axis, picking up a rotation of twice the angle between the axis followed by a rotation of the same angle as the first). This therefore allows us to transfer the angle of separation between two vectors (which is of course related to the inner-product between the two vectors) into the angle of rotation of any vector in the plane. After some work, it is possible to directly show that S is indeed a (real) rotation by an angle $\phi = 2\cos^{-1}(|\langle\psi_0|\psi_1\rangle|)$, therefore $|\langle\psi_0|\psi_1\rangle| = |\cos(\phi/2)|$.

Implementation

Sign flips along an arbitrary vector can easily be written as a sign flip P along the zero vector $|0\rangle$, by rotating to the zero vector, performing the sign flip P and rotating back. Therefore, if we specify the construction of $|\psi_0\rangle$ as the unitary V acting on $|0\rangle$, then:

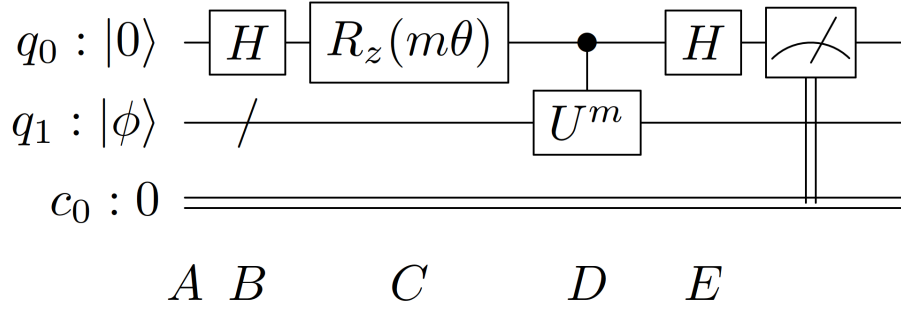
$$\begin{aligned} S_0 &= VPV^\dagger \\ S_1 &= UVPV^\dagger U^\dagger \\ S &= VPV^\dagger UVPV^\dagger U^\dagger \end{aligned} \tag{123}$$

Thus, full phase estimation of S operating on $|\psi_0\rangle$ reveals either ϕ or $-\phi$ and therefore the desired amplitude.

3.7.3 Summary of Wiebe et al.'s Bayesian Phase Estimation

Wiebe et al. introduce a variant of the iterative phase estimation algorithm. Their contribution is that, instead of determining the phase one bit at a time (e.g. from least to most significant as is usually done), they employ an approximate Bayesian approach to update a probability distribution over ϕ . This method handles noisy qubits elegantly and has the added advantage that we may employ it in our use case where the sign of ϕ changes. This ability to handle noisy qubits confers a significant degree of resiliency to the Bayesian approach even to unknown noise effects. Furthermore, known noise sources (as captured by a parameterized model, but with, of course, unknown instantaneous effect) are handled in a principled way directly in the likelihood term. This provides a very elegant way of handling various known sources of error. The Bayesian approach then becomes a joint noise-parameter-calibration plus useful-parameter-learning algorithm. The ‘nuisance’ parameters can then be integrated out, folding in their uncertainties into the final uncertainties of the useful parameters.

Here follows Wiebe et al.’s circuit:



where, $R_z(m\theta) = e^{im\theta Z/2}$ and $U|\phi\rangle = e^{i\phi}|\phi\rangle$.

The intuition for m is the same as in the full phase estimation algorithm, it peels away m copies of the eigenvalue. In the full algorithm, m runs through powers of two, in Wiebe's algorithm m is selected to maximize the information gain at each step.

In detail, here is how the circuit proceeds:

$$\begin{aligned}
A & : |0\rangle \otimes |\phi\rangle \\
B & : \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes |\phi\rangle \\
C & : \frac{1}{\sqrt{2}}(e^{im\theta/2}|0\rangle + e^{-im\theta/2}|1\rangle) \otimes |\phi\rangle \\
D & : \frac{1}{\sqrt{2}}(e^{im\theta/2}|0\rangle + e^{-im(\theta/2-\phi)}|1\rangle) \otimes |\phi\rangle \\
E & : \frac{1}{\sqrt{2}}((e^{im\theta/2} + e^{-im(\theta/2-\phi)})|0\rangle + (e^{im\theta/2} - e^{-im(\theta/2-\phi)})|1\rangle) \otimes |\phi\rangle
\end{aligned} \tag{124}$$

D, of course, corresponds to the famous 'phase kick-back' that leaves the eigenvector register unchanged but can be seen as changing the phase of the control qubit.

Therefore:

$$\begin{aligned}
P(0|\phi; \theta; m) & = \frac{1+\cos(m(\theta-\phi))}{2} \\
P(1|\phi; \theta; m) & = \frac{1-\cos(m(\theta-\phi))}{2}
\end{aligned} \tag{125}$$

The Bayesian analysis begins with a prior over ϕ , namely $P_0(\phi)$, and proceeds by Bayes' update law:

$$P_{i+1}(\phi|E_{i+1}; \theta_i, m_i) = \frac{P(E_{i+1}|\phi; \theta_i, m_i)P_i(\phi)}{N} \tag{126}$$

where, N is the normalization factor (integrating over ϕ), $E_i \in \{0,1\}$ is the outcome of measurement i and P_i is the i th updated posterior except for P_0 , which is the original prior (e.g. uniform angular prior).

Instead of performing the full Bayesian update above, Wiebe et al. choose to firstly start with a broad Gaussian prior, then draw samples from the posterior with which they fit a new Gaussian to. At every stage their best guess probability distribution is Gaussian and they use the Gaussian to decide how to set θ and m : $m_i = (1.25/\sigma_i)$ and $\theta_i \sim P_i(\phi)$. The

intuition behind this choice is to amplify (by multiplying by m) the difference between θ (a ‘guess’ for ϕ) and the true ϕ . The amplification needs to increase as the precision of the posterior improves, that is as θ becomes a better approximation, the difference $\theta - \phi$ becomes smaller and needs to be amplified even more, to be detectable. Put differently, as the variance σ_i decreases, $\theta - \phi$ approaches zero and this small difference needs to be magnified to be able to observe it in the statistics.

In the Results section, we present our Bayesian approach for Quantum Amplitude Estimation with two methods for handling multiple eigenvectors.

3.8 Algorithms - Topological Data Analysis

We continue to pursue the question of whether such a classical competitor exists for the algorithm for Topological Data Analysis (TDA) of [17]. Given a set S of points, a TDA algorithm finds certain topological properties of S at different scales of measurement, considering S at a given scale of measurement to be a *simplicial complex*, which is simply a collection of simplices, here considered to be subsets of S of a given cardinality that are “close enough” to each other.

Here a suggestion that a competitive classical algorithm might be possible is not the assumption of a pre-built binary tree, but rather the analysis condition of the Lloyd algorithm that the simplicial complex is *dense*, that is, that a constant fraction (say, one-tenth) of the subsets of S that might be simplices are in fact simplices. Under the condition, it is possible to randomly sample the simplices of the complex by rejection sampling, that is, by picking an appropriate cardinality subset of S uniformly at random, and accepting it as a randomly generated simplex if it satisfies the closeness conditions needed. Such a randomly generated simplex corresponds to a random row of the Laplacian matrix of the complex. Such uniform samples are unlikely to be enough to estimate the rank of the Laplacian (the key value needed for TDA), just as, for example, a uniform random sample of the edges of a dense undirected graph do not determine the number of connected components of the graph. However, such a sample of edges would induce a not-too-large collection of components, that may be helpful.

We note also that the paradigm of *property testing* may be appropriate to consider. Here it is assumed that an input undirected graph is either connected, or *far* from connected, where the latter means that many edges would need to be added to make it connected. The goal is a fast algorithm that can tell the difference between these two cases: to examine a small number of vertices and edges of the graph, and determine which of the two conditions holds. There are randomized property-testing algorithms that need $O(\frac{1}{\epsilon})$ work to determine that a graph is at least ϵ -far from connected, meaning that at least ϵm edges need to be added to the current set of m edges to make the graph connected. We intend to examine the question of whether there are similar classical property-testing algorithms for higher-order complexes and Betti numbers.

In addition to such "quantum negative" results, however, we are also trying to exploit the sampling paradigm for more clear-cut cases of quantum improvements, by using uniform sampling.

Besides the above discussion around classical replacements to QTDA, we also work to implement and improve QTDA on the IBM Q platform.

3.8.1 Boundary Map & Chain Complex

To describe our results for TDA, we briefly give some background on the nature of the computation. As mentioned, a simplicial complex is a collection of lists of points of a pointset, each list with an assigned orientation ± 1 . A simplex of length $k + 1$ is said to have dimension k , a k -simplex. A k -chain C_k is a formal sum of k -simplices, meaning that each simplex in a sum is assigned an integer coefficient. These chains form a group under the operation where the coefficient a simplex in the new chain is the sum of the coefficients of the simplex in the group operands; considering the coefficients as members of the reals, it is a vector space.

The boundary operator $\partial_k: C_k \rightarrow C_{k-1}$ is a linear operator taking a k -chain $\sum_i c_i \sigma_i$ to its boundary $\sum_i c_i \partial_k(\sigma_i)$, where for simplex $\sigma = (p_0, p_1, \dots, p_k)$, where $\hat{p}_j$ indicates that p_j is to be omitted. The composition $\partial_{k-1} \partial_k$ is identically zero, since a given k -simplex contributes a face (sublist) $(k - 2)$ -simplex in two different ways, and the different orders will have different signs, cancelling each other out in the formal sum. That is, the boundary of a k -chain has no boundary.



Figure 12. Boundary map

Homology Group.

Put another way, $\text{im}(\partial_{k+1}) \subset \text{ker}(\partial_k)$. A key object of homology theory is the homology group, $\text{ker}(\partial_k) / \text{im}(\partial_{k+1})$, and in particular, the rank of that group.

Considering the matrices A representing ∂_k and B representing ∂_{k+1} , we have $AB = 0$, and we are interested in $\text{ker}(A) / \text{im}(B)$. A member of this class of vectors has the form $x + \text{im}(B)$, where $x \in \text{ker}(A)$, and since we are adding $\text{im}(B)$, we can also assume $x \in \text{im}(B)^\perp = \text{ker}(B^\top)$. That is, the homology group is isomorphic to $\text{ker}(A) \cap \text{ker}(B^\top)$, which (as can be shown) is equal to $\text{ker}(A^\top A + BB^\top)$, where $(A^\top A + BB^\top)$ is the Hodge Laplacian. Returning to boundary operator notation, the k 'th Betti number b_k is the dimension of $\text{ker}(\partial_k^\top \partial_k + \partial_{k+1} \partial_{k+1}^\top)$, which is the number of zero eigenvalues of the positive-semidefinite matrix $\partial_k^\top \partial_k + \partial_{k+1} \partial_{k+1}^\top$. When $k = 0$, we can assume $\partial_k = 0$, ∂_{k+1} translates to the vertex-edge adjacency matrix, and the Hodge Laplacian translates

to the graph Laplacian. Here the Betti number tells us the number of connected components of the graph, while higher-order Betti numbers tell us the number of “holes”, “voids”, and so on.

circle		$b_0 = 1, b_1 = 1$
double loop		$b_0 = 1, b_1 = 2$
solid square		$b_0 = 1, b_1 = 0, b_2 = 0$
square with hole		$b_0 = 1, b_1 = 1, b_2 = 0$
cube surface		$b_0 = 1, b_1 = 0, b_2 = 1$
torus surface		$b_0 = 1, b_1 = 2, b_2 = 1$

Figure 13. Various shapes and their respective Betti numbers

3.8.2 Complex Filtrations and BarCodes

Tracking how Betti numbers change as function of the scale ϵ reveals how topological features come into existence (birth) and disappear (death) as the data is analyzed at different ϵ values. A topological feature that persists across broad range of length scales can be identified as a ‘true’ feature of the structure. In persistent homology, the persistence barcode encodes pairs of simplices meaning birth and death of homology classes. Persistence barcodes depend on the ordering of the simplices (called a filter) of the given simplicial complex. Figure 14. provides an example of a persistent homology barcode, on the top of the figure, a sequence of simplices corresponding to increasing scale of ϵ around each node are depicted, and underneath the filter values of each homology class. A virtue of persistent homology is the invariancy of the identified features to changes in the nodal locations that do not incur notable pair-wise modifications. That entails that such features are indifferent to various transformations (such as translations and rotations) and capable of handling relatively large distortions of nodal placement.

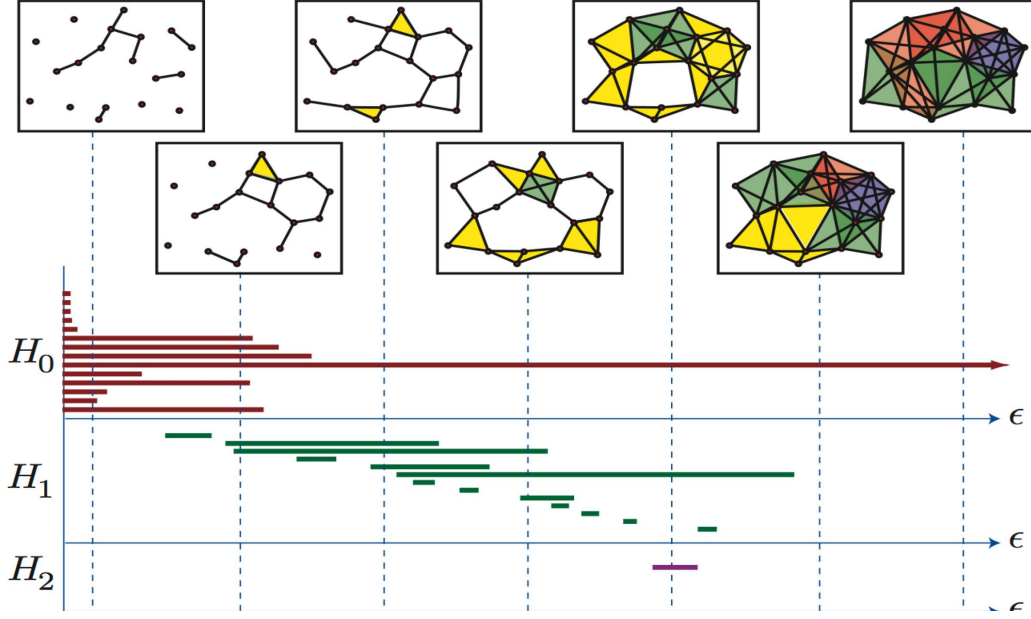


Figure 14. Barcode representing persistent homology across scales

3.9 Algorithms - Betti Number by Chebyshev Approximation

The Betti number calculation described above boils down to computing the dimension of the kernel of certain Laplacian matrices. In other words, Betti numbers can be calculated by estimating the rank of these Laplacian matrices. However, the natural approach, as used in Lloyd et. al (Lloyd, Garnerone, and Zanardi 2016) is to compute the complete eigen-decomposition of these matrices, and then estimating the kernel dimension. Here, we describe a rank estimation procedure, proposed in (Ubaru and Saad 2016) that does not require an eigen-decomposition of the matrix. The method is based on Chebyshev polynomial approximation of the matrix, and requires only computing the matrix moments $m_i = v^T A^i v$. The method has three ingredients:

The dominant spectral projector:

Let $A \in \mathbb{R}^{n \times n}$ be a symmetric SPD matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n$ and associated orthonormal eigenvectors $u_1, u_2, \dots, u_n$, respectively. We consider estimating the trace of the eigen-projector:

$$P = \sum_{\lambda_i \in [a, b]} u_i u_i^T, \quad (127)$$

where the interval $[a, b]$ is (implicitly or explicitly) selected so that it includes the relevant dominant eigenvalues that determine the rank. The eigenvalues of a projector are either zero or one and so the trace of P equals the number of terms in the sum above, i.e., the number of eigenvalues $\eta_{[a, b]}$ in $[a, b]$,

$$\eta_{[a, b]} = \text{trace}(P). \quad (128)$$

Although P is typically not available, it can be inexpensively approximated in practice by a polynomial of A . First, we can interpret P as a step function of A given by

$P = h(A)$, where $h(t) = 1$ if $t \in [a, b]$; $h(t) = 0$ otherwise

Next, this step function $h(t)$ can be approximated by a polynomial of degree m , say $\phi_m(t)$ and the projector P is expressed as $P \approx P_m(A)$. In this form, it becomes possible to estimate the trace of P by a stochastic estimator (Hutchinson 1990).

The trace estimator:

Hutchinson's unbiased estimator (Hutchinson 1990) uses only matrix-vector products to approximate the trace of a generic matrix D . The method estimates the trace $\text{trace}(D)$ by first generating random vectors $v_l, l = 1, \dots, n_v$ with equally probable entries ± 1 , and then computing the average over the samples of $v_l^T D v_l$,

$$\text{trace}(D) \approx \frac{1}{n_v} \sum_{l=1}^{n_v} v_l^T D v_l. \quad (129)$$

It is known that any random vectors v_l with mean of entries equal to zero and unit 2-norm can be used. Thus, substituting D with $P_m(A)$ in the above, will yield the following estimate of the trace of P :

$$\text{trace}(P) \approx \frac{1}{n_v} \sum_{l=1}^{n_v} v_l^T P_m(A) v_l. \quad (130)$$

We need to efficiently compute the vectors $P_m(A)v_l$ for any v_l , and this can be accomplished by a sequence of matrix-by-vector products with the matrix A .

Chebyshev filters:

Chebyshev polynomials are commonly used to expand the step function h , i.e., $h(t)$ is approximately expanded as:

$$h(t) \approx \sum_{k=0}^m \gamma_k T_k(t), \quad (131)$$

where each T_k is the k -degree Chebyshev polynomial of the first kind, formally defined as $T_k(t) = \cos(k \cos^{-1}(t))$. Since Chebyshev polynomials are based on the interval $[-1, 1]$ we will assume first that A has eigenvalues between -1 and 1 . Let a, b such that $-1 \leq a < b \leq 1$. The expansion coefficients γ_k for the polynomial to approximate a step function $h(t)$, which takes value 1 in $[a, b]$ and 0 elsewhere, are known:

$$\gamma_k = \begin{cases} \frac{1}{\pi} (\cos^{-1}(a) - \cos^{-1}(b)) & : k = 0, \\ \frac{2}{\pi} \left(\frac{\sin(k \cos^{-1}(a)) - \sin(k \cos^{-1}(b))}{k} \right) & : k > 0 \end{cases}. \quad (132)$$

Once the γ_k 's are known, the desired Chebyshev expansion of the projector P will be given by: $P \approx P_m(A) = \sum_{k=0}^m \gamma_k T_k(A)$.

The approximate matrix rank r_τ can be determined by setting the interval $[a, b] = [\tau, \lambda_1]$, where τ is a small threshold > 0 , that defines the inflection point for the step function, such that we count the nonzero eigenvalues of the matrix to compute the rank. As a result, the

approximate rank of a matrix A using the Chebyshev polynomial filtering method is estimated by:

$$r_\tau = \eta_{[\tau, \lambda_1]} \approx \frac{n}{n_v} \sum_{l=1}^{n_v} [\sum_{k=0}^m \gamma_k (v_l)^\top T_k(A) v_l]. \quad (133)$$

Chebyshev polynomial from moments:

The Chebyshev polynomial of the matrix $\phi_m(A)v_l$ is typically built using the three-term recurrence (Ubaru and Saad 2016). However, with the quantum approach, we cannot use the recurrence approach, since we can only compute the moments. Therefore, we use a general (summation) formula to compute the Chebyshev polynomials. Specifically, we have

$$T_k(x) = \sum_{j=0}^{\lfloor \frac{k}{2} \rfloor} (-1)^j 2^{k-(2j+1)} a(k, j) x^{k-2j}, \quad (134)$$

where

$$a(k, j) = \frac{\binom{2j}{j} \binom{k}{2j}}{\binom{k-1}{j}}. \quad (135)$$

Given the moments $m_i = v^T A^i v$, we compute the Chebyshev moments $T_k(m_i)$ using the above formula.

3.10 Applications - Topological Classifier

One of the applications of Topological data analysis and persistent homology, is in machine learning, for classification problems. We can use the topological features of a given data for training classifiers, which we call the topological classifiers. The idea is that the Betti number of the points at different scale form the persistence barcode, as discussed in earlier section. We can use these Betti numbers at various scales ϵ as the (topological) input features for the classifiers.

A popular application where such topological classifiers will be useful is in digit recognition. Suppose we are given a set of digit images (e.g., MNIST), we can sample n number of points from each of the images. These points from different digits between 0 and 9, and the digits (their points) will have different Betti numbers at different scale. Instead of using the whole image or the sampled points as features for classifying the digits, we can use the topological features (persistence barcode). In section 4.8, we present few experimental results for digit classification using topological classifier (Betti Numbers). We also present an improved hybrid topological and statistical digit classifier that yields superior performance compared to the topological classifier.

3.11 Applications - Pricing & Risk Analysis

Pricing derivatives or portfolios of financial products and analyzing the risk involved are important and computationally intense tasks in the financial service sector (FSS). The fair price of a financial product is usually defined as its expected payoff, which is defined by a

payoff function and an underlying uncertainty model. Classically, the expected payoff or the risk involved is estimated using Monte Carlo simulation. Monte Carlo simulation converges only slowly, which makes pricing and risk analysis two of the computationally most demanding tasks in the FSS often running overnight on large compute grids.

Quantum computing has the potential to speed up Monte Carlo simulation for certain applications [19]. By loading the uncertainty model into a quantum state and preparing the amplitude of an ancilla qubit such that it reflects the payoff function, it is possible to use the quantum amplitude estimation algorithm [20] to approximate the expected payoff with a quadratic speedup compared to Monte Carlo simulation. A million classical Monte Carlo samples could be replaced by a few thousand quantum samples while achieving the same accuracy. This could help to reduce the time needed to analyze price or risk of a financial product from many hours to a few minutes. Two common risk measures that can be computed that way are Value at Risk (VaR) and Conditional Value at Risk (CVaR), also called expected shortfall.

A schematic drawing of a quantum circuit implementing quantum amplitude estimation is shown in Figure 15. The top m qubits represent the final result, and bottom $(n+1)$ qubits represent the considered problem. An application of operator Q can be considered the quantum counterpart of a single classical Monte Carlo sample. Thus, the number of samples corresponds to 2^m . Adding another results qubit reduces the estimation error by a factor of two but also doubles the number of samples. Before the results qubits are measured an inverse QFT is performed. The integer y that corresponds to the result measured for the m results qubits is then mapped classically to the amplitude of interest via $\sin^2\left(\frac{y\pi}{2^m}\right)$. We demonstrated this approach on several small examples: a single Treasury bill and a two-asset portfolio [19].

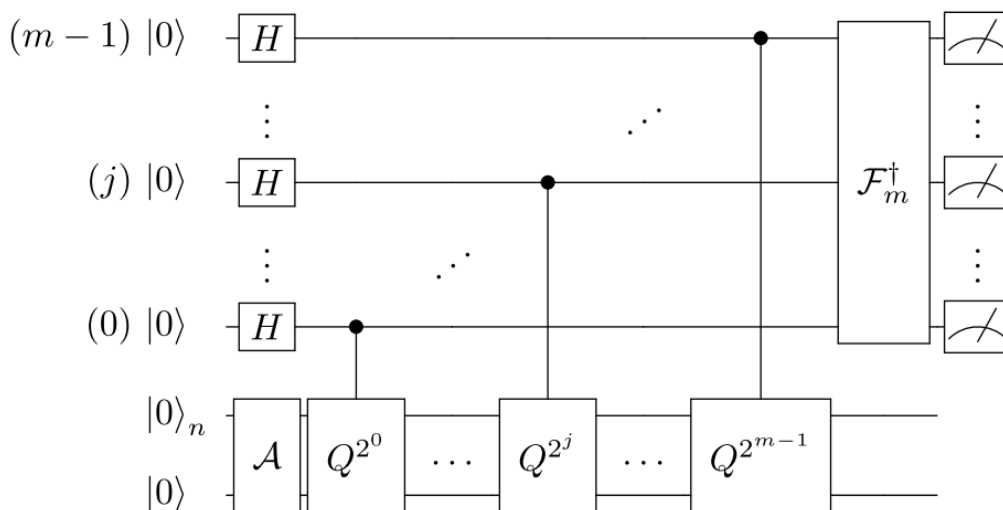


Figure 15: Amplitude estimation circuit.

3.12 Applications - Variational Heuristics for Combinatorial Optimization

The hybrid quantum/classical variational approach is an optimization algorithm devised for the early generation of universal quantum computers. Variational approaches have been applied to a variety of fields, e.g., chemistry, machine learning, optimization. In broad terms, a variational approach works by choosing a parametrization of the quantum state that depends on a relatively small set of parameters, then using classical optimization routines to try to determine values of the parameters corresponding to a quantum state that maximizes or minimizes a given utility function. Typically, the utility function is a Hamiltonian encoding the total energy of the system, to be minimized. This directly relates to an optimization context: the same idea can be applied to classical combinatorial optimization problems, provided that we can construct a Hamiltonian encoding the objective function of the optimization problem.

This section summarizes our experience in using classical derivative-free optimization methods to try to find good solutions for these problems and sheds some lights on limitations that should be overcome to increase the effectiveness of the variational approach. All our computational experiments are based on noise-free simulations of quantum hardware, and we therefore have access to the full quantum state with which we can exactly evaluate the Hamiltonians. The conclusions of the study might be considerably different if real hardware had been used, considering the inherent amount of noise that affects the computations. To the best of our knowledge, this is the most comprehensive numerical study of a hybrid quantum/classical variational method for combinatorial optimization. The most notable limitation of our study is the fact that we use only one type of variational form, and we do not test any problem-dependent variational form as advocated by some other works.

3.12.1 Hamiltonians for binary optimization problems.

The most natural formulation of combinatorial optimization problems on a quantum computer is via an Ising spin glass model, which directly translates into a Hamiltonian. Indeed, we have:

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad Z|0\rangle = |0\rangle \quad Z|1\rangle = -|1\rangle, \quad (136)$$

and the two eigenvalues ± 1 of Z correspond to the positive and negative spin. The Ising spin glass model can be seen as a quadratic unconstrained binary optimization problem, and it inherits its hardness. In general, computing the partition function of an Ising model is NP-complete. Because of this, any problem in NP can be reduced to an Ising model; we are particularly interested in combinatorial problems that have a natural mapping to Ising spin glass models, i.e., problems with the property that if the original instance has size n , we need only n qubits for an Ising spin glass representation.

As we will see in the following, some problems are naturally formulated in terms of spin variables ± 1 , whereas others have a more natural formulation in terms of 0-1 variables.

The transformation between the two types of variables is straightforward. The main idea is as follows. Consider a binary quadratic unconstrained optimization problem:

$$\min\{c^T x + x^T Q x : x \in \{0,1\}^n\}, \quad (137)$$

then transform into an Ising model using the substitution $x_j = \frac{y_j^Z + 1}{2}$, where $x_j \in \{0,1\}$ and $y_j^Z \in \{-1,1\}$ for $j = 1, \dots, n$; we use the superscript Z to distinguish ± 1 spins from 0-1 variables. Since is a quadratic model, the substitution yields a summation of terms, each of which contains either one or two y_j^Z variables. The Hamiltonian is then a summation of weighted tensor products of Z Pauli operators, where each term of the summation contains at most two Z s. Furthermore, since Z is diagonal, the Hamiltonian resulting from this transformation is diagonal.

If the original binary quadratic optimization problem is constrained, the approach mentioned above can still be applied by adding appropriate (quadratic) penalties for constraint violations in the objective function. In the cases of relevance for this paper, the additional constraints can be expressed as the requirement $Ax = b$ for some choice of A, b ; in this case, it is sufficient to add the term $\alpha \|Ax - b\|^2$ to the objective function with a sufficiently large α , to ensure that the unconstrained formulation has the same optimum as the original constrained formulation.

We study six classes of combinatorial optimization problem employed in our numerical study. These problems are: Maximum Stable Set, Maximum 3-satisfiability, Number Partitioning, Max Cut, Market Split, Traveling Salesman Problem. We do not give details for space reasons.

3.13 Applications - Additional Applications

Although this is a very industry specific application, the developed algorithms can be used to evaluate similar problems from other areas as well. For instance, the service level measures used in logistics and supply chain management are quite related to the risk measures used in finance. The algorithms described above could be used to efficiently estimate the service level and inventory holding costs in of a supply chain and use them as objective function and constraints in a simulation-based optimization framework. In the following, we outline an illustrative example that shows how the financial risk analysis could be applied to determine optimal safety stocks.

Suppose a random variable D representing the customer demand for a product P . Further on, suppose a safety stock of S units for P . The stock-out probability is a common service level measure and defined as the probability of D being larger than S . For a given S we can use amplitude estimation to estimate the stock-out probability with a quadratic speedup over Monte Carlo simulation. If we are interested in the safety stock S that leads to a certain given target service level, then the resulting problem is equivalent to computing the VaR for a financial asset and – as for VaR – a bisection search over S can provide the optimal safety stock in a logarithmic number of steps [19]. Figure 16 illustrates such a situation.

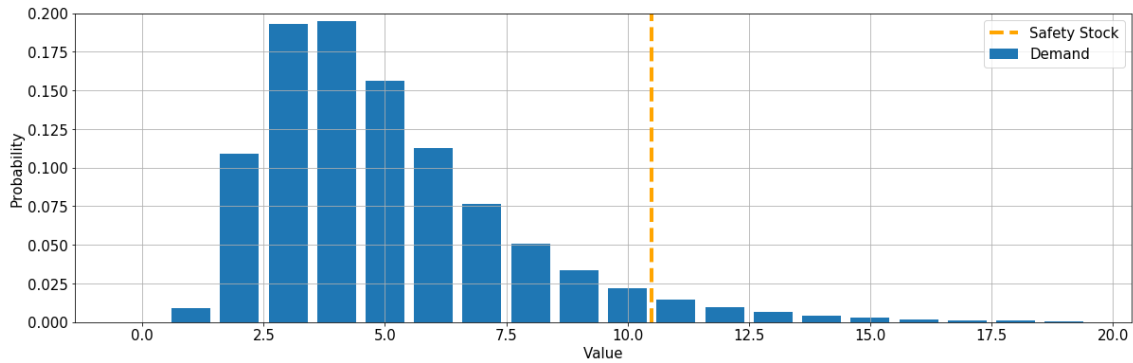


Figure 16. Illustrative example of a given demand distribution and the safety stock value that leads to a service level of 95%, i.e. a stock-out probability of 5%.

3.14 Data-Porting Overview

A fundamental issue in assessing the significance and viability of quantum algorithms relates to the problem of porting classical data to the quantum setting. Quantum algorithms are often given and analyzed under the assumption that a quantum state has been prepared as input, or alternatively, that a classical data structure has been built such that a quantum state can be efficiently prepared from it. This assumption is convenient, but not trivially realized. In the setting of randomized classical algorithms, this assumption often corresponds to the condition that data structures have been built a-priori to allow random sampling of the data, for example, that for each row i of a matrix A , its j 'th entry can be sampled with probability proportional to A_{ij}^2 . Indeed, the same binary tree data structure on the data that is used for preparing a quantum state via amplitude amplification in QC can be used for random sampling in the classical setting.

The cost of building such binary trees is not included in the analysis of the quantum algorithms that depend on them; this raises the natural question of the performance of classical randomized algorithms under the assumption that such trees have been pre-built, and can be used for random sampling. Some recent work has shown that, indeed, under the conditions used for analysis of some quantum algorithms, there are classical randomized algorithms with similar performance.

3.14.1 Approximate Quantum Fourier Transform

3.14.2 Introduction

The quantum Fourier transform is of fundamental importance and represents the key ingredient in most quantum algorithms today, from quantum phase estimation, to state quantum machine learning and ground state calculations.

Similar to its classical counterpart, the quantum Fourier transform is a unitary transformation from the position space $\{x\}_j$ into the momentum space $\{p\}_j$ given by the change of basis

$$\sum_{j=0}^{N-1} x_j |j\rangle \rightarrow \sum_{k=0}^{N-1} p_k |k\rangle = \sum_{k=0}^{N-1} e^{2\pi i a k / N} |k\rangle, \quad (138)$$

This change of basis can be performed by the quantum circuit depicted in Figure 17.

3.14.3 Quantum transformations

Although the outcome of the quantum transform is functionally similar to the classical transform, there is a significant difference in the runtime complexity. While the best-known classical result (the Fast Fourier Transform) uses $O(n^2)$ actions, the quantum Fourier transform requires merely $O(n^2)$ gates for an input state represented by $O(2^n)$ complex values.

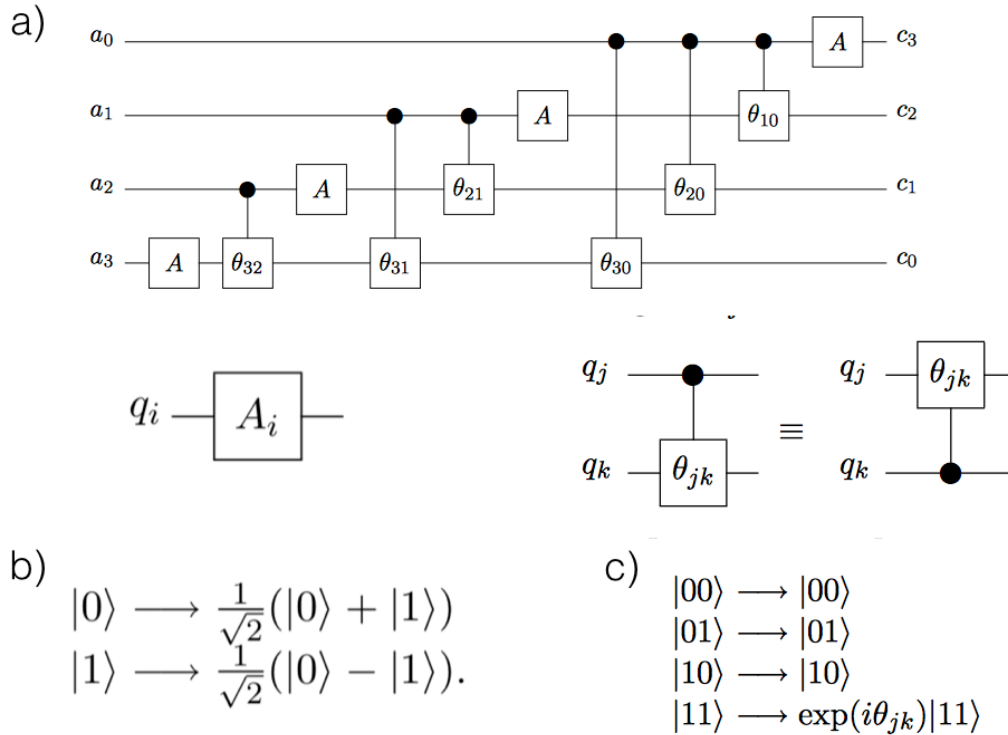


Figure 17. a) QFT circuit description b) Application of the A gate (Hadamard) c) Application of the entangling phase gates θ_{ij}

As related applications become more resource intensive, there is a greater need to formulate and execute better, faster and more efficient schemes to perform quantum Fourier transforms, either by novel new circuits or by approximation. The research question deals with potential applications, including implementation and benchmarking of an assortment of approximation schemes for the quantum Fourier transform.

To improve the algorithmic complexity, the most naive approach that can be taken is to perform partial decimation of the transformation gates shown in Figure 17 a,b.

Such decimations have been shown to be bounded in error and theoretically bounded to the model in which they are approximated [21].

In fact, applying mean field approximations decimating time scale degrees of freedoms in a cascade manner will not change the underlying universal property of the signal transformed. Such decimations and approximated will be described next.

3.14.4 Mean Field Approximation.

As a starting point, we choose to perform the simple approximation of decimating the action in a fashion akin to the mean field treatment. This decimation can be described as follows. We start by noting that a binary basis can be written as

$$a = \sum_{i=0}^{N-1} a_i 2^i, \quad k = \sum_{i=0}^{N-1} k_i 2^i, \quad (139)$$

and the product in the exponent can be written as

$$ak = a_0 c_0 + 2(a_0 c_1 + a_1 c_0) + 2^2(a_0 c_2 + a_1 c_1 + a_2 c_0) + \dots + (a_0 c_{N-1} + \dots + a_{L-1} c_0) + \mathcal{O}(2^L) \quad (140)$$

Since $(e^{2\pi i/N})^\alpha = 1$ for $\alpha > N$ we can neglect the terms of $\mathcal{O}(2^L)$ and thereby approximate the exponent at (119) as

$$e^{2\pi i ak/2^L} = e^{2\pi i a_0 k_0/2^L} e^{2\pi i (a_0 k_1 + a_1 k_0)/2^{L-1}} \dots e^{2\pi i (a_0 c_{N-1} + \dots + a_{L-1} c_0)/2} \quad (141)$$

As can be observed from the power series, for the approximation parameterized by an integer m , the $N - m$ smallest terms are neglected. In all the remaining terms, the arguments are then multiples of $2\pi/2^m$ where m is the level of the approximation.

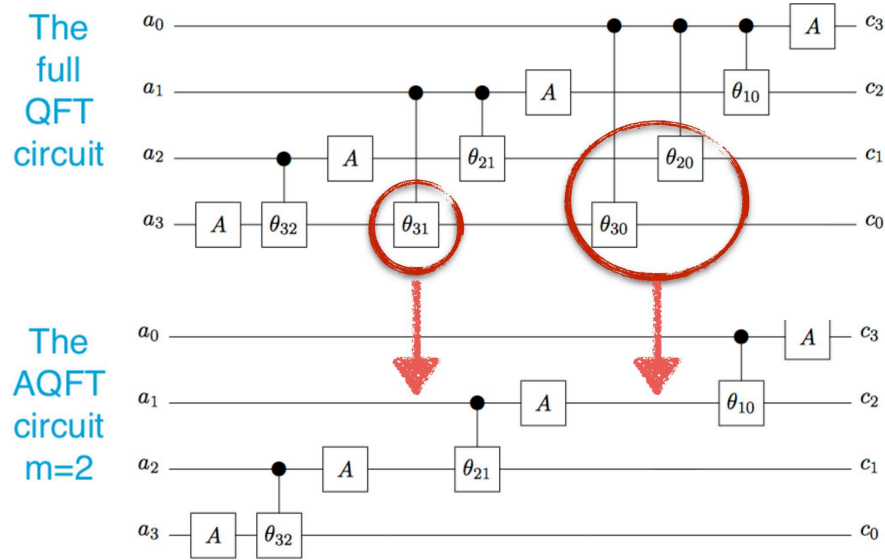


Figure 18. Top: the standard (full) QFT circuit. Bottom: the Mean Field decimated (Approximated QFT) circuit with approximation level of $m=2$

Physically, this coincides with coarse-graining the short wavelength components of the wave function and removing the appropriate controlled phase shift gates in the circuit. Figure 18 provides an example for an $m = 2$ approximation. By using the mean field approximation, we need N operations of the A gate and $(2N - m)(m - 1)/2$ operations of the phase gate θ_{ij} , and thus the run time complexity increases as $\mathcal{O}(Nm + m^2)$.

An important fact to note is that while the run time complexity now scales as Nm , the matrix elements of the full and approximate Fourier transform differ by a multiplicative factor of $e^{i\epsilon}$ with $|\epsilon| \leq 2\pi N/2^m$.

3.15 Data-Porting - Narrow Band and Sparse Quantum Fourier Transforms

A worthy goal in Quantum Computing is producing short-depth approximations of the Quantum Fourier Transform (QFT). We study the case of the QFT of a state that is guaranteed to be either narrow-band or sparse in the Fourier domain. We exploit the structure inherent in these two related special cases to produce shorter-depth, exact and approximative versions of the QFT.

3.15.1 Introduction

The Quantum Fourier Transform is arguably the most important building block of quantum computing. This statement can be made trivially true, by the observation that the single qubit Hadamard gate (used universally) is the exact QFT of a single qubit, while the tensored Hadamard gate on multiple qubits is the first order (depth 1) approximation of the Fourier

transform. Therefore, a better understanding of the Fourier transform and its approximations should prove fruitful for quantum computing in general.

One of the first papers to study the approximation of the QFT was Barenco et. al. They provide a principled method to decimate the full QFT circuit and arrive at demonstrably useful approximations. We build on this work to provide short-depth QFT circuits for narrow-band and sparse QFT. In the sparse case, the QFT competes with excellent classical techniques for the FFT of k -sparse signals, with $O(kn)$ complexity. In order to surpass classical computing in this case, we have to make use of the more recent work of Cleve et. al [22].

3.15.2 QFT Circuit Decimation and Interpretation

Barenco et. al [23] show that the QFT unitary matrix on n qubits, has entries:

$$F_{i,j} = \omega^{ij} \quad (142)$$

where $\omega = e^{\frac{2i\pi}{2^n}}$ and i is the row index (corresponding to the output frequency) and j is the column index corresponding to the input ‘time’-step variable. They proceed to show that, if i and j are written in binary (since they are, after all, represented with qubits) then only bit- k products of $\{i_{k_1}\}$ times $\{j_{k_2}\}$, corresponding to $k_1 + k_2 < n$ contribute to the matrix entry. Decimation proceeds by eliminating terms involving $k_1 + k_2 = m' < m$ for some approximation-order parameter m . Dropping these terms corresponds to eliminating control-rotation gates between $n - m'$ -neighbour gates. This works as a useful approximation because the rotation angle is inversely proportional to the gap, such that large gaps correspond to small rotations, worthy targets for elimination.

We now introduce a novel interpretation of this decimation scheme:

Consider the complex plane. The slowest non-zero frequency ($i = 1$) ω , rotates once around the plane reaching complex values ω^j , corresponding to the 2^n 2^n -th-roots of unity; higher frequencies visit a subset of these complex numbers. Now, for approximation-order parameter m , consider the 2^m 2^m -roots of unity around the complex plane. Rotate the visited complex numbers by angle $-(2\pi(2^{-m} + 2^{-n}))/2$. This center the visited complex numbers around the closest 2^m root of lesser complex angle.

Next comes the crucial step: for each of the visited complex numbers, replace it with the closest 2^m -th root. This is a kind of complex rounding (literally equivalent to rounding on the one-dimensional unit circle, thereby selecting the closest root by distance on the unit circle, or equivalently selecting the root with the largest positive overlap and normalizing. Without rotation this is equivalent to flooring to the closest root of lesser angle). The new rounded complex numbers still ‘rotate’ around the circle with the same frequency as before but now with jumps between and pauses at the 2^m roots, introducing frequency artefacts.

The above explanation suggests a novel improvement to the Barenco et. al AQFT (unrelated to our use below): After the complex rounding step, the new complex numbers should be rotated back by $-(2\pi(2^{-m} + 2^{-n}))/2$. This ensures that the real and complex parts of the

generated approximation basis states are closer to the full basis states. However, the magnitude of a given frequency is unaffected.

With the above explanation, we can also easily see that the complementary circuit to the approximation circuit, produces a basis that captures the elementwise, multiplicative error between the approximation basis and the full basis. As such, it is not immediately obvious how such a basis could be used on a quantum computer since it must be multiplied elementwise with the approximation basis before being applied to the time-domain state vector to recover the full QFT. This operation is not easily done on a quantum computer, except for commuting operators in the diagonal basis.

4.0 RESULTS AND DISCUSSION

4.1 Algorithms - Variational Quantum k-Eigensolver and Monte Carlo Minimizer

We performed extensive numerical tests of the VQkE, using our implementation within the open-source QISKit AQUA toolkit. Experiments were based on two classes of problems: problems arising in quantum chemistry, as well as randomly generated Hamiltonians.

4.1.1 Quantum chemistry experimentation

In quantum chemistry, we verified that our VQkE implementation can successfully recover the excited states (eigenvalues) of small molecules such as H₂, at various interatomic distances; this is provided that we use a sufficiently powerful variational form. However, our experiments indicate that the algorithm has difficulty identifying some excited states of larger molecules: we focused our efforts on LiH, and observed that while our implementation typically converges to an excited state, we are unable to control which excited states are returned. In other words, the numerical implementation is prone to “missing” some excited states. This is most likely due to the nonconvex nature of the classical optimization problems that are solved by VQkE in its basic iteration step: the algorithm is guaranteed to find all eigenvalues only if these nonconvex problems are solved to global optimality, and it seems that this is difficult in practice.

4.1.2 Random Hamiltonian experimentation

Similar conclusions can be drawn on random Hamiltonians. In this context, we were able to verify that the accuracy of VQkE decreases as the number of Pauli terms in the Hamiltonian increases. In the left part of the figure below we show on the x axis a measure of density of the Hamiltonian, and on the y axis a measure of error of the eigenvalues (more specifically, the average absolute deviation of computed eigenvalues, from the closest correct eigenvalue). We tested two versions of VQkE, one of which periodically increases the penalty parameters for the deflated auxiliary problem. We noticed that as the density of the Hamiltonian increases, the accuracy of the eigenvalues decreases.

4.2 Algorithms - Improving Variational Quantum Optimization Using CVaR

4.2.1 Results on IBM's Poughkeepsie quantum processor

We report in Figure 20 a summary of our computational experience when using CVaR optimization on IBM's quantum hardware.

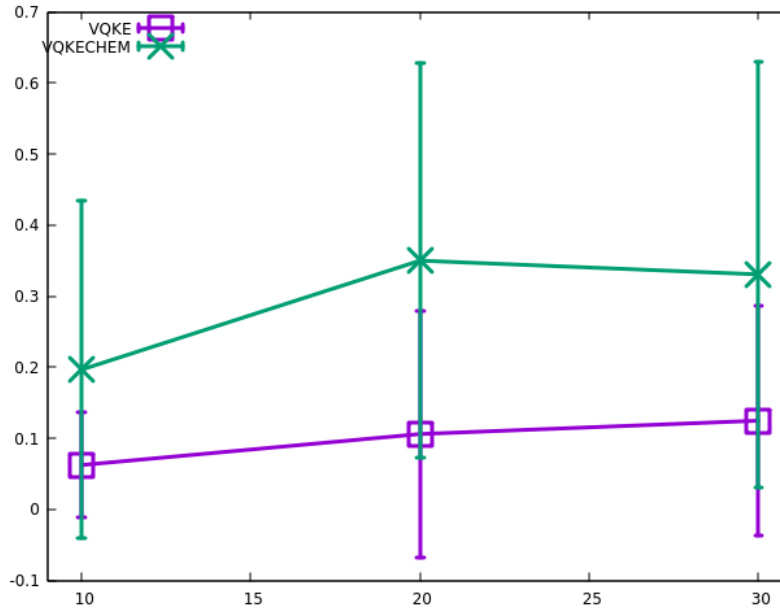


Figure 19. Eigenvalue error as a function of the Hamiltonian density assessment of the Variational Quantum k-Eigsolver with periodical penalty parameter increase (green) and without (purple)

The value of α determines the level of risk used in the definition of CVaR. The figure shows that for $\alpha = 1$ the optimization never reaches a large overlap with the ground state (reported on the right-hand side in the figure): this corresponds to the traditional expected value optimization. However, for smaller value of α we reach significant overlap with the ground state, as large as 0.3 when $\alpha = 0.25$.

A more detailed analysis of the computational experience can be found in our recent arXiv report: <https://arxiv.org/abs/1907.04769>. The paper also contains some analytical results on QAOA and VQE, highlighting an issue of “controllability” of the quantum state produced by QAOA.

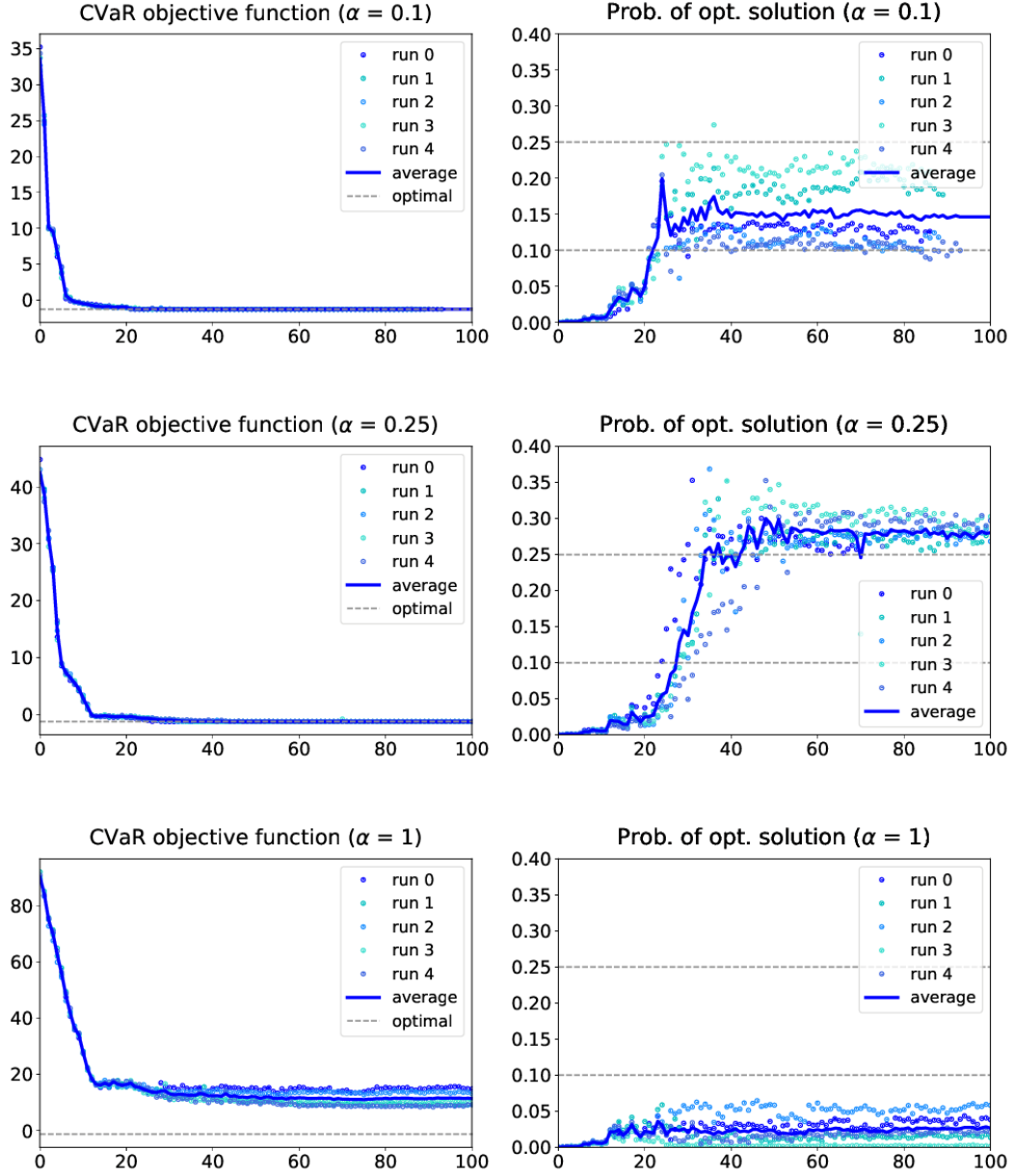


Figure 20. CVaR optimization on a portfolio optimization problem, executed on IBM's quantum hardware.

4.3 Algorithms - Robust Optimization

A focus for future research is to study classes of parametrized unitaries $V(\sigma)$ such that there exists a simple set S (e.g., a compact set) with the property that $\{V(\sigma)|\bar{\xi}\rangle : \sigma \in S\} \cong \{\xi \in P : d(\xi, \bar{\xi}) \leq \rho\}$, for an appropriate choice of d and ρ . Notice that in general if ρ is so large as to include all possible probability distributions over $\{0, \dots, 2^p - 1\}$, then $V(\sigma)$ must include all possible unitaries, hence no simple set S exists; in other words, $V(\sigma)$ would have to include a universal set of circuits, which does not have a compact representation. It follows that ρ must be small for this approach to have hope of succeeding.

Distance measures of particular interest are those akin to the Wasserstein distance d_W , leading to a distance of the form:

$$d_W(\sum_j \alpha_j |j\rangle, \sum_j \beta_j |j\rangle) = \sqrt[\ell]{\sum_j ||\alpha_j|^2 - |\beta_j|^2|^\ell}. \quad (143)$$

Defining $V(\sigma)$ so that $\{V(\sigma)|\bar{\xi}\rangle : \sigma \in S\}$ is approximately equal to $\{\xi \in P : d_W(\xi, \bar{\xi}) \leq \rho\}$ is a research question under investigation. Notice that to measure the distance between two states as in the definition of d_W , a good starting point is the modified SWAP test described in [24].

4.4 Algorithms – Structured Random Walks and Quantum Walks

In this section we consider additional results associated with the Structured Random Walks and Quantum Walks presented in Section 3.0. In particular, we consider the quantum algorithms for these Structured Random Walks and Quantum Walks to compute the equilibrium distribution of the corresponding Markov process. The equilibrium distribution of the Markov process associated with the random walk often forms the basis of various algorithms of interest as well as for other useful purposes. Examples include applications in computer vision and applications based on generating random elements in large and complicated sets (such as the set of lattice points in a convex body). Another example is the use of the equilibrium distribution to obtain hitting times, which are the expected times for the random walk to reach a given node for the first time from a given initial start node. These hitting times in turn often form the basis of various algorithms of interest as well as for other useful purposes. Examples include collaborative filtering (making automatic predictions about the interests of a user by collecting preferences from many users) and link prediction in a network (predicting the possibility of links between two nodes in a network that have not yet been connected by network information). In addition, we also consider in this section the use of Quantum Walks and Structured Random Walks as the basis of other algorithms of interest such as those related to clique detection.

4.4.1 Computational Complexity Comparison of Algorithms

We now present and discuss results that compare the computational complexity of the quantum algorithms described in Section 3.0 for Quantum Walks and Structured Random Walks. As a representative example, we consider the random walk on the nonnegative quarter plane defined over the two-dimensional lattice $\mathbb{Z}_+ \times \mathbb{Z}_+$, recalling that $\mathbb{Z}_+$ denotes the nonnegative integers. Random walks on the nonnegative quarter plane arise naturally in many areas of mathematics and physics among other scientific domains, including stochastic networks and analytic combinatorics, as well as non-scientific applications.

For consistent notation across both sets of algorithms, let ϵ denote our predefined error tolerance, let $\sqrt{N}$ denote the truncation of the state space in the second dimension, and define $\mathbb{Z}_M := \{1, \dots, M\}$. The Quantum Walk algorithms require the same truncation level

in both the first and the second dimensions, resulting in a state space of $\mathbb{Z}_{\sqrt{N}} \times \mathbb{Z}_{\sqrt{N}}$; note that this is in strong contrast to the Structured Random Walk algorithms that comparatively consider the state space $\mathbb{Z}_+ \times \mathbb{Z}_{\sqrt{N}}$, which is a more accurate representation of the desired two-dimensional lattice $\mathbb{Z}_+ \times \mathbb{Z}_+$ by only truncating in one of the two dimensions and not both. We then can show that the mixing time for the Quantum Walk algorithms on the two-dimensional lattice $\mathbb{Z}_{\sqrt{N}} \times \mathbb{Z}_{\sqrt{N}}$ is given by $O\left(\frac{\sqrt{N \log N}}{\epsilon}\right)$. This means that the average probability distribution for the Quantum Walk will be within ϵ of the true equilibrium distribution after $O\left(\frac{\sqrt{N \log N}}{\epsilon}\right)$ time steps.

Table 1. Run time analysis comparison.

METHODS	COST
Classical Direct solver (inversion)	$O(\sqrt{N^3})$
Classical Iterative solver (using CG)	$O(N\kappa s \log(1/\epsilon))$
Quantum Linear System algorithm (QLSA)	$\left(\frac{\sqrt{N} \log(N) s^2 \kappa^2}{\epsilon}\right)$ or $\left(\sqrt{N} s \kappa \text{polylog}\left(\frac{s\kappa}{\epsilon}\right)\right)$,
Quantum Phase Algorithm and Eigensolver (no data loading)	$\left(\frac{\log^2(N)s}{\epsilon}\right)$
QPA and Eigensolver (with data loading)	$\left(\frac{\sqrt{N} \log^2(N)s}{\epsilon}\right)$
Quantum walk on 2-d lattice size $\sqrt{N} \times \sqrt{N}$	$O\left(\frac{\sqrt{N \log N}}{\epsilon}\right)$

In comparison, using the Quantum Phase Algorithm and Eigensolver for the Structured Random Walk on the two-dimensional lattice $\mathbb{Z}_+ \times \mathbb{Z}_{\sqrt{N}}$, we can obtain the equilibrium distribution within an error tolerance of ϵ at a computational cost of $O\left(\frac{\log^2(\sqrt{N})s}{\epsilon}\right)$, where s denotes the row sparsity of the matrices (which is typically very small for random walks) as part of the corresponding algorithm in Section 3.5. This computational complexity does not include the time for QRAM loading, which in the worst case would require an additional time complexity of $O(\sqrt{N})$, and therefore in this case the total computation time would be $O\left(\frac{\sqrt{N} \log^2(\sqrt{N})s}{\epsilon}\right)$. Alternatively, we can exploit the improved runtime of QLSP to obtain a computational complexity of $O\left(\sqrt{N} s \kappa \text{polylog}\left(\frac{s\kappa}{\epsilon}\right)\right)$, where κ is the condition

number of the matrices as part of the corresponding algorithm in Section 3.5. Both of these Structured Random Walk algorithms provide runtime improvements over the Quantum Walk algorithms. In the cases where $O\left(\frac{1}{\epsilon}\right)$ dominates $O(\sqrt{N \log N})$, which is often the case since one desires results very close to the true equilibrium distribution, the second Structured Random Walk algorithm provides a very significant speedup over the Quantum Walk algorithm. *This result further addresses the critical limitation of the $O\left(\frac{1}{\epsilon}\right)$ computational complexity that is a well-known bottleneck for Quantum Walk algorithms.* We summarize these results in the table above.

4.4.2 Application of Clique Detection Algorithm

We next present and discuss results related to the application of Quantum Walk algorithms within the context of clique detection. In particular, we consider a representative example of applying the Quantum Walk algorithm described in Section 3.6 to estimate the clique number for a given graph. The clique number of a graph is defined as the number of vertices in a maximum clique, where a maximum clique is in turn a clique such that there is no clique with more vertices. The clique number $\alpha(G)$ of graph G satisfies the inequality $\alpha(G) \leq \lambda(G)+1$, where $\lambda(G)$ denotes the spectral radius of the adjacency matrix associated with graph G .

Recall that the k^{th} root (computed classically) of the trace of the k^{th} power of the adjacency matrix (computed using the quantum walk circuit) is one way to estimate the leading eigenvalue. The leading eigenvalue can be used to calculate an upper bound for the clique number. Once the adjacency matrix is raised to the different powers one can gain diagonal terms in the new adjacency matrix. Such terms a_{ii} are the numbers of paths from i back to i and in fact are an indication of loops.

The quantum computation involves the design of a quantum circuit which first encodes the structure of the graph and then performs parallel evaluations of the different paths on it. The paths are the different routes that can be traversed on the graph. The parallel path traverse is done by means of a circuit embodying a quantum random walk. The clique analysis can be performed by repeated application of the quantum random walk circuit. The quantum walk encodes in its spectrum the clique analysis. Eigenvalues (of the spectrum of the random walk) of higher powers of the shift followed by the quantum coin operator provide a measure of self-cliques. The action of taking the powers of the adjacency matrix can be mapped towards consecutive operations of the quantum shift operator in the random walk. Coupling this to a spectrum inference mechanism such as β -QPE or even regular QPE can give an exponential speed up with respect to a local cycle detection leveraging the classical quantum hybrid approach.

1. Execute quantum walks to gather information about the trace of $A(G)$, $A(G)^2$, $A(G)^3$, ..., where $A(G)$ denotes the adjacency matrix of graph G , and self-loops are excluded.
2. For each trace computation of $A(G)^k$, obtain its k^{th} root and compare it with the previous approximation.

3. Once $\lambda(G)$ is computed, return the upper bound of the clique number as an approximation.

As an example, we consider the graph with 7 vertices shown below. The sparsity pattern of $A(G)$ is also shown. This graph has one 3-clique, and one 4-clique (and several 2-cliques). Therefore, the clique number is $\alpha(G)=4$. The spectral radius of $A(G)$ is 3.1249 and thus we obtain $\alpha(G) \leq 4.1249$. We can then take the upper bound as an approximation of the clique number, rounded to the nearest integer.

For this particular example, the predicted and true clique number were the same. As a side note, notice that the leading eigenvalue $\lambda(G)$ gets approximated in an increasingly improved manner as ‘ k ’ increases. The convergence of the spectral radius approximation is plotted at the end of this section.

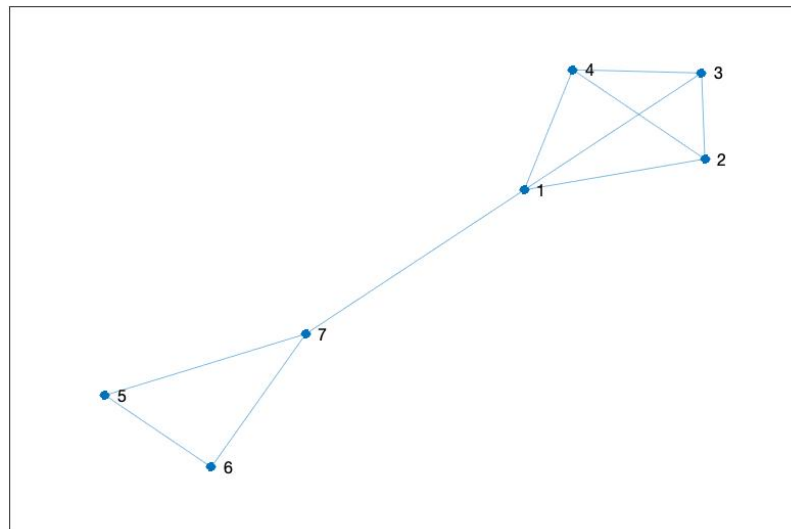


Figure 21. A sample of a graph.

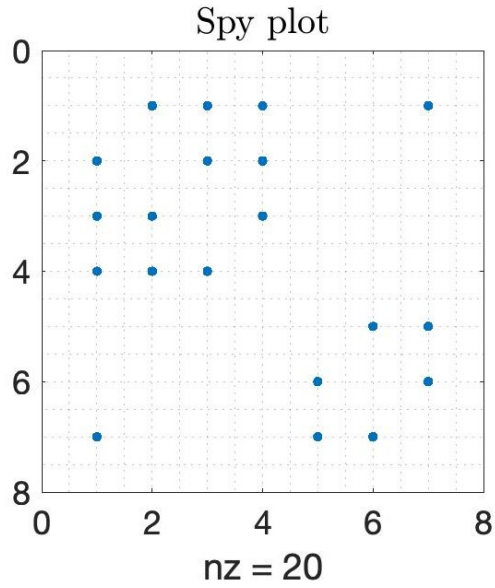


Figure 22. Sparsity plot.

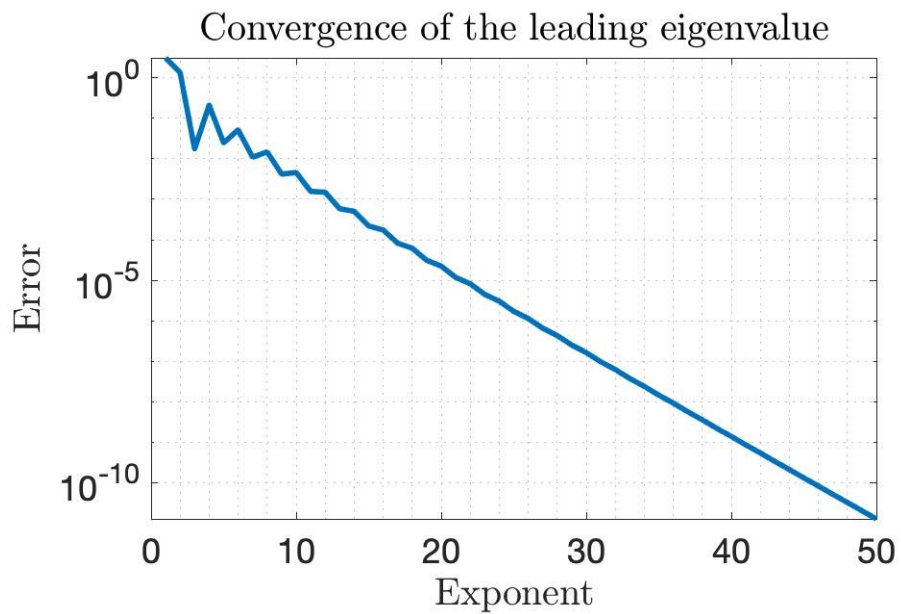


Figure 23. Convergence of the leading eigenvalue.

4.5 Algorithms - Active Learning-Quantum Amplitude Estimation

We discussed prior art for Quantum Amplitude Estimation in the previous section. Here, we present a Bayesian approach for QAE.

4.5.1 Quantum Bayesian Amplitude Estimation

Our contribution is to combine Quantum Expectation Estimation with Quantum Bayesian Phase Estimation, while sidestepping the eigenvector collapse problem in two distinct ways.

First Method: Eigenvector Invariance

For the first method, we proceed the same as in Knill et al. but instead of using the full phase estimation or iterative phase estimation we use a non-trivial variant of Quantum Bayesian phase estimation. Now, normally Quantum Bayesian Estimation demands that the state be initialized into an eigenvector. We use the insight that we can get away with the S rotation operator and the initial state $|\psi_0\rangle$, by a careful choice of θ_i . The problem is that in general $|\psi_0\rangle$ has components in the direction of both eigenvectors and therefore would pick up both phases $\pm\phi$. This would destroy the ability to construct a single pair of likelihoods (29) that is independent of the unknown eigenvector decomposition of the initial state. To get around this problem we introduce the idea of choosing θ such that the likelihood is invariant under a sign change of ϕ . This leads to two possible choices

$$\theta \in \{0, \pi/2m\}. \quad (144)$$

These two choices allow that for any m (and higher m 's should generally again be used as precision improves) there is always significant statistical distinguishability between measuring a zero versus a one.

This is because we have a difference between the two outcome probabilities that is either $\cos(m\phi)$ or $\sin(m\phi)$, the larger of which is always greater than $1/\sqrt{2}$.

This initial analysis is promising but it does not take into account the prior and which choice of m maximises the information gain (see below).

A further difference between our approach and Wiebe et al.'s, is that our prior may need to be bimodal to account for both possible phases. Finally, we envisage use-cases where prior knowledge of the phase can be incorporated in order to speed up convergence. For example, in our Quantum Homology work, we use Quantum Expectation Estimation to calculate Betti numbers, which are known to be integer values.

Maximizing Information Gain

Wiebe et al. and others study the reduction in variance after every Bayesian update to motivate choices of m and θ . A potentially more powerful approach is analyzing the information gain as defined by the expected utility, where the utility is defined as the Kullback-Liebler divergence between the prior and posterior distributions. We show that this approach not only leads to a pleasing formulation but also allows for a novel understanding of the Kitaev protocol.

We have derived an expression for the expected utility and also explored its implications for the Kitaev protocol. The main result is that the expected utility is the difference between the entropy of the outcome of the experiment averaging out prior knowledge and the entropy taking into account prior knowledge. The goal therefore is to choose m and θ to maximise this expected utility. Since we are restricting ourselves to just two possible values of θ (given m), we can simply write out the expected utility for these two choices of θ . First, we fix $\theta = 0$:

$$\text{EU}(m_i, \theta_i = 0) = H\left(\frac{1}{2} + \frac{\pi}{2} \hat{P}_i^E(m)\right) - \int_{-\pi}^{\pi} H\left(\frac{1}{2} + \frac{\cos(m\phi)}{2}\right) P_i^E(\phi) d\phi \quad (145)$$

where, $\hat{P}_i^E(m) := \frac{1}{\pi} \int_{-\pi}^{\pi} \cos(m\phi) P_i(\phi) d\phi$, is the even Fourier component (the cosine component) of the prior at frequency m and $P_i^E(\phi) := (P_i(\phi) + P_i(-\phi))/2$ is the even part of the prior.

Next, we fix $\theta = \pi/2m$:

$$\text{EU}(m_i, \theta_i = \pi/2m) = H\left(\frac{1}{2} + \frac{\pi}{2} \hat{P}_i^O(m)\right) - \int_{-\pi}^{\pi} H\left(\frac{1}{2} + \frac{\sin(m\phi)}{2}\right) P_i(\phi) d\phi \quad (146)$$

Therefore, the optimal choice of θ and m is the one that maximises the maximum of the above two expressions. The optimal choice is easily derived given the Kitaev assumptions. In the NISQ era, m is upper bounded by the coherence time and the optimal can in the worst-case scenario be calculated (in parallel, classically) after accurate numerical evaluations, since there are a finite number of m choices to simply search over. This graphically demonstrates a melding of classical and quantum methods. Furthermore, the iterative and Bayesian approach both do away with the Quantum Fourier transform, and now we can explicitly see that that work is pushed onto the classical side.

In order to gain an appreciation for maximizing the information gain, it is instructive to look at the Fisher information (a related, easier to work with quantity). As a first pass, we make use of the Discrete Fisher information (Pablo Sánchez-Moreno, et al.). Since there are only two outcomes, the maximization reduces to selecting the smallest m that makes one of two simple trigonometric expressions integer. It seems always possible to select a non-zero Fisher information value. However, as expected, m may need to continue increasing to achieve the maximum. It remains to be seen how this relates to the full information gain maximization.

Second Method: Tracking Eigenvectors

In “Efficient Bayesian phase estimation using mixed priors”, Ewout van den Berg, IBM Research, proposes a multi-eigenvalue Bayesian Phase estimation algorithm with a sophisticated posterior distribution expansion approach that intelligently switches between an accurate but expensive Fourier expansion to an acceptable and cost-effective normal distribution as the posterior becomes sufficiently accurate.

This second approach to dealing with multiple eigenvalues is now our preferred approach, since it allows for the full spectrum of parameters to be selected from, while only increasing the number of parameters to learn by one.

Indeed, since we have proposed focusing on Knill's expectation algorithm rewritten as phase estimation, we only have two eigenvalues to cater for. Furthermore, the two eigenvalues are simply the same value with opposite signs. This means there is only one degree of phase freedom and only one weight (since the two weights square sum to one) to learn.

For a quick demonstration, we execute Van den Berg's code to produce a figure of the error of the two eigenvalues of the S operator:

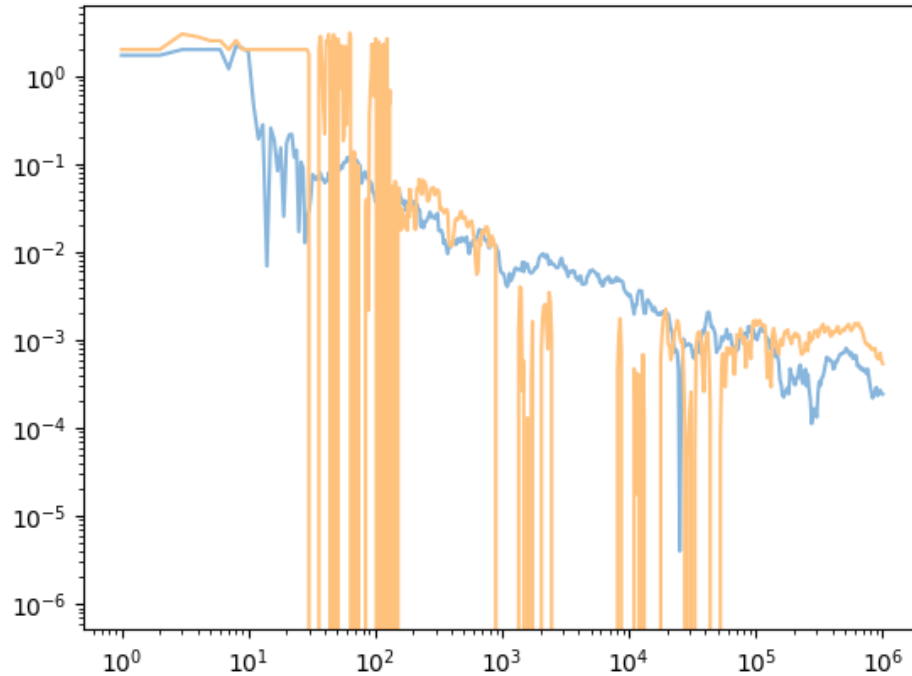


Figure 24. Mean eigenvalue error vs iteration; One curve for each of the two eigenvalues.

The problem with the above naïve multi-eigenvalue approach is that the code fails to capture the relationship between the two eigenvalues, namely the phase of the eigenvalues of S differ only in their signs. The two eigenvalues are learnt independently. Knowledge from one is not incorporated into the other and may actually be responsible for the erratic learning of one of the eigenvalues.

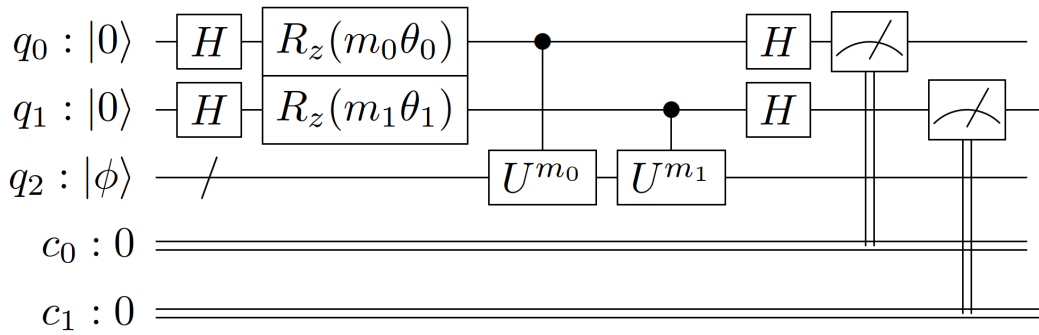
Therefore, we propose adjusting the Bayesian update formulae to account for this extra information, (using Van den Berg's notation):

$$P^{(l+1)}(\phi) = \frac{P^{(l)}(\phi) \left(P_{k,\beta}(m|\phi, w) + (-1)^m \sin(\beta) \sin(k\phi) (1 - w_1) \right)}{P_{k,\beta}(m)} \quad (147)$$

As can be seen, even though there are two eigenvalues, because of their relationship, there is no need to compute C_1 and C_2 separately. Only the difference survives with only a single weight featuring.

4.5.2 Using Extra Qubits

Finally, as an extra innovation, we propose using any extra ancilla qubits to increase the number of experiments per computational round. There are many ways to achieve this. One such way one, is by repeating the circuit in parallel across the other qubits. The same m 's and θ 's may be chosen since, after all, many samples are required to significantly update the posterior. Alternatively, the parameters may be chosen in a Bayesian experimental design fashion that chooses subsequent parameters in such a way as to maximize the subsequent expected information gain after the Bayes update of an earlier run (that is, earlier in processing time). Here is the circuit diagram for our expanded Bayesian Expectation Estimation algorithm:



It is interesting to note that these parallel circuits start to look exactly like Quantum Phase Estimation. The rotation before the controlled- U^m can be moved to after the control. Then the main difference between multi-qubit Bayesian phase estimation and the full quantum phase estimation is that there is only one un-controlled (but arbitrary) rotation in the Bayesian case but multiple (well-defined) controlled rotations in the inverse Fourier transform of the full phase estimation. These controlled rotations are the source of the speed up of the Fourier transform because all possible rotations (up to the accuracy implied by the number of qubits assigned to store the accumulated phases) of the Fourier transform are executed in one step. We forgo that advantage to save on controlled operations and rather opt to dynamically scan the ‘right’ rotations iteratively. This transfers the burden that was taken care of by the inverse Fourier transform onto the classical Bayesian post-processing. Indeed, the computational performance of the Bayesian estimation is terrible in the required number of bits of precision in the phase, namely exponential. But for fixed and relatively low precision, this does not concern us, and we are content with the exponential gains from being able to compute with exponentially large unitaries. In summary, the main advantage of not doing full Quantum Phase Estimation is that we can keep the circuit depth low (fewer controlled operations and fewer applications of U) while still being able to cumulatively learn from repeated measurements on noisy qubits.

4.6 Algorithms - Topological Data Analysis

We described the Lloyd et. al’s QTDA algorithm in the previous section. Here, we present our extension and implementation in QISKIT.

4.6.1 QTDA Implementation and Refinements

We report the successful implementation of portions of QTDA in IBM QISKit. Below we walk through the current progress, including, implementing the boundary matrix with polynomially many Pauli operators and a novel refinement of the calculation of the size of the kernel of the Laplacian.

In a celebrated paper Lloyd et. al demonstrate the first 'machine learning' algorithm with provable exponential speed-up on a quantum computer which does not suffer from the data loading problem. The authors combine Grover's search for state vector preparation (creating the Vietoris Rips simplicial complex of a given cloud of data-points) with Quantum Phase Estimation to calculate the dimensions of the kernel of the boundary matrix restricted to the different k -simplices of the simplicial complex (representing the size of different structures). The dimensions of the restricted kernel allow for the calculation of the coveted topological global invariants called the Betti numbers (representing the size of different 'non filled-in' structures: clusters, loops, voids, etc.). The whole process is repeated (in superposition) for different course-grained simplicial complexes producing a series of Betti numbers called the persistent homology (representing robust-across-scale structures). In this paper we report the successful implementation of portions of this algorithm in Qiskit and the running of this algorithm on real hardware. As part of the implementation we derive an original polynomial-number-of-terms Pauli decomposition of the boundary matrix realisable on the IBM Q architecture. Finally, we apply QTDA to a pertinent cosmology use-case, namely increasing the sensitivity of persistent homology to detect non-Gaussianity in the Cosmic Microwave Background (CMB) radiation. We show that increasing the number of data points under consideration, increases the sensitivity, thereby necessitating the use of exponentially more computation. This rapid increase in computational requirements can be met by quantum computers but not by classical supercomputers, which, after a certain number of points, would fail to calculate the Betti numbers with any realistically available space and time resources.

4.6.2 Overview of Implementation and Refinement

TDA has garnered significant interest in the applied mathematics community. It is one of the few 'big data' algorithms that can consume massive datasets and reduce them to a handful of global and interpretable signature numbers laden with predictive and analytical value. Unfortunately, the classical algorithm has some significant computational demands which are currently addressed by sampling. We demonstrate a use-case where sampling obviates the benefits of persistent homology, necessitating the full detailed calculation which is unreachable by classical supercomputers. Fortunately, a new model of computation can be employed to satisfy these demands: quantum computing.

With the realization of the power of quantum computing in the 1980's, there has been a continued search for new algorithms that can exploit the specialized power of quantum computers. Since the power of quantum computing can be characterized as the ability to perform a subset of linear algebra operations in exponentially large spaces, it is entirely

plausible that classical TDA could be cast into quantum form. Classical homology's own breakthrough was the casting of the powerful notion of continuous connected topological paths into a discrete linear algebraic setting. This has opened the door for a quantum version that can perform the core graph theoretic and linear algebraic operations. The two operations which find quantum form are clique detection and kernel dimension estimation. The first is achievable in approximation as Grover's search and the second as phase estimation as proposed by Lloyd et. al. The final result is the calculation of the Betti numbers as the dimension of the kernel of the generalized Laplacian. The kernel is revealed when QPE returns zero for the eigenvalue of the Laplacian. Detecting zero not only reveals the presence of the zero eigenspace (representing connected components, loops, voids and so on) but also the relative size of the zero eigenspace through measurement statistics. This is achieved by executing QPE on a random simplex from the simplicial complex and counting the number of times QPE projects onto the zero-eigenspace.

Finally, coupled with recent state preparation techniques including those which leverage variational methods, to potentially replace the Grover step and a novel, experimental rewriting of the eigenspace calculation as an expectation, this opens the door for a NISQ implementation of Quantum TDA.

4.6.3 Boundary Operator

The boundary operator on k -dimensional simplices is defined as

$$\partial_k |s_k\rangle = \sum_l (-1)^l |s_{k-1}(l)\rangle \quad (148)$$

where $|s_{k-1}(l)\rangle$ is a one-dimensional less simplicial complex than $|s_k\rangle$ with the same vertex set as it, except for vertex l .

4.6.4 Boundary Matrix

In order to efficiently construct the boundary operator for all k , on an IBM Q device, we need to represent it in one go, with polynomially many Pauli matrices. To assist with this description, define B_{01} :

$$\begin{aligned} B_{01} &= (\sigma_x + i\sigma_y)/2 \\ &= \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \end{aligned} \quad (149)$$

B_{01} is written in terms of Pauli matrices (σ_x and σ_y) however, note that σ_y 's co-efficient is not real, therefore the boundary operator is not Hermitian.

The boundary matrix can now be written:

$$\begin{aligned}
\partial &= \oplus \partial_k \\
&= B_{01} \otimes I \otimes I \otimes \dots I \\
&\quad + \sigma_z \otimes B_{01} \otimes I \otimes \dots I \\
&\quad + \sigma_z \otimes \sigma_z \otimes B_{01} \otimes \dots I \\
&\quad \vdots \\
&\quad + \sigma_z \otimes \sigma_z \otimes \sigma_z \otimes \dots \otimes B_{01}
\end{aligned} \tag{150}$$

which is the direct sum of the ∂_k 's. There are n single qubit operations in the tensor product in each term above, one for each vertex in the simplicial complex. The B_{01} term 'removes a vertex' from a simplex and the σ_z 's provides the right sign by noting the position of that vertex mod 2 relative to other vertices.

4.6.5 Graph Laplacian

Adding the full boundary matrix to its complex conjugate, naturally produces a Hermitian operator:

$$B = \partial^\dagger + \partial \tag{151}$$

Taking the square of this full boundary matrix produces the Laplacian:

$$L = B^2 \tag{152}$$

However, we are not interested in the full boundary matrix, but rather the boundary restricted to the actual simplices in the complex and of the right order. Therefore what is required are six projections, two before the first boundary matrix, two in the middle and two at the end:

$$L = P_{\psi,k} B P_{\psi,k-1,k+1} B P_{\psi,k} \tag{153}$$

Each P represents both a projection onto the simplicial complex and onto the right k 's. The first projection can be done classically as a classical sample, if only one Betti number is desired.

The kernel of this matrix corresponds to 'cycles' in each of the k -dimensional subspaces.

The QTDA algorithm applies the quantum phase estimation of L onto a random simplex obtained by turning the simplicial vector into a uniform mixture over the simplices, thereby randomly selecting a simplex. Such a random simplex then gets projected onto the eigenvectors of the Laplacian restricted to the simplicial complex. Since the simplex was randomly chosen, repeatedly selecting a simplex and then projecting it 'feels' out the size of the respective eigenspaces.

4.6.6 Projection onto the k -simplices

Projection onto the k -simplices can happen in quantum parallel, by conditionally rotating the different k -subspaces. This conditioning is achieved by performing a controlled-permutation operation (by permutation we mean add-one-with-wrapping) on each vertex qubit into a common count register. It has the effect that simplices with the same *number* of vertices, even if the vertices are different, all rotate the count register to the same value, thereby becoming entangled with the same basis state in the count register. When the count register is measured and k is received, it amounts to a projection onto the k -simplices. Even though the k is randomly received, since we are interested in all k nothing is lost (except when matching projections of different projectors).

Now, for the description of the actual circuit that achieves the conditional permutation. This turns out to be surprisingly complicated. A permutation circuit by itself is not too difficult. The key insight is seeing the 4×4 permutation matrix as a swapped, flipped, CNOT. Then for larger sizes the circuit is extended by simply swapping two rows in the larger matrix with a multi-controlled NOT.

The difficulty comes in when conditionalizing permutation as efficiently as possible. One particular interesting approach, is to perform the conditional gate in the diagonal basis, opening the door for an efficient diagonal matrix construction (multiplexed- R_z). First, however a unitary matrix that diagonalises the control-permutation is required, fortunately this turns out to be beautifully intuitive and easy. The permutation matrix is diagonalised by the Fourier transform. Indeed, oscillating functions are the needed eigenvectors with the eigenvalue corresponding to the phase required in the Fourier domain to translate/permute in the time domain.

We report the successful and efficient circuit construction in Qiskit. In particular the size of the count register is $\log(n)$ and the complexity of the two change of bases are $O(\log^2(n))$. We see that the simplices indeed get entangled with the correct count labels.

4.6.7 Novel Reformulation as Expectation

Since the (fully connected sub-complex) Laplacian has two eigenvectors, 0 and 1, and we are interested in the norm of the component of the random vector in the kernel, we can rewrite the QPE eigenspace calculation with an expectation calculation if we adjust L such that the zero eigenvalue becomes the one eigenvalue.

Here follows a detailed treatment of the above the discussion.

B has eigenvalues $\{-\sqrt{n}, 0, \sqrt{n}\}$, where n is the number of vertices (in the fully connected sub-complex).

Therefore $\frac{1}{n}B^2$ has eigenvalues $\{0,1\}$. Subtracting this from the identity switches the eigenvalues of the eigenspaces:

Thus K becomes the projection onto the kernel of B (equivalently the kernel of B^2 ,

$$K = I - \frac{1}{n}B^2 \quad (154)$$

Next, let $|\psi\rangle$ be the output of the Vietoris-Rips Grover's search algorithm, namely the simplicial complex:

$$|\psi\rangle = \frac{1}{|S_k|} \sum_{s_k \in S_k} |s_k\rangle \quad (155)$$

where S_k is the set of k -simplices in the simplicial complex and $|s_k\rangle$ is the binary-encoded quantum state of a specific simplex s_k .

Turn $|\psi\rangle$ into a uniform mixture by copying the label k into an ancilla register and then measure that register. This chooses a random k' from the simplicial complex.

Now write $|s_{k'}\rangle$ in the eigenbasis of the Laplacian, B^2 :

$$\begin{aligned} |s_{k'}\rangle &= \sum_j c_j^i |j\rangle \\ |s_{k'}\rangle &= \sum_{j \in \Lambda_0} c_j^i |j\rangle + \sum_{j \in \Lambda_1} c_j^i |j\rangle \end{aligned} \quad (156)$$

where, Λ_0 are the indices of the eigenvectors with eigenvalue zero and Λ_1 containing those with eigenvalue unity.

Now, calculate the expectation of K :

$$\begin{aligned} \langle s_{k'} | K | s_{k'} \rangle &= \langle s_{k'} | (I - B^2) | s_{k'} \rangle \\ &= 1 - \sum_{j \in \Lambda_1} (c_j^i)^2 \\ &= \sum_{j \in \Lambda_0} (c_j^i)^2 \end{aligned} \quad (157)$$

This is the norm of the component of the simplex in the kernel of the Laplacian. Repeating this and averaging, is proportional to the size of the kernel, allowing us to calculate the Betti number:

$$\begin{aligned} \text{Betti}_k &= |S_k| \times \text{Ker}(L) \\ &= |S_k| \times E_{k' \sim U}(\langle s_{k'} | K | s_{k'} \rangle) \end{aligned} \quad (158)$$

Such a rewriting as an expectation, allows the use of NISQ compatible Quantum Expectation Estimation algorithms including one developed by the authors called Bayesian Quantum Expectation Estimation without Eigenvector Collapse.

Now, this technique only works for fully connected sub-complexes. In order to generalize, we have to cater for arbitrary complexes. Fortunately, arbitrary spectrums may be “turned” into spectrums with just two eigenvalues. This can be achieved by applying a step function to the spectrum after which the expectation may be calculated as above. Indeed, the inspiration for this approach comes precisely from attempts to approximate the rank of a Hermitian matrix. Since Betti numbers relate to the kernel (and hence the rank) of the Laplacian, this rank approach (see next section) can directly be seen to be appropriate. Furthermore, this quantum rank approximation has potentially far-reaching uses beyond Betti number calculation.

4.7 Algorithms - Betti Number by Chebyshev Approximation

Previously, we described a stochastic polynomial approximation method to compute the Betti numbers of data points. Here, we present few numerical results illustrating the performance of this method.

Table 2. Betti number estimation

Data Shape	Scale ϵ	Betti	True	Restricted Laplacian deg = 8	Projected Boundary deg = 8
Triangle	0.5	Betti 0	1	1	1
		Betti 1	1	1	1
		Betti 2	0	0	0
Square	0.5	Betti 0	1	1	1
		Betti 1	1	1	1
		Betti 2	0	0	0
		Betti 3	0	0	0
Pyramid	0.5	Betti 0	1	1	1
		Betti 1	1	1	1
		Betti 2	0	0	0
		Betti 3	0	0	0
Pyramid	0.7	Betti 0	1	1	1
		Betti 1	0	1	0
		Betti 2	0	1	0
		Betti 3	0	0	0
		Betti 4	0	0	0
Random 5 points, 4 dim	0.4	Betti 0	2	2	2
		Betti 1	0	0	0
		Betti 2	0	0	0
		Betti 3	0	0	0
		Betti 4	0	0	0
Random 8 points, 6 dim	0.6	Betti 0	2	2	-1
		Betti 1	0	1	-2
		Betti 2	0	0	0
		Betti 3	0	0	0
		Betti 4	0	0	0

Here, we present some preliminary results for computing Betti numbers of few points and shapes using our rank estimation technique. We consider two approaches; one is to compute the kernel dimension of a restricted Laplacian matrix. Second, is to compute the kernel of a projected boundary sum matrix. Table 2 presents the results.

In the table we consider data-points that form different shapes (first column), and Betti numbers at different complex scale ϵ (second column) are computed. The true betti numbers are given in third column. The betti numbers computed by our Chebyshev method using the two different matrices (restricted Laplacian and projected boundary sum) are given in the last two columns. The Chebyshev method yields fairly accurate rank even for small degree say 6 to 8. The matrices have larger nonzero eigenvalues, therefore, we do not seem to need high degree. Our methods seem to be more accurate when the scale ϵ is small. Note that, when $\epsilon < 0.6$, our method yields accurate Betti numbers, however, when

the scale is large, we need a higher degree. The degree $m = 8$, the number of starting vectors was $n_v = 30$, and the threshold $\tau = 0.1$.

4.8 Applications - Topological Classifier

Previously, we described the topological classifier based on Betti number of datapoints. Here we present digit classification results. In Table 3, we present the Betti numbers for the ten digits (0-9) at different scale ϵ using 6-point representation of the digits. We list the true Betti value and the value estimated by our Chebyshev method. We used the restricted Laplacian approach since it is more accurate at lower degree. The degree $m = 8$, the number of starting vectors was $n_v = 30$, and the threshold $\tau = 0.1$. First, we note that the Chebyshev method is fairly accurate, yielding the correct Betti numbers in almost all cases. Next, we note that with just 6 points and Betti numbers at 4 scales, we will be able to distinguish many digits from others (1 from 8, or 4 from 6 etc). However, the Betti numbers of (1,7), (2,3,5) and (6,9) are identical in our 6-point, 4 scale setting. This implies, we will need more points and scales to distinguish between these numbers.

Table 3. Betti numbers of digits at different scale

Digits	Betti	True value				Estimated value			
		$\epsilon = 0.5$	$\epsilon = 0.75$	$\epsilon = 1.0$	$\epsilon = 1.5$	$\epsilon = 0.5$	$\epsilon = 0.75$	$\epsilon = 1.0$	$\epsilon = 1.5$
Zero 0	Betti 0	2	2	1	1	2	2	1	1
	Betti 1	0	0	1	0	0	0	1	0
	Betti 2	0	0	0	0	0	0	0	0
One 1	Betti 0	1	1	1	1	1	1	1	1
	Betti 1	0	0	0	0	0	0	0	0
	Betti 2	0	0	0	0	0	0	0	0
Two 2	Betti 0	2	1	1	1	2	1	1	1
	Betti 1	0	0	0	0	0	0	0	0
	Betti 2	0	0	0	0	0	0	0	0
Three 3	Betti 0	2	1	1	1	2	1	1	1
	Betti 1	0	0	0	0	0	1	0	0
	Betti 2	0	0	0	0	0	0	0	0
Four 4	Betti 0	1	1	1	1	1	1	1	1
	Betti 1	0	0	0	1	0	0	0	1
	Betti 2	0	0	0	0	0	0	0	0
Five 5	Betti 0	2	1	1	1	2	1	1	1
	Betti 1	0	0	0	0	0	0	0	0
	Betti 2	0	0	0	0	0	0	0	0
Six 6	Betti 0	1	1	1	1	1	1	1	1
	Betti 1	1	0	0	0	1	1	0	0
	Betti 2	0	0	0	0	0	0	0	0
Seven 7	Betti 0	1	1	1	1	1	1	1	1
	Betti 1	0	0	0	0	0	0	0	0
	Betti 2	0	0	0	0	0	0	0	0
Eight 8	Betti 0	1	1	1	1	1	1	1	1
	Betti 1	2	0	0	0	2	0	0	0
	Betti 2	0	0	0	0	0	0	1	0
Nine 9	Betti 0	1	1	1	1	1	1	1	1
	Betti 1	1	0	0	0	1	0	0	0
	Betti 2	0	0	0	0	0	0	0	0

4.8.1 Improved Hybrid Topological and Statistical Features Digits Classifier

As we observed in the previous section, the topological features (Betti Numbers) can be used to develop topological classifiers. We considered digit classification, and showed how we can distinguish certain digits from others using just the Betti numbers at few resolution scales ϵ . However, we noted that certain digits cannot be distinguished from one another (e.g., 1 and 7; 6 and 9) using just the topological features. Here, we discuss possible improvements for this digit classifier.

One potential approach to improve the topological classifier for digit classification is to consider statistical measures of the data points, such as mean, variance, Kurtosis and others, as additional features along with the topological features (Betti numbers). These global features are easy to compute for the data points. With just a small set of statistical and topological features, we can classify points that come from different digits.

Table 4. Topological and statistical features of digits

Digits	Betti	Resolution scale				Statistical		
		$\epsilon = 0.5$	$\epsilon = 0.75$	$\epsilon = 1.0$	$\epsilon = 1.5$		Dim 1	Dim 2
Zero 0	Betti 0	2	2	1	1	Mean	0.5	1.0
	Betti 1	0	0	1	0	Variance	0.25	1.0
	Betti 2	0	0	0	0	Kurtosis	-2.0	-2.0
One 1	Betti 0	1	1	1	1	Mean	1.0	1.0
	Betti 1	0	0	0	0	Variance	0.0	0.625
	Betti 2	0	0	0	0	Kurtosis	-3.0	-1.64
Two 2	Betti 0	2	1	1	1	Mean	0.5	1.0
	Betti 1	0	0	0	0	Variance	0.25	0.625
	Betti 2	0	0	0	0	Kurtosis	-2.0	-1.64
Three 3	Betti 0	2	1	1	1	Mean	0.4	1.0
	Betti 1	0	0	0	0	Variance	0.24	0.5
	Betti 2	0	0	0	0	Kurtosis	-1.83	-1.3
Four 4	Betti 0	1	1	1	1	Mean	0.5	1.25
	Betti 1	0	0	0	1	Variance	0.25	0.1875
	Betti 2	0	0	0	0	Kurtosis	-2.0	-0.67
Five 5	Betti 0	2	1	1	1	Mean	0.5	1.0
	Betti 1	0	0	0	0	Variance	0.25	0.625
	Betti 2	0	0	0	0	Kurtosis	-2.0	-1.64
Six 6	Betti 0	1	1	1	1	Mean	0.4	0.8
	Betti 1	1	0	0	0	Variance	0.24	0.56
	Betti 2	0	0	0	0	Kurtosis	-1.83	-1.15
Seven 7	Betti 0	1	1	1	1	Mean	0.75	1.25
	Betti 1	0	0	0	0	Variance	0.1875	0.6875
	Betti 2	0	0	0	0	Kurtosis	-0.67	-1.37
Eight 8	Betti 0	1	1	1	1	Mean	0.5	1.0
	Betti 1	2	0	0	0	Variance	0.25	0.67
	Betti 2	0	0	0	0	Kurtosis	-2.0	-1.5
Nine 9	Betti 0	1	1	1	1	Mean	0.6	1.2
	Betti 1	1	0	0	0	Variance	0.24	0.56
	Betti 2	0	0	0	0	Kurtosis	-1.83	-1.15

In Table 4, we present the topological and statistical features numbers for the ten digits (0-9), when we use 6 points representations of the digits on a 2D plane. Along with the Betti

numbers at 4 different scales ϵ , we include three statistical features, namely mean, variance and Kurtosis for the two dimensions. Note that we now have a unique feature set for most of the digits, and we can distinguish the digits from each other with just a small set of features. This experiment shows that adding few statistical features to the topological features improve the classifiers significantly, and we will be able to distinguish most of the digits from each other. However, we note that, we still cannot distinguish 2 and 5, since they are vertical mirror images and are statistically invariant.

4.9 Applications - Pricing & Risk Analysis

The quantum amplitude estimation algorithm finishes with the application of an inverse Quantum Fourier Transform (QFT). Although QFT is not the most expensive part in amplitude estimation, it is worth noting that it can be replaced by the Approximate QFT algorithm described §3.15. We tested our option-pricing algorithm with both, the exact and approximate inverse QFT (with approximation order equal to 1) and could not see significant differences in the results. The Approximate QFT seemed to capture the important features of the result and lead to the correct values. Figure 25 illustrates results for the exact QFT and the differences when using Approximate QFT; the differences are marginal and do not affect the result.

Another classical approach to estimate option prices is based on Fast Fourier Transforms (FFT) [25]. First, the characteristic function of the payoff given an uncertainty model is constructed, and then, FFT is used to retrieve the corresponding probability density function, which can be used to derive the expected value. Since quantum computers are very good at performing Fourier transforms, one might think this approach could be suitable for a quantum computer as well. However, it appears that preparing the corresponding characteristic function on a quantum computer can be more challenging than following the approach described above. Furthermore, applying the Fourier transform leads to a state where the amplitudes correspond to the probabilities, and not to the square root of the probabilities as usually required in this setup. Last, if this approach would work and we could produce a quantum state that represents the problem correctly, we still would have to apply quantum amplitude estimation to get the intended quadratic speedup. Thus, we conclude that QFT might be a good way to load parts of the data (cf. § 4.11), but the classical approach that uses FFT to derive the option price seems not to be applicable on a quantum computer.

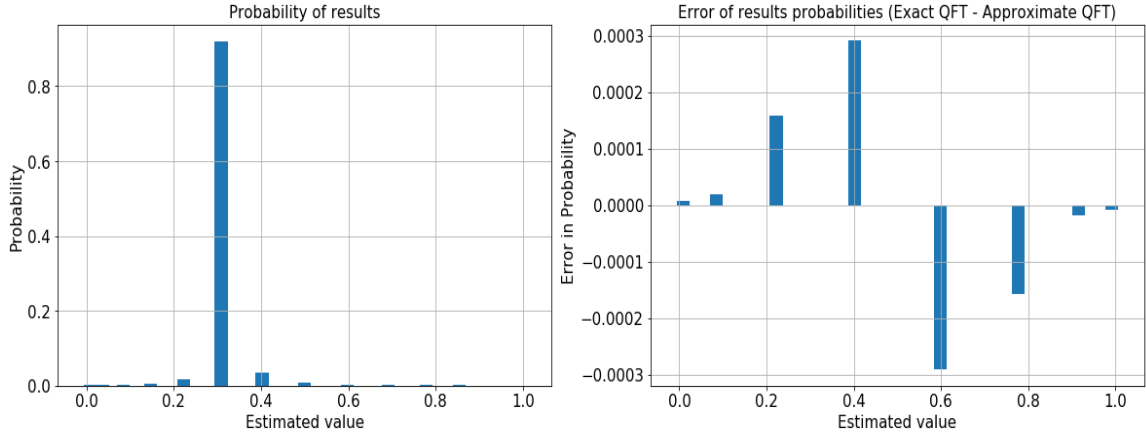


Figure 25: (left) Exact probabilities of different estimated values, where the maximum probability corresponds to the result of the amplitude estimation algorithm. (right) Error in probability if inverse QFT is replaced by inverse Approximate QFT.

4.10 Applications - Variational Heuristics for Combinatorial Optimization

Our study leads to the following observations. Even if the variational form used in our experiments is guaranteed to span the ground state, the resulting optimization problems are difficult for classical local optimization methods, e.g., gradient descent: these methods often get stuck in local minima that may be very far from the optimal solution. Global optimization seems to be more reliable, but the performance of all algorithms is very problem-dependent. In particular, the variational approach has difficulties on the problem classes that also appear to be the hardest for a classical Branch-and-Bound solver. This difficulty is likely related to the concept of “density” of the problem representation. Indeed, the performance of the classical Branch-and-Bound solver can be explained by the fact that the classical representation of these difficult problems as binary quadratic optimization problems leads to dense matrices, which are known to be harder to deal with than sparse matrices. Similarly, for the variational approach our experiments show that the number of distinct eigenvalues of the Hamiltonian is a good indicator of the difficulty of a problem instance; this can be explained in light of the fact that eigenpairs represent stationary points of the optimization problem. The number of distinct eigenvalues is often related to the number of terms in the representation of the Hamiltonian as a weighted sum of Paulis, depending on the weights. This observation may provide an easy way to quickly estimate the difficulty of finding the ground state with the variational approach on a given Hamiltonian.

Furthermore, for these classes of problems, which yield diagonal Hamiltonians, it is unclear if two-qubit entangling gates help accelerate convergence to the ground state. Finally, regardless of the method used and the problem class, attaining a good approximation ratio or attaining a good probability of sampling the optimal solution seems to require a large number of iterations of the classical optimization routine; the scaling of the performance with respect to problem size (limited to what can be ascertained in a study that considers at most 18 qubits) indicates that as problem size increases, the number of necessary iterations grows more than linearly, which is expected when dealing with nonconvex

problems. Considering the crucial role played by the variational form in this type of method, it is important that future research efforts carefully consider the choice of variational form and its effect on the performance of the optimization algorithm. An extended version of this report, with additional data and the corresponding analysis, can be found at [26].

4.11 Data-porting - Approximate Quantum Fourier Transform

4.11.1 Preliminary Results.

We have implemented this approximation in both qiskit and qutip as a testing sandbox and as a framework to use and incorporate in future algorithmic development.

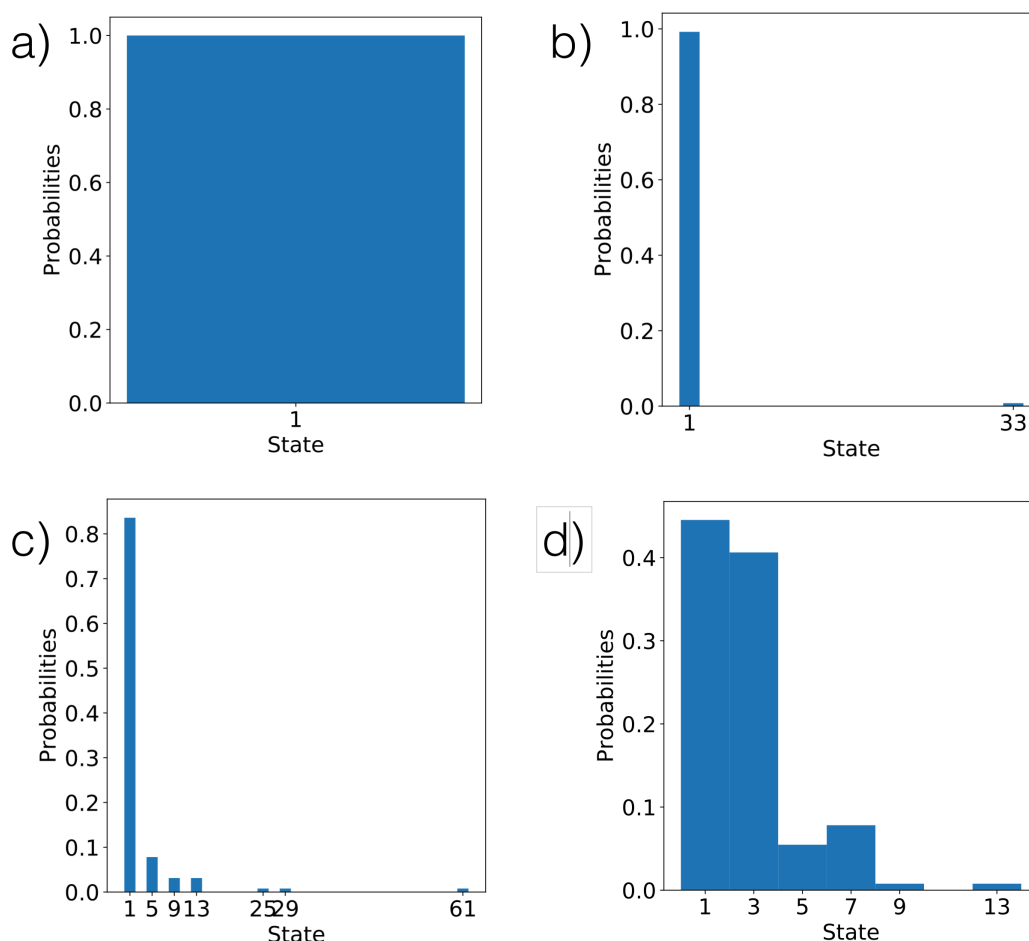


Figure 26. Approximated QFT with varying degree of approximation m

For a first result, we can test the accuracy of the approximate transform with varying degrees of decimation m , which is shown in Figure 26. The transforms are performed on an eigenstate of the full quantum Fourier transform. As can be seen, the approximation holds well up to a decimation of 50% of the gates giving a complexity advantage of $\mathcal{O}(N)$.

Another view of the error behavior of the approximate quantum Fourier transform can be seen in Figure 27, which in its top row shows a signal occupying the bottom 64 frequency components, out of 1024, represented by 10 qubits, with the Fourier transform of that signal and its AQFT with $m = 1$, and the error of that AQFT. Here the error is normalized by the Euclidean norm of the signal. In the second row, the errors for $m = 2, 3$, and 5 are shown. Note that the error is unsystematic, in the sense that it is roughly uniform noise in the parts of the spectrum where the signal is zero, holding true for $m \geq 2$ even when the total error is relatively large.

As seen also in Figure 28 and Figure 29, with similar plots but for medium and high frequency signals, the AQFT behaves similarly for these.

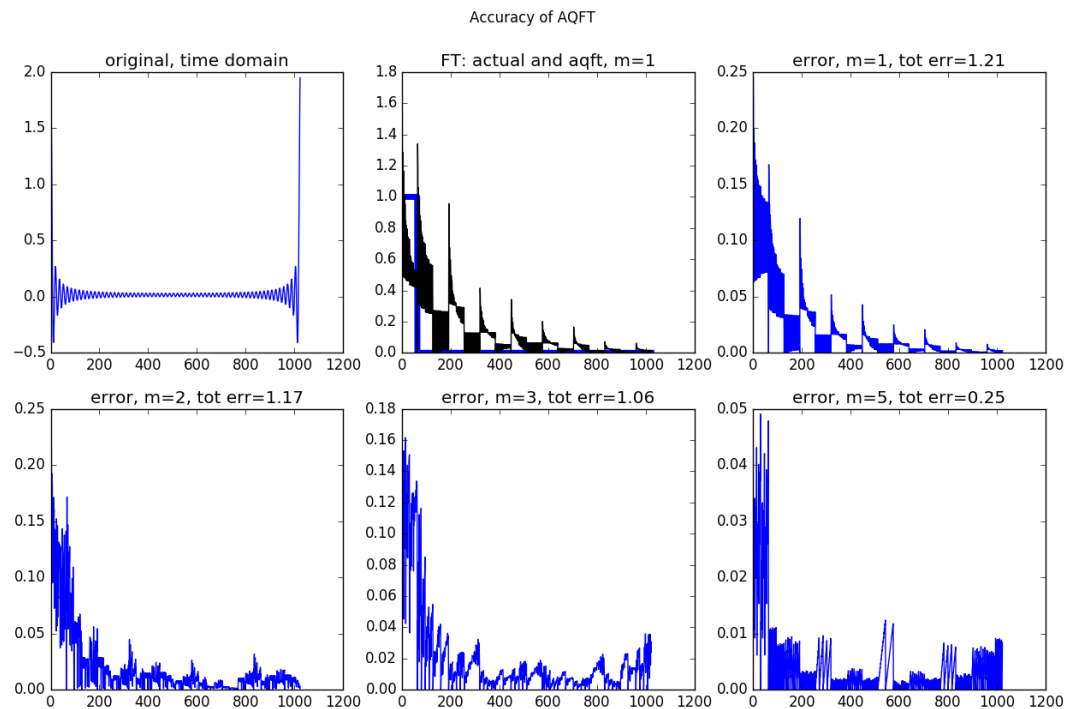


Figure 27. Error of approximate QFT, low frequency signal

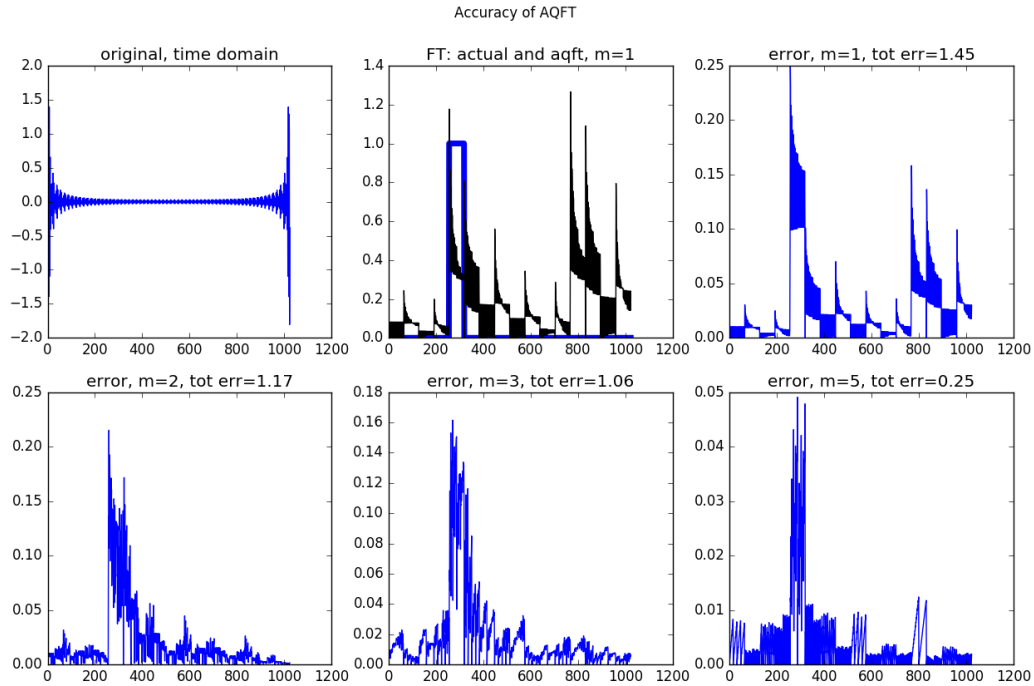


Figure 28. Error of approximate QFT, medium frequency signal

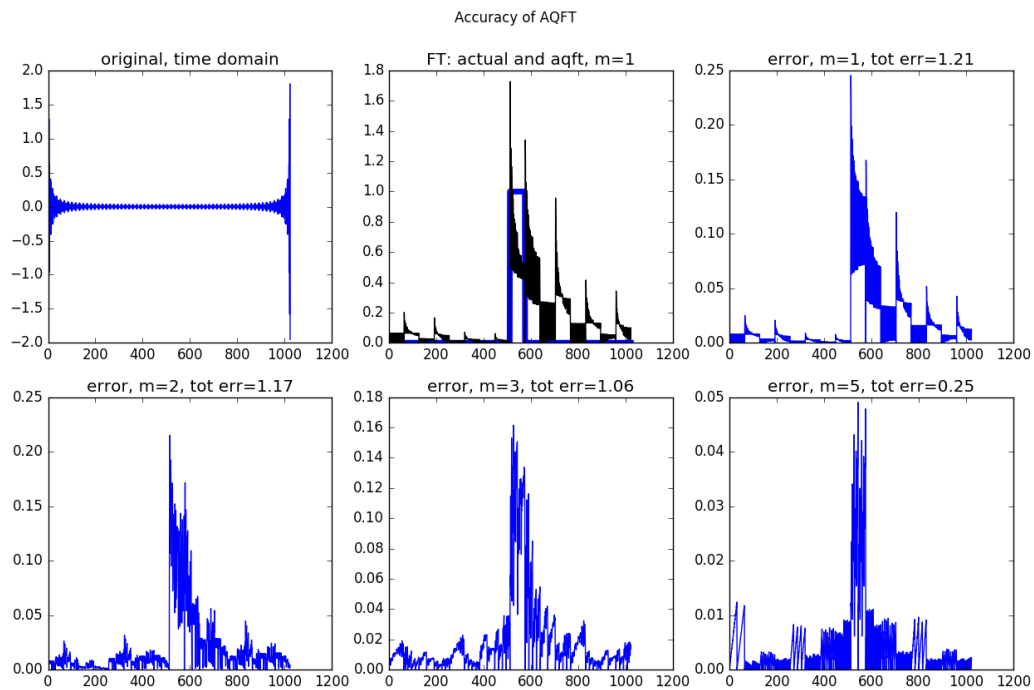


Figure 29. Error of approximate QFT, high frequency signal

4.11.2 Further Theoretical considerations

With the known sparsity of classical signal, it becomes possible to approximately search for the Fourier coefficients that are present in the signal. Once knowledge of the locations of the Fourier values are obtained, a shorter-depth full Quantum Fourier transform circuit can be designed that is no longer approximative but is more efficient and can possibly now fit within the coherence time of real devices.

This is achieved by translating the mid-point between the smallest and largest frequencies to zero and then performing a truncated full Fourier transform. The savings achieved are due to the fact that two-qubit gates between distant qubits have been systematically removed.

4.12 Data porting - Narrow Band and Sparse Quantum Fourier Transforms

4.12.1 Use of AQFT for Narrow-Band Signals

Our novel use of the AQFT to determine the full QFT of a strong narrow-band signal (high amplitude across the band, and width less than half the range), proceeds as follows. Since we are guaranteed a narrow-band signal, the first goal is to find the location of the most significant bit of the highest frequency. To do so we can perform an AQFT on the most significant bit only for a cost of $O(1)$ ($m=1$). As long as the signal is strong across the narrow-band, frequency artefacts do not alter the locations of the peaks. With the discovery of the most significant bit, say at k , at a cost of $O(n)$, the next step is to multiply the input signal by $\omega^{-2^k j}$ (we are of course assuming we have multiple copies of the input). This is achieved by constructing a diagonal unitary $D = R_z^{2^{n-1}} \otimes R_z^{2^{n-2}} \cdots \otimes R_z$, where $R_z = \begin{pmatrix} 1 & 0 \\ 0 & \omega^{2^k} \end{pmatrix}$. This is at cost $O(1)$.

This multiplication in the time domain is translation in the Fourier domain, such that the frequency corresponding to angle ω^{2^k} is now positioned at zero. The highest frequency might still have bits at $k - 1$ and lower while the lowest frequency might now have at most a k most-significant bit (if for example it was previously at zero).

We then repeat this procedure $O(k)$ (a maximum of $2k$) times for a total cost of $O(nk)$. This reads out the location of the highest and lowest frequencies perfectly but not the amplitudes. Finally, we perform a translation on the original signal such that the midpoint frequency is at zero. We may now perform an exact QFT on significantly fewer bits, the number of bits dictated by the width of the band. The depth cost of the full QFT is upper-bounded by $O(k^2)$ or possibly $O(k)$ using Cleve et. al. Most importantly, any one run of the QFT is then not more than depth $O(k)$. If we are content with a final approximative QFT, this can possibly be achieved in $O(\log k/p)$, with p the precision, using Cleve et. al.

4.12.2 Use of AQFT for Sparse Signals

For k -sparse signals (k has a different meaning, than above: the number of frequencies present), the same scheme of finding the largest and lowest frequencies can be employed. Once discovered a quicker technique may be employed than a reduced full QFT: the extreme frequencies' amplitudes can be obtained by translating them to zero and projecting onto the uniform state. This can be done by applying the Hadamard gate to each qubit (equivalent to an ($m=1$) AQFT; this tensored Hadamard operator's first row is the uniform state) and collecting statistics of the zero computational basis state. Once the location and amplitude of the extreme frequencies are known these known signals can be added to the original signal to remove that frequency and the process can continue (due to linearity, this can be affected, without actually performing the sum). Therefore, the total cost is $O(nk)$ with no need to run a full QFT at the end. Any one run has depth $O(I)$.

5.0 CONCLUSIONS

In this project we have addressed three complementary dimensions of quantum computing, namely the development of quantum algorithms, the scoping of applications, and an attempt to address the intrinsic data-porting bottleneck problem.

We made a conscious effort to conceive new algorithms that can provably offer notable (i.e., exceeding a square root factor) advantage over their classical counterparts. Our primary goal was to consider algorithms that can address, under general settings, useful problems within realistic environments comprising near-term noisy quantum devices. For this reason we avoided making overly generous assumptions on the nature of the available computation (e.g., fault-tolerant computation) as well as concerning the existence of data structures already arranged in convenient structures (e.g., QRAM) with no regard to the effort of generating such structures or the cost of porting classical data into a quantum device.

Our study into variational methods has taught us that this class of approaches may bear some promise in performing hybrid classical-quantum computation applicable for near-term devices, yet it will be essential to figure out whether a suitable ansatz can be defined so as to ensure that despite substantially (exponentially small compared to the quantum Hilbert space) reduced degrees of freedom under classical optimization control, a set of desired quantum spaces can be still be reached. The variational form needed is obviously problem dependent. In domains such as chemistry it may be possible to satisfy such suitability requirements, yet it is not clear that this can be done more generally in a broader context (e.g., finance, machine learning, etc.).

Another line of effort is the hybridization of classical and quantum computation by harnessing the machinery of artificial intelligence and statistical inference, in the form of Bayesian active learning to aid in performing general purpose phase/amplitude estimation. Our conclusion in this case is that active-like learning can often break the boundaries of noisy sampling, and enable rapid phase/amplitude estimation, as long as the underlying phase/amplitude possesses an exploitable structure. Given the popularity of kernel computations in quantum computing, we believe that the adaptation and further assessment of the value that these approaches bring may be of great impact.

Quantum walks have been studied extensively in the quantum computing literature. Under appropriate conditions, this class of methods can provably offer notable computational advantage over their classical equivalent. In addition to quantum walks, we have looked closely into structured Markov processes and structured random walks. Algorithms based on these general classes of Markov processes provide runtime improvements over the Quantum Walk algorithms in general, and especially in the cases where $O\left(\frac{1}{\epsilon}\right)$ dominates $O(\sqrt{N \log N})$. In the common latter cases our algorithm provides dramatic speedups over the Quantum Walk algorithm and, for the first time, addresses the well-known $O\left(\frac{1}{\epsilon}\right)$ computational complexity bottleneck for Quantum Walk algorithms. These quantum algorithms we devised can be exploited to address a much larger class of problems beyond

random walks. We also devised a graph analysis algorithm for the purpose of clique detection.

Our current understanding is that one of the few quantum algorithms that does not suffer from the data porting bottleneck is quantum Topological Data Analysis. While previous algorithms have made unrealistic assumptions concerning their quantum implementation, we have set to explore noise-robust alternatives for the key primitives of these algorithms, so as to circumvent noisy gates and deep circuit propositions. By bringing in notions from matrix sketching and Chebyshev approximations, we believe that we have made QTDA an appealing candidate for the task.

Lastly, we have studied various alternatives to the famous Quantum Fourier Transform (which was originally devised by a former member of the IBM Math group, Don Coppersmith). Inspired by the Approximated QFT, we have designed algorithms for two scenarios: the Fourier transform calculation of quantum states with amplitudes that have narrow band or k-spares frequencies. Both approaches make use of knowledge of the structure of the frequencies to save significantly on the depth of the circuit. Future work involves a careful integration of Cleve et al.'s work and implementation on real devices.

5.1 Recommendations

It was recently realized that quantum algorithms can be elegantly articulated as various forms of Quantum Singular Value Transformation (QSVT). This insightful generalization is important as it enables one to connect the dots between seemingly distinct algorithms.

In order to be convinced of the attainment of the promised value of quantum computing it is necessary to explicitly demonstrate an advantage over classical algorithms, including any special conditions and provisos. While originally the task may have seemed straightforward, recent studies in the field of quantum-inspired classical algorithms have proven that under similar assumptions, several classes of quantum algorithms (namely low-rank QSVT) can be matched, sometimes approximately, by classical algorithms. It is therefore critical to realize which algorithms are affected by this insight and focus upon other classes (e.g. sparse quantum singular transformation based). We therefore recommend concentrating effort on further understanding the quantum-inspired landscape, so as to not only help delineate which quantum primitives truly offer quantum advantage and which do not, but also help the research community to design new effective quantum algorithms.

In tandem with the above, bringing distinct and complementary perspectives from AI, theory, and decision making under uncertainty, is essential for further development of the field. This cannot be done in isolation, indeed, a deep understanding of hardware limitations and an exchange with experimentalists is crucially important. Such an emphasis on “full-stack” collaboration is accelerated and facilitated by efficient implementations of (publicly known) algorithms that build upon and extend open source frameworks (including QISKit) .

To devise new quantum algorithms that operate robustly across multiple hardware technologies, we believe that exploration of the following topics is essential:

1. Quantum Geometrical Algorithms
 - a. Conceive NISQ-compliant algorithms for QTDA by incorporating variational Quantum Walk as a replacement of the Grover projector
 - b. Substitute QPE by novel algorithms for quantum rank estimator
 - c. Replace non-quantum advantageous portions by classical quantum-inspired components so as to reduce circuit depth.
2. Quantum Statistical Algorithms
 - a. Gaussian Processes Regression by employing stochastic Chebyshev moments
 - b. Combine Bayesian Iterated Quantum Phase Estimation without Eigenvector Preparation and Amplitude Estimation.
3. Quantum Graph Algorithms
 - a. Investigate graph / network algorithms with various structures and various forms for the nodes and/or edges together with associated Markov processes
 - b. Establish theoretical results for limits of such graphs / networks and Markov processes as the number of nodes / edges grow large.

Such novel NISQ-aware algorithmic research is crucial in the journey towards realization of the true potential of quantum computing, both in order to avert a “quantum”-winter by paving a self-sustaining path of research and development and also in terms of discovering new “killer” quantum applications.

6.0 REFERENCES

- [1] J. R. J. R. R. B. a. A. A.-G. McClean, "The theory of variational hybrid quantum-classical algorithms," vol. 18.2, no. 023023, 2016.
- [2] A. Y. Kitaev, A. Shen and M. N. Vyalyi, *Classical and Quantum Computation*, American Mathematical Society, 2002.
- [3] L. Horesh and G. Nannicini, "Meta-Level Short-Depth Quantum Computation of K-Eigenpairs". US Patent P201707431US01, 2018.
- [4] L. Horesh and G. Nannicini, "Short Circuit Depth Variational Quantum Computation of Monte Carlo Minimization and Nth Order Moments". US Patent P201707431AUS01, 2018.
- [5] J. S. Leeuwaarden, M. S. Squillante and E. M. Winands, "Quasi-birth-and-death processes, lattice path counting, and hypergeometric functions," *Journal of Applied Probability*, vol. 46, no. 2, p. 507–520, 2009.
- [6] D. Aharonov, A. Ambainis, J. Kempe and U. Vazirani, "Quantum walks on graphs," in *In Proceedings of the thirty-third annual ACM symposium on Theory of computing*, 2001.
- [7] A. M. C. R. D. E. F. E. G. S. a. S. D. A. Childs, "Exponential algorithmic speedup by a quantum walk," in *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, 2003.
- [8] M. Szegedy, "Quantum speedup of Markov chain based algorithms," in *Proceedings of the forty-fifth annual IEEE symposium on Foundations of computer science*, 2004.
- [9] A. K. J. a. R. A. Ambainis, "Coins make quantum walks faster," in *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, 2005.
- [10] D. A. Bini, G. Latouche and B. Meini, *Numerical Methods for Structured Markov Chains*, Oxford University Press, 2005.
- [11] A. W. „ H. A. a. L. S. Harrow, "Quantum algorithm for linear systems of equations," *Physical review letters*, vol. 103, no. 15, p. 150502, 2009.
- [12] M. H. P. M. S. S. N. U. a. L. W. D. Dervovic, "Quantum linear systems algorithms: a primer," arXiv preprint arXiv:1802.08227, 2018.

- [13] A. Prakash, "Quantum algorithms for linear algebra and machine learning," PhD thesis, UC Berkeley, 2004.
- [14] A. Childs, R. Kothari and R. Somma, "Quantum algorithm for systems of linear equations with exponentially improved dependence on precision," *SIAM Journal on Computing*, 2017.
- [15] A. J. M. P. S. M.-H. Y. X.-Q. Z. P. J. L. A. A.-G. a. J. L. O. Peruzzo, "A Variational Eigenvalue Solver on a Photonic Quantum Processor.," *Nature Communications* 5: 4213., 2014.
- [16] I. Kerenidis and A. Prakash, "Quantum recommendation systems," 2016. [Online]. Available: <https://arxiv.org/abs/1603.08675>.
- [17] S. Lloyd, S. Garnerone and P. Zanardi, "Quantum algorithms for topological and geometric analysis of big data," 2014.
- [18] A. W. A. H. S. L. Harrow, "Quantum algorithm for linear systems of equations," vol. 103, no. 15 , 2009.
- [19] S. Woerner and D. J. Egger, "Quantum Risk Analysis," <http://arxiv.org/abs/1806.06893>, 2018.
- [20] G. Brassard, P. Hoyer, M. Mosca and A. Tapp, "Quantum Amplitude Amplification and Estimation," *arXiv:quant-ph/0005055*, 2000.
- [21] H. Stanley, "Scaling, universality, and renormalization: Three pillars of modern critical phenomena," *Reviews of modern physics*, vol. 71(2), p. 358, 1999.
- [22] R. Cleve and J. Watrous, "Fast parallel circuits for the quantum," <https://arxiv.org/abs/quant-ph/0006004>, 2000.
- [23] A. Barenco, K. S. A Ekert and P. Törmä, "Approximate quantum Fourier transform and decoherence," *APS Physical Review A*, 1996.
- [24] L. Cincio, Y. Subaşı and A. T. C. P. J. Sornborger, "Learning the quantum algorithm for state overlap," *arXiv*, p. 1803.04114, 2018.
- [25] Y. K. Kwok, K. S. Leung and H. Y. Wong, "Efficient Options Pricing Using the Fast Fourier Transform," in *Handbook of Computational Finance*, J. e. a. Duan, Ed., Springer Handbooks of Computational Statistics, 2012.
- [26] G. Nannicini, "Performance of hybrid quantum/classical variational heuristics for combinatorial optimization," in <https://arxiv.org/abs/1805.12037>, 2018.

- [27] S. Lloyd, M. Mohseni and P. Rebentrost, "Quantum principal component analysis," *Nature Physics*, vol. 10, p. 631, 2014.
- [28] G. H. Low and I. L. Chuang, 2017. [Online]. Available: <https://arxiv.org/abs/1707.05391>.
- [29] A. Rudi, L. Wossnig, C. Ciliberto, A. Rocchetto, M. Pontil and S. S., "Approximating Hamiltonian dynamics with the Nystrom method," *arXiv*, p. 1804.02484, 2018.
- [30] E. Tang, "A quantum-inspired classical algorithm for recommendation systems," *arxiv*, p. 1807.04271, 2018.
- [31] A. Gilyén, S. Lloyd and E. Tang, "Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension.," 12 November 2018. [Online]. Available: <https://arxiv.org/abs/1811.04909>. [Accessed 2019].
- [32] D. P. Woodruff, Sketching as a tool for numerical linear algebra, Foundations and Trends in Theoretical Computer Science, 2014.
- [33] K. L. Clarkson and D. P. Woodruff, "Low rank approximation and regression in input sparsity time.," in *acm Symposium on Theory of Computing*, ACM, 2013.
- [34] A. Ambainis, "Quantum walks and their algorithmic applications," *International Journal of Quantum Information*, vol. 1, no. 4, pp. 507--518, 2003.
- [35] J. S. Leeuwaarden, M. S. Squillante and E. M. Winands, "Quasi-birth-and-death processes, lattice path counting, and hypergeometric functions," *Journal of Applied Probability*, p. 46(2):507–520, 2009.
- [36] Y. B. Y. a. H. G. LeCun, "Deep learning," *nature*, vol. 521(7553), 2015.
- [37] M. a. C. I. Nielsen, Quantum information and quantum computation, Cambridge: Cambridge University : 2000.
- [38] N. a. G. Wiebe, "Efficient Bayesian phase estimation," *Physical review letters*, Vols. 117(1), p.010503, 2016.
- [39] J. T. H. a. R. T. Friedman, The elements of statistical learning, Springer series in statistics, 2001.
- [40] M. S. I. a. P. Schuld, An introduction to quantum machine learning, Contemporary Physics, 2015.

- [41] N. K. A. a. S. K. Wiebe, "Quantum deep learning," *arXiv preprint arXiv:1412.3489*, 2014.

ABBREVIATIONS AND ACRONYMS

AFRL	Air Force Research Laboratory
AI	Artificial Intelligence
AQFT	Approximate Quantum Fourier Transform
BQP	Bounded-Error Quantum Polynomial Time
CMB	Cosmic Microwave Background
CO	Combinatorial Optimization
CVaR	Conditional Value at Risk
IBM	International Business Machine
JL	Johnson Lindenstrauss Lemma
MNIST	Modified National Institute of Standards and Technology
NISQ	Noisy Intermediate-Scale Quantum (NISQ)
NNZ	Number of Non-Zeros
NP	Non-Deterministic Polynomial Acceptable Problems
P	Polynomial Time
QAOA	Quantum Approximate Optimization Algorithm
QFT	Quantum Fourier Transform
QISKit	Quantum Information Software Kit
QLSP	Quantum Linear System Problem
QPE	Quantum Phase Estimation
QRAM	Quantum Random Access Memory
QSVT	Quantum Singular Value Transformation
QTDA	Quantum Topological Data Analysis
RAM	Random Access Memory
SPD	Symmetric Positive-Definite
SQFT	Sparse Quantum Fourier Transform
TDA	Topological Data Analysis
VaR	Value at Risk (VaR)
VQE	Variational Quantum EigenSolver
VQkE	Variational Quantum k-EigenSolver