US 20240144066A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2024/0144066 A1**

Dalzell et al. (43) **Pub. Date: May 2, 2024**

(54) **QUANTUM INTERIOR POINT METHOD**

(71) Applicant: **Goldman Sachs & Co. LLC**, New York, NY (US)

(72) Inventors: **Alexander M. Dalzell**, San Francisco, CA (US); **B. David Clader**, Ellicott City, MD (US); **Grant Salton**, San Jose, CA (US); **Mario Berta**, Los Angeles, CA (US); **Cedrick Yen-Yu Lin**, Bellevue, WA (US); **David A. Bader**, New York, NY (US); **William J. Zeng**, New York, NY (US)

(21) Appl. No.: **18/376,532**

(22) Filed: **Oct. 4, 2023**

Related U.S. Application Data

(60) Provisional application No. 63/413,230, filed on Oct. 4, 2022.

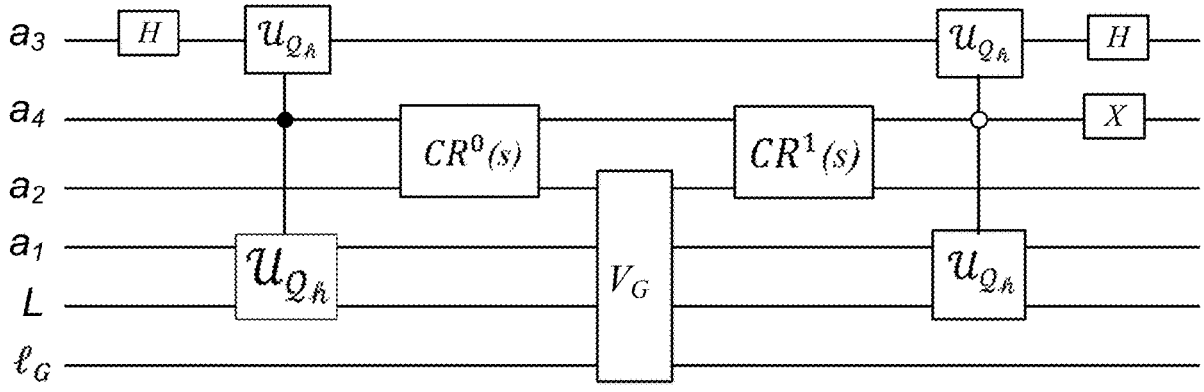
Publication Classification

(51) **Int. Cl.**
G06N 10/20 (2006.01)
G06N 10/60 (2006.01)

(52) **U.S. Cl.**
CPC ***G06N 10/20*** (2022.01); ***G06N 10/60*** (2022.01)

(57) **ABSTRACT**

In some aspects, the techniques described herein relate to a quantum method for solving a second-order cone program (SOCP) instance, the method including: defining a Newton system for the SOCP instance by constructing matrix G and vector h based on the SOCP instance; preconditioning matrix G and vector h via row normalization to reduce a condition number of matrix G; iteratively determining u until a predetermined iteration condition is met, the iterations including: causing a quantum computing system to apply matrix G and vector h to a quantum linear system solver (QLSS) to generate a quantum state; causing the quantum computing system to perform quantum state tomography on the quantum state; and updating a value of u based on a current value of u and the output of the quantum state tomography; and determining a solution to the SOCP instance based on the updated value of u.



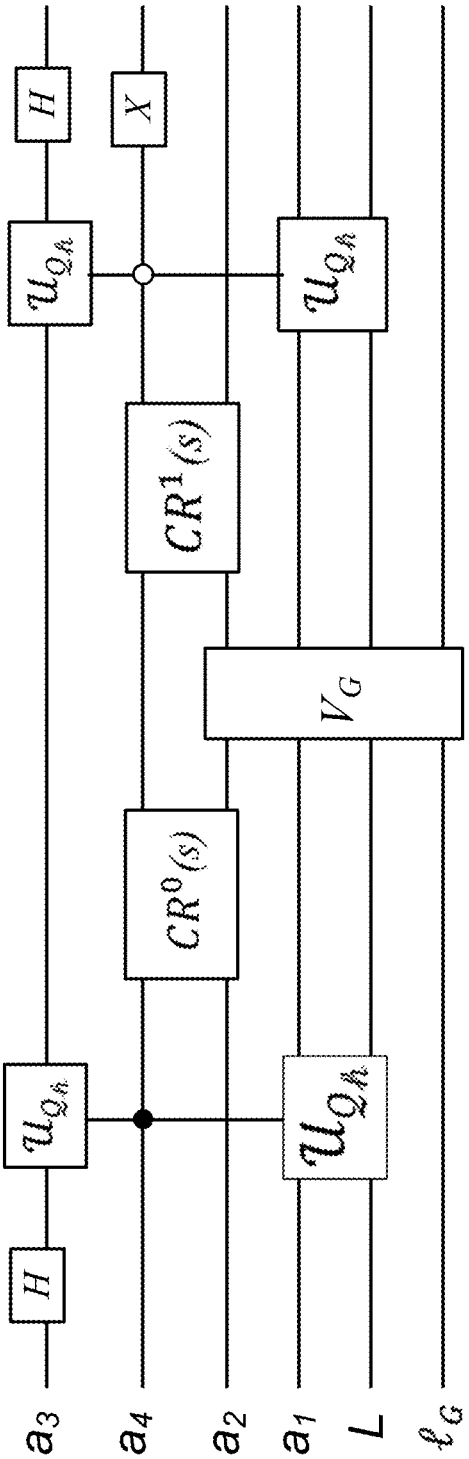

H
X
$\mathcal{U}_{Q_k}$
$\mathcal{U}_{Q_k}$
$CR^1(s)$
V_G
$CR^0(s)$
$\mathcal{U}_{Q_k}$
$\mathcal{U}_{Q_k}$
H
a_3
a_4
a_2
a_1
L
ℓ_G

FIG. 1

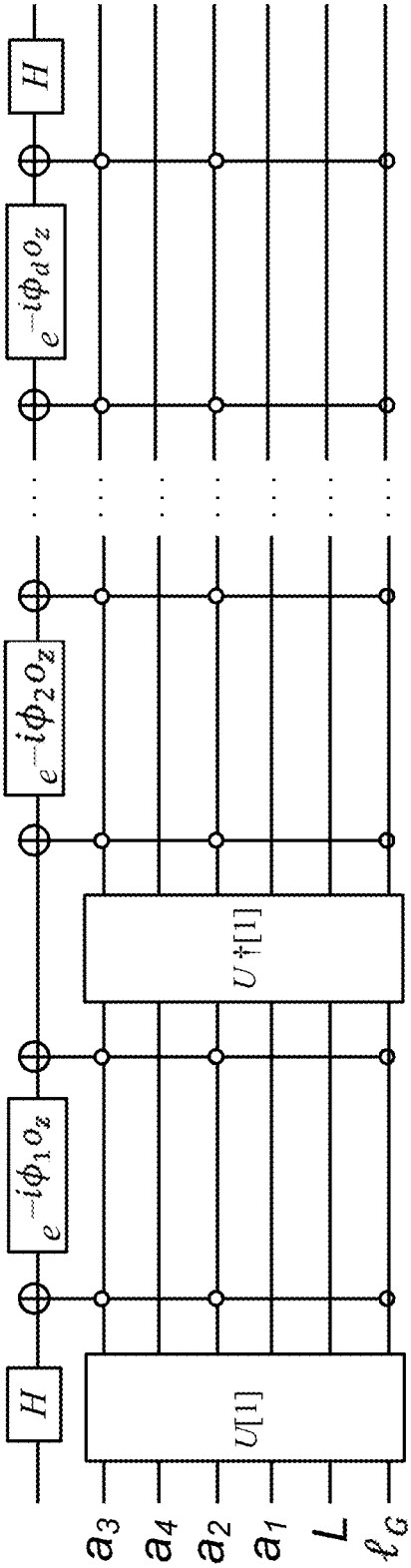


FIG. 2

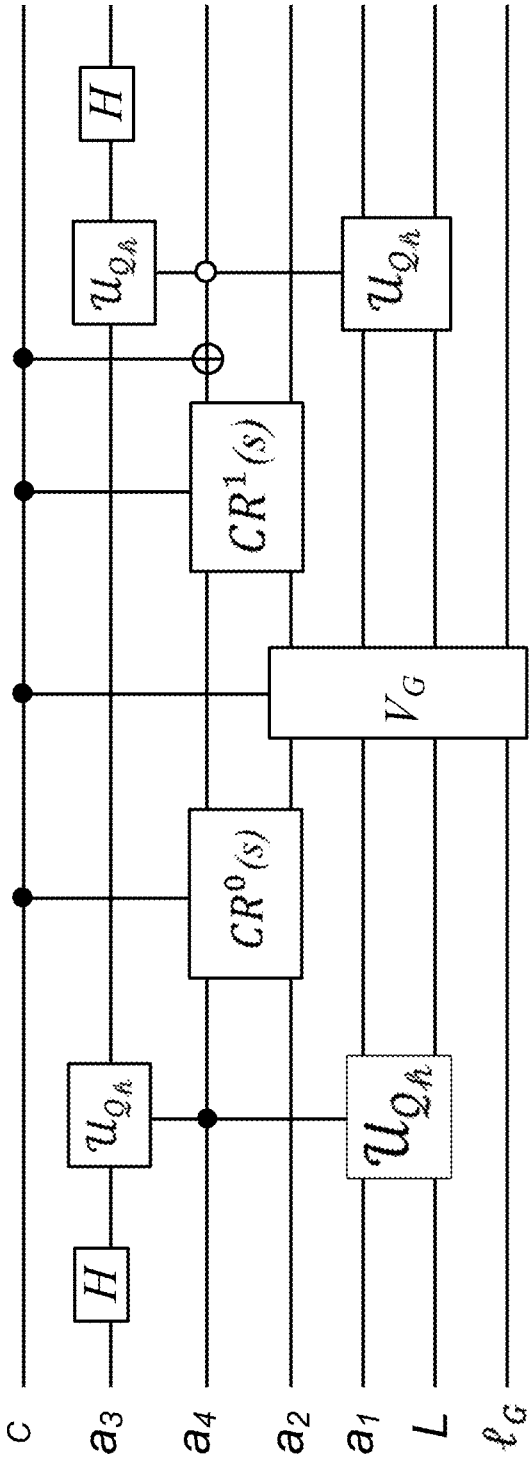


FIG. 3

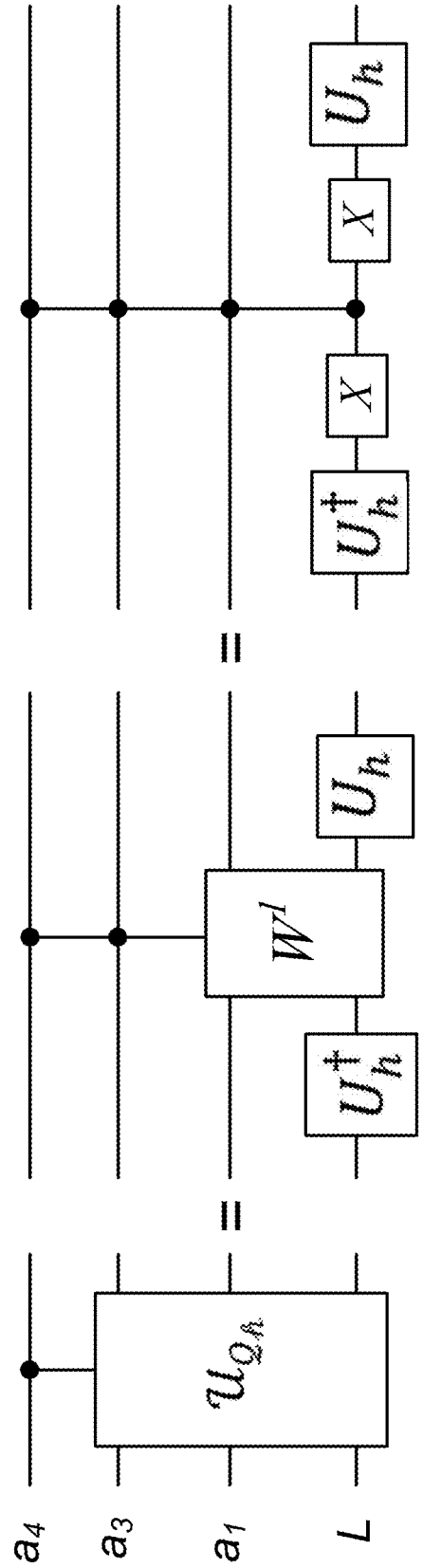


FIG. 4

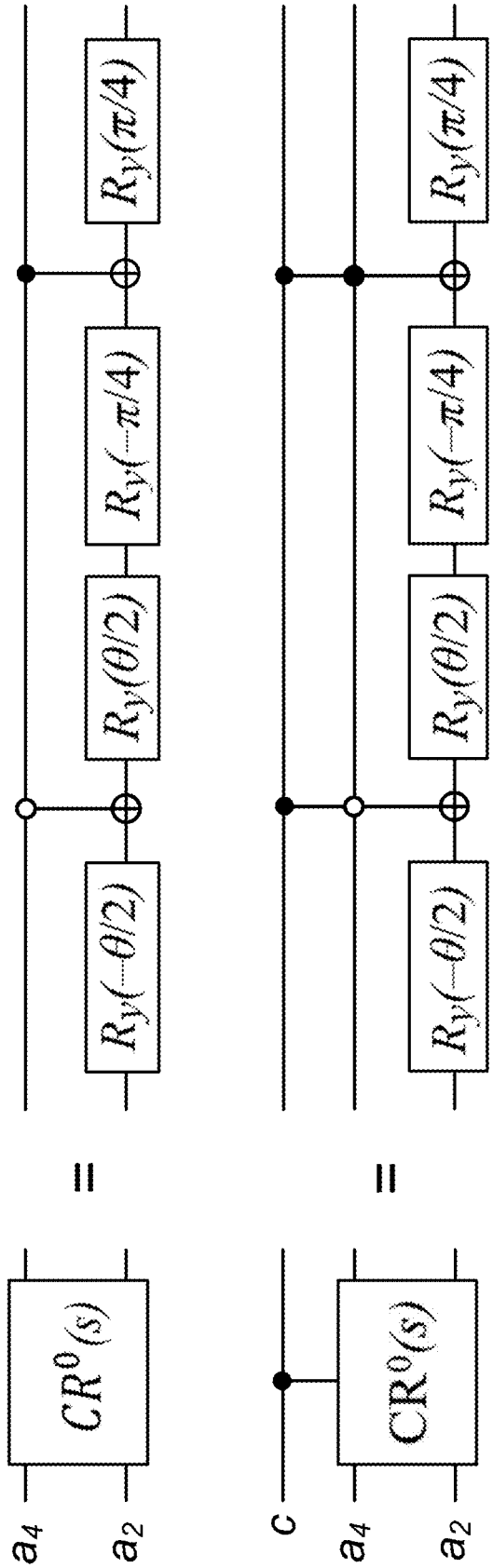

a_4
a_2
$CR^0(s)$
=
$Ry(-\theta/2)$
$Ry(\theta/2)$
$Ry(-\pi/4)$
$Ry(\pi/4)$
c
a_4
a_2
$CR^0(s)$
=
$Ry(-\theta/2)$
$Ry(\theta/2)$
$Ry(-\pi/4)$
$Ry(\pi/4)$

FIG. 5

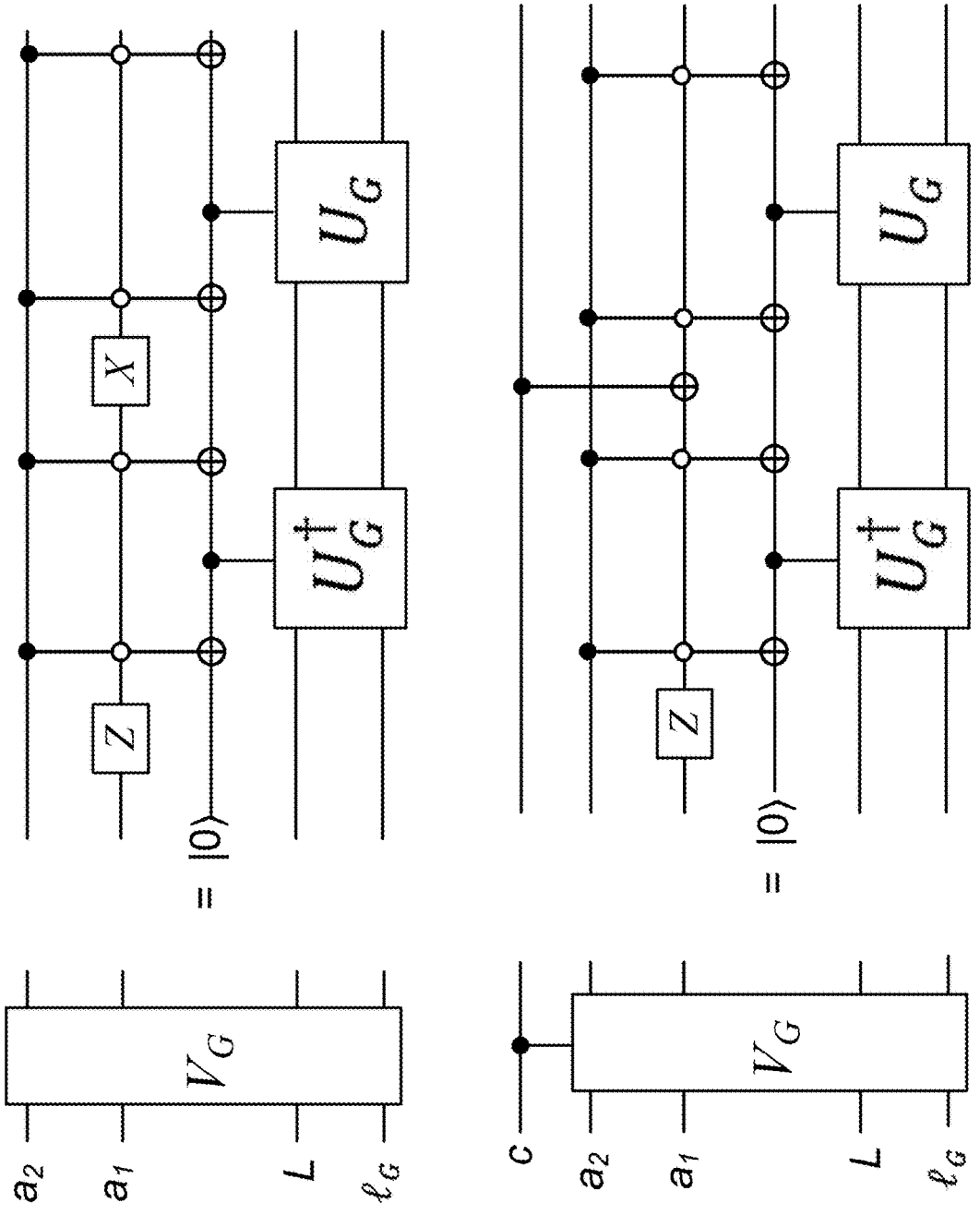

a_2
a_1
V_G
L
ℓ_G
$|0\rangle$
Z
X
U_G
$U_G^\dagger$
=
c
a_2
a_1
V_G
L
ℓ_G
$|0\rangle$
Z
U_G
$U_G^\dagger$
=

FIG. 6

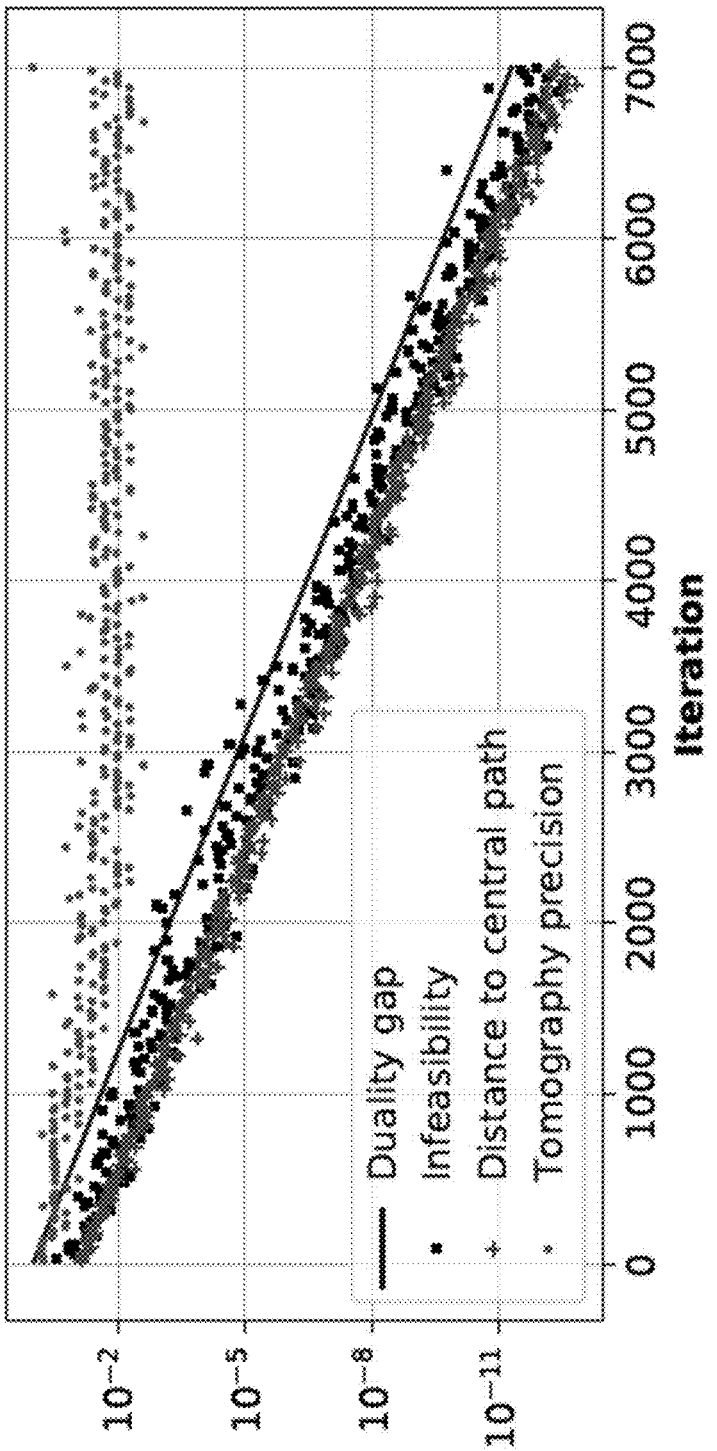

10^{-2}
10^{-5}
10^{-8}
10^{-11}
0
1000
2000
3000
4000
5000
6000
7000
Iteration
Duality gap
Infeasibility
Distance to central path
Tomography precision

FIG. 7

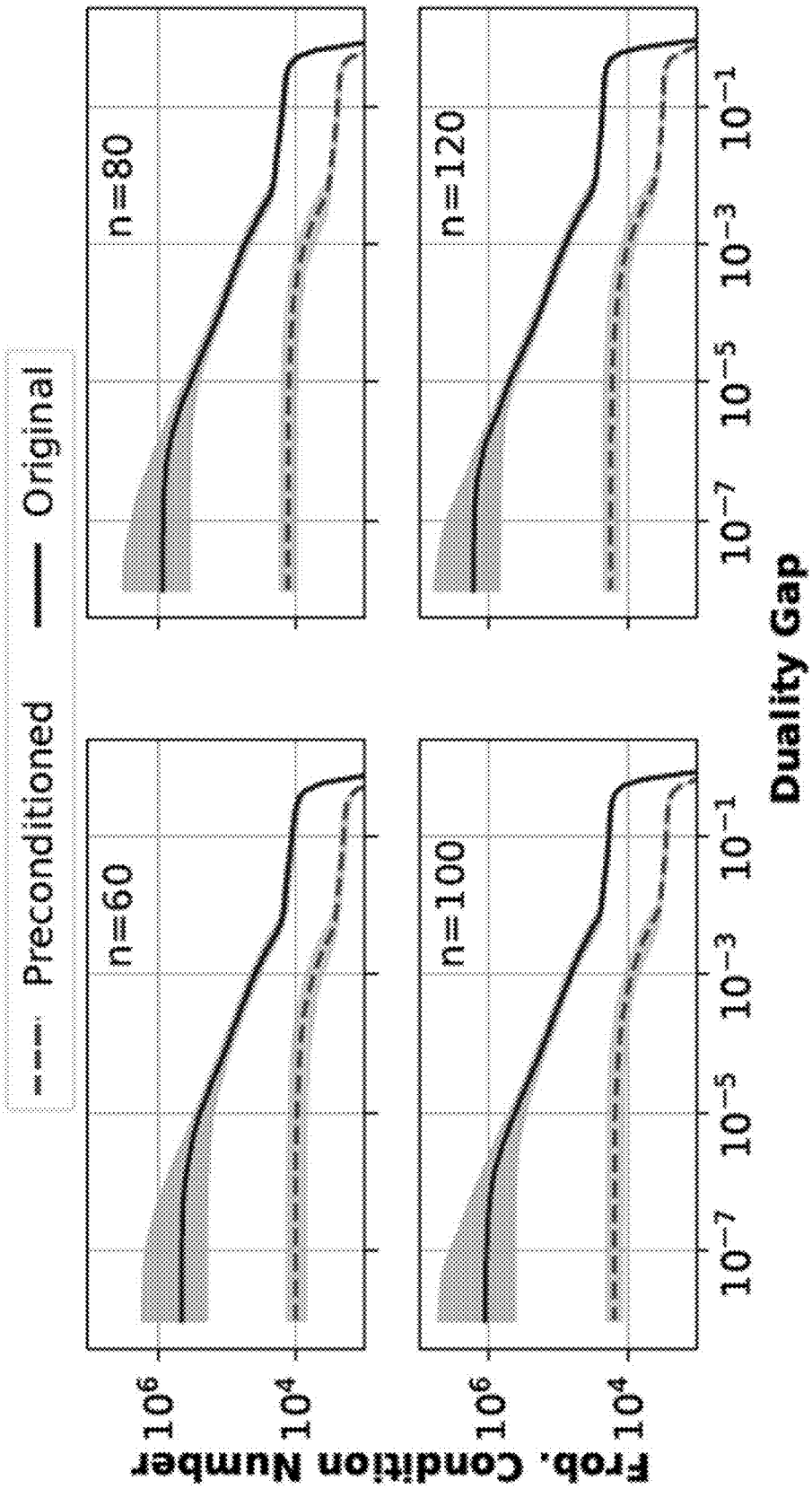

Preconditioned
Original
n=60
n=80
n=100
n=120
Frob. Condition Number
10^6
10^4
10^{-7}
10^{-5}
10^{-3}
10^{-1}
Duality Gap

FIG. 8

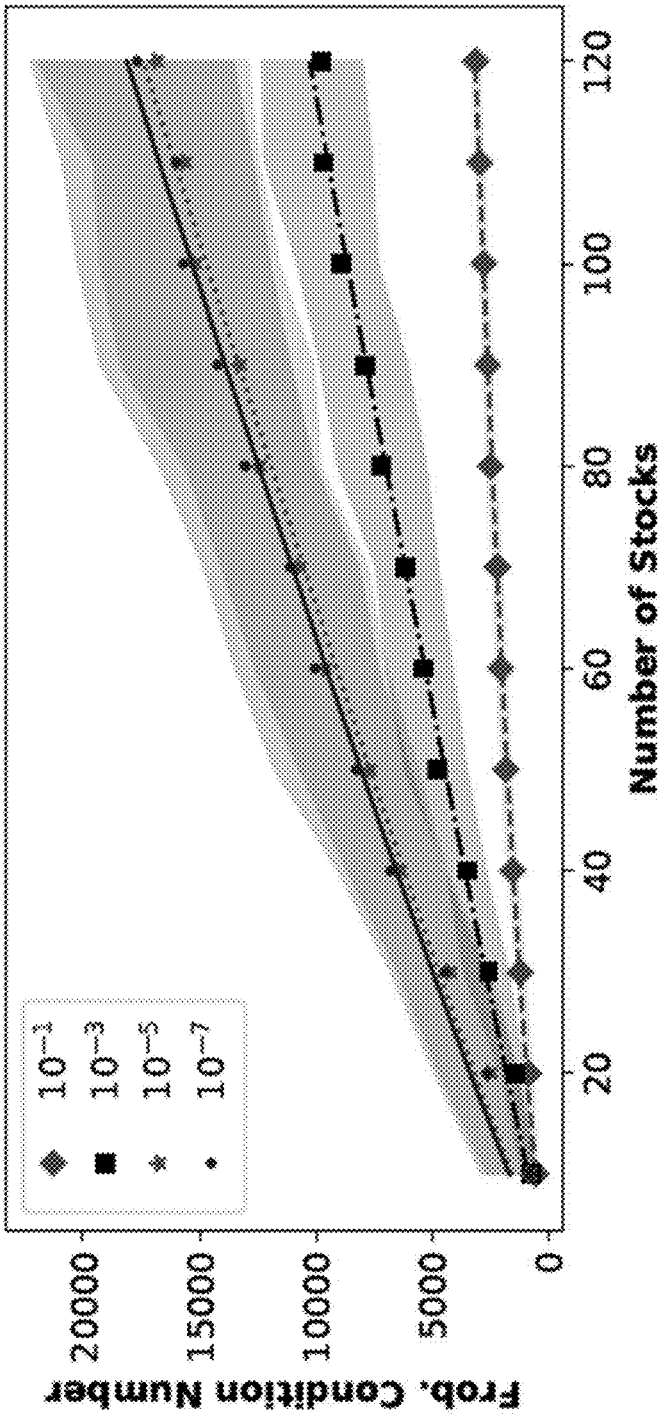

Prob. Condition Number
20000
15000
10000
5000
0
20
40
60
80
100
120
Number of Stocks
10^{-1}
10^{-3}
10^{-5}
10^{-7}

FIG. 9

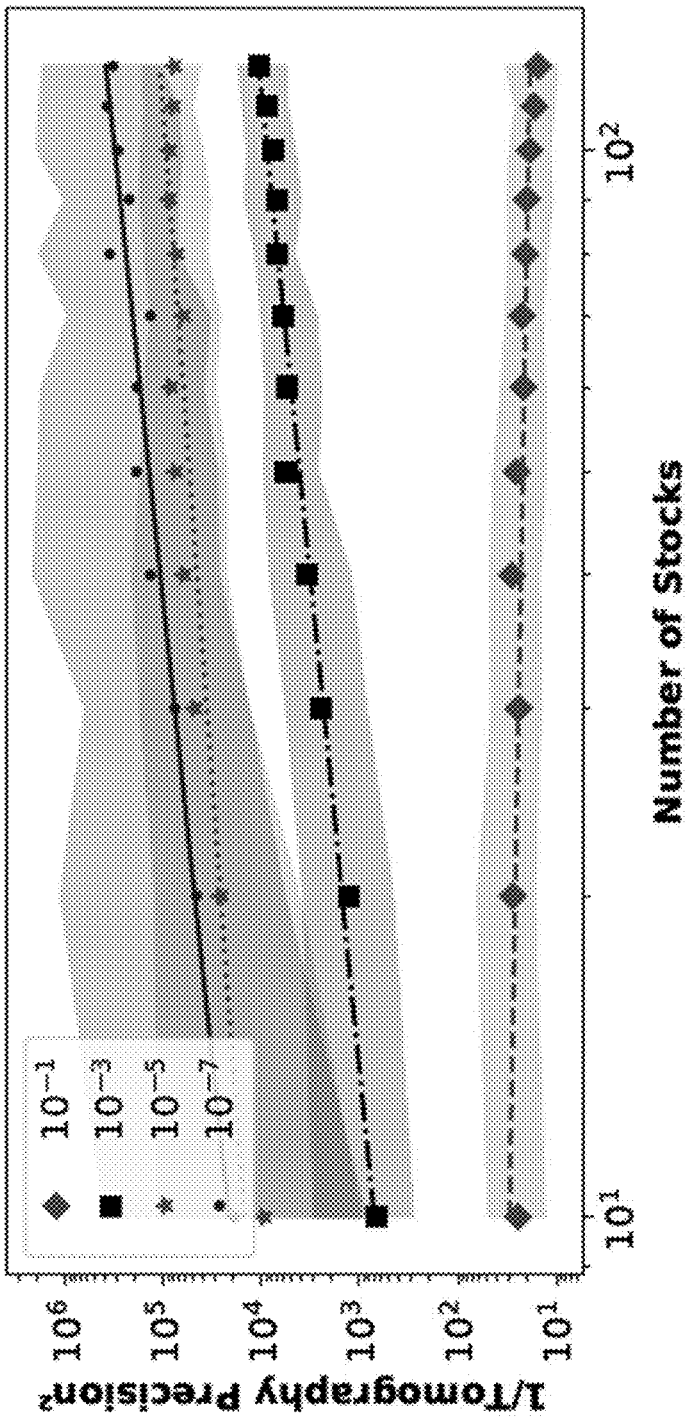

1/Tomography Precision²
10⁶
10⁵
10⁴
10³
10²
10¹
10¹
10²
Number of Stocks
10⁻¹
10⁻³
10⁻⁵
10⁻⁷

FIG. 10

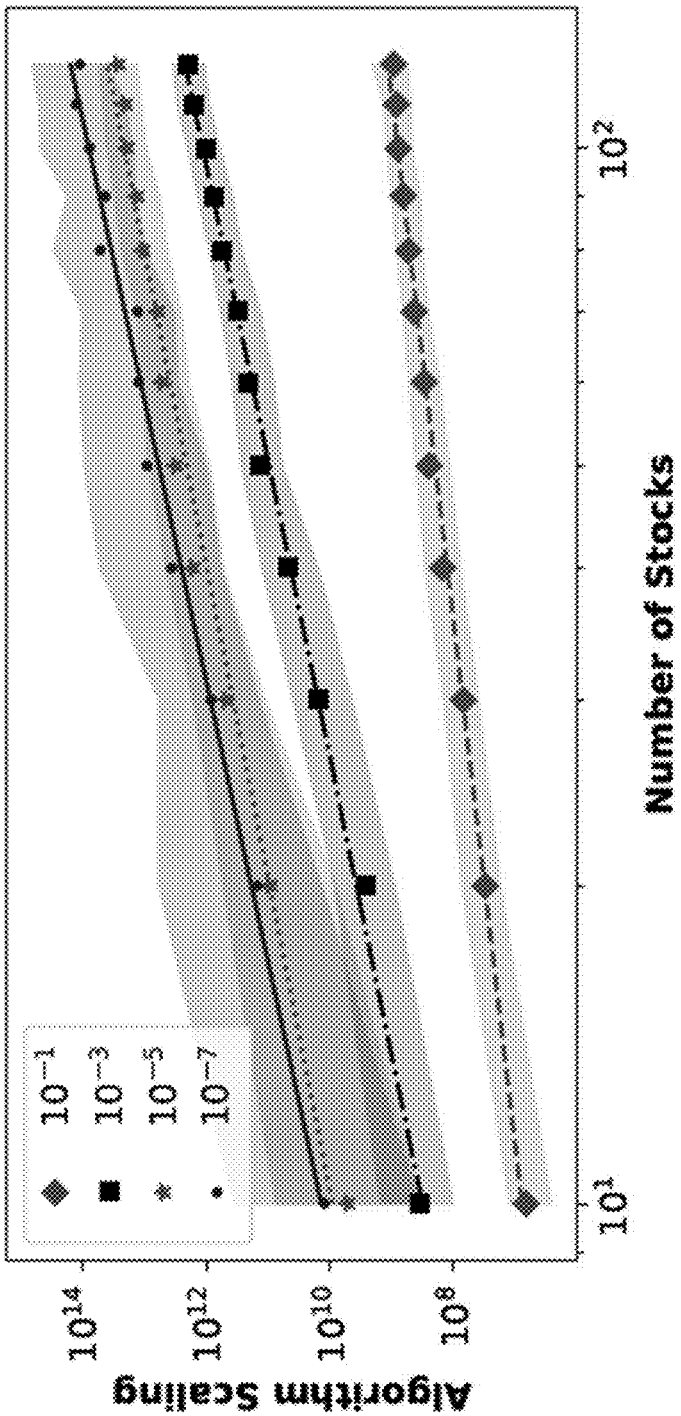

Algorithm Scaling
10^{14}
10^{12}
10^{10}
10^{8}
10^{1}
10^{2}
Number of Stocks
10^{-1}
10^{-3}
10^{-5}
10^{-7}

FIG. 11

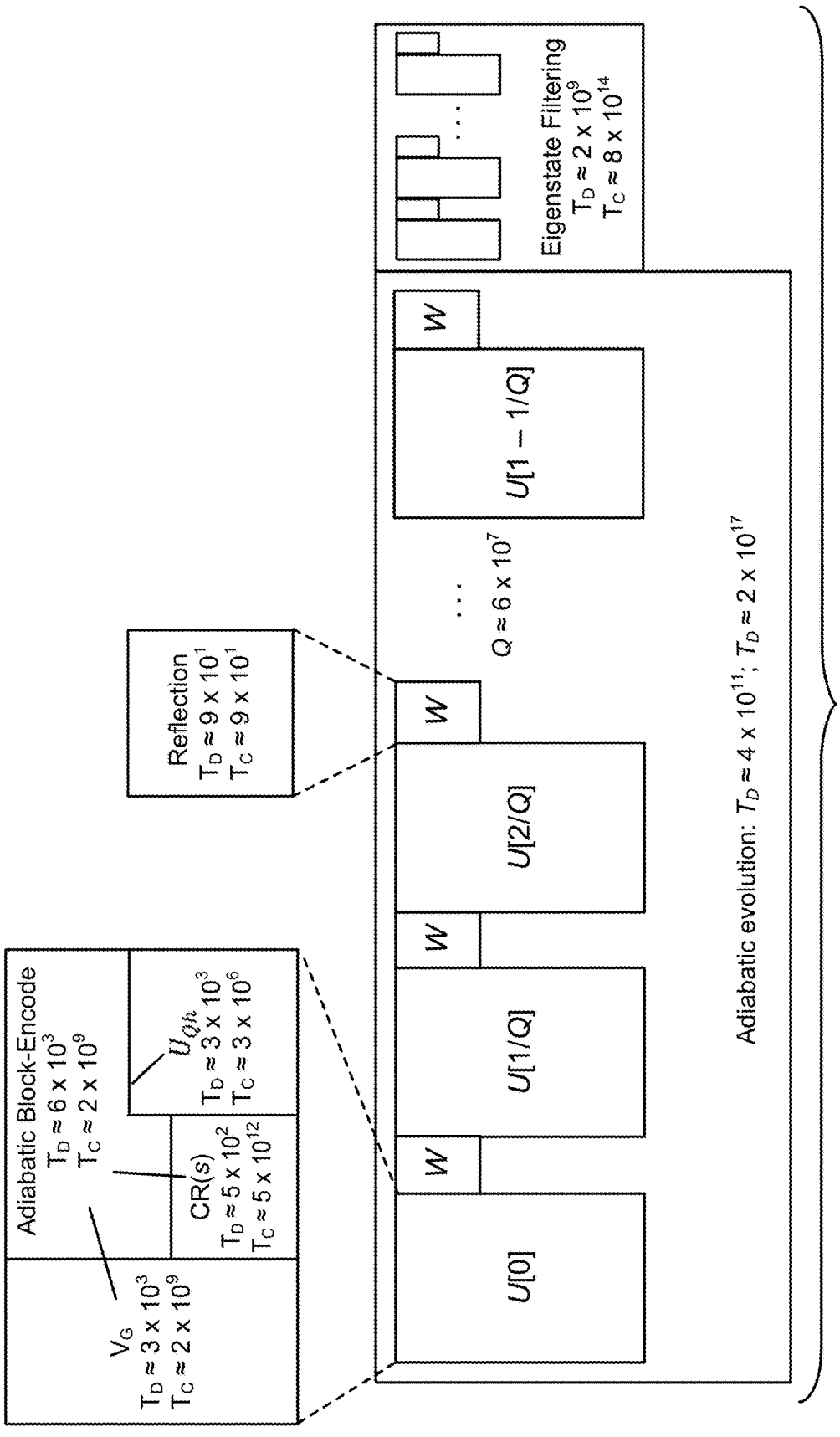

Adiabatic Block-Encode
$T_D \approx 6 \times 10^3$
$T_C \approx 2 \times 10^9$
V_G
$T_D \approx 3 \times 10^3$
$T_C \approx 2 \times 10^9$
CR(s)
$T_D \approx 5 \times 10^2$
$T_C \approx 5 \times 10^{12}$
U_{gh}
$T_D \approx 3 \times 10^3$
$T_C \approx 3 \times 10^6$
Reflection
$T_D \approx 9 \times 10^1$
$T_C \approx 9 \times 10^1$
W
U[0]
U[1/Q]
U[2/Q]
$Q \approx 6 \times 10^7$
U[1 – 1/Q]
Eigenstate Filtering
$T_D \approx 2 \times 10^9$
$T_C \approx 8 \times 10^{14}$
Adiabatic evolution: $T_D \approx 4 \times 10^{11}$; $T_D \approx 2 \times 10^{17}$
$k \times N_{it} \approx (3 \times 10^8) \times (8 \times 10^3) \approx 3 \times 10^{12}$ classical repititions
+ same number of repetitions for controlled version

FIG. 12

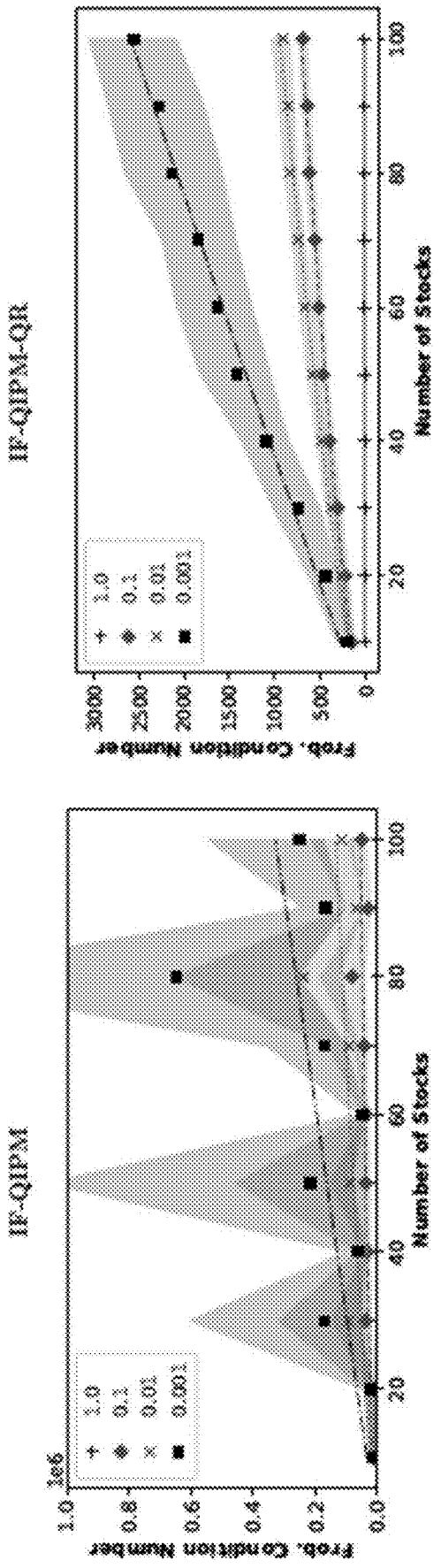

IF-QIPM-QR
Frob. Condition Number
3000
2500
2000
1500
1000
500
0
20
40
60
80
100
Number of Stocks
1.0
0.1
0.01
0.001
IF-QIPM
1e6
Frob. Condition Number
1.0
0.8
0.6
0.4
0.2
0.0
20
40
60
80
100
Number of Stocks
1.0
0.1
0.01
0.001

FIG. 13

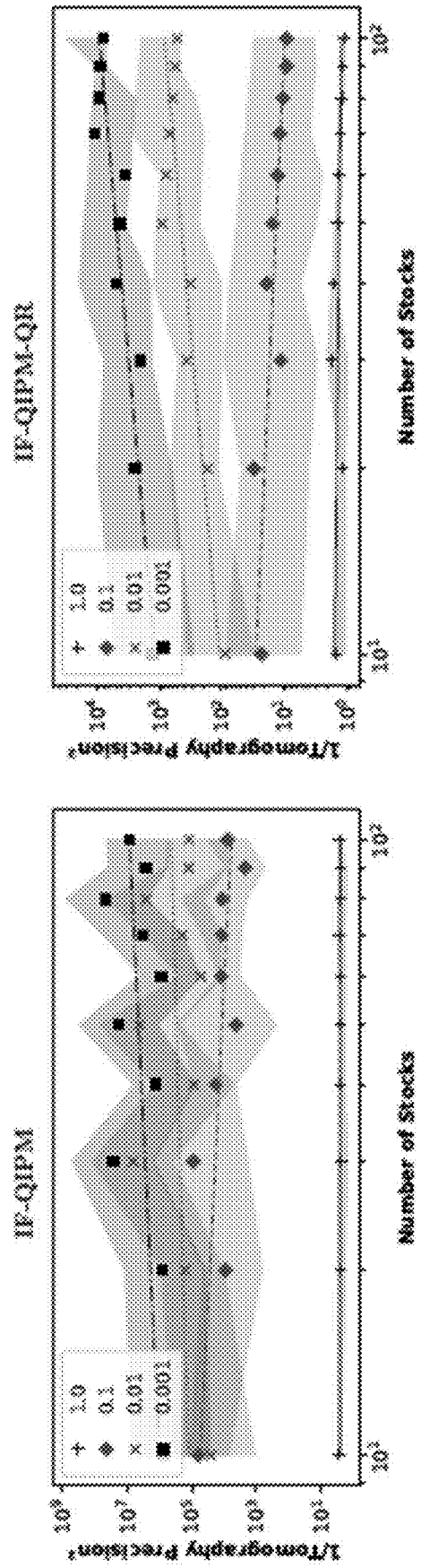

IF-QIPM
IF-QIPM-QR
1/Tomography Precision²
Number of Stocks
1.0
0.1
0.01
0.001

FIG. 14

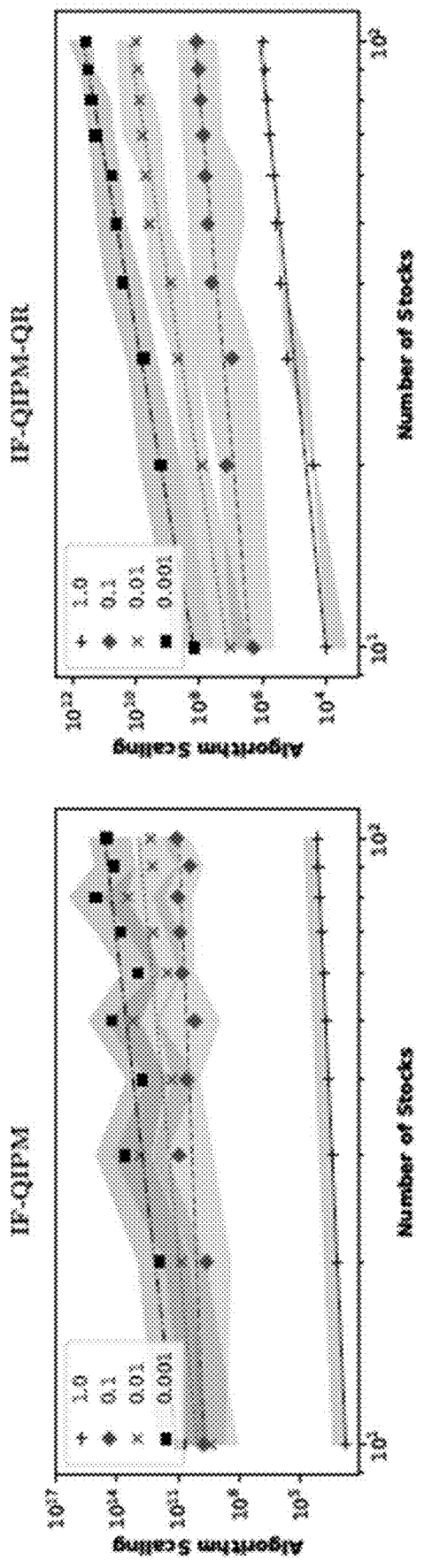

IF-QIPM-QR
IF-QIPM
Algorithm Scaling
Number of Stocks
1.0
0.1
0.01
0.001

FIG. 15

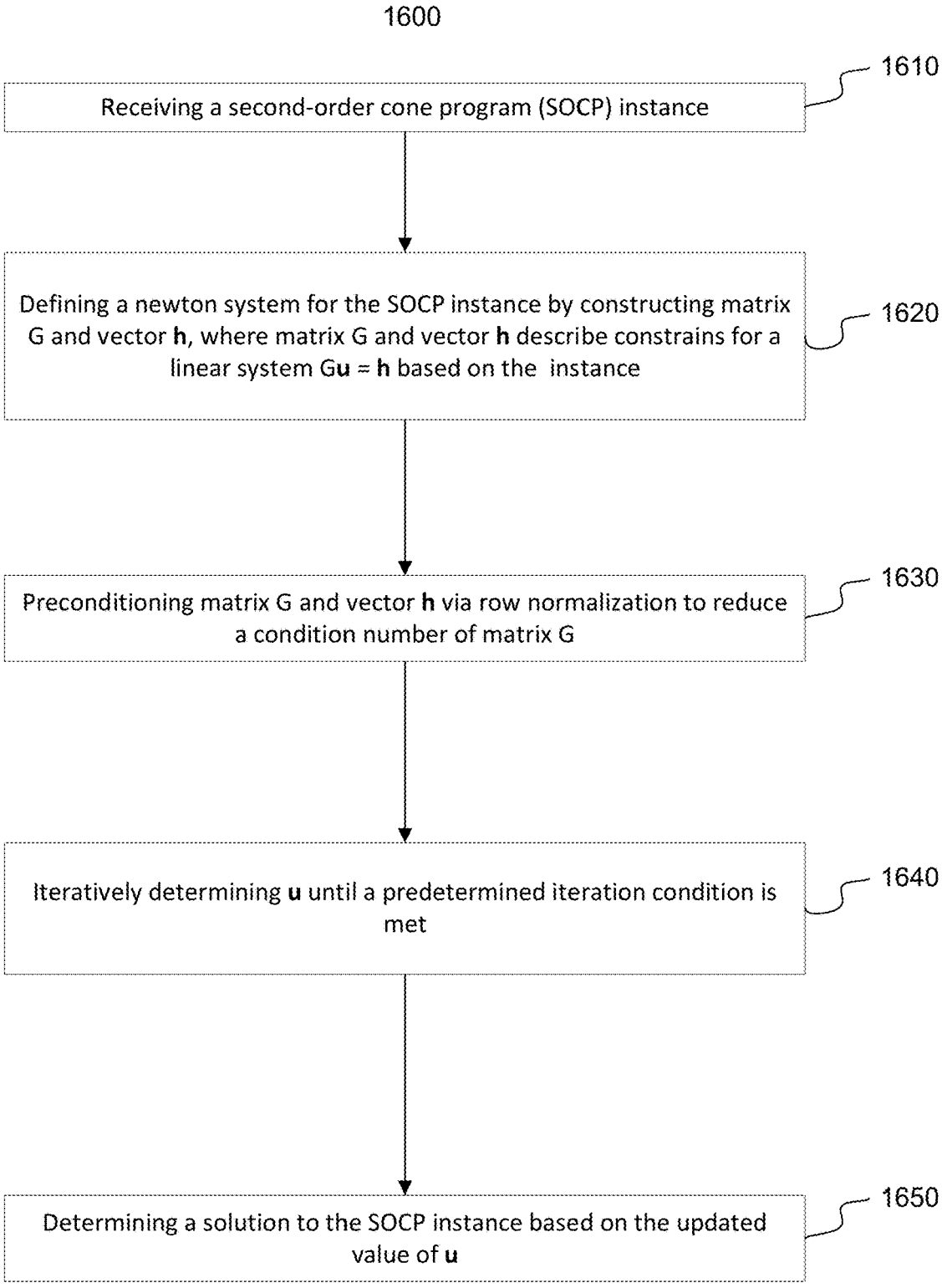

1600
Receiving a second-order cone program (SOCP) instance
1610
Defining a newton system for the SOCP instance by constructing matrix G and vector **h**, where matrix G and vector **h** describe constrains for a linear system G**u** = **h** based on the instance
1620
Preconditioning matrix G and vector **h** via row normalization to reduce a condition number of matrix G
1630
Iteratively determining **u** until a predetermined iteration condition is met
1640
Determining a solution to the SOCP instance based on the updated value of **u**
1650

FIG. 16

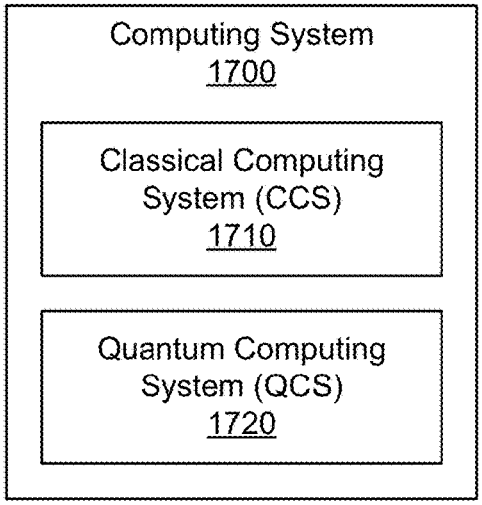


FIG. 17A

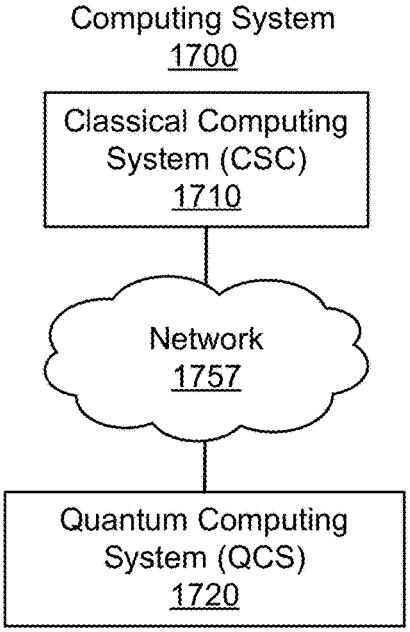


FIG. 17B

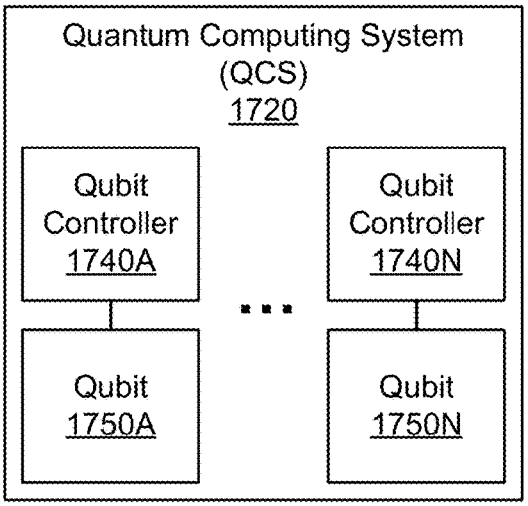


FIG. 17C

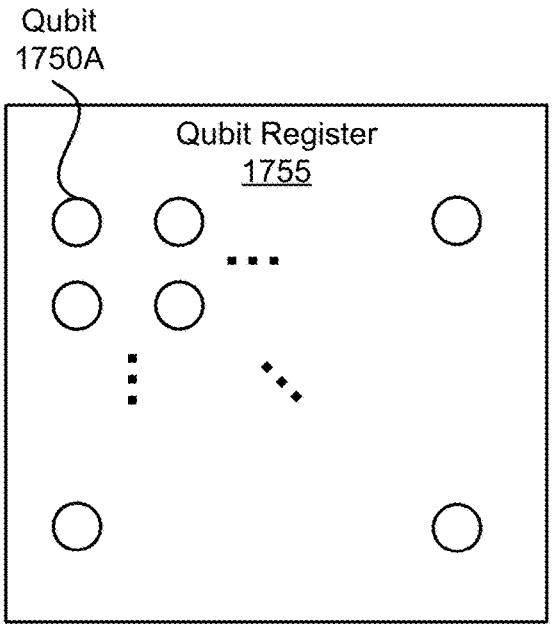


FIG. 17D

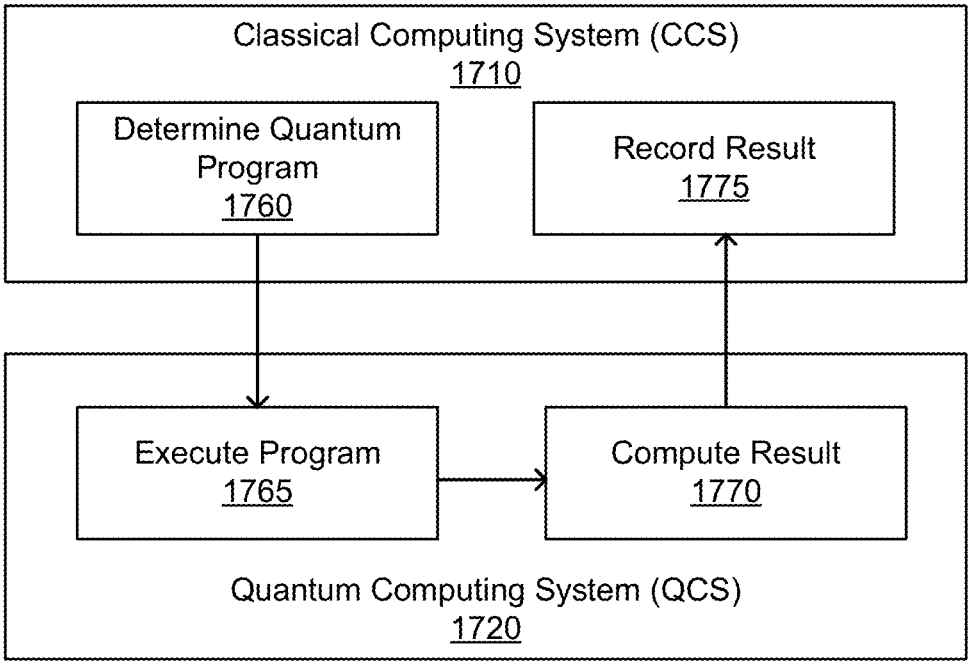

Classical Computing System (CCS)
1710
Determine Quantum Program
1760
Record Result
1775
Execute Program
1765
Compute Result
1770
Quantum Computing System (QCS)
1720

FIG. 17E

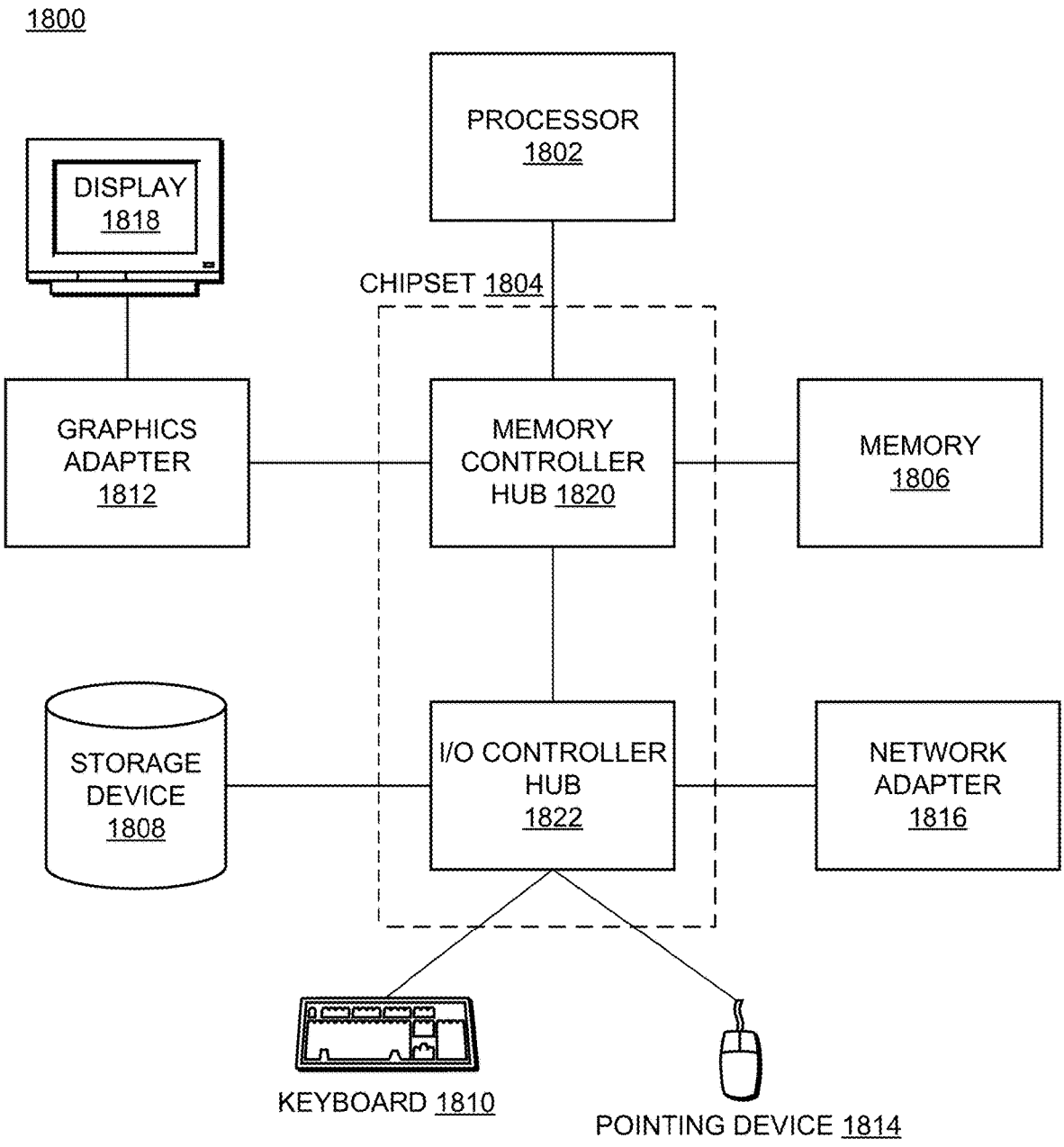

1800
PROCESSOR
1802
DISPLAY
1818
CHIPSET 1804
GRAPHICS
ADAPTER
1812
MEMORY
CONTROLLER
HUB 1820
MEMORY
1806
STORAGE
DEVICE
1808
I/O CONTROLLER
HUB
1822
NETWORK
ADAPTER
1816
KEYBOARD 1810
POINTING DEVICE 1814

FIG. 18

QUANTUM INTERIOR POINT METHOD

CROSS-REFERENCE TO RELATED APPLICATION

[0001] This application claims priority to U.S. Provisional Patent Application Ser. No. 63/413,230, "End-to End Analysis for Quantum Interior Point Methods with Improved Block-Encodings," filed on Oct. 4, 2022, which is incorporated herein by reference in its entirety.

BACKGROUND

1. Technical Field

[0002] This disclosure relates generally to quantum interior point methods (QIPMs), and more particularly to implementing quantum interior point methods (QIPMs).

2. Description of Related Art

[0003] The practical utility of finding optimal solutions to well-posed optimization problems has been known since the days of antiquity. With the advent of the quantum era, there has been great interest in developing quantum algorithms that solve optimization problems with provable speedups over classical algorithms. Unfortunately, it can be difficult to implement these quantum algorithms and evaluate whether these quantum algorithms will be practically useful.

SUMMARY

[0004] In some aspects, the techniques described herein relate to a quantum interior point method (QIPM) for solving a second-order cone program (SOCP) instance using a quantum computing system, the method including: receiving the SOCP instance; defining a Newton system for the SOCP instance by constructing matrix G and vector h, where matrix G and vector h describe constrains for a linear system Gu=h based on the SOCP instance; preconditioning matrix G and vector h via row normalization to reduce a condition number of matrix G; iteratively determining u until a predetermined iteration condition is met, the iterations including: causing the quantum computing system to apply matrix G and vector h to a quantum linear system solver (QLSS) to generate a quantum state; causing the quantum computing system to perform quantum state tomography on the quantum state; and updating a value of u based on a current value of u and the output of the quantum state tomography; and determining a solution to the SOCP instance based on the updated value of u.

[0005] Other aspects include components, devices, systems, improvements, methods, processes, applications, computer readable mediums, and other technologies related to any of the above.

BRIEF DESCRIPTION OF THE DRAWINGS

[0006] Embodiments of the disclosure have advantages and features which will be more readily apparent from the following detailed description and the appended claims, when taken in conjunction with the examples in the accompanying drawings, in which:

[0007] FIG. **1** is a diagram of an example quantum circuit configured to enact a unitary U[s] on registers on a set of registers, according to some embodiments;

[0008] FIG. **2** is a diagram of an example Quantum singular value transform (QSVT) circuit, according to some embodiments;

[0009] FIG. **3** is a diagram of an example-controlled version of the quantum circuit in FIG. **1**, controlled on qubit c, according to some embodiments;

[0010] FIG. **4** is a diagram illustrating an example decomposition of the U_{Qh} gate into a state-preparation unitary U_h and multi-controlled-Toffoli gates, according to some embodiments;

[0011] FIG. **5** is a diagram illustrating an example decomposition of the $CR^0(s)$ gate (top) and controlled-$CR^0(s)$ gate (bottom), as defined in eq. (55), according to some embodiments;

[0012] FIG. **6** is a diagram illustrating an example decomposition of the V_G unitary (top) and controlled-V_G unitary (bottom), as defined in eq. (57), according to some embodiments;

[0013] FIG. **7** is a plot illustrating simulation results of the QIPM on an SOCP instance corresponding to portfolio optimization on 30 randomly chosen stocks, according to some embodiments;

[0014] FIG. **8** includes plots of the Median Frobenius condition number for 128 randomly sampled stock portfolios from the DWCF index, according to some embodiments;

[0015] FIG. **9** is a plot of the Median Frobenius condition number κ_F for 128 randomly sampled stock portfolios from the DWCF index, according to some embodiments;

[0016] FIG. **10** is a plot of the median value of the square of the inverse tomography precision used to remain in the neighborhood of the central path for 128 randomly sampled stock portfolios from the DWCF index, according to some embodiments;

[0017] FIG. **11** is a plot of the median value of the estimated algorithm scaling factor, according to some embodiments;

[0018] FIG. **12** is a diagram illustrating the breakdown of quantum resources used for a single coherent run of the uncontrolled version of a quantum algorithm, according to some embodiments;

[0019] FIG. **13** includes two plots of the Median Frobenius condition number for 128 randomly sampled stock portfolios from the DWCF index, according to one or more embodiments;

[0020] FIG. **14** includes two plots of the Median value of the square of the required inverse tomography precision used to remain in the neighborhood of the central path for 128 randomly sampled stock portfolios from the DWCF index, according to one or more embodiments;

[0021] FIG. **15** includes two plots of the Median value of the estimated algorithm scaling factor computed as the median of $n^{1.5}\kappa_F/\xi^2$ for 128 randomly sampled stock portfolios from the DWCF index, according to one or more embodiments;

[0022] FIG. **16** is a flowchart of an example method, specifically a quantum interior point method (QIPM), for solving a second-order cone program (SOCP) instance using a quantum computing system, according to one or more embodiments;

[0023] FIGS. **17**A-**17**B are block diagrams of a computing system including a classical computing system and a quantum computing system, according to some embodiments;

[0024] FIG. **17**C-**17**D are block diagrams of components of a quantum computing system, according to some embodiments;

[0025] FIG. **17**E is a flow chart that illustrates an example execution of a quantum routine on the computing system; and

[0026] FIG. **18** is an example architecture of a classical computing system, according to some embodiments.

DETAILED DESCRIPTION

[0027] The figures and the following description relate to preferred embodiments by way of illustration only. It should be noted that from the following discussion, alternative embodiments of the structures and methods disclosed herein will be readily recognized as viable alternatives that may be employed without departing from the principles of what is claimed.

I. OVERVIEW

[0028] This disclosure studies quantum interior point methods (QIPMs) for second-order cone programming (SOCP), guided by the example use case of portfolio optimization (PO). This disclosure provides a complete quantum circuit-level description of the algorithm from problem input to problem output, making several improvements to the implementation of the QIPM. This disclosure reports the number of logical qubits and the quantity/depth of non-Clifford T-gates used to run the algorithm, including constant factors. The determined resource counts depend on instance-specific parameters, such as the condition number of certain linear systems within the problem. To determine the size of these parameters, numerical simulations of small PO instances are performed, which lead to concrete resource estimates for the PO use case. The numerical results do not probe large enough instance sizes to make conclusive statements about the asymptotic scaling of the algorithm. However, already at small instance sizes, the analysis suggests that, due primarily to large constant pre-factors, poorly conditioned linear systems, and a fundamental reliance on costly quantum state tomography, fundamental improvements to the QIPM are desired for it to lead to practical quantum advantage.

A. Introduction

[0029] The practical utility of determining optimal solutions to well-posed optimization problems has been known since the days of antiquity, with Euclid considering the minimal distance between two points using a line. In the modern era, optimization algorithms for business and financial use cases continue to be ubiquitous. Partly as a result of this utility, algorithmic techniques for optimization problems have been well studied since even before the invention of the computer, including a famous dispute between Legendre and Gauss on who was responsible for the invention of least squares fitting. With the advent of the quantum era, there has been great interest in developing quantum algorithms that solve optimization problems with provable speedups over classical algorithms.

[0030] Unfortunately, it can be difficult to evaluate whether these quantum algorithms will be practically useful. In some cases, the algorithms are heuristic, and their performance can only be measured empirically once it is possible to run them on actual quantum hardware. In other cases, the difficulty in evaluating practicality stems from the inherent complexity of combining many distinct ingredients, each with their own caveats and bottlenecks. To make an apples-to-apples comparison and quantify advantages of a quantum algorithm, an end-to-end resource analysis that accounts for all costs from problem input to problem output may be performed.

[0031] Such an end-to-end analysis for a quantum interior point method (QIPM) was performed for solving second-order cone programs (SOCPs). In particular, this disclosure focuses on a concrete use case with very broad applications, but of interest in the financial services sector: portfolio optimization (PO). In general, PO is the task of determining the optimal resource allocation to a collection of possible classes to optimize a given objective. In finance, one seeks to determine the optimal allocation of funds across a set of possible assets that maximizes returns and minimizes risk, subject to constraints. Noteworthy, many variants of the PO problem can be cast as a SOCP and subsequently solved with a classical or quantum interior point method. Indeed, classical interior point methods (CIPMs) are efficient not only in theory, but also in practice; they are the method of choice within fast numerical solvers for SOCPs and other conic programs, which encompass a large variety of optimization problems that appear in industry. Notably, QIPMs structurally mirror CIPMs, and seek improvements by replacing certain subroutines with quantum primitives. Thus, compared to other proposed quantum algorithms for conic programs not based on widely used classical techniques (e.g., solvers that leverage the multiplicative weights update method), QIPMs are uniquely positioned to provide not only a theoretical asymptotic advantage, but also a practical quantum solution for this common class of problem.

[0032] However, the QIPM is a complex algorithm that delicately combines some purely classical steps with multiple distinct quantum subroutines. The runtime of the QIPM is stated in terms of several parameters that can only be evaluated once a particular use case has been specified; depending on how these parameters scale, an asymptotic speedup may or may not be achievable. Additionally, any speedup is contingent on access to a large quantum random access memory (QRAM), an ingredient that in prior asymptotic-focused analyses has typically been assumed to exist without much further justification or cost analysis.

[0033] The resource analysis is detailed and takes care to study aspects of the end-to-end pipeline, including the QRAM component. This disclosure reports results in terms of relevant problem parameters, and then describes numerical experiments to determine the size and scaling of these parameters for actual randomly chosen instances of the PO problem, based on historical stock data. This approach allows us to estimate the exact resource cost of the QIPM for an example PO problem, including a detailed breakdown of costs by various subroutines. This estimate incorporates several optimizations to the underlying subroutines, and technical improvements to how they are integrated into the QIPM. Consequently, our analysis allows us to evaluate the prospect that the algorithm may exhibit a practical quantum advantage, and it reveals the computational bottlenecks within the algorithm that are most in need of further improvement.

[0034] While this disclosure focuses on the QIPM and its application to the PO problem, this disclosure has more

general applications and more general takeaways for quantum algorithms and for quantum computing applications. Firstly, the results emphasize the importance of end-to-end analysis when evaluating a proposed application. Furthermore, the modular treatment of the underlying algorithmic primitives produces quantitative and qualitative takeaways that are relevant for end-to-end treatments of a large number of other algorithms that also rely on these subroutines, especially those in the area of machine learning, where data access via QRAM and quantum linear algebra techniques are often used.

B. Results

[0035] The resource analysis focuses on three central quantities that determine the overall cost of algorithms implemented on fault-tolerant quantum computers: the number of logical qubits, the total number of T gates ("T-count"), and the number of parallel layers of T gates ("T-depth") useed to construct quantum circuits for solving the problem. The T-depth acts as a proxy for the overall runtime of the algorithm, whereas the T-count and number of logical qubits are helpful for determining how many physical qubits may be used for a full, fault-tolerant implementation. We justify the focus on T gates by pointing out that, in many prominent approaches to fault-tolerant quantum computation, quantum circuits are decomposed into Clifford gates and T gates, and the cost of implementing the circuit is dominated by the number and depth of the T gates. The fault-tolerant Clifford gates can be performed transversally or even in software, whereas the T gates use the expensive process of magic state distillation. This disclosure stops short of a full analysis of the algorithm at the physical level, as the logical analysis seems to suffice to evaluate the overall outlook for the algorithm and identify its main bottlenecks.

[0036] At the core of any interior point method (IPM) is the solving of a linear system of equations. The QIPM performs this step using a quantum linear system solver (QLSS) together with pure state quantum tomography. The cost of QLSS depends on a parameter κ_F, the Frobenius condition number $\|G\|_F\|G^{-1}\|$ of the matrix G that is inverted (where $\|\bullet\|_F$ denotes the Frobenius norm, and $\|\bullet\|$ denotes the spectral norm), while the cost of tomography depends on a parameter ξ, a precision parameter. These parameters are evaluated empirically by simulating the QIPM on small instances of the PO problem.

[0037] Table I reports a summary of overall resource calculation, in which the asymptotically leading term is shown (along with its constant prefactor) in terms of parameters κ_F and ξ, as well as n, the number of assets in the PO instance, and $\in$, the desired precision to which the portfolio should be optimized. It is determined (numerically) that κ_F grows with n, and that ξ shrinks with n; it is estimated that, at n=100 and $\in=10^{-7}$, the implementation of the QIPM may use 8×10^6 qubits and 8×10^{29} total T gates spread out over 2×10^{24} layers. These resource counts are decidedly out of reach both in the near and far term for quantum hardware, even for a problem of modest size by classical standards. Even if quantum computers one day match the gigahertz-level clock-speeds of modern classical computers, 10^{24} layers of T gates would take millions of years to execute. By contrast, the PO problem can be easily solved in a matter of seconds on a laptop for n=100 stocks.

[0038] This disclosure cautions that the numbers reported should not be interpreted as the final word on the cost of the QIPM for PO. Further examination of the algorithm may uncover many improvements and optimizations that may reduce the costs compared to the current calculations. On the other hand, the results do already incorporate several innovations made to reduce the resource cost, including preconditioning the linear system.

[0039] Besides the main resource calculation, this disclosure makes several additional contributions and observations:

[0040] 1. This disclosure provides explicit example quantum circuits for useful (e.g., important) subroutines of the QIPM, namely the state-of-the-art QLSS based on the discrete adiabatic theorem and pure state tomography, which complement the circuits for block-encoding (using QRAM). These example quantum circuits, and their precise resource calculations, may be useful elsewhere, as these subroutines are ubiquitous in quantum algorithms. See section IV F and section V for additional details.

[0041] 2. This disclosure breaks down the resource calculation into its constituents to illustrate which parts of the algorithm are most costly. This disclosure determines that many independent factors create significant challenges toward realizing quantum advantage with QIPMs, and this work underscores aspects of the algorithm that may be improved. This disclosure also notes that the conditions under which QIPMs would be most successful (e.g., when κ_F is small) also allow for classical IPMs based on iterative classical linear system solvers to be competitive. See section VII for additional details.

[0042] 3. This disclosure numerically simulates several versions of the full QIPM solving the PO problem on portfolios as large as n=120 stocks, and this disclosure reports the empirical size and scaling of the relevant parameters κ_F and ξ. There is considerable variability in the trends observes, depending on which version of the QIPM is chosen, and when the QIPM is terminated, which makes it difficult to draw robust conclusions. However, this disclosure determines that both κ_F and ξ^{-1} appear to grow with n. Note that previous numerical experiments on a similar formulation of the PO problem suggested κ_F does not grow with problem size, but those pervius experiments scaled the number of "time epochs" while keeping n constant. Additionally, this disclosure observes that the "infeasible" version of the QIPM originally empirically performs similarly to more sophisticated "feasible" versions, despite not enjoying the same theoretical guarantees of fast convergence. Finally, contrary to theoretical expectation, this disclosure observes that κ_F and ξ^{-1} do not diverge as $\in\rightarrow0$. See section VI for additional details.

[0043] 4. This disclosure makes various technical improvements to the underlying ingredients of QIPMs:

[0044] Tomographic precision: Performing tomography on the output of a QLSS necessarily causes the classical estimate of the solution to the linear system to be inexact. This disclosure describes how the allowable amount of tomography precision can be determined adaptively rather than relying on theoretical bounds. Nonetheless, this disclosure also improves the constant prefactor in the tomographic bounds. The total number of state preparation queries used to learn an unknown

L-dimensional pure state to ξ error using a tomography method is to leading order at most $115L\ln(L)/\xi^2$.

[0045] Norm of the linear system: Since QLSSs output a normalized quantum state, tomography does not directly yield the norm of the solution to the linear system. The norm can be learned through more complicated protocols, but it is observed that in the context of QIPMs, a sufficient estimate for the norm can be learned classically.

[0046] Preconditioning: a preconditioning method is proposed that is compatible with the QIPM, while reducing the parameter κ_F. The numerical simulations suggest the reduction is more than an order of magnitude for the portfolio optimization problem.

[0047] Feasible QIPM: A "feasible" version of a QIPM is implemended which includes determining a basis for the null space of the SOCP matrix. This disclosure identifies an explicit basis for the PO problem, thereby avoiding a costly QR decomposition. However, this disclosure observes that determines the basis via QR decomposition leads to more stable numerical results.

TABLE I illustrates asymptotic, leading-order contributions to the total quantum resources for an end-to-end portfolio optimization (including constant factors), in terms of the number of assets in the portfolio (n), the desired precision to which the portfolio should be optimized (ϵ), the maximum Frobenius condition number of matrices encountered by the QIPM (κ_F), and the minimum tomographic precision for the algorithm to succeed (ξ). The T-depth and T-count expressions represent the cumulative cost of $\mathcal{O}(\xi^{-2}\,n^{1.5}\log(n)\log(\epsilon^{-1}))$ individual quantum circuits performed serially, a quantity that we estimate evaluates to 6×10^{12} circuits at n=100; see table X for a detailed accounting. The right column uses a numerical simulation of the quantum algorithm (see section VI) to compute the instance-specific parameters in the resource expression and estimate the resource cost at n=100 and $\epsilon=10^{-7}$.

TABLE I

Resource	QIPM complexity	Estimated at n = 100
Number of logical qubits	$800\,n^2$	8×10^6
T-depth	$(2\times10^{10})\kappa_F n^{1.5}\xi^{-2}\log_2(n)\log_2(\epsilon^{-1})\log_2(\kappa_F n^{14/27}\xi^{-1})$	2×10^{24}
T-count	$(7\times10^{11})\kappa_F n^{3.5}\xi^{-2}\log_2(n)\log_2(\epsilon^{-1})\log_2(\kappa_F\xi^{-1})$	8×10^{29}

[0048] The outline for the remainder of this disclosure is as follows. Section II describes and defines the portfolio optimization problem in terms of Markowitz portfolio theory. Section III describes Second Order Cone Programming (SOCP) problems, illustrate how portfolio optimization can be represented as an instance of SOCP, and discuss how IPMs can be used for solving SOCPs. Section IV review the quantum ingredients used to turn an IPM into a QIPM. In particular, this disclosure reviews quantum linear system solvers, block-encoding for data loading, and quantum state tomography for data read out. This disclosure also presents better bounds on the tomography procedure than were previously known. Section V describes the implementation of using QIPM and quantum algorithms for SOCP for the portfolio optimization problem, including a detailed resource estimate for the end-to-end problem. Section VI shows numerical results from simulations of the full problem, and section VII reflects on the calculations performed, identifying the main bottlenecks and drawing conclusions about the outlook for quantum advantage with QIPM.

[0049] The QIPM has many moving parts using several mathematical symbols. While all symbols are defined as they are introduced in the text, this disclosure also provides a full list of symbols for the reader's reference in the section Additional Information A. Throughout the paper, all vectors are denoted in bold lowercase letters to contrast with scalar quantities (unbolded lowercase) and matrices (unbolded uppercase). The only exception to this rule will be the symbols N, K, and L, which are positive integers (despite being uppercase), and denote the number of rows or columns in certain matrices related to an SOCP instance.

II. PORTFOLIO OPTIMIZATION (PO)

A. Introduction

[0050] Portfolio optimization is the process widely used, for example, by financial analysts to assign allocations of capital across a set of assets within a portfolio, given optimization criteria such as maximizing the expected return and minimizing the financial risk. The creation of the mathematical framework for modern portfolio theory (MPT) is credited to Harry Markowitz, for which he received the 1990 Alfred Nobel Memorial Prize in Economic Sciences. Markowitz describes the process of selecting a portfolio in two stages, where the first stage starts with "observation and experience" and ends with "beliefs about the future performances of available securities." The second stage starts with "the relevant beliefs about future performances" and ends with "the choice of portfolio." The theory is also known as mean-variance analysis.

[0051] Typically, portfolio optimization strategies include diversification, which is the practice of investing in a wide array of asset types and classes as a risk mitigation strategy. Some popular asset classes are stocks, bonds, real estate, commodities, and cash. After building a portfolio, one may expect a return (or profit) after a specific period of time. Risk is defined as the fluctuations of the asset value. MPT describes how high variance assets can be combined with other uncorrelated assets through diversification to create portfolios with low variance on their return. Naturally, among equal-risk portfolios, investors prefer those with higher expected return, and among equal-return portfolios, they prefer those with lower risk.

B. Mathematical Formulation

[0052] Within a portfolio, w_i represents the amount of an asset i being held over some period of time. Often, this

amount is given as the asset's price in dollars at the start of the period. When the price is positive ($w_i>0$), this is referred to as a long position; and when the price is negative ($w_i<0$), this is referred to as a short position with an obligation to buy this asset at the end of the period. Typically, investment banks hold long positions, while hedge funds build portfolios with short positions that have higher risk due to the uncertainty of the price to buy the asset at the end of the period. The optimization variable in the portfolio optimization problem is the vector of n assets $w\in\mathbb{R}^n$ in the portfolio.

[0053] The price of each asset i varies over time. u_i is defined to be the relative change (positive or negative) during the period of interest. Then, the return of the portfolio for that period is defined as $\bar{r}=u^Tw$ dollars. The relative changes $u\in\mathbb{R}^n$ follow a stochastic process, and this can be modeled with a random vector with mean $\hat{u}$ and covariance Σ. The return $\bar{r}$ is then a random variable with mean $\hat{u}^Tw$ and covariance $w^T\Sigma w$.

[0054] To capture realistic problem formulations, one or more mathematical constraints may be added to the optimization problem corresponding to the problem-specific considerations. For example, two common constraints in portfolio optimization problems are no short positions ($w_i\geq 0$ for all i, denoted by $w\geq 0$) and that the total investment budget is limited ($1^Tw=1$, where 1 denotes the vector of ones). This forms the classical portfolio optimization problem from Markowitz's mean-variance theory:

$$\begin{aligned}&\min_w \quad w^\top\textstyle\sum w \\ &\text{s.t.} \quad \hat{u}^\top w\geq \bar{r}_{min} \\ &\quad 1^\top w=1 \\ &\quad w\geq 0\end{aligned} \tag{1}$$

This formulation is a quadratic optimization problem where risk is minimized, while achieving a target return of at least $\bar{r}_{min}$ with a fixed budget and no short positions. In practice, the portfolio optimization problem is often reformulated in other ways, for example, to maximize return subject to a fixed amount of risk, or to optimize an objective function that weighs risk against return. The current application follows the latter approach, formulated as follows, where q is a tunable risk-aversion coefficient:

$$\begin{aligned}&\min_w \quad -\hat{u}^\top w+q\sqrt{w^\top\textstyle\sum w} \\ &\text{s.t.} \quad 1^\top w=1 \\ &\quad w\geq 0\end{aligned} \tag{2}$$

This optimization problem is no longer a QO problem, but it can be mapped to a conic problem, as described later in section III B. Depending on the problem, additional constraints can be added. For instance, constraints can be added to allow short positions, component-wise short sale limits, or a total short sale limit. Another variant of this is a constraint for a collateralization requirement, which limits the total of short positions to a fraction of the total long positions. Often, buying or selling an asset results in a transaction fee that is proportional to the amount of asset that is bought or sold. Linear transaction costs or maximum transaction amounts are often included as constraints in portfolio optimization. Diversification constraints can limit portfolio risk by limiting the exposure to individual positions and groups of assets within particular sectors. To illustrate the flexibility of this analysis, a maximum transaction constraint is included and use the following problem formulation is used in the analysis in the rest of the disclosure:

$$\begin{aligned}&\min_w \quad -\hat{u}^\top w+q\sqrt{w^\top\textstyle\sum w} \\ &\text{s.t.} \quad 1^\top w=1 \\ &\quad |w-\overline{w}|\leq\zeta \\ &\quad w\geq 0,\end{aligned} \tag{3}$$

where $\overline{w}$ denotes the current portfolio, so that $|w-\overline{w}|$ is the vector of transaction quantities for each asset, which are constrained to be smaller than maximum values contained in the vector ζ.

III. SECOND ORDER CONE PROGRAMMING (SOCP) AND INTERIOR POINT METHODS (IPM)

A. Definitions

[0055] Second-order cone programming (SOCP) is a type of convex optimization that allows for a richer set of constraints than linear programming (LP), without many of the complications of semidefinite programming (SDP). Indeed, SOCP is a subset of SDP, but SOCP admits interior point methods (IPMs) that may be just as efficient as IPMs for LP. Many real-world problems can be cast as SOCP, including the example portfolio optimization problem of interest.

[0056] For any k-dimensional vector v, the following may be used $v=(v_0;\tilde{v})$, where v_0 is the first entry of v, and $\tilde{v}$ contains the remaining k−1 entries.

[0057] Definition 1. A k-dimensional second-order cone (for $k\geq 2$) is the convex set

$$\mathcal{Q}^k=\{(x_0;\tilde{x})\in\mathbb{R}^k \mid x_0\geq\|\tilde{x}\|\}, \tag{4}$$

where $\|\bullet\|$ denotes the vector two-norm (standard Euclidean norm). For k=1, $\mathcal{Q}^1=\{x_0\in\mathbb{R}\mid x_0\geq 0\}$.

[0058] Definition 2. In general, a second-order cone problem is formulated as

$$\begin{aligned}&\min_x \quad c^\top x \\ &\text{s.t.} \quad Ax=b \\ &\quad x\in\mathcal{Q},\end{aligned} \tag{5}$$

where $\mathcal{Q}=\mathcal{Q}^{N_1}\times\ldots\times\mathcal{Q}^{N_r}$ is a Cartesian product of r second-order cones of combined dimension $N=N_1+\ldots+N_r$, and A is a full-rank K×N matrix encoding K linear equality constraints, with K≤N.

[0059] Note that the special case of linear programming is immediately recovered if $N_i=1$ for all i. We say that a point x is primal feasible whenever Ax=b and $x\in\mathcal{Q}$. It is strictly primal feasible if additionally it lies in the interior of $\mathcal{Q}$.

[0060] The dual to problem in eq. (5) is a maximization problem over a variable $y \in \mathbb{R}^K$, given as follows:

$$\begin{aligned} \min_{y} \quad & b^\top y \\ \text{s.t.} \quad & A^\top y + s = c \\ & s \in \mathcal{Q}. \end{aligned} \tag{6}$$

[0061] We say that a pair (s; y) is dual feasible whenever $A^Ty+S=c$ and $s \in \mathcal{Q}$. For any point (x; y; s) with x, $s \in \mathcal{Q}$, the duality gap may be defined as

$$\mu(x, s) := \frac{1}{r} x^\top s = \frac{1}{r}(c^\top x - b^\top y), \tag{7}$$

where r is the number of cones, as in definition 2, and the second equality holds under the additional assumption that the point is primal and dual feasible. The fact that x, $s \in \mathcal{Q}$ implies that $\mu(x, s) \geq 0$. Moreover, assuming that both the primal and dual problems have a strictly feasible point, the optimal primal solution x* and the optimal dual solution (y*; s*) are guaranteed to exist and satisfy $c^Tx^*=b^Ty^*$, and hence

$$\mu = \frac{1}{r} x^{*\top} s^* = x^{*\top}(c - A^\top y^*) = c^\top x^* - b^\top y^* = 0.$$

Thus, the primal-dual condition of optimality can be expressed by the system

$$\begin{aligned} & Ax = b \\ & A^Ty + s = c \\ & x^Ts = 0 \\ & x \in \mathcal{Q}, s \in \mathcal{Q}. \end{aligned} \tag{8}$$

B. Portfolio Optimization as SOCP

[0062] The portfolio optimization problem can be solved by reduction to SOCP, and this reduction is often made in practice. Here this disclosure describes one way of translating the portfolio optimization problem, as given in eq. (3) into a second-order cone program.

[0063] The objective function in eq. (3) has a non-linear term $q\sqrt{w^T\Sigma w}$, which may be linearized by introducing a new scalar variable t, and a new constraint $t \geq \sqrt{w^T\Sigma w}$. This results in the equivalent optimization problem

$$\begin{aligned} \min_{x=(w;t)} \quad & [-\hat{u}; q]^\top (w; t) \\ \text{s.t.} \quad & 1^\top w = 1 \\ & |w_i - \overline{w}_i| \leq \zeta_i \\ & w_i \geq 0 \\ & t^2 \geq w^\top \Sigma w. \end{aligned} \tag{9}$$

[0064] A goal now is to express the constraints in eq. (9) as second-order cone constraints. Given an m×n matrix M for which $\Sigma=M^TM$, the constraint on t can be expressed by introducing an m-dimensional variable η subject to the equality constraint $\eta=Mw$ and the second-order cone constraint $(t; \eta) \in \mathcal{Q}^{m+1}$.

[0065] The matrix M can be determined from Σ via a Cholesky decomposition, although for large matrices Σ, this computation may be costly. Alternatively, if Σ and $\hat{\mu}$ are calculated from stock return vectors $u^{(1)}, \ldots, u^{(m)}$ during m independent time epochs (e.g. returns for each of m days or each of m months), then a valid matrix M^T is given by $(u^{(1)}-\hat{u}, \ldots, u^{(m)}-\hat{u})$, i.e. the columns of M^T are given by the deviation of the returns from the mean in each epoch. This is the approach taken in the numerical experiments herein (presented later).

[0066] The absolute value constraints are handled by introducing a pair of n-dimensional variables ϕ and ρ, subject to equality constraints $\phi=\zeta-(w-\overline{w})$ and $\rho=\zeta+(w-\overline{w})$. The absolute value constraints are then imposed as positivity constraints $\phi_i \leq 0$, $\rho_i \geq 0$, which are included as second-order cone constraints of dimension 1. Alternatively, the absolute value constraints may be encoded with n second-order cone constraints of dimension 2; these formulations are equivalent up to a simple coordinate change, and one may opt to use 1-dimensional cones for their simplicity of presentation.

[0067] In summary, the portfolio optimization problem from eq. (3) may be described as the following SOCP that minimizes over the variable $x=(w; \phi; \rho; t; \eta) \in \mathbb{R}^{3n+m+1}$:

$$\begin{aligned} \min_{x} \quad & [-\hat{u}; 0; 0; q; 0]^\top (w; \phi; \rho; t; \eta) = : c^\top x \\ \text{s.t.} \quad & \begin{pmatrix} 1^\top & 0^\top & 0^\top & 0 & 0^\top \\ I & I & 0 & 0 & 0 \\ I & 0 & -I & 0 & 0 \\ M & 0 & 0 & 0 & -I \end{pmatrix} \begin{pmatrix} w \\ \phi \\ \rho \\ t \\ \eta \end{pmatrix} = \begin{pmatrix} 1 \\ \overline{w} + \zeta \\ \overline{w} - \zeta \\ 0 \end{pmatrix} \\ & (w; \phi; \rho; t; \eta) \in \underbrace{\mathcal{Q}^1 \times \cdots \times \mathcal{Q}^1}_{n \text{ positivity constraints}} \\ & \times \underbrace{\mathcal{Q}^1 \times \cdots \times \mathcal{Q}^1}_{2n \text{ budget constraints}} \\ & \times \underbrace{\mathcal{Q}^{m+1}}_{\text{risk}}, \end{aligned} \tag{10}$$

where I denotes an identity block, 0 denotes a submatrix of all 0s, 0 is a vector of all 0s, 1 is a vector of all 1s, and the size of each block of A can be inferred from its location in the matrix. Thus, the total number of cones is r=3n+1, and the combined dimension is N=3n+m+1. Note that r=2n+1 cones if the absolute value constraints are represented using dimension-2 cones. The SOCP constraint matrix A is a K×N matrix, with K=2n+m+1.

[0068] Notice that many of the rows of the K×N matrix A are sparse and contain only one or two nonzero entries. However, the final m rows of the matrix A will be dense and contain n+1 nonzero entries due to the appearance of the matrix M containing historical stock data; in total a constant fraction of the matrix entries will be nonzero, so sparse matrix techniques will provide only limited benefit.

[0069] Additionally, note that the primal SOCP in eq. (10) has an interior feasible point as long as (has strictly positive entries. To better understand this, choose w to be any strictly

positive vector that satisfies $|w-\overline{w}|<\zeta$, and let $\phi=\zeta+(\overline{w}-w)$, $\rho=\zeta-(\overline{w}-w)$, $\eta=Mw$, and t equal to any number strictly greater than $\|\eta\|$. It can be verified that the dual program likewise has a strictly feasible point; this guarantees that the optimal primal-dual pair for the SOCP exists and satisfies eq. (8).

C. Interior Point Methods for SOCP

1. Introduction

[0070] Interior point methods (IPMs) are a class of efficient algorithms for solving convex optimization problems including LPs, SOCPs, and SDPs, where (in contrast to the simplex method) intermediate points generated by the method lie in the interior of the convex set, and they are guaranteed to approach the optimal point after a polynomial number of iterations of the method. Each iteration involves forming a linear system of equations that depends on the current intermediate point. The solution to this linear system determines the search direction, and the next intermediate point is formed by taking a small step in that direction. This disclosure considers path-following primal-dual IPMs, where, if the step size is sufficiently small, the intermediate points are guaranteed to approximately follow the central path, which ends at the optimal point for the convex optimization problem.

2. Central Path

[0071] To define the central path, first establish some notation related to the algebraic properties of the second-order cone. Let the product $u\circ v$ of two vectors $u=(u_0; \tilde{u})$, $v=(v_0; \tilde{v}) \in \mathcal{Q}^k$ be defined as

$$u\circ v=(u^T V;u_0\tilde{v}+v_0\tilde{u}) \tag{11}$$

and the identity element for this product is denoted by the vector $e=(1; 0)\in \mathcal{Q}^k$. For the Cartesian product $\mathcal{Q}=\mathcal{Q}^{N_1}\times \ldots \times\mathcal{Q}^{N_r}$ of multiple second-order cones, the vector e is defined as the concatenation of the identity element for each cone, and the circle product of two vectors is given by the concatenation of the circle product of each constituent. A consequence of this definition is the that e^Te is equal to the number of cones r.

[0072] Now, for the SOCP problem of eq. (5), the central path (x(v); y(v); s(v)) is the one-dimensional set of central points, parameterized by $v\in[0, \infty)$, which satisfies the conditions:

$$\begin{aligned} &Ax(v)=b\\ &A^Ty(v)+s(v)=c\\ &x(v)\circ s(v)=ve\\ &x(v)\in \mathcal{Q},s(v)\in \mathcal{Q}. \end{aligned} \tag{12}$$

Note that the central path point (x(v); y(v); s(v)) has a duality gap that satisfies $\mu(x(v), s(v))=v$, and that when v=0, eq. (12) recovers eq. (8).

3. Determining an Initial Point on the Central Path Via Self-Dual Embedding

[0073] Path-following primal-dual interior point methods determine the optimal point by beginning at a central point with v>0 and following the central path to a very small value of v, which is taken to be a good approximation of the optimal point. For a given SOCP, determining an initial point on the central path is non-trivial and, in general, can be just as hard as solving the SOCP itself. One solution to this problem is the homogeneous self-dual embedding, a slightly larger self-dual SOCP is formed with the properties that (i) the optimal point for the original SOCP can be determined from the optimal point for the self-dual SOCP and (ii) the self-dual SOCP has a trivial central point that can be used to initialize the IPM.

[0074] To do this, introduce new scalar variables τ, θ, and κ, which are used to give more flexibility to the constraints. Previously, Ax=b was used. In the larger program, this constraint is relaxed to read $Ax=b\tau-(b-Ae)\theta$, such that the original constraint is recovered when $\tau=1$ and $\theta=0$, but x=e is a trivial solution when $\tau=1$ and $\theta=1$. Similarly, the constraint $A^Ty+s=c$ is relaxed to read $A^Ty+s=c\tau-(c-e)\theta$, which has the trivial solution y=0, s=e when $\tau=\theta=1$. These can be complemented with two additional linear constraints to form the program:

$$\begin{aligned} &\min_{(x;y;\tau;\theta;s;\kappa)} \quad (r+1)\theta \\ &\begin{pmatrix} 0 & A^\top & -c & \bar{c} \\ -A & 0 & b & -\bar{b} \\ c^\top & -b^\top & 0 & -\bar{z} \\ -\bar{c}^\top & \bar{b}^\top & \bar{z} & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ \tau \\ \theta \end{pmatrix} + \begin{pmatrix} s \\ 0 \\ \kappa \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ r+1 \end{pmatrix} \\ &x, s \in \hat{\mathcal{Q}} \quad \tau, \kappa \geq 0; \quad y, \theta \text{ free}, \end{aligned} \tag{13}$$

where $\bar{b}=b-Ae$, $\bar{c}=c-e$, $\bar{z}=c^Te+1$, and $r=e^Te$ is the number of cones in the original SOCP. While eq. (13) is not exactly of the form given in eq. (5), it can still be considered a primal SOCP. Since the block matrix in eq. (13) is skew-symmetric and the objective function coefficients are equal to the right-hand-side of the equality constraints, when the dual program (c.f. eq. (6)) is computed, an equivalent program is arrived at; thus, eq. (13) is self-dual. Thus, when applying path-following primal-dual IPMs to eq. (13), in some embodiments, only the primal variables (that is x, y, τ, θ, s, K) may need to be tracked. Taking into account the addition of τ and κ, which are effectively an extra pair of primal-dual variables, the duality gap (c.f. eq. (7)) is defined as

$$\mu(x, \tau, s, \kappa) := \frac{1}{r+1}(x^\top s+\kappa\tau). \tag{14}$$

Note that if the point (x; y; τ; θ; s; K) is feasible, i.e., if it satisfies the four linear constraints in eq. (13), then the following identity is determined:

$$\begin{aligned} \mu(x, \tau, s, \kappa) &= \frac{-x^\top A^\top y+x^\top c\tau-x^\top\bar{c}\theta+\kappa\tau}{r+1} \\ &= \frac{-b^\top y\tau+\bar{b}^\top y\theta+x^\top c\tau-x^\top\bar{c}\theta+\kappa\tau}{r+1} \\ &= \frac{\bar{b}^\top y\theta-x^\top\bar{c}\theta+\bar{z}\tau\theta}{r+1} \\ &= \theta, \end{aligned} \tag{15}$$

where the first, second, third, and fourth rows of eq. (13) are invoked above in lines one, two, three, and four, respectively. This equality justifies the redefinition in eq. (14): noting that the primal objective function in eq. (13) is (r+1)θ, and (since the program is self-dual) the associated dual objective function is −(r+1)θ, note that the gap between primal and dual objective functions, divided by the number of conic constraints (2r+2), is exactly equal to θ.

[0075] The central path for the augmented SOCP in eq. (13) is defined by the feasibility conditions for the SOCP combined with the relaxed complementarity conditions x∘s=νe and κτ=ν. Thus, the point (x=e; y=0; τ=1; θ=1; s=e; κ=1) is not only a feasible point for the SOCP in eq. (13), but also a central point with ν=1.

[0076] Finally, a noteworthy property of the self-dual SOCP in eq. (13) is that the optimal point for the original SOCP in eq. (5) can be derived from the optimal point for the SOCP in eq. (13). Specifically, let (x_{sd}*; y_{sd}*; τ*; θ*; s_{sd}*; κ*) be the optimal point for eq. (13) (it can be shown that θ*=0). Then if

$$\tau^* > 0,\ (x^*; y^*; s^*) = \left(\frac{x^*_{sd}}{\tau^*}; \frac{y^*_{sd}}{\tau^*}; \frac{s^*_{sd}}{\tau^*}\right)$$

is an optimal primal-dual point for eqs. (5) and (6). If τ*=0, then at least one of the original primal SOCP in eq. (5) and the original dual SOCP in eq. (6) must be infeasible. As previously demonstrated, the specific SOCP for portfolio optimization in eq. (10) is primal and dual feasible, so τ*≠0 for that example.

[0077] What if there is only a point that is approximately optimal for the self-dual SOCP? An approximately optimal point for the original SOCP can still be determined. Suppose a feasible point for which μ (x, τ, s, κ)=∈. The point (x/τ; y/τ; s/τ) is $\mathcal{O}(\in)$ close ti feasible for the original SOCP in the sense that the equality constraints are satisfied up to $\mathcal{O}(\in)$ error

$$\left\|A\frac{x}{\tau} - b\right\| = \frac{\epsilon}{\tau}\|b - Ae\| \tag{16}$$

$$\left\|A^T\frac{y}{\tau} + \frac{s}{\tau} - c\right\| = \frac{\epsilon}{\tau}\|c - e\|. \tag{17}$$

[0078] Moreover, since κ>0 and θ=∈, assert using the third row of eq. (13) that the difference in objective function achieved by the primal and dual solutions is also $\mathcal{O}(\in)$, that is

$$c^\top\frac{x}{\tau} - b^\top\frac{y}{\tau} \le \frac{|c^\top e + 1|}{\tau}\epsilon. \tag{18}$$

In summary, by using the self-dual SOCP of eq. (13), a trivial point is obtained from which to start the IPM, and given an (approximately) optimal point, the following is obtained: either an (approximately) optimal point to the original SOCP or a certificate that the original SOCP was not feasible to begin with.

4. Iterating the IPM

[0079] Each iteration of the IPM takes as input an intermediate point (x; y; τ; θ; s; κ) that is feasible (or in some formulations, nearly feasible), has duality gap

$$\frac{1}{r+1}(x^\top s + \kappa\tau)$$

equal to μ, and is close to the central path with parameter ν=μ. The output of the iteration is a new intermediate point (x+Δx; y+Δy; τ+ΔT; θ+Δθ; s+Δs; κ+Δκ) that is also feasible and close to the central path, with a reduced value of the duality gap. Thus, many iterations lead to a solution with duality gap arbitrarily close to zero.

[0080] One additional input is the step size, governed by a parameter σ<1. The IPM iteration aims to bring the next intermediate point onto the central path with parameter ν=σμ. This is accomplished by taking one step using Newton's method, where the vector (Δx; Δy; Δτ; Δθ; Δs; Δκ) is uniquely determined by solving a linear system of equations called the Newton system. The first part of the Newton system is the conditions to be met for the new point to be feasible, given in the following system of N+K+2 linear equations:

$$\begin{pmatrix} 0 & A^\top & -c & \bar{c} \\ -A & 0 & b & -\bar{b} \\ c^\top & -b^\top & 0 & -\bar{z} \\ -\bar{c}^\top & \bar{b}^\top & \bar{z} & 0 \end{pmatrix}\begin{pmatrix}\Delta x\\ \Delta y\\ \Delta\tau\\ \Delta\theta\end{pmatrix} + \begin{pmatrix}\Delta s\\ 0\\ \Delta\kappa\\ 0\end{pmatrix} = \begin{pmatrix} -A^\top y + c\tau - \bar{c}\theta - s \\ Ax - b\tau + \bar{b}\theta \\ -c^\top x + b^\top y + \bar{z}\theta \\ \bar{c}^\top x - \bar{b}^\top y - \bar{z}\tau \end{pmatrix} \tag{19}$$

Note that if the point is already feasible, the right-hand-side is equal to zero.

[0081] The second part of the Newton system is the linearized conditions for arriving at the point on the central path with duality gap σμ. That is, aim for (x+Δx)∘(s+Δs) =σμe and (κ+Δκ)(τ+Δτ)=σμ. By ignoring second order terms (i.e., the $\mathcal{O}(\Delta x\circ\Delta s)$ and $\mathcal{O}(\Delta\kappa\Delta\tau)$ terms), these become

$$x\circ\Delta s + s\circ\Delta x = \sigma\mu e - x\circ s$$

$$\kappa\Delta\tau + \tau\Delta\kappa = \sigma\mu - \kappa\tau. \tag{20}$$

[0082] The expression above can be rewritten as a matrix equation by first defining the arrowhead matrix U for a vector u=(u_0; ũ)∈Q^k as

$$U = \begin{pmatrix} u_0 & \tilde{u}^\top \\ \tilde{u} & u_0 I \end{pmatrix} = ue^\top + eu^\top + u_0 I - 2u_0 ee^\top. \tag{21}$$

When u∈ $\mathcal{Q}$ lies in the direct product of multiple second-order cones, the arrowhead matrix is $\mathcal{Q}$ formed by placing the appropriate matrices of the above form on the block diagonal. The arrowhead matrix has the property that for any vector v, Uv=u∘v.

[0083] Using this notation, the Newton equations in eq. (20) can be written as

$$\begin{pmatrix} S & 0 & 0 & 0 & X & 0 \\ 0 & 0 & \kappa & 0 & 0 & \tau \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta \tau \\ \Delta \theta \\ \Delta s \\ \Delta \kappa \end{pmatrix} = \begin{pmatrix} \sigma \mu e - Xs \\ \sigma \mu - \kappa \tau \end{pmatrix}, \tag{22}$$

where X and S are the arrowhead matrices for vectors x and s.

[0084] Equations (19) and (22) together form the Newton system. It is observered that there are 2N+K+3 constraints to match the 2N+K+3 variables in the vector (Δx; Δy; Δτ; Δθ; Δs; Δκ). As long as the duality gap is positive and (x; y; τ; θ; s; κ) is not too far from the central path (which will be the case as long as a is chosen sufficiently close to 1 in every iteration), the Newton system has a single unique solution. Note that one can choose different search directions than the one that arises from solving the Newton system presented here; this includes first applying a scaling transformation to the product of second-order cones, then forming and solving the Newton system that results, and finally applying the inverse scaling transformation. Alternate search directions are explained in section Additional Information D, but in the main text the basic search direction illustrated above is maintained, since in the numerical simulations the simple search direction gave equal or better results than more complex alternatives, and it enjoys the same theoretical guarantee of convergence.

5. Solving the Newton System

[0085] The Newton system formed by combining eqs. (19) and (22) is an L×L linear system of the form Gu=h, where L=2N+K+3. Classically this can be solved exactly a number of ways, the most straightforward being Gaussian elimination, which scales as $\mathcal{O}(L^3)$. Using Strassen-like tricks, this can be asymptotically accelerated to $\mathcal{O}(L^{\omega})$ where ω<2.38, although practically the runtime is closer to $\mathcal{O}(L^3)$. Meanwhile, the linear system can be approximately solved using a variety of iterative solvers, such as conjugate gradient descent or the randomized Kaczmarz method. The complexity of these approaches depends on the condition number of the Newton matrix. Section IV discusses quantum approaches to solving the Newton system.

[0086] It is noteworthy to distinguish methods that exactly solve the Newton system, and methods that solve it inexactly, because inexact solutions typically lead to infeasible intermediate points. As presented above, the Newton system in eqs. (19) and (22) can tolerate infeasible intermediate points; the main consequence is that the right-hand-side of eq. (19) becomes non-zero. As discussed in section IV, exact feasibility is difficult to maintain in quantum IPMs, since the Newton system cannot be solved exactly.

[0087] The inexact-feasible IPM(IF-IPM) is a workaround by which exact feasibility can be maintained despite an inexact linear system solver. For the IF-IPM, this disclosure assumes access to a basis for the null space of the feasibility constraint equations, that is, a linearly independent set of solutions to eq. (19) when the right-hand-side is zero. These basis vectors are arranged as the columns of a matrix B; since there are N+K+2 linear feasibility constraints and 2N+K+3 variables, the matrix B should have N+1 columns. In the case of portfolio optimization, a matrix B satisfying this criterion can be deduced by inspection, as discussed in section Additional Information C; however, this choice does not yield a B with orthogonal columns. Generating a B with orthonormal columns can be done by performing a QR decomposition of the matrix in eq. (19), which would incur a large one-time classical cost of $\mathcal{O}((N+K)^3)$ operations. Better asymptotic scaling for QR decomposition can be accomplished using fast matrix multiplication. In either case, since B is a basis for the null space of the constraint equations, there is a one-to-one correspondence between vectors $\Delta z \in \mathbb{R}^{N+1}$, and vectors that satisfy eq. (19) via the relation (Δx; Δy; Δτ; Δθ; Δs; Δκ)=BΔz. Thus, the Newton system can be reduced to

$$\left[\begin{pmatrix} S & 0 & 0 & 0 & X & 0 \\ 0 & 0 & \kappa & 0 & 0 & \tau \end{pmatrix} B\right] \Delta z = \begin{pmatrix} \sigma \mu e - Xs \\ \sigma \mu - \kappa \tau \end{pmatrix} \tag{23}$$

$$(\Delta x; \Delta y; \Delta \tau; \Delta \theta; \Delta s; \Delta \kappa) = B \Delta z. \tag{24}$$

The Newton system above can be solved by first computing Δz by inverting the quantity in brackets in the first line and applying it to the right-hand-side, and then computing (Δx; Δy; Δτ; Δθ; Δs; Δκ) by performing the multiplication BΔz. This matrix-vector product can be accomplished classically in $\mathcal{O}(N^2)$ operations. Note that matrix-matrix products where one of the matrices is an arrowhead matrix (S or X) can also be carried out in $\mathcal{O}(N^2)$ classical time, as the form of arrowhead matrices given in eq. (21) implies that the product can be computed by summing several matrix-vector products. Finally, note that since the second and fourth block columns of the first matrix in eq. (22) are zero, the second and fourth block rows of B (e.g., in eq. (C1)) can be completely omitted from the calculation.

[0088] Thus, there are three main choices for how to run the IPM when the solution to linear systems is inexact: first, by solving eqs. (19) and (22) directly and allowing intermediate solutions to be infeasible; second, by determining a matrix B by inspection as described in section Additional Information C and then solving eqs. (23) and (24); third, by determining a matrix B via QR decomposition and then solving eqs. (23) and (24). When the linear system is solved using a quantum algorithm, as discussed in section IV, this disclosure refers to the algorithm that results from each of these three options by II-QIPM, IF-QIPM, and IF-QIPM-QR, respectively. The pros and cons of each method are summarized in table II.

6. Neighborhood of the Central Path and Polynomial Convergence

[0089] Prior literature establishes that if sufficiently small steps are taken (i.e., if σ is sufficiently close to 1), then each intermediate point stays within a small neighborhood of the central path. This disclosure now reviews these conclusions.

For a vector $u=(u_0; \tilde{u}) \in \mathcal{Q}^k$, the following matrix is defined:

$$T_u = \begin{pmatrix} u_0 & \tilde{u}^{\top} \\ \tilde{u} & \sqrt{u_0^2 - \|\tilde{u}\|^2}\, I + \dfrac{\tilde{u}\tilde{u}^{\top}}{u_0 + \sqrt{u_0^2 - \|\tilde{u}\|^2}} \end{pmatrix}, \tag{25}$$

which, as for the arrowhead matrix, generalizes to the product of multiple cones by forming a block diagonal of matrices of the above form. This disclosure uses the following distance metric

$$d_F(x,\tau,s,\kappa)=\sqrt{2}\sqrt{\|T_x s-\mu(x,\tau,s,\kappa)e\|^2+(\tau\kappa-\mu(x,\tau,s,\kappa))^2}. \quad (26)$$

The distance metric induces a neighborhood $\mathcal{N}$, which includes both feasible and infeasible points, as well as the neighborhood $\mathcal{N}_F$, which includes only feasible points

$$\mathcal{N}(\gamma)=\{(x;y;\tau;\theta;s;\kappa): d_F(x,\tau,s,\kappa)\leq\gamma\mu(x,\tau,s,\kappa)\} \quad (27)$$

$$\mathcal{N}_F(\gamma)=\mathcal{N}(\gamma)\cap\mathcal{P}_F, \quad (28)$$

where $\mathcal{P}_F$ denotes the set of feasible points for the self-dual SOCP. Note that the vector $T_x s$ can be computed classically in $\mathcal{O}(N)$ time given access to the entries of x and s. Thus, whether or not a point lies in $\mathcal{N}(\gamma)$ can be determined in $\mathcal{O}(N)$ time.

[0090] So long as $0\leq\gamma\leq 1/3$ and $(x; y; \tau; \theta; s; \kappa)\in\mathcal{N}_F(\gamma)$, then the following may be true:

$$(x+\Delta x;y+\Delta y;\tau+\Delta\tau;\theta+\Delta\theta;s+\Delta s;\kappa+\Delta\kappa)\in\mathcal{N}_F(\Gamma), \quad (29)$$

where

$$\Gamma=\frac{4\left(\gamma^2+2(r+1)(1-\sigma)^2\right)}{(1-3\gamma)^2\sigma}. \quad (30)$$

TABLE II

Choices on which version of the Newton system to solve lead to different versions of the QIPM, even with the same underlying quantum subroutines.

	II-QIPM	IF-QIPM	IF-QIPM-QR
Newton system	Equations (19) and (22)	Equations (23) and (24)	Equations (23) and (24)
Size of Newton system (L)	2N + K + 3	N + 1	N + 1
Feasible intermediate points	No	Yes	Yes
Caveats	Theoretical convergence guarantee uses $\mathcal{O}(r^2)$ (rather than $\mathcal{O}((\sqrt{r}))$ iterations	Ill-conditioned null-space basis leads to large condition number of Newton system	Uses classical QR decomposition, which could dominate overall runtime

[0091] Thus, if $\Gamma\leq\gamma$, and assuming the Newton system is solved exactly, every intermediate point will lie in $\mathcal{N}_F(\gamma)$. This condition is met, for example, if $\gamma=1/10$ and $\sigma=1-(20\sqrt{2}\sqrt{(r+1)})^{-1}$. Since each iteration reduces the duality gap by a factor σ, the duality gap can be reduced to ϵ after roughly only $20\sqrt{2(r+1)}\ln(1/\epsilon)$ iterations. If the Newton system is solved inexactly, but such that feasibility is preserved (e.g., by solving inexactly for Δz and then multiplying by B, as described above), then an error δ on the vector $(x; \tau; s; \kappa)$ can be tolerated, and the resulting vector can still be within the neighborhood at each iteration.

[0092] On the other hand, if the Newton system is not solved exactly, then the resulting vector may not be feasible. Thus, the II-QIPM version of the QIPM does not enjoy the theoretical guarantee of convergence in $\mathcal{O}(\sqrt{r})$ iterations that the IF-QIPM and IF-QIPM-QR versions do (see table II). The best guarantees for the II-QIPM would imply convergence only after $\mathcal{O}(r^2)$ iterations. Nevertheless, it is unclear if a small amount of infeasibility makes a substantial difference in practice: multiple versions of the QIPM were simulated and similar overall performance was observed when intermediate solutions were allowed to be infeasible, despite an inferior theoretical guarantee of success. Thus, in sections V and VI, where the QIPM implementation, resource count, and numerical analysis are described, this disclosure focuses on the II-QIPM. This disclosure presents some of the results of the numerical simulations of the IF-QIPM and IF-QIPM-QR results in the section Additional Information.

IV. QUANTUM INTERIOR POINT METHODS (QIPM)

A. Introduction to QIPM

[0093] As discussed in section III, each iteration of an IPM SOCP solver involves forming and solving a linear system of equations that depends on the intermediate point at the current iteration. For classical IPM implementations for SOCP, the linear systems of equations are typically solved exactly; for example, the numerical SOCP solving package ECOS solves linear systems with a sparse LDL (Cholesky) factorization. For arbitrary dense systems, the runtime of solving an L×L system this way is $\mathcal{O}(L^3)$, but by exploiting sparsity the actual runtime in practice could be much faster, by an amount that is hard to assess. Alternatively, it would, in principle, be possible to employ classical iterative approximate linear system solvers such as conjugate gradient descent or the randomized Kaczmarz method. The choice of the linear system solver thereby determines the overall complexity of the IPM SOCP solver. An idea of QIPM is to use a quantum subroutine to solve the linear system of equations. Other steps of QIPMs may be classical and may remain the same as described in section III. As a quantum linear system solver (QLSS) does not solve the exact same mathematical problem as classical linear system solvers and, moreover, a QLSS uses coherent (quantum) access to the classical data as given by the entries of the relevant matrices, there are various additional tools (discussed herein) that allow embedded QLSS subroutines as a step of IPM SOCP solvers.

[0094] First, this disclosure discusses in section IV B the input and output model of QLSSs and present the complexity of state-of-the-art QLSSs. Then, section IV C provides constructions based on quantum random access memory (QRAM) to load classical data as input into a QLSS and discusses the complexity overhead arising from that step. Subsequently, section IV D presents so-called pure state quantum tomography that allows to convert the output of the QLSS into an estimate of the classical solution vector of the linear system of equations. Section IV E puts all the steps together and states the overall classical and quantum com-

plexities of using QLSSs as a subroutine in IPM SOCP solvers. The idea is to compare these costs to the complexities of classical IPM SOCP solvers and point out regimes where quantum methods can potentially scale better that any purely classical methods (e.g., in terms of the SOCP size N, the matrix condition number κ, etc.)

B. Quantum Linear System Solvers

[0095] For current purposes, a linear system of equations is given by a real invertible L×L matrix G together with a real vector $h=(h_1, \ldots, h_L)$, and one is looking to give an estimate of the unknown solution vector $u=(u_1, \ldots, u_L)$ defined by Gu=h. The (Frobenius) condition number is defined as

$$\kappa_F(G):=\|G\|_F\|G^{-1}\|, \tag{31}$$

where $\|\bullet\|_F$ denotes the Frobenius norm and $\|\bullet\|$ for a matrix argument denotes the spectral norm.

[0096] For this setting, the input to a QLSS is then comprised of: (i) a preparation unitary U_h that creates the $\ell:=\lceil \log L \rceil$ qubit quantum state

$$|h\rangle:=\|h\|^{-1}\cdot\Sigma_{i=1}{}^{L}h_i|i\rangle \quad via |h\rangle=U_h|0\rangle^{\otimes\ell}, \tag{32}$$

where $\|\bullet\|$ for a vector argument denotes the vector two-norm (standard Euclidean norm), (ii) a block encoding unitary U_G in the form

$$U_G:=\begin{pmatrix} \frac{G}{\|G\|_F} & \cdot \\ \cdot & \cdot \end{pmatrix} \tag{33}$$

[0097] on $\ell+\ell_G$ qubits for some $\ell_G\in\mathbb{N}$, and (iii) an approximation parameter $\varepsilon_{QLSP}\in(0,1]$. The quantum linear system problem (QLSP) is stated as follows: For a triple (G, h, ε_{QLSP}) as above, the goal is to create an ℓ-qubit quantum state $|\tilde{v}\rangle$ such that

$$\left\||\tilde{v}\rangle-|v\rangle\right\|\le\varepsilon_{QLSP} \quad \text{for } |v\rangle:=\frac{\sum_{i=1}^{L}u_i|i\rangle}{\left\|\sum_{i=1}^{L}u_i|i\rangle\right\|}, \tag{32}$$

defined by Gu=h with $u=(u_1, \ldots, u_L)$, by employing as few times as possible the unitary operators U_G, U_h, their inverses $U_G{}^{\dagger}$, $U_t{}^{\dagger}$, controlled versions of U_G, U_h, and additional quantum gates on potentially additional ancilla qubits. The state-of-the-art QLSS using the fewest calls to U_G, U_h and their variants, is based on ideas from discrete adiabatic evolution. This disclosure notes the following explicit complexities. In this formulation, the quantum state $|v\rangle$ corresponds to the normalized solution vector of the normalized linear system Gu=h. Thus, the state $|v\rangle$ does not carry information on the norm of the solution $\|u\|$. This norm is related to v by the relationship $\|u\|=\|h\|/\|Gv\|$.

[0098] Proposition 1. The QLSP for (G, h, ε_1) can be solved with a quantum algorithm on $\lceil \log_2(L) \rceil+4$ qubits for

$$\varepsilon_1\le C\cdot\frac{\kappa_F(G)}{Q}+O\left(\frac{\sqrt{\kappa_F(G)}}{Q}\right) \tag{35}$$

for some constant C≤15307 using $Q\ge\kappa_F(G)$ controlled queries to each of U_G and $U_G{}^{\dagger}$, and 2Q queries to each of U_h and $U_h{}^{\dagger}$, and constant quantum gate overhead. If G is positive semi-definite, then C≤5632 instead.

[0099] Note that a stronger version of above proposition works with the (regular) condition number $\kappa(G):=\|G\|\|G^{-1}\|$, but it uses a block-encoding of the form eq. (33) in which the normalization factor is $\|G\|$ rather than $\|G\|_F$. In the current case, the Frobenius version $\kappa_F(G)$ is used, since there may not a straightforward method to perform U_G with normalization factor $\|G\|_F$, described in section IV C. It is then sufficient to give upper bounds for the remaining $\kappa_F(G)$ to run the algorithm from proposition 1. In practice, such upper bounds are given by using appropriate heuristics (cf. section V on implementations).

[0100] Note that proposition 1 implies a solution to the QLSP in eq. (34) with an asymptotic query complexity of $\mathcal{O}(\kappa_F/\varepsilon_{QLSP})$ to U_G, U_h and their variants and under standard complexity-theoretic assumptions this is optimal in terms of the scaling $\mathcal{O}(\kappa)$, but not in terms of the scaling $\mathcal{O}(\varepsilon_{QLSP})$. To get to an improved $\mathcal{O}(\log(1/\varepsilon_{QLSP}))$ scaling, an eigenstate filtering method may be used that additionally invokes a quantum singular value transform based on a minimax polynomial. The following overall complexities are provided.

[0101] Proposition 2. The QLSP problem for (G, h, ε_2) can be solved with a quantum algorithm on $\lceil \log_2(L) \rceil+5$ qubits that produces a quantum state

$$\sqrt{p}|0^5\rangle|\tilde{v}\rangle+\sqrt{1-p}|\perp\rangle|\text{fail}\rangle \tag{36}$$

with $\langle 0^5|\perp\rangle|=0$ and success probability $p\ge\frac{1}{2}$. With that, the sought-after ε_2-approximate solution quantum state $|\tilde{v}\rangle$ can be prepared using Q+d controlled queries to each of U_G and $U_G{}^{\dagger}$, and 2Q+2d queries to each of U_h and $U_h{}^{\dagger}$, where

$$Q=2C\kappa_F(G)+\mathcal{O}(\sqrt{\kappa_F(G)}) \tag{37}$$

$$d=2\kappa_F(G)\ln(2/\varepsilon_2). \tag{38}$$

Here, C≤15307 is the same constant as in proposition 1.

[0102] This version of the algorithm basically uses proposition 1 with constant choice of $\varepsilon_1\le\frac{1}{4}$, and then uses eigenstate filtering to measure whether the final state is the correct solution state. On average the algorithm is repeated no more than twice to produce the desired state $|\tilde{v}\rangle$. The resulting scaling that proposition 2 implies for the QLSP problem in eq. (34) is $\mathcal{O}(\kappa\log(1/\varepsilon_{QLSP}))$, which under standard complexity-theoretic assumptions is optimal in both κ and ε_{QLSP}. In practice the $Q=2C\kappa_F(G)$ dominates over d and all other terms can be safely neglected for typical settings—even for finite scale analyses. Moreover, the constant C is typically an order of magnitude smaller than the estimates given; for positive semi-definite G the constant is estimated as 638. No direct estimates for general matrices G are available, but this disclosure henceforth assumes C=2000 for the numerical estimates. Additionally, note that for the eigenstate filtering step via QSVT, the minimax polynomial and its corresponding quantum signal processing angles have to be computed. This is done as part of classical pre-processing.

[0103] Note that the implementation of the QLSS in each of proposition 1 and proposition 2 assume perfect implementation of the underlying circuits, without additional gate synthesis errors. In practice, however, these circuits will not be implemented perfectly, and hence additional sources of error are later included (e.g., block-encoding error, imper-

fect rotation gates, etc.) that also contribute to ε_{QLSP}. These additional contributions are in section IV D, for example.

[0104] The following continues by laying out the additional classical and quantum resources used to employ QLSS for estimating in an end-to-end fashion the classical solution vector $v=(v_1, \ldots, v_1)$ instead of the quantum state $|v\rangle$.

C. Block-Encoding Via Quantum Random Access Memory (QRAM)

[0105] Many quantum algorithms (and in particular for the current use case), use coherent access to classical data for use in the algorithm. Block-encodings of matrices provide a commonly used access model for the classical data by encoding matrices into unitary operators, thereby providing oracular access to the data. As mentioned above, for a matrix $G\in\mathbb{R}^{L\times L}$, a unitary matrix U_G block-encodes G when the top-left block of U_G is proportional to G, i.e.

$$U_G := \begin{pmatrix} G/\alpha & \cdot \\ \cdot & \cdot \end{pmatrix}, \tag{39}$$

where $\alpha\geq\|G\|$ is a normalization constant, chosen as $\alpha=\|G\|_F$ for the use case. The other blocks in U_G are irrelevant, but they are encoded such that U_G is unitary. For current real matrices G are focused on, but the extension to complex matrices is straightforward. A block-encoding makes use of unitaries that implement (controlled) state preparation, as well as quantum random access memory (QRAM) data structures for loading the classical data. Specifically, QRAM is referred to as the quantum circuit that allows query access to classical data in superposition:

$$\sum_j \psi_j|j\rangle|0\rangle \xrightarrow{QRAM} \sum_j \psi_j|j\rangle|a_j\rangle, \tag{40}$$

where j is the address in superposition with amplitude ψ_j and $|a_j\rangle$ is the classical data loaded into a quantum state. There are several models of QRAM one can use that differ in the way in which the data is loaded. The two most notable QRAM models are the select-swap (SS) model, which is particularly efficient in terms of T-gate utilization, and the bucket-brigade (BB) model, which has reduced susceptibility to errors when operated on potentially faulty hardware.

[0106] The block-encoding unitary U_G acts on $\ell+\ell_G$ qubits, where $\ell=\lceil\log_2(L)\rceil$ and, in our construction, $\ell_G=\ell$. To build it, U_G may be formed as the product of a pair of controlled-state preparation unitaries U_L and U_R. Specifically,

$$U_G=U_R^{\dagger}U_L, \tag{41}$$

$$U_R:|0\rangle^{\otimes\ell}|j\rangle\mapsto|\psi_j\rangle|j\rangle \tag{42}$$

$$U_L:|0\rangle^{\otimes\ell}|k\rangle\mapsto|k\rangle|\phi_k\rangle \tag{43}$$

where the ℓ-qubit states $|\psi_j\rangle$ and $|\phi_k\rangle$ are determined from the matrix elements G_{jk} of G, as follows:

$$|\psi_j\rangle=\sum_k\frac{G_{jk}}{\|G_{j,\cdot}\|}|k\rangle \tag{44}$$

$$|\psi_k\rangle=\sum_j\frac{\|G_{j,\cdot}\|}{\|G\|_F}, \tag{45}$$

where G_j, denotes the jth row of G. That is, controlled on the second ℓ-qubit register in the state $|j\rangle$, U_R prepares the ℓ-qubit state $|\psi_j\rangle$ into the first ℓ-qubit register, and U_L performs the same operation for the states $|\phi_k\rangle$ modulo a swap of the two registers. Both U_L and U_R utilize an additional ℓ' QRAM ancilla qubits that begin and end in the state $|0\rangle$. These controlled-state preparation unitaries U_R and U_L are implemented by combining a QRAM-like data-loading step with a protocol for state preparation of ℓ-qubit states. There are several combinations of state preparation procedure and QRAM model one can choose with varying benefits and resource requirements. Reference arXiv:2206.03505 (hereafter "Clader") studies the resources used to implement these block-encodings and provides explicit circuits for their implementation. This disclosure simply imports the relevant resource estimates from that work in table III. In the current setting, the matrices to block encode are typically dense, which is why the general constructions from Clader are sufficient. For the current purposes, this disclosure works with the minimum depth circuits that achieve a T-gate depth of $\mathcal{O}(\log L)$, at the price of using a total number of $\mathcal{O}(L^2)$ many qubits for the data structure implementing the block encoding unitary U_G. Additionally, the ℓ-qubit unitary U_h defined by $|h\rangle=U_h|0\rangle^{\otimes\ell}$ corresponds to the special case of quantum state preparation and is directly treated by the methods outlined in Clader. The resources used to synthesize U_h up to error ε_h are also reported in table III.

TABLE III

Logical quantum resources used to block-encode (left column) and control-block-encode (right column) an $L\times L$ matrix G to precision $\varepsilon_G\in[0, 1]$, where $L=2^{\ell}$ is assumed. Here terms are suppressed doubly and triply logarithmic in L and $1/\varepsilon_G$ (see Clader).

Resource	Block Encoding	Controlled Block Encoding
# of qubits	$N_{Qbe}:=4L^2-3L+2\ell-1$	$N_{Qche}:=N_{Qbe}+L$
T-depth	$T_{Dbe}:=10\ell+24\log_2(1/\varepsilon_G)+44$	$T_{Dcbe}:=T_{Dbe}+4$
T-count	$T_{Cbe}:=(12\log_2(1/\varepsilon_G)+56)L^2-24L-12\log_2(1/\varepsilon_G)-32\ell-32$	$T_{Ccbe}:=T_{Cbe}+16(L-1)$

TABLE IV

Logical quantum resources used to prepare an arbitrary ℓ-qubit quantum state $|h\rangle$ from classical data (left column) and a single-qubit controlled version (right column) to precision $\varepsilon_h \in [0, 1]$. Here terms are suppressed doubly and triply logarithmic in L and $1/\varepsilon_h$ (see Clader). For a single-qubit control, there are no additional Clifford gates used, which can be observed by examining the state-preparation procedure in Clader and noting that the state $|0\rangle^{\otimes \ell}|\rangle + |1\rangle|\psi\rangle$ can be prepared with minor modifications to the procedure that prepares $|\psi\rangle$. First, use the "flag" qubits to control both the angle loading and unloading steps (rather than just the unloading steps), and second, control every flip of the flag qubits in that procedure with the first single-qubit control, thus turning NOT gates into CNOT gates, which are also Clifford. When the control is ON, the procedure works as before, and when the control is OFF, none of the qubits leave the $|0\rangle$ state.

Resource	State Preparation	Controlled State Preparation
# of qubits	N_{Qsp}: $=4L+\ell-6$	N_{Qcsp}: $=N_{Qsp}+1$
T-depth	T_{Dsp}: $=3\ell+12\log_2(1/\varepsilon_h)+24$	T_{Dcsp}: $=T_{Dsp}$
T-count	T_{Csp}: $=(12\log_2(1/\varepsilon_h)+40)L-12\log_2(1/\varepsilon_h)-16\ell-40$	T_{Ccsp}: $=T_{Csp}$

[0107] The minimum-depth block encodings of Clader also incur some classical costs. Specifically, the quoted depth values are only achievable assuming a number of angles have been classically pre-computed and for each angle a gate sequence of single-qubit Clifford and T gates that synthesizes a single-qubit rotation by that angle up to small error. Calculating one of the angles can be done by summing a subset of the entries of G and computing an arcsin. Meanwhile, circuit synthesis uses applying a version of the Solovay-Kitaev algorithm. For the block-encoding procedure, L(L−1) angles and their corresponding gate sequences are computed, which uses a total runtime of L^2 poly $\log(1/\varepsilon_G)$, although this computation is amenable to parallelization. For the state preparation procedure, L−1 angles and their sequences are used.

D. Quantum State Tomography

[0108] This disclosure described how a quantum state $|\tilde{v}\rangle$ approximating the (real-valued) solution $|v\rangle$ of a linear system up to precision ε_{QLSP} can be produced. As mentioned in section IV B, in the actual circuit implementation, the approximation error ε_{QLSP} accounts for both the inherent error from eigenstate filtering captured in proposition 2 as well as additional gate synthesis error arising from imperfect implementation of block-encoding unitaries and single-qubit rotations. The next step is to approximately read out the amplitudes of $|\tilde{v}\rangle$ into classical form. To start out, this disclosure proves the following proposition, which communicates how many copies of a quantum state are used to provide a good enough classical description of it, up to a phase on each amplitude.

Proposition 3. Let $0<\varepsilon,\delta<1$ and $|\psi\rangle=\Sigma_{j\in[L]}\alpha_j|j\rangle$ be a quantum state. Then,

$$\frac{5+\sqrt{21}}{3\varepsilon^2}$$

$\ln(2L/\delta)<3.1942\varepsilon^{-2}\ln(2L/\delta)$ measurements of $|\psi\rangle$ in the computational basis suffice to learn an ε-ℓ_∞-norm estimate $|\tilde{\alpha}|$ of $|\alpha|$, with success probability at least $1-\delta$.

[0109] The proof is provided in the section Additional Information B1. Recall that proposition 2 gives a unitary U such that

$$U|0^5 0^\ell\rangle=\sqrt{p}|0^5\rangle|\tilde{v}\rangle+\sqrt{1-p}|\perp\rangle|\text{fail}\rangle \tag{46}$$

with $|\tilde{v}\rangle:=\Sigma_{i=1}^{N}\tilde{v}_i|i\rangle$, $\langle 0^5|\perp\rangle=0$, and $p\geq 1/2$. The vector $\tilde{v}$ may have complex coefficients, but it approximates a real vector v up to some error ε_{QLSP} in ℓ_2 norm. A goal is to obtain an estimate $\tilde{v}'=(v_1',\ldots,v_N')$ such that

$$\|v-\tilde{v}'\|\leq\xi \text{ for an error parameter } \xi\in[0,1]. \tag{47}$$

where ξ captures other (e.g., all) sources of error. Proposition 3 is not quite sufficient because it only gives us an estimate of the absolute value of V. However, the following procedure is sufficient:

[0110] 1. Create $k=57.5\ L\ \ln(\delta L/\delta)/(\varepsilon^2(1-\varepsilon^2/4))$ many copies of the quantum state $U|0^{5+\ell}\rangle=\sqrt{p}|0^5\rangle|\tilde{v}\rangle+\sqrt{1-p}|\perp\rangle|\text{fail}\rangle$, and measure them all in the computational basis to give empirical estimates $\{p_i\}_{i=1}^{L}$ of the probabilities $p|\tilde{v}_i|^2$.

[0111] 2. Using controlled applications of U, create $k=57.5\ L\ \ln(6L/\delta)/(\varepsilon^2(1-\varepsilon^2/4))$ copies of

$$2^{-1/2}|0^5\rangle|0\rangle\sqrt{p}\,|\tilde{v}\rangle+2^{-1/2}|0^5\rangle|1\rangle\sum_{i=1}^{L}\sqrt{p_i'}\,|i\rangle+|\perp'\rangle|\text{fail}'\rangle, \tag{48}$$

which by applying a Hadamard can be mapped to

$$|0^5\rangle|0\rangle\frac{\sqrt{p}\,|\tilde{v}\rangle+\sum_{i=1}^{L}\sqrt{p_i'}\,|i\rangle}{2}+|0^5\rangle|1\rangle\frac{\sqrt{p}\,|\tilde{v}\rangle-\sum_{i=1}^{L}\sqrt{p_i'}\,|i\rangle}{2}+|\perp'\rangle|\text{fail}''\rangle. \tag{49}$$

Here $|\perp'\rangle$ is an arbitrary state orthogonal to $|0^5\rangle$ and $|\text{fail}'\rangle$ and $|\text{fail}''\rangle$ are arbitrary unnormalized states. The quantities $\sqrt{p_i'}$ are (possibly complex) amplitudes that satisfy $|\sqrt{p_i'}-\sqrt{p_i}|\leq\varepsilon_{tsp}$ for all i; they arise because the state $\Sigma_{i=1}^{L}\sqrt{p_i'}|i\rangle$ can only be prepared up to some error. Next, measure this state in the computational basis, denoting the measurement count of the result $0^6 i$ as k_i^+ and the result $0^5\ 1 i$ as k_i^-.

[0112] 3. Define

$$a_i^+=\min\left(\sqrt{p_i},\frac{k_i^+-k_i^-}{\sqrt{p_i}}\right) \tag{50}$$

$$a_i^-=\max\left(-\sqrt{p_i},\frac{k_i^+-k_i^-}{\sqrt{p_i}}\right) \tag{51}$$

$$\text{and let } \tilde{a}_i=\begin{cases}0 & \text{if } \sqrt{p_i}\leq\dfrac{2}{3\sqrt{2L}}\varepsilon\sqrt{1-\dfrac{\varepsilon^2}{4}}+\varepsilon_{tsp}\\ a_i^+ & \text{if } \tilde{a}_i\neq 0 \text{ and } k_i^+\geq k_i^-\\ a_i^- & \text{if } \tilde{a}_i\neq 0 \text{ and } k_i^+<k_i^-\end{cases}. \tag{52}$$

Output the estimate

$$|\tilde{v}'\rangle=\Sigma_{i=1}^{L}\tilde{a}_i|i\rangle/\sqrt{\sum_{i=1}^{L}\tilde{a}_i^2}.$$

Proposition 4. Suppose that $\|\tilde{v}-v\| \leq \varepsilon_{QLSP}$ and that v is a real-valued vector. Let ε and ε_{tsp} be constants that satisfy $\varepsilon+\sqrt{2L}\varepsilon_{tsp}+\sqrt{2}\varepsilon_{QLSP} \leq 1/2$. Then the algorithm above outputs an estimate ṽ' such that $\|\tilde{v}'-v\| < \varepsilon+1.58\sqrt{L}\varepsilon_{tsp}+1.58\varepsilon_{QLSP}$ with probability 1−δ.

[0113] The proof is provided in the section Additional Information B1. The statement is used to bound the total error parameter ξ by the quantity $\varepsilon+1.58\sqrt{L}\varepsilon_{tsp}+1.58\varepsilon_{QLSP}$. Proposition 4, together with proposition 2, produces with high probability an $\mathcal{O}(\varepsilon)$ good estimate ṽ' of v by using $\mathcal{O}(L\ln(L)/\varepsilon^2)$ many samples. If a goal is to resolve the initial linear system Gu=h, then the vector ṽ' produced as in Section IV D as an estimate for the normalized vector v=u/‖u‖, gives an estimate for u via

$$\tilde{u} := \tilde{v}' \cdot \frac{\|h\|}{\|G\tilde{v}'\|},$$

for which the following is found:

$$\|u-\tilde{u}\| \leq \|v-\tilde{v}'\| \cdot (1+\kappa(G)) \cdot \frac{\|h\|}{\|G\tilde{v}'\|}.$$

[0114] There are other methods in the literature that allow to perform pure state quantum tomography with comparable query complexities, but this disclosure favors the above method because of its computational simplicity, and the fact that it does not require solving any potentially costly additional optimization problems. The sample complexity has been improved to $\mathcal{O}(L\ln(L)/\varepsilon^2)$, which comes at the cost of more complicated quantum circuits and higher constant overheads (See Reference arXiv:2206.03505). It would be interesting to work out the more involved finite complexity of this result, and further comment on the potential impact of this are provided in section VII.

E. Asymptotic Quantum Complexity

[0115] Putting things together, the steps of our QLSS for given real L×L matrix G and real vector h of size L may be:

[0116] 1. Construct the circuits that implement the block-encoding unitaries U_G and U_h up to error ε_G and ε_h via quantum state preparation and QRAM, which involves a classical pre-processing cost scaling as $L^2\text{poly}\log(1/\varepsilon_{G,h})$. The quantum resources used are described in table III. The T-gate depth (referred to as "time complexity") is $\mathcal{O}(\log L)$ and the total T-gate count is $\mathcal{O}(L^2)$.

[0117] 2. Employ the QLSS unitary from proposition 2 to approximately solve the corresponding QLSP, leading to the quantum state $|\tilde{v}\rangle$. The query complexity to U_G, U_h, their controlled versions, and their inverses, is $\mathcal{O}(\kappa_F(G)\log(1/\varepsilon))$. The number of qubits useed is $\lceil \log L \rceil+5$.

[0118] 3. Repeat the previous step $\mathcal{O}(L\ln(L/\delta)\varepsilon^{-2})$ many times to implement the pure state quantum tomography scheme from section IV D, which also includes the use of an $\mathcal{O}(L)$ qubit QRAM structure, and one ancilla qubit. Tomography leads to the sought-after classical vector estimate ṽ' with $\|\tilde{v}'-\| \leq \varepsilon$.

[0119] The QLSS can then be used for each iteration of an IPM SOCP solver, which involves forming and solving a linear system of equations, resulting in the QIPM SOCP solver. This disclosure provides the quantum circuits used to implement the solver in the section IV D.

F. Quantum Circuits

[0120] The following are the quantum circuits used for the QLSS of proposition 1. The QLSS includes applying a unitary U[s] for many different values of s, where U[s] is a block-encoding of a certain Hamiltonian related to G and h, as specified below. The unitary acts on $4+\ell+\ell_G$ total qubits, where the final ℓ_G qubits are ancillas associated with U_G. The four single-qubit registers are referred to with labels a_1, a_2, a_3, a_4, the ℓ-qubit register with label L, and the ℓ_G-qubit register with label ℓ_G. These labels are used as subscripts on bras, kets, and operators to clarify the register to which they apply. The circuit for U[s] is depicted in FIG. **1**. Specifically, the unitary U[s] is a block-encoding of the $(2+\ell)$-qubit Hamiltonian $c(s)\cdot\mathcal{H}[s]:=(1-f(s))\mathcal{H}_0+f(s)\mathcal{H}_1$ on registers a_4a_1L, where c(s) is a normalization factor (defined later in eq. (60)),

$$\mathcal{H}_0 := \begin{pmatrix} 0 & 0 & I_L-|h\rangle\langle h|_L & 0 \\ 0 & 0 & 0 & -I_L \\ I_L-|h\rangle\langle h|_L & 0 & 0 & 0 \\ 0 & -I_L & 0 & 0 \end{pmatrix} \tag{53}$$

$$\text{and } \mathcal{H}_1 := \begin{pmatrix} 0 & 0 & 0 & G \\ 0 & 0 & G^\dagger(I_L-|h\rangle\langle h|_L) & 0 \\ 0 & (I_L-|h\rangle\langle h|_L)G & 0 & 0 \\ G^\dagger & 0 & 0 & 0 \end{pmatrix}, \tag{54}$$

and where I_L denotes the identity operation on subsystem L, and the four rows and columns correspond to the sectors with qubits a_4, a_1 set to (0,0), (0,1), (1,0), (1,1). FIG. **1** features the expressions:

$$CR^0(s) := |0\rangle\langle 0|_{a_4} \otimes R(s)_{a_2} + |1\rangle\langle 1|_{a_4} \otimes H_{a_2} \tag{55}$$

$$CR^1(s) := |1\rangle\langle 1|_{a_4} \otimes R(s)_{a_2} + |0\rangle\langle 0|_{a_4} \otimes H_{a_2} \tag{56}$$

$$V_G := |0\rangle\langle 0|_{a_2} \otimes Z_{a_1} \otimes I_{L\ell_G} + |1\rangle\langle 1|_{a_2} \otimes \begin{pmatrix} 0 & U_G \\ U_G^\dagger & 0 \end{pmatrix}_{a_1 L\ell_G}, \tag{57}$$

where H denotes the single-qubit Hadamard gate, and R(s) is given by

$$R(s) := \frac{1}{\sqrt{(1-f(s))^2+f(s)^2}} \begin{pmatrix} 1-f(s) & f(s) \\ f(s) & -(1-f(s)) \end{pmatrix} \tag{58}$$

$$f(s) := \frac{\kappa_F(G)}{\kappa_F(G)-1} \cdot \left(1-\left(1+s\left(\sqrt{\kappa_F(G)}-1\right)\right)^{-2}\right) \tag{59}$$

The normalization factor of R(s) above combines with a factor of $1/\sqrt{2}$ introduced by the Hadamard gate to give an overall normalization factor for $\mathcal{H}(s)$ of

$$c(s)=(2((1-f(s))^2+f(s)^2))^{-1/2} \in [2^{-1/2},1] \tag{60}$$

and scheduling function $f(s)$ with $f(0)=0$ and $f(1)=1$. Note the self-inverse property $U[s]^2=1\ \forall s\in[0,1]$. The overall quantum circuit U for the quantum algorithm of proposition 1 is then given as:

$$U := \prod_{\square=1}^{Q} P[1-j/Q] \tag{61}$$

with the walk operator

$$P[s] := WU[s],$$

where W is the operator that acts as identity on registers a_4a_1L (which host the Hamiltonian $\mathcal{H}[s]$) while performing the reflection $(2|0|_{a_2a_3\ell_G}\rangle 0\langle -I_{a_2a_3\ell_G})$ on the remaining qubits. The unitary U makes Q controlled queries to each of U_G and $U_G^\dagger$, and 2Q queries to each of U_h and $U_h^\dagger$, and it has constant quantum gate overhead.

[0121] FIG. **1** is a diagram of an example quantum circuit that is a main component of the example quantum circuit for proposition 1, enacting the unitary U[s] on registers $a_3a_4a_2a_1L\ell_G$ of the scaled Hamiltonian $c(s)\cdot\mathcal{H}[s]$, where $\mathcal{H}[s]=(1-f(s))\mathcal{H}_0+f(s)\mathcal{H}_1$, on registers a_4a_1L. The quantum gates and functions are defined in eqs. (53) to (59) except for sub-circuit U_{Qh}, which is depicted in FIG. **4**. The unitary U[s] is then used in eq. (61) to define the overall quantum circuit U for proposition 1.

[0122] Next, give the remaining QSVT eigenstate filtering quantum circuit for the refined quantum linear system solver of proposition 2. The null space of $c(1)\cdot\mathcal{H}[1]$ is of interest, which has ground-state energy equal to zero and spectral gap at least $c(1)\kappa_F^{-1}(G)=(\sqrt{2}\kappa_F)^{-1}$. As such, employ the Chebyshev minimax polynomial:

$$R_l\left(x,\kappa_F^{-1}(G)\right) := \frac{T_l\left(-1+2\frac{x^2-\kappa_F^{-2}(G)/2}{1-\kappa_F^{-2}(G)/2}\right)}{T_l\left(-1+2\frac{-\kappa_F^{-2}(G)/2}{1-\kappa_F^{-2}(G)/2}\right)}, \tag{62}$$

where $T_l(\bullet)$ is l-th Chebyshev polynomial of the first kind, as part of the corresponding QSVT quantum circuit. R_l has even degree d equal to

$$d := 2l = 2\lceil \kappa_F(G)\ln(2/\varepsilon_{qsp})\rceil \text{ for some } \varepsilon_{qsp}\in(0,1] \tag{63}$$

where ε_{qsp} is the precision to which R_l approximates the optimal filter operator. The QSP subscript stands for "quantum signal processing."

[0123] The circuit for the eigenstate filtering step is depicted in FIG. **2**. To implement it, one may classically pre-compute the corresponding QSP angles $\{\phi_1,\ldots,\phi_d\}$. The query complexity to the block encoding U[1] is given by d, the additional gate overhead is as in FIG. **2**, and the total number of qubits is $1+4+\ell$. Finally, using the overall quantum circuit U from proposition 1 with constant approximation parameter $\varepsilon_1=1/4$ therein (to produce an input state to the quantum circuit of FIG. **2**), gives the overall quantum circuit of the QLSS from proposition 2, which then solves the QLSP to error $\varepsilon_2=\varepsilon_{qsp}$.

[0124] FIG. **2** is a diagram of an example Quantum singular value transform (QSVT) circuit, acting on the block-encoding U[1] of $\mathcal{H}(1)=\mathcal{H}_1/\sqrt{2}$, as defined in eq. (54). The circuit features one additional ancilla qubit and depends on the classically precomputed rotation angles $\{\phi_1,\ldots,\phi_d\}$.

[0125] A quantum tomography routine may also include performing controlled versions of the above circuits as described in eq. (49) and illustrated in FIG. **3** (which replaces FIG. **1**). More specifically, FIG. **3** is a diagram of an example-controlled version of the quantum circuit in FIG. **1**, controlled on qubit c. Note that not all gates need to be controlled on c, as their inverses follows in the circuit. The controlled circuits can be accomplished by modifications to the circuits in FIGS. **1** and **2** as follows.

[0126] Any QSVT circuit can be made controlled by simply controlling the application of the z rotation gates, since the rest of the circuit contains only symmetric applications of unitary gates and their inverses. Thus, a controlled version of FIG. **2** may be created by simply performing controlled-σ_z rotations, which uses two CNOT gates and an extra single qubit σ_z rotation gate.

[0127] Controlling the linear system portion is not enough to implement eq. (49). This may (e.g., must) be followed up with a controlled state-preparation routine, controlled on the value of the qubit c being in the $|1\rangle$ state. The resource analysis for controlled state-preparation is reported in Ref. Clader. The resource counts are reported here in table IV.

V. IPM IMPLEMENTATION AND RESOURCE ESTIMATES FOR PO

[0128] The previous section reviewed the ingredients used to implement the QIPM, namely, QLSS, block-encoding, and tomography. Here, combine those ingredients to describe how the QIPM is actually implemented, making several observations that go beyond prior literature. Also perform a full resource analysis of the entire protocol and report resources used to run the algorithm.

A. QIPM Loop and Pseudo-Code

[0129] A QIPM is formed from an IPM by performing the step of solving a linear system with a quantum algorithm; the rest of the steps are classical. Example Algorithm 1 presents pseudocode for the interior point method where the single quantum subroutine—approximately solving a linear system—appears in blue text. The input to Algorithm 1 is an SOCP instance with N variables, K linear constraints, and r second-order cone constraints, along with a tolerance parameter $\in$. Here note that $K=\mathcal{O}(N)$ in the case of the formulation of the PO problem simulated in section VI. The output of the QIPM is a vector x that is $\mathcal{O}(\in)$ close to feasible, and $\mathcal{O}(\in)$ close to optimal.

[0130] A description of the QIPM algorithm is provided herein along with the following new observations:

[0131] Classical costs: The IPM uses $\mathcal{O}(\sqrt{r}\log(1/\in))$ iterations. In the classical case, when solving the PO problem via SOCP with an IPM, the cost of an iteration is dominated by the time used to solve a linear system of size L×L, which is $\mathcal{O}(N^3)$ if done via Gaussian elimination, since $L\sim\mathcal{O}(N)$ in the PO problem. In the quantum case, this step is performed quantumly. However, even in the quantum case, some classical costs are incurred: the left-hand and right-hand sides of the Newton system in eq. (19) and eq. (22) are classically computed to load this classical data into quantum

circuits that perform the QLSS and tomography to gain a classical estimate of the solution to the linear system. In particular, constructing the linear system includes classical matrix-vector multiplication to compute the residuals on the right-hand-side of the Newton system in eq. (19). If the SOCP constraint matrix A is $\mathcal{O}(N)\times N$ and the number of cones $r=\mathcal{O}(N)$, then this classical matrix-vector multiplication takes $\mathcal{O}(N^2)$ time in each of the $\mathcal{O}(\sqrt{N})$ iterations. Thus, the QIPM uses at least $\mathcal{O}(N^{2.5})$ classical time. Additionally, in the resource counts the minimal depth block-encoding circuits from Reference Clader are used, which use N^2poly $\log(1/\epsilon)$ classical time per iteration (although this can be parallelized) to compute angles and corresponding gate sequences to precision ε. These classical costs limit the maximum possible speedup of the QIPM over the classical IPM, but if the quantum subroutine is sufficiently fast that classical matrix-vector multiplication and angle computation is the bottleneck step, then this is a good signal for the utility of the QIPM.

[0132] Preconditioning: Since the runtime of the QLSS depends on the condition number of the matrix G that appears in the linear system Gu=h, it is worth examining preconditioning techniques for reducing the condition number. In the implementation proposed, a simple form of preconditioning may be preformed. Let D be a diagonal matrix where entry D_{ii} is equal to the norm of row i of the matrix G. Instead of solving the linear system Gu=h, solve the equivalent system $(D^{-1}G)u=D^{-1}h$. Note that $D^{-1}G$ and $D^{-1}h$ can each be classically computed in $\mathcal{O}(N^2)$ time, roughly equal to the time used to compute h in the first place (see previous bullet), so this step is unlikely to be a bottleneck in the algorithm. In the numerical experiments herein, the condition number of $D^{-1}G$ is typically more than an order of magnitude smaller than G, and sometimes several orders of magnitude (see FIG. **9** in section VI).

[0133] Norm of linear system and step length: As discussed in section IV B, QLSSs produce a normalized state $|u\rangle$, where u is the solution to Gu=h, and quantum state tomography on $|u\rangle$ can only reveal the direction of the solution u and not its norm. The norm can be estimated separately with a comparable amount of resources, but in the context of QIPMs, it is not necessary to learn the norm of the solution. If the direction of the solution is known, the amount by which to update the vector in that direction can be determined classically in $\mathcal{O}(N)$ time as follows. If $(\Delta x;\Delta y;\Delta\tau;\Delta\theta;\Delta s;\Delta\kappa)$ is the normalized solution to the Newton linear system in eqs. (19) and (22), then the amount to step in that direction is equal to

$$\frac{\mu(x,\tau,s,\kappa)(1-\sigma)(r+1)}{-(\Delta x)^T s-(\Delta s)^T x-(\Delta\kappa)\tau-(\Delta\tau)\kappa}. \tag{64}$$

This expression is chosen such that the duality gap of the new point is (e.g., exactly) a factor of σ smaller than the old point, up to deviations that are second order in the step length. Note that if the old point is feasible and the solution to the linear system is exact, the second and higher order contributions vanish anyway.

[0134] Adaptive tomographic precision and neighborhood detection: In conventional work, the choice of tomography precision parameter is determined by a formula that aimed to guarantee staying within the neighborhood of the central path under a worst-case outcome. However, since determining whether a point is within the neighborhood of the central path can be done in classical $\mathcal{O}(N)$ time (see section III C 6), the tomography precision parameter ξ herein may instead be determined adaptively for optimal results. For example: start with ξ=½, solve the linear system to precision ξ and check if the resulting point is within the neighborhood of the central path. If yes, continue to the next iteration; if no, repeat the tomography with ξ←ξ/2. Since the complexity of tomography is $\mathcal{O}(1/\xi^2)$, the cost of this adaptive scheme is proportional to a geometric series $4+16+64+\ldots+\mathcal{O}(1/\xi^2)$ of which the final term makes up most of the cost (accordingly, for simplicity, the resource calculation may only account for the final term). This cost could be much lower than the theoretical value if the typical errors are not as adverse for the IPM as a worst-case error of the same size.

[0135] The pseudocode in Algorithm 1 illustrates the "infeasible" version of the example algorithm (II-QIPM from table II). The numbers on the left refer to steps of Algorithm 1. Text in Algorithm 1 bookended by "/*" and "*/" are comments on one or more of the steps. To implement the feasible versions (IF-QIPM and IF-QIPM-QR), minor modifications are made to reflect the process described in section III.

Algorithm 1: Quantum Interior Point Method

Input: SOCP instance (A, b, c), list of cone sizes $(N_1,\cdots,N_r)$ and tolerance ϵ
Output: Vector x that optimizes objective function (eq. (5)) to precision ϵ
/* For portfolio optimization, A, b, c are given in eq. (10). First n entries of x give optimal stock weights. */
1 $(x;y;\tau;\theta;s;\kappa)\leftarrow(e;0;1;1;e;1)$ /* initialize on central path */
2 $\mu\leftarrow 1,\ \sigma\leftarrow 1-\frac{1}{20\sqrt{2}}\frac{1}{\sqrt{r}},\ \gamma\leftarrow 1/10$ /* set parameters */
3 while $\mu\geq\epsilon$: /* Follow central path until duality gap less than ϵ */

-continued

Algorithm 1: Quantum Interior Point Method

$G \leftarrow \begin{pmatrix} 0 & A^\top & -c & \overline{c} & I & 0 \\ -A & 0 & b & -\overline{b} & 0 & 0 \\ c^\top & -b^\top & 0 & -\overline{z} & 0 & 1 \\ -\overline{c}^\top & \overline{b}^\top & \overline{z} & 0 & 0 & 0 \\ S & 0 & 0 & 0 & X & 0 \\ 0 & 0 & \kappa & 0 & 0 & \tau \end{pmatrix}$ /* from eqs. (19) and (22) */

$h \leftarrow \begin{pmatrix} -A^\top y + c\tau - \overline{c}\theta - s \\ Ax - b\tau + \overline{b}\theta \\ -c^\top x + b^\top y + \overline{z}\theta \\ \overline{c}^\top x - \overline{b}^\top y - \overline{z}\tau \\ \sigma\mu e - X S e \\ \sigma\mu - \kappa\tau \end{pmatrix}$ /* mat .-vec. mult. performed classically */

for j = 1, ... , L: /* preconditioning via row normalization */

$g \leftarrow \sqrt{\sum_k |G_{jk}|^2}$ /* norm of jth row of G */

$h_j \leftarrow h_j / g$

for $k = 1, \ldots, L$:

$|G_{jk} \leftarrow G_{jk}/g$

Classically compute L^2 angles and gate decompositions to perform block-encoding of G and state-preparation of |h>

$\xi \leftarrow 1$

repeat /* try smaller and smaller ξ until central path is found */

$\xi \leftarrow \xi/2$

$(\Delta x; \Delta y; \Delta\tau; \Delta\theta; \Delta s; \Delta\kappa) \leftarrow$ ApprSolve(G, h, ξ)

$$\text{(step length)} \leftarrow \frac{\mu(\sigma-1)(r+1)}{(\Delta x)^\top s + (\Delta s)^\top x + (\Delta\kappa)\tau + (\Delta\tau)\kappa}$$

$(x'; y'; \tau'; \theta'; s'; \kappa') \leftarrow (x; y; \tau; \theta; s; \kappa) + (\text{step length}) \cdot (\Delta x; \Delta y; \Delta\tau; \Delta\theta; \Delta s; \Delta\kappa)$

until $(x'; y'; \tau'; \theta'; s'; \kappa') \in N(\gamma)$

$(x; y; \tau; \theta; s; \kappa) \leftarrow (x'; y'; \tau'; \theta'; s'; \kappa')$

$\mu \leftarrow \sigma\mu$

return x/τ

def ApprSolve (G, h, ξ):

$L \leftarrow 2N + K + 3$

$\delta \leftarrow 0.1$

$\varepsilon \leftarrow 0.9\xi$

$k \leftarrow 57.5 L \ln(6L/\delta)/(\varepsilon^2(1-\varepsilon^2/4))$

Run tomography as described in section IV D using k applications and k controlled-applications of the QLSS algorithm on the system (G, h)

return Vector $\tilde{v}'$ for which $\|\tilde{v}'\| = 1$ and $\|\tilde{v}' - v\| \le \xi$ with probability at least $1-\delta$, where $v \propto |G^{-1}h\rangle$

B. End-to-End Quantum Resource Estimates

[0136] The QIPM described in the pseudocode takes $20\sqrt{2}\sqrt{r}\ln(\epsilon^{-1})$ iterations to reduce the duality gap to ϵ, where r is the number of second-order cone constraints. In the case of the portfolio optimization problem, r=3n+1, where n is the number of stocks in the portfolio. Choosing the constant pre-factor to be $20\sqrt{2}$ allows us to utilize theoretical guarantees of convergence (modulo the issue of infeasibility discussed in section III C 5); however, it would not be surprising if additional optimization of the parameters or heuristic changes to the implementation of the algorithm (e.g. adaptive step size during each iteration) would lead to constant-factor speedups in the number of iterations. Since the number of iterations would be the same for both the quantum and classical IPM, these sorts of improvements would not impact the performance of the QIPM relative to its classical counterpart.

1. Quantum Circuit Compilation and Resource Estimate for Quantum Circuits Appearing within QIPM

[0137] The QIPM includes repeatedly performing a quantum circuit associated with the QLSS and measuring in the computational basis. The costs of each of these individual quantum circuits is accounted for herein. There are two kinds of circuits that are used: first, the circuit that creates the output of the QLSS subroutine, given by the state in eq. (36), and second, the circuit that creates the state used to determine the signs of the amplitudes during the tomography subroutine corresponding to a controlled-QLSS subroutine, given in eq. (49).

[0138] To simplify the analysis, first compile the circuits from the previous section into a primitive gateset that includes Toffoli gates (and multi-controlled versions of

them), rotation gates, block-encoding unitaries, state-preparation and controlled state-preparation unitaries. This compilation allows combining previous in-depth resource analysis for these primitive routines (See Ref. Clader) with the additional circuits shown here.

[0139] From left to right in the U[s] circuit shown in FIG. **1**, the circuits for U_{Qh}, $CR^0(s)$ (and equivalently $CR^1(s)$), and V_G in FIGS. **4** to **6**, respectively. In addition to these circuits, controlled versions of them may be performed within the tomography routine to estimate the sign of the amplitudes. The controlled-U[s] gate is given in FIG. **3**. The implementation of the controlled versions of $CR^0(s)$ (and equivalently $CR^1(s)$), and V_G are also depicted in FIGS. **5** and **6**, respectively.

[0140] With these decompositions in place, table V reports the resources used to perform each of the two kinds of quantum circuits involved in the QIPM (which are each performed many times over the course of the whole algorithm). The resource quantities are reported in terms of the number of calls Q to the block-encoding (which scales linearly with the condition number), as well as the controlled-block-encoding and state-preparation resources given previously in tables III and IV. The expressions also depend on various error parameters which must be specified to obtain a concrete numerical value.

[0141] In section VI, after observing empirical scaling of certain algorithmic parameters, error parameters are described to arrive at a concrete number for a specific problem size.

2. Resource Estimate for Producing Classical Approximation to Linear System Solution

[0142] The resource estimates described above capture the quantum resources used for a single coherent quantum circuit that appears during the algorithm. The output of this quantum circuit is a quantum state, but the QIPM includes a classical estimate of the amplitudes of this quantum state. This classical estimate is produced through tomography, as described in section IV D, by performing $k=57.5\ L\ \ln(6L/\delta)/(\varepsilon^2(1-\varepsilon^2/4))$ repetitions each of the QLSS and controlled-QLSS circuits, where ε is the desired tomography precision and δ is the probability that the tomography succeeds. In the example implementation given in Algorithm 1, fix $\delta=0.1$. Thus, to estimate the quantum resources of a single iteration of the QIPM, the previous resource estimates reported in table V should each be multiplied by k. With P processors large enough to prepare the output of the QLSS, these k copies may be prepared in k/P parallel steps, saving a factor of P in the runtime at the expense of a factor of P additional space. The resources and scaling estimates do not account for any parallelization, and completely serial execution and runtime is assumed.

[0143] After multiplication by k, these expressions give the quantum resources used to perform the single quantum line of the QIPM, ApprSolve. This subroutine has both classical input and output and can thus be compared to classical approaches for approximately solving linear systems.

[0144] FIG. **4** is a diagram illustrating an example decomposition of the U_{Qh} gate (shown, e.g., in FIG. **1**) into a state-preparation unitary U_h and multi-controlled-Toffoli gates. The reflection operator W is given by $W'=I_{a_1L}-2|1\rangle\langle 1|_{a_1}\otimes|0\rangle\langle 0|_L$. Not pictured are additional ancillas that begin and end in $|0\rangle$ and are utilized to implement the unitary U_h in shallower depth.

[0145] FIG. **5** is a diagram illustrating an example decomposition of the $CR^0(s)$ gate (top) and controlled-$CR^0(s)$ gate (bottom), as defined in eq. (55), into single qubit rotation gates and CNOTs (top) or Toffolis (bottom). The gate $R_y(\phi)$ is defined to map $|0\rangle\mapsto\cos(\phi/2)|0\rangle+\sin(\phi/2)|1\rangle$ and $|1\rangle\mapsto-\sin(\phi/2)|0\rangle+\cos(\phi/2)|1\rangle$. The rotation angle

$$\theta = 2\arctan\left(\frac{1-f(s)}{f(s)}\right),$$

where $f(s)$ given in eq. (59). The $CR^1(s)$ gate is identical but with the control bit sign flipped. Note that the $R_y(\pm\pi/4)$ gates are Clifford conjugate to a single T or $T^\dagger$ gate.

[0146] FIG. **6** is a diagram illustrating an example decomposition of the V_G unitary (top) and controlled-V_G unitary (bottom), as defined in eq. (57), into calls to a standard block-encoding unitary U_G and other elementary gates, using a single ancilla qubit initialized to the $|0\rangle$ state. Not pictured are additional ancillas that begin and end in $|0\rangle$ and are utilized to implement the unitary U_G in shallower depth.

TABLE V shows quantum resources used to create the state output by the quantum linear system solver, given in eq. (36) (QLSS, left) or the state used to compute the signs during the tomography subroutine, given in eq. (49) (Controlled QLSS, right) for a square linear system of size $L=2^\ell$. Note that these resource quantities do not yet account for the k classical repetitions used in order to perform tomography on the output state. The parameters Q and d each scale linearly with the condition number of the linear system, as defined in proposition 2. The symbols N_{Qcbe}, T_{Dcbe}, and T_{Ccbe} denote the number of logical qubits, the T-depth, and the T-count, respectively, for performing a controlled-block-encoding, as reported in table III. The symbols T_{DSp}, and T_{Csp} are analogous quantities for state-preparation, as reported in table IV. The parameters ε_{ar}, ε_{tsp} and $\varepsilon_z\in(0,1]$ are error parameters corresponding to the gate synthesis precision used for the $CR^0(s)$ and $CR^1(s)$ rotations, controlled state-preparation step used by tomography, and the QSVT phases, respectively.

TABLE V

Resource	QLSS	Controlled QLSS
# Qubits	$N_{Qcbe}+5$	$N_{Qcbe}+6$
T-depth	$12Q\log_2(1/\varepsilon_{ar})+2(Q+d)T_{Dcbe}+4(Q+d)T_{Dsp}+Q(24\ell+31)+3d\log_2(1/\varepsilon_z)+d(32\ell-2)$	$12Q\log_2(1/\varepsilon_{ar})+2(Q+d)T_{Dcbe}+4(Q+d)T_{Dsp}+Q(24\ell+36)+6d\log_2(1/\varepsilon_z)+d(32\ell-2)+12\log_2(1/\varepsilon_{tsp})+3(\ell-1)$
T-count	$12Q\log_2(1/\varepsilon_{ar})+2(Q+d)T_{Ccbe}+4(Q+d)T_{Csp}+Q(24\ell+31)+3d\log_2(1/\varepsilon_z)+d(32\ell-2)$	$12Q\log_2(1/\varepsilon_{ar})+2(Q+d)T_{Ccbe}+4(Q+d)T_{Csp}+Q(24\ell+51)+6d\log_2(1/\varepsilon_z)+d(32\ell-2)+12(L-1)\log_2(1/\varepsilon_{tsp})+16(L-\ell-1)$

3. Estimate for End-to-End Portfolio Optimization Problem

[0147] Recall that the full QIPM algorithm is an iterative algorithm, where each iteration involves approximately solving a linear system by preparing many copies of the same quantum states. The duality gap μ, which measures the proximity of the current interior point to the optimal point, begins at **1** and decreases by a constant factor σ with each iteration. Thus, the number of iterations to reach a final duality gap ∈ is given by:

$$N_{it}=\lceil \ln(\epsilon)/\ln(\sigma)\rceil=\left\lceil \frac{\ln(\epsilon)}{\ln\left(1-\frac{1}{20\sqrt{2r}}\right)}\right\rceil \approx \lceil 20\sqrt{2}\ln(\epsilon^{-1})\sqrt{r}\rceil. \quad (65)$$

Recall from the discussion in section III C 3 that the output of the QIPM will achieve an $\mathcal{O}(\epsilon)$ approximation to the optimal value of the objective function.

[0148] Pulling this all together, the resources to perform the full QIPM algorithm can be estimated, including the multiplicative factors used to perform tomography as well as the number of iterations to converge to the optimal solution. Note that the relevant condition number $\kappa_F(G)$ and linear-system precision ξ may vary from iteration-to-iteration as the Newton matrix G changes. The overall runtime can be upper bounded using the maximum observed value of $\kappa_F(G)$, which is denoted by κ_F, and minimum observed value of ξ across all iterations. At each iteration, to achieve overall precision ξ, the tomography precision ε is chosen to be just smaller than ξ (e.g., choose ε=0.9ξ), while all other error parameters (ε_{ar}, ε_{tsp}, ε_z, etc.) are chosen to be small constant fractions of ξ, such that a total error budget of ξ is not exceeded. As the non-tomographic error parameters all appear underneath logarithms, these small constant factors will drop out of a leading order analysis, and it suffices to replace all of these error parameters with ξ.

[0149] The overall runtime may then be expressed in terms of κ_F, ξ, L (the size of the Newton system), and r (the number of second-order cone constraints) up to leading order and including (e.g., all) constant factors, which are reported in table VI. Recall that for the infeasible version of the QIPM acting on the self-dual embedding, L=2N+K+3, where N is the number of SOCP variables and K is the number of linear constraints. Note that in the leading order expression, it is assumed that the contributions proportional to $Q=\mathcal{O}(\kappa_F)$ dominate over terms proportional to $Q=\mathcal{O}(\kappa_F)$ at practical choices of ξ due to the large constant prefactor in the definition of Q (see proposition 2 and surrounding discussion). The left column of table I from the introduction is formed using the expressions in table VI, and substituting the corresponding relations between L and n, where n is the number of stocks in the portfolio optimization problem given in eq. (10). That is, substitute r=3n+1 and L=2N+K+3=8n+3m+6=14n+6 when m=2n is taken, where N is the number of SOCP variables, K is the number of SOCP constraints, n is the number of stocks, and m is the number of time epochs used to create the matrix M as described in section II.

[0150] TABLE VI shows leading order contribution to the logical qubit count, T-depth, and T-count for the entire QIPM, including constant factors. The parameter L denotes the size of the Newton linear system and r denotes the number of second-order cone constraints, while ∈ denotes the final duality gap that determines when the algorithm is terminated. For the infeasible QIPM running on an n-asset instance of portfolio optimization, as given in eq. (10), L=14n+6 and r=3n+1; these substitutions yield the results in table I. The parameter κ_F denotes the maximum observed Frobenius condition number and ξ denotes the minimum observed tomographic precision parameter across all iterations.

TABLE VI

Resource	QIPM complexity
# Qubits	$4L^2$
T-depth	$(7\times10^8)\kappa_F L\sqrt{r}\xi^{-2}\log_2(\epsilon^{-1})\log_2(L)\log_2(\kappa_F L^{14/27}\xi^{-1})$
T-count	$(2\times10^8)\kappa_F L^3\sqrt{r}\xi^{-2}\log_2(\epsilon^{-1})\log_2(L)\log_2(\kappa_F\xi^{-1})$

VI. NUMERICAL EXPERIMENTS WITH HISTORICAL STOCK DATA

[0151] The resource expressions in table VI include constant factors but leave parameters κ_F and ξ unspecified. These parameters depend on the specific SOCP being solved. As a final step, numerical simulations of small PO problems can be used to study the size of these parameters for different PO problem sizes. This information enables us to give concrete estimates for the resources used to solve realistic PO problems with our implementation of the QIPM and sheds light on whether there could be an asymptotic quantum advantage.

[0152] The numerical experiments simulate the entirety of Algorithm 1. The (e.g., only) quantum part of the algorithm is to carry out the subroutine ApprSolve (G, h, ξ). The quantum algorithm for this subroutine can be simulated by

solving the linear system exactly using a classical solver and then adding noise to the resulting estimated values to simulate the output of tomography. Since the tomography scheme illustrated in section IV D repeatedly prepares the same state and draws k samples from measurements in the computational basis, the result is a sample from the multinomial distribution. In the numerical simulations herein, samples from this same multinomial distribution are drawn, thus capturing tomographic noise in a more precise way than by simply adding uniform Gaussian noise, as was done in conventional work. For simplicity, this disclosure assumes that the part of the tomography protocol that calculates the signs of each amplitude correctly computes each sign. To numerically estimate resource counts, it is helpful to understand what level of precision ξ is used to stay close enough to the central path throughout the algorithm, as well as how large the Frobenius condition number κ_F of the Newton system is. Noteworthy, it may be desirable to know how these quantities scale with system size and duality gap μ, which decreases by a constant factor with each iteration of the QIPM.

[0153] Section III C 5 discussed three formulations of the QIPM (see table II). The first (II-QIPM) is closely related to the original formulation, which does not guarantee that the intermediate points generated by the IPM are feasible. The other two are instantiations of the inexact-feasible formulation, which uses pre-computing a basis for the null-space of the SOCP constraint matrix. The first of these computes a valid basis by hand (IF-QIPM), while the second uses a QR decomposition to determine the basis (IF-QIPM-QR). All three versions were simulated, and it was determined that the II-QIPM stayed close to the central path, despite the lack of a theoretical guarantee that this would be the case. Here the results of the II-QIPM are presented. For comparison, in the section Additional Information E, some numerical results are presented for the feasible QIPMs, which do benefit from a theoretical convergence guarantee, but have other drawbacks.

[0154] As discussed in section V A, a simple preconditioner is implemented that reduces the condition number by about at least an order of magnitude with negligible additional classical cost. Resources estimates are reported assuming a preconditioned matrix.

A. Example Instance

[0155] FIG. **7** presents as an example the results of one of the simulations. A portfolio optimization instance of eq. (3) was constructed by randomly choosing n=30 stocks from the Dow Jones U.S. Total Stock Market Index (DWCF). The following parameters were, for example, arbitrarily set to q=1, ζ=0.05-1, and it was assumed the previous portfolio $\overline{w}$ allocates weight to each stock in proportion to its market capitalization. The returns of the 30 stocks on the first m=2n=60 days in the dataset were used to construct an average return vector $\hat{u}$ and an m×n matrix M for which $M^TM=\Sigma$, the covariance matrix for the stock returns, as described in section III B.

[0156] The infeasible QIPM acting on the corresponding SOCP in eq. (10) was also simulated. The figure illustrates how the simulation successfully follows the central path to the optimal solution after many iterations. The duality gap decreases with each step, and, notably, the infeasibility and distance to the central path also decrease (exponentially) with iteration. Also plotted is the tomography precision that was used to ensure that each iteration stayed sufficiently close to the central path (determined adaptively as described in the pseudocode in Algorithm 1). The plot exemplifies how, despite the lack of theoretical convergence guarantees, our simulations suggest that in practice the II-QIPM acting on the PO SOCP will yield valid solutions.

[0157] FIG. **7** is a plot illustrating simulation results of the QIPM on an SOCP instance corresponding to portfolio optimization on n=30 randomly chosen stocks using m=60 time epochs. The duality gap μ (defined in eq. (14)), the distance to the central path d_F (defined in eq. (26)), and the infeasibility (defined as the norm of the residual on the right-hand-side in eq. (19)) each decrease exponentially with the number of iterations. The tomography precision ξ to stay near the central path (defined adaptively as outlined in Algorithm 1) initially decreases and then plateaus at about 10^{-2}.

[0158] Remarkably, for this instance, observe that both the Frobenius condition number κ_F and the inverse tomography precision ξ^{-1} initially increase but ultimately plateau with the iteration number, even as the duality gap gets arbitrarily small (see FIG. **8** for data on κ_F). This scaling behavior was a generic feature of the simulations across all the instances simulated. This contrasts with the worst-case expectation that the condition number can increase as $\kappa_F \mathcal{O}(1/\mu)$ or $\kappa_F = \mathcal{O}(1/\mu^2)$ (depending on the formulation of the Newton system). Prior literature does not say much about whether the quantity ξ^{-1} should be expected to diverge. One might expect that, since the neighborhood of the central path gets smaller as μ gets smaller (e.g., radius is proportional to μ in eq. (27)), the precision requirement to stay close to the central path would get more stringent in proportion to μ. However, recall that the step size from one iteration to the next also shrinks with μ, and that ξ represents the size of the error on the normalized Newton system solution; thus the neighborhood does not shrink relative to the distance to the optimum and the length of the next step, and there is no immediate reason that ξ^{-1}, as defined, must diverge as $\mu \to 0$. However, one does expect that in the worst case, if the condition number κ diverges, then ξ^{-1} should also diverge, as errors of constant size on the estimate of $u/\|u\|$ can lead to residual errors of divergent size $\kappa\xi$ on the normalized product $Gu/\|Gu\|$.

[0159] FIG. **8** includes plots of the Median Frobenius condition κ_F number for 128 randomly sampled stock portfolios from the DWCF index as a function of the duality gap for portfolios of size 60, 80, 100, and 120 stocks. The shaded regions indicate the 16th to 84th percentile. Observe that the condition number appears to plateau at small values of the duality gap.

B. Scaling of Condition Number

[0160] To understand the problem scaling with portfolio size, example problem instances are generated by randomly sampling n stocks from the DWCF, using returns over m=2n time epochs (days) to construct our SOCP as in eq. (10). Parameters q, ζ, $\overline{w}$, $\hat{u}$ and M are all chosen in the same way as described above. The Frobenius condition number of the Newton matrix is plotted as well as the preconditioned Newton matrix as a function of the duality gap in FIG. **8** for portfolios of size $n \in \{60, 80, 100, 120\}$. As previously mentioned, the condition number appears to plateau at a certain value of the duality gap, especially for the preconditioned matrix.

[0161] To help understand the asymptotic scaling of the quantum algorithm it may be helpful is to determine how the condition number scales as a function of the number of assets, as the runtime of the QLSS algorithm grows linearly with the condition number. FIG. **9** plots the Frobenius condition number κ_F as a function of n, the number of stocks, observed at duality gaps $\mu \in \{10^{-1}, 10^{-3}, 10^{-5}, 10^{-7}\}$. At duality gaps of 10^{-5} and 10^{-7}, the condition number κ_F has plateaued as observed in FIG. **8**. A non-linear fit to the data is performed using a power law $\kappa_F \sim an^b$ model, where a and b are fit parameters, and the exponents b are reported in table VII. All exponents appear to be near or less than unity.

[0162] FIG. **9** is a plot of the Median Frobenius condition number κ_F for 128 randomly sampled stock portfolios from the DWCF index as a function of portfolio size for duality gaps of 10^{-1}, 10^{-3}, 10^{-5}, and 10^{-7}. The shaded regions correspond to the 16th to 84th percentile. The lines represent power-law fits of the form an^b, where the values for b are reported in table VII. In all four cases, the exponent is less than 1 and in the latter three cases it is greater than 0.9 suggesting a nearly linear-in-n trend.

TABLE VII

Estimated exponent parameters for the Frobenius condition number κ_F obtained from the fits that are plotted in FIG. 9.

Duality Gap	Condition Number Scaling
10^{-1}	$\mathcal{O}(n^{0.60\pm0.02})$
10^{-3}	$\mathcal{O}(n^{0.94\pm0.04})$
10^{-5}	$\mathcal{O}(n^{0.92\pm0.04})$
10^{-7}	$\mathcal{O}(n^{0.91\pm0.05})$

TABLE VIII

Estimated exponent parameters for $1/\xi^2$ obtained from the fits that are plotted in FIG. 10.

Duality Gap	Tomography Scaling
10^{-1}	$\mathcal{O}(n^{-0.19\pm0.05})$
10^{-3}	$\mathcal{O}(n^{1.10\pm0.06})$
10^{-5}	$\mathcal{O}(n^{0.79\pm0.11})$
10^{-7}	$\mathcal{O}(n^{1.16\pm0.10})$

C. Scaling of Tomography Precision

[0163] While the depth of the individual quantum circuits that compose the QIPM scales only with the Frobenius condition number, the QIPM also includes a number of repetitions of this circuit for tomography that scales as $1/\xi^2$, the inverse of the tomography precision squared. To see how this scales with problem size, an analysis for ξ^{-2} is performed similar to the analysis performed for κ_F. These results are presented in FIG. **10** for the same four duality gaps of $\{10^{-1}, 10^{-3}, 10^{-5}, 10^{-7}\}$. To reduce the iteration-to-iteration variation in the tomography precision (which results from our adaptive approach to tomography in Algorithm 1), in calculating ξ^{-2} at duality gap μ, one may take the average over the value of ξ^{-2} at the five iterations with duality gap nearest to μ. The median of ξ^{-2} can be fit at each value of n to a linear model on a log-log plot, corresponding to a relationship $\xi^{-2}=an^b$. The implied exponent b is in table VIII. In this case, it is hard to draw robust conclusions from the fits. The fit suggests that the median of ξ^{-2} is increasing with n on the interval $n \in [10, 120]$. However, the most striking feature of the data is that the instance-to-instance variation of ξ^{-2} is significantly larger than that of κ_F. In fact, at $\mu=10^{-7}$, the 84th percentile of in stances at n=10, the smallest size simulated, had a larger value of ξ^{-2} than the 50th percentile of instances at n=120, the largest size simulated.

[0164] FIG. **10** is a plot of the median value of the square of the inverse tomography precision ξ^{-2} used to remain in the neighborhood of the central path for 128 randomly sampled stock portfolios from the DWCF index as a function of portfolio size for duality gaps of 10^{-1}, 10^{-3}, 10^{-5}, and 10^{-7}. To reduce iteration-to-iteration variation, an artifact of the adaptive approach to tomography, average over the observed value of ξ^{-2} at the five iterations for which the duality gap is nearest the indicated value. The shaded regions correspond to the 16th to 84th percentile. Here logarithmic axes are used since (unlike for κ_F) instance-to-instance variation covers multiple orders of magnitude even for a fixed value of n. The dashed lines correspond to a linear fit to the log-log data, where the slope is reported in table VIII.

D. Asymptotic Scaling of Overall Runtime

[0165] Above fits for κ_F and ξ^{-2} as a function of n on the range $n \in [10, 120]$ are provided. Here the quantity $n^{1.5}\kappa_F/\xi^2$ is studied, which determines the asymptotic scaling of the runtime of the QIPM. FIG. **11** plots this quantity at the same four duality gap values $\mu \in \{10^{-1}, 10^{-3}, 10^{-5}, 10^{-7}\}$. The implied exponents arising from linear fits on a log-log axis are reported in table IX. They are generally consistent with summing the exponents from the previously reported fits. The data inherit from ξ^{-2} the feature that the instance-to-instance variation is orders of magnitude larger than the median. Taken at face value, the fits suggest that the scaling of the median algorithmic runtime on the interval $n \in [10, 120]$ is similar to the $n^{3.5}$ scaling of classical IPMs using Gaussian elimination, and worse than the asymptotic $n^{2.87}$ arising from classical IPMs using fast matrix-multiplication techniques to solve linear systems (note that this scaling does not apply until n becomes very large, so it is not a good practical comparator). However, the large variance and imperfect fits do not give confidence that these trends can be reliably extrapolated to larger n. Accordingly, when actual resource counts are computed in the next subsection, the inventors stick to n=100.

[0166] Ultimately, it is not essential to pin down the asymptotic scaling of the algorithm, because a determination of this work is that, even if a slight asymptotic polynomial speedup exists, the size of the constant prefactors involved in the algorithm preclude an actual practical speedup, barring significant improvements to multiple aspects of the algorithm. The next subsection elaborates on this point in a more quantitative fashion.

[0167] FIG. **11** is a plot of the median value of the estimated algorithm scaling factor computed as the median of $n^{1.5}\kappa_F/\xi^2$ for 128 randomly sampled stock portfolios from the DWCF index as a function of portfolio size for duality gaps of 10^{-1}, 10^{-3}, 10^{-5}, and 10^{-7}. As in FIG. **10**, over 5 consecutive points are averaged to reduce iteration-to-iteration variance deriving from adaptive tomography. Also, the actual number of observed samples are used that were used to achieve sufficient tomographic precision in place of the tomographic factor n/ξ^2. The shaded regions correspond to

the 16th to 84th percentiles. The lines correspond to a linear fit to the log-log data, where the slope is reported in table IX.

[0168] TABLE IX shows exponent parameter estimates from the fits to the line generated by plotting $n^{1.5}\kappa_F/\xi^2$ in FIG. **11**, which determines the overall scaling of the runtime of the QIPM. For comparison, CIPMs using Gaussian elimination have runtime $\mathcal{O}(n^{3.5})$ and CIPMs using faster methods for solving linear systems have runtime $\mathcal{O}(n^{2.87})$.

TABLE IX

Duality Gap	Algorithm Scaling
10^{-1}	$\mathcal{O}(n^{2.01\pm0.05})$
10^{-3}	$\mathcal{O}(n^{3.56\pm0.07})$
10^{-5}	$\mathcal{O}(n^{3.36\pm0.14})$
10^{-7}	$\mathcal{O}(n^{3.75\pm0.12})$

E. Numerical Resource Estimates

[0169] Rather than examine algorithmic scaling, actual resource counts for the QIPM applied to PO are computed. It is these resource counts that may matter most from a practical perspective. The total circuit size is estimated in terms of the number of qubits, T-depth, and T-count for a portfolio of 100 assets. This size is chosed because it is small enough that the entire quantum algorithm can be simulated classically. However, at this size, solving the PO problem is not classically hard; generally speaking, the PO problem becomes challenging to solve with classical methods when n is on the order of 10^3 to 10^4. A similar concrete calculation can be performed at larger n by extrapolating trends observed in the numerical simulations herein, but there is not confidence that the fits on $n\in[10, 120]$ reported above are reliable predictors for larger n.

[0170] Recall that the only step in the QIPM performed by a quantum computer is the task of producing a classical estimate to the solution of a linear system to error ξ. The complexity of this task as it is performed within the QIPM depends on ξ as well as the Frobenius condition number κ_F. The first step of the calculation is to fix values for ξ and κ_F at n=100. They are choosen by taking the median over the 128 samples in the numerical simulation at duality gap $\mu=10^{-7}$.

[0171] Once κ_F and ξ are fixed, concrete values for the various other error parameters can be determined that appear in the algorithm such that overall error ξ can be achieved. Tomography dominates the complexity and overall error, but there are a number of other factors that contribute to the error in the final solution. The sources of error are enumerated and labeled here for completeness:

[0172] ε_G: Error in block-encoding the matrix G

[0173] ε_h: Error in the unitary that prepares the state $|h\rangle$

[0174] ε_{ar}: Gate synthesis error for single-qubit rotations used by $CR^0(s)$ and $CR^1(s)$ (see FIG. **5**)

[0175] ε: Tomography error

[0176] ε_z: Gate synthesis error for each single-qubit rotation used for QSVT eigenstate filtering (see FIG. **2**)

[0177] ε_{qsp}: Error due to polynomial approximation in eigenstate filtering

[0178] ε_{tsp}: Error in preparing the state $\Sigma_{i=1}^{L}\sqrt{p_i}|i\rangle$ used for computing the signs in the tomography routine $\rangle$

[0179] Section IV described a quantum circuit that prepares a state $|\tilde{v}\rangle$ (after postselection) for which $\||\tilde{v}\rangle-|v\rangle\|\leq\varepsilon_{QLSP}$. If the block-encoding unitaries, state-preparation unitaries, and single-qubit rotations were perfect, then the only contribution to ε_{QLSP} would be from eigenstate filtering and we may have $\varepsilon_{QLSP}\leq\varepsilon_{qsp}$. Note the relationship $d=2\kappa_F\ln(2/\varepsilon_{qsp})$ from proposition 2. Since the block-encoding unitary U_G, the state-preparation unitary U_h, and the single-qubit rotations are implemented imperfectly, there is additional error. In preparing the state, the unitary U_G is called 2Q+2d times and the unitary U_h is called 4Q+4d times, where Q is given in proposition 2. Additionally, there are 2Q combined appearances of $CR^0(s)$ and $CR^1(s)$ gates, where each appearance includes two single-qubit rotations. Note that the appearances of $CR^0(s)$ and $CR^1(s)$ within the eigenstate filtering portion of the circuit do not contribute to the error because at s=1 these gates can be implemented exactly. Additionally, there are another d single-qubit rotations used to implement the eigenstate filtering step. Since operator norm errors add sublinearly, one can thus say that

$$\varepsilon_{QLSP}\leq\varepsilon_{qsp}+(2Q+2d)\varepsilon_G+(4Q+4d)\varepsilon_h+4Q\varepsilon_{ar}+2d\varepsilon_z. \tag{66}$$

[0180] Now, the result of proposition 4 implies that, in order to assert that the classical estimate $\tilde{v}'$ output by tomography satisfies $\|\tilde{v}'-v\|\leq\xi$, it suffices to have

$$\xi\geq\varepsilon+1.58\sqrt{L}\varepsilon_{tsp}+1.58[\varepsilon_{qsp}+(2Q+2d)\varepsilon_G+(4Q+4d)\varepsilon_h+4Q\varepsilon_{ar}+d\varepsilon_z], \tag{67}$$

where for convenience recall the definitions (ignoring the $\mathcal{O}(\sqrt{\kappa_F})$ term) of Q and d as

$$Q=2C\kappa_F \tag{68}$$

$$d=2\kappa_F\ln(2/\varepsilon_{qsp}) \tag{69}$$

Recalling that the dominant term in the complexity of the algorithm scales as ε^{-2} but logarithmically in the other error parameters, to minimize the complexity assign the majority of the error budget to ε: let $\varepsilon=0.9\xi$, and split the remaining 0.1ξ across the remaining six terms of eq. (67). There is room for optimizing this error budget allocation, but the savings would be at most a small constant factor in the overall complexity.

[0181] Note that elsewhere in the draft, ξ has been referred to as "tomography precision" since ε will dominate the contribution to ξ. Here, the resource calculation uses differentiating ε from ξ, but when speaking conceptually about the algorithm, one can focus on ξ as it is the more fundamental parameter: it represents the precision at which the classical-input-classical-output linear system problem is solved, allowing apples-to-apples comparisons between classical and quantum approaches.

[0182] With values for κ_F, ε_G, ε_h, ε_{qsp}, ε_z, and ε_{tsp} now fixed, the resource count can be completed using the expressions in table V. Note that for gate synthesis error, the following formula is used $R_y=3\log_2(1/\varepsilon_r)$, where R_y is the number of T gates used to achieve an ε_r-precise Clifford+T gate decomposition of the rotation gate. Putting this all together yields the resource estimates for a single run of the (uncontrolled) quantum linear system solver in table X at n=100. We report these estimates both in terms of primitive block-encoding and state-preparation resources, as well as the raw numerical estimates. For the total runtime, the resources used for the controlled state-preparation routine may also be estimated. These quantities are estimated, but to

the precision of the reported estimates, the numbers are the same as the controlled version, so they are excluded for brevity.

[0183] To estimate total runtime, our estimates are multiplied by the tomography factor k (for controlled and for uncontrolled) as well as the number of iterations $N_{it}=\lceil \ln(\in)/\ln(\sigma)\rceil$, where $\in$ is the target duality gap (which is taken to be $\in=10^{-7}$), and $\sigma=1.0-1/(20\sqrt{2r})$. While k will vary from iteration to iteration, in the calculation it is assumed that the total number of repetitions is given by the simple product $(2k)N_{it}$, which, noting that the value of ξ plateaus after a certain number of iterations, will give a roughly accurate estimate. Note that these $2kN_{it}$ repetitions need not be done coherently, in the sense that the entire system is measured and reprepared in between each repetition. One can bound the tomography factor k to be $k\leq 57.5\, L\ln(L)/\xi^2$, where ξ is determined empirically. However, our numerical simulations of the algorithm yield an associated value of k used to generate the estimate to precision ξ, so this numerically determined value can be used directly. The observed median value of $k=3.3\times10^8$ from simulation is multiple orders of magnitude smaller than the theoretical bound. Using this substitution for k and N_{it}, the results are shown in the right column of table I in the introduction.

[0184] To aid in understanding which portions of the algorithm dominate the complexity, a breakdown of the resources is shown in FIG. **12**. The width of the boxes is representative of the T-depth, while the height of the boxes represents the T-count. The number of classical repetitions, composed of tomography samples as well as IPM iterations used to reach a target duality gap, contributes the largest factor to the algorithmic runtime. Of these two, quantum state tomography contributes more than the iterations used to reach the target duality gap. Our exact calculation confirms that for the individual quantum circuits involved in the QLSS, the discrete adiabatic portion of the algorithm dominates over the eigenstate filtering step in its contribution to the overall quantum circuit T-depth. Within the adiabatic subroutine, the primary driver of the T-depth and T-count is the application of the block-encoding operator Q times (see e.g., eq. (61)), where Q is proportional to the Frobenius condition number. An additional source of a large T-count arises from block-encoding the linear system, which causes the T-count to scale as $\mathcal{O}(L^2)$.

[0185] TABLE X shows estimated number of logical qubits N_Q, T-depth T_D, and T-count T_C used to perform the quantum linear system solver (QLSS) subroutine within the QIPM running on a PO instance with n=100 stocks. This calculation uses the empirically observed median value for the condition number at duality gap $\mu=10^{-7}$, which was $\kappa_F=1.6\times10^4$. The full QIPM repeats this circuit $k=\mathcal{O}(n\ln(n)\xi^{-2})$ times in each iteration to generate a classical estimate of the output of the QLSS, and also performs $N_{it}=\mathcal{O}(n^{0.5})$ iterations, where the linear system being solved changes from iteration to iteration. In the left column, the resources are written as numerical prefactors times the resources used to perform the controlled-block-$\rangle$ encoding of the matrix G (denoted by a subscript cbe), and the state-preparation of the vector $|h\rangle$ (denoted by a subscript sp), defined in tables III and IV. Written this way, one can see the large prefactors occurring from the linear system solver portion of the algorithm. In the right column the exact resources are computed, including those coming from the block-encoding. The notation AeB is short for $A\times10^B$.

TABLE X

QLSS Prefactors	Total
$N_Q = N_{Qcbe} + 5$	N_Q = 8e6
$T_D = (1e8)T_{Dcbe} + (3e8)T_{Dsp} + (5e10)$	T_D = 4e11
$T_C = (1e8)T_{Ccbe} + (3e8)T_{Csp} + (5e10)$	T_C = 2e17

VII. CONCLUSIONS

A. Bottlenecks

[0186] The resource quantities reported are large, even for the classically easy problem size of n=100 assets in the portfolio optimization instance. Our detailed analysis allows us to see exactly how this large number arises, which is helpful for understanding how to improve it. Several independent factors leading to the large resource estimates are outlined below.

[0187] The block-encoding of the classical data may be called many times by the QLSS. This data is arranged in an L×L matrix (note that for a PO instance of size n with m=2n, the Newton linear system has size roughly $L\approx14n$). These block-encodings can be implemented up to error ε_G in $\mathcal{O}(\log(L/\varepsilon_G))$ T-depth using circuits for quantum random access memory (QRAM) as a subroutine. While the asymptotic scaling is favorable, after close examination of the circuits for block-encoding, in practice the T-depth can be quite large: at n=100 and ε_G 10^{-10} (it may be necessary to take ε_G very small since the condition number of G is quite large), block-encoding to precision ε_G has a T-depth of nearly 1000. Notably, this T-depth arises even after implementing several new ideas to minimize the circuit depth.

[0188] The condition number κ_F determines how many calls to the block-encoding are to be made, and it is observed that κ_F may be quite large for the application of portfolio optimization. Even after an preconditioning, κ_F is on the order of 10^4 already for small SOCP instances corresponding to n=100 stocks, and empirical trends suggest it grows nearly linearly with n. However, additional preconditioning may significantly reduce the effective value of κ_F in this algorithm.

[0189] The constant factor in front of the $\mathcal{O}(\kappa_F)$ in QLSSs is also quite large: the theoretical analysis proves an upper bound on the prefactor of 3×10^4. Numerical simulations previously performed suggested that, in practice, it can be one order of magnitude smaller than the theoretical value. Thus, the constant prefactor may be taken to be 2000 in the numerical estimates herein, which still contributes significantly to the estimate.

[0190] Pure state tomography includes preparing many copies of the output $|v\rangle$ of the QLSS. This disclosure improved the constant prefactors in the theoretical analysis beyond what was known, but even with this improvement, the number of queries used to produce an estimate v' of the amplitudes of $|v\rangle$ up to error ε in ℓ_2 norm is $115\, L\ln(L)/\varepsilon^2$, which for n=100 and $\varepsilon=10^{-3}$ is on the order of 10^{11} (although our simulations suggest $2k=7\times10^8$ suffice in practice).

[0191] QIPMs, like CIPMs, are iterative algorithms; the number of iterations in our implementation is roughly $20\sqrt{2r}\ln(\in^{-1})$, a number chosen to utilize theoretical guarantees of convergence (note that $r\approx3n$). Taking

n=100 and $\in=10^{-7}$, our implementation would use 8×10^3 iterations. The number of iterations may be significantly decreased if more aggressive choices are made for the step size. For example, similar to the adaptive approach to tomographic precision, one may set longer step sizes first, and shorten the step size when the iteration does not succeed. This sort of optimization may apply equally to CIPMs and QIPMs.

[0192] Remarkably, the five factors described above contribute roughly equally to the overall T-depth calculation; the exception being the number of copies used to do tomography, which is a much larger number than the others. Another comment regarding tomography is that, in principle, the k tomographic samples can be taken in parallel rather than in series. Running in parallel leads to a huge overhead in memory: one can reduce the tomographic depth by a multiplicative factor P at the cost of a multiplicative factor P additional qubits. Note that even preparing a single copy uses the large number of nearly ten million logical qubits at n=100. Moreover, it is unlikely that improvements to tomography alone may make the algorithm practical, as the other four factors still contribute roughly 10^{16} to the T-depth.

[0193] Besides the rather large constant factors pointed out above for tomography and especially for the QLSS, note that the multiplicative "log factors" that are typically hidden underneath $\tilde{\mathcal{O}}$ notation in asymptotic do tomography, which is a much larger number than the analyses contribute meaningfully here. For instance, the entire block-encoding depth is $\mathcal{O}(\log(n/\varepsilon_G))$, which, in practice, is as large as 1000. Moreover, there is an additional $\ln(\in^{-1})\approx16$ coming from the iteration count, and a $\ln(L)\approx7$ from tomography.

[0194] This quantitative analysis of bottlenecks for QIPMs can inform likely bottlenecks in other applications where QLSS, tomography, and QRAM subroutines are used. While some parameters such as κ_F and ξ are specific to the application considered here, other observations such as the numerical size of various constant and logarithmic factors (e.g., block-encoding depth) would apply more generally in other situations.

[0195] FIG. **12** is a diagram illustrating the breakdown of the quantum resources used for a single coherent run of the uncontrolled version of the quantum algorithm used to produce the state eq. (36). As in table X, take the final duality gap to be $\mu=10^{-7}$ and the number of assets to be n=100. Choices for the Frobenius condition number $\kappa_F=1.6\times10^4$ and number of tomographic repetitions $k=3.3\times10^8$ are informed by our numerical experiments, as discussed in section VI. A similar breakdown for the controlled version used to produce the state eq. (49) would be basically the same. The eigenstate filtering sub-circuit follows a similar alternating structure to the adiabatic evolution, with the U[j] block-encodings replaced with either U[1] or $U[1]^\dagger$, the reflection operator W replaced with phase rotations, and only d<<Q total number of iterations (refer to FIG. **2** for details.)

B. Resource Estimate Given Dedicated QRAM Hardware

[0196] The bottlenecks above focused mainly on the T-depth and did not take into account the total T-count or the number of logical qubits, which are also large. Indeed, the estimate of 8 million logical qubits, as reported in table I, is drastically larger than estimates for other quantum algorithms, such as Shor's algorithm and algorithms for quantum chemistry, both of which can be on the order of 10^3 logical qubits. By contrast, the current generation of quantum processors have tens to hundreds of physical qubits, and no logical qubits; a long way from the resources used for this QIPM.

[0197] However, note that, as for other algorithms using repeated access to classical data, the vast majority of the gates and qubits in the QIPM arise in the block-encoding circuits, which are themselves dominated by QRAM-like data-loading subcircuits. These QRAM-like sub-circuits have several special features. Firstly, they are largely composed of controlled-swap gates, each of which can be decomposed into four T gates that can even be performed in a single layer, given one additional ancilla and classical feed-forward capability. Furthermore, in some cases, the ancilla qubits can be "dirty", i.e., initialized to any quantum state, and, if designed correctly, the QRAM circuits can possess a natural noise resilience that may reduce the resources used for error correction. Implementing these circuits with full-blown universal and fault-tolerant hardware could be unnecessary given their special structure. Just as classical computers have dedicated hardware for RAM, quantum computers may have dedicated hardware optimized for performing the QRAM operation. Preliminary work on hardware based QRAM data structures (as opposed to QRAM implemented via quantum circuits acting on logical qubits) shows promise in this direction.

[0198] Our estimates suggest that the size of the QRAM used to solve an n=100 instance of PO is one megabyte, and the QRAM size for $n=10^4$ (i.e., sufficiently large to potentially be challenging by classical standards) is roughly 10 gigabytes, which is comparable to the size of classical RAM one might use on a modern laptop. These numbers could perhaps be reduced by exploiting the structure of the Newton matrix, as certain blocks are repeated multiple times in the matrix, and many of the entries are zero (see eqs. (10) and (19)). Exploiting the sparsity of the matrix can lead to reduced logical qubit count and T-count, but not reduced T-depth. In fact, it may lead to non-negligible increases in the T-depth, since the shallowest block-encoding constructions can be hyper-optimized for low-depth, and are explicitly not compatible with exploiting sparsity.

[0199] With this in mind, one can ask the following hypothetical question: suppose access to a sufficiently large dedicated QRAM element in our quantum computer, and furthermore that the QRAM ran at a 4 GHz clock speed (which is comparable to modern classical RAM); would the algorithm become more practical in this case? Under the conservative simplifying assumption that each block-encoding and state-preparation unitary uses just a single call to QRAM and the rest of the gates are free, a rough answer can be given by referring to the expression in table X, which states that 4×10^8 total block-encoding and state-preparation queries are used. Thus, even if the rest of the estimates stay the same, the number of QRAM calls involved in just a single QLSS circuit for n=100 would be 4×10^8. Accounting for the fact that the QIPM involves an estimated 6×10^{12} repetitions of similarly sized circuits, the overall number of QRAM calls used to solve the PO problem would be larger than 10^{21}, and the total evaluation time would be on the order of ten thousand years. Thus, even at 4 GHz speed for the QRAM, the problem remains decidedly intractable. Nonetheless, if the QIPM is made practical, it may involve specialized QRAM hardware in combination with improve-

ments to the algorithm itself. Separate from the QLSS, a relatively small number of state preparation queries is used in tomography to create the state in eq. (50), but this number does not scale with K and it is neglected in this back-of-the-envelope analysis.

C. Comparison Between QIPMs and CIPMs and Comments on Asymptotic Speedup

[0200] The discussion above suggests that the current outlook for practicality with a QIPM is pessimistic, but simultaneously highlights several avenues by which to improve the results. Even with such improvements, if QIPMs are to one day be practical, they may need to at least have an asymptotic speedup over CIPMs. Here we comment on this possibility. A step of both QIPMs and CIPMs is the problem of computing a classical estimate of the solution to a linear system, a task that is also of broad use beyond interior point methods. Thus, we can compare different approaches to solving linear systems, and our conclusions are relevant in any application where linear systems are solved. Accordingly, in table XI, the asymptotic runtime of several approaches to solving an L×L linear system to precision are given, including the (QLSS+tomography) approach utilized by QIPMs, as well as two classical approaches. Whereas prior literature primarily compared against Gaussian elimination (which scales as $\mathcal{O}(L^3)$), this disclosure notes a comparison against the randomized Kaczmarz method, which scales as $\mathcal{O}(L\kappa_F^2 \ln(\xi^{-1}))$. This scaling comes from the fact that $2\kappa_F^2 \ln(\xi^{-1})$ iterations are used, and each iteration involves computing several inner products at cost $\mathcal{O}(L)$. It is observed that the worst-case cost of an iteration is 4L floating point multiplications, meaning all the constant prefactors involved are more-or-less mild. Thus, the asymptotic quantum advantage of the QIPM is limited to an amount equal to $\mathcal{O}(\min(\xi^2\kappa_F, \xi^2 L^2/\kappa_F))$, which is at most $\mathcal{O}(L)$ when $\kappa_F \propto L$ and $\xi=O(1)$. Encouragingly, our numerical results are consistent with $\kappa_F \propto L$. However, our results are not consistent with $\xi=\mathcal{O}(1)$, suggesting instead that ξ is decreasing with L.

[0201] TABLE XI shows a comparison of time complexities of different approaches for exactly or approximately solving an L×L linear system with Frobenius condition number κ_F to precision ξ. The comparison highlights how a quantum advantage only persists when κ_F is neither too large nor too small. The constant pre-factor roughly captures the T-depth determined for the quantum case (the same pre-factor from Tab. VI after discounting the $20\sqrt{2}$ IPM iteration factor) and the number of multiplications in the classical case.

TABLE XI

Solver	Type	Complexity	Pre-factor estimate
QLSS + Tomography	Quantum, Approximate	$L\kappa_F\xi^{-2}$ $\ln(L)\ln(\kappa_F\xi^{-1}L^{14/27})$	5×10^7
Gaussian Elimination	Classical, Exact	L^3	1/3
Randomized Kaczmarz	Classical, Approximate	$L\kappa_F^2\ln(\xi^{-1})$	8

[0202] If $\kappa_F \propto L$ and $\xi=\mathcal{O}(1)$, it is determined a total QIPM runtime of $\mathcal{O}(n^{2.5})$, improving over classical $\mathcal{O}(n^{3.5})$ for a portfolio with n stocks. This speedup is a material asymptotic improvement over the classical complexity, but leveraging this speedup for a practical advantage might still be difficult. Firstly, the difference in the constant prefactor between the quantum and classical algorithms would likely negate the speedup unless n is taken to be very large. Secondly, the speedup would necessarily be sub-quadratic. In the context of combinatorial optimization, where quadratic speedups can be obtained easily via Grover's algorithm, even a quadratic speedup is unlikely to exhibit actual quantum advantage after factoring in slower quantum clock speeds and error-correction overheads.

[0203] Our results suggest that determining a practical quantum advantage for portfolio optimization might require structural improvements to the QIPM itself. In particular, it may be helpful to explore whether additional components of the IPM can be quantized, and whether the costly contribution of quantum state tomography could be completely circumvented. Naively, circumventing tomography entirely is challenging, as it is useful (e.g., important) to retrieve a classical estimate of the solution to the linear system at each iteration in order to update the interior point and construct the linear system at the next iteration. Nevertheless, tomography represents a formidable bottleneck.

[0204] While our results are pessimistic on the question of whether quantum interior point methods will deliver quantum advantage for portfolio optimization (and other applications), it is our hope that by highlighting the precise issues leading to daunting resource counts, our work can inspire innovations that render quantum algorithms for optimization more practical. Finally, we conclude by noting that detailed, end-to-end resource estimations of the kind performed here may be helpful (e.g., important) for commercial viability of quantum algorithms and quantum applications. While it is helpful to discover and prove asymptotic speedups of quantum algorithms over classical, an asymptotic speedup alone does not imply practicality. For this, a detailed, end-to-end resource estimate is used, as the quantum algorithm may nevertheless be far from practical to implement.

VIII. ADDITIONAL INFORMATION

Additional Information A: Notation

[0205] Here we list the symbols that appear in this disclosure for reference.

[0206] Symbols related to portfolio optimization

[0207] n: number of stocks in the portfolio

[0208] w: length-n vector indicating fraction of portfolio allocated to each stock (the object to be optimized)

[0209] $\overline{w}$: length-n vector indicating current portfolio allocation

[0210] ζ: length-n vector indicating maximum allowable change to portfolio

[0211] $\hat{u}$: length-n vector of average returns

[0212] Σ: n×n covariance matrix capturing deviations from average returns

[0213] q: parameter in objective function that determines relative weight of risk vs. return (eq. (3))

[0214] M: m×n matrix corresponding to the square-root of Σ, i.e. $\Sigma=M^TM$

[0215] m: number of rows in M, often equal to the number of time epochs (section III B)

[0216] Symbols related to second-order cone programs

[0217] $\mathcal{Q}^k$: second-order cone of dimension k (eq. (4))

[0218] $\mathcal{Q}$: product set of several second-order cones

[0219] e: identity element for $\mathcal{Q}$ or $\mathcal{Q}^k$ (depending on context)

[0220] N: total number of variables in the SOCP

[0221] K: total number of linear constraints in the SOCP

[0222] r: number of second-order cone constraints in the program

[0223] x: length-N vector; primal variable to be optimized, constrained to $\mathcal{Q}$

[0224] y: length-K vector; dual variable to be optimized

[0225] s: length-N vector, appears in dual program, constrained to $\mathcal{Q}$

[0226] A: K×N matrix encoding linear constraints (eq. (5))

[0227] b: length-K vector encoding right-hand side of linear constraints (eq. (5))

[0228] c: length-N vector encoding objective function (eq. (5))

[0229] μ(x, s): duality gap of the primal-dual point (x, s) (eq. (7))

[0230] τ, κ, θ: additional scalar variables introduced to implement self-dual embedding (e.g., see section III C 3)

[0231] μ(x, τ, s, κ): duality gap of the point (x, τ, s, κ) of the self-dual SOCP (eq. (14))

[0232] X, S: arrowhead matrices for vectors x and s (eq. (21))

[0233] B: basis for null space of self-dual constraint matrix

[0234] Symbols related to second-order cone programs for portfolio optimization

[0235] ϕ: length-n variable introduced during reduction from PO to SOCP; part of x (eq. (10))

[0236] ρ: length-n variable introduced during reduction from PO to SOCP; part of x (eq. (10))

[0237] t: scalar variable introduced during reduction from PO to SOCP; part of x (eq. (10))

[0238] η: length-m variable introduced during reduction from PO to SOCP; part of x (eq. (10))

[0239] Symbols related to interior point methods (IPMs)

[0240] v: parameterizes central path (eq. (12))

[0241] $d_F(x, \tau, S, \kappa)$: distance of the point (x, τ, S, κ) to the central path of the self-dual SOCP (eq. (13))

[0242] $\mathcal{N}$, $\mathcal{N}$, ℓ_F: neighborhoods of the "central path" (eqs. (27) and (28))

[0243] γ: radius of neighborhood of central path

[0244] σ: step length parameter

[0245] L: size of (square) Newton matrix

[0246] ∈: input to IPM specifying error tolerance, algorithm terminates once duality gap falls beneath ∈

[0247] Relations Between Parameters

[0248] Self-dual embedding has 2N+K+3 parameters and N+K+2 linear constraints

[0249] Newton matrix has size L=2N+K+3 for infeasible approach and L=N+1 for feasible approach

[0250] For PO formulation in eq. (10), N=3n+m+1, r=3n+1, K=2n+m+1

[0251] In our numerical experiments, we choose m=2n

[0252] Symbols related to quantum linear system solvers

[0253] G: L×L matrix encoding linear constraints

[0254] h: length-L vector encoding right-hand-side of linear constraints

[0255] u: solution to linear system Gu=h

[0256] v: normalized solution to linear system u/‖u‖

[0257] ε_{QLSP}: error in solution to linear system

[0258] ṽ: normalized output of the QLSS, which should satisfy $\|v-\tilde{v}\| \leq \varepsilon_{QLSP}$

[0259] $\ell: \lceil \log_2 L \rceil$

[0260] U_G: block-encoding unitary for G

[0261] ℓ_G: number of ancilla qubits used by U_G

[0262] U_h: state-preparation unitary for $|h\rangle$

[0263] $\kappa_F(G)$: Frobenius condition number $\|G\|_F\|G^{-1}\|$ of G

[0264] Q: number of queries to U_G and U_h (proposition 1)

[0265] C: constant prefactor of κ_F (proposition 1)

[0266] d: the degree of the polynomial used in eigenstate filtering (proposition 2)

[0267] Symbols related to block encoding and state preparation

[0268] ε_G: block-encoding error for matrix G

[0269] ε_h: state-preparation error for vector h

[0270] ε_{ar}: Gate synthesis error for rotations needed by $CR^0(s)$ and $CR^1(s)$

[0271] ε_z: Gate synthesis error for rotations needed by the QSP phases

[0272] ε_{qsp}: Error due to polynomial approximation in eigenstate filtering

[0273] ε_{tsp}: Error in preparing the state $\Sigma_{i=1}^{L}\sqrt{p_i}|i\rangle$ needed for the tomography routine

[0274] N_{Qbe}, T_{Dbe}, and T_{Cbe}: number of logical qubits, T-depth, and T-count required for block-encoding.

[0275] N_{Qcbe}, T_{Dcbe}, and T_{Ccbe}: number of logical qubits, T-depth, and T-count required for controlled-block-encoding.

[0276] N_{Qsp}, T_{Dsp}, and T_{Csp}: number of logical qubits, T-depth, and T-count required for state preparation.

[0277] N_{Qcsp}, T_{Dcsp}, and T_{Ccsp}: number of logical qubits, T-depth, and T-count required for controlled-state preparation.

[0278] Symbols related to tomography

[0279] k: number of measurements on independent copies of the state

[0280] δ: probability of failure

[0281] ε: guaranteed error of tomographic estimate

[0282] ξ: overall precision of solution to linear system, dominated by tomographic error

Additional Information B: Deferred Proofs

1. Quantum State Tomography

[0283] Proof of proposition 3. Consider a single coordinate α_j with associated probability $p_j=|\alpha_j|^2$, and suppose k samples are taken to determine an estimate $\tilde{p}_j$ of p_j. By Bernstein's inequality,

$$Pr[|\tilde{p}_j - p_j| > \varepsilon_j] \leq 2\exp\left(-\frac{\varepsilon^2}{2(p_j+\varepsilon/3)}k\right) \tag{B1}$$

and so for a given component-wise target deviation in the probability ε_j, choosing

$$k \geq \frac{2(p_j+\varepsilon/3)}{\varepsilon^2}\ln(2/\delta') = \frac{2(|\alpha_j|^2+\varepsilon/3)}{\varepsilon^2}\ln(2/\delta') \tag{B2}$$

guarantees that $\Pr[|\tilde{p}_j-p_j|>\varepsilon_j]\leq\delta'$.

[0284] Now pick $\varepsilon_j=\sqrt{3\gamma}|\alpha_j|\varepsilon+\gamma\varepsilon^2$ for some yet undetermined $\gamma>0$. With this choice

$$\begin{aligned}
&\frac{2\left(|\alpha_j|^2+\frac{\varepsilon}{3}\right)}{\varepsilon^2}\ln(2/\delta') \\
&= \frac{2\left(|\alpha_j|^2+\sqrt{\frac{\gamma}{3}}\varepsilon+\frac{\gamma}{3}\varepsilon^2\right)}{\left(\sqrt{3\gamma}|\alpha_j|\varepsilon+\gamma\varepsilon^2\right)^2}\ln(2/\delta') \\
&\leq \frac{2\left(|\alpha_j|^2+2\sqrt{\frac{\gamma}{3}}\varepsilon+\frac{\gamma}{3}\varepsilon^2\right)}{3\gamma\varepsilon^2\left(|\alpha_j|=|+\sqrt{\frac{\gamma}{3}}\varepsilon\right)^2}\ln(2/\delta') \\
&= \frac{2}{3\gamma\varepsilon^2}\ln(2/\delta'),
\end{aligned} \tag{B3}$$

and hence it suffices to choose

$$k = \frac{2}{3\gamma\varepsilon^2}\ln(2/\delta').$$

Letting $\delta'=\delta/L$, the union bound implies that for

$$k = \frac{2}{3\gamma\varepsilon^2}\ln(2L/\delta),$$

all estimates $\tilde{p}_j$ satisfy $|\tilde{p}_j-p_j|\leq\varepsilon_j$. Now bound the distance between $|\tilde{\alpha}_j|$ and $|\alpha_j|$. First,

$$\begin{aligned}
|\tilde{\alpha}_j|-|\alpha_j| &\leq \sqrt{p_j+\varepsilon}-|\alpha_j| \\
&= \sqrt{|\alpha_j|^2+\sqrt{3\gamma}|\alpha_j|\varepsilon+\gamma\varepsilon^2}-|\alpha_j| \\
&\leq \left(|\alpha_j|+\sqrt{\gamma}\varepsilon\right)-|\alpha_j| \\
&= \sqrt{\gamma}\varepsilon.
\end{aligned} \tag{B4}$$

[0285] Next, bound $|\alpha_j|-|\tilde{\alpha}_j|$. If $p_j\leq\varepsilon_j$ then

$$|\alpha_j|^2 \leq \sqrt{3\gamma}|\alpha_j|\varepsilon+\gamma\varepsilon^2 \Leftrightarrow |\alpha_j| \leq \frac{(\sqrt{3}+\sqrt{7})\sqrt{\gamma}}{2}\varepsilon, \tag{B5}$$

while if $p_j>\varepsilon_j$,

-continued

$$\begin{aligned}
|\alpha_j|-|\tilde{\alpha}_j| &\leq |\alpha_j|-\sqrt{p_j-\varepsilon_j} \\
&= |\alpha_j|-\sqrt{|\alpha_j|^2-\sqrt{3\gamma}|\alpha_j|\varepsilon-\gamma\varepsilon^2} \\
&< \frac{(\sqrt{3}+\sqrt{7})\sqrt{\gamma}}{2}\varepsilon,
\end{aligned} \tag{B6}$$

which follows because the function $f(x)=x-\sqrt{x^2-\sqrt{3x}-1}$ has its maximum at

$$f\left(\frac{\sqrt{3}+\sqrt{7}}{2}\right) = \frac{\sqrt{3}+\sqrt{7}}{2}.$$

Therefore with the choice

$$\gamma = \left(\frac{\sqrt{3}+\sqrt{7}}{2}\right)^{-2},$$

we can guarantee that $||\tilde{\alpha}_j|-|\alpha_j||\leq\varepsilon$, which corresponds to

$$k = \frac{2}{3\gamma\varepsilon^2}\ln(2L/\delta) = \frac{5+\sqrt{21}}{3\varepsilon^2}\ln(2L/\delta) \tag{B7}$$

measurements.

[0286] Proof of proposition 4. Define

$$\varepsilon' = \frac{1}{\sqrt{2L}}\varepsilon\sqrt{1-\varepsilon^2/4}.$$

Then k $2.875\varepsilon'^{-2}$ ln(6L/δ). Consider the following three assertions:

[0287] 1. The estimates p_i satisfy $|\sqrt{p_i}-|\tilde{v}_i|\sqrt{p}|\leq\varepsilon'/3$ for all i.

[0288] 2. The estimates $p_i^+=k_i^+/k$ satisfy

$$\left|\sqrt{p_i^+}-\frac{|\sqrt{p}\,\tilde{v}_i+\sqrt{p_i'}|}{2}\right| \leq \varepsilon'/3,$$

and the estimates $p_i^-=k_i^-/k$ satisfy

$$\left|\sqrt{p_i^-}-\frac{|\sqrt{p}\,\tilde{v}_i+\sqrt{p_i'}|}{2}\right| \leq \varepsilon'/3,$$

for all i.

[0289] 3. The actual amplitudes $\sqrt{p_i'}$ of the state created in the second step satisfy $|\sqrt{p_i'}-\sqrt{p_i}|\leq\varepsilon_{tsp}$.

[0290] From proposition 3, we know that Assertion 1 holds with probability at least 1-δ/3, and Assertion 2 holds with probability at least 1−2δ/3. Therefore both assertions hold with probability at least 1−δ. Moreover, Assertion 3 holds by assumption. From here on it is assumed that all three assertions hold.

[0291] Let a_i be the real part and b_i be the imaginary part of the quantity $\sqrt{p}\tilde{v}_i$. Let $r_i^+=|\sqrt{p}\tilde{v}_i+\sqrt{p_i}|$, and $r_i^-=|\sqrt{p}\tilde{v}_i-\sqrt{p_i}|$. Note that r_i^+ and r_i^- are proportional to the absolute value of the ideal amplitudes of the state created in eq. (50). One can show that

$$a_i = \frac{(r_i^+)^2 - (r_i^-)^2}{4\sqrt{p_i}}. \tag{B8}$$

Define $f_i(x, y)=(x^2-y^2)/\sqrt{p_i}$; then $a_i=f_i(r_i^+/2, r_i^-/2)$. Note that the estimates $\sqrt{p_i^{\pm}}$ give good approximations of $r_i^+/2$ and $r_i^-/2$:

$$\left|\sqrt{p_i^{\pm}} - \frac{r_i^{\pm}}{2}\right| \le \frac{\varepsilon'}{3}\varepsilon + \frac{\varepsilon_{tsp}}{2}, \tag{B9}$$

which follows from Assertions 2 and 3. The amplitudes di that define the estimate output by the tomography algorithm are given in eq. (52), which can now be rewritten as

$$\tilde{a}_i = \begin{cases} 0, & \sqrt{p_i} \le \frac{2}{3}\varepsilon' + \varepsilon_{tsp}; \text{ else} \\ \min\left(\sqrt{p_i}, f_i\left(\sqrt{p_i^+}, \sqrt{p_i^-}\right)\right), & f_i\left(\sqrt{p_i^+}, \sqrt{p_i^-}\right) \ge 0 \\ \max\left(-\sqrt{p_i}, f_i\left(\sqrt{p_i^+}, \sqrt{p_i^-}\right)\right), & f_i\left(\sqrt{p_i^+}, \sqrt{p_i^-}\right) < 0 \end{cases}. \tag{B10}$$

[0292] We prove that the $\tilde{a}_i$ values approximate the a_i values, specifically

$$|\tilde{a}_i - a_i| \le \varepsilon' + \varepsilon_{tsp} + |b_i|. \tag{B11}$$

We will prove the claim above using a case-by-case analysis. Assume that $a_i \ge 0$; the case $a_i < 0$ will proceed similarly.

[0293] First, consider the case

$$\sqrt{p_i} \le \frac{2}{3}\varepsilon' + \varepsilon_{tsp}.$$

In this case $\tilde{a}_i=0$ and

$$a_i \le \sqrt{p}\,|\tilde{v}_i| \le \sqrt{p_i} + \frac{\varepsilon'}{3} \le \varepsilon' + \varepsilon_{tsp}, \text{ so } |a_i| \le \varepsilon + \varepsilon_{tsp}.$$

[0294] Second, consider the case $f_i(\sqrt{p_i^+}, \sqrt{p_i^-}) \ge a_i$. From the definition of $\tilde{a}_i$ and Assertion 1, we have

$$\tilde{a}_i \le \sqrt{p_i} \le \sqrt{p}\,|\tilde{v}_i| + \frac{\varepsilon'}{3},$$

and thus

$$\tilde{a}_i - a_i \le \sqrt{p}\,|\tilde{v}_i| - a_i + \frac{\varepsilon'}{3} = \sqrt{a_i^2 + b_i^2} - a_i + \frac{\varepsilon'}{3} \le |b_i| + \frac{\varepsilon'}{3}. \tag{B12}$$

We also have (again invoking Assertion 1)

$$a_i - \tilde{a}_i \le a_i - \sqrt{p_i} \le a_i - \sqrt{p}\,|\tilde{v}_i| + \frac{\varepsilon'}{3} \le \frac{\varepsilon'}{3} \text{ and thus,} \tag{B13}$$

$$|a_i - \tilde{a}_i| \le |b_i| + \frac{\varepsilon'}{3}.$$

[0295] Additionally, consider the case $f_i(\sqrt{p_i^+}, \sqrt{p_i^-}) < a_i$. Defining

$$\tilde{\varepsilon} = \frac{2}{3}\varepsilon' + \varepsilon_{tsp},$$

we can lower bound $f_i(\sqrt{p_i^+}, \sqrt{p_i^-})$: **[text missing or illegible when filed]**

$$\text{⍰} = a_i - \tilde{\varepsilon}\frac{r_i^+ + r_i^-}{2\sqrt{p_i}}. \tag{B14}$$

⍰ indicates text missing or illegible when filed

Here in the second line we used eq. (B9) and the fact that

$$r_i^+ \ge \sqrt{p_i} \ge \frac{2}{3}\varepsilon' + \varepsilon_{tsp}.$$

We now upper bound $r_i^+ + r_i^-$:

$$\begin{aligned} r^+ + r^- &= \sqrt{\left(a_i + \sqrt{p_i}\right)^2 + b_i^2} + \sqrt{\left(a_i - \sqrt{p_i}\right)^2 + b_i^2} \\ &\le \left|a_i + \sqrt{p_i}\right| + \left|a_i - \sqrt{p_i}\right| + 2|b_i| \\ &= 2\max\left(a_i, \sqrt{p_i}\right) + 2|b_i| \\ &\le 2\left(\sqrt{p_i} + \varepsilon'/3 + |b_i|\right), \end{aligned} \tag{B15}$$

where in the fourth line we used $a_i \le \sqrt{a_i^2+b_i^2} = \sqrt{p}\,|\tilde{v}_i| \le \sqrt{p_i} + \varepsilon'/3$ (Assertion 1).

Therefore,

[0296]

$$\begin{aligned} f_i\left(\sqrt{p_i^+},\sqrt{p_i^-}\right) &= a_i - \tilde{\varepsilon}\frac{r_i^+ + r_i^-}{2\sqrt{p_i}} \\ &\geq a_i - \tilde{\varepsilon}\frac{2\left(\sqrt{p_i} + \varepsilon'/3 + |b_i|\right)}{2\sqrt{p_i}} \\ &= a_i - \tilde{\varepsilon} - \frac{\tilde{\varepsilon}}{\sqrt{p_i}}\left(\varepsilon'/3 + |b_i|\right) \\ &\geq a_i - (\varepsilon' + \varepsilon_{tsp} + |b_i|), \end{aligned} \tag{B16}$$

where in the fourth line we used $\tilde{\varepsilon}/\sqrt{p_i}\leq 1$. This implies

$$|\tilde{a}_i - a_i| = a_i - \tilde{a}_i \leq a_i - \min\left(f_i\left(\sqrt{p_i^+},\sqrt{p_i^-}\right),\sqrt{p_i}\right) \leq \varepsilon' + \varepsilon_{tsp} + |b_i|. \tag{B17}$$

Here, we used $a_i-\sqrt{p_i}\leq\sqrt{p}|\tilde{v}_i|-\sqrt{p_i}\leq\varepsilon'/3$.

[0297] We've shown that $|\tilde{a}_i-a_i|\leq\varepsilon'+\varepsilon_{tsp}+|b_i|$ for all cases. Therefore,

$$\begin{aligned} \|\tilde{a} - a\|_2^2 &\leq \sum_i\left[(\varepsilon' + \varepsilon_{tsp})^2 + 2|b_i|(\varepsilon' + \varepsilon_{tsp}) + b_i^2\right] \\ &\leq L(\varepsilon' + \varepsilon_{tsp})^2 + 2(\varepsilon' + \varepsilon_{tsp})\sqrt{L\sum_i b_i^2} + \sum_i b_i^2 \\ &= \left(\sqrt{L}\,(\varepsilon' + \varepsilon_{tsp}) + \sqrt{\sum_i b_i^2}\right)^2, \end{aligned} \tag{B18}$$

and hence

$$\begin{aligned} \left\|\tilde{a} - \sqrt{p}\,v\right\|_2 &\leq \|\tilde{a} - a\|_2 + \left\|a - \sqrt{p}\,v\right\|_2 \\ &\leq \sqrt{L}\,(\varepsilon' + \varepsilon_{tsp}) + \sqrt{\sum_i b_i^2} + \sqrt{\sum_i\left(\sqrt{p}v_i - a_i\right)^2} \\ &\leq \sqrt{L}\,(\varepsilon' + \varepsilon_{tsp}) + \sqrt{2p}\,\varepsilon_{QLSP}, \end{aligned} \tag{B19}$$

where we used $\Sigma_i\ ((v_i-a_i/\sqrt{p})^2+b_i^2/p)\leq\varepsilon_{QLSP}^2$. Since $\tilde{v}'\propto\tilde{a}$, for some proportionality factor λ we have $\|\lambda\tilde{v}'-v\|\sqrt{2L}(\varepsilon'+\varepsilon_{tsp})+\sqrt{2}\varepsilon_{QLSP}$, where we used $p\leq\frac{1}{2}$. A bit of geometry will show that if

$$\|c - d\|_2 \leq \gamma < 1 \text{ and } \|d\|_2 = 1, \text{ then } \left\|\frac{c}{\|c\|_2} - d\right\|_2 \leq g(\gamma) \equiv 2\sin\left(\frac{1}{2}\sin^{-1}\gamma\right) = \sqrt{1+\gamma} - \sqrt{1-\gamma}.$$

Applying this with $c=\lambda\tilde{v}'$ and $d=v$ we obtain

$$\|\tilde{v}' - v\|_2 + \left(\sqrt{2L}\,\varepsilon_{tsp} + \sqrt{2}\,\varepsilon_{QLSP}\right)\frac{dg}{dx}\Big|_{x=\sqrt{2L}(\varepsilon'+\varepsilon_{tsp})+\sqrt{2}\varepsilon_{QLSP}} < \varepsilon + 1.58\sqrt{L}\,\varepsilon_{tsp} + 1.58\varepsilon_{QLSP} \tag{B20}$$

as claimed. In the second inequality we used the convexity of g; in the third inequality we used the fact that $g(\sqrt{2L}\varepsilon')=\varepsilon$, $\sqrt{2L}(\varepsilon'+\varepsilon_{tsp})+\sqrt{2}\varepsilon_{QLSP}<\varepsilon+\sqrt{2}\varepsilon_{tsp}+E\sqrt{2}\varepsilon_{QLSP}\leq\frac{1}{2}$, and $\sqrt{2}g'(\frac{1}{2})<1.58$.

Additional Information C: Null Space Matrix for Portfolio Optimization

[0298] In section III C, an inexact-feasible interior point method was described that uses as input a matrix B with columns that form a basis for the null space of the feasibility equations for the self-dual SOCP that appear in eq. (19). A straightforward way to find such a B in general may be to perform a QR decomposition of the constraint matrix, costing classical $O(N^3)$ runtime (or, using techniques for fast matrix multiplication, between $O(N^2)$ and $O(N^3)$ time). The upshot is that B can only be computed once and does not change with each iteration of the algorithm, but depending on other parameters of the problem, this classical runtime may dominate the overall complexity. Alternatively, in many specific cases including ours, a valid matrix B can be determined by inspection. For example, suppose that we have a $(N-K)\times N$ matrix Q_A with full column rank for which $AQ_A=0$, a $K\times(K-1)$ matrix P with full column rank for which $\bar{b}^TP=0$, and a point x_0 for which $Ax_0=b$. Then, letting $\gamma=b^T\bar{b}/\|b\|^2$, a valid choice for B is

$$B = \begin{array}{c} \\ x \\ y \\ \tau \\ \theta \\ s \\ \kappa \end{array}\begin{pmatrix} 0 & Q_A & e & x_0 \\ P & \bar{b}\dfrac{\bar{c}^TQ_A}{\|\bar{b}\|^2} & -\dfrac{(r+1)}{\|\bar{b}\|^2}\bar{b} & \dfrac{\bar{c}^Tx_0 - \bar{z}}{\|\bar{b}\|^2}\bar{b} \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \\ -A^TP & -A^T\bar{b}\dfrac{\bar{c}^TQ_A}{\|\bar{b}\|^2} & \dfrac{(r+1)}{\|\bar{b}\|^2}A^T\bar{b} + e & \dfrac{-\bar{c}^Tx_0 + \bar{z}}{\|\bar{b}\|^2}A^T\bar{b} + c \\ b^TP & (\gamma-1)c^TQ_A - \gamma e^TQ_A & 1-\gamma(r+1) & -\gamma\bar{z}(r+1)c^Tx_0 - \gamma e^Tx_0 \end{pmatrix} \tag{C1}$$

The leftmost column in the above block matrix corresponds to K−1 basis vectors formed by choosing y to be a vector perpendicular to $\bar{b}$ and x=0, τ=θ=0. The second column corresponds to N−K vectors formed by choosing x to be in the null space of A, and letting τ=θ=0, with

$$y = \frac{\bar{c}^{\top} x}{\|\bar{b}\|^2} \bar{b}.$$

The third column corresponds to the vector formed by choosing x=e, τ=θ=1, and then

$$y = \frac{-(r+1)}{\|\bar{b}\|^2} \bar{b}.$$

The final column corresponds to choosing $x=x_0$, τ=1, θ=0, and

$$y = \frac{\bar{c}^{\top} x_0 - \bar{z}}{\|\bar{b}\|^2} \bar{b}.$$

In each case, the choices of x, y, τ, and θ uniquely determine the values of s and κ. Note that in practice the second and fourth block rows of B can be ignored because in eq. (22) they are left-multiplied by a matrix whose second and fourth block columns are zero.

[0299] What remains is to specify P, Q_A, and x_0 for the case of portfolio optimization, given in eq. (10). Finding a valid matrix P is straightforward. Note that from eq. (10), we have b=(1; $\bar{w}$+ζ; $\bar{w}$−ζ; 0). For j=1, . . . , 2n, let p_j have a 1 in its first entry, and $a-1/b_{j+1}$ in its (j+1)th entry, with zeros elsewhere. For j=2n+1, . . . , 2n+m, let p_j have a single 1 in its (j+1)th entry, and zeros elsewhere. Thus, the p_j are independent and $b^T p_j=0$ for all j. Then define the matrix P by $P=(p_1, \ldots, p_{2n+m})$. Similarly, the columns of a valid matrix Q_A can be generated as follows: given a choice of w such that $1^T w=0$, choose ϕ=−w, ρ=w, t=0, and η=Mw. As there are n−1 linearly independent choices of w (e.g. the vectors (1; −1; 0; 0; . . . ; 0), (0; 1; −1; 0; . . . ; 0), (0; 0; 1; −1; . . . ; 0), etc.), this leads to n−1 linearly independent columns of Q_A. A final nth column can be formed by choosing t=1 and w=ϕ=ρ=0 and η=0. Next, the point x_0 can be chosen by letting w=$\bar{w}$, =ρ=ζ, t=0, and η=M$\bar{w}$.

Additional Information D: Alternative Search Directions

[0300] The solution (Δx; Δy; Δτ; Δθ; Δs; Δκ) to the Newton systems in eqs. (19) and (22) is one possible search direction for the interior point method. Alternative search directions can be found by applying a scale transformation to the convex set. For the k-dimensional second-order cone $\mathcal{Q}^k$, define the set

$$\mathcal{G}^k = \left\{ \lambda T : \lambda > 0,\ T^{\top} \begin{pmatrix} 1 & 0 \\ 0 & -I \end{pmatrix} T = \begin{pmatrix} 1 & 0 \\ 0 & -I \end{pmatrix} \right\}. \tag{D1}$$

For the product $\mathcal{Q}$ of multiple cones, let the set $\mathcal{G}$ include of direct sums of entries from $\mathcal{G}^k$. This definition implies that the matrices G∈ $\mathcal{G}$ map the set $\mathcal{Q}$ onto itself. Thus for a fixed choice G∈ $\mathcal{G}$, consider a change of variables $x'=G^T x$, $s'=G^{-1}s$, y'=y. Let X' and S' be the arrowhead matrices for x' and s', and, following the same logic as above, the following Newton system is arrived at:

$$\begin{pmatrix} S'G^T & 0 & 0 & 0 & X'G^{-1} & 0 \\ 0 & 0 & \kappa & 0 & 0 & \tau \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \\ \Delta \tau \\ \Delta \theta \\ \Delta s \\ \Delta \kappa \end{pmatrix} = \begin{pmatrix} \sigma \mu e - X'S'e \\ \sigma \mu - \kappa \tau \end{pmatrix}. \tag{D2}$$

The solution to this linear set of equations (along with the feasibility equations of eq. (19)) is distinct for different choices of G. The choice G=I recovers eq. (22) and is called the Alizadeh-Haeberly-Overton (AHO) direction. Note that the IPM can reduce the duality gap by a constant factor after $O(\sqrt{r})$ iterations for any choice of G. However, some choices of G can yield additional potentially desirable properties; for example, the Nesterov-Todd search direction scales the cone such that x'=s'. However, in the numerical simulations of the QIPM, no obvious benefits of choosing a search direction were observed other than the AHO direction.

Additional Information E: Numerical Results for Feasible QIPMs

[0301] Section VI presented numerical results for the "II-QIPM," for which intermediate points could be infeasible. Here we also present some results for two variants of the "feasible" QIPM, denoted by "IF-QIPM" and "IF-QIPM-QR," as summarized in table II. The IF-QIPM uses the null space basis B outlined in Additional Information C, whereas the IF-QIPM-QR version uses a null space basis B determined using a QR decomposition. In all cases, the algorithm was simulated for enough iterations to reduce the duality gap to 10^{-3}, whereas for the II-QIPM it was simulated down to 10^{-7}.

[0302] In FIGS. **13** to **15**, the analogous results for the feasible IPMs are presented as were displayed in FIGS. **9** to **11** for the infeasible case. The IF-QIPM-QR has the best performance, though this should be weighed against the fact that an expensive QR decomposition was classically precomputed to implement this method. However, the advantage of the IF-QIPM-QR method is not large enough for any of the qualitative conclusions in section VII to change. The IF-QIPM method has the worst performance, which may be due to the fact that the null-space basis found by inspection turns out to be a very ill-conditioned matrix (its condition number was observed to be in the vicinity of 1000). Additionally, the IF-QIPM appears to have the largest instance-to-instance variation of any of the methods, leading to lower quality numerical fits.

[0303] FIG. **13** includes two plots of the Median Frobenius condition number for 128 randomly sampled stock portfolios from the DWCF index as a function of portfolio size for duality gaps of 1.0, 0.1, 0.01, and 0.001. The error bars show the 68th percentile, which corresponds to one standard deviation if the distribution is Gaussian. A linear trend appears to work quite well for the IF-QIPM-QR case, but the IF-QIPM is quite noisy. For each duality gap, a power-law fit of the form an^b is also ploted. The values of b are in table XII.

[0304] FIG. **14** includes two plots of the Median value of the square of the required inverse tomography precision used to remain in the neighborhood of the central path for 128 randomly sampled stock portfolios from the DWCF index as a function of portfolio size for duality gaps of 1.0, 0.1, 0.01, and 0.001. The error bars show the 68th percentile, which corresponds to one standard deviation if the distribution is Gaussian. For each duality gap, a linear fit on the log-log data is also plotted. The corresponding slope is in table XIII.

[0305] FIG. **15** includes two plots of the Median value of the estimated algorithm scaling factor computed as the median of $n^{1.5}\kappa_F/\xi^2$ for 128 randomly sampled stock portfolios from the DWCF index as a function of portfolio size for duality gaps of 1.0, 0.1, 0.01, and 0.001. The error bars show the 68th percentile, which corresponds to one standard deviation if the distribution is Gaussian. For each duality gap, a linear fit on the log-log data is also ploted. The corresponding slope is in table XIV.

[0306] TABLE XII sjpws fit parameters for the Frobenius condition number for the four horizontal-axis locations considered on the scaling plot of FIG. **13**. The uncertainties correspond to one standard deviation errors on the parameter estimates from the fit. We note that both versions have similar empirical scaling, although the fits are better for IF-QIPM-QR. The constant prefactors are superior for the IF-QIPM-QR version, but calculating the QR decomposition requires a one-time classical cost proportional to $\mathcal{O}(L^3)$.

[0307] TABLE XIII shows fit parameters for the square of the inverse of the required tomography precision to stay near the central path, corresponding to FIG. **14**. The uncertainties correspond to one standard deviation errors on the parameter estimates from the fit.

[0308] TABLE XIV shows estimated scaling of the quantum algorithm as a function of portfolio size for the two feasible versions of the quantum algorithm, corresponding to FIG. **15**. The uncertainties correspond to one standard deviation errors on the parameter estimates from the fit.

TABLE XII

Dual. Gap	IF-QIPM	IF-QIPM-QR
1.0	$\kappa_F(G)\sim n^{0.57\pm0.60}$	$\kappa_F(G)\sim n^{0.228\pm0.002}$
0.1	$\kappa_F(G)\sim n^{0.58\pm0.28}$	$\kappa_F(G)\sim n^{0.66\pm0.03}$
0.01	$\kappa_F(G)\sim n^{0.81\pm0.53}$	$\kappa_F(G)\sim n^{0.73\pm0.03}$
0.001	$\kappa_F(G)\sim n^{1.01\pm0.77}$	$\kappa_F(G)\sim n^{0.98\pm0.04}$

TABLE XIII

Dual. Gap	IF-QIPM	IF-QIPM-QR
1.0	$\xi^{-2}\sim\mathcal{O}(n^{-0.01\pm0.02})$	$\xi^{-2}\sim\mathcal{O}(n^{-0.11\pm0.07})$
0.1	$\xi^{-2}\sim\mathcal{O}(n^{-0.99\pm0.41})$	$\xi^{-2}\sim\mathcal{O}(n^{-0.46\pm0.11})$
0.01	$\xi^{-2}\sim\mathcal{O}(n^{0.53\pm0.91})$	$\xi^{-2}\sim\mathcal{O}(n^{0.89\pm0.15})$
0.001	$\xi^{-2}\sim\mathcal{O}(n^{0.93\pm0.66})$	$\xi^{-2}\sim\mathcal{O}(n^{0.90\pm0.15})$

TABLE XIV

Dual. Gap	IF-QIPM	IF-QIPM-QR
1.0	$\mathcal{O}(n^{1.41\pm0.01})$	$\mathcal{O}(n^{2.07\pm0.15})$
0.1	$\mathcal{O}(n^{1.23\pm0.40})$	$\mathcal{O}(n^{1.77\pm0.15})$
0.01	$\mathcal{O}(n^{2.87\pm0.91})$	$\mathcal{O}(n^{3.13\pm0.18})$
0.001	$\mathcal{O}(n^{3.54\pm0.64})$	$\mathcal{O}(n^{3.50\pm0.10})$

[0309] The above sections describe example embodiments for purposes of illustration only. Any features that are described as essential, necessesary, required, important, or otherwise implied to be required should be interpreted as only being required for that embodiment and are not necessarily included in other embodiments.

[0310] Additionally, the above sections often use the phrase "we" (and other similar phases) to reference an entity that is performing an operation (e.g., a step in an algorithm). These phrases are used for convenience. These phrases may refer to a computing system (e.g., computing system **1700**) that is performing the described operations.

IX. EXAMPLE METHODS

[0311] FIG. **16** is a flowchart of an example method **1600**, specifically a quantum interior point method (QIPM), for solving a second-order cone program (SOCP) instance using a quantum computing system, according to one or more embodiments. As previously described, a QIPM includes one or more quantum operations that provide a computational advantage over other methods of solving SOCP instances. Specifically, the algorithmic complexity for the quantum method can be better (in some ways) than the classical method. The classical method scales as $n^{3.5}$ log $(1/\in)$ and the quantum algorithm scales as $n^{1.5}k_F$ log $(1/\in)/\xi^2$.

[0312] In the example of FIG. **16**, the method **1600** is performed from the perspective of a computing system (e.g., **1700**) including a quantum computing system (e.g., **1720**). The method **1600** can include greater or fewer steps than described herein. Additionally, the steps can be performed in different order, or by different components than described herein. In some embodiments, the method **1600** is performed by the computing system executing code stored on a (e.g., non-transitory) computer-readable storage medium that causes the computing system to perform the steps of method **1600**. Algorithm 1 is an example of method **1600**. In some embodiments, one or more steps of the method can be used to determine solutions for optimization problems other than a SOCP.

[0313] At step **1610**, the computing system receives the SOCP instance (e.g., see input of Algorithm 1). The SOCP instance may include quantities (A, b, c) as described with respect to eq. 10. The computing system may also receive a list of cone sizes $(N_1, \ldots, N_r)$ and a tolerance $\in$ (also referred to as precision $\in$). In some embodiments, computing system receives the SOCP instance with N>0 variables, K>0 linear constraints, r>0 second-order cone constraints, and the tolerance parameter E, where matrix A of the SOCP instance is a K×N matrix encoding linear constraints, vector b is a length-K vector encoding right-hand side of linear constraints, and vector c is a length-Nvector encoding an (e.g., objective) function (e.g., see eq. (5)).

[0314] At step **1620**, the computing system defines a Newton system for the SOCP instance by constructing matrix G and vector h (e.g., steps **4** and **5** of Algorithm 1). Matrix G and vector h describe constrains for a linear system

Gu=h based on the SOCP instance. Defining the system in this way enables the benefits of the quantum approach over classical approaches to be realized.

[0315] At step **1630**, the computing system preconditiones matrix G and vector h via row normalization to reduce a condition number of matrix G (e.g., steps **6-10** of Algorithm 1). Among other advantages, preconditioning matrix G and vector h reduces their computational complexity (due to the reduced condition number), which reduces computational time.

[0316] Preconditioning matrix G and vector h may include determining a diagonal matrix D where at least one (e.g., each or every) entry D_{ii} is equal to the norm of row i of matrix G. After determining the the matrix D, matrix G and vector h may be redefined according to: $G=D^{-1}G$ and $h=D^{-1}h$, where $D^{-1}G$ has a condition number less that the condition number of previous/original matrix G. More information on preconditioning can be found above at, for example, sections I B and V A.

[0317] At step **1640**, the computing system iteratively determines u until a predetermined iteration condition is met. For example, see steps **13-18** of Algorithm 1, where in this context u is (x'; y'; τ'; θ'; s'; $\varkappa'$) and the predetermine iteration condition is (x'; y'; τ'; θ'; s'; $\varkappa'$) $\in \mathcal{N}(\gamma)$). The iterations may include:

[0318] The iterations of step **1640** may include causing the quantum computing system to apply matrix G and vector h to a quantum linear system solver (QLSS) to generate a quantum state (e.g., part of step **15** of Algorithm 1). Causing the quantum computing system to apply matrix G and vector h to the QLSS to generate the quantum state may include k>0 applications of the QLSS and k controlled-applications of the QLSS to generate the quantum state. Causing the quantum computing system to apply matrix G and vector h to the QLSS may include causing the quantum computing system to execute a quantum circuit. More information on the QLSS can be found above at, for example, section IV B.

[0319] The QLSS may operate on a block encoded version of matrix G and a state-prepared version of vector h. To do this, the method **1600** may include causing matrix G to be block encoded onto the quantum computing system and the vector h to be state encoded onto the quantum computing system. As previously described, block-encodings enable quantum algorithms (the QLSS in this case) to coherently access classical data (G and h in this case). More information on block encoding can be found above at, for example, sections I B and IV C.

[0320] The interations of step **1640** may include causing the quantum computing system to perform quantum state tomography on the quantum state (e.g., part of step **15** (e.g., step **27**) of Algorithm 1). Causing the quantum computing system to perform quantum state tomography may include causing the quantum computing system to execute a quantum circuit. Conceptually, the quantum state tomography "reads out" the results of calculations "stored" in the quantum state.

[0321] The output of the of the quantum state tomography may be a unit vector $\tilde{v}'$ indicating an iteration direction for u (e.g., $\tilde{v}'$ is (Δx; Δy; Δτ; Δθ; Δs; Δ$\varkappa$) of Algorithm 1). The unit vector $\tilde{v}'$ may be characterized by $\|\tilde{v}'-v\|\leq\xi$ with probability equal to or greater than a predetermined probability threshold (e.g., probability at least 1−δ, where δ is, for example, 0.1), where $v\propto G^{-1}h$ and ξ is a tomography precision parameter (e.g., see steps **24** and **28** of Algorithm 1).

[0322] In some embodiments, the quantum state tomography is performed according to a tomography precision parameter ξ that decreases with each new iteration (e.g., see ξ in Algorithm 1). The tomography precision parameter ξ may decrease by ξ/2 with each new iteration (see e.g., step **14** of Algorithm 1). More information on quantum tomography can be found above at, for example, sections IV D and V A.

[0323] The interations of step **1640** may include updating a value of u based on a current value of u and the output of the quantum state tomography (e.g., steps **16-17** of Algorithm 1). For example, updating the value of u may include determining an updated step length σ≠0 based on the unit vector $\tilde{v}'$, and adding the current value of u to the product of: (1) the updated step length σ and (2) the unit vector $\tilde{v}'$ (e.g., see steps **16-17** of Algorithm 1, where "step length" refers to the updated step length, σ refers to the current step length, the current value of u is (x; y; τ; θ; s; $\varkappa$), and the updated value of u is (x'; y'; τ'; θ'; s'; $\varkappa'$). As used herein, a value of a vector (e.g., u or $\tilde{v}'$) may refer to the value of a single component or of multiple components of that vector.

[0324] In some embodiments, the updated step length σ is given by by:

$$\frac{\mu(x,\tau,s,\mathcal{H})(1-\sigma)(r+1)}{-(\Delta x)^{\top}s-(\Delta s)^{\top}x-(\Delta,\mathcal{H}\tau-(\Delta\tau)\mathcal{H}}$$

where Δx, Δs, Δ$\varkappa$, and Δτ are components of the unit vector $\tilde{v}'$; x, s, $\varkappa$, and τ are components of current u; μ(x, τ, s, $\varkappa$) is a duality gap; σ is the current step length; and r is the number of second-order cone constraints of the SOCP instance. The updated step length may be determined such that a duality gap μ of the updated value of u is a factor of σ (the current step length) smaller than the current value of u within a second order deviation in the step length σ. The duality gap μ may describe a difference between the current value of u and an exact solution to the linear system Gu=h. More information on step length can be found above at, for example, sections I B, V A, and IV A.

[0325] At step **1650**, the computing system determines a solution to the SOCP instance based on the updated value of u (e.g., steps **19-21** of Algorithm 1). In the example of Algorithm 1, the solution to the SOCP instance (within precision ∈) is vector x of (x; y; τ; θ; s; $\varkappa$)).

[0326] In some embodiments, steps **1620-1640** are repeated iteratively (e.g., see loop of steps **3-21** of Algorithm 1). For example, a target precision F for the solution to the SOCP instance is received (referred to as "tolerance" or "precision" in Algorithm 1), and steps **1620-1640** are repeated and a duality gap μ is iteratively updated based on the output of the quantum state tomography (e.g., the new duality gap μ is the product of current duality gap μ and the updated step length σ (e.g., see also step **20** of Algorithm 1) until the duality gap μ is less than the target precision F (e.g., see condition at step **3** of Algorithm 1), where the duality gap μ describes a difference between the current value of u and an exact solution to the linear system Gu=h. In these embodiments, matrix G and vector h may be constructed (in step **1620**) further based on the (e.g., updated) value of u (e.g., u is (x; y; τ; θ; s; $\varkappa$) and G and h depend at least on one of these quantities in steps **4** and **5** of Algorithm 1).

X. DESCRIPTION OF A COMPUTING SYSTEM

[0327] Embodiments described above may be implemented using one or more computing systems. Example computing systems are described below.

[0328] FIG. **17**A is a block diagram that illustrates a computing system **1700**, according to some embodiments. In the example of FIG. **17**A, the computing system **1700** includes a classical computing system **1710** (also referred to as a non-quantum computing system) and a quantum computing system **1720**, however a computing system may just include a classical computing system or a quantum computing system. An embodiment of the classical computing system **1710** is described further with respect to FIG. **18**. While the classical computing system **1710** and quantum computing system **1720** are illustrated together, they may be physically separate systems. For example, FIG. **17**B illustrates an example cloud computing architecture where the computing system **1710** and the quantum computing system **1720** communicate via a network **1757**. The computing system **1700** may include different, additional, or fewer elements than illustrated (e.g., multiple quantum computing systems **1720**).

[0329] The classical computing system **1710** may operate or control the quantum computing system **1720**. For example, the classical computing system **1710** causes the quantum computing system **1720** to perform one or more operations, such as to execute a quantum algorithm or quantum circuit (e.g., the classical computing system **1710** generates and transmits instructions for the quantum computing system **1720** to execute a quantum algorithm or quantum circuit). For example, the computing system **1710** causes the quantum computing system **1720** to perform one or more steps of method **1600**. Although only one classical computing system **1710** is illustrated in FIG. **17**A, any number of classical computing system **1710** or other external systems may be connected to the quantum computing system **1720**.

[0330] FIG. **17**C is a block diagram that illustrates the quantum computing system **1720**, according to some embodiments. The quantum computing system **1720** includes any number of quantum bits ("qubits") **1750** and associated qubit controllers **1740**. As illustrated in FIG. **17**D, the qubits **1750** may be in a qubit register **1755** of the quantum computing system **1720** (or multiple registers). Qubits are further described below. A qubit controller **1740** is a module that controls one or more qubits **1750**. A qubit controller **1740** may include one or more classical processors such as one or more CPUs, one or more GPUs, one or more FPGAs, or some combination thereof. A qubit controller **1740** may perform physical operations on one or more qubits **1750** (e.g., it can perform quantum gate operations on a qubit **1740**). In the example of FIG. **17**C, a separate qubit controller **1740** is illustrated for each qubit **1750**, however a qubit controller **1740** may control multiple (e.g., all) qubits **1750** of the quantum computing system **1720** or multiple controllers **1740** may control a single qubit. For example, the qubit controllers **1740** can be separate processors, parallel threads on the same processor, or some combination of both.

[0331] FIG. **17**E is a flow chart that illustrates an example execution of a quantum routine on the computing system **1700**. The classical computing system **1710** generates **1760** a quantum program to be executed or processed by the quantum computing system **1720**. The quantum program may include instructions or subroutines to be performed by the quantum computing system **1720**. In an example, the quantum program is a quantum circuit. The quantum computing system **1720** executes **1765** the program and computes **1770** a result (referred to as a shot or run). Computing the result may include performing a measurement of a quantum state generated by the quantum computing system **1720** that resulted from executing the program. Practically, this may be performed by measuring values of one or more of the qubits **1750**. The quantum computing system **1720** typically performs multiple shots to accumulate statistics from probabilistic execution. The number of shots and any changes that occur between shots (e.g., parameter changes) may be referred to as a schedule. The schedule may be specified by the program. The result (e.g., quantum state data) (or accumulated results) is recorded **1775** by the classical computing system **1710**. Results may be returned after a termination condition is met (e.g., a threshold number of shots occur). The classical computing system **1710** may determine a quantity based on the received results.

[0332] The quantum computing system **1720** exploits the laws of quantum mechanics in order to perform computations. A quantum processing device, a quantum computer, a quantum processor system, and a quantum processing unit (QPU) are each examples of a quantum computing system. The quantum computing system **1700** can be a universal or a non-universal quantum computing system (a universal quantum computing system can execute any possible quantum circuit (subject to the constraint that the circuit doesn't use more qubits than the quantum computing system)). In some embodiments, the quantum computing system **1700** is a gate model quantum computer. As previously described, quantum computing systems use so-called qubits, or quantum bits (e.g., **1750**A). While a classical bit has a value of either 0 or 1, a qubit is a quantum mechanical system that can have a value of 0, 1, or a superposition of both values. Example physical implementations of qubits include superconducting qubits, spin qubits, trapped ions, arrays of neutral atoms, and photonic systems (e.g., photons in waveguides). Additionally, the disclosure is not specific to qubits. The disclosure may be generalized to apply to quantum computing systems **1720** whose building blocks are qudits (d-level quantum systems, where $d>2$) or quantum continuous variables, rather than qubits.

[0333] A quantum circuit is an ordered collection of one or more gates. A sub-circuit may refer to a circuit that is a part of a larger circuit. A gate represents a unitary operation performed on one or more qubits. Quantum gates may be described using unitary matrices. The depth of a quantum circuit is the least number of steps used to execute the circuit on a quantum computing system. The depth of a quantum circuit may be smaller than the total number of gates because gates acting on non-overlapping subsets of qubits may be executed in parallel. A layer of a quantum circuit may refer to a step of the circuit, during which multiple gates may be executed in parallel. In some embodiments, a quantum circuit is executed by a quantum computing system. In this sense, a quantum circuit can be thought of as comprising a set of instructions or operations that a quantum computing system can execute. To execute a quantum circuit on a quantum computing system, a user may inform the quantum computing system what circuit is to be executed. A quantum computing system may include both a core quantum device and a classical peripheral/control device (e.g., a qubit con-

troller **1740**) that is used to orchestrate the control of the quantum device. It is to this classical control device that the description of a quantum circuit may be sent when one seeks to have a quantum computer execute a circuit.

[0334] The parameters of a parameterized quantum circuit may refer to parameters of the gates. For example, a gate that performs a rotation about the y axis may be parameterized by a real number that describes the angle of the rotation.

[0335] The description of a quantum circuit to be executed on one or more quantum computing systems may be stored in a non-transitory computer-readable storage medium. The term "computer-readable storage medium" should be taken to include a single medium or multiple media (e.g., a centralized or distributed database, or associated caches and servers) able to store instructions. The term "computer-readable medium" shall also be taken to include any medium that is capable of storing instructions for execution by the quantum computing system and that cause the quantum computing system to perform any one or more of the methodologies disclosed herein. The term "computer-readable medium" includes, but is not limited to, data repositories in the form of solid-state memories, optical media, and magnetic media.

[0336] FIG. **18** is an example architecture of a classical computing system **1710**, according to some embodiments. The quantum computing system **1720** may also have one or more components described with respect to FIG. **18**. FIG. **18** depicts a high-level block diagram illustrating physical components of a computer system used as part or all of one or more entities described herein, in accordance with an embodiment. A computer may have additional, less, or variations of the components provided in FIG. **18**. Although FIG. **18** depicts a computer **1800**, the figure is intended as a functional description of the various features which may be present in computer systems rather than a structural schematic of the implementations described herein. In practice, and as recognized by those of ordinary skill in the art, items shown separately could be combined and some items could be separated.

[0337] Illustrated in FIG. **18** are at least one processor **1802** coupled to a chipset **1804**. Also coupled to the chipset **1804** are a memory **1806**, a storage device **1808**, a keyboard **1810**, a graphics adapter **1812**, a pointing device **1814**, and a network adapter **1816**. A display **1818** is coupled to the graphics adapter **1812**. In one embodiment, the functionality of the chipset **1804** is provided by a memory controller hub **1820** and an I/O hub **1822**. In another embodiment, the memory **1806** is coupled directly to the processor **1802** instead of the chipset **1804**. In some embodiments, the computer **1800** includes one or more communication buses for interconnecting these components. The one or more communication buses optionally include circuitry (sometimes called a chipset) that interconnects and controls communications between system components.

[0338] The storage device **1808** is any non-transitory computer-readable storage medium, such as a hard drive, compact disk read-only memory (CD-ROM), DVD, or a solid-state memory device or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, optical disk storage devices, flash memory devices, or other non-volatile solid state storage devices. Such a storage device **1808** can also be referred to as persistent memory. The pointing device **1814** may be a mouse, track ball, or other type of pointing device, and is used in combination with the keyboard **1810** to input data into the computer **1800**. The graphics adapter **1812** displays images and other information on display **1818**. The network adapter **1816** couples the computer **1800** to a local or wide area network.

[0339] The memory **1806** holds instructions and data used by the processor **1802**. The memory **1806** can be non-persistent memory, examples of which include high-speed random access memory, such as DRAM, SRAM, DDR RAM, ROM, EEPROM, flash memory.

[0340] As is known in the art, a computer **1800** can have different or other components than those shown in FIG. **18**. In addition, the computer **1800** can lack certain illustrated components. In one embodiment, a computer **1800** acting as a server may lack a keyboard **1810**, pointing device **1814**, graphics adapter **1812**, or display **1818**. Moreover, the storage device **1808** can be local or remote from the computer **1800** (such as embodied within a storage area network (SAN)).

[0341] As is known in the art, the computer **1800** is adapted to execute computer program modules for providing functionality described herein. As used herein, the term "module" refers to computer program logic utilized to provide the specified functionality. Thus, a module can be implemented in hardware, firmware, or software. In one embodiment, program modules are stored on storage device **1808**, loaded into the memory **1806**, and executed, individually or together, by one or more processors (e.g., **1802**).

XI. ADDITIONAL CONSIDERATIONS

[0342] Some portions of the above disclosure describe the embodiments in terms of algorithmic processes or operations. These algorithmic descriptions and representations are commonly used by those skilled in the computing arts to convey the substance of their work effectively to others skilled in the art. These operations, while described functionally, computationally, or logically, are understood to be implemented by computer programs comprising instructions for execution, individually or together, by one or more processors, equivalent electrical circuits, microcodes, or the like. Furthermore, it has also proven convenient at times, to refer to these arrangements of functional operations as modules, without loss of generality. In some cases, a module can be implemented in hardware, firmware, or software.

[0343] As used herein, any reference to "one embodiment" or "an embodiment" means that a particular element, feature, structure, or characteristic described in connection with the embodiment is included in at least one embodiment. The appearances of the phrase "in one embodiment" in various places in the specification are not necessarily all referring to the same embodiment. Similarly, use of "a" or "an" preceding an element or component is done merely for convenience. This description should be understood to mean that one or more of the elements or components are present unless it is obvious that it is meant otherwise. As used herein, the terms "comprises," "comprising," "includes," "including," "has," "having" or any other variation thereof, are intended to cover a non-exclusive inclusion. For example, a process, method, article, or apparatus that comprises a list of elements is not necessarily limited to only those elements but may include other elements not expressly listed or inherent to such process, method, article, or apparatus. Further, unless expressly stated to the contrary, "or" refers to an inclusive or and not to an exclusive or. For

example, a condition A or B is satisfied by any one of the following: A is true (or present) and B is false (or not present), A is false (or not present) and B is true (or present), and both A and B are true (or present).

[0344] In addition, use of the "a" or "an" are employed to describe elements and components of the embodiments. This is done merely for convenience and to give a general sense of the disclosure. This description should be read to include one or at least one and the singular also includes the plural unless it is obvious that it is meant otherwise. Where values are described as "approximate" or "substantially" (or their derivatives), such values should be construed as accurate +/−10% unless another meaning is apparent from the context. From example, "approximately ten" should be understood to mean "in a range from nine to eleven."

[0345] Alternative embodiments are implemented in computer hardware, firmware, software, and/or combinations thereof. Implementations can be implemented in a computer program product tangibly embodied in a machine-readable storage device for execution by a programmable processor system including one or more processors that can act individually or together; and method steps can be performed by a programmable processor system executing a program of instructions to perform functions by operating on input data and generating output. Embodiments can be implemented advantageously in one or more computer programs that are executable on a programmable system including one or more programmable processors coupled to receive data and instructions from, and to transmit data and instructions to, a data storage system, at least one input device, and at least one output device. Each computer program can be implemented in a high-level procedural or object-oriented programming language, or in assembly or machine language if desired; and in any case, the language can be a compiled or interpreted language. Suitable processors include, by way of example, both general and special purpose microprocessors. Generally, a processor will receive instructions and data from a read-only memory and/or a random-access memory. Generally, a computer will include one or more mass storage devices for storing data files; such devices include magnetic disks, such as internal hard disks and removable disks; magneto-optical disks; and optical disks. Storage devices suitable for tangibly embodying computer program instructions and data include all forms of non-volatile memory, including by way of example semiconductor memory devices, such as EPROM, EEPROM, and flash memory devices; magnetic disks such as internal hard disks and removable disks; magneto-optical disks; and CD-ROM disks. Any of the foregoing can be supplemented by, or incorporated in, ASICs (application-specific integrated circuits) and other forms of hardware.

[0346] Although the above description contains many specifics, these should not be construed as limiting the scope of the disclosure but merely as illustrating different examples. It should be appreciated that the scope of the disclosure includes other embodiments not discussed in detail above. Various other modifications, changes, and variations which will be apparent to those skilled in the art may be made in the arrangement, operation, and details of the methods and apparatuses disclosed herein without departing from the spirit and scope of the disclosure.

What is claimed is:

1. A quantum interior point method (QIPM) for solving a second-order cone program (SOCP) instance using a quantum computing system, the method comprising:

(a) receiving the SOCP instance;

(b) defining a Newton system for the SOCP instance by constructing matrix G and vector h, where matrix G and vector h describe constrains for a linear system Gu=h based on the SOCP instance;

(c) preconditioning matrix G and vector h via row normalization to reduce a condition number of matrix G;

(d) iteratively determining u until a predetermined iteration condition is met, the iterations comprising:

causing the quantum computing system to apply matrix G and vector h to a quantum linear system solver (QLSS) to generate a quantum state;

causing the quantum computing system to perform quantum state tomography on the quantum state; and

updating a value of u based on a current value of u and the output of the quantum state tomography; and

(e) determining a solution to the SOCP instance based on the updated value of u.

2. The method of claim **1**, wherein preconditioning matrix G and vector h comprises:

determining a diagonal matrix D where at least one entry D_{ii} is equal to the norm of row i of matrix G.

3. The method of claim **2**, further comprising redefining matrix G and vector h according to: $G=D^{-1}G$ and $h=D^{-1}h$.

4. The method of claim **3**, wherein $D^{-1}G$ has a condition number less that the condition number of previous matrix G.

5. The method of claim **1**, wherein the output of the quantum state tomography is a unit vector $\tilde{v}'$ indicating an iteration direction for u.

6. The method of claim **5**, wherein the unit vector $\tilde{v}'$ is characterized by $\|\tilde{v}'-v\|\leq\xi$ with probability equal to or greater than a predetermined probability threshold, where $v\propto G^{-1}h$ and ξ is a tomography precision parameter.

7. The method of claim **5**, wherein updating the value of u comprises:

determining a step length $\sigma\neq 0$ based on the unit vector $\tilde{v}'$; and

adding the current value of u to the product of: (1) the step length σ and (2) the unit vector $\tilde{v}'$.

8. The method of claim **7**, wherein the step length is given by:

$$\frac{\mu(x,\tau,s,\mathcal{H})(1-\sigma)(r+1)}{-(\Delta x)^{\top}s-(\Delta s)^{\top}x-(\Delta,\mathcal{H})\tau-(\Delta\tau)\mathcal{H}}$$

where Δx, Δs, $\Delta\kappa$, and $\Delta\tau$ are components of the unit vector $\tilde{v}'$; x, s, κ, and τ are components of current u; $\mu(x, \tau, s, \kappa)$ is a duality gap; σ is the step length; and r is the number of second-order cone constraints of the SOCP instance.

9. The method of claim **7**, wherein the step length σ is determined such that a duality gap μ of the updated value of u is a factor of σ smaller than the current value of u within a second order deviation in the step length σ.

10. The method of claim **9**, wherein the duality gap μ describes a difference between the current value of u and an exact solution to the linear system Gu=h.

11. The method of claim **1**, wherein the quantum state tomography is performed according to a tomography precision parameter ξ that decreases with each new iteration.

12. The method of claim **11**, wherein the tomography precision parameter ξ decreases by $\xi/2$ with each new iteration.

13. The method of claim **1**, wherein causing the quantum computing system to apply matrix G and vector h to the QLSS to generate the quantum state comprises $k>0$ applications of the QLSS and k controlled-applications of the QLSS to generate the quantum state.

14. The method of claim **1**, further comprising:

receiving a target precision F for the solution to the SOCP instance; and

repeating steps (b)-(d) and iteratively updating a duality gap μ based on the output of the quantum state tomography until the duality gap μ is less than the target precision ε, where the duality gap μ describes a difference between the current value of u and an exact solution to the linear system Gu=h.

15. The method of claim **14**, wherein matrix G and vector h are constructed further based on the updated value of u.

16. The method of claim **1**, wherein causing the quantum computing system to apply matrix G and vector h to the QLSS comprises causing the quantum computing system to execute a quantum circuit.

17. The method of claim **1**, wherein the QLSS operates on a block encoded version of matrix G and a state-prepared version of vector h.

18. The method of claim **17**, further comprising: causing matrix G to be block encoded onto the quantum computing system and the vector h to be state encoded onto the quantum computing system.

19. A non-transitory computer-readable storage medium storing code that, when executed by a computing system including a quantum computing system, causes the computing system to perform operations comprising:

receiving a second-order cone program (SOCP) instance;

defining a Newton system for the SOCP instance by constructing matrix G and vector h, where matrix G and vector h describe constrains for a linear system Gu=h based on the SOCP instance;

preconditioning matrix G and vector h via row normalization to reduce a condition number of matrix G;

iteratively determining u until a predetermined iteration condition is met, the iterations comprising:

causing the quantum computing system to apply matrix G and vector h to a quantum linear system solver (QLSS) to generate a quantum state;

causing the quantum computing system to perform quantum state tomography on the quantum state; and

updating a value of u based on a current value of u and the output of the quantum state tomography; and

determining a solution to the SOCP instance based on the updated value of u.

20. The non-transitory computer-readable storage medium of claim **19**, wherein preconditioning matrix G and vector h comprises:

determining a diagonal matrix D where at least one entry D_{ii} is equal to the norm of row i of matrix G.

* * * * *