New Journal of Physics
The open access journal at the forefront of physics

Deutsche Physikalische Gesellschaft DPG
IOP Institute of Physics
Published in partnership with: Deutsche Physikalische Gesellschaft and the Institute of Physics

OPEN ACCESS

RECEIVED
19 March 2022

REVISED
28 June 2022

ACCEPTED FOR PUBLICATION
5 July 2022

PUBLISHED
5 August 2022

PAPER

Quantum computational quantitative trading: high-frequency statistical arbitrage algorithm

Xi-Ning Zhuang[1,2], Zhao-Yun Chen[3], Yu-Chun Wu[2,3,4,5,*] and Guo-Ping Guo[1,2,3,4,5,*]

[1] Origin Quantum Computing, Hefei, People's Republic of China
[2] CAS Key Laboratory of Quantum Information, University of Science and Technology of China, Hefei 230026, People's Republic of China
[3] Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei 230088, People's Republic of China
[4] CAS Center for Excellence and Synergetic Innovation Center in Quantum Information and Quantum Physics, University of Science and Technology of China, Hefei 230026, People's Republic of China
[5] Hefei National Laboratory, University of Science and Technology of China, Hefei 230088, People's Republic of China
[*] Authors to whom any correspondence should be addressed.

E-mail: wuyuchun@ustc.edu.cn and gpguo@ustc.edu.cn

Keywords: quantum computation, statistical arbitrage, quantitative trading, quantum finance

Abstract

Quantitative trading is an integral part of financial markets with high calculation speed requirements, while no quantum algorithms have been introduced into this field yet. We propose quantum algorithms for high-frequency statistical arbitrage trading by utilizing variable time condition number estimation and quantum linear regression. The algorithm complexity has been reduced from the classical benchmark $O(N^2d)$ to $O(\sqrt{dN}\kappa_0^2\,\log(1/\epsilon)^2))$, where N is the length of trading data, and d is the number of stocks, κ_0 is the condition number and ϵ is the desired precision. Moreover, two tool algorithms for condition number estimation and cointegration test are developed.

1. Introduction

With the rapid development of quantum computing [1–3], the qubits on the chips are up to 53 currently [3], and it will extend beyond 100 soon in the roadmap of quantum systems based on superconductivity. Hence, quantum computing shows the potential to solving practical problems, such as chemistry [4–6], materials [7], drug design [8], and so on.

Quantum computation has produced positive effects in finance [9, 10], and current quantum algorithms mainly focus on solving derivatives pricing problems and risk analysis by quantum Monte-Carlo (QMC) simulation [11–16], optimizing stocks portfolio through quadratic unstrained binary optimization (QUBO) [17–19], and financial analysis work utilizing quantum machine learning (QML) [20–23]. However, for quantitative trading and especially statistical arbitrage, there are no corresponding quantum algorithms yet.

Quantitative trading is an essential field of finance which focuses on algorithmic trading methods via studying historical trading data and developing complex trading algorithm. And statistical arbitrage is a mainstream approach of quantitative trading taken by most hedge funds [24, 25]. The main idea of arbitrage is to describe the comovement out of securities and portfolios and make profit from other traders' pricing error. While lots of classical algorithms for quantitative trading have been proposed [26–29], and traditional hardware techniques including infrared communication and field programmable gate array have been employed over the years [30, 31], still the requirement for speed cannot be satisfied when implementing those complicated statistical methods, especially in the quicker-take-all situation of high-frequency trading (HFT) whose need of computing speed is crucial [32]. In statistical arbitrage, one needs to find a potential cointegrated pair via many linear regressions and cointegration tests involving a huge matrix of historical data. For example, in US stock markets, the problem size can exceed $N = 10^7$ and the complexity is 10^{15} (see section 6 for details) which is very hard to calculate by classical computers. For this problem, quantum computation might provide an effective solution.

In this article, quantum algorithms applied to statistical arbitrage strategy are proposed. It consists of two subroutines: the first one is the variable time preselection algorithm (VTPA) that will help to find, with high probability, the potential comovement out of securities and portfolios. The second one is the quantum cointegration test algorithm (QCTA) that focuses on the efficient verification of cointegrated pairs, which is quite valuable in statistics but has not been achieved via quantum computation ever before. The classical benchmark to achieve the preselection procedure is by matrix factorization with complexity $O(N^3)$ [33], while our algorithm's complexity is $O(\sqrt{dN}\kappa_0^2\log(1/\epsilon)^2)$ where d is the number of stocks usually much less than time length N and κ_0 is the condition number. Moreover, an efficient tool named quantum condition number comparison algorithm (QCNCA) used to probe a matrix's condition number is proposed, and it can be applied to many other domains. The estimation and optimization on condition number is a crux of realistic problems with linear systems, while few work focuses on this issue before.

The structure of this article is as follows: after giving the problem statement and analysis in section 2, the global structure and main results of our work are shown in section 3. The details of VTPA and QCT are described in sections 4 and 5, respectively, followed by a discussion on complexity and quantum advantage in section 6.

2. Statistical arbitrage problem

Statistical arbitrage is a market neutral trading method to model the comovement of different assets and correct the pricing error in the market. Pioneered by Gerry Bamberger [34], statistical arbitrage has developed a lot, and the crux and core are to model the comovement. Following the framework first introduced by Vidyamurthy [27], statistical arbitrage is divided mainly into three key steps: firstly, two or more securities moved together historically in a formation period should be preselected; secondly, some version of the Engle–Granger cointegration test [35] is taken for verification; thirdly, the spread between them in a subsequent trading period is monitored by some optimal entry/exit thresholds. Since the spread of stocks will revert to its historical mean, and the profit can be made from other traders' irrational behavior by longing the oversold securities and shorting the overbought ones at the same time [26] (see figure 1 for reference).

The first and hard problem of statistical arbitrage is to find the comovement of assets. Although the covariance matrix can be applied to describe the correlation of every two assets pair, it is much more often for traders to model the multicollinearity of more than three assets in the market. In statistics, **multicollinearity** refers to a situation in which some of the explanatory variables in a multiple regression model are highly linearly related. The formal problem statement is given as follows:

Problem 1 (Multicollinear detection problem). Suppose that $P = \{p\}$ is the set of portfolios, and $p = (p_t^{(j)})_{J\times T}$ is a portfolio of stocks' historical quote data. Here $p_t^{(j)}$ is an element of p as the jth stock's price at time t. The matrix p is of full rank since no perfect linear relation exists in noisy financial market data. The target is to verify whether there are $p_t^{(j)}$s whose multicollinearity are statistically significant.

The classical method to seek the multicollinearity is by multiple regression. However, there are two difficulties one have to face: firstly the classical regression algorithm's complexity is $O(n^3)$ while the realistic problem size n is quite large; secondly, the algorithm complexity will increase quickly for ill-conditioned matrix which often appears in the situation of multicollinearity.

In numerical analysis, to detect and measure the seriousness of the multicollinearity problem, the **condition number κ** is introduced. Given problem f, it is generally a measurement to describe the change of the output value divided by the change of the input variable x:

$$\kappa(f) = \lim_{\epsilon\to 0}\sup_{\|\delta x\|\leqslant\epsilon}\frac{\|\delta f\|}{\|\delta x\|}.$$

In the case of matrices, the condition number associated with the linear equation $Ax = b$ release the dependence of accuracy on the input data. Specifically, the condition number of normal matrix A is

$$\kappa(A) = \frac{|\lambda_{\max}(A)|}{|\lambda_{\min}(A)|}.$$

It should be emphasized that condition number is a property of the matrix itself and does not depend on the algorithm or accuracy of the computer used. Hence, both classical and quantum computers have a common problem to solve an ill-conditioned (high condition number) linear system of equations.

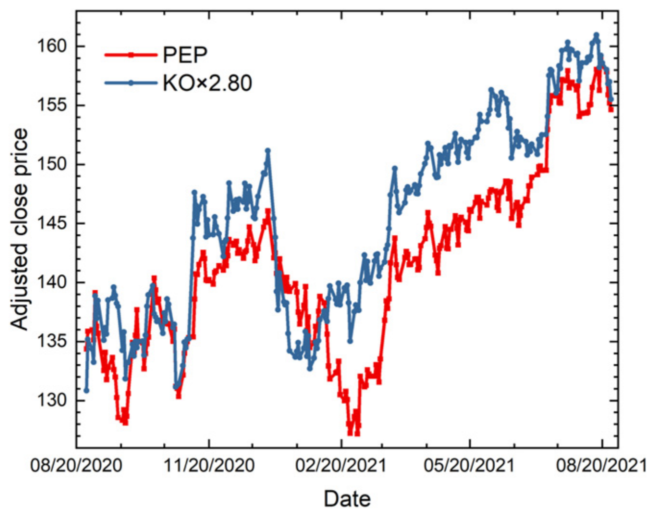

Figure 1. The comovement of adjusted close prices of PEP and KO with a rescaling multiplier 2.8. The *x*-axis is the time line and the *y*-axis represents the price. The red line is the sequence of daily close prices of Pepsi, and the blue line is the sequence of daily close prices of Coca-Cola multiplied by a factor 2.8 from August 8 2020 to August 8 2021. The factor is derived from a linear regression since the scales of average prices of these companies are different and can not be compared directly. This picture shows the comovement of these two stocks and the statistical arbitrage strategy: on February 2 2021 the spread of these stocks increased to a maximum and we can long the PEP and short the KO, and then hold them until the spread reverses in July 2021 and make profit. (The market data can be downloaded from Yahoo! Finance's API. A pythonic way to download market data can be found on the website https://pypi.org/project/yfinance/).

The larger the condition number, the more ill-conditioned the matrix is, and the algorithm complexity will increase very quickly.

The second problem of statistical arbitrage is the verification of the **multicointegration** of the preselected multicollinear securities. To make profit the price spread is assumed to revert to its historical mean, which is guaranteed by the statistical concept multicointegration. Briefly speaking, multicointegration demands a linear combination of several given time series (denoted as residuals series) to be stationary (not time varying), and it can describe the reversion property of multi variables. With the detailed explanation of statistical concepts given in appendix A, the formal problem statement is:

Problem 2 (Multicointegration test problem). Suppose that $p = (p_t^{(j)})_{J\times T}$ is a portfolio of stocks' historical quote data. Here $p^{(j)}$ is a time series as the *j*th stock's price. The target is to verify whether these time series are cointegrated.

The classical cointegration test algorithm consists of two regression procedures: one is for the derivation of the linear combination coefficients, and the other one is for the time stationary hypothesis test. The complexity of each regression is $O(n^3)$.

3. Quantum statistical arbitrage

In this section, two algorithms solving the quantum statistical arbitrage problem are proposed. One is for the case of fixed condition number threshold; the other is for a fixed number of remained portfolios. Given historical data of many stocks for a long time interval, our target is to select those stocks that are cointegrated. The algorithm mainly contains two steps: preselect multicollinear stock portfolios from the pool by applying $VTPA(p, \kappa)$ where $VPTA$ is true if the given portfolio p's condition number is larger than the threshold κ; and then verify whether the preselected portfolio p is cointegrated by implementing $QCT(p)$ to output cointegration flag f and corresponding coefficients β.

3.1. Data loading

Data loading is an important and difficult procedure for quantum algorithms considering practical application. In this subsection, the data structure and loading procedure are given, and then the time complexity is analyzed. Moreover, we proved that this procedure is efficient and optimal.

Suppose that $P = \{p\}$ is the portfolio pool, and $(p_t^{(j)})_{J\times T}$ is a portfolio of stocks' historical quote data. Here $p_t^{(j)}$ is an element of p as the *j*th stock's price at time t. The matrix p is of full rank since no perfect linear relation exists in noisy financial market data. The two quantum statistical algorithms work in

Algorithm 1. Quantum statistical arbitrage algorithm with fixed condition number preselection.

```
Input:
  $\kappa_0$: the threshold for preselection
  $T$: the length of time interval
  $J$: the total number of stocks
  $d$: number of stocks in one portfolio
  $P$: the portfolio pool set contains portfolios p
  $p_t^{(j)}$: the $j$th stock's price at time $t$
Output:
  $(p, \beta)$ cointegrated portfolios and cointegration coefficients

  Data loading
  for $p$ in $P$ do
    $|p\rangle = \sum_{t=0}^{T-1}\sum_{j=0}^{J-1} p_t^{(j)}|t\rangle|j\rangle$
    if $VTPA(p, \kappa_0) = True$ then
      $QCT(p) = f, \beta$
      if $f = True$ then
        Output $(p, \beta)$
    else
      Skip to the next loop
```

the standard oracle model, and the matrix is stored in a quantum random access memory (qRAM) [36–38]. A procedure $\mathcal{P}_x$ is assumed to perform the map

$$|j\rangle|t\rangle|z\rangle \rightarrow |j\rangle|t\rangle\left|z \oplus p_t^{(j)}\right\rangle$$

for any $j \in [1, 2, \ldots, d]$ and $t \in [1, 2, \ldots, N]$, and the price is stored as a bit string in the third register.

In order to derive the desired real symmetric matrix, the strategy of HHL [39, 40] is adopted as:

$$A = \begin{pmatrix} 0 & X \\ x^{\mathrm{T}} & 0 \end{pmatrix}.$$

Moreover, the norm of the matrix is assumed to satisfy $\|A\| = 1$ without loss of generality since otherwise let $A = \frac{A}{\|A\|}$.

For most quantum algorithms concerning realistic situation compared to classical algorithms, data loading is a common and crucial problem such as pattern recognition and recommendation system where data loading is a bottleneck problem since every tick a lot of data carrying the picture information need to be loaded [41]. However, for the case of financial data, things are very different:

For our specified financial scene, we apply the technique named incremental updating [42, 43], so that the time complexity of data loading procedure can be regarded as a constant. The data need to be loaded are divided into two parts: the historical data and the new-loading data. Although the time of historical data loading may be long for the first time, it is finished before the trading time. And in every new second of trading time, the large matrix of historical data is unchanged and only a fixed number of lines of data will be updated. Hence the complexity of data updating can be assumed as a constant, which is very different from other quantum algorithms.

When compared to classical algorithms, since this new-loading data storage procedure is also the data generation procedure, the time complexity of our algorithm's quantum data loading is in the same order of classical ones. And the quantum algorithm will benefit from the computing procedure.

Moreover, since the data based on historical trading is random, it can not be derived efficiently from a designed quantum circuit. Otherwise the history data can be predicted perfectly by a fixed circuit and this is impossible. This hints that there is no more efficient circuit to prepare the historical matrix directly, and the incremental data updating is optimal.

3.2. Fixed condition number algorithm

In the case the market is stationary and the threshold of condition number used for the preselection can be derived from historical trading data. If an efficient κ_0 derived from historical data is taken as filter threshold, the following algorithm 1 is given:

Suppose that we have the historical data of J stocks with T ticks, then for each portfolio p of d stocks, we apply the variable time preselection algorithm circuit. This quantum circuit will return whether this data matrix's condition number is larger than κ_0. It is based on the new original quantum condition number comparison algorithm that can implement a quick condition number estimation by eigenvalue

Algorithm 2. Quantum statistical arbitrage algorithm with progressive preselection.

```
Input:
    $k$: portfolio number threshold
    $T$: the length of time interval
    $J$: the total number of stocks
    $d$: the number of stocks in one portfolio
    $P$: the portfolio pool
    $p_t^{(j)}$: the $j$th stock's price at time $t$
Output:
    $(p, \beta)$ cointegrated portfolios and cointegration coefficients

    Data loading
    Step counter $j = 1$
    Portfolio counter $K = |P|$
    while $K > k$ do
        $\kappa_j = 2^j$
        for $p$ in $P$ do
            $|p\rangle = \sum_{t=0}^{T-1}\sum_{j=0}^{J-1} p_t^{(j)}|t\rangle|j\rangle$
            if $VTPA(p, \kappa_j) = True$ then
                skip
            else
                $K = K - 1$
                $P = P - \{p\}$
        $j = j + 1$
for $p$ in $P$ do
    $QCT(p) = (f, \beta)$
    if $f = True$ then
        Output $(p, \beta)$
```

computation, and the variable time structure is applied to make a second accelerated effect by handling with the case of large condition number. If the VTPA's result is true, we will know that the matrix's condition number is large enough so that there is high probability that this portfolio contains multicollinear columns of stock price sequences. Then we will apply the quantum cointegration test circuit which can compute if this multicollinear portfolio contains a linear combination of stocks that is cointegrated as desired.

3.3. Adaptive condition number algorithm

As for the case that the financial market changes drastically and no fixed threshold κ_0 can be derived, an even more efficient algorithm 2 is provided and he basic idea is as follows: since our single-step preselection sub-algorithm can be used for any given κ, a progressive κ preselection procedure can be implemented. Portfolio matrices with small κ will be directly obsoleted in the first several steps until the number of matrices left is small enough, and until then, the quantum cointegration test will be implemented.

Both of the above two algorithms are for statistical arbitrage, and the selection depends on the specific market: if the κ-threshold is stationary, the first algorithm is chosen; otherwise, the second one is preferred. Since the two subroutines VTPA and QCT are complicated and tool sub-algorithm QCNCA is developed, they will be introduced in sections 4 and 5, respectively.

4. Variable time preselection

In this section, we will explain the main idea of the first part of our work as a variable time quantum algorithm to preselect the stocks that are multicollinear and thus may be cointegrated as needed.

Although ill-conditioned matrices are commonly considered a terrible problem that one should try to avoid, we develop the heuristic idea to detect multicollinearity by searching matrices with small eigenvalues and large condition numbers. QCNCA is developed to determine whether the condition number κ of a given matrix is larger than the threshold κ_0 in subsection 4.1.

Since QCNCA's dependence on κ is quadratic, the technique of variable time quantum algorithm is introduced to accelerate the implementation of matrices selection [44], and then the VTPA is as follows:

Theorem *1.Supposing that many different linear systems are given with unknown condition number κ and P_j denote the probability that condition number satisfies $\kappa_{j-1} = 2^{j-1} \leqslant \kappa \leqslant \kappa_j = 2^j$. Then there is an efficient quantum algorithm to preselect matrices with condition numbers $\kappa \geqslant \kappa_0$. The average query complexity is*

$O(\sqrt{d}\,\log(1/\epsilon)^2(\sum_{j=1}^{M}4^j jP_j))$. *As for a uniform probability distribution, the query complexity is* $O(\sqrt{d}\kappa_0^2\,\log(1/\epsilon)^2)$ *to determine whether the condition number is larger than* κ_0.

The proofs of correctness and complexity of theorem 1 are given in subsections 4.3 and 4.4, respectively.

4.1. Tools: quantum condition number comparison algorithm

Realizing that multicollinearity appears with large κ [45, 46], and hence small eigenvalues, the following preselection algorithm is developed: repeat a simplified phase estimation sub-algorithm until an eigenvalue small enough is detected. If such an eigenvalue is found, the corresponding portfolios will be recorded as an alternative one. It worth noticing that some cointegrated pairs may be missed in our algorithm, but it does not matter since our task is to search for some collinear portfolios instead of the impossible mission to find all of the cointegrated pairs. We denote this procedure quantum condition number comparator $QCNC(\kappa, \varphi)$ and get the following result:

Lemma 2. *Supposing that* A *is an* $N\times N$ *Hermitian matrix with* $\|A\| = 1$ *with unknown condition number* κ *and the probability density function of eigenvalues is* $p(\lambda)$. *Then there is a quantum algorithm using* $O(\kappa_0\,\log(1/\epsilon)\frac{\int_{1/\kappa}^{1}p(x)\,\mathrm{d}x}{\int_{1/\kappa_0}^{1/\kappa}p(x)\,\mathrm{d}x})$ *calls of* A *to determine whether the condition number is larger than* κ_0. *In the case of a uniform probability distribution,* A*'s calls are* $O(\kappa_0^2\,\log(1/\epsilon))$ *so that whenever* $\kappa \geqslant 2\kappa_0$, *the target qubit will be 1.*

It should be noticed that this repeating time, especially when κ is large, is determined by the threshold κ_0, while traditional algorithms depend on the unknown κ. This is an algorithm finding whether the condition number of a linear system is large than the given threshold without solving the equations.

Proof of lemma 2. Without loss of generality, suppose that A is a matrix with Frobenius norm

$$\|A\|_F = \left(\sum_{i=1}^{m}\sum_{j=1}^{n}|a_{ij}|^2\right)^{1/2} = \sqrt{d} \tag{1}$$

(otherwise let $A = \frac{\sqrt{d}}{\|A\|_F}A$), and unknown rank r. A direct calculation shows that:

$$\begin{aligned} |\lambda_{\max}(A)| &= \|A\|_2 && (2)\\ &\geqslant \frac{1}{\sqrt{r}}\|A\|_F && (3)\\ &= \sqrt{d/r} && (4)\\ &\geqslant 1. && (5)\end{aligned}$$

Here in (2)

$$\|A\|_2 = \sup_{x\neq 0}\frac{\|Ax\|}{\|x\|} = \sigma_{\max}(A) \tag{6}$$

is the induced L_2 norm and equals to $|\lambda_{\max}(A)|$ (see [47]'s example 5.6.6), and it follows the inequality (3) (see [48]).

For any given eigenvector $|\lambda\rangle$, with a variant of phase estimation, it is easy for us to determine whether its corresponding eigenvalue λ is larger than $1/\kappa_0$ or not with complexity $O(\kappa_0\log(1/\epsilon))$ [49]. By the definition of the condition number of normal matrices, for any known eigenvalue λ:

$$\kappa = \frac{|\lambda_{\max}(A)|}{|\lambda_{\min}(A)|} \geqslant \frac{1}{|\lambda(A)|} \geqslant \kappa_0. \tag{7}$$

Hence a lower bound of the condition number is also given. Whenever a sufficiently small eigenvalue λ_0 is given, the matrix can be regarded with condition number greater than κ_0 and high multicollinearity as a consequence.

Obviously, there is a certain probability of success when the testing eigenvalue is larger than κ_0. Let the condition number be κ and the probability density function of eigenvalues be p; the success probability is

$$P_{\text{success}} = \frac{\int_{1/\kappa_0}^{1/\kappa}p(x)\,\mathrm{d}x}{\int_{1/\kappa}^{1}p(x)\,\mathrm{d}x}. \tag{8}$$

Under the assumption that the eigenvalues follow a uniform probability distribution, the success probability turns to be

$$P_{\text{success}} = \frac{1/\kappa_0 - 1/\kappa}{1 - 1/\kappa} \tag{9}$$

$$= \frac{1}{\kappa_0}\frac{\kappa - \kappa_0}{\kappa - 1} \tag{10}$$

$$\approx \frac{1}{\kappa_0}\left(1 - \frac{\kappa_0}{\kappa}\right). \tag{11}$$

Here κ is assumed large and $\kappa - 1 \simeq \kappa$. Moreover, whenever a matrix with $\kappa \geqslant 2\kappa_0$ is given, we have:

$$P_{\text{success}} \geqslant 1/2\kappa_0. \tag{12}$$

This procedure shall be repeated $2\kappa_0$ times to boost the success probability. Hence the total number of calls for A is $O(\kappa_0^2 \log(1/\epsilon))$. Since complexity to simulate $U = e^{iA}$ is $O(\sqrt{d}(1 + \log(\kappa_0/\epsilon)))$ [50], the total query complexity is $O(\sqrt{d}\kappa_0^2 \log(1/\epsilon)(1 + \log(\kappa_0/\epsilon)))$. ■

It should be mentioned that the assumption of uniform distribution is reasonable. Although different distributions of eigenvalues may appear in specified realistic problems, some normal conditions can be imposed to guarantee that the algorithm will still work with slight modification.

4.2. Algorithm

To see how to derive an algorithm more efficient on κ, one should notice that matrices with small condition numbers can be found quite early and need not be calculated anymore. Some clock registers are used to obsolete those matrices with small condition numbers by starting from small threshold κ_0. The larger the threshold κ_0 is, the fewer the matrices need to be tested. Repeating this procedure several times can make the acceleration.

Suppose that $M = \lceil \log \kappa_0 \rceil$. The clock registers $C_{1,\ldots,M}$ are used to control and store the result of subprocedure $\mathcal{A}_j$ defined later. Another one-qubit register $\mathcal{F}$ is used as a flag register to donate if the algorithm is stopped. For all $j \in \{1, \ldots, M\}$, let $\phi_j = 1/\kappa_j = 2^{-j}$, and let ϵ be the desired precision. Since it is our target to verify whether matrix A contains components corresponding to small eigenvalues, the algorithm is defined as $\mathcal{A} = \mathcal{A}_M\mathcal{A}_{M-1}\ldots\mathcal{A}_1$, where $\mathcal{A}_j$ is defined as follows:

Algorithm $\mathcal{A}_j$ conditional on first $j-1$ qubits of $\mathcal{H}_{\mathcal{C}}$ being $|1\rangle$, apply QCNC(κ_j, ϵ) using C_j as the output qubit and additional fresh qubits from $\mathcal{P}$ as ancillary (denoted by P_j). If C_j is left $|0\rangle$ in the first term, the qubit on stop flag register $\mathcal{F}$ will be flipped.

4.3. Correctness

We shall now prove the correctness of this algorithm.

Proof of theorem 1 (Correctness part). Given a matrix A, the condition number is either in some interval $[\kappa_j, 2\kappa_j]$ or greater than κ_0.

(i) Suppose that a matrix with condition number $\kappa_j \leqslant \kappa \leqslant 2\kappa_j$ is given:

State after $\mathcal{A}_1$ **to** $\mathcal{A}_{j-1}$. Since $\kappa \leqslant \phi_{1,\ldots,j-1}$, the clock registers $C_1, \ldots, C_{j-1}$ are at position $|1\rangle$ while the stop flag register $\mathcal{F}$ stays $|0\rangle$ with high probability. After $j-1$ steps the state is left as

$$|1\rangle_{C_1,\ldots,C_{j-1}}|0\rangle_{C_j,\ldots,C_M}|0\rangle_{\mathcal{F}}|\gamma_1^1\rangle_{P_1}\ldots|\gamma_1^{j-1}\rangle_{P_{j-1}}|0\rangle_{P_j\ldots P_M},$$

where $\left|\gamma_1^i\right\rangle$ is the ancillary state produced by the ith call to QCNC.

State after $\mathcal{A}_j$. Because $\phi_j \leqslant \kappa \leqslant 2\phi_j$, QCNC will split the jth control register to $|1\rangle_{C_j}$ with high probability:

$$\beta_0|\mathcal{U}_{j-1}\rangle_{\mathcal{C}}|0\rangle_{\mathcal{F}}|\gamma_1^1\rangle_{P_1}\ldots\left|\gamma_1^{j-1}\right\rangle_{P_{j-1}}\left|\gamma_0^j\right\rangle_{P_j}|0\rangle_{P_{j+1}\ldots P_M} + \beta_1|\mathcal{U}_j\rangle_{\mathcal{C}}|0\rangle_{\mathcal{F}}|\gamma_1^1\rangle_{P_1}\ldots\left|\gamma_1^{j-1}\right\rangle_{P_{j-1}}\left|\gamma_1^j\right\rangle_{P_j}|0\rangle_{P_{j+1}\ldots P_M}$$

$$\text{where}\,\mathcal{U}_j = 1^j0^{m-j}.$$

Since C_j is left $|0\rangle$ in the first term, the qubit on register $\mathcal{F}$ is flipped:

$$\beta_0|\mathcal{U}_{j-1}\rangle_{\mathcal{C}}|1\rangle_{\mathcal{F}}|\gamma_1^1\rangle_{P_1}\ldots\left|\gamma_1^{j-1}\right\rangle_{P_{j-1}}\left|\gamma_0^j\right\rangle_{P_j}|0\rangle_{P_{j+1}\ldots P_M} + \beta_1|\mathcal{U}_j\rangle_{\mathcal{C}}|0\rangle_{\mathcal{F}}|\gamma_1^1\rangle_{P_1}\ldots\left|\gamma_1^{j-1}\right\rangle_{P_{j-1}}\left|\gamma_1^j\right\rangle_{P_j}|0\rangle_{P_{j+1}\ldots P_M}.$$

State after $\mathcal{A}_{j+1}$. This will affect the two parts of the state in different way: in the case that the jth control qubit is split, the step QCNC(κ_{j+1}, ϵ) is implemented. Notice that $\kappa \leqslant \phi_{j+1}$, the state turns to be:

$$\beta_1|\mathcal{U}_j\rangle_{\mathcal{C}}|1\rangle_{\mathcal{F}}|\gamma_1^1\rangle_{P_1}\ldots\left|\gamma_1^j\right\rangle_{P_j}\left|\gamma_0^{j+1}\right\rangle_{P_{j+1}}|0\rangle_{P_{j+2}\ldots P_M}.$$

As for the case that jth control qubit is $|0\rangle$, nothing will be done and the state is:

$$\beta_0|\mathcal{U}_{j-1}\rangle_{\mathcal{C}}|1\rangle_{\mathcal{F}}\left|\gamma_1^1\ldots\gamma_1^{j-1}\right\rangle_{P_1\ldots P_{j-1}}\left|\gamma_0^j\right\rangle_{P_j}|0\rangle_{P_{j+1}\ldots P_M}+\beta_1|\mathcal{U}_j\rangle_{\mathcal{C}}|1\rangle_{\mathcal{F}}\left|\gamma_1^1\ldots\gamma_1^j\right\rangle_{P_1\ldots P_j}\left|\gamma_0^{j+1}\right\rangle_{P_{j+1}}|0\rangle_{P_{j+2}\ldots P_M}.$$

State after $\mathcal{A}$

Given a matrix A with condition number $\kappa_j \leqslant \kappa < \kappa_{j+1}$, the final state at the end of algorithm $\mathcal{A}$ is:

$$\beta_0|\mathcal{U}_{j-1}\rangle_{\mathcal{C}}|1\rangle_{\mathcal{F}}\left|\gamma_1^1\ldots\gamma_1^{j-1}\right\rangle_{P_1\ldots P_{j-1}}\left|\gamma_0^j\right\rangle_{P_j}|0\rangle_{P_{j+1}\ldots P_M}+\beta_1|\mathcal{U}_j\rangle_{\mathcal{C}}|1\rangle_{\mathcal{F}}\left|\gamma_1^1\ldots\gamma_1^j\right\rangle_{P_1\ldots P_j}\left|\gamma_0^{j+1}\right\rangle_{P_{j+1}}|0\rangle_{P_{j+2}\ldots P_M}.$$

(ii) As for matrix A with condition number $\kappa \geqslant \kappa_0$, we have:

$$|1\rangle_{\mathcal{C}}|0\rangle_{\mathcal{F}}|\gamma_1^1\ldots\gamma_1^M\rangle_{P_1\ldots P_M}.$$

It should be noticed that whenever the flag register is split to $|1\rangle_{\mathcal{F}}$, the algorithm stops at some step and reject the hypothesis of the condition number larger than κ_0. Hence a measurement can be implemented on $|1\rangle_F$ to decide whether the matrix contains a component with eigenvalues less than some given $1/\kappa_0$. Besides, a control counter circuit can be employed on the M clock registers to probe the range for κ.

4.4. Complexity analysis

In this subsection, the algorithm's complexity analysis is given to finish the proof of theorem 1. It should be mentioned that the algorithm complexity depends on the specific distribution of condition numbers and eigenvalues of the problem. In this work, a theoretical framework is developed for analysis. Moreover, as an example, the result assuming that $\log\kappa$ follows a uniform probability distribution is calculated. This assumption is common and reasonable since there are relatively fewer matrices with large condition number.

Proof of theorem 1 (Complexity part). Suppose that there are n matrices and the condition number threshold for comparison is κ_0. Let $M = \lceil\log\kappa_0\rceil$. Let $\kappa_j = 2^j$ and P_j be the probability that the matrix's condition number satisfies $\kappa_{j-1} \leqslant \kappa \leqslant \kappa_j$. Then the cumulative number of queries T_j for this kind of matrix is:

$$T_j = \sum_{k=1}^{j}\mathrm{QCNC}(\kappa_k, \epsilon) \tag{13}$$

$$= \sum_{k=1}^{j}\kappa_k^2\log(1/\epsilon)\sqrt{d}(1+\log(\kappa_k/\epsilon)) \tag{14}$$

$$= \sqrt{d}\log(1/\epsilon)^2\sum_{k=1}^{j}2^{2k+1}k \tag{15}$$

$$= \sqrt{d}\log(1/\epsilon)^2\frac{(j-1/3)4^{j+1}+4/3}{3} \tag{16}$$

$$\leqslant \frac{4^{j+1}j}{3}\sqrt{d}\log(1/\epsilon)^2. \tag{17}$$

Hence considering the probability, the arithmetic average number of queries is:

$$T_{\mathrm{avg}} = \sum_{j=1}^{M}P_jT_j \tag{18}$$

$$\leqslant \sum_{j=1}^{M}\frac{4^{j+1}j}{3}\sqrt{d}\log(1/\epsilon)^2P_j \tag{19}$$

$$= \frac{4}{3}\sqrt{d}\log(1/\epsilon)^2\left(\sum_{j=1}^{M}4^jjP_j\right). \tag{20}$$

Supposing that $\log\kappa$ follows a uniform probability contribution, the probability is

$$P_j = 1/M = 1/\log\,\kappa_0$$

and the average time is:

$$T_{\text{avg}} \leqslant \frac{4}{3}\sqrt{d}\,\log\,(1/\epsilon)^2\left(\sum_{j=1}^{M}4^j jP_j\right) \tag{21}$$

$$= \frac{4}{3}\sqrt{d}\,\log\,(1/\epsilon)^2\left(\sum_{j=1}^{M}4^j j/M\right) \tag{22}$$

$$= \frac{4}{3M}\sqrt{d}\,\log\,(1/\epsilon)^2\left(\sum_{j=1}^{M}4^j j\right) \tag{23}$$

$$= \frac{4}{3M}\sqrt{d}\,\log\,(1/\epsilon)^2\frac{M4^{M+1}}{3} \tag{24}$$

$$\leqslant \frac{16}{9}\sqrt{d}\,\log\,(1/\epsilon)^2 4^M \tag{25}$$

$$= \frac{16}{9}\sqrt{d}\kappa_0^2\,\log\,(1/\epsilon)^2. \tag{26}$$

Hence the complexity is $O(\sqrt{d}\kappa_0^2\,\log\,(1/\epsilon)^2)$ as claimed in theorem 1. ■

The complexity of this algorithm also depends on the fixed threshold κ_0 instead of an unknown κ. Hence besides the acceleration compared to classical algorithms, the stability and robustness are also improved to satisfy financial problems.

5. Quantum cointegration test

To finish the last piece of quantum statistical arbitrage, it needs to be verified whether the preselected matrices contain a cointegrated pair. The global structure and details of QCT are described in the first subsection, and the analysis of complexity is given in the second subsection. These two parts yield the following result:

Theorem 3. *Suppose that d and N are the number of kinds of stocks and the time length of stock prices, ϵ is the precision desired, and κ is the condition number. Then the cointegration test with L lag-length augmented dickey fuller test can be implemented with complexity* $O(\frac{d^{2.5}\kappa^3}{\delta^2}\text{poly}(\log_2\frac{d\kappa}{\delta}) + dN + \frac{(L+2)^{2.5}\kappa'^3}{\delta'^2}\text{poly}(\log_2\frac{(L+2)\kappa'}{\delta'})$, *where* $\delta = \min\{1/d,\epsilon\}$, $\delta' = \min\{1/(L+2),\sqrt{L+2}\epsilon^2\}$.

5.1. Algorithm

First of all, the following procedure is used to generate the residual sequence of linear regression. Since the residuals sequence is needed instead of regression coefficients or predicted values [51, 52], known quantum linear algorithms should be employed with some further modification. The work of [51]'s theorem 2 is used to derive an approximation β of the regression coefficients $\hat{\beta}$.

Lemma 4 (QLR, theorem 2 in [51]). *Let* $\mathbf{X} = (x_{i,j})$ *be an* N^*d *balanced matrix such that its singular values are in range* $[1/\kappa, 1]$. *Let* $\mathbf{y} = (y_1, y_2, \dots, y_N)^{\text{T}}$ *be a balanced unit vector. Suppose* $(\mathbf{X}, \mathbf{y})$ *is well behaved. Given* $\epsilon > 0$ *and access to the procedures* P_x *and* P_y *described above. Then the problem to output a vector* $\beta = (\beta_1, \beta_2, \dots, \beta_d)^{\text{T}}$ *such that* $\left|\beta - \hat{\beta}\right| \leqslant \epsilon$ *and* $\beta = \hat{\mathbf{X}}^{\dagger}\mathbf{y}$ *can be solved by a gate-efficient quantum algorithm that makes* $O(\frac{d^{2.5}\kappa^3}{\delta^2}\text{poly}[\log_2(\frac{d\kappa}{\delta})])$ *uses of* P_x *and* P_y, *where* $\delta = \min\{1/d,\epsilon\}$.

This quantum linear regression procedure is denoted as $QLR(d,\delta,\kappa)$. Then the predicted value vector $\hat{y}$ is calculated by the matrix multiplication

$$\hat{y} = X\beta, \tag{27}$$

and the residuals sequence is derived by a vector subtraction between the predicted values $\hat{y}$ and real values y:

$$u = y - \hat{y}. \tag{28}$$

Algorithm 3. Quantum cointegration test algorithm.

```
Input:
    $\kappa_0$: the threshold for preselection
    $T$: the length of time interval
    $J$: the total number of stocks
    $p_t^{(j)}$: the $j$th stock's price at time $t$
Output:
    $(f, \beta)$flag and cointegrated coefficients
    Data loading:
        $|\psi_x\rangle = \sum_{t=0}^{T-1}\sum_{j=0}^{J-1} r_t^{(j)}|t\rangle|j\rangle$: amplitude encoding
    Residual construction module:
        $QLR(d, \delta, \kappa)$ to derive $\beta$
        Classical matrix multiplication $\hat{y} = X\beta$
        Classical vector subtraction $\hat{u} = y - \hat{y}$
    Statistics calculation module:
        Lagged residuals $\Delta u_t = u_t - u_{t-1}$
        $QLR(L+1, \delta', \kappa')$ to derive $\gamma$
        Classical test statistic $\mathrm{DF}_T$
        Comparison with Critical value table ([53])
```

This should be a hybrid algorithm since classical algorithms can calculate matrix multiplications and subtractions with fewer restrictions and more efficiently.

Next, another regression $QLR(L+1, \delta', \kappa')$ on time variable and lagged residuals will be employed to derive the statistical index. The lagged residuals Δu_t is defined as the first-order difference and can be calculated efficiently by a vector subtraction:

$$\Delta u_t = u_t - u_{t-1}. \tag{29}$$

Then $QLR(L+1, \delta', \kappa')$ procedure shows:

$$\Delta u_t = \alpha + \beta t + \gamma u_{t-1} + \sum_{i=1}^{L-1} \delta_i \Delta u_{t-i} + \epsilon_t, \tag{30}$$

where L is the lag-length used in the ADF test, and β is the coefficient of the time variable t, and γ is the coefficient of the first-order difference Δu_t. The test statistic $\mathrm{DF}_T = \frac{\hat{\gamma}}{\mathrm{SE}(\hat{\gamma})}$, where SE means standard error, can be computed by classical computer more efficiently.

Finally, the result will be sent to be compared with a critical value table [53]. And the total algorithm is summarised as algorithm 3.

5.2. Complexity analysis

In the following subsection, a detailed analysis of the algorithm's complexity is given. Suppose a single-round cointegration test on an Nd design matrix where N is the number of samples and d is the number of variables. By lemma 4, the regression coefficients can be derived directly with complexity to be $O(\frac{d^{2.5}\kappa^3}{\delta^2}\mathrm{poly}(\log_2 \frac{d\kappa}{\delta}))$. Then the residuals can be computed directly in $O(Nd)$ steps. The result of this hybrid residual generation procedure is as follows:

Lemma 5 (Complexity of residuals sequence generation procedure). *Suppose X is an N^*d design matrix and y the target vector, also we have ϵ the precision desired, and κ is the condition number. Then the residuals sequence of regression can be derived with complexity $O(\frac{d^{2.5}\kappa^3}{\delta^2}\mathrm{poly}(\log_2 \frac{d\kappa}{\delta}) + dN)$.*

Besides this, it should be mentioned that an alternative method use [52]'s work to derive a predictor of a linear model. This method should be repeated N times to derive the residuals sequence. Hence the total algorithm is $O(N\log N\kappa^2\epsilon^{-3})$.

Since the residuals derived from the above subroutine are intermediate instead of final results, it is important for us to analyze the error propagation of the cointegration test to control the global error:

Lemma 6 (Bounded error propagation). *Suppose the error of the first regression (for residuals) be $|\beta - \beta'| \leqslant \epsilon$, then the error of the second regression (for cointegration test) is bounded by $\sqrt{L+2}\epsilon^2$ where L is the lag length in the ADF test.*

Proof of lemma 6. We can compute the error of residuals as follows: suppose that

$$u_t = X\beta - y \tag{31}$$

and

$$u'_t = X\beta' - y \tag{32}$$

are the residuals and estimated residuals, respectively. The error of the second regression variable u_t is

$$\begin{aligned} |u_t - u'_t| &= |(X\beta - y) - (X\beta' - y)| && (33)\\ &= |X(\beta - \beta')| && (34)\\ &\leqslant \epsilon. && (35)\end{aligned}$$

Here (35) follows from $\|X\| = 1$, and the errors of Δu_t can be calculated as:

$$\begin{aligned} |\Delta u_t - \Delta u'_t| &= |(u_t - u_{t-1}) - (u'_t - u'_{t-1})| && (36)\\ &\leqslant |u_t - u_{t-1}| + |u'_t - u'_{t-1}| && (37)\\ &\leqslant \epsilon + \epsilon = 2\epsilon. && (38)\end{aligned}$$

Regard these two error sequences as 2ϵ-bounded perturbation terms of the design matrix

$$\hat{U} = U + E, \tag{39}$$

in the second regression (30), by [54–56]'s work the error propagation is bounded as:

$$\|\gamma - \hat{\gamma}\| \leqslant \sum f_j^2 \|\delta_j\|. \tag{40}$$

Here $f_j = \sqrt{\gamma^2 + \sum c_j e_j^2}$ is the sensitivity of the dependence on the jth variable, and is bounded by $O(\sqrt{L + 2\epsilon})$. And δ_j is the error of jth term and hence is bounded by $O(\epsilon)$. Hence the total error propagation is bounded by $O(\sqrt{L + 2\epsilon^2})$. ■

Proof of theorem 3. With the facts above can the total complexity be calculated: the generation of the residuals will cost $O(\frac{d^{2.5}\kappa^3}{\delta^2}\text{poly}(\log_2 \frac{d\kappa}{\delta}) + dN)$; a second regression on residuals is implemented by *QLR* again with propagated error $\epsilon' = \sqrt{L + 2\epsilon^2})$, condition number κ' and $d = L + 2$, and by lemma 4, the complexity is $O(\frac{(L+2)^{2.5}\kappa'^3}{\delta'^2}\text{poly}(\log_2 \frac{(L+2)\kappa'}{\delta'}))$, where $\delta' = \min\{1/(L+2), \sqrt{L + 2\epsilon^2}\}$. The final complexity follows by a direct sum. ■

6. Realistic case analysis

This section will analyze the quantum advantage of QSA in the realistic financial scenario of US stock markets. There are mainly two kinds of characteristics data having significant influences on the algorithm complexity. One is the number of stocks: there are about 8000 stocks in the US stock markets. Another is the trading time. The regular trading time of the New York stock exchange and the NASDAQ are both 6.5 h per day. For the half-second time intervals aggregated quotes data, the length of data in one day is $N_0 = 6.5 \times 3600 \times 2 = 46\,800$. Furthermore there are about $l = 253$ trading days one year on average. Hence the typical size of the time series data can be computed as

$$N = N_0 \times l = 46\,800 \times 253 \approx 1.2 \times 10^7. \tag{41}$$

Under the consideration of the realistic case discussed above, there are mainly three reasons why QSA is more efficient than classical ones: first of all, in financial scenario, there are many different stocks, and it occupies only a tiny proportion of the searching space to find a multicollinearity portfolio out of thousands of stocks. The number of three-stock portfolios can exceed $M = C_{8000}^3 \approx 10^9$ while M_0, the number of multicointegrated pairs, is usually less than 1000. The proportion of non-multicollinear portfolios is estimated as

$$M_0/M \leqslant 10^{-6}. \tag{42}$$

By (41), the classical benchmark is $O(N^2 d) = 10^{18}$, and the average complexity of our algorithm is mainly determined by the first preselection subroutine with complexity $O(\sqrt{dN}\kappa_0^2 \log{(1/\epsilon)^2})) = 10^8$ (see details below). The primary reason for this acceleration is that the preselection procedure can search the multicollinearity without large matrix factorizations and regressions. Secondly, the problem size determined by sample number N is supposed to be very large for our problem of high-frequency trading: on the one hand, for high-frequency trading, there is a short time interval and a large number N_0 of trading date quotes of every single trading day. On the other hand, it does make sense in finance to consider a long time interval l since it is a statistical arbitrage model instead of some models for prediction such as momentum trading. Finally, for the specific case of statistical arbitrage trading strategy, it is common and unavoidable to handle matrices with large condition number κ, resulting in high cost of computing resources and time complexity. Utilizing the ability to detect κ by QCNCA, our algorithm is time variable one and adaptive to κ. Since most portfolios are with small κ as discussed above, giving a bound $\kappa_0 = 1000$, our algorithm's complexity is about $O(\sqrt{dN}\kappa_0^2 \log(1/\epsilon)^2)) = 10^11$.

The number of qubits needed can be estimated as follows: according to the data size discussed above, the qubits needed to prepare for the initial state is about $\log(1.2^*10^7) + \log 8000 \approx 35$. The $QCNC(\kappa, \phi)$ circuit consists of simplified phase estimation subcircuits, and each subcircuit with 0.1 precision needs more than 4 qubits. Moreover, the *VTPA* circuit consists of $QCNC(\kappa_j, \phi)$ circuits for different κ_j, and hence more than 50 qubits are needed, which are hard for us to simulate.

7. Conclusion

In this article, we introduce quantum algorithms for quantitative trading in the case of high-frequency statistical arbitrage and show the quantum advantage. Besides exploring new financial applications, two heuristic algorithms are also developed as instruments: one is for the estimation of the condition number of a given matrix, which has not been considered and proposed before as far as we know. This algorithm can be applied to solve other problems where condition number is a primary influencing factor of the algorithm's complexity, such as quantum computational fluid dynamics and differential equation solution [57–61]. The other is the implementation of statistical cointegration test, which has many applications in time series, finance analysis. Some modifications and exploration will be considered later to suit these exciting problems.

During the analysis of QCNCA and VTPA's complexity, we provide a theoretical framework and show the quantum advantage under the assumption of uniform distribution. Since the real problems are complicated, many other statistical models and different distributions will be taken into consideration. By some modification in equations (8)–(11), this method might still work with different results of complexity, and this is our further research direction. Moreover, the work of circuit simplification and simulation will be done in the future.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 12034018), and Innovation Program for Quantum Science and Technology No. 2021ZD0302300.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Appendix A. Cointegration test

In this appendix, some statistical concepts and facts about stochastic process and time series analysis are provided. Following those, an explicit demonstration is also given on the relationship between multicollinearity and cointegration, which may be confusing for some readers.

A (weakly) **stationary** time series, x_t, is a finite variance process with an unconditional joint probability distribution. Thus it does not change when shifted in time: (i) the mean value function $\mu_{xt} = E(x_t)$ is constant and (ii) the covariance function $\gamma_x(s, t) = E[(x_s - \mu_s)(x_t - \mu_t)]$ depends on s and t only through their difference $|s - t|$. In autoregressive-moving average models of unknown order, to test whether a given time series denoted as Y_t is stationary or not, the augmented Dickey–Fuller (ADF) unit root test may be employed [62].

Cointegration (multi-cointegration) is a relevant statistical property of two or more time series which are individually integrated of order d while their combination is integrated of order less than d. Here the **order of integration** is a summary statistic denoting the minimum number of differences taken to obtain a covariance-stationary series. Without loss of generality, $d = 1$ is assumed in this article. Under different financial hypotheses, there are mainly three kinds of cointegration tests: the Engle–Granger test [35], the Johansen test [63], and the Phillips–Ouliaris test [64]. In our work, Engle–Granger two-step method is used as the most popular and famous one:

Suppose that x_t^i are non-stationary and integrated of order $d = 1$, then a linear combination

$$\hat{u}_t = \sum \beta_i x_t^i$$

is expected to be stationary for some specific coefficient of β_i. In the general case that β_i is not decided yet, some estimation must be made first, usually by ordinary least squares regression. Next, the stationary test will be implemented on the residuals $\hat{u}_t$. It is a regression on $\hat{u}_t$, and the lagged residuals $\hat{u}_{t-1}$ are included as a regressor:

$$\Delta u_t = \alpha + \beta t + \gamma u_{t-1} + \sum_{i=1}^{p-1} \Delta u_{t-i} + \epsilon_t.$$

Here α and β are the intercept and the coefficient on the time trend, respectively, and p denotes the lag order of the autoregressive process to be decided. γ is the coefficient on the historical data u_{t-1}. The unit root test is carried out under the null hypothesis $\gamma = 0$ which means that the time series is integrated. The test statistic to be computed is

$$\mathbf{DF}_\tau = \frac{\hat{\gamma}}{\mathrm{SE}(\hat{\gamma})}.$$

What follows is a comparison with the Dickey–Fuller distribution critical value table [53].

Whenever such a cointegrated stock portfolio is found, the linear combination is expected to have the property of mean-reverting and can be utilized in statistical arbitrage.

Appendix B. Quantum linear regression

Quantum linear regression is the primary tool of QCT and is introduced as follows. Wiebe, Braun, and Lloyd (WBL) firstly introduced an algorithm for quantum data fitting [40]. Building on Harrow, Hassidim, and Lloyd's (HHL) quantum algorithm for linear systems of equations [39], WBL developed a least-squares estimation using Moore–Penrose pseudo inverse. WBL's algorithms are mainly suited for data sets whose design matrices are sparse and well-conditioned. Given an N dimension s sparse data matrix, the time complexity is $O(\log Ns^3\kappa^6\epsilon^{-1})$, where the condition number given is κ and the accuracy desired is ϵ^{-1}. With the technique of quantum principal component analysis (qPCA) and singular value decomposition (SVD) [65], Schuld, Sinayskiy, and Petruccione (SSP) came with an algorithm for prediction based on a linear regression model with least-squares optimization [52]. The sparseness condition is removed, and the existence of a low-rank approximation is supposed instead. The time complexity is $O(\log N\kappa^2\epsilon^{-3})$, where an improvement of factor κ^4 is made on the condition number at the cost of worse dependence on accuracy by a factor ϵ^{-2}. Recently, Guoming Wang presents a new quantum algorithm for fitting a linear regression model using least-squares approach [51]. This algorithm builds on Low and Chuang's method for Hamiltonian simulation based on qubitization and quantum signal processing [66, 67]. Childs, Kothari, and Somma (CKS)'s approach is introduced to inverse the matrix derived from SVD [49]. Imposing restrictions on the number of adjustable parameters d, and hence the rank of the design matrix, the gate complexity is $O(\frac{d^{1.5}\kappa^3}{\epsilon^2}\mathrm{poly}[\log_2(\frac{\kappa}{\epsilon\delta})])$ with the succeeding probability is at least $1-\epsilon$.

ORCID iDs

Xi-Ning Zhuang https://orcid.org/0000-0001-5118-5066
Zhao-Yun Chen https://orcid.org/0000-0002-5181-160X
Yu-Chun Wu https://orcid.org/0000-0002-8997-3030
Guo-Ping Guo https://orcid.org/0000-0002-2179-9507

References

[1] DiVincenzo D P 2000 *Fortschr. Phys.* **48** 771
[2] Kalai G 2011 arXiv:1106.0485
[3] Arute F *et al* 2019 *Nature* **574** 505
[4] McArdle S, Endo S, Aspuru-Guzik A, Benjamin S C and Yuan X 2020 *Rev. Mod. Phys.* **92** 015003
[5] Outeiral C, Strahm M, Shi J, Morris G M, Benjamin S C and Deane C M 2021 *Wiley Interdiscip. Rev.-Comput. Mol. Sci.* **11** e1481
[6] Emani P S *et al* 2021 *Nat. Methods* **18** 701–9
[7] Ma H, Govoni M and Galli G 2020 *npj Comput. Mater.* **6** 85
[8] Cao Y, Romero J and Aspuru-Guzik A 2018 *IBM J. Res. Dev.* **62** 6

[9] Orús R, Mugel S and Lizaso E 2019 *Rev. Phys.* **4** 100028
[10] Egger D J, Gambella C, Marecek J, McFaddin S, Mevissen M, Raymond R, Simonetto A, Woerner S and Yndurain E 2020 *IEEE Trans. Quantum Eng.* **1** 1–24
[11] Ceperley D and Alder B 1986 *Science* **231** 555
[12] Montanaro A 2015 *Proc. R. Soc.* A **471** 20150301
[13] Stamatopoulos N, Egger D J, Sun Y, Zoufal C, Iten R, Shen N and Woerner S 2020 *Quantum* **4** 291
[14] Martin A, Candelas B, Rodríguez-Rozas Á, Martín-Guerrero J D, Chen X, Lamata L, Orús R, Solano E and Sanz M 2019 arXiv:1904.05803
[15] Rebentrost P, Gupt B and Bromley T R 2018 *Phys. Rev.* A **98** 022321
[16] Woerner S and Egger D J 2019 *npj Quantum Inf.* **5** 15
[17] Rosenberg G, Haghnegahdar P, Goddard P, Carr P, Wu K and De Prado M L 2016 *IEEE J. Sel. Top. Signal Process.* **10** 1053
[18] Lopez de Prado M 2015 Generalized optimal trading trajectories: a financial quantum computing application *SSRN Electron. J.* https://doi.org/10.2139/ssrn.2575184
[19] Rosenberg G 2016 *1QB Information Technologies Write Paper* p 1
[20] Wittek P 2014 *Quantum Machine Learning: What Quantum Computing Means to Data Mining* (New York: Academic)
[21] Schuld M, Sinayskiy I and Petruccione F 2015 *Contemp. Phys.* **56** 172
[22] Lloyd S, Mohseni M and Rebentrost P 2013 arXiv:1307.0411
[23] Buhrman H, Cleve R, Watrous J and de Wolf R 2001 *Phys. Rev. Lett.* **87** 167902
[24] Krauss C 2017 *J. Econ. Surv.* **31** 513
[25] Kanamura T, Rachev S T and Fabozzi F J 2008 The application of pairs trading to energy futures markets Yale School of Management
[26] Gatev E, Goetzmann W N and Rouwenhorst K G 2006 *Rev. Financ. Stud.* **19** 797
[27] Vidyamurthy G 2004 *Pairs Trading: Quantitative Methods and Analysis* vol 217 (New York: Wiley)
[28] Caldeira J and Moura G V 2013 Selection of a portfolio of Pairs based on cointegration: a statistical arbitrage strategy *SSRN Electron. J.* 28
[29] Elliott R J, Van Der Hoek J and Malcolm W P 2005 *Quant. Finance* **5** 271
[30] Leber C, Geib B and Litz H 2011 *2011 21st Int. Conf. on Field Programmable Logic and Applications* (IEEE) pp 317–22
[31] Narang R K 2013 *Inside the Black Box: A Simple Guide to Quantitative and High Frequency Trading* vol 846 (New York: Wiley)
[32] Gomber P and Haferkorn M 2015 *Encyclopedia of Information Science and Technology* 3rd edn (Hershey, Pennsylvania: IGI Global) pp 1–9
[33] Trefethen L N and Bau D III 1997 *Numerical Linear Algebra* vol 50 (Philadelphia, PA: SIAM)
[34] Bookstaber R 2007 *A Demon of Our Own Design: Markets, Hedge Funds, and the Perils of Financial Innovation* (New York: Wiley)
[35] Engle R F and Granger C W J 1987 *Econometrica* **55** 251
[36] Giovannetti V, Lloyd S and Maccone L 2008 *Phys. Rev. Lett.* **100** 160501
[37] Giovannetti V, Lloyd S and Maccone L 2008 *Phys. Rev.* A **78** 052310
[38] Hong F-Y, Xiang Y, Zhu Z-Y, Jiang L-Z and Wu L-N 2012 *Phys. Rev.* A **86** 010306
[39] Harrow A W, Hassidim A and Lloyd S 2009 *Phys. Rev. Lett.* **103** 150502
[40] Wiebe N, Braun D and Lloyd S 2012 *Phys. Rev. Lett.* **109** 050505
[41] Kerenidis I and Prakash A 2016 arXiv:1603.08675
[42] Aldridge I 2013 *High-Frequency Trading: A Practical Guide to Algorithmic Strategies and Trading Systems* vol 604 (New York: Wiley)
[43] Weiming J M 2019 *Mastering Python for Finance: Implement Advanced State-Of-The-Art Financial Statistical Applications Using Python* (Birmingham: Packt Publishing Ltd)
[44] Ambainis A 2012 *STACS'12 (29th Symp. on Theoretical Aspects of Computer Science)* vol 14 (LIPIcs) pp 636–47
[45] Pesaran M H 2015 *Time Series and Panel Data Econometrics* (Oxford: Oxford University Press) pp 67–72
[46] Belsley D A, Kuh E and Welsch R E 2005 *Regression Diagnostics: Identifying Influential Data and Sources of Collinearity* vol 571 (New York: Wiley)
[47] Horn R A and Johnson C R 1985 *Matrix Analysis* (Cambridge: Cambridge University Press)
[48] Golub G H *et al* 1996 *Matrix Computations* 3rd edn (Baltimore, Maryland: JHU Press) pp 56–7
[49] Childs A M, Kothari R and Somma R D 2017 *SIAM J. Comput.* **46** 1920
[50] Berry D W, Childs A M and Kothari R 2015 *2015 IEEE 56th Annual Symp. on Foundations of Computer Science* (IEEE) pp 792–809
[51] Wang G 2017 *Phys. Rev.* A **96** 012335
[52] Schuld M, Sinayskiy I and Petruccione F 2016 *Phys. Rev.* A **94** 022342
[53] Fuller W A 1976 *Introduction to Statistical Time Series* (Hoboken, New Jersey: Wiley)
[54] Davies R B and Hutton B 1975 *Biometrika* **62** 383
[55] Beaton A E, Rubin D B and Barone J L 1976 *J. Am. Stat. Assoc.* **71** 158
[56] Stewart G W 1977 Sensitivity coefficients for the effects of errors in the independent variables in a linear regression *Tech. Rep.* Maryland Univ College Park Dept of Computer Science
[57] Rebentrost P, Mohseni M and Lloyd S 2014 *Phys. Rev. Lett.* **113** 130503
[58] Berry D W 2014 *J. Phys. A: Math. Theor.* **47** 105301
[59] Berry D W, Childs A M, Ostrander A and Wang G 2017 *Commun. Math. Phys.* **356** 1057
[60] Childs A M, Liu J-P and Ostrander A 2020 arXiv:2002.07868
[61] Clader B D, Jacobs B C and Sprouse C R 2013 *Phys. Rev. Lett.* **110** 250504
[62] Said S E and Dickey D A 1984 *Biometrika* **71** 599
[63] Johansen S and Juselius K 1990 *Oxf. Bull. Econ. Stat.* **52** 169–210
[64] Phillips P C B and Perron P 1988 *Biometrika* **75** 335
[65] Lloyd S, Mohseni M and Rebentrost P 2014 *Nat. Phys.* **10** 631
[66] Low G H and Chuang I L 2017 *Phys. Rev. Lett.* **118** 010501
[67] Low G H and Chuang I L 2019 *Quantum* **3** 163