Toward pricing financial derivatives with an IBM quantum computer

Ana Martin,[1,*] Bruno Candelas,[1,*] Ángel Rodríguez-Rozas,[2] José D. Martín-Guerrero,[3] Xi Chen,[1,4] Lucas Lamata,[5] Román Orús,[6,7,8] Enrique Solano,[1,4,7,9] and Mikel Sanz[1,7,9,†]

[1]*Department of Physical Chemistry, University of the Basque Country UPV/EHU, Apartado 644, 48080 Bilbao, Spain*
[2]*Risk Division, Banco Santander, Avenida de Cantabria S/N, 28660 Boadilla del Monte, Madrid, Spain*
[3]*IDAL, Electronic Engineering Department, University of Valencia, Avinguda de la Universitat s/n, 46100 Burjassot, Valencia, Spain*
[4]*International Center of Quantum Artificial Intelligence for Science and Technology (QuArtist) and Physics Department, Shanghai University, 200444 Shanghai, China*
[5]*Departamento de Física Atómica, Molecular y Nuclear, Universidad de Sevilla, 41080 Sevilla, Spain*
[6]*Donostia International Physics Center, Paseo Manuel de Lardizabal 4, 20018 San Sebastián, Spain*
[7]*IKERBASQUE, Basque Foundation for Science, Plaza Euskadi 5, 48009 Bilbao, Spain*
[8]*Multiverse Computing, Pio Baroja 37, 20008 San Sebastián, Spain*
[9]*IQM, Nymphenburgerstr. 86, 80636 Munich, Germany*

(Received 6 August 2019; revised 1 November 2019; accepted 18 December 2020; published 22 February 2021)

Pricing interest-rate financial derivatives is a major problem in finance, in which it is crucial to accurately reproduce the time evolution of interest rates. Several stochastic dynamics have been proposed in the literature to model either the instantaneous interest rate or the instantaneous forward rate. A successful approach to model the latter is the celebrated Heath-Jarrow-Morton framework, in which its dynamics is entirely specified by volatility factors. In its multifactor version, this model considers several noisy components to capture at best the dynamics of several time-maturing forward rates. However, as no general analytical solution is available, there is a trade-off between the number of noisy factors considered and the computational time to perform a numerical simulation. Here, we employ the quantum principal component analysis to reduce the number of noisy factors required to accurately simulate the time evolution of several time-maturing forward rates. The principal components are experimentally estimated with the five-qubit IBMQX2 quantum computer for 2×2 and 3×3 cross-correlation matrices, which are based on historical data for two and three time-maturing forward rates. This paper is a step towards the design of a general quantum algorithm to fully simulate on quantum computers the Heath-Jarrow-Morton model for pricing interest-rate financial derivatives. It shows indeed that practical applications of quantum computers in finance will be achievable in the near future.

DOI: 10.1103/PhysRevResearch.3.013167

I. INTRODUCTION

In finance, derivatives are contracts whose value derives from the value of an underlying financial asset or a set of assets, like an index, bonds, currency rates, stocks, market indices, or interest rates. Typical financial derivatives contracts include forwards, futures, swaps (currency swaps or interest rate swaps), caps, floors, and swaptions, among many others. They are typically used either to manage (mitigate) risk exposure (hedging), or for pure speculation. In the case of pricing interest-rate financial derivatives under the risk-neutral assumption [1,2], it is crucial to model accurately the time evolution of interest rates. Several stochastic dynamics have been proposed in the literature to model either the instantaneous interest rate $r(t)$ (also known as the instantaneous spot rate or, simply, as the short rate) or the instantaneous forward rate, which is the forward rate at a future, infinitesimal period $(T, T + \delta t)$ forecasted at a previous time t, denoted by $f(t, T)$ [1]. Simple dynamics based on one or two noisy (random) factors for modeling both the short rate and the forward rates have been proposed [1,3–5]. For short rates, one- and two-factor models became popular, such as the the Vasicek model, the Hull-White model, the Cox-Ingesroll-Ross (CIR) model and its CIR++ extension as one-factor models, and the Gaussian-Vasicek model and the Hull-White model, as two-factor models. Furthermore, their corresponding algorithms are straightforward to implement. However, these models suffer from the strong requirements which arise from the necessity to calibrate to market data and to capture, at the same time, correlation and covariance structures from the time evolution of different forward rates. A highly successful approach proposed to overcome these constraints is the celebrated Heath-Jarrow-Morton (HJM) framework [6–9], which directly models the time evolution of forward rates. Indeed, the HJM model is a general family of models from which

[*] These authors contributed equally to this work.
[†] Corresponding author: mikel.sanz@ehu.es

most of the aforementioned models may be derived [1]. To get a deeper understanding of this model, we refer the reader to Appendix A. In the HJM model, the dynamics is entirely specified by its volatility factors. Although general, there is a trade-off between the number of noisy factors considered and the computational time when executing the algorithm. Therefore, the computational power limits the accuracy of the model. Here is where a quantum computer becomes a useful tool. A quantum computer has a significantly bigger computational capacity than a classical one; therefore, using it would allow us to increase the accuracy of the HJM model.

Quantum computing (QC) has emerged in the past years as one of the most exciting applications of quantum technologies [10], which promises to revolutionize the computational power at our disposal. In QC, entanglement, probably the most characteristic signature of quantum physics, is employed as an extra resource to speed up the performance of the computation, since it allows us to parallelize the calculations. Multiple algorithms with provable quantum speedup with respect to their best classical counterparts have been proposed for prime factorization [11], searching in a list [12], solving systems of linear equations [13], and finding the largest eigenvalues and eigenvectors of a given matrix [14], among many others. A particularly relevant application is quantum simulation, in which a controllable quantum system simulates the dynamics of another quantum system of interest whose classical simulation would be highly inefficient. Examples of applications of quantum simulations can be found in spin systems [15,16], quantum chemistry [17–20], quantum field theories [21,22], fluid dynamics [23], and quantum artificial life [24–26]. Some applications of quantum technologies to finances have already been proposed [27–31], but only few experiments have been carried out so far [32–35]. State-of-the-art technology, however, only provides us with small noisy quantum chips, which limits the applicability of digital quantum simulations to toy models.

In this article, we employ an efficient quantum principal component analysis (qPCA) algorithm to effectively reduce the number of noisy factors needed to accurately simulate the joint dynamics of several time-maturing forward rates, according to the multifactor HJM model. Although this is a general implementation that can be run in any superconductive-circuit-based quantum processor, we have implemented this algorithm in the five-qubit IBMQX2 superconducting quantum processor of IBM due to its easy accessibility. The volatility factors,

$$\bar{\sigma}_i(\tau_j) = \sqrt{\lambda_i}(\mathbf{v}_i)_j, \tag{1}$$

where λ_i and $\mathbf{v}_i$ are the eigenvalues and eigenvectors of the covariance matrix, respectively, are estimated from 2×2 and 3×3 cross-correlation matrices between different time-maturing forward rates based on historical data. To illustrate our qPCA algorithm, we apply this technique to the covariance matrix appearing in Fig. 19.3 in Ref. [36], based on historical data for 1-, 3- and 6-month rates:

$$\sigma_3 = \begin{pmatrix} 0.000189 & 0.000097 & 0.000091 \\ 0.000097 & 0.000106 & 0.000101 \\ 0.000091 & 0.000101 & 0.000126 \end{pmatrix}. \tag{2}$$

This is, to our knowledge, both the first quantum computing experiment in financial option pricing and the largest implementation of the qPCA algorithm on a quantum platform. Although for small matrices the problem can be easily solved on a classical computer, this contribution represents a promising attempt towards the quantum computation of large-scale financial problems which are today prohibitively expensive. In the present noisy intermediate-scale quantum (NISQ) technology era [22], we extend the applications of quantum computers to the field of finance, paving the way for achieving useful quantum supremacy or advantage in the following years.

Quantum principal component analysis

Principal component analysis (PCA) is a mathematical technique which allows us to find the optimal low-rank approximation of a given matrix by computing its spectral decomposition in eigenvalues and eigenvectors. Indeed, this approximation discards the smallest eigenvalues of the matrix, keeping only the principal components of the spectral decomposition. This technique is of paramount importance for a variety of applications that go from dimensionality reduction problems to problems related to finding patterns in data of big dimension. Unfortunately, the computational cost is too high when the size of the matrix is elevated. It is in this context in which quantum algorithms and quantum computers may play a relevant role. Indeed, in Ref. [14] the authors provided an elegant quantum algorithm to perform PCA with an exponential speedup. The authors assumed that the matrix can be represented by a quantum state; i.e., it is a non-negative matrix with trace equal to 1, which covers a wide range of interesting cases, including the case under study in this paper of covariance matrices associated to volatilities.

In this article, we explore the use of this qPCA technique to effectively reduce the dimensionality of the HJM model, by reducing the number of noisy factors without detriment to its accuracy. The qPCA algorithm proposed is specially suited for this financial problem since it is only applicable to density matrices, such as the one given by the correlation matrix used as input for the HJM model. Finally, succeeding in reducing the effective number of noisy factors turns out to be critical for the construction of any quantum Monte Carlo algorithm for this model, after reducing the potential number of quantum gates of the resulting quantum circuit, thus alleviating the limitation in terms of decoherence times of the quantum processor.

We implement a slightly modified version of the aforementioned algorithm, which is better adapted to be run in a small and noisy quantum chip, typical in this NISQ technology era [37]. This allows us to reduce the number of noisy factors presented within the HJM model. This is the first step toward the construction of a general quantum computing algorithm to fully simulate the HJM model on the IBM quantum computer for pricing interest-rate financial derivatives. In the following section, we briefly describe the algorithm.

II. QUANTUM CIRCUIT

Let us consider a non-negative matrix $\sigma_N \in \mathbb{R}^N \times \mathbb{R}^N$ with $\mathrm{tr}[\sigma_N] = 1$, which is the matrix whose principal components we want to compute. Let us assume that we can efficiently

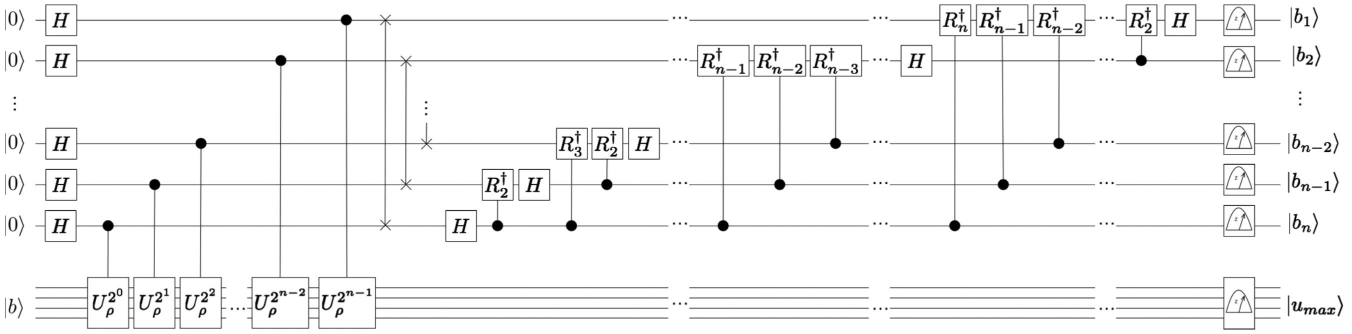

FIG. 1. Quantum circuit implementation for $n + \log N$ qubits. The first n qubits are dedicated to the binary codification of the maximum eigenvalue of the matrix σ_N and they are initialized in the site $|0\rangle$. The rest of the qubits, a total of $\log N$, encode the estimation of the corresponding eigenvector and are initialized on a random state $|b\rangle$. The single qubit gate H corresponds to the Hadamard gate. The rest of the gates are controlled operations. The controlled $U_\rho^{2^k}$ gate applies the matrix $U = e^{it\sigma_N}$ 2^k times on the last set of qubits. The controlled $R_k^\dagger$ gate applies the matrix $\begin{pmatrix}1 & 0\\ 0 & e^{2\pi i/2^k}\end{pmatrix}$ to each target qubit. After performing all the operations, if the initial $|b\rangle$ is appropriate, one gets the final state $|b_1 b_2 \cdots b_n\rangle \otimes |u_{\max}\rangle$, where $|b_1 b_2 \cdots b_n\rangle$ is the n-bit estimation of the eigenvalue $\lambda_j^{(n)}$ and $|u_{\max}\rangle$ is the best estimation of the exact eigenvector of σ_N.

generate the unitary $e^{it\sigma_N}$. It has been proven in the literature that, under certain conditions such as sparsity of the matrix [38,39] or access to several copies of σ_N [14], this is possible. In our case the covariance matrix σ_N is not sparse, but we can access several copies of it codified in quantum states. Thus, the best way to efficiently generate the unitary operation $e^{it\sigma_N}$ with accuracy ϵ in $\mathcal{O}(t^2\epsilon^{-1})$ steps is the one described by Lloyd *et al* in Ref. [14]. This matrix admits a spectral decomposition $\sigma_N = \sum_{j=1}^N \lambda_j |u_j\rangle\langle u_j|$, with $0 \leqslant \lambda_j \leqslant 1$ and $\sum_{j=1}^N \lambda_j = 1$, and we assume that σ_N can be very well approximated by a matrix $\rho_r = \sum_{j=1}^r \lambda_j |u_j\rangle\langle u_j|$ with rank $r \ll N$. Therefore, the goal of the algorithm is the determination of the r largest eigenvalues of σ_N and their corresponding eigenvectors. If we want to determine the eigenvalues with an n-bit precision, we will need $n + \log N$ qubits, as depicted in Fig. 1, which represents the gate decomposition of the algorithm. *A priori*, we do not know the eigenvectors of our algorithm. Hence, we cannot make use of quantum phase estimation to compute directly the corresponding eigenvalue. Consequently, we initialize our system in a random state $|b\rangle$ whose (unknown) decomposition in terms of the eigenbasis is given by $|b\rangle = \sum_{j=1}^N \beta_j |u_j\rangle$. If we take a random vector, the probability that there exists a component $\beta_k = 0$ is zero. The quantum state after the quantum Fourier transform can be written as $|\Psi_b\rangle = \sum_{j=1}^N \beta_j |\Lambda_j^{(n)}\rangle \otimes |u_j\rangle$, so eigenvalues and eigenvectors are entangled. $\Lambda_j^{(n)} = 0.b_1 b_2 \cdots b_n$ is the n-bit precision binary representation of the jth eigenvalue of ρ_r. However, if our assumption that σ_N is well approximated by the r-rank matrix ρ_r is correct, then the highest eigenvalues should be around $1/r \approx \sum_{k=1}^n y_k 2^{-k}$. Calling $|y^{(n)}\rangle = |y_1 y_2 \cdots y_n\rangle$ the vector of these components, it means that, by projecting the eigenvalue component $|\Lambda_j^{(n)}\rangle$ of the state $|\Psi_b\rangle$ around this component, one may obtain the eigenvector corresponding to the maximum eigenvalue, i.e., $\langle y^{(n)}| \otimes \mathbb{1}|\Psi_b\rangle \approx |u_{\max}\rangle$. It is possible, especially when n is small, as may happen in the NISQ chips, that the n-bit approximation of the eigenvalue cannot be able to distinguish between two or more eigenvectors. In this case, the projection is not into the maximum eigenvalue, but into a K-dimensional subspace containing the indistinguishable components $\langle y^{(n)}| \otimes \mathbb{1}|\Psi_b\rangle = \sum_{j=1}^K \tilde{\beta}_j |u_j\rangle$, where the $\tilde{\beta}_j$ are the normalized β_j in the subspace. As we do not know *a priori* whether $K > 1$ or not, we could start with a different random state $|c\rangle = \sum_{j=1}^N \gamma_j |u_j\rangle$, which leads to $|\Psi_c\rangle = \sum_{j=1}^N \gamma_j |\Lambda_j^{(n)}\rangle \otimes |u_j\rangle$. After projecting into $|y^{(n)}\rangle$ the expected state is a different superposition $\sum_{j=1}^L \tilde{\gamma}_j |u_j\rangle$ with high probability, which helps us to check whether we have actually identified the eigenvector corresponding to the maximum eigenvalue. Otherwise, we must increase the n-bit precision until a unique eigenvalue is identified.

Let us assume now that the n-bit precision is sufficient to determine a unique eigenvector. Taking into account the constraints due to the small number of qubits and the noise of the chip and the operations, we can sequentially improve the result of the eigenvector. As described above, we start the protocol with a random quantum state $|b_0\rangle$, to which the noisy algorithm is applied and the projection into the $|y^{(n)}\rangle$ subspace is performed. Let us call the result $|\Psi_{b_0}\rangle$, which is an approximation for the eigenvector. If we employ now this state as the initial state in the protocol, $|\Psi_{b_0}\rangle = |b_1\rangle$, then one expects that the approximation for the eigenvector provided by $|\Psi_{b_1}\rangle$ improves the fidelity due to the cancellation of coherent errors associated to the β components. Nonetheless, there is a limitation in this sequential improvement related to the decoherence of the qubits and the statistical error of the measurement. In any case, the result can be (slightly) further improved by performing measurements in different bases and averaging, since this cancels some systematic errors of the gates.

III. RESULTS

As described in the previous section, the protocol is divided into two parts. First, we estimate the eigenvector $|u_{\max}\rangle$ corresponding to the largest eigenvalue $\lambda_{\max}$. We start with a random state, apply the circuit implementation shown in Fig. 2, project on the binary n-bit estimation for the largest eigenvalue $|y^{(n)}\rangle$, and use this state as the initial state of the process, which sequentially approaches the exact eigenvector. Afterwards, we use this eigenvector to get a more accurate

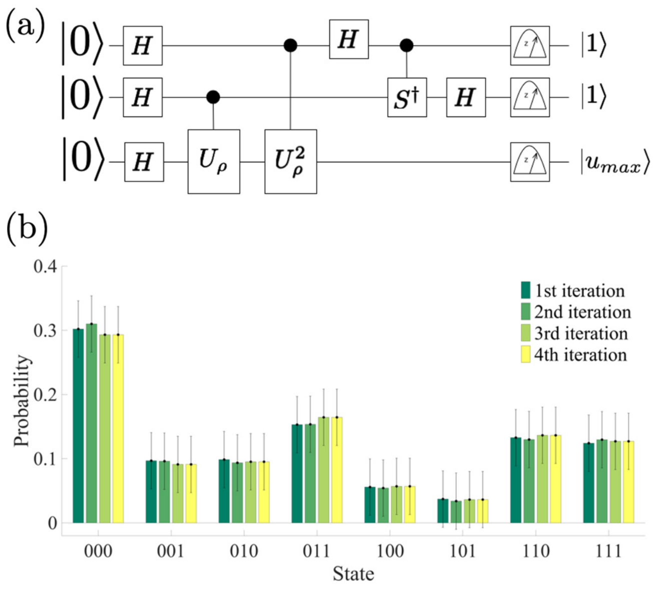

FIG. 2. (a) Quantum circuit implementation for the 2 × 2 matrix. The first two qubits encode the 2-bit estimation of the greatest eigenvalue of ρ_2 and are initialized on the state $|0\rangle$. The last qubit is dedicated to the estimation of the corresponding eigenvector. It is initialized on a random state $|b\rangle$. For the first iteration, we initialize it on the state $|+\rangle$ by applying a Hadamard gate. The single-qubit gates represented by the letter H refer to the Hadamard gate. The controlled U gates represent the unitary controlled operations called $U_\rho^{2^k}$ in Fig. 1. The last two-qubit gate is a controlled $S^\dagger$ gate, controlling the first qubit and acting on the second. The final state of the system after running the circuit and taking measures is $|11\rangle \otimes (0.719|0\rangle + 0.659|1\rangle)$. (b) Populations for each iteration. The graphic shows the experimental probabilities of finding the three qubits in each state and its corresponding errors for the four iterations of the algorithm. We have considered both statistical and experimental errors, assuming for the latter an error of 8% for each two-qubit gate.

approximation for the eigenvalue $\lambda_{\max}$ by means of quantum phase estimation.

For the estimation of the eigenvector, we start with a random state $|b_0\rangle$. Hence, the initial state of the system is $|0\rangle \otimes |0\rangle \otimes |b_0\rangle$. After the first iteration and projecting on the computational basis the eigenvector, we obtain a first estimation, which we call $|b_1\rangle$, and use it as the initial state of the system on the next iteration. This is $|0\rangle \otimes |0\rangle \otimes |b_1\rangle$. We continue this process and iterate k times until $|b_{k-1}\rangle \approx |b_k\rangle$. Once we reach that point, we can say that $|b_k\rangle \approx |u_{\max}\rangle$.

Let us now estimate the eigenvalue $\lambda_{\max}$. Once the first part is finished and we have an accurate approximation for $|u_{\max}\rangle$, we can apply quantum phase estimation [10] to obtain $\lambda_{\max}$ with n-bit precision. The precision is limited in this case by the size of the processor. Our aim is to apply the algorithm to the 3 × 3 matrix given in Eq. (2) in the five-qubit IBMQX2 quantum processor. First, we solve the 2 × 2 submatrix of σ_3 containing only two maturities, and afterwards, we solve the 4 × 4 expansion of the same matrix. Despite the small size of the problem, the volume of the quantum algorithms allowed in this processor is almost achieved, but we can still obtain relatively accurate results. We have run the algorithm in both the simulator provided by QISKIT [40] and the real IBM quantum processor, reaching accurate results in both cases.

A. 2 × 2 matrix

First, we need to codify the covariance matrix in a quantum state, so we only need to normalize it with respect to its trace,

$$\rho_2 = \frac{\sigma_2}{\mathrm{tr}(\sigma_2)} = \begin{pmatrix} 0.6407 & 0.3288 \\ 0.3288 & 0.3593 \end{pmatrix}, \tag{3}$$

whose spectral decomposition is given by

$$\lambda_1 = 0.8576, \qquad |u_1\rangle = 0.8347|0\rangle + 0.5508|1\rangle, \tag{4}$$

$$\lambda_2 = 0.1424, \qquad |u_2\rangle = 0.5508|0\rangle - 0.8347|1\rangle. \tag{5}$$

Let us remark that $\lambda_{\max} \gg \lambda_2$, a usual characteristic of these correlation matrices, so we can apply the PCA technique to find the optimal low-rank approximation of ρ_2. Let us now define the unitary

$$U_{\rho_2} = e^{2\pi i \rho_2} = \begin{pmatrix} 0.6260 - 0.3068i & -0.7170i \\ -0.7170i & 0.6260 + 0.3068i \end{pmatrix}. \tag{6}$$

For the first part of the protocol, we make use of three qubits, two for a 2-bit approximation of the eigenvalue, and a third one one to represent the eigenvector. We apply the first part of the protocol as described above, starting with a quantum state $|b_0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ and projecting into the $|y^{(n)}\rangle = |11\rangle$ state. After the fourth iteration, each of them averaged over 8192 realizations, the outcome vector estimating the eigenvector stabilizes and we stop. With this final eigenvector, we also rotate the measurement basis in x, y, and an arbitrary direction $r = (\cos\alpha, -e^{i\beta}\sin\alpha; e^{i\beta}\sin\alpha, e^{i\gamma}\cos\alpha)$ to compute the relative phase and to improve the accuracy of the solution provided. We have chosen the set of angles $\alpha = 1.00$, $\beta = 0.80$, and $\gamma = 0.16$, but any other choice would be valid as long as all of the angles are different from zero. Our estimation for $|u_{\max}\rangle$ is consequently given by

$$\begin{aligned} |u_{\max}\rangle = {} & [(0.87 \pm \delta) - i(0.10 \pm \delta)]|0\rangle \\ & + [(0.47 \pm \delta) + i(0.10 \pm \delta)]|1\rangle, \end{aligned} \tag{7}$$

with $\delta = 0.9$ the error estimated from the two-qubit gates and measurement fidelity provided by IBM and the statistical error related to the number of repetitions. This δ should be understood as an upper bound of all possible errors, including not only the gate and qubit errors provided by IBMQ, but also other errors, which do not increase with the depth of the algorithm, such as T_1 and T_2, measurement errors, crosstalk, non-Markovian errors, etc. As we had no direct access to the processor, it is not possible to distinguish among them, so we consider the worst possible case. In Appendix B we provide more information about how we have estimated the error δ. Let us remark that we have split the complex phase between both states using the global phase. The estimation for the coefficients after each iteration in the z basis is provided in Table I. We can observe that the algorithm has already converged in the first iteration, and the variations are within the estimated error. We take the eigenvector produced in the last iteration and repeat the algorithm with this one as the initial state measuring in x, y, and r-random directions to check possible relative phases and to try to remove systematic errors, which yields the states $|b_x\rangle =$

TABLE I. Estimated coefficients of the eigenvector for consecutive iterations of the algorithm in modulus and measured in the z basis. Here, the state of the previous iteration is employed as the initial state in the following iteration until the values are stabilized. Measurements of the eigenvector are performed in the z basis and repeated for 8192 realizations.

Iteration	c_0^z	c_1^z
1	0.719	0.695
2	0.707	0.707
3	0.720	0.694
4	0.680	0.734

$0.878|0\rangle + (0.421 + i0.230)|1\rangle$, $|b_y\rangle = 0.878|0\rangle + (0.427 + i0.220)|1\rangle$, and $|b_r\rangle = 0.985|0\rangle + 0.175|1\rangle$.

Let us remark that the previous estimation of the eigenvector was performed by projecting into the subspace estimating the eigenvalue into the 2-bit string $\Lambda_{\max} = 0.11$. However, we can now apply quantum phase estimation to improve the estimation for the eigenvalue. We divide the problem into these two stages for two reasons. First, we do not know *a priori* the value of the maximum eigenvalue, only the approximate rank, and hence a low n-bit approximation covers a larger range, as explained in the previous section. Additionally, we observe a lower error when the protocol is performed in this manner, probably due to the accumulation of two-qubit gates and the error in the projection for the eigenvalue estimation. However, we cannot be sure, since IBM does not provide the exact quantum circuit which they are performing in the processor.

Let us now use three qubits for the eigenvalue estimation $\Lambda_{\max} = 0.b_1b_2b_3$, keeping one qubit to encode the corresponding eigenvector. The depth of the circuit implementation grows and leads us to the decoherence of the system when we run it on the real quantum processor, as depicted in Fig. 3. However, the result provided by the QISKIT simulator, producing the quantum state $|111\rangle \otimes [0.8150|0\rangle + 0.5794|1\rangle]$, is an almost ideal result for the 3-bit string estimation of the eigenvalue. Indeed, the predicted eigenvalue is $\Lambda = 0.111$ in binary representation and corresponds to the number $\lambda = 0.875$ and the fidelity between $|u_{\max}\rangle$ and the one obtained after performing the quantum phase estimation in the QISKIT simulator $|u_{\mathrm{QPE}}\rangle$ is

$$F = |\langle u_{\mathrm{QPE}}|u_{\max}\rangle|^2 = 0.965. \tag{8}$$

This shows that, with few improvements in the gates and chips or with a lower level programming in the chip, one could substantially improve the results.

B. 4 × 4 matrix

In this case, the matrix σ_3 will be represented by the two-qubit quantum state

$$\rho_4 = \frac{\sigma_4}{\mathrm{tr}(\sigma_4)} = \begin{pmatrix} 0.4489 & 0.2304 & 0.2162 & 0 \\ 0.2304 & 0.2518 & 0.2399 & 0 \\ 0.2162 & 0.2399 & 0.2993 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}. \tag{9}$$

Thus, the unitary generated, $U_{\rho_4} = e^{2\pi i \rho_4}$, is given by

$$U_{\rho_4} = \begin{pmatrix} 0.415 + 0.048i & -0.108 - 0.566i & -0.029 - 0.702i & 0 \\ -0.108 - 0.566i & 0.744 - 0.030i & -0.285 - 0.181i & 0 \\ -0.029 - 0.702i & -0.285 - 0.181i & 0.618 + 0.099i & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

The spectral decomposition of ρ_4, for the sake of comparability, is given by

$$\begin{aligned} \lambda_1 &= 0.000, & |u_1\rangle &= (0.000, 0.000, 0.000, 1.000), \\ \lambda_2 &= 0.031, & |u_2\rangle &= (-0.119, 0.786, -0.607, 0.000), \\ \lambda_3 &= 0.169, & |u_3\rangle &= (0.734, -0.342, -0.587, 0.000), \\ \lambda_4 &= 0.800, & |u_4\rangle &= (0.669, 0.516, 0.536, 0.000), \end{aligned}$$

where the vectors are expressed in the basis $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$. This problem is much more complicated than the previous one, since we do not implement U_{ρ_4}, but the controlled U_{ρ_4}. This matrix must be decomposed in terms of two-qubit gates, which dramatically increases the depth of the algorithm and, consequently, the decoherence and the errors. The quantum circuit implementation for this problem is shown in Fig. 4. Following the aforementioned protocol, we start with the state $|b_0\rangle = (|00\rangle + |01\rangle + |10\rangle + |11\rangle)/2$ and provide the coefficients in the z basis, for both the simulator and the real processor, in Table II.

Afterwards, we measure in different bases in order to compute the relative phases, and take the average to cancel systematic errors. The estimation of the eigenvector is, therefore,

$$\begin{aligned} |u_{\max}\rangle = {} & (0.6287 + i0.3991)|00\rangle + (0.4010 + i0.0693i)|01\rangle \\ & + (0.4807 - i0.1964)|10\rangle + (0.0305 + i0.0959)|11\rangle. \end{aligned}$$

The number of entangling gates performed for this algorithm is at least 18, so the total estimated error δ in the coefficients, assuming the 8% error per gate observed in the previous section, is over 100%, which makes in principle the result meaningless.

A possible way to address this issue is to make use of some error mitigation methods [41], similar to how it is done in Ref. [32]. By measuring in different bases, we have mitigated the systematic errors associated to measurements. However, the errors that occur during the quantum circuit can be mitigated by employing, for instance, Richardson's extrapolation

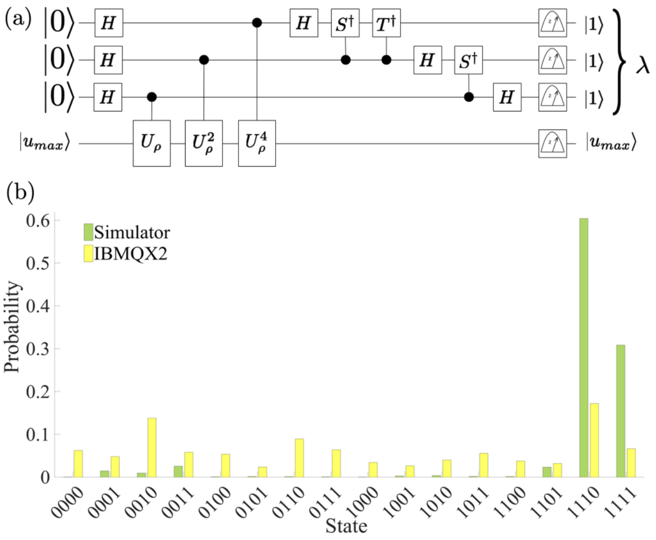

FIG. 3. (a) Quantum phase estimation circuit implementation for the 3-bit estimation of the greatest eigenvalue, $\Lambda^{(3)}$. After estimating the eigenvector $|u_{\max}\rangle$, it is used to improve the estimation of the corresponding eigenvalue λ by applying the quantum phase estimation algorithm. In this case we dedicate three qubits to the binary codification of λ. The fourth qubit is initialized on the estimated eigenvector $|u_{\max}\rangle$. The fifth two-qubit gate represents a controlled $T^\dagger$ gate, controlling the first qubit and acting on the third one. The rest of the gates are the same that have been applied on the previous part of the algorithm. Finally, one takes measures and gets the final state of the system: $|\Lambda^{(3)}\rangle \otimes |u_{\max}\rangle = |111\rangle \otimes [0.8150|0\rangle + 0.5794|1\rangle]$. (b) Populations of the 3-bit eigenvalue estimation. This chart shows the probabilities of each state in the QISKIT simulator (green bars) and in the real quantum processor (yellow bars) for the quantum phase estimation algorithm taking the previously obtained eigenvector given in Eq. (C1). The first qubit refers to the subspace of the eigenvector estimation. The next three qubits refers to the subspace of the binary estimation of the eigenvalue. The quantum circuit comprises at least six entangling gates, which leads the system to an almost total decoherence, as reflected in the homogeneous distribution of probabilities in the real chip.

[42] to eliminate powers of the noise perturbation. Additionally, to improve the error cancellation during measurements, readout error mitigation [43] could be applied.

IV. CONCLUSIONS

We have proposed and implemented an efficient quantum algorithm to reduce the number of noisy factors present in the time evolution of forward rates according to the multifactor Heath-Jarrow-Morton model. Indeed, this model considers several noisy components to accurately describe the dynamics of several time-maturing forward rates, which can be gathered in a cross-correlation matrix. The eigenvectors corresponding to the largest eigenvalues of this matrix provide the principal components of the correlations. When the considered data set is large, this calculation turns out to be challenging. The principal components are experimentally estimated using a hybrid classical-quantum algorithm with the five-qubit IBMQX2 quantum computer for 2 × 2 and 3 × 3 cross-correlation matrices, which are based on historical data for two and three time-maturing forward rates. We have obtained a reasonable approximation for both the maximum eigenvalue and its corresponding eigenvector in the 2 × 2 case. For the 4 × 4 matrix, the depth of the algorithm is too high and the experimental errors in the quantum processor prevent us from extracting any useful information. For the 4 × 4 matrix, the depth of the algorithm is too large and the experimental errors in the quantum processor prevent us from extracting any useful information. The application of some error mitigation techniques could be a solution for this problem. Simultaneously, the simulation in QISKIT shows that it would be achievable in a better experimental set. This means that we have exhausted the computational power provided by the current quantum processor in terms of gate fidelities, connectivity, and number of qubits. Nonetheless, this paper is a step towards the design of a general quantum algorithm to fully simulate on quantum computers the HJM model for pricing interest-rate financial derivatives, and shows that practical applications of quantum computers in finance will be achievable even in the NISQ technology era.

The codes we have used to run our experiments both on the real chip and on the QISKIT [40] built-in simulator are available on the following GitHub public repository [44]. These codes have been written according to the 0.7.1 version of QISKIT.

ACKNOWLEDGMENTS

The authors acknowledge the use of IBM QISKIT for this work. The views expressed are those of the authors and do not reflect the official policy or position of IBM. We also acknowledge funding from projects QMiCS (Grant No. 820505) and OpenSuperQ (Grant No. 820363) of the EU Flagship on Quantum Technologies; the FET-OPEN project QuroMorphic, Spanish Government, Grant No. PGC2018-095113-B-I00 (MCIU/AEI/FEDER, UE); Basque Government, Grant No. IT986-16; Spanish Ramón y Cajal Grant No. RYC-2017-22482; Shanghai Municipal Science and Technology Commission (Grants No. 18010500400 and No. 18ZR1415500); and the Shanghai Program for Eastern Scholar. This work is supported by the US Department of Energy, Office of Science, Office of Advanced Scientific Computing Research (ASCR) quantum algorithm teams program, under field work Proposal No. ERKJ333. We also acknowledge support from NSFC (Grant No. 12075145), STCSM (Grants No. 2019SHZDZX01-ZX04, No. 18010500400, and No. 18ZR1415500), and from Spanish Government PID2019-104002GB-C21, PID2019-104002GBC22 (MCIU/AEI/FEDER, UE).

A.M., as the first author, has been responsible for the development of this work with the support of B.C. M.S. suggested the seminal ideas and the adaptation of the algorithm to a NISQ chip. A.R.-R. helped with the financial background and the comparison with the classical results. J.M.-G., X.C., L.L., R.O., and E.S. helped to improve the ideas and results shown in the paper. All authors have carefully proofread the manuscript. M.S. supervised the project throughout all stages.

The authors declare that there are no competing interests.

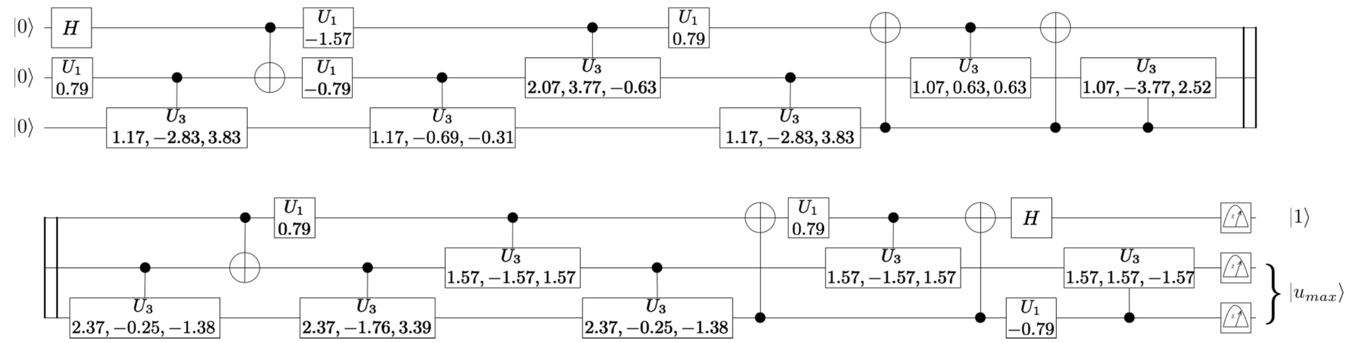

FIG. 4. Quantum circuit implementation for the 4 × 4 matrix. The first qubit is the only one dedicated for the binary codification of the greatest eigenvalue λ of the matrix ρ_4. It is initialized on the state $|0\rangle$. The last two qubits encode the estimation of the corresponding eigenvector and are initialized on a random state $|b\rangle$. The final state of the system after measuring is $|1\rangle \otimes |u_{\text{max}}\rangle$.

APPENDIX A: THE HEATH-JARROW-MORTON FRAMEWORK

In Sec. I, we introduced the HJM model to model the time evolution of forward rates. Here we present a brief description of such a framework.

Calculating the fair price of financial derivatives represents a challenge from both the mathematical modeling and the computational power perspectives. Traditionally, simplistic assumptions on the dynamics of financial markets in terms of mathematical pricing models, such as the Black-Scholes model, provided simple analytical tools for both pricing and hedging purposes. However, these simple models have limited features in terms of modeling and, in particular, do not properly account for all the relevant financial risk factors. To solve this issue, sophisticated models such as the HJM framework considered in this work gained increasing interest, requiring one, however, to resort to numerical techniques due to the fact that its solution (price) is no longer available analytically.

Solving sophisticated pricing models that account for multiple risk factors using numerical techniques represents a formidable computational task. The most popular technique used by the global financial community is based on Monte Carlo, which, due to its low convergence rate, requires a huge amount of computational resources in order to meet acceptable levels of accuracy. Generally speaking, designing an efficient Monte Carlo algorithm requires two subsequent stages: first, to reduce the computational complexity of the model when possible, resorting to dimensionality reduction techniques, and second, to accelerate the Monte Carlo convergence from the resulting reduced model. Quantum computing appears as a promising alternative to speed up both stages.

TABLE II. Estimated coefficients of the eigenvector for consecutive iterations of the algorithm in modulus and measured in the z basis. Here, the state of the previous iteration is employed as the initial state in the following iteration until the values are stabilized. Measurements of the eigenvector are performed in the z basis and repeated for 8192 realizations.

Iteration	c_{00}^{z}	c_{01}^{z}	c_{10}^{z}	c_{11}^{z}
		Chip		
1	0.542	0.503	0.466	0.487
2	0.531	0.498	0.493	0.477
3	0.543	0.493	0.494	0.468
4	0.502	0.492	0.523	0.482
		Simulator		
1	0.719	0.695	0.695	0.695
2	0.707	0.707	0.695	0.695
3	0.720	0.694	0.695	0.695
4	0.680	0.734	0.695	0.695

A T-maturity zero-coupon bond (also known as a pure discount bond) is a contract that ensures its investor to accrue one unit of currency at time T (its maturity), whose price at a previous time t is denoted by $P(t, T)$. From this definition, it is clear that at expiry of the contract we must have $P(T, T) = 1$. This time-dependent curve represents a fundamental element in the theory of risk-neutral derivative pricing [1] and will extensively be used throughout this article. $P(t, T)$ is also known as the curve of discount factors, since it is employed to calculate the present value of future cash flows. The inverse of this amount is called the capitalization factor, providing the capitalization of a present quantity to a future time.

In finance, the instantaneous interest rate r_t (also known as the instantaneous spot rate, or simply the short rate) is the rate of return of a risk-free investment at time t (for example, a U.S. Treasury bond) (see Ref. [1]). This is also the interest rate applied when borrowing money from the money market and it is given as an annual percentage. We denote by $B(t)$ the time-t value $B(t)$ of the money market account, defined as

$$B(t) = \exp\left(-\int_0^t r(s)ds\right). \tag{A1}$$

Of course, if today is the time t, the value of $r(t)$ can be observed in the money market and, therefore, it constitutes a known value. However, for future times $T > t$, $r(T)$ is uncertain and modeled through a stochastic process. In the risk-neutral framework, when using the money market account $B(t)$ as numéraire, the link between the short rate and the zero coupon is indeed materialized by the risk-neutral pricing formula

$$P(t, T) = \mathbb{E}^{\mathbb{Q}_B}\left[\frac{B(t)}{B(T)} \times 1|\mathcal{F}_t\right] = \mathbb{E}^{\mathbb{Q}_B}\left[e^{\left(-\int_t^T r(s)ds\right)}|\mathcal{F}_t\right], \tag{A2}$$

where $\mathbb{Q}_B$ is the equivalent martingale measure associated to the numéraire $B(t)$, and $\mathcal{F}_t$ denotes the filtration of the information observed in the market until time t. Therefore, if

TABLE III. Complete outcome of each iteration of the first part of the algorithm. The first two qubits correspond to the state $|\lambda^{(n)}\rangle$, which encodes the 2-bit estimation of the largest eigenvalue $\lambda_{\max}$. The last qubit corresponds to the estimation of the corresponding eigenvector $|u_{\max}\rangle$. Each experiment has been run a total of 8192 times. The table shows both the number of times the experiment ends up on each state (under the heading "Counts") and the percentage it represents over the total number of runs ("%").

	Simulator		Chip	
State	Counts	%	Counts	%
		First iteration		
000	2205	27	2474	30
001	428	5	793	10
010	181	2	808	10
011	558	7	1253	15
100	325	4	457	6
101	316	4	304	4
110	2692	33	1087	13
111	1487	18	1016	12
		Second iteration		
000	2190	27	2538	31
001	422	5	787	10
010	181	2	766	9
011	516	6	1257	15
100	297	4	443	5
101	264	3	277	3
110	2757	34	1063	13
111	1565	19	1061	13
		Third iteration		
000	2196	27	2401	29
001	418	5	746	9
010	169	2	779	10
011	510	6	1347	16
100	292	4	465	6
101	287	4	297	4
110	2755	34	1117	14
111	1565	19	1040	13
		Fourth iteration		
000	2179	27	2182	27
001	414	5	671	8
010	191	2	754	9
011	469	6	1100	13
100	317	4	605	7
101	306	4	400	5
110	2801	34	1145	14
111	1515	18	1335	16

today is the time t, then $P(t,T)$ is deterministic and it should match the information observed in the market. However, at any future time t_f from today, $t < t_f < T$, $P(t_f,T)$ represents a random variable whose value is model dependent.

Models for short rate are typically classified depending upon the number of noisy factors that defines their dynamics. Popular one-factor short-rate models include the Vasicek, the Hull-White model, and the CIR model and its CIR++ extension, among others. They quickly became of lesser interest due to their limitation when pricing financial instruments whose payoffs involve the joint distribution of several of such rates at different maturities, mainly due to their incapability to exhibit the intrinsic decorrelation among them. Motivated from this observation, multifactor models appeared to enrich the correlation structure. As a result, several two-factor models were proposed, such as the Gaussian-Vasicek model and the Hull-White two-factor model.

Despite the freedom when modeling the instantaneous short rate in the models mentioned above, some limitations may appear when attempting to calibrate a particular model to the current (observed) market curve of discount factors and to capture, at the same time, the correlation and covariance structure of forward rates. The first sound alternative to short-rate models was introduced by Heath, Jarrow, and Morton in 1992 [8], developing a general framework for modeling the instantaneous forward rates. In its multifactor version, determining the number of noisy factors needed becomes a trade-off between the ability, with increasing noisy factors, to better reproduce correlation and covariance structures while capturing market data, and the computational cost when performing a numerical simulation.

The connection between the forward rates $f(t,T)$ and the short rate $r(t)$ is established through the bond price as

$$f(t,T) = -\frac{\partial}{\partial T}\log P(t,T). \tag{A3}$$

When $f(t,T)$ is known for all T, we must have

$$P(t,T) = e^{-\int_t^T f(t,s)ds}. \tag{A4}$$

By differentiating Eq. (A2) with respect to T we obtain

$$-\frac{\partial P(t,T)}{\partial T} = \mathbb{E}^{\mathbb{Q}_B}\left[\exp\left(-\int_t^T r(s)ds\right)r(T)\middle|\mathcal{F}_t\right]. \tag{A5}$$

By changing to the T-forward measure $\mathbb{Q}_T$ associated to the bond price numéraire $P(t,T)$ we have

$$\begin{aligned}
-\frac{\partial P(t,T)}{\partial T} \\
&= \mathbb{E}^{\mathbb{Q}_T}\left[\exp\left(-\int_t^T r(s)ds\right)r(T)\frac{P(t,T)}{\exp\left(-\int_t^T r(s)ds\right)}\middle|\mathcal{F}_t\right] \\
&= P(t,T)\mathbb{E}^{\mathbb{Q}_T}[r(T)|\mathcal{F}_t],
\end{aligned} \tag{A6}$$

and therefore,

$$f(t,T) = \mathbb{E}^{\mathbb{Q}_T}[r(T)|\mathcal{F}_t]. \tag{A7}$$

As such, in the HJM multifactor model, the evolution of a risk-neutral zero-coupon bond price satisfies the following equation:

$$dP(t,T) = P(t,T)\left\{r(t)dt + \sum_{i=1}^{N}\left(\int_t^T \sigma_i(t,s)ds\right)dW_i(t)\right\}, \tag{A8}$$

where dW_i, $i = 1,\dots,N$, are the uncorrelated Brownian increments associated to the volatilities σ_i. Using the bond price dynamics (A8) and (A3), we have

$$df(t,T) = \alpha(t,T)dt + \sum_{i=1}^{N}\sigma_i(t,T)dW_i(t), \tag{A9}$$

where

$$\alpha(t,T) = \sum_{i=1}^{N}\sigma_i(t,T)\int_t^T \sigma_i(t,s)ds. \tag{A10}$$

TABLE IV. After the fourth iteration, we rotate the measurement basis in x, y, and an arbitrary direction $r = (\cos\alpha, -e^{i\beta}\sin\alpha; e^{i\beta}\sin\alpha, e^{i\gamma}\cos\alpha)$, with $\alpha = 1.00$, $\beta = 0.80$, and $\gamma = 0.16$. Here we present the complete outcome for each rotation. The first two qubits correspond to the state $|\Lambda^{(n)}\rangle$, which encodes the 2-bit estimation of the largest eigenvalue λ_{max}. The last qubit corresponds to the estimation of the corresponding eigenvector $|u_{\text{max}}\rangle$. Each experiment has been run a total of 8192 times. The table shows both the number of times the experiment ends up on each state (under the heading "Counts") and the percentage it represents over the total number of runs ("%").

	x rotation				y rotation				r rotation			
	Simulator		Chip		Simulator		Chip		Simulator		Chip	
State	Counts	%	Counts	%	Counts	%	Counts	%	Counts	%	Counts	%
000	2215	27	2642	32	2229	27	2503	31	519	6	752	9
001	340	4	599	7	436	5	570	7	2089	26	2341	29
010	633	8	2183	27	631	8	2072	25	236	3	810	10
011	77	1	297	4	145	2	360	4	542	7	1413	17
100	588	7	557	7	549	7	567	7	75	1	160	2
101	4	0	125	2	3	0	133	2	508	6	422	5
110	4249	52	1555	19	4126	50	1736	21	475	6	449	5
111	86	1	234	3	73	1	251	3	3748	46	1845	23

This unique choice for $\alpha(t, T)$ as a function of the volatility terms is what prevents arbitrage. As mentioned before, this is a general framework from which many short-rate models may be derived, upon the particular choices for the σ terms. However, not every choice generates a Markovian dynamics. They must also be carefully selected in order to derive practical algorithms that are efficient in terms of computational times. One possibility that ensures Markovianity is to assume that the volatility factors only depend on the time to maturity, so $\sigma_i = \bar{\sigma}_i(T - t) = \bar{\sigma}_i(\tau)$. At this point, we can use time series data to calculate the functions $\bar{\sigma}_i$. For this purpose, we build the covariance matrix between the changes in the forward rates for different time maturities τ_j (typically for maturities at 1 month, 3 months, 6 monts, 1 year, 2 years, etc.). The result is a symmetric matrix whose diagonal terms are the variances of the rates, while the off-diagonal terms represent the covariances between each pair of rates.

Considering all possible time-maturing forwards is computationally costly for the numerical simulations. Using PCA we can obtain the most relevant eigenvectors and their associated eigenvalues. As seen in the literature, most of the evolution of the curve can be explained by considering two or three of such factors. Typically, it is observed that whenever the entries of the first principal component are all similar, then the dominant movement of the curve will be a parallel shift. Also, the second component typically accounts for a twist in the curve. In general, if the eigenvalues are λ_i and the eigenvectors are v_i, the volatility factors will be given by

$$\bar{\sigma}_i(\tau_j) = \sqrt{\lambda_i}(\mathbf{v}_i)_j. \tag{A11}$$

where λ_i and $\mathbf{v}_i$ are the eigenvalues and eigenvectors of the covariance matrix, respectively, and are estimated from 2×2 and 3×3 cross-correlation matrices between different time-maturing forward rates based on historical data.

APPENDIX B: ERROR δ

As we mention in Sec. III, the error δ is an upper bound of all possible errors. We cannot differentiate where the errors that we get are coming from since they are a mixture between the two-qubit gates and measurement errors and the inherent errors of the system, which does not increase with the depth of the circuit. Thus, we have to consider the worst possible case, where all of the possible errors are taking place.

We first run the experiment for the 2×2 covariance matrix and compute the fidelity of our circuit. After running this first experiment successfully and assuming that all of the errors that we get are a consequence of the two-qubit gate error, we estimate the error per two-qubit gate. Thus, the total error δ of the process might be described as

$$\delta = \sum \frac{\text{Fidelity}}{\text{number of two-qubit gates}}, \tag{B1}$$

and the error per gate, $\delta_{\text{two-qubit gate}}$, is defined as

$$\delta_{\text{two-qubit gate}} = \frac{\text{Fidelity}}{\text{number of two-qubit gates}}. \tag{B2}$$

To estimate the upper bound of the error that we expect to suffer when we run the experiment for the 4×4 covariance matrix, we simply multiply $\delta_{\text{two-qubit gate}}$ by the number of two-qubit gates of the corresponding quantum circuit.

APPENDIX C: COMPLETE RESULTS

1. 2 × 2 matrix

For the first part of the circuit we use two qubits for a 2-bit approximation of the eigenvalue λ_{max} and one to represent the corresponding eigenvector. We start this part by initializing the last qubit, which corresponds to the eigenvector estimation, on the state $|+\rangle = 1/\sqrt{2}(|0\rangle + |1\rangle)$ and project into the state $|y^{(n)}\rangle = |11\rangle$. In Table III we show the complete outcome we get on each iteration. After the fourth iteration the outcome vector estimating the eigenvector stabilizes and we stop.

In order to improve the accuracy of the solution provided and to compute the relative phase, we rotate the measurement basis in x, y, and an arbitrary direction $r = (\cos\alpha, -e^{i\beta}\sin\alpha; e^{i\beta}\sin\alpha, e^{i\gamma}\cos\alpha)$, with $\alpha = 1.00$, $\beta =$

TABLE V. Complete outcome of the quantum phase estimation algorithm. The first three qubits are dedicated to the binary codification of the eigenvalue λ. The last qubit corresponds to the eigenvector estimated on the previous part of the protocol.

	Simulator		Chip	
State	Counts	%	Counts	%
0000	1	0	509	6
0001	119	1	393	5
0010	78	1	1126	14
0011	209	3	475	6
0100	7	0	440	5
0101	13	0	194	2
0110	10	0	730	9
0111	7	0	523	6
1000	2	0	281	3
1001	22	0	216	3
1010	29	0	328	4
1011	14	0	458	6
1100	13	0	308	4
1101	192	2	260	3
1110	4949	60	1409	17
1111	2527	31	542	7

0.80, and $\gamma = 0.16$. In Table IV it is possible to see the outcome we get for each rotation.

After all these operations we get the following estimation of the eigenvector:

$$|u_{\max}\rangle = [(0.87 \pm \delta) - i(0.10 \pm \delta)]|0\rangle + ((0.47 \pm \delta) + i(0.10 \pm \delta))|1\rangle, \qquad (C1)$$

where $\delta = 0.9$ is the error estimated from the qubit and measurement fidelity provided by IBM and the statistical error related to the number of repetitions. We also get the 2-bit string estimation of the eigenvalue: $\Lambda_{\max} = 0.11$.

We now implement the second part of the protocol, which consists of applying the quantum phase estimation algorithm on a four-qubit system. The first three qubits are dedicated to get a more accurate estimation of the eigenvalue $\Lambda_{\max}$ and the last qubit encodes the previously estimated eigenvector. In Table V we show the outcome we get after performing this last part of the protocol. Due to the depth of the circuit implementation we end up losing all coherence when we run the experiment on the real quantum processor, which can be verified by looking at the results in Table V. However, looking at the results we get after running the same experiment on the QISKIT simulator, we see that the final state of the system is $|111\rangle \otimes [0.8150|0\rangle + 0.5794|1\rangle]$, which is an almost ideal result for the 3-bit string estimation of the eigenvalue. The predicted eigenvalue $\Lambda = 0.111$ is the binary representation of the number 0.875, which is a good approximation of the exact eigenvalue we are looking for.

2. 4 x 4 matrix

TABLE VI. Complete outcome of each iteration of the first part of the algorithm for the 4×4 case. The first qubit corresponds to the state $|\Lambda^{(1)}\rangle$, which encodes the 1-bit estimation of the largest eigenvalue $\lambda_{\max}$. The next two qubits hold the estimation of the corresponding eigenvector $|u_{\max}\rangle$. Each experiment has been run a total of 8192 times. The table shows both the number of times the experiment ends up on each state (under the heading "Counts") and the percentage it represents over the total number of runs ("%").

	Simulator		Chip	
State	Counts	%	Counts	%
		First iteration		
000	639	8	1131	14
001	1096	13	1098	13
010	195	2	1073	13
011	1570	19	991	12
100	2217	27	1144	14
101	684	8	985	12
110	1730	21	848	10
111	61	1	922	11
		Second iteration		
000	833	10	1414	17
001	1302	16	1105	13
010	59	1	1122	14
011	134	2	916	11
100	3360	41	1025	13
101	910	11	902	11
110	1533	19	883	11
111	61	1	825	10
		Third iteration		
000	829	10	1401	17
001	1720	21	1069	13
010	116	1	1209	15
011	71	1	890	11
100	3087	38	1069	13
101	906	11	879	11
110	1405	17	883	11
111	58	1	792	10
		Fourth iteration		
000	797	10	1325	6
001	1735	21	1036	13
010	144	2	1055	13
011	87	1	1249	15
100	3011	37	889	11
101	899	11	854	10
110	1464	18	964	12
111	55	1	820	10

In the 4×4 case we follow the same steps as in the 2×2 case. We start with the state $|b_0\rangle = (|00\rangle + |01\rangle + |10\rangle + |11\rangle)/2$, and then we apply four times the previously described iterative process. In Table VI we provide a complete description of the results obtained for each iteration. As in the previous case, we then measure in different bases in order to compute the relative phases (see Table VII). Due to the quantum circuit implementation being much deeper now than in the previous case, decoherence prevents us from recovering any remarkable result when running the experiment on the real quantum processor. The estimation of the eigenvector we get after running the protocol on the simulator is

$$|u_{\max}\rangle = (0.6287 + i0.3991)|00\rangle + (0.4010 + i0.0693)|01\rangle + (0.4807 - i0.1964)|10\rangle + (0.0305 + i0.0959)|11\rangle.$$

TABLE VII. List of the complete results obtained for the 4 × 4 case after rotating the basis in three arbitrary directions. The first rotation consists in applying a Hadamard gate to the first qubit, which corresponds to the change of basis $r = (\cos\theta, -e^{i\lambda}\sin\theta; e^{i\phi}\sin\theta, e^{i(\lambda+\phi}\cos\theta)$, with $\theta = \frac{\pi}{4}$, $\lambda = \frac{\pi}{2}$, and $\phi = 0$ in the planes $(|00\rangle, |10\rangle)$ and $(|01\rangle, |11\rangle)$. The second rotation consists in applying a Hadamard gate to the second qubit, which corresponds to the same change of basis but in the planes $(|00\rangle, |01\rangle)$ and $(|10\rangle, |11\rangle)$. The third rotation is the combination of the two previous rotations. The first qubit corresponds to the state $|\Lambda^{(1)}\rangle$ and encodes the 1-bit estimation of the largest eigenvalue $\lambda_{\max}$. The last two qubits correspond to the estimation of the eigenvector $|u_{\max}\rangle$. Each experiment has been run a total of 8192 times. The table shows both the number of times the experiment ends up on each state (under the heading "Counts") and the percentage it represents over the total number of runs ("%").

	Rotation I				Rotation II				Rotation III			
	Simulator		Chip		Simulator		Chip		Simulator		Chip	
State	Counts	%	Counts	%	Counts	%	Counts	%	Counts	%	Counts	%
000	150	2	1239	15	1229	15	987	12	1344	16	1008	12
001	4866	59	988	12	197	2	1128	14	686	8	998	12
010	384	5	1253	15	335	4	915	11	187	2	963	12
011	186	2	1077	13	770	9	1016	12	263	3	1117	14
100	1830	22	789	10	2262	28	1184	14	2382	29	980	12
101	66	1	777	9	821	10	1059	13	2633	32	787	10
110	79	1	770	9	450	5	944	12	392	5	1198	15
111	631	8	1299	16	2128	26	959	12	305	4	1141	14

APPENDIX D: ERROR ESTIMATION

The purpose of this section is to explain shortly how we obtained the error bars shown in Fig. 2. The argument we used to estimate the errors in state populations is the following. We suppose an experimental error of 8% in each two-qubit gate, which makes a total error of 24% for the experimental setup used in the estimation of the eigenvector in the 2 × 2 case. We also have to take into account the statistical error, which goes as $N^{-\frac{1}{2}}$. Since in our case the number of shots is 8192, this error is around 11%. Thus, we assumed a total error of 35%. This means that from 8192 results approximately 2900 are wrong. Dividing this quantity by the number of states (8) we get an error of 360 for the number of counts per state, which is the value we used to generate the error bars.

[1] D. Brigo and F. Mercurio, *Interest Rate Models—Theory and Practice* (Springer, New York, 2006).

[2] S. E. Shreve, *Stochastic Calculus for Finance II: Continuous-Time Models* (Springer, New York, 2004).

[3] S. Kogan, D. Levin, B. R. Routledge, J. S. Sagi, and N. A. Smith, Predicting risk from financial reports with regression, in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (Association for Computational Linguistics, Boulder, Colorado, 2009), pp. 272–280.

[4] M. F. Kanevski and V. Timonin, Machine learning analysis and modeling of interest rate curves, in *Proceedings of the 18th European Symposium on Artificial Neural Networks-Computational and Machine Learning (ESAAN 2010)*, Bruges, Belgium (2010).

[5] S. Lahmiri, Interest rate next-day variation prediction based on hybrid feedforward neural network, particle swarm optimization, and multiresolution techniques, Physica A **444**, 388 (2016).

[6] D. Heath, R. Jarrow, and A. Morton, Bond pricing and the term structure of interest rates: A discrete time approximation, J. Financ. Quant. Anal. **25**, 419 (1990).

[7] D. Heath, R. Jarrow, and A. Morton, Contingent claims valuation with a random evolution of interest rates, Rev. Futures Markets **9**, 54 (1991).

[8] D. Heath, R. Jarrow, and A. Morton, Bond pricing and the term structure of interest rates: A new methodology for contingent claims valuation, Econometrica **60**, 77 (1992).

[9] R. Jarrow, *Modelling Fixed Income Securities and Interest Rate Options* (Stanford Economics and Finance, Stanford, 2002)

[10] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000).

[11] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM J. Comput. **26**, 1484 (1997).

[12] L. K. Grover, From Schrödinger's equation to quantum search algorithm, Am. J. Phys. **69**, 769 (2001).

[13] A. W. Harrow, A. Hassidim, and S. Lloyd, Quantum Algorithm for Linear Systems of Equations, Phys. Rev. Lett. **103**, 150502 (2009).

[14] S. Lloyd, M. Mohseni, and P. Rebentrost, Quantum principal component analysis, Nat. Phys. **10**, 631 (2014).

[15] B. P. Lanyon, C. Hempel, D. Nigg, M. Müller, R. Gerritsma, F. Zähringer, P. Schindler, J. T. Barreiro, M. Rambach, G. Kirchmair, M. Hennrich, P. Zoller, R. Blatt, and C. F. Roos, Universal digital quantum simulation with trapped ions, Science **334**, 57 (2011).

[16] U. Las Heras, A. Mezzacapo, L. Lamata, S. Filipp, A. Wallraff, and E. Solano, Digital Quantum Simulation of Spin Systems in Superconducting Circuits, Phys. Rev. Lett. **112**, 200501 (2014).

[17] L. García-Álvarez, U. Las Heras, A. Mezzacapo, M. Sanz, E. Solano, and L. Lamata, Quantum chemistry and charge

transport in biomolecules with superconducting circuits, Sci. Rep. **6**, 27836 (2016).

[18] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature **549**, 242 (2017).

[19] J. Argüello-Luengo, A. González-Tudela, T. Shi, P. Zoller, and J. I. Cirac, Analog quantum chemistry simulation, Nature **574**, 215 (2019).

[20] R. Babbush, D. W. Berry, J. R. McClean, and H. Neven, Quantum simulation of chemistry with sublinear scaling to the continuum, npj Quantum Inf. **5**, 92 (2019).

[21] N. Klco, E. F. Dumitrescu, A. J. McCaskey, T. D. Morris, R. C. Pooser, M. Sanz, E. Solano, P. Lougovski, and M. J. Savage, Quantum-classical computation of Schwinger model dynamics using quantum computers, Phys. Rev. A **98**, 032331 (2018).

[22] J. Preskill, Simulating quantum field theory with a quantum computer, arXiv:1811.10085.

[23] A. Mezzacapo, M. Sanz, L. Lamata, I. L. Egusquiza, S. Succi, and E. Solano, Quantum simulator for transport phenomena in fluid flows, Sci. Rep. **5**, 13153 (2015).

[24] U. Alvarez-Rodriguez, M. Sanz, L. Lamata, and E. Solano, Biomimetic cloning of quantum observables, Sci. Rep. **4**, 4910 (2014).

[25] U. Alvarez-Rodriguez, M. Sanz, L. Lamata, and E. Solano, Quantum artificial life in quantum technologies, Sci. Rep. **6**, 20956 (2016).

[26] U. Alvarez-Rodriguez, M. Sanz, L. Lamata, and E. Solano, Quantum artificial life in an IBM quantum computer, Sci. Rep. **8**, 14793 (2018).

[27] B. E. Baaquie, *Quantum Finance Hamiltonians and Path Integrals for Options* (Cambridge University Press, Cambridge, 2004).

[28] R. Orús, S. Mugel, and E. Lizaso, Forecasting financial crashes with quantum computing, Phys. Rev. A **99**, 060301 (2019).

[29] P. Rebentrost, B. Gupt, and T. R. Bromley, Quantum computational finance: Monte Carlo pricing of financial derivatives, Phys. Rev. A **98**, 022321 (2018).

[30] R. Orús, S. Mugel, and E. Lizaso, Quantum computing for finance: Overview and prospects, Rev. Phys. **4**, 100028 (2019).

[31] J. Gonzalez-Conde, A. Rodríguez-Rozas, E. Solano, and M. Sanz, Pricing financial derivatives with exponential quantum speedup, arXiv:2101.04023.

[32] N. Stamatopoulos, D. J. Egger, Y. Sun, C. Zoufal, R. Iten, N. Shen, and S. Woerner, Option pricing using quantum computers, Quantum **4**, 291 (2020).

[33] D. Venturelli and A. Kondratyev, Reverse quantum annealing approach to portfolio optimization problems, Quantum Mach. Intell. **1**, 17 (2019).

[34] S. Woerner and D. J. Egger, Quantum risk analysis, npj Quantum Inf. **5**, 15 (2019).

[35] Y. Ding, J. Gonzalez-Conde, L. Lamata, J. D. Martín-Guerrero, E. Lizaso, S. Mugel, X. Chen, R. Orús, E. Solano, and M. Sanz,, Towards prediction of financial crashes with a D-Wave quantum computer, arXiv:1904.05808.

[36] P. Wilmott, *Paul Wilmott Introduces Quantitative Finance* (Wiley, New York, 2007).

[37] J. Preskill, Quantum computing in the NISQ era and beyond, Quantum **2**, 79 (2018).

[38] D. W. Berry, G. Ahokas, R. Cleve, and B. C. Sanders, Efficient quantum algorithms for simulating sparse Hamiltonians, Commun. Math. Phys. **270**, 359 (2007).

[39] A. M. Childs, On the relationship between continuous- and discrete-time quantum walk, Commun. Math. Phys. **294**, 581 (2010).

[40] G. Aleksandrowicz, T. Alexander, P. Barkoutsos, L. Bello, Y. Ben-Haim, D. Bucher, F. J. Cabrera-Hernádez, J. Carballo-Franquis, A. Chen, Ch.-F. Chen, J. M. Chow, A. D. Córcoles -Gonzales, A. J. Cross, A. Cross, J. Cruz-Benito, C. Culver, S. De La Puente González, E. De La Torre, D. Ding, E. Dumitrescu, I. Duran, P. Eendebak, M. Everitt, I. Faro Sertage, A. Frisch, A. Fuhrer, J. Gambetta, B. Godoy Gago, J. Gomez-Mosquera, D. Greenberg, I. Hamamura, V. Havlicek, J. Hellmers, L. Herok, H. Horii, S. Hu, T. Imamichi, T. Itoko, A. Javadi-Abhari, N. Kanazawa, A. Karazeev, K. Krsulich, P. Liu, Y. Luh, Y. Maeng, M. Marques, F. J. Martín-Fernández, D. T. McClure, D. McKay, S. Meesala, A. Mezzacapo, N. Moll, D. Moreda Rodríguez, G. Nannicini, P. Nation, P. Ollitrault, L. J. O'Riordan, H. Paik, J. Pérez, A. Phan, M. Pistoia, V. Prutyanov, M. Reuter, J. Rice, A. Rodríguez Davila, R. H. Putra Rudy, M. Ryu, N. Sathaye, C. Schnabel, E. Schoute, K. Setia, Y. Shi, A. Silva, Y. Siraichi, S. Sivarajah, J. A. Smolin, M. Soeken, H. Takahashi, I. Tavernelli, C. Taylor, P. Taylour, K. Trabing, M. Treinish, W. Turner, D. Vogt-Lee, C. Vuillot, J. A. Wildstrom, J. Wilson, E. Winston, C. Wood, S. Wood, S. Wörner, I. Y. Akhalwaya, and C. Zoufal, QISKIT: An open-source framework for quantum computing, 2019, https://github.com/Qiskit.

[41] A. Kandala, K. Temme, A. D. Corcoles, A. Mezzacapo, J. M. Chow, and J. M. Gambetta, Error mitigation extends the computational reach of a noisy quantum processor, Nature **567**, 491 (2019).

[42] K. Temme, S. Bravyi, and J. M. Gambetta, Error Mitigation for Short-Depth Quantum Circuits, Phys. Rev. Lett. **119**, 180509 (2017).

[43] A. Dewes, F. R. Ong, V. Schmitt, R. Lauro, N. Boulant, P. Bertet, D. Vion, and D. Esteve, Characterization of a Two-Transmon Processor with Individual Single-Shot Qubit Readout, Phys. Rev. Lett. **108**, 057002 (2012).

[44] https://github.com/amartinfer/QPCA.git.