Journal of
Quantum Computing

Tech Science Press

DOI: 10.32604/jqc.2022.036706
Article

Near Term Hybrid Quantum Computing Solution to the Matrix Riccati Equations

Augusto González Bonorino[1,*], Malick Ndiaye[2] and Casimer DeCusatis[2]

[1]Claremont Graduate University, Claremont, 91711, USA
[2]Marist College, Poughkeepsie, 12601, USA
*Corresponding Author: Augusto González Bonorino. Email: augusto.gonzalez-bonorino@cgu.edu
Received: 10 October 2022; Accepted: 07 April 2023; Published: 03 July 2023

Abstract: The well-known Riccati differential equations play a key role in many fields, including problems in protein folding, control and stabilization, stochastic control, and cybersecurity (risk analysis and malware propagation). Quantum computer algorithms have the potential to implement faster approximate solutions to the Riccati equations compared with strictly classical algorithms. While systems with many qubits are still under development, there is significant interest in developing algorithms for near-term quantum computers to determine their accuracy and limitations. In this paper, we propose a hybrid quantum-classical algorithm, the Matrix Riccati Solver (MRS). This approach uses a transformation of variables to turn a set of nonlinear differential equation into a set of approximate linear differential equations (i.e., second order non-constant coefficients) which can in turn be solved using a version of the Harrow-Hassidim-Lloyd (HHL) quantum algorithm for the case of Hermitian matrices. We implement this approach using the Qiskit language and compute near-term results using a 4 qubit IBM Q System quantum computer. Comparisons with classical results and areas for future research are discussed.

Keywords: Quantum computing; matrix ricatti equations; differential equations; qiskit; hybrid algorithm; HHL algorithm

1 Introduction

Quantum computing is an emerging field which utilizes the principles of quantum physics to perform computations in fundamentally different ways from classical computers. For certain problems, quantum algorithms offer significant improvements in execution time over their classical counterparts. One of the most famous examples is Shor's algorithm for factoring large (100–200 digit) numbers into the product of two primes, thus making it possible to break modern public-private key encryption schemes [1]. While the theoretical basis of this field has been understood for many decades, only within the past few years have small working quantum computers become available. One of the largest currently available quantum computers is the IBM Q System One, which

is programmed using the Qiskit language [2]. Much of the near-term research in this field involves proof of concept implementations, which are limited by the scalability of current quantum computer hardware. Nevertheless, it's critical to study these applications at a small scale now, in preparation for larger quantum computers becoming available within the next few years (current systems are limited to a handful of qubits; however, IBM roadmaps plan for a 1,000-qubit machine by 2024 or sooner [3]). For example, IBM has recently announced quantum resistant encryption features on the latest model z16 enterprise server [4], even though we are years away from building a large enough quantum computer to seriously threaten conventional encryption.

While there are currently only a few known examples where quantum computers offer a potential performance advantage, one promising area is the solution of systems of linear differential equations. For this problem, it has been shown [5] that under certain conditions, quantum algorithms can yield an exponential improvement in execution time compared with the best-known classical algorithms. In this paper, we investigate the use of quantum algorithms to generate approximate solutions to systems of nonlinear differential equations. Our approach involves a transformation which turns a set of nonlinear differential equations into an approximation using linear differential equations with second order non-constant coefficients. We can then solve a matrix representing this set of linear differential equations using a version of the Harrow-Hassidim-Lloyd (HHL) quantum algorithm, for the common case of Hermitian matrices.

Specifically, we are interested in the well-known Matrix Riccati nonlinear differential equations [6], which play a role in a wide range of fields including problems in protein folding, control and stabilization, stochastic control, and cybersecurity (risk analysis and malware propagation) [7–10]. These equations have been studied extensively using classical computing techniques, including significant efforts by national research labs [11], and approximate solutions with varying degrees of accuracy have been demonstrated. We propose a novel hybrid quantum-classical algorithm, the Matrix Riccati Solver (MRS), which is implemented on a near-term 4 qubit quantum computer using Qiskit. We then compare the accuracy achievable with the MRS and conventional approaches over a limited neighborhood of values. It is possible to achieve useful accuracy with this approach, and future quantum computers with additional qubits are expected to offer even better performance.

The remainder of this paper is organized as follows. Following a short introduction, we describe the fundamental Riccati equation formulations, the quantum linear systems problem, and the HHL algorithm. We then describe our implementation of the MRS hybrid algorithm, and present experimental results running on a 4-qubit quantum computer. We compare the MRS results with the best available classical approximations and suggest areas for further research.

2 Riccati Equations Fundamentals

In general, a Riccati equation is any first order ordinary differential equation that is quadratic in the unknown function, such as equations of the form

$$y' = a(t) \cdot y^2 + b(t) \cdot y + c(t) \tag{1}$$

where a(t), b(t) and c(t) are nonzero, continuous functions of t. More generally, the term Riccati equation is used to refer to matrix equations with an analogous quadratic term, which occur in both continuous-time and discrete-time linear quadratic Gaussian control systems. The steady-state (non-dynamic) version of these is referred to as the algebraic Riccati equation. The non-linear Riccati equation can always be converted to a second order linear ordinary differential equation (ODE).

To date, there is no general approach for finding solutions to the Riccati equations, although there are analytical techniques which allow solutions under certain conditions as well as numerical approximation solutions for other cases. For example, suppose we have a particular solution y_1 to this general equation. Then we can make the following change of variables

$$\mathrm{y} = \mathrm{y1} + \mathrm{u} \tag{2}$$

and so, we can rewrite the general equation as

$$(y1 + u)' = a(t) \cdot (y_1 + u)^2 + b(t) \cdot (y_1 + u) + c(t) \tag{3}$$

$$y_1' + u' = a(t) \cdot (y_1^2 + 2 \cdot y_1 \cdot u + u^2) + b(t) \cdot y_1 + b(t) \cdot u + c(t) \tag{4}$$

$$y_1' + u' = a(t) \cdot y_1^2 + a(t) \cdot 2 \cdot y_1 \cdot u + a(t) \cdot u + b(t) \cdot u + b(t) \cdot y_1 + c(t) \tag{5}$$

It follows that, since y_1 is a particular solution to the Riccati equation we can cancel out y_1', a (t) $\cdot$ y_1^2, b t $\cdot$ y_1, and $c(t)$. Hence, we obtain

$$u' = a\,(t) \cdot u^2 + u \cdot (2 \cdot a\,(t) \cdot y_1 + b\,(t)) \tag{6}$$

$$u' - u \cdot (2 \cdot a\,(t) \cdot y_1 + b\,(t)) = a\,(t) \cdot u^2 \tag{7}$$

$$\frac{u' - u\,(2 \cdot a\,(t) \cdot y_1 + b\,(t))}{u^2} = a\,(t) \tag{8}$$

Note that u' is a Bernoulli equation and thus we can apply well-established methods to find the general solution. First, we make the substitution $z = u^{1-m}$, where $m = 2$ in this case, to transform the Bernoulli into the following linear differential equation:

$$-\mathrm{z}' - (2 \cdot \mathrm{a}\,(\mathrm{t}) \cdot \mathrm{y}_1 + \mathrm{b}\,(\mathrm{t})) \cdot \mathrm{z} = \mathrm{a}\,(\mathrm{t}) \tag{9}$$

We will expand these methods to a matrix formalism, which is the standard approach used in quantum computing state machines.

3 Quantum Linear Systems Problem

The "Classical" Linear System Problem (LSP) states the following:

Given an $N \times N$ matrix A, an N–dimensional vector $\vec{b}$ and the equation

$$A \cdot \vec{x} = \vec{b} \tag{10}$$

Solve for $\vec{x} = (x1, x2, \ldots, xN)^T$. To date, the best algorithm for this task is called the Conjugate Transpose method, first introduced by Hestenes et al. in 1952 [12]. It has an asymptotic complexity of $O\left(N \cdot \sqrt{k}\right)$ and thus, complexity grows in polynomial time as the number of inputs increases.

Analogously, the Quantum Linear System Problem (QLSP) states that given a Hermitian Given an $N \times N$ matrix A, an N –dimensional vector $\vec{b}$, and the equation

$$A \cdot \vec{x} = \vec{b} \tag{11}$$

Prepare a quantum state that approximates the column vector

$$\vec{x} = \frac{\sum_{i=1}^{N} x_i \cdot \vec{i}}{\sqrt{\sum_{i=1}^{N} |x_i|^2}} \tag{12}$$

where $\vec{x} = (x_1, x_2, \ldots, x_N)^2$. Specifically, for a given precision $\epsilon > 0$, the approximate (mixed or pure) quantum state ρ_x satisfies

$$\frac{1}{2} \cdot Tr \left| \rho_x - \vec{x^T} \cdot vecx \right| \leq \epsilon \tag{13}$$

4 The HHL Algorithm

The Harrow-Hassidim-Lloyd (HHL) quantum algorithm [3] may be applied to expressions like those we developed in the previous section. Given a Hermitian matrix A, the HHL algorithm will compute the vector of unknowns $\vec{x}$ in the form

$$\vec{x} = \sum_{j=1}^{N} \beta_j \cdot \frac{1}{\lambda_j} \cdot \vec{u}_j \tag{14}$$

where $\{\vec{u}_j\}$ denotes the eigenvectors of A and $\{\lambda_j\}$ its eigenvalues. Then, HHL performs three important operations:

- Quantum Phase Estimation (QPE)
- Invert the eigenvalues
- Measurement

QPE is a central building block for many quantum algorithms. As shown in the quantum circuit of Fig. 1, QPE estimates the phase of an eigenvalue of the unitary operator U that we create by using the properties of Hermitian matrices. Given a unitary operator, U, QPE estimates the phase angle theta in

$$U \cdot \vec{\psi} = e^{2 \cdot \pi \cdot i \cdot \vec{\psi}} \tag{15}$$

where $\vec{\psi}$ is an eigenvector (i.e., state vector) and $e^{2 \cdot \pi \cdot i \cdot \vec{\psi}}$ is the eigenvalue. Note that since U is a unitary operator, all eigenvalues have a norm of 1.

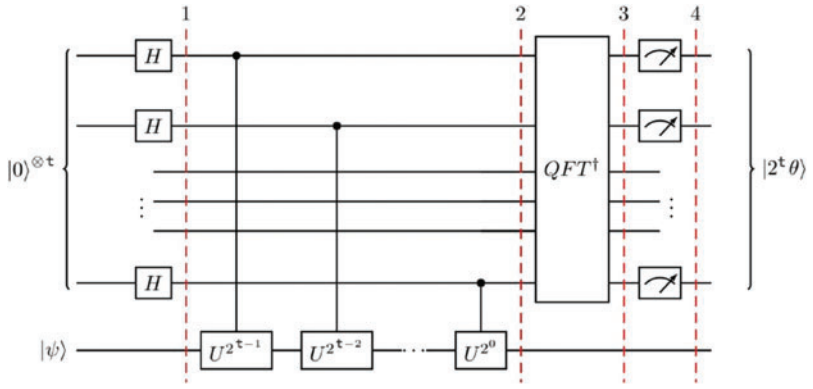

Figure 1: Quantum Phase Estimation (QPE) circuit using quantum fourier transform (QFT), hadamard gates (H), and various unitary transforms (U)

A quantum circuit for this algorithm is shown in Fig. 2, incorporating the QPE circuit from Fig. 1 and leveraging the lin_alg package from Qiskit so that we can treat the HHL algorithm as a black box.

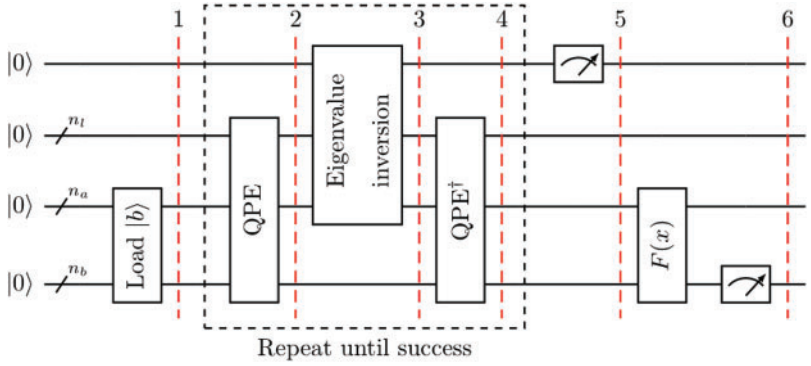

Figure 2: Quantum circuit for the HHL algorithm

After encoding the solution vector $\vec{b}$ as a linear combination of the input matrix A's eigenvectors, the algorithm undergoes an iterative process of computing the inverse of A's eigenvalues, by applying Quantum Phase Estimation with $U = e^{i \cdot A \cdot t}$, and its inverse until the measurement (depicted between demarcation lines 4 and 5 of Fig. 2) yields 1. Then, it returns the estimated solution

$$\vec{x} = \sum_{j=1}^{n} \frac{\vec{u_j^T} \cdot \vec{b}}{\lambda_j \cdot \vec{u_j}} \tag{16}$$

5 The Matrix Riccati Solver (MRS)

We propose a hybrid algorithm, the Matrix Riccati Solver (MRS), to find solutions for the matrix form of these equations given a set of initial conditions. First, we show that some Matrix Riccati equation can be re-written as a Hermitian matrix, given certain conditions are met, and encode this fact in the classical component of the proposed algorithm. Then, we feed the Hermitian matrix to HHL on IBM's 4-qubit near-term quantum computer to obtain the solution.

Let R denote the following Matrix Riccati equation

$$Y' = Y \cdot A(t) \cdot Y + C(t) \cdot Y + Y \cdot B(t) + D(t) \tag{17}$$

where $Y \in \mathbb{R}^{n \times m}$, $D(t) \in \mathbb{R}^{n \times m}$, $C(t) \in \mathbb{R}^{n \times n}$, $B(t) \in \mathbb{R}^{m \times m}$ and $A(t) \in \mathbb{R}^{m \times n}$. The goal is, given an initial condition $Y(0)$, to obtain solutions $Y(t)$, $\forall\, t > 0$.

We propose the following theorems (see Appendix for full proof):

Theorem 1: *Assuming $m = n$, if $B(t) = 0$ and if $A(t)$ is invertible, then Eq. (17) can be converted to the following second order matrix differential equation:*

$$u'' - (ACA^{-1} + A'A^{-1}) \cdot u' + AD \cdot u = 0 \tag{18}$$

By using the change of variables $Y = -A^{-1} \cdot u' \cdot u^{-1}$, where u is invertible.

Theorem 2: *If $ACA^{-1} + A'A^{-1} = S$ where S is a matrix with constant entries and diagonalizable, and $A \cdot D = -I$, then (18) can be converted into the following equations:*

$$v'' - D_1 \cdot v' - v = 0 \tag{19}$$

where D_1 is a diagonal matrix.

From (18) we derive the following linear system

$$e^{-tMii} \cdot x_{ii} = w_{ii\,(See\ Appendix)} \tag{20}$$

where $M_{11} = 0$, $M_{12} = 1$, $M_{21} = 1$, $M_{22} = -\alpha_i$, and α_i denotes the entries in the diagonal matrix D_1,

$wii = [v_{ii}\,(0),\ v'_{ii}\,(0)]^{\mathrm{T}}$, where v_{ij} are the entries of v so $v = (v_{ij})$ and $1 \le i, j \le n$.

Solving (20) is equivalent to solving the system

$$H \cdot \vec{x} = \vec{w} \tag{21}$$

where $\vec{x} = (x_{11}, x_{22}, \ldots, x_{nn})^T$, H has diagonal entries are $H_{ij} = e^{-t \cdot Mnn}$ and $\vec{\mathrm{w}} = (w_{11}, w_{22}, \ldots, w_{nn})^T$. Note that H is sparse and Hermitian for any i. Thus, we can leverage the HHL algorithm once we normalize the vector $\vec{w}$. In the particular case where $m = m = 1$, we get the following Riccati equation:

$$dy/dt = A\,(t) \cdot y^2 + B\,(t) \cdot y + C\,(t) \tag{22}$$

where $A(t)$, $B(t)$, $C(t)$ are real-valued functions of t. We seek to compute $y(t)$ given an initial condition $y(0)$. We convert the Riccati equation to a second order differential equation by applying the change of variables

$$y = -\frac{v'}{A \cdot v} \tag{23}$$

Such that we obtain the following equation:

$$v'' - v' \cdot \left(B + \frac{A'}{A}\right) + A \cdot C \cdot v = 0 \tag{24}$$

Next, let $\alpha = \mathrm{B} + \dfrac{A'}{A}$ and assume $A \cdot C = -1$, which must hold for the resulting matrix to be Hermitian.

From this, we obtain a system with vector of unknowns $\vec{u} = (v,\ z\ )^T$ and a matrix M $M_{11} = 0$, $M_{12} = 1$, $M_{21} = 1$, $M_{22} = -\alpha$. Therefore, we obtain the following equation $u' = M \cdot u$. We proceed by computing the characteristic polynomial to obtain the following eigenvalues

$$\lambda_i = \frac{\alpha \pm \sqrt{\alpha^2 + 4}}{2}, i \in \{1, 2\} \tag{25}$$

and thus, the corresponding eigenvectors are:

$$V_i = (1, \lambda_i)^T \tag{26}$$

Finally, we construct a 2×2 passage matrix $P = (V_1, V_2)$, such that we can rewrite our matrix M as $H = P \cdot D \cdot P^{-1}$ where D is a diagonal matrix with entries λ_i.

It follows that to solve this system we must compute $\vec{x} = e^{tH} \cdot \vec{x}_{\vec{0}}$ where $\vec{x}_{\vec{0}} = (u(0),\ z(0))^T$. Thus, we must rewrite the problem as $e^{-tH} \cdot \vec{x} = \vec{x}_{\vec{0}}$, where $e^{-tH} = P \cdot e^{-tD} \cdot P^{-1}$. Since we now know e^{-tH} is Hermitian, we conclude by normalizing $\vec{x}_{\vec{0}}$ such that $u(0)^2 + z(0)^2 = 1$. Fig. 3 is a high-level pseudocode description of the algorithm.

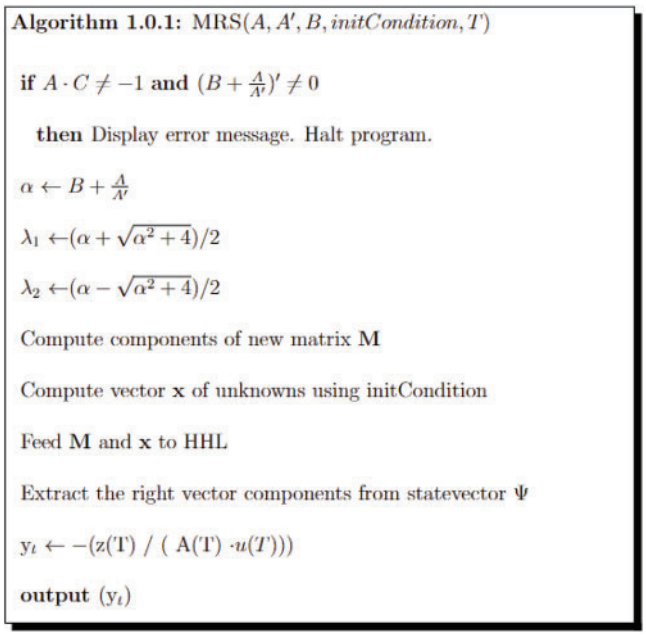

Figure 3: Matrix Riccati Solver (MRS) algorithm's pseudocode

The MRS checks that the required conditions are met. If not, the program displays an error and halts. To ensure that α is a constant MRS checks its derivative equals zero and then registers the variable as a constant quantity using the Python package SimPy constant quantity [13]. Then, MRS computes the eigenvalues λ_i of the system of equations we derived in the previous section. The eigenvalues are then utilized to compute the values of each component in our new matrix . We solve for $M \cdot \vec{x} = \vec{b}$ to compute the normalized vector of unknowns $\vec{x}$ for the specified initial condition $Y(0)$ by calculating the values of its components

$$x_0 \frac{1}{\sqrt{1 + A(0)^2 + Y(0)^2}} \tag{27}$$

$$x_{1=} \frac{A(0) \cdot Y(0)}{\sqrt{1 + (0)^2 + Y(0)^2}} \tag{28}$$

We can then input M and $\vec{x}$ to HHL to find the solution to the given system. This is achieved by extracting the state vector ψ, containing the quantum state of each qubit, from the circuit generated by the quantum computer. Finally, MRS extracts the states of the qubits containing our answers and computes a numeric approximation of the solution by solving for which denotes the time for which we are evaluating the expression. The initial condition refers to the value $y(T = 0)$.

$$y_t = -\left(\frac{z(T)}{(A(T) \cdot u(T))}\right) \tag{29}$$

6 Experimental Results

To test the algorithm's performance, we used the following example. Consider the following Riccati equation with initial condition $y(0) = 1$

$$\frac{dy}{dt} = (2 - sin\,(t)) \cdot y^2 + \frac{2 - sin\,(t) + cos\,(t)}{2 - sin\,(t)} \cdot y - \frac{1}{2 - sin\,(t)} \tag{30}$$

Recall the Uniqueness and Existence theorem [2] states that there is a unique solution for a given initial value problem within a local neighbourhood, defined by $\varepsilon > 0$. Fig. 4 shows the results of both the quantum solution (i.e., Matrix Riccati Solver) and classical solution over the local neighborhood (0.30, 0.38) with a time step of 0.001.

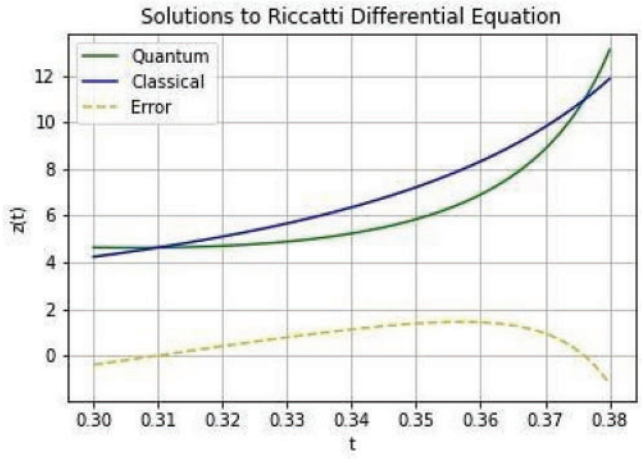

Figure 4: Comparison of classical solution and hybrid quantum-classical MRS solution

The accuracy of the quantum algorithm varies from a nearly exact solution near the edges of this neighborhood to a worst-case error of -1.444 at T $= 0.3570$. For some applications this is an acceptable error tolerance; in other cases, such as risk assessment analysis, the quantum algorithm will slightly under-estimate risk factors over this neighborhood. This error is likely attributed to the relatively small number of qubits used in the near-term approximate solution. Although there will always be some error due to quantum fluctuations, future quantum computers are expected to provide a better approximation of the classical solution over a larger neighborhood.

7 Conclusion

We propose a hybrid classical-quantum algorithm to generate solutions to nonlinear differential equations, specifically the Riccati equations, by approximating them as a series of linear differential equations. Previous research indicates that quantum assisted solutions should provide an exponential speedup in execution time. Although this may not be fully realized using near-term quantum computers, we investigate algorithms which can be scaled to larger numbers of qubits in future systems. Near term results indicate that over neighborhoods of interest the hybrid MRS algorithm offers good accuracy for a number of applications. Additional research is planned to investigate larger quantum computers as they become commercially available.

Acknowledgement: We would like to thank the Mathematics and Computer Science department at Marist College for supporting our academic endeavors with insightful conversations and advice.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] P. Shor, "Polynomial time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997. https://doi.org/10.1137/S0097539795293172

[2] A. Asfaw, L. Bello, Y. Ben-Haim, S. Bravyi, N. Bronn *et al.,* "Learn quantum computing using Qiskit," 2020. [Online]. Available: https://community.qiskit.org/textbook

[3] J. Hruska, "IBM unveils quantum roadmap, plans 1,000 qubit chips by 2023," English. 2020. Available: https://www.extremetech.com/computing/315020-ibm-unveils-quantum-roadmap-plans-1000-qubit-chip-by-2023

[4] IBM, "Announcing IBM z16: Real-time AI for Transaction Processing at Scale and Industry's First Quantum- Safe System," 2022. Available: https://newsroom.ibm.com/2022-04-05-Announcing-IBM-z16-Real-time-AI-for-Transaction-Processing-at-Scale-and-Industrys-First-Quantum-Safe-System?

[5] A. W. Harrow, A. Hassidim and S. Lloyd, "Quantum algorithm for linear systems of equations," *Physical Review Letters*, vol. 103, no. 15, pp. 150502, 2009. https://doi.org/10.1103/PhysRevLett.103.150502

[6] R. Haberman, "Applied Partial Differential Equations with Fourier Series and Boundary Value Problems," in *English*, 5th ed., London, UK: Pearson, 2018.

[7] A. Mohammadi and M. Spong, "Quadratic optimization based nonlinear control for protein conformation prediction," *IEEE Control Systems Letters*, vol. 6, pp. 2755–2760, 2022. https://doi.org/10.1109/LCSYS.2022.3176433

[8] A. Anees and I. Hussain, "A novel method to identify initial values of chaotic maps in cybersecurity," *Symmetry*, vol. 11, no. 2140, 2019. https://doi.org/10.3390/sym11020140

[9] W. Zhongru, L. Chen, S. Song, C. Pei Xin and R. Qiang, "Automatic cybersecurity risk assessment based on fuzzy fractional ordinary differential equations," *Alexandria Engineering Journal*, vol. 59, no. 4, pp. 2725–2731, 2020.

[10] L. Lystad, N. Per-Ole and H. Ralph, "The Riccati equation–an economic fundamental equation which describes marginal movement in time," *Modeling, Identification and Control*, vol. 27, no. 1, pp. 3–21, 2006. https://doi.org/10.4173/mic.2006.1.1

[11] Lawrence Livermore National Laboratory research brief, "NSDE: Nonlinear solvers and differential equations," 2005. [Online]. Available: https://computing.llnl.gov/projects/nsde (accessed on 08/20/2022).

[12] M. Hestenes and E. Stiefel, "Methods of conjugate gradients for solving linear systems," *Journal of Research of the National Bureau of Standards*, vol. 49, no. 6, pp. 409, 1952. https://doi.org/10.6028/jres.049.044

[13] A. Meurer, M. Paprocki, C. P. Smith, O. Čertik, S. B. Kirpichev *et al.,* "SymPy: Symbolic computing in Python," *PeerJ Computer Science*, vol. 3, no. 3, pp. e103, 2017.

Appendix

Let R denote the following Matrix Riccati equation

$$Y' = Y \cdot A(t) \cdot Y + C(t) \cdot Y + Y \cdot B(t) + D(t) \tag{A1}$$

where $Y \in \mathbb{R}^{n \times m}$, $D(t) \in \mathbb{R}^{n \times m}$, $C(t) \in \mathbb{R}^{n \times n}$, $B(t) \in \mathbb{R}^{m \times m}$ and $A(t) \in \mathbb{R}^{m \times n}$.

Theorem 1: *Assume $m = n$. If $B(t) = 0$ and if $A(t)$ is invertible, then (R) can be converted to the following second order differential equation:*

$$u'' - (A \cdot C \cdot A^{-1} + A' \cdot A^{-1}) \cdot u' + A \cdot D \cdot u = 0 \tag{A2}$$

By using the change of variables $Y = -A^{-1} \cdot u' \cdot u^{-1}$, where u is invertible.

Proof.

$$Y' = -(A^{-1})' \cdot u' \cdot u^{-1} - A^{-1} \cdot u'' \cdot u^{-1} - A^{-1} \cdot u' \cdot (u^{-1})' \tag{A3}$$

Recall $(A^{-1})' = -A^{-1} \cdot A' \cdot A^{-1}$. Then,

$$Y' = A^{-1} \cdot A' \cdot A^{-1} \cdot u' \cdot u^{-1} - A^{-1} \cdot u'' \cdot u^{-1} + A^{-1} \cdot u' \cdot u^{-1} \cdot u' \cdot u^{-1} \tag{A4}$$

$$Y' = A^{-1} \cdot u' \cdot u^{-1} \cdot A \cdot A^{-1} \cdot u' \cdot u^{-1} + A^{-1} \cdot u' \cdot u^{-1} \cdot B - C \cdot A^{-1} \cdot u' \cdot u^{-1} + D \tag{A5}$$

It follows that, since $B(t) = 0$

$$Y' = A^{-1} \cdot u' \cdot u^{-1} \cdot A \cdot A^{-1} \cdot u' \cdot u^{-1} - C - A^{-1} \cdot u' \cdot u^{-1} + D \tag{A6}$$

and so

$$A^{-1} \cdot A' \cdot A^{-1} \cdot u' \cdot u^{-1} - A^{-1} \cdot u'' \cdot u^{-1} = -C \cdot A^{-1} \cdot u' \cdot u^{-1} + D \tag{A7}$$

By right multiplying by u we get

$$A^{-1} \cdot A' \cdot A^{-1} \cdot u' - A^{-1} \cdot u'' = -C \cdot A^{-1} \cdot u' + D \cdot u \tag{A8}$$

Then, left multiply by A

$$A' \cdot A^{-1} \cdot u' = -A \cdot C \cdot A^{-1} \cdot u' + A \cdot D \cdot u \tag{A9}$$

Which yields

$$u'' - (A \cdot C \cdot A^{-1} + A' \cdot A^{-1}) \cdot u' + A \cdot D \cdot u = 0 \tag{A10}$$

Theorem 2: *If* $A \cdot C \cdot A^{-1} + A' \cdot A^{-1} = S$ **,** *where S is constant and diagonalizable, and* $A \cdot D = -I$.

Then, (A1) can be converted to the following equation

$$v'' - D_1 \cdot v' - Y = 0 \tag{A11}$$

where D_1 *denotes a diagonal matrix*.

Proof.

We know the S is diagonalizable, thus we can rewrite it as $S = P \cdot D_1 \cdot P^{-1}$ where P denotes the passage matrix $P^{-1} \cdot S = D1 \cdot P^{-1}$. Next, let F denote the following equation:

$$u'' - S \cdot u' - u = 0 \tag{A12}$$

By left multiplying both sides by P^{-1} we get

$$P^{-1} \cdot u'' - P^{-1} \cdot S \cdot u' - P^{-1} \cdot u = 0 \tag{A13}$$

or

$$P^{-1} \cdot u'' - D_1 \cdot P^{-1} \cdot u' - P^{-1} \cdot u = 0 \tag{A14}$$

Finally, let $v = P^{-1} \cdot u$ which yields

$$v'' - D_1 \cdot v' - v = 0 \tag{A15}$$

Obtaining the system from (A15)

Let α_i denote the eigenvalues of D_1 , where $1 \leq i \leq n$. Then, the diagonal entries of D_1 are $D_i = \alpha_i$.

Next, let $V_{ij} \in \mathbb{R}$, $1 \leq i \leq n$ and $1 \leq j \leq n$ be the entries of v. Then, (A15) leads to

$$v_{ij}'' - \alpha_i \cdot v_{ij}' - v_{ij} = 0 \tag{A16}$$

Now, notice that $v_{ii} = v_{ij}$ since α_i does not depend on j. Thus, we get

$$v_{ii}'' - \alpha_i \cdot v_{ii}' - v_{ii} = 0 \tag{A17}$$

Next, let $w_{ii} = v_{ii}'$ and $w_{ii}' = v_{ii}' = v_{ii} + \alpha_i \cdot w_{ii}$. Then, let $\vec{x}_{ii} = (v_{ii}, w_{ii})^T$ denote the vector unknowns. We can rewrite this as the following system of equations:

$$\frac{d\vec{x}_{ii}}{dt} = M_{ii} \cdot \vec{x}_{ii} \tag{A18}$$

To find the solution to (A18) we solve for $e^{-t \cdot M_{ii}} \cdot \vec{x}_{ii} = x_{0ii}$ where $M_{ii} = (c_1, c_2)$, $c_1 = (0, 1)^{\mathrm{T}}$, $c_2 = (0, \alpha_i)^{\mathrm{T}}$, and x_{0ii} denotes the initial value of $\vec{x}_{ii}$.

Computing e^{-tMi}

Let λ_{i} denote an eigenvalue of $\mathrm{M_{ii}}$, given by $\det(\mathrm{M_{ii}} - \lambda_{\mathrm{i}} \cdot \mathrm{I}) = 0$. It follows that $\lambda_{\mathrm{i}} = \dfrac{\alpha_{\mathrm{i}} \pm \sqrt{\alpha_{\mathrm{i}}^2 + 4}}{2}$. Hence, we have two eigenvalues $\lambda_{\mathrm{i}}^1 = \dfrac{\alpha_{\mathrm{i}} + \sqrt{\alpha_{\mathrm{i}}^2 + 4}}{2}$ and $\lambda_{\mathrm{i}}^2 = \dfrac{\alpha_{\mathrm{i}} - \sqrt{\alpha_{\mathrm{i}}^2 + 4}}{2}$, with corresponding eigenvectors $\mathrm{w_i^1} = \left(1, \lambda_{\mathrm{i}}^1\right)^{\mathrm{T}}$ and $\mathrm{w_i^2} = \left(1, \lambda_{\mathrm{i}}^2\right)^{\mathrm{T}}$. Then, the passage matrix P_i is given by $P_i = \left(\mathrm{w_i^1 w_i^2}\right)$. So, $\mathrm{M_{ii}} = \mathrm{P_i} \cdot \mathrm{N_i} \cdot {P_i}^{-1}$, where $\mathrm{N_i}$ is a diagonal matrix with entries $\lambda_{\mathrm{i}}^{\mathrm{k}}$, $\mathrm{k} \in \{1, 2\}$.

If we multiply both sides by -t and exponentiate we obtain the following equation

$$\boldsymbol{e^{-t \cdot M_{ii}} = P_i \cdot e^{-t \cdot N_i} \cdot P_i^{-1}} \tag{A19}$$

And therefore the following matrix:

$$\boldsymbol{e^{-t \cdot M_{ii}} = \begin{pmatrix} \dfrac{\lambda_i^2 \cdot e^{-t \cdot \lambda_i^1} - \lambda_i^1 \cdot e^{-t \cdot \lambda_i^2}}{\lambda_i^2 - \lambda_i^1} & \dfrac{-e^{-t \cdot \lambda_i^1} + e^{-t \cdot \lambda_i^2}}{\lambda_i^2 - \lambda_i^1} \\ \dfrac{-e^{-t \cdot \lambda_i^1} + e^{-t \cdot \lambda_i^2}}{\lambda_i^2 - \lambda_i^1} & \dfrac{\lambda_i^1 \cdot e^{-t \cdot \lambda_i^1} - \lambda_i^2 \cdot e^{-t \cdot \lambda_i^2}}{\lambda_i^2 - \lambda_i^1} \end{pmatrix}} \tag{A20}$$

Normalizing the solution vector

For the input system to be valid, HHL requires that the solution vector $\vec{b} = x_{0ii} = (v_{ii}(0), w_{ii}(0))^T$ is normalized such that $v_{ii}(0)^2 + w_{ii}(0)^2 = 1$.

Let $v = P^{-1} \cdot u$, $v_{ij} = \sum_{k=1}^{n} P_{ik} \cdot u_{kj}$, $w' = v_{ij}'$, $P^{-1} = P_{ij}$ and $u = u_{ij}$. Then,

$$v_{ij}(0) = \sum_{k=1}^{n} P_{ik} \cdot u_{kj}(0) \tag{A21}$$

$$w_{ij}(0) = \sum_{k=1}^{n} P_{ik} \cdot u_{kj}'(0) \tag{A22}$$

Since $Y = -A^{-1} \cdot u' \cdot u^{-1}$ we know that

$$A(0) \cdot Y(0) \cdot u(0) + u'(0) = 0 \tag{A23}$$

$$u'(0) = -A(0) \cdot Y(0) \cdot u(0) \tag{A24}$$

Expressed as sums instead of matrix notation

$$u'_{ki}(0) = -\sum_{l=1}^{n}\sum_{s=1}^{n}\alpha_{kl}(0)\cdot Y_{ls}(0)\cdot y_{si}(0) \tag{A25}$$

$$v_{ii}(0) = -\sum_{k=1}^{n} P_{ik}\cdot u_{ki}(0) \tag{A26}$$

$$w_{ii}(0) = \sum_{k=1}^{n} P_{ik}\cdot\left(-\sum_{l=1}^{n}\sum_{s=1}^{n}\alpha_{kl}(0)\cdot Y_{ls}(0)\cdot y_{si}(0)\right) \tag{A27}$$

We know apply the normalization condition $v_{ii}^2 + w_{ii}^2 = \frac{1}{n}$, $1 \le i \le n$, and obtain

$$\left(-\sum_{k=1}^{n} P_{ik}\cdot u_{ki}(0)\right)^2 + \left(\sum_{k=1}^{n} P_{ik}\cdot\left(-\sum_{l=1}^{n}\sum_{s=1}^{n}\alpha_{kl}(0)\cdot Y_{ls}(0)\cdot y_{si}(0)\right)\right)^2 = \frac{1}{n} \tag{A28}$$

where P_{ij} denotes the entries of the inverse of the passage matrix of $S = A\cdot C\cdot A^{-1} + A'\cdot A^{-1}$.

Putting everything together yields the solution $y(t) = -A^{-1}(T)\cdot u'(T)\cdot u^{-1}(T)$, which is the equation plotted in the results sections.