

Alma Mater Studiorum Università di Bologna
Archivio istituzionale della ricerca

Quantum advantage for differential equation analysis

This is the final peer-reviewed author's accepted manuscript (postprint) of the following publication:

Published Version:

Kiani, B.T., De Palma, G., Englund, D., Kaminsky, W., Marvian, M., Lloyd, S. (2022). Quantum advantage for differential equation analysis. PHYSICAL REVIEW A, 105(2), 1-15 [10.1103/PhysRevA.105.022415].

Availability:

This version is available at: <https://hdl.handle.net/11585/858463> since: 2022-03-30

Published:

DOI: <http://doi.org/10.1103/PhysRevA.105.022415>

Terms of use:

Some rights reserved. The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (<https://cris.unibo.it/>).
When citing, please refer to the published version.

(Article begins on next page)

This is the final peer-reviewed accepted manuscript of:

Kiani, B. T., De Palma, G., Englund, D., Kaminsky, W., Marvian, M., & Lloyd, S. (2022). Quantum advantage for differential equation analysis. Physical Review A, 105(2)

The final published version is available online at
<https://dx.doi.org/10.1103/PhysRevA.105.022415>

Rights / License:

The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (<https://cris.unibo.it/>)

When citing, please refer to the published version.

Quantum advantage for differential equation analysis

Bobak Toussi Kiani,^{1,2} Giacomo De Palma,^{2,3} Dirk Englund,^{1,2,4}
William Kaminsky,⁵ Milad Marvian,^{2,6} and Seth Lloyd^{7,2}

¹*Department of Electrical Engineering and Computer Science,
Massachusetts Institute of Technology, Cambridge, MA 02139*

²*Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139*

³*Scuola Normale Superiore, 56126 Pisa, Italy*

⁴*QuEra Computing, Boston, MA 02143*

⁵*Independent Contractor, Cambridge, MA 02139*

⁶*Department of Electrical and Computer Engineering,
Center for Quantum Information and Control (CQuIC),
University of New Mexico, Albuquerque, NM 87131*

⁷*Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139*

(Dated: April 27, 2022)

Quantum algorithms for differential equation solving, data processing, and machine learning potentially offer an exponential speedup over all known classical algorithms. However, there also exist obstacles to obtaining this potential speedup in useful problem instances. The essential obstacle for quantum differential equation solving is that outputting useful information may require difficult post-processing, and the essential obstacle for quantum data processing and machine learning is that inputting the data is a difficult task just by itself. In this study, we demonstrate, when combined, these difficulties solve one another. We show how the output of quantum differential equation solving can serve as the input for quantum data processing and machine learning, allowing dynamical analysis in terms of principal components, power spectra, and wavelet decompositions. To illustrate this, we consider continuous time Markov processes on epidemiological and social networks. These quantum algorithms provide an exponential advantage over existing classical Monte Carlo methods.

I. INTRODUCTION

One of the primary proposed applications of quantum computers is the solution of linear differential equations on high-dimensional spaces. The ability of quantum computers to represent N -dimensional vectors as the state of $\log_2 N$ qubits, and to perform linear algebraic transformations of those states in time $\text{poly}(\log N)$, then translates into a potential exponential speedup over classical algorithms for solving such high-dimensional differential equations. The output of the quantum computer presents the solution to the equation as a quantum ‘history state’ that is a superposition of the solution at different points in time. The problem then arises: How do we extract useful information from that quantum solution state? We can measure the expectation value of different quantities of interest: however, in the example of Markov chains, such expectation values can often be evaluated efficiently classically via Monte Carlo techniques. To obtain a quantum advantage that reveals essential features of the solution to the linear differential equations, we need to perform quantum post-processing on the history state.

Quantum machine learning and data processing algorithms provide potential exponential speedups over classical counterparts for methods such as high-

dimensional regression, principal component analysis, and support vector machines [1–4]. A basic problem with such quantum algorithms is that the input to the algorithm is a quantum state that encodes the classical data, and to construct such a state requires the implementation of a large-scale quantum random access memory (qRAM), a difficult technological task. The central observation of this study is that the problem with quantum linear equation solvers – they give quantum states as output – and the problem with quantum machine learning and data processing algorithms – they require quantum states as input – effectively solve each other: the output from the quantum linear equation solver can be used as the input to the quantum machine learning or data processing algorithm. In particular, we show that the history-state quantum solution to high-dimensional linear differential equations takes exactly the form needed to perform quantum analysis of the solution via quantum machine learning and data analysis. We show how to produce the singular values and singular vectors of the solution via quantum principal component analysis, and how to extract the power spectrum of the solution by performing quantum Fourier transforms. This analysis reveals the dominant components of the time evolution, corresponding to large singular values and

eigenvalues of the transition matrix with small negative real part. Finally we show how to perform quantum wavelet analysis to reveal rapid transitions and emergent features in the solution at different time scales. These quantum algorithms for post-processing the solution of linear differential equations can yield an exponential speedup over classical methods, and could potentially be performed on near term intermediate scale quantum computers.

The quantum post-processing of the history state to reveal salient features of the history can be applied to any linear differential equation. To show how quantum post-processing reveals such features, we focus on the case of continuous time Markov chains on high-dimensional spaces, with an emphasis on the spread of disease and opinion in complex social networks. Previous quantum algorithms for Monte Carlo methods yielded a square root speedup over classical algorithms [5]. By contrast, the quantum algorithms presented here for analyzing linear dynamics in terms of singular values, power spectra, and wavelets represent an exponential speedup over existing classical Monte Carlo methods.

II. RESULTS

A. Quantum algorithms for linear differential equations

Quantum numerical algorithms solve differential equations by quantizing classical numerical procedures and performing matrix operations on finite, high-dimensional state spaces [6–9]. We write a general linear differential equation for a vector in \mathbb{R}^N with $N \gg 1$ in the form

$$\frac{d\vec{x}(t)}{dt} = \mathcal{M}\vec{x}(t) + \vec{c} \quad (1)$$

where $x(t)$ represents the state of the system at time t , \mathcal{M} is the $N \times N$ differential equation matrix, and c is a forcing term. For example, in the case of Markov models, the state space is represented by a probability vector $x(t)$ with entries $x_i(t)$ indicating the probability of the system existing in state i at time t , and \mathcal{M} the transition matrix.

Well-known quantum algorithms [6, 7] can solve linear differential equations of the form of Eq. 1, where \mathcal{M} is a sparse matrix, by applying the quantum algorithm for linear systems of equations [10, 11]. The algorithms of Refs. [6, 7] are based on a simple underlying idea, but require highly nontrivial technical improvements in order to achieve a low

computational complexity. For the sake of a clearer exposition, we present here only the basic idea, and refer the reader to Refs. [6, 7] for a complete presentation of the algorithms. We consider the differential equation (Eq. 1) for $0 \leq t \leq t_{\max}$. The idea is based on the standard classical methods for discretizing Eq. 1 in T time steps each of size $h = t_{\max}/T$ and re-framing it as the solution to an equation of the form

$$A\vec{x} = \vec{b} \quad (2)$$

where

$$\vec{x} = \begin{pmatrix} \vec{x}_0 \\ \vec{x}_1 \\ \vec{x}_2 \\ \vdots \\ \vec{x}_T \end{pmatrix} = \begin{pmatrix} \vec{x}(0) \\ \vec{x}(h) \\ \vec{x}(2h) \\ \vdots \\ \vec{x}(Th) \end{pmatrix} \quad (3)$$

is a ‘vector of vectors’ that contains the values of the state vectors for the system, $\vec{x}_\ell = \vec{x}(t_\ell)$, at different moments of discretized time $t_\ell = \ell h$. A is a matrix that represents the updating action of the discretized differential operator. The form of A and \vec{b} depends on which discretization method one employs for the differential equation (*e.g.*, Euler forward, Euler backward, Crank-Nicolson, etc.). The simplest method is Euler forward, where

$$A = \begin{pmatrix} I & 0 & \dots & 0 & 0 \\ -(I + \mathcal{M}\Delta t) & I & \dots & 0 & 0 \\ & & \ddots & & \\ 0 & 0 & \dots & I & 0 \\ 0 & 0 & \dots & -(I + \mathcal{M}\Delta t) & I \end{pmatrix} \quad (4)$$

and

$$\vec{b} = \begin{pmatrix} \vec{x}_0 \\ \vec{c} \\ \vdots \\ \vec{c} \end{pmatrix} \quad (5)$$

The solution to the differential equation is then obtained by inverting the matrix in Eq. 2:

$$\vec{x} = A^{-1}\vec{b} \quad (6)$$

Roughly speaking, the quantum algorithms of [6, 7] map the classical states onto quantum states, $\vec{x} \rightarrow |x\rangle$, $\vec{b} \rightarrow |b\rangle$, and solve the problem using quantum matrix inversion to construct the normalized version of the unnormalized quantum state $|x\rangle = A^{-1}|b\rangle$,

which represents a quantum superposition of the solutions of Eq. 1 at different points in time.

We will base our results on the algorithm of Ref. [7], which has the lowest computational complexity. This algorithm sets $T = \lceil t_{\max} \|\mathcal{M}\| \rceil$, employs the Taylor expansion of the matrix exponential to solve the differential Eq. 1, and produces a coherent superposition of the quantum state $|x\rangle$ with some garbage state associated to the terms of the Taylor series. We show in Appendix A that the normalized version $|\bar{x}\rangle = |x\rangle / \| |x\rangle \|$ of $|x\rangle$ can be recovered from the quantum state produced by the algorithm of Ref. [7] with $O(1)$ success probability.

Form of the quantum solution The (unnormalized) quantum state $|x\rangle$ contains registers for the states and time-steps:

$$|x\rangle = \sum_{i=0}^T |x_i\rangle |i\rangle \quad (7)$$

where the entries of $|x_i\rangle$ are the values of the state for timestep $|i\rangle$. $|x\rangle$ is a quantum ‘history state’: a superposition of the different timesteps $|i\rangle$, correlated/entangled with the corresponding state vectors $|x_i\rangle$ at that timestep.

The quantum state in Eq. 7 can now be post-processed using efficient quantum algorithms such as those for wavelet transforms or quantum machine learning. Alternatively, history states encoding the evolution of an arbitrary quantum circuit at different times can be created by preparing the ground state of a local Hamiltonian [12, 13] and then subsequently post-processed.

Computational runtime and scaling of the error The computational complexity of the quantum algorithm of Ref. [7] is linear in the condition number and the time, and logarithmic in the error. We show in Appendix A that, by running this algorithm and by suitably projecting the generated quantum state, we can obtain a quantum state that is ϵ -close to the state $|x\rangle$ in 2-norm with

$$O\left(\kappa' \text{poly log} \left(\left(1 + \frac{t_{\max} \|\vec{c}\|}{\|\vec{x}(0)\|} \right) \frac{\kappa' N}{\epsilon} \right)\right) \quad (8)$$

elementary quantum gates. Here $\kappa' = \kappa t_{\max} \|\mathcal{M}\| s$, κ is the condition number of the matrix that is used to diagonalize \mathcal{M} and s is the sparsity of \mathcal{M} .

Furthermore, input states $|b\rangle$ for differential equation solvers can often be efficiently constructed via efficient algorithms that apply sparse matrix or local operations. This is especially true in the case of Markov chains, further discussed later, where initial

states are often supported on a sparse number of entries or locally uniformly throughout the possible set of states (see Appendix B).

Application to the Schrödinger equation In the case where \mathcal{M} is anti-Hermitian, then Eq. 1 is the Schrödinger equation [12]. In this case, the Fourier analysis of the history state reveals the eigenvalues and eigenvectors of \mathcal{M} . For example, when \mathcal{M} is the Feynman Hamiltonian, the history state encodes the history of quantum computation and the Fourier analysis reveals its eigenstates. Finding the eigenstates of the Feynman Hamiltonian is at least as hard as performing the quantum computation [14].

B. Continuous time Markov chains

To illustrate the power of our methodology, in this study, we focus on differential equations for continuous time Markov chains which update probability distributions over state space by assuming that the probability of making a transition to the next state only depends on the current state of the Markov chain. Here, we focus on Markov chains that represent dynamics of complex networks, particularly those modeling the spread of opinion and disease. The dynamics of complex networks are modeled via Markovian techniques by assuming that each node in a network exists in one of q states. A central challenge in this setup is in handling the dimension of the Markov state which grows exponentially with the number of nodes in a graph, often rendering the problem intractable for classical computers. For example, in epidemiology, modeling the dynamics of infection and recovery for a system of individuals whose interactions make up a complex network of n nodes is a hard computational problem that involves predicting the behavior of a very large q^n dimensional continuous time probabilistic dynamics [15, 16]. The exact solution for the dynamics of such epidemiological models lies beyond the realm of capability of even the most powerful classical computers. Consequently, classical approaches typically rely on various approximations, such as mean field theory [15].

Continuous time Markov chains are defined by differential equations of the form

$$\frac{d\vec{x}(t)}{dt} = \mathcal{M}\vec{x}(t) \quad (9)$$

where $\vec{x}(t)$ is the vector of probabilities for the underlying state of the system at time t , and \mathcal{M} is a matrix of transition rates [17, 18].

Eq. 9 is precisely of the form required in Eq. 1 for efficiently solving differential equations using quantum algorithms. As shown in Appendix A, the quantum algorithm of Ref. [7] can be employed to produce a normalized version of the quantum state $|x\rangle = \sum_{i=0}^T |x_i\rangle |i\rangle$ storing the state probabilities at discretized times. Note that the quantum algorithm represents the vector of probabilities as quantum vector of probability amplitudes.

Generalization to non-Markovian models

We can generalize our algorithms to incorporate prior histories in a “non-Markovian” model. The Markovian nature to the methods for solving differential equations is reflected in the form of the matrix A in Eq. 4 above. The fact that the Euler forward method for discretizing a differential equation depends only on the current and previous state of the system implies that A only has entries on the diagonal and directly below the diagonal. If we wish to include influences on the present from further in the past (including the distant past), then we can simply add additional entries to each row: adding a matrix entry A_{ij} , $j < i$ to the i ’th row of A allows the state of the system at time $j < i$ to influence the updating at time i . This change cannot be directly incorporated in the algorithm of Ref. [7], which relies on the Taylor expansion of the matrix exponential, but it can easily be incorporated in the previous algorithm of Ref. [6], which directly solves Eq. 2.

C. Quantum post-processing

The solution of our quantum differential equation solver $|x\rangle$ exists in a very high dimensional Hilbert space, and here, we discuss methods to obtain useful information from $|x\rangle$ using various quantum algorithms that can offer exponential speedups over classical counterparts. In this study, we focus on the particular case of post-processing outputs from continuous time Markov chain models. The algorithms we list here are by no means comprehensive of the full catalog of algorithms available to quantum computer scientists for extracting information from these states.

Post-processing: expectation values of quantities The most basic information that can be extracted from a Markov chain is the expectation value at a given time of a real-valued observable on the state space. Classically, such expectation values can be estimated efficiently by Monte Carlo sampling of the Markov chain up to the required time. In the quantum case, expectation values of

quantities can be obtained by estimating the overlap of the history state with a state encoding the values of the quantity we wish to calculate. In Appendix C, we consider two quantum algorithms to compute such expectation values. The first is based on a post-processing of the quantum history state of the Markov chain, and the second is based on the coherent version of the classical Monte Carlo simulation of the Markov chain. As shown in Appendix C, one can obtain a quadratic speedup in the error of estimating an observable using quantum techniques for Monte Carlo sampling.

Post-processing: principal component analysis of data matrix If the history state is effectively low rank, *i.e.*, there are only a few large singular values, then the description of the time evolution of the Markov process can be compressed by expressing it in terms of the corresponding singular vectors, which the quantum principal component analysis also reveals. The history state will be effectively low rank, for example, when the Markov transition matrix has only a few eigenvalues with small negative parts, so that the dynamics is dominated over longer times by the corresponding eigenvectors.

In the case of continuous time Markov chains, the quantum state in Eq. 7 can be interpreted as a $q^n \times T$ data matrix \mathbf{X}_{mat} where each column j corresponds to the probabilities of the Markov state at timestep j . The dominant singular values and corresponding singular vectors of this matrix can be extracted from $|x\rangle$ by performing quantum principal component analysis (qPCA) on the $|x_j\rangle$ and $|j\rangle$ registers which runs in $O(Rn \log q)$ time where R is the rank of \mathbf{X}_{mat} [2]. Note, that qPCA in this setting is equivalent to performing a Schmidt decomposition on the Hilbert spaces spanned by registers $|x_j\rangle$ and $|j\rangle$. The qPCA performs this decomposition via density matrix exponentiation [2]. It is often the case that the effective rank R (the number of large singular values) of \mathbf{X}_{mat} is small with respect to the number of states q^n , and later, we show that the effective rank is in fact very small for the example models we consider. Note that because our method acts directly on a quantum state, quantum inspired algorithms for PCA do not have access to the data structure for extracting the singular vectors and singular values of the history state [19–21]. Specifically, in Appendix D, we show that classical Monte Carlo methods cannot efficiently extract the singular vectors and the singular values of the history state (Eq. 7) whenever the support of the probability distribution is exponentially large in the number of nodes of the network.

After performing qPCA via density matrix expo-

mentation on copies of $|x\rangle$, we have a decomposition of the data matrix into left and right singular vectors:

$$\text{qPCA} : |x\rangle \rightarrow \sum_{j=0}^T |l_j\rangle |r_j\rangle |\tilde{\sigma}_j\rangle, \quad (10)$$

where $|l_j\rangle$ are the left singular vectors corresponding to the Markov states, $|r_j\rangle$ are the right singular vectors corresponding to the temporal states, and $|\tilde{\sigma}_j\rangle$ are estimates of the singular values. The singular values $|\tilde{\sigma}_j\rangle$ represent the weight of the left (Markov state) and right (temporal state) singular vectors in the solution. It is conventional to take the ordering in j in Eq. 10 to be from the largest to smallest singular values.

The left singular vectors $|l_j\rangle$ can be interpreted as the most common profile of Markov states. The first left singular vector corresponds to the profile with the greatest contribution to the data matrix, often the steady state of the Markov process. The next few singular vectors typically correspond to the profile of states in the early progression of the Markov simulation before steady state is achieved (see simulations later for examples).

The right singular vectors $|r_j\rangle$ detail the progression of the corresponding left singular vectors over time. For example, the first singular vector is typically weak during the early progression and grows to a constant value as the steady state arises. The next few right singular vectors show when the corresponding left singular vectors take prominence, often highest in magnitude at early points in time (see simulations later for examples).

Decomposing the data into singular vectors also allows one to apply efficient transformations to the singular vectors using quantum post-processing methods. For example, if one is interested in analyzing the Markov states in the frequency domain, a quantum Fourier transform can be applied to the right singular vectors. Later, in our example, we show that the dominant singular vectors correspond to slowly varying dynamics at low frequencies and the later singular vectors correspond to more rapidly varying dynamics at higher frequencies.

Post-processing: efficient quantum transformations Fourier transforms and wavelet transforms are commonly used in the analysis of large datasets, especially time series [22–25]. Discrete wavelet transforms, for example, can be used to identify statistical patterns in a time series. With a quantum computer, Fourier transforms and certain discrete wavelet transforms can be performed exponentially faster than their classical counterparts

[26–28]. These transforms can be applied to the data contained in our output quantum state (Eq. 7). For example, a Fourier transform or wavelet transform can be applied to the time register, *e.g.* to observe the data in the frequency domain or to compress the data in terms of the dominant wavelets. Let $U_{jk} = \langle k|j\rangle$ be the element of the unitary matrix U that maps the states $|j\rangle$ to the transform states $|k\rangle$ (frequency states in the case of the quantum Fourier transform; wavelets in the case of the discrete wavelet transform). Applying U to the temporal register, we obtain the state

$$\sum_{j=0}^T |x_j\rangle U |j\rangle = \sum_{j,k=0}^T |x_j\rangle U_{jk} |k\rangle = \sum_{k=0}^T |y_k\rangle |k\rangle, \quad (11)$$

where $|y_k\rangle = \sum_{j=0}^T U_{jk} |x_j\rangle$, is the state of the system correlated with the k th frequency or wavelet state in the temporal register. Sampling from the temporal register then yields the dominant frequencies/wavelets, and the spatial register yields the state of the system correlated with those frequencies/wavelets. For example, as noted above in the discussion of the Schrödinger equation, in Eq. 1, if \vec{c} is 0 and the matrix \mathcal{M} is anti-Hermitian, performing a quantum Fourier transform on the temporal register yields the purely imaginary eigenvalues of \mathcal{M} and the output contains the corresponding eigenvectors [29]. More generally, when the eigenvalues of \mathcal{M} have both real and imaginary components, performing the quantum Fourier transform reveals the power spectrum of the solution: a complex eigenvalue $a + ib$ manifests itself as a Lorentzian peaked at the natural frequency $\sqrt{a^2 + b^2}$.

By contrast, classical Monte Carlo sampling does not obviously extract the proper information for performing Fourier or wavelet transforms on the quantum state (see Appendix D).

Post-processing: quantum machine learning In the past few years, many quantum algorithms for machine learning have been proposed that can be performed exponentially faster than classical counterparts when data inputs are quantum states [1, 3, 4, 30–34]. When data in the form of Eq. 7 is used as input, applications of machine learning algorithms are numerous. Here we list some of these applications, grouped by the type of model used. First, quantum models have been proposed for compression of data or efficient readout. These models include quantum auto-encoders [34, 35] and qPCA as discussed before [2]. Second, a wide range of algorithms implementing kernel methods can be used to classify data, identify key features in the data, or

measure similarity between different datasets [4, 36–38]. Indeed, if we trace out the temporal register in Eq. 7, the state register is described by the (unnormalized) density matrix $\sum_j |x_j\rangle\langle x_j|$, which is the covariance matrix for the synthetically generated data which can be analyzed directly. Third, output states can be input into parameterized quantum circuits or quantum neural networks to identify key features or perform machine learning tasks [32, 39, 40].

D. Example simulation for epidemic processes

The analysis of epidemic spreading – viral or social – is often modeled as a dynamical process on a complex network [15]. Theoretical approaches to epidemic processes typically assume transitions (*e.g.* rates of infection) occur as Poisson processes which correspond to models of continuous time Markov chains [15, 41]. Classically, numerical simulation of continuous time Markov processes is intractable for large networks as the dimension of the Markov state grows exponentially with the number of nodes in the network or graph.

To demonstrate the applicability of our quantum algorithm, we simulate continuous time Markov chain models on simple seven node networks. We choose a relatively small network so that we can still visually represent the full solution to the continuous time Markov chain. Here, we present models for analyzing viral epidemics and perform similar analysis for social epidemics in Appendix E. For ease of graphical presentation, we implement susceptible-infected-susceptible (SIS) epidemiological models which have only two states per node: susceptible and infected. This is in contrast to the more realistic susceptible-infected-recovered (SIR) models which also fits within the framework of our algorithms, but can be hard to visualize and plot since they have many more states (3^n as opposed to 2^n).

Epidemic simulations of viral contagion

We present analysis of a Markov simulation for a susceptible-infected-susceptible (SIS) model on a single network shown in Fig. 1a. Our simple model features many common properties of continuous time Markov chain simulations. Notably, it is common that Markov transition matrices have only a small number of dominant eigenvalues (*i.e.*, those whose values are close to zero) thus rendering them suitable for analysis similar to that performed here for small networks. Of course, network models with more nodes will likely have emergent phenomena that will not appear in this small network – phenomena that one may hope to analyze using quan-

tum computers [15, 41].

For an SIS model, the full state at time t is described by a vector $\vec{x}(t)$ of length 2^n ($n = 7$ for our example). All transitions are modeled as Poisson processes with transition matrix \mathbf{Q} .

$$\frac{d\vec{x}(t)}{dt} = \mathbf{Q}\vec{x}(t) \quad (12)$$

We begin in an initial state where any node has a 35% probability of being infected and conduct analysis over the intermediate phase of the epidemic, between days one and two. To numerically estimate the state at discretized times \vec{x}_t , we employ a Forward Euler method that begins at the state of the epidemic on day one and ends at day two over $T = 1027$ timesteps (step size $h \approx 0.001$ days):

$$\vec{x}_{t+1} = \vec{x}_t + h\mathbf{Q}\vec{x}_t \quad (13)$$

From day one to day two, the epidemic has spread sufficiently that multiple individuals are likely to be infected, and the probability distribution has spread throughout the space. As noted above, this is the regime where the quantum algorithm provides an exponential advantage over existing classical Monte Carlo techniques.

Fully simulating the above for T steps constructs a data matrix \mathbf{X}_{mat} where column i is equal to x_{i-1} (we assume initial state x_0 is included in this matrix as well).

$$\mathbf{X}_{mat} = \begin{bmatrix} | & | & \dots & | \\ \vec{x}_0 & \vec{x}_1 & & \vec{x}_T \\ | & | & & | \end{bmatrix} \quad (14)$$

where we note that this data matrix can be interpreted as a classical version of the quantum output state shown in Eq. 7.

In Fig. 1b, we plot the probabilities of states of the Markov process over time providing a visualization of the data matrix \mathbf{X}_{mat} . Subsequent post-processing of this data matrix is performed on the segment of data between days one and two. We note that though Fig. 1b contains a complete description of the Markov history state, extracting trends and analyzing this figure can be challenging. For larger networks, visualization of this data matrix cannot be efficiently performed and we now turn our attention to efficient quantum algorithms for extracting salient features from this data matrix.

Simulations: quantum principal component analysis (qPCA) The output of our continuous time Markov chain algorithm is stored in a data matrix, visualized in Fig. 1b. In the quantum setting, this data matrix is a quantum state which we

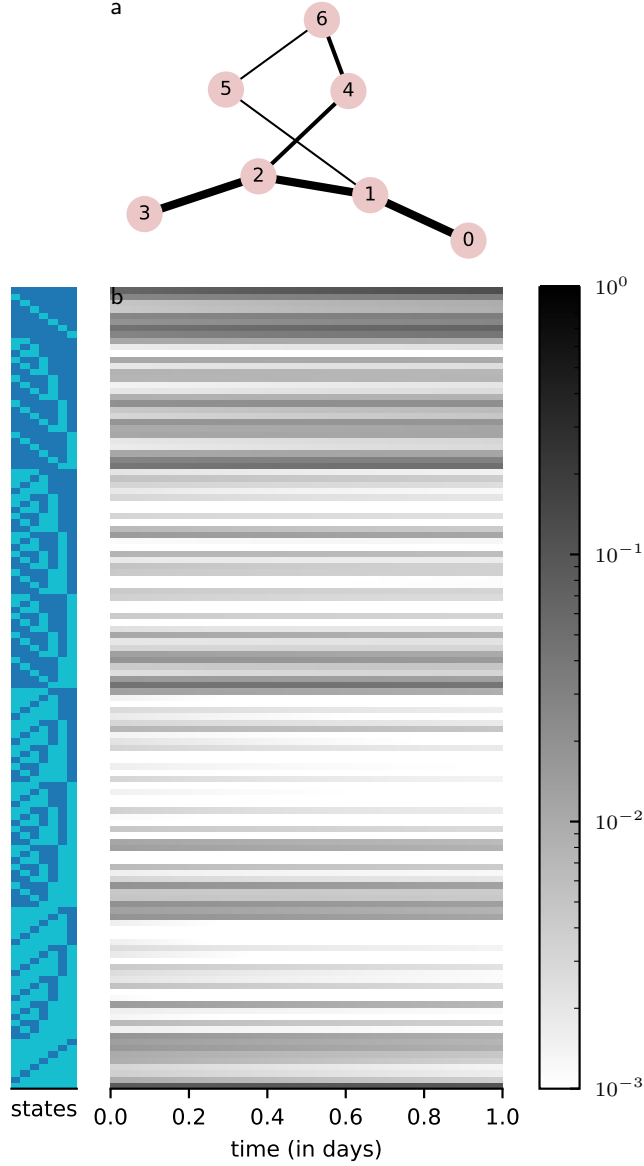


FIG. 1: **(a)** The 7 node network used for simulation of the continuous time Markov chain. Line widths correspond to the rate of infection from infected neighbor nodes ($r_{SI} = \{0.4, 0.8, 1.6\}$ from thinnest to thickest lines). Network has the feature that some nodes (*e.g.*, node 2 and 3) are strongly connected to neighboring nodes and others (*e.g.*, node 5) are not. **(b)** Probabilities of Markov states over the course of three days shown as a colorbar chart (logarithmically scaled). Post-processing is performed on the data between days one and two contained in the orange box. States are enumerated as rows on the left hand side, each denominated by a 7 node colorbar numbered node 0 on the left to node 6 on the right.

Dark/light color indicates a node in that state is infected/susceptible respectively.

can subsequently post-process using various efficient quantum algorithms. One available post-processing algorithm is quantum Principal Component Analy-

sis (qPCA) as in Eq. 10, which can compress the data into its singular vectors [2]. If the matrix is low rank, as in our example with exponentially de-

caying singular values (see Fig. 2a), qPCA can be performed in time logarithmic in the dimension of the full Markov state [2]. The principal components can be subsequently transformed or even measured.

The most dominant left (Fig. 2b) and right (Fig. 2c) singular vectors can be interpreted as the most common profile of states (left vectors) and their corresponding trajectories in time (right vectors). In our example, the first singular vector corresponds to the steady state of our epidemic. Note that it takes prominence almost completely throughout the course of the simulation (see right singular vector in Fig. 2c). The second singular vector plots important changes in the epidemic as more nodes become infected. The corresponding right singular vector plots the steady, almost linear, transition over time as this singular vector takes prominence. Similarly, the third and fourth singular vectors plot trends in the progression of the epidemic, especially in early phases where nodes become infected.

Simulations: Fourier and wavelet analysis For small networks such as the one studied here, the data in the Fourier domain is dominated by the steady state contributions as shown in Fig. 3a. With quantum algorithms, one also has the option of transforming the individual singular vectors into their frequency components as shown in Fig. 3b. The second to fourth singular vectors all have strong contributions from low frequencies, whose values provide an indication of the rate of change in the progression of the Markov chain. Given the small network size, this dominance of low frequency components is perhaps not altogether surprising. Larger networks encounter phenomena not observed in small networks, and may potentially reveal interesting features in the Fourier domain [15, 41] if they are analyzed with a quantum computer.

Beyond the standard Fourier transform, quantum computers offer the advantage of efficient post-processing via wavelet transforms [26]. Continuing the example shown in the main text, here we transform the time dimension of our data matrix using a Haar wavelet transform, which can be performed efficiently on a quantum computer [26]. When viewed in the Haar wavelet basis (form of wavelets shown in Fig. 4a), one can analyze the characteristic timescales over which differences in the Markov state probabilities become apparent. Perhaps unsurprisingly, as shown in Fig. 4b, the zeroth Haar vector is most prominent as this corresponds to the steady state of the Markov chain. More interesting results are observed in analysis of the singular vectors, which can also be transformed into the Haar domain as shown in Fig. 4c. Here, clear differences

can be observed in the Haar basis of the steady state singular vector (first singular vector) and later vectors. The first singular vector is dominated by the zeroth Haar wavelet (constant wavelet) since that singular vector corresponds to the steady state. The next few singular vectors corresponding to changes in the intermediate progression of the epidemic are dominated by Haar vectors with support over various phases. For example, the third and fourth singular vectors take large values over Haar vectors with support in the early phases of the simulation (*e.g.*, fourth and eighth Haar vectors).

E. Potential for realization on near term quantum devices

The algorithms proposed here are potentially suitable for near term quantum devices [42] with around 100 qubits. We note that the presence of noise in near term quantum devices, which is not analyzed here, may present a challenge to the successful implementation of these algorithms. Nevertheless, if challenges with noise are addressed via error correction or other means, our algorithms can be used to analyze the output of high dimensional differential equations outside the reach of available classical algorithms. For example, Markov states of dimensionality 2^{50} can be simulated for a million timesteps (approximately 2^{20}) on a quantum computer with 100 qubits using algorithms from [7]. Given the output state, quantum principal component analysis and wavelet transforms can subsequently be performed to analyze the history state generated by the near term quantum device [2, 26, 43].

III. DISCUSSION

Common quantum algorithms for solving linear differential equations output quantum states corresponding to solutions of physical models in high dimensional vector spaces. These output states store the complete history of the solution to a differential equation, allowing one to perform efficient quantum post-processing on these solution states. Our approach avoids a commonly cited drawback of many quantum machine learning and data processing algorithms – that classical inputs must be mapped into quantum states or stored in qRAM. We focus here on the case of Markov models and propose efficient quantum algorithms for evaluating continuous time Markovian and non-Markovian models. Our algorithms allow for efficient simulation of Markov mod-

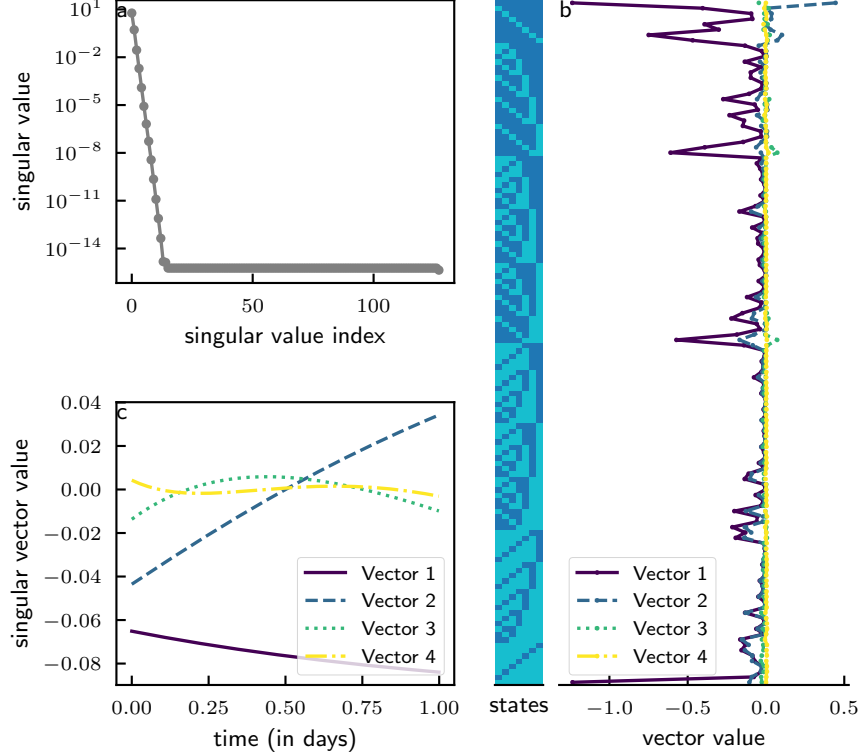


FIG. 2: **(a)** Singular values of data matrix X_{mat} decay exponentially fast. **(b)** First four left singular vectors scaled by the square root of their corresponding singular values show that much of the disease progression can be understood by just observing the first few vectors. States are enumerated as rows on the left hand side, each denominated by a 7 node colorbar numbered node 0 on the left to node 6 on the right. Dark/light colors indicate a node in that state is infected/susceptible respectively. **(c)** Values of the right singular vectors scaled by the square root of their corresponding singular values show the progression of the epidemic over time. The first singular vector depicts the steady state and the next few singular vectors detail the intermediate course of the Markov process.

els on the complete state of a Markov chain. Outputs of our models are quantum states which can be efficiently generated and then passed as inputs for other efficient quantum post-processing algorithms (*e.g.*, quantum signal analysis and machine learning). The quantum post-processing reveals features of the data such as the singular value decomposition, the power spectrum, and wavelet decompositions, which cannot be reconstructed efficiently using classical sampling algorithms. When applied to complex networks, the quantum algorithms may be used to reveal such fundamental features of the dynamics of epidemics potentially exponentially faster than classical algorithms.

Data Availability All data generated or analysed during this study are included in this article (and supplementary information files).

Code Availability To recreate figures and results, please access our code repository at: <https://github.com/bkiani/Quantum-DiffEQ-Advantage>

ACKNOWLEDGEMENTS

This work was supported by NSF, ARO, DOE, AFOSR, and IARPA. MM is supported by the NSF Grant No. CCF-1954960. The authors thank Peter Love and Clemens Wittenbecher for helpful discussions.

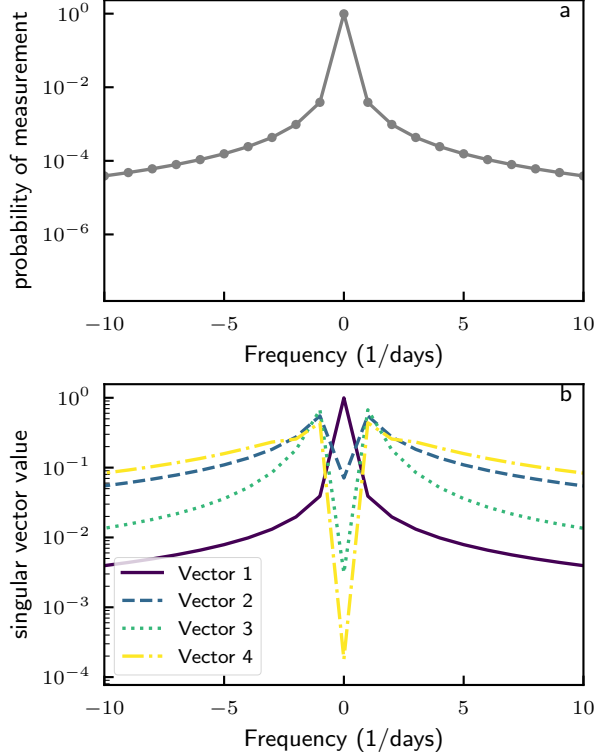


FIG. 3: **(a)** Measurements made in the Fourier domain (plotted only for low frequency components) have high probability in the zero frequency (steady state) and low frequency components. **(b)** The first singular vector, plotted in the Fourier domain, is dominated by the zero frequency component corresponding to the steady state. Later singular vectors take prominence in low frequency components which indicate the rate of change during the intermediate progression of the Markov chain.

All authors designed the research and analyzed results. BK and GP performed the research including the mathematical proofs and computations. BK, GP, and SL wrote the paper.

-
- [1] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, *Nature* **549**, 195 (2017).
 - [2] S. Lloyd, M. Mohseni, and P. Rebentrost, *Nature Physics* **10**, 631 (2014).
 - [3] M. Schuld, I. Sinayskiy, and F. Petruccione, *Physical Review A* **94**, 022342 (2016).
 - [4] P. Rebentrost, M. Mohseni, and S. Lloyd, *Physical review letters* **113**, 130503 (2014).
 - [5] A. Montanaro, *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **471**, 20150301 (2015).
 - [6] D. W. Berry, *Journal of Physics A: Mathematical and Theoretical* **47**, 105301 (2014).
 - [7] D. W. Berry, A. M. Childs, A. Ostrander, and G. Wang, *Communications in Mathematical Physics* **356**, 1057 (2017).
 - [8] S. Lloyd, G. De Palma, C. Gokler, B. Kiani, Z.-W. Liu, M. Marvian, F. Tennie, and T. Palmer, *arXiv preprint arXiv:2011.06571* (2020).
 - [9] J.-P. Liu, H. Ø. Kolden, H. K. Krovi, N. F. Loureiro, K. Trivisa, and A. M. Childs, *arXiv preprint arXiv:2011.03185* (2020).
 - [10] A. W. Harrow, A. Hassidim, and S. Lloyd, *Physical review letters* **103**, 150502 (2009).
 - [11] A. M. Childs, R. Kothari, and R. D. Somma, *arXiv preprint arXiv:1511.02306* (2015).

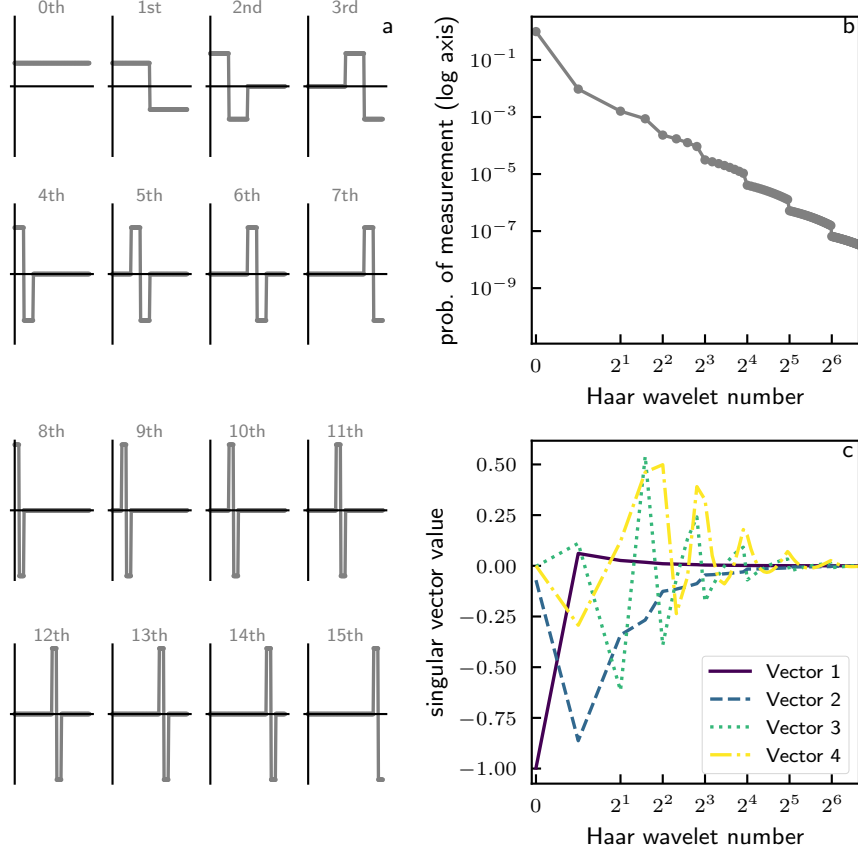


FIG. 4: **(a)** Each Haar wavelet has support over a characteristic timescale indicated by the wavelet number. Haar wavelet numbers between powers of 2 take the same form and are offset from each other in the time dimension. As a visual aid, we show here the first 16 discrete Haar wavelets. **(b)** Quantum state has high probability in the zeroth Haar wavelet (steady state). **(c)** Wavelet transform of singular vectors shows that the first singular vector is strongest in the zeroth Haar vector (steady state). Later singular vectors take prominence in Haar vectors with support in the early stages of epidemic.

- [12] R. P. Feynman, Optics news **11**, 11 (1985).
- [13] A. Y. Kitaev, A. Shen, M. N. Vyalyi, and M. N. Vyalyi, *Classical and quantum computation* (American Mathematical Soc., 2002).
- [14] S. Lloyd, arXiv preprint **arXiv:0805.2757** (2008).
- [15] R. Pastor-Satorras, C. Castellano, P. Van Mieghem, and A. Vespignani, Reviews of modern physics **87**, 925 (2015).
- [16] F. D. Sahneh, C. Scoglio, and P. Van Mieghem, IEEE/ACM Transactions on Networking **21**, 1609 (2013).
- [17] W. J. Anderson, *Continuous-time Markov chains: An applications-oriented approach* (Springer Science & Business Media, 2012).
- [18] A. Kolmogoroff, Mathematische Annalen **104**, 415 (1931).
- [19] E. Tang, arXiv preprint **arXiv:1811.00414** (2018).
- [20] N.-H. Chia, A. Gilyén, T. Li, H.-H. Lin, E. Tang, and C. Wang, in *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing* (2020) pp. 387–400.
- [21] M. Rudelson and R. Vershynin, Journal of the ACM (JACM) **54**, 21 (2007).
- [22] K.-M. Lau and H. Weng, Bulletin of the American meteorological society **76**, 2391 (1995).
- [23] D. B. Percival and A. T. Walden, *Wavelet methods for time series analysis*, Vol. 4 (Cambridge university press, 2000).
- [24] B. Cazelles, M. Chavez, G. C. d. Magny, J.-F. Guégan, and S. Hales, Journal of the Royal Society Interface **4**, 625 (2007).
- [25] A. Grinsted, J. C. Moore, and S. Jevrejeva, Nonlinear Processes in Geophysics **11**, 561 (2004).

- [26] P. Hoyer, arXiv preprint **quant-ph/9702028** (1997).
- [27] A. Fijany and C. P. Williams, in *NASA International Conference on Quantum Computing and Quantum Communications* (Springer, 1998) pp. 10–33.
- [28] A. Klappenecker and M. Rotteler, in *ISPA 2001. Proceedings of the 2nd International Symposium on Image and Signal Processing and Analysis. In conjunction with 23rd International Conference on Information Technology Interfaces (IEEE Cat. (IEEE, 2001) pp. 464–468.*
- [29] C. Shao, arXiv preprint **arXiv:1912.08015** (2019).
- [30] Z. Li, X. Liu, N. Xu, and J. Du, Physical review letters **114**, 140504 (2015).
- [31] G. Wang, Physical review A **96**, 012335 (2017).
- [32] E. Farhi and H. Neven, arXiv preprint **arXiv:1802.06002** (2018).
- [33] A. B. Grilo, I. Kerenidis, and T. Zijlstra, Physical Review A **99**, 032314 (2019).
- [34] J. Romero, J. P. Olson, and A. Aspuru-Guzik, Quantum Science and Technology **2**, 045001 (2017).
- [35] A. Khoshaman, W. Vinci, B. Denis, E. Andriyash, H. Sadeghi, and M. H. Amin, Quantum Science and Technology **4**, 014001 (2018).
- [36] M. Schuld and N. Killoran, Physical review letters **122**, 040504 (2019).
- [37] V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, and J. M. Gambetta, Nature **567**, 209 (2019).
- [38] S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, and N. Killoran, arXiv preprint **arXiv:2001.03622** (2020).
- [39] I. Cong, S. Choi, and M. D. Lukin, Nature Physics **15**, 1273 (2019).
- [40] M. Schuld, A. Bocharov, K. M. Svore, and N. Wiebe, Physical Review A **101**, 032308 (2020).
- [41] P. Van Mieghem and E. Cator, Physical Review E **86**, 016116 (2012).
- [42] J. Preskill, Quantum **2**, 79 (2018).
- [43] A. Bellante, A. Luongo, and S. Zanero, arXiv preprint arXiv:2104.08987 (2021).
- [44] I. Kerenidis and A. Prakash, arXiv preprint arXiv:1603.08675 (2016).
- [45] I. Kerenidis and A. Prakash, Physical Review A **101**, 022316 (2020).
- [46] L. Grover and T. Rudolph, arXiv preprint **quant-ph/0208112** (2002).
- [47] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant, Nature **585**, 357 (2020).
- [48] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, Í. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, Nature Methods **17**, 261 (2020).
- [49] P. L. Simon, M. Taylor, and I. Z. Kiss, Journal of mathematical biology **62**, 479 (2011).

Appendix A: Proofs of runtimes and error bounds

1 Generation of the quantum history state

Let $T = \lceil t_{\max} \|\mathcal{M}\| \rceil$, let $\delta > 0$ and let $k \in \mathbb{N}$ with $k \geq 5$ and $(k+1)! \geq 2T$. The quantum algorithm of Ref. [7] discretizes the differential equation in T time steps of size $h = t_{\max}/T$, truncates the Taylor expansion of the matrix exponential at the k -th order and produces an approximate normalized version $|\phi\rangle$ of the quantum state

$$|y\rangle = \sum_{i=0}^{T-1} \sum_{j=0}^k |i, j\rangle |y_{i,j}\rangle + |T, 0\rangle |y_{T,0}\rangle. \quad (\text{A1})$$

For sufficiently large k , the vectors $|y_{i,0}\rangle$ are close to the solution of the linear differential equation at the i -th time step [7, Theorem 6]:

$$\|\vec{x}(ih) - |y_{i,0}\rangle\| \leq 2.8 \kappa i \frac{\|\vec{x}(0)\| + t_{\max} \|\vec{c}\|}{(k+1)!}, \quad (\text{A2})$$

and the vectors $|y_{i,j}\rangle$ for $j \geq 1$ are some garbage vectors associated to the terms of the Taylor expansion of the matrix exponential, of which we do not need the exact form. Their norms are upper bounded by [7, eqs.

(99), (100)]

$$\| |y_{i,j}\rangle \| \leq \frac{\| |y_{i+1,0}\rangle \| + \| |y_{i,0}\rangle \|}{(3-e)j!} . \quad (\text{A3})$$

Let

$$|\bar{y}\rangle = \frac{|y\rangle}{\| |y\rangle \|} \quad (\text{A4})$$

be the normalized version of $|y\rangle$. The quantum state $|\phi\rangle$ satisfies

$$\| |\phi\rangle - |\bar{y}\rangle \| \leq \delta \quad (\text{A5})$$

and the algorithm requires

$$O\left(\kappa k^2 t_{\max} \|\mathcal{M}\| s \text{poly log } \frac{\kappa k t_{\max} \|\mathcal{M}\| s N}{\delta}\right) \quad (\text{A6})$$

elementary quantum gates [7, eq. (128)].

An approximation of the quantum history state $|x\rangle$ can be obtained by projecting the second register of the quantum state $|\phi\rangle$ on the 0 value. Let $\langle 0|\phi\rangle$ be the unnormalized projection. In the following, we show that its success probability is $O(1)$, and that choosing

$$\begin{aligned} \delta &= O(\epsilon) , \\ k &= O\left(\log\left(\left(1 + \frac{t_{\max} \|\vec{c}\|}{\|\vec{x}(0)\|}\right) \frac{\kappa t_{\max} \|\mathcal{M}\|}{\epsilon}\right)\right) \end{aligned} \quad (\text{A7})$$

we can achieve

$$\left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - |\bar{x}\rangle \right\| \leq \epsilon . \quad (\text{A8})$$

From Eq. A6, this modification of the algorithm of Ref. [7] requires

$$\begin{aligned} &O(\kappa t_{\max} \|\mathcal{M}\| s \\ &\text{poly log } \left(\left(1 + \frac{t_{\max} \|\vec{c}\|}{\|\vec{x}(0)\|}\right) \frac{\kappa t_{\max} \|\mathcal{M}\| s N}{\epsilon}\right)) \end{aligned} \quad (\text{A9})$$

elementary quantum gates.

Success probability

We assume that

$$\delta \leq \frac{1}{2\sqrt{66}} . \quad (\text{A10})$$

We have from Eq. A3

$$\begin{aligned} \sum_{j=1}^k \| |y_{i,j}\rangle \|^2 &\leq \frac{(\| |y_{i+1,0}\rangle \| + \| |y_{i,0}\rangle \|^2)}{(3-e)^2} \sum_{j=1}^{\infty} \frac{1}{j!^2} \\ &\leq 1.28 * 2 \frac{\| |y_{i+1,0}\rangle \|^2 + \| |y_{i,0}\rangle \|^2}{(3-e)^2} , \end{aligned} \quad (\text{A11})$$

and

$$\begin{aligned} \sum_{i=0}^{T-1} \sum_{j=1}^k \| |y_{i,j}\rangle \|^2 &\leq \frac{1.28 * 4}{(3-e)^2} \sum_{i=0}^T \| |y_{i,0}\rangle \|^2 \\ &\leq 65 \sum_{i=0}^T \| |y_{i,0}\rangle \|^2. \end{aligned} \quad (\text{A12})$$

Let $\langle 0|y\rangle$ be the projection of $|y\rangle$ onto the 0 value of the second register. The success probability of such projection is

$$\begin{aligned} \|\langle 0|\bar{y}\rangle\|^2 &= \frac{\sum_{i=0}^T \| |y_{i,0}\rangle \|^2}{\sum_{i=0}^T \| |y_{i,0}\rangle \|^2 + \sum_{i=0}^{T-1} \sum_{j=1}^k \| |y_{i,j}\rangle \|^2} \\ &\geq \frac{1}{66}. \end{aligned} \quad (\text{A13})$$

Let $\langle 0|\phi\rangle$ be the projection of $|\phi\rangle$ onto the 0 value of the second register. We have from Eq. A5, A13 and A10

$$\begin{aligned} \|\langle 0|\phi\rangle\| &\geq \|\langle 0|\bar{y}\rangle\| - \|\langle 0|\bar{y}\rangle - \langle 0|\phi\rangle\| \\ &\geq \|\langle 0|\bar{y}\rangle\| - \delta \geq \frac{1}{2\sqrt{66}}, \end{aligned} \quad (\text{A14})$$

therefore the success probability of the projection satisfies

$$p = \|\langle 0|\phi\rangle\|^2 \geq \frac{1}{264}. \quad (\text{A15})$$

Error analysis

From Eq. A2, the distance between the quantum history state $|x\rangle$ and the projection $\langle 0|y\rangle$ satisfies

$$\begin{aligned} \| |x\rangle - \langle 0|y\rangle \|^2 &= \sum_{i=0}^T \| \vec{x}(ih) - |y_{i,0}\rangle \|^2 \\ &\leq 2.8^2 \kappa^2 \frac{(\|\vec{x}(0)\| + t_{\max} \|\vec{c}\|)^2}{(k+1)!^2} \frac{T(T+1)(2T+1)}{6}, \end{aligned} \quad (\text{A16})$$

then

$$\| |x\rangle - \langle 0|y\rangle \| \leq \kappa \frac{\|\vec{x}(0)\| + t_{\max} \|\vec{c}\|}{2^{k+4}} \sqrt{3T} (T+1). \quad (\text{A17})$$

We have from Lemma 1

$$\begin{aligned} \left\| |\bar{x}\rangle - \frac{\langle 0|y\rangle}{\|\langle 0|y\rangle\|} \right\| &\leq \frac{\|\vec{x}(0)\| + t_{\max} \|\vec{c}\|}{\| |x\rangle \|} \frac{\kappa \sqrt{3T} (T+1)}{2^{k+3}} \\ &\leq \left(1 + \frac{t_{\max} \|\vec{c}\|}{\|\vec{x}(0)\|} \right) \frac{\kappa \sqrt{3T} (T+1)}{2^{k+3}}. \end{aligned} \quad (\text{A18})$$

We choose

$$\begin{aligned} k &= \left\lceil \log_2 \left(\left(1 + \frac{t_{\max} \|\vec{c}\|}{\|\vec{x}(0)\|} \right) \frac{\kappa \sqrt{3T} (T+1)}{4\epsilon} \right) \right\rceil \\ &= O \left(\log \left(\left(1 + \frac{t_{\max} \|\vec{c}\|}{\|\vec{x}(0)\|} \right) \frac{\kappa t_{\max} \|\mathcal{M}\|}{\epsilon} \right) \right), \end{aligned} \quad (\text{A19})$$

such that

$$\left\| |\bar{x}\rangle - \frac{\langle 0|y\rangle}{\|\langle 0|y\rangle\|} \right\| \leq \frac{\epsilon}{2}. \quad (\text{A20})$$

We have from Eq. A5, A13 and Lemma 1 again

$$\left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - \frac{\langle 0|\bar{y}\rangle}{\|\langle 0|\bar{y}\rangle\|} \right\| \leq \frac{2\|\langle 0|\phi\rangle - \langle 0|\bar{y}\rangle\|}{\|\langle 0|\bar{y}\rangle\|} \leq 2\sqrt{66}\delta, \quad (\text{A21})$$

such that choosing

$$\delta = \frac{\epsilon}{4\sqrt{66}} \quad (\text{A22})$$

we have

$$\left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - \frac{\langle 0|\bar{y}\rangle}{\|\langle 0|\bar{y}\rangle\|} \right\| \leq \frac{\epsilon}{2} \quad (\text{A23})$$

and

$$\begin{aligned} \left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - |\bar{x}\rangle \right\| &\leq \left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - \frac{\langle 0|\bar{y}\rangle}{\|\langle 0|\bar{y}\rangle\|} \right\| \\ &\quad + \left\| \frac{\langle 0|\bar{y}\rangle}{\|\langle 0|\bar{y}\rangle\|} - |\bar{x}\rangle \right\| \leq \epsilon \end{aligned} \quad (\text{A24})$$

as required.

Lemma 1. *Let v, w be vectors in a normed vector space. Then,*

$$\left\| \frac{v}{\|v\|} - \frac{w}{\|w\|} \right\| \leq 2 \frac{\|v - w\|}{\|v\|}. \quad (\text{A25})$$

Proof. We have

$$\left\| \frac{v}{\|v\|} - \frac{w}{\|w\|} \right\| \leq \frac{\|v - w\| + \|\|w\| - \|v\|\|}{\|v\|} \leq 2 \frac{\|v - w\|}{\|v\|}. \quad (\text{A26})$$

□

Appendix B: Input state preparation

Input states to differential equations encode boundary conditions and initial states. These input states can be prepared via two different methods discussed here.

a Efficiently constructed quantum initial states The most optimal setting which commonly occurs in differential equation analysis is when input vectors are sparse or efficient to construct on a quantum computer. For a standard linear differential equation in the form $\frac{d\vec{x}(t)}{dt} = \mathcal{M}\vec{x}(t) + \vec{c}$, the boundary conditions take the form below (copied from the main text):

$$\vec{b} = \begin{pmatrix} \vec{x}_0 \\ \vec{c} \\ \vdots \\ \vec{c} \end{pmatrix}.$$

Note, that in quantum algorithms, copies of the vector above are encoded into the quantum state at appropriate locations. It is often the case that \vec{b} is sparse or easily computed by local operations. This is especially true in the case of Markov chains algorithms where $c = 0$ and the initial state is either sparsely supported or locally independent. For example, applying Hadamard gates to all qubits in the initial state constitutes one easy method to construct the initial state with uniform support over all states. Similarly, applying single qubit operations to each node provides an easy method to form an initial state where one has knowledge of individual nodes independent of other nodes.

b Input states via qRAM If it is not possible to efficiently construct the initial state via sparse matrix or local operations as discussed above, another option one can use is inputting states via a quantum random access memory (qRAM) data structure [44, 45]. qRAM data structures store data $(i, x_i) \in [n]$ in a form that allows for quantum queries in superposition $|i, 0\rangle \rightarrow |i, x_i\rangle$ in time $O(\text{poly log}(n))$. Constructing such a data structure would, in general, require $O(n)$ quantum operations [45]. However, in settings where subsequent computations require longer runtimes, say $O(n^2)$ or $O(n^3)$ time, then such a qRAM data structure can be efficiently employed in our algorithms.

Appendix C: Sampling and calculating observables of Markov states

To determine the Tq^n individual probabilities of a Markov state $x_j(k)$, for $j = 0$ to T , $k = 1$ to $N = q^n$ stored in a quantum state, one would have to make an exponentially large number of measurements. However, one can use quantum measurements to reveal a wide variety of desired properties of the quantum system at time j . To extract the expectation value of some desired quantity Q (total number infected, variance of the infection rate across the graph, existence of hot spots, etc.) we need to make a quantum measurement to estimate the expectation value

$$\langle Q \rangle = \sum_{k=1}^N x_j(k) Q(k) \quad (C1)$$

where $Q(k)$ is the value of Q on the k 'th state of the vertices of the graph.

First, we have to make sure that we obtain the state $|\vec{x}_j\rangle$ with high probability. In the formulation given above, this state only occurs in the overall superposition $|x\rangle$ with amplitude $O(1/\sqrt{T})$. The standard way to amplify the probability of finding the answer at the desired time j [6] is to pad out the matrix A following the j 'th row with $O(T)$ rows of the form

$$(0 \dots 0 \quad -I \quad I \quad 0 \dots 0) \quad (C2)$$

where the I term in each $-I \quad I \quad 0$ sequence lies on the diagonal. These rows induce a trivial dynamics in which all the states in the solution following the j 'th state also contain the state $|\vec{x}_j\rangle$. This technique allows us to obtain the state $|\vec{x}_j\rangle$ with probability $O(1)$.

To obtain $\langle Q \rangle = \sum_{k=1}^N x_j(k) Q(k)$, we use standard techniques of quantum state preparation [46]. We assume that we are given access in quantum superposition to the individual values of the variable $Q(k)$ together with its partial sums over ranges of k ; The techniques of [46] then allow us to construct the (unnormalized) state

$$|Q\rangle = \sum_{k=1}^N Q(k) |k\rangle \quad (C3)$$

in time $O(\log N) = O(n \log q)$. Define $Z_Q = \sum_{k=1}^N Q(k)^2$. Even if Z_Q is not known beforehand, its value is revealed during the state preparation process [46]. The normalized version of $|Q\rangle$ is then

$$|\tilde{Q}\rangle = Z_Q^{-1/2} |Q\rangle \quad (C4)$$

We can now use a swap test between $|\tilde{Q}\rangle$ and $|\tilde{x}_j\rangle$ to measure the overlap $\langle\tilde{Q}|\tilde{x}_j\rangle$. This overlap in turn allows us to measure

$$\langle Q \rangle = \sum_{k=1}^N x_j(k) Q(k) = \langle Q | x_j \rangle = Z_j^{1/2} Z_Q^{1/2} \langle \tilde{Q} | \tilde{x}_j \rangle \quad (\text{C5})$$

In conclusion, even though we don't have access to the individual probabilities for states, the quantum algorithm allows us to measure expectation values for a wide variety of observables efficiently. This method allows us to use the quantum algorithm to extract expectation values of the desired quantities.

Comparison to classical complexity Extracting expectation values could also be done classically using classical Monte Carlo to sample from the probabilistic dynamics. Because of the local form of the probabilistic updating rule, the number of computational steps required to draw one sample of the Markov chain at time t scales as

$$O(n(t/h) \log q). \quad (\text{C6})$$

The average of Q over $O(\log \frac{1}{\delta} / \epsilon^2)$ samples is ϵ -close to $\langle Q \rangle$ with probability at least $1 - \delta$, and its computation has complexity

$$O\left(\frac{n(t/h) \log \frac{1}{\delta} \log q}{\epsilon^2}\right). \quad (\text{C7})$$

Quantum speedup of Monte Carlo sampling The quantum algorithm of Ref. [5] provides a quadratic improvement in the dependence of the complexity Eq. C7 of Monte Carlo sampling on the precision ϵ . More precisely, let us assume that $0 \leq Q(k) \leq 1$ for any $k = 1, \dots, N$ (this can always be achieved by a suitable linear redefinition of Q). Let U be a quantum unitary operator that implements a unitary dilation of the classical algorithm for the Monte Carlo sampling of the Markov chain. We can assume that the complexity of U has at worst a constant overhead with respect to the classical algorithm, and therefore has the same scaling Eq. bC6. Then, Theorem 2.3 of [5] implies that, for any $0 < \epsilon < 1$ and any $0 < \delta < 1$ there exists a quantum algorithm that, with $O(\log \frac{1}{\delta} / \epsilon)$ applications of U , outputs $\mu \in \mathbb{R}$ such that $|\mu - \langle Q \rangle| < \epsilon$ with probability at least $1 - \delta$. The complexity of the algorithm is therefore

$$O\left(\frac{n(t/h) \log \frac{1}{\delta} \log q}{\epsilon}\right), \quad (\text{C8})$$

with the promised quadratic improvement in the ϵ -dependence with respect to Eq. C7.

Appendix D: Classical algorithms for Principal Component Analysis

Here we show that the singular vectors and singular values of the data matrix \mathbf{X} cannot be efficiently estimated with classical methods whenever $\| |x_j\rangle \|^2$ is exponentially small in the number of nodes for any time step, *i.e.*, if the size of the support of the probability distribution is always exponential. More precisely, we show that none of the entries of $\mathbf{X}^\dagger \mathbf{X}$ can be estimated efficiently. Indeed, if $x_j(k)$ is the probability of the k -th state at the time step j , the jj' entry of $\mathbf{X}^\dagger \mathbf{X}$ is

$$(\mathbf{X}^\dagger \mathbf{X})_{jj'} = \sum_k x_j(k) x_{j'}(k), \quad (\text{D1})$$

and is equal to the probability that, in a couple of independent Monte Carlo simulations of the Markov chain, the state of the first simulation at the step j is equal to the state of the second simulation at the step j' . This probability can be estimated by running many couples of simulations of the Markov chain. However, the estimate will be zero until a couple with the state of the first simulation at the step j equal to the state of the second simulation at the step j' is found. This event will typically happen after

$$O\left(\frac{1}{(\mathbf{X}^\dagger \mathbf{X})_{jj'}}\right) \geq O\left(\frac{1}{\max_i \| |x_i\rangle \|^2}\right) \quad (\text{D2})$$

runs, which is exponentially large in the number of nodes if $\| |x_j\rangle \|^2$ is exponentially small for any time step.

Appendix E: Additional simulations and figures

1 Epidemic simulations of social opinion

Continuous time Markov chains can also be implemented to simulate the spread of social opinion. Here, we consider a model where nodes can exist in one of three states: conservative, liberal, or undecided. As in our analysis on viral epidemics, we perform simulations on the same 7-node network as in the main text. Similar to before, transitions from undecided to liberal or conservative occur at rates dependent on the strength of connection to other liberal or conservative nodes respectively. The data matrix is analyzed between days one and two of the "social epidemic", where at day zero, all states are equally likely.

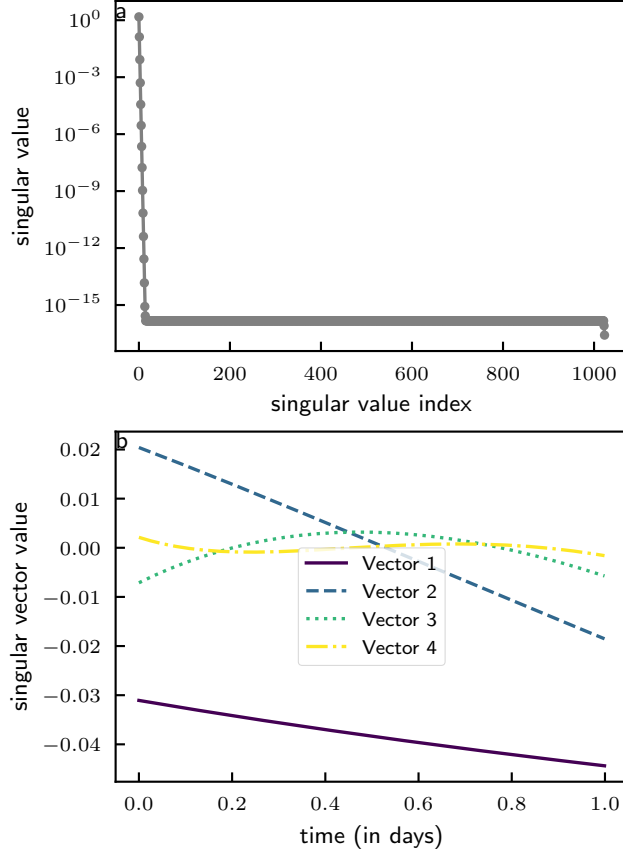


FIG. 5: Quantum principal component analysis of the spread of social opinion. **(a)** Singular values of data matrix \mathbf{X}_{mat} decay exponentially fast. **(b)** Values of the right singular vectors scaled by the square root of their corresponding singular values show the progression of social opinion over time. The results are consistent with prior results for the analysis of viral epidemics on the same network.

As shown in Fig. 5a, the data matrix in this case is similarly low rank. Furthermore, we see similar progressions over time in the right singular vectors as shown in Fig. 5b. The first singular vector corresponds to steady state contributions, whereas later singular vectors chart the most prominent changes in the data matrix over time.

2 Model of social distancing

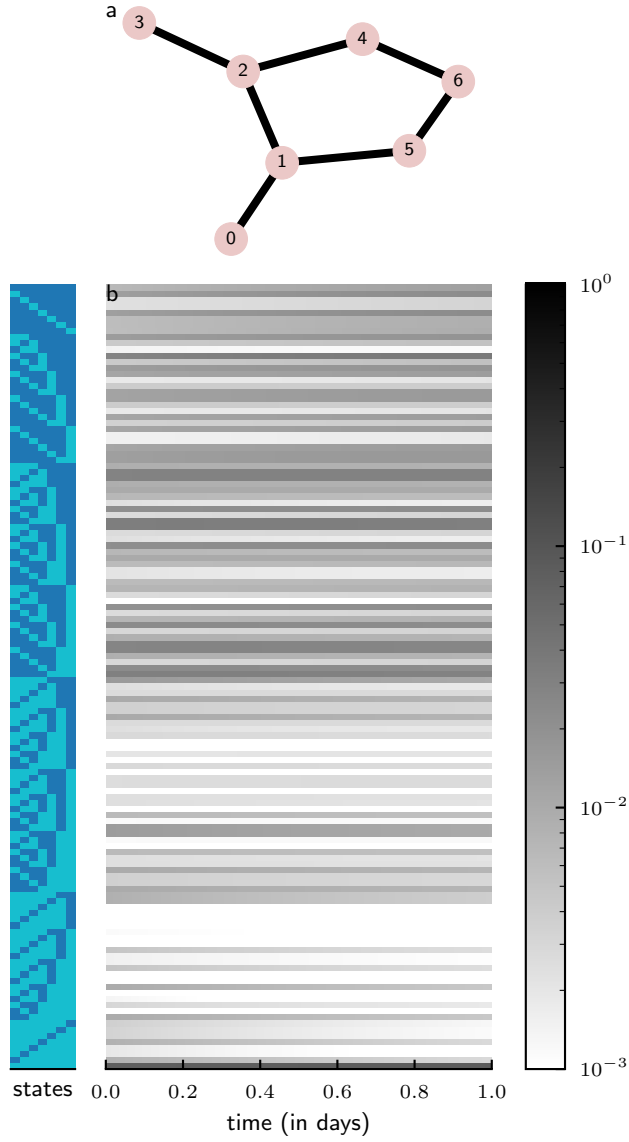


FIG. 6: **(a)** The 7 node network used for simulation of continuous time Markov chain where transitions from susceptible to infected and vice-versa occur with rates $r_{SI} = 1.5$, reduced by a factor of five when four or more nodes are infected. **(b)** Probabilities of Markov states over time shown as colorbar chart (logarithmically scaled). States are enumerated as rows on the left hand side, each denominated by a 7 node colorbar numbered node 0 on the left to node 6 on the right. Dark/light color indicates a node in that state is infected/susceptible respectively.

Supplementary to the main text, we show results here for a simulation of a Markov chain which incorporates effects of "social distancing" in the Markov model. Specifically, we simulate a viral epidemic on the same network as in the main text where transitions from susceptible to infected and vice-versa occur with rates $r_{SI} = 1.5$ and $r_{IS} = 0.33$ respectively as long as three or fewer nodes are infected. When four or more nodes

are infected, "social distancing" is enacted and transitions from susceptible to infected occur at a fifth of the original rate ($r_{SI} = 0.3$). As expected, this shifts the steady state away from situations where all nodes are infected to those where four nodes are infected.

The complete progression of this model is plotted in Fig. 6b. Note, that the state where all nodes are infected is now unlikely; instead, the states where four or five nodes are infected become very likely (*i.e.*, the social distancing works).

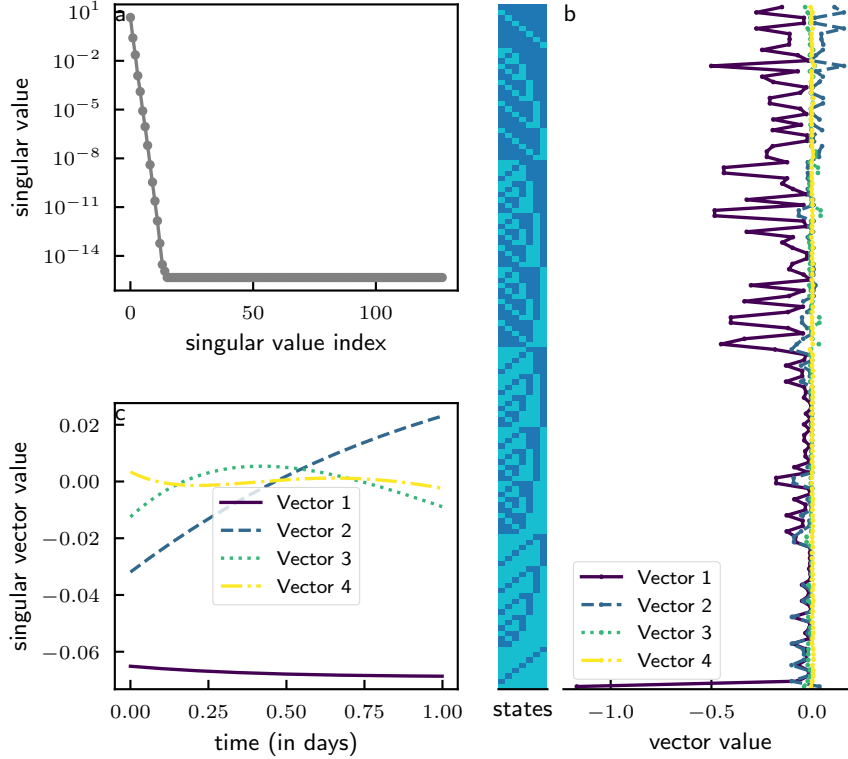


FIG. 7: **(A)** Singular values of data matrix \mathbf{X} decay exponentially fast. **(B)** First four left singular vectors scaled by the square root of their corresponding singular value show that states where four nodes are infected become prominent as this is the inflection point for social distancing. States are enumerated as rows on the left hand side, each denominated by a 7 node colorbar numbered node 0 on the left to node 6 on the right. Dark/light colors indicate a node in that state is infected/susceptible respectively. **(C)** Values of the right singular vectors scaled by the square root of their corresponding singular value show the progression of the epidemic over time. The first singular vector depicts the steady state and the next few singular vectors detail the intermediate course of the Markov process.

As with the original model, singular values decay exponentially rapidly (see Fig. 7a). The first four left singular vectors are plotted in Fig. 7b scaled by their corresponding singular value. The steady state singular vector has clearly changed with respect to the original model. Analysis of the first singular vector shows that the dominant states are those where no node is infected and four nodes are infected (*i.e.*, the transition point of social distancing).

Appendix F: Markov states incorporating more than just simple nodes

Markov models can incorporate nodes of different types which interact in a customized fashion. Fig. 8 outlines some of the different options available to one modeling epidemiological processes. Utilizing quantum

algorithms, nodes of different types can be stored in separate registers. Analysis and post-processing of the Markov states using a quantum state can take advantage of the structure inherent in these expanded models.

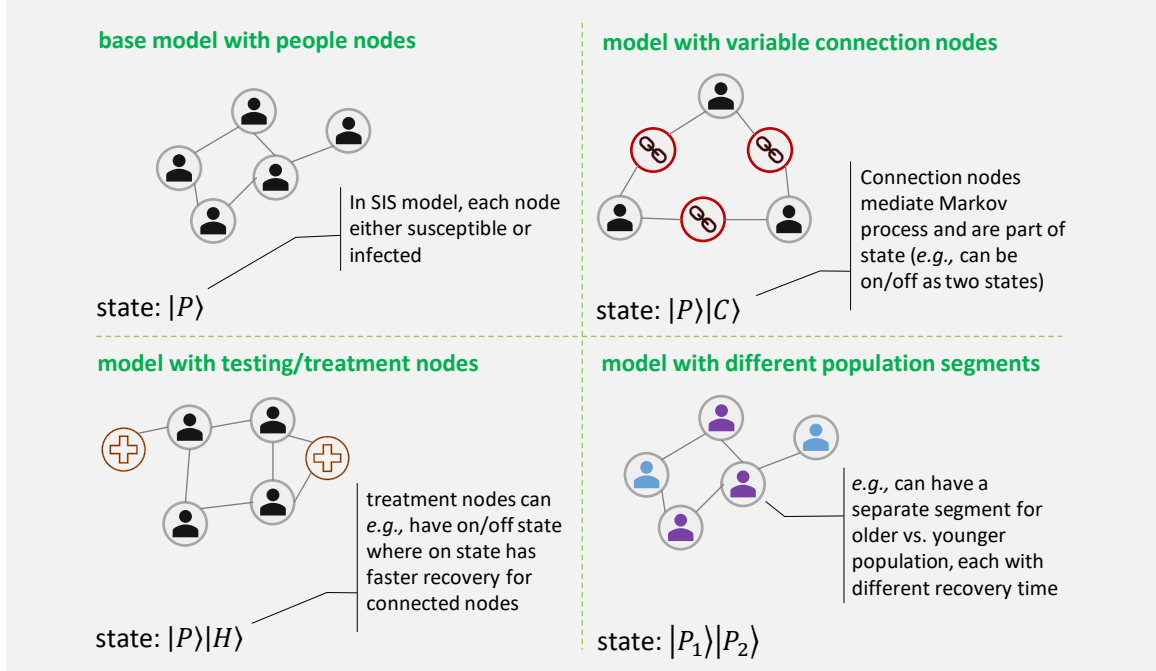


FIG. 8: By customizing the properties of nodes in a network, Markov states can model various different phenomenon. Here, we show some of the options available to expand the functionality of a Markov state. In these cases, states are stored in a tensor product structure where information corresponding nodes of different types can be stored in separate registers.

Appendix G: Experimental details

All experiments were performed in Python using the packages Numpy [47] and Scipy [48]. For simulations of epidemic spreading, to construct the transition matrix Q for our experiments, we use the method detailed in [49]. We assume transitions from infected to susceptible (*i.e.*, recovery rate) occur at rate $r_{IS} = 0.33$ indicating that it takes about three days on average to recover from infection. Transitions from susceptible to infected occur at a rate $r_{SI} \in \{0.4, 0.8, 1.6\}$ depending on a node's connection strength to a neighboring node.

To simplify the exposition, our experiments and simulations were performed classically. In Appendix A, we further discuss, the asymptotic runtimes and errors for performing our algorithms on a gate based quantum computer. We also show that input states for Markov models can typically be efficiently constructed using simple quantum operations.