Quantum advantage for differential equation analysis

Bobak T. Kiani,[1*] Giacomo De Palma,[2,5] Dirk Englund,[1,2,3]
William Kaminsky, Milad Marvian,[4] Seth Lloyd[2,5]

[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]QuEra Computing, Boston, MA 02143
[4]Department of Electrical and Computer Engineering, Center for Quantum Information and Control (CQuIC),University of New Mexico, Albuquerque NM 87131
[5]Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, MA 02139

*To whom correspondence should be addressed; E-mail: bkiani@mit.edu.

Quantum algorithms for both differential equation solving and for 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 machine learning is that inputting the training set is a difficult task just by itself. In this paper, 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 machine learning, allowing dynamical analysis in terms of princi-

pal 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.

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 $\mathrm{poly}(\log N)$, 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 algorithms for the analysis of data provide potential exponential speedups over classical machine learning algorithms for methods such as high-dimensional regression, principal component analysis, and support vector machines [*1–4*]. A basic problem with such quantum machine learning 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 paper is that the problem with quantum linear equation solvers –

they give quantum states as output – and the problem with quantum machine learning 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 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. The singular value decomposition and the Fourier analysis reveal the dominant components of the time evolution, corresponding to large singular values, and to 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 here 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. We have chosen to investigate Markov processes because Monte Carlo methods represent a powerful classical method for revealing the features of such processes. Previous quantum algorithms for Monte Carlo 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.

Results

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. We write a general linear differential equation for a vector in $\mathbb{R}^N$ with $N \gg 1$ in the form

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

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 $\mathcal{M}$ is 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 [*8, 9*]. 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

$$A\vec{x} = \vec{b} \tag{2}$$

where

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

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

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

and

$$\vec{b} = \begin{pmatrix} \vec{x}_0 \\ \vec{c} \\ \vdots \\ \vec{c} \end{pmatrix} \tag{5}$$

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

$$\vec{x} = A^{-1}\vec{b} \tag{6}$$

Roughly speaking, the quantum algorithms of [6, 7] map the classical states onto quantum states, $\vec{x} \to |x\rangle$, $\vec{b} \to |b\rangle$, 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. [6], 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 the Methods section 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. [*6*] with $O(1)$ success probability.

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

$$|x\rangle = \sum_{i=0}^{T} |x_i\rangle\,|i\rangle \tag{7}$$

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. Such algorithms can provide useful information about the solution which cannot be obtained efficiently using classical computation, and we overview these algorithms in later sections.

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 [*10, 11*]. The quantum post-processing techniques that we apply here to history states of linear differential equations can also be applied to the history state of the ground state of a local Hamiltonian.

Scaling of the error The computational complexity of the quantum algorithm of Ref. [*6*] is linear in the condition number and the time, and logarithmic in the error. We show in the Methods section 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 $|\bar{x}\rangle$ in 2-norm with

$$O\left(\kappa\, t_{\max}\, \|\mathcal{M}\|\, s\, \mathrm{poly}\log\left(\left(1+\frac{t_{\max}\,\|\vec{c}\|}{\|\vec{x}(0)\|}\right)\frac{\kappa\, t_{\max}\,\|\mathcal{M}\|\, s\, N}{\epsilon}\right)\right) \tag{8}$$

elementary quantum gates. Here κ is the condition number of the matrix that is used to diagonalize $\mathcal{M}$ and s is the sparsity of $\mathcal{M}$.

Application to the Schrödinger equation In the case where $\mathcal{M}$ is anti-Hermitian, then Eq. 1 is the Schrödinger equation [*10*]. 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 [*12*].

Continuous time Markov chains To illustrate the power of our methodology, in this paper, we focus on differential equations for continuous time Markov chains. Markov chains are a primary tool in mathematics and statistics for modeling the transitions over time of a system which can exist in one of a set of states. Markov models 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. Markov chains have applications in many fields including bioinformatics, population dynamics, and statistical controls [*13*].

Here, we focus on Markov chains that represent dynamics of complex networks, particularly those modeling the spread of opinion and disease in social networks. 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 [*14, 15*]. 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 to modeling the spread of epidemics via complex networks of interactions typically rely on various approximations, such as mean field theory [*14*].

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

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

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 [*16, 17*].

Eq. 9 is precisely of the form required in Eq. 1 for efficiently solving differential equations using quantum algorithms. As shown in the Methods section, the quantum algorithm of Ref. [*6*] 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. Since $|x\rangle$ contains the complete information of the Markov state at different points in time, post-processing algorithms, which we now outline, can be used to extract useful information from the state.

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. [*6*],

which relies on the Taylor expansion of the matrix exponential, but it can easily be incorporated in the previous algorithm of Ref. [7], which directly solves Eq. 2.

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 the Supplementary Material, 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 the Supplementary Material, 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 In contrast to the expectation values of observables, features of the solution such as the singular value decomposition, the power spectrum, and wavelet analysis cannot be reconstructed efficiently by sampling from

the Monte Carlo solution. Here, the power of quantum computation for performing linear algebra on high dimensional spaces provides an exponential speedup over the best known classical methods.

Principal component analysis of the history state of the differential equation yields useful information about the solution. First of all, 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. The space spanned by these eigenvectors in turn has high overlap with the space spanned by the dominant singular vectors.

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 $\boldsymbol{X}$ where each column j corresponds to $|x_j\rangle$, 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 $\boldsymbol{X}$ [*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 $\boldsymbol{X}$ 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 [*18–20*]. Specifically, in the Supplementary Material, 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 exponentiation on copies of $|x\rangle$, we have a decomposition of the data matrix into left and right singular vectors:

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

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 trans-

form 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.

We noted above that when the Markov transition matrix has only a few eigenvalues with small negative parts, the history state will be effectively low-rank, with only a few large singular values. The quantum singular value analysis can be used to probe other aspects of the history as well. For example, we can perform quantum principal component analysis on a sliding window within the history: as this window moves forward, a fundamental change in the dynamics of the process can have a signature in changes in the singular values, and in the form of the corresponding singular vectors. These changes can be revealed by performing swap tests between the dominant singular vectors at different points during the evolution.

Post-processing: efficient quantum transformations Fourier transforms and wavelet transforms are commonly used in the analysis of large datasets, especially time series [*21–24*]. 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 [*25–27*]. 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

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

where $|y_k\rangle = \sum_{j=0}^{T} U_{jk} |x_j\rangle$, is the state of the system correlated with the kth 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 complex eigenvalues of $\mathcal{M}$ and the output contains the corresponding eigenvectors [*28*]. 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 section on classical algorithms for principal component analysis in the Supplementary Material).

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 in cases where data is input as a quantum state [*1, 3, 4, 29–33*]. Using quantum algorithms for continuous time Markov chain models proposed here, input states for these algorithms are generated efficiently thus preserving the exponential speedup of these algorithms even in the construction of input states.

When data in the form given by Eq. 7 is input into machine learning algorithms, applications of machine learning algorithms are numerous. Here we list some of these applications, grouped based on the type of model that can be used. First, quantum models have been proposed for compression of data or efficient readout. These models include quantum auto-encoders [*33, 34*] 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, 35–37*]. 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; similarly, if we trace out the state register, the temporal register is described by the density matrix $\sum_{ij} \langle x_i|x_j\rangle |j\rangle \langle i|$, which is the kernel matrix for the data. That is, the quantum differential equation solver gives us direct access to the states required to attain the potential exponential speedups of quantum kernel methods. Third, output states can be inputted into parameterized quantum circuits or quantum neural networks to identify key features or perform machine learning tasks [*31, 38, 39*].

Example simulation for epidemic processes The analysis of epidemic spreading – viral or social – is often modeled as a dynamical process on a complex network [*14*]. 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 [*14, 40*]. 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 both for analyzing viral and social epidemics. Also, for the sake of brevity and ease of graphical presentation, we implement susceptible-infected-susceptible (SIS) epidemiological models which have only two states per node: susceptible and infected. Nodes can become infected multiple times in this model – *i.e.,* there is no recovered state as in a susceptible-infected-recovered (SIR) model. For contagion on social networks, we consider

models where nodes can exist in one of three states labeled liberal, conservative, and undecided.

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 quantum computers [*14, 40*].

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

$$\frac{d\vec{x}}{dt} = \boldsymbol{Q}\vec{x} \tag{12}$$

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. Specifically, during that period, we numerically construct the history state of the Markov simulation using 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):

$$\vec{x}_{t+1} = \vec{x}_t + h\boldsymbol{Q}\vec{x}_t \tag{13}$$

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.

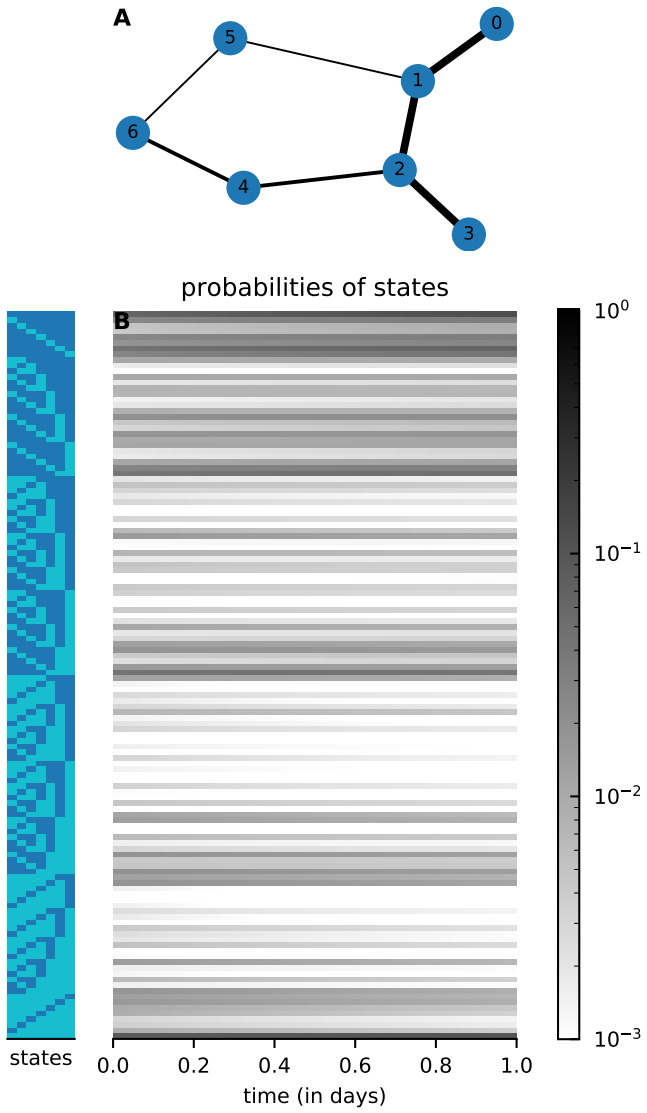

Figure 1: **(A)** The 7 node network used for simulation of the continuous time Markov chain. Line widths correspond to the rate of infection from infected neighbor nodes ($r_{SI} = \{0.4, 0.8, 1.6\}$ from thinnest to thickest lines). Network has the feature that some nodes (*e.g.,* node 2 and 3) are strongly connected to neighboring nodes and others (*e.g.,* node 5) are not. **(B)** Probabilities of Markov states over time shown as a 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.

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

$$\boldsymbol{X} = \begin{bmatrix} | & | & & | \\ \vec{x}_0 & \vec{x}_1 & \dots & \vec{x}_T \\ | & | & & | \end{bmatrix} \tag{14}$$

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

To construct the matrix $\boldsymbol{Q}$, we use the method detailed in [*41*]. 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 (see Fig. 1A caption).

In Fig. 1B, we plot the probabilities of states of the Markov process over time providing a visualization of the data matrix $\boldsymbol{X}$. 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 can subsequently post-process using various efficient quantum algorithms. One available post-processing algorithm is quantum Principal Component Analysis (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 decaying 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.

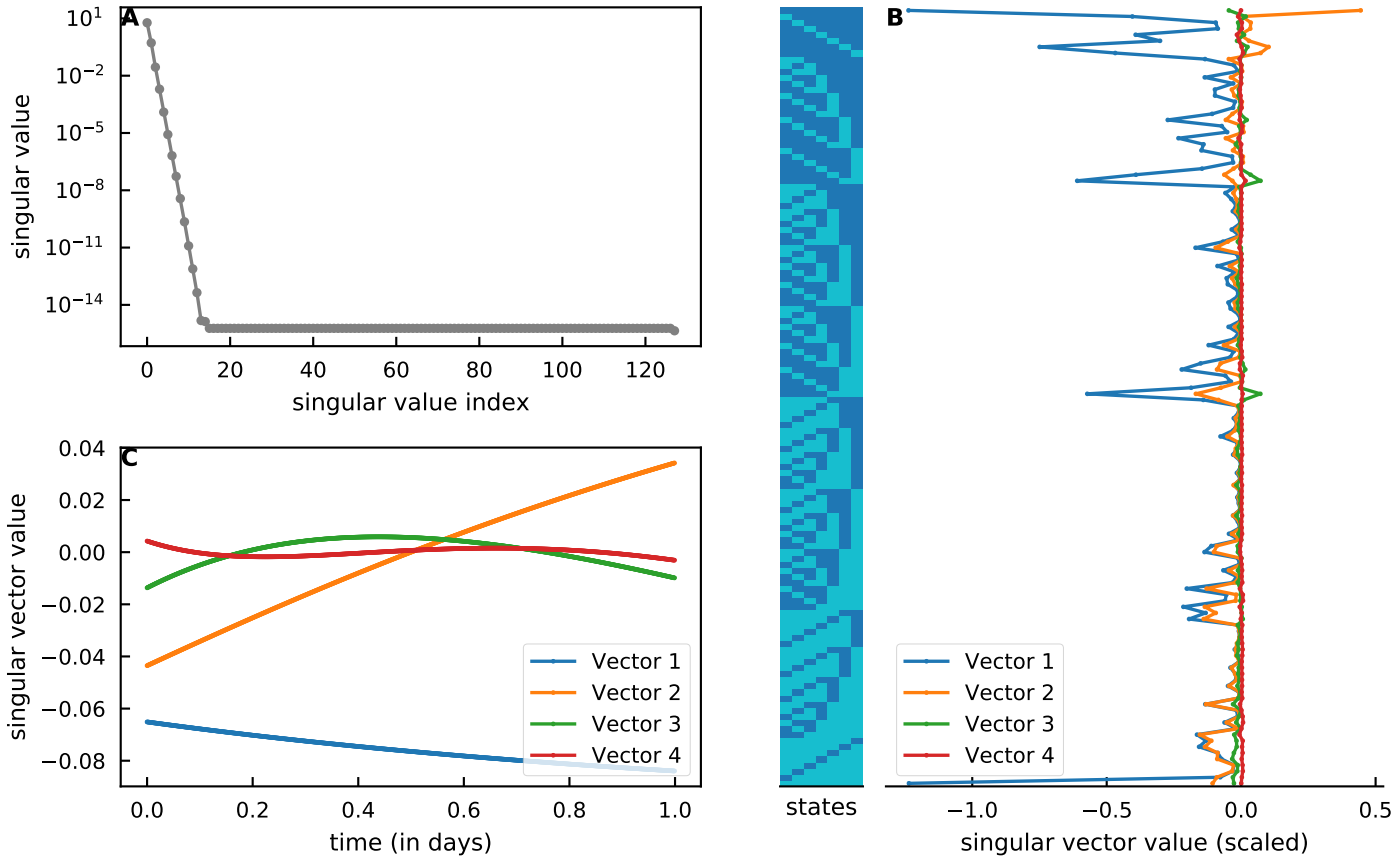

Figure 2: **(A)** Singular values of data matrix $\boldsymbol{X}$ decay exponentially fast. **(B)** First four left singular vectors scaled by the square root of their corresponding singular values show that much of the disease progression can be understood by just observing the first few vectors. States 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.

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 transition to recovery or infection.

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 [*14, 40*] if they are analyzed with a quantum computer.

Quantum computers offer the advantage of efficient post-processing via wavelet transforms. Here, we transform the time dimension of our data matrix using a Haar wavelet transform, which can be performed efficiently on a quantum computer [*25*]. 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

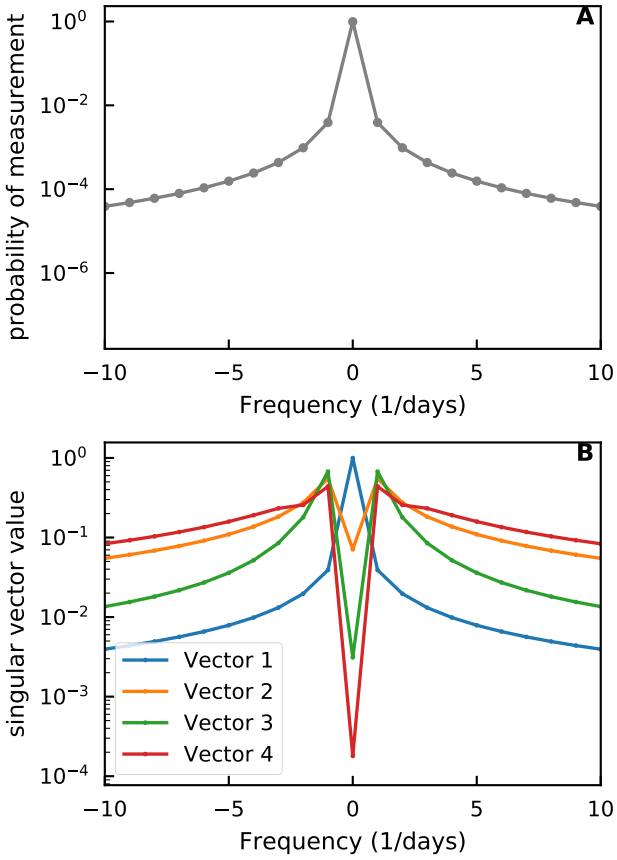

Figure 3: **(A)** Measurements made in the Fourier domain (plotted only for low frequency components) have high probability in the zero frequency (steady state) and low frequency components. **(B)** The first singular vector, plotted in the Fourier domain, is dominated by the zero frequency component 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.

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).

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 (see figure 1A). 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.

As shown in figure 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 figure 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.

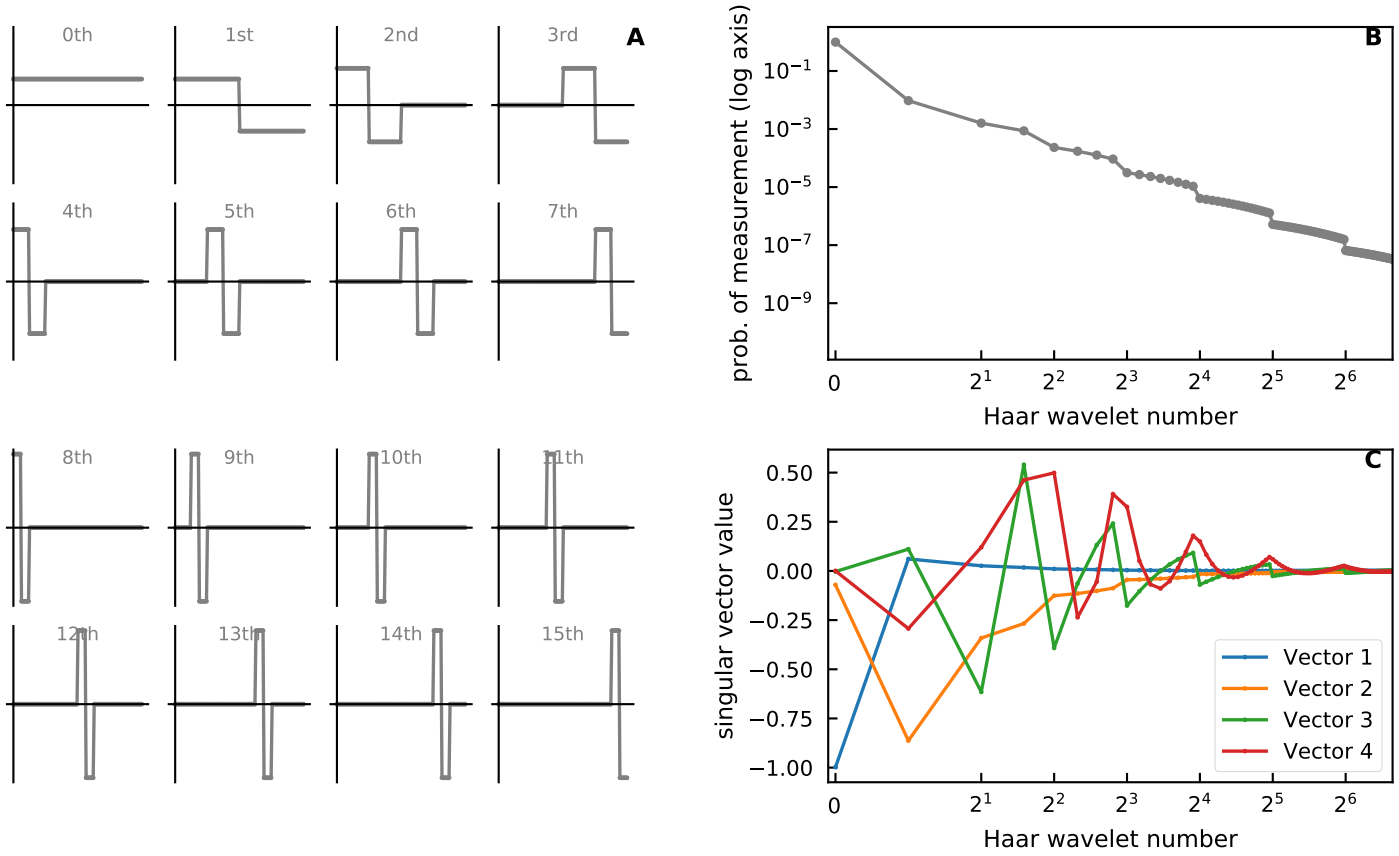

Figure 4: **(A)** Each Haar wavelet has support over a characteristic timescale indicated by the wavelet number. Haar wavelet numbers between powers of 2 take the same form and are offset from each other in the time dimension. As a visual aid, we show here the first 16 discrete Haar wavelets. **(B)** 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.

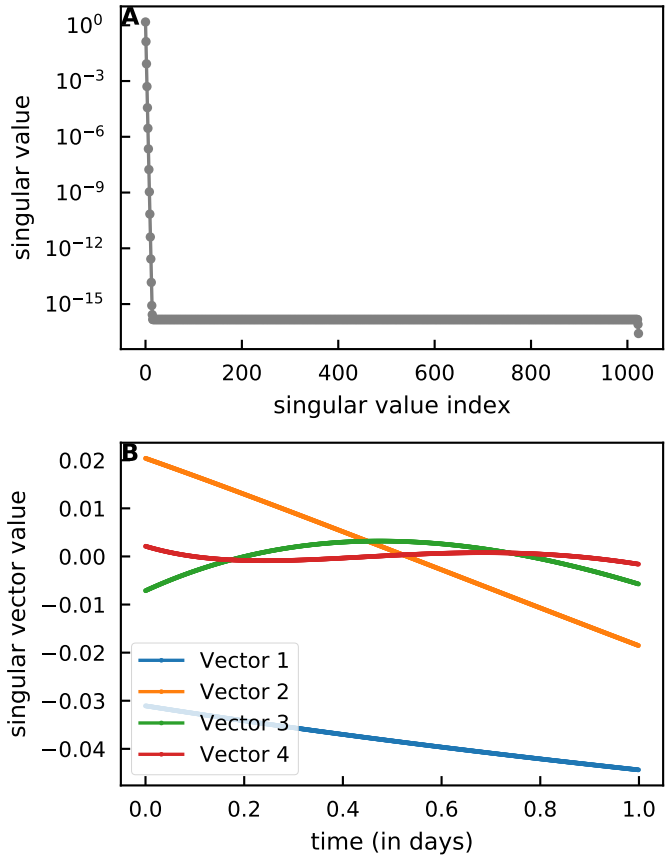

Figure 5: Quantum principal component analysis of the spread of social opinion. **(A)** Singular values of data matrix $\boldsymbol{X}$ decay exponentially fast. **(B)** Values of the right singular vectors scaled by the square root of their corresponding singular values show the progression of social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

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 signals processing algorithms – that classical inputs must be mapped into quantum states or stored in qRAM. The quantum linear differential equation solver efficiently produces as outputs states of exactly the form required for quantum machine learning and signals processing algorithms. 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 models 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.

Materials and Methods

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. [6] discretizes the differential equation (Eq. 1) 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

$$|y\rangle = \sum_{i=0}^{T-1} \sum_{j=0}^{k} |i,j\rangle |y_{i,j}\rangle + |T,0\rangle |y_{T,0}\rangle \,. \tag{15}$$

For sufficiently large k, the vectors $|y_{i,0}\rangle$ are close to the solution of the linear differential equation (Eq. 1) at the i-th time step [*6, Theorem 6*]:

$$\| \vec{x}(ih) - |y_{i,0}\rangle \| \leq 2.8\, \kappa\, i\, \frac{\|\vec{x}(0)\| + t_{\max}\, \|\vec{c}\|}{(k+1)!} \,, \tag{16}$$

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 [*6, eqs. (99), (100)*]

$$\| |y_{i,j}\rangle \| \leq \frac{\| |y_{i+1,0}\rangle \| + \| |y_{i,0}\rangle \|}{(3-\mathrm{e})\, j!} \,. \tag{17}$$

Let

$$|\bar{y}\rangle = \frac{|y\rangle}{\| |y\rangle \|} \tag{18}$$

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

$$\| |\phi\rangle - |\bar{y}\rangle \| \leq \delta \tag{19}$$

and the algorithm requires

$$O\left(\kappa\, k^2\, t_{\max}\, \|\mathcal{M}\|\, s\, \mathrm{poly}\log \frac{\kappa\, k\, t_{\max}\, \|\mathcal{M}\|\, s\, N}{\delta} \right) \tag{20}$$

elementary quantum gates [*6, eq. (128)*].

An approximation of the quantum history state $|x\rangle$ of Eq. 7 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

$$\delta = O(\epsilon) \,, \qquad k = O\left(\log\left(\left(1 + \frac{t_{\max}\, \|\vec{c}\|}{\|\vec{x}(0)\|} \right) \frac{\kappa\, t_{\max}\, \|\mathcal{M}\|}{\epsilon} \right) \right) \tag{21}$$

we can achieve

$$\left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - |\bar{x}\rangle \right\| \leq \epsilon \,. \tag{22}$$

From Eq. 20, this modification of the algorithm of Ref. [6] requires

$$O\left(\kappa\, t_{\max}\, \|\mathcal{M}\|\, s\, \mathrm{poly}\log\left(\left(1+\frac{t_{\max}\,\|\vec{c}\|}{\|\vec{x}(0)\|}\right)\frac{\kappa\, t_{\max}\,\|\mathcal{M}\|\, s\, N}{\epsilon}\right)\right) \tag{23}$$

elementary quantum gates.

Success probability

We assume that

$$\delta \leq \frac{1}{2\sqrt{66}} \,. \tag{24}$$

We have from Eq. 17

$$\sum_{j=1}^{k} \||y_{i,j}\rangle\|^2 \leq \frac{\left(\||y_{i+1,0}\rangle\| + \||y_{i,0}\rangle\|\right)^2}{(3-\mathrm{e})^2} \sum_{j=1}^{\infty} \frac{1}{j!^2} \leq 1.28 * 2\, \frac{\||y_{i+1,0}\rangle\|^2 + \||y_{i,0}\rangle\|^2}{(3-\mathrm{e})^2} \,, \tag{25}$$

and

$$\sum_{i=0}^{T-1}\sum_{j=1}^{k} \||y_{i,j}\rangle\|^2 \leq \frac{1.28*4}{(3-\mathrm{e})^2} \sum_{i=0}^{T} \||y_{i,0}\rangle\|^2 \leq 65 \sum_{i=0}^{T} \||y_{i,0}\rangle\|^2 \,. \tag{26}$$

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

$$\|\langle 0|\bar{y}\rangle\|^2 = \frac{\sum_{i=0}^{T} \||y_{i,0}\rangle\|^2}{\sum_{i=0}^{T} \||y_{i,0}\rangle\|^2 + \sum_{i=0}^{T-1}\sum_{j=1}^{k} \||y_{i,j}\rangle\|^2} \geq \frac{1}{66} \,. \tag{27}$$

Let $\langle 0|\phi\rangle$ be the projection of $|\phi\rangle$ onto the 0 value of the second register. We have from Eq. 19, 27 and 24

$$\|\langle 0|\phi\rangle\| \geq \|\langle 0|\bar{y}\rangle\| - \|\langle 0|\bar{y}\rangle - \langle 0|\phi\rangle\| \geq \|\langle 0|\bar{y}\rangle\| - \delta \geq \frac{1}{2\sqrt{66}} \,, \tag{28}$$

therefore the success probability of the projection satisfies

$$p = \|\langle 0|\phi\rangle\|^2 \geq \frac{1}{264} \,. \tag{29}$$

Error analysis

From Eq. 16, the distance between the quantum history state $|x\rangle$ and the projection $\langle 0|y\rangle$ satisfies

$$\| |x\rangle - \langle 0|y\rangle \|^2 = \sum_{i=0}^{T} \| \vec{x}(ih) - |y_{i,0}\rangle \|^2 \leq 2.8^2 \, \kappa^2 \frac{\left(\|\vec{x}(0)\| + t_{\max} \|\vec{c}\| \right)^2}{(k+1)!^2} \frac{T\,(T+1)\,(2T+1)}{6} \,, \tag{30}$$

then

$$\| |x\rangle - \langle 0|y\rangle \| \leq \kappa \frac{\|\vec{x}(0)\| + t_{\max} \|\vec{c}\|}{2^{k+4}} \sqrt{3T}\,(T+1) \ . \tag{31}$$

We have from Lemma 1

$$\left\| |\bar{x}\rangle - \frac{\langle 0|y\rangle}{\|\langle 0|y\rangle\|} \right\| \leq \frac{\|\vec{x}(0)\| + t_{\max} \|\vec{c}\|}{\||x\rangle\|} \frac{\kappa\sqrt{3T}\,(T+1)}{2^{k+3}} \leq \left(1 + \frac{t_{\max} \|\vec{c}\|}{\|\vec{x}(0)\|} \right) \frac{\kappa\sqrt{3T}\,(T+1)}{2^{k+3}} \,. \tag{32}$$

We choose

$$\begin{aligned} k &= \left\lceil \log_2 \left(\left(1 + \frac{t_{\max} \|\vec{c}\|}{\|\vec{x}(0)\|} \right) \frac{\kappa\sqrt{3T}\,(T+1)}{4\epsilon} \right) \right\rceil \\ &= O\left(\log \left(\left(1 + \frac{t_{\max} \|\vec{c}\|}{\|\vec{x}(0)\|} \right) \frac{\kappa\, t_{\max} \|\mathcal{M}\|}{\epsilon} \right) \right) \,, \end{aligned} \tag{33}$$

such that

$$\left\| |\bar{x}\rangle - \frac{\langle 0|y\rangle}{\|\langle 0|y\rangle\|} \right\| \leq \frac{\epsilon}{2} \,. \tag{34}$$

We have from Eq. 19, 27 and Lemma 1 again

$$\left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - \frac{\langle 0|\bar{y}\rangle}{\|\langle 0|\bar{y}\rangle\|} \right\| \leq \frac{2\,\|\langle 0|\phi\rangle - \langle 0|\bar{y}\rangle\|}{\|\langle 0|\bar{y}\rangle\|} \leq 2\sqrt{66}\,\delta \,, \tag{35}$$

such that choosing

$$\delta = \frac{\epsilon}{4\sqrt{66}} \tag{36}$$

we have

$$\left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - \frac{\langle 0|\bar{y}\rangle}{\|\langle 0|\bar{y}\rangle\|} \right\| \leq \frac{\epsilon}{2} \tag{37}$$

and

$$\left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - |\bar{x}\rangle \right\| \leq \left\| \frac{\langle 0|\phi\rangle}{\|\langle 0|\phi\rangle\|} - \frac{\langle 0|\bar{y}\rangle}{\|\langle 0|\bar{y}\rangle\|} \right\| + \left\| \frac{\langle 0|y\rangle}{\|\langle 0|y\rangle\|} - |\bar{x}\rangle \right\| \leq \epsilon \tag{38}$$

as required.

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

$$\left\| \frac{v}{\|v\|} - \frac{w}{\|w\|} \right\| \leq 2 \frac{\|v - w\|}{\|v\|} . \tag{39}$$

Proof. We have

$$\left\| \frac{v}{\|v\|} - \frac{w}{\|w\|} \right\| \leq \frac{\|v - w\| + |\|w\| - \|v\||}{\|v\|} \leq 2 \frac{\|v - w\|}{\|v\|} . \tag{40}$$

□

References

1. J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, S. Lloyd, Quantum machine learning. *Nature* **549**, 195–202 (2017).

2. S. Lloyd, M. Mohseni, P. Rebentrost, Quantum principal component analysis. *Nature Physics* **10**, 631–633 (2014).

3. M. Schuld, I. Sinayskiy, F. Petruccione, Prediction by linear regression on a quantum computer. *Physical Review A* **94**, 022342 (2016).

4. P. Rebentrost, M. Mohseni, S. Lloyd, Quantum support vector machine for big data classification. *Physical review letters* **113**, 130503 (2014).

5. A. Montanaro, Quantum speedup of Monte Carlo methods. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **471**, 20150301 (2015).

6. D. W. Berry, A. M. Childs, A. Ostrander, G. Wang, Quantum algorithm for linear differential equations with exponentially improved dependence on precision. *Communications in Mathematical Physics* **356**, 1057–1081 (2017).

7. D. W. Berry, High-order quantum algorithm for solving linear differential equations. *Journal of Physics A: Mathematical and Theoretical* **47**, 105301 (2014).

8. A. W. Harrow, A. Hassidim, S. Lloyd, Quantum algorithm for linear systems of equations. *Physical review letters* **103**, 150502 (2009).

9. A. M. Childs, R. Kothari, R. D. Somma, Quantum linear systems algorithm with exponentially improved dependence on precision. *arXiv preprint arXiv:1511.02306* **83** (2015).

10. R. P. Feynman, Quantum mechanical computers. *Optics news* **11**, 11–20 (1985).

11. A. Y. Kitaev, A. Shen, M. N. Vyalyi, M. N. Vyalyi, *Classical and quantum computation* (American Mathematical Soc., 2002).

12. S. Lloyd, Robustness of adiabatic quantum computing. *arXiv preprint arXiv:0805.2757* (2008).

13. P. A. Gagniuc, *Markov chains: from theory to implementation and experimentation* (John Wiley & Sons, 2017).

14. R. Pastor-Satorras, C. Castellano, P. Van Mieghem, A. Vespignani, Epidemic processes in complex networks. *Reviews of modern physics* **87**, 925 (2015).

15. F. D. Sahneh, C. Scoglio, P. Van Mieghem, Generalized epidemic mean-field model for spreading processes over multilayer complex networks. *IEEE/ACM Transactions on Networking* **21**, 1609–1620 (2013).

16. W. J. Anderson, *Continuous-time Markov chains: An applications-oriented approach* (Springer Science & Business Media, 2012).

17. A. Kolmogoroff, Über die analytischen methoden in der wahrscheinlichkeitsrechnung. *Mathematische Annalen* **104**, 415–458 (1931).

18. E. Tang, Quantum-inspired classical algorithms for principal component analysis and supervised clustering. *arXiv preprint arXiv:1811.00414* (2018).

19. N.-H. Chia, A. Gilyén, T. Li, H.-H. Lin, E. Tang, C. Wang, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing* (2020), pp. 387–400.

20. M. Rudelson, R. Vershynin, Sampling from large matrices: An approach through geometric functional analysis. *Journal of the ACM (JACM)* **54**, 21–es (2007).

21. K.-M. Lau, H. Weng, Climate signal detection using wavelet transform: How to make a time series sing. *Bulletin of the American meteorological society* **76**, 2391–2402 (1995).

22. D. B. Percival, A. T. Walden, *Wavelet methods for time series analysis*, vol. 4 (Cambridge university press, 2000).

23. B. Cazelles, M. Chavez, G. C. d. Magny, J.-F. Guégan, S. Hales, Time-dependent spectral analysis of epidemiological time-series with wavelets. *Journal of the Royal Society Interface* **4**, 625–636 (2007).

24. A. Grinsted, J. C. Moore, S. Jevrejeva, Application of the cross wavelet transform and wavelet coherence to geophysical time series. *Nonlinear Processes in Geophysics* (2004).

25. P. Hoyer, Efficient quantum transforms. *arXiv preprint quant-ph/9702028* (1997).

26. A. Fijany, C. P. Williams, *NASA International Conference on Quantum Computing and Quantum Communications* (Springer, 1998), pp. 10–33.

27. A. Klappenecker, M. Rotteler, *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.

28. C. Shao, Computing eigenvalues of matrices in a quantum computer. *arXiv preprint arXiv:1912.08015* (2019).

29. Z. Li, X. Liu, N. Xu, J. Du, Experimental realization of a quantum support vector machine. *Physical review letters* **114**, 140504 (2015).

30. G. Wang, Quantum algorithm for linear regression. *Physical review A* **96**, 012335 (2017).

31. E. Farhi, H. Neven, Classification with quantum neural networks on near term processors. *arXiv preprint arXiv:1802.06002* (2018).

32. A. B. Grilo, I. Kerenidis, T. Zijlstra, Learning-with-errors problem is easy with quantum samples. *Physical Review A* **99**, 032314 (2019).

33. J. Romero, J. P. Olson, A. Aspuru-Guzik, Quantum autoencoders for efficient compression of quantum data. *Quantum Science and Technology* **2**, 045001 (2017).

34. A. Khoshaman, W. Vinci, B. Denis, E. Andriyash, H. Sadeghi, M. H. Amin, Quantum variational autoencoder. *Quantum Science and Technology* **4**, 014001 (2018).

35. M. Schuld, N. Killoran, Quantum machine learning in feature hilbert spaces. *Physical review letters* **122**, 040504 (2019).

36. V. Havlíček, A. D. Córcoles, K. Temme, A. W. Harrow, A. Kandala, J. M. Chow, J. M. Gambetta, Supervised learning with quantum-enhanced feature spaces. *Nature* **567**, 209–212 (2019).

37. S. Lloyd, M. Schuld, A. Ijaz, J. Izaac, N. Killoran, Quantum embeddings for machine learning. *arXiv preprint arXiv:2001.03622* (2020).

38. I. Cong, S. Choi, M. D. Lukin, Quantum convolutional neural networks. *Nature Physics* **15**, 1273–1278 (2019).

39. M. Schuld, A. Bocharov, K. M. Svore, N. Wiebe, Circuit-centric quantum classifiers. *Physical Review A* **101**, 032308 (2020).

40. P. Van Mieghem, E. Cator, Epidemics in networks with nodal self-infection and the epidemic threshold. *Physical Review E* **86**, 016116 (2012).

41. P. L. Simon, M. Taylor, I. Z. Kiss, Exact epidemic models on graphs using graph-automorphism driven lumping. *Journal of mathematical biology* **62**, 479–508 (2011).

42. L. Grover, T. Rudolph, Creating superpositions that correspond to efficiently integrable probability distributions. *arXiv preprint quant-ph/0208112* (2002).

Acknowledgments

This work was supported by ARO, DOE, AFOSR, and IARPA. The authors thank Peter Love and Clemens Wittenbecher for helpful discussions.

Supplementary Material

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

$$\langle Q \rangle = \sum_{k=1}^{N} x_j(k) Q(k) \tag{41}$$

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 [*7*] is to pad out the matrix A following the j'th row with $O(T)$ rows of the form

$$(0 \ldots 0 \ -I \ I \ 0 \ldots 0) \tag{42}$$

where the I term in each $-I \ I \ 0$ 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 [*42*]. 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 [*42*] then

allow us to construct the (unnormalized) state

$$|Q\rangle = \sum_{k=1}^{N} Q(k)|k\rangle \tag{43}$$

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 [*42*]. The normalized version of $|Q\rangle$ is then

$$|\tilde{Q}\rangle = Z_Q^{-1/2}|Q\rangle \tag{44}$$

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

$$\langle Q\rangle = \sum_{k=1}^{N} x_j(k)Q(k) = \langle Q|x_j\rangle = Z_j^{1/2} Z_Q^{1/2} \langle\tilde{Q}|\tilde{x}_j\rangle \tag{45}$$

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

$$O(n(t/h)\log q)\,. \tag{46}$$

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

$$O\left(\frac{n(t/h)\log\frac{1}{\delta}\log q}{\epsilon^2}\right)\,. \tag{47}$$

Quantum speedup of Monte Carlo sampling The quantum algorithm of Ref. [5] provides a quadratic improvement in the dependence of the complexity Eq. 47 of Monte Carlo sampling on the precision ϵ. More precisely, let us assume that $0 \leq Q(k) \leq 1$ for any $k = 1, \ldots, N$ (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. b46. 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 $O\left(\log \frac{1}{\delta} / \epsilon\right)$ 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

$$O\left(\frac{n(t/h) \log \frac{1}{\delta} \log q}{\epsilon}\right), \tag{48}$$

with the promised quadratic improvement in the ϵ-dependence with respect to Eq. 47.

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

$$\left(\mathbf{X}^\dagger \mathbf{X}\right)_{jj'} = \sum_k x_j(k)\, x_{j'}(k)\,, \tag{49}$$

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

$$O\left(\frac{1}{\left(\mathbf{X}^{\dagger}\mathbf{X}\right)_{jj'}}\right) \geq O\left(\frac{1}{\max_i \left\| |x_i\rangle \right\|^2}\right) \tag{50}$$

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

Additional figures and simulations

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).

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).

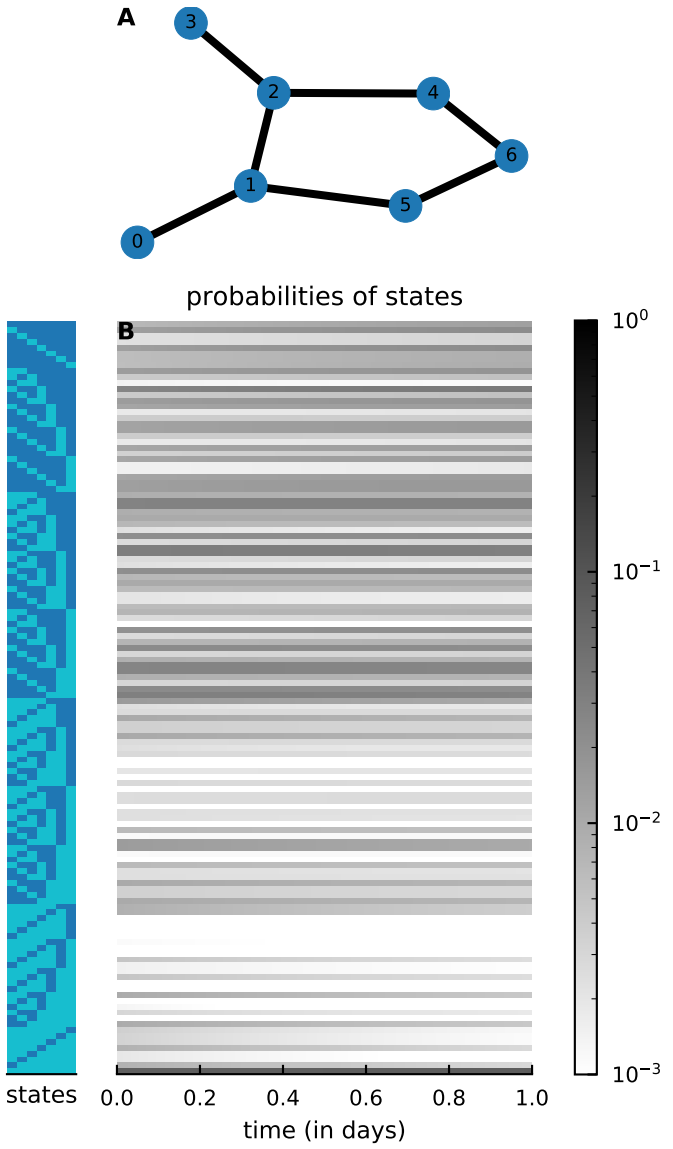

Figure 6: **(A)** The 7 node network used for simulation of continuous time Markov chain where transitions from susceptible to infected and vice-versa occur with rates $r_{SI} = 1.5$, reduced by a factor of five when four or more nodes are infected. **(B)** Probabilities of Markov states over time shown 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.

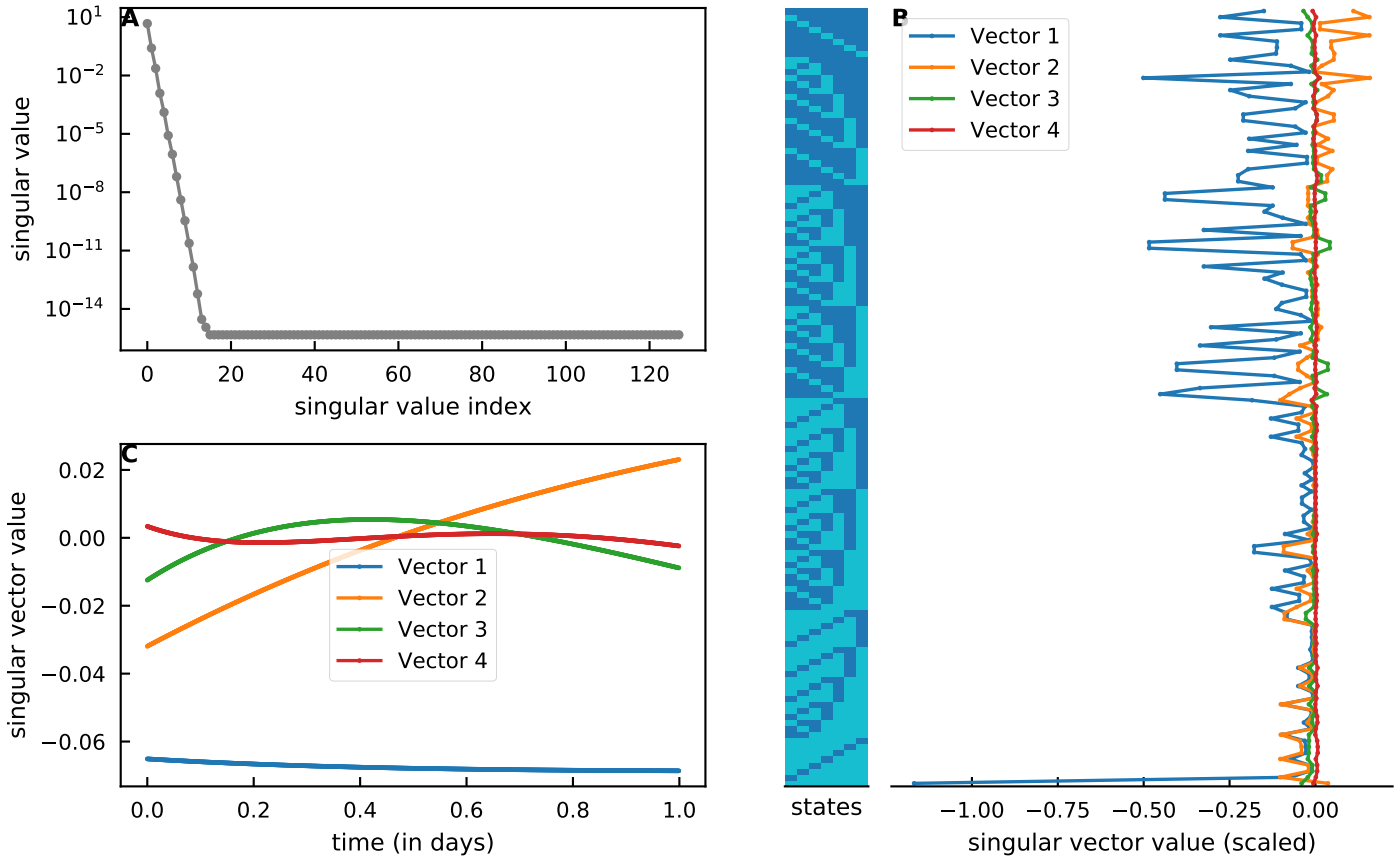

Figure 7: **(A)** Singular values of data matrix $\boldsymbol{X}$ decay exponentially fast. **(B)** First four left singular vectors scaled by the square root of their corresponding singular value show that states where four nodes are infected become prominent as this is the inflection point for social 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.

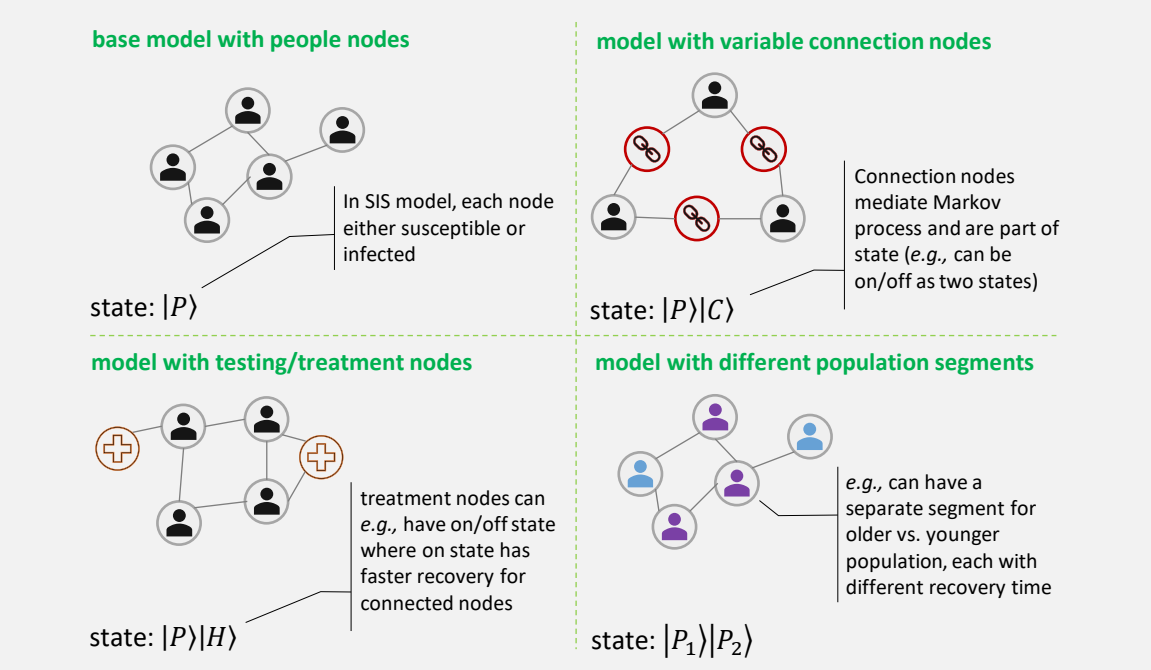

Figure 8: By customizing the properties of nodes in a network, Markov states can model various different phenomenon. Here, we show some of the options available to expand the functionality of a Markov state. In these cases, states are stored in a tensor product structure where information corresponding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

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.

Computational details

All simulations were performed in Python. Code is available upon request.