



HAL
open science

Temporal Gillespie algorithm: Fast simulation of contagion processes on time-varying networks

Christian L. Vestergaard, Mathieu Géniois

► **To cite this version:**

Christian L. Vestergaard, Mathieu Géniois. Temporal Gillespie algorithm: Fast simulation of contagion processes on time-varying networks. 2015. hal-01140134v1

HAL Id: hal-01140134

<https://hal.science/hal-01140134v1>

Preprint submitted on 9 Apr 2015 (v1), last revised 2 Nov 2015 (v2)

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Temporal Gillespie algorithm: Fast simulation of contagion processes on time-varying networks

Christian L. Vestergaard^{1,*} and Mathieu G enois¹

¹*Aix Marseille Universit e, Universit e de Toulon,
CNRS, CPT, UMR 7332, 13288 Marseille, France*

(Dated: April 7, 2015)

Stochastic simulations are one of the cornerstones of the analysis of dynamical processes on complex networks, and are often the only accessible way to explore their behavior. The development of fast algorithms is paramount to allow large-scale simulations. The Gillespie algorithm can be used for fast simulation of stochastic processes, and variants of it have been applied to simulate dynamical processes on static networks. However, its adaptation to temporal networks remains non-trivial. We here present a temporal Gillespie algorithm that solves this problem. Our method is applicable to general Poisson (constant-rate) processes on temporal networks, stochastically exact, and up to orders of magnitude faster than traditional simulation schemes based on rejection sampling. We also show how it can be extended to simulate non-Markovian processes. The algorithm is easily applicable in practice, and as an illustration we detail how to simulate both Poissonian and non-Markovian models of epidemic spreading. Namely, we provide pseudocode and its implementation in C++ for simulating the paradigmatic Susceptible-Infected-Susceptible and Susceptible-Infected-Recovered models and a Susceptible-Infected-Recovered model with non-constant recovery rates.

I. INTRODUCTION

Networks have emerged as a natural description of complex systems and their dynamics [1], notably in the case of spreading phenomena, such as social contagion, rumor and information spreading, or epidemics [1–3]. The dynamics of contagion processes occurring on a network are usually complex, and analytical results are attainable only in special cases [3, 4]. Furthermore, such results almost systematically involve approximations [3, 4]. Numerical studies based on stochastic simulations are therefore necessary, both to verify analytical approximations, and to study the majority of cases for which no analytical results exist. The development of fast algorithms is thus important for the characterization of contagion phenomena, and for large-scale applications such as simulations of world-wide epidemics [2, 5].

The Doob-Gillespie algorithm [6–11], originally proposed by David Kendall in 1950 for simulating birth-death processes and made popular by Daniel Gillespie in 1976 for the simulation of coupled chemical reactions, offers an elegant way to speed up such simulations by doing away with the many rejected trials of traditional Monte Carlo methods. Instead of checking at each time-step if each possible reaction takes place, as rejection-sampling algorithms do, the Gillespie algorithm draws directly the time elapsed until the next reaction takes place and what reaction takes place at that time. It is readily adapted to the simulation of Poisson processes on static networks [12] and can be generalized to non-Markovian processes [13].

Systems in which spreading processes take place, e.g., social, technological, infrastructural, or animal systems,

are not static though. Individuals create and break contacts at time-scales comparable to the time-scales of such processes [14–16], and the dynamics of social networks themselves profoundly affect dynamical processes taking place on top of them [17–23]. One thus needs to take the network’s dynamics into account, e.g., by representing it as a time-varying network [24] (also known as a time-varying graph, temporal network, or dynamical network). The dynamical nature of time-varying networks makes the adaptation of the Gillespie algorithm to such systems non-trivial.

The main difficulty in adapting the Gillespie algorithm to time-varying networks is taking into account the variation of the set of possible transitions at each time step. We show that by renormalizing time by the instantaneous cumulative transition rate, we can construct a temporal Gillespie algorithm that is applicable to Poisson (constant rate) processes on time-varying networks. We give pseudocode and C++ implementations for its application to simulate the paradigmatic Susceptible-Infected-Susceptible (SIS) and Susceptible-Infected-Recovered (SIR) models of epidemic spreading, for both homogeneous and heterogeneous [25] populations. We verify the accuracy of the temporal Gillespie algorithm numerically by comparison with a classical rejection sampling algorithm, and we show that it is up to orders of magnitude faster.

While Poissonian models are of widespread use, real contagion phenomena show memory effects, i.e., they are non-Markovian. Notably, for realistic infectious diseases, the rate at which an infected individual recovers is not constant over time [26, 27]. Social contagion may also show memory effects, e.g., one may be more (or less) prone to adopt an idea the more times one has been exposed to it. To treat this larger class of models, we show how the temporal Gillespie algorithm can be extended to non-Markovian processes. We give in particular an

* cvestergaard@gmail.com

algorithm for simulating SIR epidemic models with non-constant recovery rates.

The manuscript is organized as follows. Section II defines the stochastic processes which can be simulated using the temporal Gillespie algorithm, and describes the class of compartmental models for contagion phenomena on networks, the class we will use in examples throughout this paper. Section III gives a quick overview of the traditional rejection-sampling algorithms. Section IV outlines a derivation of the static Gillespie algorithm. Section V derives the temporal Gillespie algorithm for Poisson (constant-rate) processes. In Section VI we validate the temporal Gillespie algorithm through numerical comparison with a rejection-sampling algorithm; we also compare their speeds for simulating SIR and SIS processes on synthetic time-varying networks. Section VII shows how the temporal Gillespie algorithm can be extended to simulate non-Markovian processes; the approach is verified numerically and the speed of the non-Markovian temporal Gillespie algorithm is compared to rejection sampling. Section VIII concludes.

Four short appendices provide supporting information: Appendix A lists notation used in the article. Appendix B gives pseudocode for application of the temporal Gillespie algorithm to specific contagion processes on time-varying networks. Appendix C details how Monte Carlo simulations were performed. Finally, in Appendix D we apply the temporal Gillespie algorithm to empirical time-varying networks of face-to-face contacts, confirming the results obtained on synthetic networks; we also discuss an approach for further optimization of the algorithm for empirical networks.

II. STOCHASTIC PROCESSES ON TIME-VARYING NETWORKS

We define in this section the type of stochastic processes to which the temporal Gillespie algorithm can be applied. At the time of writing, the main domain of application of the algorithm is the class of compartmental models for contagion processes on time-varying networks, which we introduce below. For definiteness, algorithms detailing the application of the temporal Gillespie algorithm will concern this class of stochastic processes.

In general, we consider a system whose dynamics can be described by a set of stochastic transition events. We assume that the system can be modeled at any point in time by a set, $\Omega(t)$, of $M(t)$ independent stochastic processes m , which we term *transition processes*; the rate at which the transition m takes place is denoted λ_m . The set $\Omega(t)$ in general changes over time, and λ_m may or may not vary over time. For the classic “static” Gillespie algorithm to be applicable, $\Omega(t)$ can change only when a transition (or chemical reaction in the context of Gillespie’s original article) takes place. For processes taking place on time-varying networks, the medium of the process—the network—also changes with time. As these

changes may allow or forbid transitions, $\Omega(t)$ is not only modified by every reaction, but also by every change in the network. This is the domain of the temporal Gillespie algorithm, which only requires that the number of points in which $\Omega(t)$ changes be finite over a finite time-interval [28].

The assumption that the transition processes are independent is essential to the validity of the Gillespie algorithm, as it allows the calculation of the distribution of waiting times between consecutive transitions. This assumption is not overly restrictive, as it only requires a transition process to be independent of the evolution of the other simultaneous transition processes. A transition process may depend on all earlier transitions, and the current and past states of all nodes. As such, Gillespie algorithms can notably be applied to models of cooperative infections and other non-linear processes such as threshold models [13], and even the dynamics of ant battles [29].

A. Compartmental models of contagion

In a network based description of the population in which a contagion process takes place, an individual is modeled as a node i [Fig. 1(a)]. A contact between two individuals taking place at time t is represented by an edge $(i, j)_t$ in the graph describing the population at the instant t [Fig. 1(a)]. In a compartmental model, each node is in a certain state, which belongs to a fixed, finite set of q different states (compartments) [3]. A random variable $x_i(t) \in \{\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_q\}$ specifies the state of the node i at time t (i.e. to which compartment it belongs). Nodes may stochastically transition between states, governed by the set $\Omega(t)$ of transition processes. One is usually interested in the evolution of the number of nodes in each state, which we denote X_1, X_2, \dots, X_q .

As an example, we consider the classic SIR model of epidemic spreading with constant transition rates in a homogeneous population (rates are the same for all individuals) [Fig. 1(b)]. Here nodes can be in one of three states: susceptible, infected, and recovered, $\{\mathcal{S}, \mathcal{I}, \mathcal{R}\}$. Two different transition types let nodes change state: i) a node i in the \mathcal{S} state switches to the \mathcal{I} state with rate $k_{\mathcal{I}}(t)\beta$ (an $\mathcal{S} \rightarrow \mathcal{I}$ reaction), where $k_{\mathcal{I}}(t)$ is the number of infected neighbors i has at time t , i.e., the number of contacts i has with nodes in the \mathcal{I} state [Fig. 1(a)]; ii) a node i in the \mathcal{I} state switches to the \mathcal{R} state at rate μ (an $\mathcal{I} \rightarrow \mathcal{R}$ reaction).

In general, the transition processes can be divided into three types:

- a. spontaneous transitions, which only depend on the current state of the node, $x_i(t)$ [Fig. 1(c)]—e.g. an infected node recovers spontaneously in the SIR model [Fig. 1(b)];
- b. contact-dependent transitions, which may take place only when the node i is in contact with other

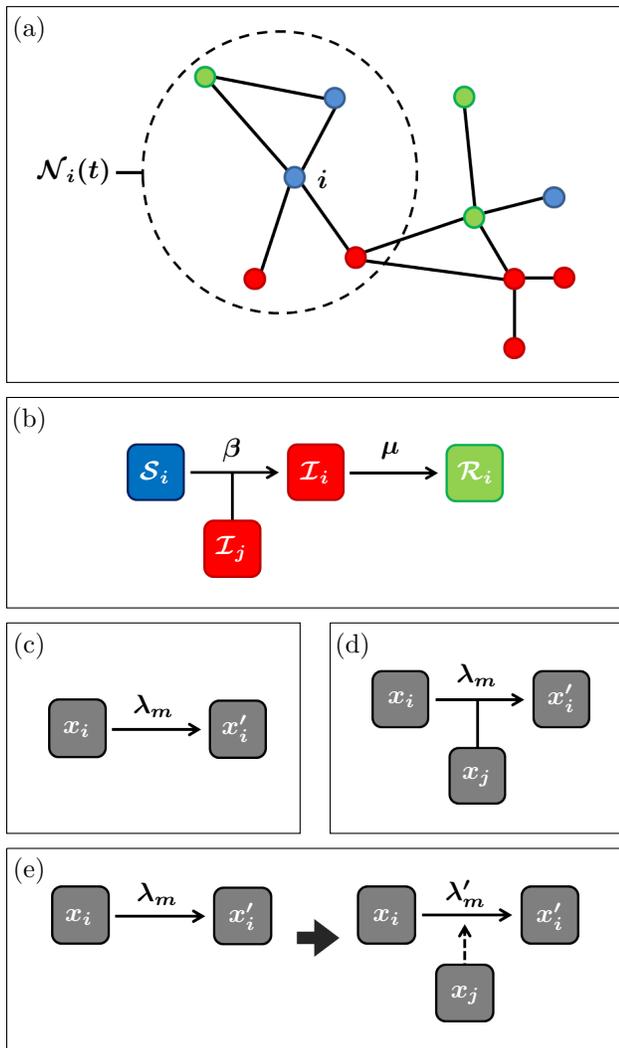


FIG. 1. **Schematic representation of a compartmental contagion process on a network.** (a) Snapshot of a contagion process on a sample network where the colors of nodes correspond to their current state; edges denote a current contact between nodes; $\mathcal{N}_i(t)$ denotes the set of nodes in contact with i at time t — i 's neighborhood. (b) Example: reaction types in the SIR model. (c) Spontaneous reaction: a node i may spontaneously transition from its current state x_i to x'_i with rate λ_m . (d) Contact-dependent reaction: a node i may transition from its current state x_i to x'_i with rate λ_m upon contact with the node j in state x_j . (e) Mixed transition: a node i may spontaneously transition from its current state x_i to another state, x'_i with rate λ_m ; contact with another node j , in state x_j , may alter the transition rate of m , $\lambda_m \rightarrow \lambda'_m$. After the contact $(i, j)_t$ ends, the transition rate may revert to λ_m or change to a third value.

nodes in a given state; it thus depends on the states of the node's current contacts, x_j for $j \in \mathcal{N}_i(t)$ (where $\mathcal{N}_i(t)$ is the neighborhood of i , i.e., the nodes j in contact with i at time t) [Fig. 1(d)]—e.g. a susceptible node may be infected upon contact with an infectious node in the SIR model

[Fig. 1(b)].

- c. mixed transitions, which take place spontaneously, but may depend on the node's past and current contacts [Fig. 1(e)]—e.g. in rumor spreading, an individual may learn on his own that a rumor is false (spontaneous) or may be convinced by another individual who knows the rumor is false (contact-dependent).

This division is important for practical application of the temporal Gillespie algorithm as transition processes of type a need only be updated after a transition has taken place, and processes of type c need only be updated whenever a relevant contact takes place, but not at each time-step. Using this distinction is crucial to obtaining large speed-increase that the temporal Gillespie algorithm offers over rejection sampling, as discussed below (Secs. V and VI).

III. REJECTION SAMPLING FOR MONTE CARLO SIMULATIONS

A straightforward way to simulate a stochastic process is to use a rejection sampling algorithm, akin to the classical Metropolis algorithm. Here one divides the time-axis in small time-steps Δt , where Δt should be chosen sufficiently small such that this discretization does not influence the outcome of the process significantly; on time-varying networks, the choice of Δt often comes naturally as the time-resolution at which the network is measured or simulated [Fig. 2(a)].

At each time-step $t = 0, \Delta t, 2\Delta t, \dots$, we test whether each possible transition $m \in \Omega(t)$ takes place or not. In practice this is done by drawing a random number r_m that is uniformly distributed on $[0, 1)$ for each m and comparing it to $\lambda_m \Delta t$: if $r_m < \lambda_m \Delta t$ the reaction takes place, if $r_m \geq \lambda_m \Delta t$ nothing happens [Figs. 2(b) and 2(c)]. (Note that one should technically compare r_m to $1 - \exp(-\lambda_m \Delta t)$ to ensure that λ_m defines a proper transition rate for finite Δt . However, the two procedures are equivalent in the limit $\Delta t \rightarrow 0$.)

From the design of the rejection sampling algorithm we see that the proportion of trials that are rejected is equal to a weighted average over $\{1 - \lambda_m \Delta t\}_m$. Thus, since we require $\lambda_m \Delta t \ll 1$ in order to avoid discretization errors, the vast majority of trials are rejected and the rejection sampling algorithm is computationally inefficient.

IV. GILLESPIE ALGORITHM ON STATIC NETWORKS

The Gillespie algorithm lets us perform stochastically exact Monte Carlo simulations without having to reject trials. For Poisson processes on static networks, it works by recognizing that the waiting time between two consecutive transitions is exponentially distributed, and that

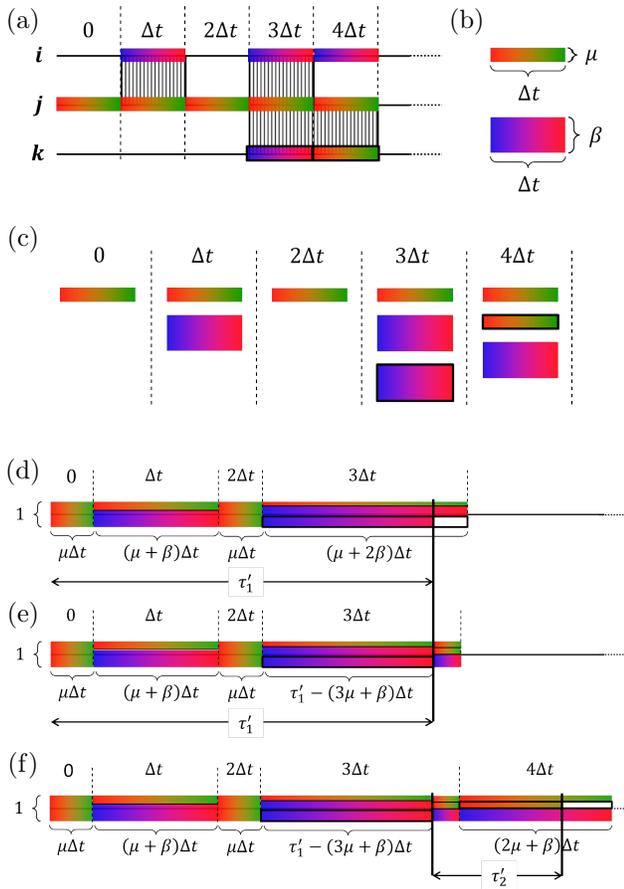


FIG. 2. **How the temporal Gillespie algorithm works for the example of an SIR process on a time-varying network.** (a) Representation of a simple time-varying network consisting of 3 nodes. Rastered grey areas between the timelines of two nodes indicate that the nodes are in contact during the given time-step. Colored rectangles on the timeline of a node indicate that the node may undergo a transition from one state [\mathcal{S} (blue), \mathcal{I} (red), \mathcal{R} (green)] to another during that time-step. Transitions that do take place are outlined in black. (b) The probability that an $\mathcal{S} \rightarrow \mathcal{I}$ or $\mathcal{I} \rightarrow \mathcal{R}$ transition takes place is given by the “area” $\beta\Delta t$ or $\mu\Delta t$, respectively. (c) Possible reactions during each time-step. A rejection sampling algorithm decides individually whether each transition takes place. (d) Representation of the possible transitions on the renormalized timeline. The renormalization can be thought of as an area-conserving transformation of the rectangles of (b) and (c), which represent the probability of each transition. (Note that the drawing is not to scale since in general $\lambda_m\Delta t \ll 1$.) A transition takes place when the integrated cumulative transition rate, $\mathbb{L}(t; t^*)$ [Eq. (7)], is equal to the renormalized waiting time, τ'_1 . (e) A transition that takes place immediately modifies the set of possible transitions which, in turn, modifies the cumulative transition rate Λ . The length of the remaining part of the time-step, measured in renormalized time, is modified accordingly. (f) A new renormalized waiting time τ'_2 is drawn and the process is repeated.

each transition happens with a probability that is pro-

portional to its rate.

Specifically, the (survival) probability that the transition m has not taken place after a time τ since the last transition event is

$$S_m(\tau) = e^{-\lambda_m\tau} . \quad (1)$$

Since each transition takes place independently, the probability that no event takes place during the interval τ since the last event is

$$S(\tau) = \prod_m S_m(\tau) = e^{-\Lambda\tau} , \quad (2)$$

where $\Lambda = \sum_{m=1}^M \lambda_m$ is the cumulative transition rate. The above result is obtained by using the fact that while Ω and M do depend on t , they only change when an event takes place and not in-between. They can thus be treated as constant for the purpose of calculating the waiting time between events. The distribution of the waiting times τ is then $p(\tau) = \Lambda e^{-\Lambda\tau}$, while the probability that reaction m is the next reaction that takes place and that it takes place after exactly time τ is equal to $p_m(\tau) = \lambda_m e^{-\Lambda\tau}$.

The static Gillespie algorithm thus consists in drawing the waiting time $\tau \sim \text{Exp}(\Lambda)$ until the next transition and then drawing which transition m takes place with probability $\pi_m = \lambda_m/\Lambda$. (Here $\tau \sim \text{Exp}(\Lambda)$ is short for: τ is exponentially distributed with mean $1/\Lambda$.)

V. TEMPORAL GILLESPIE ALGORITHM

For processes taking place on time-varying networks however, $\Omega(t)$ changes with time independently of the transition events, e.g., for the case of an SIR process nodes may become infected only when in contact with an infected individual [Figs. 2(a) and 2(c)]. This means that the survival probability does not reduce to a simple exponential as in Eq. (1); it is instead given by

$$S_m(\tau) = \exp\left(-\int_{t^*}^{t^{**}} I_m(t)\lambda_m dt\right) , \quad (3)$$

where t^* is the time at which the last transition took place, $t^{**} = t^* + \tau$ is the time when the next transition takes place, and $I_m(t)$ is an indicator function that is equal to one when the process m may take place, e.g., when two given nodes are in contact, and zero when m may not take place. The meaning of I_m is exemplified in Fig. 2(a): the node i may transition from the \mathcal{S} to the \mathcal{I} state when in contact with the infected node j ; if we let m denote this transition process, $I_m(t)$ is then one for $t = \Delta t, 3\Delta t, 4\Delta t$ and zero for $t = 0, 2\Delta t$.

Using, as in the previous section, that transition processes are independent, we can write the probability that no event takes place during an interval τ (the waiting

time survival function):

$$S(\tau) = \prod_{m \in \Omega} S_m(\tau) = \exp \left(- \sum_{m \in \Omega} \int_{t^*}^{t^{**}} I_m(t) \lambda_m dt \right), \quad (4)$$

where Ω is the set of all transition processes on the interval $(t^*, t^{**}]$, i.e., Ω is the union over $\Omega(t)$ for $t \in (t^*, t^{**}]$. We finally use that $\sum_{m \in \Omega} I_m(t) f_m(t) = \sum_{m \in \Omega(t)} f_m(t)$, for any $f_m(t)$, to write

$$S(\tau) = \exp \left(- \int_{t^*}^{t^{**}} \Lambda(t) dt \right), \quad (5)$$

where

$$\Lambda(t) = \sum_{m \in \Omega(t)} \lambda_m \quad (6)$$

is the cumulative transition rate at time t . The dynamics of empirical time-varying networks is highly intermittent and we cannot describe $\Omega(t)$ analytically. This means that we cannot perform the integral of Eq. (5) directly. We may instead renormalize time by the instantaneous cumulative transition rate, $\Lambda(t)$: We define the renormalized waiting time between two consecutive transitions, τ' , as

$$\tau' = \mathbb{L}(t^{**}; t^*) = \int_{t^*}^{t^{**}} \Lambda(t) dt, \quad (7)$$

i.e., equal to the cumulative transition rate integrated over $(t^*, t^{**}]$. The survival function of τ' has the following simple form:

$$S(\tau') = \exp(-\tau'). \quad (8)$$

The time t^{**} when a new transition takes place is given implicitly by $\mathbb{L}(t^{**}; t^*) = \tau'$, while the probability that m is the transition that takes place at time $t = t^{**}$ is, as above, given by:

$$\pi_m = \lambda_m / \Lambda(t). \quad (9)$$

This lets us define a Gillespie-type algorithm for time-varying networks by first drawing a renormalized waiting time until the next event $\tau' \sim \text{Exp}(1)$. Second, we may in theory compare $\mathbb{L}(t; t^*)$ numerically to τ' for each t and let a transition take place when $\mathbb{L}(t; t^*) = \tau'$ [Fig. 2(d)]. In practice, since $\Lambda(t)$ only changes when a transition takes place or at $t_n = n\Delta t$ with $n \in \mathbb{N}$, we may instead compare τ' to

$$\mathbb{L}[(n+1)\Delta t; t^*] = [(n^*+1)\Delta t - t^*] \Lambda(t^*) + \Delta t \sum_{i=n^*+1}^n \Lambda(i\Delta t). \quad (10)$$

Here n^* is the time-step during which the last transition took place, and $\Lambda(t^*)$ is the cumulative transition rate at

t^* , immediately after the last transition has taken place. The first term of Eq. (10) is the cumulative transition rate integrated over the remainder of the n^* th time-step left after the last transition; the second term is equal to $\mathbb{L}[(n+1)\Delta t; (n^*+1)\Delta t]$. A new transition takes place during the time-step n^{**} where $\mathbb{L}[(n^{**}+1)\Delta t; t^*] \geq \tau'$ [Fig. 2(d)]; the precise time of this new transition is

$$t^{**} = n^{**} \Delta t + \frac{\tau' - \mathbb{L}(n^{**} \Delta t; t^*)}{\Lambda(n^{**} \Delta t)}, \quad (11)$$

while the reaction m that takes place is drawn with probability given by Eq. (9). We then update Ω and Λ as $\Omega(t^{**})$ and $\Lambda(t^{**})$ [Figs. 2(e)], draw a new waiting time, $\tau' \sim \text{Exp}(1)$, and reiterate the above procedure [Fig. 2(f)].

The algorithm can be implemented for contagion processes on time-varying networks as follows (see Appendix B for pseudocode for specific contagion models and Supplemental Files for implementation in C++) [30]:

1. Draw a renormalized waiting time until the first event, $\tau' \sim \text{Exp}(1)$.
2. At each time-step $t_n = n\Delta t$, with $n = 0, 1, 2, \dots$, let $\Omega \equiv \Omega(t_n)$ and $\Lambda \equiv \Lambda(t_n)$; here, only contact-dependent processes (type *b*, Sec. II A) and mixed (type *c*, Sec. II A) processes that depend on contacts taking place at t_n or t_{n-1} need to be updated—an important point, as it lets the temporal Gillespie algorithm be orders of magnitude faster than rejection sampling (see discussion in Sec. VI). Then, compare τ' to $\Lambda\Delta t$:

if $\Lambda\Delta t \leq \tau'$: Subtract $\Lambda\Delta t$ from τ' , continue to next time-step and repeat 2.

if $\Lambda\Delta t > \tau'$: Let the reaction m take place, chosen from Ω with probability $\pi_m = \lambda_m/\Lambda$. The fraction that is left of the time-step when the transition takes place is $\xi = 1 - \tau'/(\Lambda\Delta t)$ and the precise time of the transition is $t^{**} = t_n + \tau'/\Lambda$. Next, update Ω and Λ ; this time all transition processes should be updated, as spontaneous processes (type *a*, Sec. II A) may change, emerge, or disappear when a transition takes place. Then:

- (a) draw a new renormalized waiting time, $\tau' \sim \text{Exp}(1)$;
- (b) compare τ' to $\xi\Lambda\Delta t$:
 - if** $\tau' \geq \xi\Lambda\Delta t$: subtract $\xi\Lambda\Delta t$ from τ' , continue to next time-step and repeat 2.
 - if** $\tau' < \xi\Lambda\Delta t$: Another transition takes place during the present time-step (at time $t^{***} = t^{**} + \tau'/\Lambda$, where t^{**} is the time of the last transition during the same time-step): choose m from Ω with probability $\pi_m = \lambda_m/\Lambda$; let $\xi \rightarrow \xi - \tau'/\Lambda\Delta t$, and update Ω and Λ . Repeat a) and b).

By construction, the above procedure produces realizations of a stochastic process for which the waiting times for each transition follow exactly their correct distributions. The temporal Gillespie algorithm is thus what we term *stochastically exact*: all distributions and moments of a stochastic process evolving on a time-varying network obtained through Monte Carlo simulations using the algorithm converge to their exact values. Rejection based sampling algorithms are stochastically exact only in the limit $\lambda_m \Delta t \rightarrow 0$.

VI. COMPARISON OF GILLESPIE AND REJECTION SAMPLING ALGORITHMS

A. Numerical validation

We compare the outcome of SIR and SIS processes on activity-driven time-varying networks [31] simulated using the temporal Gillespie algorithm to simulations using traditional rejection sampling. For sufficiently small $\lambda_m \Delta t$, the outcomes are indistinguishable (Fig. 3), confirming the validity of the temporal Gillespie algorithm. Note that rejection sampling is only expected to be accurate for $\lambda_m \Delta t \ll 1$, while the temporal Gillespie algorithm is stochastically exact for all $\lambda_m \Delta t$; the results of the two algorithms thus differ when the assumption $\lambda_m \Delta t \ll 1$ does not hold (Supplemental Fig. 1) [32].

B. Comparison of simulation speed

Next, we compare the speeds of the temporal Gillespie and the rejection sampling algorithms for SIR and SIS processes (see Appendix C for details on how simulations were performed). Figures 4 and 5 show that the temporal Gillespie algorithm is up to multiple orders of magnitude faster than traditional rejection sampling. These results are confirmed by simulations on empirical time-varying networks (Appendix D). The speed gain is higher for larger systems (compare $N = 1\,000$ to $N = 100$ in Figs. 4 and 5) We also see that the speed gain is larger the sparser the network is. This is because the calculation of the contacts between susceptible and infected nodes at each time-step, necessary to determine the possible $\mathcal{S} \rightarrow \mathcal{I}$ transitions, is the performance limiting step of the temporal Gillespie algorithm. In the extreme case of a contagion model where all transitions are contact-dependent (type *b*, Sec. II A), such as the classic Maki-Thompson model of rumor spreading, the temporal Gillespie algorithm is approximately a factor two faster than the rejection sampling algorithm.

VII. NON-MARKOVIAN PROCESSES

For real-world contagion processes, transition rates are typically not constant but in general depend on the his-

tory of the process [26, 27]. Such processes are termed *non-Markovian*. The survival probability for a single non-Markovian transition process taking place on a time-varying network is given by:

$$S_m(\tau; \mathcal{F}_t^{(m)}) = \exp\left(-\int_{t^*}^{t^{**}} I_m(t) \lambda_m(t; \mathcal{F}_t^{(m)}) dt\right). \quad (12)$$

Here $\mathcal{F}_t^{(m)}$ is a filtration for the process m , i.e., all information relevant to the transition process available up to and including time t ; typically, $\mathcal{F}_t^{(m)}$ will be its starting time and relevant contacts that have taken place since. As in Sec. V, t^* is the time of the last transition and $t^{**} = t^* + \tau$ is the time of the next.

We use again that the transition processes are independent, to write the waiting time survival probability:

$$S(\tau; \mathcal{F}_t) = \exp\left(-\int_{t^*}^{t^{**}} \Lambda(t; \mathcal{F}_t) dt\right), \quad (13)$$

with

$$\Lambda(t; \mathcal{F}_t) = \sum_{m \in \Omega(t)} \lambda_m(t; \mathcal{F}_t^{(m)}), \quad (14)$$

and where \mathcal{F}_t is the union over $\mathcal{F}_t^{(m)}$ for $m \in \Omega$.

For a static network Eq. (13) reduces to the result found in [13, Eq. (7)]. This can be seen by noting that $M(t) = M$ and $\Omega(t) = \Omega$ are then constant, and that $\lambda_m(t; \mathcal{F}_t^{(m)}) = -[dS_m(t; \mathcal{F}_t^{(m)})/dt]/S_m(t; \mathcal{F}_t^{(m)}) = d\{\ln[1/S_m(t; \mathcal{F}_t^{(m)})]\}/dt$ and $S_m(t; \mathcal{F}_t^{(m)}) = S_m(t; \mathcal{F}_t^{(m)})/S_m(t_m; \mathcal{F}_t^{(m)})$.

As in the Poissonian case (Sec. V) we define the renormalized waiting time, τ' , as

$$\tau' = \mathbb{L}(t^{**}; t^*, \mathcal{F}_t) = \int_{t^*}^{t^{**}} \Lambda(t; \mathcal{F}_t) dt. \quad (15)$$

This gives us the same simple forms as above for the survival function of the renormalized waiting time, τ' ,

$$S(\tau') = \exp(-\tau'), \quad (16)$$

and the probability that m is the transition that takes place at $t = t^{**}$,

$$\pi_m(t; \mathcal{F}_t) = \frac{\lambda_m(t; \mathcal{F}_t^{(m)})}{\Lambda(t; \mathcal{F}_t)}. \quad (17)$$

Until now our approach and results are entirely equivalent to the Poissonian case of Section V. However, since $\lambda_m(t)$ in general depend continuously on time, the transition time t^{**} is not simply found by linear interpolation as in Eq. (11). Instead, one would need to solve the implicit equation $\mathbb{L}(t^{**}; t^*) = \tau'$ numerically to find t^{**} exactly. To keep things simple and speed up calculations, we may approximate $\Lambda(t)$ as constant over a time-step. This assumes that $\Delta\Lambda(t)\Delta t \ll 1$, where $\Delta\Lambda(t)$ is the change of

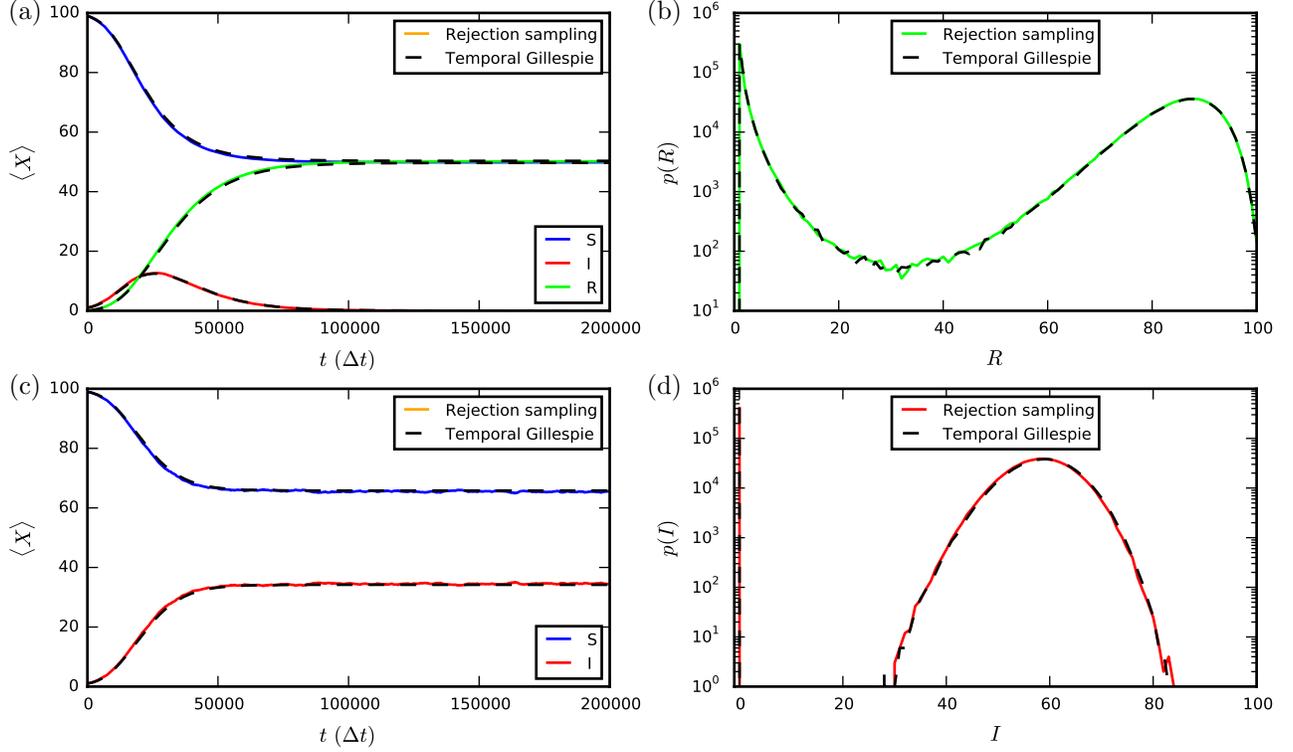


FIG. 3. **Comparison of numerical results from temporal Gillespie and rejection sampling algorithms.** (a) Mean number of nodes in each state of the SIR model as function of time. (b) Distribution of final epidemic size (number of recovered nodes when $I = 0$) in the SIR model. (c) Mean number of nodes in each state of the SIS model as function of time. (d) Distribution of the number of infected nodes in the stationary state ($t \rightarrow \infty$) of the SIS model. All simulations were performed 1 000 000 times with the root node chosen at random on an activity driven network [31] consisting of $N = 100$ nodes, with activities $a_i = \eta z_i$, where $\eta = 0.1$ and $z_i \sim z_i^{-3.2}$ for $z_i \in [0.03, 1)$, and a node formed two contacts when active. Parameters of the epidemic processes were $\beta\Delta t = 10^{-2}$ and $\mu\Delta t = 10^{-4}$.

$\Lambda(t)$ during a single time-step. It is a more lenient assumption than the assumption that $\Lambda(t)\Delta t \ll 1$ which rejection sampling relies on, as can be seen by noting that in general $\Delta\Lambda(t)/\Lambda(t) \ll 1$. The same assumption also lets us calculate $\mathbb{L}[(n+1)\Delta t; t^*]$ as in the Poissonian case:

$$\mathbb{L}[(n+1)\Delta t; t^*, \mathcal{F}_t] = [(n^*+1)\Delta t - t^*] + \Delta t \sum_{i=n^*+1}^n \Lambda(i\Delta t; \mathcal{F}_t) \quad (18)$$

and the time, t^{**} , at which the next transition takes place:

$$t^{**} = n^{**}\Delta t + \frac{\tau' - \mathbb{L}(n^{**}\Delta t; t^*, \mathcal{F}_t)}{\Lambda(n^{**}\Delta t; \mathcal{F}_t)}. \quad (19)$$

Using the above equations, we can now construct a temporal Gillespie algorithm for non-Markovian processes.

This algorithm updates all $\lambda_m(t)$ that depend on time at each time-step, where for the Poissonian case we only had to initialize new processes, i.e., contact-dependent processes (type b and c , Sec. II A). This means the algorithm is less than a factor two faster than rejection sampling [compare dotted red/blue lines (Temporal Gillespie,

$\epsilon = 0$) to full black (Rejection Sampling) lines in Fig. 6]. To speed up the algorithm, we may employ a first-order cumulant expansion of $S(\tau; \mathcal{F}_t)$ around $\tau = 0$, as proposed in [13] for a static non-Markovian Gillespie algorithm. It consists in approximating $\lambda_m(t; \mathcal{F}_t^{(m)})$ by the constant $\lambda_m(t^*; \mathcal{F}_t^{(m)})$ for $t^* < t < t^{**}$ and gives a considerable speed increase of the algorithm [full red/blue lines (Temporal Gillespie, $\epsilon \rightarrow \infty$) in Fig. 6]. However, the approximation is only valid when $M(t) \gg 1$, which is not always the case for contagion processes. Notably, at the beginning and end of an SIR process, and near the epidemic threshold for an SIS process, M is small and the approximation breaks down; the approximate algorithm for example overestimates the peak number of infected nodes in a SIR process with recovery rates that increase over time [compare full black line ($\epsilon \rightarrow \infty$) to the quasi-exact full red line ($\epsilon = 0$) in Fig. 7(a)]. An intermediate approach, which works when the number of transition processes is small, but is not too slow to be of practical relevance, is needed. We propose one such approach below.

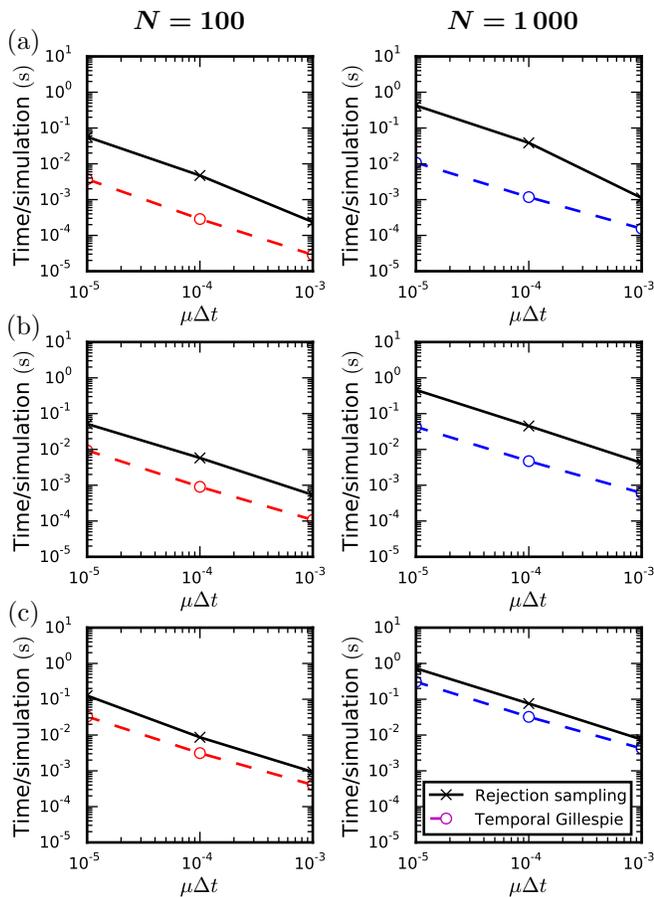


FIG. 4. Comparison of the speed of the temporal Gillespie and the rejection sampling algorithms: **SIR process**. Time per realization of a single SIR process on an activity driven network: (a) for an average instantaneous degree of the network of $\overline{k(t)} = 0.002$, (b) for $\overline{k(t)} = 0.02$, and (c) for $\overline{k(t)} = 0.2$ [for empirical contact networks, $\overline{k(t)} \approx 0.002$ – 0.03 (Appendix D)]. Networks consisted of $N = 100$ (left) or $N = 1000$ (right) nodes, with activities $a_i = \eta z_i$ and $z_i \sim z_i^{-3.2}$ for $z_i \in [0.03, 1)$; a node formed two contacts each time it was active. Simulations were run for 20 000, 200 000, or 2 000 000 time-steps for $\mu\Delta t$ equal to 10^{-3} , 10^{-4} or 10^{-5} , respectively, or until $I = 0$; for $\Delta t = 20$ s (as for the empirical data treated in Appendix D), $\mu\Delta t \approx 3 \cdot 10^{-5}$ corresponds to a recovery time of roughly one week, typical of flu-like diseases. The infection rate was $\beta = 10^3\mu$ for networks with $\overline{k(t)} = 0.002$, $\beta = 10^2\mu$ for networks with $\overline{k(t)} = 0.02$, and $\beta = 10\mu$ for networks with $\overline{k(t)} = 0.2$.

A. Efficient non-Markovian temporal Gillespie algorithm

As discussed above, we neither want to update all transition rates at each time-step as this makes the temporal Gillespie algorithm slow, nor do we want to only update them when a transition event takes place as this makes the algorithm inaccurate.

An intermediate approach is found by looking at the

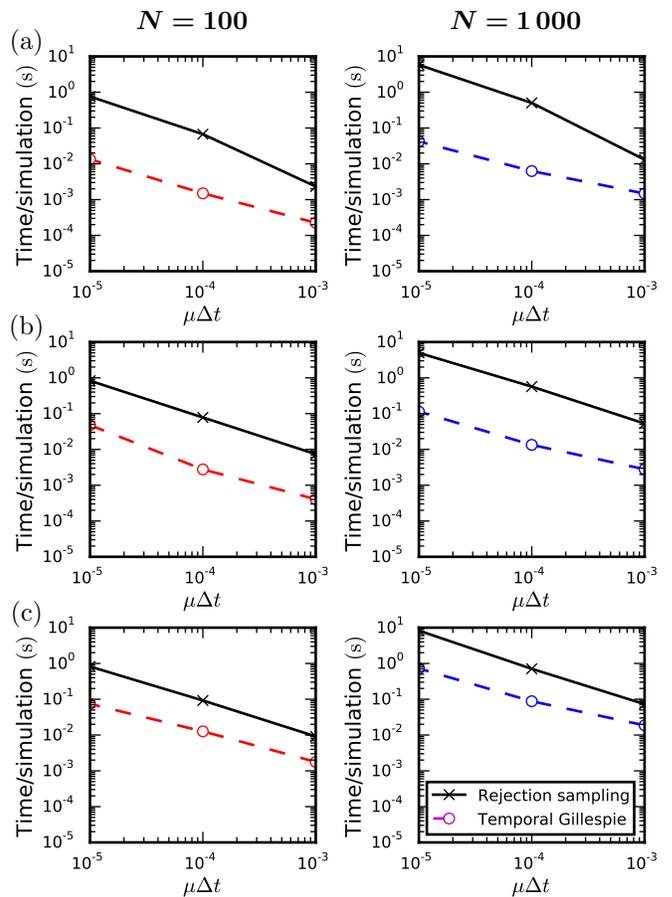


FIG. 5. Comparison of the speed of the temporal Gillespie and the rejection sampling algorithms: **SIS process**. Time per realization of a single SIS process on an activity driven network: (a) for an average instantaneous degree of the network of $\overline{k(t)} = 0.002$, (b) for $\overline{k(t)} = 0.02$, and (c) for $\overline{k(t)} = 0.2$ [for empirical contact networks, $\overline{k(t)} \approx 0.002$ – 0.03 (Appendix D)]. Networks consisted of $N = 100$ (left) or $N = 1000$ (right) nodes, with activities $a_i = \eta z_i$ and $z_i \sim z_i^{-3.2}$ for $z_i \in [0.03, 1)$; a node formed two contacts each time it was active. Simulations were run for 20 000, 200 000, or 2 000 000 time-steps for $\mu\Delta t$ equal to 10^{-3} , 10^{-4} or 10^{-5} , respectively, or until $I = 0$; for $\Delta t = 20$ s (as for the empirical data treated in Appendix D), $\mu\Delta t \approx 3 \cdot 10^{-5}$ corresponds to a recovery time of roughly one week, typical of flu-like diseases. The infection rate was $\beta = 10^3\mu$ for networks with $\overline{k(t)} = 0.002$, $\beta = 10^2\mu$ for networks with $\overline{k(t)} = 0.02$, and $\beta = 10\mu$ for networks with $\overline{k(t)} = 0.2$.

relevant physical time-scales of the transition processes: the average waiting time before they take place, $\langle\tau^{(m)}\rangle$. If the time elapsed since we last updated $\lambda_m(t)$ is small compared to $\langle\tau^{(m)}\rangle$ we do not make a large error by treating it as constant over the interval; however, if the elapsed time is comparable to or larger than $\langle\tau^{(m)}\rangle$, the error may be considerable. Thus, instead of updating λ_m at each time-step, we may update it only after a time $t > \epsilon\langle\tau^{(m)}\rangle$ has elapsed since it was last updated. Here ϵ controls the

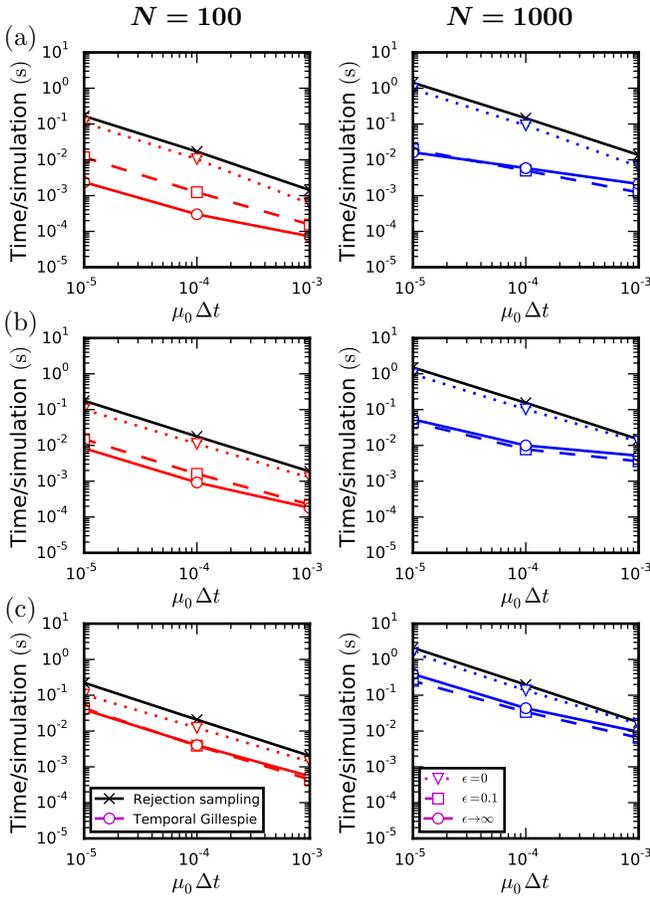


FIG. 6. Comparison of the speed of the temporal Gillespie and the rejection sampling algorithms: non-Markovian SIR process. Time per realization of a single simulation of an SIR process with Weibull distributed recovery times (Sec. VII A) on an activity driven network: (a) for an average instantaneous degree of the network of $\overline{k(t)} = 0.002$, (b) for $\overline{k(t)} = 0.02$, and (c) for $\overline{k(t)} = 0.2$ [for empirical contact networks, $\overline{k(t)} \approx 0.002\text{--}0.03$ (Appendix D)]. The parameter ϵ controls the accuracy of the temporal Gillespie algorithm: for $\epsilon = 0$, approximating $\lambda_m(t)$ as constant over a single time-step, it is most accurate; for $\epsilon \rightarrow \infty$, approximating the λ_m as constant between two consecutive transition events, it is the least accurate. Networks consisted of $N = 100$ (left) or $N = 1000$ (right) nodes, with activities $a_i = \eta z_i$ and $z_i \sim z_i^{-3.2}$ for $z_i \in [0.03, 1)$; a node formed two contacts each time it was active. The recovery rate of an infected node was given by Eq. (20) with $k = 1.5$. Simulations were run for 20 000, 200 000, or 2 000 000 time-steps for $\mu_0 \Delta t$ equal to 10^{-3} , 10^{-4} or 10^{-5} , respectively, or until $I = 0$. The infection rate was $\beta = 10^3 \mu_0$ for networks with $\overline{k(t)} = 0.002$, $\beta = 10^2 \mu_0$ for networks with $\overline{k(t)} = 0.02$, and $\beta = 10 \mu_0$ for networks with $\overline{k(t)} = 0.20$.

precision of the algorithm.

We use below this approach to simulate a non-Markovian SIR process, where the recovery times of infected nodes follows a Weibull distribution (see Appendix B for an algorithm written in pseudocode and

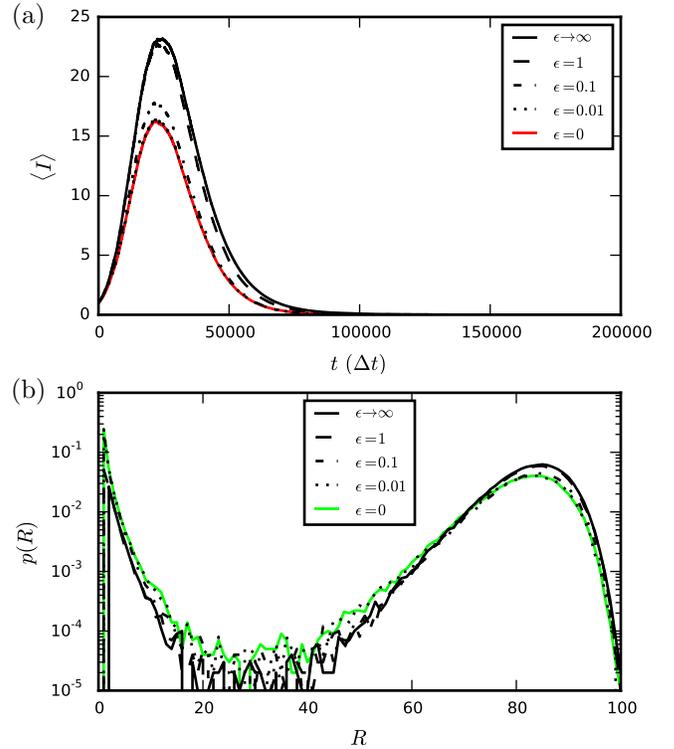


FIG. 7. Comparison of the outcome of non-Markovian SIR processes for different values of the parameter ϵ . (a) Average number of infected nodes, $\langle I \rangle$, as function of time for an SIR process with Weibull distributed recovery times (Sec. VII A). (b) Distribution of the numbers of recovered nodes after the infection has died out (i.e. when $I = 0$). For $\epsilon = 0$ the temporal Gillespie algorithm is quasi-exact (see Supplemental Fig. 2 for comparison with rejection sampling) [32]; for $\epsilon \rightarrow \infty$, corresponding to a first-order cumulant expansion of $\Lambda(t; \mathcal{F}_t)$ around $t = t^*$ (see main text), it is least accurate. As ϵ is decreased, both $\langle I \rangle(t)$ and $p(R)$ rapidly approach the quasi-exact result obtained with $\epsilon = 0$. Simulations were performed on an activity-driven network consisting of $N = 100$ nodes with activities $a_i = z_i/10$, where $z_i \sim z_i^{-3.2}$; nodes' recovery times followed Eq. (20) with $k = 1.5$ and $\mu_0 = 10^{-4}$ Hz; the length of a time-step was $\Delta t = 1$ s and the infection rate $\beta = 100 \mu_0 = 10^{-2}$ Hz.

Supplemental Files for implementation in C++ [30]. The recovery rate of an infected node is here given by

$$\mu(t; t^{(m)}) = k \mu_0^k (t - t^{(m)})^{k-1}, \quad (20)$$

where μ_0 sets the scale, $t^{(m)}$ is time when the node was infected, and k is a shape parameter of the distribution. For $k = 1$, we recover the constant-rate Poissonian case with $\mu(t; t^{(m)}) = \mu_0$. For realistic modeling of infections, $k > 1$; here $\mu(t; t^{(m)})$ is zero at $t = t^{(m)}$ and grows with time. In this case, we thus update the recovery rates $\mu(t; t^{(m)})$ whenever the time elapsed since a transition last took place exceeds $\langle \tau^{(m)} \rangle = \Gamma(1 + 1/k)/\mu_0$.

The parameter ϵ lets us control the precision of the

non-Markovian temporal Gillespie algorithm: the smaller ϵ is, the more precise the algorithm is, on the other hand, the larger ϵ is, the faster the algorithm is (Fig 8). At $\epsilon = 0$, the temporal Gillespie algorithm is maximally accurate, but also slowest, corresponding to the quasi-exact approximation that $\Lambda(t; \mathcal{F}_t)$ stays constant over a single time-step. Letting $\epsilon \rightarrow \infty$ corresponds to the first order cumulant expansion of [13], and is the fastest, but least accurate. Intermediate ϵ gives intermediate accuracy and speed, and permits one to obtain the desired accuracy without sacrificing performance. In the case of the SIR process with Weibull-distributed recovery times, $\epsilon = 0.1$ gives an error of no more than a few percent [Figs. 8(a)–8(d) and 7]—which is usually acceptable as the discrepancy between model and reality can be expected to be larger—with an almost optimal computation time [Figs. 8(e) and 6].

VIII. CONCLUSION

We have presented a fast temporal Gillespie algorithm for simulating stochastic processes on time-varying networks. The temporal Gillespie algorithm is up to multiple orders of magnitude faster than current algorithms for simulating stochastic processes on time-varying networks. For Poisson (constant-rate) processes, where it is stochastically exact, its application is particularly simple. The algorithm is also applicable to non-Markovian processes, where a control parameter lets one choose the desired accuracy and performance in terms of simulation speed. We have shown how to apply it to compartmental models of contagion in human contact networks. The scope of the temporal Gillespie algorithm is more general than this, however, and it may be applied e.g. to diffusion-like processes or systems for which a network description is not appropriate.

ACKNOWLEDGEMENT

The authors thank Alain Barrat for helpful discussions and critical reading of the manuscript and Thomas L. Vestergaard for help with debugging and code review. The authors also thank the SocioPatterns collaboration for privileged access to data sets. C.L.V. is supported by the EU FET project Multiplex 317532 and M.G. by the French ANR project HarMS-flu (ANR-12-MONU-0018).

Appendix A: Notation

Tables I and II list the notation used in the manuscript. Table I gives notation pertaining to the temporal Gillespie algorithm, and Table II lists notation pertaining to time-varying networks and compartmental contagion processes.

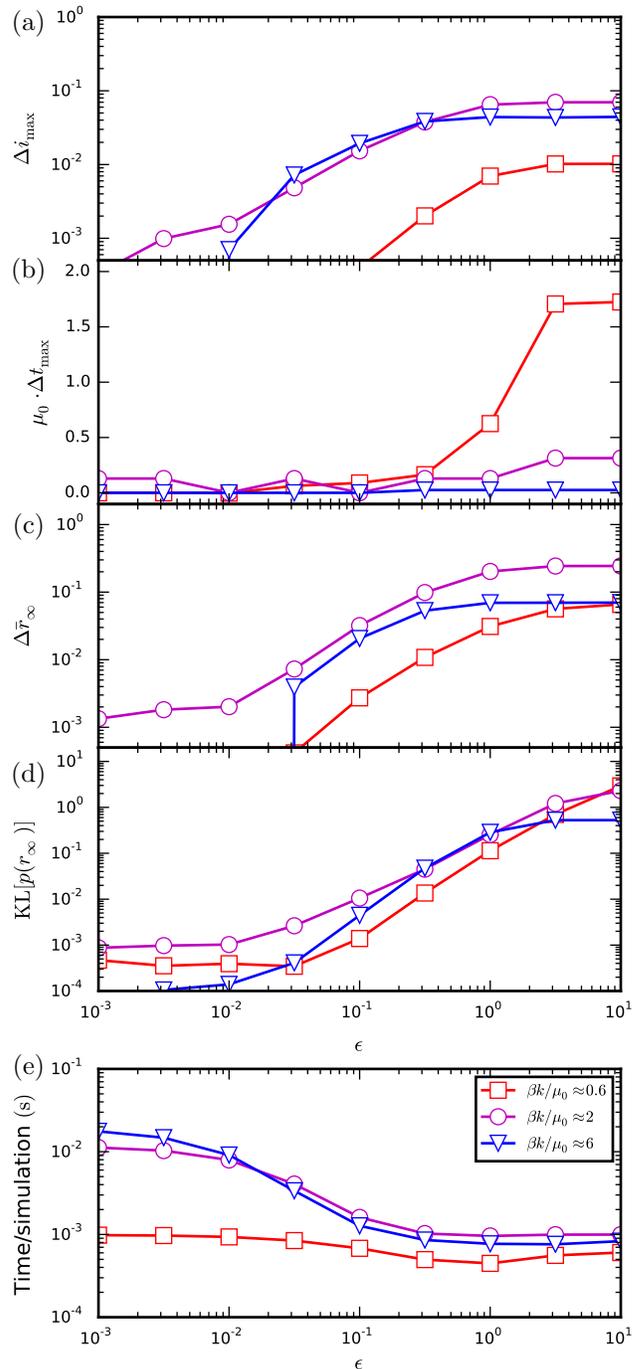


FIG. 8. **Accuracy and speed of the non-Markovian temporal Gillespie algorithm as function of ϵ .** (a)–(d) Different measures of the difference in outcome of simulations between algorithms with $\epsilon > 0$ and $\epsilon = 0$ (quasi-exact). (a) Difference, $\Delta \bar{i}_{\max} = \bar{i}_{\max}(\epsilon) - \bar{i}_{\max}(0)$, in the peak average fraction of infected nodes, $\bar{i}_{\max} = \bar{I}_{\max}/N$. (b) Difference Δt_{\max} between the times at which this peak takes place, normalized by μ_0 . (c) Difference, $\Delta \bar{r}_{\infty}$, in the average fraction of nodes affected by the infection—the average *attack rate*. (d) Kullback-Leibler divergence, $\text{KL}[p(r_{\infty})]$, between the distributions of attack rates (e) Time per simulation of the process. Simulations were performed on an activity-driven network with $N = 100$ nodes and activities $a_i = z_i/10$ and $z_i \sim z_i^{-3.2}$, for $z_i \in [0.03, 1]$; nodes’ recovery times followed Eq. (20) with $k = 1.5$, $\mu_0 = 10^{-4}$ Hz, and $\Delta t = 1$ s.

TABLE I. Notation pertaining to the temporal Gillespie algorithm.

Symbol	Description	First appearance(s)
t	Real time.	Sec. II
Δt	Duration of a time-step.	Sec. III
n	Time-step number.	Sec. V
m	Possible transition / transition process.	Sec. II
λ_m	Transition rate for m .	Sec. II
$I_m(t)$	Function indicating whether the transition m may take place at time t .	Sec. V
$\Omega(t)$	Set of transition processes at time t .	Secs. II,V
$M(t)$	Number of transition processes at time t .	Secs. II,V
Ω	Set of total possible transitions between two consecutive transition events.	Secs. II,V
M	Number of total possible transitions between two consecutive transition events.	Secs. II,V
$\Lambda, \Lambda(t)$	Cumulative transition rate (at time t): $\Lambda(t) = \sum_{m \in \Omega(t)} \lambda_m$.	Secs. IV,V
$\mathbb{L}(t; t^*)$	Integrated cumulative transition rate (from t^* to t).	Sec. V
τ	Waiting time between two consecutive transitions.	Sec. IV
$S(\tau)$	Waiting time survival function.	Sec. IV
t^*, t^{**}	Times when the last/next transition took/takes place, respectively.	Sec. V
n^*, n^{**}	Time-steps during which the last/next transition took/takes place, respectively.	Sec. V
τ'	Renormalized waiting time between two consecutive transition events.	Sec. V
$S(\tau')$	Renormalized waiting time survival function.	Sec. V
$\tau' \sim \text{Exp}(1)$	τ' is exponentially distributed with unit mean.	Sec. V (Sec. IV)
$\mathcal{F}_t^{(m)}$	Filtration for the transition process m .	Sec. VII
\mathcal{F}_t	Union of all $\mathcal{F}_t^{(m)}$.	Sec. VII

TABLE II. Notation pertaining to compartmental contagion models and time-varying networks.

Symbol	Description	First appearance
i, j	Node.	Sec. II A
N	Number of nodes in network.	Sec. II A
$(i, j)_t$	Contact taking place at time t between nodes i and j .	Sec. II A
$\mathcal{N}_i(t)$	Neighborhood of i : set of nodes in contact with i at time t .	Sec. II A
$k_{\mathcal{I}}(t)$	Number of infected nodes in contact with i at time t	Sec. II A
$\overline{k}(t)$	Average degree (number of contacts per node) of network at time t .	Fig. 4
$x_i(t)$	Random variable specifying the state (compartment) of node i at time t .	Sec. II A
$\mathcal{X} \in \{\mathcal{X}_1, \mathcal{X}_2 \dots \mathcal{X}_q\}$	Possible node states (compartments).	Sec. II A
X_p	Number of nodes in state \mathcal{X}_p .	Sec. II A
$\mathcal{S}, \mathcal{I}, \mathcal{R}$	Possible node states in SIS and SIR models of epidemic spreading.	Sec. II A
S, I, R	Number of nodes in each of the states \mathcal{S}, \mathcal{I} , and \mathcal{R} , respectively.	Sec. II A
β	Rate of $\mathcal{S} \rightarrow \mathcal{I}$ transition of a susceptible node in contact with a single infectious node.	Sec. II A
μ	Rate of spontaneous $\mathcal{I} \rightarrow \mathcal{R}$ or $\mathcal{I} \rightarrow \mathcal{S}$ transition of an infectious node.	Sec. II A
μ_0	Scale parameter of the Weibull distribution.	Sec. VII A
k	Shape parameter of the Weibull distribution.	Sec. VII A
$t^{(m)}$	Starting time for transition process m (e.g. the time when a node got infected).	Sec. VII A

Appendix B: Algorithms for simulating specific contagion models

We here give pseudocode for the application of the temporal Gillespie algorithm to three specific models: section B1 treats the Poissonian SIR process, section B2 treats the Poissonian SIS process, and section B3 treats

a non-Markovian SIR process with Weibull distributed recovery times.

We assume in the following that the time-varying network is represented by a list of lists of individual contacts taking place during each time-step. An individual contact, termed **contact**, is represented by a tuple of nodes, **i** and **j**. The list `contactLists[t]` gives

the contacts taking place during a single time-step, t , for $t=0,1,\dots,T_{\text{simulation}}-1$, where $T_{\text{simulation}}$ is the desired number of time-steps to simulate. The state of each node is given by the vector x , where the entry $x[i] \in \{S, I, R\}$ gives the state of node i .

As one may always renormalize time by the duration of a time-step, Δt , we have in the following set $\Delta t = 1$. Note that `beta` and `mu` in the code then corresponds to $\beta\Delta t$ and $\mu\Delta t$, respectively.

1. SIR process

The classical SIR model with constant infection and recovery rates is the simplest epidemic model where individuals can gain immunity. As discussed in the main text, nodes may be in one of three states: susceptible (\mathcal{S}), infectious (\mathcal{I}), or recovered (\mathcal{R}). Two different transition types let the nodes switch state: a spontaneous $\mathcal{I} \rightarrow \mathcal{R}$ transition which takes place with rate μ , and a contact-dependent $\mathcal{S} \rightarrow \mathcal{I}$ transition which takes place with rate β upon contact with an infectious node. Figure 9 shows how the temporal Gillespie algorithm may be implemented for an SIR process on a time-varying contact network.

2. SIS process

In the SIS model, nodes can be in one of two states: susceptible (\mathcal{S}) or infectious (\mathcal{I}). As for the SIR model, two different transition types let the nodes switch state: a spontaneous $\mathcal{I} \rightarrow \mathcal{S}$ transition which takes place with rate μ , and a contact-dependent $\mathcal{S} \rightarrow \mathcal{I}$ transition which takes place with rate β upon contact with an infectious node.

The SIS model is implemented in a manner very similar to the SIR model; an implementation can be found by using the code of Figure 9 with lines 07 and 40 removed and line 37 replaced by `x[m] = S`. C++ code is given in Supplemental Files [30] for both homogeneous and heterogeneous populations.

3. Non-Markovian SIR process

We consider in the main text (Sec. VII) an SIR model with non-constant recovery rates; instead of being exponentially distributed (as in the constant-rate SIR model), the individual recovery times, $\tau^{(m)}$, are here Weibull distributed,

$$\tau^{(m)} \sim k\mu_0 \left(\mu_0 \tau^{(m)} \right)^{k-1} e^{-\mu_0 \tau^{(m)}}. \quad (\text{B1})$$

As for the classical SIR model, nodes may be in one of three states: susceptible (\mathcal{S}), infectious (\mathcal{I}), or recovered (\mathcal{R}). Two different transition types let the nodes switch

```

//Initialize:
01 FOR i=1,...,N
02   x[i] = S //set node states to S
03 ENDFOR
04 x[root] = I //set state of root node to I
05 m_I = [root] //list of infected nodes
06 N_I = 1 //number of infected nodes
07 N_R = 0 //number of recovered nodes
08 Mu = mu //cumulative recovery rate
09 tau = randexp(1) //draw tau ~ exp(1)

//Run through the time-steps:
10 FOR t=0,1,...,T_simulation-1
    //Update list of possible S->I transitions:
11   CLEAR m_SI //S nodes in contact with I nodes
12   FOR contact in contactLists[t]
13     (i,j) = contact
14     IF (x[i],x[j])== (S,I)
15       APPEND i to m_SI
16     ELSE IF (x[i],x[j])== (I,S)
17       APPEND j to m_SI
18     ENDIF
19   ENDFOR
20   M_SI = length of m_si
21   Beta = beta*M_SI //cumulative infection rate
22   Lambda = Mu+Beta //cumulative transition rate
    //Check if a transition takes place:
23   IF Lambda<tau //no transition
24     tau -= Lambda
25   ELSE //at least one transition
26     xi = 1. //remaining fraction of time-step
27     WHILE xi*Lambda>=tau
28       DRAW z uniformly from [0,Lambda)
29       IF z<Beta //S->I transition
30         DRAW m at random from m_SI
31         x[m] = I
32         APPEND m to m_I
33         N_I += 1
34         Mu += mu
35       ELSE //I->R transition
36         DRAW m at random from m_I
37         x[m] = R
38         REMOVE m from m_I
39         N_I -= 1
40         N_R += 1
41         Mu -= mu
42       ENDIF
43     xi -= tau/Lambda //update remaining fraction
    //Update list of S->I transitions and rates:
44     REDO lines 11-22
45     tau = randexp(1) //draw new tau
46   ENDWHILE
47 ENDIF
    //Read out the desired quantities:
48   WRITE N_I, N_R, ...
49 ENDFOR

```

FIG. 9. Pseudocode for an SIR process with constant and homogeneous transition rates. C++ code for homogeneous and heterogeneous populations is given in Supplemental Files [30].

```

//Initialize:
01 FOR i=1,...,N
02   x[i] = S //set nodes states to S
03 ENDFOR
04 x[root] = I //set state of root node to I
05 t_inf[root] = 0 //time of infection = 0
06 m_I = [root] //list of infected nodes
07 mus = [mu(0)] //list of recovery rates
08 Mu = mu(0) //cumulative recovery rate
09 N_I = 1 //number of infected nodes
10 N_R = 0 //number of recovered nodes
11 tau = randexp(1) //draw tau ~ exp(1)

//Run through the time-steps:
12 FOR t=0,1,...,T_simulation-1
//Update mus if t-t*>=epsilon*mu_avg:
13 IF t-t*>=epsilon*mu_avg
14   CLEAR mus
15   FOR m in m_I
16     APPEND mu(t-t_inf[m]) to mus
17   ENDFOR
18   Mu = sum of mus
19 ENDIF
//Update list of possible S->I transitions:
20 CLEAR m_SI //S nodes in contact with I nodes
21 FOR contact in contactLists[t]
22   (i,j) = contact
23   IF (x[i],x[j])== (S,I)
24     APPEND i to m_SI
25   ELSE IF (x[i],x[j])== (I,S)
26     APPEND j to m_SI
27   ENDIF
28 ENDFOR
29 M_SI = length of m_si
30 Beta = beta*M_SI //cumulative infection rate
31 Lambda = Mu+Beta //cumulative transition rate

//Check if transition takes place:
32 IF Lambda<tau //no transition
33   tau -= Lambda
34 ELSE //at least one transition
35   xi = 1. //remaining fraction of time-step
36   t* = t //for calculating transition times
37   WHILE xi*Lambda>=tau
38     t* += tau/Lambda //transition time
39     DRAW z uniformly from [0,Lambda)
40     IF z<Beta //S->I transition
41       DRAW m at random from m_SI
42       x[m] = I
43       t_inf[m] = t*
44       APPEND m to m_I
45       N_I += 1
46     ELSE //I->R transition
47       DRAW m from m_I with weight mus[m]
48       x[m] = R
49       REMOVE m from m_I
50       N_I -= 1
51       N_R += 1
52     ENDIF
53   xi -= tau/Lambda //update remaining fraction
//Update mus:
54   CLEAR mus
55   FOR m in m_I
56     APPEND mu(t*-t_inf[m]) to mus
57   ENDFOR
58   Mu = sum of mus
//Update list of S->I transitions and rates:
59   REDO lines 20-31
60   tau = randexp(1) //draw new tau
61 ENDWHILE
62 ENDIF
//Read out the desired quantities:
63 WRITE N_I, N_R, ...
64 ENDFOR

```

FIG. 10. Pseudocode for a non-Markovian SIR process with non-constant recovery rates. The function μ returns the instantaneous recovery rate as function of $(t - t^*)$; for Weibull distributed recovery times, μ is given by Eq. (20). C++ code is given in Supplemental Files [30].

state: a contact-dependent $\mathcal{S} \rightarrow \mathcal{I}$ transition with constant rate β upon contact with an infectious node, and a spontaneous $\mathcal{I} \rightarrow \mathcal{R}$ transition which takes place with rate $\mu(t; t^{(m)})$, given by Eq. (20).

The implementation of the SIR model with Weibull distributed waiting times requires some extension of the code for the constant-rate SIR model to account for the heterogeneous and time-dependent recovery rates. To this end, we introduce the following variables: t_inf lists the times at which each node was infected (if applicable); t^* is the exact time at which the last transition took place; μ is a function of time that is called as $\mu(t-t_inf[m])$ to return the instantaneous recovery rate of m at time t ; μ_avg is the expected recovery time of an infected node and is used together with the precision control parameter `epsilon` in the approximate simulation scheme discussed in Sec. VII. Figure 10 shows pseudocode for an implementation of the SIR model with

Weibull distributed recovery times.

Appendix C: Details on how Monte Carlo simulations were performed

All simulations for comparing the speed of algorithms were performed sequentially on a HP EliteBook Folio 9470m with a dual-core (4 threads) Intel Core i7-3687U CPU @ 2.10 GHz. The system had 8 GB 1600 MHz DDR3 SDRAM and a 256 GB mSATA Solid State Drive. Code was compiled with TDM GCC 64 bit using g++ with the optimization option `-O2`. Speedtests were also performed using `-O3` and `-Ofast`, but `-O2` gave the fastest results, both for rejection sampling and temporal Gillespie algorithms.

TABLE III. Summary statistics for empirical face-to-face contact networks from the *SocioPatterns* collaboration [33]: social setting; number of nodes in the network, N ; total duration of measurements, T ; average instantaneous degree, $\overline{k(t)}$.

Setting	N	T	$\overline{k(t)}$	Reference
Conference	399	32 hours	0.035	[34]
High school	327	4 days	0.032	[35]
Hospital	80	4 days	0.032	[36]
Workplace	92	11 days	0.002	[37]

Appendix D: Empirical time-varying networks

Empirical networks describing human contact differ from simulated networks in a number of ways, for example, their structure and dynamics are more complex, but perhaps most importantly in the perspective of optimizing simulations, they are of finite length. One is often interested in long-time behavior or slowly evolving processes compared to the length of available data. To overcome this limitation, one usually loops over the data set. This means that if a node enters an absorbing state (compartment) such as the recovered (\mathcal{R}) state in the SIR model, one may remove all following contacts to this node from the data, thus reducing the number of contacts that one must go through during the following loop. Furthermore, since the $\mathcal{I} \rightarrow \mathcal{R}$ transition is independent of the network, one may also remove all contacts between two infected nodes.

```

01 FOR t=0,1,...,T_simulation-1
    //Update list of possible S->I transitions:
02 CLEAR m_SI //S nodes in contact with I nodes
03 FOR contact in contactLists[t]
04     (i,j) = contact
05     IF x[i]==S
06         IF x[j]==I
07             APPEND i to m_SI
08         ELSE IF x[j]==R //remove if x[j]==R
09             REMOVE contact from contactLists[t]
10         ENDIF
11     ELSE IF x[i]==I
12         IF x[j]==S
13             APPEND j to m_SI
14         ELSE //remove if (x[i],x[j])==I or x[i]==R
15             REMOVE contact from contactLists[t]
16         ENDIF
17     ELSE //remove if x[i]==R
18         REMOVE contact from contactLists[t]
19     ENDIF
20 ENDFOR
21 .
22 .
23 .
XX ENDFOR

```

FIG. 11. Pseudocode for counting possible $\mathcal{S} \rightarrow \mathcal{I}$ transitions with removal of outdated contacts.

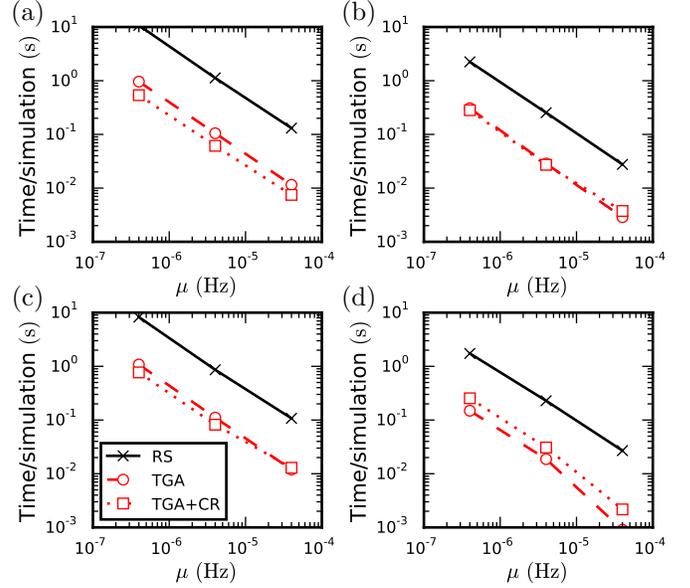
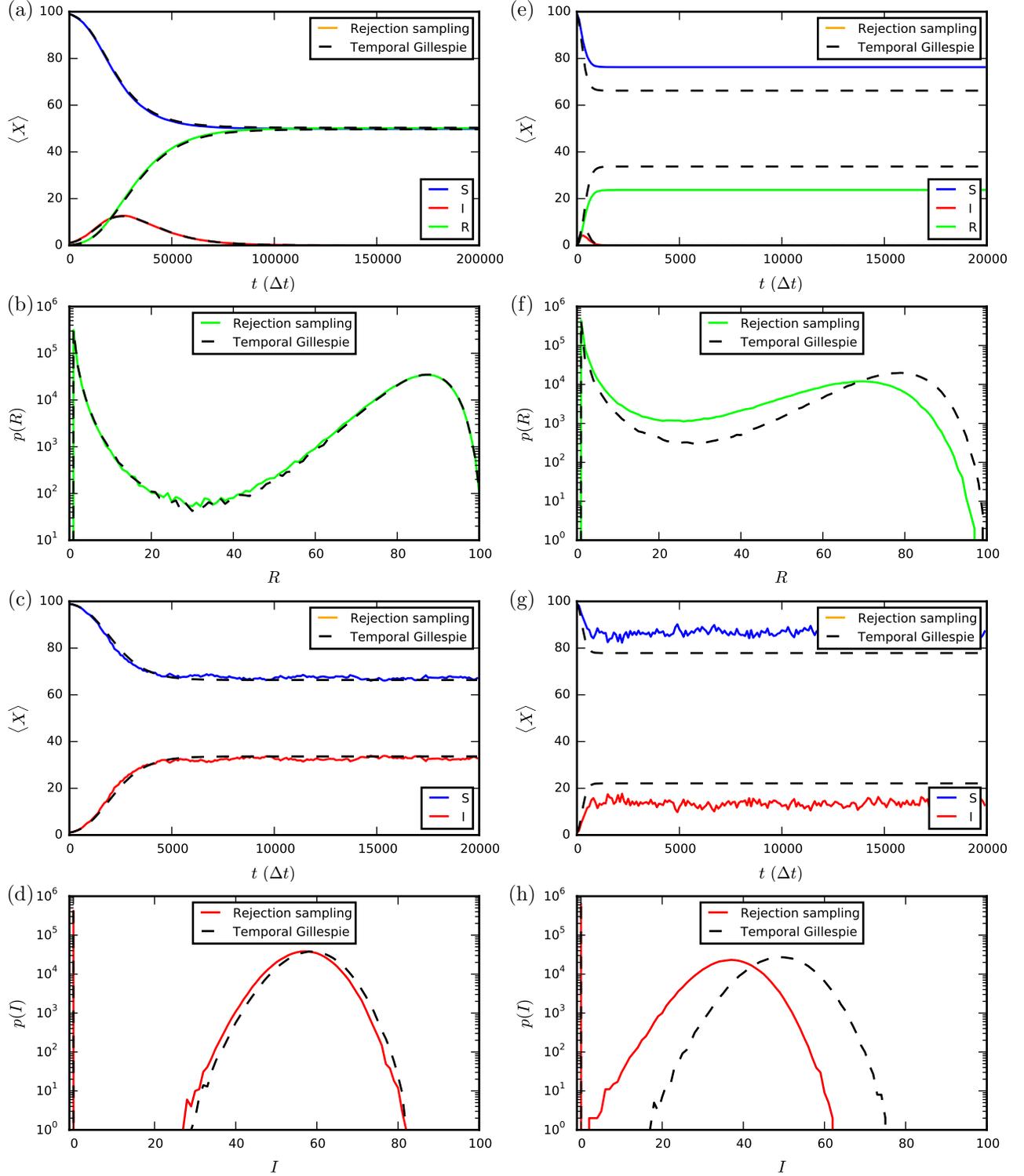


FIG. 12. **Comparison of the speed of the temporal Gillespie algorithm, with (TGA+CR) and without (TGA) contact removal, and the rejection sampling algorithm (RS): empirical time-varying networks.** Time per realization of a single simulation on empirical face-to-face contact networks: (a) at a scientific conference, $N = 396$, $T = 5750 \Delta t$, and $\overline{k(t)} \approx 0.03$; (b) in a hospital ward, $N = 80$, $T = 17382 \Delta t$, and $\overline{k(t)} \approx 0.03$; (c) in a high school, $N = 327$, $T = 18179 \Delta t$, and $\overline{k(t)} \approx 0.03$; (d) at a workplace, $N = 92$, $T = 49382 \Delta t$, and $\overline{k(t)} \approx 0.002$ (Table III). Simulations were performed with $\beta = 100\mu$ in A–C and $\beta = 1000\mu$ in D.

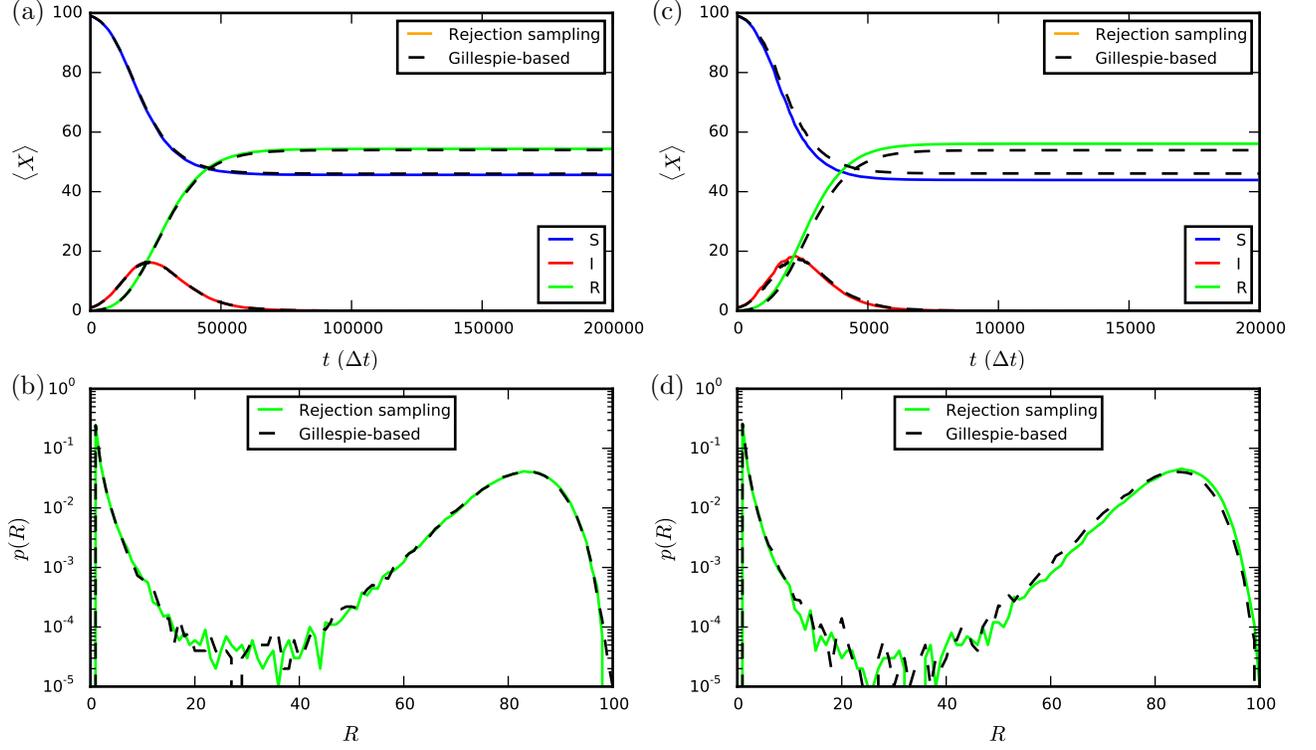
Pseudocode for an algorithm that removes obsolete contacts is given in Fig. 11. Figure 12 compares the speed of the temporal Gillespie algorithm with and without contact removal for simulations of constant-rate SIR process on empirical networks of face-to-face contacts (Table III); the computation speed is also compared to rejection sampling. The simulations on empirical data (Fig. 12) confirm the speed gain obtained by using the temporal Gillespie algorithm found for synthetic networks (Fig. 4). Comparison between algorithms with and without contact removal, however, reveal no clear gain in removing obsolete contacts; for short datasets a small speed gain may be obtained [Fig. 12(a)], while for longer datasets contact removal makes the algorithm slower [Fig. 12(d)] due to the added overhead involved in identifying and removing obsolete contacts.

-
- [1] Barrat, A., Barthélemy, M. & Vespignani, A. *Dynamical Processes on Complex Networks* (Cambridge University Press, 2008).
- [2] Balcan, D. *et al.* Multiscale mobility networks and the spatial spreading of infectious diseases. *Proc. Natl. Acad. Sci. USA* **106**, 21484–21489 (2009).
- [3] Pastor-Satorras, R., Castellano, C., Van Mieghem, P. & Vespignani, A. Epidemic processes in complex networks. *arXiv:1408.2701* .
- [4] Ferreira, S. C., Castellano, C. & Pastor-Satorras, R. Epidemic thresholds of the susceptible-infected-susceptible model on networks: a comparison of numerical and theoretical results. *Phys. Rev. E* **86**, 041125 (2012).
- [5] Tizzoni, M. *et al.* Real-time numerical forecast of global epidemic spreading: case study of 2009 A/H1N1pdm. *BMC Med.* **10**, 165 (2012).
- [6] Doob, J. L. Topics in the theory of Markoff chains. *T. Am. Math. Soc.* **52**, 37–64 (1942).
- [7] Doob, J. L. Markoff chains—denumerable case. *T. Am. Math. Soc.* **58**, 455–473 (1945).
- [8] Kendall, D. G. An artificial realization of a simple "birth-and-death" process. *J. R. Stat. Soc. Ser. B Stat. Methodol.* **12**, 116–119 (1950).
- [9] Bartlett, M. S. Stochastic processes or the statistics of change. *J. R. Stat. Soc. Ser. C Appl. Stat.* **2**, 44–64 (1953).
- [10] Gillespie, D. T. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *J. Comput. Phys.* **22**, 403–434 (1976).
- [11] Gillespie, D. T. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.* **81**, 2340–2361 (1977).
- [12] Holme, P. Model versions and fast algorithms for network epidemiology. *arXiv:1403.1011v1* .
- [13] Boguñá, M., Lafuerza, L., Toral, R. & Serrano, M. A. Simulating non-Markovian stochastic processes. *Phys. Rev. E* **90**, 042108. (2014).
- [14] Onnela, J.-P. *et al.* Structure and tie strengths in mobile communication networks. *Proc. Natl. Acad. Sci. USA* **104**, 7332–7336 (2007).
- [15] Rybski, D., Buldyrev, S. V., Havlin, S., Liljeros, F. & Makse, H. A. Scaling laws of human interaction activity. *Proc. Natl. Acad. Sci. USA* **106**, 12640–12645 (2009).
- [16] Cattuto, C. *et al.* Dynamics of person-to-person interactions from distributed RFID sensor networks. *PLoS One* **5**, e11596 (2010).
- [17] Vázquez, A., Rácz, B., Lukács, A. & Barabási, A.-L. Impact of non-Poissonian activity patterns on spreading processes. *Phys. Rev. Lett.* **98**, 158702 (2007).
- [18] Miritello, G., Moro, E. & Lara, R. Dynamical strength of social ties in information spreading. *Phys. Rev. E* **83**, 045102 (2011).
- [19] Karsai, M. *et al.* Small but slow world: how network topology and burstiness slow down spreading. *Phys. Rev. E* **83**, 025102 (2011).
- [20] Panisson, A. *et al.* On the dynamics of human proximity for data diffusion in ad-hoc networks. *Ad Hoc Networks* **10**, 1532–1543 (2012).
- [21] Gauvin, L., Panisson, A., Cattuto, C. & Barrat, A. Activity clocks: spreading dynamics on temporal networks of human contact. *Sci. Rep.* **3**, 3099 (2013).
- [22] Holme, P. & Liljeros, F. Birth and death of links control disease spreading in empirical contact networks. *Sci. Rep.* **4**, 4999 (2014).
- [23] Karsai, M., Perra, N. & Vespignani, A. Time varying networks and the weakness of strong ties. *Sci. Rep.* **4**, 4001 (2014).
- [24] Holme, P. & Saramäki, J. Temporal networks. *Phys. Rep.* 1–28 (2012).
- [25] Cai, C.-R., Wu, Z.-X. & Guan, J.-Y. Behavior of susceptible-vaccinated-infected-recovered epidemics with diversity in the infection rate of individuals. *Phys. Rev. E* **88**, 062805 (2013).
- [26] Ferguson, N. M. *et al.* Strategies for mitigating an influenza pandemic. *Nature* **442**, 448–452 (2006).
- [27] Lloyd, A. L. Realistic distributions of infectious periods in epidemic models: changing patterns of persistence and dynamics. *Theor. Popul. Biol.* **60**, 59–71 (2001).
- [28] Note that while the principal field of application of the algorithm is time-varying networks, the algorithm may also be applied to systems where a network-based description is not appropriate.
- [29] Martelloni, G., Santarlasci, A., Bagnoli, F. & Santini, G. Modeling ant battles by means of a diffusion-limited Gillespie algorithm. *arXiv:1503.06094v1* .
- [30] C++ code for implementations of the temporal Gillespie algorithm can be found at github.com/CLVestergaard/TemporalGillespieAlgorithm
- [31] Perra, N., Gonçalves, B., Pastor-Satorras, R. & Vespignani, A. Activity driven modeling of time varying networks. *Sci. Rep.* **2**, 469 (2012).
- [32] Supplemental figures are found at the end of the manuscript.
- [33] www.sociopatterns.org.
- [34] Stehlé, J. *et al.* Simulation of an SEIR infectious disease model on the dynamic contact network of conference attendees. *BMC Med.* **9**, 87 (2011).
- [35] Fournet, J. & Barrat, A. Contact patterns among high school students. *PLoS One* **9**, e107878 (2014).
- [36] Vanhems, P. *et al.* Estimating potential infection transmission routes in hospital wards using wearable proximity sensors. *PLoS One* **8**, e73970 (2013).
- [37] Géniois, M., Vestergaard, C. L., Cattuto, C. & Barrat, A. Data on face-to-face contacts in an office building suggests a low-cost vaccination strategy based on community linkers. *Network Science* **FirstView**, 1–22 (2015).

SUPPLEMENTAL FIGURES



Supplemental FIG. 1. **Comparison of numerical results from temporal Gillespie and rejection sampling algorithms for high transition probability per time-step.** (a)–(d) $\beta\Delta t = 10^{-1}$ and $\mu\Delta t = 10^{-3}$; (e)–(h) $\beta\Delta t = 1$ and $\mu\Delta t = 10^{-2}$. (a),(e) Mean number of nodes in each state of the SIR model as function of time. (b),(f) Distribution of final epidemic size (number of recovered nodes when $I = 0$) in the SIR model. (c),(g) Mean number of nodes in each state of the SIS model as function of time. (d),(h) Distribution of the number of infected nodes in the stationary state ($t \rightarrow \infty$) of the SIS model. All simulations were performed 1 000 000 times with the root node chosen at random on an activity driven network consisting of $N = 100$ nodes, with activities $a_i = \eta z_i$, where $\eta = 0.1$ and $z_i \sim z_i^{-3.2}$ for $z_i \in [0.03, 1)$, and a node formed two contacts when active.



Supplemental FIG. 2. **Comparison of numerical results from temporal Gillespie and rejection sampling algorithms for a non-Markovian SIR process.** (a),(c) Mean number of nodes in each state as function of time in the SIR model with Weibull distributed recovery times (Sec. VII A); the parameter controlling the precision of the temporal Gillespie algorithm was set to $\epsilon = 0$ (quasi-exact). (b),(d) Distribution of final epidemic size (number of recovered nodes when $I = 0$). (a),(b) $\beta\Delta t = 10^{-2}$ and $\mu\Delta t = 10^{-4}$; (c),(d) $\beta\Delta t = 10^{-1}$ and $\mu\Delta t = 10^{-3}$. The outcome of the rejection sampling algorithm approaches that of the temporal Gillespie algorithm as $\beta\Delta t$ and $\mu\Delta t$ become smaller. All simulations were performed 100 000 times with the root node chosen at random on an activity driven network consisting of $N = 100$ nodes, with activities $a_i = \eta z_i$, where $\eta = 0.1$ and $z_i \sim z_i^{-3.2}$ for $z_i \in [0.03, 1)$, and a node formed two contacts when active. Nodes' recovery times followed Eq. (20) with $k = 1.5$ and the length of a time-step was $\Delta t = 1$ s.