PLOS COMPUTATIONAL BIOLOGY

CORRECTION

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

Christian L. Vestergaard, Mathieu Génois

There is an error in Pseudocode 1, and 2, where a line was omitted after line 46 (in 1), 61 (in 2). The following line was omitted:

tau - = xi*Lambda //subtract remainder of time-step

Citation: Vestergaard CL, Génois M (2019) Correction: Temporal Gillespie algorithm: Fast simulation of contagion processes on time-varying networks. PLoS Comput Biol 15(7): e1007190. https://doi.org/10.1371/journal.pcbi.1007190

Published: July 3, 2019

Pseudocode 1:

```
   //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     tau -= xi*Lambda //subtract remainder of time-step
48   ENDIF
     //Read out the desired quantities:
49   WRITE N_I, N_R, ...
50 ENDFOR
```

Pseudocode 1. Pseudocode for an SIR process with constant and homogeneous transition rates. C++ code for homogeneous and heterogeneous populations is given in S1 Files.

https://doi.org/10.1371/journal.pcbi.1007190.g001

Pseudocode 2:

```
   //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     tau -= xi*Lambda //subtract remainder of time-step
63   ENDIF
     //Read out the desired quantities:
64   WRITE N_I, N_R, ...
65 ENDFOR
```

Pseudocode 2. Pseudocode for a non-Markovian SIR process with non-constant recovery rates. The function mu returns the instantaneous recovery rate as function of $(t-t^*)$; for Weibull distributed recovery times, mu is given by Eq. (23). C++ code is given in S1 Files.

https://doi.org/10.1371/journal.pcbi.1007190.g002

This change in the pseudocode results in changes to supporting information S1, S2 and S3 Figs. The corrected supporting information figures are:

Supporting information

S1 Fig. Numerical results from temporal Gillespie and rejection sampling algorithms for contagion processes taking place on empirical networks. (A)–(D) for a SIR process and (E)–(H) a SIS process. (A),(B),(E), and (F) for $\beta\Delta t = 10^{-2}$ and $\mu\Delta t = 10^{-4}$; (C),(D),(G), and (H) for $\beta\Delta t = 10^{-1}$ and $\mu\Delta t = 10^{-3}$. (A),(C) Mean number of nodes in each state of the SIR model as function of time. (B),(D) Distribution of final epidemic size (number of recovered nodes when $I = 0$) in the SIR model. (E),(G) Mean number of nodes in each state of the SIS model as function of time. (F),(H) Distribution of the number of infected nodes in the stationary state ($t \to \infty$) of the SIS model. All simulations were performed 1 000 000 times with the root node chosen at random on a face-to-face contact network recorded in a high school (Table 1).
(PDF)

S2 Fig. Comparison of numerical results from temporal Gillespie and rejection sampling algorithms for high transition probability per time-step. (A)–(D) for a SIR process and (E)–(H) a SIS process. (A),(B),(E), and (F) for $\beta\Delta t = 10^{-1}$ and $\mu\Delta t = 10^{-3}$; (C),(D),(G), and (H) for $\beta\Delta t = 1$ and $\mu\Delta t = 10^{-2}$. (A),(C) Mean number of nodes in each state of the SIR model as function of time. (B),(D) Distribution of final epidemic size (number of recovered nodes when $I = 0$) in the SIR model. (E),(G) Mean number of nodes in each state of the SIS model as function of time. (F),(H) Distribution of the number of infected nodes in the stationary state ($t \to \infty$) 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.
(PDF)

S3 Fig. 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. 6: "Non-Markovian processes"); 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 $\gamma = 1.5$ and the length of a time-step was $\Delta t = 1$ s.
(PDF)

Reference

1. Vestergaard CL, Génois M (2015) Temporal Gillespie Algorithm: Fast Simulation of Contagion Processes on Time-Varying Networks. PLoS Comput Biol 11(10): e1004579. https://doi.org/10.1371/journal.pcbi.1004579 PMID: 26517860