Course EN. 520.622 Principles of Complex Networked Systems

Project 10: Stochastic modeling of nonlinear epidemiology

Yingheng Wang
ywang584@jhu.edu

Dec/16/2021

Abstract

In this project, I study and re-implement a paper on stochastic analysis of nonlinear epidemic processes, in which the authors conducted numerical simulation independently by the algorithm derived from the master equations, as well as by an event-driven Monte Carlo algorithm, leading to the identical results.

1 Overview

The selected project 10 is aiming to simulate the nonlinear stochastic process of epidemiology by applying the algorithm derived from the master equations and the Monte Carlo algorithm. The objective of the project is to reproduce Fig. 2, Fig. 3 and Fig. 4 in Chen [2005]. Specifically, my contributions are four folds:

- I read the paper and re-derived each equation included in the main paper and appendix. (I also found a typpo in Fig. 1 of Chen [2005])
- I re-implemented the ME simulation in Matlab and reproduced related graphs in Fig. 2 - Fig. 4 of Chen [2005]. I completed two versions of the code, which included the plain version (in Sec 2.1) with a set of differential equations and the matrix version (in Sec 2.2) with simplified computation and clearer code formatting.
- I re-implemented the Monte Carlo simulation in Matlab and reproduced the related graphs in Fig. 2 - Fig. 4 of Chen [2005].
- I tuned the parameters both in the ME simulation and the Monte Carlo simulation to reproduce the best-matched figures.

2 Simulation based on the ME

For simulating the temporal profile of the epidemic, I solved all the differential equations from Eq. (18) to Eq. (35) simultaneously by using a function, ode45 (Petzold [1997]), in Matlab. Among these equations, Eq. (18)-(20) are for the evolution of the macroscopic behavior of the system, Eq. (26)-(28) are the expectations of random variables Y_1, Y_2 and Y_3, Eq. (30)-(32) are their variances and Eq. (33)-(35) are their covariances.

I also implemented the ME simulation by solving Eq. (30)-(32) and Eq. (33)-(35) in matrix operations according to the matrix **A** in Eq. (23) and **B** in Eq. (24). Formally, given a matrix **C** that

$$\mathbf{C} = \begin{bmatrix} E(Y_1^2) & E(Y_2Y_1) & E(Y_3Y_1) \\ E(Y_1Y_2) & E(Y_2^2) & E(Y_2Y_3) \\ E(Y_1Y_3) & E(Y_2Y_3) & E(Y_3^2) \end{bmatrix},$$

we can obtain a matrix **D** by calculating $\mathbf{D} = \mathbf{AC} + \mathbf{B}$. Such that we can easily get the variances and covariances as follows:

$$\frac{d}{dt}E[Y_1^2] = D_{11};\ \frac{d}{dt}E[Y_2^2] = D_{22};\ \frac{d}{dt}E[Y_3^2] = D_{33};$$

$$\frac{d}{dt}E[Y_1Y_2] = D_{12} + D_{21} - B_{12};$$

$$\frac{d}{dt}E[Y_1Y_3] = D_{13} + D_{31} - B_{13};$$

$$\frac{d}{dt}E[Y_2Y_3] = D_{23} + D_{32} - B_{23}.$$

The above equations are also equivalent to Eq. (B.10)-(B.12) and Eq. (B.13)-(B.15) in the appendix of Chen [2005].

I provided the pseudocode of the calculation of variances and covariances in the ME simulation in Appendix 4). Then I refactored the codes and inserted them into my implementation.

2.1 The plain version

I simulated the master equations by solving the differential equations provided in Eq. (18)-(35) of Chen [2005]. The implementation, reproduction and parameter sensitivity analysis are given in this section.

2.1.1 Implementation

```
[t, y] = ode45(@mean, [0 25], [762;1;0;0;0;0;0;0;0;0;0;0]);
n1 = y(:, 1:3) + y(:, 4:6);
std1 = y(:, 1:3) + y(:, 4:6) + sqrt(y(:, 7:9));
std2 = y(:, 1:3) + y(:, 4:6) - sqrt(y(:, 7:9));

%plot temporal profile of the S,I,R population
%without their standard deviation envelop by master-equation.
plot(t,n1(:,1),'-',t,n1(:,2),'-',t,n1(:,3),'-');
title('Temporal profile of the S,I,R population.');
xlabel('Time (days)');
ylabel('Population (N)');
legend('Susceptibles','Infectives','Recovered')

%plot temporal profile of the susceptibles population
%with their standard deviation envelop by master-equation.
plot(t,n1(:,1),'-',t,std1(:,1),'-',t,std2(:,1),'-');
title('Temporal profile of the susceptibles population.');
```

```
xlabel('Time (days)');
ylabel('Susceptibles (S)');
legend('ME mean','mean+std','mean-std')

% plot temporal profile of the infected population
% with their standard deviation envelop by master-equation.
plot(t,n1(:,2),'-',t,std1(:,2),'-',t,std2(:,2),'-');
title('Temporal profile of the infected population.');
xlabel('Time (days)');
ylabel('Infectives (I)');
legend('ME mean','mean+std','mean-std')

% plot temporal profile of the recovered population
% with their standard deviation envelop by master-equation.
plot(t,n1(:,3),'-',t,std1(:,3),'-',t,std2(:,3),'-');
title('Temporal profile of the recovered population.');
xlabel('Time (days)');
ylabel('Recovery (R)');
legend('ME mean','mean+std','mean-std')

function dydt = mean(t,y)
    % function that solves the differential equations
    % from the system size expansion of master equations
    % to simulate the temporal profile of the epidemic.
    omega = 763; lambda1 =  0.000003;
    lambda1_ = omega * lambda1; lambda2 = 0.5;

    phi = y(1); theta = y(2); gamma = y(3);
    e = y(4:6); v = y(7:9); c = y(10:12);

    dydt(1) = - lambda1_ * phi * theta;
    dydt(2) = lambda1_ * phi * theta - lambda2 * theta;
    dydt(3) = lambda2 * theta;

    dedt(1) = - lambda1_ * theta * e(1) - lambda1_ * phi * e(2);
    dedt(2) = lambda1_ * theta * e(1) + (lambda1_ * phi - lambda2) * e
        (2);
    dedt(3) = lambda2 * e(2);

    C(12) = c(1); C(13) = c(2); C(23) = c(3);

    dvdt(1) = -2 * theta * lambda1_ * v(1) - 2 * phi * lambda1_ * C(12)
         + phi * theta * lambda1_;
    dvdt(2) = 2 * theta * lambda1_ * C(12) + 2 * (phi * lambda1_ -
        lambda2) * v(2) + phi * theta * lambda1_ + lambda2 * theta;
    dvdt(3) = 2 * lambda2 * C(23) + lambda2 * theta;

    dcdt(1) = (phi * lambda1_ - lambda2 - theta * lambda1_) * C(12) +
        theta * lambda1_ * v(1) - phi * lambda1_ * v(2) - lambda1_ * phi
         * theta;
    dcdt(2) = (-theta * lambda1_) * C(13) + (-phi * lambda1_) * C(23) +
         lambda2 * C(12);
    dcdt(3) = (theta * lambda1_) * C(13) + (phi * lambda1_ - lambda2) *
         C(23) + lambda2 * v(2) - lambda2* theta;

    dydt = [dydt(1); dydt(2); dydt(3); dedt(1); dedt(2); dedt(3); ...,
         D(1,1); D(2,2); D(3,3); D(1,2) + D(2,1) - B(1,2); ...,
         D(1,3) + D(3,1) - B(1,3); D(2,3) + D(3,2) - B(2,3)];
end
```

2.1.2 Reproduction

The reproduced figures are shown below. These figures (Fig. 1, Fig. 2 and Fig. 3) are the temporal profiles of the susceptible, infected and recovered population with their standard deviation envelop by master equations, respectively.

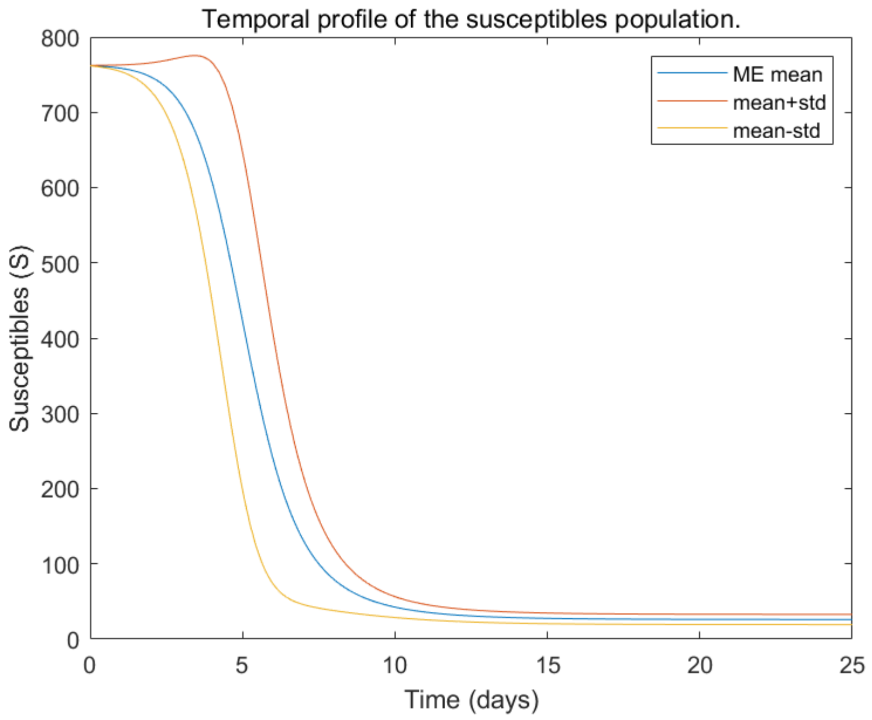


Figure 1: Temporal profile of the susceptible population with their standard deviation envelop by master equations.

2.1.3 Parameter sensitivity

I also evaluated parameter sensitivity to study the influence of the transition-intensity functions λ_1' and λ_2 and find the best-matched values to reproduce the figures in Section 2.1.2.

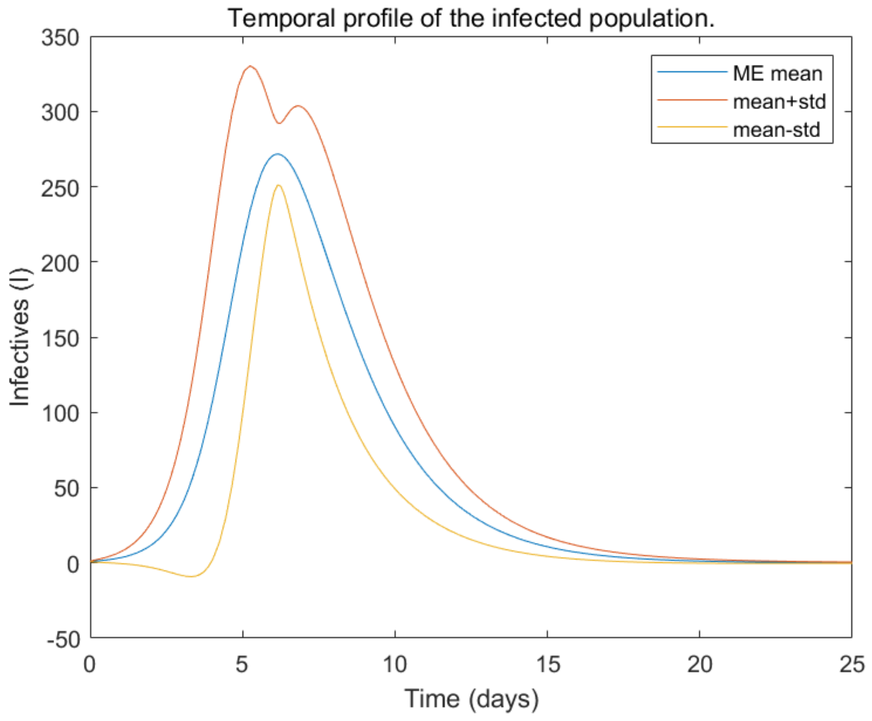


Figure 2: Temporal profile of the infected population with their standard deviation envelop by master equations.

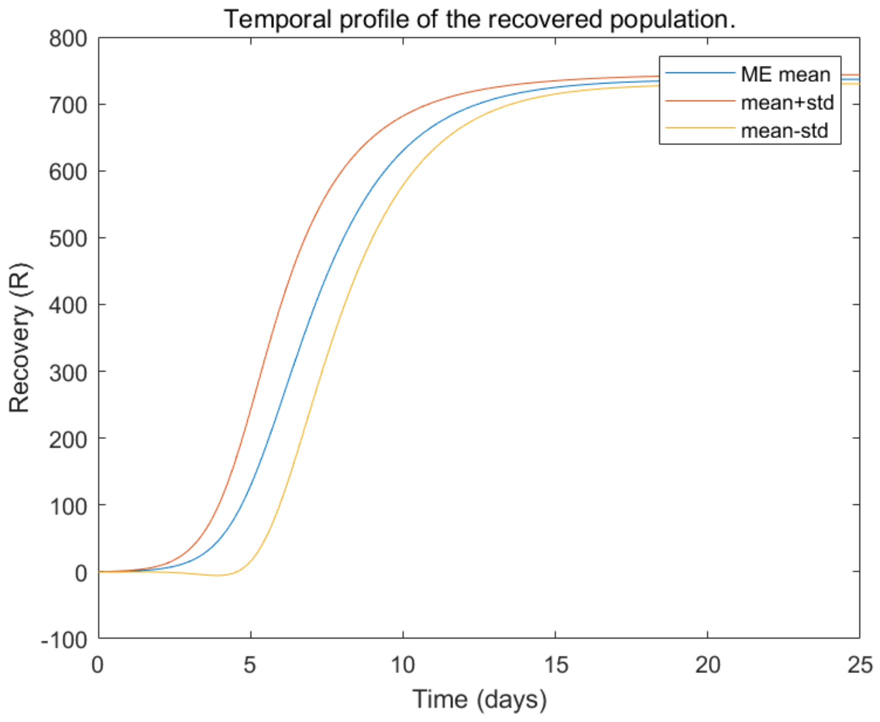
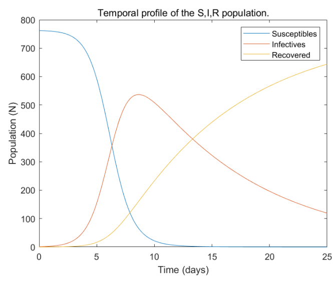
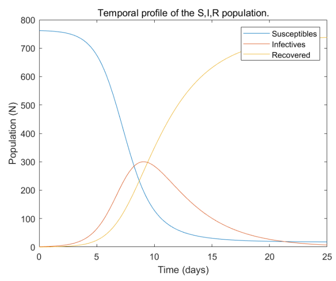


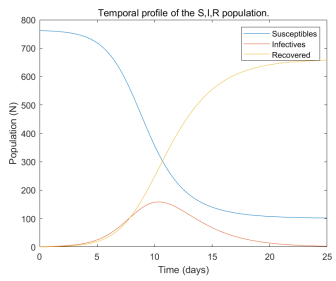
Figure 3: Temporal profile of the recovered population with their standard deviation envelop by master equations.



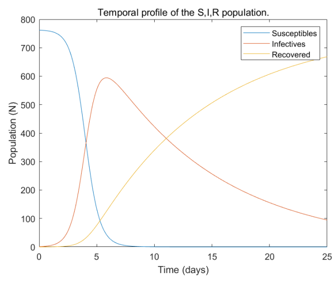
(a) $\lambda_1' = 0.002, \lambda_2 = 0.1$



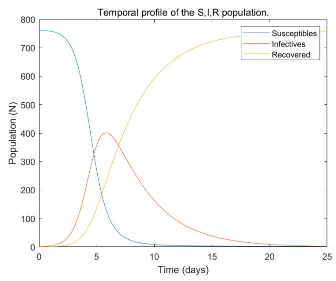
(b) $\lambda_1' = 0.002, \lambda_2 = 0.3$



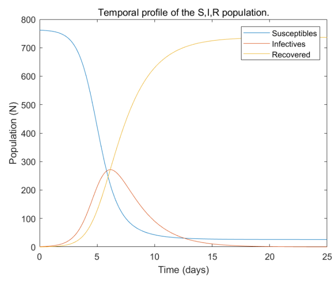
(c) $\lambda_1' = 0.002, \lambda_2 = 0.5$



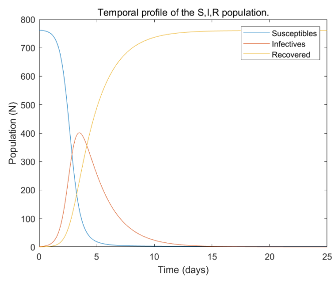
(d) $\lambda_1' = 0.003, \lambda_2 = 0.1$



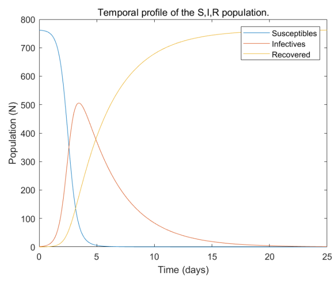
(e) $\lambda_1' = 0.003, \lambda_2 = 0.3$



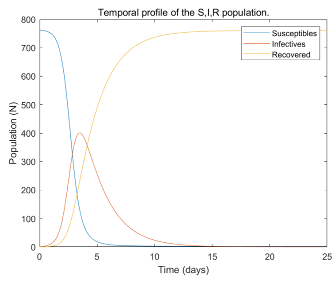
(f) $\lambda_1' = 0.003, \lambda_2 = 0.5$



(g) $\lambda_1' = 0.005, \lambda_2 = 0.1$



(h) $\lambda_1' = 0.005, \lambda_2 = 0.3$



(i) $\lambda_1' = 0.005, \lambda_2 = 0.5$

Figure 4: Parameter sensitivity analysis of the transition-intensity functions λ_1' and λ_2.

2.2 The matrix version

I also simplified the code by applying matrix multiplication. Due to the different numerical simulation codes, the reproduction is also a bit different compared to the plain version. The main modification of the implementation and the reproduction are given in this section.

2.2.1 Implementation

```
%......
%The main difference between these two versions.
%Other parts of the implementation remain the same.
 A = [ - theta * lambda1_, - phi * lambda1_, 0; ...,
      theta * lambda1_, (phi * lambda1_ - lambda2), 0; ...,
      0, lambda2, 0];
 B = [phi * theta * lambda1_, - phi * theta * lambda1_, 0; ...,
      - phi * theta * lambda1_, phi * theta * lambda1_ + theta *
         lambda2, - theta * lambda2; ...,
      0, - theta * lambda2, theta * lambda2];
 C = [v(1), c(1), c(2); c(1), v(2), c(3); c(2), c(3), v(3)];
 D = A * C + B;

 dydt = [dydt(1); dydt(2); dydt(3); dedt(1); dedt(2); dedt(3); ...,
      D(1,1); D(2,2); D(3,3); D(1,2) + D(2,1) - B(1,2); ...,
      D(1,3) + D(3,1) - B(1,3); D(2,3) + D(3,2) - B(2,3)];
 %......
```

2.2.2 Reproduction

The reproduced figures are shown below. These figures (Fig. 5, Fig. 6 and Fig. 7) are the temporal profiles of the susceptible, infected and recovered population with their standard deviation envelop by master equations, respectively.

2.3 Discussion

Firstly, the reproduced figures of the plain version and the matrix version are a bit different. In the temporal profile of the infected population, the 'mean-std' graph of the plain version will be below zero in the beginning of the simulation. In contrast, the matrix version will keep the graph stay above the time axis by a smaller standard deviation.

Besides, I also notice that the reproduced figures match the figures in Chen [2005] best when the parameters are given by $\lambda_1' = 0.003$ and $\lambda_2 = 0.5$. Also, after evaluating the parameters, I find that the growth trend of the infected population will be steep if the infecting rate λ_1' is large. On the contrary, the growth trend of the recovered population will be rapid if the recovery rate λ_2 is large.

Moreover, I find that there is a typpo in Fig. 1 of Chen [2005]. The coefficients of the equation on the second branch should be $\lambda_2(n_2 + 1)$.

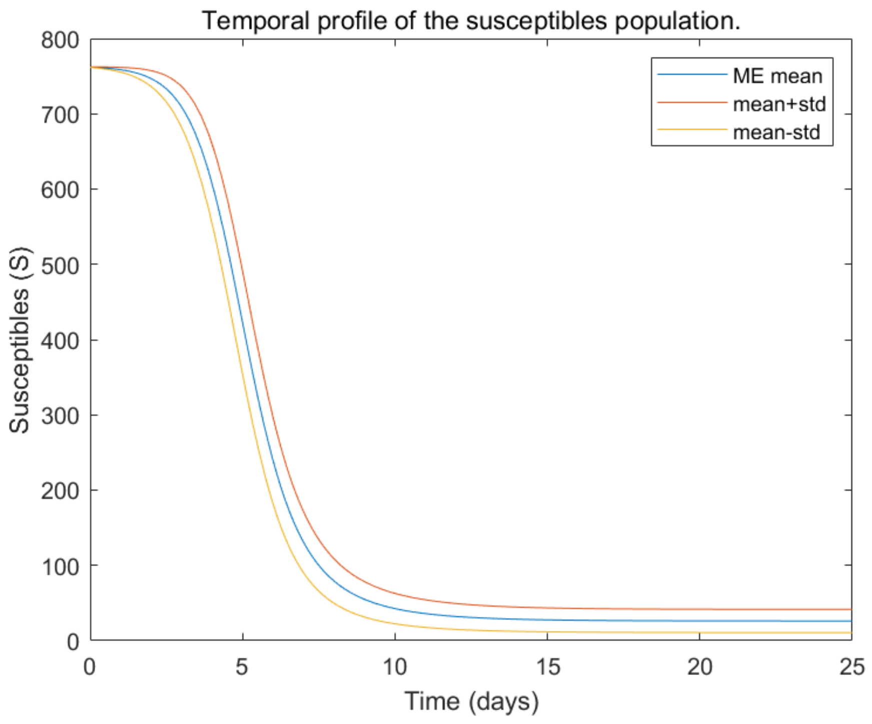


Figure 5: Temporal profile of the susceptible population with their standard deviation envelop by master equations. (matrix version)

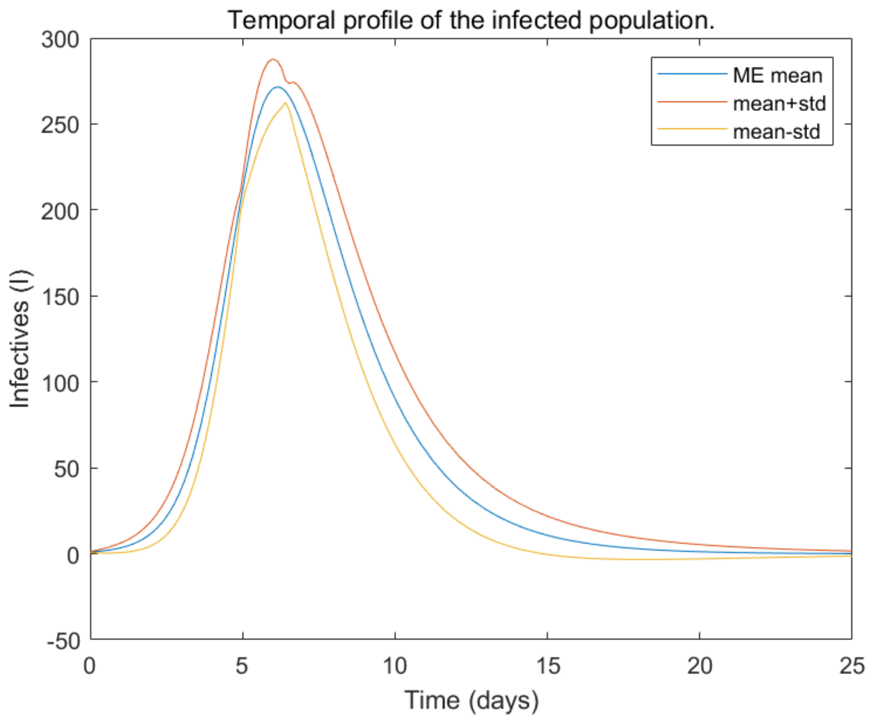


Figure 6: Temporal profile of the infected population with their standard deviation envelop by master equations. (matrix version)

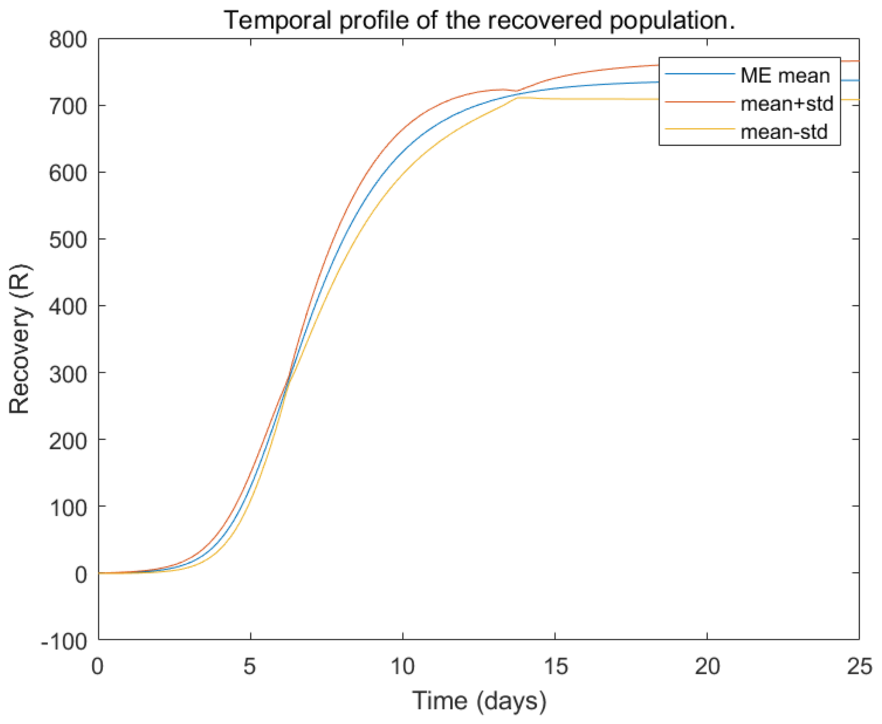


Figure 7: Temporal profile of the recovered population with their standard deviation envelop by master equations. (matrix version)

3 Simulation based on Monte Carlo

The Monte Carlo simulation is shown in detail. However, I summarize the pseudocode in Chen [2005] and tune the parameters to find the best-matched figures. I also conducted parameter sensitivity analysis with the Monte Carlo simulation.

3.1 Implementation

```
Initial_N = [762; 1; 0];
lambda1 = 0.003; lambda2 = 0.3;
Tf = 20;
Omega = 100;

time_list_list = {};
state_list_list = {};
for iter = 1:Omega
    N = Initial_N;

    simulated_time = 0;
    time_list = [];
    state_list = [];
    while simulated_time < Tf
        time_list = [time_list simulated_time];
        state_list = [state_list N];
        simulated_time = simulated_time - log(1 - rand(1)) / (N(1)*N(2)
           *lambda1 + N(2)*lambda2);
        Q = N(1) * N(2) * lambda1 / (N(1)*N(2)*lambda1 + N(2)*lambda2);
        if rand(1) < Q
            N(1) = N(1) - 1;
            N(2) = N(2) + 1;
        elseif N(2) > 0
            N(2) = N(2) - 1;
            N(3) = N(3) + 1;
        end
    end
    time_list_list = [time_list_list time_list];
    state_list_list = [state_list_list state_list];
end

collect_time = [];
for iter = 1:Omega
    collect_time = [collect_time time_list_list{iter}];
end
collect_time = sort(collect_time);

collect_state = zeros(3, length(collect_time));
for iter = 1:Omega
    state_list_list{iter} = interp1(time_list_list{iter}, transpose(
       state_list_list{iter}), collect_time, 'previous', 'extrap');
    collect_state = collect_state + transpose(state_list_list{iter} /
       Omega);
end

plot(collect_time, collect_state);
```

3.2 Reproduction and parameter sensitivity

I show the reproduction and parameter sensitivity analysis together because the temporal profiles of the susceptible, infected and recovered population are drawn in the same figure.

4 Conclusion

The validity of the stochastic model of nonlinear epidemiology is amply demonstrated by numerically calculating the evolution of population of all types and their fluctuations over time through two simulation algorithms, one based on the master equations and the other based on the event-driven Monte Carlo procedure. These two algorithms are implemented totally independently of each other but with the same set of system parameters, i.e. the transition-intensity functions. Hence, it is indeed remarkable that the two algorithms have yielded essentially identical results.

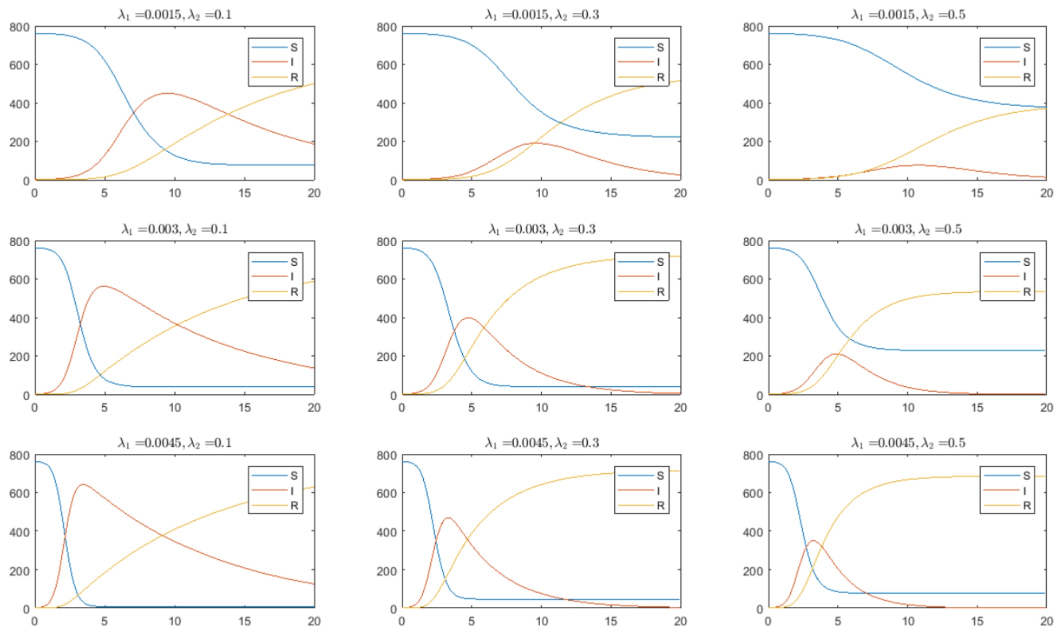


Figure 8: Reproduction and parameter sensitivity analysis. (Monte Carlo)

References

[1] Chen W Y and Bokka S., Journal of theoretical biology, 2005, 234(4): 455-470.

[2] Petzold, L.R., Hindmarsh, A.C., A systematized collection of ODE solvers, Report of Lawrence Livemore National Laboratory to the U.S. Department of Energy under Contract W-7405-eng-48, 1997, also available on web at http://www.netlib.org/odepack.

Appendix: The pseudocode

```
dEY1Y1 = -2 * theta * lambda1_ * EY1Y1 - 2 * phi * lambda1_ * EY1Y2 +
    phi * theta * lambda1_;
dEY2Y2 = 2 * theta * lambda1_ * EY1Y2 + 2 * (phi * lambda1_ - lambda2)
    * EY2Y2 + phi * theta * lambda1_ +
lambda2 * theta;
dEY3Y3 = 2 * lambda2 * EY2Y3 + lambda2 * theta;
dEY1Y2 = (phi * lambda1_ - lambda2 - theta * lambda1_) * EY1Y2 + theta
    * lambda1_ * EY1Y1 - phi * lambda1_
* EY2Y2 - lambda1_ * phi * theta;
dEY1Y3 = (-theta * lambda1_) * EY1Y3 + (-phi * lambda1_) * EY2Y3 +
    lambda2 * EY1Y2;
dEY2Y3 = (theta * lambda1_) * EY1Y3 + (phi * lambda1_ - lambda2) *
    EY2Y3 + lambda2 * EY2Y2 - lambda2*
theta;
```