A simple stochastic model for an epidemic – numerical experiments with MATLAB

Keng-Cheng Ang
kengcheng.ang@nie.edu.sg
National Institute of Education
Nanyang Technological University
1, Nanyang Walk, Singapore 637616
Singapore

Abstract

In this paper, we examine the use of a simple stochastic differential equation in the modelling of an epidemic. Real data for the Singapore SARS outbreak are used for a detailed study. The model is solved numerically and implemented on MATLAB, with further analysis and refinement. This article is built around several MATLAB programs and serves to provide a practical and accessible introduction to numerical methods for a stochastic model for epidemics.

1 Introduction

Mathematical modelling forms an important part of many tertiary courses in mathematics and engineering. In recent times, it has gained more attention and awareness in other fields such as bioengineering, economics, biology, epidemiology and the medical sciences. Traditionally, the techniques used and skills involved are confined to deterministic models based on concepts from algebra, vector calculus, regression, differential equations and so on [1].

Notwithstanding their popularity, one major shortcoming of deterministic models is their inability to include an element of uncertainty or noise. On the other hand, a purely empirical model developed from known data often lacks the predictive value that may be gained from the knowledge of the dynamics of the problem. Therefore, a more desirable model would be one that harnesses the best aspects of both approaches. This may be done by including some stochastic noise components into a deterministic model.

In this paper, we discuss the use of a stochastic differential equation (SDE) in the modelling of an infectious disease, using the outbreak of Severe Acute Respiratory Syndrome, or SARS, in Singapore in 2003 as a practical example for a more detailed study. The theory of SDEs is kept to a minimum intentionally, while practical implementations of the numerical solution are discussed in detail.

A complete understanding of SDE theory demands prerequisite knowledge and skills in advanced probability theory and stochastic processes. This paper does not aim to provide such theoretical background. Instead, the intent here is to demonstrate how an SDE may be used to simulate the progression of a disease outbreak, and to introduce a practical and accessible way in which one could experiment with an SDE model.

Nevertheless, it is acknowledged that there are readers who may wish to have a deeper understanding of SDEs. Those who are keen to learn more about SDEs may wish to refer to [2] and [3], both of which are comprehensive references for the subject. For details on the numerical treatment of SDEs, [4] is recommended.

In the next section, a model for the proliferation of a highly communicable disease based on an SDE is introduced. A numerical scheme, namely the Euler-Maruyama method, for approximating the solution of an SDE is then described. The scheme is then applied to the case of a SARS outbreak in Singapore and implemented on MATLAB. This is described in Section 3, where numerical simulations and their results are also briefly discussed. Section 4 examines some ways of modifying and refining the model, and results from more simulations are presented. In Section 5, a brief mention of some important issues surrounding such models is made.

All programs in this paper are written and tested on MATLAB R2006b. MATLAB is an ideal platform for numerical simulations of this nature because of the availability of a high level random number generator, graphics facilities and vector computation features. Program listings as well as video-clips of simulation runs may be downloaded at

`http://math.nie.edu.sg/kcang/ejmt0702`
or `https://ejmt.mathandtech.org/Content/v1n2p3/kcang.htm`.

2 A stochastic model for a SARS outbreak

A simple deterministic model for the spread of a highly communicable disease such as SARS is given by the ordinary differential equation [5]

$$\frac{dx}{dt} \; = \; \lambda x\,(n-x)\,, \tag{1}$$

where $x(t)$ is the number of infected and susceptible individuals at time t (in, say, days), λ is some constant of proportionality, and n is the total number of individuals in that community. Equation (1) is the well known logistic equation.

This model is based on the two-compartment “S-I” (or Susceptible-Infectious) epidemic model. For details on the derivation and justification of the S-I model, one may refer to [6] and [7].

While this model provides a predictive value to some extent, it does not take into account the stochastic nature (or “white noise”) that are often present in such situations. To introduce a stochastic component into the model, an SDE of the form

$$dX(t) \; = \; \lambda X(t)(n-X(t))\,dt + \mu X(t)\,dW(t), \quad X(0)=X_0, \quad 0 \le t \le T, \tag{2}$$

may be used [4]. Here, λ and μ are real constants, and $W(t)$ is a random variable representing a standard *Wiener process* or *Brownian motion.* The function in the first term on the right hand side of Equation (2) is known as the *drift* while that in the second term is the *diffusion.* It is clear that drift represents the deterministic portion of the model, while diffusion represents the stochastic component.

To solve the equation numerically, we employ the Euler-Maruyama (EM) method. The EM method is discussed in some detail in [8]. For the reader's convenience, the method is briefly described here.

First, we need a way to construct or simulate a Brownian motion. In numerical simulations, it is usual to consider a *discretized* Brownian motion, in which $W(t)$ is sampled at discrete t values. Setting $\delta t = \frac{T}{N}$ for some positive integer N, we have

$$dW_j \;=\; W_j - W_{j-1}, \quad j = 1, 2, \ldots, N, \tag{3}$$

where $dW_j \sim \sqrt{\delta t} N(0,1)$.

In MATLAB, the function `randn` may be used to generate a random number drawn from the $N(0,1)$ distribution. This feature is used to generate a discretized Brownian path for all the simulations here. In order to simulate a state of randomness, the command

```
randn('state',sum(i*clock))
```

is used. Since the initial state depends on the "value" of the clock at the time the command is executed, each time the program is run, presumably a new set of random numbers will be generated. If, instead, one wishes to reproduce a particular set of random numbers, the value of "`state`" in the command may be fixed by replacing "`sum(i*clock)`" by an integer. As an example, one could use `randn('state',123)` to generate a fixed set of random numbers. The random numbers so generated are then scaled by a factor of $\sqrt{\delta t}$ and stored as an array in the variable, `dW`.

The EM method may now be applied to Equation (2) over an interval $[0, T]$. The time interval is first discretized. We define $\Delta t = \frac{T}{L}$, for some positive integer L and let $\tau_j = j\Delta t$. Then, the EM method yields

$$X_j \;=\; X_{j-1} + \lambda X_{j-1}(n - X_{j-1})\Delta t + \mu X_{j-1}(W(\tau_j) - W(\tau_{j-1})), \quad j = 1, 2, \ldots, L, \tag{4}$$

where X_j is the numerical approximation to $X(\tau_j)$. Note that if $\mu = 0$, Equation (4) is simply the Euler approximation to the deterministic model given by the logistic equation (1).

For computational convenience, we set the stepsize Δt to be some integer multiple R of the increment δt; that is, $\Delta t = R \times \delta t$. Doing so will force the set of points $\{t_j\}$ on which the Brownian path is based to contain the set $\{\tau_j\}$.

In addition, the increment $W(\tau_j) - W(\tau_{j-1})$ needs to be computed on a general step in the EM method. This is given by

$$W(\tau_j) - W(\tau_{j-1}) \;=\; W(jR\delta t) - W((j-1)R\delta t) \;=\; \sum_{k=jR-R+1}^{jR} dW_k. \tag{5}$$

In all the programs in this article, this quantity is computed using the MATLAB command

```
winc = sum(dW((j-1)*R+1:j*R)),
```

which provides the values of the increments needed in Equation (4). In addition, for each run of the program, the number of sample paths is specified by the user. The final numerical solution is computed as an average of all admissable results.

3 Numerical Simulations

Table 1 below shows the cumulative number of individuals infected with SARS in Singapore in 2003 [9]. The outbreak began with one case on Day 0, and continued till Day 70 when a total of 206 cases was recorded.

Table 1: Cumulative number of individuals infected with SARS in Singapore from 24 February to 7 May 2003

Day	Number	Day	Number	Day	Number
0	1	24	84	48	184
1	2	25	89	49	187
2	2	26	90	50	188
3	2	27	92	51	193
4	3	28	97	52	193
5	3	29	101	53	193
6	3	30	103	54	195
7	3	31	105	55	197
8	5	32	105	56	199
9	6	33	110	57	202
10	7	34	111	58	203
11	10	35	116	59	204
12	13	36	118	60	204
13	19	37	124	61	204
14	23	38	130	62	205
15	25	39	138	63	205
16	26	40	150	64	205
17	26	41	153	65	205
18	32	42	157	66	205
19	44	43	163	67	205
20	59	44	168	68	205
21	69	45	170	69	205
22	74	46	175	70	206
23	82	47	179		

The MATLAB M-file `sars1.m` (see program listing in Appendix) simulates the progression of the outbreak based on the data in Table 1. In the data set, we have $n = 206$ and $X_0 = 1$. Unfortunately, in MATLAB, there is a minor inconvenience in that arrays begin with index 1 and not 0. To overcome this slight problem, we begin our simulation from Day 1. Thus, we set $X_1 = 2$ in the model, and discard X_0.

It is also neater, in terms of programming, to set $T = 1$ and scale the time interval of 70 days by letting $N = 70 \times R$, where $R = 2^{10}$ is an arbitrarily chosen positive integer. The other parameter that has to be specified is λ, and in all our simulation runs, we set $\lambda = 0.05$.

In this script (`sars1.m`), the user specifies the values of μ (which should be non-negative) and the number of sample paths in one run. As an example, if we choose $\mu = 0.25$ and use 500 paths for a simulation run, we would type `sars1(0.25,500)` at the command prompt.

Note that the deterministic model (that is, logistic equation) may be easily recovered by letting $\mu = 0$ and running the simulation with just one sample path. Figures 1 and 2 below show the graphical output from the program for $\mu = 0$ (with one path) and a sample case for $\mu = 0.25$ (with 500 paths) respectively, together with the real data for comparison.

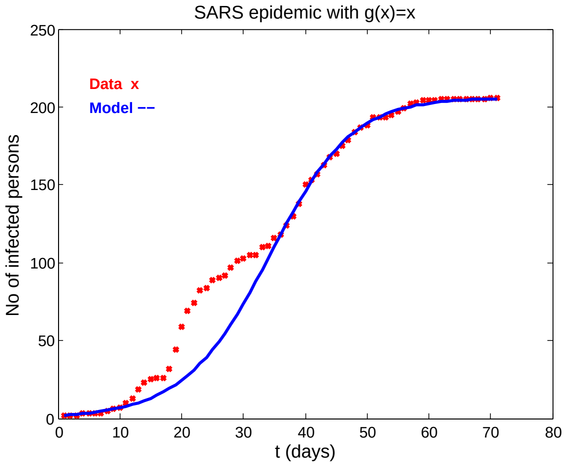


Figure 1: Graphical output from `sars1.m` with $\mu = 0$

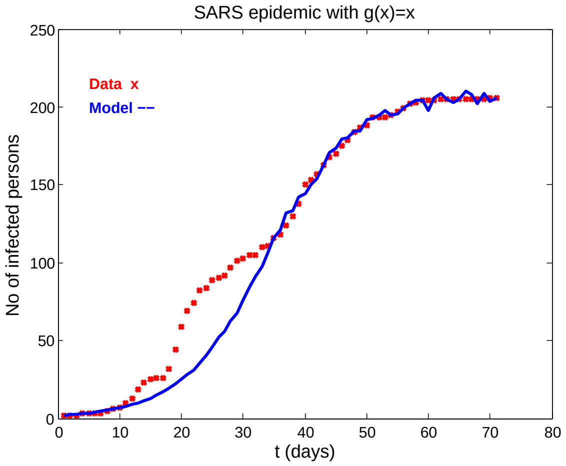


Figure 2: Graphical output from `sars1.m` with $\mu = 0.25$ and 500 sample paths

We define the "average" error, denoted by E, of a simulation run to be given by

$$E = \frac{\sqrt{\sum_{i=1}^{70} (\hat{X}_i - X_i)^2}}{70} \tag{6}$$

where $\hat{X}_i$ and X_i are values obtained from the data and the model respectively. Using this definition, it is found that $E = 2.0148$ for the case of the deterministic model (that is, when $\mu = 0$). For the case when $\mu = 0.25$, with 500 paths in the simulation run, the sample output shown in Figure 2 has an average error $E = 1.952567$. It is important to point out that this is only a sample and if one runs the program again with the same values of μ and number of trials, one could get a different output with a different value for E.

As can be seen from both graphs, the model does not seem to agree with the data very well for values of X between about 20 and about 100, although towards the latter part of the outbreak period, a better fit is obtained. If the outbreak can indeed be modelled by a logistic equation with some stochastic effects, then the data set seems to suggest that these effects are more significant in this region. That is, for $20 \leq X \leq 100$, there should be a greater presence of diffusion in the model. In physical terms, doing so may be viewed as having the model take into account a variety of factors (including human intervention, changes in disease transmission rates, and so on) that could have influenced the outbreak.

4 Modifying the model

In the SDE model given in Equation (2), the diffusion term takes the form of $\mu X(t)$. From Figure (2), it can be seen that the model exhibits fairly significant “noise” at higher values of X. This is to be expected since diffusion in this model is an increasing function of X.

From the actual data collected, however, it appears that for such an outbreak, stochastic effects are less crucial near the beginning and end of the outbreak episode. This could be due to the fact that at the beginning, as the number of infected individuals is small, the disease is only spreading gradually and has yet to progress to an epidemic state. Towards the end of the episode, perhaps due to intervention and control measures, the spread of the disease is brought under some control. It is therefore not unreasonable to assume that white noise should be more pronounced towards the middle portion of the outbreak episode.

There are many ways in which one could modify the diffusion term in the model to take into account the characteristic of the outbreak stated in the preceding paragraph. Here, we describe two ways.

4.1 Piecewise-defined $g(X)$

The SDE in Equation (2) may be written as

$$dX(t) \;=\; \lambda X(t)(n - X(t))\,dt + \mu g(X(t))\,dW(t), \tag{7}$$

where $g(X) = X$. To emphasize a greater significance of diffusion at some portion of the outbreak episode, the function $g(X)$ may be defined in some form of step function. In the script `sars2.m`[1], we define

$$g(X) \;=\; \begin{cases} 1.75, & 0 \le X < 20 \\ 2.75, & 20 \le X < 100 \\ 1.75, & X \ge 100 \end{cases} \tag{8}$$

so that when X falls between 20 and 100, a higher degree of diffusion is introduced into the model. There are, of course, other ways of specifying such a function, depending on how one wishes to model the relative impact of diffusion.

Figures 3 and 4 show the graphs of two typical simulation runs of this modified model with $\mu = 5$ with 500 sample paths, and $\mu = 8$ with 1000 sample paths respectively. In the cases presented here, the average errors are 1.334566 and 1.024202 respectively.

It is evident from the graphs in both Figures 3 and 4 that the model in this case seems to represent the data values more closely than in the preceding case where $g(X) = X$. Note that the values of μ are chosen arbitrarily, and they depend on the magnitudes of the “steps” in the definition of $g(X)$ in Equation (8).

[1] available from `http://math.nie.edu.sg/kcang/ejmt0702`

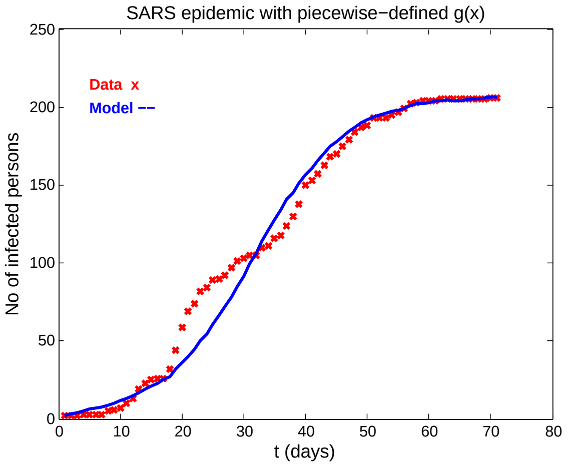


Figure 3: Graphical output from `sars2.m` with $\mu = 5$ and 500 sample paths

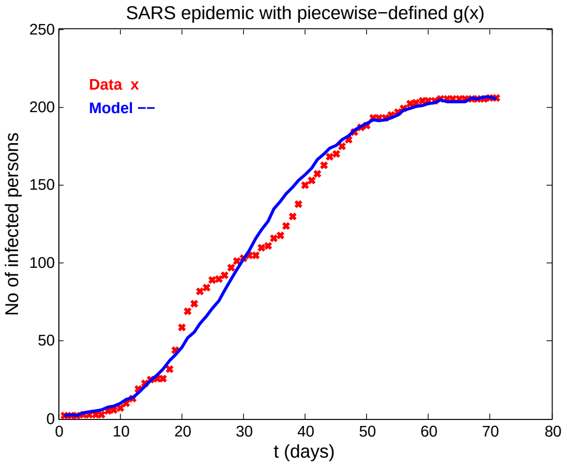


Figure 4: Graphical output from `sars2.m` with $\mu = 8$ and 1000 sample paths

4.2 Bell-shaped $g(X)$

Another way is to define $g(X)$ as a "bell-shaped" curve, centred at around the middle portion of the total number of infected individuals. To do so, we can define

$$g(X) \;=\; \exp\left(-k^2(X-103)^2\right), \tag{9}$$

where k is some constant. This provides a continuous function for $g(X)$ and by varying k, one could model the "spread" of influence of diffusion over the outbreak episode. Figure 5 shows the graphs of $g(X)$ for a few values of k. The script is now modified to include an additional parameter `k` and is stored as `sars3.m`.

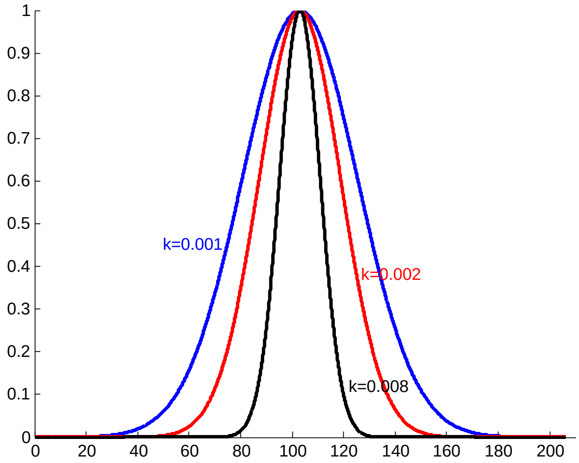


Figure 5: Graph of $g(X)$ for various values of k

Results from two typical simulation runs are shown in Figures 6 and 7. In Figure 6, we set $\mu = 20$, $k = 0.001$ and performed the simulation with 500 sample paths, obtaining an average error of 1.347614. Figure 7 shows the model results with $\mu = 20$ and $k = 0.002$, with 1000 sample paths. The average error in this case is 1.000880.

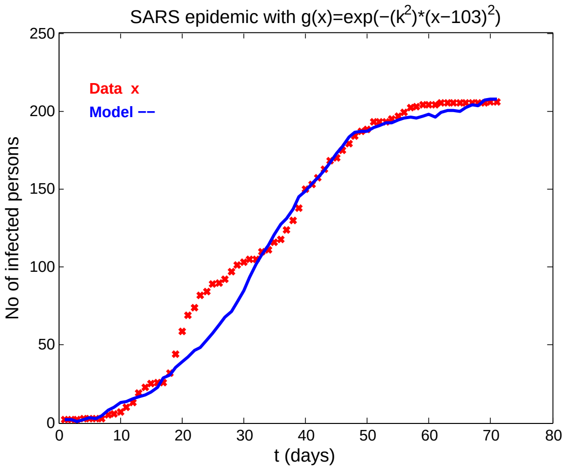


Figure 6: Graphical output from `sars3.m` with $\mu = 20$, $k = 0.001$, and 500 sample paths

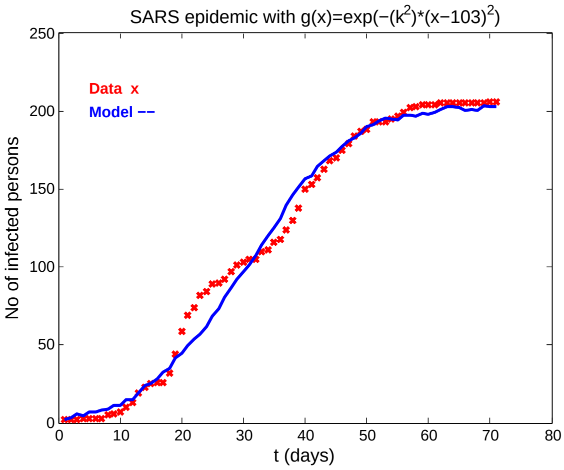
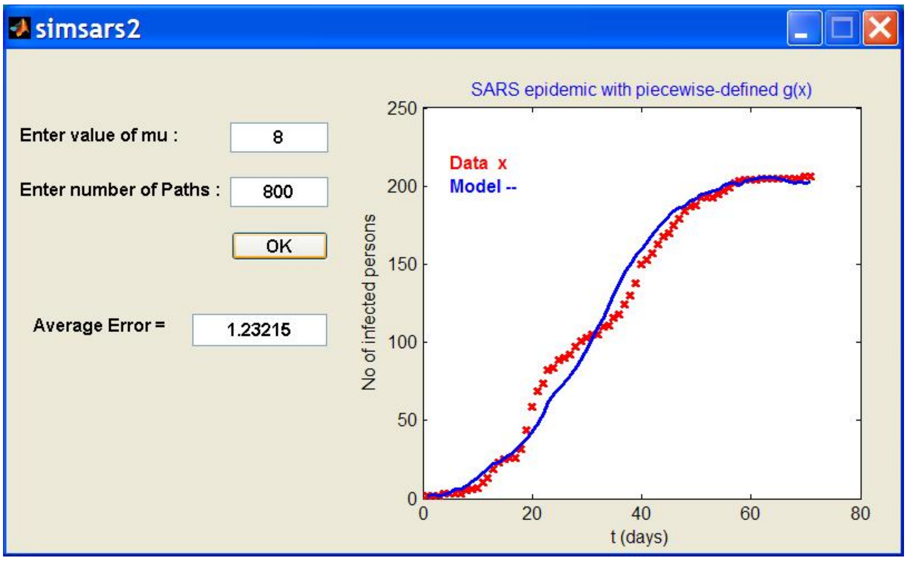


Figure 7: Graphical output from `sars3.m` with $\mu = 20$, $k = 0.002$, and 1000 sample paths

It is possible to modify the model in other ways to include other forms of diffusion. However, it is important to bear in mind that all these modifications are not simply attempts to obtain a better fit between the model and the data. The more pertinent issue here is obtaining a more realistic and reasonable representation of the physical situation, yet keeping the essence of the model intact.

In terms of implementation, it is perhaps more convenient to a user if a Graphics User Interface (GUI) version of the MATLAB programs to run the simulations in the modified models is available. This is also where MATLAB's GUI features may be capitalized. Two versions, `simsars2.m` and `simsars3.m`, have been produced and screenshots of typical runs are shown in Figures 8 and 9.



simsars2

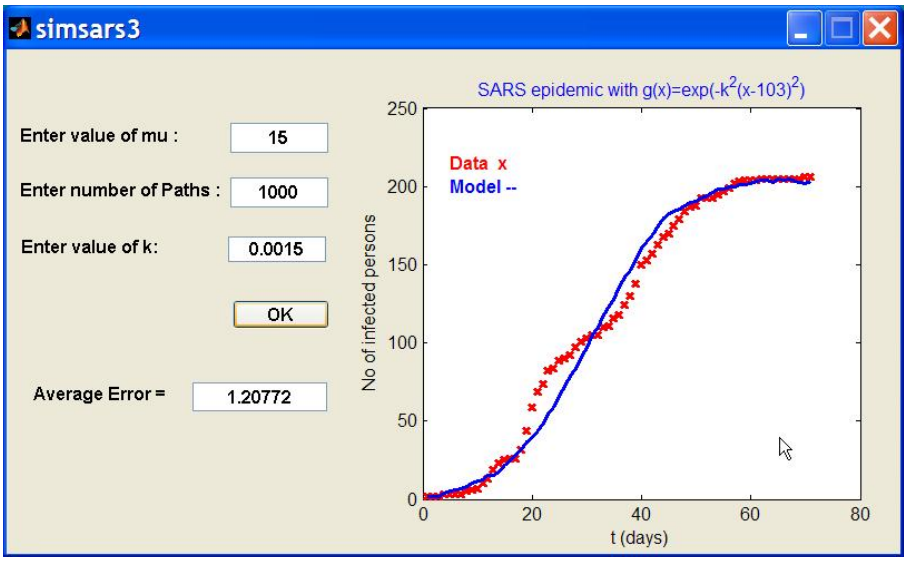
Enter value of mu : 8

Enter number of Paths : 800

OK

Average Error = 1.23215

Figure 8: Typical screenshot from GUI version, `simsars2.m`



simsars3

Enter value of mu : 15

Enter number of Paths : 1000

Enter value of k: 0.0015

OK

Average Error = 1.20772

Figure 9: Typical screenshot from GUI version, `simsars3.m`

5 Concluding Remarks

Modelling of epidemics and studies in epidemiology often involves systems of differential equations. Traditionally, the SIR or SEIR models are used in such studies (see [6], [7]). Although these are useful instruments, they are essentially deterministic in nature, and are not capable of capturing uncertainties or noise into the model.

In this paper, a simple stochastic model for the spread of a highly communicable disease is presented. Empirical data from the Singapore SARS outbreak in 2003 provided the models discussed a means to test and experiment with the models. In addition, the supplied MATLAB programs implementing the numerical scheme has made this modelling process involving SDEs more accessible.

There are some parameters (such as μ and k) that appear in the models. As in typical empirical models, such parameters do not have any physical significance other than being arbitrary constants chosen carefully through experiments to suit a particular set of data. In the case of the parameter μ, it represents the "amount" of noise in the model, and it is difficult to predict or estimate without the aid of data.

The S-I model discussed in this paper assumes a closed community. In other words, all the n individuals in the community do not leave the "system", and eventually all will be infected. Even in more sophisticated models (such as SIR and SEIR), such an assumption is sometimes necessary in order to reach a solution. The question is not how many susceptible individuals will become infectious (since, according to the model, all will be infected), but perhaps how quickly the epidemic is spreading and when will the spread slow down.

It is not the intention of this paper to construct an accurate stochastic model for a SARS outbreak. Detailed treatment of SDEs involves prerequisite knowledge in various areas of mathematics and it is neither practical nor wise to do so in this article. Instead, through simulation runs of the MATLAB programs, this paper serves to provide a way for readers to experiment with SDEs using a simple case of a disease outbreak.

References

[1] Bissell, C. and Dillon, C., 2000, Telling Tales: Models, Stories and Meanings, *For the Learning of Mathematics*, Vol. 20, No. 3, pp. 3–11.

[2] Gard, T.C., 1988, *Introduction to Stochastic Differential Equations*, Marcel Dekker, New York.

[3] Øksendal, B., 1998, *Stochastic Differential Equations*, 5th ed., Springer-Verlay, Berlin.

[4] Kloeden, P.E., and Platen, E., 1999, *Numerical Solutions of Stochastic Differential Equations*, Springer-Verlag, Berlin.

[5] Ang, K.C., 2004, A simple model for a SARS epidemic, *Teaching Mathematics and Its Applications*, Vol. 23, No. 4, pp. 181–188.

[6] Anderson, R.M. and May, R.M., 1991, *Infectious diseases of humans: dynamics and control*, Oxford University Press, Oxford.

[7] Anderson, R.M. and Nokes, D.J., 1991, Mathematical models of transmission and control, In Holland, W.W., Detels, R. and Knox, G. (eds), *Oxford Textbook of Public Health*, Oxford University Press, Oxford, pp. 225–252.

[8] Higham, D.J., 2001, An algorithmic introduction to numerical simulation of stochastic differential equations, *Society for Industrial and Applied Mathematics Review*, Vol. 43, No. 3, pp. 525–546.

[9] Heng, B.H. and Lim, S.W., 2003, Epidemiology and control of SARS in Singapore, *Epidemiological News Bulletin*, Vol. 29, pp. 42–47.

Appendix: Program Listing for `sars1.m`

```
% Euler-Maruyama method for a SARS epidemic model
% dx = lambda*(x*(n-x))dt + mu*x dW,  x(1)=x1
%
% KC Ang (April 2007)

function [X]=sars1(mu,Paths)

x1=2;                               % initial value (from data)
n=206;                              % final value (from data)
lambda = 0.05;                      % set parameter values
T  = 1;     R  = 2^10;   N = 70*R;
dt = T/N;   Dt = R*dt;   L = N/R;
X = zeros(1,L);                     % initialize solution vector

for i = 1:Paths
   x = zeros(1,L);
   xt = x1;
   randn('state',sum(i*clock));     % vary state
   dW = sqrt(dt)*randn(1,N);        % Brownian increments

   for j = 1:L
      winc = sum(dW((j-1)*R+1:j*R));
      xt = xt + lambda*xt*(n-xt)*Dt + mu*g(xt)*winc;
      x(j)=xt;
   end

   if sum(x)>0
      if i == 1
         X = x;
      else
         X = (X+x)/2;
      end
   end
end

% real data
sdata=[  2   2   2   3   3   3   3   5   6   7  10  13  19  23 ...
        25  26  26  32  44  59  69  74  82  84  89  90  92  97 ...
```

```
       101 103 105 105 110 111 116 118 124 130 138 150 153 157 ...
       163 168 170 175 179 184 187 188 193 193 193 195 197 199 ...
       202 203 204 204 204 205 205 205 205 205 205 205 205 206 ];
plot(sdata,'rx','LineWidth',2);
hold on
X=[x1,X];
plot(X,'b-','LineWidth',2);
hold off
xlabel('t (days)','FontSize',12);
ylabel('No of infected persons','FontSize',12,'Rotation',90);
title('SARS epidemic with g(x)=x','FontSize',12);
text(5,215,'Data  x','Color','r','FontWeight','bold');
text(5,200,'Model --','Color','b','FontWeight','bold');
%
err=0.0;
for i=1:70
    err=err+(X(i)-sdata(i))^2;
end
err=sqrt(err)/70;
fprintf('Average Error = %10.6f \n',err);

% -------------------------------------------------------------
% function g
function y = g(x)
y=x;
```