Stochastic simulation of epidemics

Level 2 module in "Modelling course in population and evolutionary biology"

(701-1418-00)

Module author: Roland Regoes

Course director: Sebastian Bonhoeffer
Theoretical Biology
Institute of Integrative Biology
ETH Zürich

1 About stochastic models

Many biological processes have an element of uncertainty to them. For example, humans reproduce some time between 15 and 40 years, but the exact time of reproduction for a given individual cannot be predicted. It is possible to incorporate this randomness or *stochasticity* in the mathematical formalism. This module deals with the simulation and analysis of stochastic models in the context of epidemics.

The way to incorporate randomness into a mathematical model is to formulate the terms as *probabilities* at which an event occurs, and not, as in deterministic models, as *rates*. In the stochastic SIR model, we will assume that epidemic processes, such as infection or death due to infection, but also every other process governing the demography are stochastic.

Important concepts in the context of stochastic processes are the *index space* and the *state space*. The index space is often the time or generation. The state space consists of the values that the variables of the model can attain. In our case, the state space will consist of positive integers from 1 to the total population size N. For the mathematical formulation and the simulation of stochastic models it is important to know whether the index and the state space are continuous or discrete. The model we are going to deal with belongs to the class of stochastic processes with continuous index space and discrete state space.

Lastly, I would like to emphasize that stochastic processes should not be confused with individual-based models. Individual-based models are, as the name suggests, models in which distinct individuals are simulated. Although they often involve stochastic processes, this is not part of their definition and needs not always be the case. In contrast, many stochastic models, such as the one we are going to simulate are population based. Even though our model will account for the fact that people cannot be divided (i.e. the state space will be discrete), we will not be able to track individuals.

2   The deterministic SIR model

Most of you will have encountered the susceptible-infected-recovered (SIR) model. One variant of this model is given in the following set of differential equations:

$$\frac{dS}{dt} = m(S + I + R) - mS - bSI \tag{1}$$

$$\frac{dI}{dt} = bSI - (m + v)I - rI \tag{2}$$

$$\frac{dR}{dt} = rI - mR \tag{3}$$

The variables mean:
$$
\begin{array}{ll}
S & \text{number of susceptible individuals} \\
I & \text{number of infected individuals} \\
R & \text{number of recovered individuals}
\end{array}
$$

And the parameters mean:
$$
\begin{array}{ll}
m & \text{host death rate} \\
b & \text{infection rate} \\
v & \text{pathogen-induced mortality rate} \\
r & \text{rate of recovery}
\end{array}
$$

The term that describes the birth of susceptible hosts, $m(S + I + R)$, ensures that deaths due to non-pathogen-related causes are balanced, and the total population $(S + I + R)$ remains constant over time, as long as there is no death due to the epidemic (expressed by $-vI$).

I have written down the equations of the deterministic model because from these equations the stochastic version can be easily infered. To write down the stochastic model formally is too cumbersome.

3   The stochastic SIR model

In the stochastic version of the SIR model, the continuous variables are replaced by discrete numbers, and the process rates are replaced by process probabilities. Let us denote the probability of the ith process by a_i; a will thus be a vector holding the probabilities of all possible

Process	Probability
Host birth	$a_1 = m(S + I + R)$
Death of susceptible host	$a_2 = mS$
Death of infected host	$a_3 = (m + v)I$
Death of recovered host	$a_4 = mR$
Infection	$a_5 = bSI$
Recovery	$a_6 = rI$

Table 1: Processes in the stochastic SIR model.

processes. There are six such processes in our stochastic SIR models, which are listed in Table 1.

For example, at time t, the probability that a new susceptible host is infected is:

$$P(\text{Infection}) = bSI = a_5 \tag{4}$$

If this happens the value of S jumps down by 1, and I jumps up by 1. Note that the probabilities so generated will typically not add up to one (the sum might even go above one): they give the *relative* probability of each process.

4   Simulation of the stochastic SIR model

Although conceptually our stochastic SIR model is more difficult than the deterministic one, it is not more difficult to simulate. This is due to the fact that the discrete variables are easier for computers to handle than continuous ones. To simulate the stochasticity of processes, however, requires the use of random number generators. Although random number generators are not straight-forward to develop on computers (which are inherently deterministic) almost every programming language has in-built ones nowadays.

The brute-force method to simulate our stochastic SIR model (with its continuous index space and discrete state space) would be to set very small time-steps in your simulation — so small that, at most, a single process happens at a given step. This small time step would be necessary to avoid problems arising from two or more processes happening at the same time step. The problem with this brute force approach is that in most time steps nothing will happen. This is not efficient. Fortunately, there are much cleverer algorithms. The following subsection introduces the so-called *Gillespie algorithm* which is a very efficient, but still accurate way to simulate stochastic models.

4.1   Gillespie algorithm

The idea of the Gillespie algorithm is that one first determines **when** something happens next. Suppose the current time is t. The time $t+\tau$ at which something happens next is an exponentially distributed random number scaled by the sum of all process rates, $\sum_i a_i$:

```
tau <- rexp(1, rate=sum(a))
```

Look up `?rexp`.

Then, the Gillespie algorithm determines **what** happens next. This is done by drawing a process randomly from all possible processes according to their respective probabilities. This can be done easily in `R` by drawing the index of the next process with a weighted sample command[1]:

```
sample(length(a),1,prob=a)
```

When we have determined which process happens, we can update the variables (the so-called *state* of the system). Then we iterate this process as long as we want.

4.2   Rudimentary R-script

We supplied a starting script with a skeleton of a Gillespie algorithm for the stochastic SIR model. Please fill in the missing commands and try to make it work:

```
# set parameters
parms=c(m=1e-4,b=0.02,v=0.1,r=0.3)
initial=c(S=50, I=1, R=0)
time.window=c(0, 100)

# initialize state and time variables and write them into output
state <- initial
time <-  time.window[1]

# define output dataframe
output <- data.frame(t=time,
                     S=state["S"], I=state["I"], R=state["R"],
                     row.names=1)

# define how state variables S, I and R change for each process
processes <- matrix(0, nrow=6, ncol=3,
                    dimnames=list(c("birth",
                        "death.S",
                        "infection",
                        "death.I",
                        "recovery",
                        "death.R"),
                        c("dS","dI","dR")))
```

---

[1]Check `?sample` to see how it handles a probability vector that does not add up to one—which is impertinent behaviour for a total probability.

```
<...>

# process probabilities
probabilities <- function(state){
  <...>
}

while(time < time.window[2] & state["I"]>0){

  # calculate process probabilities for current state
  <...>

  # WHEN does the next process happen?
  <...>

  # update time
  <...>

  # WHICH process happens after tau?
  <...>

  # update states
  <...>

  # write into output
  output <- rbind(output,c(time,state))
}

output
```

This script is supplied in a file called **start_stochSIR.r**. Once you have completed this script and it works you should plot a few epidemics. Later on, when you are working on the exercises you will want to run the algorithm repeatedly with different parameters. An efficient way to do this is to write a function that includes the algorithm, and then you only need to call this function with the appropriate parameters from the main body of your program. See the "functionized" version of the incomplete starting script **start_stochSIR_f.r** on the webpage.

5 Exercises

5.1 Basic exercises

Eb1. **Extinction.** One of the main aspects in which deterministic and stochastic models differ is extinction. In the deterministic SIR model, an epidemic never truly goes extinct in a

limited time frame because the number of infected hosts declines exponentially and reaches zero only at infinity. To work around this shortcoming, in a deterministic framework an epidemic is said to go extinct if it has a negative growth rate. In contrast, in our stochastic SIR model an epidemic can become extinct in a more direct sense, i.e. the number of infected hosts can become zero without waiting forever.

As you may have learned in previous courses, in a deterministic model an epidemic will go extinct (according to the definition above) if the basic reproduction number, R_0 of the infection is less than one. We will test if this is also true in our stochastic model. The basic reproduction number in our model is given by:

$$R_0 = \frac{bN}{m + v + r} \tag{5}$$

Here N denotes the initial number of hosts in the simulation.

Does the basic reproduction number also divide the dynamics into extinction and non-extinction in our stochastic SIR model?

To investigate this question, we could vary the R_0 by changing the value of the virulence parameter v, and check in how many cases the epidemic goes extinct. To get accurate results you will have to run the epidemic at least 100 times for each value of the virulence parameter. In these multiple runs of the epidemic you should keep the initial conditions the same (`initial=c(S=50, I=1, R=0)`), and only vary the parameter `v`. Please write a script which estimates the extinction probability for different values of R_0.

Eb2. **Compare deterministic and stochastic dynamics.** To further compare deterministic and stochastic models, we provide a script for the deterministic SIR model in the file `SIR-determ.R`:

```
sir <- function(t,x,parms){
        S <- x["S"]
        I <- x["I"]
        R <- x["R"]
        with(as.list(c(parms)),{
            ds <- m*(S+I+R) - m*S - b*I*S
            di <- b*I*S - (m+v+r)*I
            dr <- r*I - m*R
            der <- c(ds,di,dr)
            list(der)
        })
}


SIR.determ <- function(parms, initial, time.window){
  require(deSolve)
  times <- seq(time.window[1], time.window[2], length=101)
  as.data.frame(lsoda(initial, times, sir, parms))
}
```

This script produces output of the same form as `SIR-stoch.R`. Plot the dynamics for the default parameters of `SIR.stoch`.

The output from a single run of the stochastic SIR model is never the same. Therefore, to compare the dynamics of the stochastic SIR model to the deterministic one we will have to average the outcome of many runs of the stochastic SIR model. Thus, first run `SIR.stoch` 100 times and save the results in a data frame. One column of the data frame should contain the number of the run, and should be called `run`. Save this data frame as `sims.100`:

```
sims.100<-cbind(run=1,SIR.stoch(parms,init,tw))
for(r in 2:100){
  sims.100 <- rbind(sims.100,cbind(run=r,SIR.stoch(parms,init,tw)))
}
rm(r)
```

One technical problem with the data frame that contains multiple runs of `SIR.stoch` is that the times at which something happens are irregular and do not coincide across runs. (This is a consequence of the Gillespie algorithm.) To work around this problem, we supply the function `intpol` which can be found in the file `intpol.R`.

```
intpol <- function(sim, var, t){
  var.out <- vector("numeric", length=length(unique(sim$run)))
  for(r in unique(sim$run)){
    ind <- sim$run==r
    var.out[r] <- stepfun(sim[ind,"t"],c(NA,sim[ind,var]))(t)
  }
  var.out
}
```

This function takes as input a data frame `sim` with the results of multiple runs of `SIR.stoch` and puts out a vector containing the interpolated value of a variable ("S", "I" or "R") at time `t` for each run of `SIR.stoch`.

Now we can start comparing the dynamics of the stochastic with that of the deterministic SIR model. First define a vector with times at which we want to calculate the means of "S", "I" and "R" in `sims.100`, for example:

```
t.seq <- seq(1,100,1)
```

Now source the definition in `intpol.R` and calculate the means of `S` using the command `sapply`:

```
source("intpol.R")
S.seq <- sapply(t.seq, function(t) mean(intpol(sims.100,"S",t)))
```

Do this also with `I` and `R` and then plot the results onto the same plot as the results of `SIR.determ`.

What is the difference? Can you explain this difference?

Eb3. **More on extinction**: How does the extinction probability change if you seed the epidemic with more than one infected individual?

5.2 Advanced exercises

Ea1. **Distribution of S, I and R**

If you wanted to fit the deterministic SIR model to data you could minimise the sum squares of the deviations of the model prediction from the data. By doing this you would assume that the errors are normally distributed. But is this a reasonable assumption?

There is a lot of variability across runs of our stochastic SIR model. Rather than a nuisance, this is actually a virtue of stochastic models because the variability tells you about the error distribution (at least partially) that is necessary for statistical inference[2].

To look at the distribution of the variables plot histograms of `S`, `I` and `R` at different time. Use the data frame `sims.100` you have previously generated. Histograms are plotted with the command `hist`.

To check the normality, we can produce a so-called *Quantile-Quantile plot*, or *Q-Q plot*. On such a plot, normally distributed data will fall approximately onto a straight line. In R the Q-Q plot is implemented as the function `qqnorm`.

Ea2. **More on the dynamics**: Is there also a difference between deterministic and stochastic dynamics in a parameter range in which there is almost no extinction (eg $v = 0$)?

Ea3. To move from understanding the technicalities of the model to actually using it to simulate interesting epidemiological scenarios, please, refer to the Level 1 module on 'SIR models of epidemics'. You can use the stochastic model to answer any of the questions posed in the frame of the deterministic model. Stochastic effects are important if there are events that are rare. Can you think of further situations where this might be relevant?

---

[2] *Statistical inference* is the area of statistics that deals with the fitting of models to data and the estimation of parameters.