Numerical methods for studying disease epidemics

Suzanne O'Regan (with modifications and updates from Chris Dibble)

Monday, June 08, 2015

Introduction:

Infectious diseases cause a substantial global burden of morbidity and mortality. Understanding infectious diseases involves multiple facets including medicine, microbiology, genomics, immunology and vaccine and drug development. However, none of these approaches address important questions at the population level.

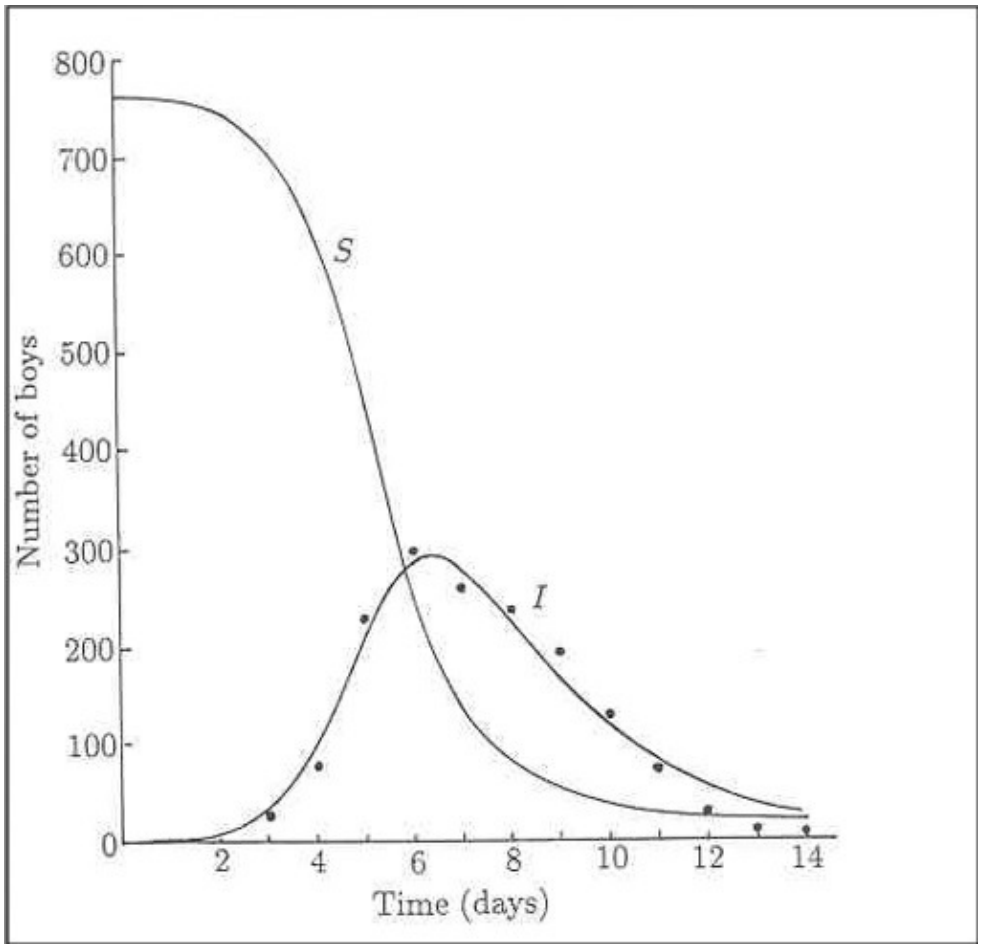


Figure shows the number of children confined to bed as a result of contracting influenza in boarding school. Modeling can help us answer the following basic questions arising from the epidemic: - What determines invasion of the bug?

- What does the growth rate tell us?
- Why does the epidemic turn over?
- Why did the pathogen go extinct?
- How many children will become infected through the course of the epidemic?

If we are able to predict the epidemic curve with a model, we can also evaluate the effects of different control measures. We can answer questions such as, is it better to randomly vaccinate children in the school, or target a

particular age group?

What is a mathematical model?

We distinguish between mathematical and statistical models. *Mathematical* or *dynamical* models are abstract models that uses mathematical language to describe the behavior of a system. The epidemiological processes (e.g., infection of susceptible individuals, recovery of infected individuals) are represented by equations. *Statistical* models attempt to describe relationships between observed quantities and independent variables. Otto and Day (1) give a nice overview of the differences between these two approaches. Models can help us to understand nature and to make predictions. In this workshop, we will show you how to formulate simple mathematical models of epidemics and how to solve these models in R.

Why use mathematical models? A mathematical model can give us key insights into how various forces act to change a system over time. For example, in a population of hosts for a parasite, birth of susceptibles into a population, transmission of the infection, and recovery from infection, are processes that lead to changes in the number of susceptible, infectious, and recovered hosts through time. A model allows us to keep track of quantities that change over time (*variables*). The *dynamics* of a system is the pattern of changes that occur over time. In reality, models are a convenient way to simplify the world.

There are innumerable things to keep in mind when thinking about models. Some are, in no specific order:
- All models are wrong, but some are useful (attributed to George Box)
- There is no 'perfect' model - There are often trade-offs between model realism and generality - Models should have a purpose - you should always be able to say *why* a particular model was created, or what question it answers, or what insight it generates about the real-world

Model types

Mathematical models can be deterministic or stochastic. A *deterministic model* assumes that the future is entirely predicted by the model. A *stochastic model* assumes that random events affect the system, in which case the model can only predict the probability of various outcomes in the future.

A key decision when developing a model is how to treat time.

Discrete time models track changes to variables in discrete time steps. *Continuous time models* allow variables to change at any point in time (i.e., time is represented by a continuous axis).

Which type of model to use depends on the biological question. Continuous variables are often used if the variables take on large enough values that treating them as continuous introduces very little error to the results. Another justification for their use is that it is typically easier mathematically to treat variables as continuous – calculus can be used to analyse the model.

Typically, discrete time models are written as *recursion equations* that describe the value of variables at the next time step in terms of a function of it at the current time step, e.g.,

$$N_{t+1} = ``\text{some function of } N_t",$$

where N_t is the value of the (discrete) variable (e.g., population size) at the current time t.

Continuous time models are written as *differential equations* that describe the *rate* at which variables changes over time. Many of the core theories of epidemic propagation are expressed as a *system* of differential equations known as a *compartmental model*. Suppose that $N(t)$ is the value of a (continuous) variable at time t. Then we may express its rate of change over time as a differential equation:

$$\frac{dN}{dt} = ``\text{some function of } N(t)", (1)$$

Often, the right hand side of the differential equation takes the form

$$\text{some function of } N(t) = \text{rate of increase of } N(t) - \text{rate of decrease of } N(t).$$

This side of the equation describes how the various biological forces change the value of the variable $N(t)$. All sources of change in $N(t)$ are included.

Here time t is the *independent variable* and population size is the *dependent variable*. If we plot $N(t)$ as a function of time, then the slope of the curve would be $dN(t)/dt$. If the variable is increasing over time, the slope is positive; if it is decreasing, then it is negative. If the magnitude of $dN(t)/dt$ is small, then $N(t)$ changes slowly over time, whereas if it is large, it changes rapidly.

We can use this equation to infer the value of $N(t)$, i.e., to obtain a *solution* of the differential equation. A solution is a function that satisfies the differential equation. If we can find this function, then we have *solved* the differential equation. Solving a differential equation allows us to deduce *long-term behavior* – to answer ``what will happen in the future?".

A simple mathematical model

In many biological systems, differential equations arise because the rates of change of a process of interest themselves change as the system dows. A simple differential equation that describes, for example, the rate of change of the size of a population of *E. coli* bacteria, is

$$\frac{dN}{dt} = rN. \quad (2)$$

Here r is the instantaneous per-capita growth rate. Equation 2 is an example of an *ordinary differential equation* because it has a single independent variable (here, it is time t).

We would like to know the solution of equation 2, or the value of the dependent variable $N(t)$ at time t. How can we obtain this from equation 2?

Typically when we talk about solving ordinary differential equations we mean solving an *initial value problem*: We want to start with a condition on the state variables (the *initial condition* – the value of each variable at time $t = 0$) and inquire about the future values of the state variables as the system evolves over time.

This requires performing an integration. For example, we can solve for $N(t)$ in equation (2) by taking the integral of both sides with respect to t, i.e.,

$$\int \frac{dN}{dt} dt = \int rN(t) dt.$$

Taking the integral is shorthand for adding all these pieces up for each small interval of time dt.

Sometimes (as is the case for equation (2)), the integration can be done by hand. Often it is very difficult or even impossible to obtain an analytical solution to a differential equation. But this is not a problem – we can use the computer to calculate the right hand side numerically. In general, this can be a tricky business. Fortunately, this is a well studied problem in numerical analysis and (when the equations are smooth, well-behaved functions of a relatively small number of variables) standard numerical integration schemes are available to approximate the integral with arbitrary precision. Particularly, R has a very sophisticated ordinary differential equation solver, which (for many problems) will give highly accurate solutions.

To solve an ordinary differential equation (ODE), the basic numerical approach used by ODE solvers is to consider the change in each variable during a very small time interval Δt, i.e., approximate the derivative by a discrete time difference equation,

$$\frac{dN}{dt} \approx \frac{N(t+\Delta t) - N(t)}{\Delta t}.$$

Rearrange this equation to obtain

$$N(t+\Delta t) = N(t) + \frac{dN}{dt}\Delta t$$

and then iterate to get a numerical solution for $N(t)$. This is Euler's method, but it is not efficient because very small time intervals are necessary to obtain accurate predictions when $N(t)$ is changing rapidly. Numerical algorithms, such as the 4th order Runge-Kutta algorithm (implemented in R), take advantage of Euler's approximation to obtain an approximate solution by solving a sequence of (tractable) linear approximations at smaller and smaller step sizes until a specified tolerance is achieved. The LSODA adaptive step size solver in R is a powerful algorithm because it can automatically adjust its step size according to how rapidly the solution is changing.

Solving ordinary differential equations with LSODA

In R, numerical integration of ordinary differential equations (ODEs) is readily performed using the package deSolve. Particularly, the function LSODA is useful since it automatically selects the optimal solving algorithm based on numerical performance.

To solve an ode, we need to provide an initial condition. To use the numerical integration functions in the deSolve library, type in `require(deSolve)`

The LSODA function takes in four (non-optional) arguments

```
out <-  lsoda(nstart, times, func, params)
```

nstart: a vector of initial conditions. There is one entry for each variable in the differential equation.
times: a vector of times for which you want to list the values of all variables you are finding with the differential equation.
func: A function representing the right hand side of the system of differential equations
params: A vector of parameter values. These are the values of any constants that appear in the differential equation. You can choose the order in which these constants appear, but the order you choose must be consistent with the definition of func.

The function `func` must be defined as `FunctionName<-function(t, y, params)`

The `t` argument represents the current time point of the integration. The `y` argument is an R vector of all the variables in our differential equation. The values of the entries in `y` are what get updated as time moves forward. Lastly, the `params` argument is a vector of parameter values, exactly the same vector as the `params` argument of LSODA.

To illustrate solving ODEs using LSODA, we return to equation (2). Starting from a single individual cell, how many cells will there be in 2 hours, if the per-capita intrinsic growth rate is 0.75 per hour and growth obeys equation (2)? Thus we are interested in solving:

$$\frac{dN}{dt} = 0.75N, \quad N(0) = 1,$$

for the population size $N(t)$ in all times in the interval $0 \leq t \leq 2$. The statement `$N(0) = 1$' is the initial condition—it means that there is a single individual in the population initially.

In this example, the only variable in the problem is N. The value $r = 0.75$ is a constant, and thus a *parameter*.

The ODE solver needs to know the right-hand sides of the ODE. We give it this information as a function (sub-routine). Note that the form of the arguments and output of this function must exactly match what is expected by the LSODA routine. Thus, for instance, the time variable `t` must be the first argument even if the function is *time-invariant* so that `t` is neglected in the calculation. Here we write a function to return the derivatives of the exponential growth model.

The first step is to write `func`, the function that represents the right hand side of the differential equation. We will call this function `ExponentialGrowth` because the solution of this model is an exponential function.

```
ExponentialGrowth <- function(t, y, params) {

  # Computes the rate of change of the variable y over time
  #
  # Args:
  #   t: vector of time points in the integration
  #   y: vector of variables in the differential equation
  #   params: vector of parameter values
  #
  # Returns:
  #   The rate of change of the variable y.

  N<-y #create local variable N

  with(as.list(c(params)), {

    #this argument to "with" lets us use the variable names

    dN <- r*N     # Right hand side of the differential equation
    res <- c(dN) #combine results into a single vector dx
    list(res) #return list of results

  })

}
```

Note: `with` function makes the parameters `params` available to the expressions in the brackets, as if they were variables. One could achieve the same effect by, for example, `dN <- params["r"]*N`.

The R variable `res` holds the “returned vector,” the vector of right hand side values, which consists of exactly one value in this simple case.

We now state the times at which we want a value of `N`:

```
times <- seq(0,2,by=0.5) #function seq returns a sequence
```

Next we assign some value to the parameter `r`:

```
params <- c(r=0.75) #function c "c"ombines values into a vector
```

Finally we specify the initial condition, i.e., the value of the state variable `N` at the beginning of the simulation:

```
Nstart <- c(N=1) #initial conditions
```

We are now ready to use LSODA to solve this differential equation.

```
#convert matrix to data frame
source("D:/OneDrive - University of Georgia/OneDrive - University of Georgia/UGA Non-AERO/REUWorkshop/REUWorkshop/REUworkshopworkedexamplesCD.R")
```

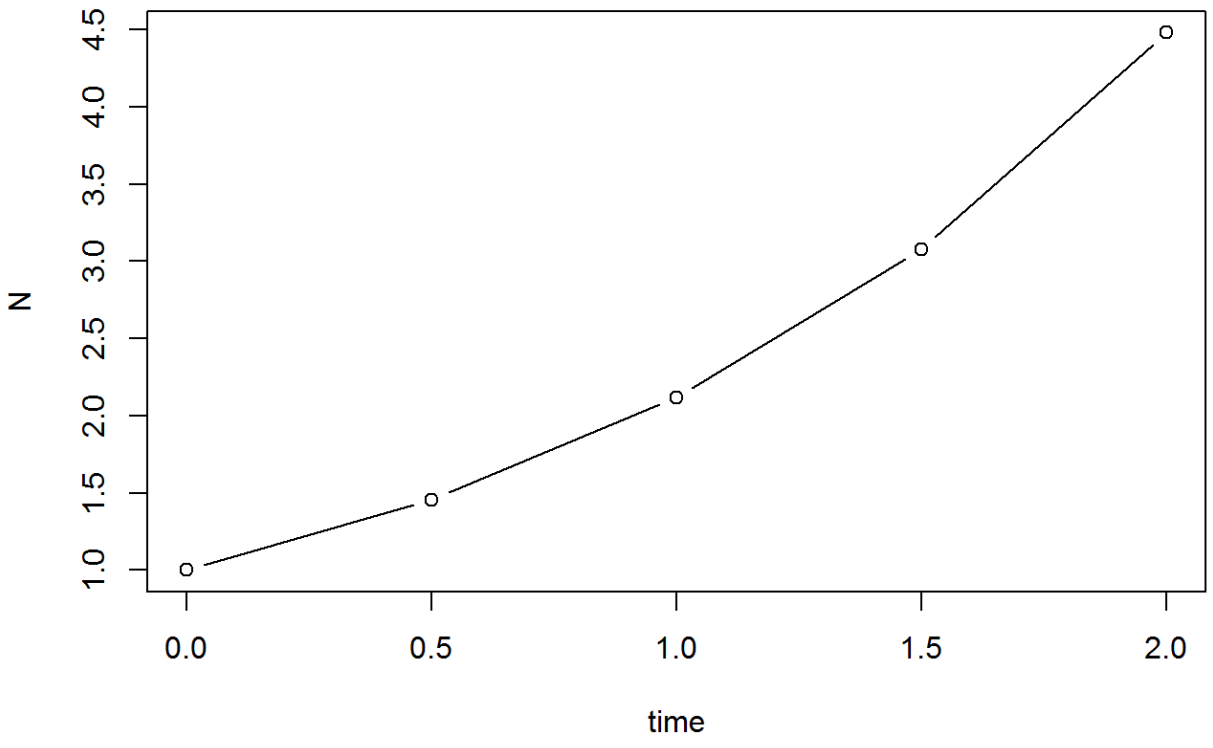
```
## Warning: package 'deSolve' was built under R version 3.1.3
```

```
out <-  as.data.frame(lsoda(Nstart, times, ExponentialGrowth, params))
head(out)
```

```
##   time        N
## 1  0.0 1.000000
## 2  0.5 1.454994
## 3  1.0 2.117003
## 4  1.5 3.080223
## 5  2.0 4.481700
```

The differential equation solver LSODA returns an R matrix, which we have converted to a data frame. The first column represents the `times` vector and the second column is the solution `N` corresponding to those times. We can plot the result by typing

```
with(out,plot(N~time,type="b"))
```



Exercise 1. What does the population look like after 10 hours?

Exercise 2. Solve the initial value problem

$$\frac{dN}{dt} = -0.5N, \quad N(0) = 100,$$

for times in the interval $0 \leq t \leq 10$. Plot the solution.

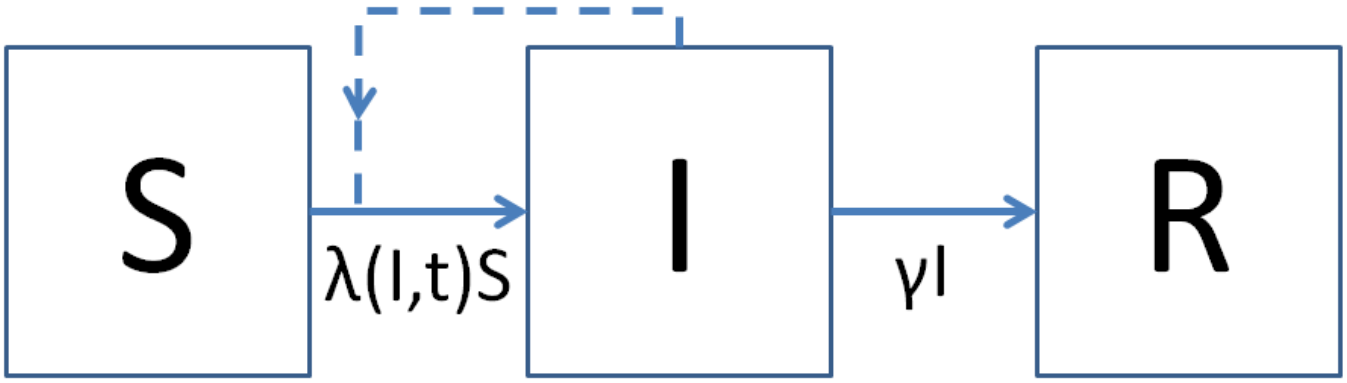
The SIR model

The classical SIR compartmental model tracks the proportions of the population in each of three classes (susceptible, denoted by $S(t)$, infectious ($I(t)$) and recovered ($R(t)$)). Here the variables of interest are susceptible, infectious and recovered individuals. As the epidemic progresses, these quantities can change over time.

In a *demographically closed* system, flow out of one class must enter another class, giving rise to a *conservation property*: the population size does not change over time. Thus no births, deaths or migration events occur. Hence, the proportions of each class sum to one, i.e., $S + I + R = 1$. The key assumptions of the SIR model are:

- Susceptibles become infectious at a rate $\lambda(I, t)$ per susceptible individual per unit time, leading to a total flow $\lambda(I, t)S$ from the susceptible to the I class. The function $\lambda(I, t)$ is called the *force of infection*. It is the per-capita rate at which susceptibles contract the infection.
- Infectious individuals recover at a rate γ per infectious individual per unit time, leading to a total flow rate of γI from the infected to the recovered class.

To clarify and organise the processes in a SIR model (or the processes of any model), it is useful to draw a flow diagram. A flow diagram illustrates how each variable affects its own dynamics and those of the other variables.



Here, the boxes represent the S, I and R classes. Arrows that start at one box and go to another box indicate conversion of members of one class to another class. Solid arrows between boxes indicate the process removes an amount of the variable (the arrow leaves the box) or contributes some amount to a variable (enters the box). If an arrow exits and returns to the same box, then that variable can regenerate itself. Dashed arrows indicate there is an interaction between the variables, i.e., when a variable influences the flow into another box but does not represent a decline in the variable from which the arrow begins (e.g., an infectious individual does not lose the infection when she passes it on to someone else).

From the flow diagram, we can write down how the state variables S, I and R change according to the following system of differential equations:

$$\frac{dS}{dt} = -\lambda(I,t)\,S$$
$$\frac{dI}{dt} = \lambda(I,t)\,S - \gamma\,I$$
$$\frac{dR}{dt} = \gamma\,I.$$

These equations have the initial conditions $S(0) > 0$, $I(0) > 0$ and $R(0) \geq 0$.

The probability of an individual becoming infected in these models depends on several factors, in addition to the type of transmission: - N = Population size
- c = contact rate (number of contacts = C(N))
- ϵ = probability of transmission per individual contact
- I = number of infected hosts
- I/N = fraction of infected hosts

Density-depedent transmission

Often, we assume that the likelihood of contacting another individual depends on the population size, or density, of the given system. (i.e. sneezing in a crowded room vs. an empty room)

If contacts are a linear function of population size, then the *probability of an individual getting infected* …

$$= c \times N \times \epsilon \times I/N$$
$$= c \times \epsilon \times I$$
$$= \beta \times I$$

So, this reads as “the probability of me getting infected depends on the number of contacts I have, the probability of getting infected from one of those contacts with an infectious individual, and the probability that a contact is with an infected individual”.

Because the contact and per-contact-transmissibility rates are hard to estimate on their own, we simplify them to β and call that the *transmission rate*. With density-dependent transmission, the likelihood of transmission increases with increasing population size, because the N’s cancel out in the above equation.

Frequency-dependent transmission

On the other hand, many diseases don’t show density-dependent transmission. For sexually-transmitted and vector-borne diseases in particular, what seems to matter for transmission is the *frequency* or fraction of infected individuals in the population.

Specifically, we assume that the number of contacts shows an asymptotic relationshipwith population size - after a certain pop. size, contacts don’t increase. We’ll replace the density-dependent contact rate, $c \times N$, with the asymptotic contact value, which we’ll call k. Then, the likelihood of an individual getting infected depends on…

$$= k \times \epsilon \times I/N$$
$$= \beta \times I/N$$

The big difference here is that N remains in the denominator for the calculation of the transmission rate β. We’ll see some consequences of that later on.

Now back to the SIR model

For the force of infection $\lambda(I,t)$ we’ll assume that it has the form

$$\lambda(I,t) = \frac{\beta\, I}{N}$$

so that the risk of infection a susceptible faces is proportional to the fraction of the population that is infectious. In other words, transmission is frequency-dependent. This function for the force of infection is appropriate for some directly transmitted diseases of humans and vector-borne diseases. Notice that by writing the force of infection as $\lambda(I,t)$ we allow for the possibility of a contact rate that varies with time t.

Note this model is not suitable for all infectious disease scenarios. The model is suitable for spread of infection in a large naive population into which a low level of infectious agent is introduced and where the resulting epidemic occurs sufficiently quickly that demographic processes are not important.

In a *demographically open* system, the proportion of individuals in the population may change due to births and deaths that happen at *per capita* rates b and μ. Then we have

$$\begin{aligned}
\frac{dS}{dt} &= b\, S - \lambda(I,t)\, S - \mu\, S \\
\frac{dI}{dt} &= \lambda(I,t)\, S - \gamma\, I - \mu\, I \\
\frac{dR}{dt} &= \gamma\, I - \mu\, R.
\end{aligned}$$

If we set $b = \mu$ then births exactly balance deaths and the population remains at a constant size, yielding

$$\begin{aligned}
\frac{dS}{dt} &= \mu\, S - \lambda(I,t)\, S - \mu\, S \\
\frac{dI}{dt} &= \lambda(I,t)\, S - \gamma\, I - \mu\, I \\
\frac{dR}{dt} &= \gamma\, I - \mu\, R.
\end{aligned}$$

Exercise 3.

Draw the flow diagram for the demographically open SIR model.

Like many epidemiological models, one can't solve the SIR equations analytically. Rather, to find a solution of a continuous-time model such as the SIR, we integrate these equations numerically. The ODE solver needs to know the right-hand sides of the ODE; i.e. for each of S, I, and R, write down the arguments that add to each, as well as those that remove each.

We give the ODE solver this information as a function. Note that the form of the arguments and output of this function must exactly match what is expected by the LSODA routine. Thus, for instance, the time variable `t` must be the first argument even if the function is time-invariant so that `t` is neglected in the calculation. Here we write a function to return the derivatives of the closed SIR model.

```
ClosedSIRModel <- function (t, x, params) {

  # Computes the rate of change of the variables S, I and R over time
  #
  # Args:
  #   t: vector of time points in the integration
  #   x: vector of variables in the differential equation
  #   params: vector of parameter values
  #
  # Returns:
  #   The rate of change of the variables S, I and R.


  S <- x[1]                                 #create local variable S, the first element of x
  I <- x[2]                                 #create local variable I
  R <- x[3]                                 #create local variable R


  with(                                     #we can simplify code using "with"
    as.list(params),                      #this argument to "with" lets us use the variable n
ames
{                                    #the system of rate equations

  N <- S + I + R

  dS <- -beta*S*I*(1/N)      ### make 1/N explicit, infected # vs prop now consistent
  dI <- beta*S*I*(1/N)-gamma*I
  dR <- gamma*I
  dx <- c(dS,dI,dR)                 #combine results into a single vector dx
  list(dx)                          #return result as a list,
  #note order must be same as the state variables
}
  )
}
```

Notice that here, we've assumed β is constant.

We now state the times at which we want solutions:

```
times <- seq(0,120,by=5)    #function seq returns a sequence
```

We also require numerical values for each constant (parameter) in the SIR equation. The parameters are the contact rate β and the per-capita recovery rate γ. Here we assign some values to the parameters:

```
params <- c(beta=0.3,gamma=1/7)  #function c "c"ombines values into a vector
```

Finally we specify the initial conditions, the values of the state variables S, I, and R at the beginning of the simulation: here we will define them as proportions of the overall population size (what would happen if we changed that?)

```
xstart <- c(S=9999/10000,I=1/10000,R=0)     #initial conditions
```

We are now ready to solve the SIR model for each time in the `times` vector with the `lsoda` command:

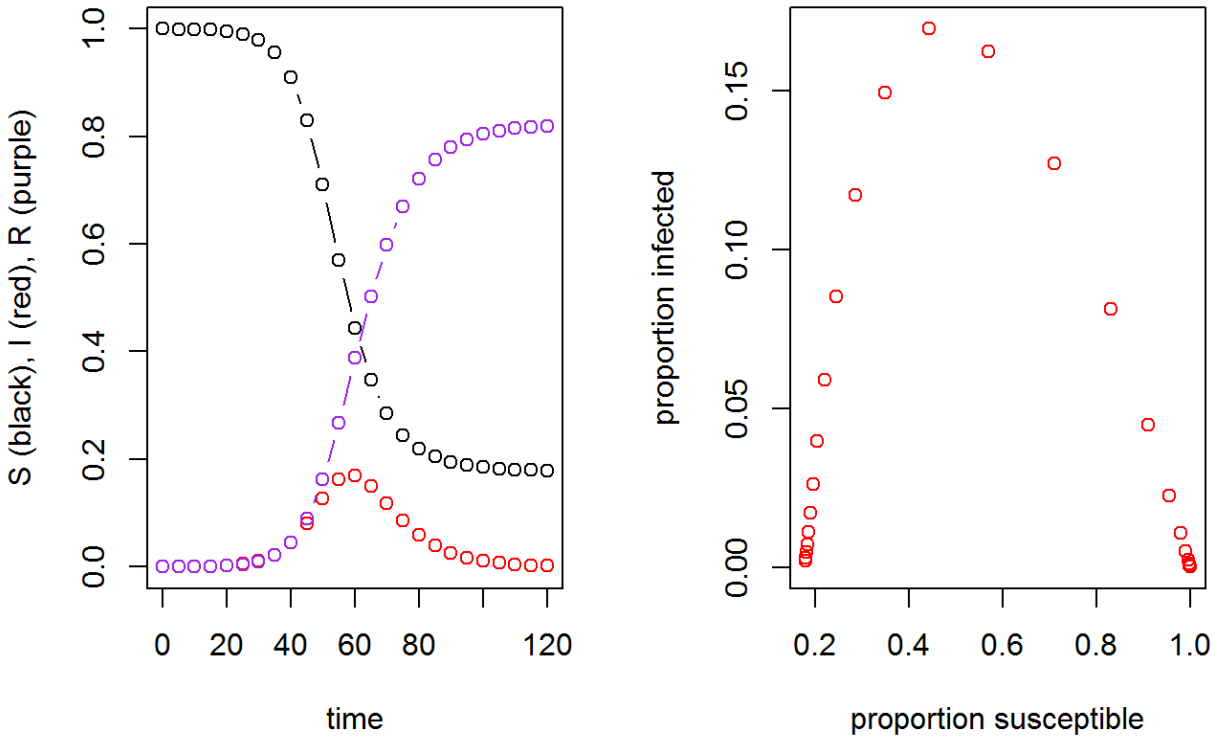
```
out.SIR <- as.data.frame(lsoda(xstart,times,ClosedSIRModel,params))  #result stored in dataframe
```

Each row of the `out.SIR` data frame corresponds to one of the times specified in the times vector that was given to LSODA when it was called. The first column of the data frame gives the time for that row. The second column gives the values for the S variable, the third column gives the values for the I variable and the fourth column gives the values for the R variable. These columns can then be used to produce various plots.

Let's compare how 1) the infected population changes over time to 2) the *trajectory* on the S-I phase space:

```
par(mfrow=c(1,2))
with(out.SIR,plot(S/(S+I+R)~time,type="b",col="black",ylim=c(0,1),ylab="S (black), I (red)
, R (purple)"))
with(out.SIR,lines(I/(S+I+R)~time,type="b",col="red"))
with(out.SIR,lines(R/(S+I+R)~time,type="b",col="purple"))

with(out.SIR,plot(I(I/(S+I+R))~I(S/(S+I+R)),col="red",xlab="proportion susceptible",ylab="
proportion infected"))
```




```
par(mfrow=c(1,1))
```

The solution of the model at time t, $(S(t), I(t))$, corresponds to a point in the *phase space*. (Note that we can neglect $R(t)$ since we can determine it from $S(t)$ and $I(t)$ because the proportions sum to 1). The curve is called a trajectory of the SIR system. The phase space is completely filled with trajectories since each point in the space can serve as an initial condition.

In summary, there are 4 steps to numerically solving a system of ODEs in R.

1. Write the system of equations that you would like to solve as a function that can be given as an argument to the ODE solver.

2. Set the times you require the solution, numerical values for the parameters and the starting values for the simulation.
3. Solve the system using LSODA.
4. Plot the results.

To complete the following exercises, you will need to implement steps 2–4.

Exercise 4. *Exploring different initial conditions*. To complete this exercise, keep β and γ the same as above, set the end time in the `times` vector to 300 and set the initial fraction of infectives $I(0)$ to 0.01.

a. Simulate a trajectory of the SIR model starting from $S(0) = 0.9, I(0) = 0.01, R(0) = 0.09$ using LSODA. Label the object that LSODA returns `out.e4a`. Plot the infected population as a function of time and the trajectory in the phase space, side by side, like the figures shown.

b. Now we will simulate two more trajectories. Solve the closed SIR system by choosing an initial fraction of susceptibles $S(0) > \gamma/\beta$ and initial proportion of recovered individuals equal to $1 - S(0) - I(0)$. Save the data frame as `out.e4b`. Then solve the closed SIR system, choosing an initial fraction of susceptibles $S(0) < \gamma/\beta$ and saving the data frame as `out.e4c`.

c. Now plot A) the three time series of proportion infected over time, and B) the three trajectories in the S-I phase space. You can do this using the `plot` function for the `out.e4a` trajectory and then calling the `points` function to plot the other two trajectories. Give each a different color with the `col =` option in the `plot` call. What do you notice?

The fraction of susceptibles γ/β is related to a threshold for invasion of the infection, called the basic reproduction ratio (or R_0, see Anderson and May (2) for more information).

R_0 helps define a threshold for pathogens to invade a population. It can be thought of as "the average number of new infections a single infection gives rise to in a completely susceptible host population."

What might be a 'threshold' value of R_0?

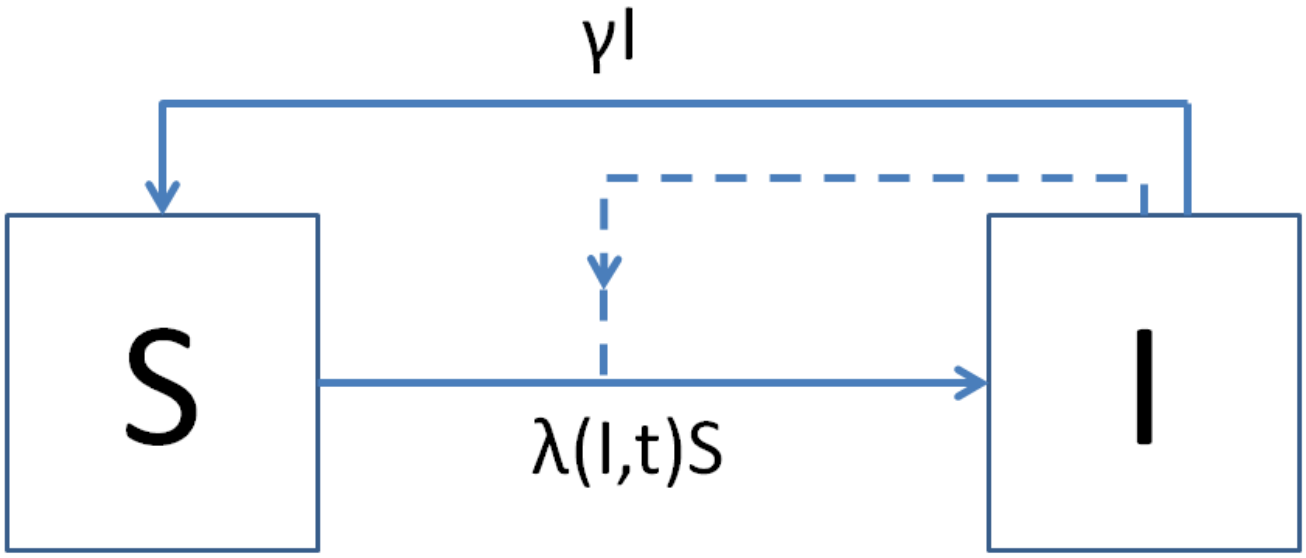
Exercise 5. Explore the dynamics of the system for different values of the β and γ parameters by simulating and plotting trajectories as time series and in phase space (e.g., I vs. S): - What happens as β gets larger? γ? - What do you think the quantity 1/γ represents? (Hint: try plotting the recovered population as well)

SIS model

We will now explore a different compartmental epidemic model. The demographically closed SIS compartmental model tracks the fraction of the population in each of two classes (susceptible and infectious). The model is suitable for representing illnesses that confer no immunity, e.g., influenza, sexually transmitted diseases. The assumptions of the model are:

- Susceptibles become infectious at a rate $\lambda(I, t)$ per susceptible individual per unit time. We let $\lambda(I, t) = \beta I$, where β denotes the contact rate per susceptible.
- Infectious individuals recover at a rate γ per infectious individual per unit time. Recovery does not confer immunity, leading to a total flow rate of γI from the infected to the susceptible class.

These assumptions are represented in the flow diagram (Figure 3). The processes in the flow diagram are represented by the following system of ordinary differential equations,



$$\frac{dS}{dt} = \gamma I - \beta SI$$
$$\frac{dI}{dt} = \beta SI - \gamma I.$$

Since the population is closed, then $S + I = 1$. We can use this equation to eliminate the S variable from the SIS model. Thus the SIS model can be represented by the equation for the change in I. Substituting $S = 1 - I$ into the equation for the change in I yields

$$\frac{dI}{dt} = \beta(1 - I)I - \gamma I$$

The initial condition $I(0) > 0$ can be used to solve this equation numerically. (Note that this is equavalent to logisitic equationn for dd pop growth in ecology)

Exercise 6.

Modify the codes given to study the dynamics of a demographically closed SIS model. Using $\beta = 0.3$ and $\gamma = 1/7$, plot the susceptible and infectious fractions of the population against time starting from various initial conditions. What do you notice? Try different β and γ values. Are any other `long-term', or *equilibrium*, solutions possible?

Exercise 7.

Modify the codes given to study the dynamics of a *demographically open* SIR model assuming that the per-capita birth rate is equal to the per-capita death rate. Assume that μ = 0.01, γ = 1/7, and β = 1/3. Plot the susceptible and infectious fractions of the population against time starting from $S(0)$ = 0.99, $I(0)$ = 0.01, and $R(0) = 0$ for 500 time steps. What do you notice?

Pause for some discussion

- What are the assumptions underlying the SIR and SIS models? Are they realistic? When might they be problematic? Are these models even useful?

Chain binomial models

In the previous sections we considered deterministic continuous time models. Now we consider a stochastic,

discrete-time model. Deterministic models are often appropriate if the population size is large, but if the population size is small, it is appropriate to consider *demographic stochasticity*, or the random nature of demographic processes, such as births, deaths, immigration and emigration. In an epidemic, the birth in question is the `birth' of a new infectious individual, which occurs if a susceptible individual contacts an infectious individual and contracts the infection from them.

Such an occurrence is referred to as a *trial*– a trial can be any sort of occurrence. For example, a coin toss can lead to heads or tails. A plant could produce any number of seeds from zero to thousands. Because more than one outcome is possible, we can consider the outcome to be a (discrete) *random variable*. Once a trial has occurred, the random variable takes a specific value. But, before the trial has occurred, we can consider the chance, or *probability* that it will take a certain value, or a particular outcome will occur. For example, there is a 50% probability that an unbiased coin may come up heads, but if that coin is biased, the probability may be much higher or lower than that. Tossing a coin is called a *Bernoulli trial* because only two outcomes, heads or tails, are possible.

Formally, a random variable, which we denote by X, has a *probability distribution* associated with it. This means that the variable is randomly distributed according to some function. In other words, the function can be used to calculate the probability that X takes some value k.

Suppose a coin is tossed n times. Each toss is an independent trial with two possible outcomes – heads or tails. Suppose that we want to quantify the probability of k tosses coming up as heads. The random variable X is the total number of heads obtained in n tosses, and the *binomial distribution* is associated with X. This means that the probability that $X = k$ heads can be calculated using

$$\text{Prob}[X = k] = \binom{n}{k} p^k (1-p)^{n-k},$$

where $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ is the total number of different ways k heads can occur in n trials, p is the probability of a head occurring in a single toss and $(1-p)$ is the probability of a tail occurring in a single toss (since these are mutually exclusive outcomes and so their probabilties must sum to 1).

Mathematically $X \sim \text{binomial}(n, p)$ or X has a binomial distribution with parameters n and p.

Where does the formula for the binomial distribution come from? Suppose we want to know in 3 coin tosses, what the probability of obtaining 2 heads is. We toss the coin and heads occurs, toss again and it comes up tails and on the final toss, we get heads again. The probability of the combination HTH of coin tosses occurring is $p \times (1-p) \times p$ (each coin toss is *independent* of the others, meaning that the probability of observing all three outcomes is equal to the product of the probabilities of observing each.). But THH can also occur with the same probability, as can HHT. The $\binom{n}{k}$ formula counts up the number of possible combinations.

Exercise 8. Suppose the probability of observing tails in a biased coin is 0.3. Calculate the probability that in 10 coin tosses, tails comes up 6 times. (Hint: Type `?dbinom`).

Reed-Frost chain binomial model We now consider the Reed-Frost chain binomial model for epidemics. Daley & Gani (3) provide a good introduction to chain binomial models. Susceptibles are converted to infectious individuals in discrete, non-overlapping intervals of time. There are two discrete random variables in the model – - S_t is the number of susceptibles at time t, and
- I_t is the number of infected individuals at time t

The population of susceptibles is now discrete, meaning that the number of susceptibles in the population at time t, denoted by S_t, is a non-negative integer–the number of susceptibles can take any number between 0 and N, the total number of individuals in the population. Time is also discrete– the times we count susceptibles are non-negative integers, where time t is counted in discrete steps from the initial time that the infection enters the population to the time T that the epidemic ends, $t = 0, 1, 2, \ldots, T$.

Let q denote the probability of contact between an infected individual and a susceptible individual, and assume infection occurs as a result of contact with probability β. Then $1-q$ is the probability there is no contact and $1-\beta$ is the probability that contact does not result in infection. Contact and infection are independent, and so the probability of observing both outcomes is equal to the product of the probabilities of observing each. Then the probability that a susceptible individual at time t does *not* become infected due to any single infectious individual is:

$$1-q+q(1-\beta)=\alpha.$$

In words, α combines the probabilities of two mutually exclusive events that may occur— a susceptible does not become infected either because they do not make contact with an infectious individual *or* there is contact but it does not result in infection.

The key assumptions of the Reed-Frost model are:

1. Time is discrete and the time steps of the model are equal to the length of the infectious period. Any individual that becomes infected remains infected for exactly one unit of time. This means that if an individual is infected at time t, at time $t+1$ that individual has died. On the other hand, it is not until time $t+1$ that susceptible individuals who have contracted the disease at time t are infectious.
2. The states S_t and I_t are discrete. At time $t=0$, the number of susceptibles is $S_0=s_0$ and the number of infected individuals is $I_0=i_0\geq 1$. The number of susceptibles can take any number between 0 and N, the total number of individuals in the population.
3. At time $t+1$, $S_{t+1}+I_{t+1}=S_t$, since all infected individuals at time t are removed by time $t+1$ due to assumption (1).
4. In one time step, a susceptible individual can become infected or it can avoid infection (remain susceptible). The probability that a susceptible individual at time t does not become infected due to any single infectious individual is α.
5. An individual that is susceptible at time t is still susceptible at time $t+1$ only if contact with infected individuals that leads to successful transmission of the infection is avoided. The probability of this event occurring is $p=\alpha\times\alpha\times\cdots\times\alpha=\alpha^{I_t}$. This is because each infected individual is independent of the others, and so meeting each infected individual are mutually exclusive outcomes.

Assumptions (4) and (5) imply that the number of susceptibles remaining at time $t+1$ is a binomial random variable, i.e., $S_{t+1}\sim\text{binomial}(S_t,p)$. In one time step, a susceptible individual can avoid infection (remain susceptible) with probability p or become infected with probability $1-p$. So there are just two possible outcomes for a susceptible individual. At time t, there are S_t individuals, each having a probability of remaining susceptible p. Thus, S_t represents the number of trials, since all susceptible individuals are independent of each other.

The binomial distribution describing the transition probability from state (s_t,i_t) to (s_{t+1},i_{t+1}) is

$$\text{Prob}[S_{t+1}=s_{t+1}]=\binom{s_t}{s_{t+1}}p^{s_{t+1}}(1-p)^{s_t-s_{t+1}}.$$

The name `chain binomial model` arises from this expression– to estimate the probability of a particular realization of the model, or of some simulation occurring, we multiply or `chain' these probabilties together.

Putting all these assumptions together, the equations of the Reed-Frost model are

$$\begin{aligned}S_{t+1}&=\text{binomial}(S_t,\alpha^{I_t}),\\I_{t+1}&=S_t-S_{t+1}.\end{aligned}$$

To simulate the Reed-Frost model, we need to set α and the initial conditions for the simulation. We let `S0` be the initial number of susceptibles and let `i` be the initial number of infectious individuals at time $t=0$. We put these into the first row of the output matrix `out.RF`.

```
alpha <- 0.96 #probability a susceptible individual does not become infected
s0 <- 100
i <- 1
t <- 0
out.RF <- c(t, s0,i) #first row of output matrix

###Next, we use a `while` loop to execute a simulation:

while(i>0){

         s <- rbinom(1, s0,alpha^i) #binomial random variable
         i <- s0 - s #calculate i_t+1 from s_t and and s_t+1
         t <- t+1 #update time vector
         out.RF <- rbind(out.RF, c(t, s,i)) # appends the new row of data to the output ma
trix
         s0 <- s # use s to calculate new number of susceptibles in the simulation
 }

colnames(out.RF) <- c("time", "S", "I")
head(out.RF)
```

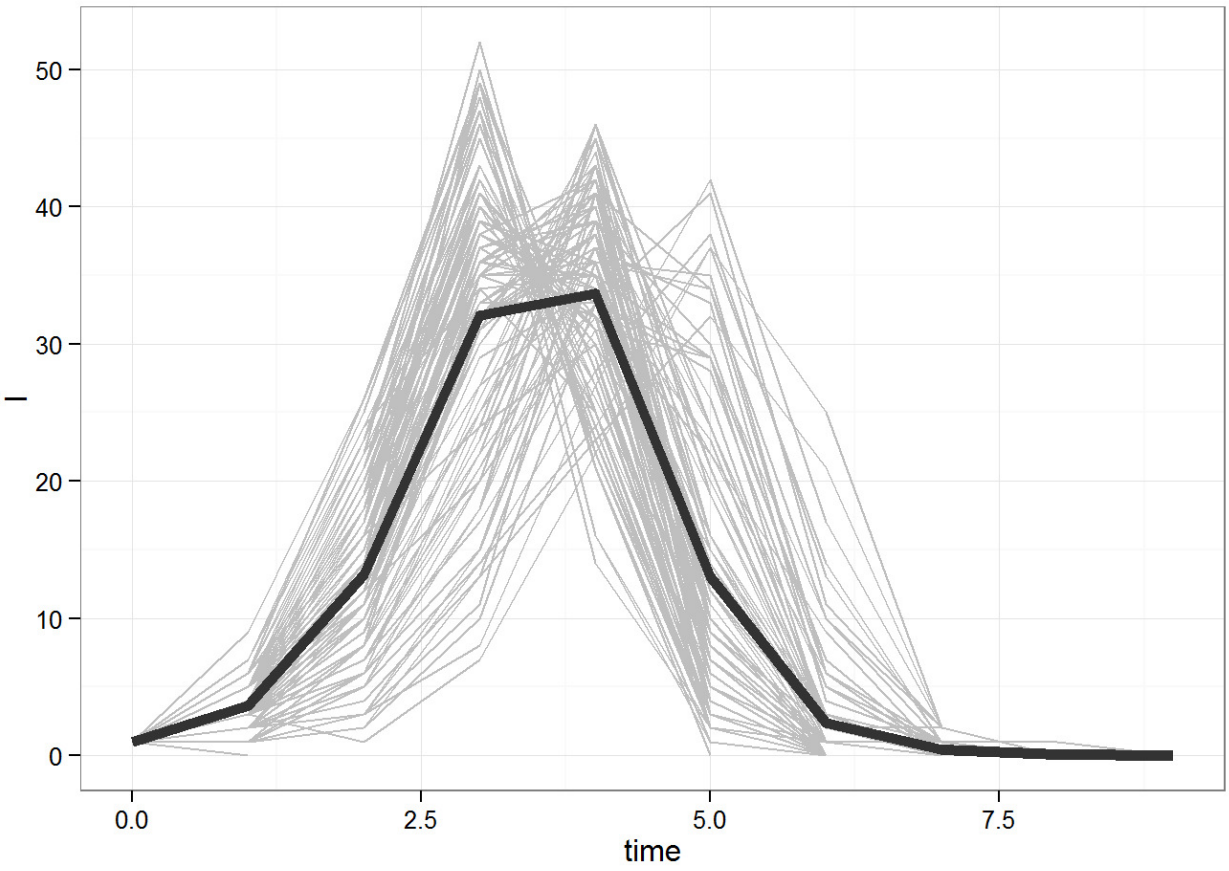
```
##        time   S  I
## out.RF    0 100  1
##           1  96  4
##           2  85 11
##           3  60 25
##           4  22 38
##           5   6 16
```

The `while` loop generates a random number of susceptibles from a binomial distribution, updates the number of infectious individuals and increments the time vector by 1 unit. These are stored into an output matrix, `out.RF`, by adding the new row of data from each iteration of the loop. The advantage of using the `while` command is that it terminates the loop when the condition inside the parentheses is false, *i.e.*, when no more infectious individuals remain.

Exercise 9. Plot the infected population against time.

Exercise 10 Calculate the mean epidemic *duration* and plot its distribution. (Hint: To generate multiple simulations, enclose the `while` loop in a `for` loop and calculate the time each epidemic ends using `out[dim(out)[1],1]`.

When epidemics are stochastic…



Final exercise (with the group or on your own if we run out of time)

- Make one of the deterministic models we've explored better. Think of an assumption and relax it. Think of something that's missing and add it. Make it more realistic for a disease of your choosing. Or anything else we've brainstormed, or something you come up with. I'd like you to;

1. Write down what your model is *for* - what new insights will we gain from it?
2. Make some predictions about how you think the model will behave - what do you expect will happen?
3. Sketch a flow diagram for your new model, with classes, transition rates, etc.
4. Write an `R` function for your model
5. Solve your function
6. Plot it over a range of 'realistic' parameter values
7. Summarize what you did, what you predicted, and what you found. Did the model do what you expected? How does it differ from what we've done previously?

Help each other with the implementation part. Ask for feedback on the ideas. Have some fun with it.

---

(1)Otto, S. P. and Day, T. A Biologists Guide to Mathematical Modeling in Ecology and Evolution. Princeton University Press, Princeton and Oxford, 2007.
(2) Anderson, R. M. and May, R. M. Infectious Diseases of Humans: Dynamics and Control. Oxford University Press, Oxford, 1991.
(3) Daley, D. J. and Gani, J. Epidemic Modelling: An Introduction. Cambridge University Press, Cambridge, 1999.