# Computer Practical ‘Deterministic models of infectious disease dynamics’.

Introduction

The purpose of this practical is to obtain a better understanding of the dynamics of infections in large populations. We will focus on a general situation, where some infection (e.g. avian influenza) is introduced in a population by infection of one individual from outside. This individual becomes infectious and starts spreading the virus to other members of the population. This spreading can be modelled by a compartmental **SIR** model, a model in which all individuals are at any time in one of the compartments (or states) **S**, **I**, and **R**. Before infection, everyone is **S**usceptible. After infection, people become **I**nfectious to others that are still susceptible. Infected people then develop an immune response and **R**ecover and stay immune for this particular infection during the rest of their lives (at least, as long as the virus is present in the population). The following **SIR** flow diagram shows the infection process:

R

*recovery*

*infection*

**I**

S

The simplest way to model the transitions from **S** to **I** (infection), and from **I** to **R** (recovery) is to assume a certain probability per unit of time for these events to happen. For infection of a susceptible individual, we assume that every susceptible makes ** contacts per unit of time, a proportion *Y/N* of which are made with infecteds. Thus, the mean number of contacts with infecteds in in a small time interval *t* is

*# infectious contacts* = (* Y / N*) × *t*

The number of people that will (on average) be infected during the time step *t* will be equal to the number of susceptible people, multiplied by the number of contacts they make:

*# new infections* = (* X Y / N*) × *t*

Similarly, if the individual recovery rate is **, the mean number of new recoveries in a time interval *t* is equal to

*# new recoveries* = (** *Y*)× *t*.

In a deterministic model, the above mean numbers of new infections and recoveries are used to exactly calculate the numbers of *X*, *Y*, and *Z*, in each very small time step. The model is written in the form of differential equations:

*X*(*t* + *t*) = *X*(*t*) – *# new infections*

⇔*X*(*t* + *t*) – *X*(*t*) = – (* X*(*t*) *Y*(*t*)/ *N*) × *t*



The last step is taken by letting *t* → 0, which is why *t* is replaced by *dt* (*dt* denotes an infinitesimally small time step). Similarly, for *Y*(*t*) and *Z*(*t*) the following differential equations are obtained:



1. Deterministic SIR model.

The modelling will be done here by using the R package deSolve, that numerically solves differential equations.

The first time you use this package, make sure you have installed it:

install.packages("deSolve")

Then open the accompanying R-code file (“Det\_no\_demo.R”), from which you can run:

library(deSolve)

##########################################################################

#Ordinary differential Equations

sir <- function(Time, State, Pars){

with(as.list(c(State,Pars)),{

dx <- -1 \* beta \* X \* Y / sum(State)

dy <- beta \* X \* Y / sum(State) - gamma \* Y

dz <- gamma \* Y

return(list(c(dx,dy,dz)))

})

}

#Parameters

pars <- c(beta = 0.6, #beta = transmission rate

gamma = 0.2 #gamma = recovery rate

)

#initial values

N0 = 30000 #population size

Y0 = 1 #initial fraction infected

Z0 = 0# initial fraction recovered,

X0 = N0 - Y0 - Z0

init <- c(X = X0,Y = Y0,Z = Z0)

#Data storage time

dt = 0.5#timestep for storing data

sim\_duration = 150 #length of the simulation

times <- seq(0, sim\_duration, by = dt)

#Solve the ordinary differential equations

ode.out <- ode(init, times, sir, pars)

#plot X, Y, and Z against time

plot(x = ode.out[,1], y = ode.out[,2], type = "l", xlab = "time", ylab = "animals", col = "purple",lwd =2)

lines(x = ode.out[,1], y = ode.out[,3], type = "l", xlab = "time", ylab = "animals", col = "red",lwd =2)

lines(x = ode.out[,1], y = ode.out[,4], type = "l", xlab = "time", ylab = "animals", col = "palegreen1",lwd =2)

legend(x = 60, y = 15000, c("X","Y","Z"),lwd =2, col = c("purple", "red", "palegreen1"))

A small explanation:

- the function sir is defined, which contains the model equations. If you want to simulate a different model, you should define each differential equation in a separate line, and end with the list containing all equation names.

- the vector pars is defined, which contains the transmission rate parameter ** (called beta), the recovery rate ** (called gamma),

- the vector init contains the initial number of infecteds Y0, the initial number of recovereds Z0, the initial number of susceptibles Y0 population size (30 000) minus the infecteds and recoverds.

- the vector times gives the times to store the time points of the simulation with time steps of the simulation dt and simulation length sim\_duration.

In object ode.out we store the simulation result. These can be plotted as count-versus time or by plotting a trajectory in the phase plane of X and Y.

Run the model with the default values.

**(a)**

- How long does the outbreak take?

- What is the peak number of infected people?

- How many people are still susceptible at the end of the outbreak?

Change the values for beta and gamma and see how the outbreak dynamics changes.

If you enter ode.out after a simulation and press enter, you can see how the simulation is stored in a table. This makes it possible to see exactly how many individuals are still susceptible at the end of the outbreak.

Using the code below you can plot the number of susceptibles against the number of infecteds.

#plot X against Y

plot(x = ode.out[,2], y = ode.out[,3], type = "l", xlab ="X", ylab ="Y")

s <- seq(length(ode.out[,2])-1)

arrows(x0 = ode.out[s,2], y0 = ode.out[s,3], x1 = ode.out[s + 1,2], y1 = ode.out[s + 1 ,3],length = .075)

**(b)** Look at the plots produced by the above commands, after running the simulation given above question (a).

- Where are start and end of the outbreak in the graph?

- Where is the peak?

- How many individuals are still susceptible at the end of the outbreak?

- The plot connects all values of the simulation with an arrow. Why are the arrow heads at the top of the graph further apart than the points close to the X-axis?

In the lectures we have learnt that the basic reproduction ratio *R*0 is an important parameter in infectious disease epidemiology: it is the mean number of new secondary cases caused by a single primary case in a susceptible population. Another important parameter is the duration of the infectious period *D*. In the SIR-model, *R*0 = */*, and *D* = 1/**.

**(c)** Simulate outbreaks with several combinations of ** and ** resulting in *R*0 = 3, and with several combinations resulting in *R*0 = 1.5, or *R*0 = 5 (or other values of *R*0 as you like). Look at the plots, but also directly at the output table. How does the disease dynamics (final size, duration of the outbreak) depend on the values of *R*0? How do the disease dynamics depend on the duration of the infectious period *D*?

Now we will create a plot for *R0* and the final size. To this end we will use the function runsteady of the package rootSolve.

#plot final size againt R0.

# For simplicity we set gamma = 1 so that beta = R0

finalsize <- NULL

for(R0 in seq(0,2,0.05)){

#add the outcomes to the results

finalsize <- rbind(finalsize,

data.frame(R0= R0, #R0 value

#final size at equilibrium (Y ==0)

finalsize = runsteady(y = init, func = sir,

parms = c(beta = R0,

gamma = 1))$y["Z"]))

}

plot(finalsize$R0,finalsize$finalsize, type ="l")

**(d)** Create a plot for *R0* and the final size. Study the effect of the initial number of infected individuals “Y” on the final size.

Why is taking x0 – X a better expression for the final size?

**(e)** Change the initial conditions to Y0 = 1 and Z0 = 70000 (with ** = 0.6 and ** = 0.2). What is the final size of the outbreak, ie how many animals got infected? Try more initial conditions, and more combinations of ** and **. What is the relation between *R*0, initial condition, and final size?

Now make the assumption that after infection, people first become latently infected (**E**xposed), before becoming **I**nfectious. In the model, we add an extra **E** compartment between the **S** and **I** compartments. The number of animals in the **E** state is *W*, and the rate at which animals leave the **E** state is *W*. The model equations become:



The adjusted model is shown at the bottom of the accompanying R code document, in the function seir. For **, the assumptions is that the mean latent period is 2 days.

**(f)** Repeat (a) for this model.

**(g)** Repeat (c) for the SEIR model, where you also adjust the mean latent period. How do final size and outbreak duration depend on the latent period?

2. Birth/immigration and death/emigration (deterministic)

In the **SIR** model of the previous exercise, the assumption was made that the population is closed, which means that no new individuals arrive due to birth or immigration and no individuals leave due to death or emigration. This model is realistic as long as the time scale of the epidemic is much faster than the time scale of the demography, e.g. with infections like influenza and common cold. With more slowly developing infections like bovine tuberculosis, however, demography should be included into the model for a more relevant description of the disease dynamics.

Now we will make a new **SIR** model *with* replacement of the host population. Newborn and immigrated individuals are all susceptible, whereas death or emigration takes place independent of the infection status. The new flow diagram becomes:

*birth/immigration*

S

*infection*

*recovery*

**I**

R

*death/emigration*

The rates at which individuals leave the **S**, **I**, and **R** states by death/emigration are *X*, *Y*, and *Z*, respectively. Birth takes place at rate **. The model then reads:



For simplicity we will assume that the population size remains constant, so that each death/emigration event is coupled to a birth/immigration event. That means that ** = *N*.

Now open the accompanying R-code file (“Det\_with\_demo.R”), from which you can run:

library(deSolve)

########################################################################

#Ordinary differential Equations

sir <- function(Time, State, Pars){

with(as.list(c(State,Pars)),{

dX <- -1 \* beta \* X \* Y /sum(State) +mu \*Y + mu \* Z

dy <- beta \* X \* Y / sum(State) - gamma \* Y - mu \* Y

dz <- gamma \* Y - mu \* Z

return(list(c(dX,dy,dz)))

})

}

#Parameters

pars <- c(beta = 0.6, #beta = transmission rate

gamma = 0.3, #gamma = recovery rate

mu = 0.05#mu = mortality rate

)

#initial values

N0 = 30000 #population size

Y0 = 1 #initial fraction infected

Z0 = 0# initial fraction recovered,

X0 = N0 - Y0 - Z0

init <- c(X = X0,Y = Y0,Z = Z0)

#Data storage time

dt = 0.5#timestep for storing data

sim\_duration = 150 #length of the simulation

times <- seq(0, sim\_duration, by = dt)

#Solve the ordinary differential equations

ode.out <- ode(init, times, sir,pars)

#plot X, Y, and Z against time

plot(x = ode.out[,1], y = ode.out[,2], type = "l", xlab = "time", ylab = "Count", col = "purple",lwd =2,ylim=c(0,N0))

lines(x = ode.out[,1], y = ode.out[,3], type = "l", xlab = "time", ylab = "Count", col = "red",lwd =2)

lines(x = ode.out[,1], y = ode.out[,4], type = "l", xlab = "time", ylab = "Count", col = "palegreen1",lwd =2)

legend(x = 200, y = 27000., c("X","Y","Z"),lwd =2, col = c("purple", "red", "palegreen1"))

#plot X against Y

plot(x = ode.out[,2], y = ode.out[,3], type = "l", xlab ="X", ylab ="Y")

s <- seq(length(ode.out[,2])-1)

arrows(x0 = ode.out[s,2], y0 = ode.out[s,3], x1 = ode.out[s + 1,2], y1 = ode.out[s + 1 ,3],length = .075)

}

**(a)** Study the dynamics of *X*, *Y*, and *Z* in time, and make a graph of *Y* against *X* (as in exercise 1b). How does the infection develop in time (extinction ?, equilibrium ? etc). What is the main difference between the **SIR** model without demography (exercise 1) and the **SIR** model with demography?

**(b)** Change the parameter values for **, **, and ** in your model, calculate *R*0 (see Learning Activity “Model Analysis”) and find the steady state number of susceptibles *X* from the simulation. How does the steady state *X* depend on *R*0? How does the oscillatory behaviour depend on *R*0, **, and **?

**ADDITIONAL ADVANCED QUESTIONS**

Suppose a veterinary epidemiologist is interested in reducing the prevalence of an infection in chickens by changing their life span. The research question is: how does host life span affect prevalence of infection? In order to see what the effect will be, he uses our model. He is not interested in short-term dynamics, but only what will happen in the long run. The focus is therefore on the steady state: when the prevalence has reached an equilibrium.

(d) Study the effect of changing the life span on the prevalence of an infection in the steady state. Interpret the results. Is a shorter or longer life span effective in reducing prevalence?

She presents the results to a colleague epidemiologist, who does not understand part of the results. She reasons: “if the life span becomes very short, you would expect that more chickens die (are removed from the farm) before being infected, and that if it is infected, it cannot effectively transmit the infection itself. However, in the results I see a high prevalence with short life span.”

(e) Explain why the model does not show the expected effect of a short life span?

Now there are two possibilities: either the explanation gives us a better understanding and new insight into the effect of life span, or the explanation shows us that the model is incorrectly formulated to answer the research question. The two epidemiologists agree that there must be something wrong in the model.

(f) How should the model be adjusted to provide an answer to the research question?

(g) Make the adjustment and answer the research question with the new model. Is a shorter or a longer life span effective in reducing prevalence?