Disease Dynamics and Modelling Assignment 2

Solving the SEIR model

Walkthrough: how to numerically solve an SEIR model

A novel pathogen has been discovered in 3 individuals in a population of 1000 individuals, with 2 individuals exposed but not infectious and 1 individual exposed and infectious. It has been found to have an incubation/progression period of 5 days, an infectious period of 10 days and a daily transmission rate of 1.7. You are to analyse the spread of this pathogen during the first 100 days of the outbreak using an SEIR model. Find and plot the solution.

Step 1: write a function that computes the system of ODEs and returns a list containing the derivatives. The input parameters of this function are time `t`, the compartments as a vector `x`, and model parameters `parameters`. The outputs of the function are a list of derivatives. We name this function as `SEIR.model`.

```
SEIR.model <- function(t, x, parameters){
  # Get the current number of individuals in each compartment.
  S <- x[1]
  E <- x[2]
  I <- x[3]
  R <- x[4]

  # Get the model parameters.
  beta <- parameters$beta
  rho <- parameters$rho
  gamma <- parameters$gamma

  # Calculate the total number of individuals in the system.
  # This is just all the compartments added together.
  N <- S + E + I + R

  # Calculate the derivatives (the system of ODEs).
  dS <- -beta*S*I/N
  dE <- beta*S*I/N - rho*E
  dI <- rho*E - gamma*I
  dR <- gamma*I

  # Return the derivatives inside a list.
  derivatives <- list(c(dS, dE, dI, dR))
  return(derivatives)
}
```

Step 2: define the initial conditions, parameters and times at which you wish to compute the solution.

```
# The initial parameters: how many people are in each compartment at the start of
# the model?
init <- c(S = 997, E = 2, I = 1, R = 0.0)

# The model parameters: what is the transmission rate (beta), progression rate (rho) and recovery rate
```

```r
pars <- list(beta = 1.7, rho = 0.2, gamma = 0.1)

# The solution times: the sequence of times for which to compute the solution.
time <- seq(from = 0, to = 100, by = 1)
```

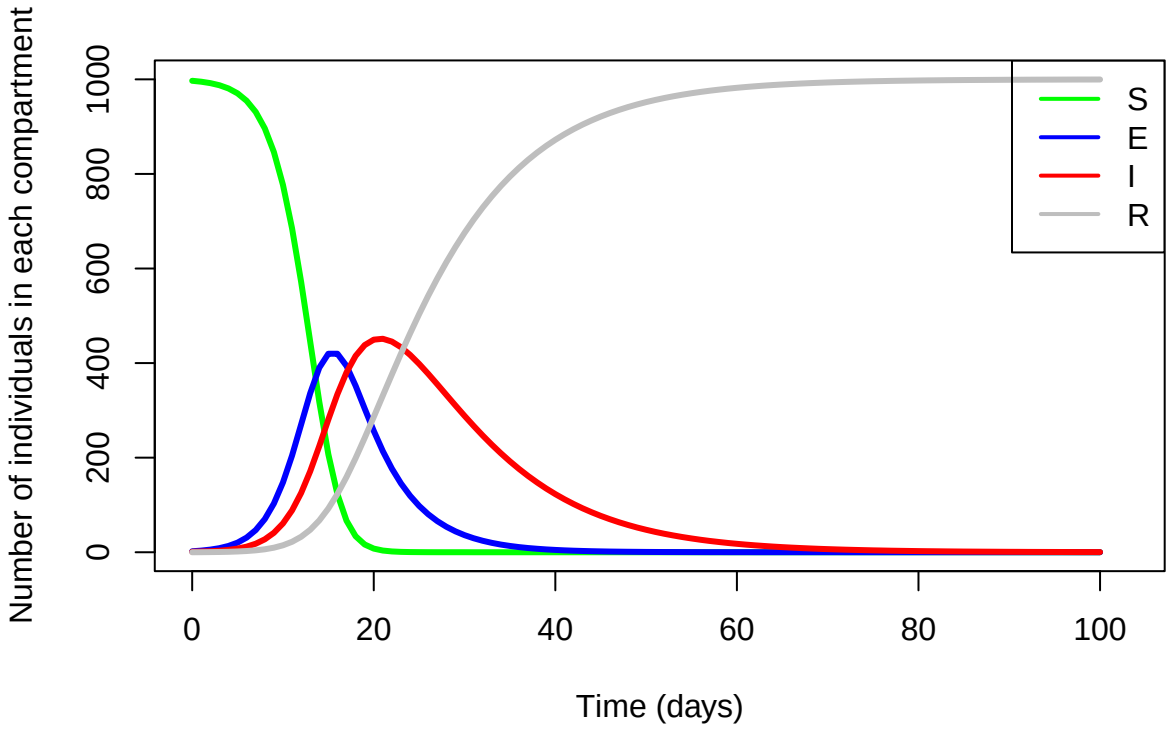
Step 3: solve the system using the 'lsoda' function from the 'deSolve' package (try help(lsoda) or ??Isoda for help). ## ode vs lsoda

```r
# Load in the deSolve package.
#install.packages("deSolve")
library(deSolve)

# lsoda is a function with inputs: ODE system (as a function), initial conditions,
# parameters and times to compute the solution. The solution for the SIR model is
# stored in 'out'.
out <- lsoda(func = SEIR.model, y = init, parms = pars, times = time)

# Convert the solution into a data.frame. This makes plotting easier.
out <- as.data.frame(out)
head(out)
```

```
##   time        S         E        I         R
## 1    0 997.0000  2.000000 1.000000 0.0000000
## 2    1 994.9855  3.474041 1.421520 0.1189792
## 3    2 992.0300  5.541006 2.135024 0.2939657
## 4    3 987.5578  8.618297 3.264161 0.5597502
## 5    4 980.7382 13.280410 5.014047 0.9673675
## 6    5 970.3516 20.352437 7.702291 1.5936649
```

Step 4: plot the solution for all compartments on a single plot.

```r
# First plot time (x-axis) against the number of susceptible (y-axis).
plot(out$time, out$S,
     type = 'l', col = 'green', ylab = "Number of individuals in each compartment",
     xlab = "Time (days)", lwd = 3, ylim = c(0, 1000), xlim = c(0,103))

# Next, we can add lines for time against infecteds, and time against recovered/removed.
lines(out$time, out$E, col = 'blue', lwd = 3)
lines(out$time, out$I, col = 'red', lwd = 3)
lines(out$time, out$R, col = 'grey', lwd = 3)

# Finally, we can add a legend so we know what lines correspond to what compartment.
legend("topright", legend = c("S","E","I","R"), col = c("green","blue","red","grey"), lwd = 2)
```




```
# view the last days of the model
tail(round(out))
```

```
##     time S E I    R
## 96    95 0 0 1  999
## 97    96 0 0 0 1000
## 98    97 0 0 0 1000
## 99    98 0 0 0 1000
## 100   99 0 0 0 1000
## 101  100 0 0 0 1000
```

Exercises: exploring the SEIR model

Now we can make some modifications to the model parameters, and observe how the model output changes.

Exercise 1

Add vital dynamics to the model i.e. birth and death, assuming; i) All individuals can reproduce regardless of the compartment ii) All newborn individuals are born into the susceptible compartment iii) The population is constant i.e. birth and death rate are equal

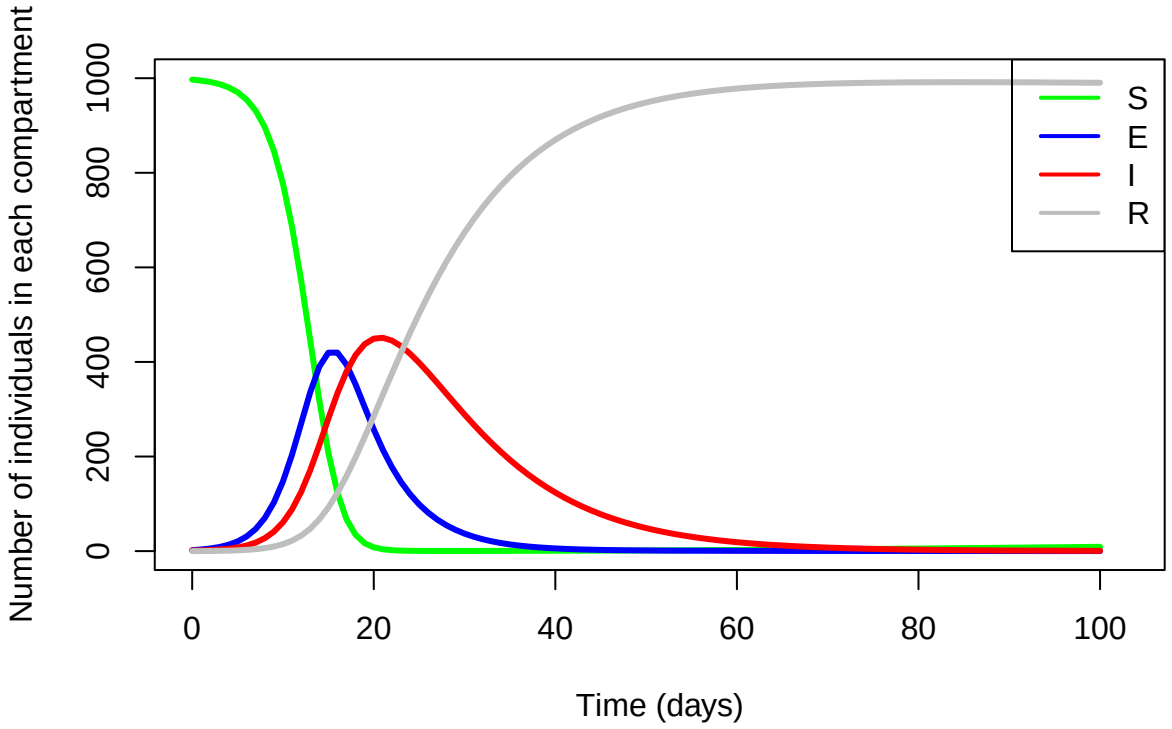
```
SEIR.model.exerciseone <- function(t, x, parameters){
  # Get the current number of individuals in each compartment.
  S <- x[1]
  E <- x[2]
  I <- x[3]
  R <- x[4]

  # Get the model parameters.
  beta <- parameters$beta
  rho <- parameters$rho
```

```
  gamma <- parameters$gamma
  mu <- parameters$mu

  # Calculate the total number of individuals in the system.
  # This is just all the compartments added together.
  N <- S + E + I + R

  # Calculate the derivatives (the system of ODEs).
  dS <- mu*N - beta*S*I/N - mu*S
  dE <- beta*S*I/N - rho*E - mu*E
  dI <- rho*E - gamma*I - mu*I
  dR <- gamma*I - mu*R

  # Return the derivatives inside a list.
  derivatives <- list(c(dS, dE, dI, dR))
  return(derivatives)
}

# The initial parameters: how many people are in each compartment at the start of
# the model?
init <- c(S = 997, E = 2, I = 1, R = 0.0)

# The model parameters: what is the transmission rate (beta), progression rate (rho), recovery rate (ga
# and birth/death rate (mu).
pars <- list(beta = 1.7, rho = 0.2, gamma = 0.1, mu = 0.0002)

# The solution times: the sequence of times for which to compute the solution.
time <- seq(from = 0, to = 100, by = 1)

# Solve the system using lsoda.
out <- lsoda(func = SEIR.model.exerciseone, y = init, parms = pars, times = time)

# Convert the solution into a data.frame. This makes plotting easier.
out <- as.data.frame(out)
```

Given the same population and parameters, with the daily birth/death rate (mu) = 0.0002, plot the trajectory of all the compartments in 100 days.

```
# Solve the system of ODEs using lsoda.
out <- lsoda(func = SEIR.model.exerciseone, y = init, parms = pars, times = time)

# Convert the solution into a data.frame.
out <- as.data.frame(out)

# Plot the results.
plot(out$time, out$S,
     type = 'l', col = 'green', ylab = "Number of individuals in each compartment",
     xlab = "Time (days)", lwd = 3, ylim = c(0, 1000), xlim = c(0,103))

# Next, we can add lines for time against infecteds, and time against recovered/removed.
lines(out$time, out$E, col = 'blue', lwd = 3)
lines(out$time, out$I, col = 'red', lwd = 3)
lines(out$time, out$R, col = 'grey', lwd = 3)
```

```
# Finally, we can add a legend so we know what lines correspond to what compartment.
legend("topright", legend = c("S","E","I","R"), col = c("green","blue","red","grey"), lwd = 2)
```




```
# view the last days of the model
tail(round(out))
```

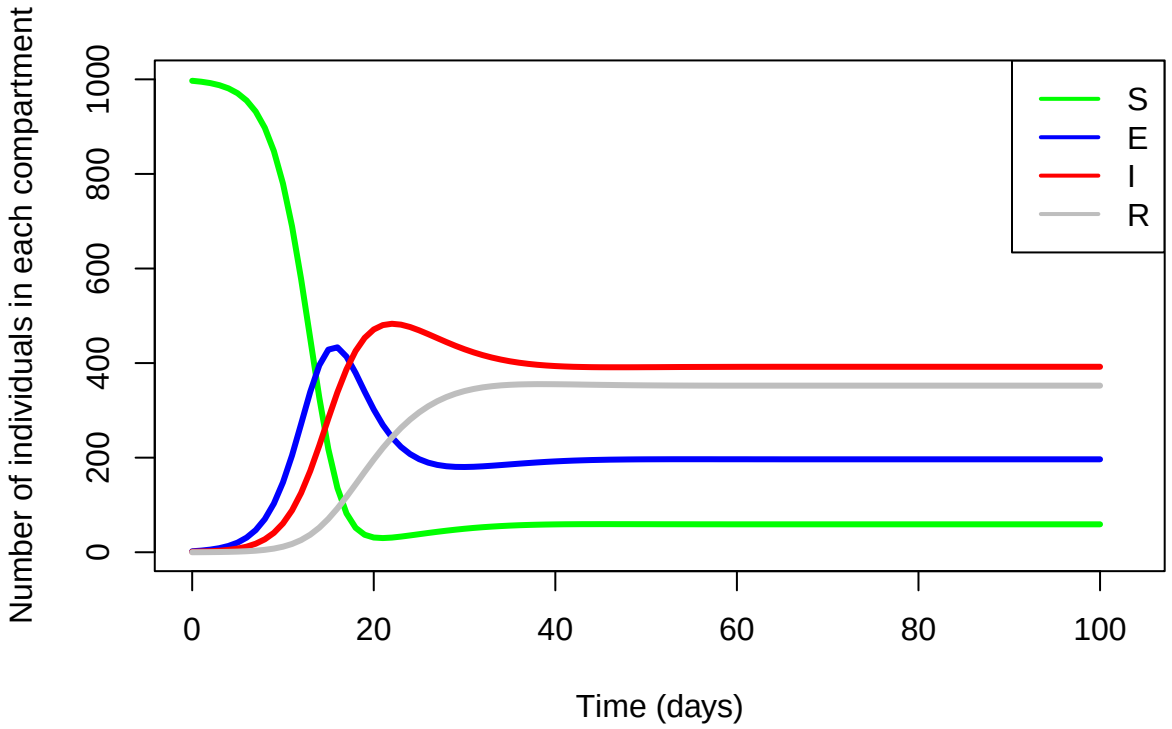
```
##     time S E I   R
## 96    95 8 0 1 991
## 97    96 8 0 1 991
## 98    97 8 0 1 991
## 99    98 9 0 1 991
## 100   99 9 0 1 990
## 101  100 9 0 1 990
```

Exercise 2

Add temporary immunity of 9 days

```
SEIRT.model <- function(t, x, parameters){
  # Get the current number of individuals in each compartment.
  S <- x[1]
  E <- x[2]
  I <- x[3]
  R <- x[4]

  # Get the model parameters.
  beta <- parameters$beta
  rho <- parameters$rho
  gamma <- parameters$gamma
  mu <- parameters$mu
  alpha <- parameters$alpha
```

```
  # Calculate the total number of individuals in the system.
  # This is just all the compartments added together.
  N <- S + E + I + R

  # Calculate the derivatives (the system of ODEs).
  dS <- alpha*R + mu*N - beta*S*I/N - mu*S
  dE <- beta*S*I/N - rho*E - mu*E
  dI <- rho*E - gamma*I - mu*I
  dR <- gamma*I  - mu*R - alpha*R

  # Return the derivatives inside a list.
  derivatives <- list(c(dS, dE, dI, dR))
  return(derivatives)
}

# The initial parameters: how many people are in each compartment at the start of
# the model?
init <- c(S = 997, E = 2, I = 1, R = 0)

# The model parameters: what is the transmission rate (beta), progression rate (rho), recovery rate (ga
# birth/death rate (mu), and rate at which individuals move from the infectious compartment to the temp
pars <- list(beta = 1.7, rho = 0.2, gamma = 0.1, mu = 0.0002, alpha = 1/9)

# The solution times: the sequence of times for which to compute the solution.
time <- seq(from = 0, to = 100, by = 1)

# Solve the system using lsoda.
out <- lsoda(func = SEIRT.model, y = init, parms = pars, times = time)

# Convert the solution into a data.frame. This makes plotting easier.
out <- as.data.frame(out)


# Plot the results.
plot(out$time, out$S,
     type = 'l', col = 'green', ylab = "Number of individuals in each compartment",
     xlab = "Time (days)", lwd = 3, ylim = c(0, 1000), xlim = c(0,103))

# Next, we can add lines for time against infecteds, and time against recovered/removed.
lines(out$time, out$E, col = 'blue', lwd = 3)
lines(out$time, out$I, col = 'red', lwd = 3)
lines(out$time, out$R, col = 'grey', lwd = 3)

# Finally, we can add a legend so we know what lines correspond to what compartment.
legend("topright", legend = c("S","E","I","R"), col = c("green","blue","red","grey"), lwd = 2)
```




```
# view the last days of the model
tail(round(out))
```

```
##     time  S   E   I   R
## 96    95 59 196 392 352
## 97    96 59 196 392 352
## 98    97 59 196 392 352
## 99    98 59 196 392 352
## 100   99 59 196 392 352
## 101  100 59 196 392 352
```

Exercise 3

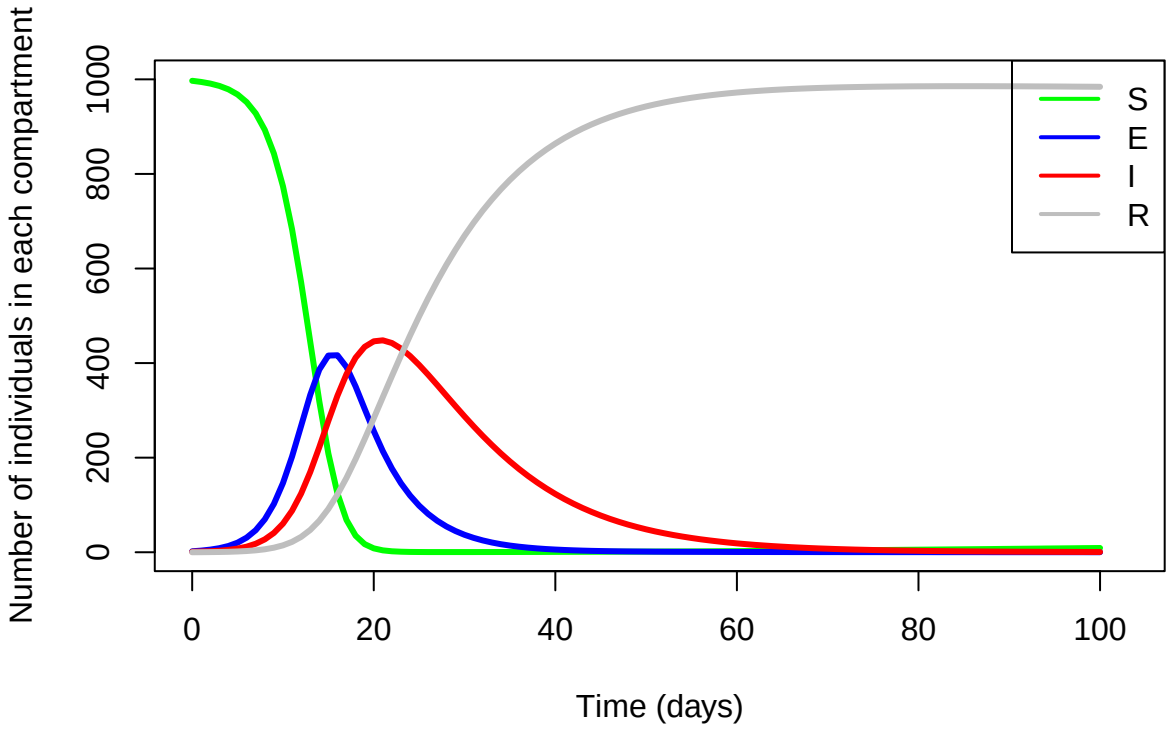
With the vital dynamics mentained, add a vaccination rate of 0.0005 with permanent immunity after both vaccination and recovery.

```
SEIRV.model <- function(t, x, parameters){
  # Get the current number of individuals in each compartment.
  S <- x[1]
  E <- x[2]
  I <- x[3]
  R <- x[4]
  V <- x[5]

  # Get the model parameters.
  beta <- parameters$beta
  rho <- parameters$rho
  gamma <- parameters$gamma
  mu <- parameters$mu
  v <- parameters$v

  # Calculate the total number of individuals in the system.
```

```r
  # This is just all the compartments added together.
  N <- S + E + I + R + V

  # Calculate the derivatives (the system of ODEs).
  dS <- mu*N - beta*S*I/N - mu*S - v*S
  dE <- beta*S*I/N - rho*E - mu*E
  dI <- rho*E - gamma*I - mu*I
  dR <- gamma*I - mu*R
  dV <- v*S - mu*V

  # Return the derivatives inside a list.
  derivatives <- list(c(dS, dE, dI, dR, dV))
  return(derivatives)
}

# The initial parameters: how many people are in each compartment at the start of
# the model?
init <- c(S = 997, E = 2, I = 1, R = 0.0, V = 0.0)

# The model parameters: what is the transmission rate (beta), progression rate (rho), recovery rate (ga
# birth/death rate (mu), vaccination rate (v), and recovery rate for vaccinated individuals (gamma_v).
pars <- list(beta = 1.7, rho = 0.2, gamma = 0.1, gamma_v = 0.2, mu = 0.0002, v = 0.0005)

# The solution times: the sequence of times for which to compute the solution.
time <- seq(from = 0, to = 100, by = 1)

# Solve the system using lsoda.
out <- lsoda(func = SEIRV.model, y = init, parms = pars, times = time)

# Convert the solution into a data.frame. This makes plotting easier.
out <- as.data.frame(out)
head(round(out))
```

```
##   time   S  E I R V
## 1    0 997  2 1 0 0
## 2    1 994  3 1 0 0
## 3    2 991  6 2 0 1
## 4    3 986  9 3 1 1
## 5    4 979 13 5 1 2
## 6    5 968 20 8 2 2
```

```r
# Plot the results.
plot(out$time, out$S,
     type = 'l', col = 'green', ylab = "Number of individuals in each compartment",
     xlab = "Time (days)", lwd = 3, ylim = c(0, 1000), xlim = c(0,103))

# Next, we can add lines for time against infecteds, and time against recovered/removed.
lines(out$time, out$E, col = 'blue', lwd = 3)
lines(out$time, out$I, col = 'red', lwd = 3)
lines(out$time, out$R, col = 'grey', lwd = 3)

# Finally, we can add a legend so we know what lines correspond to what compartment.
legend("topright", legend = c("S","E","I","R"), col = c("green","blue","red","grey"), lwd = 2)
```




```
# view the last days of the model
tail(round(out))
```

```
##     time S E I   R V
## 96    95 8 0 1 985 6
## 97    96 8 0 1 985 6
## 98    97 8 0 1 985 6
## 99    98 9 0 1 985 6
## 100   99 9 0 1 984 6
## 101  100 9 0 1 984 6
```

'i) What is the Ro of the final model and what does it imply?

```
beta <- 1.7
rho <- 0.2
gamma <- 0.1
alpha <- 0.0005
mu <- 0.0002

R0 = beta*mu*rho/((alpha*gamma*mu)+(alpha+gamma)*(mu*mu)+(mu*mu*mu)+((alpha*gamma)+(alpha+gamma)*mu+(mu
print(R0)
```

```
## [1] 4.842605
```

```
# An R0 of 4.842605 in disease modeling means that each infected person, on average, will transmit the
# An R0 value above 1 indicates that the disease is likely to spread through a population.
```

ii) What proportion of individuals would have to be vaccinated to attain herd immunity for the final model?

```
# Calculate the herd immunity threshold
R0 <- 4.842605
HIT <- (1 - 1/R0)
HIT
```

```
## [1] 0.7934996
```