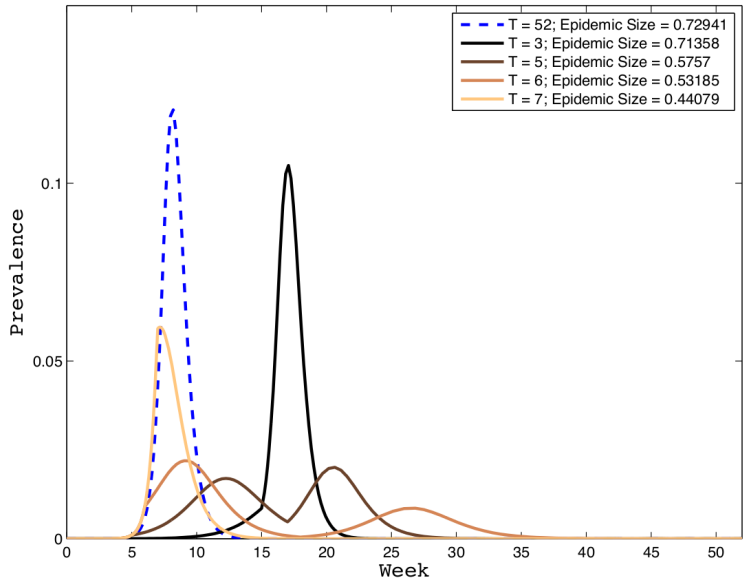
# Social Distancing*

Pejman Rohani & John M. Drake

## 1 Introduction

In this lab, we reproduce the following figure from the lecture. In the process, we will again work with loops and numerical integration.




As you will recall, in this scenario we were concerned with the population of the UK (58 million) and considered what happens when a single individual infected with a novel pathogen is introduced. We assumed the mean infectious period is 2.6 days, there is no latent period, and that $R_0$ is 1.8. The blue dotted line represents the situation where no control measures are implemented, so this is our baseline scenario. The other lines depict what happens if we reduce the transmission rate, $\beta$, by a factor $\phi = 0.3333$ starting on week 3, 5, 6 and 7. These social distancing strategies are maintained for a period $D$ weeks, in this instance $D = 12$.

## 2 Basic setup

Our starting point, as before, is to define a function that describes the set of differential equations that govern our system. Since we're dealing with a simple epidemic, we won't worry about host

demography (i.e., no births/deaths). We can reuse some code from before, specifically the function `sir.model.closed`.

```
> require(deSolve)
> #define a color palette
> pulp <- c(rgb(101,48,47, maxColorValue=255),    #brown
+           rgb(210,202,203, maxColorValue=255),  #gray
+           rgb(211,141,101, maxColorValue=255),  #peach
+           rgb(223,45,39, maxColorValue=255),    #red
+           rgb(250,208,10, maxColorValue=255),   #yellow
+           rgb(16,16,18, maxColorValue=255),     #black
+           rgb(76,106,147, maxColorValue=255)    #blue
+           )
```

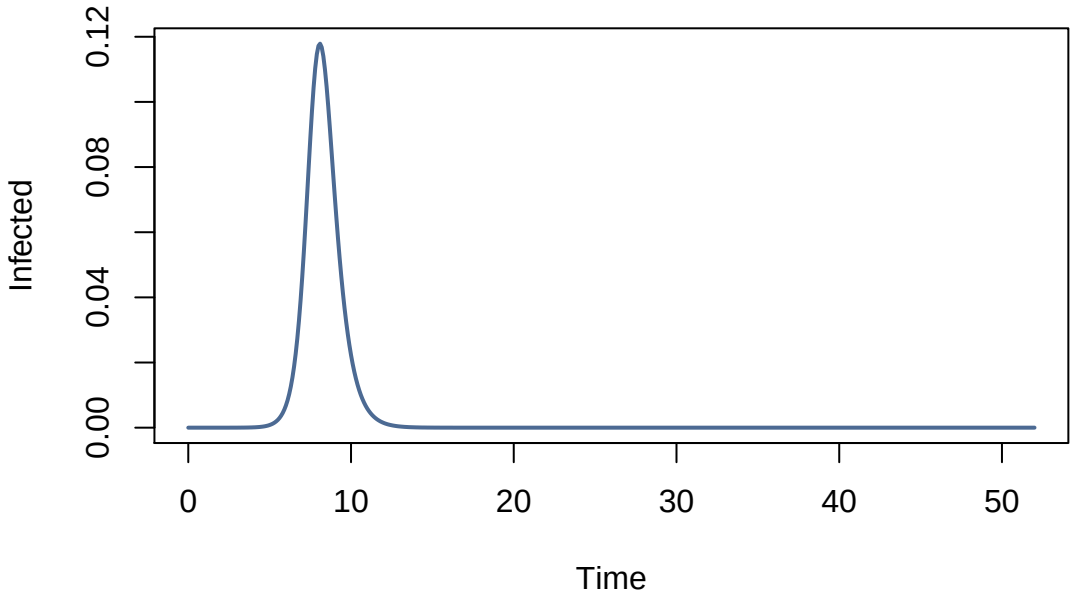
```
> sir.model.closed <- function (t, x, params) {    #here we begin a function with three arguments
+   S <- x[1]                                      #create local variable S, the first element of x
+   I <- x[2]                                      #create local variable I
+   R <- x[3]                                      #create local variable R
+   with(                                          #we can simplify code using "with"
+        as.list(params),                          #this argument to "with" lets us use the variable names
+        {                                         #the system of rate equations
+          dS <- -beta*S*I
+          dI <- beta*S*I-gamma*I
+          dR <- gamma*I
+          dx <- c(dS,dI,dR)                       #combine results into a single vector dx
+          list(dx)                                #return result as a list
+        }
+        )
+ }
```

As before, we define the model parameters: $\beta = R_0\gamma/N$, $\gamma = 7/2.6$, $T_{max} = 52$, $D = 12$ and the vector of initial conditions. We run the model once (without any social distancing) and plot the output to make sure it's working correctly. This is the blue dotted line in the figure above.

```
> R0 <- 1.8
> N <-  1                                                 #population size
> gamma <- 7/2.6                                        #recovery rate (in years)
> beta <- R0*(gamma)/N                 #transmission rate
> phi <- 0.3333
> D <- 12
> xstart <- c(S=0, I=0, R=0)          #initial conditions, must sum to one
> xstart[2] <- 1/58000000
> xstart[1] <- 1-xstart[2]
> xstart[3] <- 1-sum(xstart)
> Tmax <- 52                     #integrate for 200 years after transients
> params <- c(beta=beta, gamma=gamma)   #parameter vector
> tau <- 0.1                     #size of time step
> times <- seq(0, Tmax, by=tau) #function seq returns a sequence
```

```
> out <- as.data.frame(ode(xstart,times,sir.model.closed,params, method='ode45', rtol=1e-7))
```

```
> plot(out$time,
+      out$I,
+      type='l',
+      lwd=2,
+      xlab='Time',
+      ylab='Infected',
+      col=pulp[7])  #plot the I variable against time
```




Yup. That looks about right.

## 3 Adding social distancing

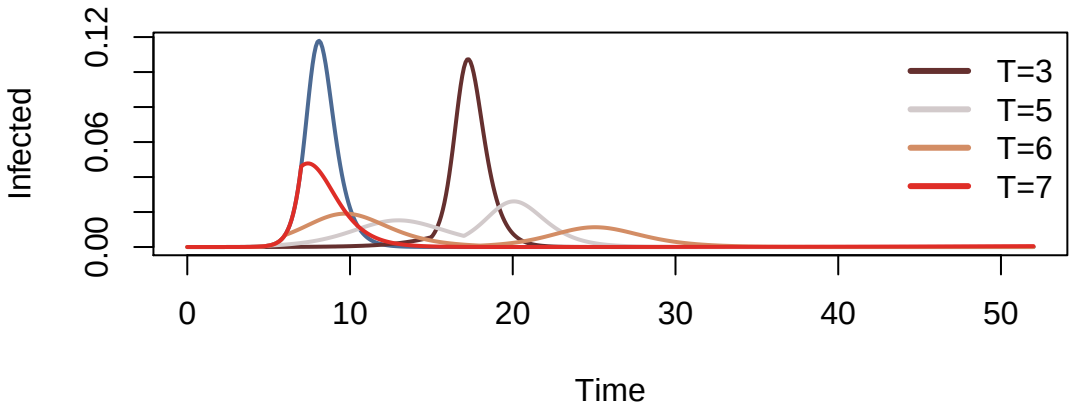
To implement social distancing in the model, we need to run the baseline model up to the week that our control measure is being implemented, set the end state of our system as the new initial conditions, reduce the transmission rate by $\phi = 0.3333$, integrate with this new value of $\beta$ for $D$ weeks, record the end state of this run, go back to our default value of $\beta$ (without any social distancing) and integrate until 52 weeks is reached. To inspect the result, we plot the epidemic curve. We could specify different values of $T$ and manually run the model a number of times, but that wouldn't be efficient. So, let's define a vector of the time transmission reduction is initiated,

```
> Tvec <- c(3, 5, 6, 7)
```

and loop over these values.

```
> # plot baseline again
> plot(out$time,
```

```
+      out$I,
+      type='l',
+      lwd=2,
+      xlab='Time',
+      ylab='Infected',
+      col=pulp[7])  #plot the I variable against time
> # create a list to store information for the legend
> legend.info <- data.frame(T=numeric(0),col=character(0),stringsAsFactors=FALSE)
> for(i in 1:length(Tvec)){
+   T <- Tvec[i]
+
+   out1 <- ode(xstart,                                  # start at initial condition
+               seq(0,T,tau),                            # solve from 0 to T
+               sir.model.closed,params,
+               method='ode45',
+               rtol=1e-7)
+
+   out2 <- ode(tail(out1,1)[2:4],                       # start at end of last solution
+               seq(T,T+D,tau),                          # solve from T to T+D
+               sir.model.closed,
+               c(beta=beta*(1-phi), gamma=gamma),       # change beta
+               method='ode45',
+               rtol=1e-7)
+
+   out3 <- ode(tail(out2,1)[2:4],                       # start at end of last solution
+               seq(T+D, Tmax, tau),                     # solve from T+D to Tmax
+               sir.model.closed,
+               params,                                  # reset parameters
+               method='ode45',
+               rtol=1e-7)
+
+   data <- as.data.frame(rbind(out1, out2, out3))
+
+   lines(data$time,
+         data$I,
+         col=pulp[i],
+         lwd=2)
+
+   legend.info[i,] <- c(T=T, col=as.character(pulp[i]))
+ }
> legend('topright',
+        legend=paste('T=',legend.info$T, sep=''),
+        lty=1,
+        lwd=3,
+        col=legend.info$col,
+        bty='n')
```




A few things about this code deserve some attention. First, we use the colormap `pulp` in different iterations of the loop and store the relevant information for inclusion in a legend that is added at the end. Second, pay attention to the initial conditions, time intervals for solutions, and parameter sets for each of thethree phases of the epidemic. These are handled differently, and somewhat subtly, in the loop. Finally, the results are store, but some information is redundance. (What information is this? Does it interfere with drawing our plot? Why or why not?)

***Exercise 1.** Using the approach outlined in this lab, show that the final epidemic size over a one-year period looks as below. Assume that initially everyone is susceptible and 1 out of 58 million are infectious. The social distancing strategy is in place for 12 weeks and $R_0 = 1.8$ with a mean infectious period of 2.6 days. You can ignore host births/deaths.