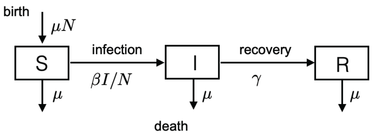
The SIR model, Integrating ODE's and a ShinyApp

Ottar N. Bjørnstad

 The app was originally developed as part of the epimdr2-package (https://cran.r-project.org/package=epimdr2; Bjørnstad 2023). The app requires epimdr2 to run



The basic equations for the flow of hosts between **S**usceptible, **I**nfectious and **R**ecovered compartments are:

$$\frac{dS}{dt} = \underbrace{\mu N}_{\text{birth}} - \underbrace{\beta I \frac{S}{N}}_{\text{infection}} - \underbrace{\mu S}_{\text{death}}$$

$$\frac{dI}{dt} = \underbrace{\beta I \frac{S}{N}}_{\text{infection}} - \underbrace{\gamma I}_{\text{recovery}} - \underbrace{\mu I}_{\text{death}}$$

$$\frac{dR}{dt} = \underbrace{\gamma I}_{\text{recovery}} - \underbrace{\mu R}_{\text{death}}$$

The assumptions of this version of the SIR model are:

- The infection circulates in a population of size N, with a per capita ``background" death rate, μ, which is balanced by a birth rate μN. From the sum of the equations $dN/dt = 0$ and $N = S + I + R$ is constant.
- The infection causes morbidity (not mortality).
- Newborns are recruited directly into the susceptible class at birth.
- Transmission of infection from infectious to susceptible individuals is controlled by a bilinear contact term $\beta I \frac{S}{N}$, from the assumption that the I infectious individuals are independently and randomly mixing with all other individuals, so the fraction S/N is with susceptible individuals; β is the transmission rate.
- Infected individuals move directly into the the infectious class and remains there for an average infectious period of $1/(\gamma + \mu)$ time units.
- Recovered individuals are immune from re-infection for life. numerical integration of a variety of different ODE models will be required. While the models differ, the basic recipe is generally the same: (1) define a R-function for the general system of equations, (2) specify the time points at which we want the integrator to save the state of the system, (3) provide values for the parameters, (4) give initial values for all state variables, and finally (5) invoke the ode()-function from the deSolve package.

```
require(epimdr2)
require(deSolve)
```

STEP 1: Define the function (often called the gradient functions) for the equation systems. The deSolve package requires the function to take the following parameters: time *t* a vector with the values for the state variables (in this case S, I and R) *y* and parameter values (for β, μ, γ, and N) *parameters*:

```
sirmod=function(t, y, parameters){
  #Pull state variables from y vector
  S=y[1]
  I=y[2]
  R=y[3]
  #Pull parameter values from the input vector
  beta=parameters["beta"]
  mu=parameters["mu"]
  gamma=parameters["gamma"]
  N=parameters["N"]
  #Define equations
  dS = mu * (N  - S)  - beta * S * I / N
  dI = beta * S * I / N - (mu + gamma) * I
  dR = gamma * I - mu * R
  res=c(dS, dI, dR)
  #Return list of gradients
  list(res)
}
```

STEP 2–4: Specify the time points at which we want *ode* to record the states of the system (here we use a half year with weekly time increments as specified in the vector *times*), the parameter values (in this case as specified in the vector *paras*), and starting conditions (specified in *start*). If we model the fraction of individuals in each class, we set $N = 1$. Let's consider a disease with an infectious period of 2 weeks ($\gamma = 365/14$ per year) for the closed epidemic (no births or deaths so $\mu = 0$). A reproduction number of 4 which implies a transmission rate $$ of 2. For our starting conditions assume that 0.1% of the initial population is infected and the remaining fraction is susceptible.

```
times  = seq(0, 0.5, by=1/365)
paras  = c(mu = 0, N = 1, R0=4, gamma = 365/14)
paras["beta"]=paras["R0"]*(paras["gamma"]+paras["mu"])
start = c(S=0.999, I=0.001, R = 0)*paras["N"]
```

STEP 5: Feed *start* values, *times*, the gradient function *sirmod* and parameter vector *paras to the ode()-function as suggested by* args(ode). *For further details on usage, do* ?function on the R command line, ?ode" in this instance. For convenience we convert the output to a data frame (ode() returns a list). The head() function shows the first 5 rows of *out* and *round(,3)* rounds the number to three decimals.

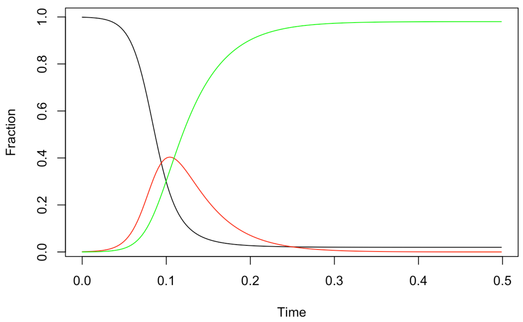
Question: 1. What are the interpretation of the values of the parameters?

```
out = ode(y = start, times = times, func = sirmod,
     parms = paras)
out=as.data.frame(out)
head(round(out, 3))
```

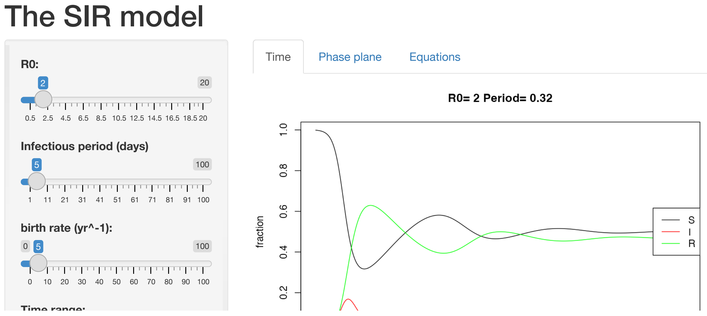
```
##    time     S     I     R
## 1 0.000 0.999 0.001 0.000
## 2 0.003 0.999 0.001 0.000
## 3 0.005 0.998 0.002 0.000
## 4 0.008 0.998 0.002 0.000
## 5 0.011 0.997 0.002 0.000
## 6 0.014 0.996 0.003 0.001
```

Plot:

```
plot(x = out$time, y = out$S, ylab = "Fraction",
     xlab = "Time", type = "l")
lines(x = out$time, y = out$I, col = "red")
lines(x = out$time, y = out$R, col = "green")
```



R has advanced capabilities of directly producing interactive web pages through the **shiny** package using the ui() and server() syntax… This is a bit elaborate elaborate for this workshop but a silly basic example is in the "shinyex.rmd" file. A shiny app for the SIR model is launched by "run document" on menu line:



References:

Bjørnstad, O.N. (2018) Epidemics: Models and Data using R. Springer (312 pp) ISBN 978-3-319-97487-3
https://www.springer.com/gp/book/9783319974866