Advanced disease modelling - Practical

Clara Champagne

Motivation

We have data on a flu epidemic in an English boarding school in 1978 source. Among the 763 boys in the school, 512 were infected and put in bed during their infection, which allows us to follow the number of infected boys over time.

```
time = c(3,4,5,6,7,8,9,10,11,12,13,14,15)
infected = c(31,82,216,299,269,242,190,125,81,52,25,22,7)
data=cbind(time, infected)
```

The objective of this exercise is to recover the parameters associated with this epidemic. We use for this an SIR-type model, which can be simulated with the following code:

```
library(deSolve)
```

```
## Warning: package 'deSolve' was built under R version 4.3.3
```

```
##SIR model
SIR<-function(t,x,parms){
  ##taille de chaque compartiment et de la population
  S = x[1]
  I = x[2]
  R = x[3]
  N = x[1]+x[2]+x[3]

  ##valeurs des parametres
  lambda = parms["lambda"]
  gamma = parms["gamma"]

  ##variations
  dS=-lambda*S*I/N
  dI=lambda*S*I/N-gamma*I
  dR=gamma*I
  res  = c(dS,dI,dR)
  list(res)
}

simulate_SIR=function(parameters){

  #parameters
  parms = c(parameters["lambda"],parameters["gamma"])
  N=parameters["N"]

  #initial conditions
  init <- c(N-parameters["initI"],parameters["initI"],0)
```

```
  #simulation
  temps <- seq(0,15)
  solveSIR <- lsoda(y =init, times=temps, func = SIR,
                    parms = parms)
  solutionSIR=as.data.frame(solveSIR)
  names(solutionSIR)=c("time","S","I","R")

  #merge with data
  sir_data=merge(data,solutionSIR)

  return(sir_data)
}

theta_init =c("lambda"=1.7,"gamma"=0.44,"initI"=1, "N"=763)
simul=simulate_SIR(theta_init)
simul
```

```
##    time infected         S         I         R
## 1     3       31 708.92573  39.81687  14.25740
## 2     4       82 604.66673 112.66174  45.67153
## 3     5      216 414.29233 228.36691 120.34076
## 4     6      299 226.78000 296.87794 239.34207
## 5     7      269 118.18036 276.76512 368.05452
## 6     8      242  67.95755 217.71490 477.32755
## 7     9      190  44.76844 158.47863 559.75293
## 8    10      125  33.23410 111.17714 618.58876
## 9    11       81  27.02160  76.52240 659.45599
## 10   12       52  23.45243  52.11566 687.43191
## 11   13       25  21.30197  35.27341 706.42461
## 12   14       22  19.96202  23.78333 719.25465
## 13   15        7  19.10746  15.99762 727.89492
```

The column "infected" contains the data. The columns S, I and R correspond to the number of individuals in the compartments S, I and R respectively. The column time indicates the time in days.

Question 1

Using the previous code, visualise the number of infected boys over time and compare the data with model simulation for various parameters

Question 2

We assume that the number of infected boys observed each day follows a Poisson distribution whose parameter is the number of infected simulated by the model.

Prior distributions are chosen as follows: Unif[0,10] for the transmission rate λ, Unif[0,1] pour le cure rate γ.

a) With R, write a function that calculates the model's likelihood for a given set of parameters. Write another function that calculates the model's log-likelihood (logarithm of the likelihood).
b) With R, write a function that calculates the model's posterior distribution for a given set of parameters. Write another function that calculates the model's log-posterior (logarithm of the posterior).

Help: These R functions might be helpful: dunif, dpois.

Question 3

Implement the Metropolis-Hastings algorithm in R.

We can use the following proposal distribution:

```
#SIMULATE PARAMETER SET FROM PROPOSAL
proposal = function(theta){
  return(c("lambda"=rnorm(1,mean=theta[["lambda"]],sd=0.01),
           "gamma"=rnorm(1,mean=theta[["gamma"]],sd=0.01),
           "initI"=1,
           "N"=763))
}

proposal_density = function(theta_before, theta_after){
  return(dnorm(theta_after[["lambda"]],mean=theta_before[["lambda"]],sd=0.01)*
           dnorm(theta_after[["gamma"]],mean=theta_before[["gamma"]],sd=0.01))
}
```

Help: in order to evaluate the acceptance probability, you can use this function: runif

Question 4

Apply the algorithm to the data and comment the results. Can you check if the MCMC chain has converged?

Question 5

Compare the fitted model and the data. Try to include uncertainty in fitted parameters in your simulations.