Estimating dynamical parameters

Micaela Martinez: Adapted from John M. Drake & Pejman Rohani SISMID 2019

July 1, 2021

In order to completely describe the epidemic trajectory we require estimates for all parameters of a dynamic model. In this exercise, we introduce a simple approach to model estimation called least squares fitting, sometimes called trajectory matching. The basic idea is that we find the values of the model parameters that minimize the squared differences between model predictions and the observed data. To demonstrate least squares fitting, we consider an outbreak of measles in Niamey, Niger, reported on by Grais et al. 2006 (Grais, R.F., et al. 2006. Estimating transmission intensity for a measles outbreak in Niamey, Niger: lessons for intervention. Transactions of the Royal Society of Tropical Medicine and Hygiene 100:867-873.).

1 Data

Let's load and plot the data:

```
# load the data
niamey<- read.csv(file='Niamey_measles_data.csv',header=TRUE)

# plot the data
plot(niamey$biweek,niamey$cases,type='p',col='grey',xlab='Biweek'
,ylab='Cases')
lines(niamey$biweek[niamey$site==1],niamey$cases[niamey$site==1],col='black')
lines(niamey$biweek[niamey$site==2],niamey$cases[niamey$site==2],col='red')
lines(niamey$biweek[niamey$site==3],niamey$cases[niamey$site==3],col='blue')
```

Question. Now that you have plotted the cases from each site, what are some of the differences you observed between the three sites?

2 SIR Model

Next we write a function for simulating the SIR model in a case where the recovery rate is predefined and there are no individuals entering or exiting the population (i.e., no demography). In the code we are using X to represent susceptibles, Y to represent infected individuals, and we are not explicitely keeping track of recovered individuals. We are assuming that there is a 13 day infectious period, represented in units of a year: 13/365 yr; thus, the recovery rate is 365/13.

```
# SIR model equations
closed.sir.model <- function (t, x, params) {
  X <- x[1]
  Y <- x[2]
  beta <- params
  dX <- -beta*X*Y
  dY <- beta*X*Y-(365/13)*Y
```

```
  list(c(dX,dY))
}
```

3 Objective function

Now we set up a function that will calculate the sum of the squared differences between the observations and the model at any parameterization, more commonly known as sum of squared errors (SSE). In this example the SSE is our objective function because it is the quantity that we seek to optimize. Here we will minimize the SSE in the optimization process.

```
# define function to calculate sum of squared errors
sse.sir <- function(params0,data,site){
  data<-data[data$site==site,] # working dataset, based on site
  t <- data[,1]*14/365              # time in biweeks
  cases <- data[,3]                 # number of cases
  beta <- exp(params0[1])           # parameter beta
  X0 <- exp(params0[2])             # initial susceptibles
  Y0 <- exp(params0[3])             # initial infected
  # output dataframe with simulated cases from the SIR model
  out <- as.data.frame(ode(c(X=X0,Y=Y0),times=t,closed.sir.model,beta,hmax=1/120))
  sse<-sum((out$Y-cases)^2)         # sum of squared errors
}
```

Notice that the code for sse.sir makes use of the following modeling trick. We know that beta, X0 (initial susceptible), and Y0 (initial infected) must be positive, but our search to optimize these parameters will be over the entire number line. We could constrain the search using a more sophisticated algorithm, but this might introduce other problems (i.e., stability at the boundaries). Instead, we parameterize our objective function (sse.sir) in terms of some alternatve variables log(beta), log(X0), and log(Y0). While these numbers range from $-\infty$ to ∞ (the range of our search) they map to our model parameters on a range from 0 to ∞ (the range that is biologically meaningful).

3.1 Optimization

Our final step is to use the function optim() to find the values of beta, X0, and Y0 that minimize the sum of squared errors as calculated using our function. The optim() function will try out different values and try to identify the combination of values that minimizes the SSE.

```
# use the optim() function to fit the model
# fit by minimizing the SSE

# define vector of initial guesses for the parameters
# initial gesses are on a log scale: log(beta), log(X0), log(YO)
params0<-c(-3.2,7.3,-2.6)

# run the optimizer for each time series individually
fit1 <- optim(params0,sse.sir,data=niamey,site=1) # fit site 1
fit2 <- optim(params0,sse.sir,data=niamey,site=2) # fit site 2
fit3 <- optim(params0,sse.sir,data=niamey,site=3) # fit site 3

# save results from optim and back-transform parameters to the natural scale
fit1.pars<- exp(fit1$par)
fit2.pars<- exp(fit2$par)
fit3.pars<- exp(fit3$par)
```

```
names(fit1.pars)<- c('beta','initial.susceptible','initial.infected')
names(fit2.pars)<- c('beta','initial.susceptible','initial.infected')
names(fit3.pars)<- c('beta','initial.susceptible','initial.infected')

# the SSE for each fit
sse.fit1<- fit1$value
sse.fit2<- fit2$value
sse.fit3<- fit3$value
```

Question. Look at the fitted parameter values, specifically the initial number of infected idividuals in each site (i.e., the third entry in the fitted parameter vectors). Do these values seem realistic to you?

4 Simulate the fitted models

Now that the model has been fit independently to each time series, we can simulate the model for each site and compare the simulated cases to the actual cases to see how well the simulated trajectories capture the dynamics of the real-world epidemic curves.

To simulate the fitted model for site 1, take the fitted parameters from site 1 and use them to run the SIR model. Then compare the simulated infections with the reported cases.

```
# make a time vector in units of a year
t <- subset(niamey,site==1)[,'biweek']*14/365

# ode() to simulate for each site using the parameter estimates from optim()
# note the input for the ode solver is:
# ode(y, times, func, parms)
sim.site1<- ode(c(X=fit1.pars[2],Y=fit1.pars[3]),times=t,
  closed.sir.model,fit1.pars[1],hmax=1/120)

# put the simulations in a data frame
sim.site1<- as.data.frame(sim.site1)

# plot the simulated susceptibles and infected for site 1
dev.new()
par(mfrow=c(1,2))
plot(t,sim.site1$X,col='black',type='l',xlab='time (years)',
  ylab='Susceptible Individuals')
title('Site 1 simulated')
plot(t,sim.site1$Y,col='red',type='l',xlab='time (years)',
  ylab='Infected Individuals')

# compare the actual cases in site 1 with the simulated infections
dev.new()
plot(t,sim.site1$Y,col='red',type='l',xlab='time (years)',
  ylab='Infected Individuals')
lines(t,subset(niamey,site==1)[,'cases'],col='black',lwd=2)
legend(0.1,1000,legend=c("data",'simulated infections'),
  col=c('black','red'),lwd=c(2,1))
```

Task. On your own, write code to simulate the fitted model for sites 2 and 3. You can modify the code for site 1 to do this.

5 Testing alternative assumptions

Next let's test alternative assumptions. Assume that this was a new introduction of infection and there was 1 infected individual at time 0; therefore, we only need to estimate the initial number of susceptibles and the transmission rate. To do so we need to change our objective function such that $Y0$ is set to 1.

```
# make a version 2 of our objective function with a fixed value for Y0

sse.sir.v2 <- function(params0,data,site){
  data<-data[data$site==site,]     # working dataset, based on site
  t <- data[,1]*14/365             # time in biweeks
  cases <- data[,3]                # number of cases
  beta <- exp(params0[1])          # parameter beta
  X0 <- exp(params0[2])            # initial susceptibles
  Y0 <- 1                          # initial infected assumed to be 1
  # output dataframe
  out <- as.data.frame(ode(c(X=X0,Y=Y0),times=t,
    closed.sir.model,beta,hmax=1/120))
  sse<-sum((out$Y-cases)^2)        # sum of squared errors
}

# Now we only need initial guesses for log(beta) and log(X0)
params0.v2<-c(-3.2,7.3)
```

Tasks & Questions. Fit the model to the data for each site with $Y0 = 1$. You should first use optim() to fit the model to each site then extract and back transform the results and run the new simulations.

How do the estimates for initial susceptibles and the transmission rate change now that you assume the epidemic was seeded by 1 infected individual at time 0?

Use plots to compare the actual cases for site 1 with the simulated infections from the two fitted models?

Based on the SSE, was does the model fit better when we assume 1 infeted as the initial condition or the original estimate from optim()? Which estimates are most realistic and does this differ by site?

6 Additional Exercises

Go back to your initial optimazation function sse.sir and try running optim() with different starting guesses for beta and the initial conditions. Do the results differ?

Try to fit the infectious period and the transmission rate. To do this you will need to create a new SIR model function with a *gamma* parameter for the recovery rate. Remember that the recovery rate is the inverse of the infectious period. You will also need to make a new sse function that also includes *gamma* as a parameter to be optimized.