SISMID Exercise 2 – Solution

Fanny Bergström

2023-06-01

Exercise 2.1 (Estimation of R_0)

Assume that a large outbreak occurs in a homogeneously mixing population.

(a) First assuming there is no preventive measures, estimate the R_0 if there were 20% infected during the outbreak.

```
################
# YOUR CODE
################
## first write the estimated R_0 as a function of the final size tau
R0 <- function(tau) {
  -log(1 - tau) / tau
}
## compute the estimated value of R_0 when r=0.2
R0_hat <- R0(tau = 0.2)
R0_hat
```

```
## [1] 1.115718
```

(b) Suppose now there were a fraction 30% of initially immune. Estimate R_0 in this case.

```
# fraction r of initially immune
r <- 0.3
################
# YOUR CODE
################

## write the estimated R_0 when there is immune people
# note this tau represents the raction infected among those initially susceptibles
R0_immune <- function(tau) {
  -log(1 - tau) / (tau * (1 - r))
}
# Note that the overall fraction infected that we observed equals tau * (1-r)!
tau_overall <- 0.2

## compute the estimated value of R_0
R0_hat <- R0_immune(tau = tau_overall / (1 - r))
R0_hat
```

```
## [1] 1.682361
```

Exercise 2.2 (Least-squares Fit the SIR model)

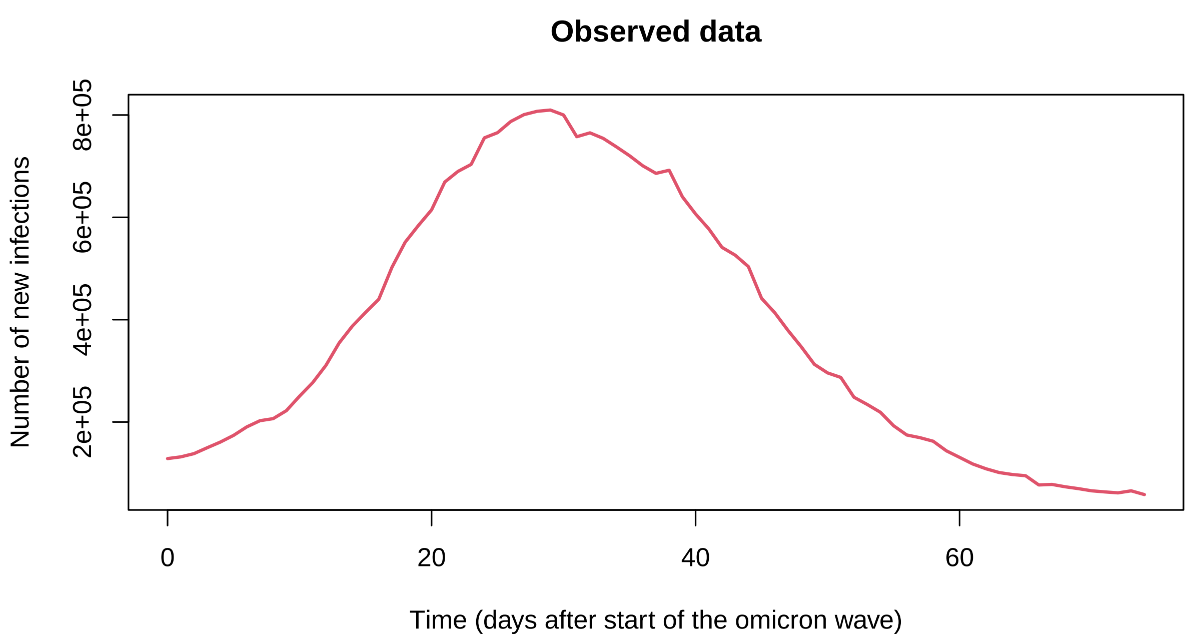
(a) First define the log-likelihood function.

```
## First, we plot the observed data:

covid <- read_csv("../data/WHO-COVID-19-global-data.csv")

covid_df <- covid %>%
  mutate(I = zoo::rollmean(New_cases, 7, na.pad = T)) %>%
  filter(Country_code == "US", Date_reported > "2021-12-15", Date_reported < "2022-03-01") %>%
  mutate(t = 0:74, I = I)

plot(covid_df$t, covid_df$I,
  lwd = 2, type = "l", lty = 1,
  ylab = "Number of new infections",
  xlab = "Time (days after start of the omicron wave)", col = 2
)
title("Observed data")
```




```
# We see that the USA has approximately 331.9 mil inhabitants pig herds
# We assume 3.3 mil infected detected
N <- 331.9 * 10^6
sum_y <- sum(covid_df$New_cases)
f_size <- sum_y / N
R0_hat <- -log(1 - f_size) / f_size

#################################################
# YOUR CODE Please fill in the blank spaces###
#################################################
```

```r
## construct a deterministic SIR Model as same in exercise 2.1

deter_sir <- function(t, y, parms) {
  beta <- parms[1]
  gamma <- parms[2]
  S <- y[1]
  I <- y[2]
  return(list(c(S = -beta / N * S * I, I = beta / N * S * I - gamma * I)))
}

##  create a log-likelihood function of theta

ll.gauss <- function(theta) {
  # Solve ODE using the parameter vector theta
  res <- lsoda(
    y = c(N - covid_df$I[1], covid_df$I[1]), # initial conditions
    times = covid_df$t,
    func = deter_sir,
    parms = exp(theta)
  ) # note here parms = e^(theta)

  return(sum(dnorm(covid_df$I, # the observed number of infected case
    mean = res[, 3], # the solution from the deterministic sir
    sd = 1, # the fixed variance of observation noise is 1
    log = TRUE
  )))
}
```

(b) Maximize the log-likelihood using function *optim* and compute the MLE.

```r
################################################
# YOUR CODE Please fill in the blank spaces###
################################################
# Determine MLE
mle <- optim(log(c(1, 3)), # initial values for theta to be optimized over, e.g. log3
  fn = ll.gauss, # our function,
  control = list(fnscale = -1) ## maximize the function
)
# Show parameter estimates
beta.hat <- exp(mle$par)[1]
beta.hat
```

```
## [1] 1.547095
```

```r
gamma.hat <- exp(mle$par)[2]
gamma.hat
```

```
## [1] 1.450124
```

```r
# and resulting R0 estimate
R0.hat <- beta.hat / gamma.hat

R0.hat
```

```
## [1] 1.066871
```

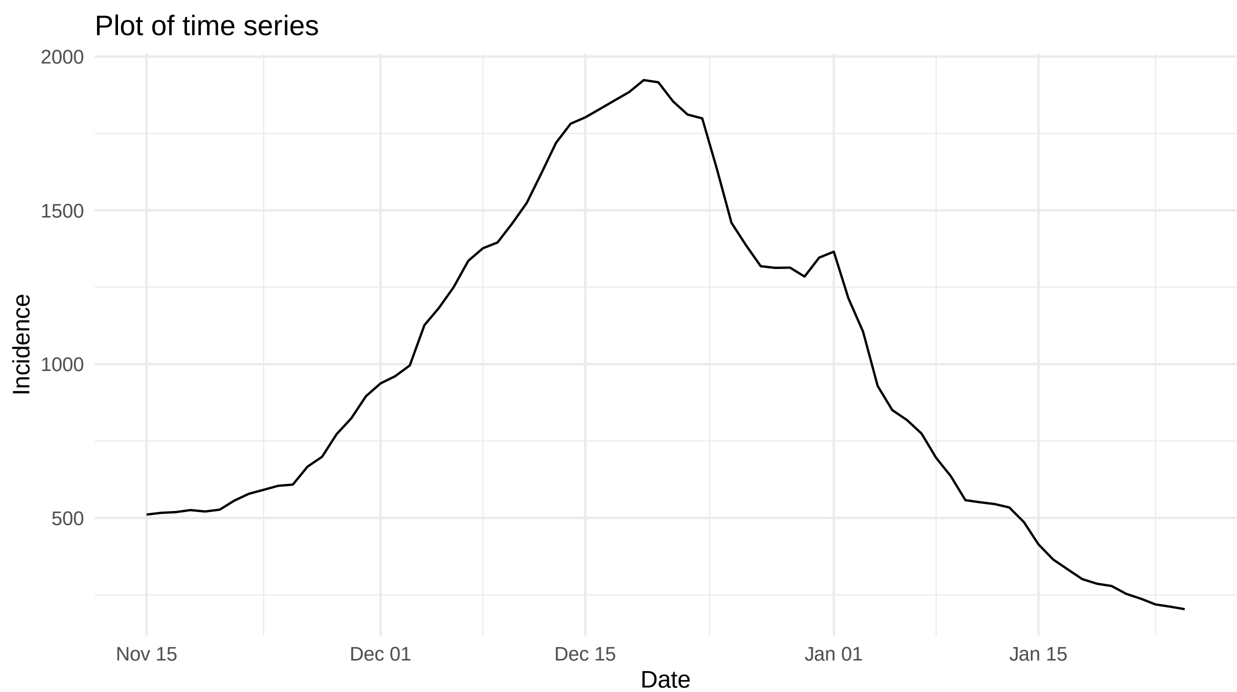
(c) Inserting the values of MLE to find the solution of SIR differential equation system. And plot the fitted curve and the real data of COVID-19 outbreak together. Does it fit well?

```
#################################################
# YOUR CODE Please fill in the blank spaces###
#################################################
## find the solution of SIR differential equation system.
mu <- lsoda(
  y = c(N - covid_df$I[1], covid_df$I[1]), # initial condition
  times = covid_df$t,
  func = deter_sir,
  parms = exp(mle$par)
) # parameters take the the values of MLE from part b

## plot the fitted curve and the real data of covid outbreak
matplot(mu[, 1], # time
  cbind(covid_df$I, mu[, 3]), # cbind(the real data of I, the fitted values)
  type = "l", lwd = 3, lty = 1,
  ylab = "Number of new infectiion",
  xlab = "Time (days after first infection)", col = c(2, 3)
)
legend(x = "topright", c("COVID outbreak", "LS fit"), lty = 1, col = c(2, 3), lwd = 3)
```

Exercise 2.3 (Least-squares Fit the SEIR model)

```r
## Filter the data
ts <- covid %>%
  mutate(Incidence = zoo::rollmean(New_cases, 7, na.pad = T), Date = Date_reported) %>%
  filter(Country == "Sweden", Date_reported > "2022-11-14", Date_reported < "2023-01-26") %>%
  mutate(t = 0:71)
## let's plot of time series
ggplot(ts, aes(x = Date, y = Incidence)) +
  geom_line() +
  ggtitle(expression("Plot of time series")) +
  theme_minimal()
```




```r
N <- 11 * 10^6 # the size of Swedish population
I0 <- ts$Incidence[1] %>% round() # number of initial infectives
################################################
# YOUR CODE Please fill in the blank spaces###
################################################
## create a deterministic SEIR model with time-dependent beta(t) as in exercise 1.3

seir_change <- function(t, y, parms) {
  beta0 <- parms[1]
  beta1 <- parms[2]
  t1 <- parms[3]
  w <- parms[4]
  rho <- 1 / 5
  gamma <- parms[5]

  S <- y[1]
  E <- y[2]
  I <- y[3]
```

```
  # time-dependent rate \beta(t):

  beta <- function(t) {
    ifelse(t <= t1 - w, beta0, ifelse(t > t1 + w, beta1,
      beta0 + (beta1 - beta0) / (2 * w) * (t - (t1 - w))
    ))
  }

  return(list(c(
    S = -beta(t) / N * S * I, E = beta(t) / N * S * I - rho * E,
    I = rho * E - gamma * I
  )))
}
```

(a) The parameters to optimize for are $\theta = (\beta_0, \beta_1, t_1, w, \gamma)'$. Use the simple least-squares approach for fitting and report your estimate for θ.

```
#################################################
# YOUR CODE Please fill in the blank spaces###
#################################################

# least-squares fit:

ll.sq <- function(theta, I0) {
  "
  This function takes a vector theta (consisting of the parameters beta0,
  beta1, t1, w, gamma) as the input,
  and it solves the ODE system (of the SEIR model) using the exponential
  values of the parameters.
  (Use the log of the parameters to ensure valid parameter values at all times.)
  The function returns:
  the sum over all the (number of incidence I - deterministic I(t))^2.
  "


  res <- lsoda(
    y = c(N - I0, I0, I0),
    times = ts$t,
    func = seir_change,
    parms = exp(theta)
  )

  return(sum((ts$Incidence - res[, 3])^2))
}
```

```
##  next, set starting values: (beta_0, beta_1, t_1, w,gamma)
# Note: hand-tuning the starting values of your optimization
# might be necessary in order to get a reasonable fit.
# Try out more starting values.
# such as  t1 is not too small, gamma not too large, beta1 shall be smaller than beta0...

#################################################
```

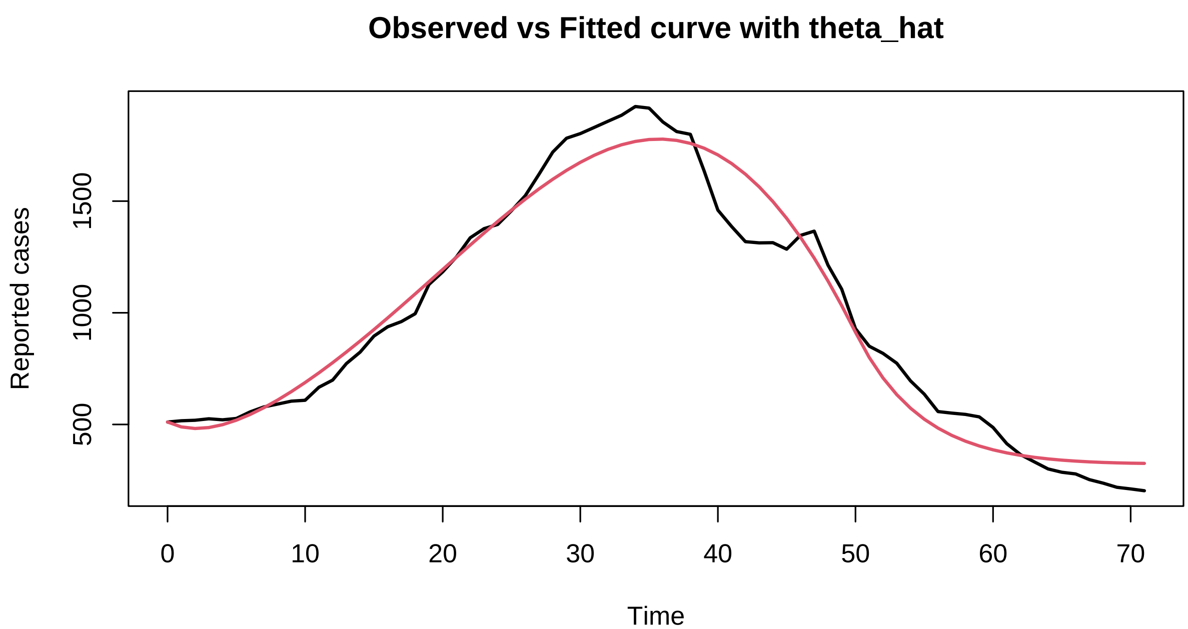
```r
# YOUR CODE Please fill in the blank spaces###
#################################################
##  Set starting values: (beta_0,beta_1, t_1, w,gamma)
theta0 <- c(0.07, 0.02, 3, 23, 0.2)


##
theta_hat <- optim(log(theta0),
  fn = ll.sq,
  method = "Nelder-Mead",
  I0 = I0
)

# plug-in of theta_hat:

mu <- lsoda(
  y = c(N - I0, I0, I0), # initial conditions
  times = ts$t, # time
  func = seir_change,
  parms = exp(theta_hat$par)
)

# plot the observed and fitted curve with estimated theta
matplot(mu[, 1], # time
  cbind(ts$Incidence, mu[, 3]), # cbind(incidence data, fitted value mu)
  type = "l", lwd = 2, lty = 1,
  ylab = "Reported cases",
  xlab = "Time"
)
title("Observed vs Fitted curve with theta_hat")
```



```r
# A nicer plot of the number of reported cases per day
ts_df <- ts %>%
  mutate(fitted = mu[, 3]) %>%
  pivot_longer(
    cols = c("fitted", "Incidence"),
    names_to = "type", values_to = "No."
  )

ggplot(ts_df, aes(x = Date, y = No., color = type)) +
  geom_line() +
  ylab("Number of daily reported cases") +
  xlab("Date") +
  scale_color_brewer(palette = "Set2") +
  theme_minimal()
```