감염병 수리 모형 실습

한국보건정보통계학회 춘계연수교육

김종훈

11 June 2021

Goals

1. Implementing a differential equation-based SIR model in R
2. Solving the SIR model using numerical integration (deSolve package and Euler)
3. Implement and numerically solve an $SEIR$ model
4. Estimate a model parameter in $SEIR$ model (e.g., $\mathcal{R}_0$)

Session information

1. R version 4.1.0 (2021-05-18)
2. RStudio Version 1.3.1093
3. packages
    - deSolve: numerical integration of ODE's
    - tidyverse: data manipulation and plotting
    - lubridate: date format
    - xaringan: slide generation

Implementing an SIR model using differentail equation (DE) framework

- Differential equations (DEs) are commonly used to describe the nature.
- DEs describe rate of change of the variables.
- Complex DEs don't have analytic solutions and require numerical integration

SIR model in differential equations

- The SIR model in a closed system

$$
\begin{aligned}
\mu_{\bullet S} &= \mu_{S\bullet} = \mu_{I\bullet} = \mu_{R\bullet} = 0 \\
\mu_{SI} &= \beta \frac{I}{N} \\
\mu_{IR} &= \gamma \\
\frac{dS}{dt} &= -\mu_{SI} S \\
\frac{dI}{dt} &= \mu_{SI} S - \mu_{IR} I \\
\frac{dR}{dt} &= \mu_{IR} I
\end{aligned}
$$

R implementation of the SIR model

```r
closed_sir_model <- function(t,  y, params) {
  ## first extract the state variables
  S <- y[1]
  I <- y[2]
  R <- y[3]
  ## now extract the parameters
  beta <- params["beta"]
  gamma <- params["gamma"]
  ## define variables to be consistent with the figure
  N <- S + I + R
  muSI <- beta*I/N
  muIR <- gamma
  ## now code the model equations
  dSdt <- - muSI*S
  dIdt <- muSI*S - muIR*I
  dRdt <- muIR*I
  ## combine results into a single vector
  dydt <- c(dSdt, dIdt, dRdt)
  ## return result as a list!
  list(dydt)
}
```

Model inputs

- Parameters, initial values, simulation times

```
params <- c(beta = 1/2, gamma = 1/5) # model parameters
times <- seq(from = 0, to = 60, by = 1) # simulation times
y0 <- c(S = 999, I = 1, R = 0) # initial values
```

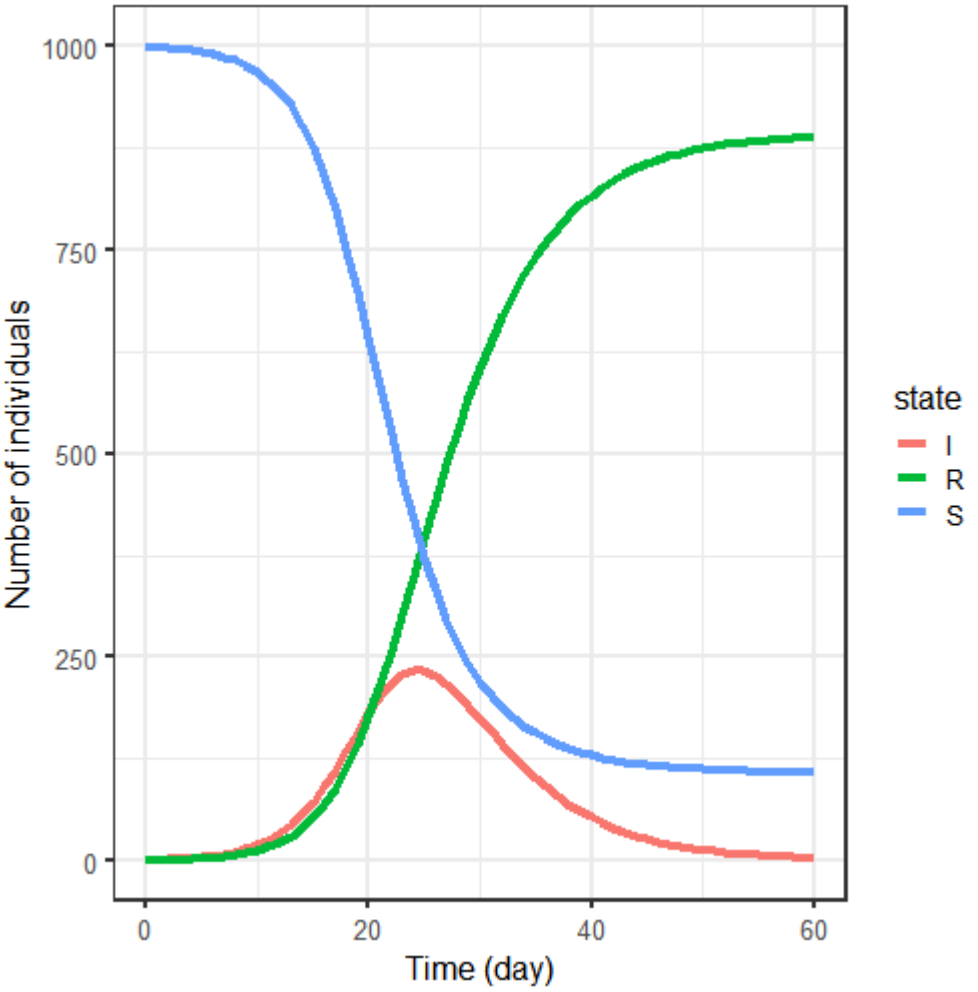
- Simulation using the deSolve::ode function

```
library(deSolve) # numerical integration algorithm for ODE's
library(tidyverse) # data manipulation methods

deSolve::ode(func = closed_sir_model, y = y0, times = times, parms =
  as.data.frame() -> out
```

Exercise 1: Explain β, γ in plain language. What are their units?

Time evolution of S, I, R



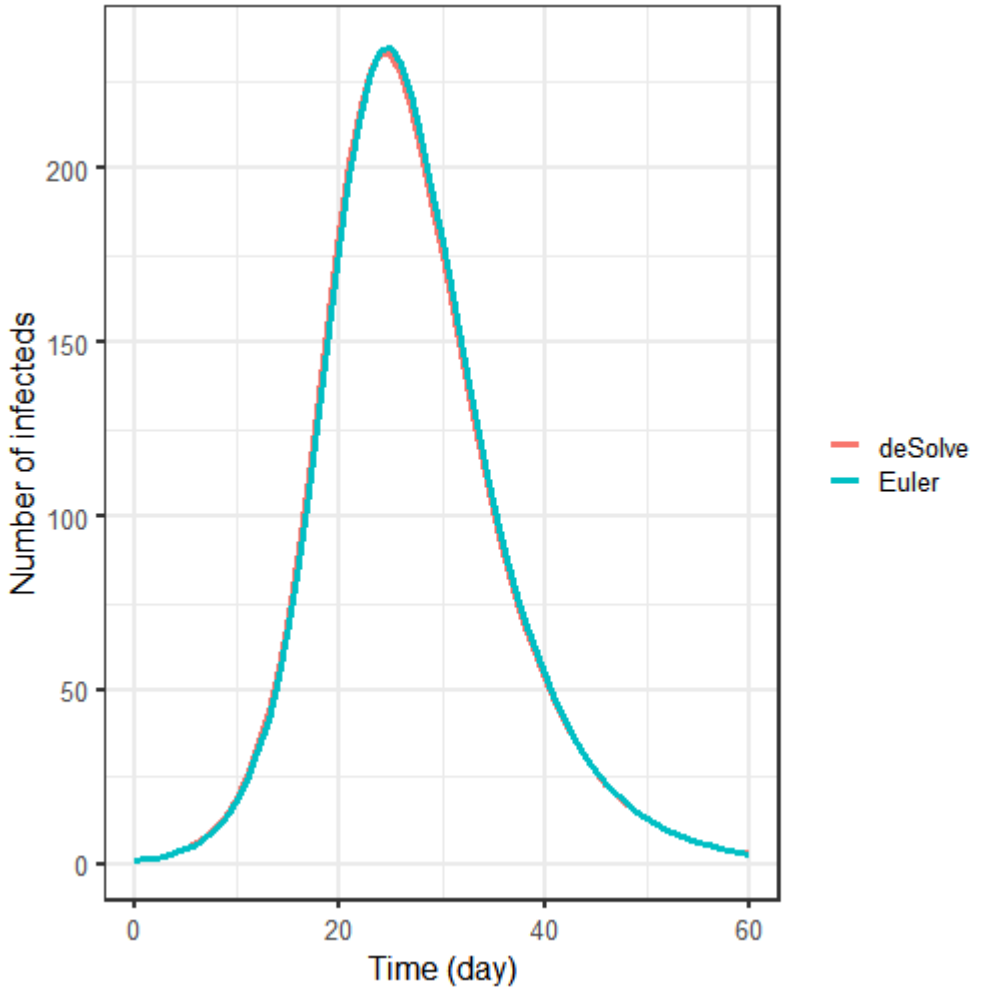
Euler method to solve the SIR model

```r
closed_sir_model_euler <- function(tend, y0, params, dt = 0.1) {
  ## extract the parameters
  beta <- params["beta"]; gamma <- params["gamma"]; muIR <- gamma
  ## simulation time
  t <- seq(0, tend, dt); iter <- length(t)
  ## variables
  S <- I <- R <- rep(NA, iter)
  ## extract the initial values
  S[1] <- y0["S"]; I[1] <- y0["I"]; R[1] <- y0["R"]
  ## now code the model equations
  for (i in 2:iter) {
    N <- S[i-1] + I[i-1] + R[i-1]
    muSI <- beta * I[i-1] / N
    dSdt <- - muSI * S[i-1]
    dIdt <- muSI * S[i-1] - muIR * I[i-1]
    dRdt <- muIR * I[i-1]
    S[i] <- S[i-1] + dSdt * dt
    I[i] <- I[i-1] + dIdt * dt
    R[i] <- R[i-1] + dRdt * dt
  }
  return(data.frame(t = t, S = S, I = I, R = R)) ## combine results i
}
```

Model parameters, initial values, and simulation times

```
params <- c(beta = 1/2, gamma = 1/5) # model parameters
tend <- 60
dt = 0.1
y0 <- c(S = 999, I = 1, R = 0) # initial values
out_euler <- closed_sir_model_euler(tend = tend, y0 = y0, params = pa
```

Check if simulation results match with deSolve



I in response to β, γ

- β, γ range

```
beta_range <- c(0.2, 0.5, 1)
gamma_range <- 1/c(2, 5, 10)
```

- Simulation

```
expand.grid(beta = beta_range, gamma = gamma_range) %>%
  group_by(beta, gamma) %>%
  do({ode(func = closed_sir_model, y = y0, times = times,
          parms = c(beta=.$beta, gamma=.$gamma)) %>%
      as.data.frame()}) %>%
  ggplot(aes(x = time, y = I)) +
  geom_line()+
  facet_grid(beta~gamma, scales = 'free_y', labeller = label_both)
```

- Exercise 2: Select one parameter set (e.g., parameters that lead to $R_0 > 1$) and conduct numerical integration of S, I, and R

'manipulate' package

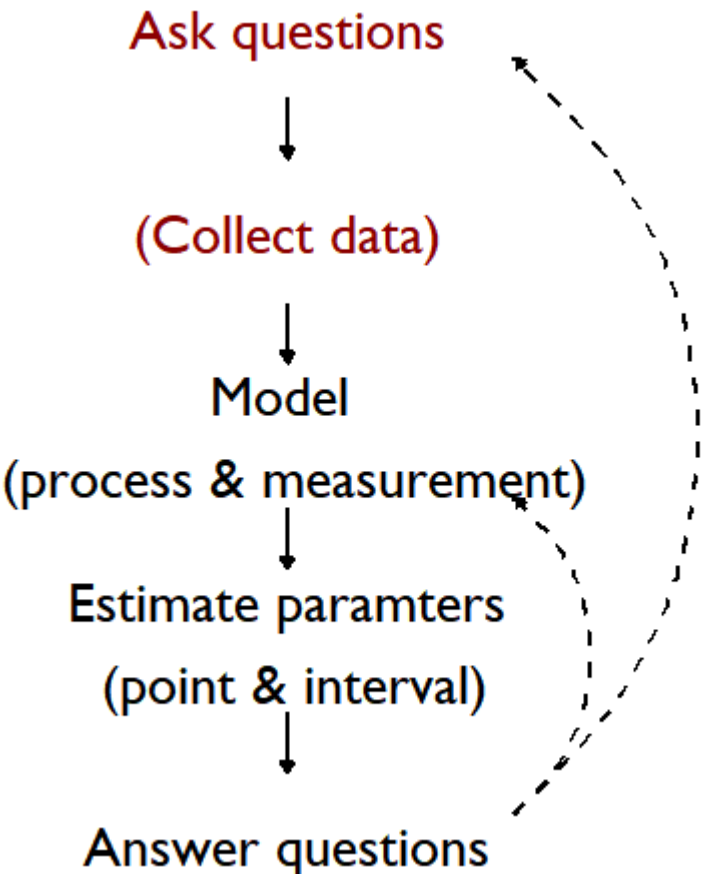
```
library(manipulate)
tmax <- 60
times <- seq(from = 0, to = tmax, by = 1) # simulation times
y0 <- c(S = 999, I = 1, R = 0) # initial values
beta <- 0.5
gamma <- 0.2
manipulate(
  ode(func = closed_sir_model, y = y0, times = seq(0, tmax, 1),
      parms = c( beta = beta, gamma = gamma)) %>%
  as.data.frame() %>%
  gather(state, value, -time ) %>% # wide to long format data
  ggplot(aes(x = time, y = value, color = state)) +
  geom_line(size = 1.5) +
  labs(x = 'Time (day)', y = 'Number of individuals') +
  theme_bw(base_size = 16),
  beta = slider(0.1, 1),
  gamma = slider(0.1, 1),
  tmax = slider(50, 200))
```

- Exercise 3: Execute the codes above and check how S, I, R change across β and γ

SEIR model practice

Estimate $\mathcal{R}_0$ using Wuhan data

Modeling process: Ask questions, collect data

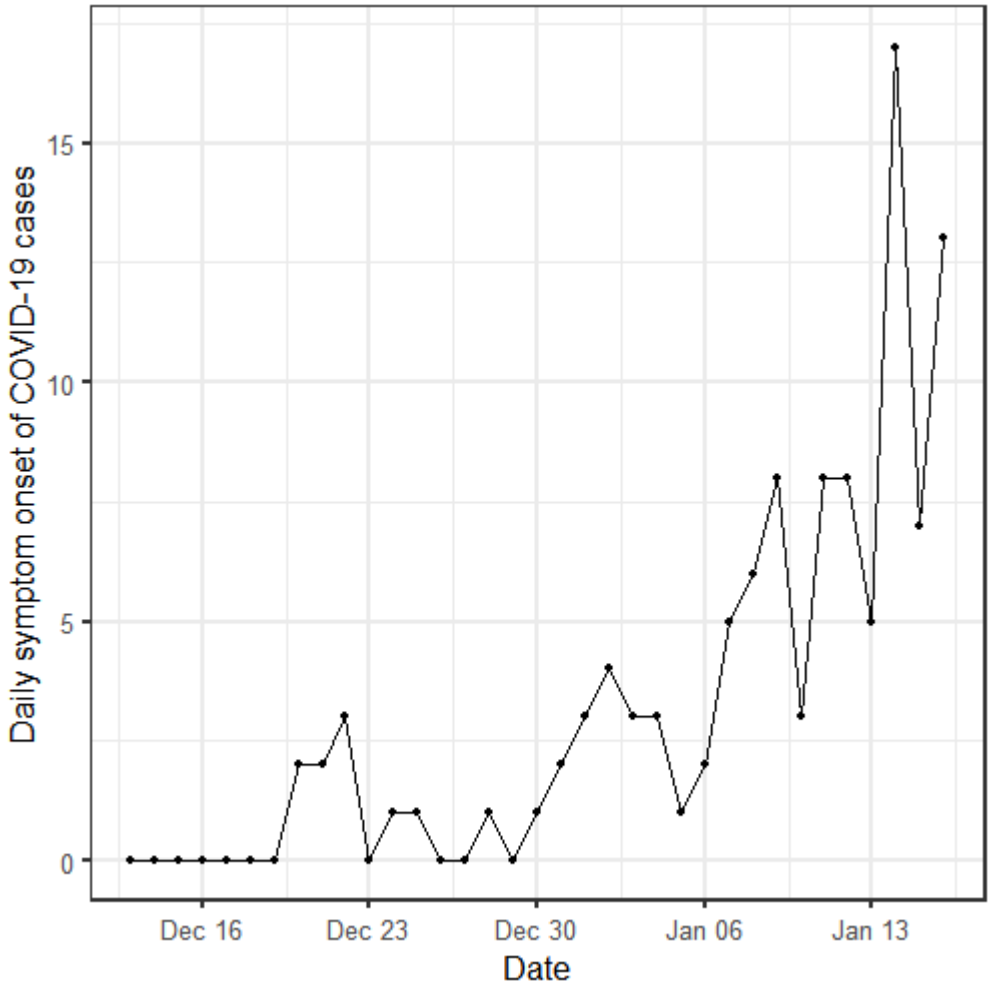


Modeling procedure: Ask questions, collect data

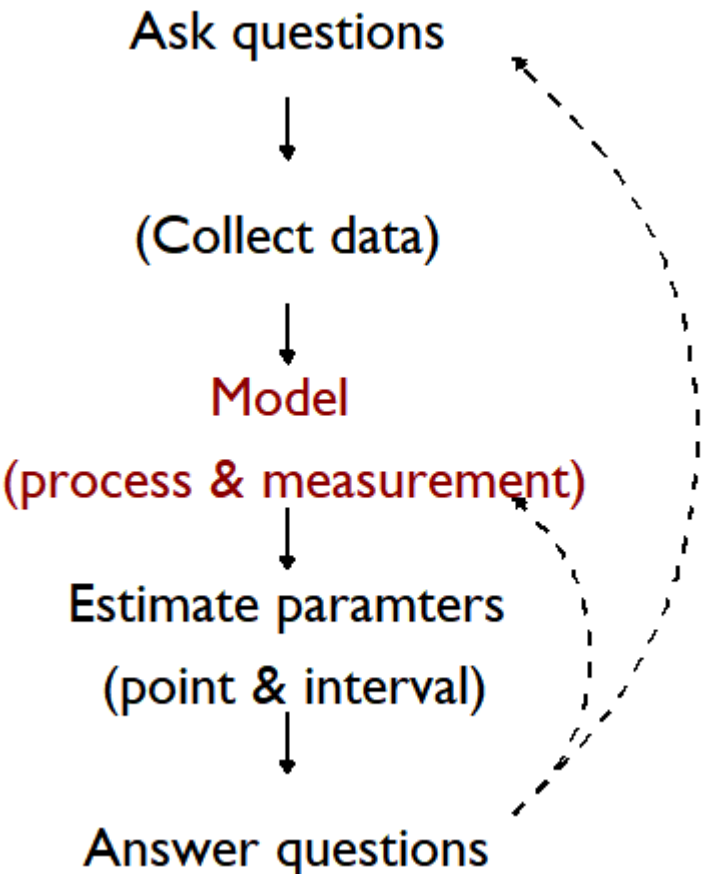
1. Question: Estimate basic reproduction number, $\mathcal{R}_0$, based on the early data from Wuhan
2. Data: Following paper

- Findings: $R_0 = 2.35\ [95\%\text{CI} : 1.15, 4.77]$

Reported daily symptom onset



Modeling procedure: modeling



Modeling the COVID-19 transmission using the SEIR framework

```
seir <- function(t, y, params) {
  S <- y[1]; E <- y[2]; I <- y[3]; R <- y[4]; C <- y[5]
  beta <- params["beta"]
  sigma <- params["sigma"]
  gamma <- params["gamma"]

  muSE <- beta * I / (S + E + I + R)
  muEI <- sigma
  muIR <- gamma

  dS <- - muSE*S
  dE <-  muSE*S - muEI*E
  dI <-  muEI*E - muIR*I
  dR <-  muIR*I
  dC <-  muEI*E ## cumulative symtom onset

  return(list(c(dS, dE, dI, dR, dC)))
}
```

Model inputs

- Parameters, initial values, simulation times

```
y0 <- c(S = 11e6 - 1, E = 0, I = 1, R = 0, C = 1) # initial values
params <- c(beta = 2.5/4.5, sigma = 1/5.2, gamma = 1/4.5)
times <- seq(from = 0, to = 35, by = 1)
```

Exercise 4: What does $1/\sigma$ represent?

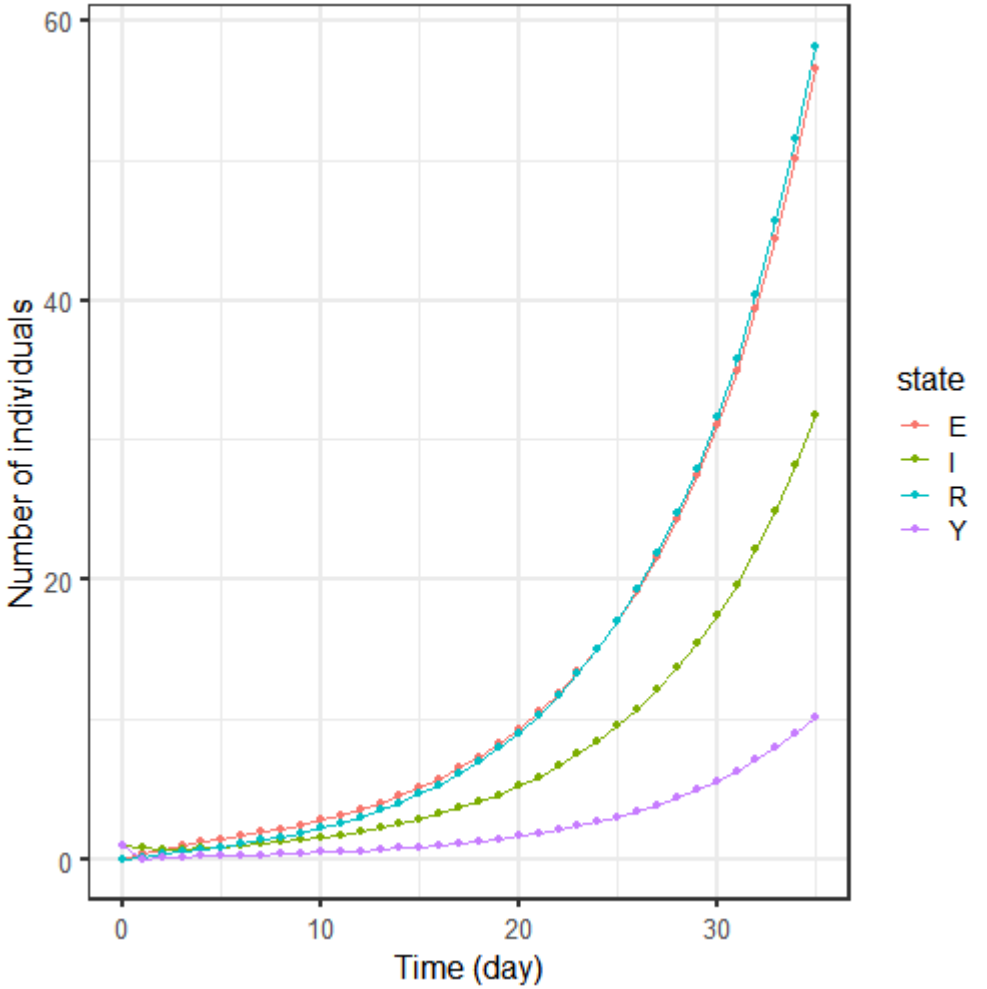
- Simulation

```
library(deSolve)
daily_case <- function( params ){
   ode(y = y0, times = times, func = seir, parms = params) %>%
   as.data.frame() -> x
   n <- nrow(x)
   x[2:n,"C"] - x[1:(n-1),"C"]
}
```

Exercise 5: What is the function 'daily_case' for?

E, I, R, Y over time

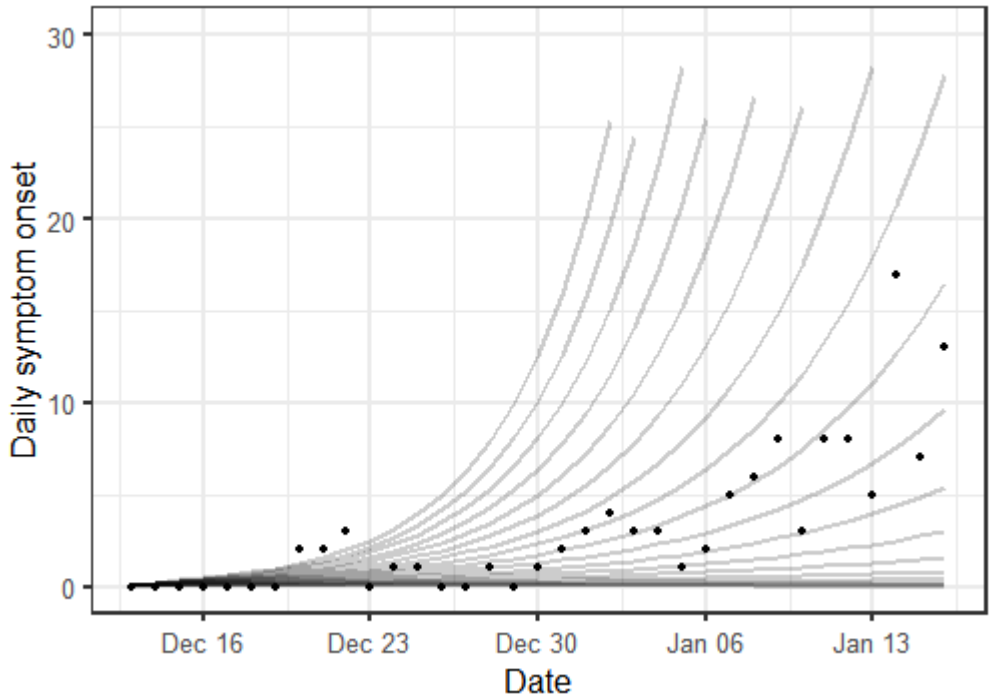
- Daily symptom onset, Y



$\mathcal{R}_0$ vs. daily symptom onset Y

- $\mathcal{R}_0 = \beta/\gamma$. Vary β while keeping γ constant

```
beta <- seq(from = 0.1, to = 1, by = 0.05)
```

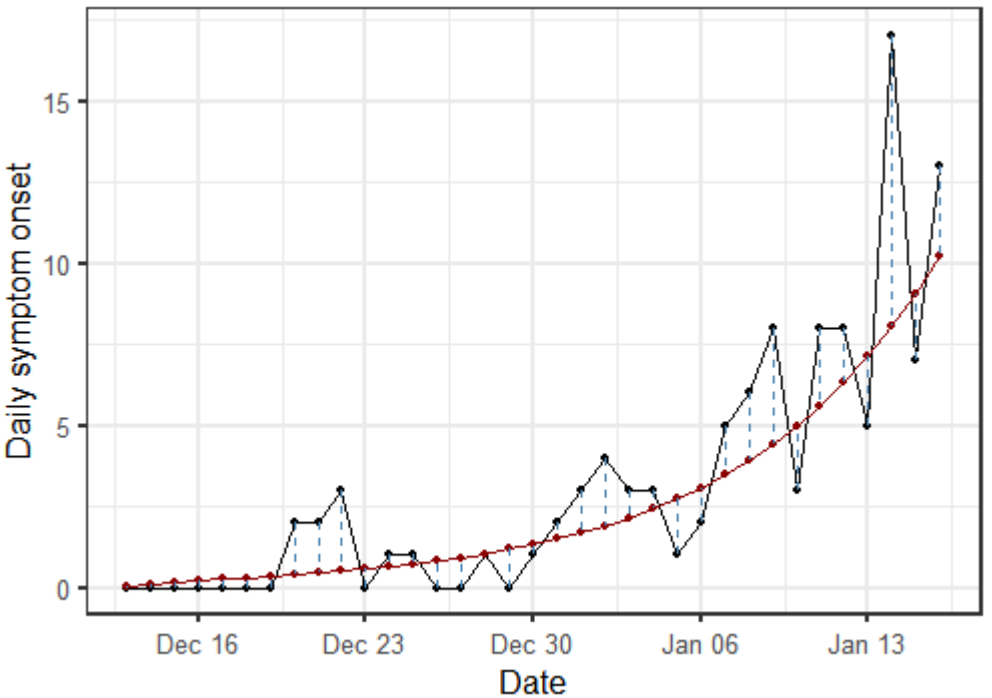


Measurement error modeling

- Poisson distribution

$$y_t \sim \text{Poisson}(Y_t)$$

$$\mathcal{L}(\theta) = \prod_{t=1}^{n} f(y_t|\theta) = \prod_{t=1}^{n} \frac{Y_t^{y_t} e^{-Y_t}}{y_t!}$$



Implementing Poisson likelihood

- 'stats::dpois', log = TRUE

```
poisson_loglik <- function(params) {
  model <- daily_case(params = params)
  sum(dpois(x = wuhan$case, lambda = model, log = TRUE)) # sum of log
}
```

- Define a function that returns poisson_loglik value for a given par value

```
f2 <- function(par) {
  params <-  c(beta = par[1], sigma = 1/5.2, gamma = 1/4.5)
  poisson_loglik(params)
}
```

Maximize the poisson_loglik: grid search

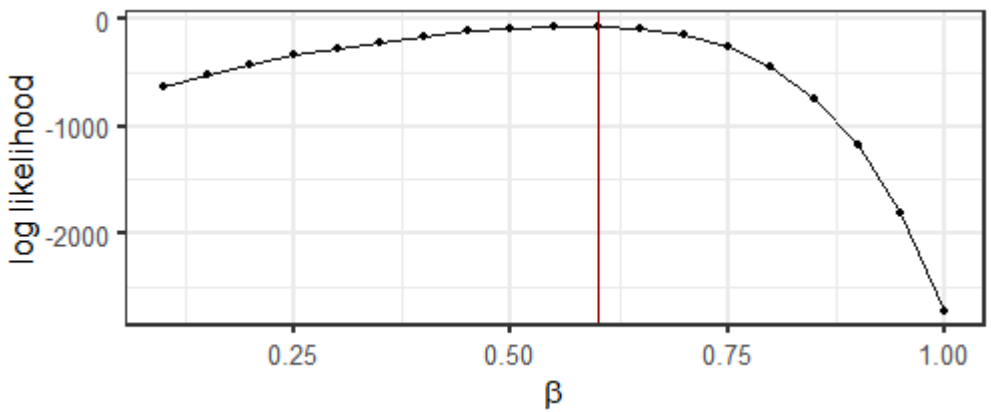
```
res <- data.frame(beta = seq(from = 0.1, to = 1, by = 0.05))
res$loglik <- sapply(res$beta, f2)
(theta <- res$beta[which.max(res$loglik)])
```

```
## [1] 0.6
```

```
gamma <- 1/4.5; (R0 <- theta/gamma)
```

```
## [1] 2.7
```

Exercise 6: Try different resolutions for 'res'. Do you get the same results?



Maximize the poisson_loglik: 'stats::optim'

- Define a function that returns a negative log likelihood

```
f3 <- function(par) {
  params <-  c(beta = par[1], sigma=1/5.2, gamma = 1/4.5)
  - poisson_loglik(params)
}
```

Exercise 7: Why do we define a negative log likelihood function?

- Optimize

```
fit3 <- optim(f3, par = c(0.1), method = "Brent", lower = 0, upper =
(theta <- fit3$par)
```

```
## [1] 0.577124
```

```
gamma <- 1/4.5; (R0 <- theta/gamma)
```

```
## [1] 2.597058
```

Exercise 8: Try different 'par'. Do you get the same result?

Estimate parameters: confidence interval

- 95% confidence interval for $\hat{\theta}$ consists of all the values θ^* for which

$$\log\mathcal{L}(\hat{\theta}) - \log\mathcal{L}(\theta^*) < 1.92$$

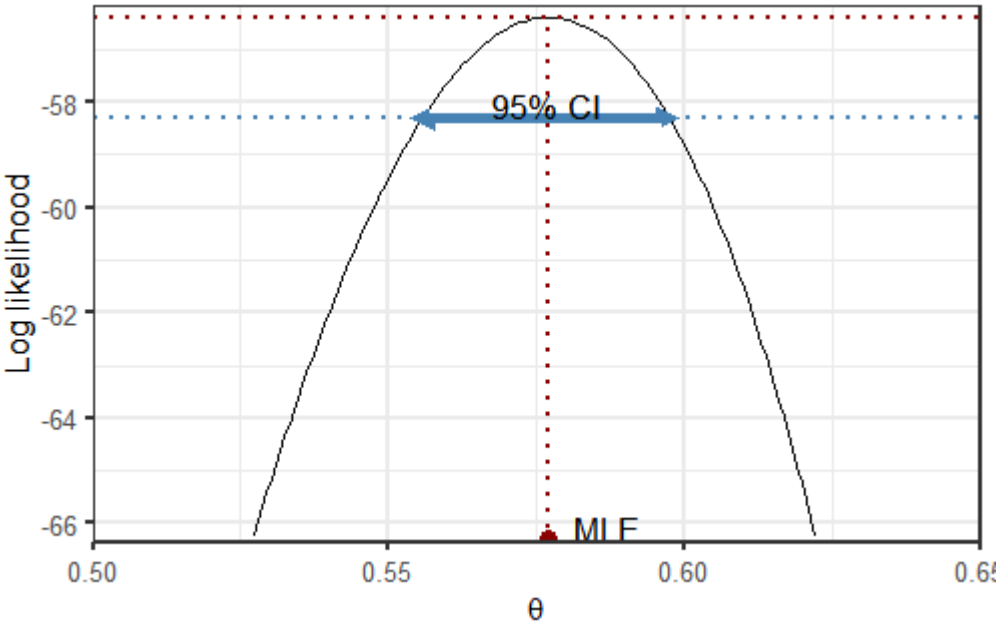
```
maxloglik <- - fit3$value
prof_b <- expand.grid(b = seq(0.5, 0.65, length = 200))
prof_b$loglik <- -sapply(prof_b$b, f3)
cutoff <- maxloglik - qchisq(p = 0.95, df = 1) / 2
(limits <- range(subset(prof_b, loglik > cutoff)$b))
```

```
## [1] 0.5557789 0.5972362
```

```
(R0_interval <- limits/gamma)
```

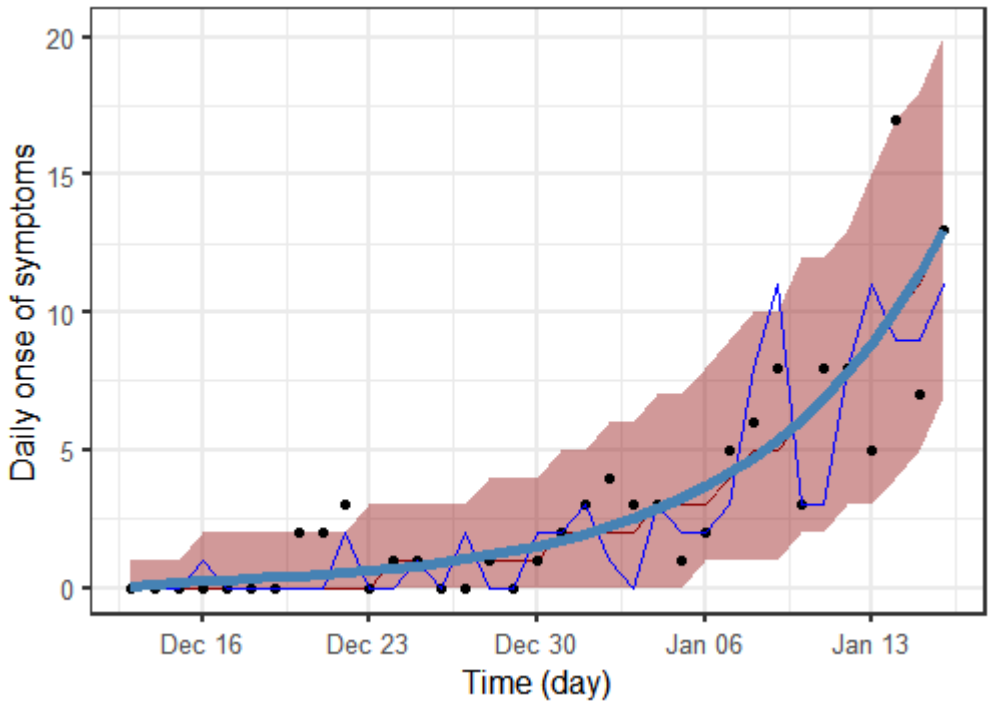
```
## [1] 2.501005 2.687563
```

95% confidence interval



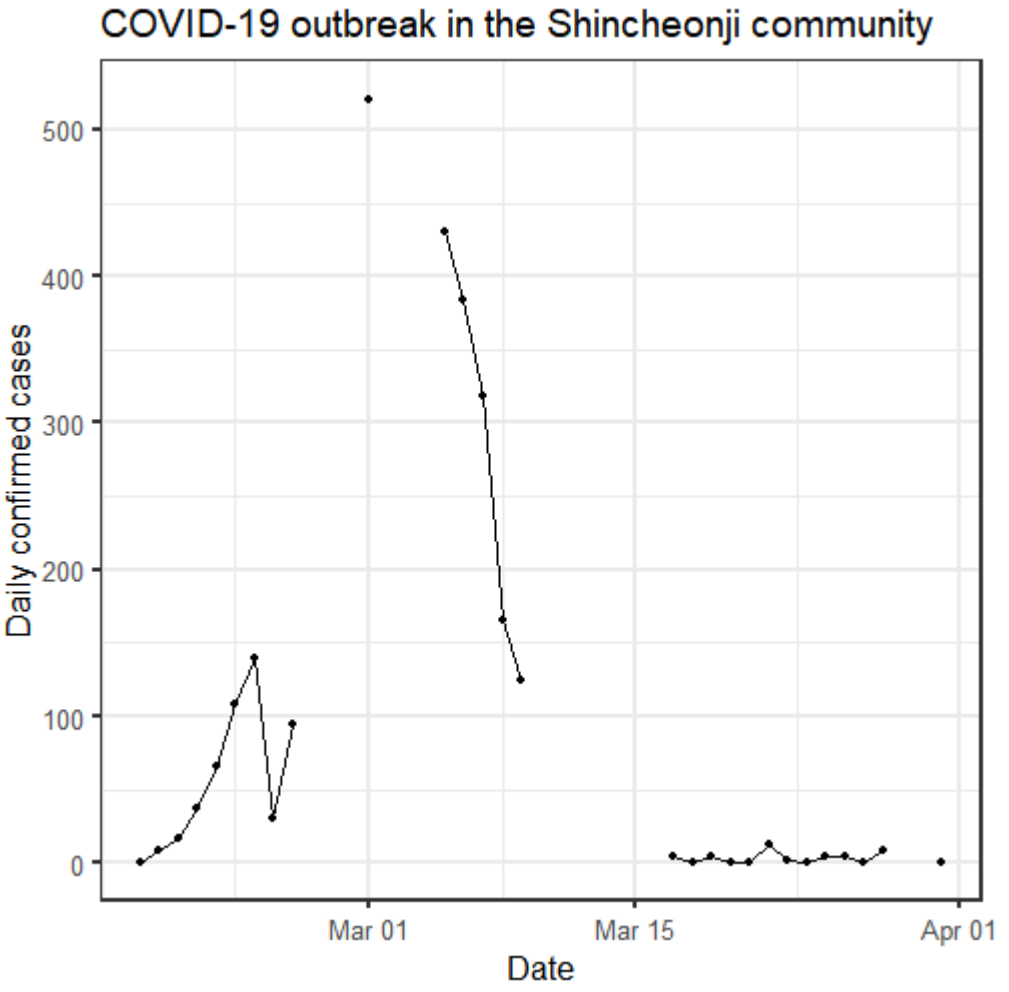
Simulation the model using β_{MLE}

```
params <- c(beta = fit3$par, sigma = 1/5.2, gamma = 1/4.5)
model_pred <- daily_case(params)
simdat <- replicate(2000, rpois(n=length(model_pred), lambda = model_
quantiles <- apply(simdat, 1, quantile, probs = c(0.025,0.5,0.975))
typical <- sample(ncol(simdat), 1)
```



Estimate parameters for the Shincheonji COVID-19 outbreak

Daegu Shincheonji COVID-19



Model inputs

- Daily confirmed cases

```
daily_confirmed <- function(params){
  ode(y = y0, times = times, func = seir, parms = params) %>%
  as.data.frame() -> x
  n <- nrow(x)
  x[2:n, "R"] - x[ 1:(n-1), "R"]
}
```

- Parameters, initial values, simulation times

```
y0 <- c(S = 9334-1, E = 0, I = 1, R = 0, C = 0)
times <- seq(0, 43, 1)
R0 <- 4
params <- c(beta = R0/4.5, sigma = 1/5.2, gamma = 1/4.5)
```

Exercise 9: Examine the 'daily_confirmed' function and params. How long is the average delay from symtom onset to confirmation?

Maximize log likelihood

- negative log likelihood

```
negloglik <- function(params) {
  p <-  c(beta = params[1], sigma = 1/5.2, gamma = 1/4.5)
  model <- daily_confirmed(params = p)
  d <- data.frame( data=dat$daily_confirmed, model = model)
  d <- d[complete.cases(d), ] # remove rows with missing values for t
  - sum(dpois(x=d$data, lambda = d$model, log = TRUE))
}
```

- 'stats::optim'

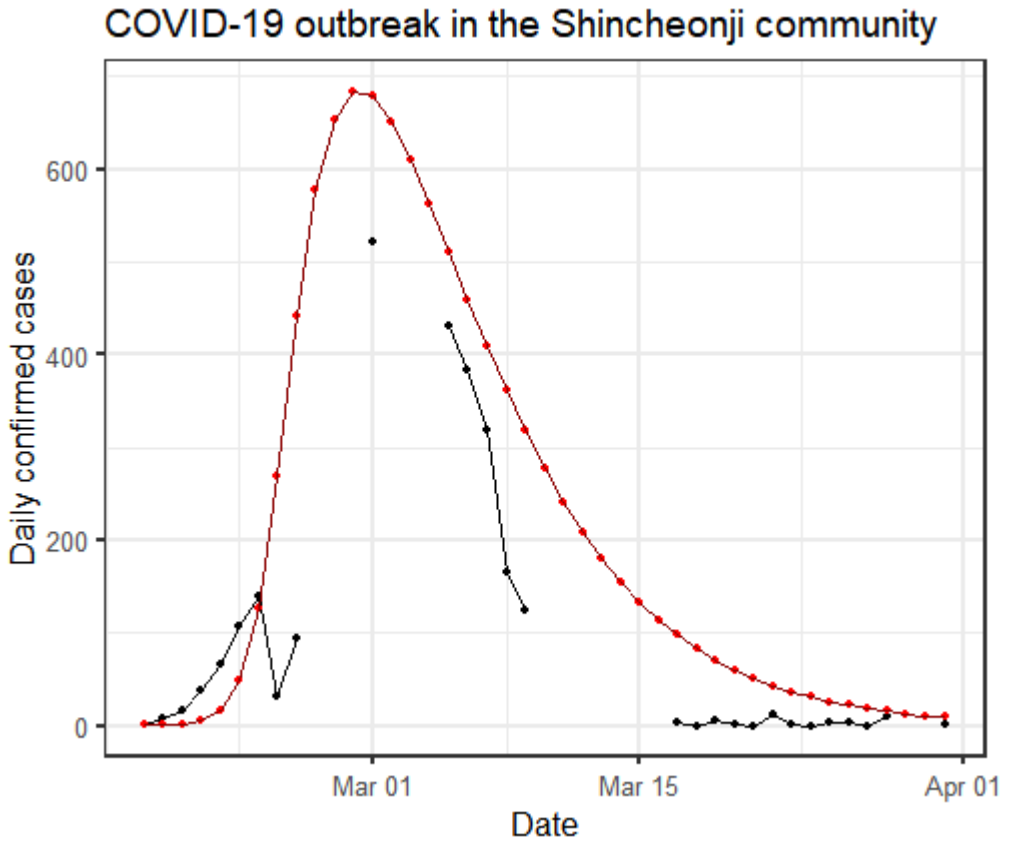
```
fit5 <- optim(par = c(0.5), fn = negloglik,  method = "Brent", lower
fit5$par; gamma <- 1/ 4.5 # 1/gamma = infectious period
```

```
## [1] 9.074375
```

```
(R0 <- fit5$par/gamma) ##
```

```
## [1] 40.83469
```

Comparing model prediction with data



Estimate two paramters: β and initial infecteds

- Define the negative log likelihood function

```
negloglik2 <- function(params) {
  p <- c(beta = params[1], sigma = 1/5.2, gamma = 1/4.5)
  y0 <- c(S = 9334 - params[2], E = 0, I = params[2], R = 0, C = 0)

  model <- daily_confirmed(params = p)
  d <- data.frame(data = dat$daily_confirmed, model = model)
  d <- d[complete.cases(d), ]
  - sum(dpois(x = d$data, lambda = d$model, log = TRUE))
}
```

'stats::optim'

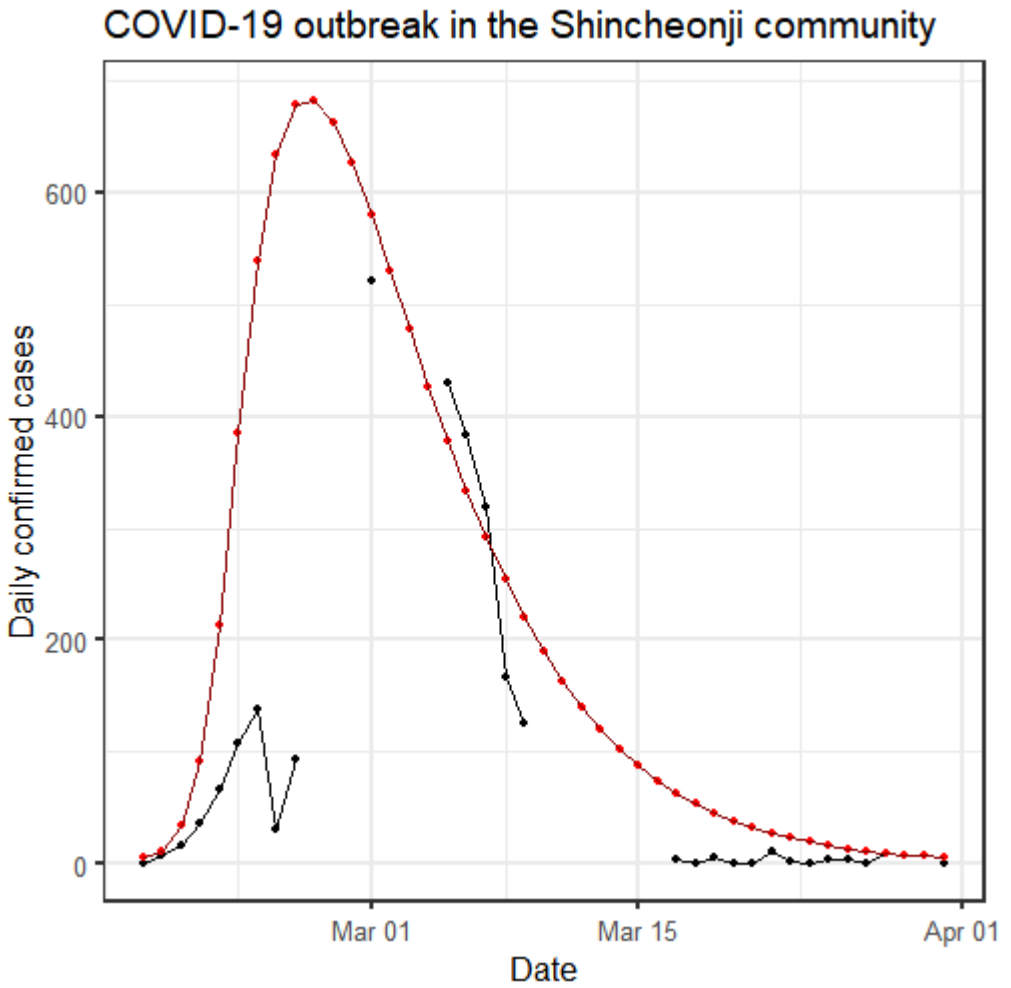
```
fit6 <- optim(par = c(1e-3, 20), fn = negloglik2)
(theta <- fit6$par)
```

```
## [1]  9.075726 19.275180
```

```
gamma <- 1 / 4.5 # 1/gamma = infectious period
(R0 <- theta[1]/gamma)
```

```
## [1] 40.84077
```

Comparing model prediction with data



Estimate parameters: β, γ, and initial infecteds

- Define the negative log likelihood function

```
negloglik3 <- function(params) {
  p <-  c(beta = params[1], sigma = 1/5.2, gamma = params[2])
  y0 <- c(S = 9334 - params[3], E = 0, I = params[3], R = 0, C = 0)

  model <- daily_confirmed(params = p)
  d <- data.frame(data=dat$daily_confirmed, model = model)
  d <- d[complete.cases(d), ]
  - sum(dpois(x = d$data, lambda = d$model, log=TRUE))
}
```

- 'stats::optim'

```
fit8 <- optim(par = c(3, 0.5, 20), fn = negloglik3)
(theta <- exp(fit8$par))
(R0 <- theta[1]/theta[2])
```

Exercise 10: Execute the fit8. Are parameter values reasonable? Why do we get NaN?

Estimate parameters 2: Only positive values!

- negative log likelihood

```
negloglik4 <- function(params) {
  p <-  c(beta = exp(params[1]), sigma=1/5.2, gamma = exp(params[2]))
  y0 <- c(S=9334-params[3], E = 0, I = exp(params[3]), R = 0, C = 0)

  model <- daily_confirmed(params = p)
  d <- data.frame(data = dat$daily_confirmed, model = model)
  d <- d[complete.cases(d), ]
  - sum(dpois(x = d$data, lambda = d$model, log = TRUE))
}
```

'stats::optim'

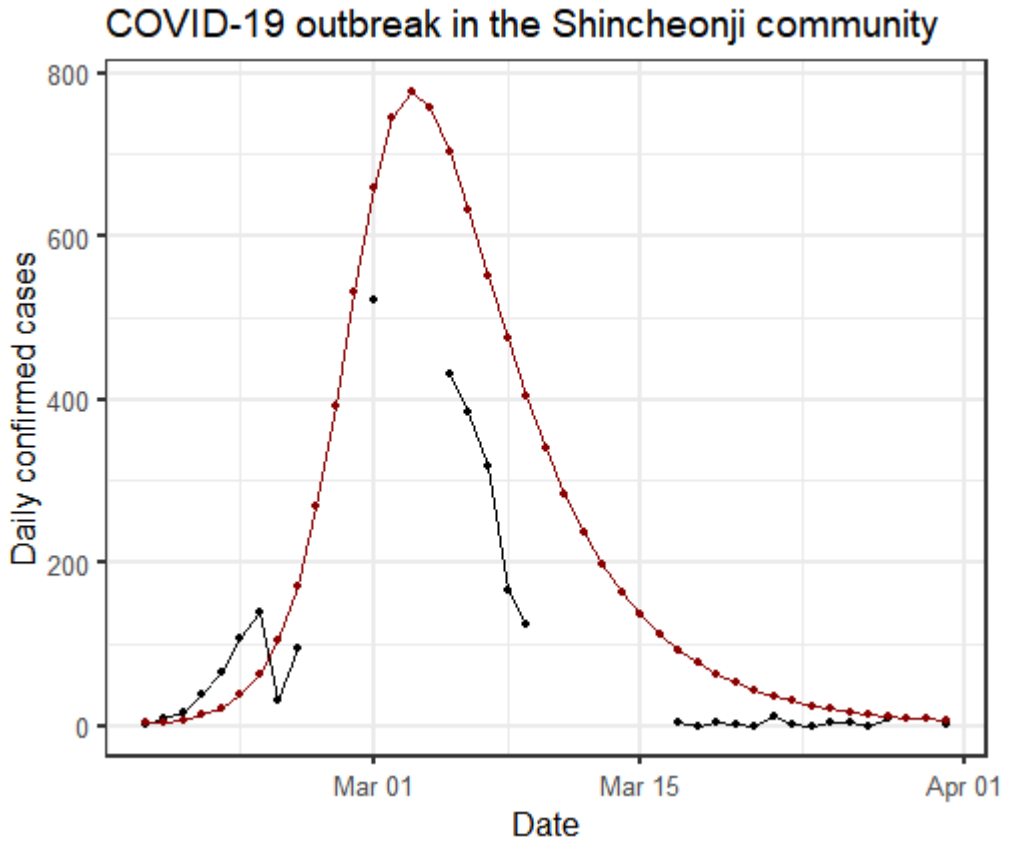
```
fit9 <- optim(par = c(log(2), log(0.4), log(50)), fn = negloglik4)
(theta <- exp(fit9$par))
```

```
## [1] 4.8164855 0.6786712 6.8912274
```

```
(R0 <- theta[1]/theta[2])
```

```
## [1] 7.096935
```

Comparing model predictions with data



Exercise 11: Try different initial values for fit9. Did you get the same results?

Exercise 12: Do you think that parameter values are reasonable?

Summary

1. Implementing differential equation-based SIR, $SEIR$ model in R
2. Numerical integration of the model to obtain S, E, I, R (deSolve package and Euler)
3. Estimating parameters based on the 'process' and 'measurement' concepts.
4. Implementing log likelihood function and maximizing the log likelihood function using the stats::optim function
5. Applying scientific investigation process and estimating multiple parameters.

Remaining questions

- How could we improve the model for Shincheonji outbreak?
    - Can we assume that rate of isolation increased over time?
- Which one would you choose for the model with 1 parameter vs. the one with 3 parameters?
- How do you calculate confidence intervals for multiple parameters?
- Are there other algorithms for maximizing log likelihood than stats::optim?