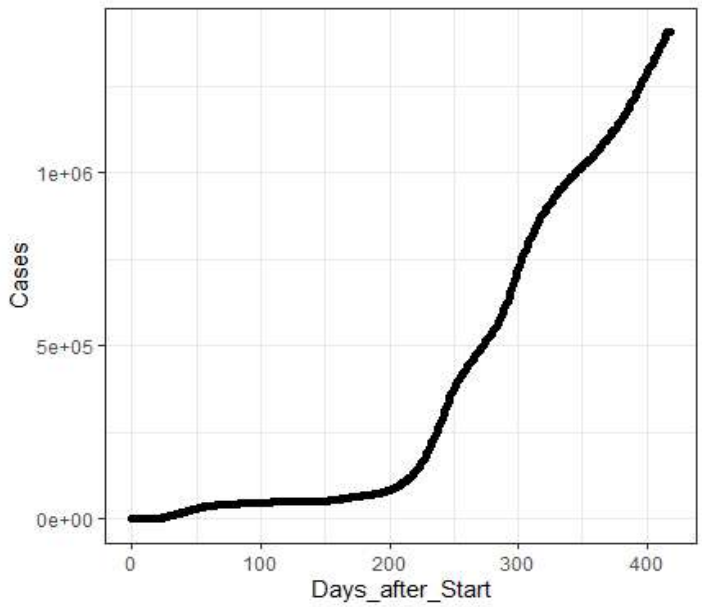
COVID-19 Cases Analysis with Growth Models

: Netherland 국가를 중심으로

1. 본인에게 배정된 두 국가 중 한 국가의 데이터를 이용해 다음 분석을 시행하라.

분석을 시행하기에 앞서, data를 시각화해보자.



<그림 1> cumulant cases after 2020-02-27

3월 20일까지의 자료를 바탕으로 4월 20일까지의 누적확진자수 예측값을 구해야 하기 때문에, 우선 3월 20일 전까지의 data set을 따로 분리하여 저장하였다. 그리고 이 data set에 대하여 각 model들을 적용한 다음, predict를 이용하여 4월 20일까지의 누적확진자수를 예측해보았다.

(1) 최근 3/20까지의 자료를 이용하여 예측한 linear regression, logistic, Gompertz의 모형 결과로부터 4/20일까지의 누적확진자수 예측값을 구하시오.

① linear regression $N_t = \beta_0 + \beta_1 t$

우선 <그림 2>의 결정계수 R^2값을 보면 0.8042로, 모형에 대한 귀무가설을 기각할 수 없다. 이제 각 linear model의 계수들에 대한 유의성을 확인해보면, p-value가 매우 작으므로 계수들에 대한 귀무가설도 기각할 수 없게 된다. 따라서 우리는 <표 1>과 같이 linear regression model의 coefficient들을 예측할 수 있게 된다. fitting된 결과는 <그림 3>을 통해 확인해볼 수 있다.

```
Call:
lm(formula = fo_lin, data = nld_cases_320)

Residuals:
    Min      1Q  Median      3Q     Max
-284997 -146948    3692  170956  270043

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)      -270042.18   17564.17  -15.38   <2e-16 ***
Days_after_Start    3128.33      78.56   39.82   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 173300 on 386 degrees of freedom
Multiple R-squared:  0.8042,    Adjusted R-squared:  0.8037
F-statistic:  1586 on 1 and 386 DF,  p-value: < 2.2e-16
```

<그림 2> linear regression 결과

이제, 이 모형을 바탕으로 4월 20일까지의 누적 확진자 수를 예측해보면, **1037599명**이라고 예측할 수 있게 된다. 이때 linear model로 예측하였음을 plot 상에서도 확인할 수 있도록, 데이터의 크기가 매우 크지만 log scale을 사용하지 않았다.

② logistic model $N_t = \frac{a}{1+\exp(b-ct)}$

우선, “Brute-Force” algorithm을 적용해 초기값을 찾아보면, $a = 1.19 \times 10^6$, $b = 1.00 \times 10^2$, $c = 3.33 \times 10^{-1}$을 얻는다. 이 값을 start로 지정하여 “Gauss-Newton” 방법을 적용시키면, 최종적으로 <그림 3>에서와 같은 추정값을 얻는다.

```
Formula: Cases ~ a/(1 + exp(b - c * Days_after_Start))

Parameters:
   Estimate Std. Error t value Pr(>|t|)
a 1.236e+06  8.878e+03  139.19   <2e-16 ***
b 7.563e+00  8.251e-02   91.66   <2e-16 ***
c 2.631e-02  3.383e-04   77.77   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 25260 on 385 degrees of freedom

Number of iterations to convergence: 8
Achieved convergence tolerance: 5.532e-06
```

<그림 3> logistic model의 계수 추정값

결정계수의 값을 반환해주는 사용자 정의 함수인 “getRsq”(자세한 사항은 부록의 r코드 참고)를 이용하여 R^2의 값을 구해보면, **0.996**란 값을 얻는다. R^2가 1에 매우 가까운 값을 가지므로, logistic model이 유의하지 않다고 볼 근거가 충분하지 않다. 또한, 각 계수들의 p-value가 매우 작은 값을 갖기 때문에 각 계수들이 유의하지 않다고 볼 근거가 충분하지 않다. 때문에 Logistic model을 이용해서는 아래와 같이 fitting을 할 수 있게 된다. (<표 2>와 <그림 6> 참고)

이제, 이 모형을 바탕으로 4월 20일까지의 누적 확진자 수를 예측해보면, **1305946**명이라고 예측할 수 있게 된다.

③ gompertz model $N_t = a\exp(-be^{-ct})$

※ Gompertz model에서의 fitting 과정은 Logistic model에서와 동일하므로, 세세한 설명을 생략하고 결과값 제시와 그에 대한 간략한 설명만 덧붙이는 식으로 서술하겠다.

"Brute-Force" algorithm으로 구한 초기값은, $a = 1.20 \times 10^6$, $b = 1.00 \times 10^1$, $c = 1.00 \times 10^{-2}$이다. 이 값을 start로 지정하여 "Gauss-Newton" algorithm으로 적용해보면, estimated a, b, c를 <그림 6>에서와 같이 얻게 된다.

```
Formula: Cases ~ a * exp(b * (-1) * exp((-1) * c * Days_after_Start))

Parameters:
   Estimate Std. Error t value Pr(>|t|)
a 1.469e+06  2.368e+04   62.06   <2e-16 ***
b 4.497e+01  3.226e+00   13.94   <2e-16 ***
c 1.379e-02  3.231e-04   42.68   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 30820 on 385 degrees of freedom

Number of iterations to convergence: 14
Achieved convergence tolerance: 4.213e-06
```

<그림 4> Gomperzt model의 계수 추정결과 by Gauss-Newton

$R^2 = 0.994$고 각 계수들의 p-value들이 매우 작은 값을 갖기 때문에, 모델과 각 계수들이 유의하지 않다고 볼 근거가 충분하지 않다. 때문에 fitting된 model은 data를 설명하는 도구로서 적절하다고 볼 수 있으며, 정리된 결과는 <표 3>에 제시돼 있다. 위와 같이 적합된 model로, 모형의 예측값과 실제 관측값을 하나의 plot으로 나타내보면 <그림 7>과 같다.

이제, 이 모형을 바탕으로 4월 20일까지의 누적 확진자 수를 예측해보면, **1275902**명이라고 예측할 수 있게 된다.

(2) 각 모델의 coefficients 값을 정리

① linear regression

Model	$\widehat{\beta_0}$	$\widehat{\beta_1}$	R-square
Linear Regression	-2.70×10^5	3.13×10^3	0.804

<표 1> linear regression model에서의 coefficients & R-square

② logistic model

Model	a	b	c	R-square
Logistic	1.24×10^6	7.56	2.63×10^{-2}	0.996

<표 2> logistic model에서의 coefficients & R-square

③ gompertz model

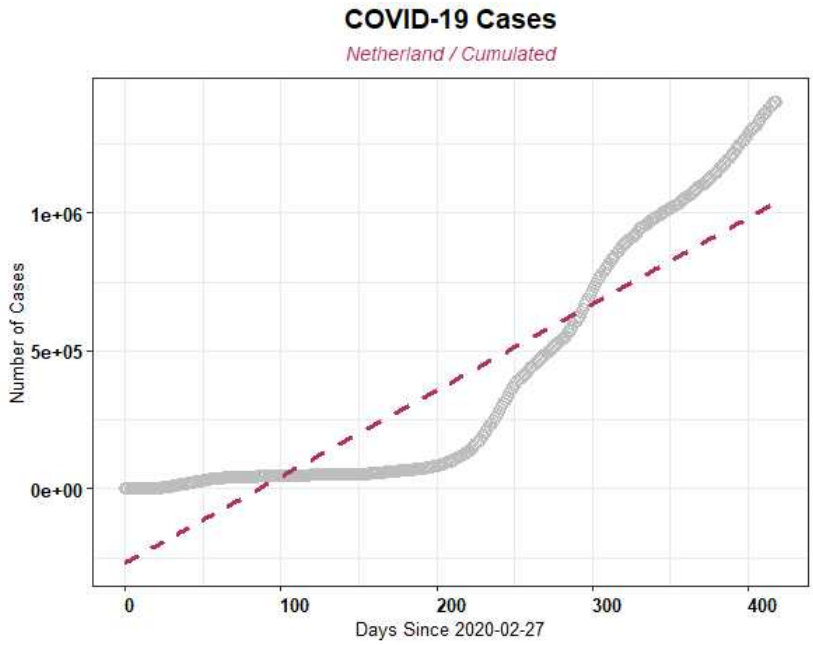
Model	a	b	c	R-square
Logistic	1.47×10^6	4.50	1.38×10^{-2}	0.994

<표 3> gompertz model에서의 coefficients & R-square

(3) 모형의 예측값과 실제 관측값을 하나의 plot에 나타낼 것.

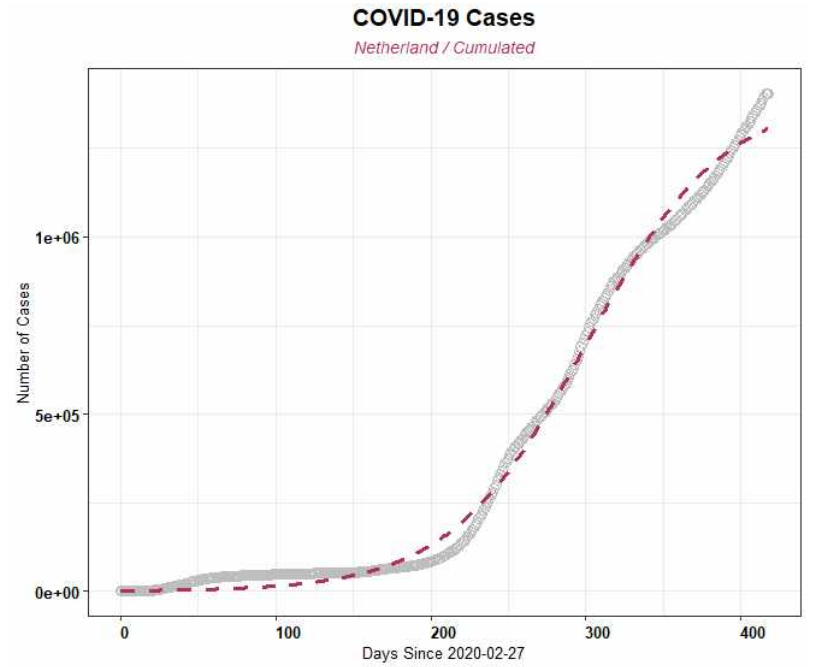
※ 이때, linear model을 이용해서 fitting을 시키는 것을 plot 상에서도 한눈에 관측할 수 있도록, case의 수가 몹시 큼에도 log scale을 이용하지 않았다. 또한, fitting된 model들끼리 시각적으로 비교할 수 있도록, logistic과 gompertz 모델에 대해서도 original scale을 이용하였다.

① linear regression



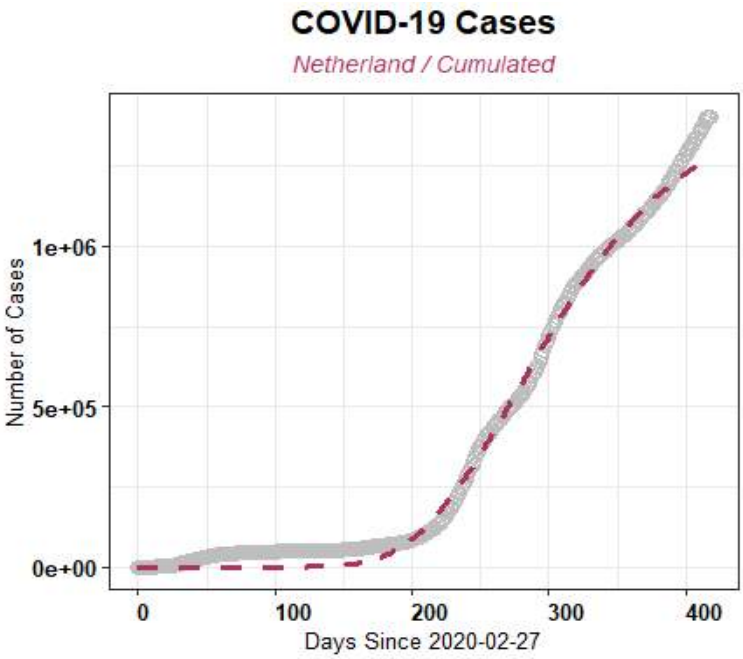
<그림 5> linear model로 예측한 값과 실제 관측값

② logistic model



<그림 6> logistic model로 예측한 값과 실제 관측값

③ gompertz model



<그림 7> Gompertz model로 fitting한 결과

(4) 3/21 - 4/20 까지의 일별 관찰값과 예측값을 사용하여 test MSE를 구할 것

	Linear	Logistic	Gompertz
MS_E	108474871086	3265231856	7455768586

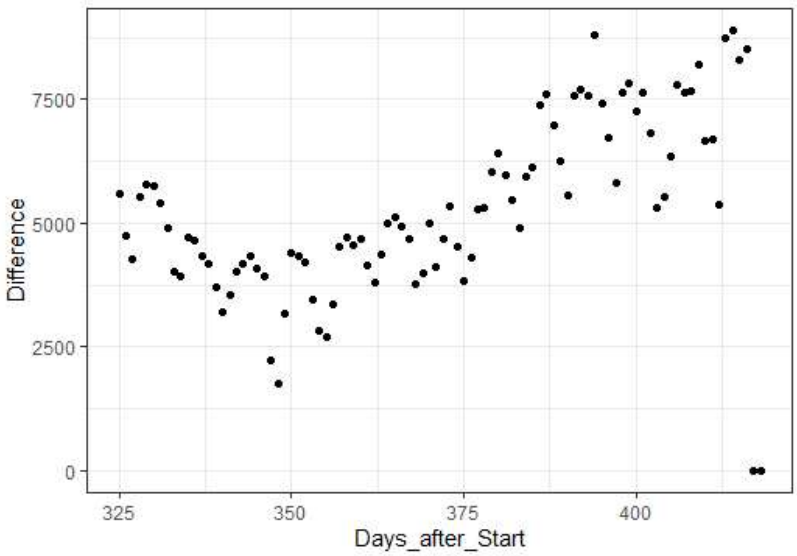
<표 4> 세 model에서의 MSE 비교 (3/21~4/20 data 이용)

(5) MSE를 기반으로 세 모형 중에서 어떤 모형이 가장 제일 자료를 잘 적합하는지 판단하시오.

위 세 모델의 mse값을 비교해보면 logistic model에서가 가장 작은 값을 가지므로, logistic model이 가장 자료를 잘 적합한다고 할 수 있다.

2. covid_vaccine 데이터의 최근 3/20까지의 자료를 이용하여, predictor를 days만 이용하여 일별확진자수에 대한 segmented poisson regression model 모형을 적합하고 적합 결과를 그래프로 나타낼 것.

우선 data를 먼저 시각화해보면, <그림 8>과 같이 나타나진다.



<그림 8> covid_vaccine의 data visualization

현재 우리는 3/20일 이전까지의 data를 갖고 분석을 시행한 다음, 그 결과를 통해 3/21 ~ 4/20일의 data를 예측고자 하므로, 위 그림에서 Days_after_Start가 378일 이전의 것만 확인해보면 된다. 그러면, break point를 2개로 설정할 예정이므로, 대략적으로 break_point가 (Days_after_Start 값으로) 350 부근, 그리고 375 부근에서 발생할 것이라고 예측할 수 있게 된다.

따라서 이 값을 초기값으로 부여한 다음, segmented 함수를 통해 더 적합한 break point를 찾고자 하였다.

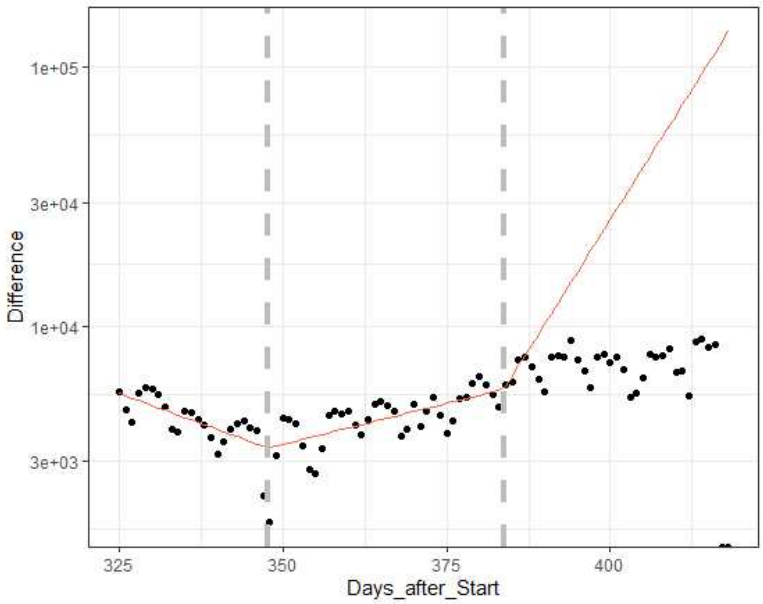
(1) 모형의 예측값과 실제 관측값의 하나의 plot에 나타낼 것

fitting 결과 break point가 <그림 9>와 같이 나오게 된다. 이 모델을 활용한 예측값과 실제 관측값을 하나의 plot으로 나타내보면 <그림 10>과 같다.

```
Estimated Break-Point(s):
                          Est. St.Err
psi1.Days_after_Start 347.693  0.240
psi2.Days_after_Start 383.762  0.163
```

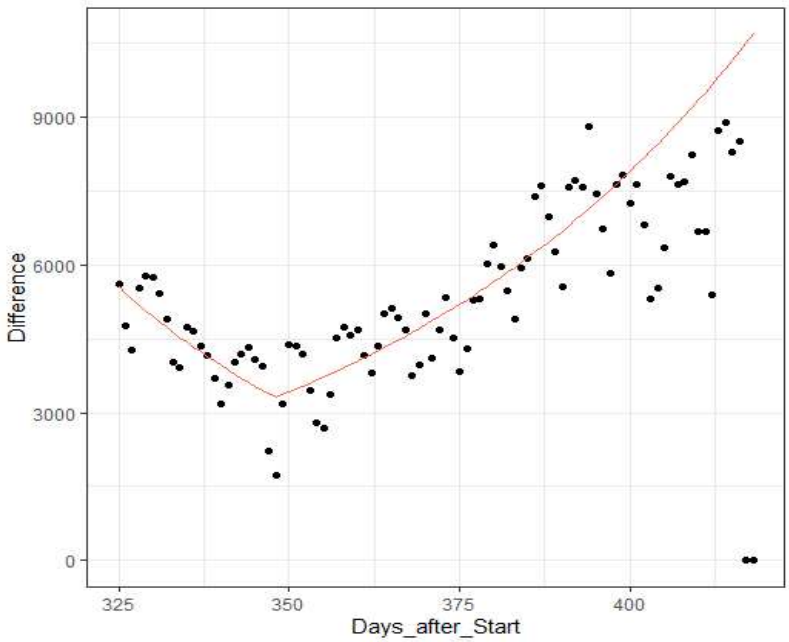
<그림 9> estimated break-points

이 모형의 경우 두 번째 breakpoint를 이후로 3/20일까지 급격하게 증가하는 추세를 보이고 있기 때문에, 이 형태로 4/20일까지의 data를 예측해보게 되면 무한대의 값이 나오게 된다. (<그림 10>의 경우 3/21부터의 예측값이 급증하는 형태를 보여, segmented의 형태를 잘 나타내기 위해 y를 log scale로 변환하여 사용하였다.)



<그림 10> break point 2개인 경우 (log scale 이용)

따라서 이 경우에는 break point를 하나만 설정(<그림11> 참고)하는 것이 일일확진자수를 더욱 잘 예측해주고 있는 것으로 보여진다.



<그림 11> break point 하나로 fitting한 결과

(2) 3/21-4/20까지의 일별 관측값과 예측값을 사용하여 test MSE를 구할 것

3785925756가 나오게 된다. 기울기가 변하는 지점을 찾아서 segmented로 fitting을 했음에도 이처럼 큰 값이 나오는 것은, <그림 8>을 살펴보면 알 수 있듯이 두 번째 breakpoint 이후로 3/20일까지의 증가 추세와 3/21에서 4/20 까지의 추세가 다르기 때문이다.

3. covid_vaccine 데이터의 최근 3/20까지의 자료를 이용하여, predictor를 days와 백신접종변수 people_vaccinated를 이용하여 일별확진자수에 대한 segmented poisson regression model 모형을 적합하고 적합 결과를 그래프로 나타낼 것.

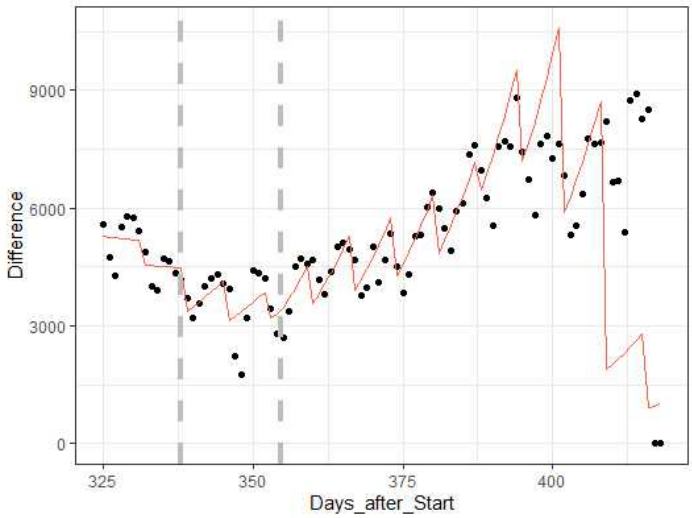
(1) 모형의 예측값과 실제 관측값의 하나의 plot에 나타낼 것

3번의 경우에는, psi의 초기값을 340과 380으로 부여하고 fit를 시도하였다. 그 결과 <그림 12>와 같은 break point를 얻게 되었다.

```
Estimated Break-Point(s):
                          Est. St.Err
psi1.Days_after_Start 337.997  0.312
psi2.Days_after_Start 354.624  0.330
```

<그림 12> estimated break points

이 모델을 활용한 에측값과 실제 관측값을 하나의 plot으로 나타내보면 <그림 13>과 같다.



<그림 13> days와 people_vaccinated를 이용해서 fitting한 결과

(2) 3/21-4/20까지의 일별 관측값과 예측값을 사용하여 test MSE를 구할 것

11466940가 나오게 된다.

(3) 백신의 효과의 여부를 데이터에 기반하여 서술하시오.

<그림 13>을 보면 두 번째 breakpoint 이후로 백신 접종 후에도 확진자가 증가하는 추세를 보이고 있음을 확인해볼 수 있다. 이 부분만 본다면 백신의 효과가 없다고 해석할 수도 있겠으나, Days_after_Start가 400 이후 부분까지 주목한다면 해석은 반대가 된다. 왜냐하면 이 부분에 대해서는 일일 확진자 수가 급격하게 감소하고 있기 때문이다.

백신 접종 후 항체가 생성되는 데에도 시간이 걸리며, 네덜란드의 국민이 동시대에 모두 접종을 받은 것이 아니라 몇 차로 나눠서 접종받았기 때문에 국민들 사이에서도 항체가 형성되는 일자가 다를 것이다. 정리하자면, 백신 접종을 시작한다 할지라도 집단 면역이 당장 이루어지는 것이 아니라, 집단 면역에 성공하기까지는 어느 정도 시간이 필요하단 사실을 알 수 있다. 이러한 관점으로 생각해본다면, 지금까지의 data를 토대로 분석한다면, 백신의 효과가 유의하다는 결론을 내릴 수 있을 것이다.

4. r 코드

```
############################## hw6 ################################

setwd("C:/2021s")
getwd()

library(ggplot2)
library(dplyr)
library(nls2)
library(segmented)
###################### 0. Set Functions to get MSE ######################


getRsq <- function(y, yhat){
  Rsq <- 1 - (sum((y-yhat)^2) / sum((y-mean(y))^2))
  return(Rsq)
}

getMSE_lin<-function(y,yhat){
  MSE <- sum((y-yhat)^2)/(length(y)-2)
  # linear model의 df = n-2
  return(MSE)
}

getMSE <- function(y, yhat){
  MSE <- sum((y-yhat)^2)/(length(y)-3)
  # 우리가 사용할 3개의 model은 전부 3개의 parameter => df = n-3
  return(MSE)
}

# length(y)가 parameter 개수에 비해 꽤 크기 때문에 length(y)로 나눠줘도 큰 차이는 없을 테지만 .... 그래도 df=n-k니까
# 위와 같이 getMSE 함수 작성


################### 1. Data Importing & Preprocessing ##################


# Read
cases<-read.csv("global_confirmed_cases_210420.csv")
nld_cases <- cases %>% filter(CountryCode == "NLD")
```

```
# Date formatting
nld_cases$Date <- sapply(nld_cases$Date, function(d) {
  vec <- strsplit(d, split = ".", fixed = T) %>%
    unlist %>% as.character()
  vec_ch <- sapply(vec, function(i) {
    ifelse(nchar(i) < 2, paste0("0",i), i)
  })
  paste(vec_ch, collapse = "-")
})


# 첫 case가 발생한 날짜의 데이터부터 이용
start_date <- nld_cases$Date[min(which(nld_cases$Cases > 0))]

# Days_after_Start column 추가
nld_cases_fin <- mutate(nld_cases,Days_after_Start = as.integer(as.Date(Date) -
as.Date(start_date))) %>%
  filter(Days_after_Start >= 0)


# 3월 20일 자료까지
nld_cases_320 <- nld_cases_fin %>% filter(Days_after_Start<=387)



############################## 2. Model Fitting ##############################


# data visualization
nld_cases_fin %>% ggplot(aes(x=Days_after_Start, y=Cases)) +
  theme_bw() + geom_point()


# fitting에 실패하면 fit_iterate_~~~에는 NULL 만 존재하게 됨
fit_iterate_logi<-NULL
fit_iterate_bert<-NULL
fit_iterate_gomp<-NULL


####### Usig Linear Regression #######
```

```
fo_lin <- Cases ~ Days_after_Start
lin.test<-lm(fo_lin,data=nld_cases_320)
summary(lin.test)

# Days_after_Start가 418일 때가 2021.04.20임
predict(lin.test,data.frame(Days_after_Start=418))
nld_predict_lin<-data.frame(x=nld_cases_fin$Days_after_Start,
                            predict=predict(lin.test,nld_cases_fin))

day_start <- as.character(sort(nld_cases_fin$Date)[1])
### linear model
cumulated_data_lin <- ggplot(data=nld_cases_fin,aes(x=Days_after_Start,y=Cases))+
  geom_point(color='gray', shape =1, size=3)+ theme_bw() +
  labs(title = paste0("COVID-19 Cases"),
       subtitle = paste0("Netherland", " / ", "Cumulated"),
       x=paste0('Days Since ', as.character(day_start)),
       y='Number of Cases')+
  geom_line(data=nld_predict_logi,
            aes(x=x,y=predict), color="maroon", linetype="dashed",size=1.2) +
  theme(plot.title=element_text(size=14, hjust=0.5, face="bold", colour="black", vjust=2),
        plot.subtitle=element_text(size=10.5, hjust=0.5, face="italic", color="maroon",
vjust=2),
        axis.text=element_text(size=10, face = "bold", colour = "black"),
        axis.text.x = element_text(size = 10, hjust = 0),
        axis.title=element_text(size=10, colour = "black"))

cumulated_data_lin


### evaluate mse
nld_predict_lin_321420 <- nld_predict_lin %>% filter(x>387)
nld_cases_321420 <- nld_cases_fin %>% filter(Days_after_Start>387)

(MSE_lin <- getMSE_lin(nld_cases_321420$Cases,nld_predict_lin_321420$predict))




########  Using Logistic Model #######
fo_logi<- Cases ~ a / (1 + exp(b-c*Days_after_Start))
```

```
### Set grid
grid_logi<- data.frame(a=c(0,max(nld_cases_320$Cases)),b=c(0,100),c=c(0,1))

### find initial value with brute-force algorithm
rough_fit_logi<- nls2(fo_logi, data=nld_cases_320, start=grid_logi, algorithm="brute-force")
coef(rough_fit_logi)


### apply gauss-newton algorithm to find better coefficients
fit_iterate_logi<- nls2(fo_logi,data=nld_cases_320,start=coef(rough_fit_logi))


### check Rsq and estimate coefficients
(Rsq_logi<-getRsq(nld_cases_320$Cases,predict(fit_iterate_logi,nld_cases_320)))
summary(fit_iterate_logi)
coef(fit_iterate_logi)


# Days_after_Start가 418일 때가 2021.04.20임
predict(fit_iterate_logi,data.frame(Days_after_Start=418))
nld_predict_logi<-data.frame(x=nld_cases_fin$Days_after_Start,
                             predict=predict(fit_iterate_logi,nld_cases_fin))


day_start <- as.character(sort(nld_cases_fin$Date)[1])
### logistic model
cumulated_data_logi <- ggplot(data=nld_cases_fin,aes(x=Days_after_Start,y=Cases))+
  geom_point(color='gray', shape =1, size=3)+ theme_bw() +
  labs(title = paste0("COVID-19 Cases"),
       subtitle = paste0("Netherland", " / ", "Cumulated"),
       x=paste0('Days Since ', as.character(day_start)),
       y='Number of Cases')+
  geom_line(data=nld_predict_logi,
            aes(x=x,y=predict), color="maroon", linetype="dashed",size=1.2) +
  theme(plot.title=element_text(size=14, hjust=0.5, face="bold", colour="black", vjust=2),
        plot.subtitle=element_text(size=10.5, hjust=0.5, face="italic", color="maroon",
vjust=2),
        axis.text=element_text(size=10, face = "bold", colour = "black"),
        axis.text.x = element_text(size = 10, hjust = 0),
        axis.title=element_text(size=10, colour = "black"))
```

```
cumulated_data_logi


nld_predict_logi_321420 <- nld_predict_logi %>% filter(x>387)
nld_cases_321420 <- nld_cases_fin %>% filter(Days_after_Start>387)

(MSE_logi <- getMSE(nld_cases_321420$Cases,nld_predict_logi_321420$predict))




########## Using Gompertz Model #########
fo_gomp <- Cases ~ a * exp(b * (-1) * exp((-1) * c * Days_after_Start))


### Set grid
grid_gomp <- expand.grid(a = max(nld_cases_320$Cases), b = seq(1, 10, 1), c = seq(0, .1,
0.001))


### find initial values with brute_force algorithm
rough_fit_gomp = nls2(fo_gomp, data = nld_cases_320, start = grid_gomp,
                      algorithm = "brute-force")
rough_fit_gomp


### apply gauss-newton algorithm to find better coefficients
fit_iterate_gomp<-nls2(fo_gomp,data=nld_cases_320,start=coef(rough_fit_gomp))
summary(fit_iterate_gomp)


### result of estimate the coefficients
(Rsq_gomp<-getRsq(nld_cases_320$Cases,predict(fit_iterate_gomp,nld_cases_320)))
coef(fit_iterate_gomp)


# Days_after_Start가 418일 때가 2021.04.20임
nld_predict_gomp<-data.frame(x=nld_cases_fin$Days_after_Start,
                             predict=predict(fit_iterate_gomp,nld_cases_fin))
nld_predict_gomp %>% filter(x==418)

day_start <- as.character(sort(nld_cases_fin$Date)[1])
```

```
### gompertz model
cumulated_data_gomp <- ggplot(data=nld_cases_fin,aes(x=Days_after_Start,y=Cases))+
  geom_point(color='gray', shape =1, size=3)+ theme_bw() +
  labs(title = paste0("COVID-19 Cases"),
       subtitle = paste0("Netherland", " / ", "Cumulated"),
       x=paste0('Days Since ', as.character(day_start)),
       y='Number of Cases')+
  geom_line(data=nld_predict_gomp,
            aes(x=x,y=predict), color="maroon", linetype="dashed",size=1.2) +
  theme(plot.title=element_text(size=14, hjust=0.5, face="bold", colour="black", vjust=2),
        plot.subtitle=element_text(size=10.5, hjust=0.5, face="italic", color="maroon",
vjust=2),
        axis.text=element_text(size=10, face = "bold", colour = "black"),
        axis.text.x = element_text(size = 10, hjust = 0),
        axis.title=element_text(size=10, colour = "black"))

cumulated_data_gomp


nld_predict_gomp_321420 <- nld_predict_gomp %>% filter(x>387)
nld_cases_321420 <- nld_cases_fin %>% filter(Days_after_Start>387)

(MSE_gomp <- getMSE(nld_cases_321420$Cases,nld_predict_gomp_321420$predict))




######################## 2. segmented poisson regression model ################
vac<-read.csv("covid_vaccine.csv")
vac_nld<-vac %>% filter(CountryCode=="NLD")


# Date formatting
vac_nld$Date <- sapply(vac_nld$Date, function(d) {
  vec <- strsplit(d, split = ".", fixed = T) %>%
    unlist %>% as.character()
  vec_ch <- sapply(vec, function(i) {
    ifelse(nchar(i) < 2, paste0("0",i), i)
  })
  paste(vec_ch, collapse = "-")
})
```

```
# 첫 case 발생
start_date <- "2020-02-27"

# Days_after_Start column 추가
vac_nld_fin <- mutate(vac_nld,Days_after_Start = as.integer(as.Date(Date) -
as.Date(start_date))) %>%
  filter(Days_after_Start >= 0)


# 3월 20일 자료까지
vac_nld_320 <- vac_nld_fin %>% filter(Days_after_Start<=387)


# data 시각화
ggplot(vac_nld_fin, aes(x = Days_after_Start, y = Difference)) +
  geom_point() + theme_bw()

 # ----------------------------

# 일별 확진자수를 보고 있으므로 days_after_Start 이용
pois_fit <- glm(Difference~Days_after_Start, data = vac_nld_320, family=poisson)
summary(pois_fit)

set.seed(777)
seg_fit <- NULL
try(seg_fit <- segmented(pois_fit, seg.Z = ~ Days_after_Start, psi=c(350,370),
                         npsi = 2, control = seg.control(it.max = 100000, n.boot = 50)))
summary(seg_fit)
(psi_1<-seg_fit$psi[3])
(psi_2<-seg_fit$psi[4])

# 적합된 모델로 fitting

nld_predict_seg<-data.frame(Days_after_Start=vac_nld_fin$Days_after_Start,
                            predict=exp(predict(seg_fit,vac_nld_fin)))

nld_predict_seg2<-data.frame(Days_after_Start=vac_nld_320$Days_after_Start,
                            predict=seg_fit$fitted.values)
```

```
### segmented model with days parameter

p_1 <- ggplot(vac_nld_fin, aes(x = Days_after_Start, y = Difference)) +
  geom_point() +
  geom_line(mapping = aes(x = Days_after_Start, y = predict),
            data = nld_predict_seg, color = "tomato") +
  geom_vline(xintercept = psi_1, color = "grey", size = 1.5, linetype="dashed") +
  geom_vline(xintercept = psi_2, color = "grey", size = 1.5, linetype="dashed") +
  theme_bw() +
  scale_y_log10()

p_1


# evaluate mse

nld_predict_seg_321420<-nld_predict_seg %>% filter(Days_after_Start>387)
vac_nld_321420 <- vac_nld_fin %>% filter(Days_after_Start>387)

# 여기서 mse가 n-6인데, getMSE 함수 자체는 n-3을 하는 것으로 돼 있으므로,
# 아래와 같이 값 조정을 해주었다.
n<-nrow(nld_predict_seg_321420)
(MSE_seg1<-getMSE(vac_nld_321420$Difference,nld_predict_seg_321420$predict)*((n-3)/(n-6
)))



################################### 3. ################################

# Cases
pois_fit_1 <- glm(Difference~ Days_after_Start+people_vaccinated,
              data = vac_nld_320, family = poisson)

summary(pois_fit_1)
# Segmented Poisson

set.seed(777)
seg_fit_1 <- NULL
try(seg_fit_1 <- segmented(pois_fit_1, seg.Z = ~ Days_after_Start,
                           psi=c(340, 380), npsi = 2,
```

```
                              control = seg.control(it.max = 10000, n.boot = 50)))
summary(seg_fit_1)

(psi_1<-seg_fit_1$psi[3])
(psi_2<-seg_fit_1$psi[4])

# 적합된 모델로 fitting
nld_predict_seg1<-data.frame(Days_after_Start=vac_nld_fin$Days_after_Start,
                             predict=exp(predict(seg_fit_1,vac_nld_fin)))


### segmented model with days parameter

p_1 <- ggplot(vac_nld_fin, aes(x = Days_after_Start, y = Difference)) +
  geom_point() +
  geom_line(mapping = aes(x = Days_after_Start, y = predict),
            data = nld_predict_seg1, color = "tomato") +
  geom_vline(xintercept = psi_1, color = "grey", size = 1.5, linetype="dashed") +
  geom_vline(xintercept = psi_2, color = "grey", size = 1.5, linetype="dashed") +
  theme_bw()

p_1


# evaluate mse

nld_predict_seg1_321420<-nld_predict_seg1 %>% filter(Days_after_Start>387)
vac_nld_321420 <- vac_nld_fin %>% filter(Days_after_Start>387)

# 여기서 mse가 n-5인데, getMSE 함수 자체는 n-3을 하는 것으로 돼 있으므로,
# 아래와 같이 값 조정을 해주었다.
n<-nrow(nld_predict_seg1_321420)
(MSE_seg1<-getMSE(vac_nld_321420$Difference,nld_predict_seg1_321420$predict)*((n-3)/(n-
5)))
```