BIOSTAT 285 Spring 2020 Covid-19 State Data Project

Nan Liu

Analysis on New York Times Covid-19 State Data

Import and Subset data First let's import the `covid-19-state-level-data.csv` dataset and create subsets of the data for States New York, California, and Washington.

```
#import csv file
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.3.1     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## Warning: package 'ggplot2' was built under R version 3.6.2
```

```
## Warning: package 'tibble' was built under R version 3.6.2
```

```
## Warning: package 'tidyr' was built under R version 3.6.2
```

```
## Warning: package 'purrr' was built under R version 3.6.2
```

```
## Warning: package 'dplyr' was built under R version 3.6.2
```

```
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
covid <- read.csv("covid-19-state-level-data.csv")
#subset new york
ny <- filter(covid, state == "New York")
dim(ny)
```

```
## [1] 58  5
```

```
head(ny,6)
```

```
##         date    state fips cases deaths
## 1 2020-03-01 New York   36     1      0
## 2 2020-03-02 New York   36     1      0
## 3 2020-03-03 New York   36     2      0
## 4 2020-03-04 New York   36    11      0
## 5 2020-03-05 New York   36    22      0
## 6 2020-03-06 New York   36    44      0
```

```
#california
ca <- filter(covid, state == "California")
dim(ca)
```

```
## [1] 94  5
```

```
head(ca,6)
```

```
##         date      state fips cases deaths
## 1 2020-01-25 California    6     1      0
## 2 2020-01-26 California    6     2      0
## 3 2020-01-27 California    6     2      0
## 4 2020-01-28 California    6     2      0
## 5 2020-01-29 California    6     2      0
## 6 2020-01-30 California    6     2      0
```

```
#washington
wa <- filter(covid, state == "Washington")
dim(wa)
```

```
## [1] 98  5
```

```
head(wa,6)
```

```
##         date      state fips cases deaths
## 1 2020-01-21 Washington   53     1      0
## 2 2020-01-22 Washington   53     1      0
## 3 2020-01-23 Washington   53     1      0
## 4 2020-01-24 Washington   53     1      0
## 5 2020-01-25 Washington   53     1      0
## 6 2020-01-26 Washington   53     1      0
```

Cubic Splines and Natural Cubic Splines

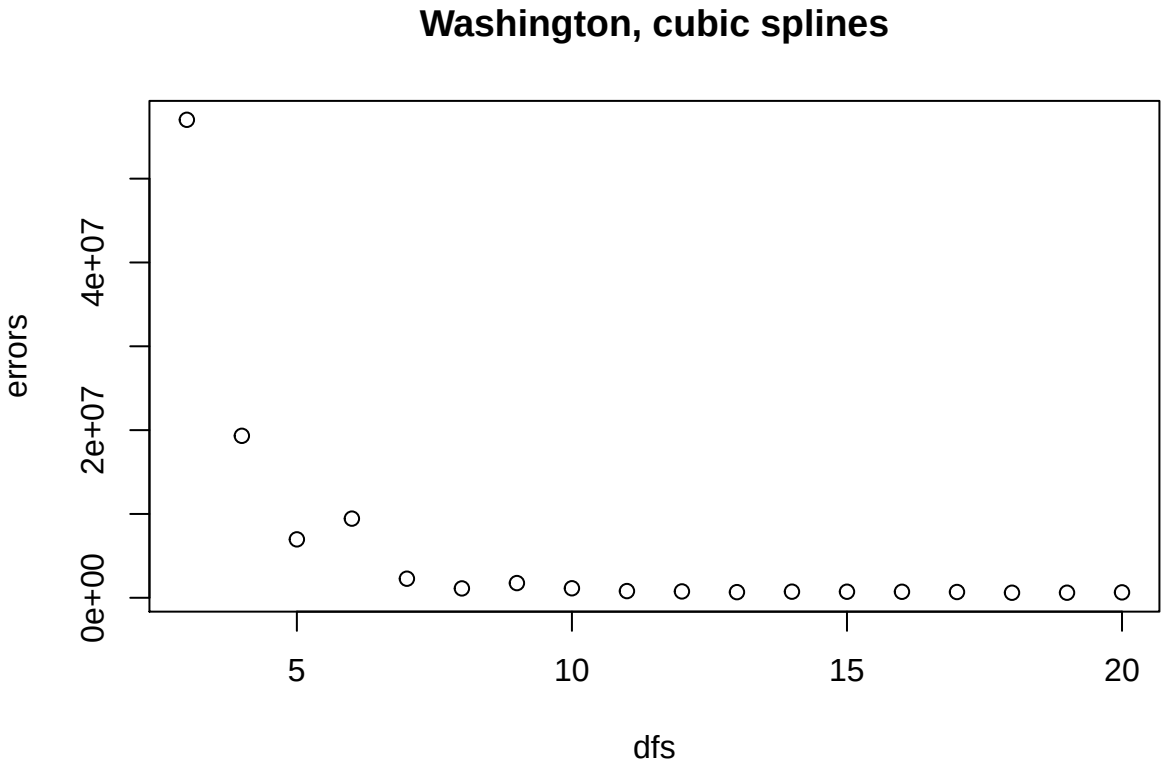
Fit both cubic splines and natural cubic splines on each state subset, with the number of cases as response, and the number of days since first case as predictor. Find the optimal degrees of freedom (`df`) for the basis.

```
#generate the number of days since first case
ny <- ny %>%
  mutate(days = as.numeric(as.Date(as.character(date))-
                  as.Date(as.character("2020-03-01"))))
ca <- ca %>%
  mutate(days = as.numeric(as.Date(as.character(date))-
```

```
                  as.Date(as.character("2020-01-25"))))
wa <- wa %>%
  mutate(days = as.numeric(as.Date(as.character(date))-
                  as.Date(as.character("2020-01-21"))))
```

Washington with cubic splines:

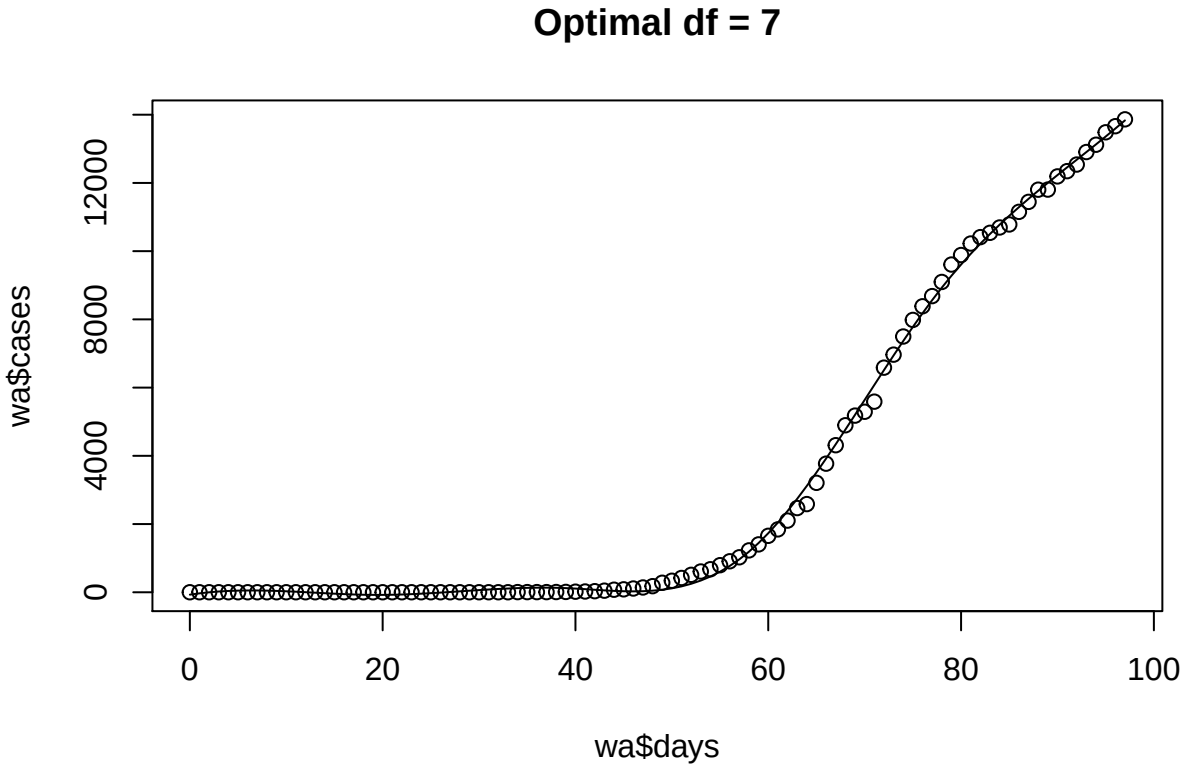
```
library(splines)
dfs <- (3 : 20)
errors <- c()
for (i in dfs){
fit <- lm(cases ~ bs(days, df = i), data = wa)
errors <- c(errors, sum(fit$residuals^2))
}
plot(dfs, errors, main = "Washington, cubic splines")
```



From the plot we see, when the df is lower than 7, the errors are quite large. If df is greater than 7, the error will not change much. So we conclude that for Washington dataset, the optimal degree of freedom in cubic spline method is 7.

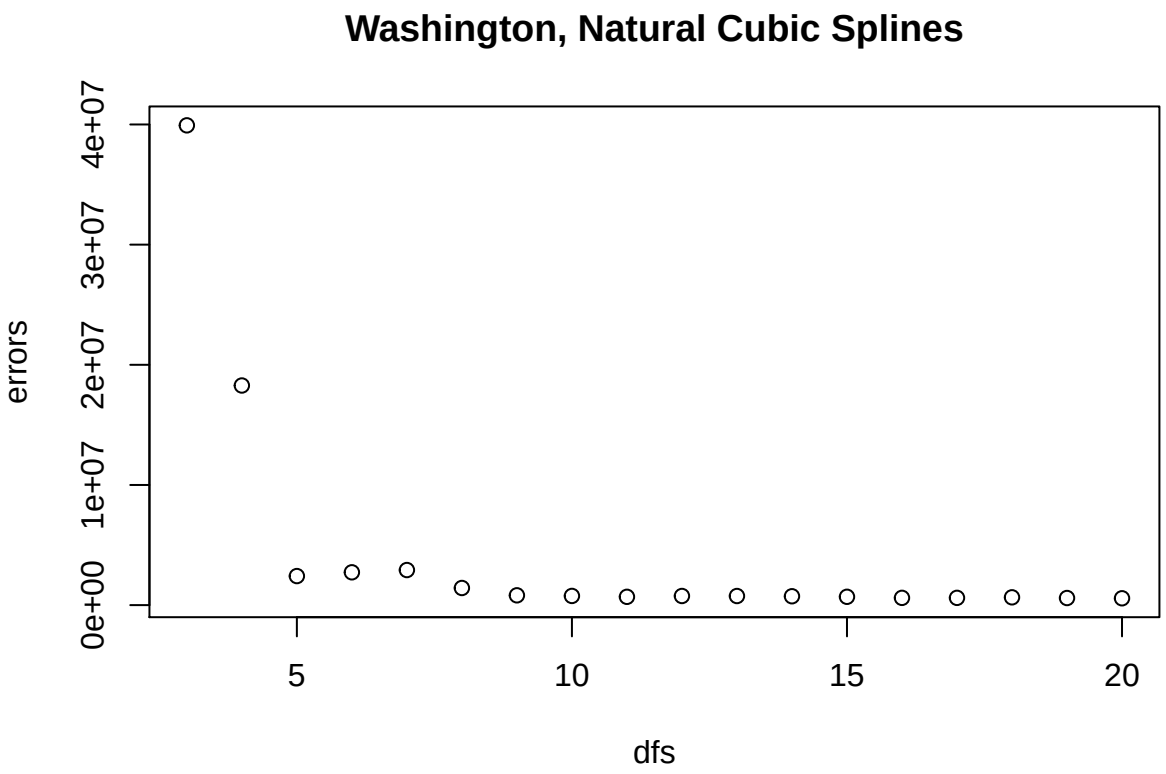
Then fit the cubic splines with df = 7 on Washington data.

```
bs.fit.wa <- lm(cases ~ bs(days, df = 7), data = wa)
plot(wa$days, wa$cases, main = ("Optimal df = 7"))
lines(wa$days, (bs.fit.wa$fitted.values))
```



Washington with natural cubic splines:

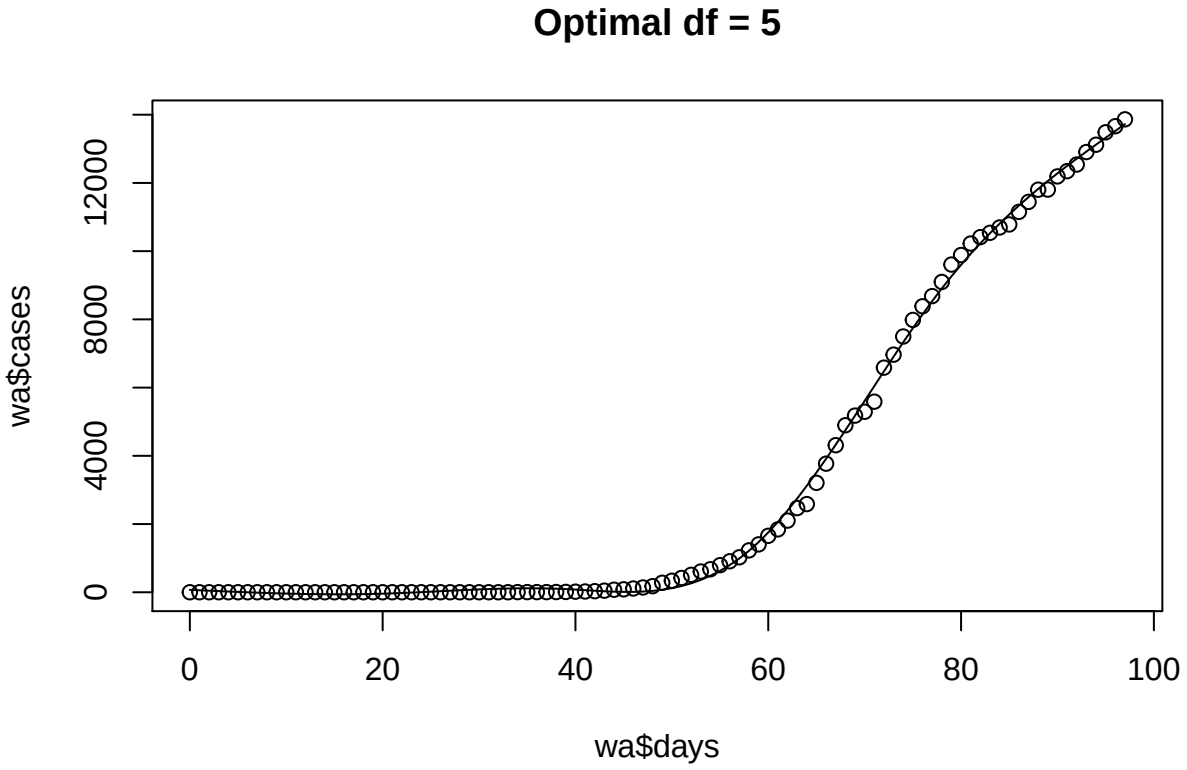
```
errors <- c()
for (i in dfs){
fit <- lm(cases ~ ns(days, df = i), data = wa)
errors <- c(errors, sum(fit$residuals^2))
}
plot(dfs, errors, main = "Washington, Natural Cubic Splines")
```



We conclude that the optimal df is 5.

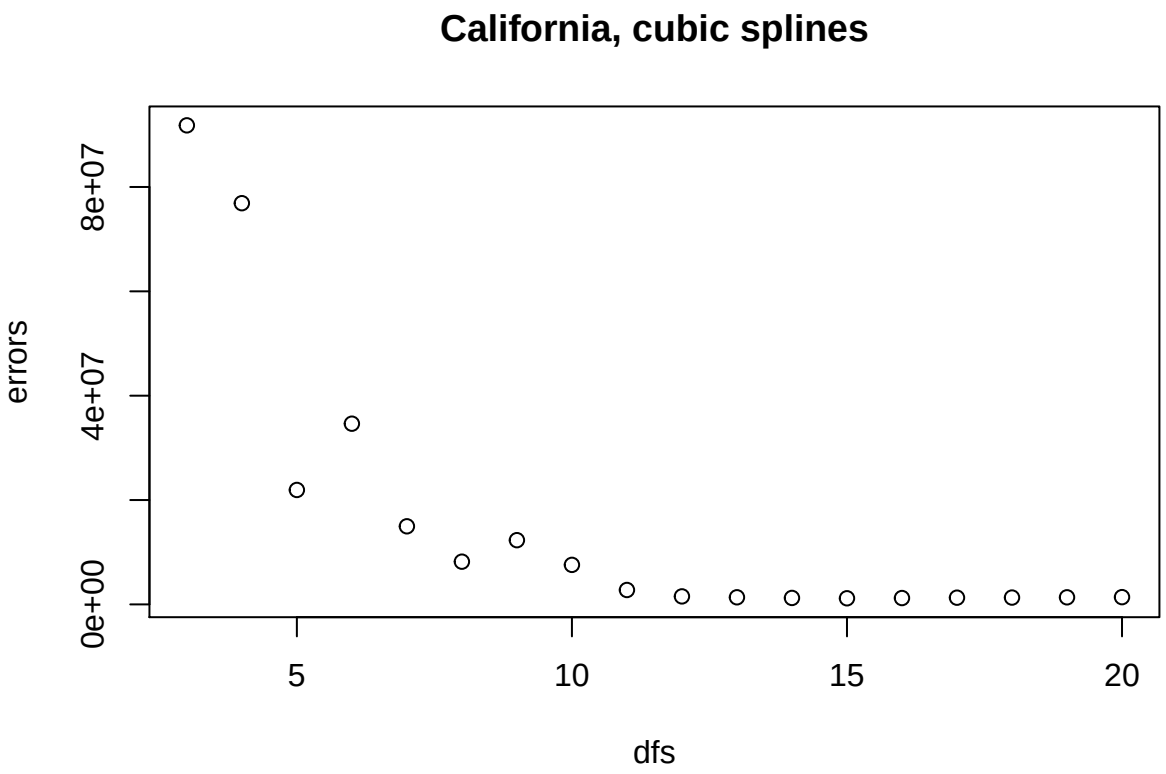
Then fit the natural cubic splines with df = 5 on Washington data.

```
ns.fit.wa <- lm(cases ~ ns(days, df = 5), data = wa)
plot(wa$days, wa$cases, main = ("Optimal df = 5"))
lines(wa$days, (ns.fit.wa$fitted.values))
```



California with cubic splines:

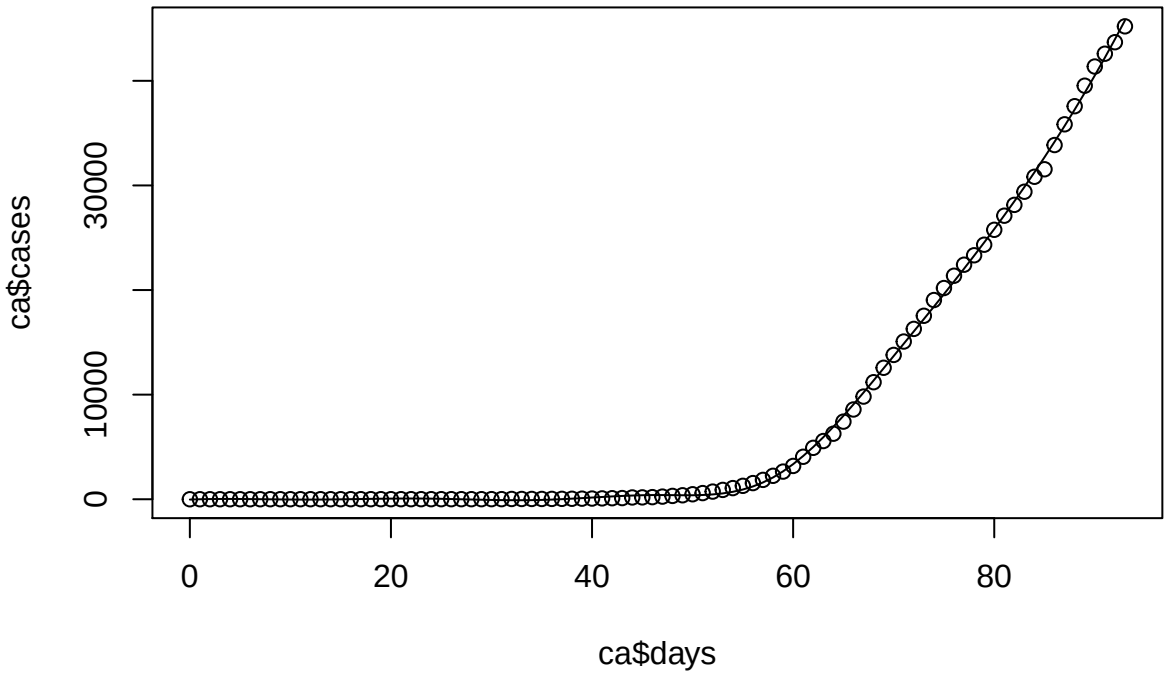
```
library(splines)
#bs
dfs <- (3 : 20)
errors <- c()
for (i in dfs){
fit <- lm(cases ~ bs(days, df = i), data = ca)
errors <- c(errors, sum(fit$residuals^2))
}
plot(dfs, errors, main = "California, cubic splines")
```



From the plot we see, when the df is lower than 10, the errors are quite large. If df is greater than 10, the error will not change much. So we conclude that for California dataset, the optimal degree of freedom in cubic spline method is 11.

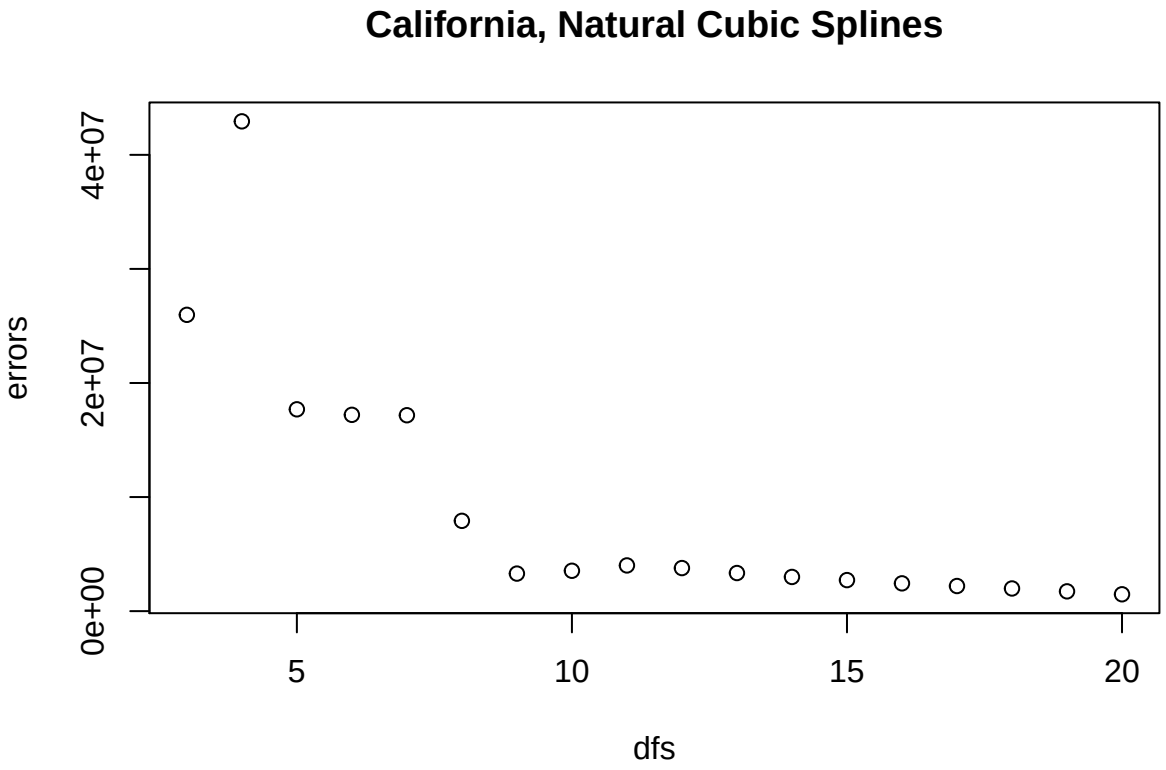
Then fit the cubic splines with df = 10 on California data.

```
bs.fit.ca <- lm(cases ~ bs(days, df = 10), data = ca)
plot(ca$days, ca$cases, main = ("Optimal df = 10"))
lines(ca$days, (bs.fit.ca$fitted.values))
```



California with natural cubic splines:

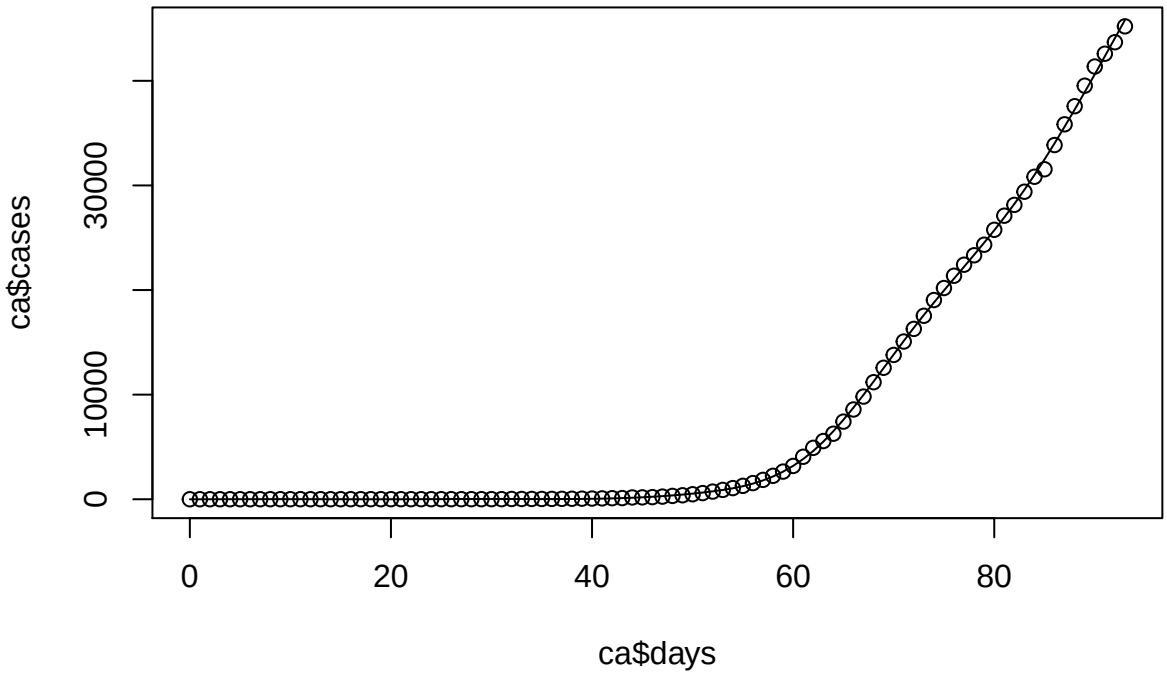
```
errors <- c()
for (i in dfs){
fit <- lm(cases ~ ns(days, df = i), data = ca)
errors <- c(errors, sum(fit$residuals^2))
}
plot(dfs, errors, main = "California, Natural Cubic Splines")
```



We conclude that the optimal df is 9.

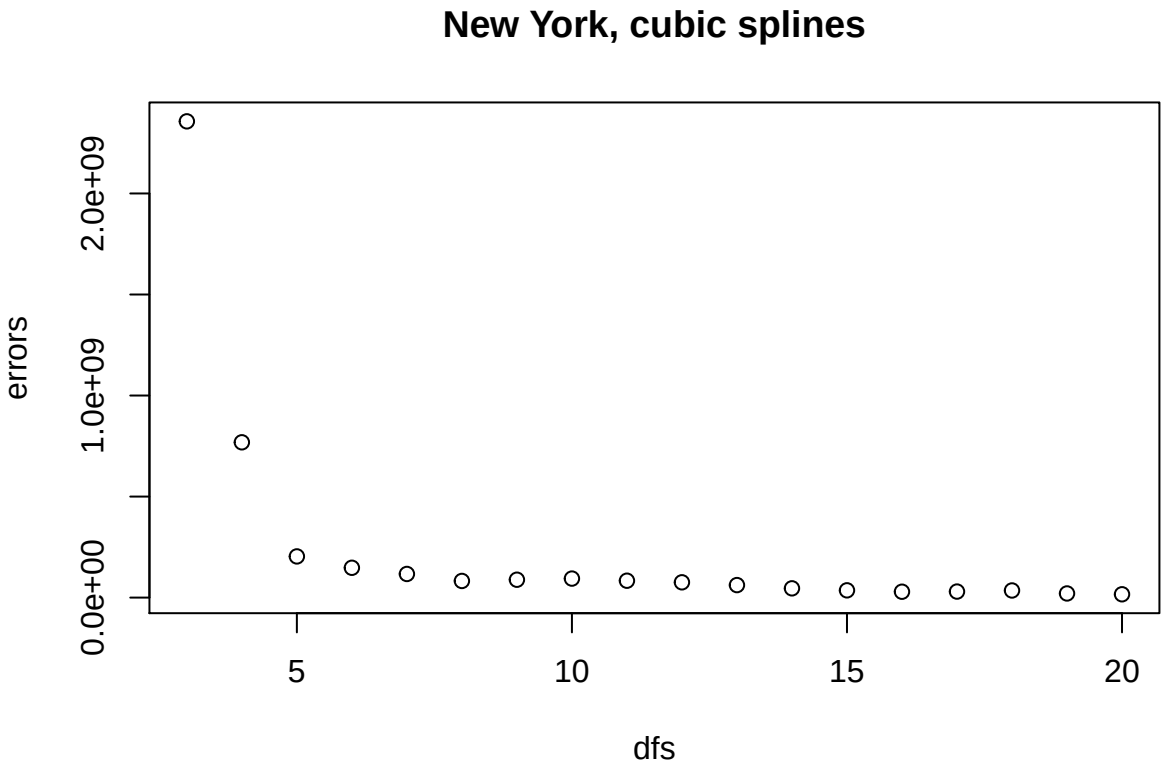
Then fit the natural cubic splines with df = 9 on California data.

```
ns.fit.ca <- lm(cases ~ ns(days, df = 9), data = ca)
plot(ca$days, ca$cases, main = ("Optimal df = 9"))
lines(ca$days, (ns.fit.ca$fitted.values))
```



New York with cubic splines:

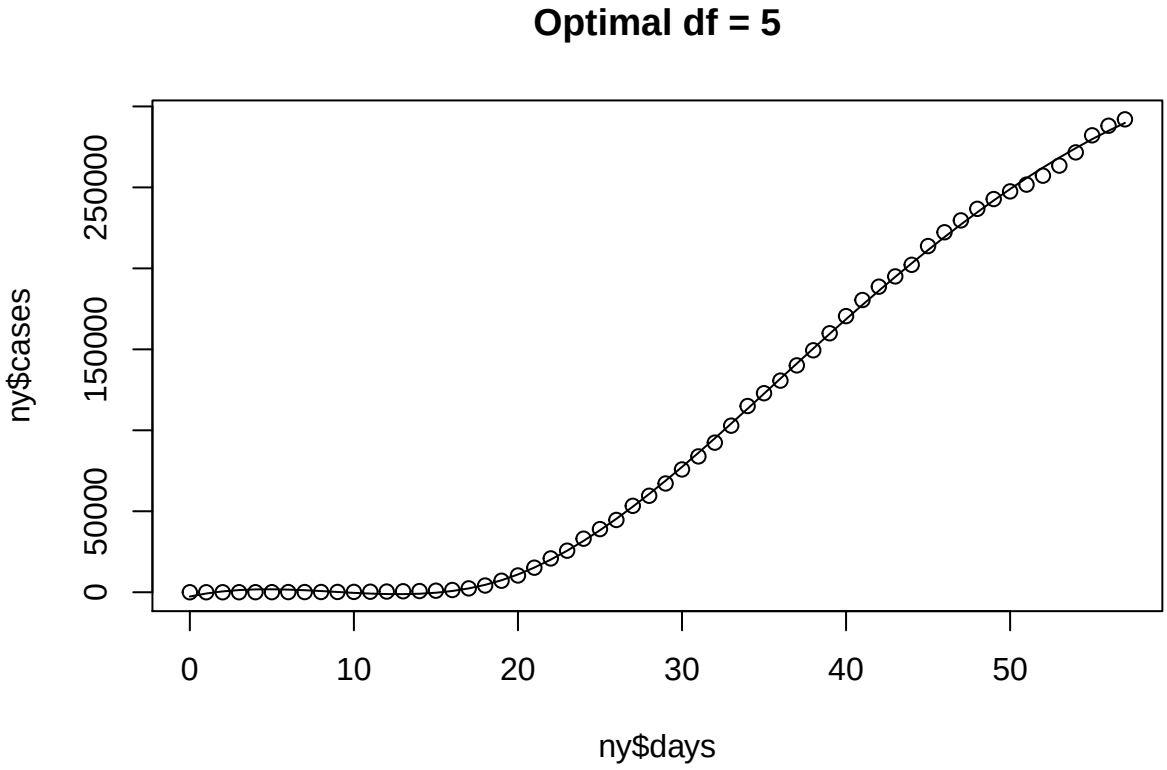
```
library(splines)
#bs
dfs <- (3 : 20)
errors <- c()
for (i in dfs){
fit <- lm(cases ~ bs(days, df = i), data = ny)
errors <- c(errors, sum(fit$residuals^2))
}
plot(dfs, errors, main = "New York, cubic splines")
```



From the plot we see, when the df is lower than 5, the errors are quite large. If df is greater than 5, the error will not change much. So we conclude that for New York dataset, the optimal degree of freedom in cubic spline method is 5.

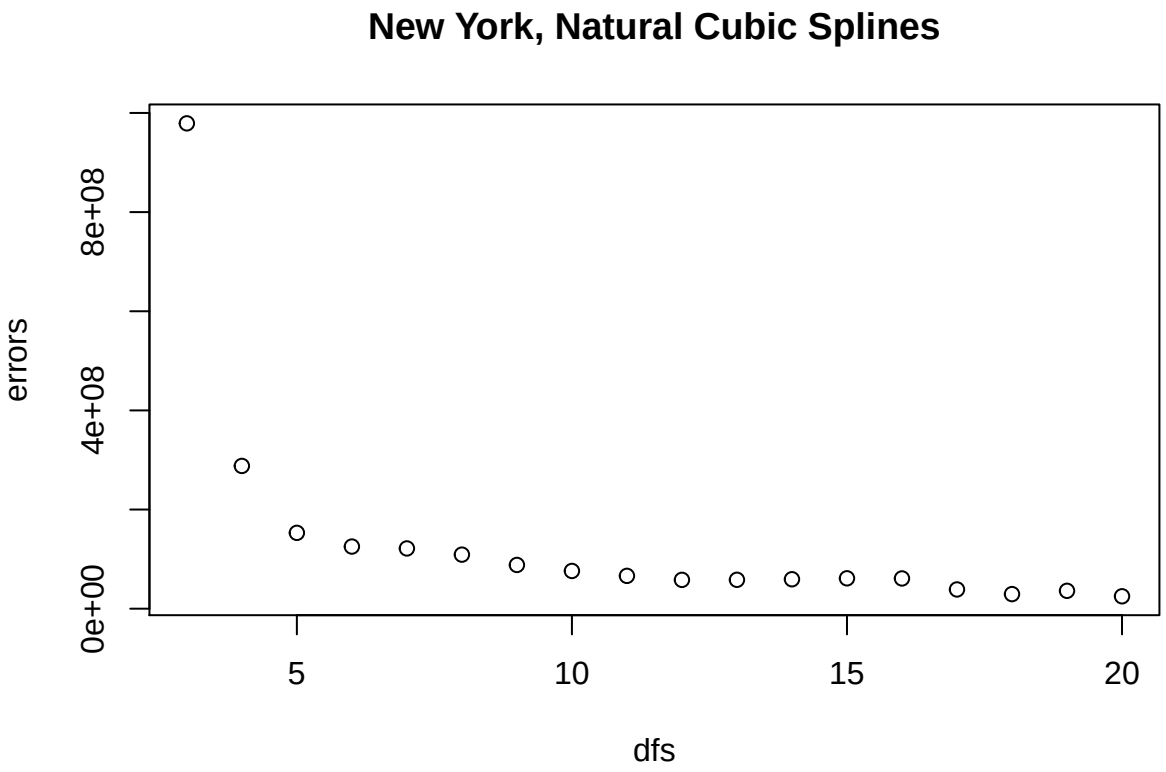
Then fit the cubic splines with df = 5 on New York data.

```
bs.fit.ny <- lm(cases ~ bs(days, df = 5), data = ny)
plot(ny$days, ny$cases, main = ("Optimal df = 5"))
lines(ny$days, (bs.fit.ny$fitted.values))
```



New York with natural cubic splines:

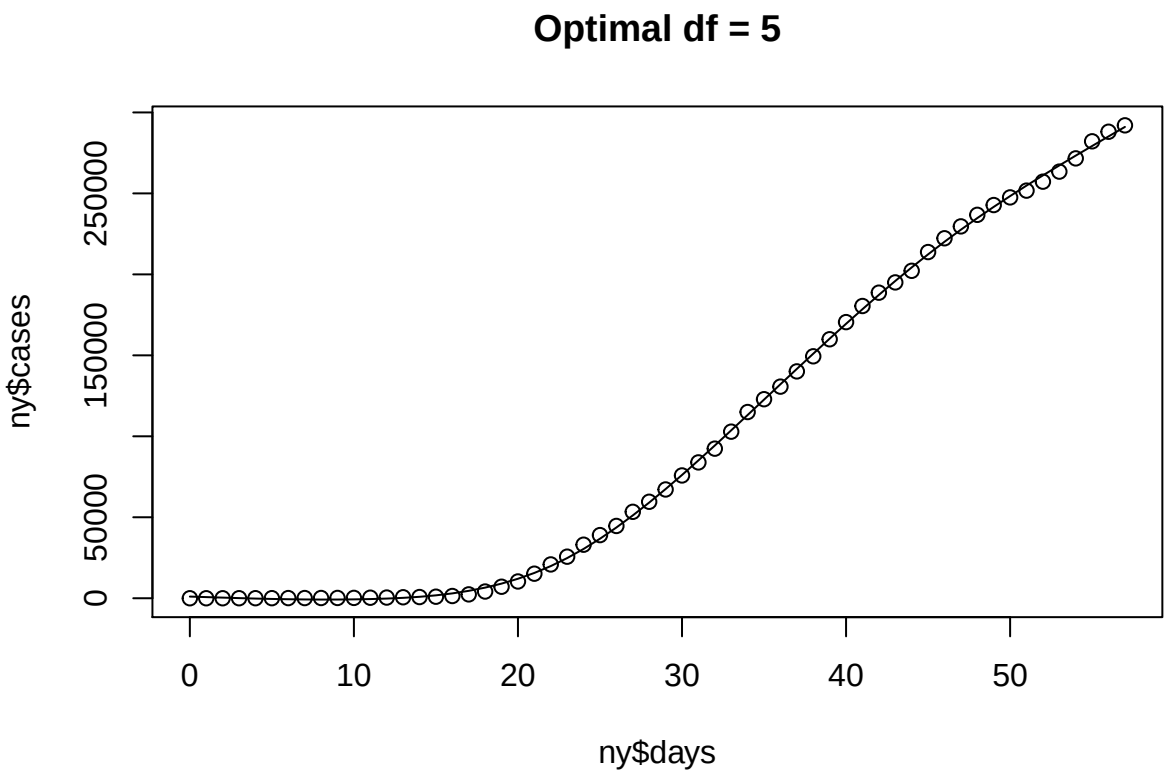
```
errors <- c()
for (i in dfs){
fit <- lm(cases ~ ns(days, df = i), data = ny)
errors <- c(errors, sum(fit$residuals^2))
}
plot(dfs, errors, main = "New York, Natural Cubic Splines")
```



We conclude that the optimal df is 5.

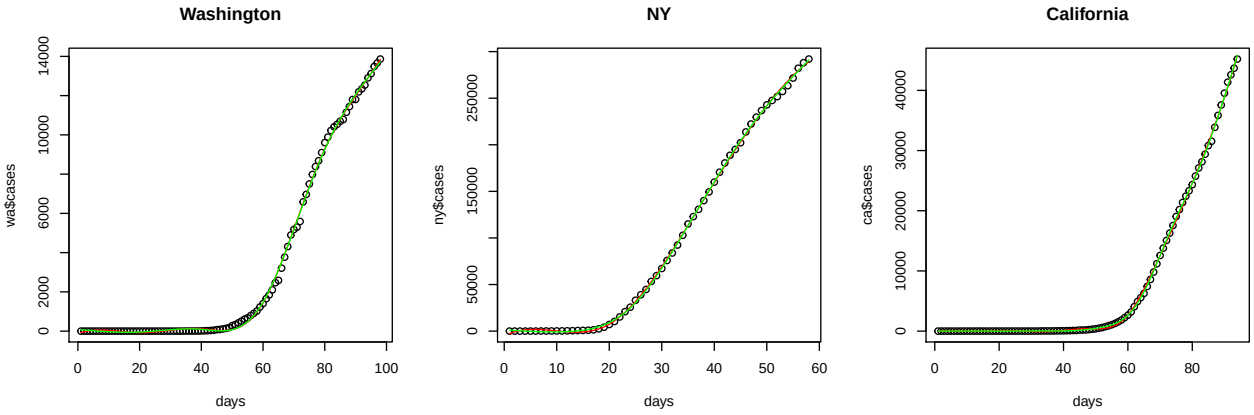
Then fit the natural cubic splines with df = 5 on New York data.

```
ns.fit.ny <- lm(cases ~ ns(days, df = 5), data = ny)
plot(ny$days, ny$cases, main = ("Optimal df = 5"))
lines(ny$days, (ns.fit.ny$fitted.values))
```



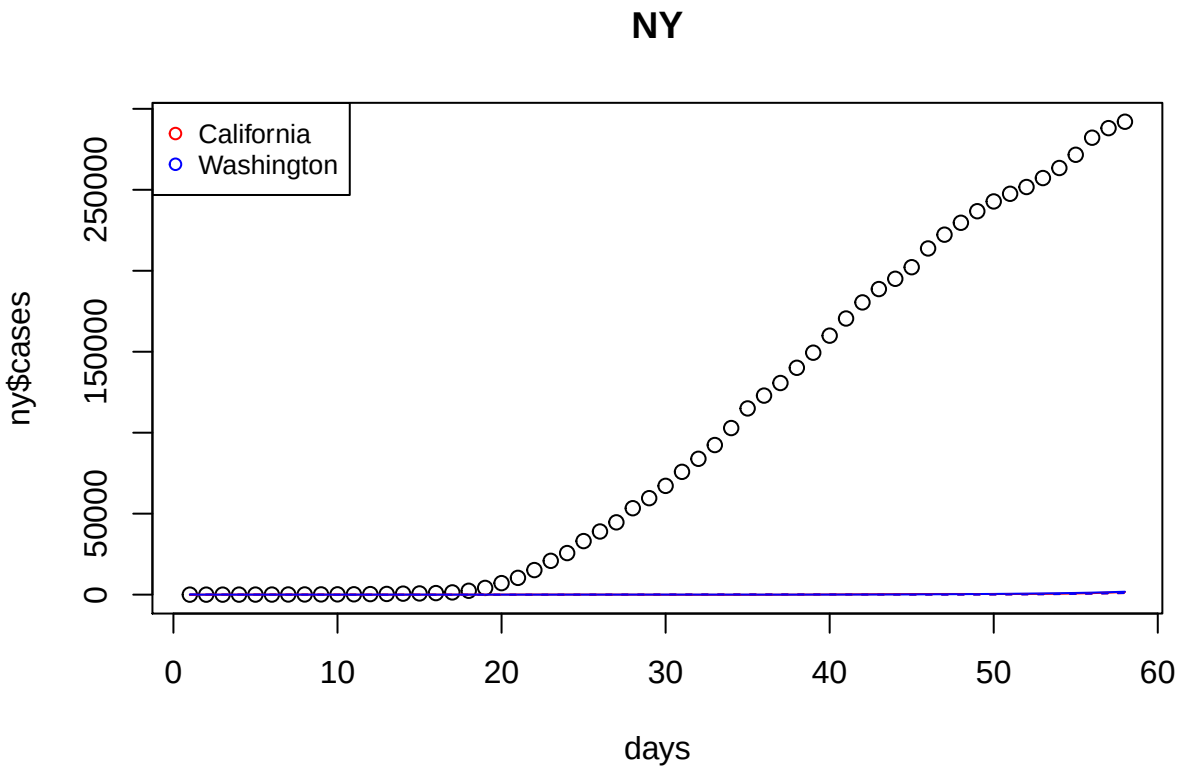
Compare different splines in different state

We plot the observed data, cubic splines and natural cubic splines.



We conclude that the cubic spline and natural cubic spline are quite similar after we chose the optimal degree of freedom. From the plot we know the cubic splines and natural cubic splines can both describe the model pretty well on the 3 subsets.

Predict New York cases



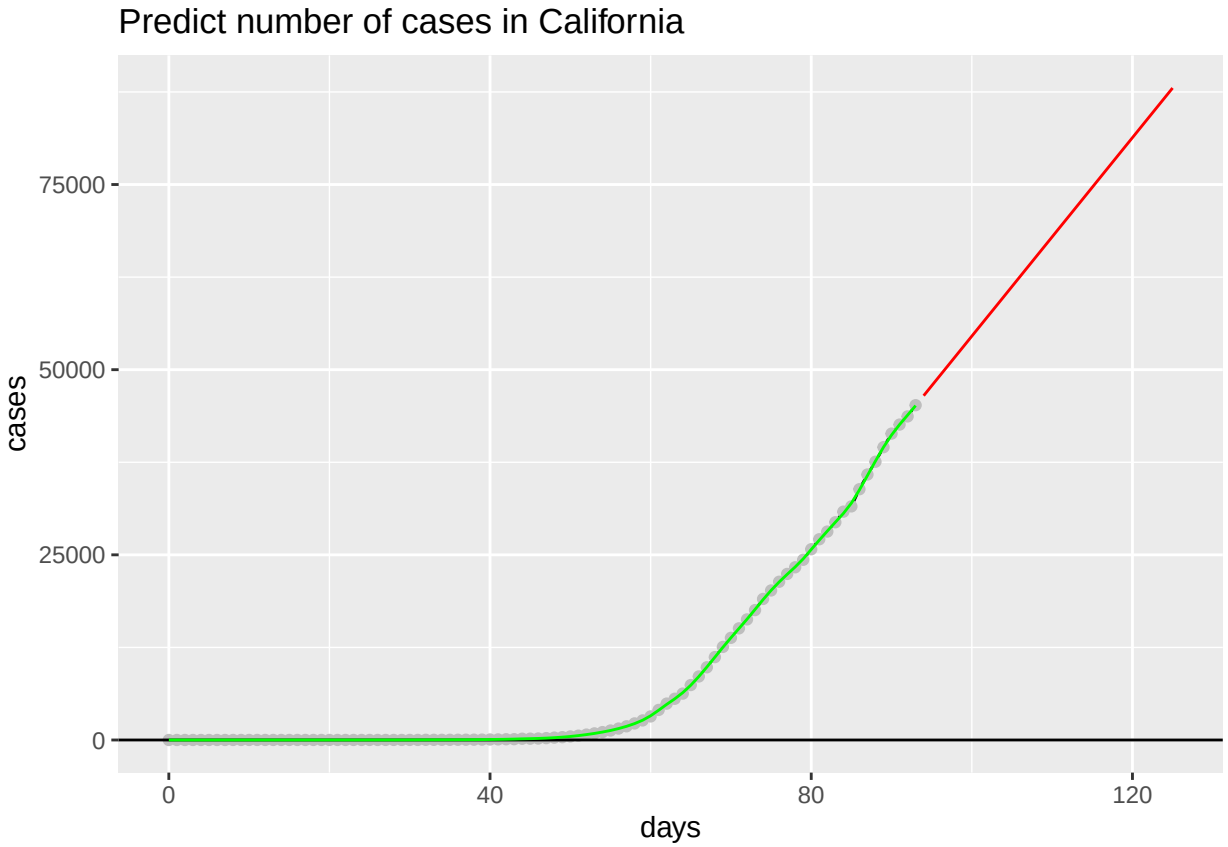
From the plot we see none of the predictions are good. The predictions are not good because New York has far more cases than those in California and Washington. Also, the growth rate of cases in New York is much higher than those of the other 2 states.

Smoothing Spline for California and Predict

```
## develop a model
smooth.fit <- smooth.spline(x =ca[,"days"],
                            y = ca[,"cases"], w = NULL, cv = FALSE)
cafit <- data.frame(days = ca$days)
cafitcases <- predict(smooth.fit, x = ca$days)
cafit$cases <- cafitcases$y

capred<- data.frame(days = 94:125)
predcases <- predict(smooth.fit, x = c(94:125))
capred$cases <- predcases$y

## plot the data
p1 <- ggplot(ca, aes(x = days, y=cases)) +
  geom_line() +
  geom_point(colour = "gray") +
  geom_hline(aes(yintercept=0)) +
  geom_line(color="red", data=capred) +
  geom_line(color="green", data=cafit) +
  labs(title = "Predict number of cases in California")
print(p1)
```



The gray dots are observed data, the green line is fitted line and the red line is extrapolation. In the first 50 days since the first case, the increasing rate of the number of cases is quite low. After 50 days since the first case, the number of cases increases very fast.

The extrapolation of smoothing splines is linear, which avoids erratic predictions in most cases.