STAT 435 HW7

Chongyi Xu

May 24, 2018

Question 1

First this problem, you will analyze a data set of your choice. Choose a data set that has $n >> p$.

(a) Describe the data in words. Where did you get it from and what is the data about?

The dataset I will use is `California Housing Data (1990)` Dataset that I found at kaggle. This dataset appeared in a 1997 paper titled Sparse Spatial Autoregressions by Pace, R. Kelley and Ronald Barry, published in the Statistics and Probability Letters journal. They built it using the 1990 California census data. It contains one row per census block group. A block group is the smallest geographical unit for which the U.S. Census Bureau publishes sample data (a block group typically has a population of 600 to 3,000 people)

```
dat <- read.csv('housing.csv', header=T)
colnames(dat)
```

```
##  [1] "longitude"          "latitude"           "housing_median_age"
##  [4] "total_rooms"        "total_bedrooms"     "population"
##  [7] "households"         "median_income"      "median_house_value"
## [10] "ocean_proximity"
```

- `longtitude`: the longtitude of the housing
- `latitude`: the longtitude of the housing
- `housing_median_age`: the median age of the housing
- `total_rooms`: total number of rooms in the district
- `total_bedrooms`: total number of bedrooms in the district
- `population`: total population in the district
- `households`: total households in the district
- `median_income`: median of income in the district
- `median_house_value`: median of the house value in the district
- `ocean_proximity`: a factor indicating the distance to ocean (`near bay`, `near ocean`, `inland`, `<1H ocean`)

We also want to check if there is any missing values.

```
sum(is.na(dat))
```

```
## [1] 207
```

So we have to deal with the missing values. For this problem, I want to use `median_house_value` as the response Y. `median_income`, `households`, `population`, `total_bedrooms`, `total_rooms`, `housing_median_age` as my features X_j.

```
dat <- dat[-1][-1]
dat$ocean_proximity <- NULL
dat <- na.omit(dat)
print(paste('n=', nrow(dat)))
```

```
## [1] "n= 20433"
```

```
print(paste('p=', ncol(dat[colnames(dat)!='median_house_value'])))
```

```
## [1] "p= 6"
```

(b) Fit a generalized additive model

I will use 5-fold CV to find the level of complexity.

```
library(gam)
```

```
## Warning: package 'gam' was built under R version 3.4.4
```

```
## Loading required package: splines
```

```
## Loading required package: foreach
```

```
## Warning: package 'foreach' was built under R version 3.4.4
```

```
## Loaded gam 1.15
```

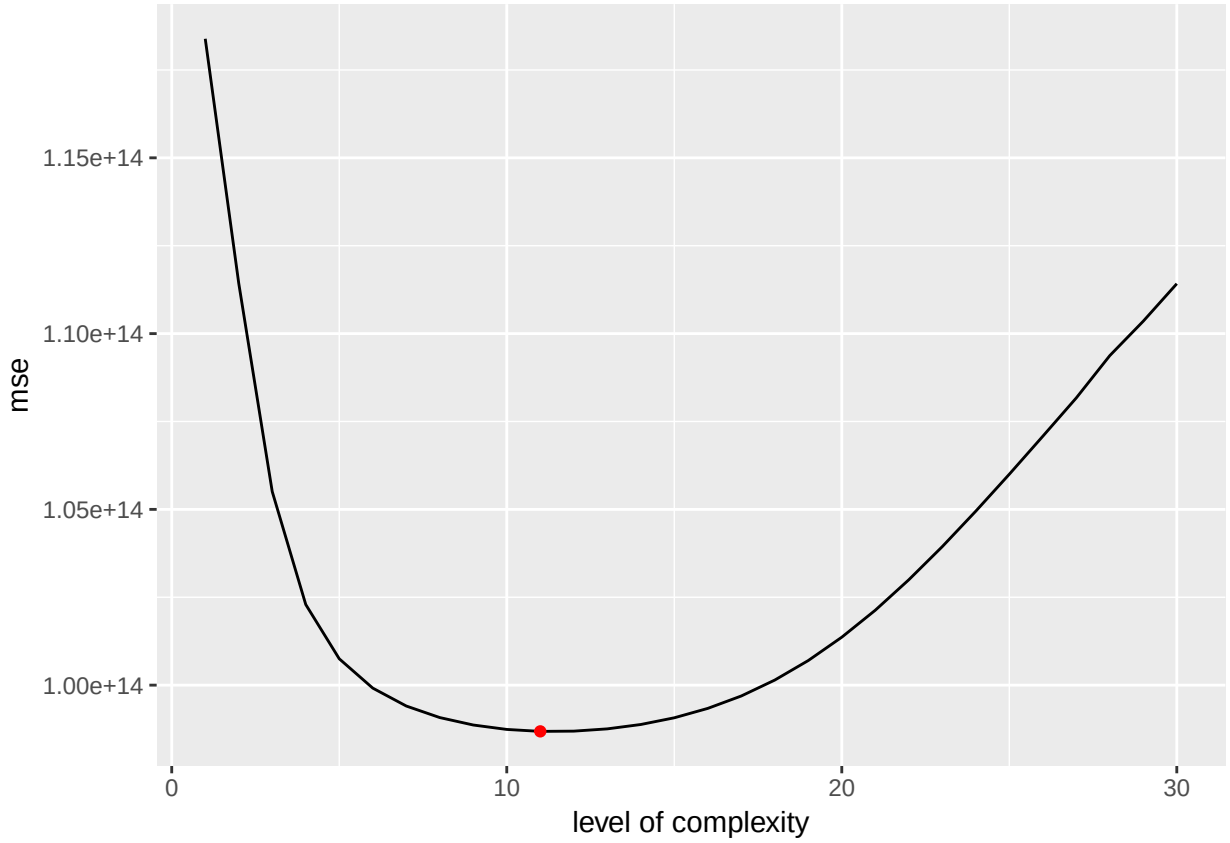
```
n <- nrow(dat)
lvls <- seq(1,30)
mse <- rep(0, length(lvls))

set.seed(435)
index <- sample(1:n, n)

for (k in 1:5) {
  for (i in 1:length(lvls)) {
    lvl <- lvls[i]
    i_train <- index[index %% 5 != k-1]
    dat.train <- dat[i_train,]
    dat.test <- dat[-i_train,]
    model <- gam(median_house_value ~
                   s(median_income, lvl) +
                   s(households, lvl) +
                   s(population, lvl) +
                   s(total_bedrooms, lvl) +
                   s(total_rooms, lvl) +
                   s(housing_median_age, lvl),
                 data=dat.train)
    y_pred <- predict(model, newdata=dat.test)
    mse[i] <- mse[i] +
      sum((y_pred-dat.test$median_house_value)^2)
  }
}

gam.mse <- mse/n
lvl <- lvls[which.min(gam.mse)]
print(paste('The level of complexity found is ', lvl))
```

```
## [1] "The level of complexity found is  11"
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.4
```

```
ggplot() + geom_line(aes(lvls, mse)) + geom_point(aes(x=lvl, y=mse[which(lvls==lvl)]), color='red') + xl
```




```
library(visreg)
```

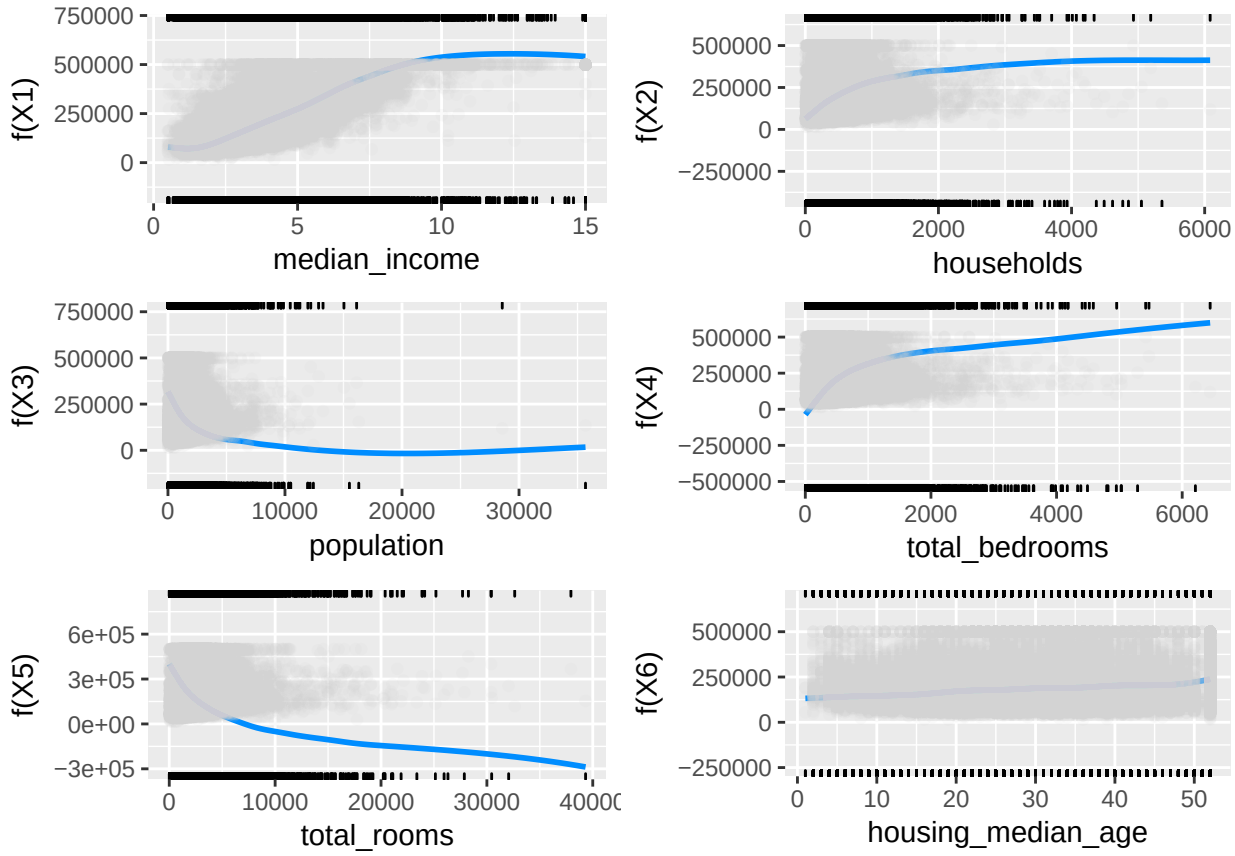
```
## Warning: package 'visreg' was built under R version 3.4.4
```

```
library(gridExtra)
```

```
## Warning: package 'gridExtra' was built under R version 3.4.4
```

```
lvl <- lvls[which.min(mse)]
model <- gam(median_house_value ~
                   s(median_income, lvl) +
                   s(households, lvl) +
                   s(population, lvl) +
                   s(total_bedrooms, lvl) +
                   s(total_rooms, lvl) +
                   s(housing_median_age, lvl),
                 data=dat)

p1 <- visreg(model, 'median_income',
             gg=T, partial=F) +
  geom_point(aes(x=dat$median_income,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X1)')
```

```
p2 <- visreg(model, 'households',
             gg=T, partial=F) +
  geom_point(aes(x=dat$households,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X2)')

p3 <- visreg(model, 'population',
             gg=T, partial=F) +
  geom_point(aes(x=dat$population,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X3)')

p4 <- visreg(model, 'total_bedrooms',
             gg=T, partial=F) +
  geom_point(aes(x=dat$total_bedrooms,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X4)')

p5 <- visreg(model, 'total_rooms',
             gg=T, partial=F) +
  geom_point(aes(x=dat$total_rooms,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X5)')

p6 <- visreg(model, 'housing_median_age',
             gg=T, partial=F) +
  geom_point(aes(x=dat$housing_median_age,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X6)')

grid.arrange(p1,p2,p3,p4,p5,p6,nrow=3)
```

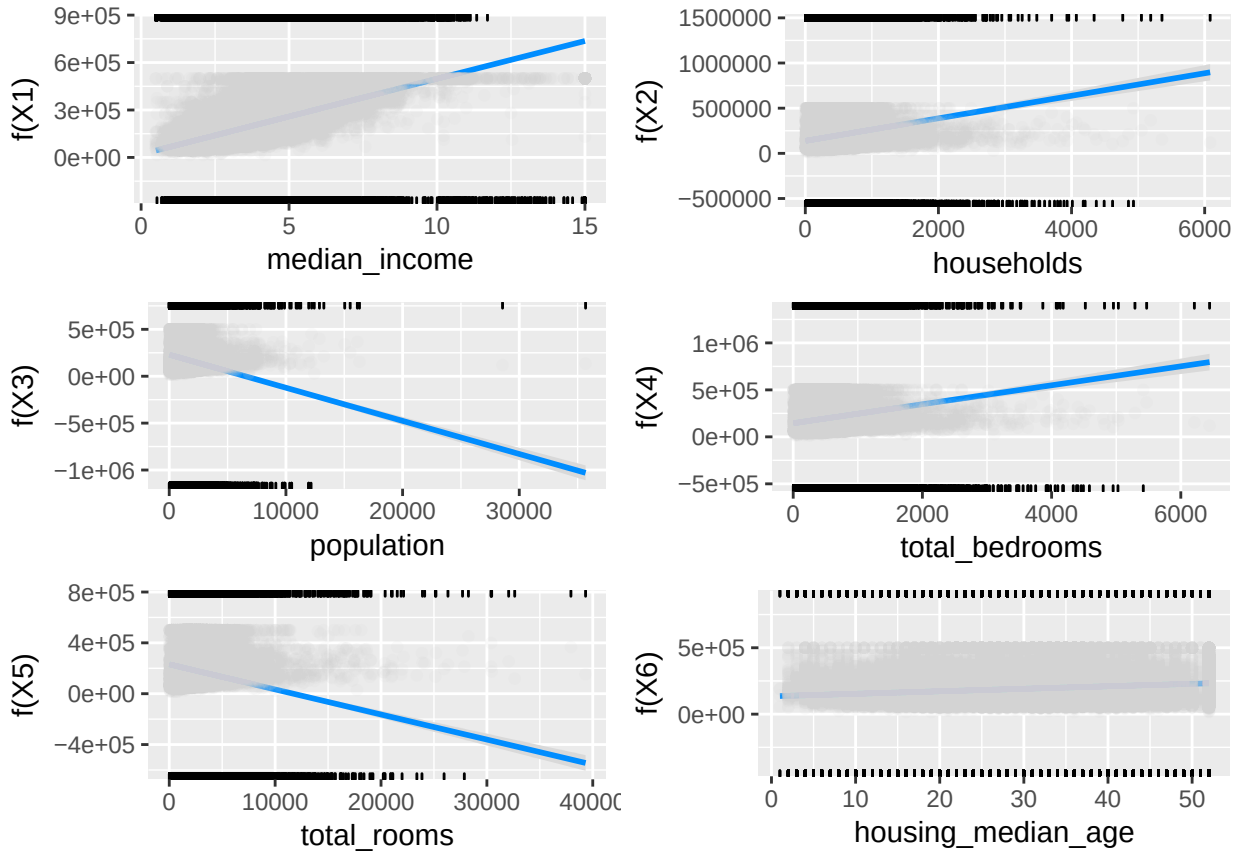


From the plot, we can see that the `housing_median_age` seems not affecting the housing value. The other features have some exponential relations with the housing value.

(c) Now fit a linear model.

```
model <- lm(median_house_value ~
                    median_income +
                    households +
                    population +
                    total_bedrooms +
                    total_rooms +
                    housing_median_age,
                  data=dat)

p1 <- visreg(model, 'median_income',
             gg=T, partial=F) +
  geom_point(aes(x=dat$median_income,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X1)')

p2 <- visreg(model, 'households',
             gg=T, partial=F) +
  geom_point(aes(x=dat$households,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X2)')
```

```r
p3 <- visreg(model, 'population',
             gg=T, partial=F) +
  geom_point(aes(x=dat$population,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X3)')

p4 <- visreg(model, 'total_bedrooms',
             gg=T, partial=F) +
  geom_point(aes(x=dat$total_bedrooms,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X4)')

p5 <- visreg(model, 'total_rooms',
             gg=T, partial=F) +
  geom_point(aes(x=dat$total_rooms,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X5)')

p6 <- visreg(model, 'housing_median_age',
             gg=T, partial=F) +
  geom_point(aes(x=dat$housing_median_age,
                 y=dat$median_house_value),
             color='lightgray', alpha=0.1) +
  ylab('f(X6)')

grid.arrange(p1,p2,p3,p4,p5,p6,nrow=3)
```



(d) Estimate the test error of the generalized additive model and test error of the linear model.

```
lm.model <- lm(median_house_value ~
                   median_income +
                   households +
                   population +
                   total_bedrooms +
                   total_rooms +
                   housing_median_age,
                 data=dat.train)

lm.y_pred <- predict(lm.model, newdata=dat.test)

lm.mse <- sum((lm.y_pred-dat.test$median_house_value)^2)/n
```

```
methods <- c('Generalized additive model', 'Linear model')
mses <- c(min(gam.mse), lm.mse)
print(paste('The test error of generalized additive model with chosen level of complexity is ', min(gam
```

```
## [1] "The test error of generalized additive model with chosen level of complexity is  4829680678.546
```

```
print(paste('The test error of linear model is ', min(lm.mse)))
```

```
## [1] "The test error of linear model is  1194041083.46532"
```

```
print(paste('So the method obtained a smaller test error is ', methods[which(mses==min(mses))]))
```

```
## [1] "So the method obtained a smaller test error is  Linear model"
```

Question 2

In this problem, we'll play around with regression splines.

(a) Generate data.

```
set.seed(7)
x <- 1:1000
y <- sin((1:1000)/100)*4 + rnorm(100)
```

Consider the model

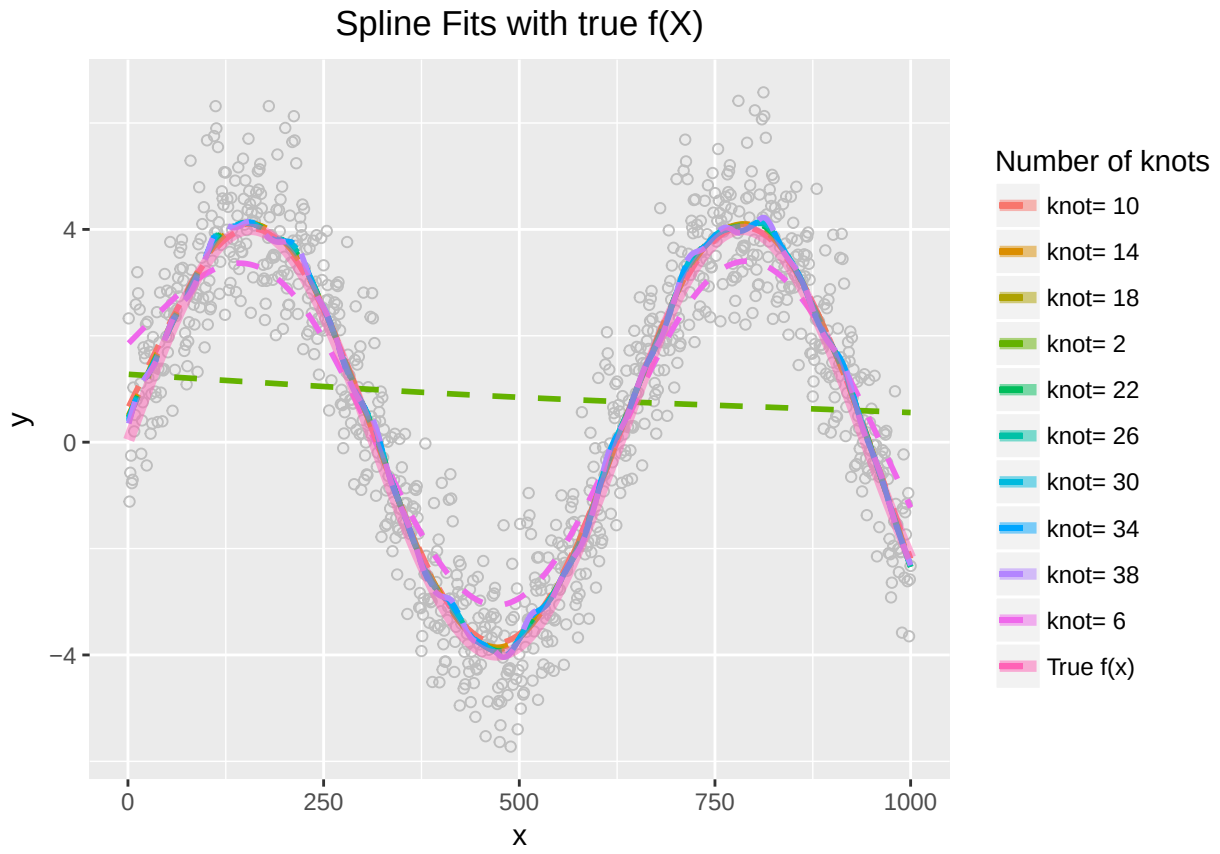
$$Y = f(X) + \epsilon$$

What is the form of $f(X)$ for this simulation setting? What is the value of $\text{Var}(\epsilon)$? What is the value of $\text{E}(Y - f(X))^2$?

From the data generating process, we know that $f(x) = 4sin(\frac{x}{100})$. Since `rnorm(100)` is equivalent to `rnorm(n=100,mean=0,sd=1)`, $\text{Var}(\epsilon) = 1$, and $\text{E}(Y - f(X))^2$ =$\text{E}(\epsilon^2)$ =$\text{Var}(\epsilon)$+$\text{E}(\epsilon^2) = 1$

(b) Fit regression splines for various numbers of knots to this simulated data, in order to get spline fits ranging from very wiggly to very smooth. Make a plot of the results, showing the raw data, the true $f(X)$, and the spline fits.

```
knots <- seq(2,40, by=4)
yhat <- matrix(NA, nrow=length(knots), ncol=length(x))
for (i in 1:length(knots)) {
  model <- smooth.spline(x=x,y=y,df=knots[i])
  yhat[i,] <- predict(model)$y
}

p <- ggplot() + geom_point(aes(x, y), color='gray', shape=1)
for (i in 1:length(knots)) {
  p <- p + geom_line(aes_string(x, y=yhat[i,],
                                col=as.factor
                                (paste('knot=', knots[i]))),
                     lwd=1, linetype='dashed')
}
p + geom_line(aes(x=x, y=4*sin(x/100), col='True f(x)'), lwd=2, alpha=0.5) +
  guides(color=guide_legend(title='Number of knots')) +
  ggtitle('Spline Fits with true f(X)') +
  theme(plot.title = element_text(hjust = 0.5))
```



(c) Based on the visual inspection, how many knots seem to give the "best" fit?

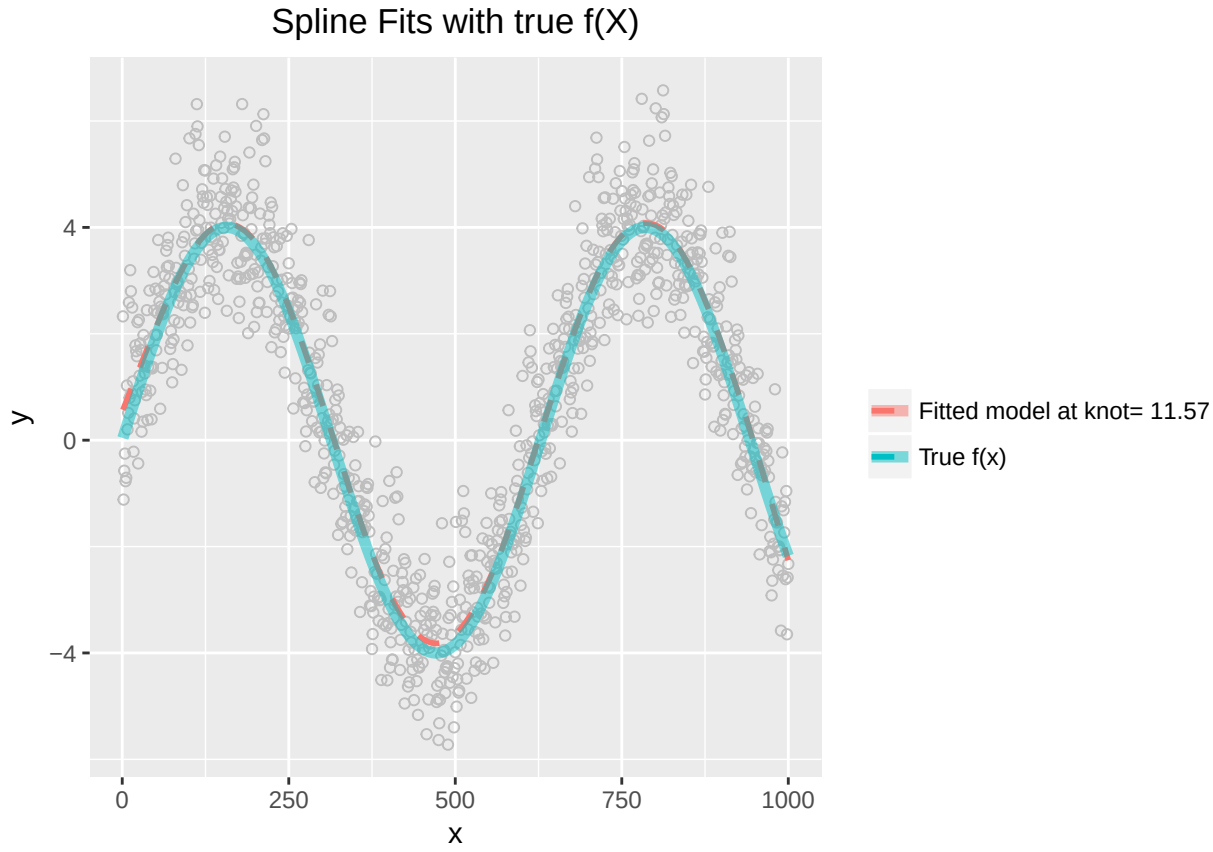
From visual inspection, knots from 10 to 30 all fit the true $f(X)$ pretty well. With knots less than $10(2, 6)$, there are some errors. And with knots greater than $22(26, 30, 34, 38)$, the model seems to be overfitting.

(d) Now perform cross-validation in order to select the optimal number of knots.

```
model.fit <- smooth.spline(x=x, y=y, cv=T)
print(paste('The best number of knots is', model.fit$df))
```

```
## [1] "The best number of knots is 11.5672723208619"
```

```
model.y_pred <- predict(model.fit)$y
ggplot() + geom_point(aes(x, y), color='gray', shape=1) +
  geom_line(aes(x, y=model.y_pred,
                col=paste('Fitted model at knot=', round(model.fit$df, 2))),
                    lwd=1, linetype='dashed') +
  geom_line(aes(x=x, y=4*sin(x/100), col='True f(x)'), lwd=2, alpha=0.5) +
  guides(color=guide_legend(title='')) +
  ggtitle('Spline Fits with true f(X)') +
  theme(plot.title = element_text(hjust = 0.5))
```



We can see that at the knot number = 11.57, the fitted model fits the true $f(X)$ pretty well.

(e) Provide an estimate of the test error $E(Y - \hat{f}(X))^2$, associated with the spline $\hat{f}(\cdot)$ from (d). How does this related to your answer in (a)?

We can simply obtain the estimation of the test error using the CV fitted model that we used in part(d). `model.fit$cv.crit` will give the test error.

```
model.fit$cv.crit
```

```
## [1] 0.9298117
```

As we can see, the estimation error is close to 1, as I stated in part(a) ($\mathrm{E}(Y - f(X))^2 = \mathrm{E}(\epsilon^2) = \mathrm{Var}(\epsilon) + \mathrm{E}(\epsilon^2) = 1$)

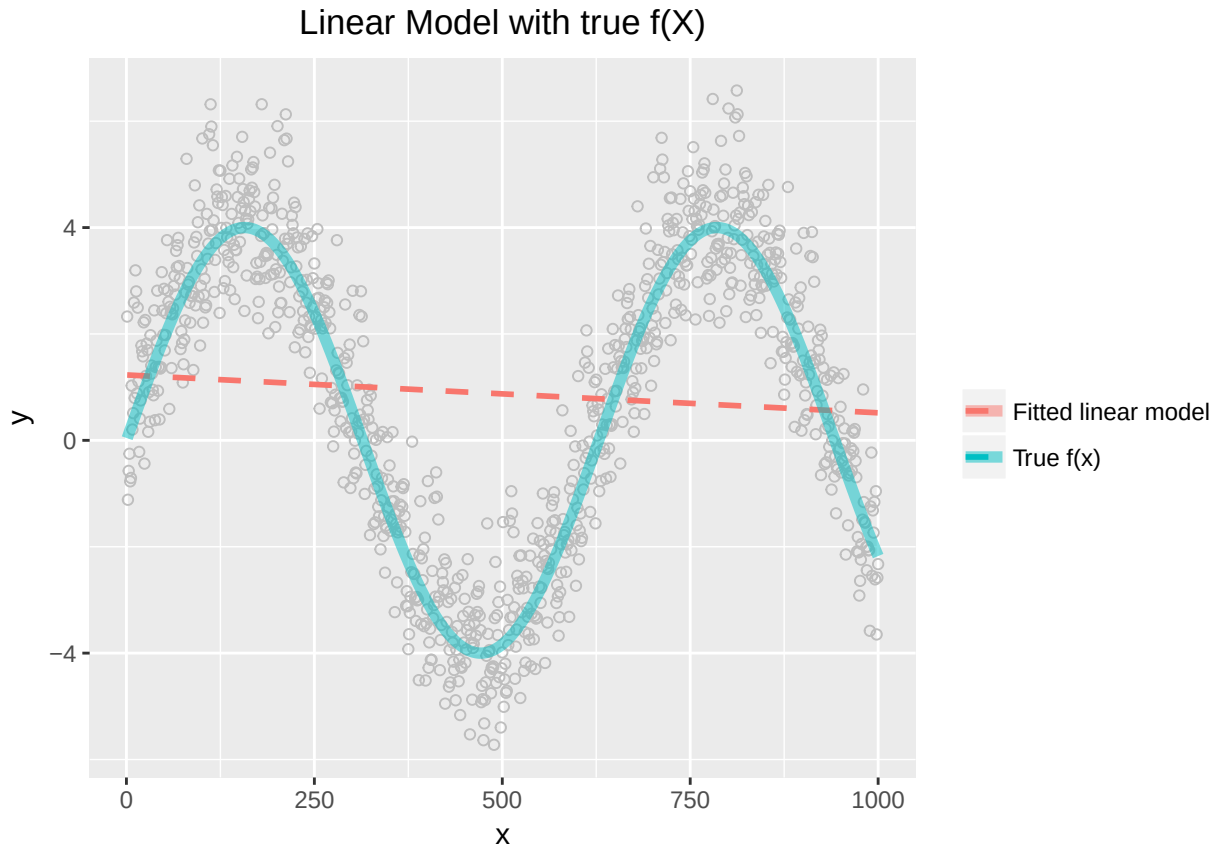
(f) Now fit a linear model of the form

$$Y = \beta_0 + \beta_1 X + \epsilon$$

to the data instead. Plot the new raw data and the fitted model and the true function $f(\cdot)$. Provide an estimate of the test error associated with the fitted model.

```
model.fit <- lm(y ~ x)
model.y_pred <- predict(model.fit)

ggplot() + geom_point(aes(x, y), color='gray', shape=1) +
  geom_line(aes(x, y=model.y_pred,
                col='Fitted linear model'),
                     lwd=1, linetype='dashed') +
```

```
  geom_line(aes(x=x, y=4*sin(x/100), col='True f(x)'), lwd=2, alpha=0.5) +
  guides(color=guide_legend(title='')) +
  ggtitle('Linear Model with true f(X)') +
  theme(plot.title = element_text(hjust = 0.5))
```



Using 5-fold CV to find the estimation test error.

```
n <- length(x)
index <- sample(1:n, n)
dat <- data.frame(x=x, y=y)
mse <- 0

for (k in 1:5) {
  i_train <- index[index %% 5 != k-1]
  dat.train <- dat[i_train,]
  dat.test <- dat[-i_train,]
  model.fit <- lm(data=dat.train, y ~ x)
  y_pred <- predict(model.fit, newdata=dat.test)
  mse <- mse + sum((dat.test$y - y_pred)^2)
}

mse <- mse/n
print(paste('The test error of the linear model is ', mse))
```

```
## [1] "The test error of the linear model is  8.0031983558057"
```

The linear model obtained a test error of 8.003 which is much larger than the test error in part(e).