HW1

Nan Tang

4/9/2020

Part 1

(a)

```
ksmooth.train <- function(x.train, y.train, kernel=c('box', 'normal'),
                          bandwidth=0.5, CV=FALSE) {
  train.length <- length(x.train)
  yhat <- numeric(train.length)

  if(kernel=='box') {
    if(CV) {
      for (i in 1:train.length) {
        x.i <- x.train[i]
        x.train.cv <- x.train[-i]
        y.train.cv <- y.train[-i]
        yhat[i] <- mean(y.train.cv[which(abs(x.train.cv - x.i) < 0.5 * bandwidth)])
      }
    } else {
      for (i in 1:train.length) {
        x.i <- x.train[i]
        yhat[i] <- mean(y.train[which(abs(x.train - x.i) < 0.5 * bandwidth)])
      }
    }
  } else{
    # get sigma
    gk.sigma <- bandwidth * 0.25 / qnorm(0.75, 0, sd=1)

    if (CV) {
      for (i in 1:train.length) {
        gk.out <- dnorm(x.train[-i], x.train[i], gk.sigma)
        yhat[i] <- sum(y.train[-i] * gk.out) / sum(gk.out)
      }
    } else {
      for (i in 1:train.length) {
        gk.out <- dnorm(x.train, x.train[i], gk.sigma)
        yhat[i] <- sum(y.train * gk.out) / sum(gk.out)
      }
    }
  }

  # output
```

```r
  output.list <- list(x.train, yhat)
  names(output.list) <- c('x.train', 'yhat.train')
  return(output.list)
}
```

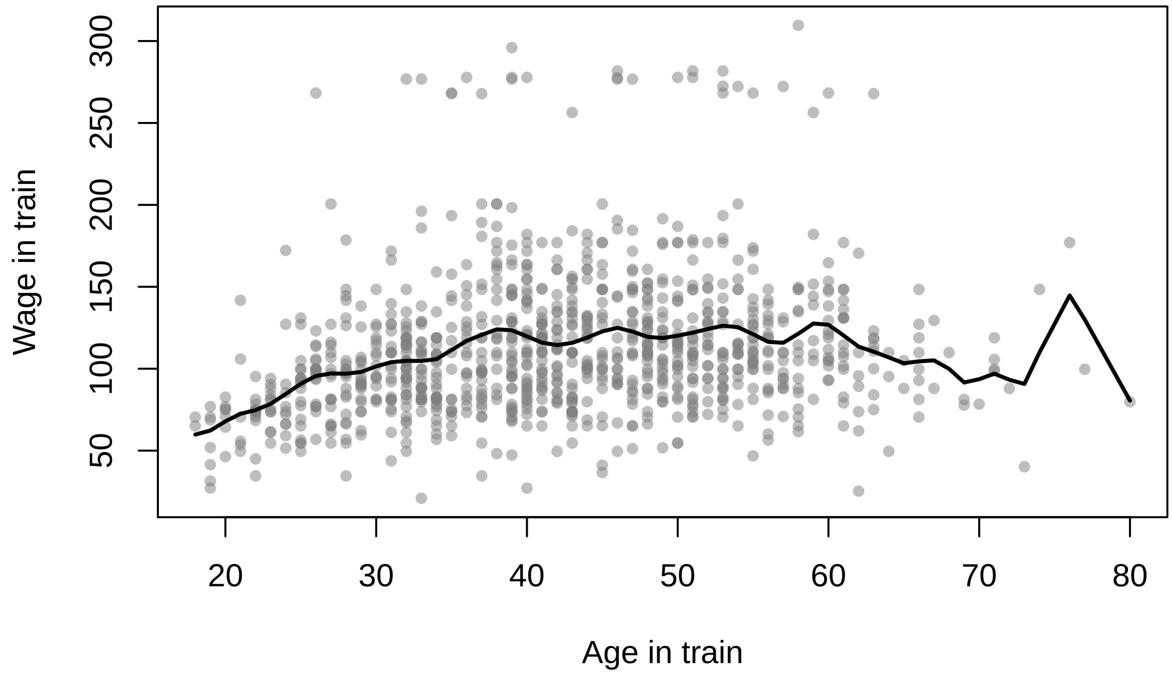
(b)

```r
ksmooth.predict <- function(ksmooth.train.out, x.query) {
  y.pred <- numeric(length(x.query))

  # linear interpolation inside range
  # values outside range are stored as NA
  y.pred <- approx(ksmooth.train.out$x.train, ksmooth.train.out$yhat.train, x.query,
                   method='linear', ties=mean)[[2]]

  x.train.min <- min(ksmooth.train.out$x.train)
  x.train.max <- max(ksmooth.train.out$x.train)
  yhat.extrap.down <- mean(ksmooth.train.out$yhat.train[which(ksmooth.train.out$x.train==x.train.min)])
  yhat.extrap.up <- mean(ksmooth.train.out$yhat.train[which(ksmooth.train.out$x.train==x.train.max)])

  extrap.down.index <- which(x.query < x.train.min)
  extrap.up.index <- which(x.query > x.train.max)

  # constant extrapolation for x values smaller than min of x.train
  if (length(extrap.down.index) > 0) {
    y.pred[extrap.down.index] <- yhat.extrap.down
  }

  # constant extrapolation for x values greater than max of x.train
  if (length(extrap.up.index) > 0) {
    y.pred[extrap.up.index] <- yhat.extrap.up
  }

  # output
  output.list <- list(x.query, y.pred)
  names(output.list) <- c('x.query', 'y.pred')
  return(output.list)
}
```

c

```r
train.age <- Wage.train$age
train.wage <- Wage.train$wage

ksmooth.train.out <- ksmooth.train(train.age, train.wage, kernel='normal', bandwidth=3)

ks.out.df <- data.frame(ksmooth.train.out$x.train, ksmooth.train.out$yhat.train)
ks.out.df <- ks.out.df[order(ks.out.df[,1]),]

plot(train.age, train.wage, col=alpha('gray50', alpha=0.5),
```

```r
     pch=16, cex=0.8, xlab='Age in train', ylab='Wage in train')
lines(ks.out.df, lwd=2)
```

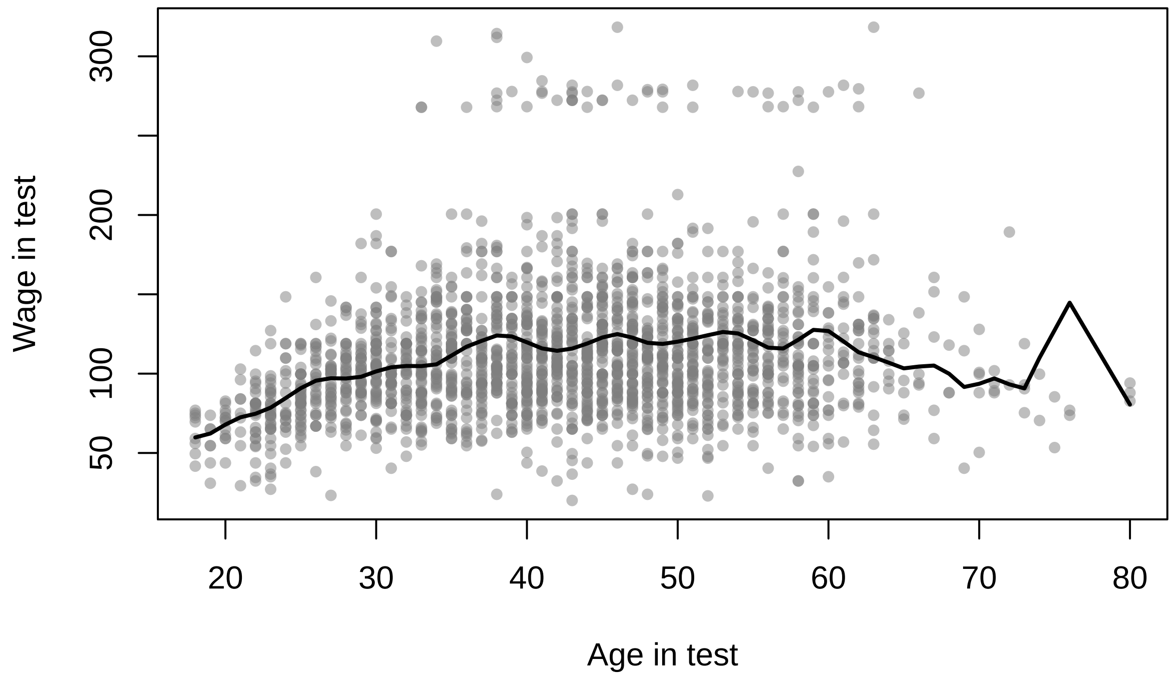



```r
RSS.train <- sum((train.wage - ksmooth.train.out$yhat.train)^2)
print(RSS.train)
```

```
## [1] 1625121
```

d

```r
test.age <- Wage.test$age
test.wage <- Wage.test$wage

ksmooth.pred.out <- ksmooth.predict(ksmooth.train.out, test.age)

ks.pred.df <- data.frame(ksmooth.pred.out$x.query, ksmooth.pred.out$y.pred)
ks.pred.df <- ks.pred.df[order(ks.pred.df[,1]),]

plot(test.age, test.wage, col=alpha('gray50', alpha=0.5),
     pch=16, cex=0.8, xlab='Age in test', ylab='Wage in test')
lines(ks.pred.df, lwd=2)
```

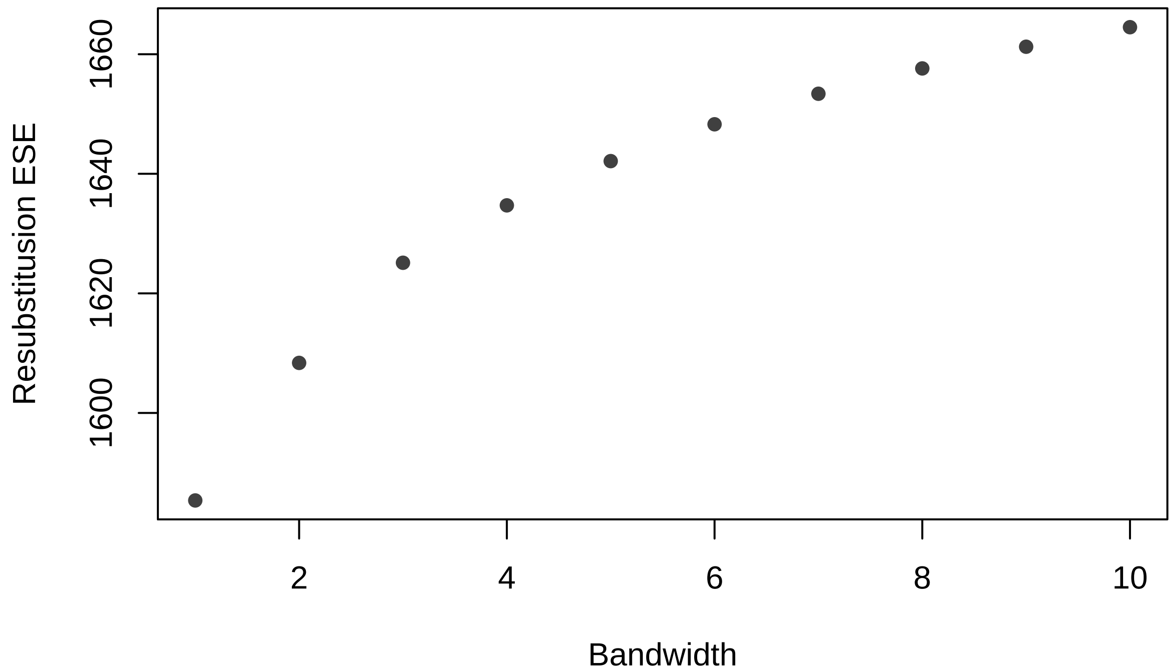



```
RSS.test <- sum((test.wage - ksmooth.pred.out$y.pred)^2)
print(RSS.test)
```

```
## [1] 3168000
```

e

```
ESE.resub <- numeric(length(1:10))

for (i in 1:10) {
  ks.train.temp <- ksmooth.train(train.age, train.wage, kernel='normal', bandwidth=i)
  ese.train <- sum((train.wage - ks.train.temp$yhat.train)^2) / length(train.wage)
  ESE.resub[i] <- ese.train
}

plot(1:10, ESE.resub, pch=16, col='gray25',
     xlab='Bandwidth', ylab='Resubstitusion ESE')
```

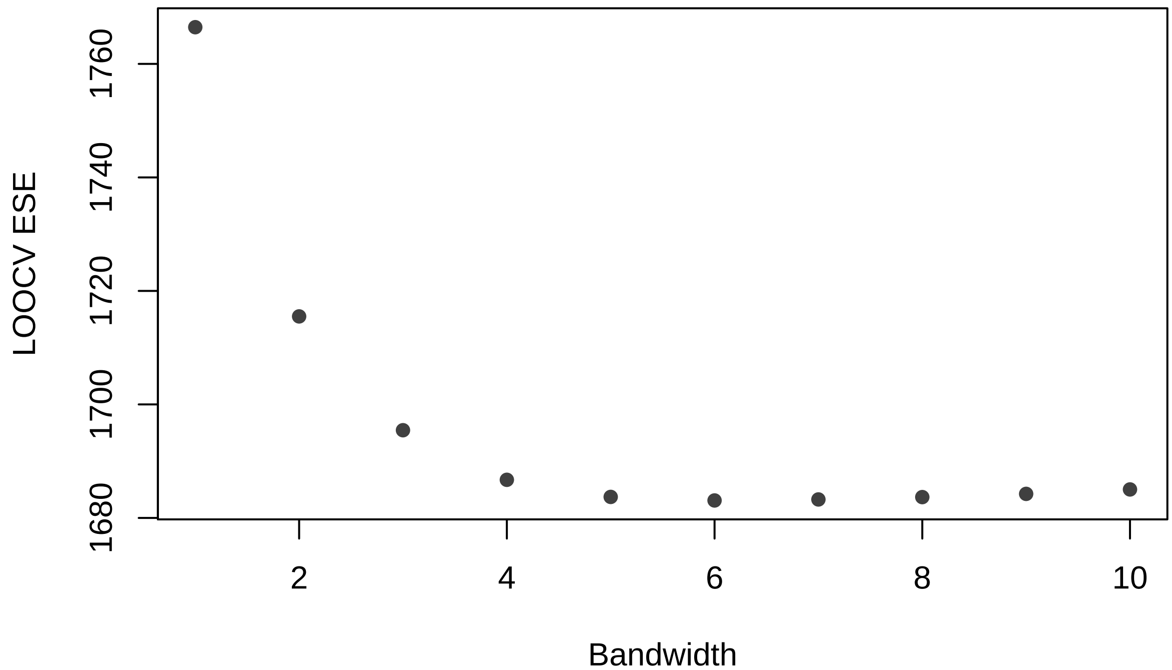



```
print(ESE.resub)
```

```
##  [1] 1585.364 1608.370 1625.121 1634.722 1642.120 1648.282 1653.387 1657.624
##  [9] 1661.252 1664.519
```

f

```
ESE.loocv <- numeric(length(1:10))

for (i in 1:10) {
  ks.train.cv.temp <- ksmooth.train(train.age, train.wage, kernel='normal', bandwidth=i, CV=TRUE)
  ese.train.cv <- sum((train.wage - ks.train.cv.temp$yhat.train)^2) /length(train.wage)
  ESE.loocv[i] <- ese.train.cv
}

plot(1:10, ESE.loocv, pch=16, col='gray25',
     xlab='Bandwidth', ylab='LOOCV ESE')
```



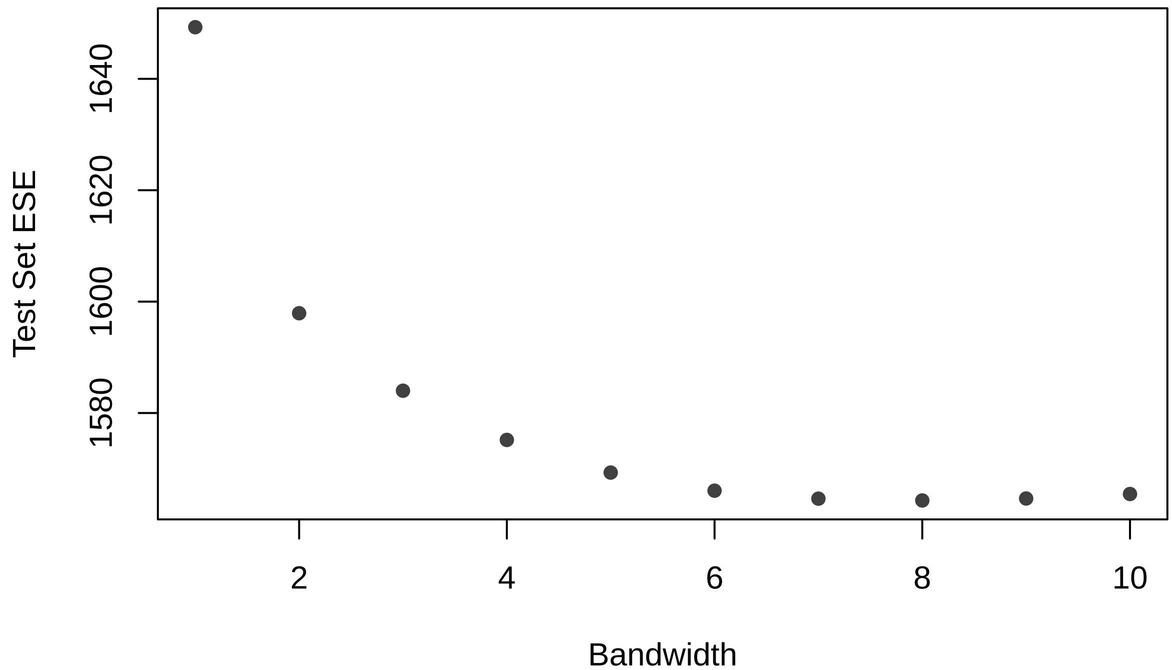

```
print(ESE.loocv)
```

```
##  [1] 1766.466 1715.508 1695.453 1686.715 1683.705 1683.079 1683.256 1683.671
##  [9] 1684.239 1685.021
```

I will choose bandwidth = 6.

g

```
ESE.est <- numeric(10)
for (i in 1:10) {
  ks.train.temp <- ksmooth.train(train.age, train.wage, kernel='normal', bandwidth=i)
  ks.pred.temp <- ksmooth.predict(ks.train.temp, test.age)
  ese.test <- sum((test.wage - ks.pred.temp$y.pred)^2) / length(test.wage)
  ESE.est[i] <- ese.test
}

plot(1:10, ESE.est, pch=16, col='gray25',
     xlab='Bandwidth', ylab='Test Set ESE')
```




```
print(ESE.est)
```

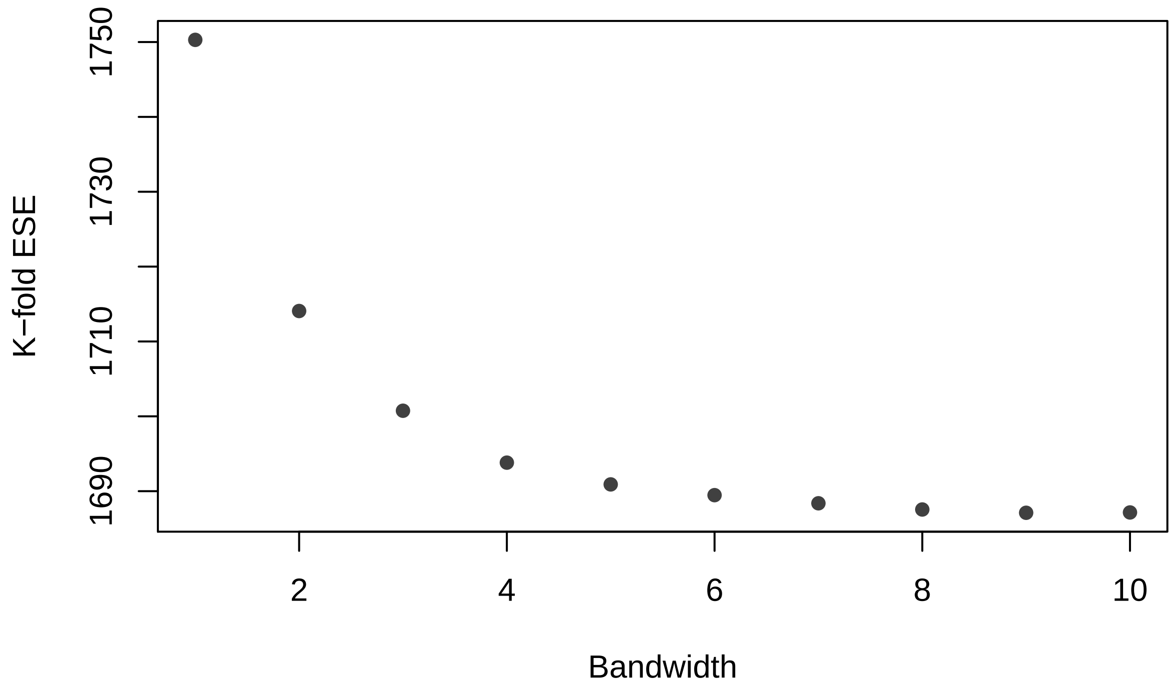
```
## [1] 1649.268 1597.913 1584.000 1575.168 1569.304 1566.052 1564.621 1564.297
## [9] 1564.647 1565.453
```

I will choose bandwidth = 8.

h

```
k <- 5
ESE.est <- numeric(10)

for (bw in 1:10) {
  ESE.5fold <- numeric(k)
  for (i in 1:k) {
    age.test.fold <- Wage.train$age[which(fold==i)]
    wage.test.fold <- Wage.train$wage[which(fold==i)]
    age.train.fold <- Wage.train$age[-which(fold==i)]
    wage.train.fold <- Wage.train$wage[-which(fold==i)]

    ks.train.temp <- ksmooth.train(age.train.fold, wage.train.fold, kernel='normal', bw)
    ks.pred.temp <- ksmooth.predict(ks.train.temp, age.test.fold)

    ESE.temp <- sum((wage.test.fold - ks.pred.temp$y.pred)^2) / length(wage.test.fold)

    ESE.5fold[i] <- ESE.temp
  }

  ESE.est[bw] <- sum(ESE.5fold) / k
}

plot(1:10, ESE.est, pch=16, col='gray25',
```

```
    xlab='Bandwidth', ylab='K-fold ESE')
```




```
print(ESE.est)
```

```
##  [1] 1750.292 1714.070 1700.746 1693.811 1690.903 1689.468 1688.382 1687.550
##  [9] 1687.115 1687.157
```

I will choose bandwidth = 9.

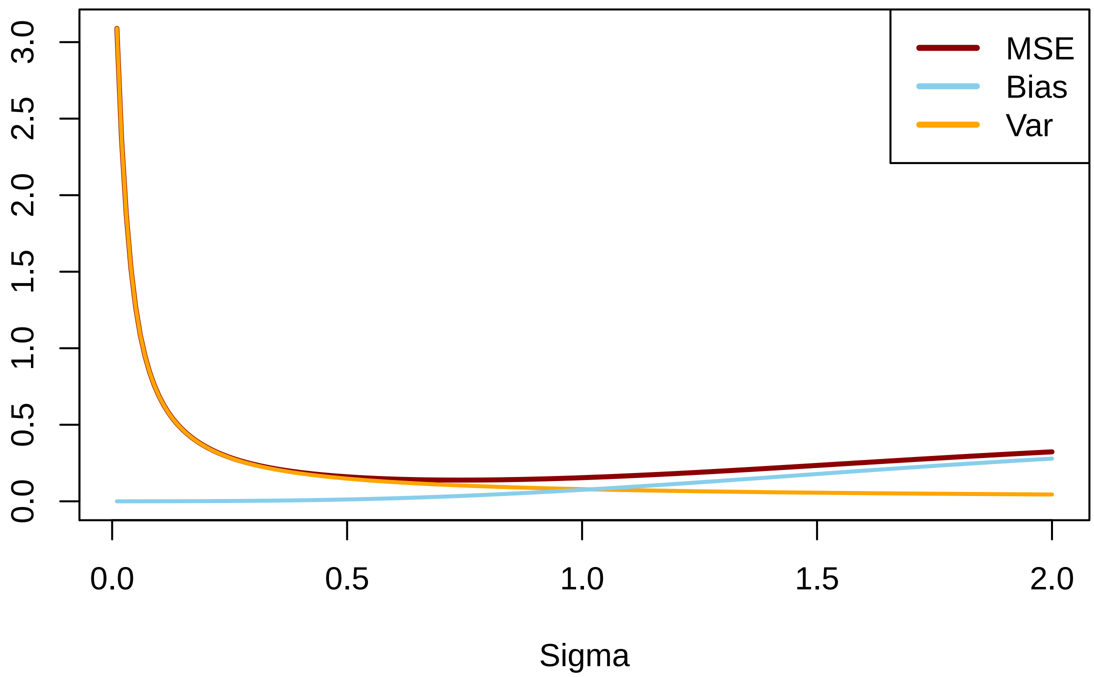
Part 2

Proof of question (a) is attached on the last page.

b

```
# return matrix of W
W_cal <- function(training, sigma_val) {
  dt.size <- length(training)

  W.result <- matrix(, nrow=dt.size, ncol=dt.size)

  for (i in 1:dt.size) {
    for (j in 1:dt.size) {
      W.result[i, j] <- dnorm(training[j], training[i], sigma_val)
    }
    W.result[i, ] <- W.result[i, ] / sum(W.result[i, ])
  }

  return(W.result)
}
```

```
gk.sigma <- seq(from=0.01, to=2, by=0.01)

dt.size=length(x.train)

gk.result <- matrix(nrow=length(gk.sigma), ncol=3)

for (i in 1:length(gk.sigma)) {
  W <- W_cal(x.train, gk.sigma[i])

  gk.bias.sq <- sum(((W - diag(dt.size)) %*% f)^2) / dt.size
  gk.var <- noise.var * sum(diag(t(W) %*% W)) / dt.size
  gk.sum <- gk.bias.sq + gk.var

  gk.result[i, 1] <- gk.bias.sq
  gk.result[i, 2] <- gk.var
  gk.result[i, 3] <- gk.sum
}

plot(gk.sigma, gk.result[,3], ylim=c(min(gk.result), max(gk.result)), xlab='Sigma', ylab='',
     type='l', col='red4', lwd=2.5)
lines(gk.sigma, gk.result[,2], col='orange', lwd=2, lty=1)
lines(gk.sigma, gk.result[,1], col='skyblue', lwd=2, lty=1)
legend('topright', legend=c('MSE', 'Bias', 'Var'), col=c('red4', 'skyblue', 'orange'), lwd=3)
```



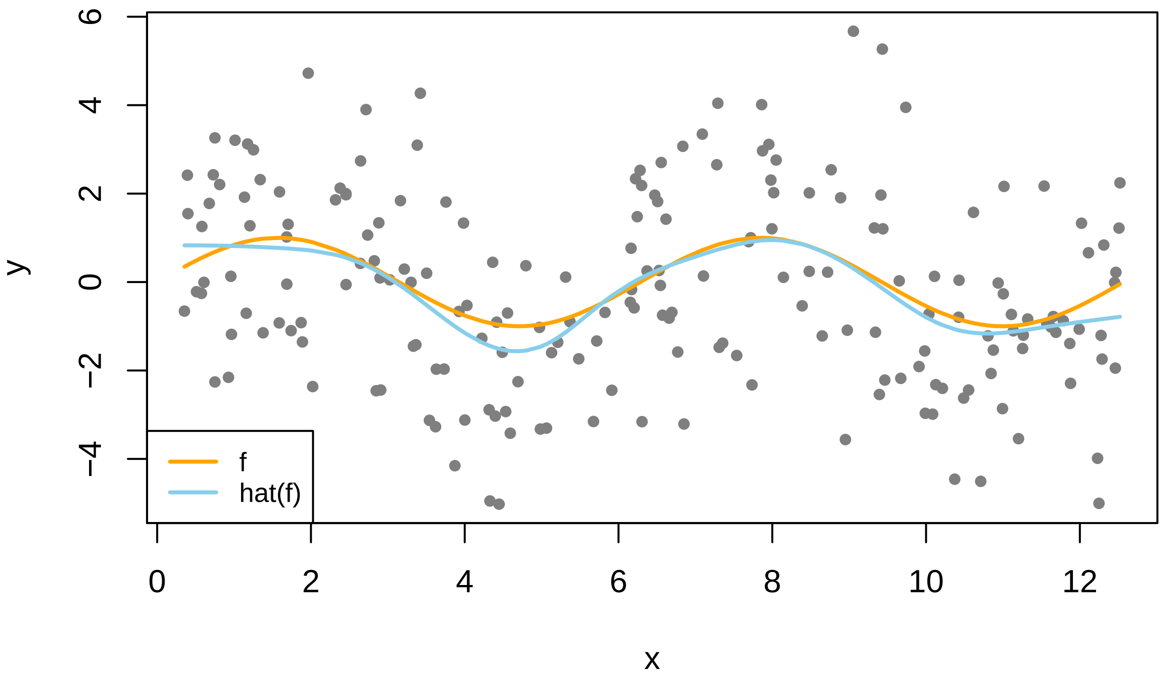

```
# sigma for smallest MSE
opt.sigma <- gk.sigma[which(gk.result[,3] == min(gk.result[,3]))]
print(opt.sigma)
```

```
## [1] 0.74
```

Sigma = 0.74 minimizes expected squared error.

c

```r
opt.fhat <- W_cal(x.train, opt.sigma) %*% y.train

plot(x.train, y.train, col='gray50', pch=16, cex=0.8, xlab='x', ylab='y')
lines(x.train, f, col='orange', lwd=2, lty=1)
lines(x.train, opt.fhat, col='skyblue', lwd=2, lty=1)
legend('bottomleft', legend=c('f', 'hat(f)'), col=c('orange', 'skyblue'), lwd=2, cex=0.8)
```



part2 - a

$$
\begin{aligned}
D &= E(\frac{1}{n}\|f-\hat{f}\|^2) \\
&= E(\frac{1}{n}\|f-E(\hat{f})-\hat{f}+E(\hat{f})\|^2) \\
&= E(\frac{1}{n}\|f-E(\hat{f})-\hat{f}+E(\hat{f})\|^2) \\
&= \frac{1}{n}[E(\|f-E(\hat{f})\|^2)+E(\|\hat{f}-E(\hat{f})\|^2)+2E(E(\|f-E(\hat{f})\|)E(\|\hat{f}-E(\hat{f})\|)]
\end{aligned}
$$

Note that $E(E(\|\hat{f}-E(\hat{f})\|))=0$, two terms are independent to each other, therefore the cross term $2E(E(\|f-E(\hat{f})\|)E(\|\hat{f}-E(\hat{f})\|)=0$.

$$D=\frac{1}{n}[E(\|f-E(\hat{f})\|^2)+E(\|\hat{f}-E(\hat{f})\|^2)]$$

Separately write two terms.

Since $\hat{f}=Wy$ and $y=f+\epsilon$, where y, f, ϵ are vectors, $E(\hat{f})=E(Wy)=WE(f+\epsilon)=Wf$

$$
\begin{aligned}
E(\|f-E(\hat{f})\|^2) &= E(\|f-Wf\|^2) \\
&= E(\|(W-I)f\|^2)
\end{aligned}
$$

Since W depends only on bandwidth, and f is parameter, this term can be written as $\|(W-I)f\|^2$

$E(\|\hat{f}-E(\hat{f})\|^2)$ is the variance of $\|\hat{f}\|$, where $\hat{f}=Wy$.

$$
\begin{aligned}
E(\|\hat{f}-E(\hat{f})\|^2) &= Var(\|W\|y) \\
&= \|W^2\|Var(y) \\
&= trace(W^TW)\sigma^2
\end{aligned}
$$

Combine these two term, we proved:

$$D=\frac{1}{n}(\|(W-I)f\|^2+trace(W^TW)\sigma^2)$$