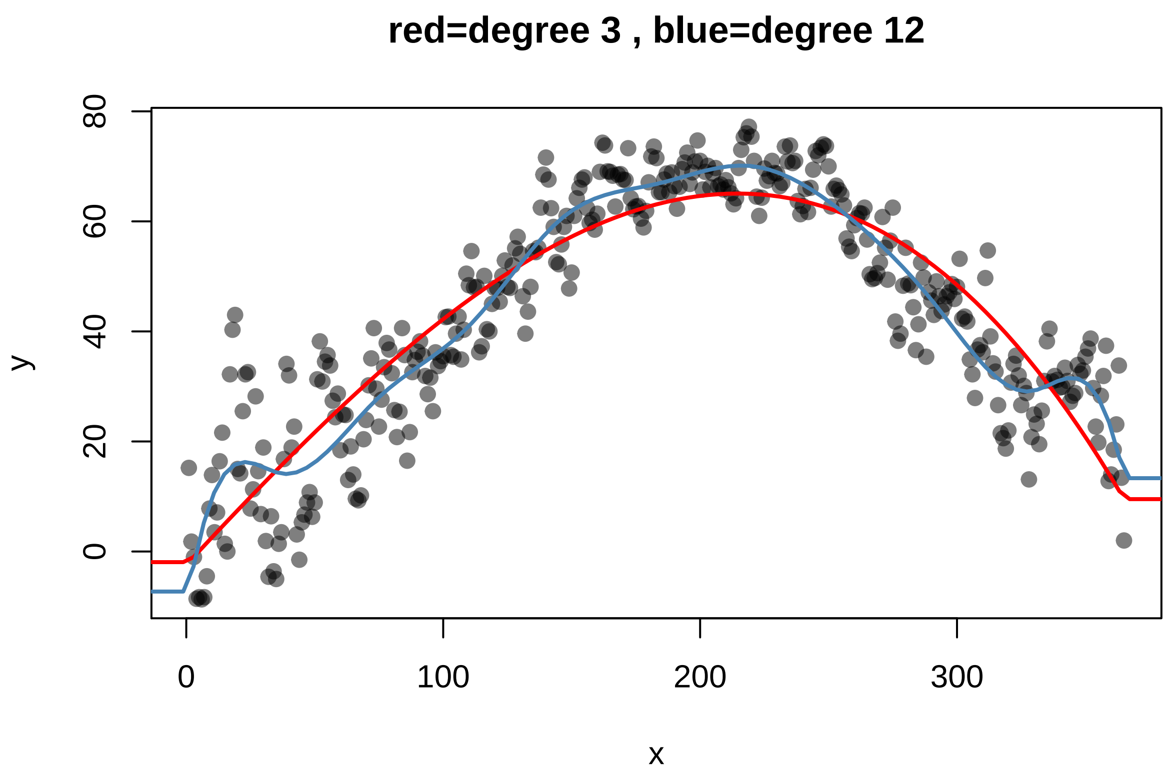
A6Q1

a)

```
ot1996 <- read.csv('ottawaTemp1996.csv', header=T)
temp <- data.frame(x = 1:365, y = ot1996$Temp)

muhat3  <- getmuhat(temp, 3)
muhat12 <- getmuhat(temp, 12)

xlim <- extendrange(temp[,1])

plot(temp,
     pch=19, col= adjustcolor("black", 0.5))
curve(muhat3, from = xlim[1], to = xlim[2],
      add = TRUE, col="red", lwd=2)
curve(muhat12, from = xlim[1], to = xlim[2],
      add = TRUE, col="steelblue", lwd=2)
title(main="red=degree 3 , blue=degree 12")
```



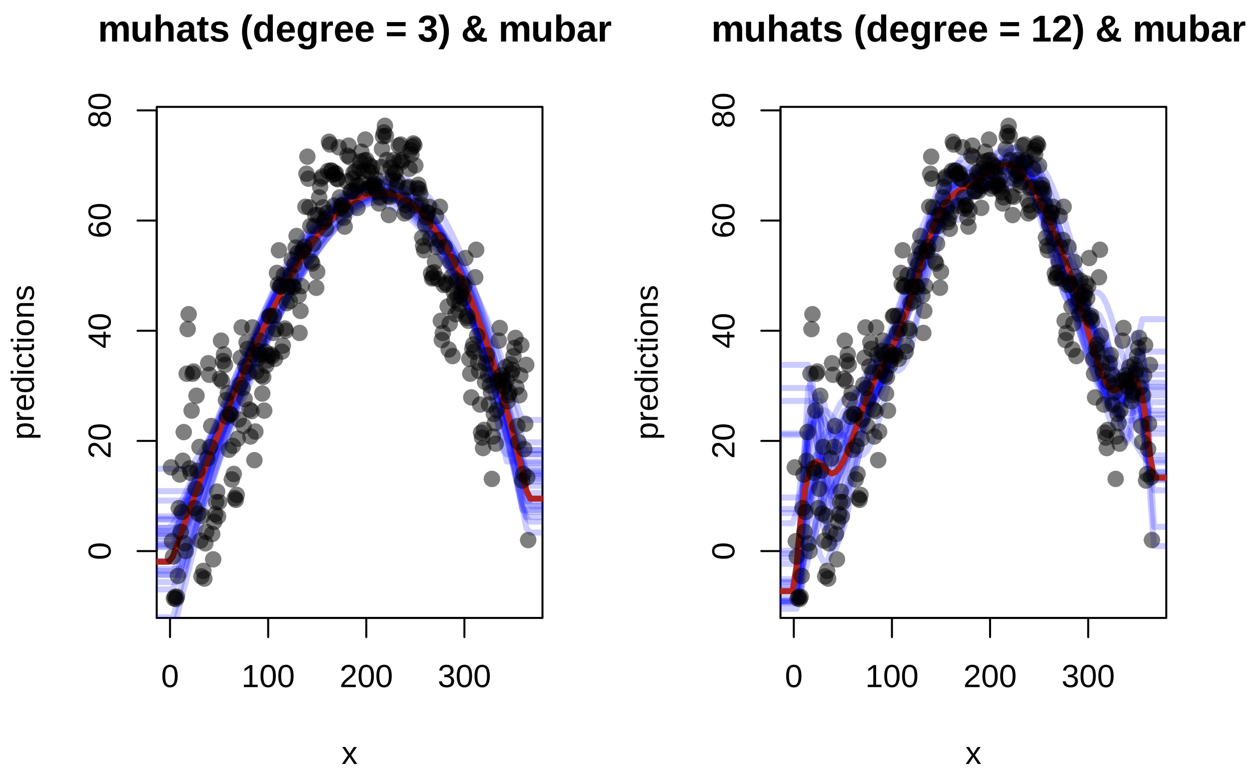
b)

```r
N_S <- 25
set.seed(341)  # for reproducibility

n= 50
samps    <- lapply(1:N_S, FUN= function(i){getSampleComp(temp, n)})
Ssamples <- lapply(samps, FUN= function(Si){getXYSample("x", "y", Si, temp)})
Tsamples <- lapply(samps, FUN= function(Si){getXYSample("x", "y", !Si, temp)})

muhats3 <- lapply(Ssamples, getmuhat, complexity = 3)

muhats12 <- lapply(Ssamples, getmuhat, complexity = 12)
```

c)

```r
par(mfrow=c(1,2))

xvals <- seq(xlim[1], xlim[2], length.out = 200)
plot(temp,
     pch=19, type='n',
     xlab="x", ylab="predictions",
     main= " muhats (degree = 3) & mubar")


for (i in 1:N_S) {
  curveFn <- muhats3[[i]]
  curve(curveFn, from = xlim[1], to = xlim[2], add=TRUE,
        col=adjustcolor("blue", 0.2), lwd=3, lty=(1))
}

curve(muhat3,  from = xlim[1], to = xlim[2],
      add=TRUE, col="firebrick", lwd=3)

points(temp,
     pch=19, col= adjustcolor("black", 0.5))


plot(temp,
     pch=19, type='n',
     xlab="x", ylab="predictions",
     main= " muhats (degree = 12) & mubar")

for (i in 1:N_S) {
  curveFn <- muhats12[[i]]
  curve(curveFn, xlim[1], xlim[2], add=TRUE,
        col=adjustcolor("blue", 0.2), lwd=3, lty=1)
}

curve(muhat12, xlim[1], xlim[2], add=TRUE, col="firebrick", lwd=3)
```

```r
points(temp,
     pch=19, col= adjustcolor("black", 0.5))
```



d)

```r
var_mutilde(Ssamples, Tsamples, complexity=3)
```

```
## [1] 6.724687
```

```r
var_mutilde(Ssamples, Tsamples, complexity=12)
```

```
## [1] 20.5846
```

e)

```r
muhat = getmuFun(temp, "x", 'y')
bias2_mutilde(Ssamples, Tsamples, muhat, complexity=3)
```

```
## [1] 84.02313
```

```r
bias2_mutilde(Ssamples, Tsamples, muhat, complexity=12)
```

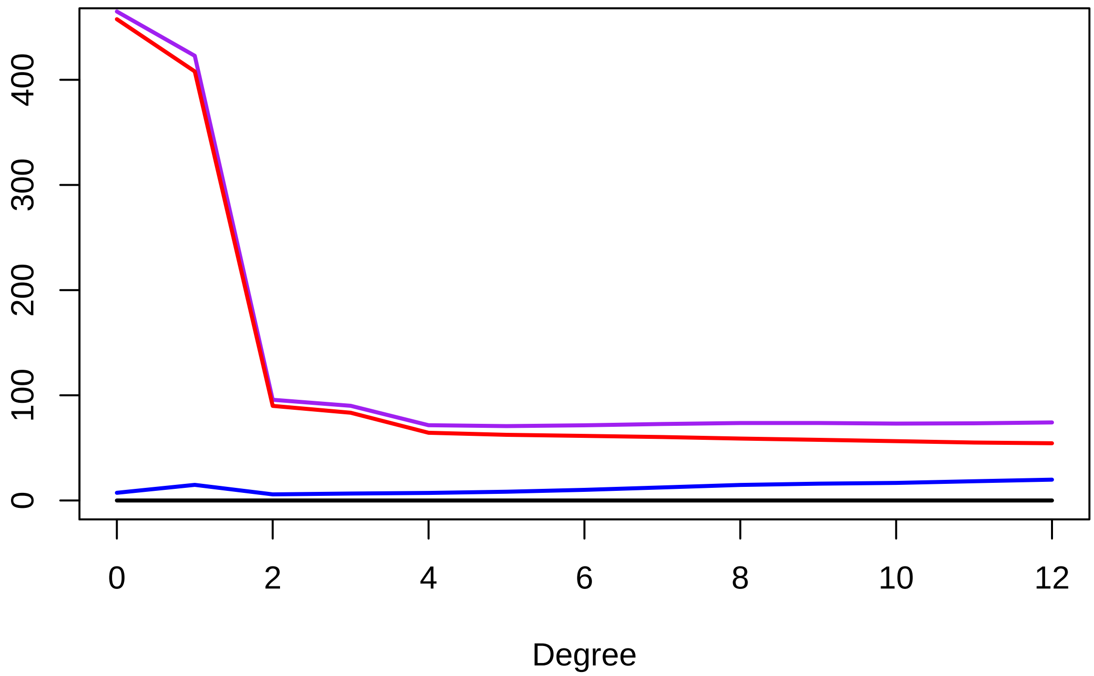
```
## [1] 54.80545
```

f)

```r
complexities <- 0:12
muhat = getmuFun(temp, "x", 'y')


apse_vals <-  sapply(complexities,
                     FUN = function(complexity){
                       apse_all(Ssamples, Tsamples, complexity = complexity,
                                muhat)
                     }
)

round( t(rbind(complexities, apse=round(apse_vals,5))),1)
```

```
##       complexities  apse var_mutilde bias2 var_y
##  [1,]            0 464.9         7.3 457.6     0
##  [2,]            1 422.8        14.8 408.0     0
##  [3,]            2  95.7         5.8  89.9     0
##  [4,]            3  90.0         6.6  83.4     0
##  [5,]            4  71.5         7.2  64.4     0
##  [6,]            5  70.7         8.3  62.4     0
##  [7,]            6  71.5        10.1  61.4     0
##  [8,]            7  72.7        12.4  60.3     0
##  [9,]            8  73.6        14.8  58.9     0
## [10,]            9  73.7        16.0  57.7     0
## [11,]           10  73.1        16.7  56.4     0
## [12,]           11  73.4        18.3  55.1     0
## [13,]           12  74.2        19.8  54.4     0
```

```r
plot( complexities, apse_vals[1,], xlab="Degree", ylab="", type='l',
      ylim=c(0, 450), col="purple", lwd=2 )
lines(complexities, apse_vals[2,], col="blue", lwd=2 )
lines(complexities, apse_vals[3,], col="red", lwd=2)
lines(complexities, apse_vals[4,], col="black", lwd=2)
```



Conclusion: The polynomial with degree 5 has the lowest APSE. The bias starts off really high and continue to decrease from 1 to 12.

g)

i)

```
sample.kfold <- function(k=NULL, pop=NULL, xvarname=NULL, yvarname=NULL ) {

  N = nrow(pop)
  kset = rep_len(1:k, N)
  kset = sample(kset)

  samps = list()
  for (i in 1:k) {
    samps[[i]] = logical(N)
    samps[[i]][kset != i] = TRUE
  }

set.seed(341)
Ssamples <- lapply(samps,
                   FUN= function(Si){getXYSample(xvarname, yvarname, Si, pop)})

Tsamples <- lapply(samps,
                   FUN= function(Si){getXYSample(xvarname, yvarname, !Si, pop)})

    list(Ssamples=Ssamples,Tsamples=Tsamples)
}
```

ii)

```
temp.muFun = getmuFun(temp, "x", 'y')
kfold.samples = sample.kfold(k=10, pop=temp, "x", "y")
apse_all(kfold.samples$Ssamples, kfold.samples$Tsamples, complexity = 3,
         temp.muFun)
```

```
##        apse var_mutilde       bias2       var_y
##  85.3552922   0.1074539  85.2478384   0.0000000
```

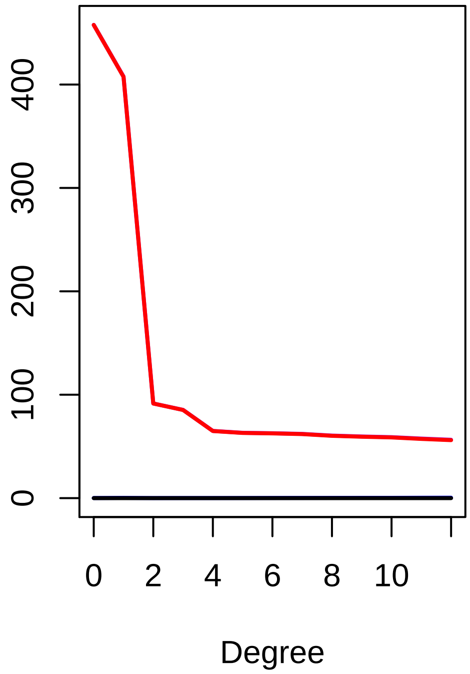
iii)

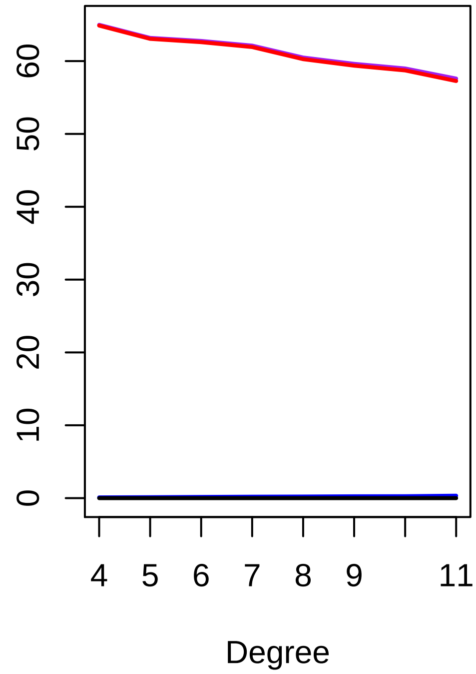
```
complexities <- 0:12


apse_vals <- sapply(complexities,
      FUN = function(complexity){
          apse_all(kfold.samples$Ssamples, kfold.samples$Tsamples,
              complexity = complexity, temp.muFun) })

# Print out the results
t(rbind(complexities, apse=round(apse_vals,5)))
```

```
##       complexities      apse var_mutilde     bias2 var_y
##  [1,]            0 457.72379     0.10453 457.61925     0
##  [2,]            1 407.75998     0.24421 407.51577     0
##  [3,]            2  91.54928     0.06873  91.48055     0
##  [4,]            3  85.35529     0.10745  85.24784     0
##  [5,]            4  64.97441     0.09901  64.87540     0
##  [6,]            5  63.18109     0.12512  63.05597     0
##  [7,]            6  62.76258     0.16067  62.60190     0
##  [8,]            7  62.12142     0.19203  61.92939     0
##  [9,]            8  60.48264     0.21393  60.26871     0
## [10,]            9  59.62014     0.24802  59.37212     0
## [11,]           10  58.98205     0.26032  58.72173     0
## [12,]           11  57.59472     0.32863  57.26609     0
## [13,]           12  56.45077     0.37286  56.07791     0
```

```
par(mfrow=c(1,2))
plot( complexities, apse_vals[1,], xlab="Degree", ylab="", type='l',
      ylim=c(0, max(apse_vals) ), col="purple", lwd=2 )
lines(complexities, apse_vals[2,], col="blue", lwd=2 )
lines(complexities, apse_vals[3,], col="red", lwd=2)
lines(complexities, apse_vals[4,], col="black", lwd=2)

# The increase in apse is too sharp in higher complexities. Let's zoom in a bit
zoom = 5:12
plot( complexities[zoom], apse_vals[1, zoom], xlab="Degree", ylab="",
      type='l', ylim=c(0, max(apse_vals[,zoom])  ), col="purple", lwd=2 )
lines(complexities[zoom], apse_vals[2, zoom], col="blue", lwd=2 )
```

```
lines(complexities[zoom], apse_vals[3, zoom], col="red", lwd=2)
lines(complexities[zoom], apse_vals[4, zoom], col="black", lwd=2)
```





Conclusion: The polynomial with degree 12 has the lowest APSE. We can pick degree 12.

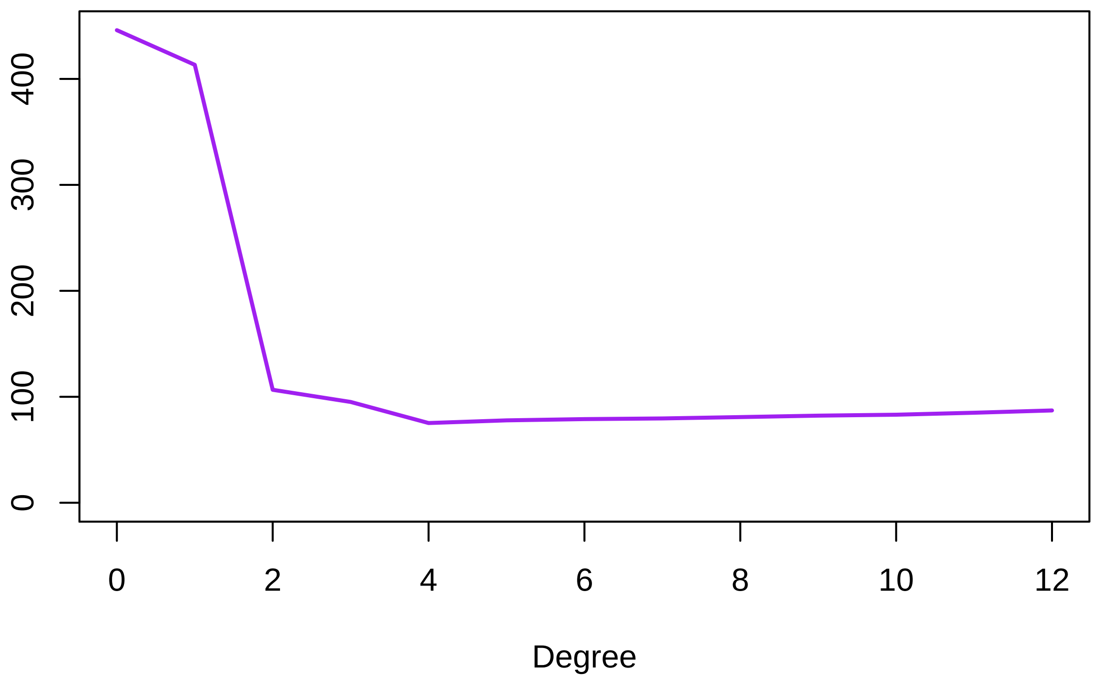
h)

```
other <- read.csv('ottawaOtherYears.csv', header=T)
t <- data.frame(x = rep_len(1:365, nrow(other)), y = other$Temp)

apse <- function(y, x, predfun){
  mean((y - predfun(x))^2, na.rm = TRUE)
}

complexities <- 0:12


apse_vals <-  sapply(complexities,
                     FUN = function(complexity){
                       apse(t$y, t$x, getmuhat(temp, complexity))
                    }
)

round( t(rbind(complexities, apse=round(apse_vals,5))),1)
```

```
##       complexities  apse
##  [1,]            0 446.0
##  [2,]            1 413.3
##  [3,]            2 106.6
```

```
## [4,]            3  95.1
## [5,]            4  75.2
## [6,]            5  77.8
## [7,]            6  78.9
## [8,]            7  79.6
## [9,]            8  80.8
## [10,]           9  82.2
## [11,]          10  83.1
## [12,]          11  85.0
## [13,]          12  87.1
```

```
plot( complexities, apse_vals, xlab="Degree", ylab="", type='l',
      ylim=c(0, max(apse_vals) ), col="purple", lwd=2 )
```



Conclusion: The polynomial with degree 4 has the lowest APSE. Comparing to f) and g), the complexity is the lowest. As we want to use the function that is the least complex, we prefer to use degree 4.