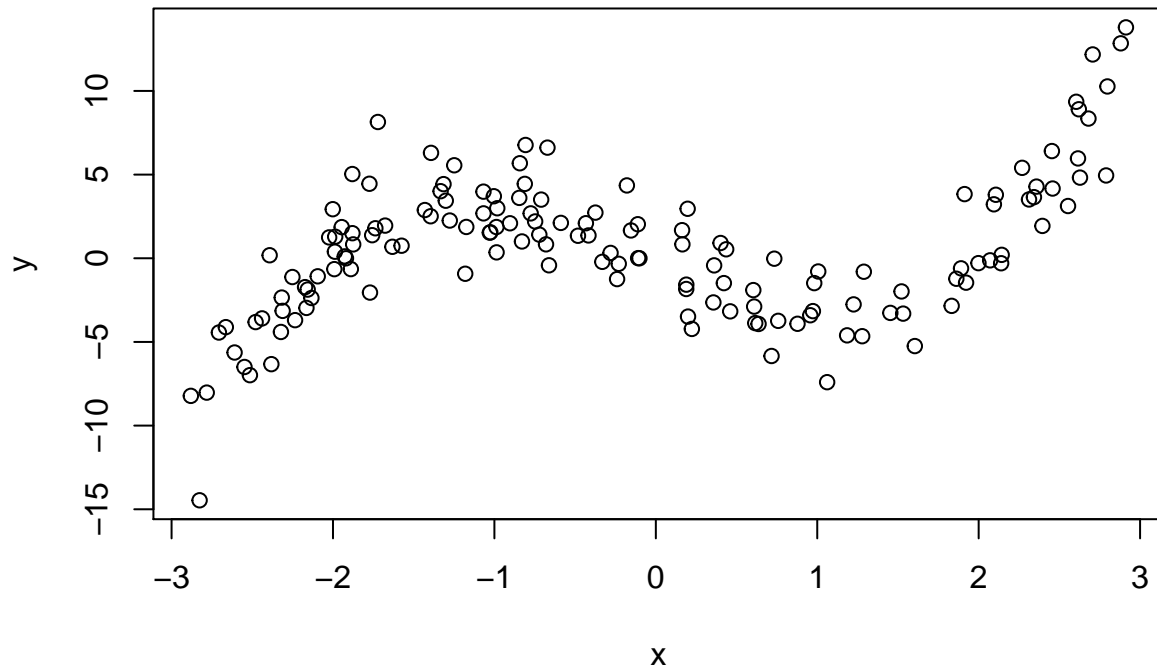


2.(a) Advantage of K-fold cross validation: Since we are interested in the actual prediction error, not the prediction error of a certain set. By using K-fold cross validation the average of these errors estimates the expected prediction error. Using validation set approach may not result a good representation of the prediction error (variance may be large). Disadvantage of K-fold cross validation: The choice of k . Large k results an approximately unbiased estimator. But it also results high variance. Similarly, small k results low variance but high biasness.

2.(b)

```
set.seed(444)
x = runif(150, -3, 3)
y = (x+2)*(x)*(x-2) + rnorm(150, sd=2)
plot(x, y)
```

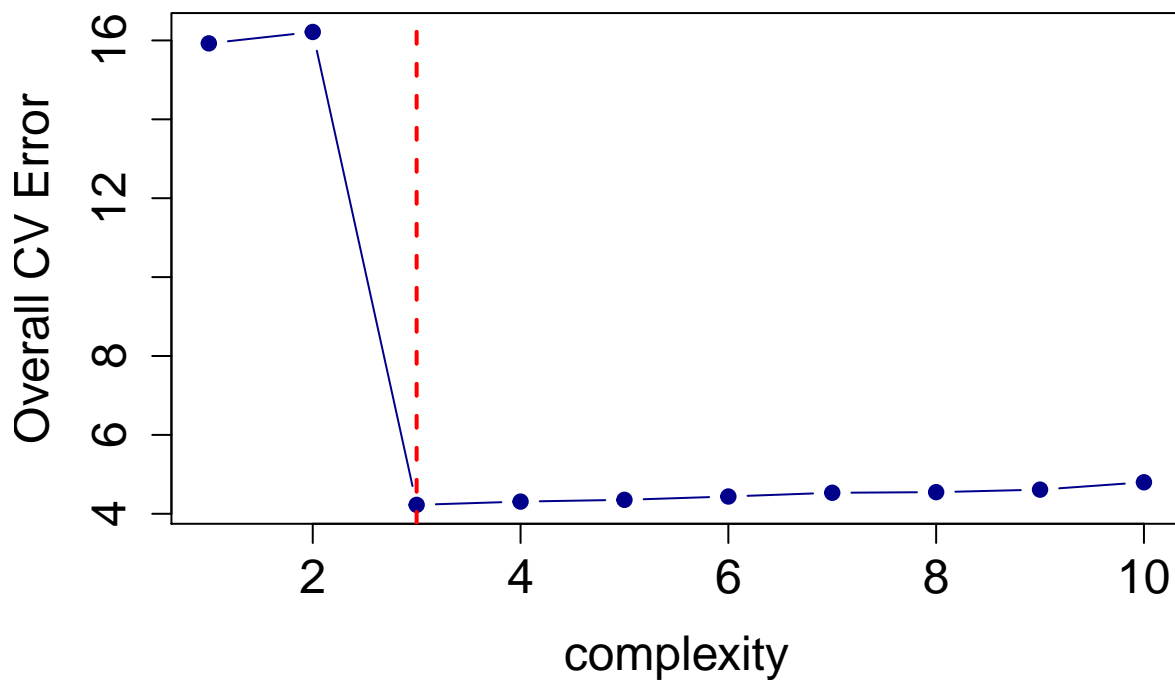


$$Y|X = X(X+2)(X-2) + \epsilon$$

2.(c)

```
library(boot)
complexity = c(1:10)
cv.err = vector("numeric", length(complexity))
data = data.frame('x' = x, 'y' = y)
set.seed(444)
for(i in 1:length(complexity)){
  glm.fit = glm(y ~ poly(x, complexity[i]))
  cv.err[i] = cv.glm(data, glm.fit)$delta[1]
}

plot(complexity, cv.err, pch=19, col="darkblue", type="b",
      cex.axis = 1.5, cex.lab=1.5, ylab="Overall CV Error")
indx = which.min(cv.err)
abline(v=indx, lty=2, lwd=2, col='red')
```



the LOOCV errors are

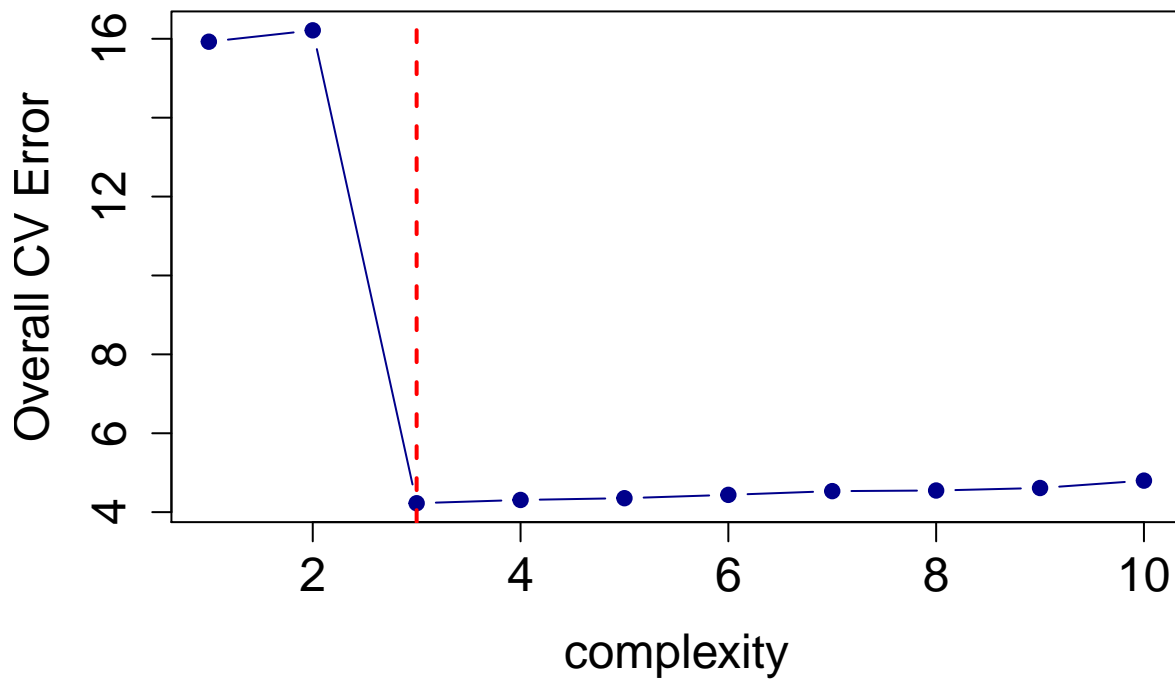
```
cv.err
## [1] 15.925694 16.214299 4.226020 4.308672 4.353616 4.437838 4.532520
## [8] 4.548312 4.611843 4.798791
```

We choose the model with complexity 3.

2.(d)

```
data = data.frame('x' = x, 'y' = y)
set.seed(844)
for(i in 1:length(complexity)){
  glm.fit = glm(y ~ poly(x, complexity[i]))
  cv.err[i] = cv.glm(data, glm.fit)$delta[1]
}

plot(complexity, cv.err, pch=19, col="darkblue", type="b",
      cex.axis = 1.5, cex.lab=1.5, ylab="Overall CV Error")
indx = which.min(cv.err)
abline(v=indx, lty=2, lwd=2, col='red')
```



the LOOCV errors are

```
cv.err
## [1] 15.925694 16.214299 4.226020 4.308672 4.353616 4.437838 4.532520
## [8] 4.548312 4.611843 4.798791
```

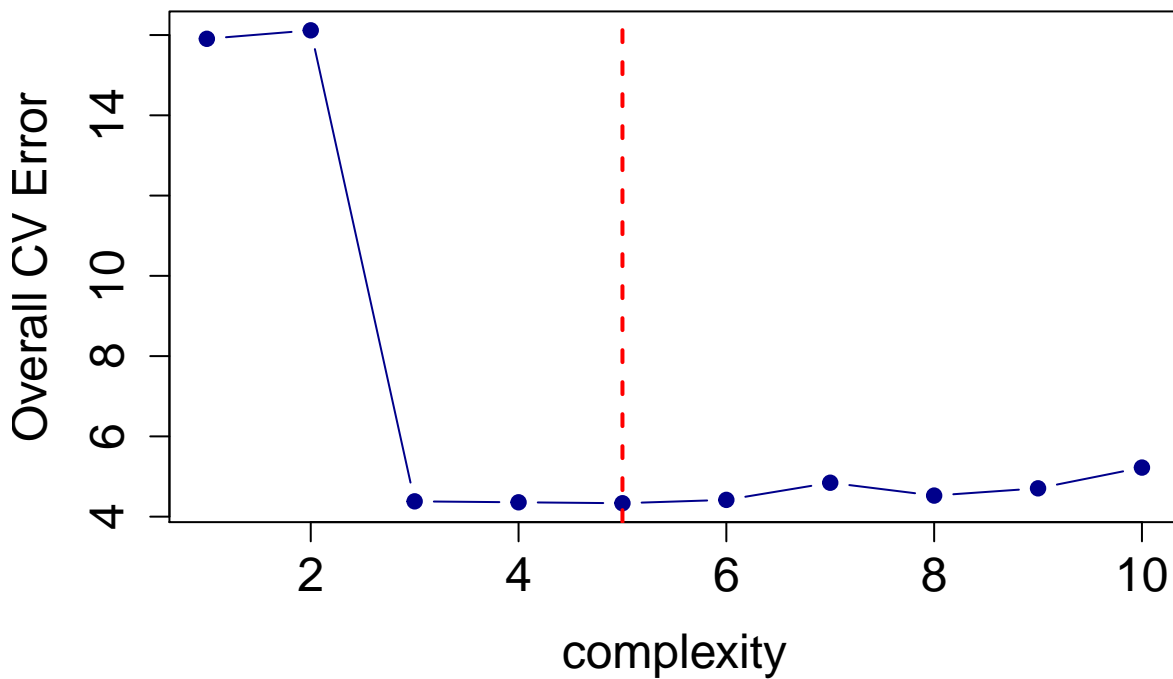
We choose the model with complexity 3.

There is no difference and that is what we would expect. Since we are performing LOOCV, every point is in exactly one test set and every set has size 1. So it does not change the result when we change the seed.

2.(e) 10-fold CV with seed number 444:

```
data = data.frame('x' = x, 'y' = y)
set.seed(444)
for(i in 1:length(complexity)){
  glm.fit = glm(y ~ poly(x, complexity[i]))
  cv.err[i] = cv.glm(data, glm.fit, K = 10)$delta[1]
}

plot(complexity, cv.err, pch=19, col="darkblue", type="b",
      cex.axis = 1.5, cex.lab=1.5, ylab="Overall CV Error")
indx = which.min(cv.err)
abline(v=indx, lty=2, lwd=2, col='red')
```



cv.err

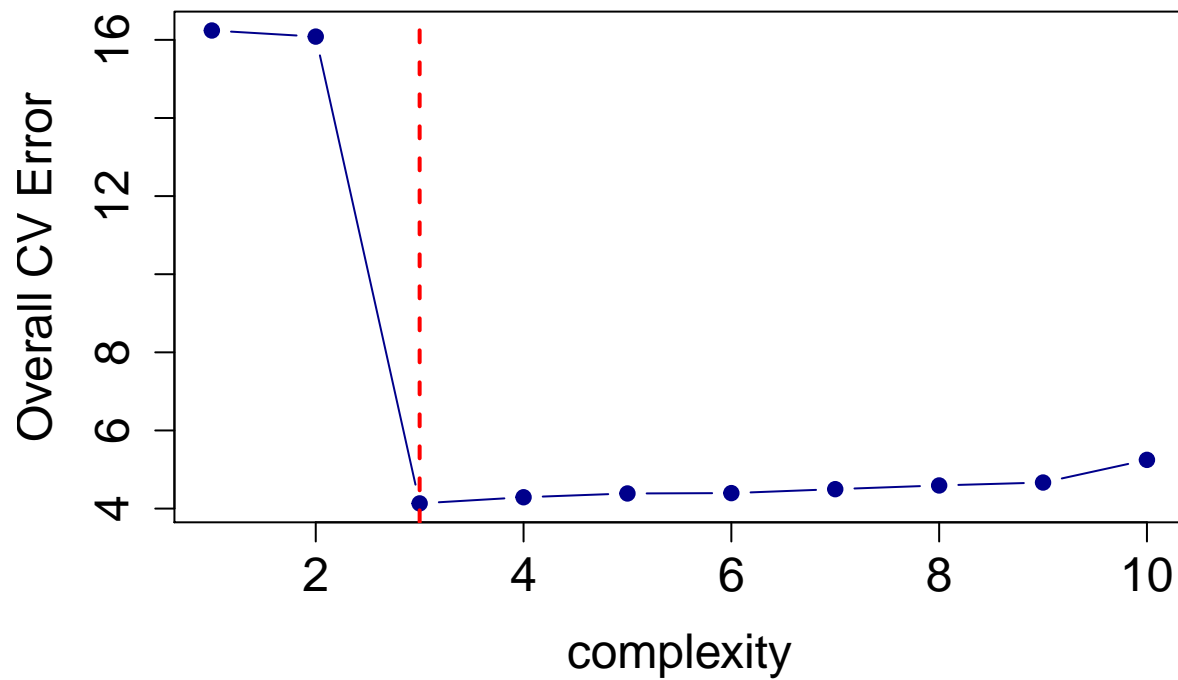
```
## [1] 15.906805 16.117685 4.381324 4.357353 4.334346 4.418009 4.845047
## [8] 4.524723 4.705431 5.222740
```

we choose the model with complexity 5.

10-fold CV with seed number 844:

```
data = data.frame('x' = x, 'y' = y)
set.seed(844)
for(i in 1:length(complexity)){
  glm.fit = glm(y ~ poly(x, complexity[i]))
  cv.err[i] = cv.glm(data, glm.fit, K = 10)$delta[1]
}
```

```
plot(complexity, cv.err, pch=19, col="darkblue", type="b",
     cex.axis = 1.5, cex.lab=1.5, ylab="Overall CV Error")
indx = which.min(cv.err)
abline(v=indx, lty=2, lwd=2, col='red')
```



```
cv.err
```

```
## [1] 16.239569 16.083618 4.132356 4.289846 4.386585 4.395705 4.496957
## [8] 4.591881 4.666508 5.247140
```

we choose the model with complexity 3.

changing the seed changes the cv error and that is what we would expect. Since we are performing a 10-fold CV. The random seed affects the elements in test and train sets. So different seed will result a slightly different errors.

2.(f) All models choose the polynomial with degree 3 except 10-fold CV with seed 444 which agree with the actual model $f(x) = x(x+2)(x-2)$. But if we plot the polynomial with degree 5 and 3, there is not much difference between two of them and the cv error of degree 3 and 5 are reasonable close.

```
xorder = order(x)
plot(x, y)
lines(x[xorder], predict(lm(y ~ poly(x, 5)))[xorder], type="l", col="blue", lwd=2)
lines(x[xorder], predict(lm(y ~ poly(x, 3)))[xorder], type="l", col="red", lwd=2)
```

