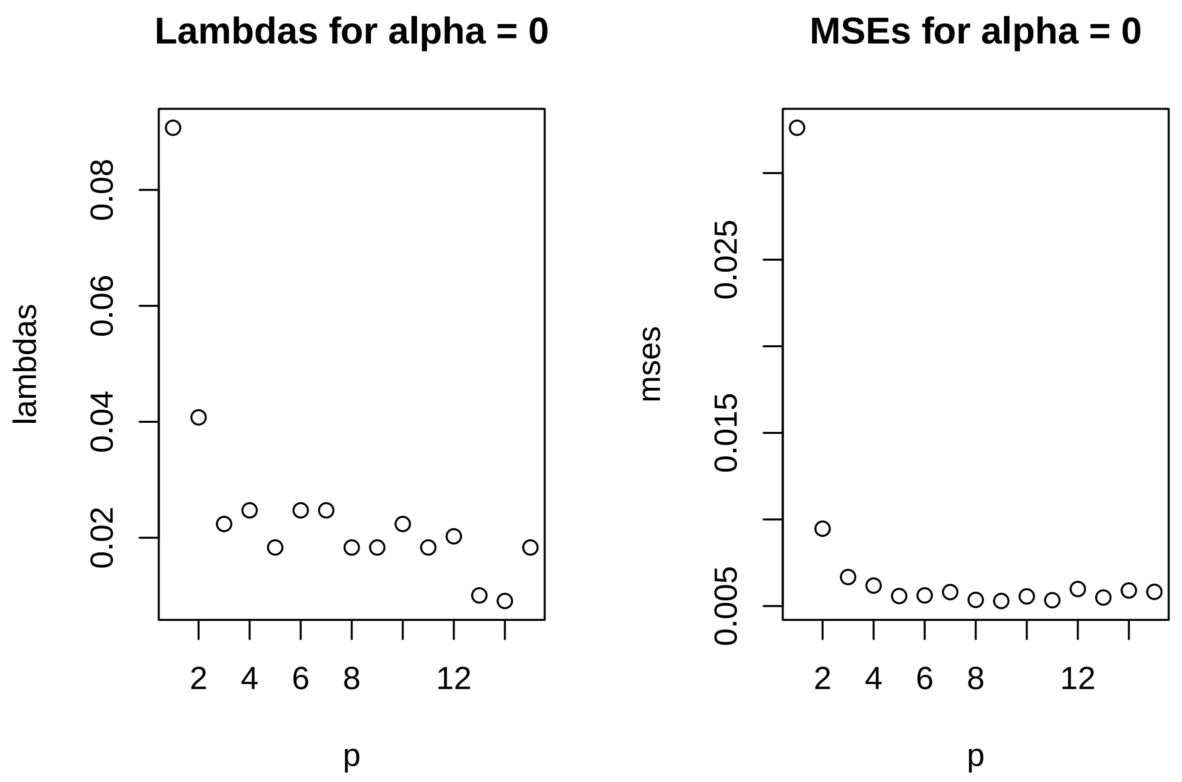
Q2)

Part a)

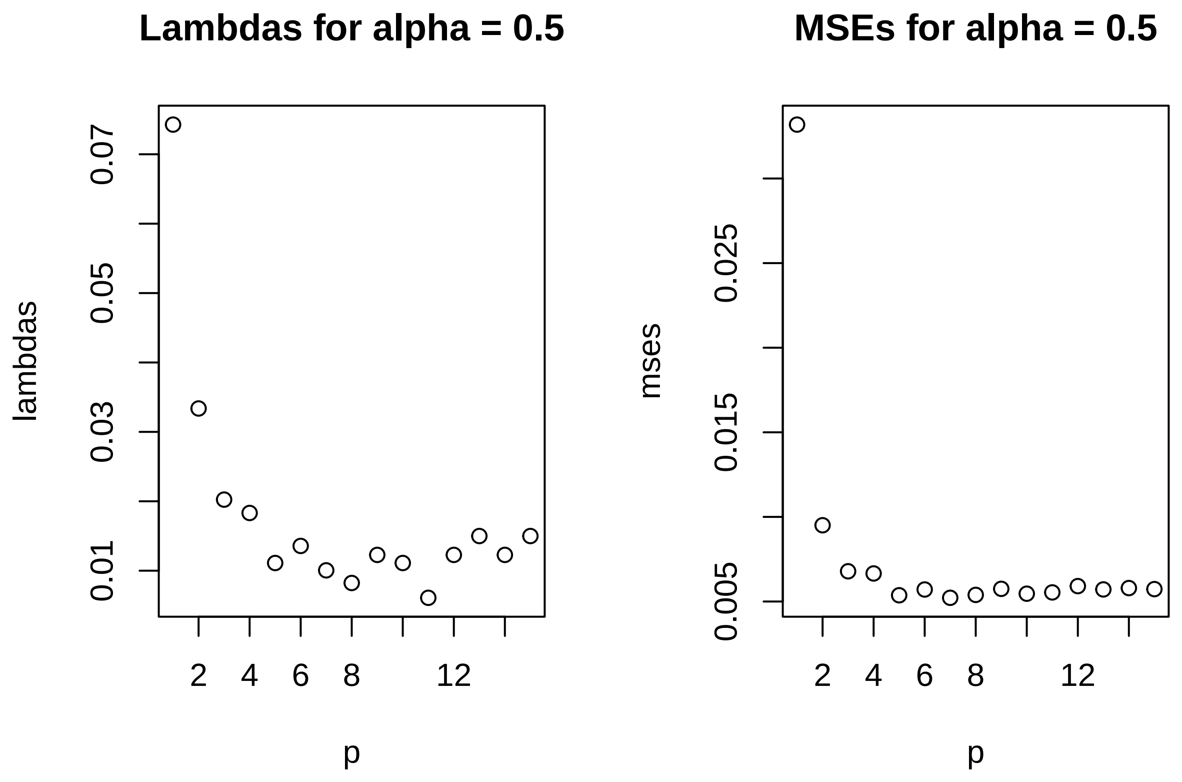
```
library(glmnet())
```

```
## Loading required package: Matrix

## Loaded glmnet 4.1-8
```

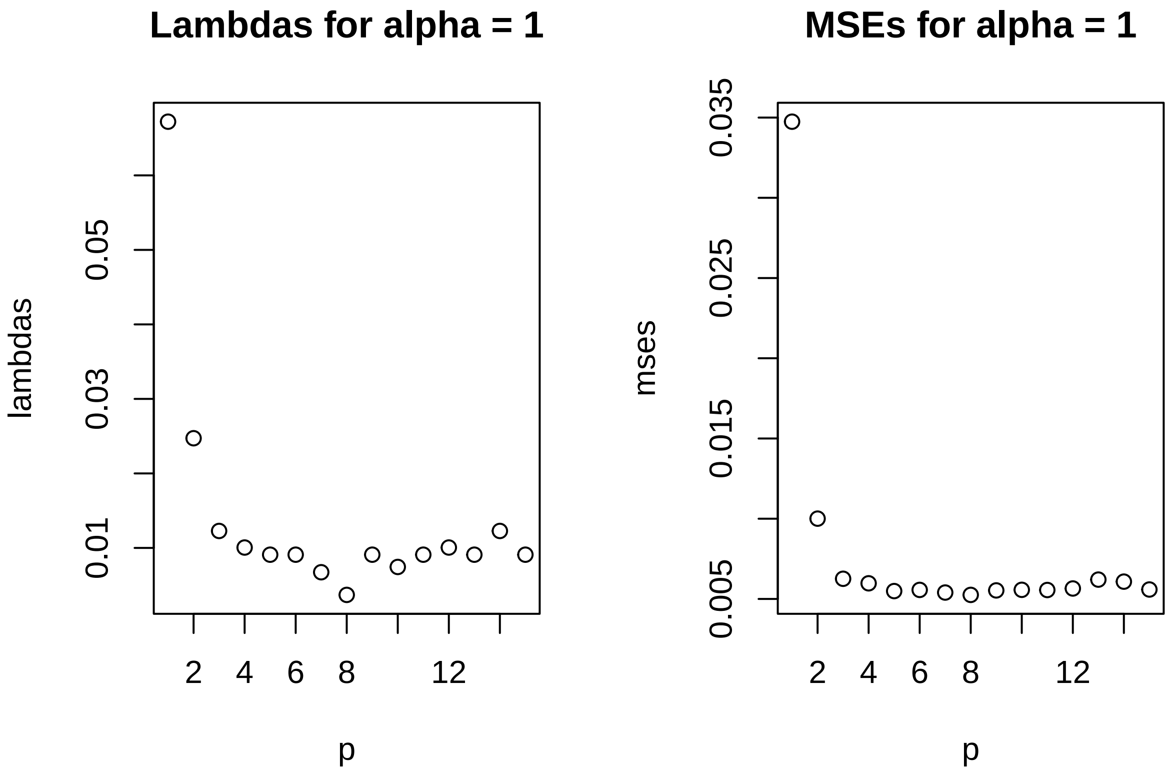
```
set.seed(42)
data <- read.csv("Quality.csv")
Y <- ts(data)
t <- as.vector(time(Y))
poly.Time = poly(t, 15)

X <- model.matrix(~ poly.Time) # 180x15
y <- as.vector(Y) # 180x1


Time.test = c(5, 14, 29, 30, 36, 39, 66, 71, 79, 84, 85, 96, 109,
              112, 135, 136, 139, 146, 156, 171)
Time.train = t[-Time.test]

X.train <- X[Time.train, 2:16]
X.test <- X[Time.test, 2:16]

y.train <-y[Time.train]
y.test <- y[Time.test]

Log.Lambda.Seq = seq(-7, 3, by = 0.1)
Lambda.Seq = c(0, exp(Log.Lambda.Seq))

for (ALPHA in c(0, 0.5, 1)) {
  lambdas <- c()
  mses <- c()
  for (p in 1:15){
    # 1) Fit model to training data
    Xmatrix <- X.train[, 1:p]
    if (p == 1){
      Xmatrix <- cbind(0, Xmatrix)
    }

    CV = cv.glmnet(Xmatrix, y.train, type.measure="mse", lambda= Lambda.Seq,  alpha=ALPHA, family="gaus
    lambdas <- c(lambdas, CV$lambda.1se)
    mses <- c(mses, CV$cvm[CV$index[2]])
  }
  p <- c(1:15)
  par(mfrow=c(1, 2))
  plot(x=p, lambdas, main = sprintf("Lambdas for alpha = %s", ALPHA))
  plot(x=p, mses, main = sprintf("MSEs for alpha = %s", ALPHA))
  print("Chosen p = ")
  print(which.min(mses))
}
```


Lambdas for alpha = 0
lambdas
0.02
0.04
0.06
0.08
2
4
6
8
12
p
MSEs for alpha = 0
mses
0.005
0.015
0.025
2
4
6
8
12
p

```
## [1] "Chosen p = "
## [1] 9
```


Lambdas for alpha = 0.5
lambdas
0.01
0.03
0.05
0.07
2
4
6
8
12
p
MSEs for alpha = 0.5
mses
0.005
0.015
0.025
2
4
6
8
12
p

```
## [1] "Chosen p = "
## [1] 7
```



```
## [1] "Chosen p = "
## [1] 8
```

We use the MSE plots to choose the polynomial degree that minimizes the MSE of the regularized model.

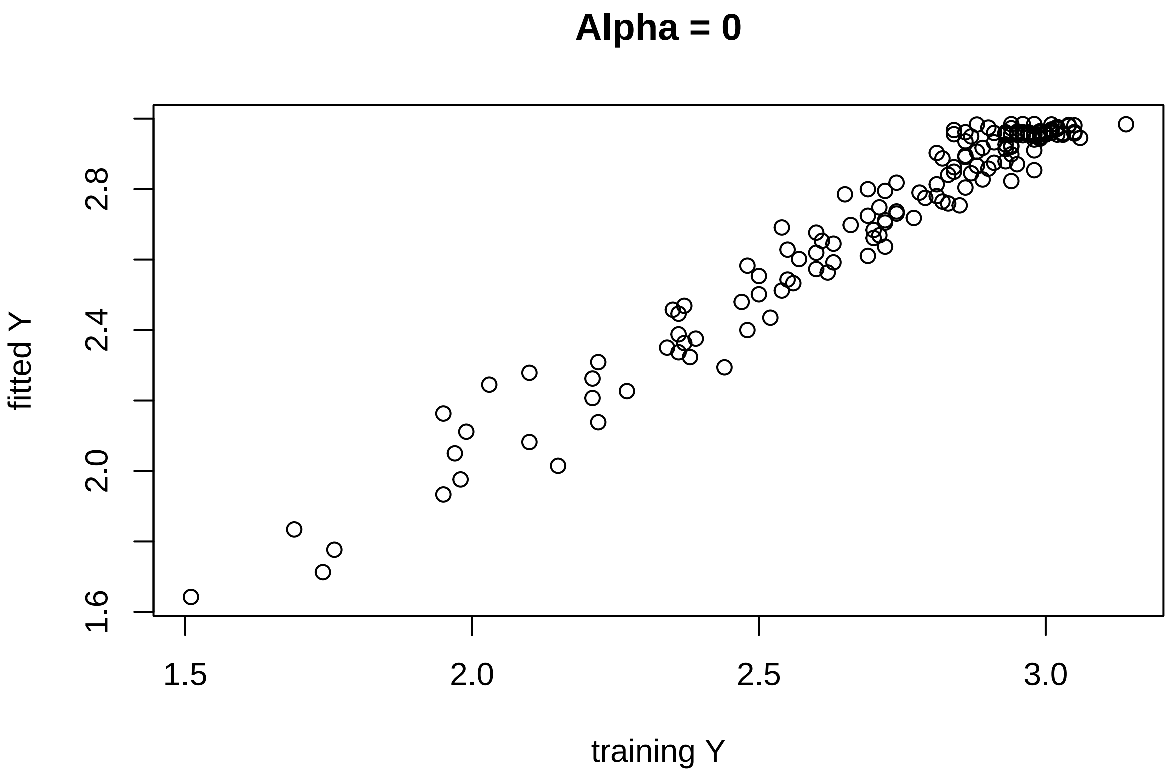
- For ridge ($\alpha = 0$) choose $p = 9$.
- For lasso ($\alpha = 1$) choose $p = 8$
- For elastic net ($\alpha = 0.5$) choose $p = 7$

Do you think that simpler models compared to what you have chosen here may have similar performances? Comment on your observation.

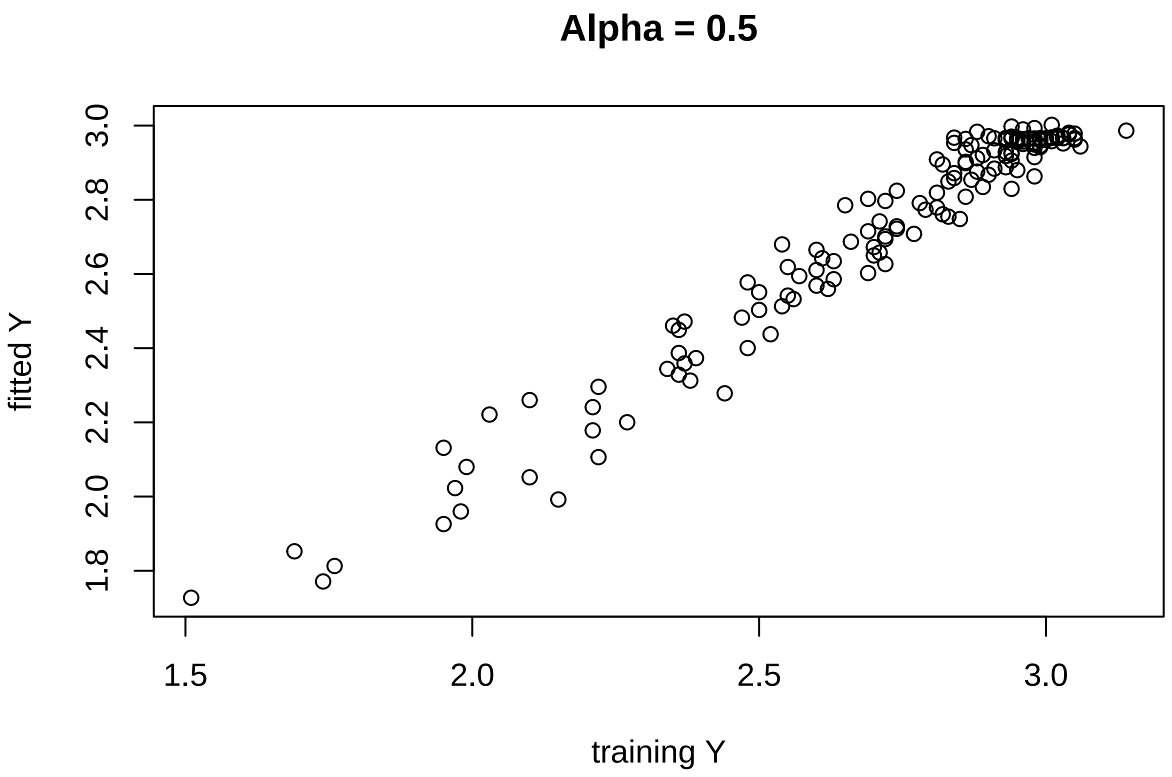
- Yes. I think a square root model $Y_t = \alpha\sqrt{t}$ can have a better fit than polynomial models.

Part b)

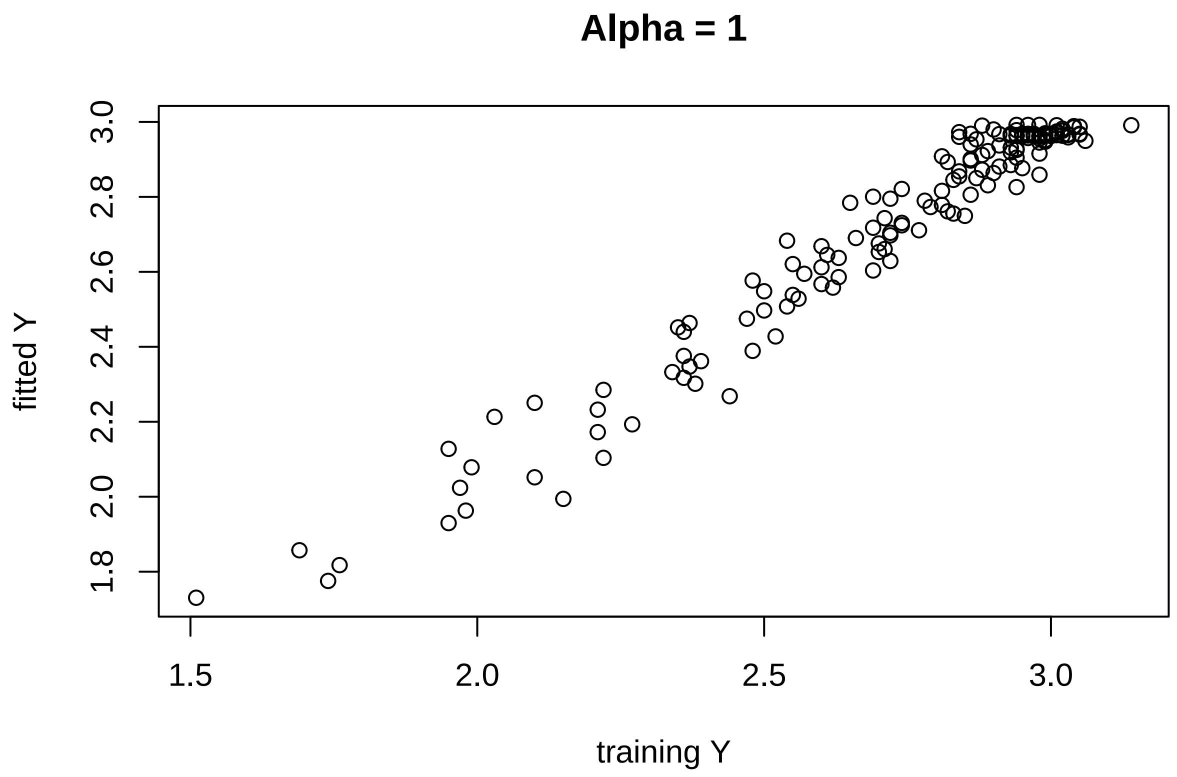
```
## alpha = 0, p=9
X.train.alpha0 <- X.train[, 1:9]
model.alpha0 = cv.glmnet(X.train.alpha0, y.train, type.measure="mse", lambda= Lambda.Seq, alpha=0, famil
alpha0.coeffs = matrix(predict(model.alpha0,type="coef"),ncol=1)
fitted.alpha0 = cbind(1,X.train.alpha0)%*%alpha0.coeffs
plot(y.train, fitted.alpha0,
    xlab = "training Y", ylab = "fitted Y",
    main = "Alpha = 0")
```



```
## alpha = 0.5, p=7
X.train.alpha05 <- X.train[, 1:7]
model.alpha05 = cv.glmnet(X.train.alpha05, y.train, type.measure="mse", lambda= Lambda.Seq, alpha=0.5, f
alpha05.coeffs = matrix(predict(model.alpha05,type="coef"),ncol=1)
fitted.alpha05 = cbind(1,X.train.alpha05)%*%alpha05.coeffs
plot(y.train, fitted.alpha05,
     xlab = "training Y", ylab = "fitted Y",
     main = "Alpha = 0.5")
```



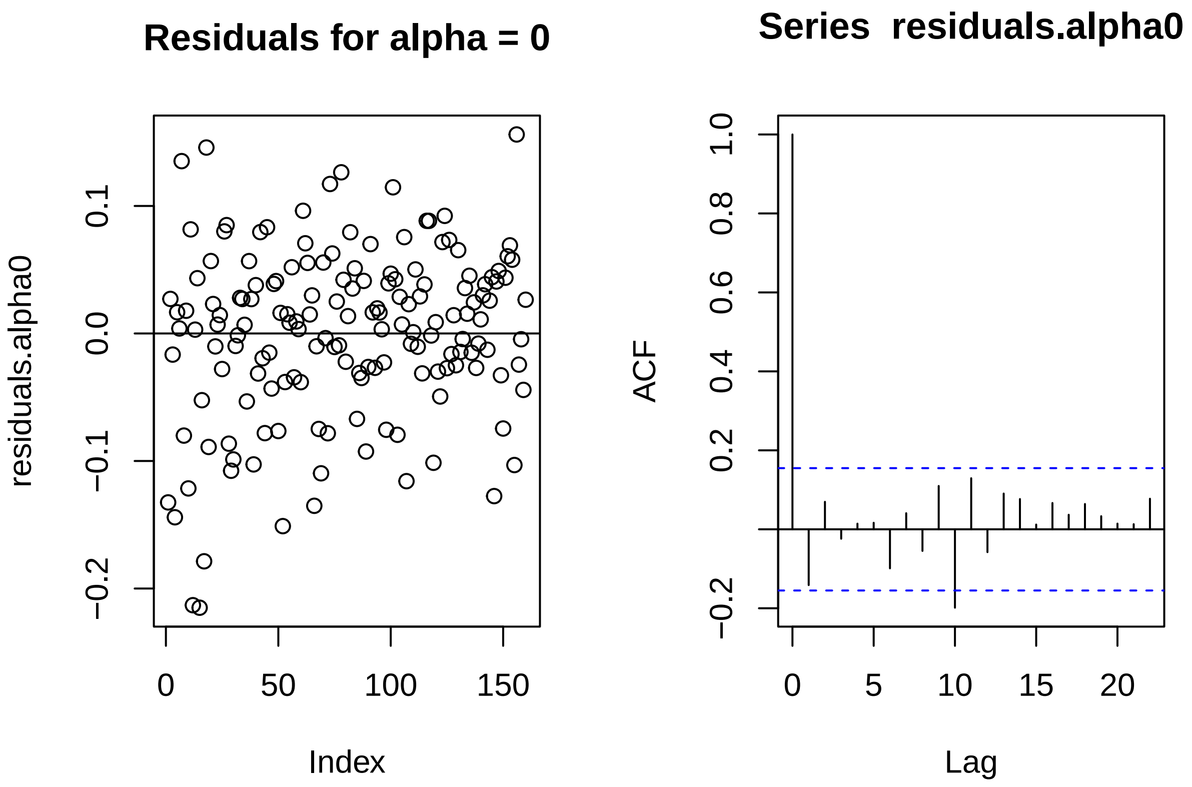
```
## alpha = 1, p=8
X.train.alpha1 <- X.train[, 1:8]
model.alpha1 = cv.glmnet(X.train.alpha1, y.train, type.measure="mse", lambda= Lambda.Seq, alpha=1, famil
alpha1.coeffs = matrix(predict(model.alpha1,type="coef"),ncol=1)
fitted.alpha1 = cbind(1,X.train.alpha1)%*%alpha1.coeffs
plot(y.train, fitted.alpha1,
     xlab = "training Y", ylab = "fitted Y",
     main = "Alpha = 1")
```


Alpha = 1
fitted Y
training Y
1.5
2.0
2.5
3.0
1.8
2.0
2.2
2.4
2.6
2.8
3.0

Part c)

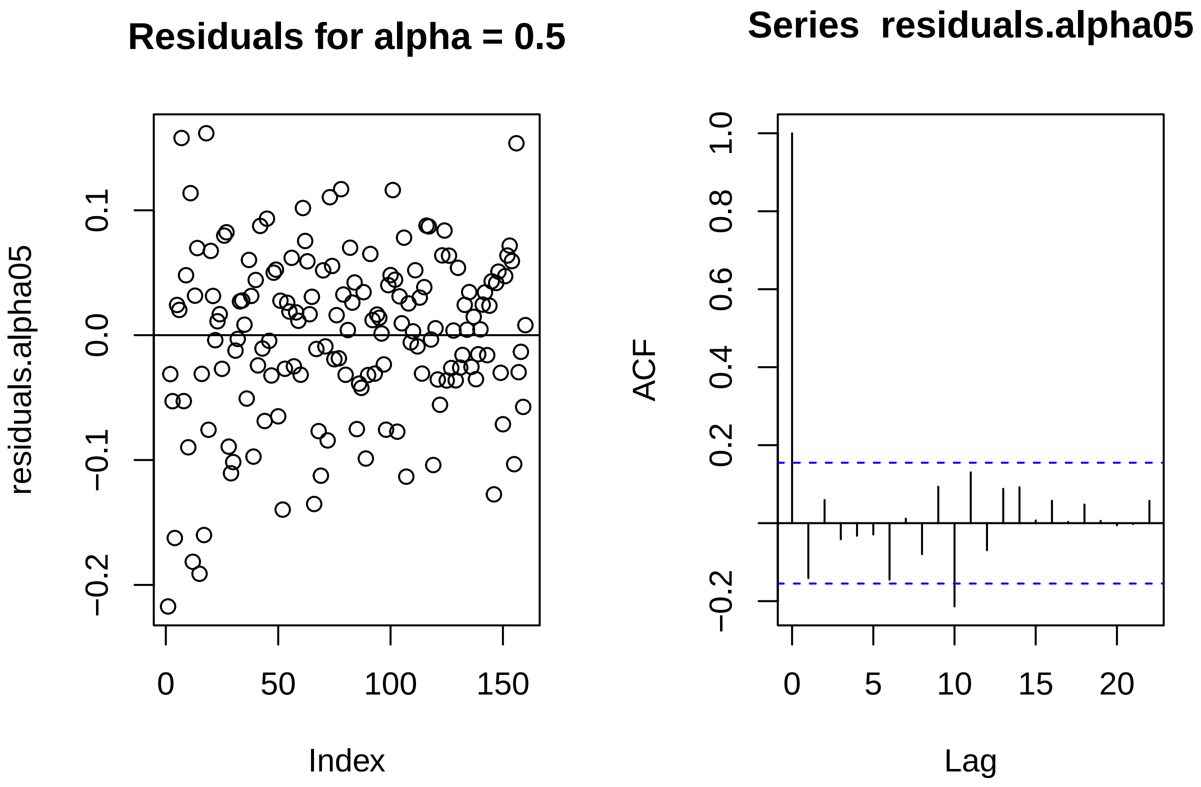
plot of residuals for $\alpha = 0$

```r
residuals.alpha0 = y.train- fitted.alpha0
par(mfrow=c(1, 2))
plot(residuals.alpha0, main = "Residuals for alpha = 0")
abline(h=0)
acf(residuals.alpha0)
```



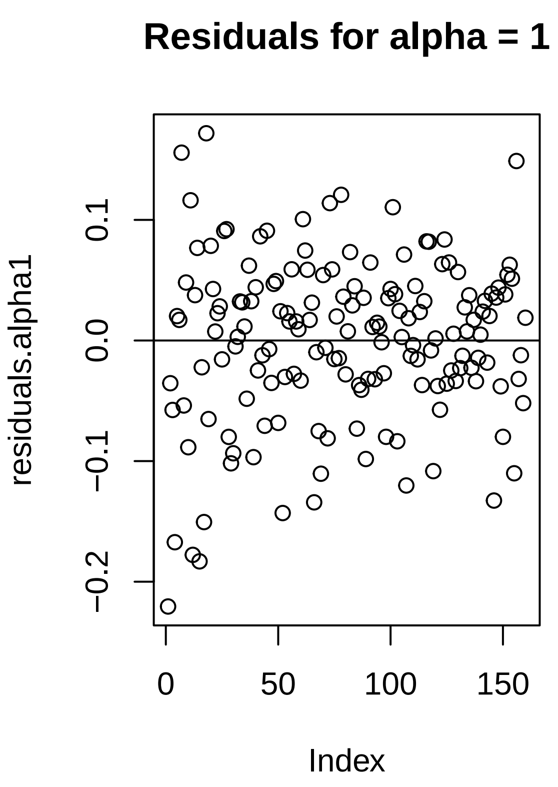
plot of residuals for $\alpha = 0.5$

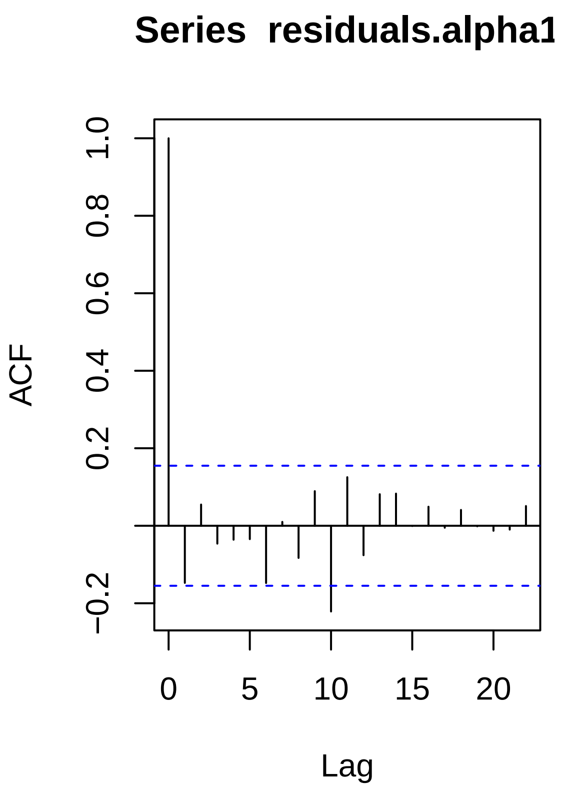
```r
residuals.alpha05 = y.train- fitted.alpha05
par(mfrow=c(1, 2))
plot(residuals.alpha05, main = "Residuals for alpha = 0.5")
abline(h=0)
acf(residuals.alpha05)
```



plot of residuals for $\alpha = 1$

```
residuals.alpha1 = y.train- fitted.alpha1
par(mfrow=c(1, 2))
plot(residuals.alpha1, main = "Residuals for alpha = 1")
abline(h=0)
acf(residuals.alpha1)
```





All ACF's have: - Only a small amount of peaks outside the blue lines - No slow decreasing trend that could indicate a trend in the residuals - No period.

All plots of residuals look randomly spread around zero, so we conclude the residuals from all three models represent iid noise.