MA 4780 Take Home Exam 1

Benjamin Hendrick

February 25, 2016

Problem 1

Part A

(i)

The MA(2) process $Y_t = e_t + 0.8e_{t-1} - 0.1e_{t-2}$ can be set in `R` as such.

```
y <- ARMAacf(ma = c(-0.8,0.1), lag.max = 20)
```

Plot the MA(2) using the `plot` and `abline` commands.

```
plot(y, x = 0:20,
     type = "h",
     ylim = c(-1,1),
     xlab = "k",
     ylab = "Autocorrelation",
     main = "Population ACF of an MA(2) model with coefficients 0.8 and -0.1")
abline(h=0)
```

Figure 1 shows the population ACF plot of the MA(2) process. The ACF plot has only three non-zero lags (at lag zero, one, two). The first lag has a significantly negative auto-correlation.

(ii)

Set up the variables and sample size $n = 100$ in `R`. Fit the MA(2) process using the `lm` function.

```
set.seed(12345)
n <- 100
e <- rnorm(n+1)
y <- ts(e[3:(n+2)]+0.8*e[2:(n+1)]-0.1*e[1:n])
fm <- lm(y~1)
```

Use the `acf` function to create the ACF plot of the sample MA(2) process.

```
acf(resid(fm),main ="Sample ACF plot for the MA(2) model with coefficients 0.8 and -0.1")
```

The sample ACF plot in Figure 2 differs from the population ACF plot in Figure 1 slightly. The population ACF plot is zero from lag 3 onward. The sample ACF plot has a sinusoidal quality to it.

(iii)

The follow code blocks creates new unique sample ACF plots.

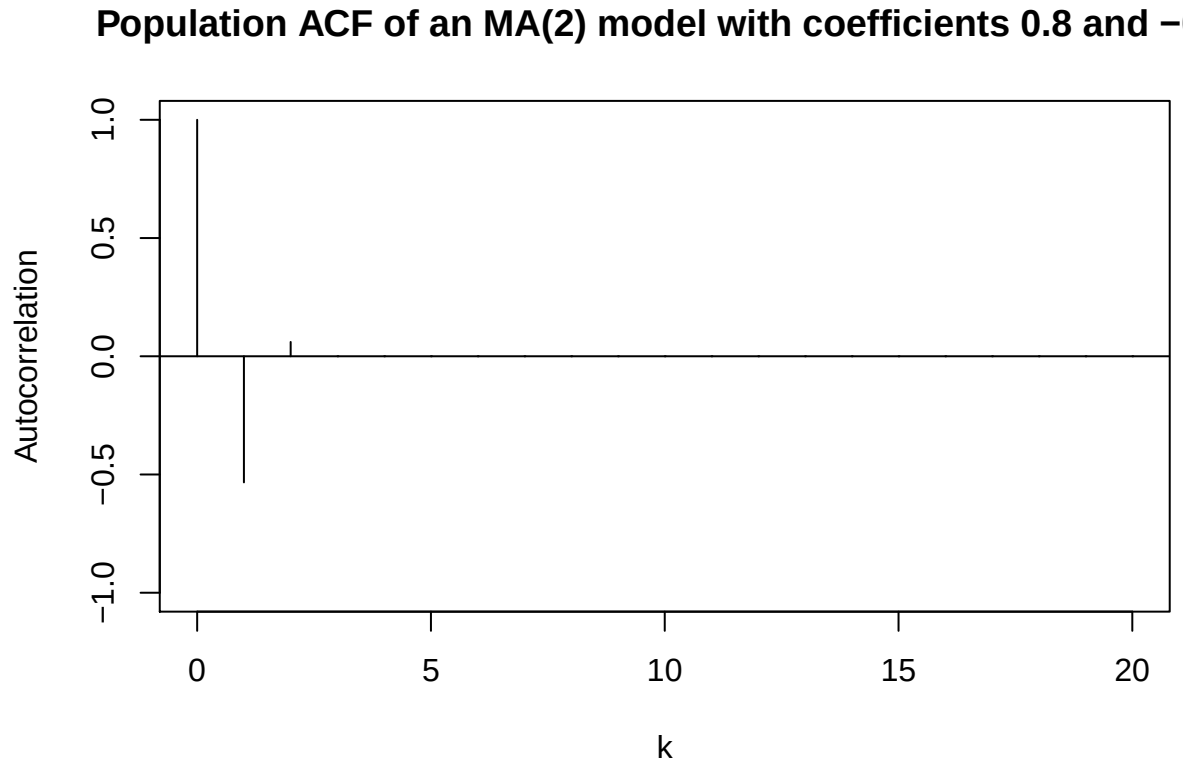

Figure 1: ACF plot of the MA(2) population.

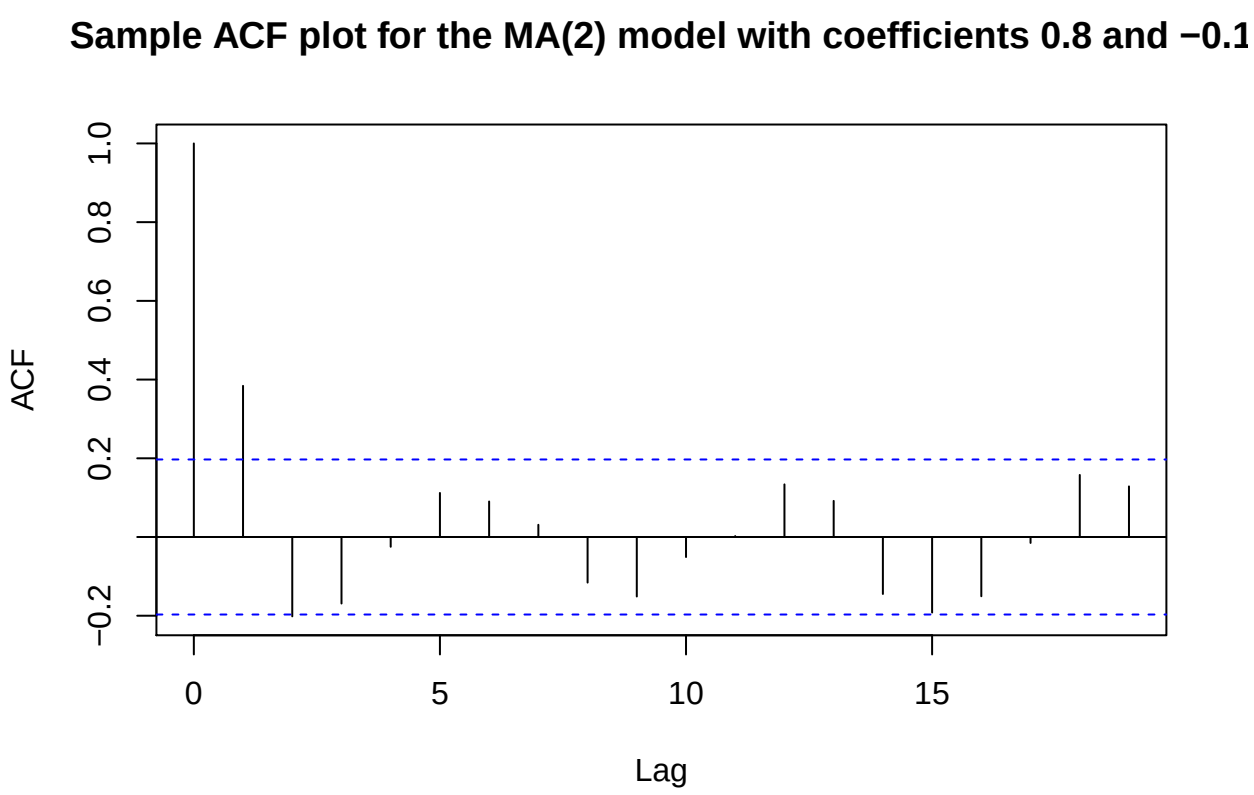

Figure 2: ACF plot of sample MA(2) process with a sample size of 100.

```r
set.seed(1)
n <- 100
e <- rnorm(n+1)
y <- ts(e[3:(n+2)]+0.8*e[2:(n+1)]-0.1*e[1:n])
fm <- lm(y~1)
acf(resid(fm),main ="Sample ACF plot for the MA(2) model with coefficients 0.8 and -0.1")
```

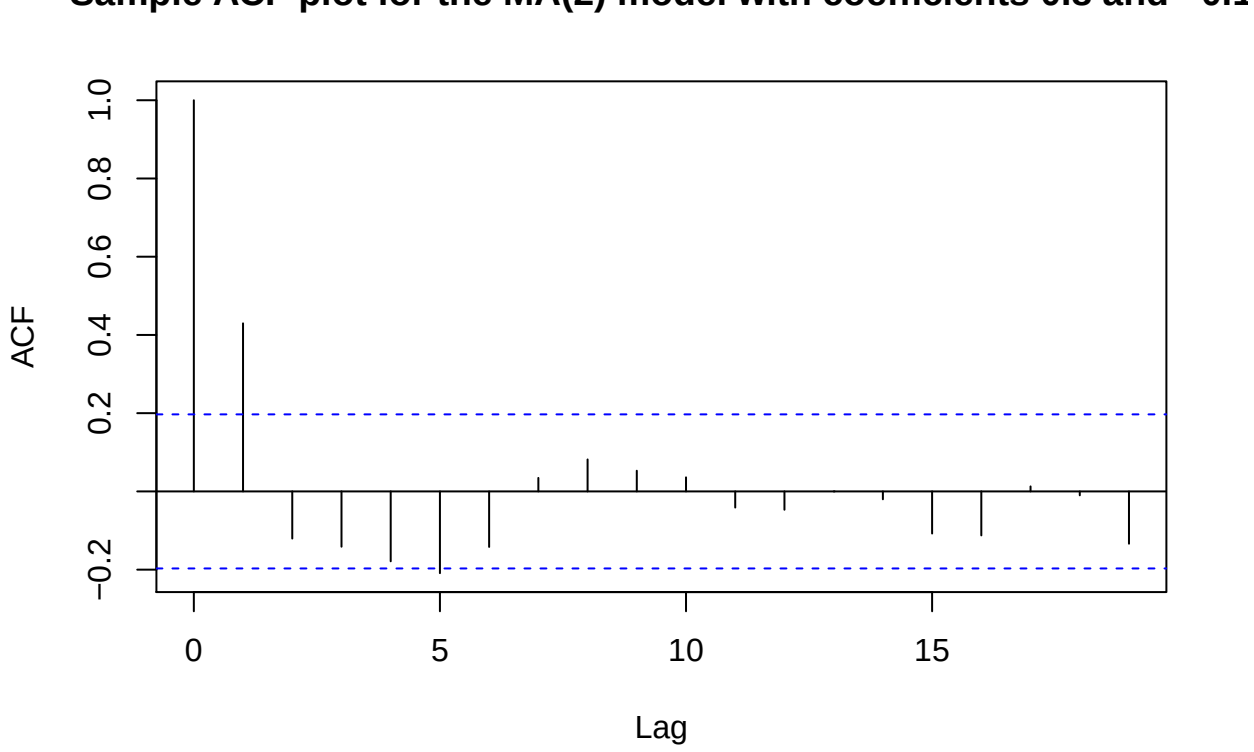

Figure 3: Sample MA(2) ACF plot 1/5

The sample ACF plot in Figure 10 does not share the same shape as the population ACF. The autocorrelations go negative much faster than previous plots.

```r
set.seed(2)
n <- 100
e <- rnorm(n+1)
y <- ts(e[3:(n+2)]+0.8*e[2:(n+1)]-0.1*e[1:n])
fm <- lm(y~1)
acf(resid(fm),main ="Sample ACF plot for the MA(2) model with coefficients 0.8 and -0.1")
```

The sample ACF plot in Figure 4 does not share the same shape as the population ACF. There is significant positive auto-correlation at lag one.

```r
set.seed(3)
n <- 100
e <- rnorm(n+1)
y <- ts(e[3:(n+2)]+0.8*e[2:(n+1)]-0.1*e[1:n])
fm <- lm(y~1)
acf(resid(fm),main ="Sample ACF plot for the MA(2) model with coefficients 0.8 and -0.1")
```

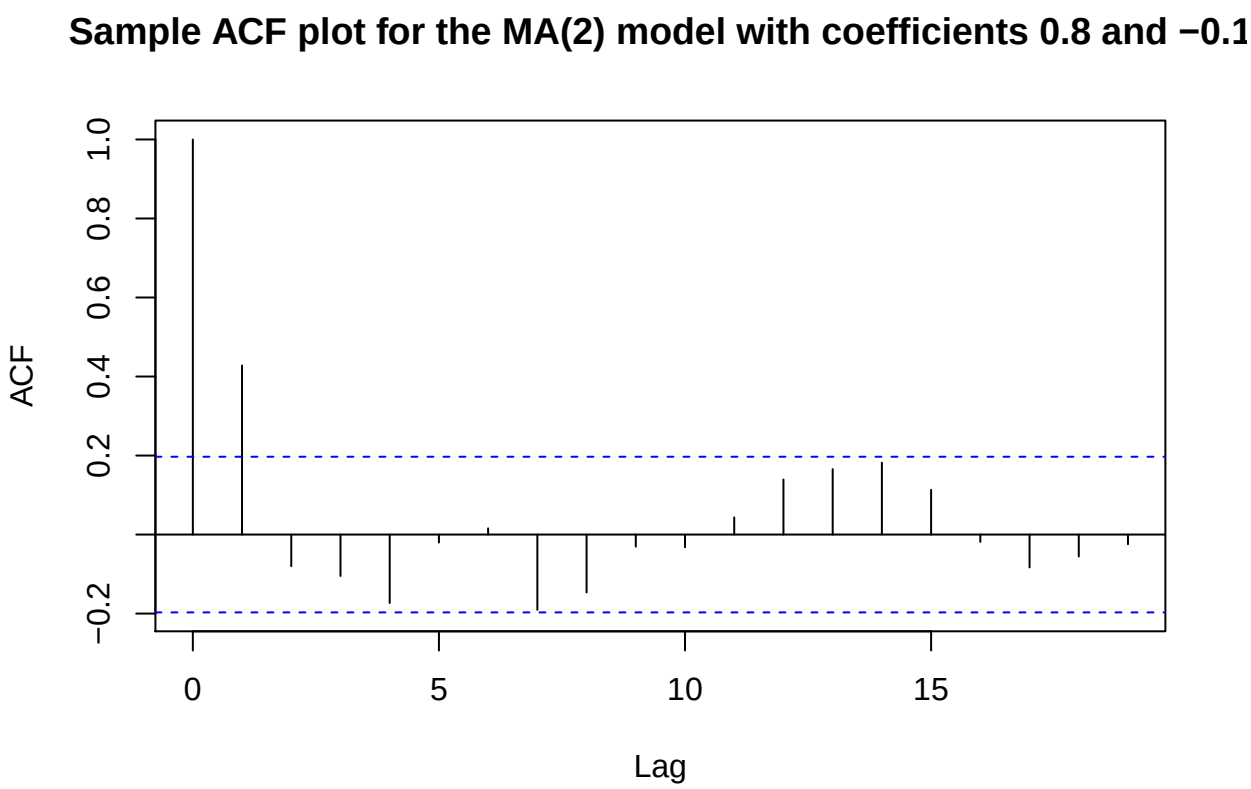

Figure 4: Sample MA(2) ACF plot 2/5

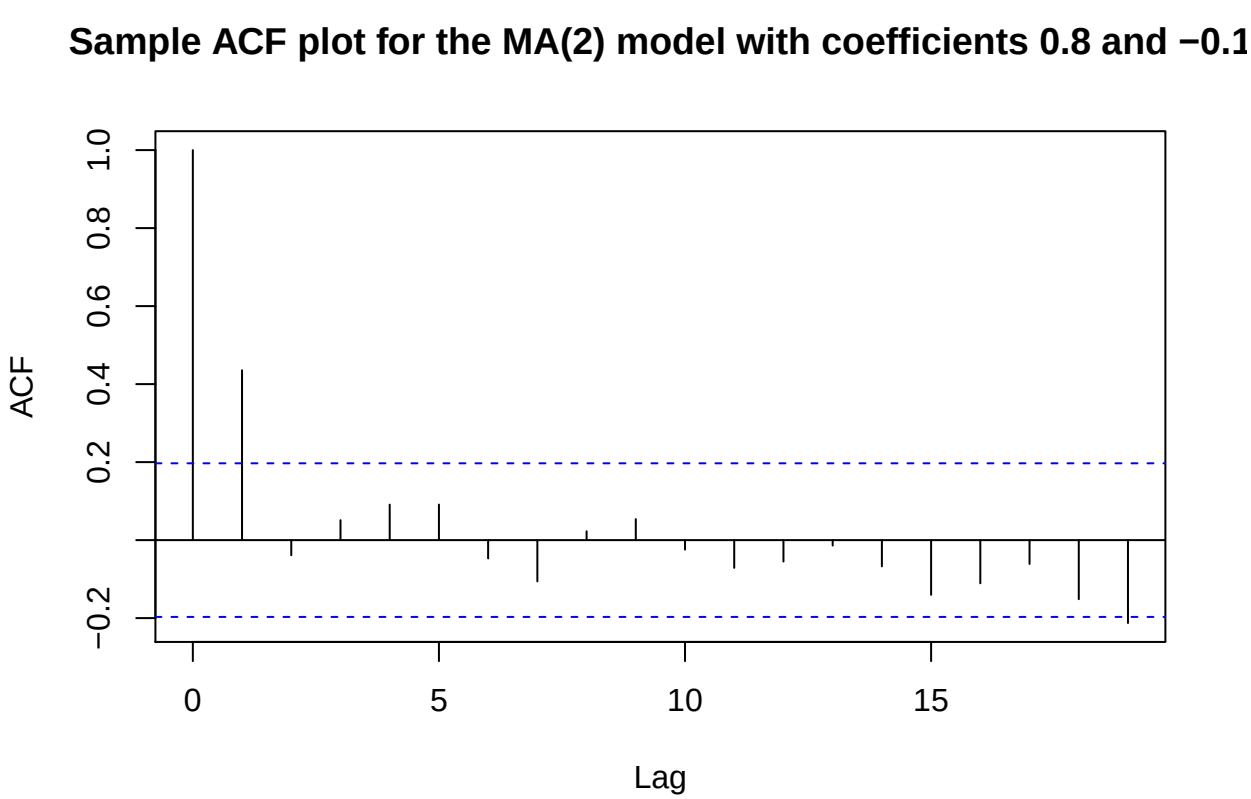

Figure 5: Sample MA(2) ACF plot 3/5

The sample ACF plot in Figure 5 does not share the same shape as the population ACF. There is significant positive auto-correlation at lag one.

```
set.seed(4)
n <- 100
e <- rnorm(n+1)
y <- ts(e[3:(n+2)]+0.8*e[2:(n+1)]-0.1*e[1:n])
fm <- lm(y~1)
acf(resid(fm),main ="Sample ACF plot for the MA(2) model with coefficients 0.8 and -0.1")
```

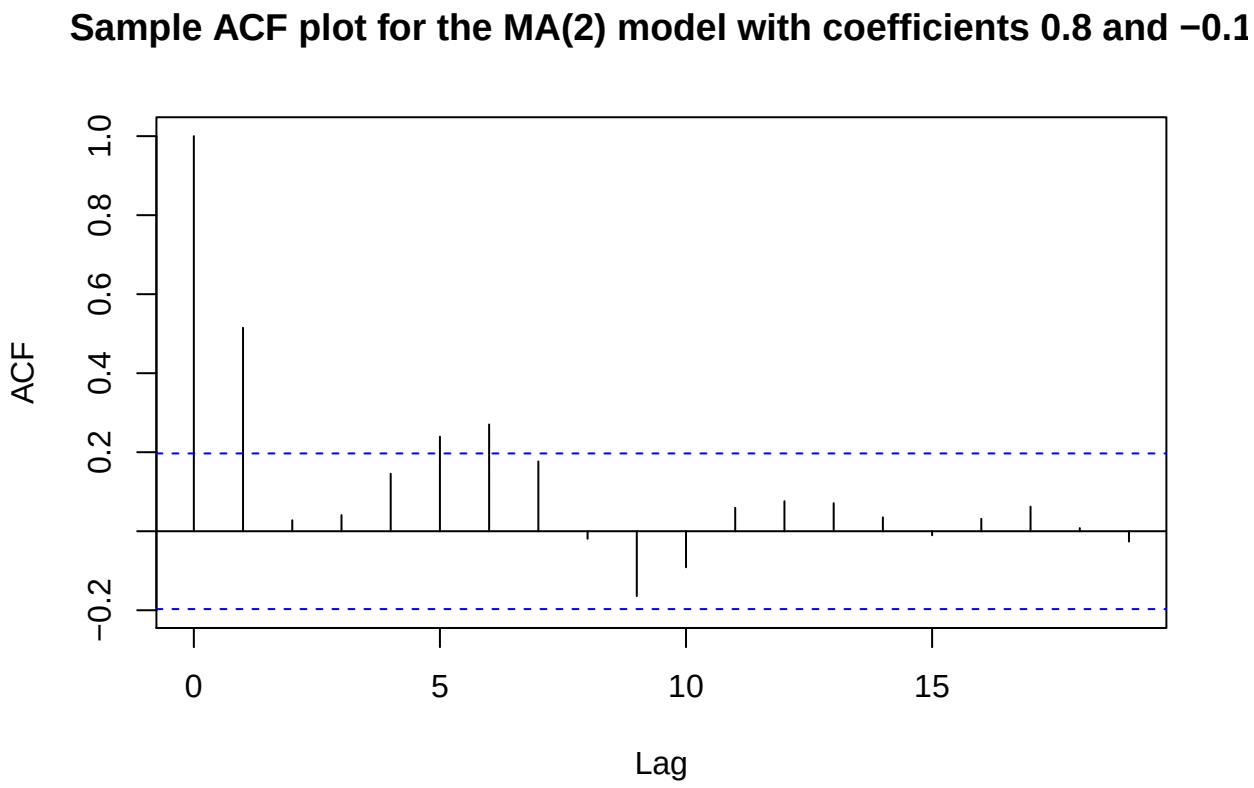

Figure 6: Sample MA(2) ACF plot 4/5

The sample ACF plot in Figure 6 does not share the same shape as the population ACF. There is significant positive auto-correlation at lag one.

```
set.seed(5)
n <- 100
e <- rnorm(n+1)
y <- ts(e[3:(n+2)]+0.8*e[2:(n+1)]-0.1*e[1:n])
fm <- lm(y~1)
acf(resid(fm),main ="Sample ACF plot for the MA(2) model with coefficients 0.8 and -0.1")
```

The sample ACF plot in Figure 7 does not share the same shape as the population ACF. There is significant positive auto-correlation at lag one.

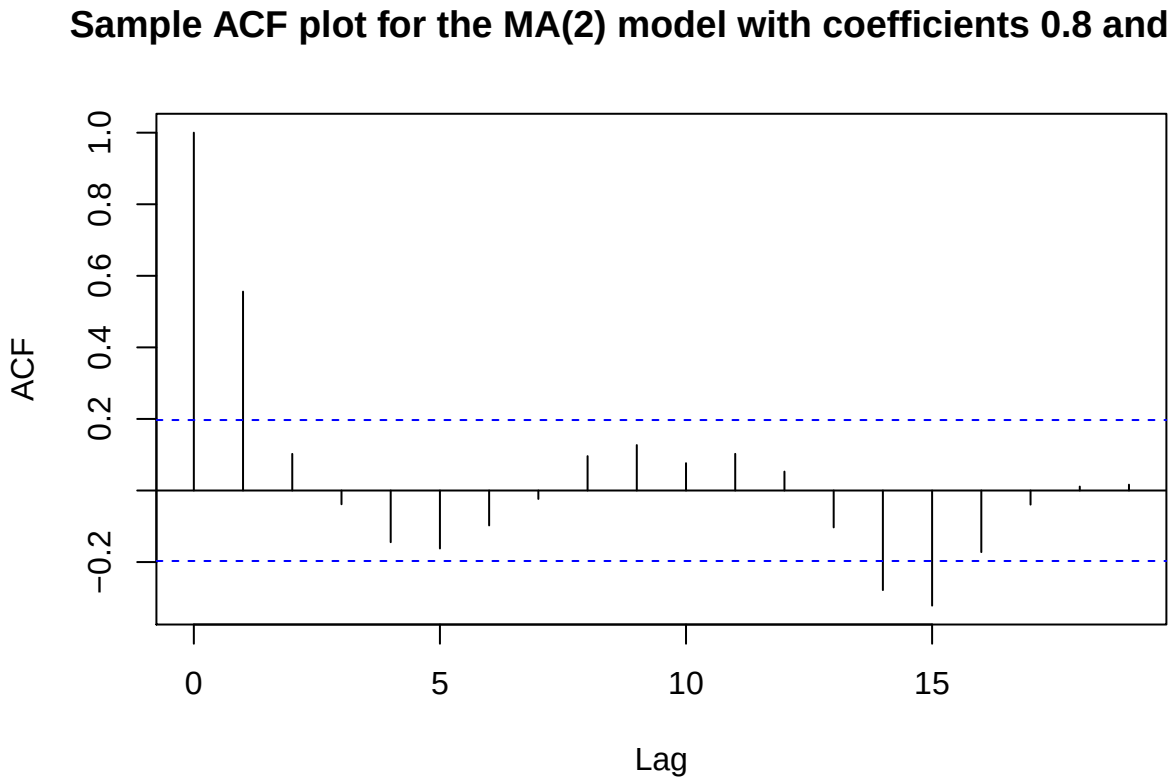

Figure 7: Sample MA(2) ACF plot 5/5

Part B

(i)

The AR(1) process $Y_t = 0.9Y_{t-1} + e_t$ can be set in `R` as such.

```
y <- ARMAacf(ar = 0.9, lag.max = 20)
```

Plot the AR(1) using the `plot` and `abline` commands.

```
plot(y, x = 0:20,
     type = "h",
     ylim = c(-1,1),
     xlab = "k",
     ylab = "Autocorrelation",
     main = "Population ACF of an AR(1) model with coefficient 0.9")
abline(h=0)
```

Figure 8 shows the population ACF plot of the AR(1) process. The ACF plot has not outstanding lags and decreases toward zero, never crossing zero. Many of the lower k lags are significantly auto-correlated.

(ii)

Set up the variables and sample size $n = 100$ in `R`. Fit the AR(1) process using the `lm` function.

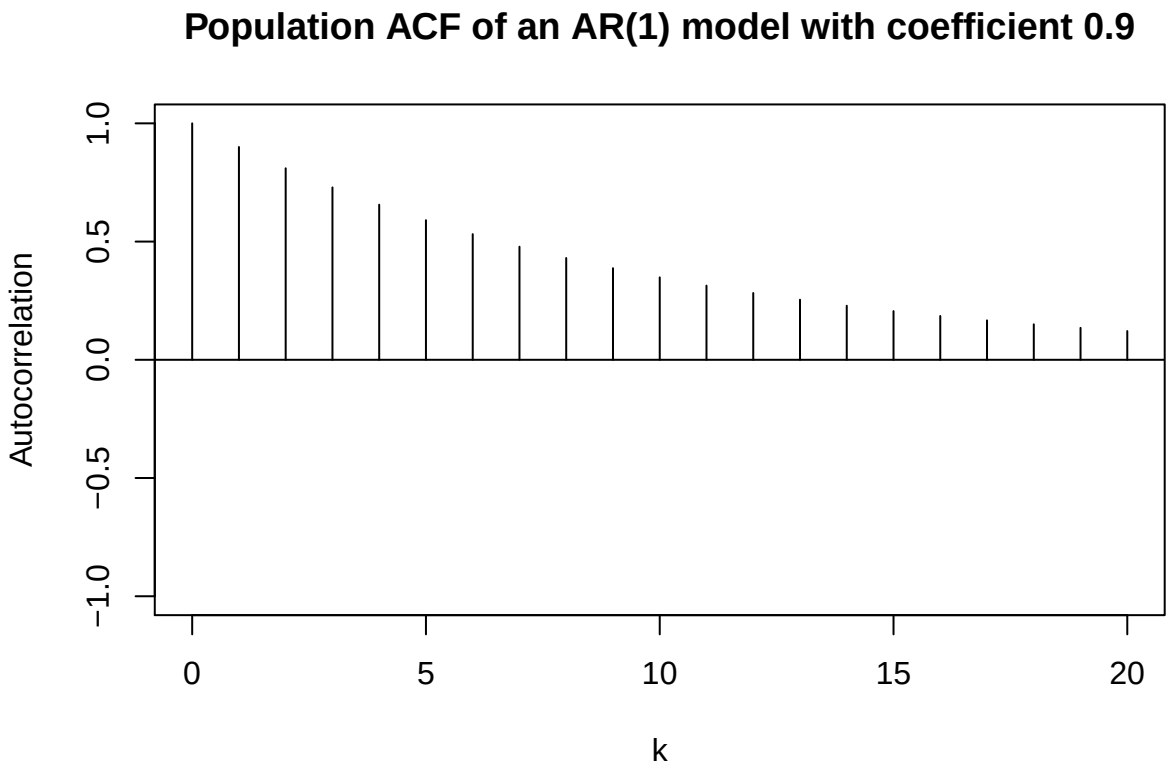

Figure 8: ACF plot of the AR(1) population.

```
set.seed(2190)
n = 100
e = rnorm(n+1)
phi1 = 0.9
y = rep(0,n); y[1] = e[1];
for (i in 2:n){ y[i] = phi1*y[i-1]+e[i]}
```

Use the `acf` function to create the ACF plot of the sample MA(2) process.

```
acf(y,main="Sample ACF of an AR(1) model with coefficient 0.9")
```

The sample ACF plot in Figure **??** is not too different from the population ACF plot in Figure 8. Both plots have decreasing auto-correlations whic are significantly positive until about lag 13. However, the sample ACF plot has negative auto-correlation after lag 17. This is not the case in the population ACF plot.

(iii)

The follow code blocks creates new unique sample ACF plots.

```
  set.seed(1)
  n = 100
  e = rnorm(n+1)
  phi1 = 0.9
  y = rep(0,n); y[1] = e[1];
```

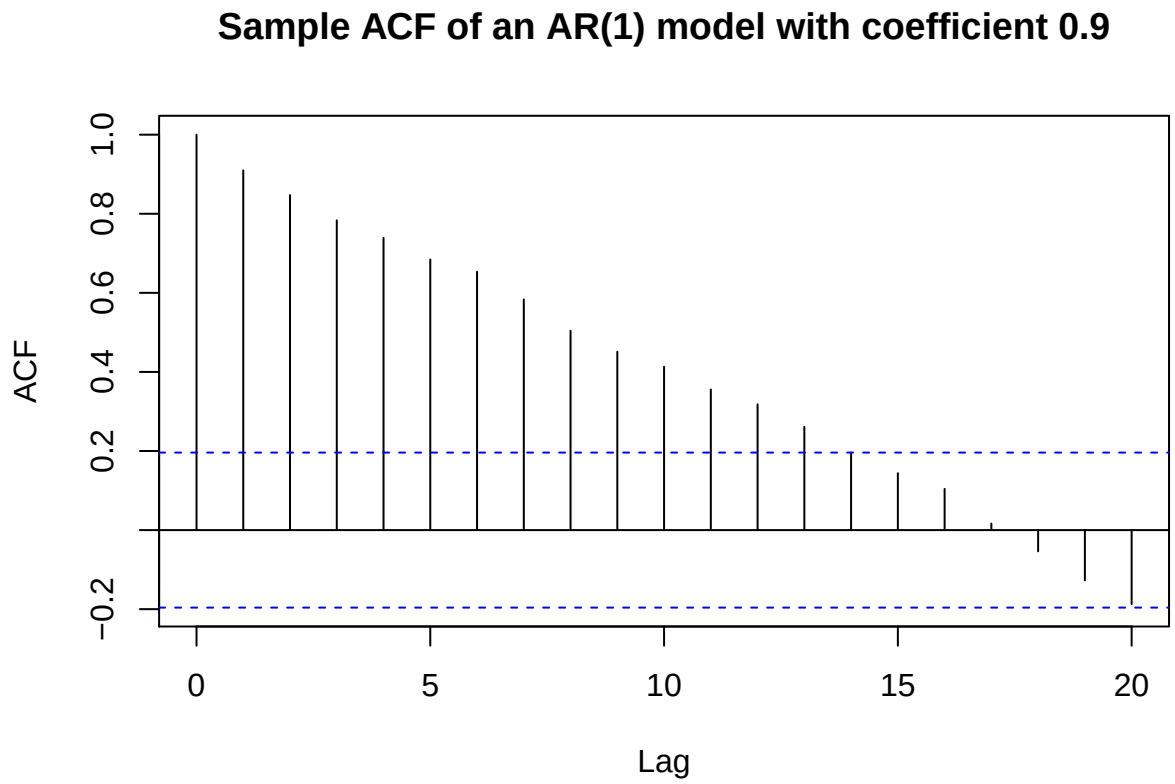

Figure 9: ACF plot of sample AR(1) process with a sample size of 100.

```
for (i in 2:n){ y[i] = phi1*y[i-1]+e[i]}
acf(y,main="Sample ACF of an AR(1) model with coefficient 0.9")
```

The sample ACF plot in Figure 10 does not share the same shape as the population ACF. There is significant positive auto-correlation at lag one.

```
set.seed(2)
n = 100
e = rnorm(n+1)
phi1 = 0.9
y = rep(0,n); y[1] = e[1];
for (i in 2:n){ y[i] = phi1*y[i-1]+e[i]}
acf(y,main="Sample ACF of an AR(1) model with coefficient 0.9")
```

The sample ACF plot in Figure 11 does not share the same shape as the population ACF. The auto-correlation deacres and increases as the significance level before becoming negative.

```
set.seed(3)
n = 100
e = rnorm(n+1)
phi1 = 0.9
y = rep(0,n); y[1] = e[1];
for (i in 2:n){ y[i] = phi1*y[i-1]+e[i]}
acf(y,main="Sample ACF of an AR(1) model with coefficient 0.9")
```

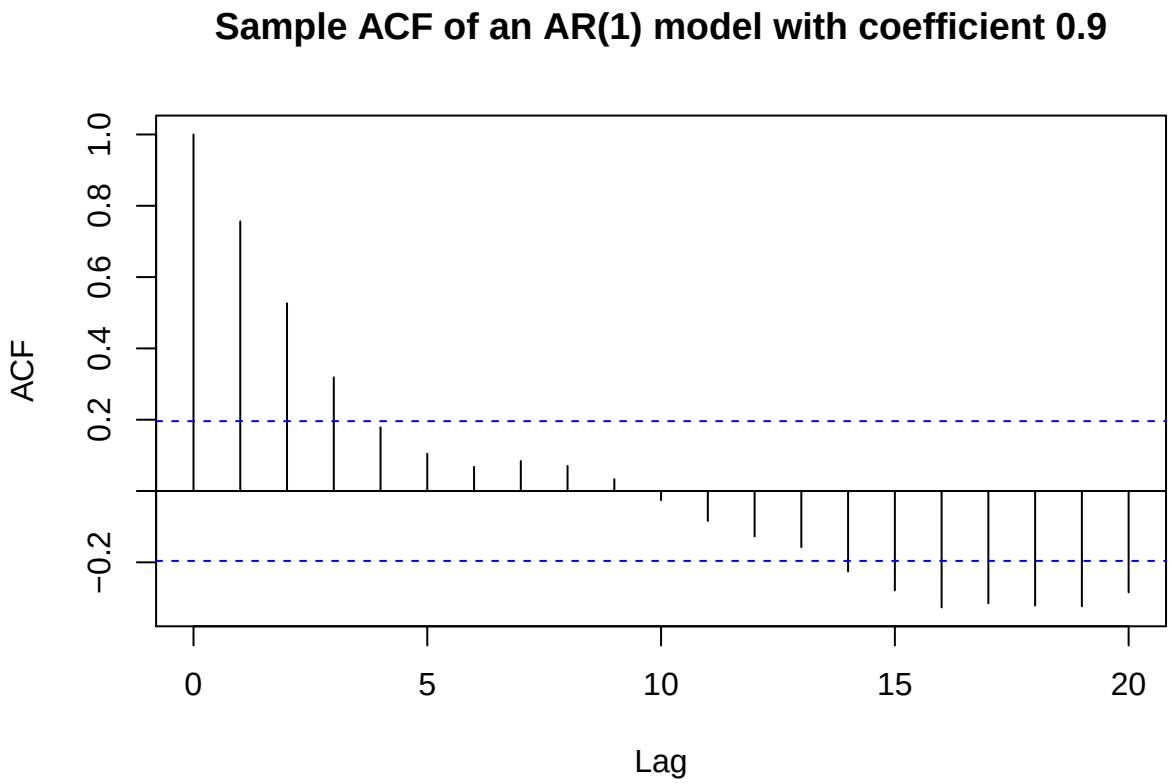

Figure 10: Sample AR(1) ACF plot 1/5

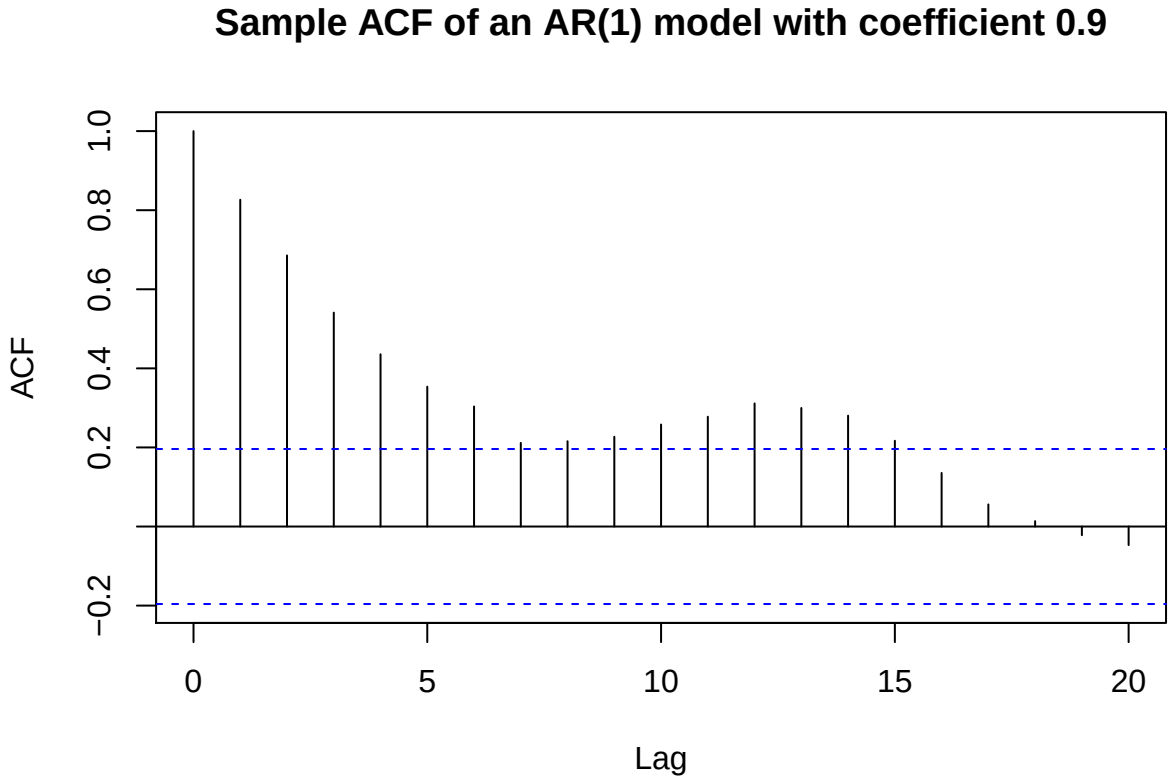

Figure 11: Sample AR(1) ACF plot 2/5

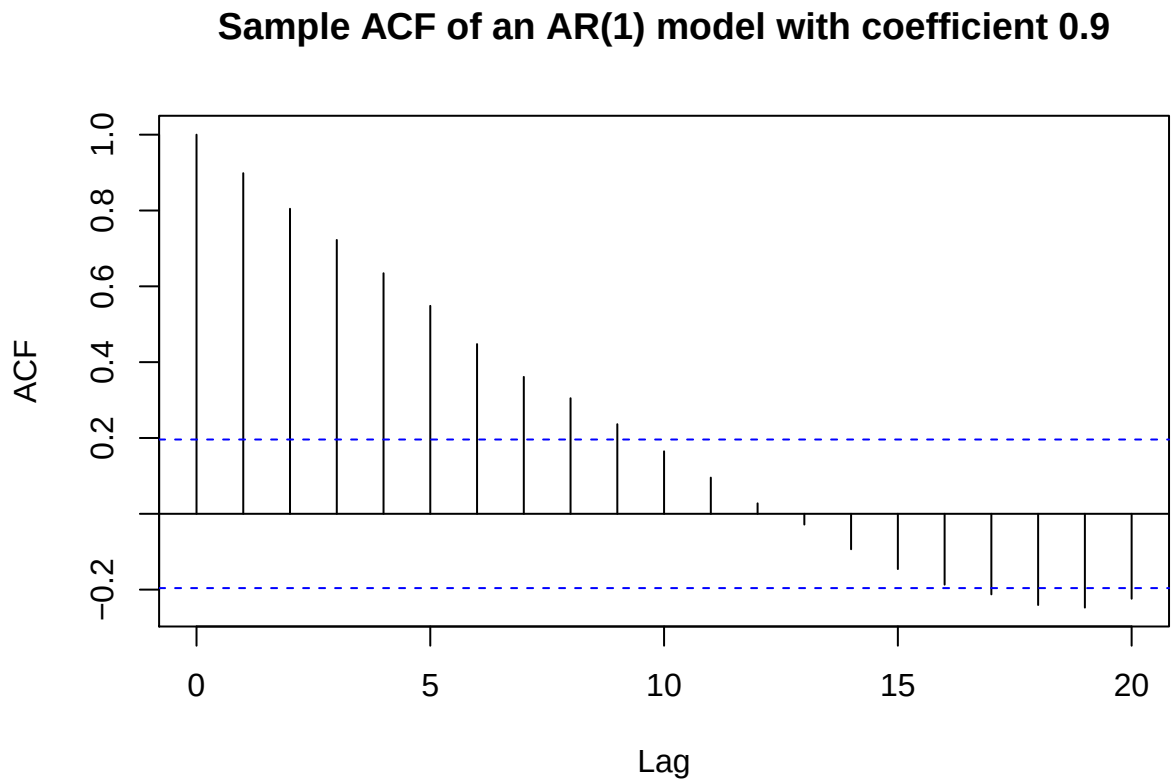

Figure 12: Sample AR(1) ACF plot 3/5

The sample ACF plot in Figure 12 does not share the same shape as the population ACF. The auto-correlation goes negative at lag 13 and continues to decrease into significant negative auto-correlation.

```
set.seed(4)
n = 100
e = rnorm(n+1)
phi1 = 0.9
y = rep(0,n); y[1] = e[1];
for (i in 2:n){ y[i] = phi1*y[i-1]+e[i]}
acf(y,main="Sample ACF of an AR(1) model with coefficient 0.9")
```

The sample ACF plot in Figure **??** does not share the same shape as the population ACF. The plot is similar, but still decreases toward zero negative auto-correlation. This is a large difference.

```
set.seed(5)
n = 100
e = rnorm(n+1)
phi1 = 0.9
y = rep(0,n); y[1] = e[1];
for (i in 2:n){ y[i] = phi1*y[i-1]+e[i]}
acf(y,main="Sample ACF of an AR(1) model with coefficient 0.9")
```

The sample ACF plot in Figure 14 does not share the same shape as the population ACF. The auto-correlation goes negative at lag 13 and continues to decrease into significant negative auto-correlation.

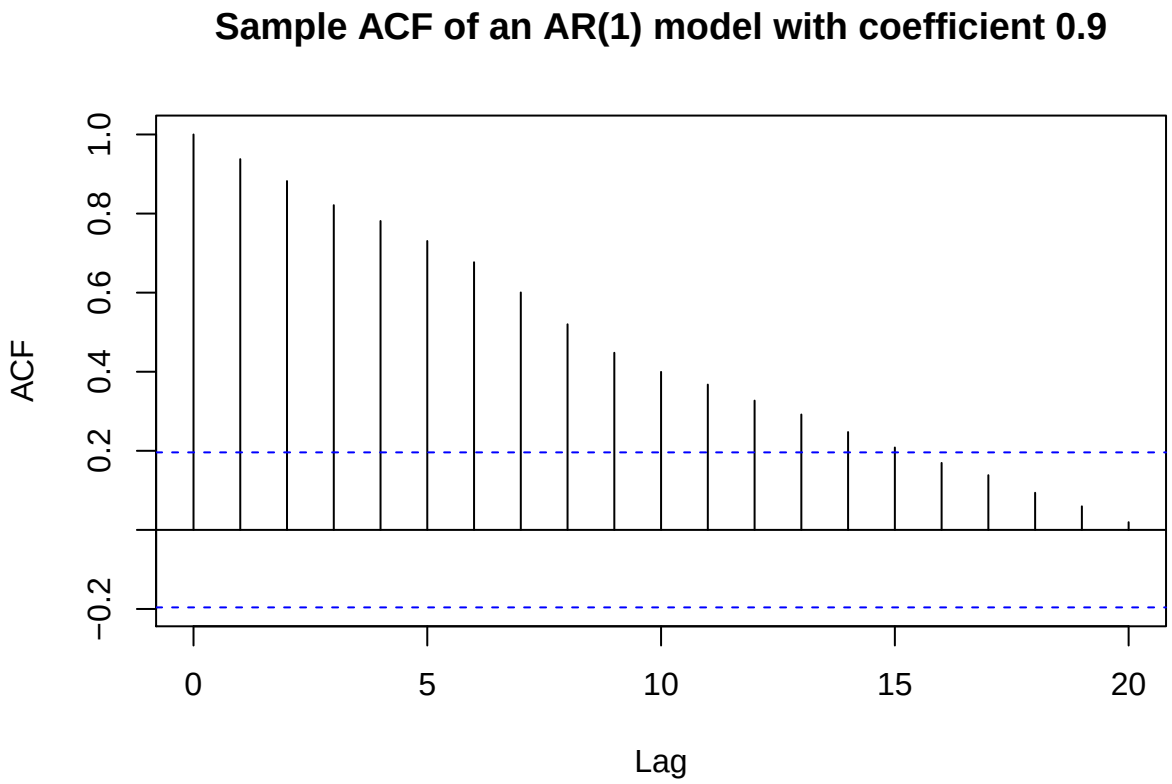

Figure 13: Sample AR(1) ACF plot 4/5

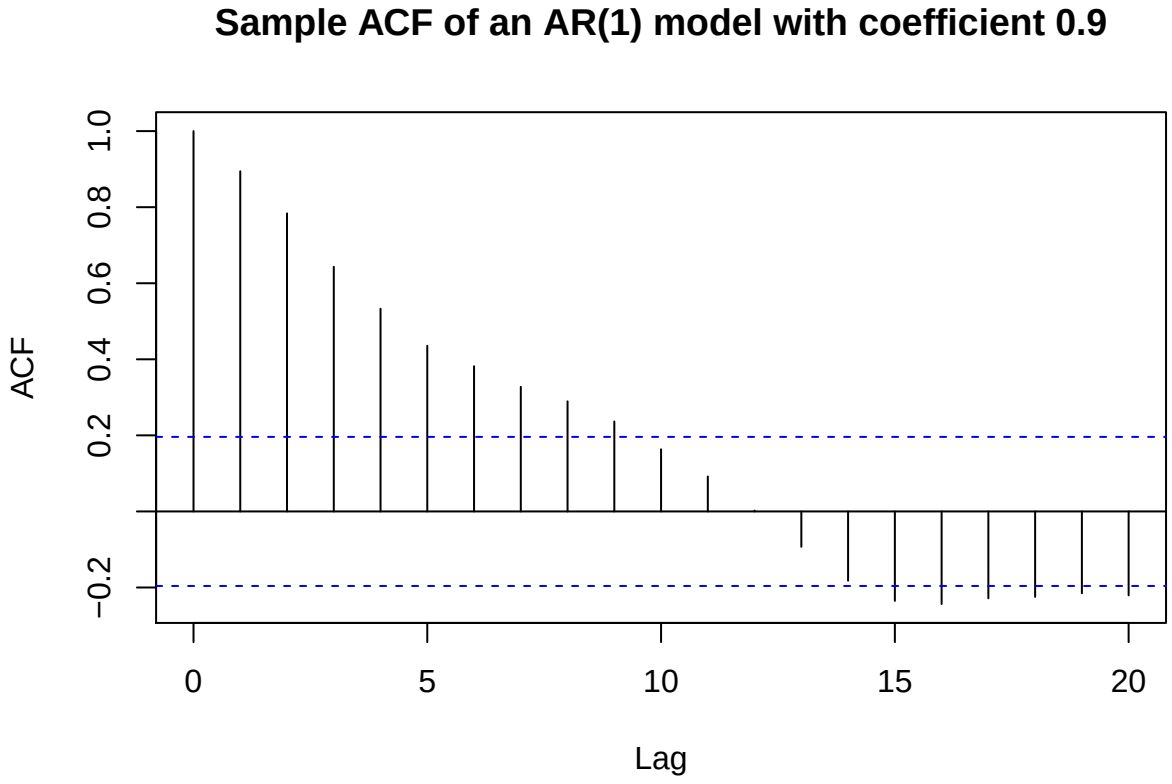

Figure 14: Sample AR(1) ACF plot 5/5

Question 2

Load the data into `R` and clean the time series.

```
library(TSA)
```

```
## Loading required package: leaps
## Loading required package: locfit
## locfit 1.5-9.1    2013-03-22
## Loading required package: mgcv
## Loading required package: nlme
## This is mgcv 1.8-7. For overview type 'help("mgcv-package")'.
## Loading required package: tseries
##
## Attaching package: 'TSA'
##
## The following objects are masked from 'package:stats':
##
##     acf, arima
##
## The following object is masked from 'package:utils':
##
##     tar
```

```
filePath <- "~/GitHub/MA-4780/Exam 1/Exam1-permit.csv"
permit <- read.csv(filePath, header=FALSE)
permit <- ts(permit$V2, start = c(1991,1), end = c(2005,12), frequency = 12)
```

Part A

Plot the data with the `plot` funciton and create the seasonal points with the `points` function.

```
plot(permit,
     type="l",
     ylab="New Residential Construction",
     xlab="Time",
     main="New Residential Constrution over Time")
points(y = permit,
       x = time(permit),
       pch = as.vector(season(permit)))
```

Figure 15 shows the plot of the time series of the `permit` data set. The trend of the data is increasing. There is seasonality where there are peaks in July and trouhs in Janurary.

Part B

Fit the regression function of the seasonal means model is found using the `lm` function. The residuals are found using the `resid` function

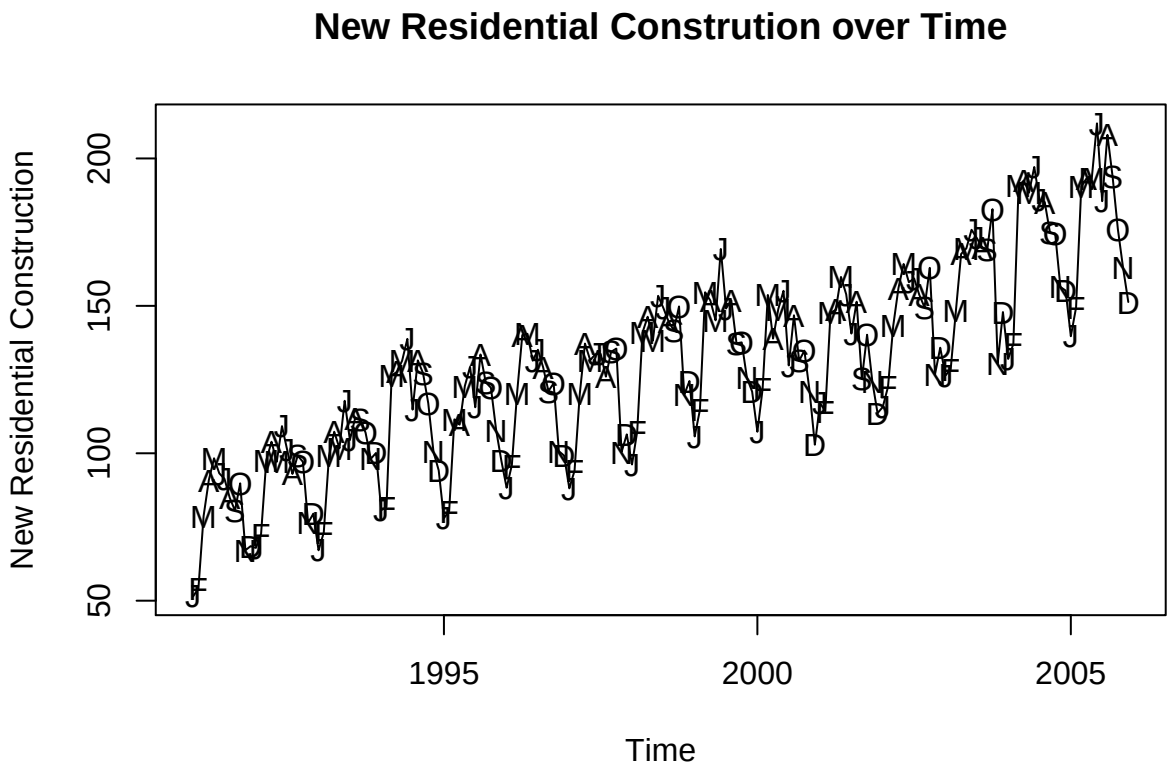

Figure 15: Time series plot the permit data with seasonal point markers.

```
permit.lm <- lm(permit~season(permit))
summary(permit.lm)$coefficients
```

```
##                           Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)              97.44000   7.512591 12.9702251 4.220493e-27
## season(permit)February    5.74000  10.624408  0.5402654 5.897295e-01
## season(permit)March      37.46667  10.624408  3.5264708 5.430238e-04
## season(permit)April      43.33333  10.624408  4.0786585 6.982115e-05
## season(permit)May        44.60000  10.624408  4.1978808 4.358473e-05
## season(permit)June       51.08000  10.624408  4.8077971 3.370157e-06
## season(permit)July       40.46667  10.624408  3.8088395 1.955157e-04
## season(permit)August     43.99333  10.624408  4.1407796 5.468702e-05
## season(permit)September 37.28667  10.624408  3.5095287 5.762728e-04
## season(permit)October    39.28667  10.624408  3.6977745 2.942348e-04
## season(permit)November   17.12667  10.624408  1.6120113 1.088365e-01
## season(permit)December   15.72000  10.624408  1.4796118 1.408496e-01
```

```
permit.resid <- resid(permit.lm)
```

The coefficients of the regression function are all significant because their p-values are less than $\alpha = 0.05$ (and even $\alpha = 0.01$). This means that each coefficient is non-zero and can be used in the model.

Part C

Plot the residuals `permit.resid` using the `plot` function.

```
plot(permit.resid,
     type = "o",
     ylab = "Raw Residuals",
     main = "Reesiduals of New Residential Construction over Time")
```

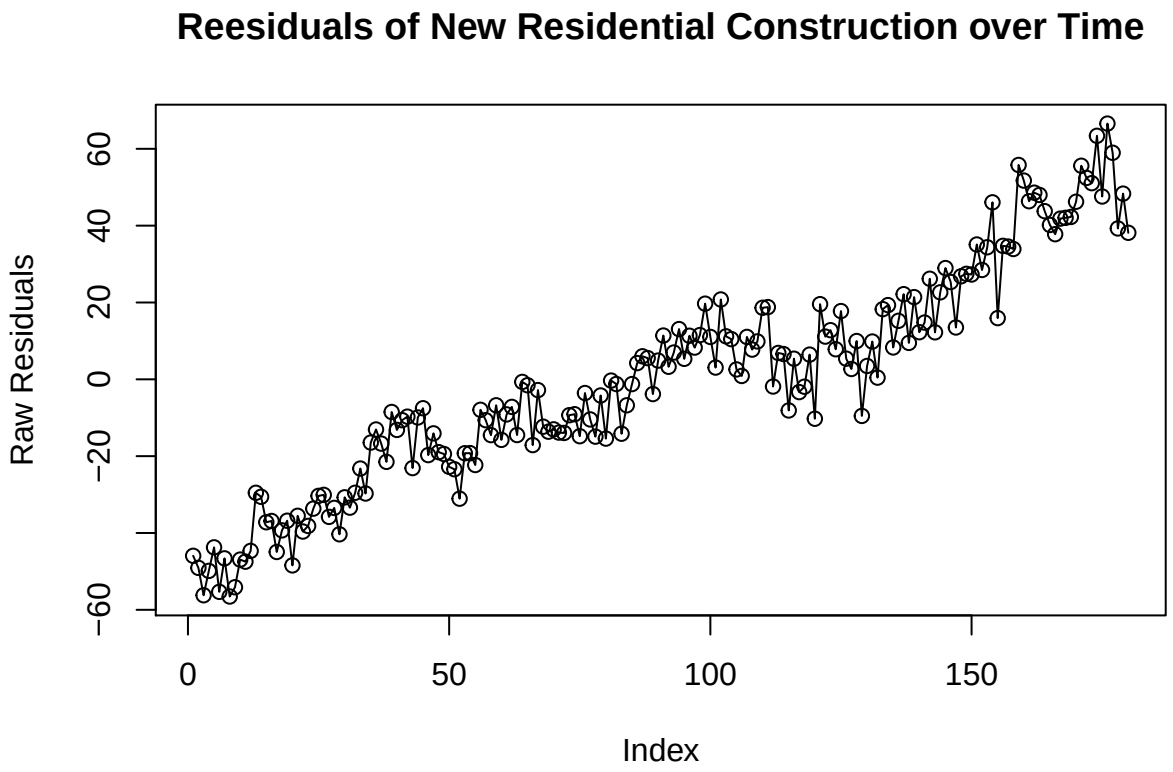

Figure 16: Residuals of New Residential Construction over Time

The residual plot in Figure 16 has a positive trend over time. There is not a strong indication of seasonality in the residuals. Although it is not necessarily homoscedastic, the residuals appear to have random variance.

Part D

Fit the residuals to a regression model by using the `lm` funcitin.

```
permit.resid.lm <- lm(permit.resid~time(permit.resid))
```

Plot the residuals against the fitted values by using the `plot` function.

```
plot(permit.resid.lm, which=1)
```

Find the summary of the residuals model by using the `summary` function.

```
summary(permit.resid.lm)
```

```
##
## Call:
```

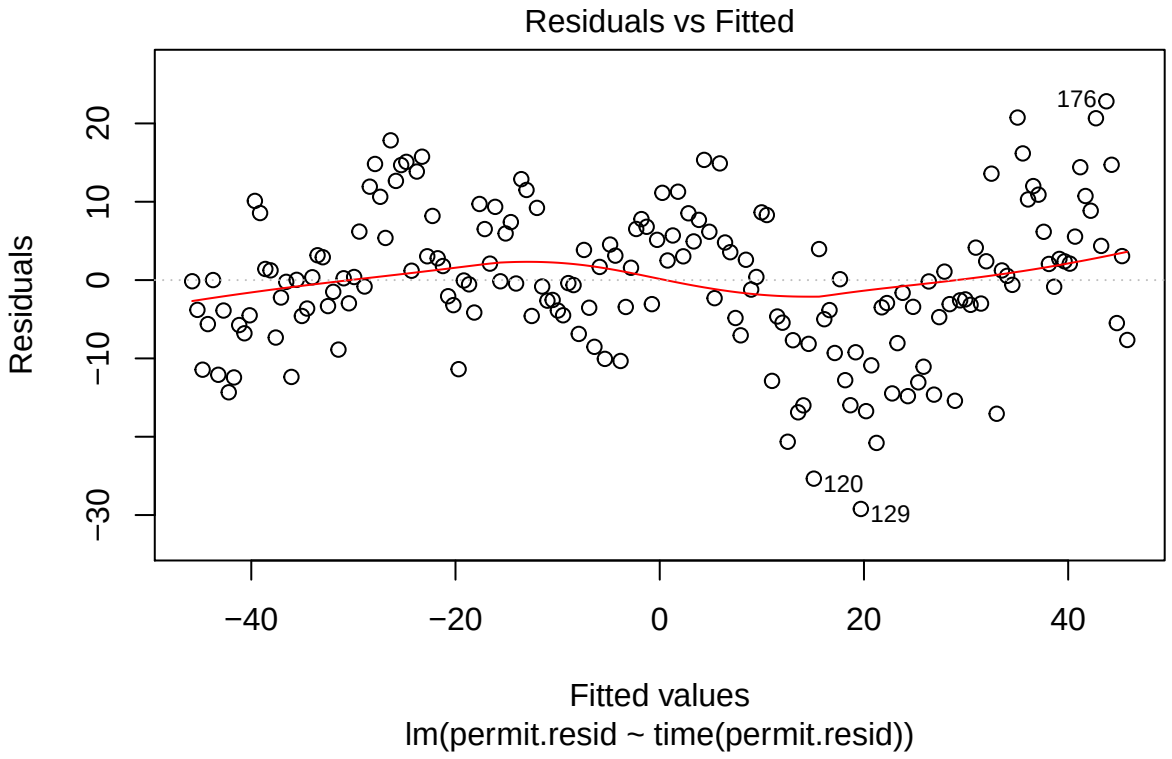

Figure 17: Plot of permit residuals vs. fitted values \label{permit.resid.v.fitted

```
## lm(formula = permit.resid ~ time(permit.resid))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -29.223  -4.671  -0.104   6.005  22.826
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -46.29899    1.37562  -33.66   <2e-16 ***
## time(permit.resid)   0.51159    0.01318   38.81   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.19 on 178 degrees of freedom
## Multiple R-squared:  0.8943, Adjusted R-squared:  0.8937
## F-statistic:  1506 on 1 and 178 DF,  p-value: < 2.2e-16
```

(i)

The p-value of the F-statistic is $< 2.2e-16$. Because it is less than $\alpha = 0.05$, we can say that the linear model is significant. Similarly, both coeficients are significant becasause their p-values are both less than $\alpha = 0.05$.

(ii)

The estimated slope of the linear model is 0.51159.

(iii)

The estimated slope can be interpreted by saying that the expected value of the error of the seasonal trends model increases by 0.51159 for every unit time.

Part E

(i)

Check if the error process has mean equal zero.

```
mean(fitted(permit.resid.lm))
```

```
## [1] -7.226858e-16
```

By a rounding error, the mean is effectively zero. Therefore the assumption holds.

(ii)

Check if the errors are independent.

```
runs(rstudent(permit.resid.lm))[1]
```

```
## $pvalue
## [1] 4.05e-07
```

The runs test suggests a lack of independence in the errors beacuse the p-value is less than $\alpha = 0.05$.

(iii)

Plot the fitted residuals using the `plot` function to see if the error is homogenous.

```
plot(fitted(permit.resid.lm),
     type = "o",
     xlab = "Index",
     ylab = "Fitted Residuals",
     main = "Plot of Fitted Residuals")
```

Based on the plot in Figure 18 the errors are homogenous.

(iv)

Plot the normal probability plot of the residuals linear model by using the `plot` funciton.

```
plot(permit.resid.lm, which=2)
```

Create a histogram of the fitted regression errors with the `hist` function.

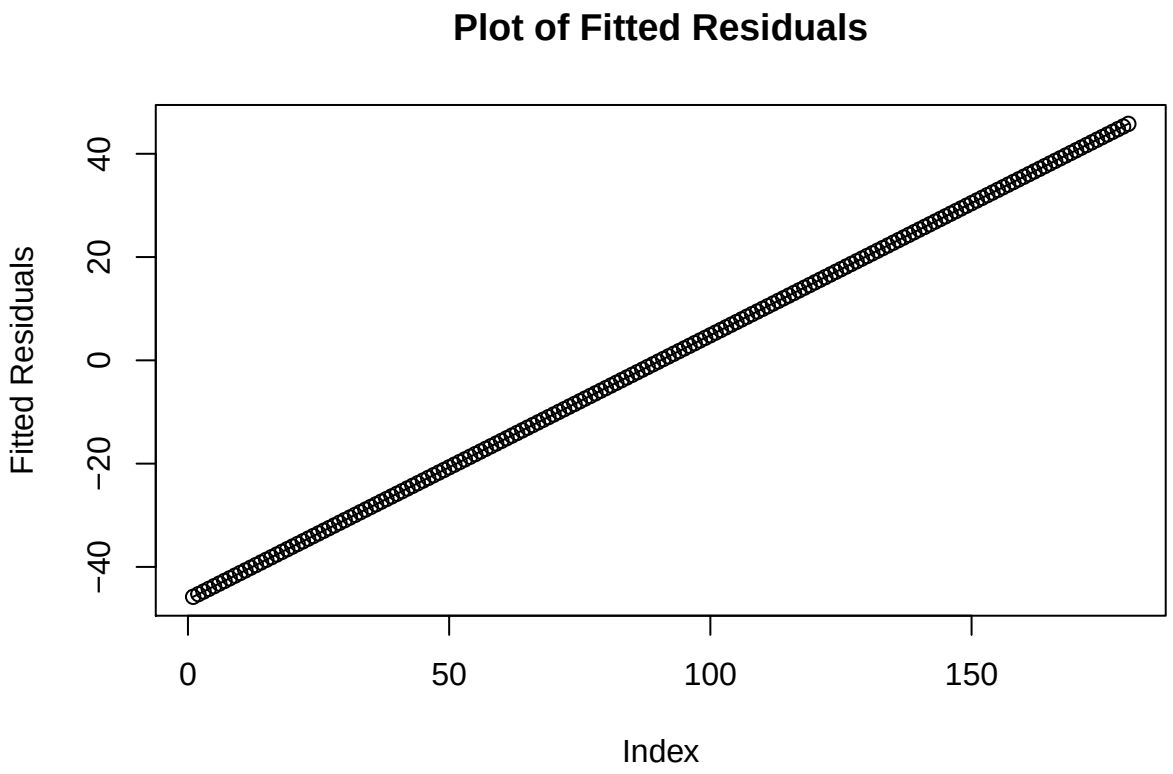

Figure 18: Plot of Fitted Residuals for Permit Data

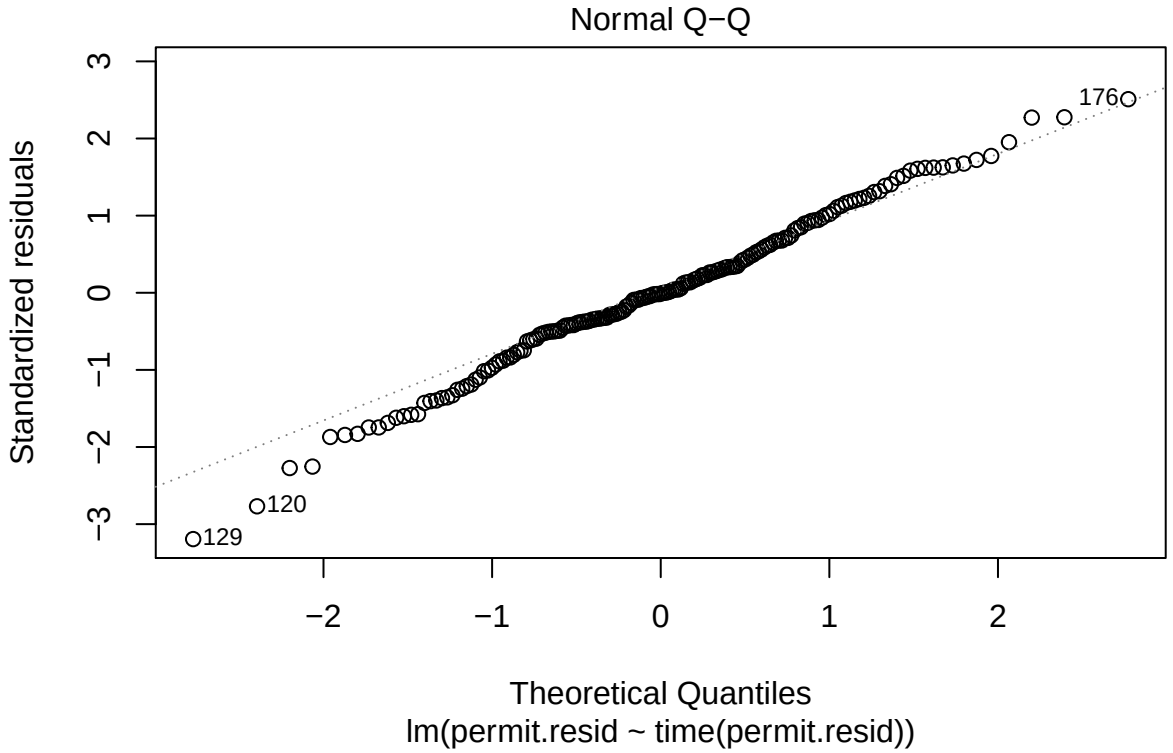

Figure 19: Normal Probability Plot of Permit Residuals

```
hist(fitted(permit.resid.lm),
     xlab = "Fitted Linear Error Process",
     main = "Histogram of Fitted Linear Error Process")
```

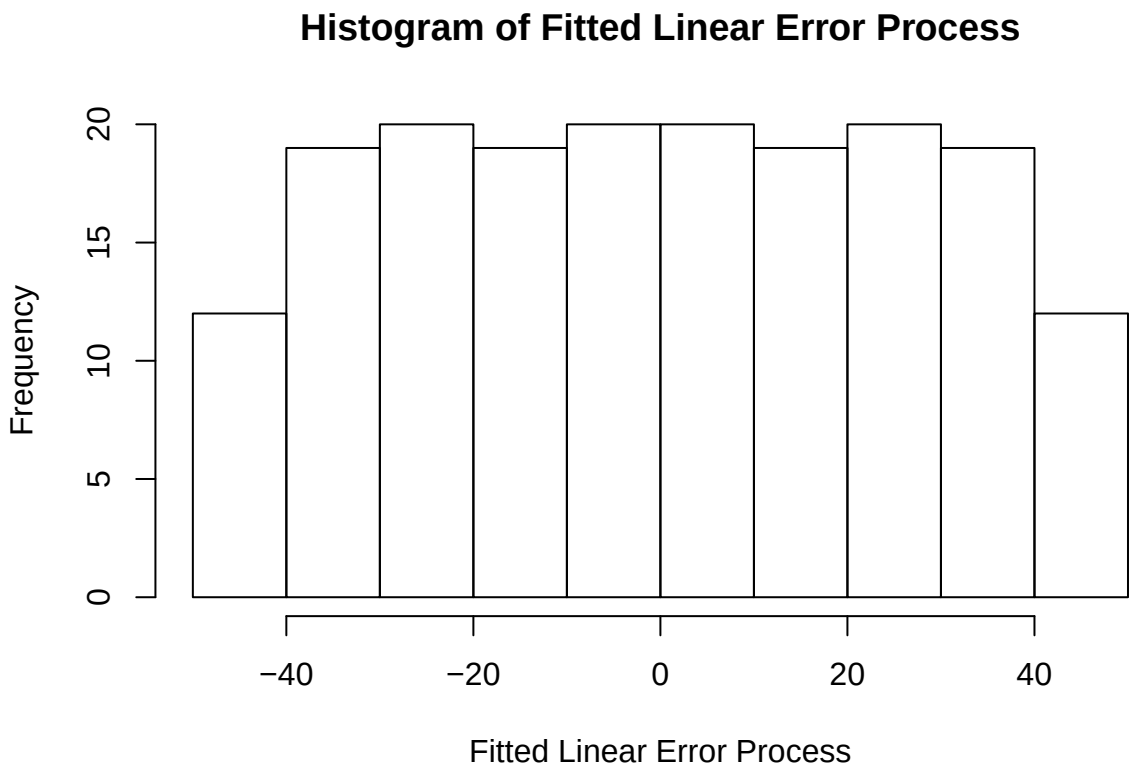

Figure 20: Histograpm of Permit Residuals

Based on the normal probability plot and the histogram in Figure 19 and **??**, the fitted errors do not appear to be normally distributed well. The distribution is still symetric, but just very wide.

Problem 3

Load the data into `R` and clean the time series.

```
library(TSA)
filePath <- "~/GitHub/MA-4780/Exam 1/Exam1-VMT.csv" # Mac
vmt <- read.csv(filePath, header=FALSE)
vmt <- ts(vmt, start = c(2000,1), end = c(2015,12), frequency = 12)
```

Part A

Plot the data with the `plot` funciton and create the seasonal points with the `points` function.

```
plot(vmt,
     type="l",
     ylab="Vehicle Miles Traveled",
```

```
     xlab="Time",
     main="Vehicle Miles Traveled over Time")
points(y = vmt,
       x = time(vmt),
       pch = as.vector(season(vmt)))
```

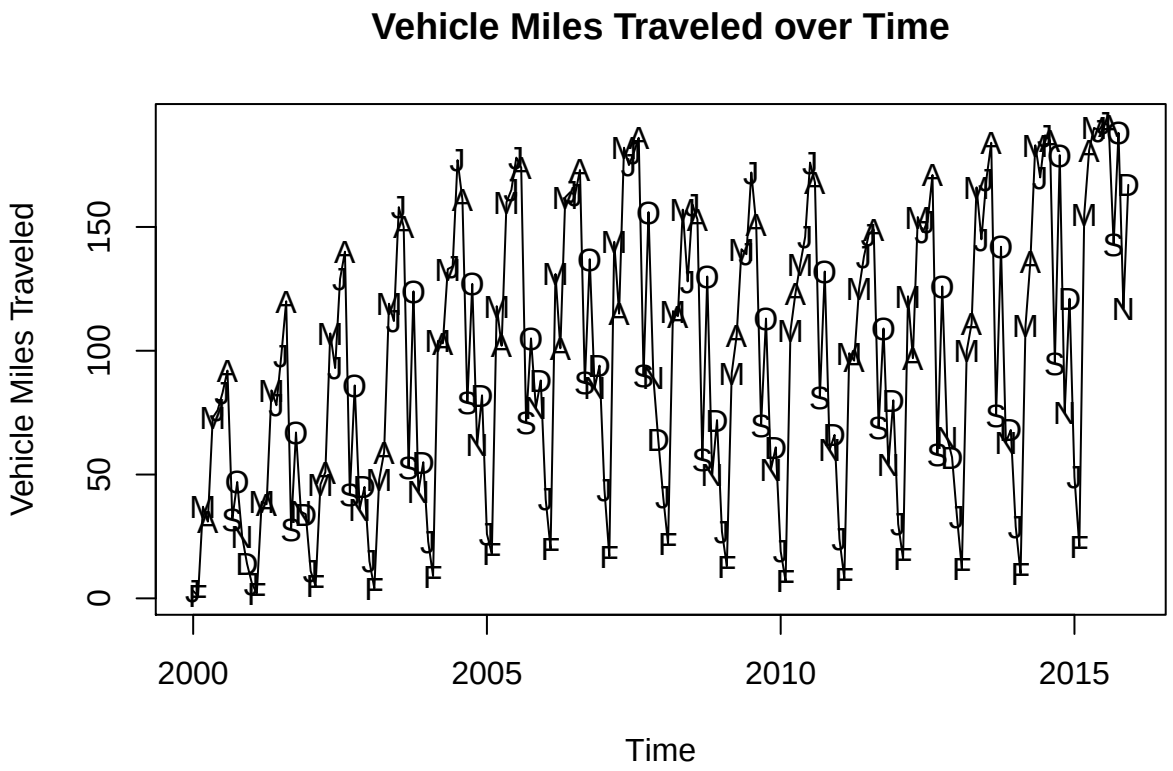

Figure 21: Time series plot the permit data with seasonal point markers.

Figure 21 shows the plot of the time series of the `vmt` data set. The trend of the data is cyclic. There is seasonality where there are peaks in August and trouhs in February.

Part B

Fit the regression function of the cosine model is found using the `lm` function. The residuals are found using the `resid` function

```
vmt.coslm <- lm(vmt~harmonic(vmt,1))
summary(vmt.coslm)$coefficients
```

```
##                                Estimate Std. Error    t value     Pr(>|t|)
## (Intercept)                  96.500000   2.763960  34.913678 2.386687e-84
## harmonic(vmt, 1)cos(2*pi*t) -56.892253   3.908829 -14.554806 1.107240e-32
## harmonic(vmt, 1)sin(2*pi*t)   4.402727   3.908829   1.126354 2.614437e-01
```

```
vmt.resid <- resid(vmt.coslm)
```

The coefficients of the regression function are all significant because their p-values are less than $\alpha = 0.05$ (and even $\alpha = 0.01$). This means that each coefficient is non-zero and can be used in the model.

Part C

Plot the residuals `permit.resid` using the `plot` function.

```
plot(vmt.resid,
     type = "o",
     ylab = "Raw Residuals",
     main = "Residuals of Vehicle Miles Traveled over Time")
```

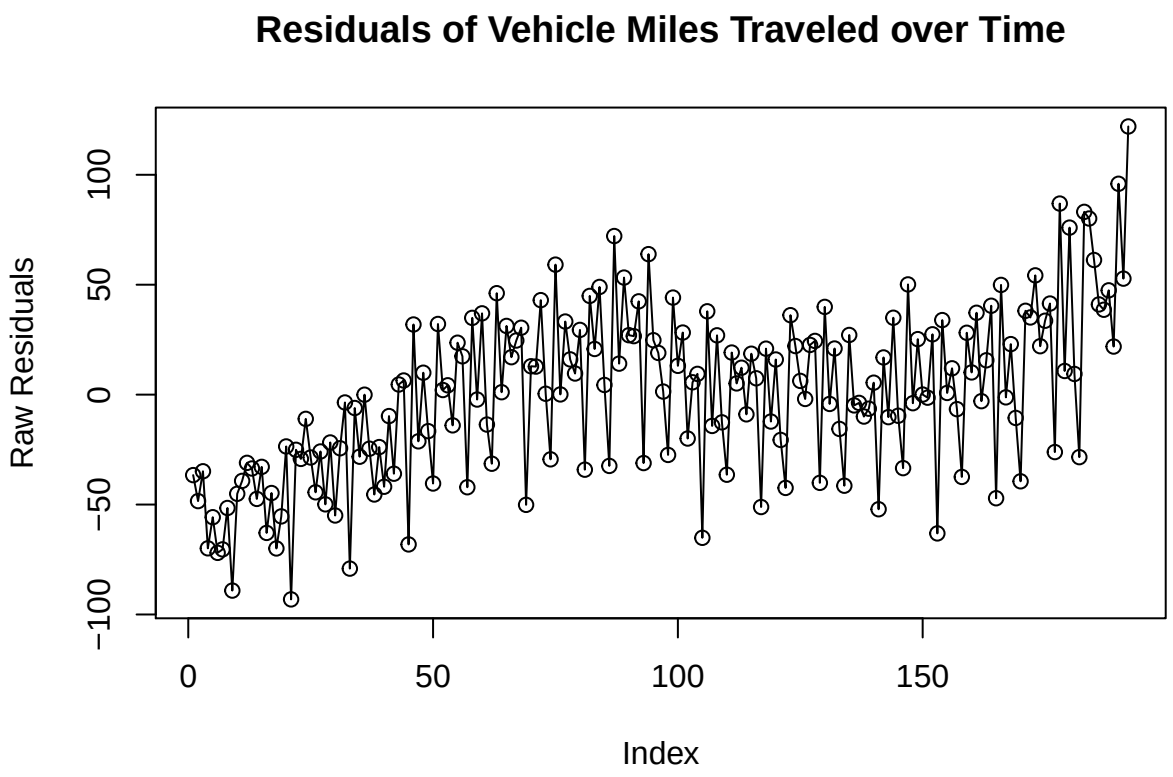

Figure 22: Residuals of Vehicle Miles Traveled over Time

The residual plot in Figure 16 has a positive trend over time. The trend is also a little sinusoidal. The variance of the residuals does not appear to be constant.

Part D

Fit the residuals to a regression model by using the `lm` funcitin.

```
vmt.resid.coslm <- lm(vmt.resid~time(vmt.resid)+time(vmt.resid)^2+time(vmt.resid)^3)
```

Plot the residuals against the fitted values by using the `plot` function.

```
plot(vmt.resid.coslm, which=1)
```

Find the summary of the residuals model by using the `summary` function.

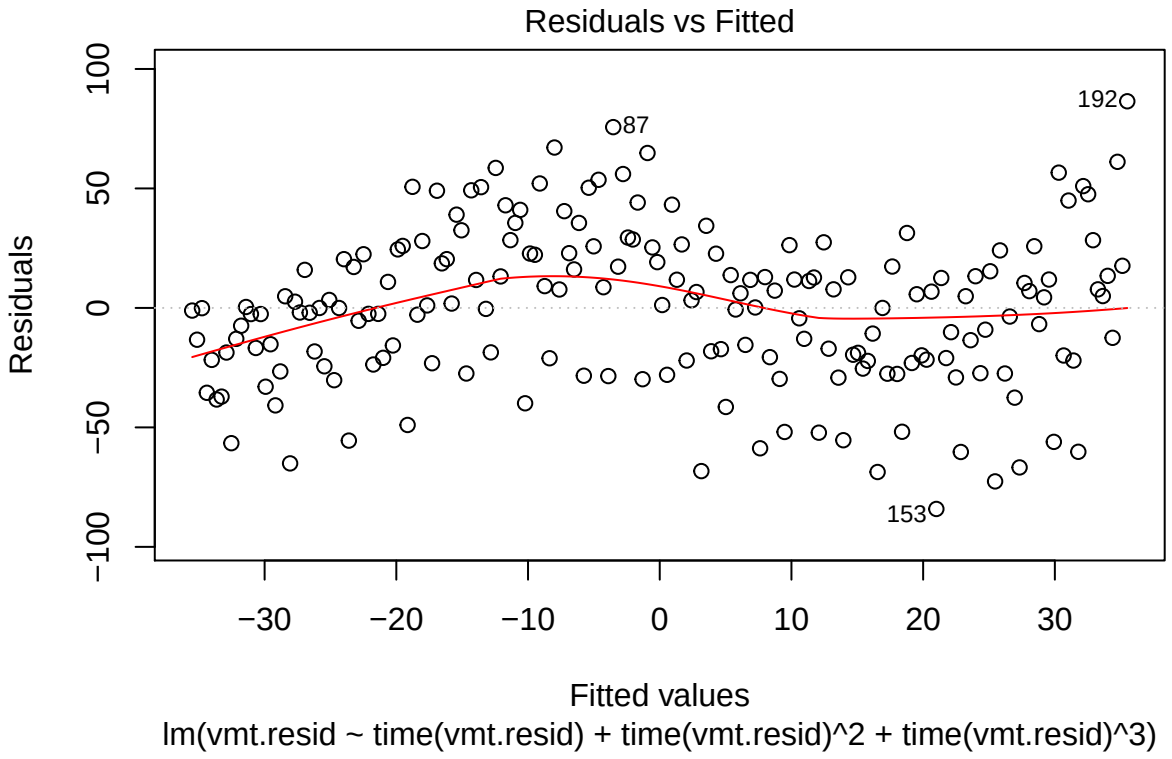

Figure 23: Plot of vmt residuals vs. fitted values \label{vmtresid.v.fitted

```
summary(vmt.resid.coslm)
```

```
##
## Call:
## lm(formula = vmt.resid ~ time(vmt.resid) + time(vmt.resid)^2 +
##     time(vmt.resid)^3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -84.136 -21.675   0.716  20.868  86.471
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -35.87250    4.65067  -7.713 6.71e-13 ***
## time(vmt.resid)   0.37174    0.04179   8.895 4.56e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32.1 on 190 degrees of freedom
## Multiple R-squared:  0.294,  Adjusted R-squared:  0.2903
## F-statistic: 79.12 on 1 and 190 DF,  p-value: 4.558e-16
```

(i)

The first coefficient of the cubic trend is significant because it has a p-value much less than $\alpha = 0.05$. However, neither coefficients two or three are significant because they have p-values greater than $\alpha = 0.05$. ### (ii)

The proportion of variability of the cubic trend is $R^2 = 0.294$.

Part E

(i)

Check if the error process has mean equal zero.

```
mean(fitted(vmt.resid.coslm))
```

```
## [1] 3.985961e-15
```

By a rounding error, the mean is effectively zero. Therefore the assumption holds.

(ii)

Check if the errors are independent.

```
runs(rstudent(vmt.resid.coslm))[1]
```

```
## $pvalue
## [1] 0.938
```

The runs test suggests independence in the errors beacuse the p-value is greater than $\alpha = 0.05$.

(iii)

Plot the fitted residuals using the `plot` function to see if the error is homogenous.

```
plot(fitted(vmt.resid.coslm),
     type = "o",
     xlab = "Index",
     ylab = "Fitted Residuals",
     main = "Plot of Fitted Residuals")
```

Based on the plot in Figure 24 the errors are not homogenous.

(iv)

Plot the normal probability plot of the residuals linear model by using the `plot` funciton.

```
plot(vmt.resid.coslm, which=2)
```

Create a histogram of the fitted regression errors with the `hist` function.

```
hist(fitted(vmt.resid.coslm),
     xlab = "Fitted Linear Error Process",
     main = "Histogram of Fitted Linear Error Process")
```

Based on the normal probability plot and the histogram in Figure 19 and **??**, the fitted errors appear to be normally distributed.

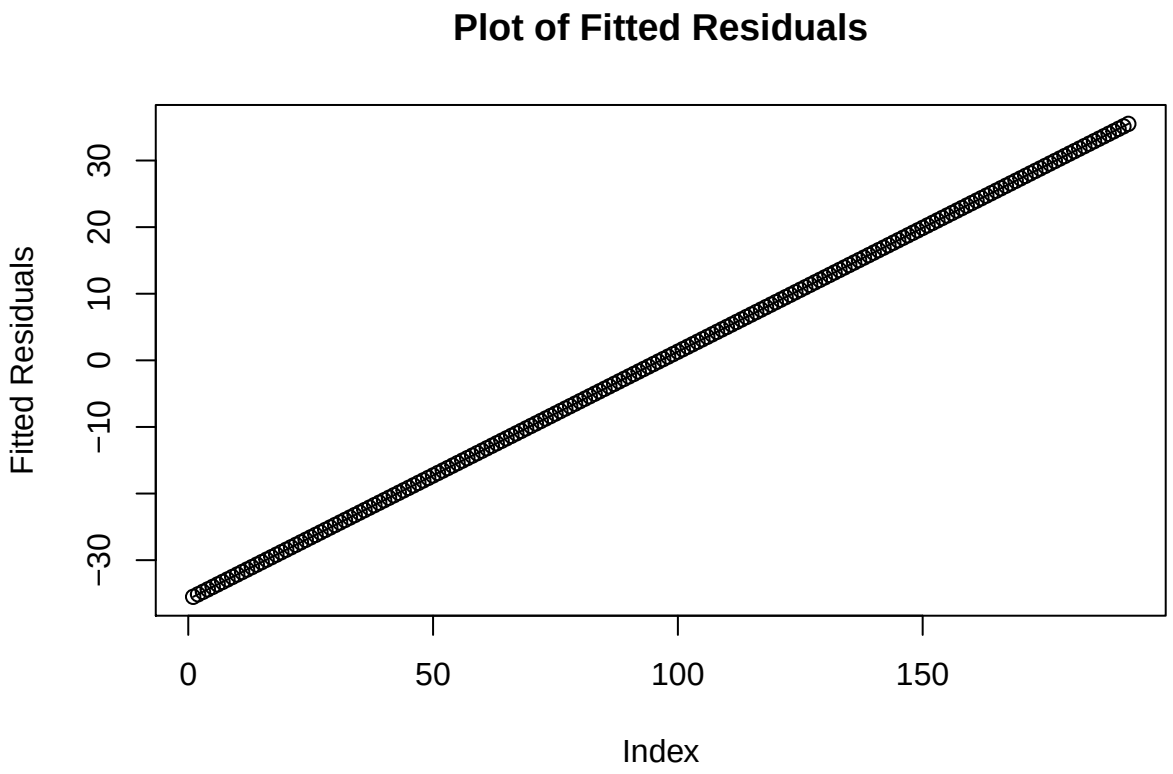

Figure 24: Plot of Fitted Residuals for Permit Data

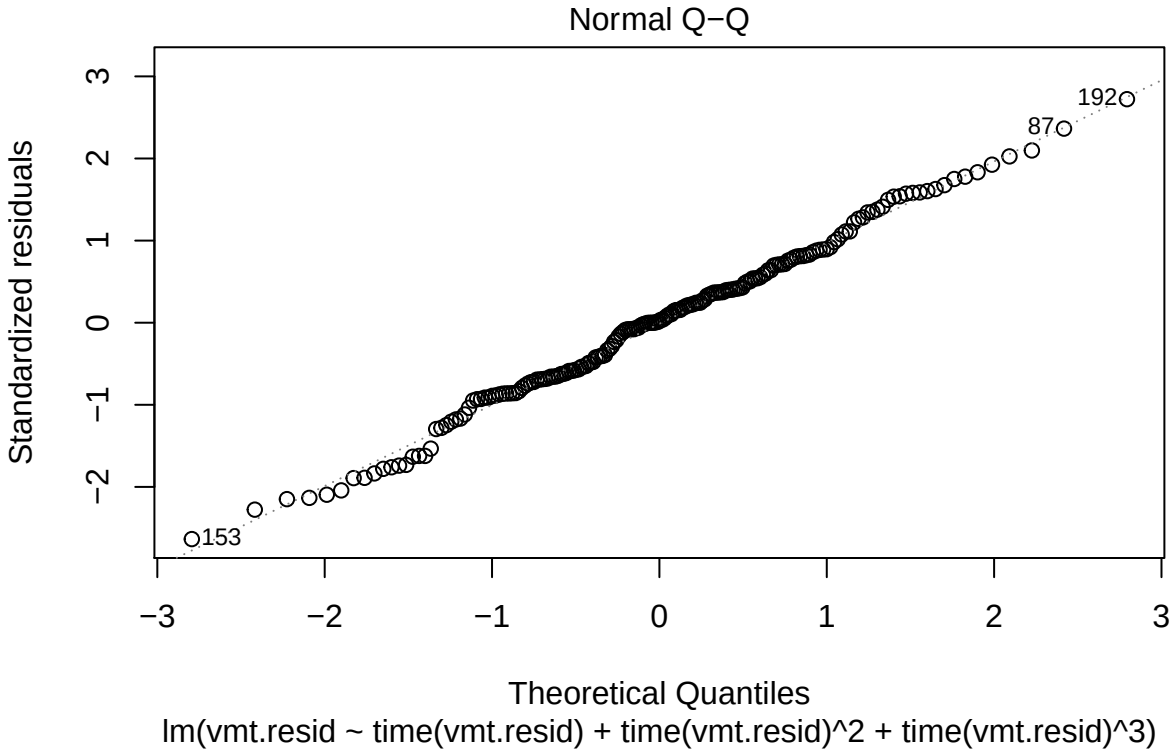

Figure 25: Normal Probability Plot of Permit Residuals

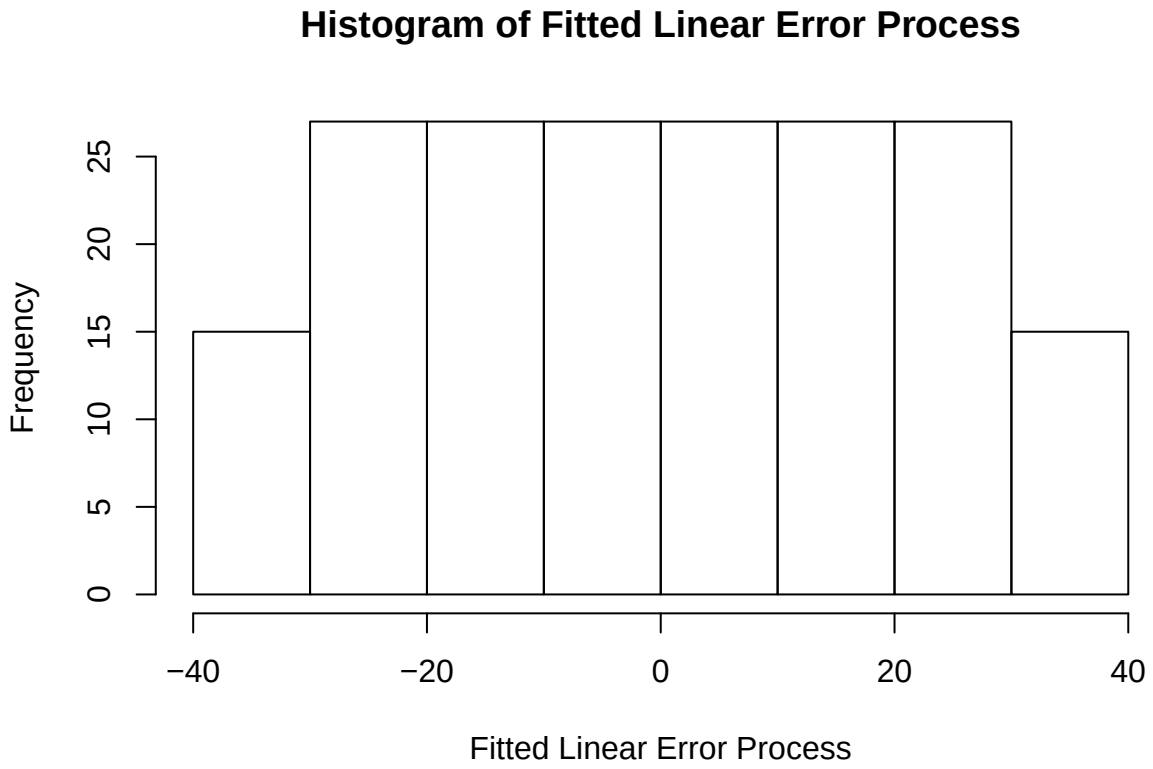

Figure 26: Histograpm of Permit Residuals