TS - Module 2 - AR(I)MA

```
library(TSA)
```

```
## Warning: package 'TSA' was built under R version 4.0.3
```

```
##
## Attaching package: 'TSA'
```

```
## The following objects are masked from 'package:stats':
##
##     acf, arima
```

```
## The following object is masked from 'package:utils':
##
##     tar
```

```
library(mgcv)
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-31. For overview type 'help("mgcv-package")'.
```

```
library(lmtest)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
data = read.csv('chart_monthly.csv',header = TRUE)
head(data)
```

```
##     month count
## 1 1965-01    23
## 2 1965-02    29
## 3 1965-03    25
## 4 1965-04    23
## 5 1965-05    31
## 6 1965-06    21
```

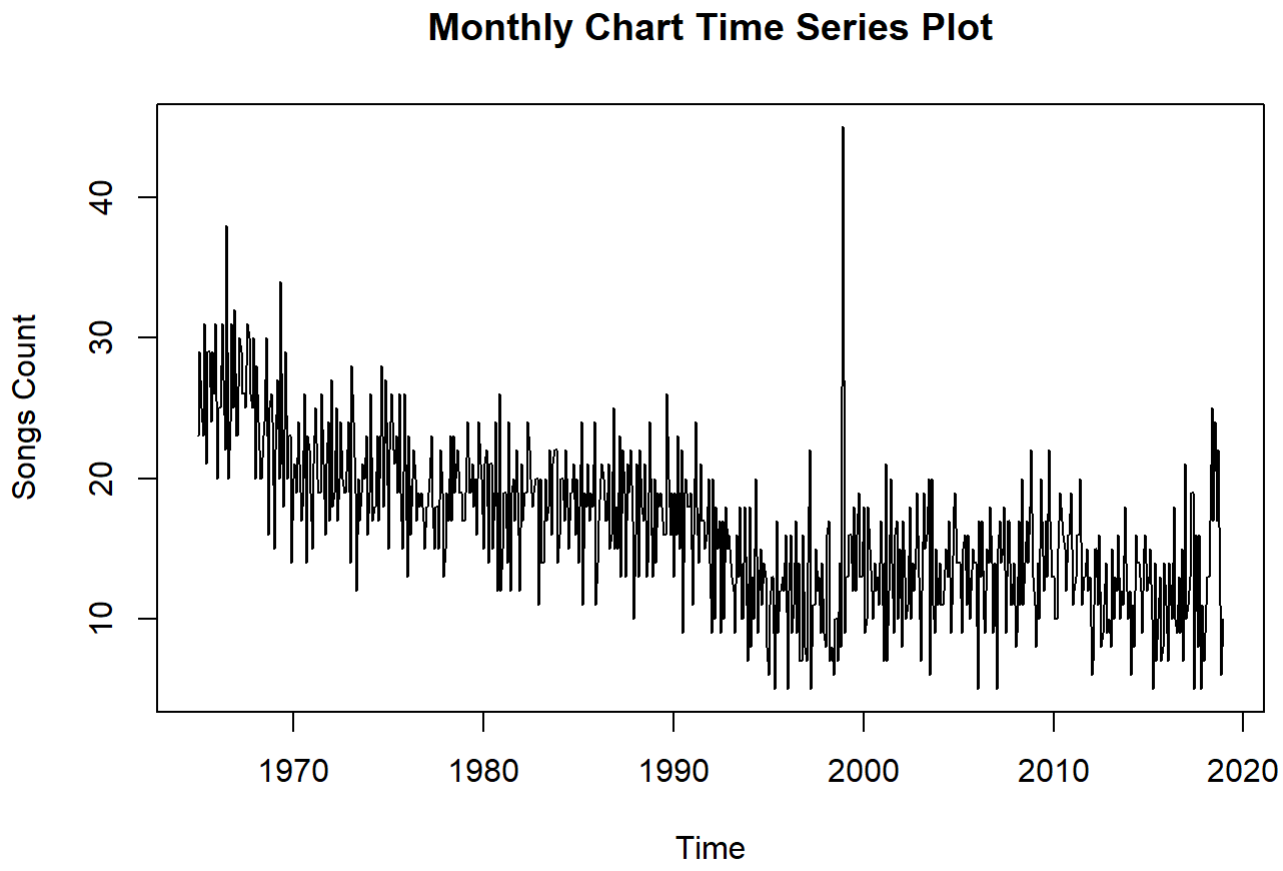
```
dates = as.Date(as.yearmon(data$month))
data_1 = data
data_1$date = dates

data = data[,2]

chart = ts(data,start=c(1965,1),freq=12)

chart.dif = diff(chart)
```

1a. Exploratory Analysis

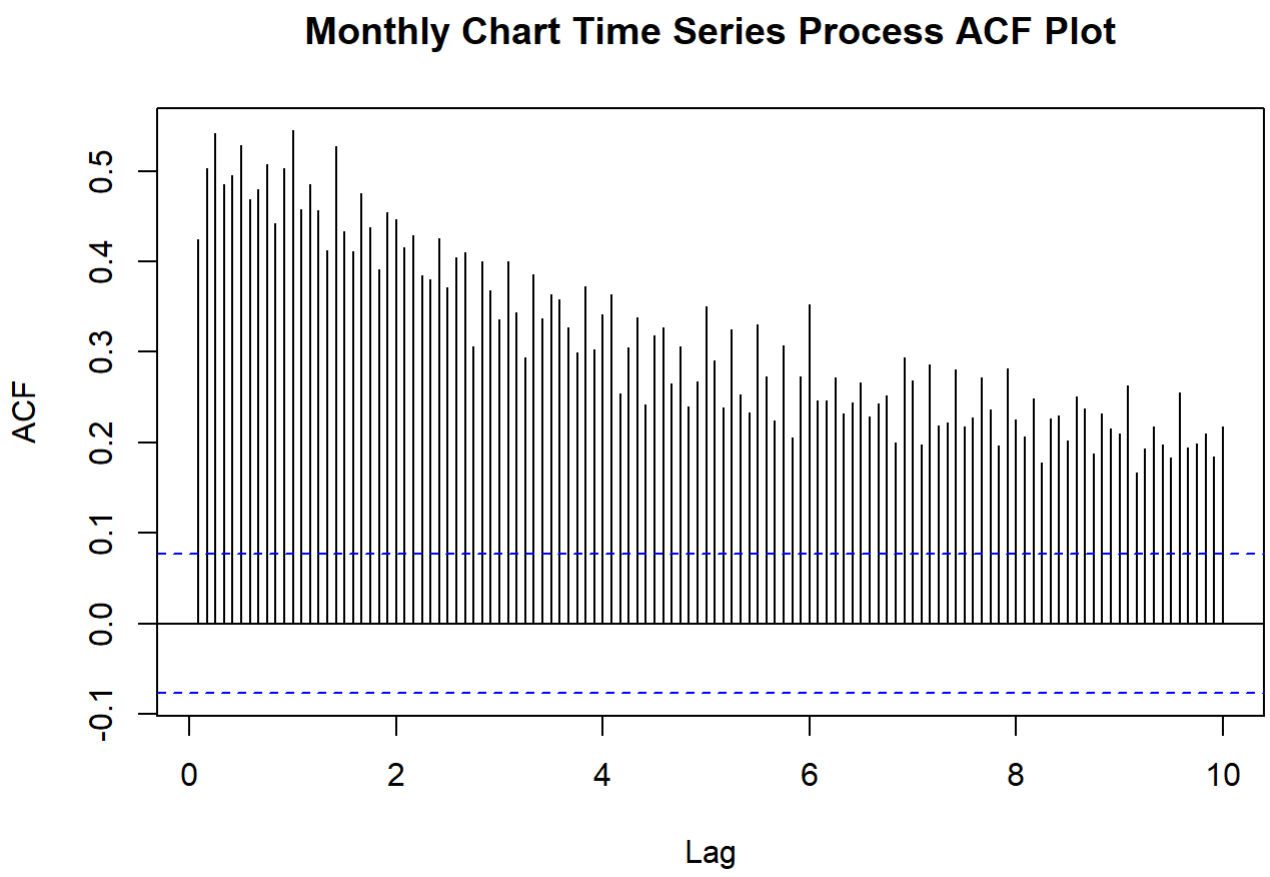
i. Plot the Time Series and ACF plots. Comment on the main features, and identify what (if any) assumptions of stationarity are violated.

ii. Perform a differencing on the data, and perform the same analysis. How do assumptions of stationarity hold for the differenced data? Do you expect the differencing data be suitable for ARMA forecasting?

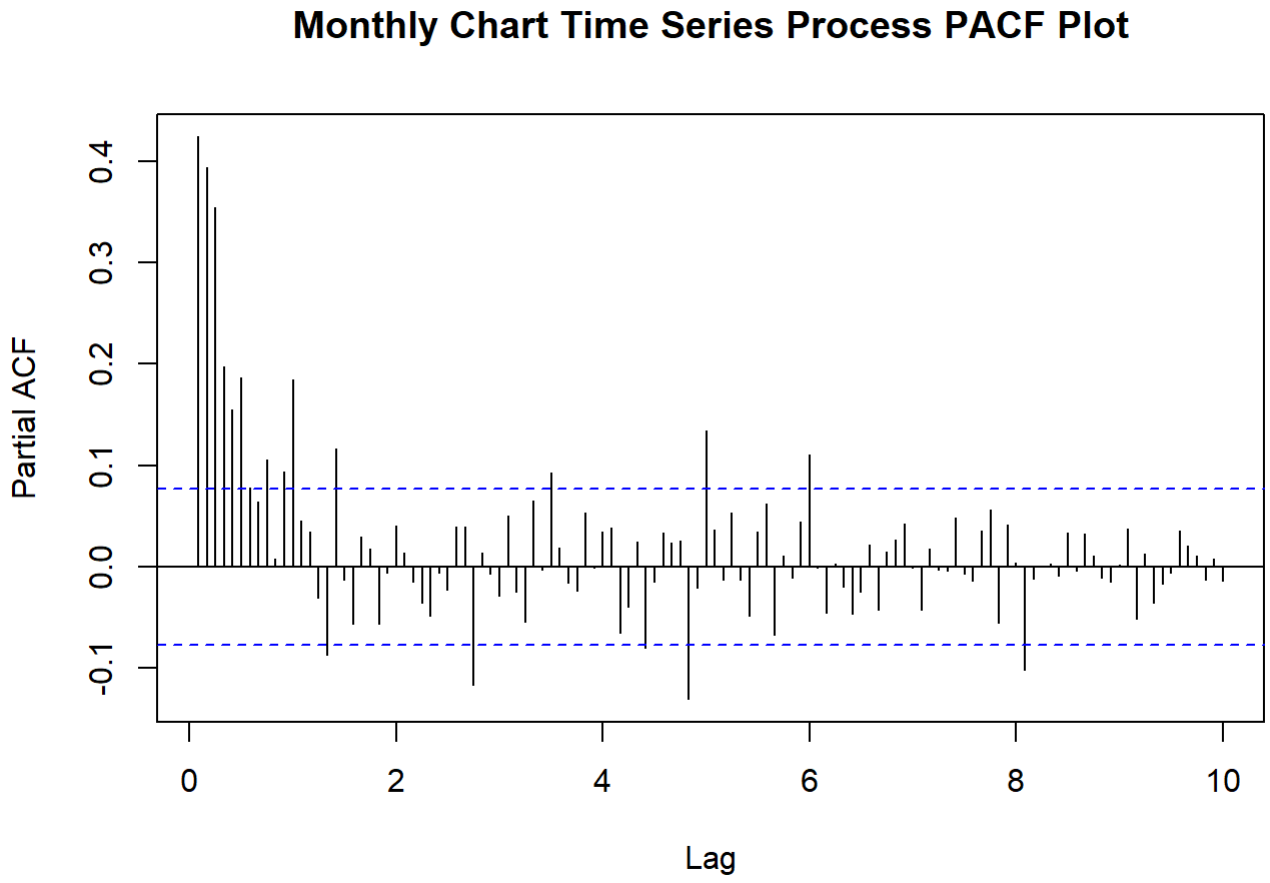
```
# Plot original time series data
ts.plot(chart, ylab = "Songs Count", main = "Monthly Chart Time Series Plot")
```



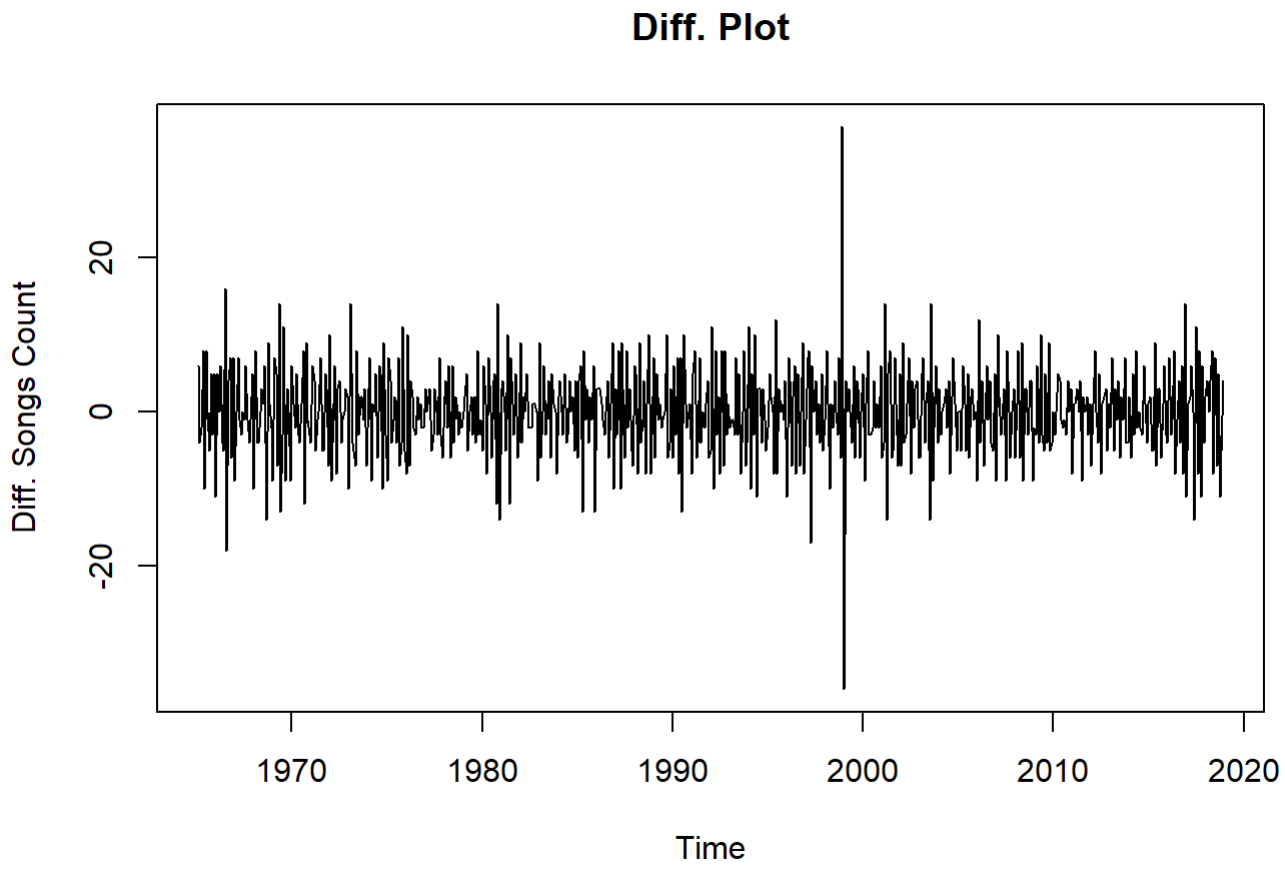

```
acf(chart,lag.max=12*10, main="Monthly Chart Time Series Process ACF Plot")
```



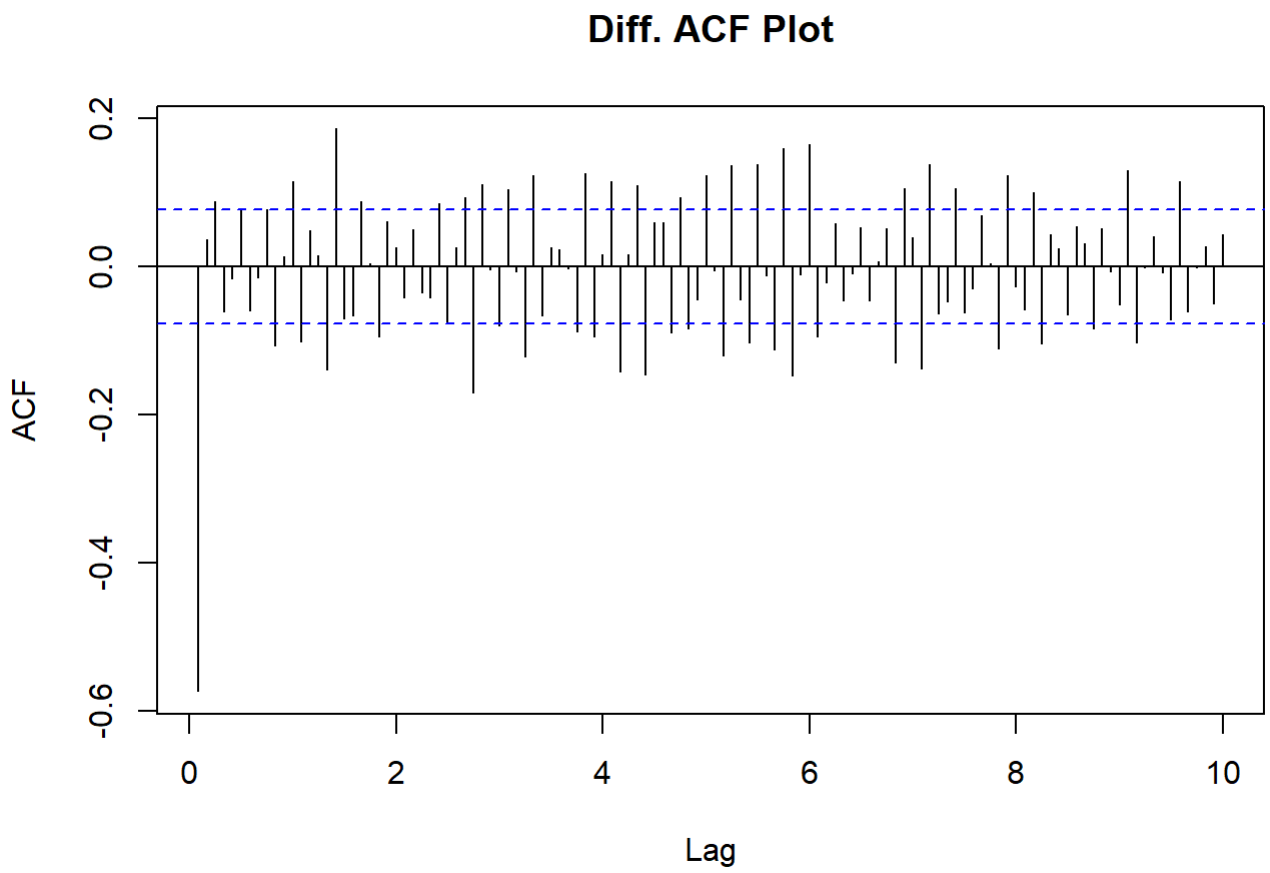

```
pacf(chart,lag.max=12*10, main="Monthly Chart Time Series Process PACF Plot")
```



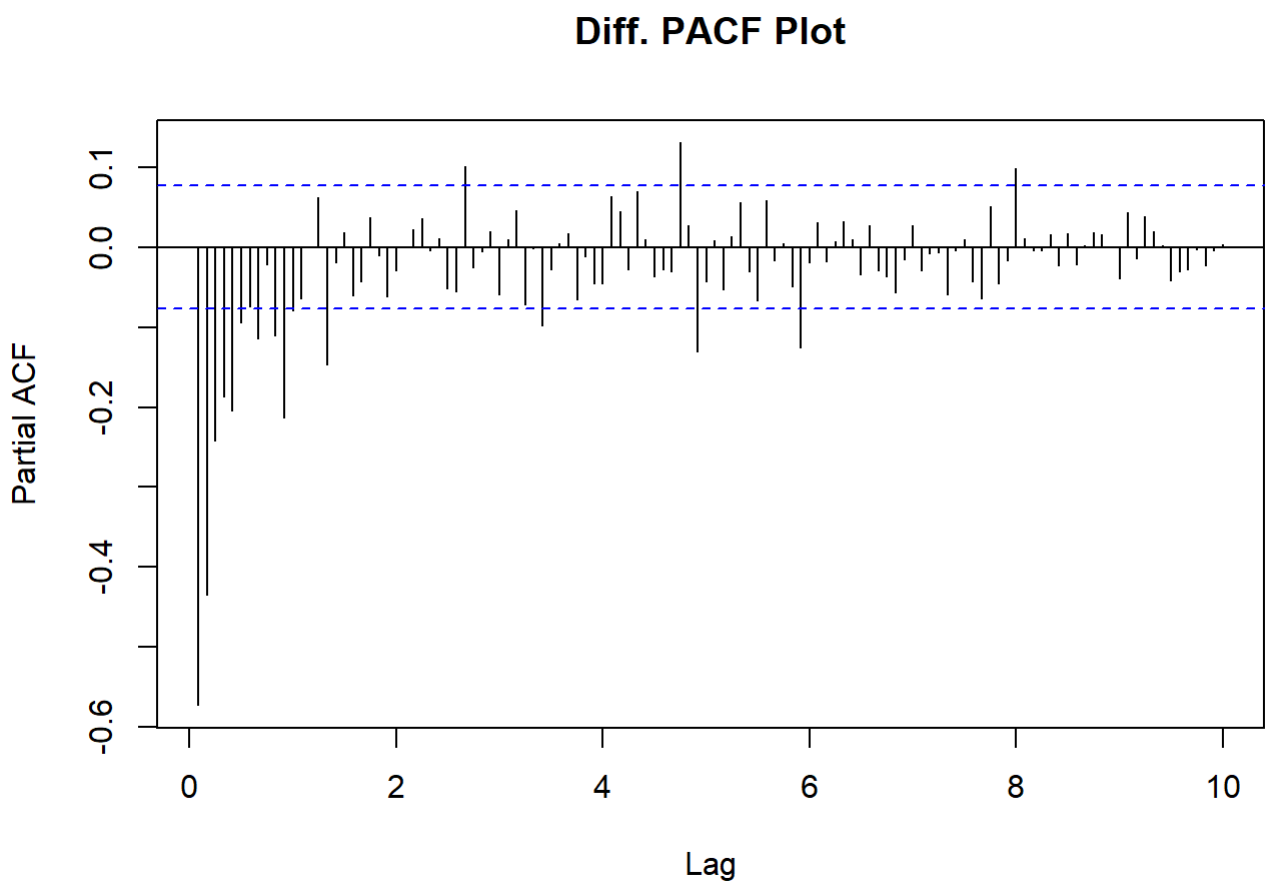

```
# Plot differenced time series data
ts.plot(chart.dif, ylab = "Diff. Songs Count", main = "Diff. Plot")
```




```
acf(chart.dif,lag.max=12*10, main="Diff. ACF Plot")
```




```
pacf(chart.dif,lag.max=12*10, main="Diff. PACF Plot")
```



Answer

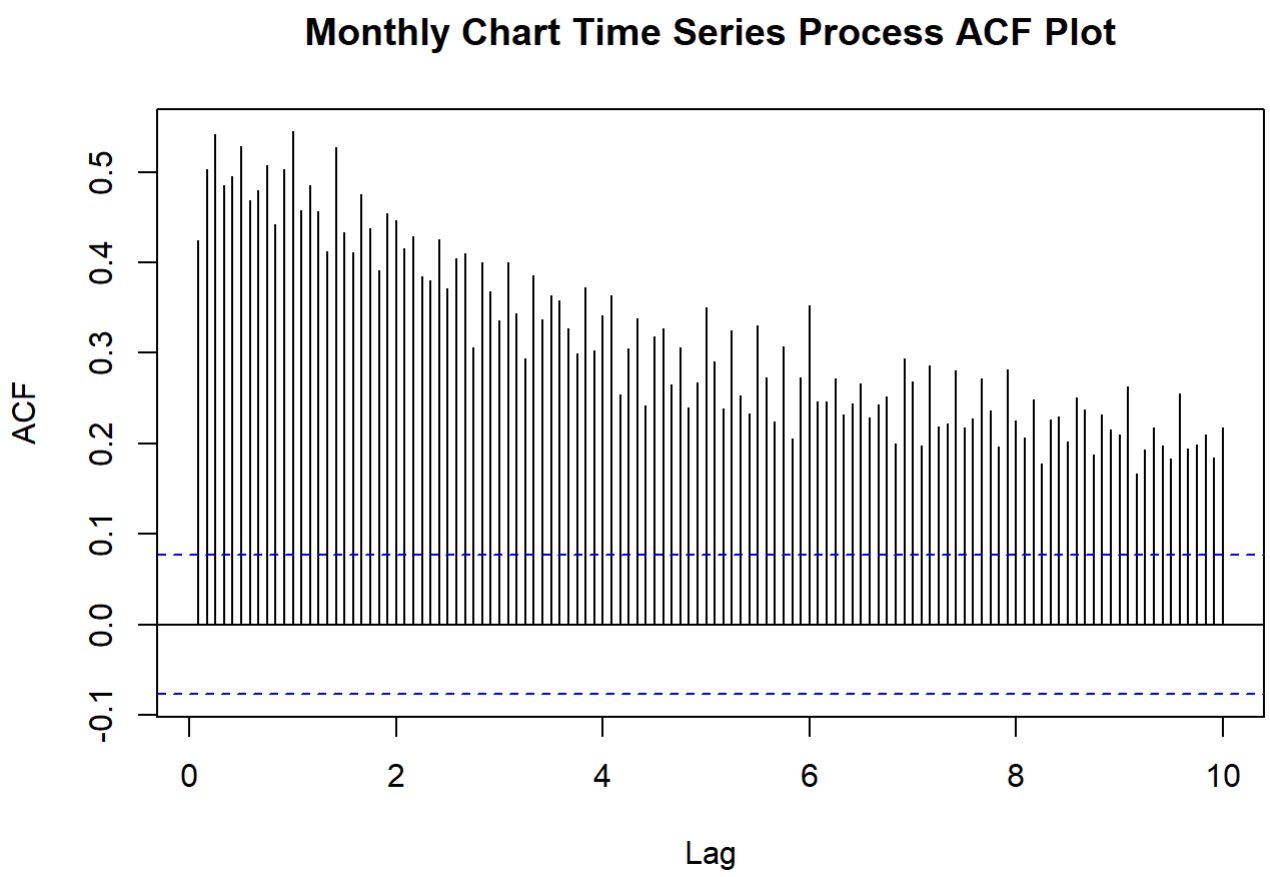
i. Looking at the original time series process, we can observe a downward trend and a cyclical pattern. While the original process indicates a finite and constant variance (other than a spike before the year 2000), the process' mean is not constant due to the trend component which is a violation of stationarity assumption. A stationary process is described by the behavior of its auto-covariance or auto-correlation function. Here we are plotting the auto-correlation (ACF) and the partial auto-correlation plots of the time series process. We can identify the downward trend in the ACF plot by the slow decay in the auto-correlation values. In addition, the auto-correlation values are large and outside of the significance bands, and thus we can conclude that the original time series process is non-stationary.

ii. The Diff. plot indicates that the differenced data has both constant mean (meaning trend has been removed), and finite and constant variance (other than the spike around the year 200). Moreover, the Diff. ACF plot confirms the trend has been removed, but despite the fact that the auto-correlation values are small, other than the value at lag 1 which is expected to be high in absolute value, they are outside the significance bands for many of the lags included in the plot, an indication of non-stationary process. Hence, since modeling an ARMA process requires stationarity, I do not expect the differencing data will be suitable for ARMA forecasting. ***Note however that while the auto-correlation values of the differenced process are outside the confidence bands, they are indeed small and very close to the bands which might suggest that the process could be weakly stationary and suitable for ARMA modeling, but will required further investigation.***

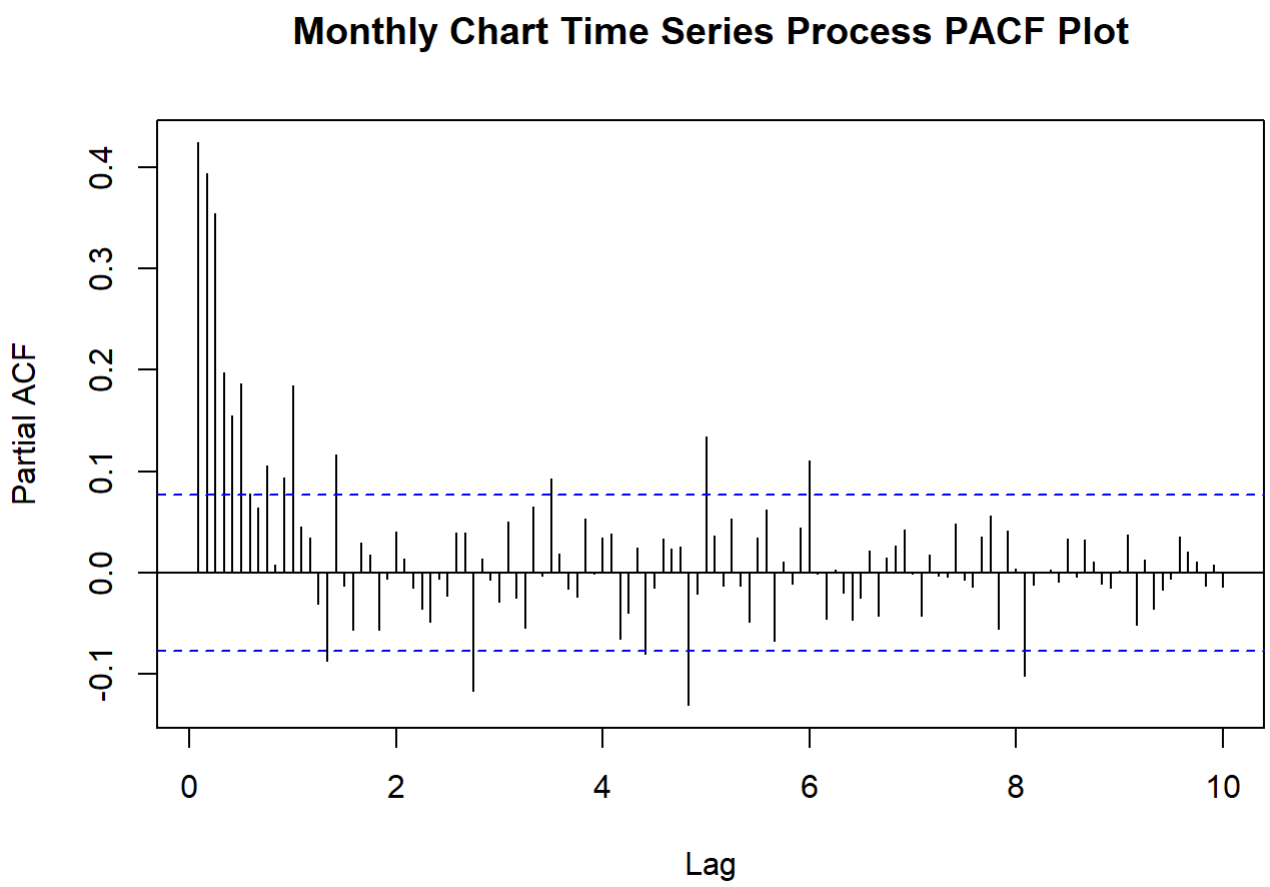
1b. ARIMA Modelling

Using graphical analysis of ACF and PACF plots as well as the iterative approach, fit the an ARIMA(p,d,q) model with max order = 3, max differencing = 1. Using an AIC significance threshold of 2, choose which order to use and explain your reasoning for choosing it. Evaluate the model residuals with relevant plots and tests.

```
# Plot ACF and PACF of original process
acf(chart,lag.max=12*10, main="Monthly Chart Time Series Process ACF Plot")
```




```
pacf(chart,lag.max=12*10, main="Monthly Chart Time Series Process PACF Plot")
```


Monthly Chart Time Series Process PACF Plot
Partial ACF
-0.1
0.0
0.1
0.2
0.3
0.4
0
2
4
6
8
10
Lag

```
test_modelA = function(ts_data, max_p, max_d, max_q){
  orders = data.frame()
  for (p in 0:max_p){
    for (d in 0:max_d){
      for (q in 0:max_q) {
        possibleError <- tryCatch({
          mod = arima(ts_data, order=c(p,d,q), method="ML")
          current.aic = AIC(mod)
          orders<-rbind(orders,c(p,d,q,current.aic))
        },
          error=function(e) e
        )
        if(inherits(possibleError, "error")) next

      }
    }
  }
  names(orders) <- c("p","d","q","AIC")
  return(orders[order(orders$AIC),])
}

max_p = 3
max_q = 3
max_d = 1
ts_data = chart

# Observe first best models based on smallest AICs
orders = test_modelA(ts_data, max_p, max_d, max_q)
```

```
## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1
```

```
## Warning in log(s2): NaNs produced
```

```
## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1
```

```
head(orders)
```

```
##    p d q      AIC
## 32 3 1 3 3612.688
## 28 3 0 3 3636.772
## 7  0 1 2 3646.283
## 22 2 1 1 3647.237
## 30 3 1 1 3647.687
## 23 2 1 2 3648.002
```

```
# Optimal order
opt_p = orders[1,1] # 3
opt_d = orders[1,2] # 1
opt_q = orders[1,3] # 3

# Fit the optimal model
opt_model = arima(chart,order=c(opt_p,opt_d,opt_q), method="ML")
```

```
## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1
```

```
print(paste("Oprtimal p order is: ", opt_p))
```

```
## [1] "Oprtimal p order is:  3"
```

```
print(paste("Oprtimal d order is: ", opt_d))
```

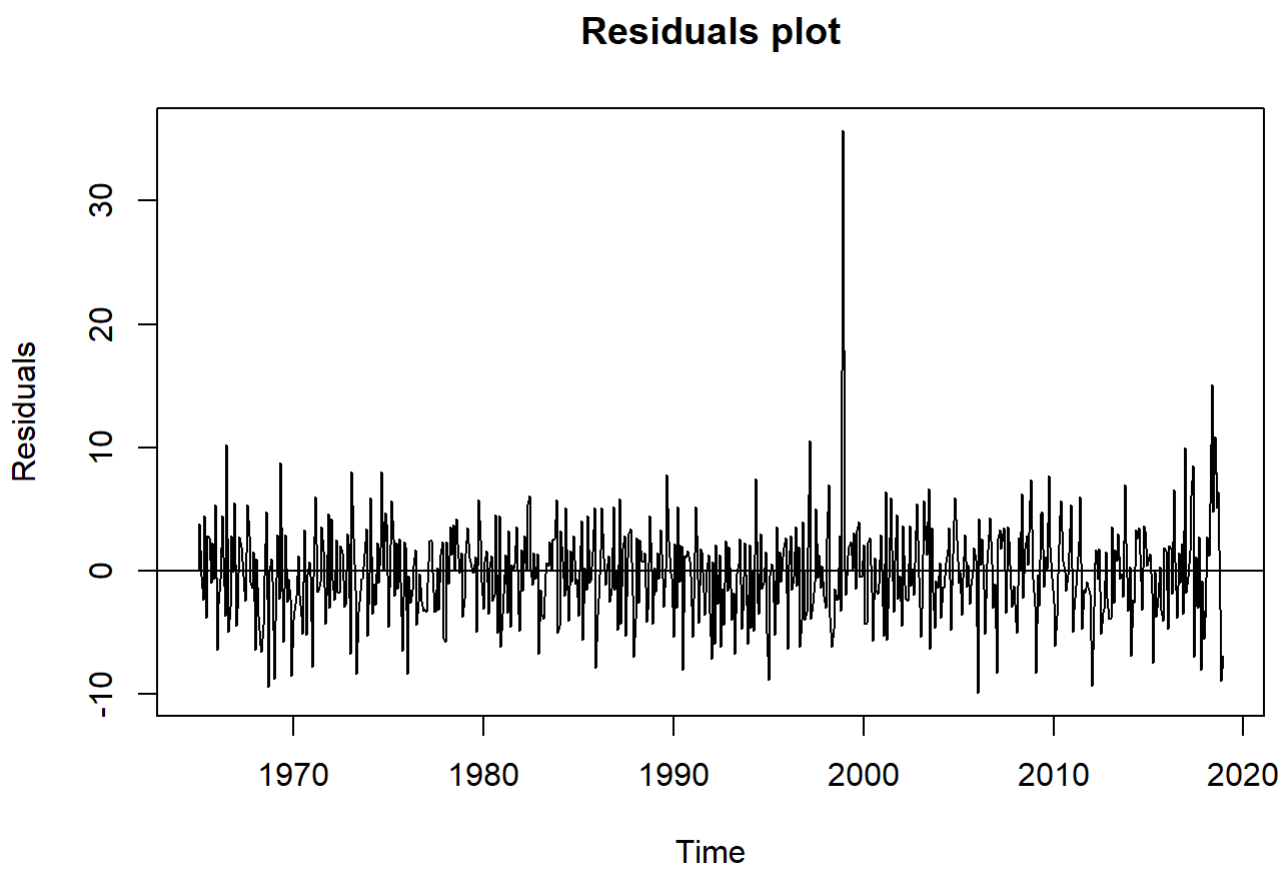
```
## [1] "Oprtimal d order is:  1"
```

```
print(paste("Oprtimal q order is: ", opt_q))
```

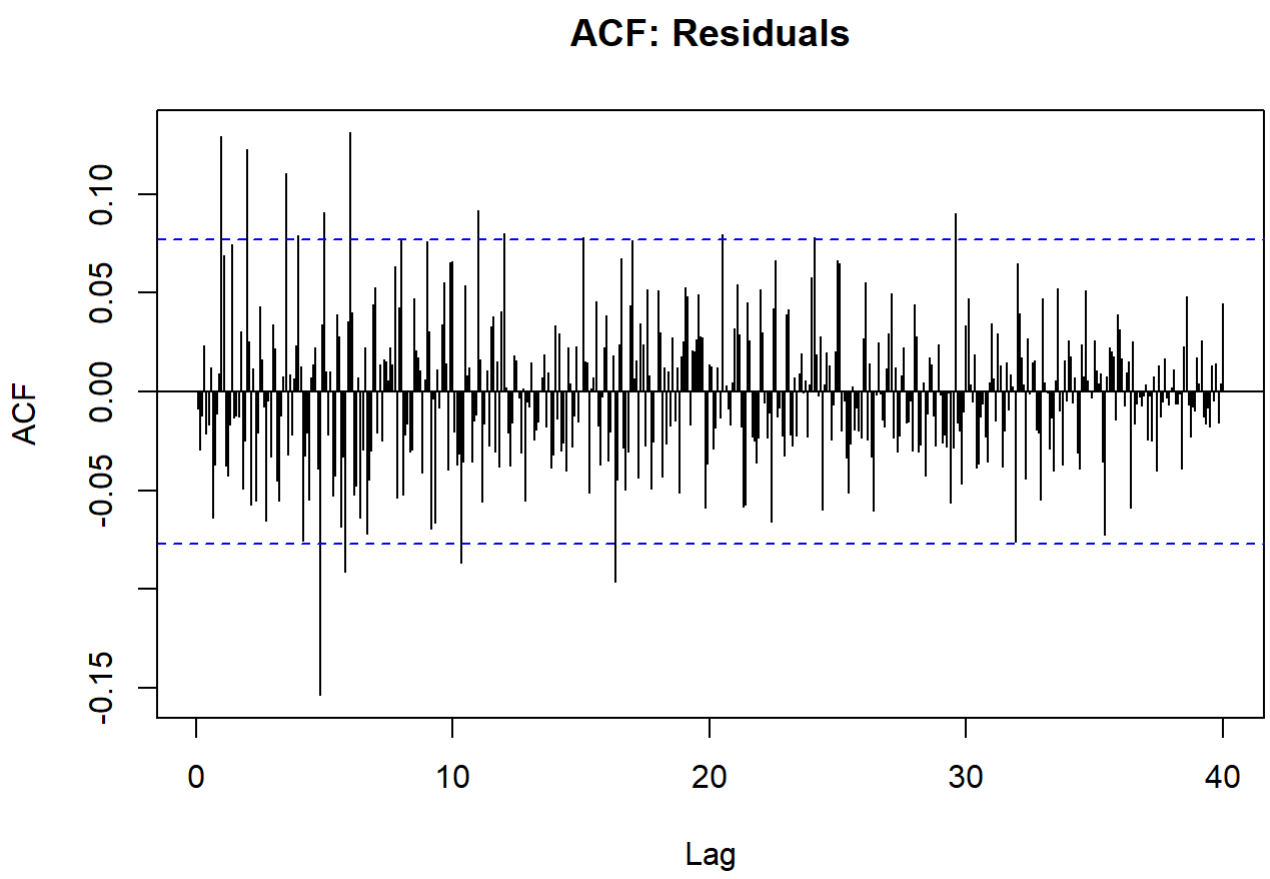
```
## [1] "Oprtimal q order is:  3"
```

Evaluate the model residuals with relevant plots and tests.

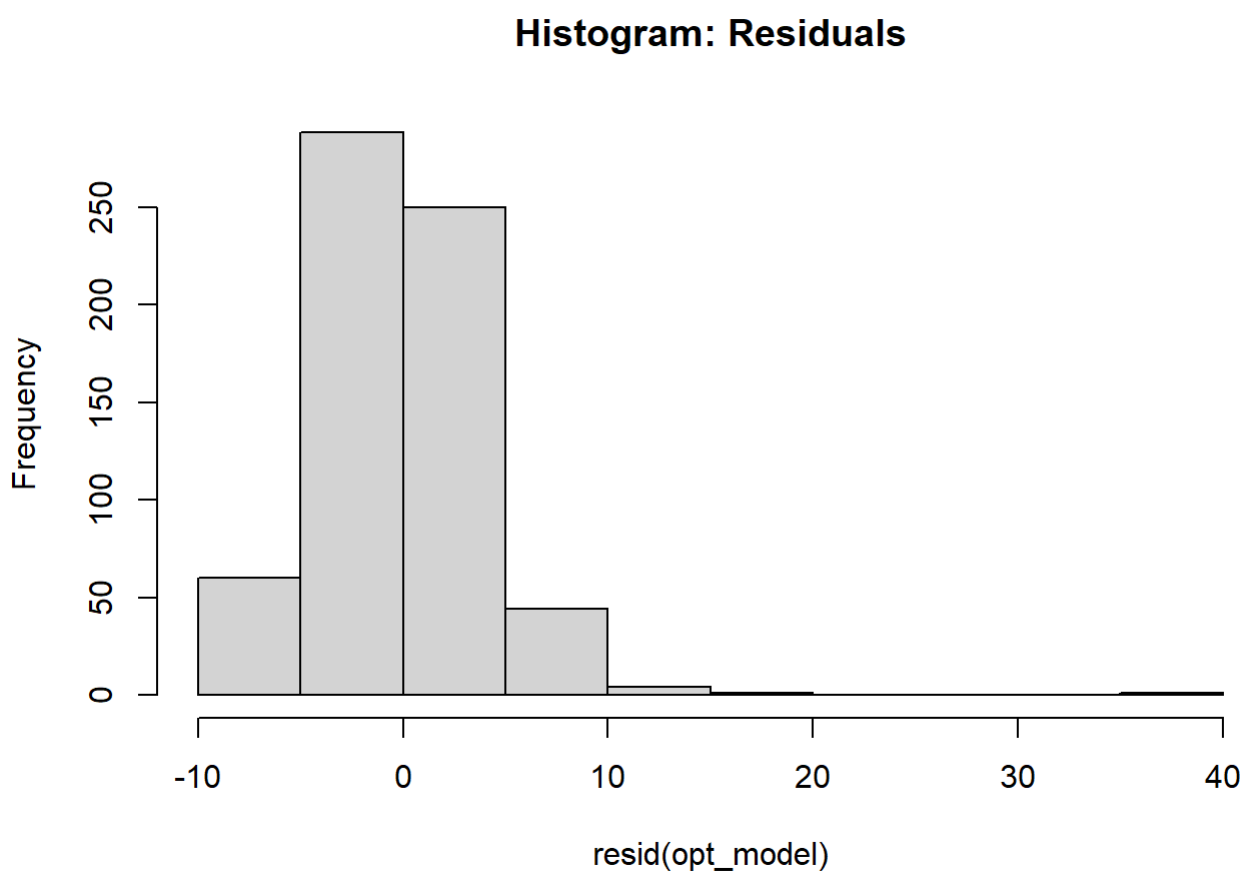
```
# Plot residuals of optimal model
plot(resid(opt_model), ylab="Residuals", main="Residuals plot")
abline(h=0)
```



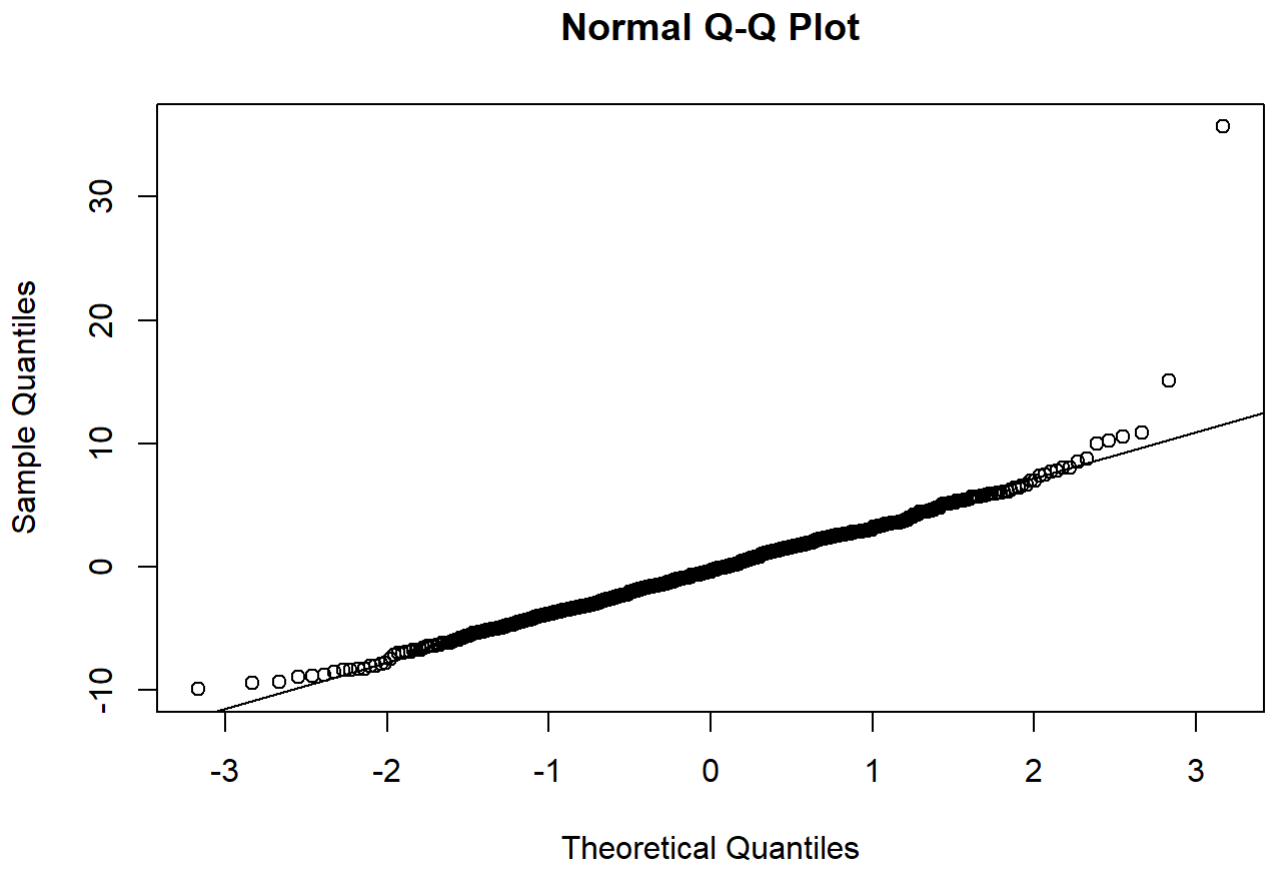

```
acf(resid(opt_model), lag.max=12*40, main="ACF: Residuals")
```




```
hist(resid(opt_model), main="Histogram: Residuals")
```




```
qqnorm(resid(opt_model))
qqline(resid(opt_model))
```




```
# Test for uncorrelated residuals
#Null hypothesis is that the residuals are Uncorrelated. Hence want to see a larg p-value
# lag = p + q + 1; fitdf = p + q (disregard the d order on both lag and fitdf)
Box.test(resid(opt_model), lag = (opt_p+opt_q+1), type = "Box-Pierce", fitdf = (opt_p+opt_q))
```

```
##
##  Box-Pierce test
##
## data:  resid(opt_model)
## X-squared = 1.623, df = 1, p-value = 0.2027
```

Answer

We started with the Original process ACF and PACF plots. As indicated in answer 1a, we concluded that the process is not stationary. Based on the ACF and PACF plots, since the the ACF tails-off and the PACF cuts-off at around lag 8, we might suggest an AR(8) model, ***but since the original process is an ARIMA process, we cannot determine order*** p ***and order*** q ***based on the ACF and PACF plots.*** We than perform and AIC comparison of different models with max order p and q equal to 3 and max order d equal to 1. Based on this AIC analysis, we determined that that optimal model is order p = 3, order d = 1, order q = 3, or in short ARIMA(3,1,3), with an AIC of 3612.688. When we evaluate the residuals of the ARIMA(3,1,3) model, we can see the residuals have constant mean and constant variance (other than one spike before the year 2000). The residuals ACF plot indicates that they are white noise since for the most part all values are within the confidence bands and ~ 0. We

used Maximum Likelihood and hence assumed normality. The Histogram and Q-Q plots investigate the assumption of normality. The residuals show only very little skewness, but overall look normal-distributed. Also, based on the Box-Pierce test's high p-value of 0.2027, we conclude that the residuals are uncorrelated.

1c. Forecasting

Build an ARIMA(2,1,4) model. Display model coefficients, comment on anything that stands out about them, and write out the model formula in full form. Then, keep the last 6 data points for testing. Generate forecasts of those 6 months and compare the predicted values to the actual ones. Include 95% confidence interval for the forecasts and provide plots. Calculate Mean Absolute Prediction Error (MAE), Mean Absolute Percentage Error (MAPE), and Precision Measure (PM); comment on the accuracy of predictions.

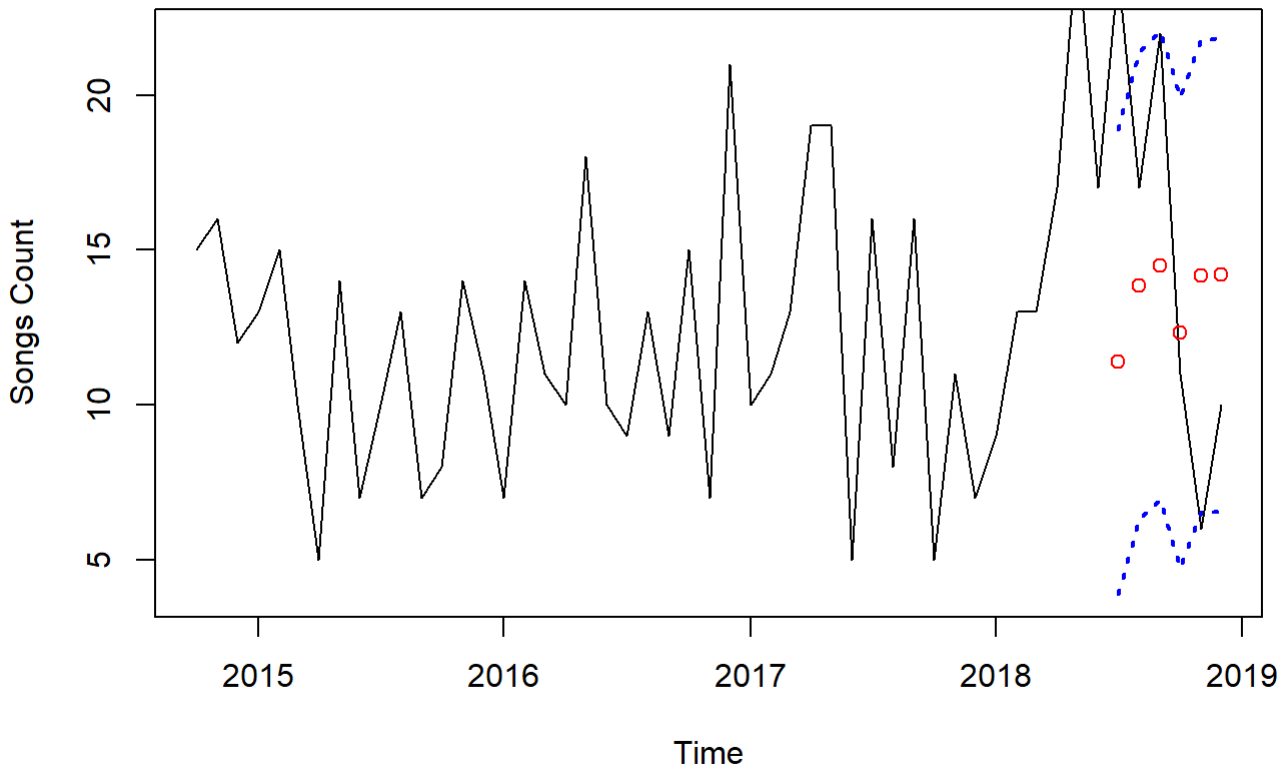
```
# Fit ARIMA(2,1,4) model
model = arima(chart,order=c(2,1,4), method="ML")
model
```

```
##
## Call:
## arima(x = chart, order = c(2, 1, 4), method = "ML")
##
## Coefficients:
##           ar1      ar2     ma1      ma2      ma3     ma4
##       -1.1589  -0.9993  0.0912  -0.0963  -0.8896  0.1598
## s.e.   0.0017   0.0011  0.0397   0.0186   0.0187  0.0387
##
## sigma^2 estimated as 15.05:  log likelihood = -1799.39,  aic = 3610.78
```

```
coeftest(model)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error   z value  Pr(>|z|)
## ar1 -1.1588756  0.0016674 -695.0250 < 2.2e-16 ***
## ar2 -0.9992763  0.0011326 -882.2592 < 2.2e-16 ***
## ma1  0.0911947  0.0396880    2.2978   0.02157 *
## ma2 -0.0963333  0.0185708   -5.1874 2.133e-07 ***
## ma3 -0.8895629  0.0187035  -47.5613 < 2.2e-16 ***
## ma4  0.1598070  0.0386777    4.1318 3.600e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
n = length(chart)
nfit = n-6

pred_model = arima(chart[1:nfit], order = c(2,1,4),method = "ML")
pred = predict(pred_model,n.ahead=6)
ubound = pred$pred+1.96*pred$se
lbound = pred$pred-1.96*pred$se
ymin = min(lbound)
ymax = max(ubound)

plot(data_1$date[(n-50):n], chart[(n-50):n], type="l", ylim=c(ymin,ymax), xlab="Time", ylab="Son
gs Count")
points(data_1$date[(nfit+1):n], pred$pred,col="red")
lines(data_1$date[(nfit+1):n], ubound,lty=3,lwd= 2, col="blue")
lines(data_1$date[(nfit+1):n], lbound,lty=3,lwd= 2, col="blue")
```




```
obs_count = chart[(nfit+1):n]
pred_count = pred$pred

print(paste("The observed values are: ", data.frame(obs_count)))
```

```
## [1] "The observed values are:  c(24, 17, 22, 11, 6, 10)"
```

```
print(paste("The predicted values are: ", data.frame(pred_count)))
```

```
## [1] "The predicted values are:  c(11.3961213087569, 13.8379080019396, 14.4822062618425, 12.33
91305198433, 14.1787351483795, 14.1885076822548)"
```

```
## Compute Accuracy Measures

### Mean Absolute Prediction Error (MAE)
MAE = mean(abs(pred_count-obs_count))

### Mean Absolute Percentage Error (MAPE)
MAPE = mean(abs(pred_count-obs_count)/obs_count)

### Precision Measure (PM)
PM = sum((pred_count-obs_count)^2)/sum((obs_count-mean(obs_count))^2)

print(paste("Mean Absolute Prediction Error (MAE) is: ", MAE))
```

```
## [1] "Mean Absolute Prediction Error (MAE) is:  6.16502296298977"
```

```
print(paste("Mean Absolute Percentage Error (MAPE) is: ", MAPE))
```

```
## [1] "Mean Absolute Percentage Error (MAPE) is:  0.492766225349615"
```

```
print(paste("Precision Measure (PM) is: ", PM))
```

```
## [1] "Precision Measure (PM) is:  1.21719680600809"
```

Answer

We started with fitting an ARIMA(2,1,4) model. Looking at the coefficients test, we can see all of them are significant at the 95% level since all of the coefficients' p-value is smaller than 0.05. The ar1 and ar2 are the AR process coefficients and the coefficients ma2 through ma4 are the MA process coefficients. The model's formula is X_t = -1.159X_{t-1} - 0.999X_{t-2} + 0.091Z_{t-1} - 0.096Z_{t-2} -0.889Z_{t-3} + 0.159Z_{t-4} + Z_t, when $Z_t \sim WN$(0, 15.05).

We can observe in the output above that the observed values are c(24, 17, 22, 11, 6, 10), while the predicted values are c(11.3961213087569, 13.8379080019396, 14.4822062618425, 12.3391305198433, 14.1787351483795, 14.1885076822548). Moreover, the Mean Absolute Error (MAE) computed as the mean of the absolute values of the differences between predicted and observed values is 6.165. The Mean Absolute Percentage Error (MAPE) computed as the mean of the absolute values of the differences scaled by the observed responses is 0.492, and the Precision Measure (PM) computed as the ratio between MSPE and the sum of square differences between the response and the mean of the response is 1.217. The more the predictions are closer to the observed values, the closer the PM is to 0.

2a. ARIMA Fitting

For both time series, use the iterative model to fit an ARIMA(p,d,q) model with max order = 3, max differencing = 2. Evaluate the models for ACF and PACF plots as well as relevant tests.

```
data1 = read.csv('USD to EU.csv',header = TRUE)
EU.date = as.Date(as.character(data1$DATE), format="%Y-%m-%d")
data_EU = data1
data_EU$DATE = EU.date
head(data_EU)
```

```
##         DATE  DEXUSEU
## 1 2014-01-01 1.376275
## 2 2014-01-08 1.362320
## 3 2014-01-15 1.363760
## 4 2014-01-22 1.356500
## 5 2014-01-29 1.367280
## 6 2014-02-05 1.351960
```

```
data1 = data1[,2]
EU = ts(data1,start=c(2014),freq=52)

data2 = read.csv('USD to GBP.csv',header = TRUE)
GBP.date = as.Date(as.character(data2$DATE), format="%Y-%m-%d")
data_GBP = data2
data_GBP$DATE = GBP.date
head(data_GBP)
```

```
##         DATE DEXUSUK
## 1 2014-01-01 1.65030
## 2 2014-01-08 1.64318
## 3 2014-01-15 1.64266
## 4 2014-01-22 1.64510
## 5 2014-01-29 1.65666
## 6 2014-02-05 1.63704
```

```
data2 = data2[,2]
GBP = ts(data2,start=c(2014),freq=52)
```

USD to EU

```
model_EU = function(ts_data, max_p, max_d, max_q){
  orders = data.frame()
  for (p in 0:max_p){
    for (d in 0:max_d){
      for (q in 0:max_q) {
        possibleError <- tryCatch({
          mod = arima(ts_data, order=c(p,d,q), method="ML")
          current.aic = AIC(mod)
          orders<-rbind(orders,c(p,d,q,current.aic))
        },
          error=function(e) e
        )
        if(inherits(possibleError, "error")) next

      }
    }
  }
  names(orders) <- c("p","d","q","AIC")
  return(orders[order(orders$AIC),])
}

max_p = 3
max_q = 3
max_d = 2
ts_data = EU

# Observe first best models based on smallest AICs
orders = model_EU(ts_data, max_p, max_d, max_q)
```

```
## Warning in log(s2): NaNs produced

## Warning in log(s2): NaNs produced
```

```
## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1
```

```
## Warning in log(s2): NaNs produced
```

```
## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1
```

```
head(orders)
```

```
##    p d q       AIC
## 29 2 1 2 -1798.008
## 42 3 1 3 -1796.761
## 30 2 1 3 -1796.100
## 41 3 1 2 -1796.099
## 6  0 1 1 -1788.708
## 16 1 1 0 -1788.229
```

```
# Optimal order
opt_p = orders[1,1] # 2
opt_d = orders[1,2] # 1
opt_q = orders[1,3] # 2

# Fit the optimal model
opt_EU = arima(EU,order=c(opt_p,opt_d,opt_q), method="ML")
```

```
## Warning in log(s2): NaNs produced
```

```
## Warning in log(s2): NaNs produced
```

```
print(paste("Oprtimal p order is: ", opt_p))
```

```
## [1] "Oprtimal p order is:  2"
```

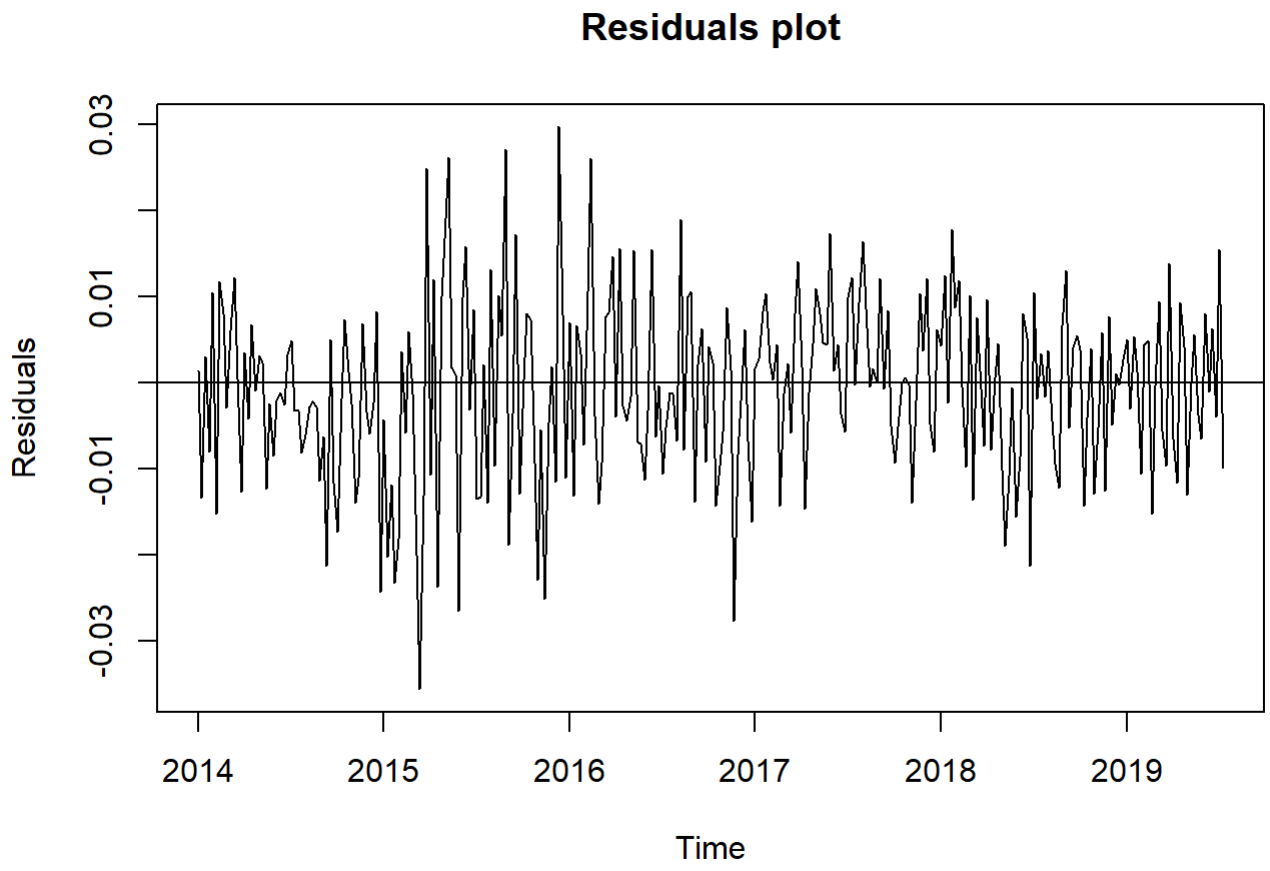
```
print(paste("Oprtimal d order is: ", opt_d))
```

```
## [1] "Oprtimal d order is:  1"
```

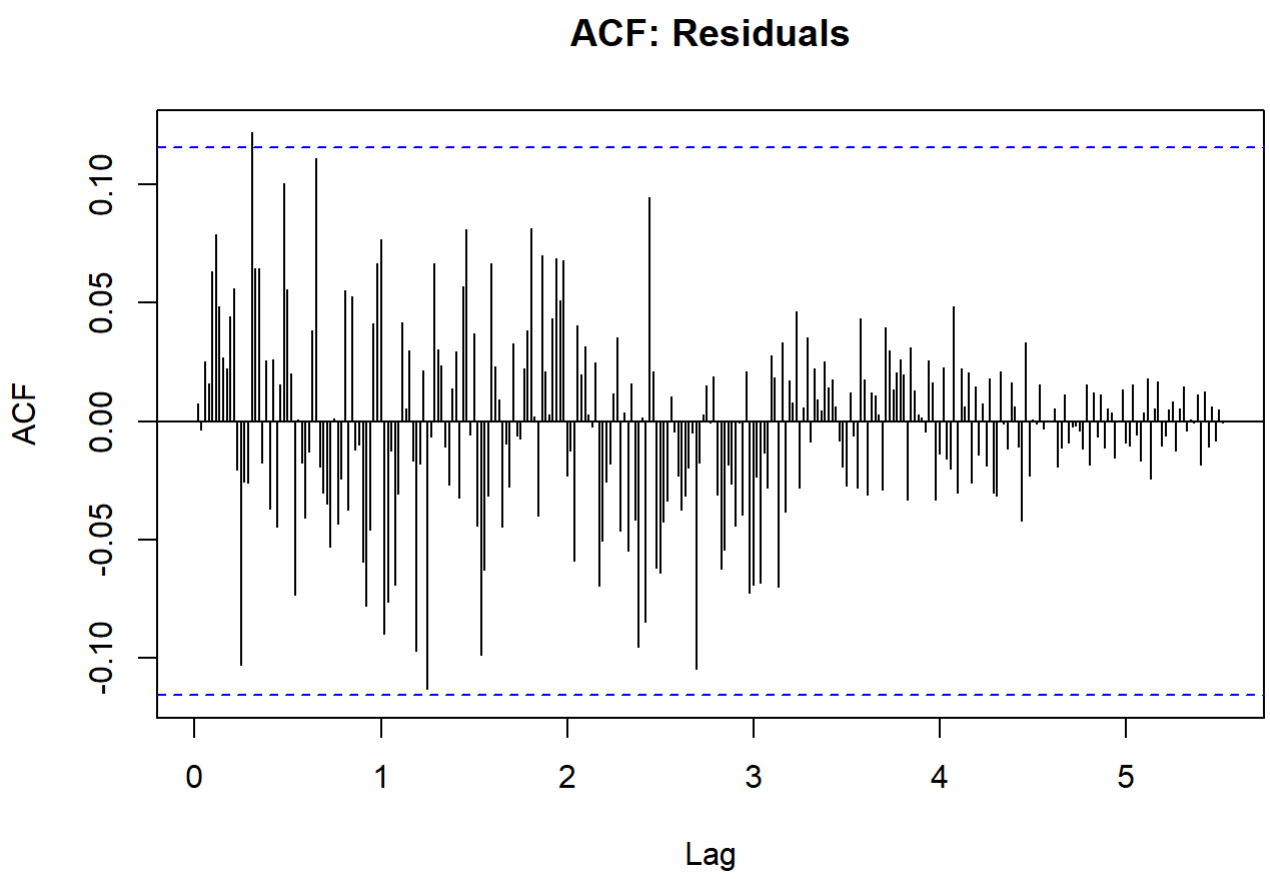
```
print(paste("Oprtimal q order is: ", opt_q))
```

```
## [1] "Oprtimal q order is:  2"
```

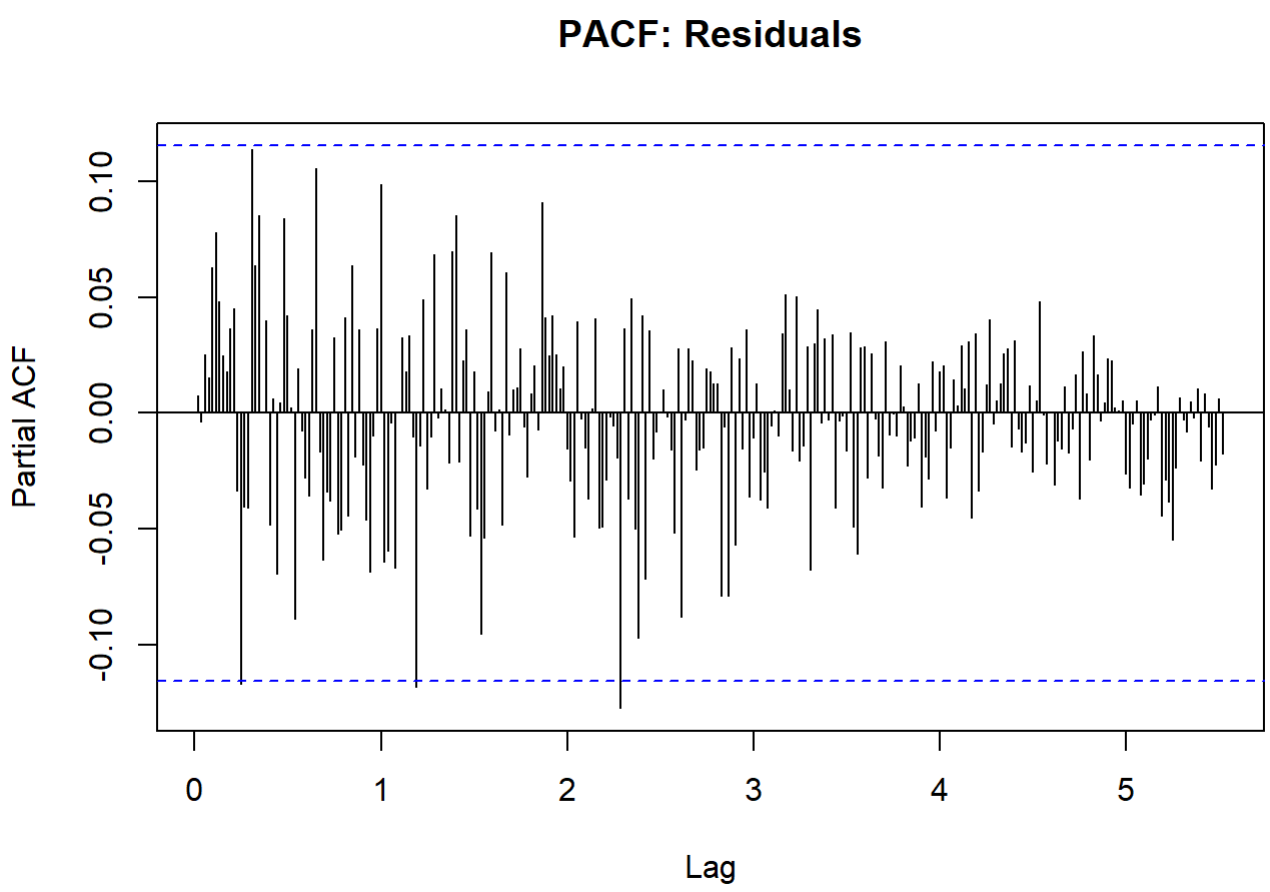
```
##############################################################################################
# Plot residuals of optimal model
plot(resid(opt_EU), ylab="Residuals", main="Residuals plot")
abline(h=0)
```



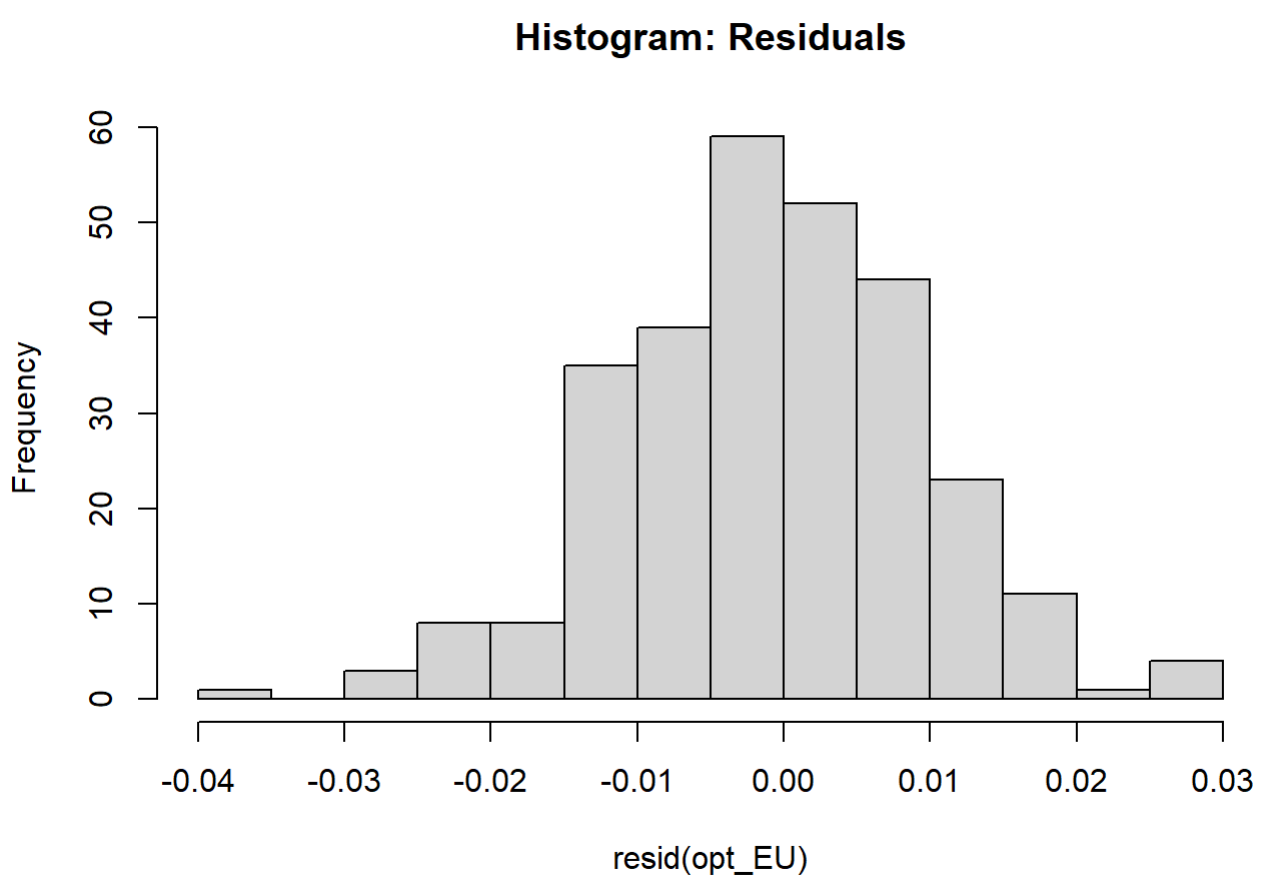

```
acf(resid(opt_EU), lag.max=52*40, main="ACF: Residuals")
```



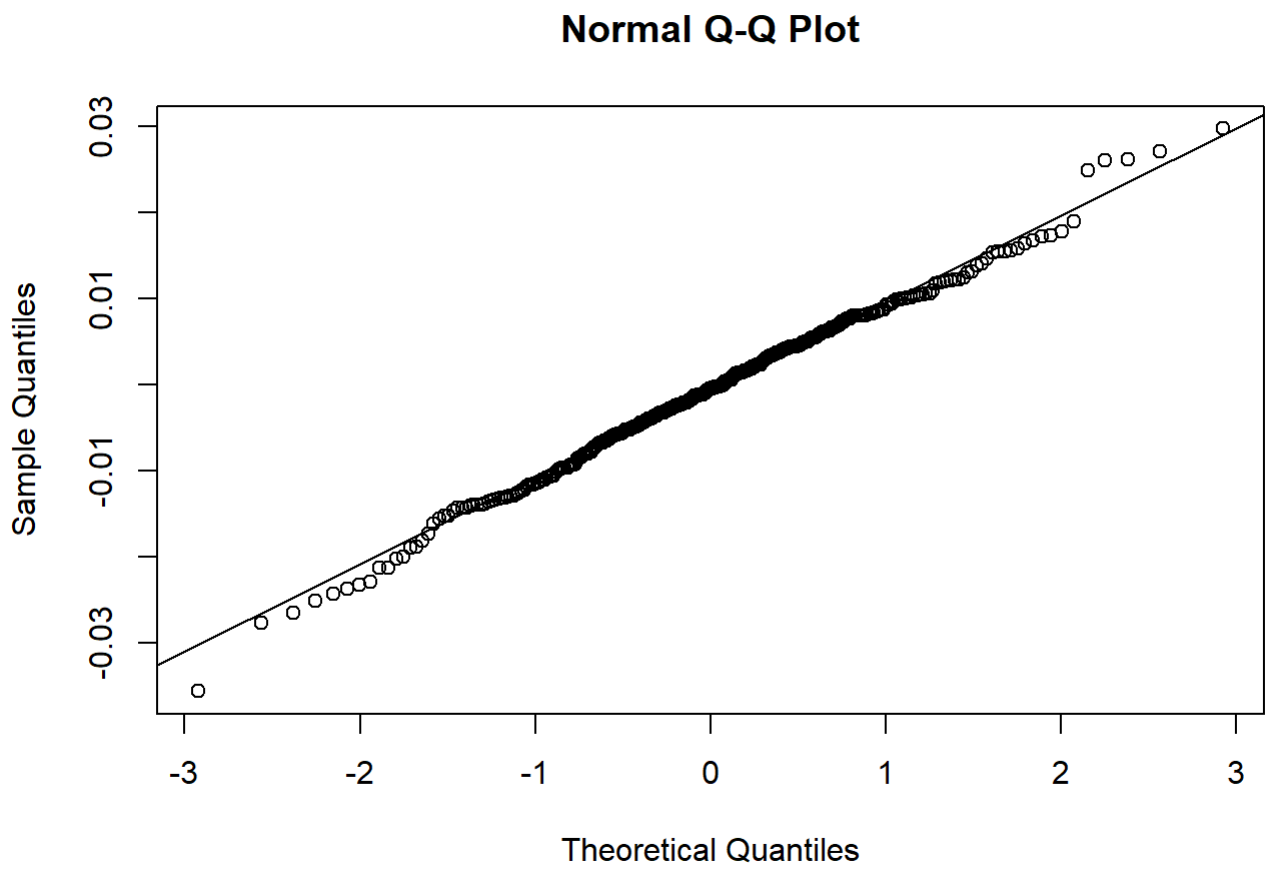

```
pacf(resid(opt_EU), lag.max=52*40, main="PACF: Residuals")
```




```
hist(resid(opt_EU), main="Histogram: Residuals")
```




```
qqnorm(resid(opt_EU))
qqline(resid(opt_EU))
```




```
# Test for uncorrelated residuals
#Null hypothesis is that the residuals are Uncorrelated. Hence want to see a larg p-value
# lag = p + q + 1; fitdf = p + q (disregard the d order on both lag and fitdf)
Box.test(resid(opt_EU), lag = (opt_p+opt_q+1), type = "Box-Pierce", fitdf = (opt_p+opt_q))
```

```
##
##  Box-Pierce test
##
## data:  resid(opt_EU)
## X-squared = 1.4178, df = 1, p-value = 0.2338
```

Answer

USD to EU

We used the iterative model to fit an ARIMA(p,d,q) model and determined based on the lowest AIC value of -1798.008 that the optimal model is ARIMA(2,1,2). Observe in the residuals plot a constant mean. Regarding the constant variance assumption, it seems like the variance is wider during the year of 2015, but for the most part we can say we that the constant variance also holds. Per the ACF/ PACF plots, the residuals process is white noise as all the values are within the confidence bands and the Q-Q and histogram plots indicates normality. Also, based on the Box-Pierce test's high p-value of 0.2338, we conclude that the residuals are uncorrelated.

USD to GBP

```
model_GBP = function(ts_data, max_p, max_d, max_q){
  orders = data.frame()
  for (p in 0:max_p){
    for (d in 0:max_d){
      for (q in 0:max_q) {
        possibleError <- tryCatch({
          mod = arima(ts_data, order=c(p,d,q), method="ML")
          current.aic = AIC(mod)
          orders<-rbind(orders,c(p,d,q,current.aic))
        },
          error=function(e) e
        )
        if(inherits(possibleError, "error")) next

      }
    }
  }
  names(orders) <- c("p","d","q","AIC")
  return(orders[order(orders$AIC),])
}

max_p = 3
max_q = 3
max_d = 2
ts_data = GBP

# Observe first best models based on smallest AICs
orders = model_GBP(ts_data, max_p, max_d, max_q)
```

```
## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1

## Warning in stats::arima(x = x, order = order, seasonal = seasonal, xreg =
## xreg, : possible convergence problem: optim gave code = 1
```

```
head(orders,10)
```

```
##    p d q       AIC
## 29 2 1 0 -1632.058
## 6  0 1 1 -1631.801
## 7  0 1 2 -1631.018
## 27 2 0 2 -1630.633
## 18 1 1 1 -1630.534
## 8  0 1 3 -1630.533
## 20 1 1 3 -1630.338
## 40 3 1 0 -1630.148
## 30 2 1 1 -1630.106
## 31 2 1 2 -1629.963
```

```
# Selecting second lowest AIC as optimal order since the order is more simple and AIC within sig
nificance threshold of 2
opt_p = orders[2,1] # 0
opt_d = orders[2,2] # 1
opt_q = orders[2,3] # 1

# Fit the optimal model
opt_GBP = arima(GBP,order=c(opt_p,opt_d,opt_q), method="ML")

print(paste("Oprtimal p order is: ", opt_p))
```

```
## [1] "Oprtimal p order is:  0"
```

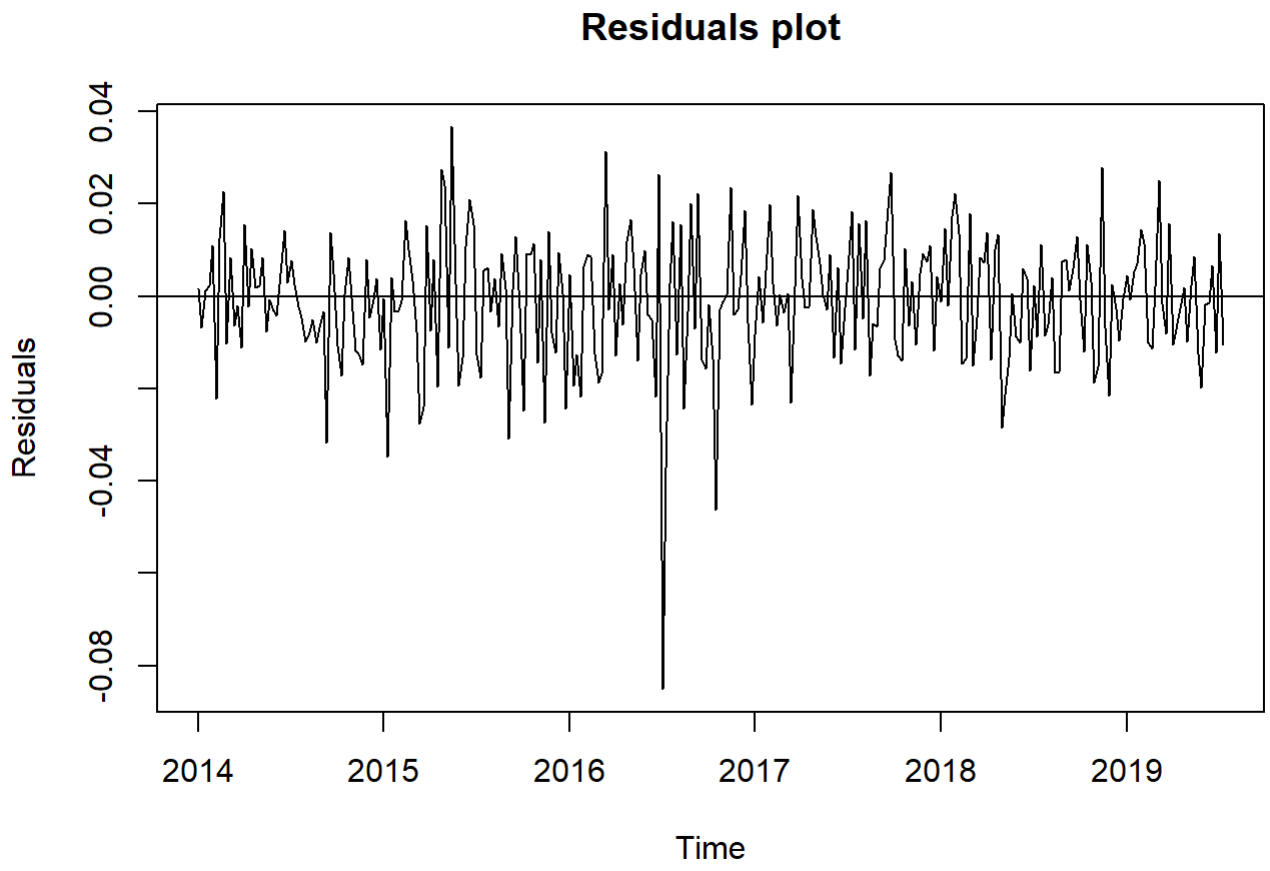
```
print(paste("Oprtimal d order is: ", opt_d))
```

```
## [1] "Oprtimal d order is:  1"
```

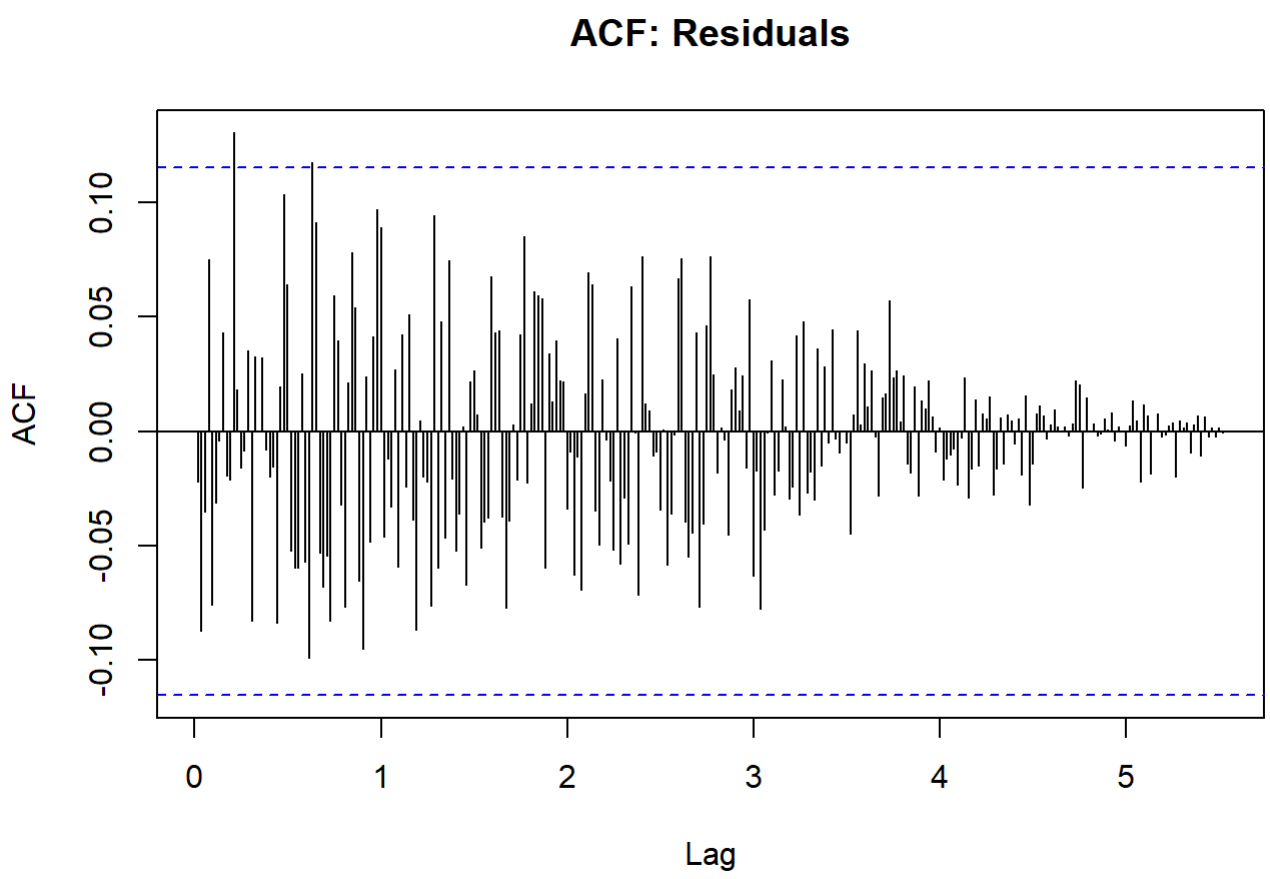
```
print(paste("Oprtimal q order is: ", opt_q))
```

```
## [1] "Oprtimal q order is:  1"
```

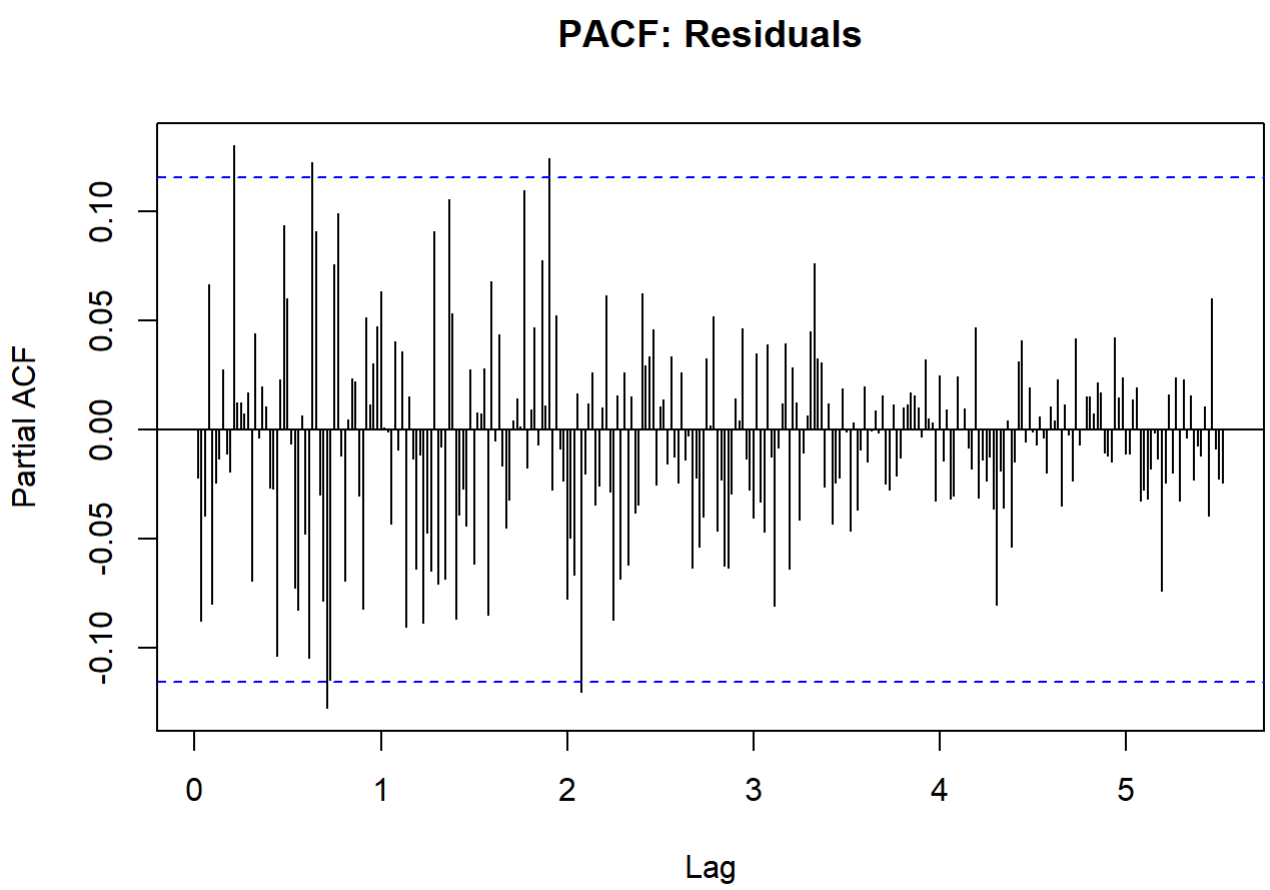
```
##############################################################################################
# Plot residuals of optimal model
plot(resid(opt_GBP), ylab="Residuals", main="Residuals plot")
abline(h=0)
```



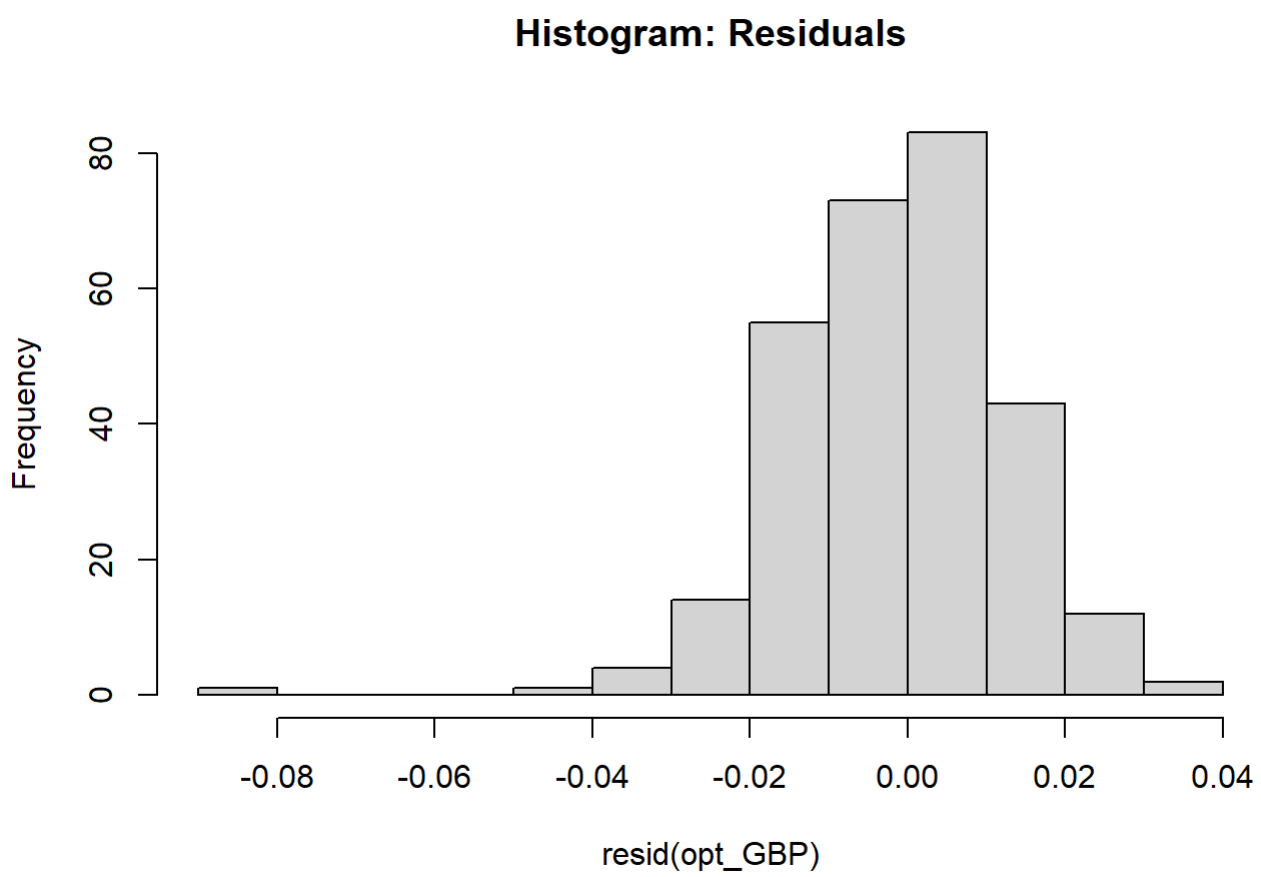

```
acf(resid(opt_GBP), lag.max=52*40, main="ACF: Residuals")
```



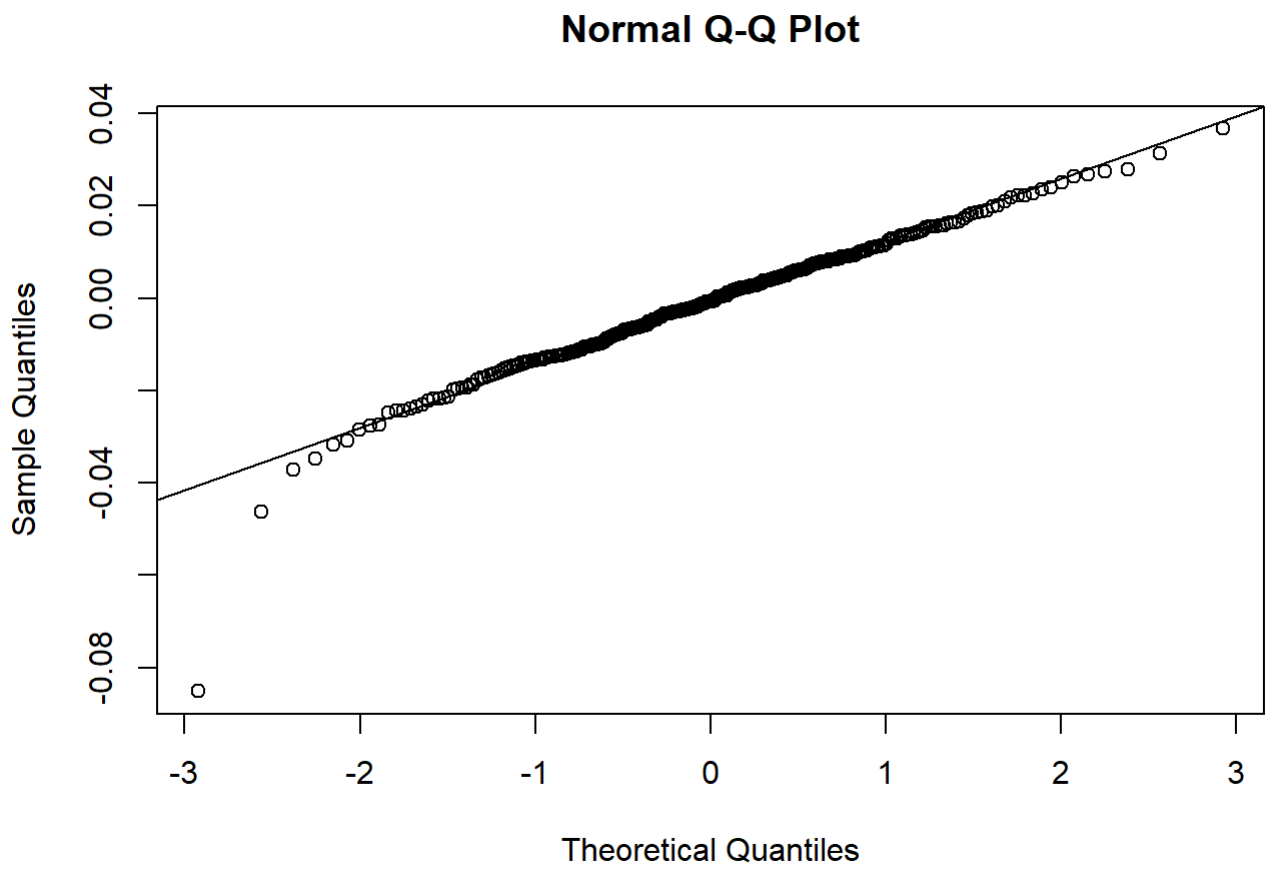

```
pacf(resid(opt_GBP), lag.max=52*40, main="PACF: Residuals")
```




```
hist(resid(opt_GBP), main="Histogram: Residuals")
```




```
qqnorm(resid(opt_GBP))
qqline(resid(opt_GBP))
```




```
# Test for uncorrelated residuals
#Null hypothesis is that the residuals are Uncorrelated. Hence want to see a larg p-value
# lag = p + q + 1; fitdf = p + q (disregard the d order on both lag and fitdf)
Box.test(resid(opt_GBP), lag = (opt_p+opt_q+1), type = "Box-Pierce", fitdf = (opt_p+opt_q))
```

```
##
##  Box-Pierce test
##
## data:  resid(opt_GBP)
## X-squared = 2.3393, df = 1, p-value = 0.1261
```

Answer

USD to GBP

Similarly to the EU model, we used the iterative model to fit an ARIMA(p,d,q) model. ***While the smallest AIC is -1632.058 with orders (2,1,0), we selected the orders (0,1,1) because it is a simpler model and within the AIC significance threshold of 2 (AIC -1631.801) as determined in question 1.*** Thus, an ARIMA(0,1,1) was fitted. Observe in the residuals plot a constant mean. Other than the one downward spike in the year of 2016, the variance is also constant. Per the ACF/ PACF plots the residuals process is white noise as all the values are within the confidence bands and the Q-Q and histogram plots indicates normality. Also, based on the Box-Pierce test's high p-value of 0.1261, we conclude that the residuals are uncorrelated.

2b. Forecasting

Show coefficients for both models and compare significance of coefficients. Next, for each series keep the last 12 data points for testing. Generate forecasts of those 12 weeks and compare the predicted values to the actual ones. Include 95% confidence interval for the forecasts and provide plots. Calculate Mean Absolute Percentage Error (MAPE) and Precision Measure (PM) for each; compare the models.

USD to EU model's coefficients

```
opt_EU
```

```
##
## Call:
## arima(x = EU, order = c(opt_p, opt_d, opt_q), method = "ML")
##
## Coefficients:
##           ar1      ar2     ma1     ma2
##       -0.4504  -0.8491  0.5958  0.9286
## s.e.   0.0798   0.0582  0.0614  0.0573
##
## sigma^2 estimated as 0.0001072:  log likelihood = 904,  aic = -1800.01
```

```
coeftest(opt_EU)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
## ar1 -0.450384   0.079828  -5.6419 1.681e-08 ***
## ar2 -0.849137   0.058169 -14.5979 < 2.2e-16 ***
## ma1  0.595798   0.061439   9.6973 < 2.2e-16 ***
## ma2  0.928562   0.057338  16.1945 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

USD to GBP model's coefficients

```
opt_GBP
```

```
##
## Call:
## arima(x = GBP, order = c(opt_p, opt_d, opt_q), method = "ML")
##
## Coefficients:
##          ma1
##       0.2400
## s.e.  0.0602
##
## sigma^2 estimated as 0.0001959:  log likelihood = 817.9,  aic = -1633.8
```

```
coeftest(opt_GBP)
```

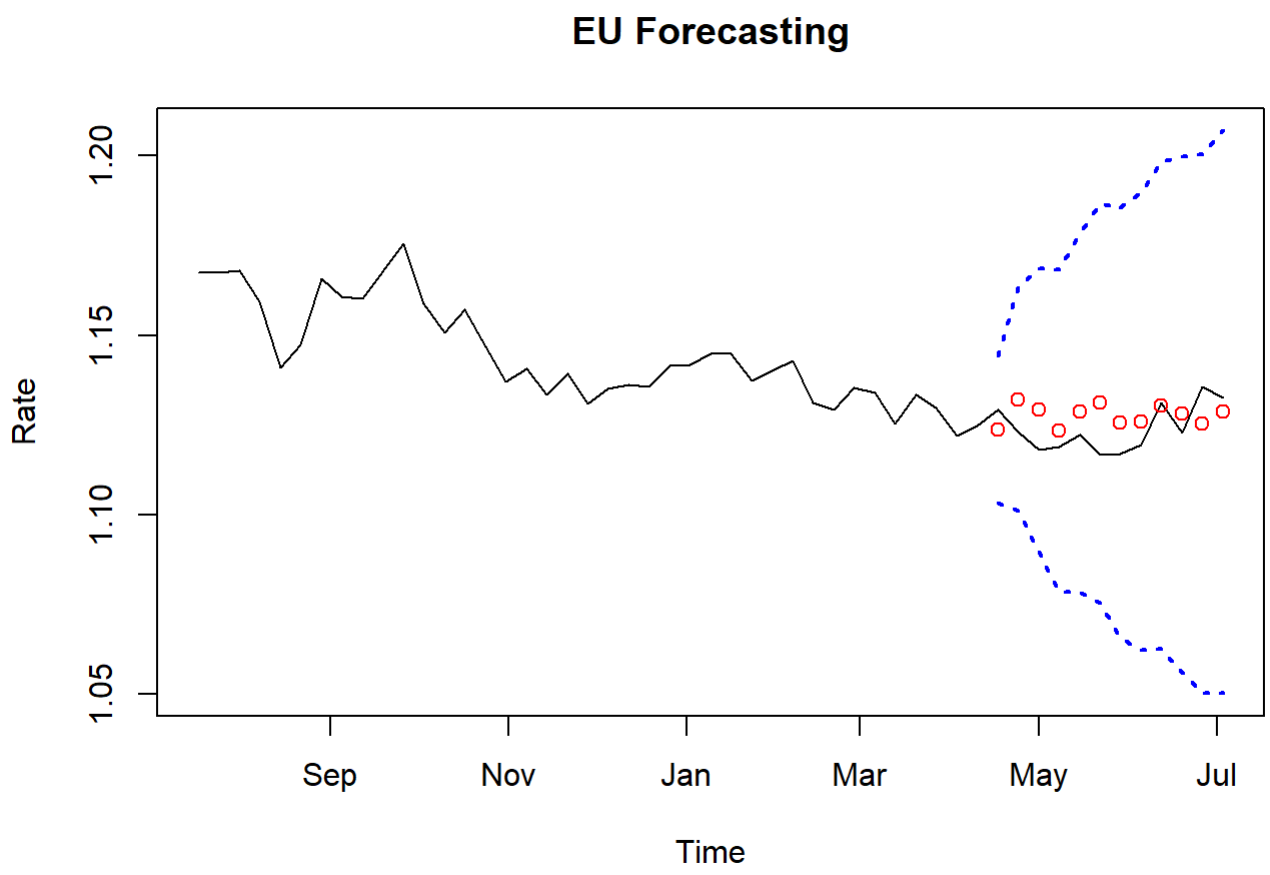
```
##
## z test of coefficients:
##
##     Estimate Std. Error z value  Pr(>|z|)
## ma1 0.240041   0.060222  3.9859 6.722e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Answer

Observe above the EU ARIMA(2,1,2) model's coefficients are ar1 = -0.45; ar2 = -0.849; ma1 = 0.595; and ma2 = 0.928. the GBP ARIMA(0,1,1) model's coefficient is ma1 = 0.24. All the coefficients are significant.

Predict USD to EU

```
n = length(data1)
nfit = n-12

pred_model_EU = arima(EU[1:nfit], order = c(2,1,2),method = "ML")
pred = predict(pred_model_EU,n.ahead=12)
ubound = pred$pred+1.96*pred$se
lbound = pred$pred-1.96*pred$se
ymin = min(lbound)
ymax = max(ubound)

plot(data_EU$DATE[(n-50):n], EU[(n-50):n], type="l", ylim=c(ymin,ymax), xlab="Time", ylab="Rate"
, main="EU Forecasting")
points(data_EU$DATE[(nfit+1):n], pred$pred,col="red")
lines(data_EU$DATE[(nfit+1):n], ubound,lty=3,lwd= 2, col="blue")
lines(data_EU$DATE[(nfit+1):n], lbound,lty=3,lwd= 2, col="blue")
```




```
obs_EU = EU[(nfit+1):n]
pred_EU = pred$pred

print(paste("The observed values are: ", data.frame(obs_EU)))
```

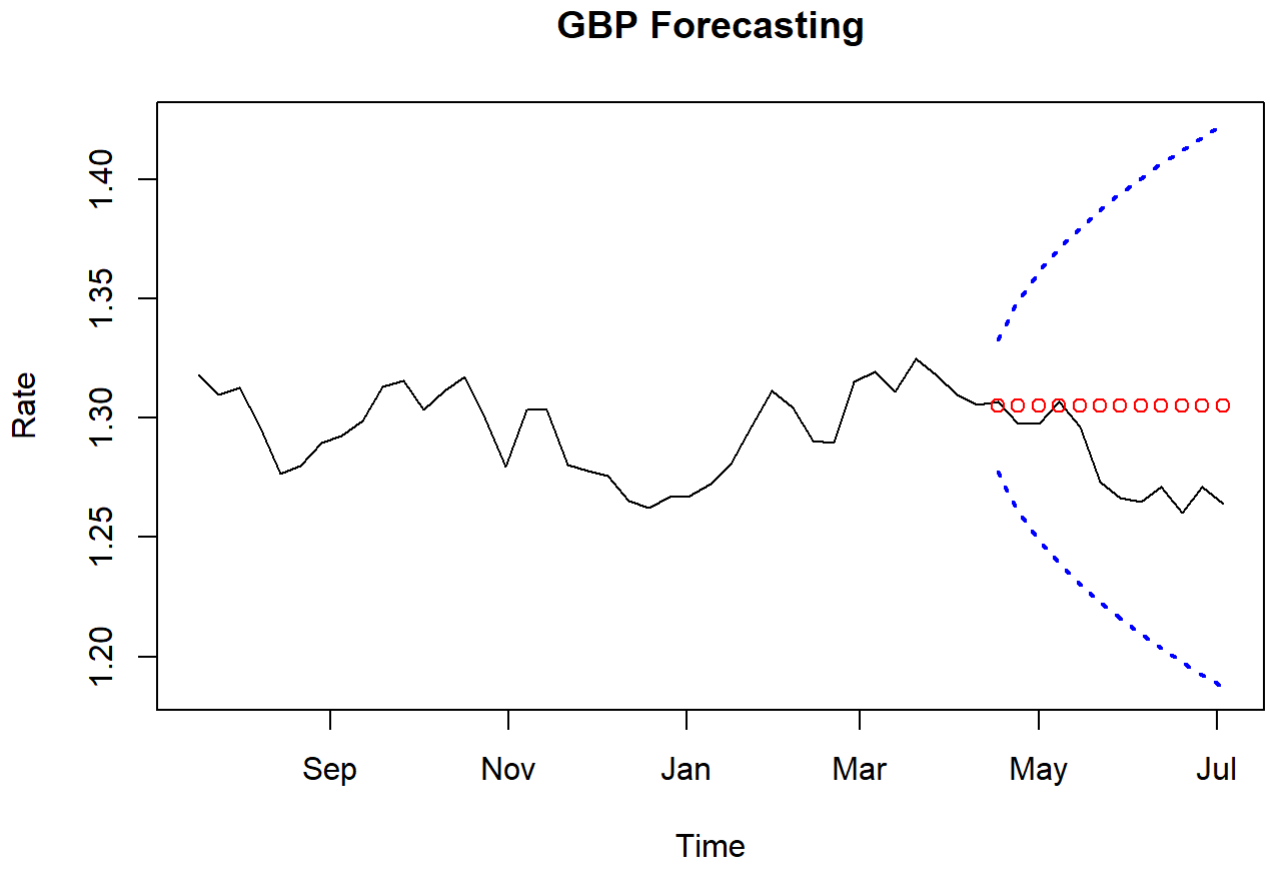
```
## [1] "The observed values are:  c(1.1292, 1.123, 1.11814, 1.11902, 1.12232, 1.11656, 1.11705,
1.11946, 1.1311, 1.12274, 1.13562, 1.13258)"
```

```
print(paste("The predicted values are: ", data.frame(pred_EU)))
```

```
## [1] "The predicted values are:  c(1.12379896911638, 1.13210330002379, 1.12914589234691, 1.123
43968811396, 1.12855756515754, 1.13106983720381, 1.12557424256461, 1.12594832354548, 1.130447223
35504, 1.12807557448755, 1.12533548549678, 1.12860157836568)"
```

Predict USD to EU

```
n = length(data2)
nfit = n-12

pred_model_GBP = arima(GBP[1:nfit], order = c(0,1,1),method = "ML")
pred = predict(pred_model_GBP,n.ahead=12)
ubound = pred$pred+1.96*pred$se
lbound = pred$pred-1.96*pred$se
ymin = min(lbound)
ymax = max(ubound)

plot(data_GBP$DATE[(n-50):n], GBP[(n-50):n], type="l", ylim=c(ymin,ymax), xlab="Time", ylab="Rat
e", main="GBP Forecasting")
points(data_GBP$DATE[(nfit+1):n], pred$pred,col="red")
lines(data_GBP$DATE[(nfit+1):n], ubound,lty=3,lwd= 2, col="blue")
lines(data_GBP$DATE[(nfit+1):n], lbound,lty=3,lwd= 2, col="blue")
```




```
obs_GBP = GBP[(nfit+1):n]
pred_GBP = pred$pred

print(paste("The observed values are: ", data.frame(obs_GBP)))
```

```
## [1] "The observed values are:  c(1.30688, 1.29758, 1.29752, 1.3066, 1.29618, 1.2733, 1.26662
5, 1.26482, 1.27108, 1.26038, 1.27098, 1.26378)"
```

```
print(paste("The predicted values are: ", data.frame(pred_GBP)))
```

```
## [1] "The predicted values are:  c(1.30492178478156, 1.30492178478156, 1.30492178478156, 1.304
92178478156, 1.30492178478156, 1.30492178478156, 1.30492178478156, 1.30492178478156, 1.304921784
78156, 1.30492178478156, 1.30492178478156, 1.30492178478156)"
```

EU Model Accuracy Measures

```
## Compute Accuracy Measures

### Mean Absolute Percentage Error (MAPE)
MAPE = mean(abs(pred_EU-obs_EU)/obs_EU)

### Precision Measure (PM)
PM = sum((pred_EU-obs_EU)^2)/sum((obs_EU-mean(obs_EU))^2)

print(paste("Mean Absolute Percentage Error (MAPE) is: ", MAPE))
```

```
## [1] "Mean Absolute Percentage Error (MAPE) is:  0.00638001466158603"
```

```
print(paste("Precision Measure (PM) is: ", PM))
```

```
## [1] "Precision Measure (PM) is:  1.6163090715705"
```

GBP Model Accuracy Measures

```
## Compute Accuracy Measures

### Mean Absolute Percentage Error (MAPE)
MAPE = mean(abs(pred_GBP-obs_GBP)/obs_GBP)

### Precision Measure (PM)
PM = sum((pred_GBP-obs_GBP)^2)/sum((obs_GBP-mean(obs_GBP))^2)

print(paste("Mean Absolute Percentage Error (MAPE) is: ", MAPE))
```

```
## [1] "Mean Absolute Percentage Error (MAPE) is:  0.0190740986506718"
```

```
print(paste("Precision Measure (PM) is: ", PM))
```

```
## [1] "Precision Measure (PM) is:  2.88425178605797"
```

Answer

Observe above the forecast plots of both EU and GBP forecasting. Based on the MAPE and PM, the EU ARIMA model predicts better than the GBP model because both measures are smaller for the EU than the GBP model. We can also see that the EU model predicts better by looking at the prediction plots. While the EU model captures the movement in rate to some extent, the GBP prediction is basically flat for all 12 prediction periods.

Question 3: Reflection on ARIMA (5 Pts)

Considering your understanding of the model as well as what you experiences completing the above questions, how would you personally regard the effectiveness of ARIMA modeling? Where would it be appropriate to use for forecasting and where would you recommend against? What are some specific points of caution one would need to consider when considering using it?

Answer

From my understanding of the model, answering the questions above and my personal experience, I do not think the ARIMA modeling is suitable for forecasting financial instruments such as publicly traded stocks or for predicting future exchange rates. If we know that yesterday's price of a stock was 75 dollars for example, I think anyone could fairly assume that tomorrow's price will be between plus/ minus 5 dollars, and can be 95% confident that it will be between plus/ minus 15 dollars. We do not need complicated models for that. Actually, thinking that a stock price (or a company's revenue) can be predicted solely based on past prices (or past revenue) is ridiculous. By doing so, we exclude the real factors that affect financial instruments' prices such as market and economic consideration, interest rate, industry factors, political factors, internal factors such as mergers and acquisitions, etc. One could still argue that all those factors are "built-in" or embedded in the past prices, hence we do account for those factors. But such factors, especially within the financial space, must be directly built into a prediction model. Quant Hedge Funds would lose their entire investment if they were to apply ARIMA as their investing strategy.

In my opinion, the ARIMA model is more suitable for a "sterile" environment in which the process is not effected by so many external factors. One example for such process in which I believe ARIMA would be a good modeling approach is in manufacturing where we can control the input and we would like to forecast output or a machine decay based on its past quantity produced. When considering using ARIMA, one should pay attention to whether the process is stationary or not and what differencing lag (or other methods such as log transforming the data) to use in order to achieve stationarity as well as the option to use Seasonal ARIMA in case of "seasoned-process." When using ARIMA prediction we must remember that the model is not so good for predicting long-run observations.