ISYE 6402 Homework 4

Background

For this data analysis, you will again analyze the currency exchange data but to a greater extent and including two different currencies for comparison. File DailyCurrencyData.csv contains the daily exchange rate of EUR/USD and GBP/USD from January 1999 through December 31st 2020. File MonthlyCurrencyData.csv contains the monthly exchange rate of EUR/USD and GBP/USD for the same time period. Similarly to homework 2, we will aggregate the daily data into weekly data. We will compare our analysis using ARMA modeling on both weekly and monthly data for the two currencies.

```
library(zoo)
library(lubridate)
library(mgcv)
library(TSA)
library(xts)
library(astsa)
library(tseries)
```

Instructions on reading the data

To read the data in R, save the file in your working directory (make sure you have changed the directory if different from the R working directory) and read the data using the R function `read.csv()`

```
daily <- read.csv("~/Downloads/DailyCurrencyData-2.csv", head = TRUE)
monthly <- read.csv("~/Downloads/MonthlyCurrencyData-1.csv", head = TRUE)

daily$Date <- as.Date(daily$Date, "%m/%d/%Y")
monthly$Date <- as.Date(paste0(monthly$Date, "-01"), "%Y-%m-%d")
colnames(monthly) <- colnames(daily)
```

#Question 1. Weekly vs Monthly Exploratory Data Analysis (20 points)

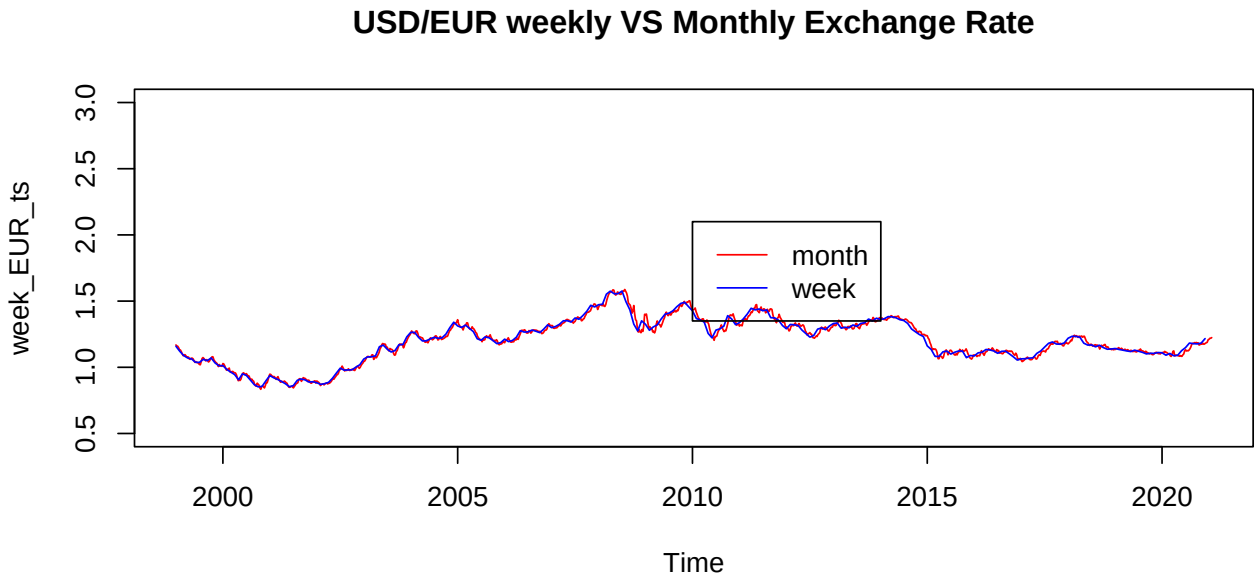
1a. Based on your intuition, when would you use weekly vs monthly time series data?

Response Based on common sense, the exchange rate does not change significantly on a daily or weekly basis, but does varies on a day-to-day basis with a narrow range; and is related to the previous closing rate. When I'm trying to project the exchange rate in a near future, I would use weekly time series data to capture to variety; when I'm trying to oversea a longer term or a trend of the exchange, I may use the monthly time series data to better capture the trend while saving time in data processing.

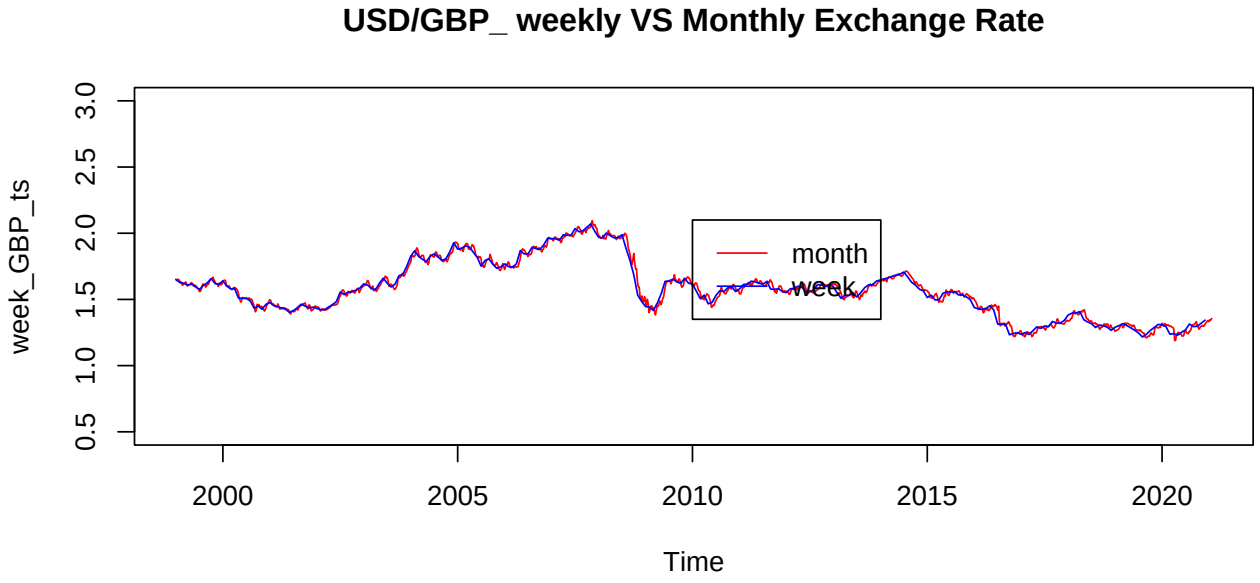
1b. Plot the time series plots for both currency exchange rates comparing weekly vs monthly data. How do the weekly and monthly time series data compare? How do the time series for the two currencies compare?

```
daily <- na.locf(daily)

weekly <- daily
weekly$Date <- floor_date(weekly$Date, "week")
weekly <- aggregate(weekly[, 2:3], by = list(weekly$Date), FUN = mean)
colnames(weekly)[1] <- "Date"
```

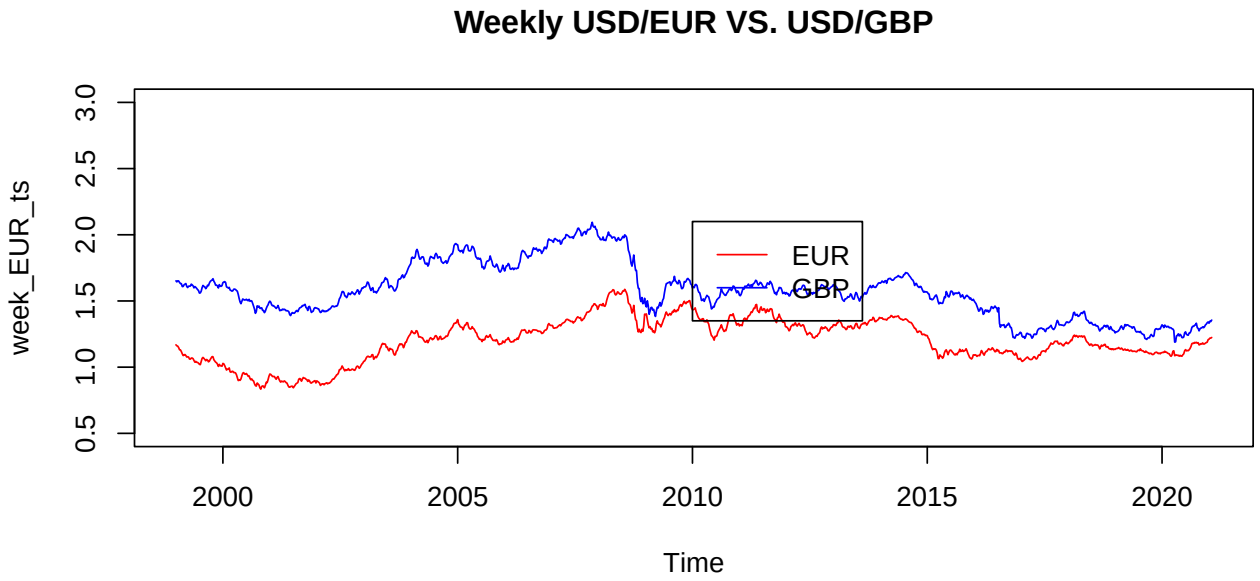
```
week_EUR_ts = ts(weekly$EUR,start=1999,freq=52)
month_EUR_ts = ts(monthly$EUR,start=1999,freq=12)

week_GBP_ts = ts(weekly$GBP,start=1999,freq=52)
month_GBP_ts = ts(monthly$GBP,start=1999,freq=12)


#EUR weekly and monthly
ts.plot(week_EUR_ts,main='USD/EUR weekly VS Monthly Exchange Rate',col='red',ylim=c(0.5,3))
lines(month_EUR_ts,col='blue')
legend(x=2010,y=2.1,legend=c("month","week"),lty=1, col=c("red","blue"))
```



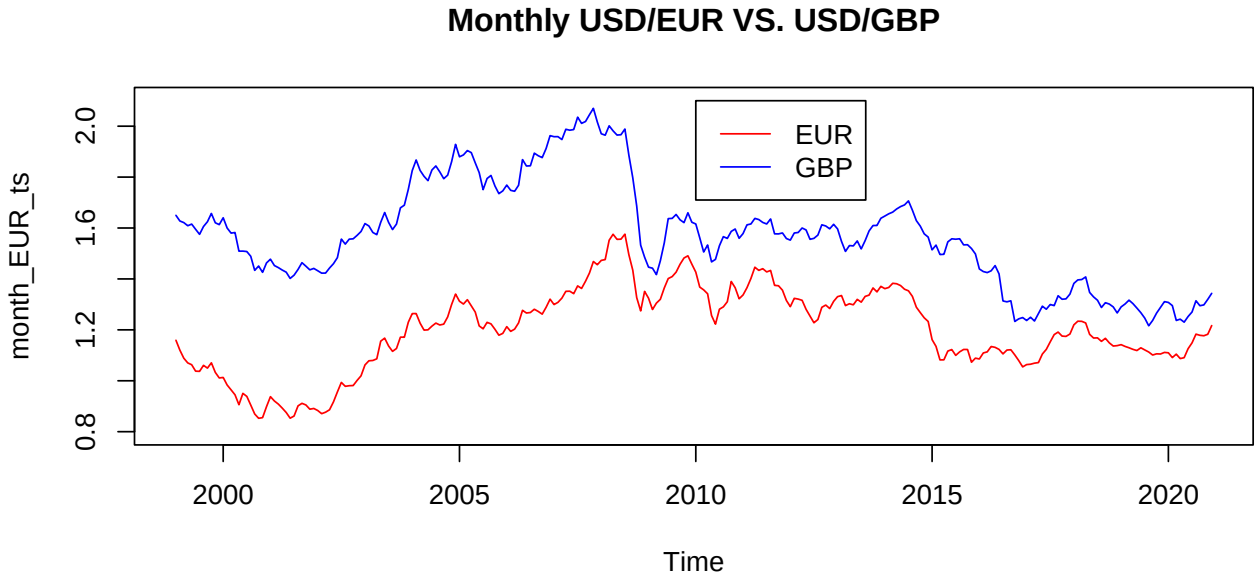

```
#GBP weekly and monthly
ts.plot(week_GBP_ts,main='USD/GBP_ weekly VS Monthly Exchange Rate',col='red',ylim=c(0.5,3))
lines(month_GBP_ts,col='blue')
legend(x=2010,y=2.1,legend=c("month","week"),lty=1, col=c("red","blue"))
```



```
#weekly of EUR and GBP
ts.plot(week_EUR_ts,main='Weekly USD/EUR VS. USD/GBP',col='red',ylim=c(0.5,3))
lines(week_GBP_ts,col='blue')
legend(x=2010,y=2.1,legend=c("EUR","GBP"),lty=1, col=c("red","blue"))
```

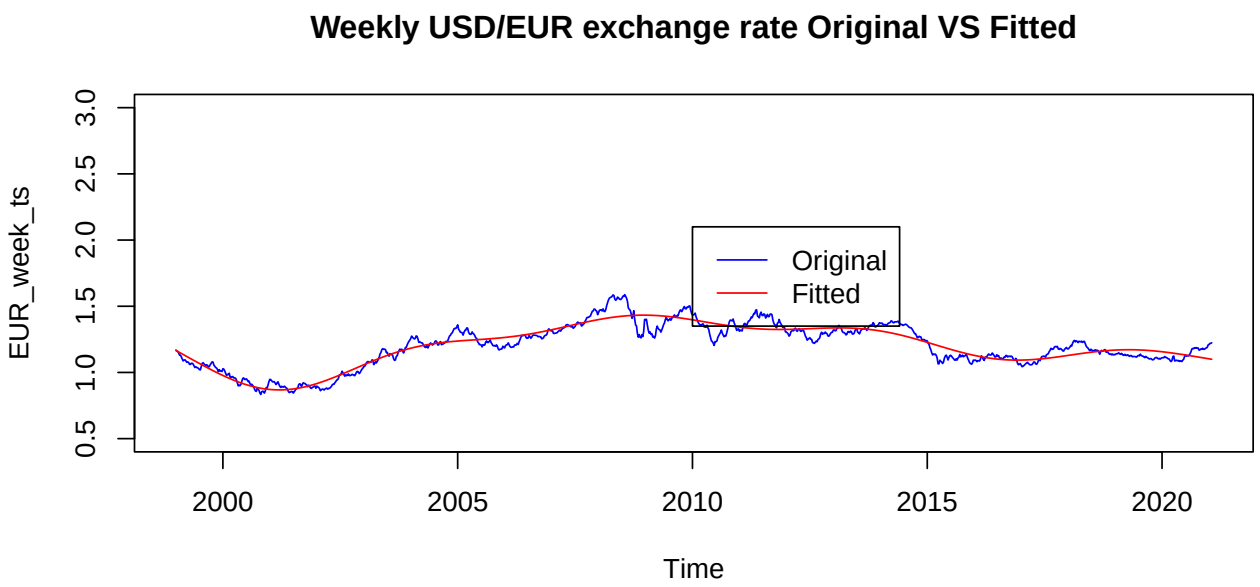



```
#monthly of EUR and GBP
ts.plot(month_EUR_ts,main='Monthly USD/EUR VS. USD/GBP',col='red',ylim=c(0.8,2.1))
lines(month_GBP_ts,col='blue')
legend(x=2010,y=2.1,legend=c("EUR","GBP"),lty=1, col=c("red","blue"))
```

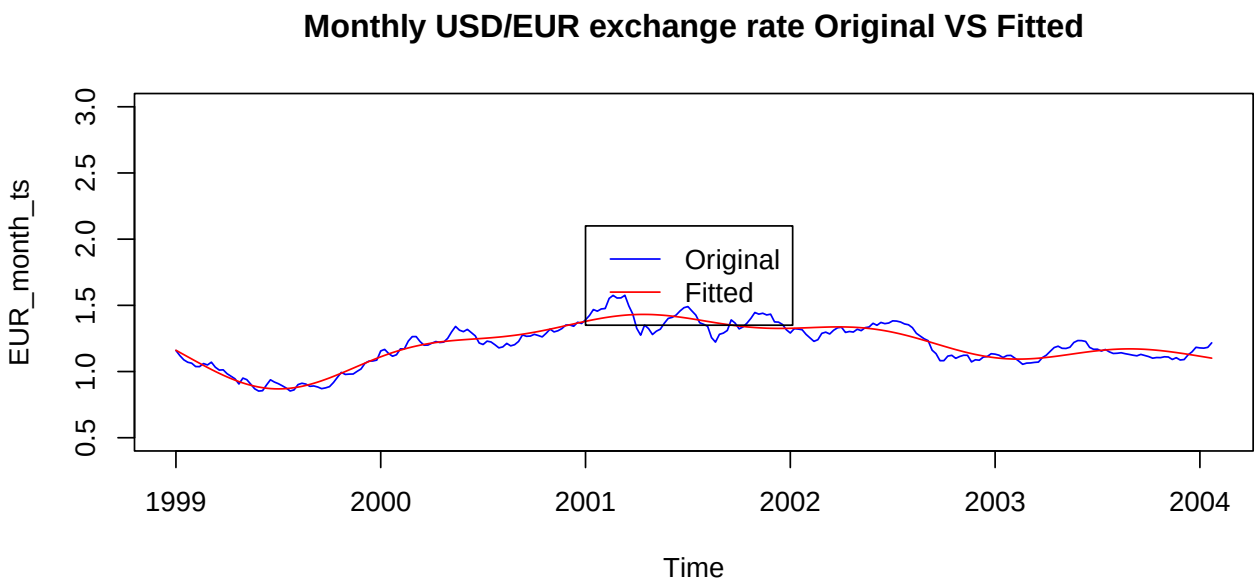


Response: Weekly vs Monthly Time Series data comparison Both weekly and monthly time series data capture similar variation of the time series data on EUR and GBP; yet the weekly data has fluactuate more while the monthly data tend to be more smoothly. The time series of two currencies shows similar non-constant variance patterns but there's no appear ant trend or seasonality. EUR overall has a higher exchange rate than GBP over years. **1c.** Fit a non-parametric trend using splines regression to both the weekly and monthly time series data for both currencies. Overlay the fitted trends for each of the currency separately. How do the trends compare when comparing those fitted using the weekly and monthly data? How do the trends for the two currencies compare?

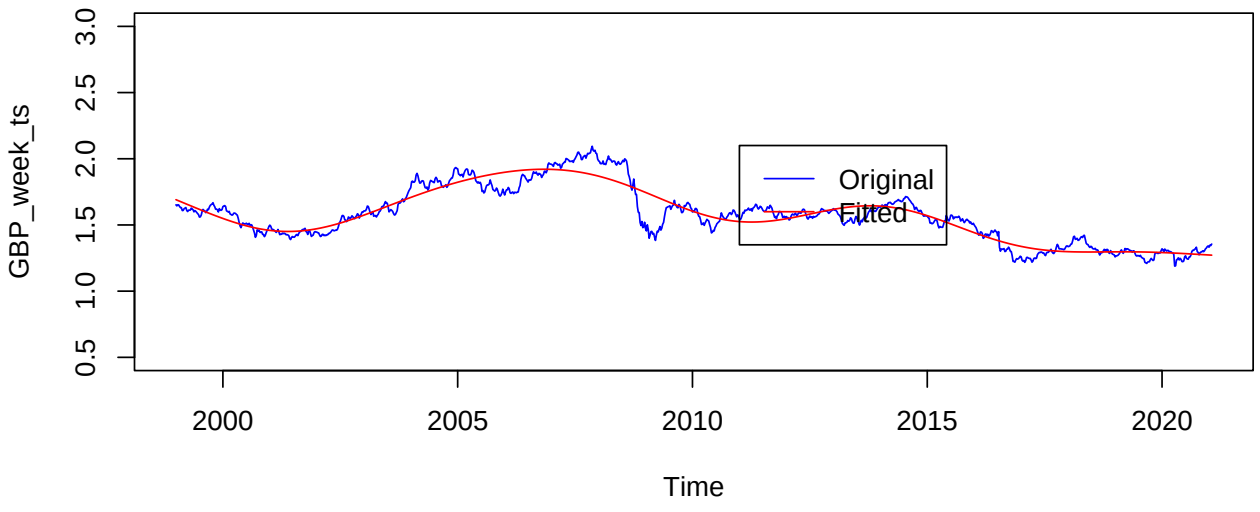
```r
#Spline regression EUR-WEEKLY
EUR_week_ts <- ts(weekly$EUR, start = 1999, freq = 52)
time.pts = c(1:length(EUR_week_ts))
time.pts = c(time.pts - min(time.pts))/max(time.pts)
gam.fit_week_EUR = gam(EUR_week_ts~s(time.pts))
gam.fit_EUR_weekly = ts(fitted(gam.fit_week_EUR),start=1999,frequency=52)
#Overlay the fitted EUR-WEEKLY and Original
ts.plot(EUR_week_ts,main='Weekly USD/EUR exchange rate Original VS Fitted',col='blue',ylim=c(0.5,3))
lines(gam.fit_EUR_weekly,col='red')
legend(x=2010,y=2.1,legend=c("Original","Fitted"),lty=1, col=c("blue","red"))
```




```r
#Spline regression EUR-MONTHLY
EUR_month_ts <- ts(monthly$EUR, start = 1999, freq = 52)
time.pts.me= c(1:length(EUR_month_ts))
time.pts.me= c(time.pts.me- min(time.pts.me))/max(time.pts.me)
gam.fit_month_EUR = gam(EUR_month_ts~s(time.pts.me))
gam.fit_EUR_monthly = ts(fitted(gam.fit_month_EUR),start=1999,frequency=52)
#Overlay the fitted EUR-MONTHLY and Original
ts.plot(EUR_month_ts,main='Monthly USD/EUR exchange rate Original VS Fitted',col='blue',ylim=c(0.5,3))
lines(gam.fit_EUR_monthly,col='red')
legend(x=2001,y=2.1,legend=c("Original","Fitted"),lty=1, col=c("blue","red"))
```

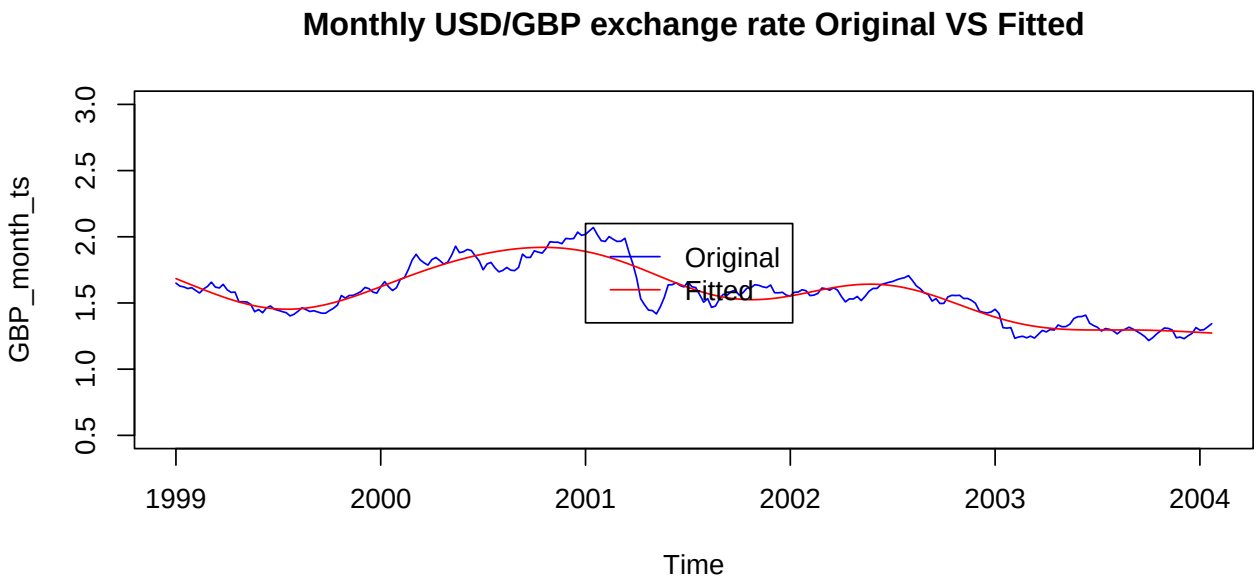



```
#Spline regression GBP-WEEKLY
GBP_week_ts <- ts(weekly$GBP, start = 1999, freq = 52)
time.pts.wg= c(1:length(GBP_week_ts))
time.pts.wg= c(time.pts.wg- min(time.pts.wg))/max(time.pts.wg)
gam.fit_week_GBP = gam(GBP_week_ts~s(time.pts.wg))
gam.fit_GBP_weekly = ts(fitted(gam.fit_week_GBP),start=1999,frequency=52)
#Overlay the fitted GBP-WEEKLY and Original
ts.plot(GBP_week_ts,main='Weekly USD/GBP exchange rate Original VS Fitted',col='blue',ylim=c(0.5,3))
lines(gam.fit_GBP_weekly,col='red')
legend(x=2011,y=2.1,legend=c("Original","Fitted"),lty=1, col=c("blue","red"))
```

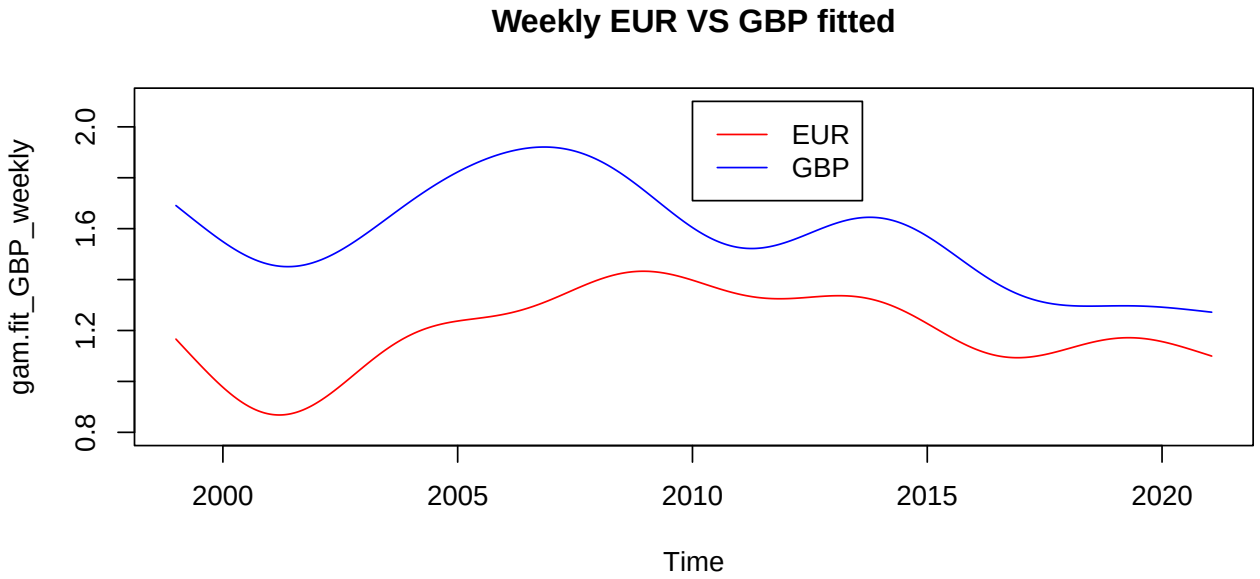



```
#Spline regression GBP-MONTHLY
GBP_month_ts <- ts(monthly$GBP, start = 1999, freq = 52)
time.pts.mg= c(1:length(GBP_month_ts))
time.pts.mg= c(time.pts.mg- min(time.pts.mg))/max(time.pts.mg)
gam.fit_month_GBP = gam(GBP_month_ts~s(time.pts.mg))
gam.fit_GBP_monthly = ts(fitted(gam.fit_month_GBP),start=1999,frequency=52)
```

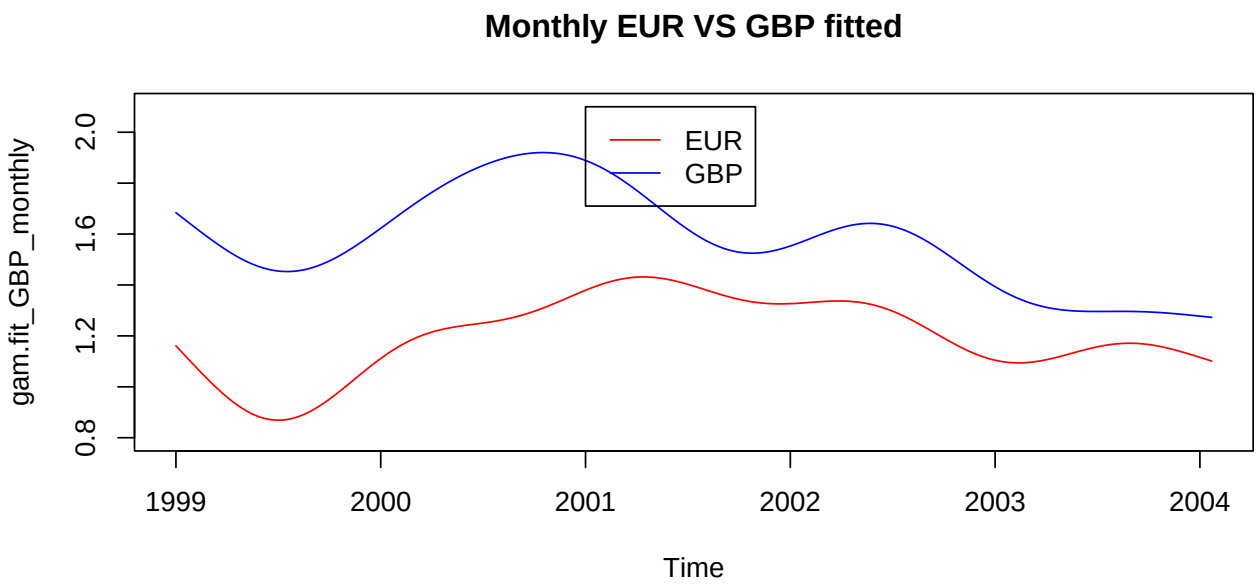
```
#Overlay the fitted GBP-MONTHLY and Original
ts.plot(GBP_month_ts,main='Monthly USD/GBP exchange rate Original VS Fitted',col='blue',ylim=c(0.5,3))
lines(gam.fit_GBP_monthly,col='red')
legend(x=2001,y=2.1,legend=c("Original","Fitted"),lty=1, col=c("blue","red"))
```




```
ts.plot(gam.fit_GBP_weekly,main='Weekly EUR VS GBP fitted',col='blue',ylim=c(0.8,2.1))
lines(gam.fit_EUR_weekly,col='red')
legend(x=2010,y=2.1,legend=c("EUR","GBP"),lty=1, col=c("red","blue"))
```



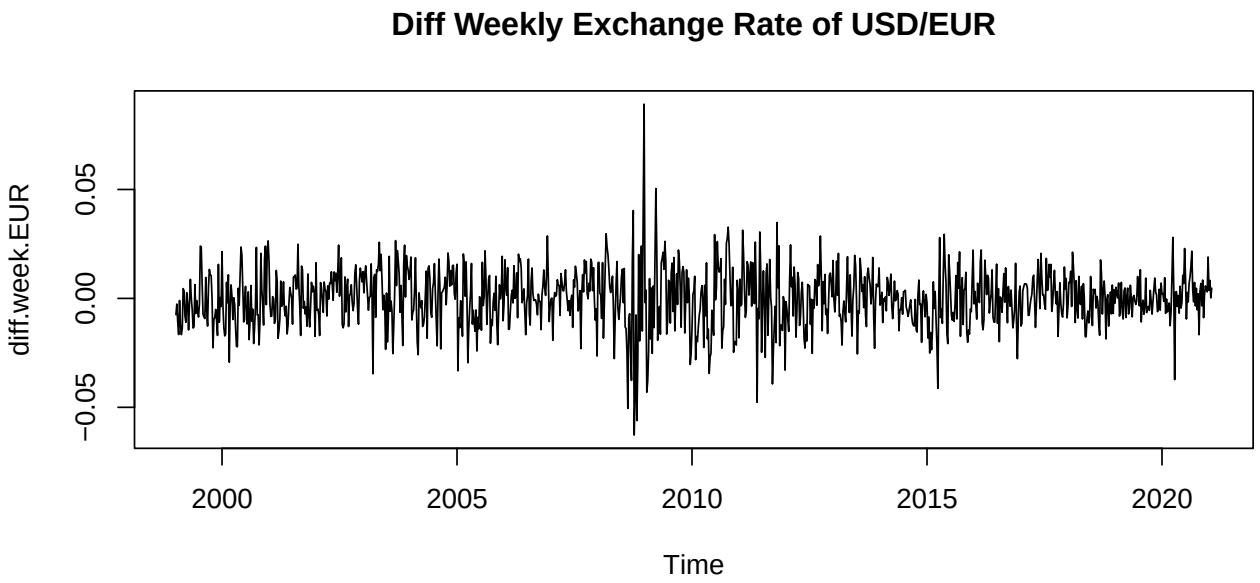

```
#monthly
ts.plot(gam.fit_GBP_monthly,main='Monthly EUR VS GBP fitted',col='blue',ylim=c(0.8,2.1))
lines(gam.fit_EUR_monthly,col='red')
legend(x=2001,y=2.1,legend=c("EUR","GBP"),lty=1, col=c("red","blue"))
```



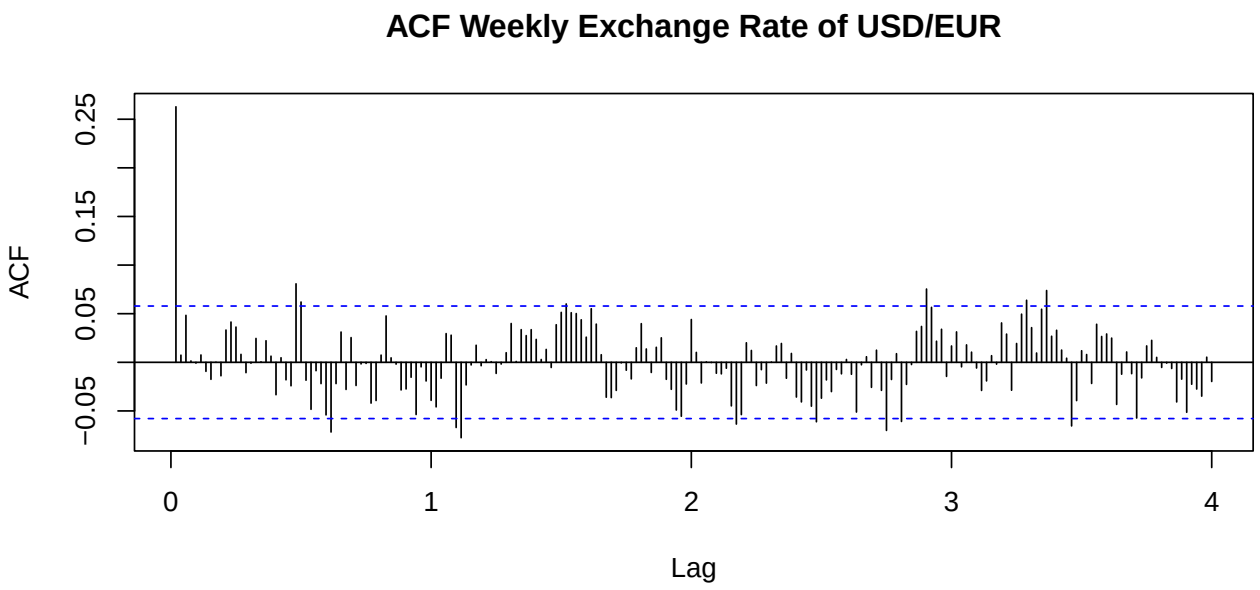
Response: Comparing Trend Estimation using weekly vs Monthly Data The Spline regression captures general trend in weekly and monthly plot but still missing some fluctuations.Generally, the weekly fitted trend estimation of both EUR and GBP captures more variance than the monthly ones; however none of the trend estimation are constant or showing any seasonalities/cyclincal patterns.

1d. Take the 1st order difference of the time series weekly vs monthly data. Plot the ACF plots and compare. How do the difference time series for weekly and monthly data compare in terms of stationarity? How do the difference time series for the two currencies compare in terms of serial dependence and stationarity?

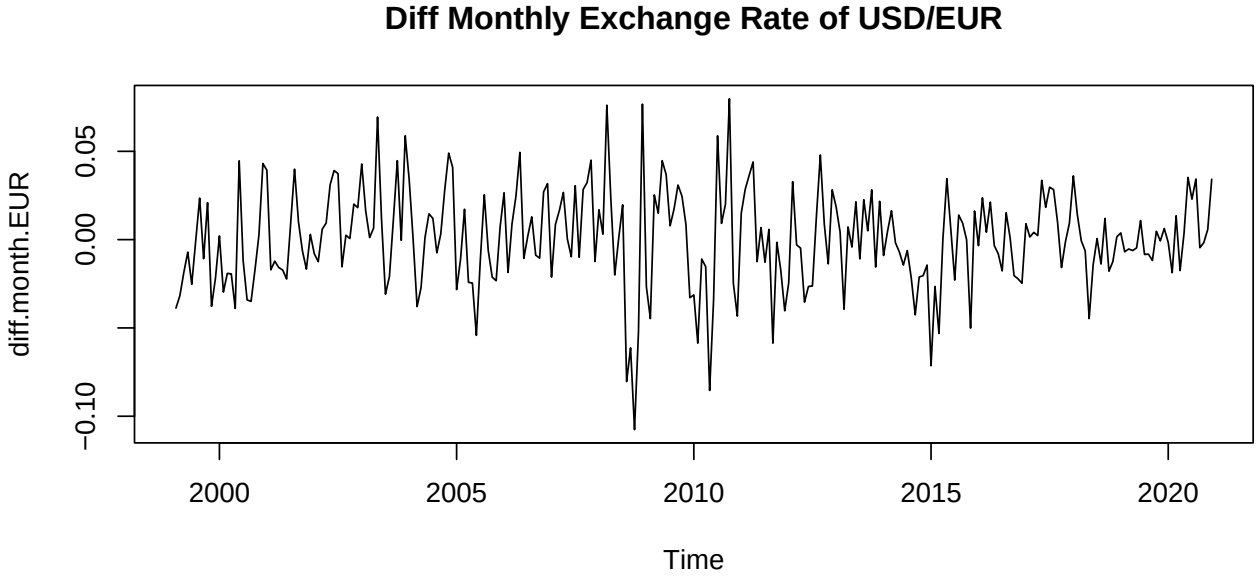
```
#Differencing Weeky data-EUR
week_EUR_ts = ts(weekly$EUR,start=1999,freq=52)
diff.week.EUR = diff(week_EUR_ts)
plot(diff.week.EUR,main = "Diff Weekly Exchange Rate of USD/EUR")
```



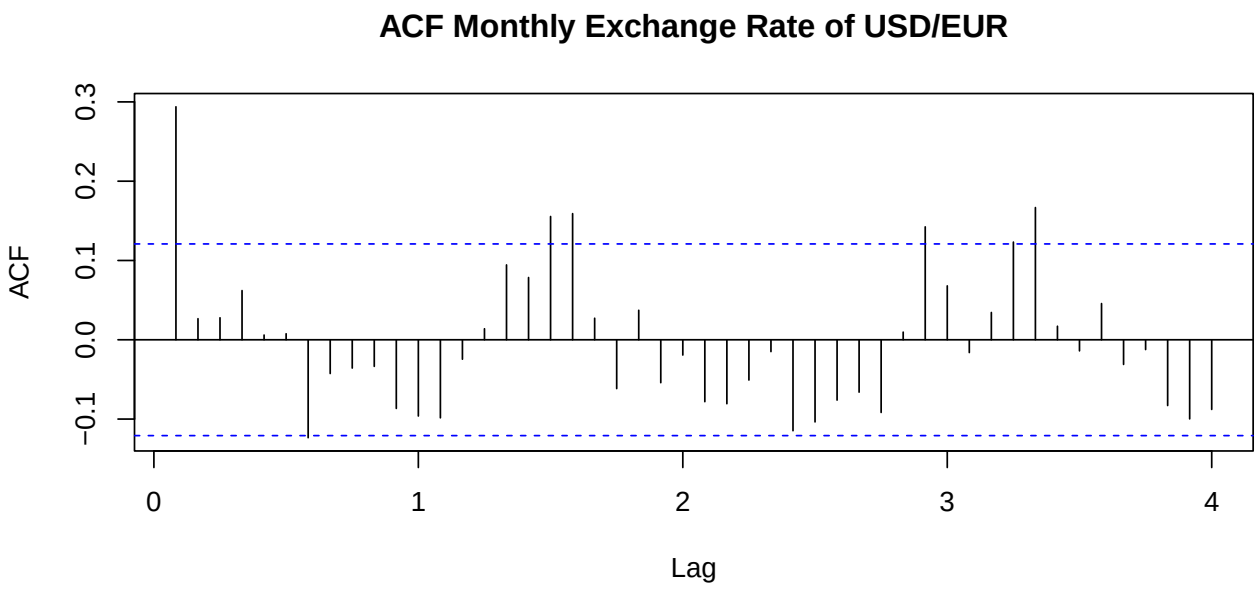

```
acf(diff.week.EUR,main='ACF Weekly Exchange Rate of USD/EUR',lag.max = 52*4)
```



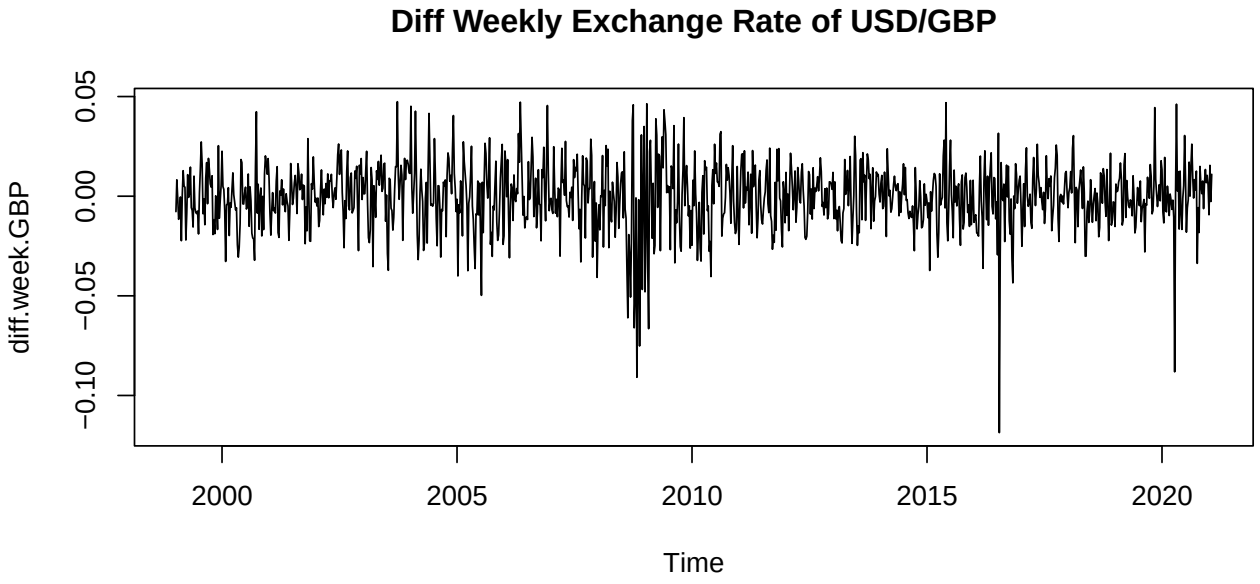

```
#Differencing Monthly data -EUR
month_EUR_ts = ts(monthly$EUR,start=1999,freq=12)
diff.month.EUR = diff(month_EUR_ts)
plot(diff.month.EUR,main = "Diff Monthly Exchange Rate of USD/EUR")
```



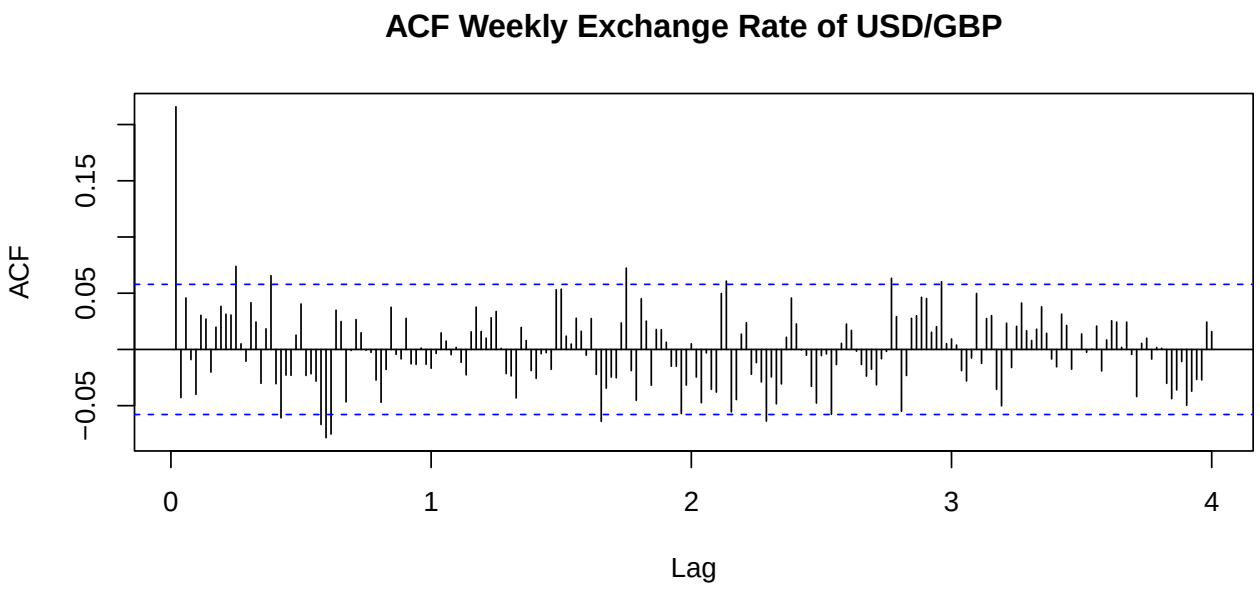

```
acf(diff.month.EUR,main='ACF Monthly Exchange Rate of USD/EUR',lag.max = 12*4)
```



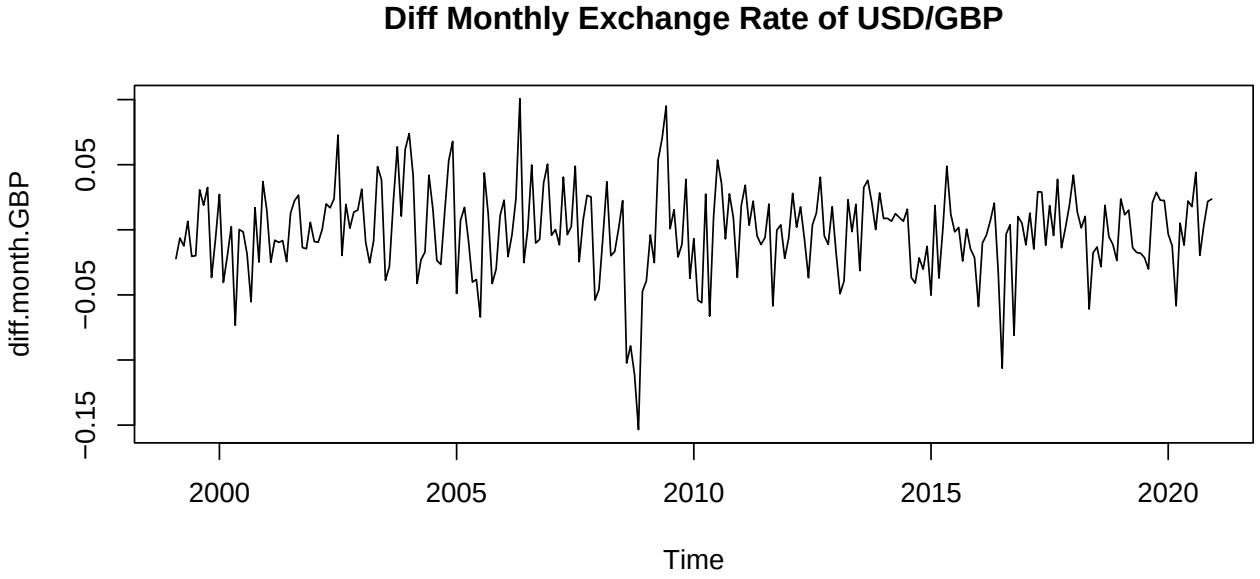

```
#Differencing Weeky data -EUR
week_GBP_ts = ts(weekly$GBP,start=1999,freq=52)
diff.week.GBP = diff(week_GBP_ts)
plot(diff.week.GBP,main = "Diff Weekly Exchange Rate of USD/GBP")
```



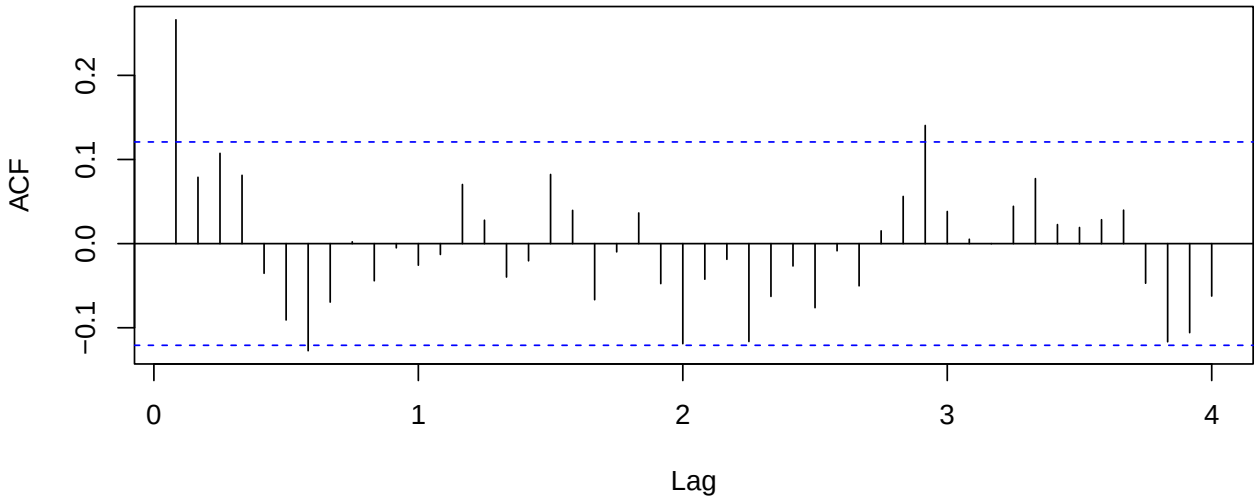

```
acf(diff.week.GBP,main='ACF Weekly Exchange Rate of USD/GBP',lag.max = 52*4)
```




```
#Differencing Monthly data -GBP
month_GBP_ts = ts(monthly$GBP,start=1999,freq=12)
diff.month.GBP = diff(month_GBP_ts)
plot(diff.month.GBP,main = "Diff Monthly Exchange Rate of USD/GBP")
```




```
acf(diff.month.GBP,main='ACF Monthly Exchange Rate of USD/GBP',lag.max = 12*4)
```




```
lag.length = 52*4
kpss.test(diff(week_EUR_ts),null="Trend")
```

```
##
##  KPSS Test for Trend Stationarity
##
## data:  diff(week_EUR_ts)
## KPSS Trend = 0.083197, Truncation lag parameter = 7, p-value = 0.1
```

```
Box.test(diff(week_EUR_ts), lag=lag.length, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  diff(week_EUR_ts)
## X-squared = 339.95, df = 208, p-value = 2.04e-08
```

```
kpss.test(diff(month_EUR_ts),null="Trend")
```

```
##
##  KPSS Test for Trend Stationarity
##
## data:  diff(month_EUR_ts)
## KPSS Trend = 0.074968, Truncation lag parameter = 5, p-value = 0.1
```

```
Box.test(diff(month_EUR_ts), lag=lag.length, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  diff(month_EUR_ts)
## X-squared = 285.04, df = 208, p-value = 0.0003096
```

```
kpss.test(diff(week_GBP_ts),null="Trend")
```

```
##
##  KPSS Test for Trend Stationarity
##
## data:  diff(week_GBP_ts)
```

```
## KPSS Trend = 0.059529, Truncation lag parameter = 7, p-value = 0.1
```

```
Box.test(diff(week_GBP_ts), lag=lag.length, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  diff(week_GBP_ts)
## X-squared = 296.28, df = 208, p-value = 5.547e-05
```

```
kpss.test(diff(month_GBP_ts),null="Trend")
```

```
##
##  KPSS Test for Trend Stationarity
##
## data:  diff(month_GBP_ts)
## KPSS Trend = 0.049235, Truncation lag parameter = 5, p-value = 0.1
```

```
Box.test(diff(month_GBP_ts), lag=lag.length, type="Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  diff(month_GBP_ts)
## X-squared = 217.81, df = 208, p-value = 0.3063
```

Response: Exploratory Analysis of 1st Order Difference Data The difference data show stationarity to certain extend. The EUR weekly shows more stationary than the EUR monthly. Overall, the weekly data is more stationary than the monthly data.For both EUR and GBP, the weekly data shows stronger serial dependence as the box tests return signifiant smaller P value compared to the monthly data.Also, we can tell that USD/EUR Exchange is more stationary and has higher seriel dependence compared to the GBP.

#Question 2. ARIMA Fitting and Forecasting: Weekly Data Analysis (23 points)

2a. Divide the data into training and testing data set, where the training data exclude the last eight weeks of data (November and December 2020) with the testing data including the last eight weeks of data. For both currency exchange rates and using the training datasets, use the iterative model to fit an ARIMA(p,d,q) model with max AR and MA orders of 8, and a differencing order of 1 or 2. Display the summary of the final model fit. Compare statistical significance of the coefficients. Would a lower order model be suggested based on the statistical significance of the coefficients?

Analyzing weekly data with ARIMA model fitting

```
daily <-na.locf(daily)
weekly <- daily
weekly$Date <- floor_date(weekly$Date, "week")
weekly <- aggregate(weekly[,2:3], by=list(weekly$Date), FUN=mean)
colnames(weekly[1]) <- "Date"

#EUR-weekly
EUR_week <- ts(weekly$EUR, start=1999, freq=52)
#divide the data into training and testing
n_forward=8
n = length(t(EUR_week))
nfit = n-n_forward
trainEUR.week = EUR_week[1:nfit]

## Fit an ARMA model to the 8-lag difference time series with *d=1*
```

```
norder = 8
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(trainEUR.week,order = c(p[i],1,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
  }
}
# Extract the "best" one according to AIC
aicv <- as.vector(aic)
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder_1 <- indexp[indexaic]-1
qorder_1 <- indexq[indexaic]-1

#fit the best ARIMA model
EUR_week_mode_1 <- arima(trainEUR.week,order=c(porder_1,1,qorder_1),method="ML")
print(EUR_week_mode_1)
```

```
##
## Call:
## arima(x = trainEUR.week, order = c(porder_1, 1, qorder_1), method = "ML")
##
## Coefficients:
##          ma1
##       0.2931
## s.e.  0.0295
##
## sigma^2 estimated as 0.0001586:  log likelihood = 3366.38,  aic = -6730.77
```

```
porder_1
```

```
## [1] 0
```

```
qorder_1
```

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

```
## Fit an EUR-WEEKLY ARMA model to the 8-lag difference time series with *d=2*
norder = 8
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(trainEUR.week,order = c(p[i],2,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
  }
}
# Extract the "best" one according to AIC
aicv <- as.vector(aic)
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder_2 <- indexp[indexaic]-1
```

```
qorder_2 <- indexq[indexaic]-1


# Fit the "best" arima model
EUR_week_model_2 <- arima(trainEUR.week, order = c(porder_2,2,qorder_2), method = "ML")
print(EUR_week_model_2)
```

```
##
## Call:
## arima(x = trainEUR.week, order = c(porder_2, 2, qorder_2), method = "ML")
##
## Coefficients:
##           ma1      ma2
##       -0.7062  -0.2938
## s.e.   0.0296   0.0295
##
## sigma^2 estimated as 0.0001587:  log likelihood = 3359.67,  aic = -6715.33
```

```
porder_2
```

```
## [1] 0
```

```
qorder_2
```

```
## [1] 2
```

```
#EUR_week_model_1 has a smaller AIC
```

```
daily <-na.locf(daily)
weekly <- daily
weekly$Date <- floor_date(weekly$Date, "week")
weekly <- aggregate(weekly[,2:3], by=list(weekly$Date), FUN=mean)
colnames(weekly[1]) <- "Date"

#GBP-weekly
GBP_week <- ts(weekly$GBP, start=1999, freq=52)
#divide the data into training and testing
n_forward=8
n = length(t(GBP_week))
nfit = n-n_forward
trainGBP.week = GBP_week[1:nfit]

## Fit an ARMA model to the 8-lag difference time series with *d=1*
norder = 8
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(trainGBP.week,order = c(p[i],1,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
  }
}
# Extract the "best" one according to AIC
aicv <- as.vector(aic)
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
```

```
indexaic <- which(aicv == min(aicv))
porder_3 <- indexp[indexaic]-3
qorder_3 <- indexq[indexaic]-3

#fit the best ARIMA model
GBP_week_mode_1 <- arima(trainGBP.week,order=c(porder_3,1,qorder_3),method="ML")
print(GBP_week_mode_1)
```

```
##
## Call:
## arima(x = trainGBP.week, order = c(porder_3, 1, qorder_3), method = "ML")
##
## Coefficients:
##          ma1
##       0.2632
## s.e.  0.0312
##
## sigma^2 estimated as 0.0002622:  log likelihood = 3080.07,  aic = -6158.14
```

```
porder_3
```

```
## [1] 0
```

```
qorder_3
```

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

```
## Fit an GBP-WEEKLY ARMA model to the 8-lag difference time series with *d=2*
norder = 8
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(trainGBP.week,order = c(p[i],2,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
  }
}
# Extract the "best" one according to AIC
aicv <- as.vector(aic)
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder_4 <- indexp[indexaic]-1
qorder_4 <- indexq[indexaic]-1


# Fit the "best" arima model
GBP_week_model_2 <- arima(trainGBP.week, order = c(porder_4,2,qorder_4), method = "ML")
print(GBP_week_model_2)
```

```
##
## Call:
## arima(x = trainGBP.week, order = c(porder_4, 2, qorder_4), method = "ML")
##
## Coefficients:
##          ar1      ar2      ma1     ma2      ma3      ma4
```

```
##       -0.5726  -0.9685  -0.1657  0.2731  -0.8285  -0.2789
## s.e.   0.0221   0.0210   0.0360  0.0261   0.0310   0.0303
##
## sigma^2 estimated as 0.0002576:  log likelihood = 3084.06,  aic = -6156.12
```

```
porder_4
```

```
## [1] 2
```

```
qorder_4
```

```
## [1] 4
```

Response: Analysis of the ARIMA Fit for the Weekly Data Both USD/EUR and usd/GBP have a lower AICC value when d=1; it means with a differecing order of 1, the data is better fitted into the model hence more plausible to predict. We also pick up d=1 because it is simpler. Overall, the simplicity and lower AICC both support that thhe order selection of d is better to be at 1 compared to 2.

```
## p-value function for the z-test taking as input the test statistic
EUR_week_mode_1
```

```
##
## Call:
## arima(x = trainEUR.week, order = c(porder_1, 1, qorder_1), method = "ML")
##
## Coefficients:
##          ma1
##       0.2931
## s.e.  0.0295
##
## sigma^2 estimated as 0.0001586:  log likelihood = 3366.38,  aic = -6730.77
```

```
GBP_week_mode_1
```

```
##
## Call:
## arima(x = trainGBP.week, order = c(porder_3, 1, qorder_3), method = "ML")
##
## Coefficients:
##          ma1
##       0.2632
## s.e.  0.0312
##
## sigma^2 estimated as 0.0002622:  log likelihood = 3080.07,  aic = -6158.14
```

```
pvalue.coef <- function(tv) {
  2 * (1 - pnorm(abs(tv)))
}
t.EURweek_1 <-as.numeric(GBP_week_mode_1$coef)/as.numeric(sqrt(diag(EUR_week_mode_1$var.coef)))

t.GBPweek_1  <-as.numeric(GBP_week_mode_1$coef)/as.numeric(sqrt(diag(GBP_week_mode_1$var.coef)))

pvalues.EUR_1 <- sapply(t.EURweek_1, pvalue.coef)
pvalues.EUR_1
```

```
## [1] 0
```

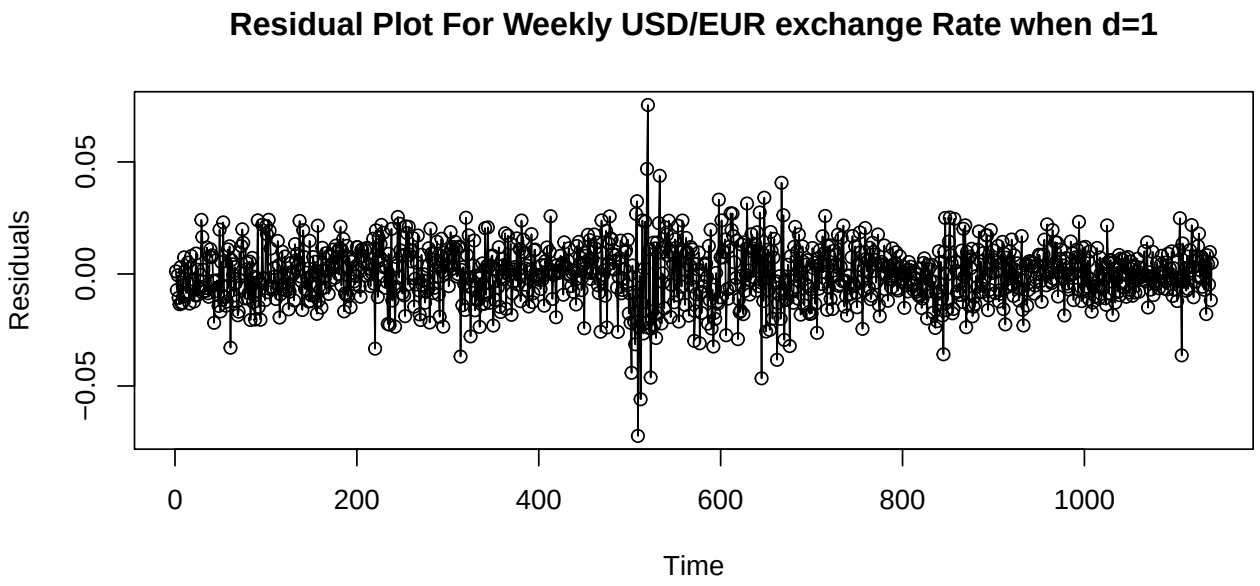
```
pvalues.GBP_1 <- sapply(t.GBPweek_1, pvalue.coef)
pvalues.GBP_1
```

```
## [1] 0
```

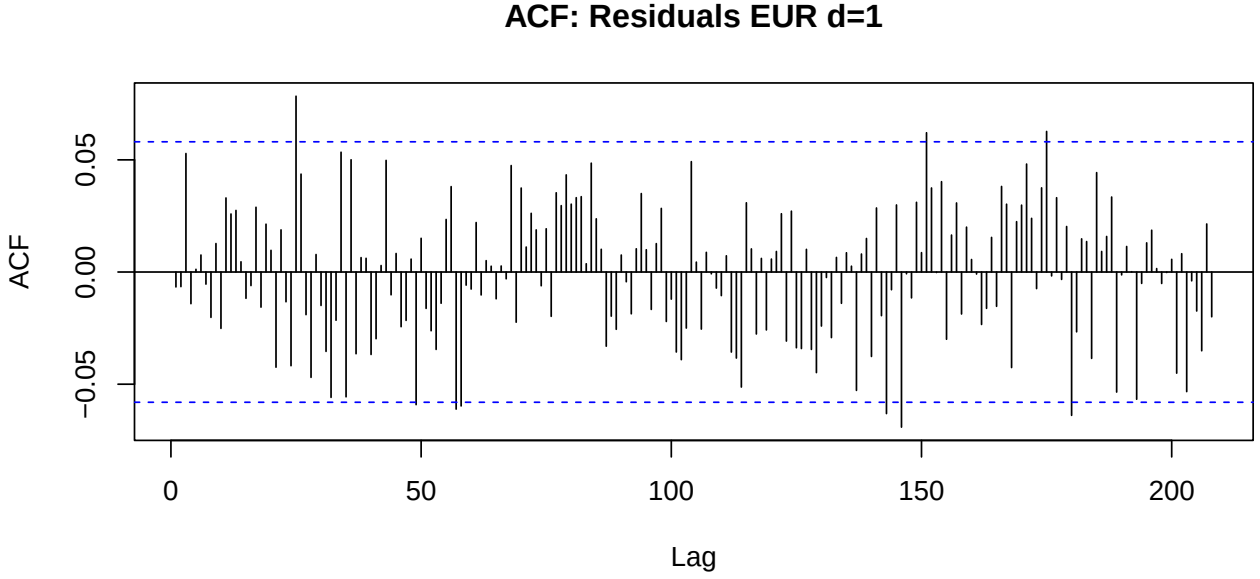
##Significant analysis Both P values equals to 0, meaning that the regression coefficients corresponding to those predictors are statistically significant. Also all the t-values are less than 0.05, showing the coefficients are plausible equal to the true parameters. They are therefore a good model fit. Both EUR and GBP with d=1 performs better than the model with d=2 when comparing AICC values. The GBP model with d=1 is less complex than when d=2 since the AR order here is 0 when d=1.

2b. Evaluate the model residuals using the ACF and PACF plots as well as hypothesis testing for serial correlation for the models applied to the weekly data for the two currencies. Does the model fit the time series data? Compare the model fit for the two currency exchange rates.

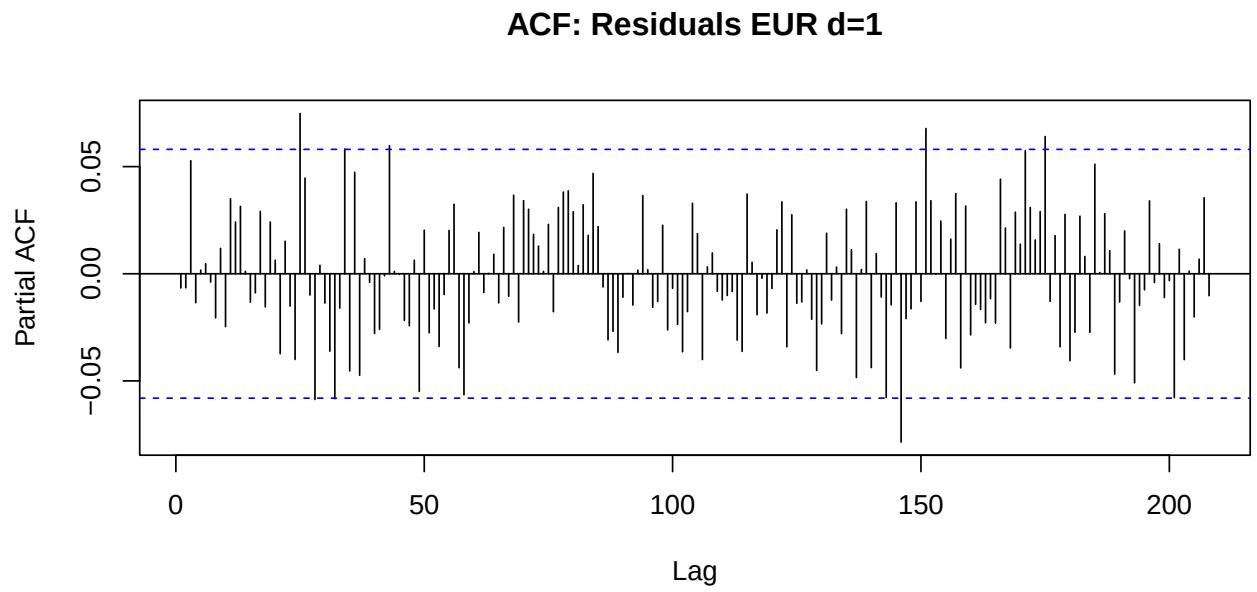
```
resids_1 <- resid(EUR_week_mode_1)
plot(resids_1, ylab='Residuals',type='o',main="Residual Plot For Weekly USD/EUR exchange Rate when d=1")
```



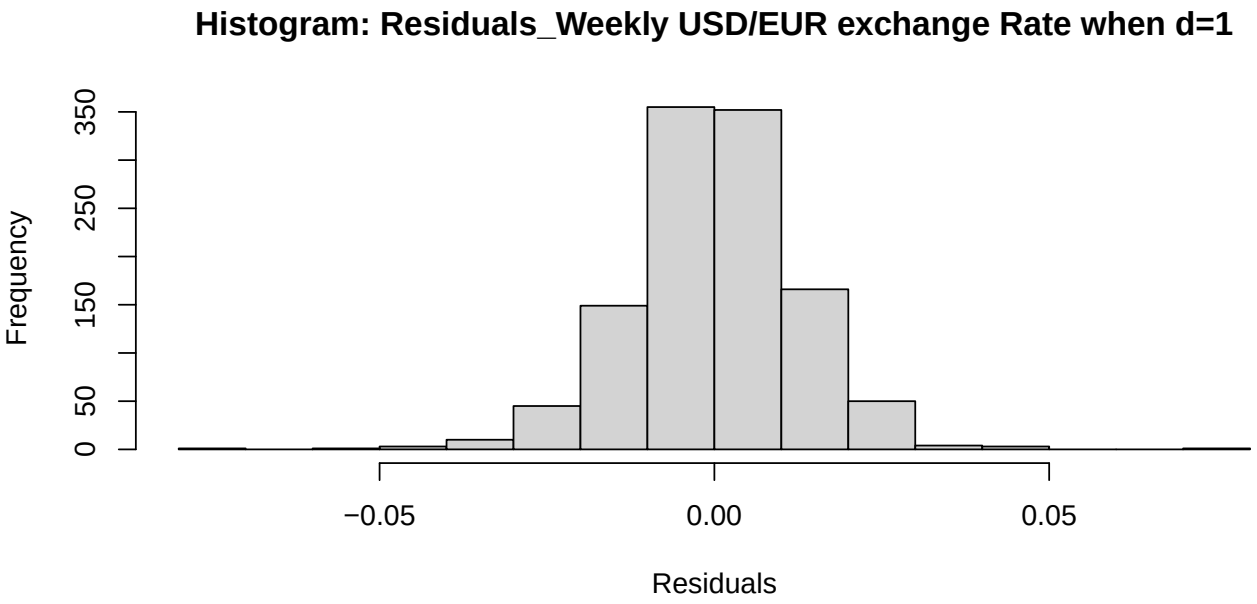

```
acf(resids_1,main="ACF: Residuals EUR d=1",lag.max=52*4)
```



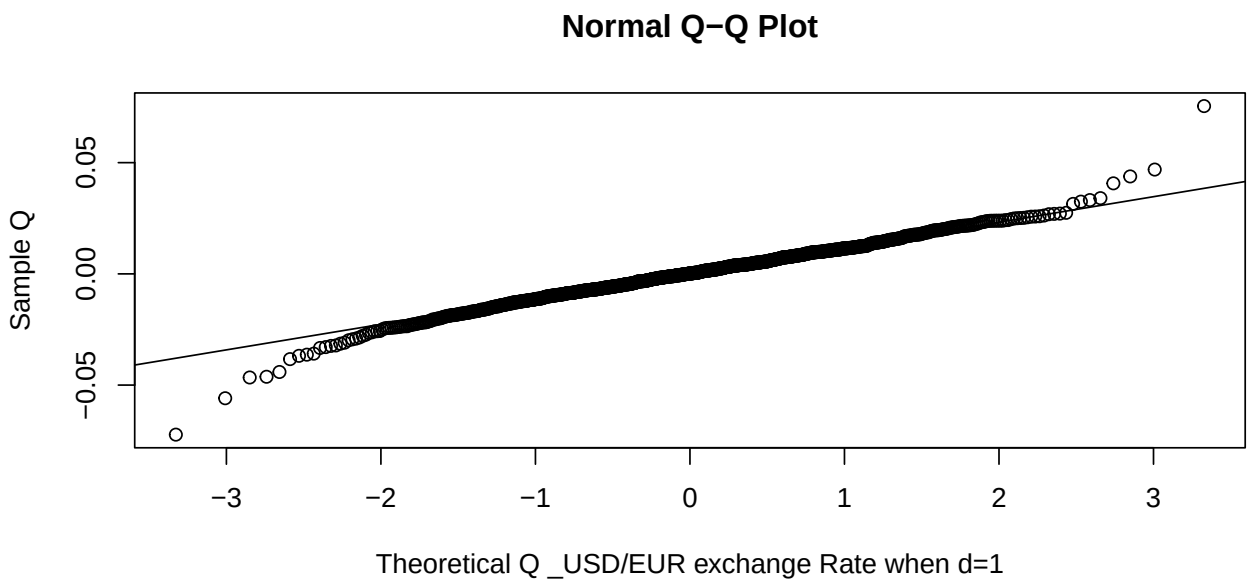

```
pacf(resids_1,main="ACF: Residuals EUR d=1",lag.max=52*4)
```




```
hist(resids_1,xlab='Residuals',main='Histogram: Residuals_Weekly USD/EUR exchange Rate when d=1')
```




```
qqnorm(resids_1,ylab="Sample Q",xlab="Theoretical Q _USD/EUR exchange Rate when d=1")
qqline(resids_1)
```



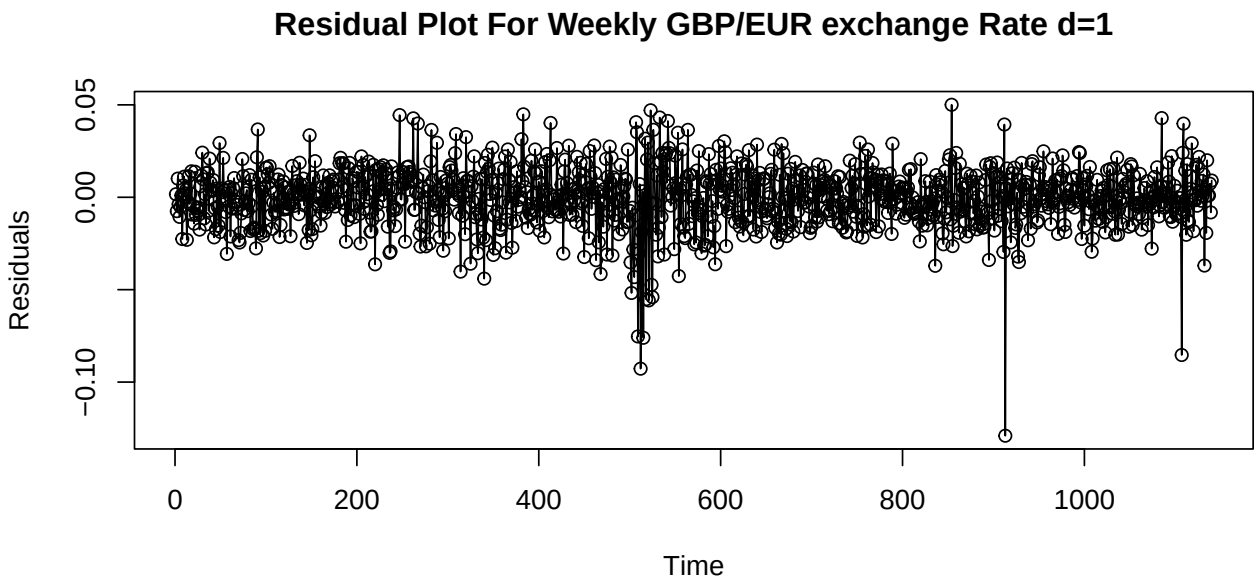

```
# Test and see if residuals are correlated
Box.test(EUR_week_mode_1$resid, lag = (porder_1+qorder_1+1), type = "Box-Pierce", fitdf = (porder_1+qorc
```

```
##
##  Box-Pierce test
##
## data:  EUR_week_mode_1$resid
## X-squared = 0.098854, df = 1, p-value = 0.7532
```

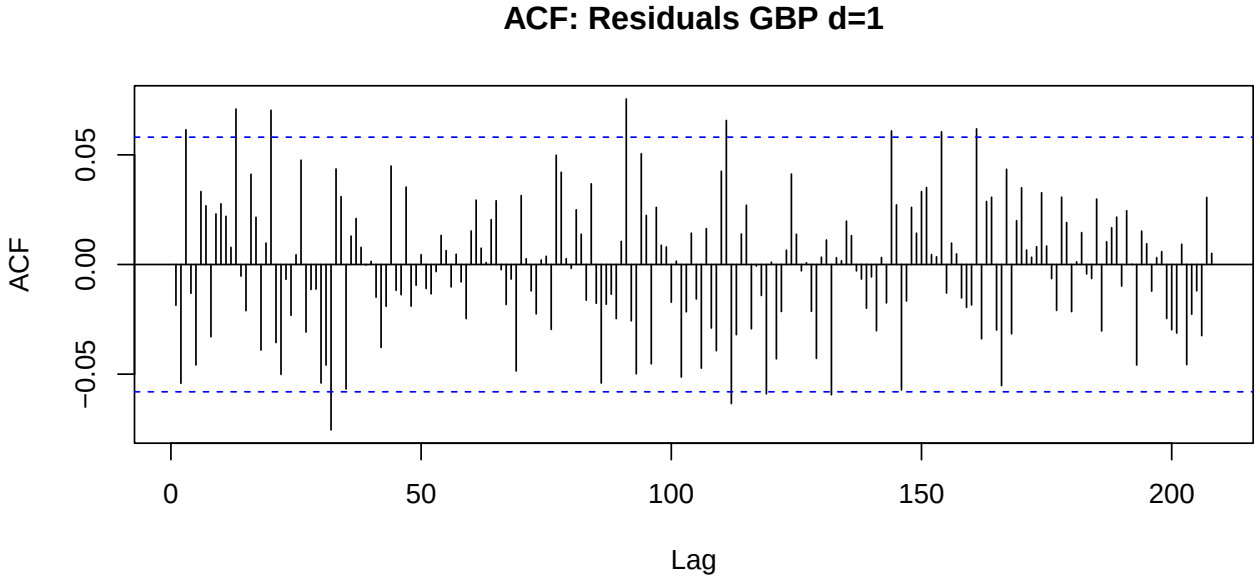
```
Box.test(EUR_week_mode_1$resid, lag = (porder_1+qorder_1+1), type = "Ljung-Box", fitdf = (porder_1+qord
```

```
##
##  Box-Ljung test
##
## data:  EUR_week_mode_1$resid
## X-squared = 0.099157, df = 1, p-value = 0.7528
```

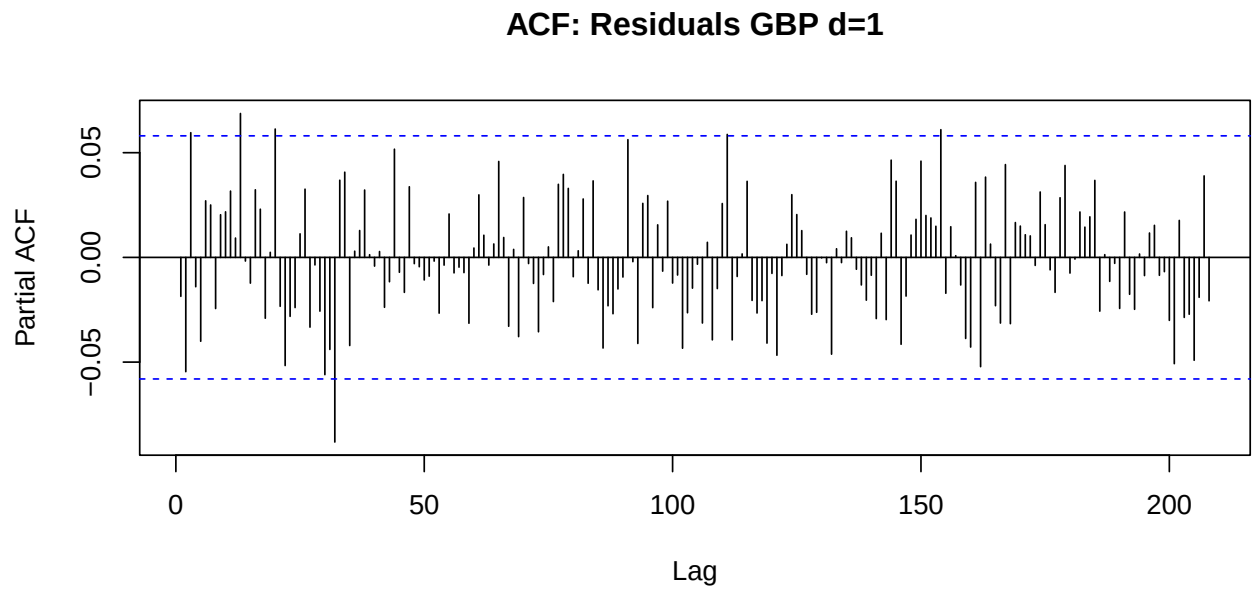
```
resids_2 <- resid(GBP_week_mode_1)
plot(resids_2, ylab='Residuals',type='o',main="Residual Plot For Weekly GBP/EUR exchange Rate d=1")
```



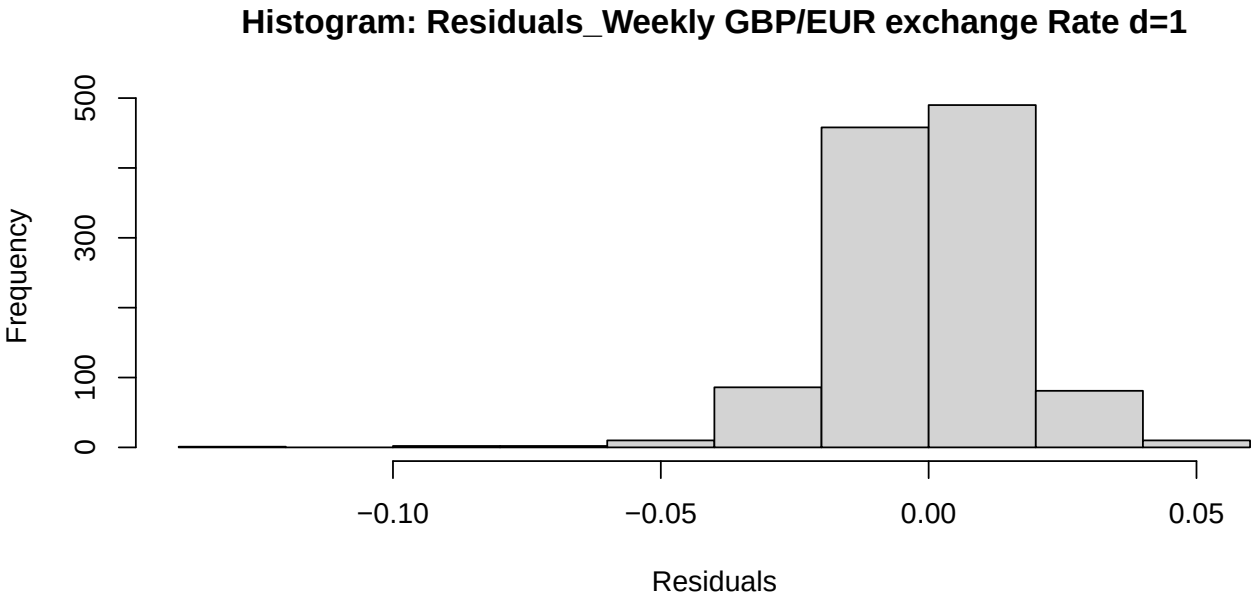

```
acf(resids_2,main="ACF: Residuals GBP d=1",lag.max=52*4)
```



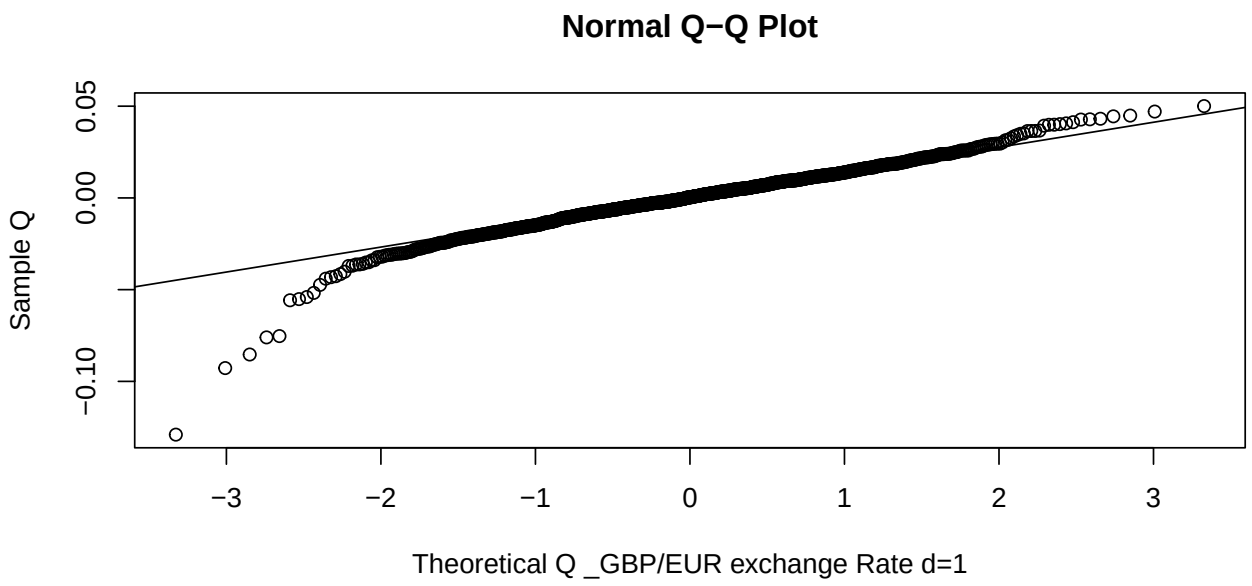

```
pacf(resids_2,main="ACF: Residuals GBP d=1",lag.max=52*4)
```




```
hist(resids_2,xlab='Residuals',main='Histogram: Residuals_Weekly GBP/EUR exchange Rate d=1')
```




```
qqnorm(resids_2,ylab="Sample Q",xlab="Theoretical Q _GBP/EUR exchange Rate d=1")
qqline(resids_2)
```




```
# Test and see if residuals are correlated
Box.test(GBP_week_mode_1$resid, lag = (porder_3+qorder_3+1), type = "Box-Pierce", fitdf = (porder_3+qor
```

```
##
##  Box-Pierce test
##
## data:  GBP_week_mode_1$resid
## X-squared = 3.7433, df = 1, p-value = 0.05302
```

```
Box.test(GBP_week_mode_1$resid, lag = (porder_3+qorder_3+1), type = "Ljung-Box", fitdf = (porder_3+qord
```

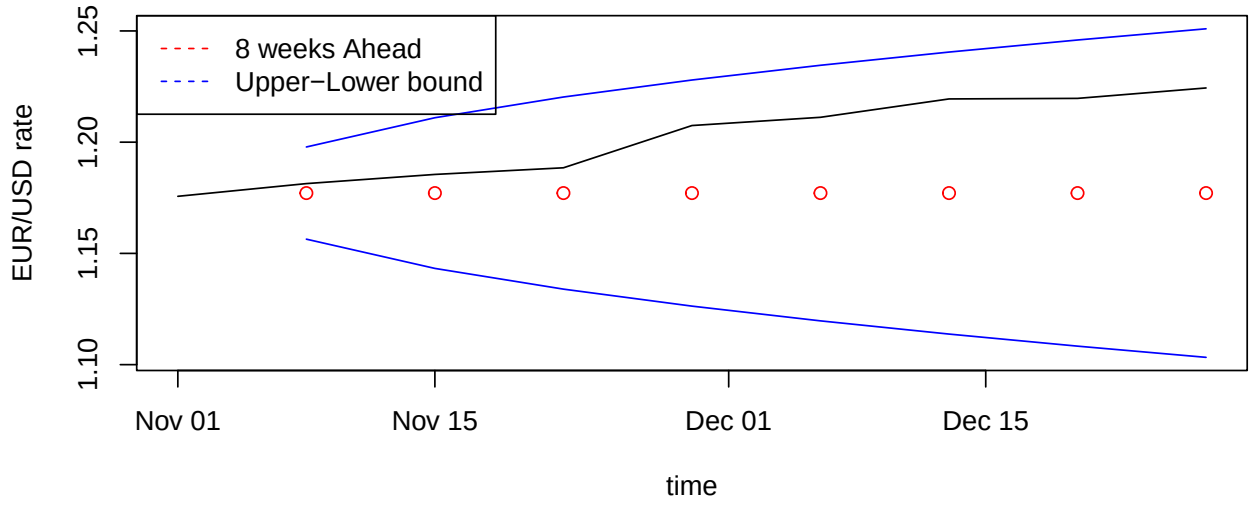
```
##
##  Box-Ljung test
##
## data:  GBP_week_mode_1$resid
## X-squared = 3.7561, df = 1, p-value = 0.05262
```

Response: Residual Analysis The mean and variance are constant in both EUR and GBP residual plots with several exceptions. The ACF and PACF both show that residuals are non-related, so we can treat the residuals as white noise. The P value is greater than 0.05 so we reject the null hypothesis, which means there is no significant auto correlations for these two exchange currencies.

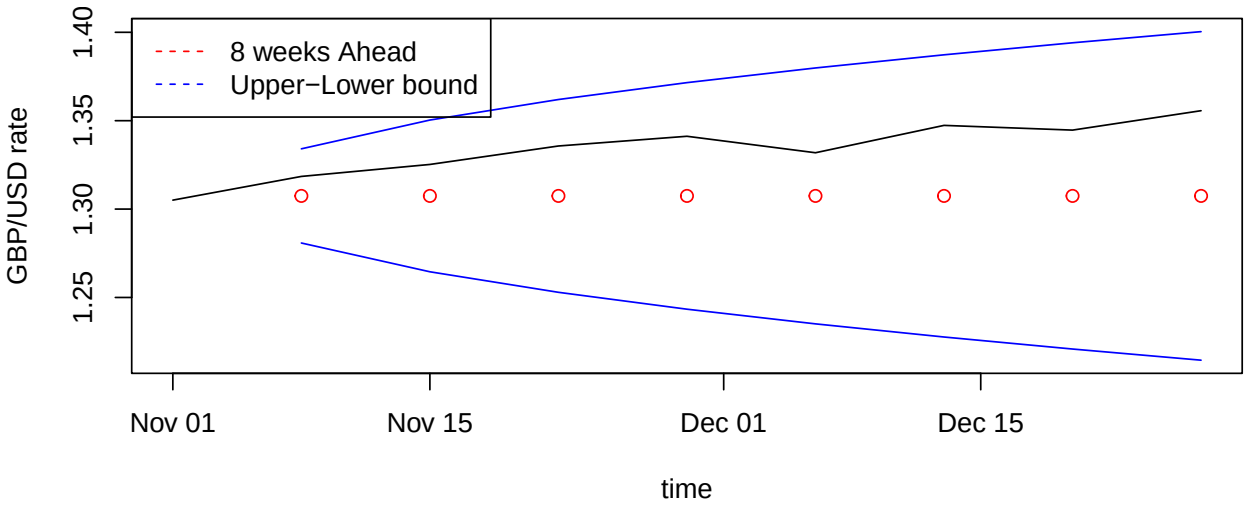
**2c.* For each currency exchange, apply the model identified in (2a) and forecast the last eight weeks of data. Plot the predicted data to compare the predicted values to the actual observed ones. Include 90% confidence intervals for the forecasts in the corresponding plots.

```
## Forecasting USD/EUR with ARMA, 8 Weeks Ahead:
daily <- na.locf(daily)

weekly <- daily
weekly$Date <- floor_date(weekly$Date, "week")
weekly <- aggregate(weekly[, 2:3], by = list(weekly$Date), FUN = mean)
colnames(weekly)[1] <- "Date"

week_EUR_ts = ts(weekly$EUR,start=1999,freq=52)
month_EUR_ts = ts(monthly$EUR,start=1999,freq=12)
week_GBP_ts = ts(weekly$GBP,start=1999,freq=52)
month_GBP_ts = ts(monthly$GBP,start=1999,freq=12)
```

```
## Forecasting USD/EUR with ARMA, 8 Weeks Ahead:
week_EUR_ts <- ts(weekly$EUR, start=1999, freq=52)

n <- length(t(week_EUR_ts))
nfit = n - 8
EUR_weekly_pred <- predict(EUR_week_mode_1,n.ahead=8)
ubound_EUR_weekly <- EUR_weekly_pred$pred+1.645*EUR_weekly_pred$se
lbound_EUR_weekly<- EUR_weekly_pred$pred-1.645*EUR_weekly_pred$se
ymin <- min(lbound_EUR_weekly)
ymax <- max(ubound_EUR_weekly)
plot(weekly$Date[(1148-8):1148], week_EUR_ts[(1148-8):1148], type = "l", ylim=c(ymin,ymax), xlab = "tim
points(weekly$Date[1141:1148], EUR_weekly_pred$pred, col = "red")
lines(weekly$Date[1141:1148],ubound_EUR_weekly, col = "blue")
lines(weekly$Date[1141:1148],lbound_EUR_weekly, col = "blue")
legend('topleft', legend=c("8 weeks Ahead ","Upper-Lower bound"),lty = 2, col=c("red","blue"))
```




```
n <- length(t(week_GBP_ts))
nfit = n - 8
GBP_weekly_pred <- predict(GBP_week_mode_1,n.ahead=8)
ubound_GBP_weekly <- GBP_weekly_pred$pred+1.645*GBP_weekly_pred$se
lbound_GBP_weekly<- GBP_weekly_pred$pred-1.645*GBP_weekly_pred$se
ymin <- min(lbound_GBP_weekly)
ymax <- max(ubound_GBP_weekly)
plot(weekly$Date[(1148-8):1148], week_GBP_ts[(1148-8):1148], type = "l", ylim=c(ymin,ymax), xlab = "tim
points(weekly$Date[1141:1148], GBP_weekly_pred$pred, col = "red")
lines(weekly$Date[1141:1148],ubound_GBP_weekly, col = "blue")
lines(weekly$Date[1141:1148],lbound_GBP_weekly, col = "blue")
legend('topleft', legend=c("8 weeks Ahead ","Upper-Lower bound"),lty = 2, col=c("red","blue"))
```



2d. Calculate Mean Absolute Percentage Error (MAPE) and Precision Measure (PM). How many observations are within the prediction bands? Compare the accuracy of the predictions for the two time series using these two measures.

```
forcast_week_EUR = EUR_weekly_pred$pred
real_EUR_weekly = week_EUR_ts[(nfit+1):n]
#Mean Absolute Percentage Measure (MAPE)
mean(abs(forcast_week_EUR-real_EUR_weekly)/real_EUR_weekly)
```

```
## [1] 0.02271062
```

```
#Precision Measure (PM).
sum((forcast_week_EUR-real_EUR_weekly)^2)/sum((real_EUR_weekly-mean(real_EUR_weekly))^2)
```

```
## [1] 3.960071
```

```
forcast_week_GBP = GBP_weekly_pred$pred
real_GBP_weekly = week_GBP_ts[(nfit+1):n]
#Mean Absolute Percentage Measure (MAPE)
mean(abs(forcast_week_GBP-real_GBP_weekly)/real_GBP_weekly)
```

```
## [1] 0.02240783
```

```
#Precision Measure (PM).
sum((forcast_week_GBP-real_GBP_weekly)^2)/sum((real_GBP_weekly-mean(real_GBP_weekly))^2)
```

```
## [1] 7.960084
```

Response: Prediction Accuracy The smaller the MAPE is, the better the model is and usually we consider MAPE less than 10 is good. In this analysis, both USD/EUR and UDS/GBP has a 0.02 MAPE meaning it may accurately forecast the data. However MAPE is not always the best approach to evaluate prediction accuracy, while precision Measure is mostly frequently used in MLE models. The smaller PM is the better the narrow the proportion between the variability in the prediction and the variability in the new data. In other words, the smaller the PM is, the more preceise the measure is. We see a 3.96 PM in USD/EUR exchange and a 7.96 PM in USD/GBP exchange rate; hence the USD/EUR model is better than the USD/GBP weekly exchange rate model.

#Question 3. ARIMA Fitting: Monthly Data Analysis (17 points)

3a. Divide the data into training and testing data set, where the training data exclude the last two months of data (November and December 2020) with the testing data including the last two months. For both currency exchange rates and using the training datasets, use the iterative model to fit an ARIMA(p,d,q) model with

max AR and MA orders of 8, and a differencing order of 1 or 2. Display the summary of the final model fit. Compare statistical significance of the coefficients. Compare the order selection from using monthly versus weekly data for each of the two currencies.

```
#divide the data into training and testing
trainEUR.month =  month_EUR_ts[1:nfit]
n_forward=2
n = length(t(month_EUR_ts))
nfit = n-n_forward


## Fit an ARMA model to the 8-lag difference time series with *d=1*
n=length(trainEUR.month)
norder = 8
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(trainEUR.month,order = c(p[i],1,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
  }
}
# Extract the "best" one according to AIC
aicv <- as.vector(aic)
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder.1 <- indexp[indexaic]-1
qorder.1 <- indexq[indexaic]-1

#fit the best ARIMA model
EUR_month_mode_1 <- arima(trainEUR.month,order=c(porder.1,1,qorder.1),method="ML")
print(EUR_month_mode_1)
```

```
##
## Call:
## arima(x = trainEUR.month, order = c(porder.1, 1, qorder.1), method = "ML")
##
## Coefficients:
##          ma1
##       0.3074
## s.e.  0.0563
##
## sigma^2 estimated as 0.000693:  log likelihood = 583.36,  aic = -1164.71
```

```
porder.1
```

```
## [1] 0
```

```
qorder.1
```

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

```
#d=2
aic = matrix(0,norder,norder)

for(i in 1:norder){
```

```
  for(j in 1:norder){
    modij = stats::arima(trainEUR.month,order = c(p[i],2,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
  }
}


# Extract the one according to AIC
aicv <- as.vector(aic)
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder.2 <- indexp[indexaic]-1
qorder.2 <- indexq[indexaic]-1


# Fit the arima model
EUR_month_model_2 <- arima(trainEUR.month, order = c(porder.2,2,qorder.2), method = "ML")
print(EUR_month_model_2)
```

```
##
## Call:
## arima(x = trainEUR.month, order = c(porder.2, 2, qorder.2), method = "ML")
##
## Coefficients:
##           ar1     ma1      ma2      ma3
##       -0.9567  0.3057  -0.9447  -0.3609
## s.e.   0.0406  0.0715   0.0393   0.0602
##
## sigma^2 estimated as 0.0006827:  log likelihood = 580.56,  aic = -1153.11
```

```
#divide the data into training and testing
trainGBP.month =  month_GBP_ts[1:nfit]
n_forward=2
n = length(t(month_GBP_ts))
nfit = n-n_forward


## Fit an ARMA model to the 8-lag difference time series with *d=1*
n=length(trainGBP.month)
norder = 8
p = c(1:norder)-1; q = c(1:norder)-1
aic = matrix(0,norder,norder)
for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(trainGBP.month,order = c(p[i],1,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
  }
}
# Extract the "best" one according to AIC
aicv <- as.vector(aic)
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
```

```
porder.3 <- indexp[indexaic]-1
qorder.3 <- indexq[indexaic]-1

#fit the best ARIMA model
GBP_month_mode_1 <- arima(trainGBP.month,order=c(porder.3,1,qorder.3),method="ML")
print(GBP_month_mode_1)
```

```
##
## Call:
## arima(x = trainGBP.month, order = c(porder.3, 1, qorder.3), method = "ML")
##
## Coefficients:
##          ar1      ar2     ar3      ma1     ma2
##       0.9852  -1.1422  0.2930  -0.7424  1.0000
## s.e.  0.0597   0.0476  0.0591   0.0141  0.0167
##
## sigma^2 estimated as 0.0009942:  log likelihood = 529.77,  aic = -1049.53
```

```
porder.3
```

```
## [1] 3
```

```
qorder.3
```

```
## [1] 2
```

```
#d=2
aic = matrix(0,norder,norder)

for(i in 1:norder){
  for(j in 1:norder){
    modij = stats::arima(trainGBP.month,order = c(p[i],2,q[j]), method='ML')
    aic[i,j] = modij$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
  }
}


# Extract the one according to AIC
aicv <- as.vector(aic)
indexp <- rep(c(1:norder),norder)
indexq <- rep(c(1:norder),each=norder)
indexaic <- which(aicv == min(aicv))
porder.4 <- indexp[indexaic]-1
qorder.4 <- indexq[indexaic]-1


# Fit the arima model
GBP_month_model_2 <- arima(trainGBP.month, order = c(porder.4,2,qorder.4), method = "ML")
print(GBP_month_model_2)
```

```
##
## Call:
## arima(x = trainGBP.month, order = c(porder.4, 2, qorder.4), method = "ML")
##
## Coefficients:
##          ar1      ar2     ar3      ma1     ma2      ma3
```

```
##       0.9891  -1.1447  0.2968  -1.7424  1.7424  -1.0000
## s.e.  0.0601   0.0478  0.0594   0.0189  0.0278   0.0209
##
## sigma^2 estimated as 0.0009971:  log likelihood = 524.94,  aic = -1037.88
```

```
EUR_month_mode_1
```

```
##
## Call:
## arima(x = trainEUR.month, order = c(porder.1, 1, qorder.1), method = "ML")
##
## Coefficients:
##          ma1
##       0.3074
## s.e.  0.0563
##
## sigma^2 estimated as 0.000693:  log likelihood = 583.36,  aic = -1164.71
```

```
GBP_month_mode_1
```

```
##
## Call:
## arima(x = trainGBP.month, order = c(porder.3, 1, qorder.3), method = "ML")
##
## Coefficients:
##          ar1      ar2     ar3      ma1     ma2
##       0.9852  -1.1422  0.2930  -0.7424  1.0000
## s.e.  0.0597   0.0476  0.0591   0.0141  0.0167
##
## sigma^2 estimated as 0.0009942:  log likelihood = 529.77,  aic = -1049.53
```

```
#significance and coefficience
t.EURmonth <-as.numeric(EUR_month_mode_1$coef)/as.numeric(sqrt(diag(EUR_month_mode_1$var.coef)))

t.GBPmonth  <-as.numeric(GBP_month_mode_1$coef)/as.numeric(sqrt(diag(GBP_month_mode_1$var.coef)))

pvalues.EUR <- sapply(t.EURmonth, pvalue.coef)
pvalues.EUR
```

```
## [1] 4.692422e-08
```

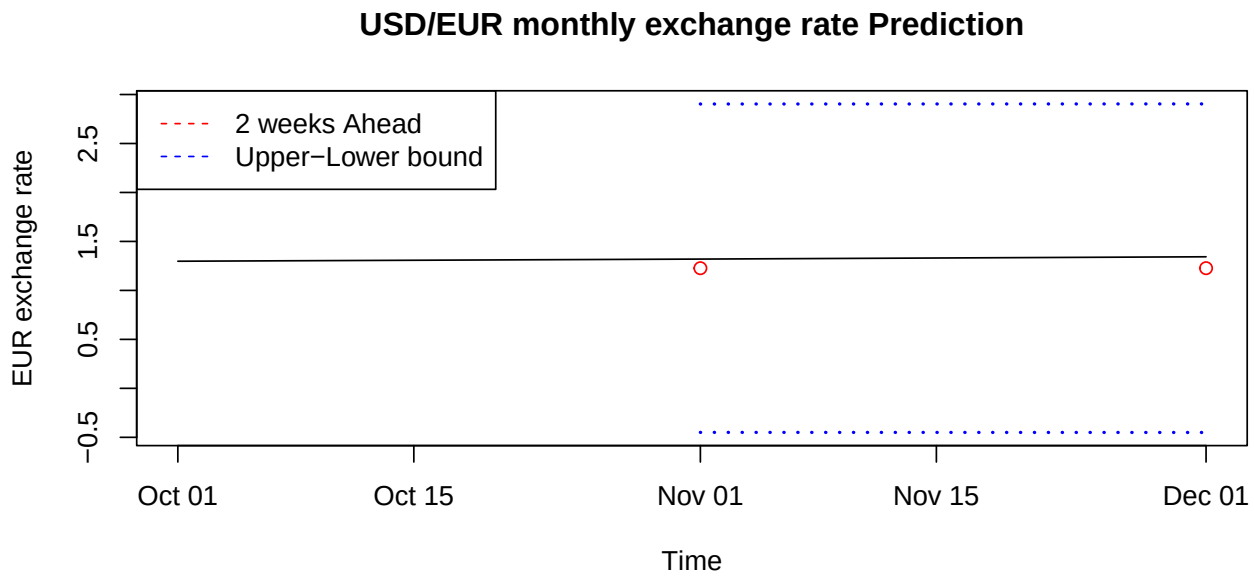
```
pvalues.GBP <- sapply(t.GBPmonth, pvalue.coef)
pvalues.GBP
```

```
## [1] 0.00000e+00 0.00000e+00 7.07024e-07 0.00000e+00 0.00000e+00
```

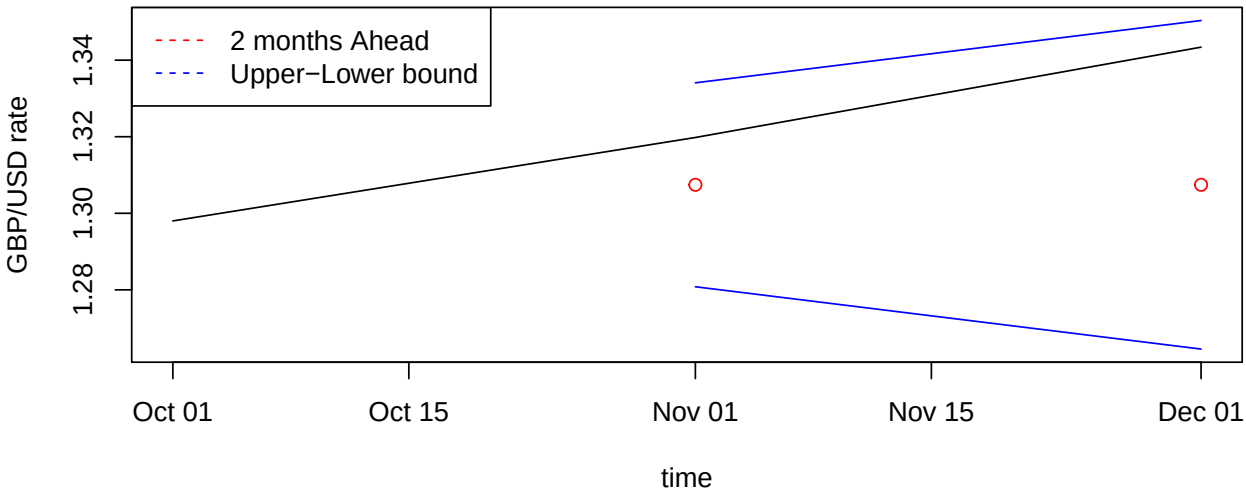
Response: Analysis of the ARIMA Fit for the Monthly Data As we want to lowest AIC to find the best fit model, from the result above, we can conclude that both USD/EUR and USD/GBP monthly exchange rate fit better in the d=1 order selection than when d=2, because the AIC value is lower than when d=1. When comparing the coefficience and significancant, we found p value are extremely small hence both USD/EUR and USD/GBP monthly exchange rate are statistically significant. Comparing the order selection, when it comes to order selection, the weekly has larger AIC so we select the weekly for both USD/EUR and USD/GBP for forecasting purpose.

3b. For each currency exchange, apply the model identified in (3a) and forecast the last two months of data. Plot the predicted data to compare the predicted values to the actual observed ones. Include 90% confidence intervals for the forecasts in the corresponding plots.

```
## Forecasting USD/EUR with ARMA, 2 months Ahead:
n = 264  #length of monthly$EUR
nfit = n - 2
EUR_monthly_pred = predict(EUR_month_mode_1,n.ahead=2)
ubound_EUR_monthly = EUR_monthly_pred$pred+1.645*EUR_monthly_pred$se
lbound_EUR_monthly = EUR_monthly_pred$pred-1.645*EUR_monthly_pred$se
ymin = min(lbound_EUR_monthly)
ymax = max(ubound_EUR_monthly)
plot(monthly$Date[262:264],month_GBP_ts[262:264],type="l", ylim=c(ymin,ymax), xlab="Time", ylab = 'EUR
points(monthly$Date[263:264],EUR_monthly_pred$pred,col="red")
lines(monthly$Date[263:264],ubound_EUR_monthly,lty=3,lwd= 2, col="blue")
lines(monthly$Date[263:264],lbound_EUR_monthly,lty=3,lwd= 2, col="blue")
legend('topleft', legend=c("2 weeks Ahead ","Upper-Lower bound"),lty = 2, col=c("red","blue"))
```




```
n <- 264 #length of monthly
nfit = n - 2 #length of monthly -2
GBP_monthly_pred <- predict(GBP_week_mode_1,n.ahead=2)
ubound_GBP_monthly <- GBP_monthly_pred$pred+1.645*GBP_monthly_pred$se
lbound_GBP_monthly<- GBP_monthly_pred$pred-1.645*GBP_monthly_pred$se
ymin <- min(lbound_GBP_monthly)
ymax <- max(ubound_GBP_monthly)
plot(monthly$Date[(n-2):n], month_GBP_ts[(n-2):n], type = "l", ylim=c(ymin,ymax), xlab = "time", ylab =
points(monthly$Date[(nfit+1):n], GBP_monthly_pred$pred, col = "red")
lines(monthly$Date[(nfit+1):n],ubound_GBP_monthly, col = "blue")
lines(monthly$Date[(nfit+1):n],lbound_GBP_monthly, col = "blue")
legend('topleft', legend=c("2 months Ahead ","Upper-Lower bound"),lty = 2, col=c("red","blue"))
```



3c. Calculate Mean Absolute Percentage Error (MAPE) and Precision Measure (PM). How many observations are within the prediction bands? Compare the accuracy of the predictions for the two time series using these two measures.

```
forcast_month_EUR = EUR_monthly_pred$pred
real_EUR_monthly = month_EUR_ts[(nfit+1):n]
#Mean Absolute Percentage Measure (MAPE)
mean(abs(forcast_month_EUR-real_EUR_monthly)/real_EUR_monthly)
```

```
## [1] 0.02282993
```

```
#Precision Measure (PM).
sum((forcast_month_EUR-real_EUR_monthly)^2)/sum((real_EUR_monthly-mean(real_EUR_monthly))^2)
```

```
## [1] 3.518953
```

```
length(real_EUR_monthly)-sum(real_EUR_monthly<lbound_EUR_monthly) - sum(real_EUR_monthly>ubound_EUR_mon
```

```
## [1] 2
```

```
forcast_month_GBP = GBP_monthly_pred$pred
real_GBP_monthly = month_GBP_ts[(nfit+1):n]
#Mean Absolute Percentage Measure (MAPE)
mean(abs(forcast_month_GBP-real_GBP_monthly)/real_GBP_monthly)
```

```
## [1] 0.01806419
```

```
#Precision Measure (PM).
sum((forcast_month_GBP-real_GBP_monthly)^2)/sum((real_GBP_monthly-mean(real_GBP_monthly))^2)
```

```
## [1] 5.191027
```

```
length(real_GBP_monthly)-sum(real_GBP_monthly<lbound_GBP_monthly) - sum(real_GBP_monthly>ubound_GBP_mon
```

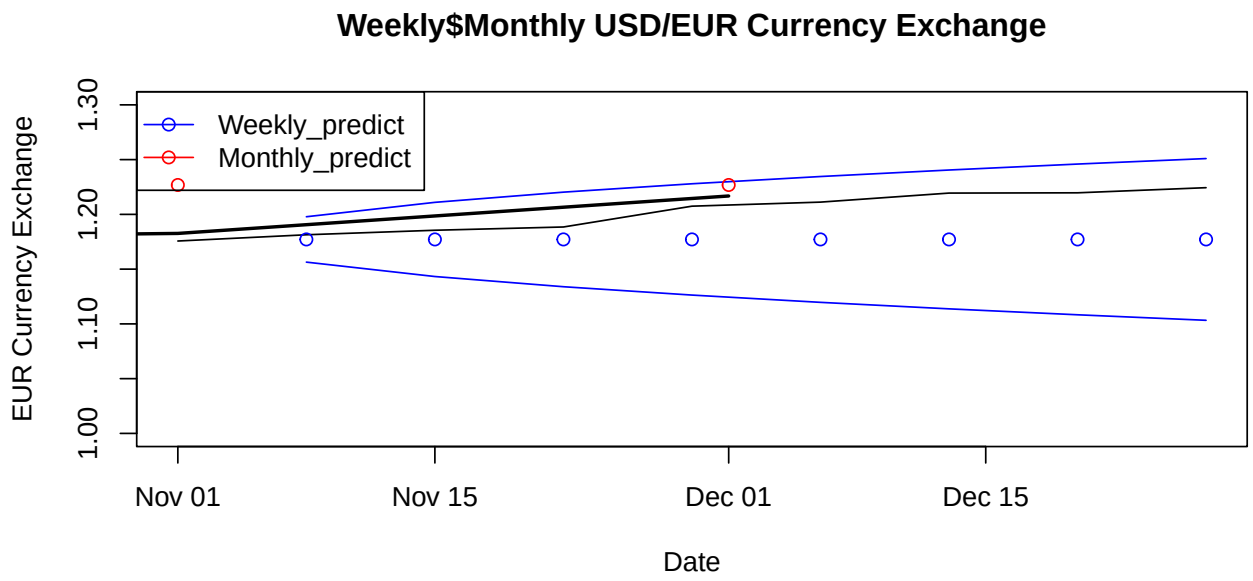
```
## [1] 2
```

Response: Predictions Two observations are within the prediction band. The monthly exchange rate data of USD/EUR and USD/GBP both show small MAPE value less than 3% which shows good fit; however the GBP has PM value over 5 and EUR has PM value of over 3; indicating none of the models are good forecasting models that can grant high accuracies.

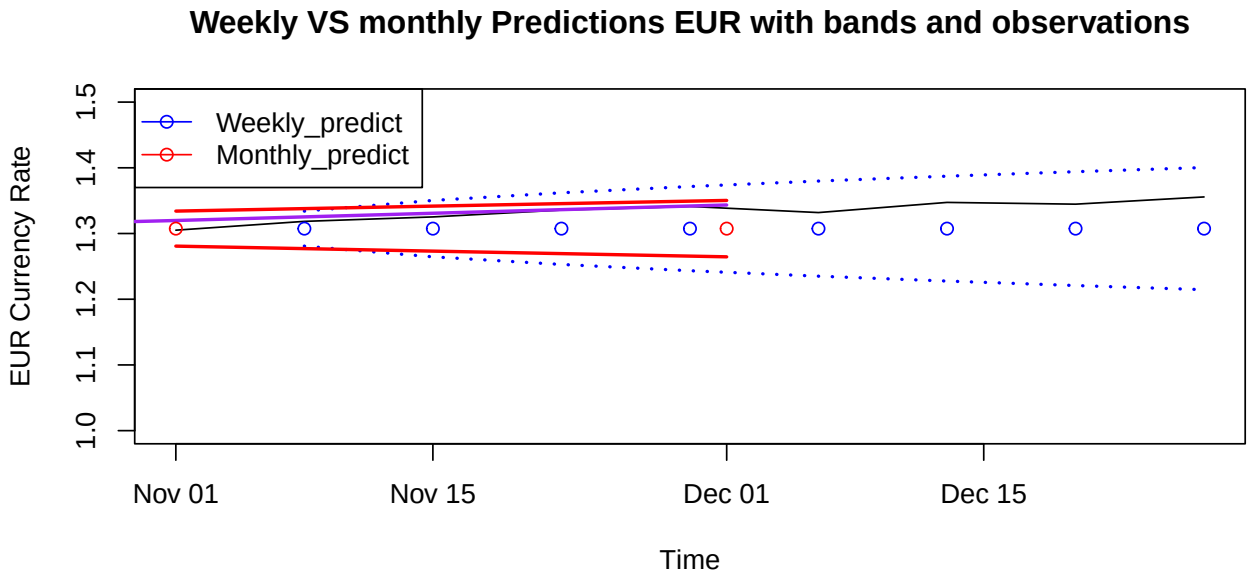
#Question 4. Weekly vs Monthly Forecasting (5 points)

Compare the forecasts based on the weekly versus monthly data. Overlay the forecast into one single plot for each of the two currency exchange rates. What can you say about using weekly versus monthly data?

```
nweek=1148 #length(weekly)
nfitweek=nweek-8 #1140
nmonth=264 #length(monthly)
nfitmonth=nmonth-2 #262

#Overlaying weekly&monthly USD/EUR currency exchange forecasting model
plot(weekly$Date[1140:1148], week_EUR_ts[1140:1148],xlab="Date",ylab="EUR Currency Exchange", ylim=c(1,
points(weekly$Date[1141:1148], EUR_weekly_pred$pred, col = "blue")
lines(weekly$Date[1141:1148],ubound_EUR_weekly, col = "blue")
lines(weekly$Date[1141:1148],lbound_EUR_weekly, col = "blue")

lines(monthly$Date[262:264],month_EUR_ts[262:264],lty=1,lwd= 2)
points(monthly$Date[263:264],EUR_monthly_pred$pred,col="red")
lines(monthly$Date[263:264],ubound_EUR_monthly,lty=1,lwd= 2, col="red")
lines(monthly$Date[263:264],lbound_EUR_monthly,lty=1,lwd= 2, col="red")
legend('topleft', legend=c("Weekly_predict ","Monthly_predict"),lty = 1, pch=c(1,1),col=c("blue","red"))
```




```
plot(weekly$Date[1140:1148],week_GBP_ts[1140:1148],ylim=c(1,1.5),type="l", xlab="Time", ylab = 'EUR Curr
points(weekly$Date[1141:1148],GBP_weekly_pred$pred,col="blue")
lines(weekly$Date[1141:1148],ubound_GBP_weekly,lty=3,lwd= 2, col="blue")
lines(weekly$Date[1141:1148],lbound_GBP_weekly,lty=3,lwd= 2, col="blue")

lines(monthly$Date[262:264],month_GBP_ts[262:264],lty=1,lwd= 2,col="purple")
points(monthly$Date[263:264],GBP_monthly_pred$pred,col="red")
lines(monthly$Date[263:264],ubound_GBP_monthly,lty=1,lwd= 2, col="red")
lines(monthly$Date[263:264],lbound_GBP_monthly,lty=1,lwd= 2, col="red")
legend('topleft', legend=c("Weekly_predict ","Monthly_predict"),lty = 1, pch=c(1,1),col=c("blue","red"))
```



Response: Prediction Comparison Both model captures the prediction within the range; but the width between the upper-line and lower-line are tend to be wider as time goes by. The monthly prediction has a relatively narrower width therefore is better at predicting.

#Question 5. Reflection on ARIMA (5 points)

Considering your understanding of the ARIMA model in general as well as what your understanding of the behavior of the currency exchange data based on the completion of the above questions, how would you personally regard the effectiveness of ARIMA modelling? Where would it be appropriate to use it for forecasting and where would you recommend against? What are some specific points of caution one would need to consider when considering using it?

Response: Take Home Points Besides supply and demand, there are other factors affecting the currency exchange rate, such as the political events, global and domestic economy, natural disasters and so on. With that being said, the exchange rates are most likely following the random walk model–non-stationary, and deterministically disordered. The ARIMA model could have been effective if we could find a good data transformation and good order of selection; yet it will be overconfident in predicting currency rates merely based on time-series. When the currency-exchange rate is stable over years or has certain trend or is stationary, such as swiss franc, ARIMA model is mostly likely to be useful; yet when the currency exchange rate tend to be voliatble, such as USD-Cryptocurrencies, we wouldn't recommond using ARIMA model since the data itself has no constant mean, variance or stationarity. It's price varies significantly so there's no meaning to predict with a wider range. When considering using ARIMA to predict currency exchanges, we should observe the data and try to find the order of selection with the lowest AIC as well as checking other accurancy indicators such as PM to make sure the model fits well.