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 USD/JPY and USD/CNY from January 1999 through December 31st 2020. File *MonthlyCurrencyData.csv* contains the *monthly* exchange rate of USD/JPY and USD/CNY 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)
```

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("DailyCurrencyData.csv", head = TRUE)
monthly <- read.csv("MonthlyCurrencyData.csv", head = TRUE)

daily$Date <- as.Date(daily$Date, "%Y-%m-%d")
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)

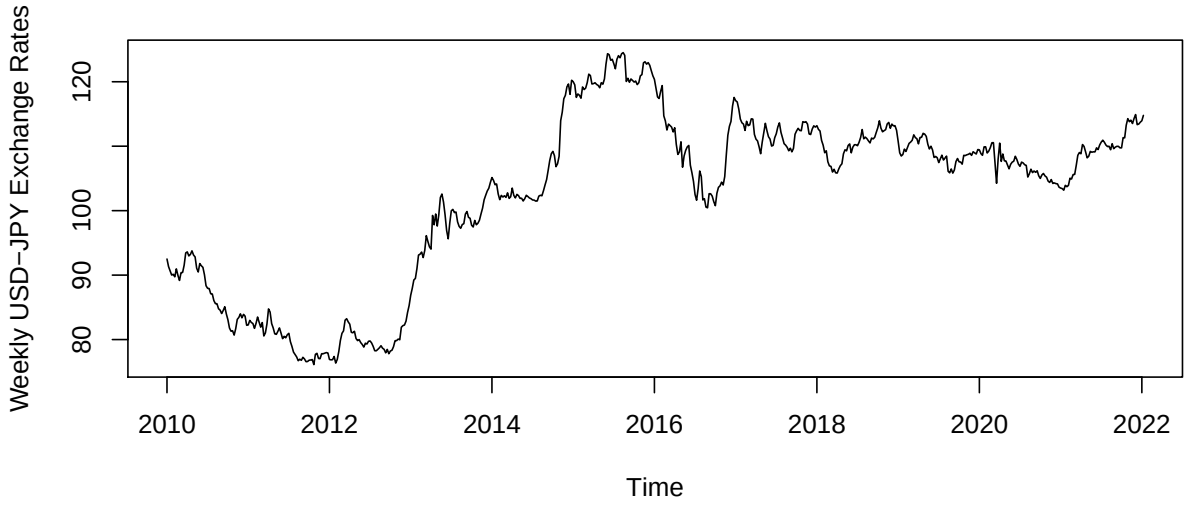
1a. Based on your intuition, when would you use weekly instead of monthly time series data?

Response

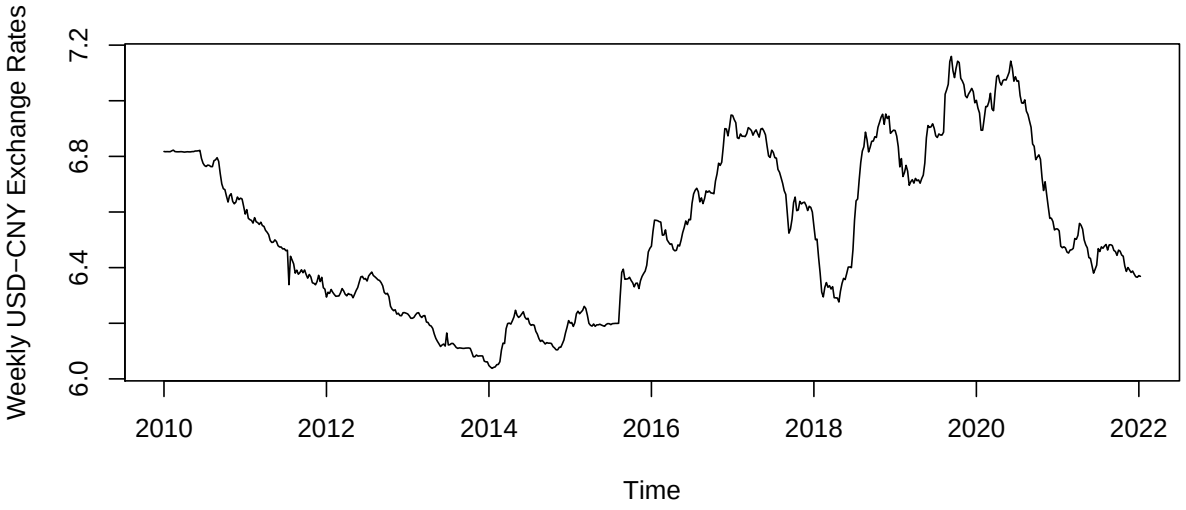
My intuition is that you would use weekly time series data when you are interested in a more granular look at the data. For example, if conducting a time series analysis over a shorter period of time, such as one year, you might be interested in using weekly data because it would give you 52 data points, rather than monthly data, which would only give you 12 data points. On the other hand, if you are conducting time series analysis over a longer period of time, such as 10 years, you might use monthly data, because weekly data would be too granular and not have as much meeting. It also depends on the specific data being used and the intended analysis/outcomes. Monthly data is probably more likely to be normally distributed since data is bucketed over a wider range of data points.

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?

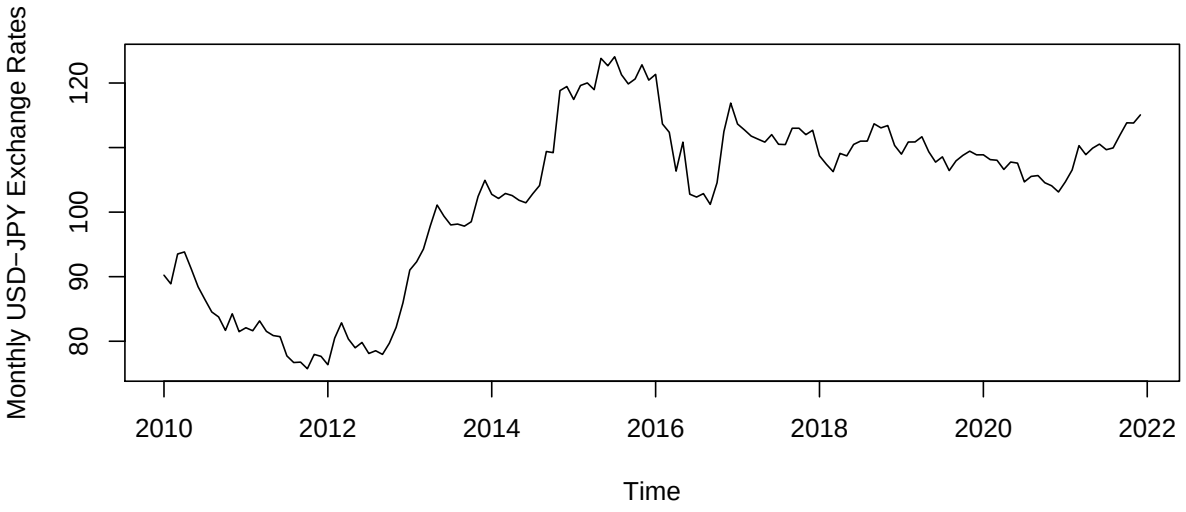
```r
daily <- na.locf(daily)

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

jpy.weekly.ts <- ts(weekly$JPY, start = 2010, freq = 52)
cny.weekly.ts <- ts(weekly$CNY, start = 2010, freq = 52)

jpy.monthly.ts <- ts(monthly$JPY, start = 2010, freq = 12)
cny.monthly.ts <- ts(monthly$CNY, start = 2010, freq = 12)

ts.plot(jpy.weekly.ts, ylab="Weekly USD-JPY Exchange Rates")
```



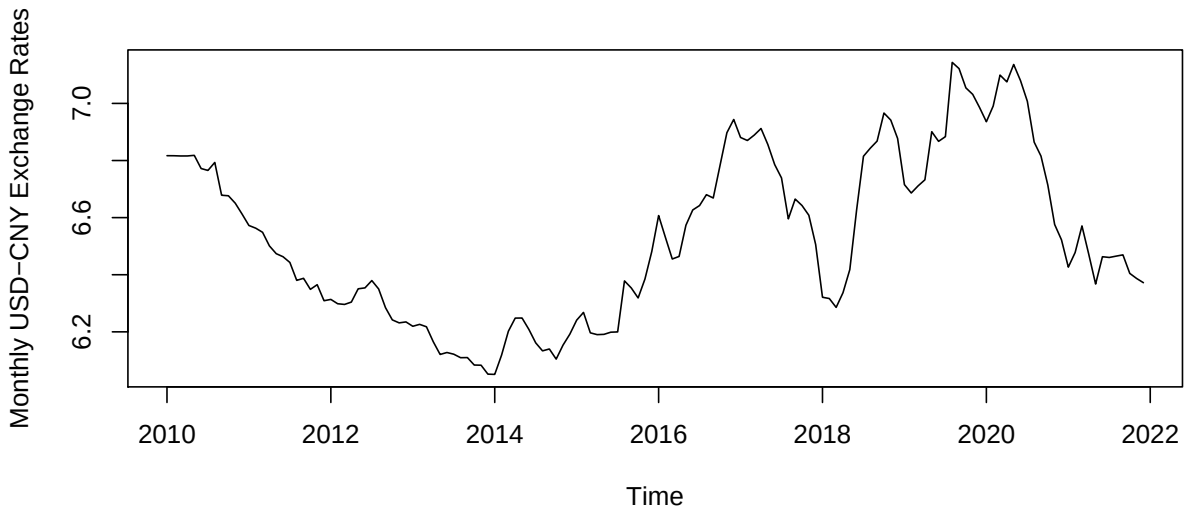

```r
ts.plot(cny.weekly.ts,ylab="Weekly USD-CNY Exchange Rates")
```



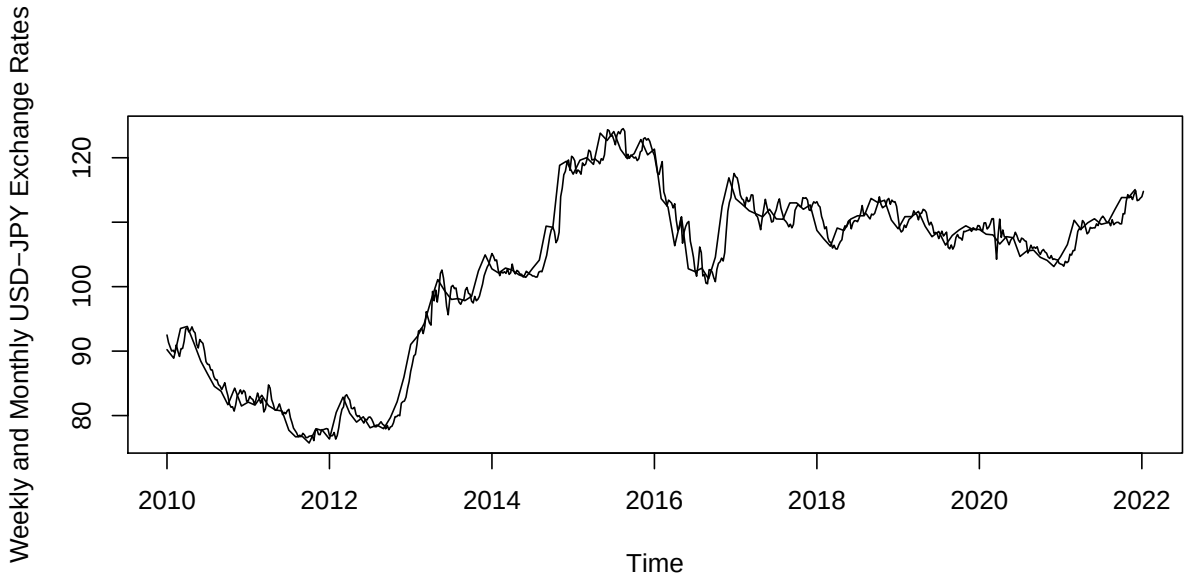

```
ts.plot(jpy.monthly.ts, ylab="Monthly USD-JPY Exchange Rates")
```



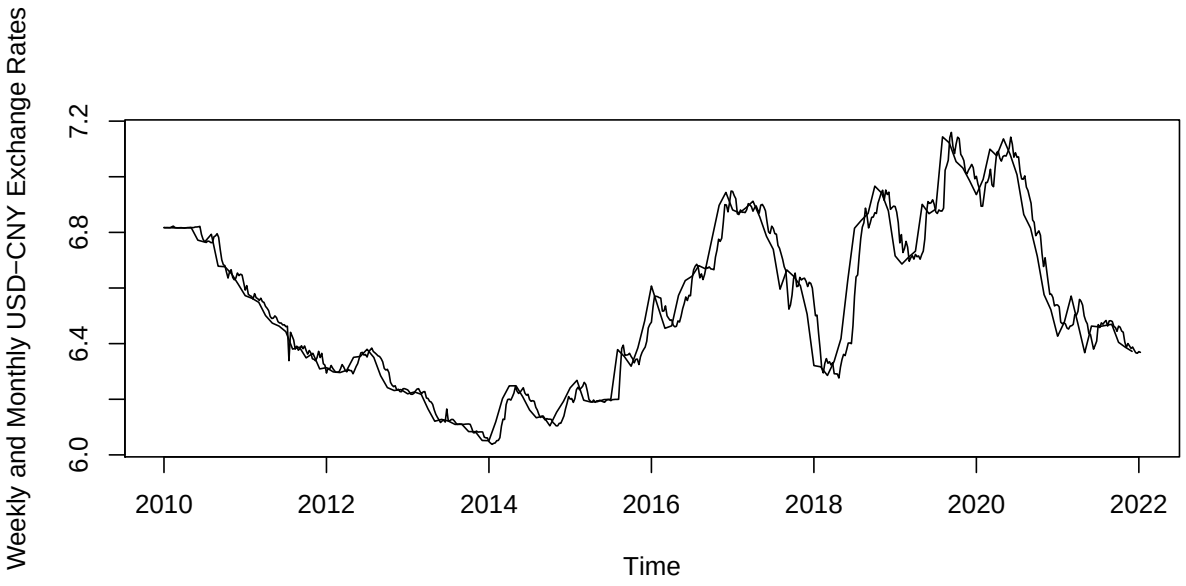

```
ts.plot(cny.monthly.ts,ylab="Monthly USD-CNY Exchange Rates")
```




```
ts.plot(jpy.weekly.ts, ylab="Weekly and Monthly USD-JPY Exchange Rates")
lines(jpy.monthly.ts)
```




```
ts.plot(cny.weekly.ts,ylab="Weekly and Monthly USD-CNY Exchange Rates")
lines(cny.monthly.ts)
```



Response: Weekly vs Montly Time Series data comparison

All plots can be seen above. It was not clear to me if the instructions wanted us to overlay the weekly and monthly plots or separate them, so I did both.

The main difference between the weekly and monthly plot is that the monthly plot is a bit smoother, with fewer data points. The weekly plot has more granular spikes in the data and is less smooth. The trends in the data are essentially the same between the weekly and monthly data, since it is the same underlying data just aggregated at different intervals of time.

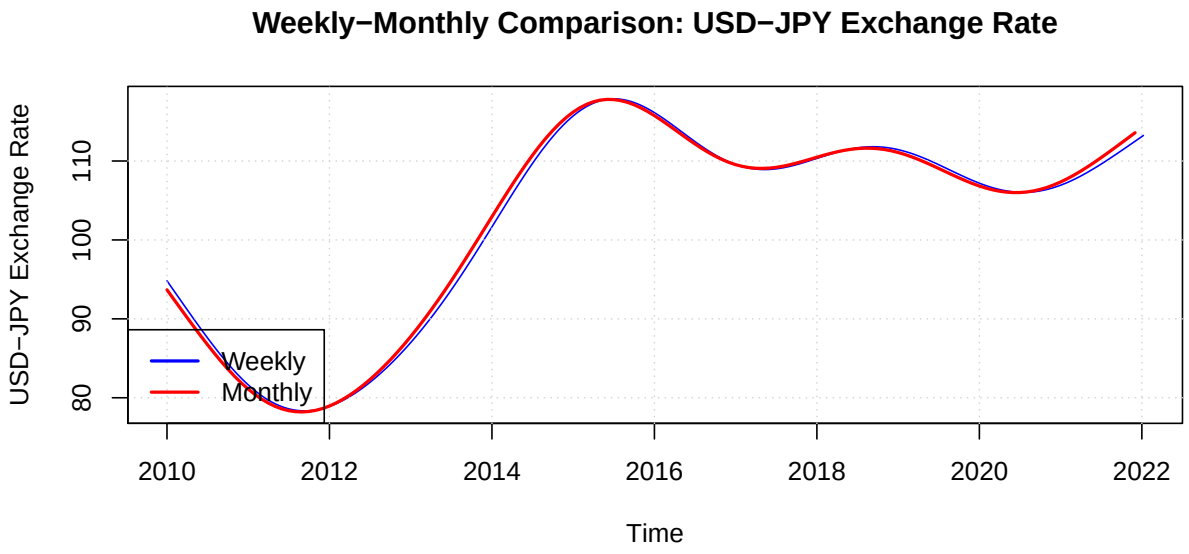
As for the comparison of the time series between the two different currencies (CNY-Chinese Yuan and JPY-Japanese Yen), the first that thing stands out is that the order of magnitude is completely different. The USD-CNY time series fluctutes between 6 and 7 CNY per US dollar. The USD-JPY time series fluctuates between 80 and 120 JPY per US dollar. The trends in the exchange rates are also completely different. The USD-JPY time series shows a moderate decrease, followed by a rapid and substantial increase, then stays on a somewhat similar level from 2016-2022. On the other hand, the USD-CPY time series shows a gradual but substantial decrease, followed by a gradual but substantial increase, before running to similar levels as the starting point in 2010, even a bit lower in 2021 than 2010.

The USD-JPY plot has a definite increasing trend over the overall time period, wehereas the USD-CNY plot has less of a trend pattern.

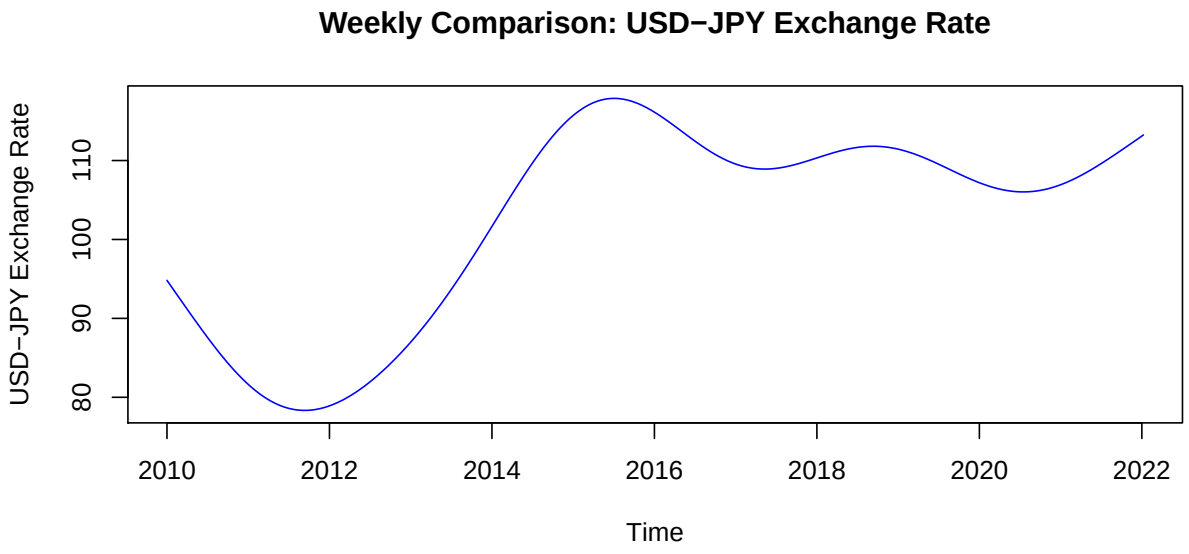
1c. Fit a non-parametric trend using splines regression to both the weekly and monthly time series data for both currencies. Overlay the fitted trends for each of the currency separately. How do the trends compare when comparing those fitted using the weekly and monthly data? How do the trends for the two currencies compare?

```
#USD-JPY

#Create Equally Spaced Time Points--Weekly
time.pts.weekly = c(1:length(jpy.weekly.ts))
time.pts.weekly = c(time.pts.weekly - min(time.pts.weekly))/max(time.pts.weekly)

#Splines Trend Estimation Weekly
gam.fit.weekly = gam(jpy.weekly.ts~s(time.pts.weekly))
jpy.fit.weekly.gam = ts(fitted(gam.fit.weekly), start = 2010, frequency = 52)
```

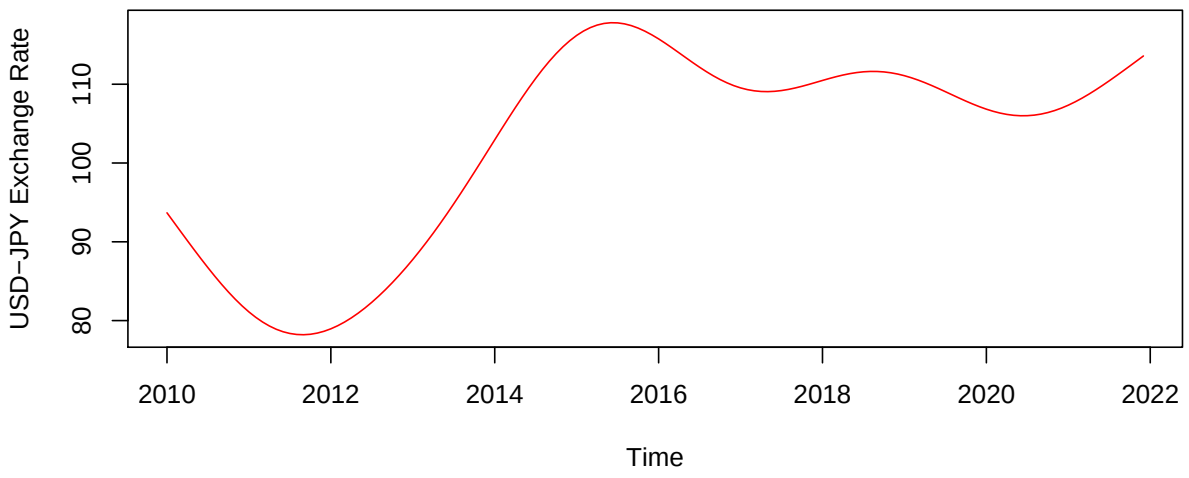
```r
#Create Equally Spaced Time Points--Monthly
time.pts.monthly = c(1:length(jpy.monthly.ts))
time.pts.monthly = c(time.pts.monthly - min(time.pts.monthly))/max(time.pts.monthly)

#Splines Trend Estimation Monhtly
gam.fit.monthly = gam(jpy.monthly.ts~s(time.pts.monthly))
jpy.fit.monthly.gam = ts(fitted(gam.fit.monthly), start = 2010, frequency = 12)

#Plot
ts.plot(jpy.fit.weekly.gam, xlab = "Time", ylab = "USD-JPY Exchange Rate", main = "Weekly-Monthly Compa
grid()
lines(jpy.fit.monthly.gam, lwd = 2, col = "red")
legend("bottomleft", legend = c("Weekly", "Monthly"),
col = c("blue", "red"), lwd = 2)
```




```r
ts.plot(jpy.fit.weekly.gam, xlab = "Time", ylab = "USD-JPY Exchange Rate", main = "Weekly Comparison: US
```

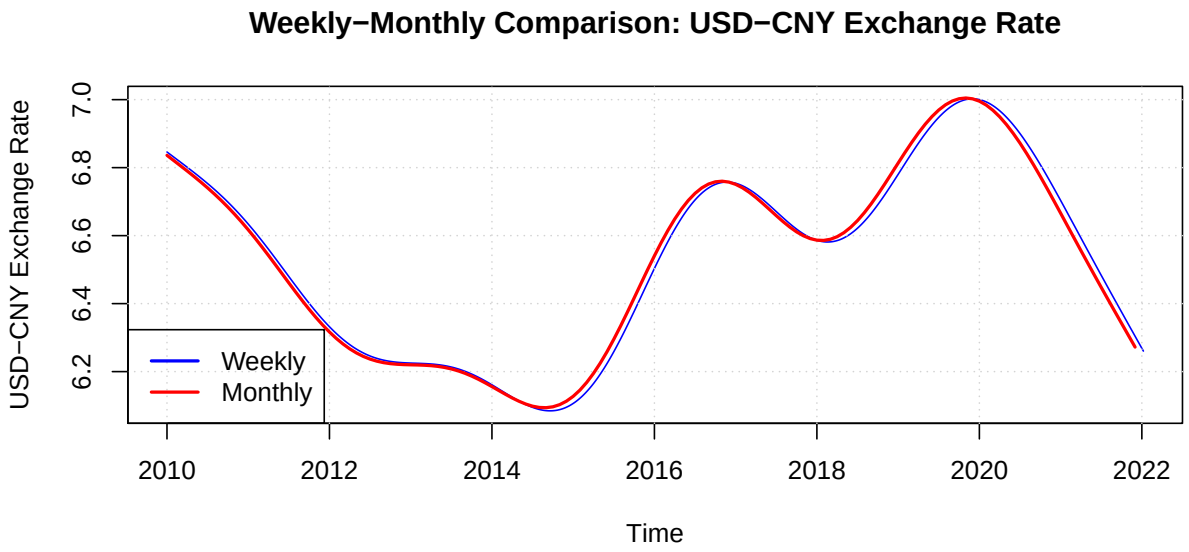



```
ts.plot(jpy.fit.monthly.gam, xlab = "Time", ylab = "USD-JPY Exchange Rate", main = "Monthly Comparison:
```

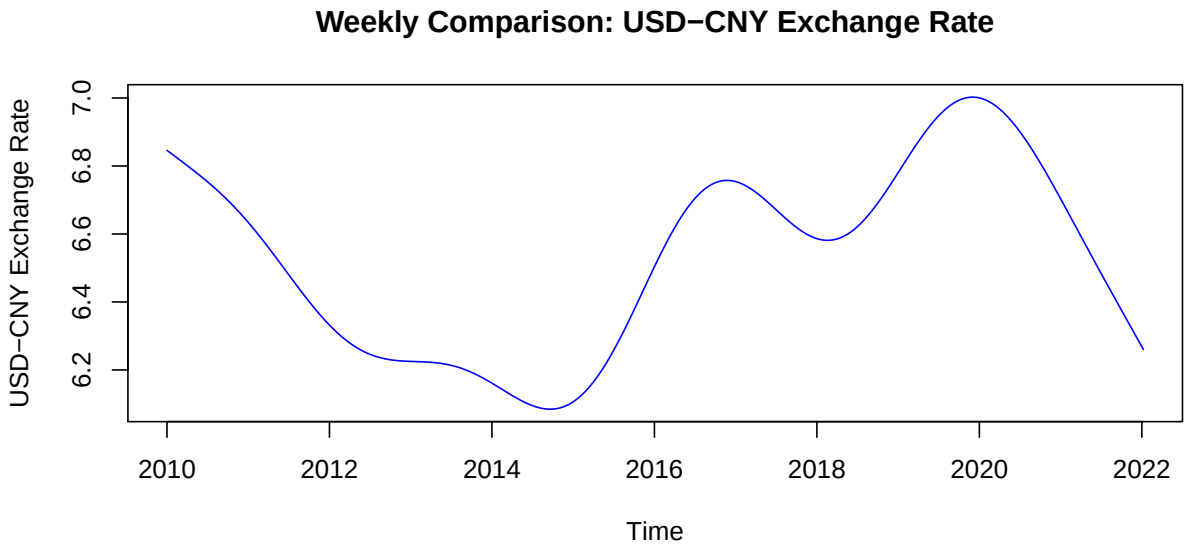



```
#USD-CNY

#Splines Trend Estimation Weekly
gam.fit.weekly.cny = gam(cny.weekly.ts~s(time.pts.weekly))
cny.fit.weekly.gam = ts(fitted(gam.fit.weekly.cny), start = 2010, frequency = 52)

#Splines Trend Estimation Monhtly
gam.fit.monthly.cny = gam(cny.monthly.ts~s(time.pts.monthly))
cny.fit.monthly.gam = ts(fitted(gam.fit.monthly.cny), start = 2010, frequency = 12)
```

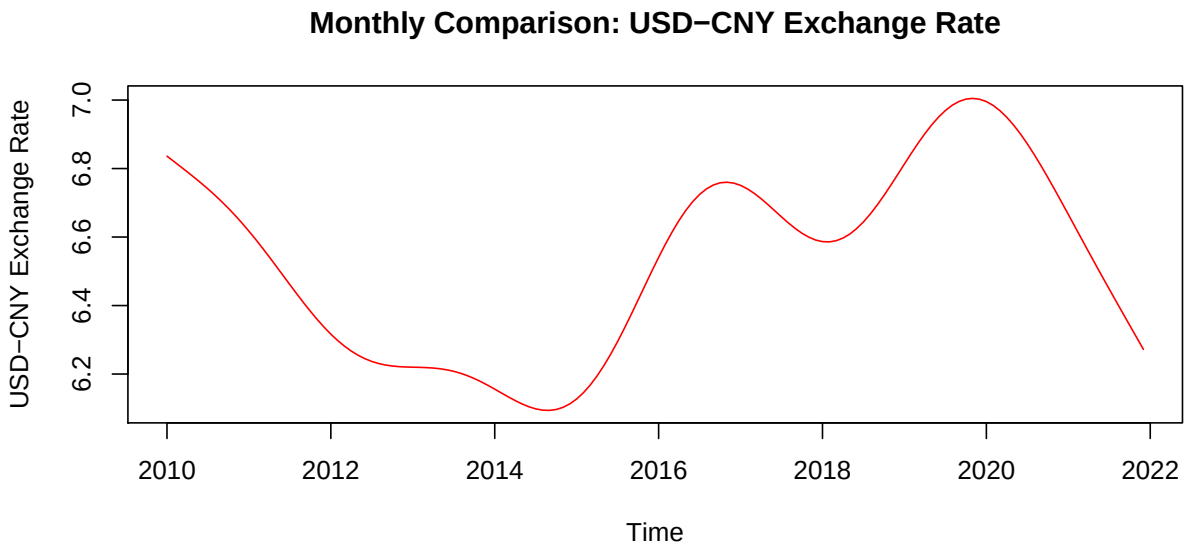
```r
#Plot
ts.plot(cny.fit.weekly.gam, xlab = "Time", ylab = "USD-CNY Exchange Rate", main = "Weekly-Monthly Compa
grid()
lines(cny.fit.monthly.gam, lwd = 2, col = "red")
legend("bottomleft", legend = c("Weekly", "Monthly"),
col = c("blue", "red"), lwd = 2)
```




```r
ts.plot(cny.fit.weekly.gam, xlab = "Time", ylab = "USD-CNY Exchange Rate", main = "Weekly Comparison: US
```




```r
ts.plot(cny.fit.monthly.gam, xlab = "Time", ylab = "USD-CNY Exchange Rate", main = "Monthly Comparison:
```



Response: Comparing Trend Estimation using weekly vs Monthly Data

I overlayed a plot showing weekly versus monthly for each currency exchange rate. I also plotted weekly and monthly separately for each currency exchange rate.

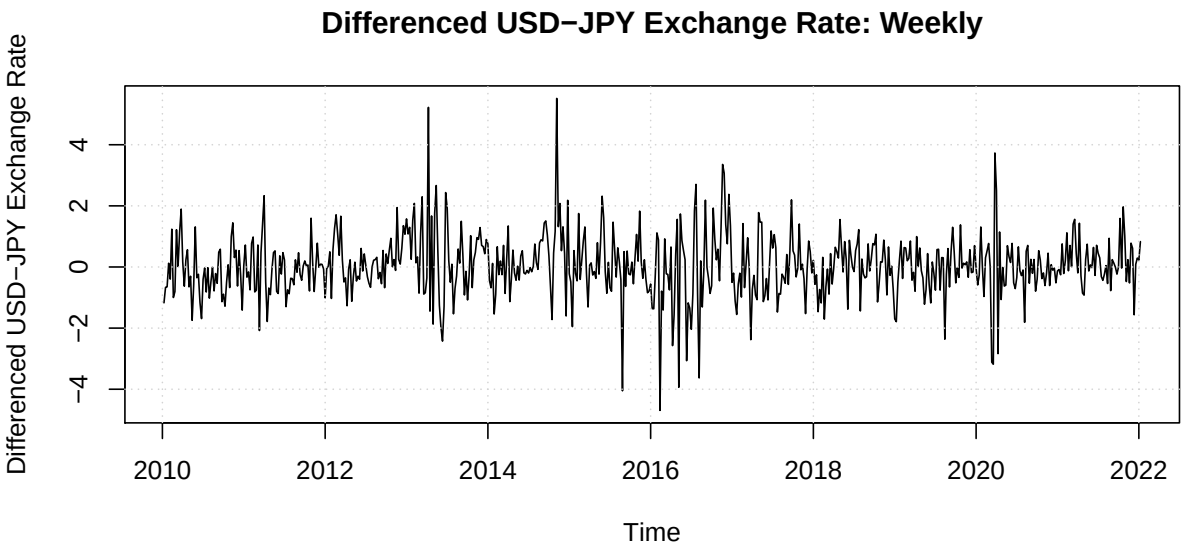
For the USD-JPY exchange rate, the trend estimation from splines regression is almost identical whether using weekly or monthly data. In the plot, you can see just how close these trend lines are. Both plots are much smoother than the time series of the raw data. The fitted trend decreases before increasing, then does not fluctuate too much from 2015 to 2022.

Again for the USD-CNY exchange rate, the trend estimation from splines regression is almost identical whether using weekly or monthly data. In the plot, you can see just how close these trend lines are. Both plots are much smoother than the time series of the raw data. The fitted trend decreases steeply and gradually before increasing, then sharply decreases again from 2020 to 2022.

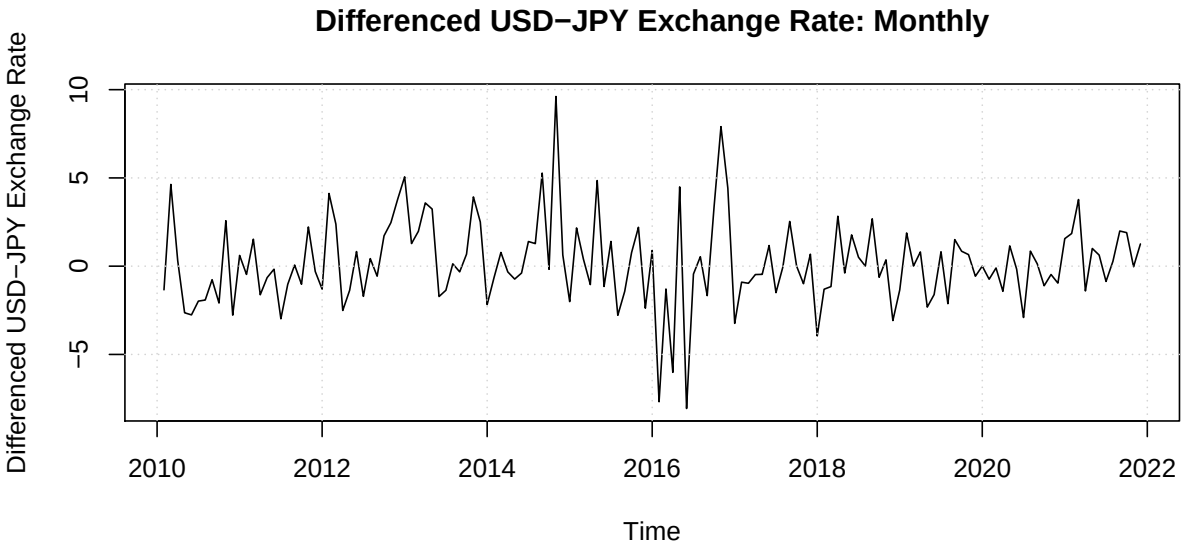
The trends for the USD-JPY exchange rate and the USD-CNY exchange rate are pretty different from each other over time. The USD-JPY exchange rate shows an overall increasing trend over the complete timeframe. The USD-CNY exchange rate is much more up-and-down, showing an initial increasing trend before ending with a noticeable decreasing trend. The ending point is lower than the starting point for the USD-CNY exchange rate.

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?

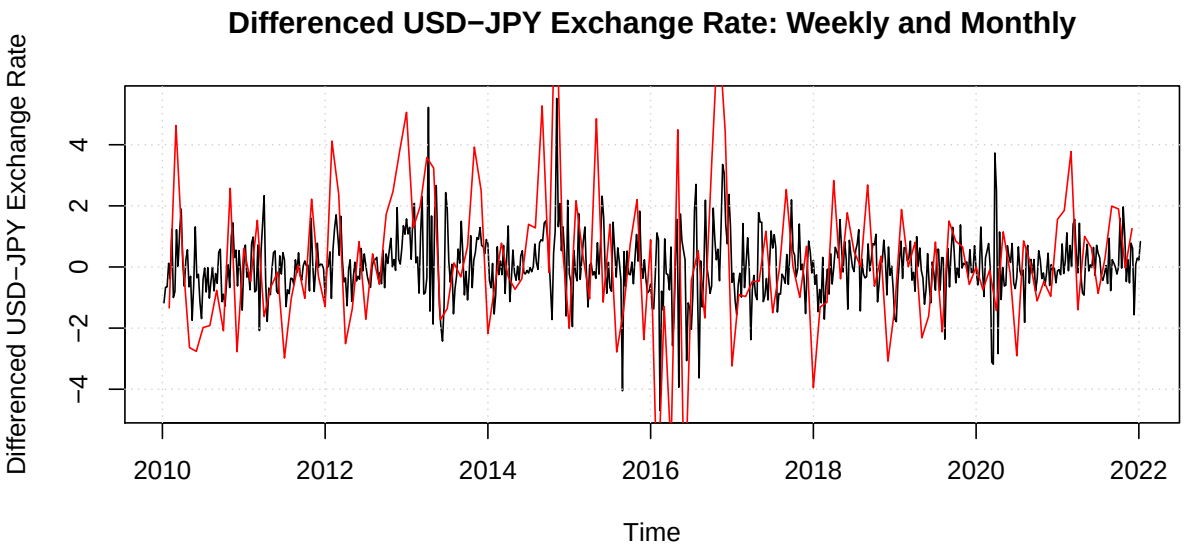
```
ts.plot(diff(jpy.weekly.ts), col = "black", xlab = "Time", ylab = "Differenced USD-JPY Exchange Rate",
main = "Differenced USD-JPY Exchange Rate: Weekly")
grid()
```



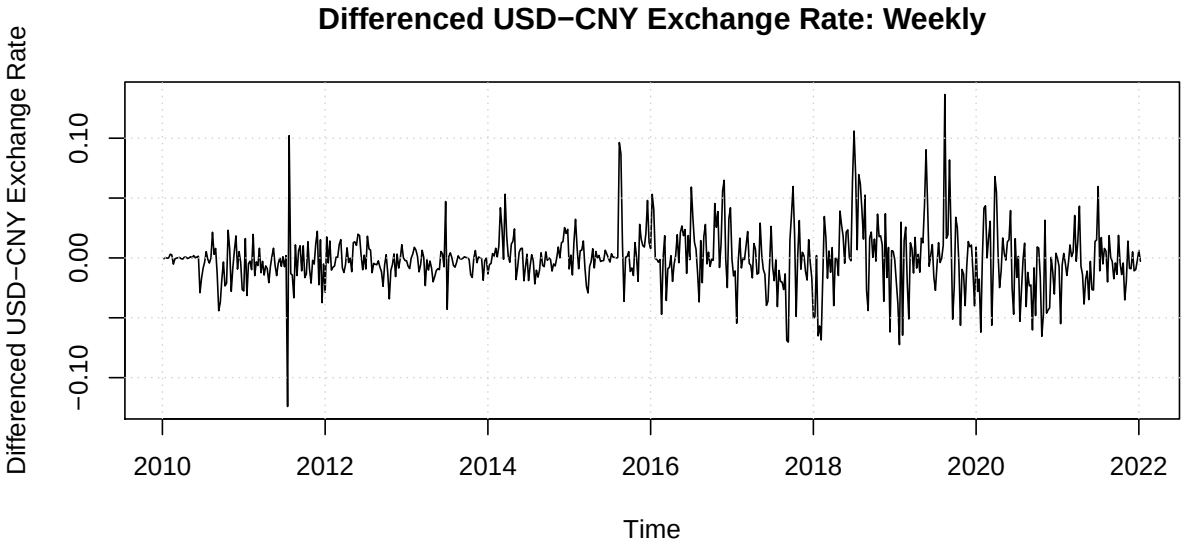

```
ts.plot(diff(jpy.monthly.ts), col = "black", xlab = "Time", ylab = "Differenced USD-JPY Exchange Rate",
main = "Differenced USD-JPY Exchange Rate: Monthly")
grid()
```



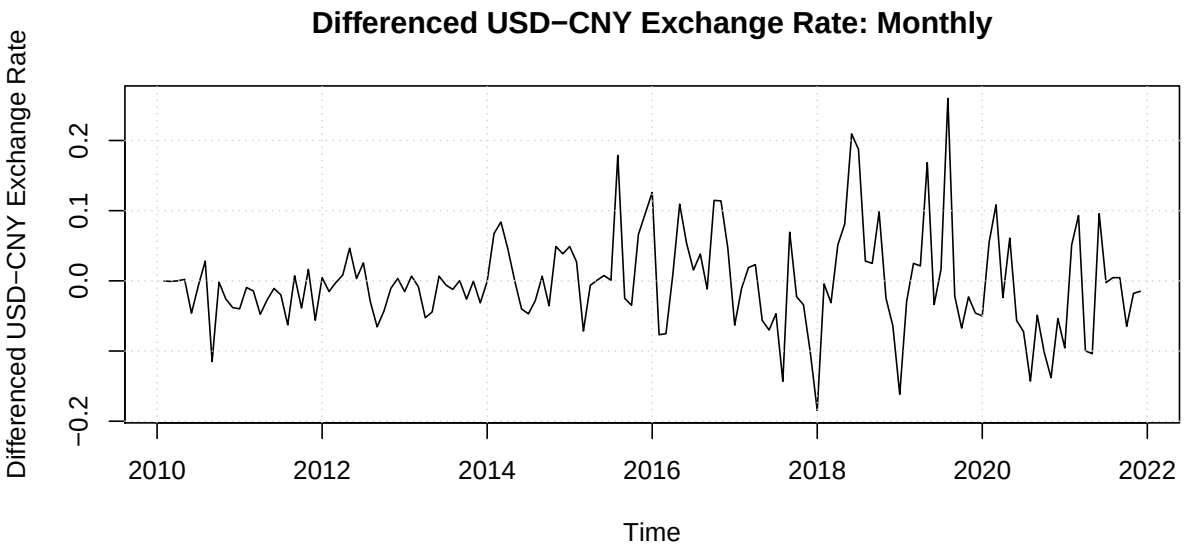

```
ts.plot(diff(jpy.weekly.ts), col = "black", xlab = "Time", ylab = "Differenced USD-JPY Exchange Rate",
main = "Differenced USD-JPY Exchange Rate: Weekly and Monthly")
grid()
lines(diff(jpy.monthly.ts), col = "red")
```



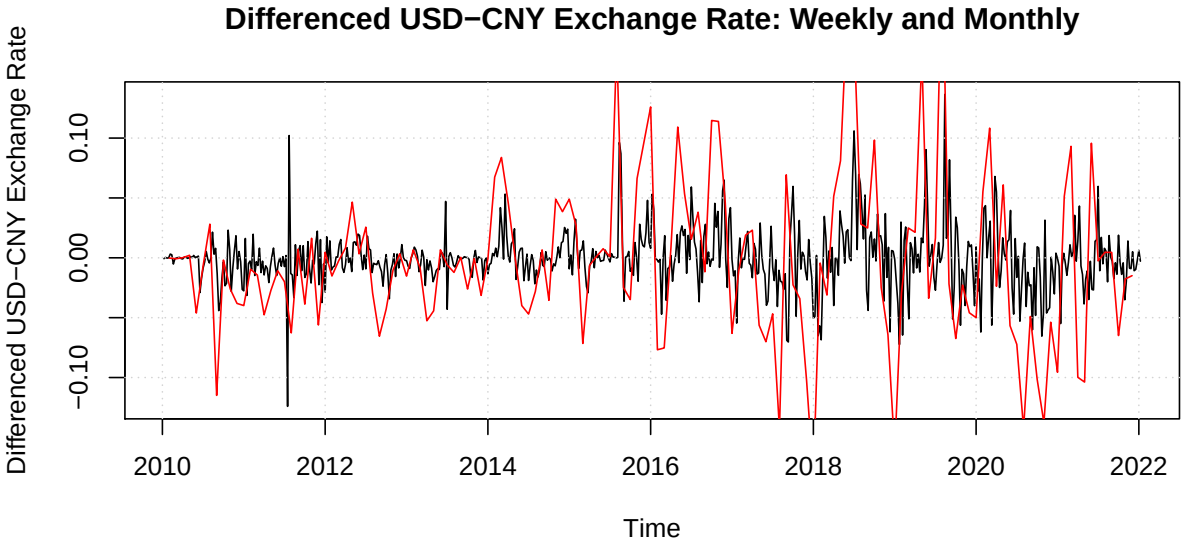

```
ts.plot(diff(cny.weekly.ts), col = "black", xlab = "Time", ylab = "Differenced USD-CNY Exchange Rate",
main = "Differenced USD-CNY Exchange Rate: Weekly")
grid()
```



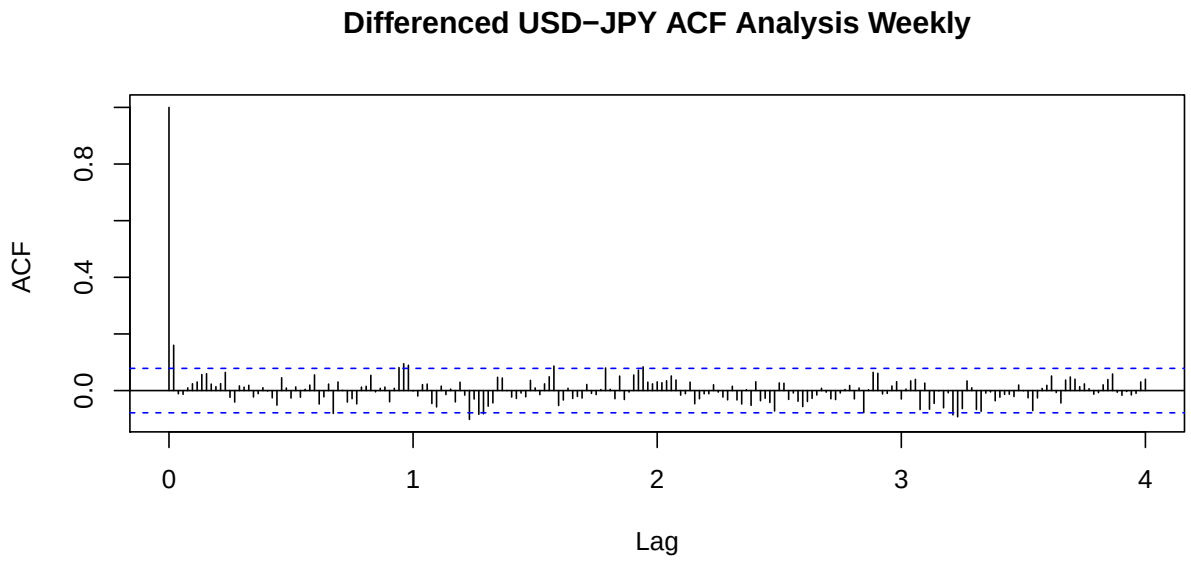

```
ts.plot(diff(cny.monthly.ts), col = "black", xlab = "Time", ylab = "Differenced USD-CNY Exchange Rate",
main = "Differenced USD-CNY Exchange Rate: Monthly")
grid()
```



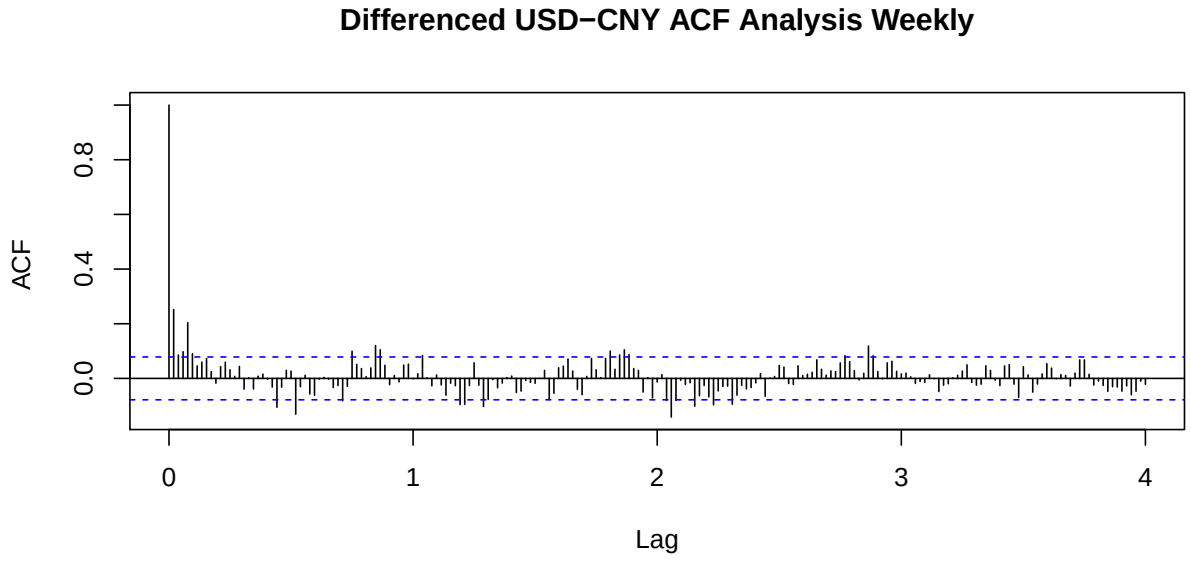

```
ts.plot(diff(cny.weekly.ts), col = "black", xlab = "Time", ylab = "Differenced USD-CNY Exchange Rate",
main = "Differenced USD-CNY Exchange Rate: Weekly and Monthly")
grid()
lines(diff(cny.monthly.ts), col = "red")
```



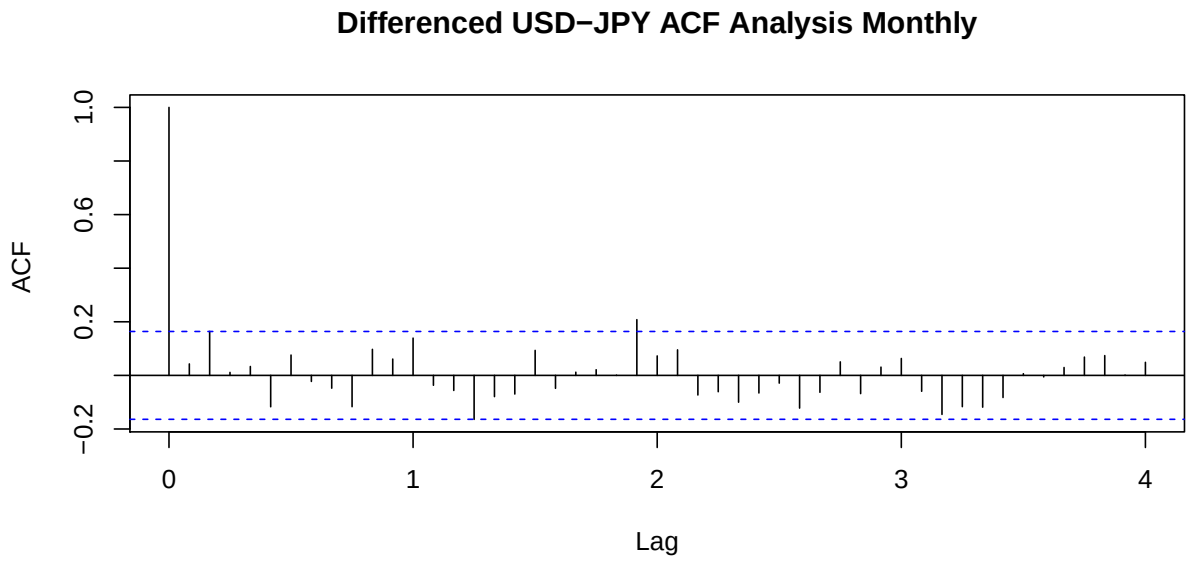

```
#ACF
acf(diff(jpy.weekly.ts), lag.max = 52 * 4, xlab = "Lag", ylab = "ACF ", main = "Differenced USD-JPY ACF
```



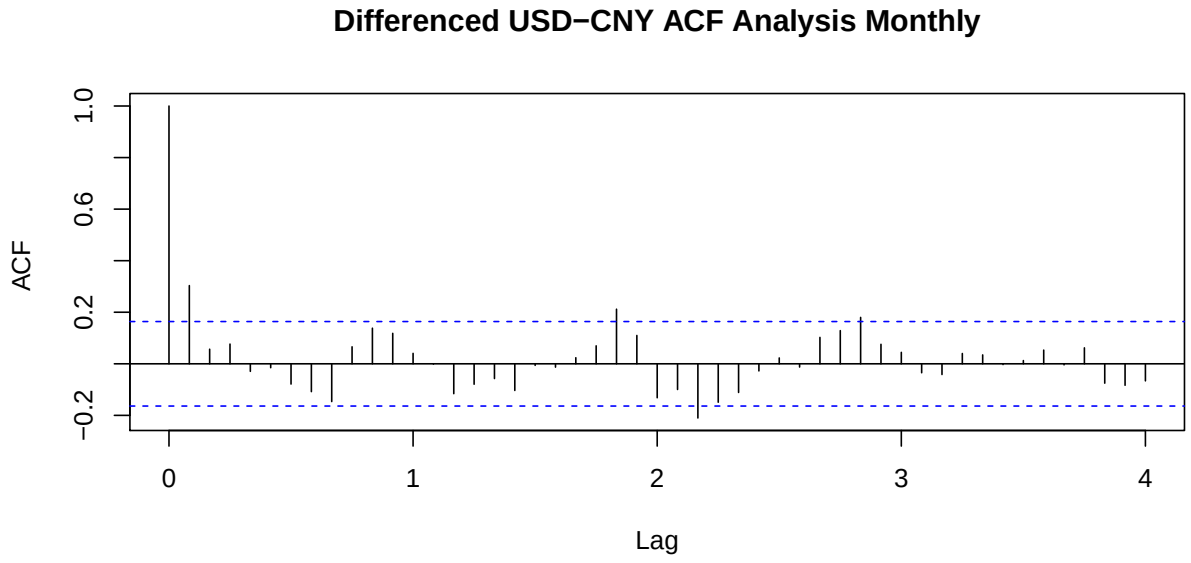

```
#ACF
acf(diff(cny.weekly.ts), lag.max = 52 * 4, xlab = "Lag", ylab = "ACF ", main = "Differenced USD-CNY ACF
```




```
#ACF
acf(diff(jpy.monthly.ts), lag.max = 12 * 4, xlab = "Lag", ylab = "ACF ", main = "Differenced USD-JPY ACF
```




```
#ACF
acf(diff(cny.monthly.ts), lag.max = 12 * 4, xlab = "Lag", ylab = "ACF ", main = "Differenced USD-CNY ACF
```



Response: Exploratory Analysis of 1st Order Difference Data

It appears based on the ACF plots for weekly and monthly data, that both datasets are weakly stationary. This applies for both the USD-JPY and USD-CNY exchange rates. There does not appear to be much of a difference with regards to stationarity between the weekly and monthly data. Both datasets have large autocorrelation values at lag 0, then have ACF values mostly within the confidence bands for the rest of the data. There are, however, a few data points outside the confidence bands. As such, I would claim that the data are weakly stationary.

When comparing the differenced data for the exchange rates of USD-JPY and USD-CNY, the exchange rates do not appear to differ much with respect to stationarity and serial dependence. Both exchange rates

again appear weakly stationary. The main difference is that the first order differences are much larger for the USD-JPY exchange rate than USD-CNY.

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 2021) 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

```
jpy.weekly.train <- jpy.weekly.ts[1:618]
jpy.weekly.test <- jpy.weekly.ts[619:626]

cny.weekly.train <- cny.weekly.ts[1:618]
cny.weekly.test <- cny.weekly.ts[619:626]

#ARIMA-USD-JPY, d = 1
n = length(jpy.weekly.train)
norder = 9
p = c(1:norder)-1; q = c(1:norder)-1
aic1 = matrix(0,norder,norder)
try(
for(i in 1:norder){
   for(j in 1:norder){
    modij1 = arima(jpy.weekly.train,order = c(p[i],1,q[j]), method='ML', optim.control = list(maxit = 1
    aic1[i,j] = modij1$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
   }
})

#ARIMA-USD-JPY, d = 2
#aic2 = matrix(0,norder,norder)
#for(i in 1:norder){
   #for(j in 1:norder){
    #modij2 = arima(jpy.weekly.train,order = c(p[i],2,q[j]), method='ML', optim.control = list(maxit =
    #aic2[i,j] = modij2$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
   #}
#}

#ARIMA-USD-CNY, d = 1
#n = length(cny.weekly.train)
#norder = 9
#p = c(1:norder)-1; q = c(1:norder)-1
#aic3 = matrix(0,norder,norder)
#try(
#for(i in 1:norder){
   #for(j in 1:norder){
    #modij3 = arima(cny.weekly.train,order = c(p[i],1,q[j]), method='ML', optim.control = list(maxit =
```

```
    #aic3[i,j] = modij2$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
   #}
#})

#ARIMA-USD-CNY, d = 2
#n = length(cny.weekly.train)
#norder = 9
#p = c(1:norder)-1; q = c(1:norder)-1
#aic4 = matrix(0,norder,norder)
#try(
#for(i in 1:norder){
   #for(j in 1:norder){
    #modij4 = arima(cny.weekly.train,order = c(p[i],2,q[j]), method='ML', optim.control = list(maxit =
    #aic4[i,j] = modij2$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n/(n-p[i]-q[j]-2)
   #}
#})

aicv1 = as.vector(aic1)
indexp1 = rep(c(1:norder),norder)
indexq1 = rep(c(1:norder),each=norder)
indexaic1 = which(aicv1 == min(aicv1))
porder1 = indexp1[indexaic1]-1
qorder1 = indexq1[indexaic1]-1


final_model1 = arima(jpy.weekly.train, order = c(porder1,1,qorder1), method = "ML")

#min(aicv1)
#aicv2 = as.vector(aic2)
#min(aicv2)

#aicv3 = as.vector(aic3)
#indexp3 = rep(c(1:norder),norder)
#indexq3 = rep(c(1:norder),each=norder)
#indexaic3 = which(aicv3 == min(aicv3))
#porder3 = indexp3[indexaic3]-1
#qorder3 = indexq3[indexaic3]-1

final_model3 = arima(cny.weekly.train, order = c(0,1,0), method = "ML")

#aicv4 = as.vector(aic4)
#indexp4 = rep(c(1:norder),norder)
#indexq4 = rep(c(1:norder),each=norder)
#indexaic4 = which(aicv4 == min(aicv4))
#porder4 = indexp4[indexaic4]-1
#qorder4 = indexq4[indexaic4]-1

#final_model4 = arima(cny.weekly.train, order = c(porder4,2,qorder4), method = "ML")
```

```
summary(final_model1)
```

```
##           Length Class  Mode
## coef        9    -none- numeric
```

```
## sigma2      1    -none- numeric
## var.coef   81    -none- numeric
## mask        9    -none- logical
## loglik      1    -none- numeric
## aic         1    -none- numeric
## arma        7    -none- numeric
## residuals 618    ts     numeric
## call        4    -none- call
## series      1    -none- character
## code        1    -none- numeric
## n.cond      1    -none- numeric
## nobs        1    -none- numeric
## model      10    -none- list
```

```
summary(final_model3)
```

```
##           Length Class  Mode
## coef        0    -none- numeric
## sigma2      1    -none- numeric
## var.coef    0    -none- numeric
## mask        0    -none- logical
## loglik      1    -none- numeric
## aic         1    -none- numeric
## arma        7    -none- numeric
## residuals 618    ts     numeric
## call        4    -none- call
## series      1    -none- character
## code        1    -none- numeric
## n.cond      1    -none- numeric
## nobs        1    -none- numeric
## model      10    -none- list
```

```
#summary(final_model4)

final_model1
```

```
##
## Call:
## arima(x = jpy.weekly.train, order = c(porder1, 1, qorder1), method = "ML")
##
## Coefficients:
##          ar1     ar2     ar3      ar4      ma1      ma2      ma3     ma4
##       0.1758  1.3058  0.2037  -0.9647  -0.0126  -1.3595  -0.4089  0.9533
## s.e.  0.0125  0.0150  0.0117   0.0140   0.0419   0.0176   0.0559  0.0174
##          ma5
##       0.1809
## s.e.  0.0406
##
## sigma^2 estimated as 1.013:  log likelihood = -881.8,  aic = 1783.6
```

```
final_model1$aic
```

```
## [1] 1783.601
```

```
final_model1$coef
```

```
##        ar1        ar2        ar3        ar4        ma1        ma2        ma3
##  0.1758177  1.3057818  0.2036811 -0.9647432 -0.0125564 -1.3594860 -0.4088735
##        ma4        ma5
##  0.9532521  0.1809303
```

```
porder1
```

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

```
qorder1
```

```
## [1] 5
```

```
#Select best model for USD-CNY
final_model3$aic
```

```
## [1] -2804.395
```

```
#final_model4$aic
final_model3
```

```
##
## Call:
## arima(x = cny.weekly.train, order = c(0, 1, 0), method = "ML")
##
##
## sigma^2 estimated as 0.0006197:  log likelihood = 1403.2,  aic = -2804.39
```

```
final_model3$coef
```

```
## numeric(0)
```

Response: Analysis of the ARIMA Fit for the Weekly Data

For the USD-JPY weekly data, the best ARIMA model (identified by AIC value) has parameters of p = 4, d = 1, and q = 5. This model has an AIC of 1783.601. The model summary estimates sigma^2 as 1.013 and log likelihood = -881.8. The 9 coefficients have p-values of 0.000000e+00, 0.000000e+00, 0.000000e+00, 0.000000e+00, 7.644600e-01, 0.000000e+00, 2.589040e-13, 0.000000e+00, 8.532468e-06. This means that the coefficient for MA1 is not statistically significant, whereas all other coefficients are statistically significant. However, overall, I would not recommend a lower order model because only 1 of the 9 coefficients is not statistically significant.

For the USD-CNY weekly data, the best ARIMA model (identified by AIC value) has parameters of p = 0, d = 1, and q = 0. This model has an AIC of -2804.395. The model summary estimates sigma^2 as 0.0006197 and log likelihood as 1403.2. I would conclude that that ARIMA model is not a good fit on the weekly data for USD-CNY, because the AR and MA processes are not included in the final model, with p and q equal to 0. There are no coefficients to report since the orders for p and q are equal to 0. This means there is no statistical significance of coefficients to consider. I would suggest just using the differenced data for the USD-CNY weekly exchange rates, rather than an ARIMA model. Lower order models would fit the data better.

```
## p-value function for the z-test taking as input the test statistic
pvalue.coef <- function(tv) {
  2 * (1 - pnorm(abs(tv)))
}

## Sample Code to compute the test statistics
 tv.jpy.weekly <-as.numeric(final_model1$coef)/as.numeric(sqrt(diag(final_model1$var.coef)))
 tv.cny.weekly <-as.numeric(final_model3$coef)/as.numeric(sqrt(diag(final_model3$var.coef)))

## Apply the pvalue.coef function
 pvalues.jpy.weekly <- sapply(tv.jpy.weekly, pvalue.coef)
 pvalues.cny.weekly <- sapply(tv.cny.weekly, pvalue.coef)

 pvalues.jpy.weekly
```

```
## [1] 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 7.644600e-01
## [6] 0.000000e+00 2.589040e-13 0.000000e+00 8.532468e-06
```

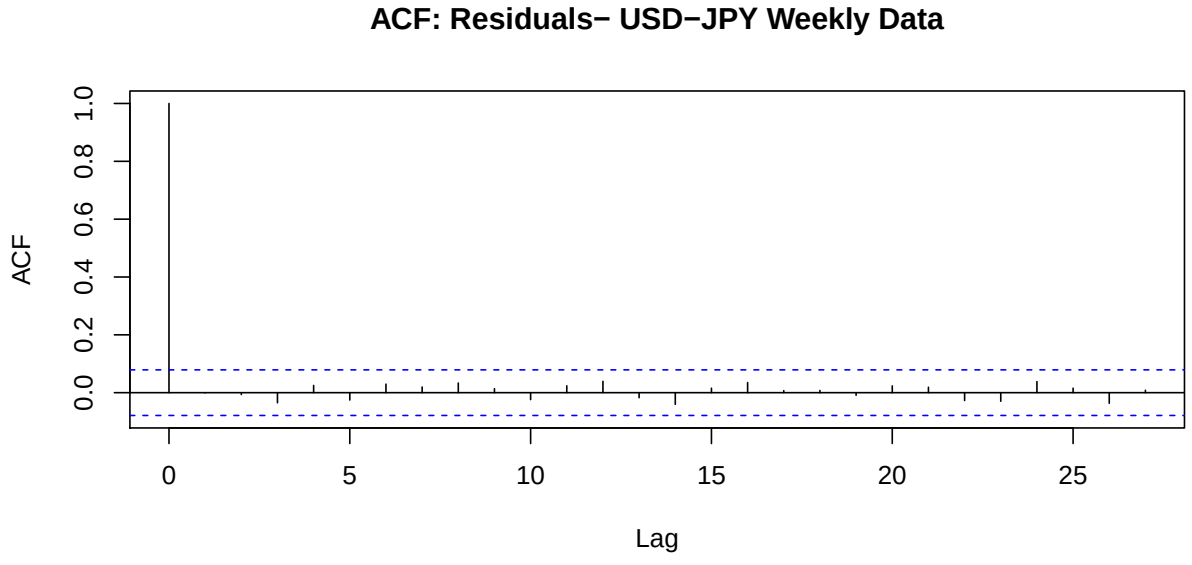
```
 pvalues.cny.weekly
```

```
## list()
```

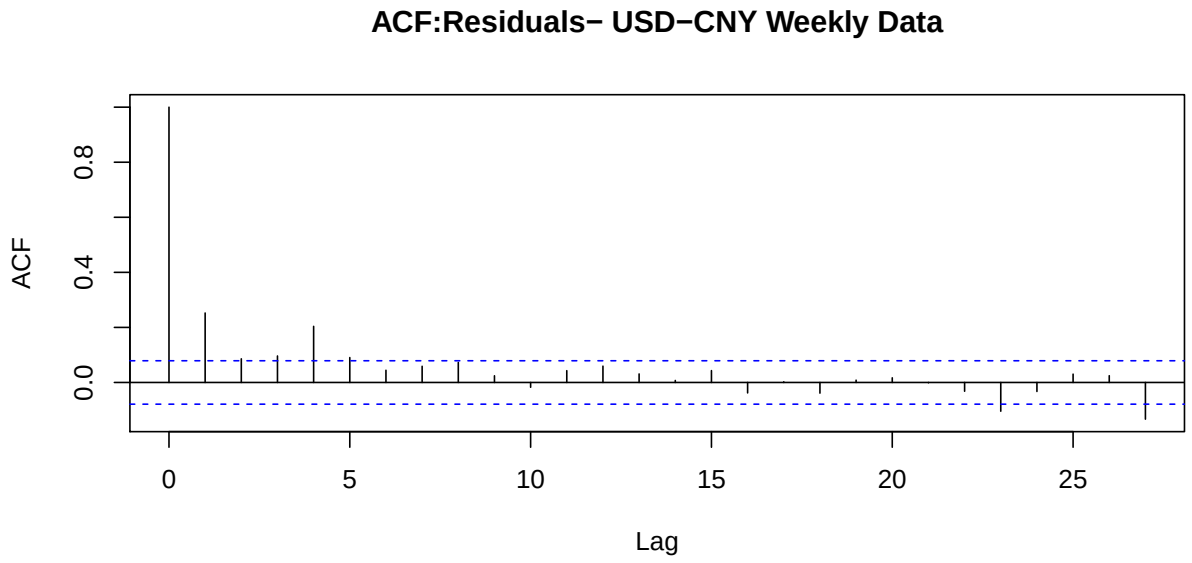
Response: Statistical Significance

2b. Evaluate the model residuals using the ACF and PACF plots, the residual plot and residuals' histogram as well as hypothesis testing for serial correlation for the selected models in (2a) for the two currencies. Does the model fit the time series data? Compare the model fit for the two currency exchange rates.

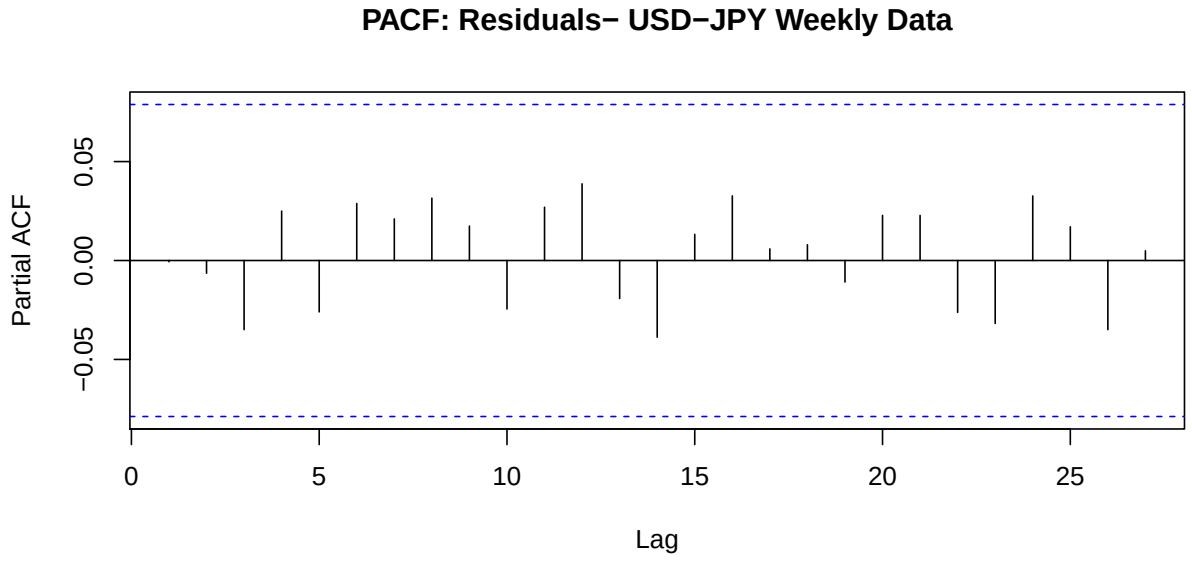
```
acf(resid(final_model1),main="ACF: Residuals- USD-JPY Weekly Data")
```



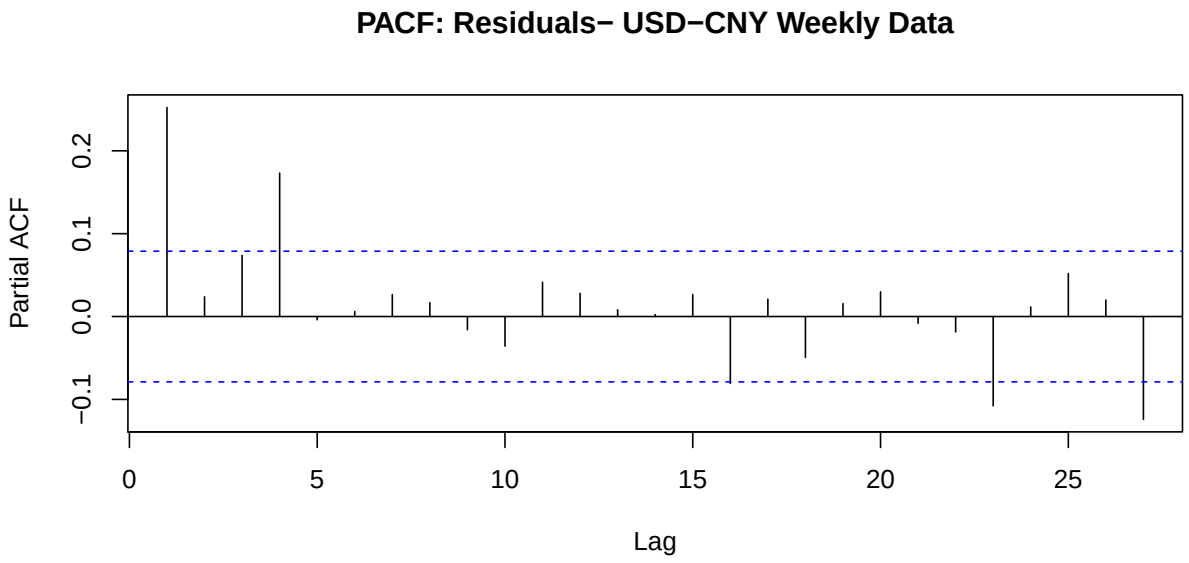
```
acf(resid(final_model3),main= "ACF:Residuals- USD-CNY Weekly Data")
```



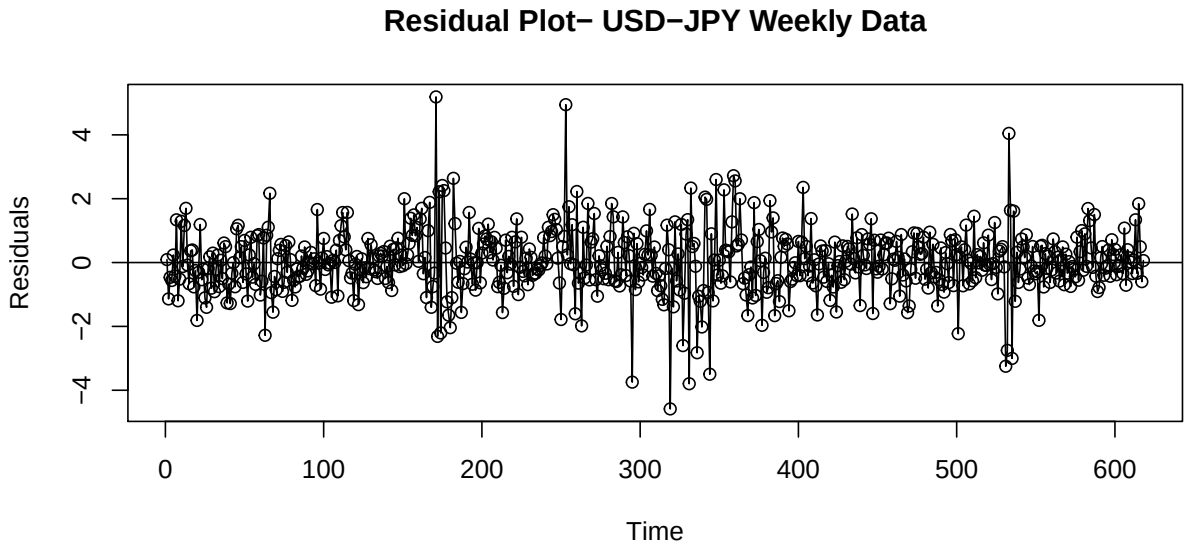

```
pacf(resid(final_model1),main="PACF: Residuals- USD-JPY Weekly Data")
```



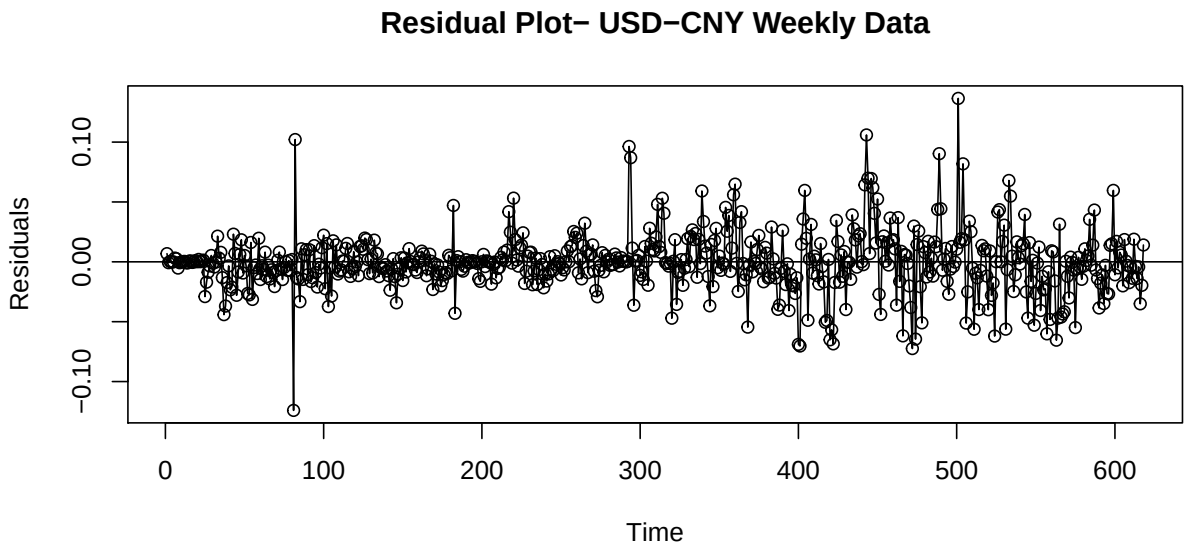

```
pacf(resid(final_model3),main="PACF: Residuals- USD-CNY Weekly Data")
```



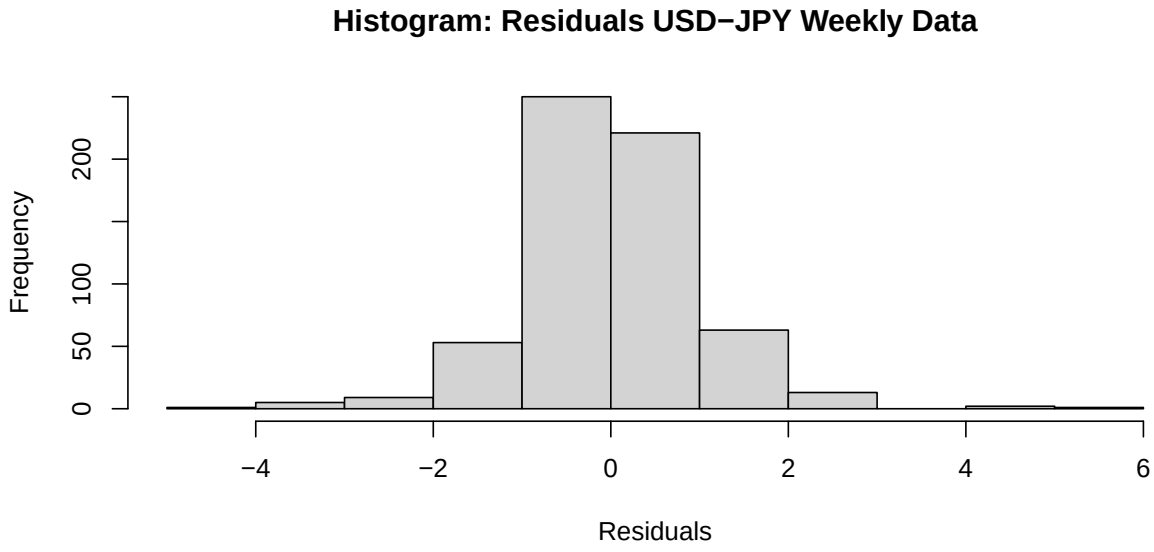

```
plot(resid(final_model1), ylab='Residuals',type='o',main="Residual Plot- USD-JPY Weekly Data")
abline(h=0)
```



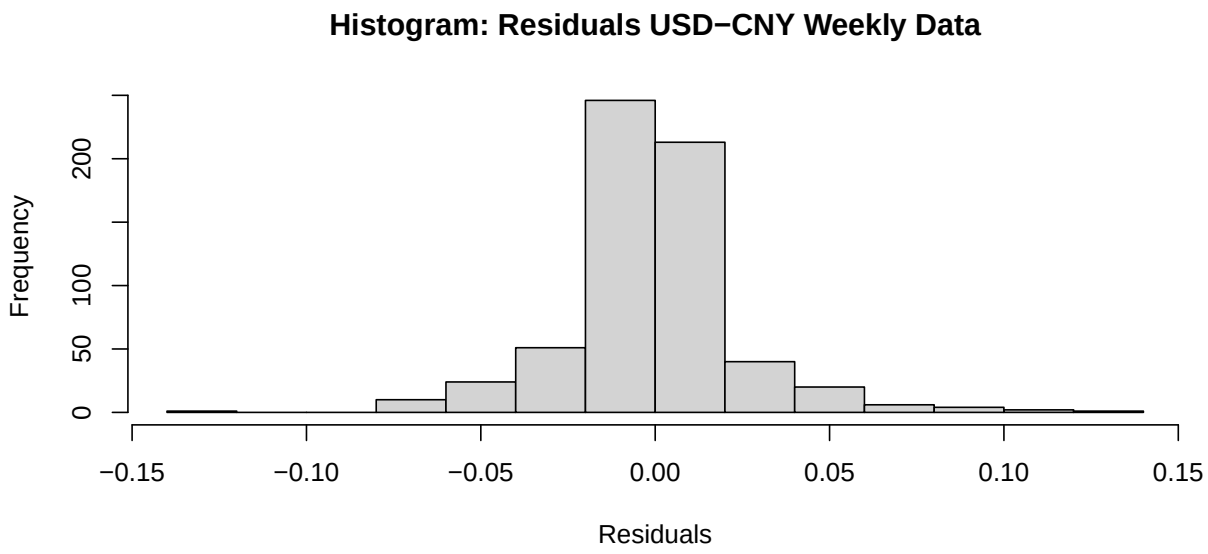

```
plot(resid(final_model3), ylab='Residuals',type='o',main="Residual Plot- USD-CNY Weekly Data")
abline(h=0)
```




```
hist(resid(final_model1),xlab='Residuals',main='Histogram: Residuals USD-JPY Weekly Data')
```




```
hist(resid(final_model3),xlab='Residuals',main='Histogram: Residuals USD-CNY Weekly Data')
```




```
#Hypothesis Test
Box.test(final_model1$resid, lag = (porder1 + qorder1 + 1), type = "Box-Pierce", fitdf = (porder1 + qor
```

```
##
##  Box-Pierce test
##
## data:  final_model1$resid
## X-squared = 3.5187, df = 1, p-value = 0.06068
```

```
Box.test(final_model1$resid, lag = (porder1 + qorder1 + 1), type = "Ljung-Box", fitdf = (porder1 + qord
```

```
##
##  Box-Ljung test
##
## data:  final_model1$resid
## X-squared = 3.5645, df = 1, p-value = 0.05903
```

```
Box.test(final_model3$resid, lag = (0 + 0 + 1), type = "Box-Pierce", fitdf = (0 + 0))
```

```
##
##  Box-Pierce test
##
## data:  final_model3$resid
## X-squared = 39.364, df = 1, p-value = 3.516e-10
```

```
Box.test(final_model3$resid, lag = (0 + 0 + 1), type = "Ljung-Box", fitdf = (0 + 0))
```

```
##
##  Box-Ljung test
##
## data:  final_model3$resid
## X-squared = 39.556, df = 1, p-value = 3.188e-10
```

Response: Residual Analysis

The residual analysis plots are shown above.

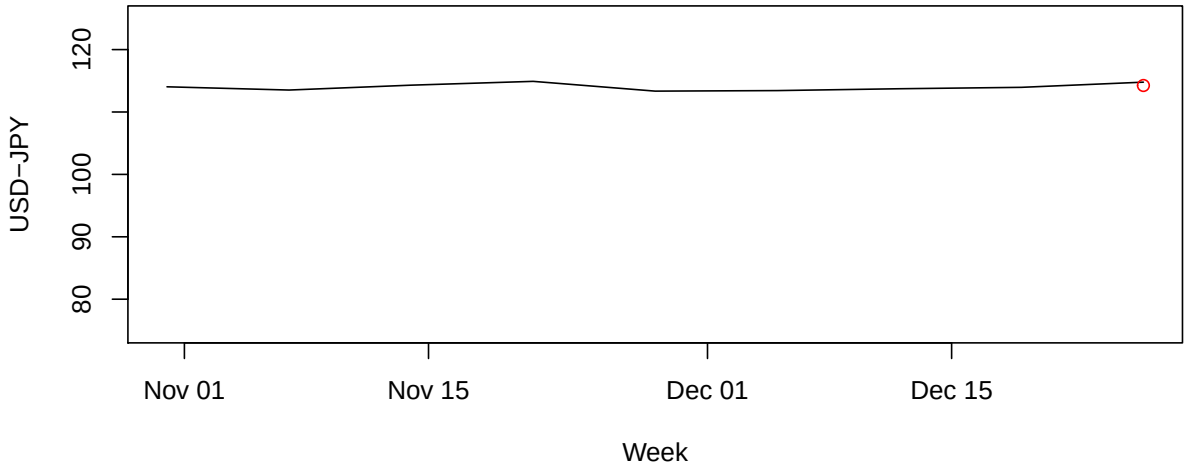
Examining the ACF plot, we can see that optimal model for the weekly USD-JPY data does not appear to violate the stationarity assumption. There is a high autocorrelation value for lag 0, then low amounts within the confidence bands for the remaining lags. The PACF plot is within the confidence band at all points. PACF is not large for any lags. The residual plot and histogram indicate lots of residuals that are close to 0 and only a few big residuals. The residuals appear to be evenly above and below 0. We run a hypothesis test using Box-Pierce as well as Ljung-Box. For this exchange rate, the Box-Pierce test yields a p-value of 0.06068 while the Box-Ljung test yields a p-value of 0.05903. As such, the results are marginally significant, and right on the cutoff of rejecting and failing to reject the null hypothesis. Using a 0.05 significance level, we would just barely conclude that the ARIMA models are an acceptable fit from a serial correlation point of view and fail to reject the null hypothesis. That said, the p-values are still pretty low, so the fit is only marginal.

For the weekly USD-CNY data, the ACF plot indicates that the optimal model may violate stationary. At lag 0, there is a high autocorrelation value. There are data points for other lags that have ACF values outside the confidence band. There are only a few points like this and they are not too far outside the confidence band, so I would conclude that this data either slightly violates stationarity or is weakly stationary. The PACF plot has values outside the confidence band at multiple lags. The residual plot and histogram indicate lots of residuals that are close to 0 and only a few big residuals. The residuals appear to be evenly above and below 0. We run a hypothesis test using Box-Pierce as well as Ljung-Box. For this exchange rate, the Box-Pierce test yields a p-value of 3.516e-10 while the Box-Ljung test yields a p-value of 3.188e-10. As such, the results are statistically significant, and we reject the null hypothesis. Using a 0.05 significance level, we would conclude that ARIMA model is not a good fit for the USD-CNY weekly data.

Comparing the two exchange rates, we can say that the recommended ARIMA model is a more acceptable fit for the USD-JPY weekly data than the USD-CNY data.

**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.

```
## Weekly Prediction over a period of 8 weeks
outpred.8 = NULL
ubound.8 = NULL
lbound.8 = NULL
n = length(jpy.weekly.ts)
for(i in 1:8){
  nfit = n-(8-i+1)
   outjpy.weekly = arima(jpy.weekly.train[1:nfit], order = c(porder1,1,qorder1),method = "ML")
   outpred = predict(outjpy.weekly,n.ahead=1)
   outpred.8 = c(outpred.8,outpred$pred)
   ubound.8 = c(ubound.8, outpred$pred+1.645*outpred$se)
   lbound.8 = c(lbound.8, outpred$pred-1.645*outpred$se)
}


plot(weekly$Date[(n-8):n], jpy.weekly.ts[(n-8):n],type="l", ylim=c(75,125), xlab="Week", ylab="USD-JPY")
points(weekly$Date[(nfit+1):n],outpred$pred,col="red")
```




```
#lines(weekly$Date[(nfit+1):n],ubound.8,lty=3,lwd= 2, col="blue")
#lines(weekly$Date[(nfit+1):n],lbound.8,lty=3,lwd= 2, col="blue")
#points(weekly$date[(nfit+1):n],outpred.8,col="green")
```

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.

```
obs = (jpy.weekly.train[(nfit+1):n])
### Mean Absolute Percentage Error (MAPE)
#mean(abs(pred.1-obs)/obs)
### Precision Measure (PM)
#sum((predprice.10-obs)^2)/sum((obs-mean(obs))^2)
```

Response: Prediction Accuracy

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 2021) 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.

```
jpy.monthly.train <- jpy.monthly.ts[1:142]
jpy.monthly.test <- jpy.monthly.ts[143:144]

cny.monthly.train <- cny.monthly.ts[1:142]
cny.monthly.test <- cny.monthly.ts[143:144]
```

```
#ARIMA-USD-JPY, d = 1
n_month = length(jpy.monthly.train)
norder = 9
p = c(1:norder)-1; q = c(1:norder)-1
aic1_month = matrix(0,norder,norder)
try(
for(i in 1:norder){
   for(j in 1:norder){
    modij1_month = arima(jpy.monthly.train,order = c(p[i],1,q[j]), method='ML', optim.control = list(ma
    aic1_month[i,j] = modij1_month$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n_month/(n_month-p[i]-q[j]-2)
   }
})

#ARIMA-USD-JPY, d = 2
aic2_month = matrix(0,norder,norder)
try(
for(i in 1:norder){
   for(j in 1:norder){
    modij2_month = arima(jpy.monthly.train,order = c(p[i],2,q[j]), method='ML', optim.control = list(ma
    aic2_month[i,j] = modij2_month$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n_month/(n_month-p[i]-q[j]-2)
   }
})

#ARIMA-USD-CNY, d = 1
n_month = length(cny.monthly.train)
norder = 9
p = c(1:norder)-1; q = c(1:norder)-1
aic3_month = matrix(0,norder,norder)
try(
for(i in 1:norder){
   for(j in 1:norder){
    modij3_month = arima(cny.monthly.train,order = c(p[i],1,q[j]), method='ML', optim.control = list(ma
    aic3_month[i,j] = modij3_month$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n_month/(n_month-p[i]-q[j]-2)
   }
})

#ARIMA-USD-CNY, d = 2
aic4_month = matrix(0,norder,norder)
try(
for(i in 1:norder){
   for(j in 1:norder){
    modij4_month = arima(cny.monthly.train,order = c(p[i],2,q[j]), method='ML', optim.control = list(ma
    aic4_month[i,j] = modij4_month$aic-2*(p[i]+q[j]+1)+2*(p[i]+q[j]+1)*n_month/(n_month-p[i]-q[j]-2)
   }
})
```

```
aicv1_month = as.vector(aic1_month)
indexp1_month = rep(c(1:norder),norder)
indexq1_month = rep(c(1:norder),each=norder)
indexaic1_month = which(aicv1_month == min(aicv1_month))
porder1_month = indexp1_month[indexaic1_month]-1
qorder1_month = indexq1_month[indexaic1_month]-1
```

```r
final_model1_month = arima(jpy.monthly.train, order = c(porder1_month,1,qorder1_month), method = "ML")

aicv2_month = as.vector(aic2_month)
indexp2_month = rep(c(1:norder),norder)
indexq2_month = rep(c(1:norder),each=norder)
indexaic2_month = which(aicv2_month == min(aicv2_month))
porder2_month = indexp2_month[indexaic2_month]-1
qorder2_month = indexq2_month[indexaic2_month]-1

final_model2_month = arima(jpy.monthly.train, order = c(porder2_month,2,qorder2_month), method = "ML")

aicv3_month = as.vector(aic3_month)
indexp3_month = rep(c(1:norder),norder)
indexq3_month = rep(c(1:norder),each=norder)
indexaic3_month = which(aicv3_month == min(aicv3_month))
porder3_month = indexp3_month[indexaic3_month]-1
qorder3_month = indexq3_month[indexaic3_month]-1

final_model3_month = arima(cny.monthly.train, order = c(porder3_month,1,qorder3_month), method = "ML")

aicv4_month = as.vector(aic4_month)
indexp4_month = rep(c(1:norder),norder)
indexq4_month = rep(c(1:norder),each=norder)
indexaic4_month = which(aicv4_month == min(aicv4_month))
porder4_month = indexp4_month[indexaic4_month]-1
qorder4_month = indexq4_month[indexaic4_month]-1

final_model4_month = arima(cny.monthly.train, order = c(porder4_month,2,qorder4_month), method = "ML")

final_model1_month$aic
```

```
## [1] 652.551
```

```r
final_model2_month$aic
```

```
## [1] 655.6765
```

```r
final_model3_month$aic
```

```
## [1] -370.8565
```

```r
final_model4_month$aic
```

```
## [1] -359.4961
```

```r
final_model1_month
```

```
##
## Call:
## arima(x = jpy.monthly.train, order = c(porder1_month, 1, qorder1_month), method = "ML")
```

```
##
## Coefficients:
##          ar1      ar2      ma1     ma2
##       0.1681  -0.8348  -0.1998  1.0000
## s.e.  0.0652   0.0503   0.0409  0.2974
##
## sigma^2 estimated as 5.424:  log likelihood = -321.28,  aic = 652.55
```

```
summary(final_model1_month)
```

```
##           Length Class  Mode
## coef        4    -none- numeric
## sigma2      1    -none- numeric
## var.coef   16    -none- numeric
## mask        4    -none- logical
## loglik      1    -none- numeric
## aic         1    -none- numeric
## arma        7    -none- numeric
## residuals 142    ts     numeric
## call        4    -none- call
## series      1    -none- character
## code        1    -none- numeric
## n.cond      1    -none- numeric
## nobs        1    -none- numeric
## model      10    -none- list
```

```
final_model1_month$coef
```

```
##        ar1        ar2        ma1        ma2
##  0.1680540 -0.8347661 -0.1998165  0.9999584
```

```
porder1_month
```

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

```
qorder1_month
```

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

```
final_model3_month
```

```
##
## Call:
## arima(x = cny.monthly.train, order = c(porder3_month, 1, qorder3_month), method = "ML")
##
## Coefficients:
##          ar1      ar2      ar3     ar4      ma1      ma2     ma3
##       0.3714  -0.0691  -0.7374  0.2946  -0.0535  -0.0535  1.0000
## s.e.  0.0828   0.0717   0.0654  0.0813   0.0460   0.0467  0.0432
##
## sigma^2 estimated as 0.003588:  log likelihood = 193.43,  aic = -370.86
```

```
summary(final_model3_month)
```

```
##           Length Class  Mode
## coef        7    -none- numeric
## sigma2      1    -none- numeric
## var.coef   49    -none- numeric
## mask        7    -none- logical
## loglik      1    -none- numeric
## aic         1    -none- numeric
## arma        7    -none- numeric
## residuals 142    ts     numeric
## call        4    -none- call
## series      1    -none- character
## code        1    -none- numeric
## n.cond      1    -none- numeric
## nobs        1    -none- numeric
## model      10    -none- list
```

```
final_model3_month$coef
```

```
##         ar1         ar2         ar3         ar4         ma1         ma2
##  0.37137202 -0.06909669 -0.73736181  0.29460944 -0.05349535 -0.05349285
##         ma3
##  0.99999593
```

```
porder3_month
```

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

```
qorder3_month
```

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

```
## p-value function for the z-test taking as input the test statistic
pvalue.coef <- function(tv) {
  2 * (1 - pnorm(abs(tv)))
}

## Sample Code to compute the test statistics
 tv.jpy.monthly <-as.numeric(final_model1_month$coef)/as.numeric(sqrt(diag(final_model1_month$var.coef))
 tv.cny.monthly <-as.numeric(final_model3_month$coef)/as.numeric(sqrt(diag(final_model3_month$var.coef))

## Apply the pvalue.coef function
 pvalues.jpy.monthly <- sapply(tv.jpy.monthly, pvalue.coef)
 pvalues.cny.monthly <- sapply(tv.cny.monthly, pvalue.coef)

 pvalues.jpy.monthly
```

```
## [1] 9.955193e-03 0.000000e+00 1.052228e-06 7.718771e-04
```

```
pvalues.cny.monthly
```

```
## [1] 7.266286e-06 3.350346e-01 0.000000e+00 2.922070e-04 2.450232e-01
## [6] 2.524896e-01 0.000000e+00
```

Response: Analysis of the ARIMA Fit for the Monthly Data

Using the monthly data, the best ARIMA model for the USD-JPY data has parameters of p = 2, d = 1, and q = 2. This model has an AIC value of 652.551. This model estimates sigma^2 as 5.424 and log likelihood as -321.28. The model coefficients are as follows: AR1 = 0.1680540, AR2 = -0.8347661, MA1 = -0.1998165, and MA2 = 0.9999584. Each coefficient is statistically significant at a level of 0.05. The calculated p-values for these coefficients are 9.955193e-03, 0.000000e+00, 1.052228e-06, and 7.718771e-04, respectively.

Using the monthly data, the best ARIMA model for the USD-CNY data has parameters of p = 4, d = 1, and q = 3. This model has an AIC value of -370.8565. This model estimates sigma^2 as 0.003588 and log likelihood as 193.43. The model coefficients are as follows: AR1 = 0.37137202, AR2 = -0.06909669, AR3 = -0.73736181, AR4 = 0.29460944, MA1 = -0.05349535, MA2 = -0.05349285, and MA3 = 0.99999593. Again, each coefficient is statistically significant at a level of 0.05. The calculated p-values for these coefficients are 7.266286e-06, 3.350346e-01, 0.000000e+00, 2.922070e-04, 2.450232e-01, 2.524896e-01, and 0.000000e+00, respectively.

Response: Monthly vs Weekly Data

It is interesting to compare the ARIMA model results between the monthly and weekly data. The weekly USD-JPY data had orders p = 4, d = 1, and q = 5. The monthly USD-JPY data has orders p = 2, d = 1, and q = 2. The weekly USD-CNY data had orders p = 0, d = 1, and q = 0. The monthly USD-CNY data has orders p = 4, d = 1, and q =3.

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.

```
## Monthly Prediction over a period of 2 months
#outpred.2 = NULL
#ubound.2 = NULL
#lbound.2 = NULL
#n = length(jpy.monthly.ts)
#for(i in 1:2){
  #nfit = n-(2-i+1)
  # outjpy.monthly = arima(jpy.monthly.train[1:nfit], order = c(porder1,1,qorder1),method = "ML")
  # outpred = predict(outjpy.monthly,n.ahead=1)
  # outpred.8 = c(outpred.8,outpred$pred)
  # ubound.8 = c(ubound.8, outpred$pred+1.645*outpred$se)
  # lbound.8 = c(lbound.8, outpred$pred-1.645*outpred$se)
#}
```

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.

Response: Predictions

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?

Response: Prediction Comparison

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: Reflection on ARIMA

I would regard the effectiveness of ARIMA modeling as mixed. On the one hand, there do not appear to be major issues of stationarity with either exchange rate (USD-JPY or USD-CNY) using weekly or monthly data. On the other hand, the residual analysis and hypothesis indicating present issues that indicate the data may not be a good fit for the ARIMA models selected. I would recommend forecasting in the short term and not the long term. A point of caution is that due to the residual analysis, the forecasting should be taken with a grain of salt.