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?

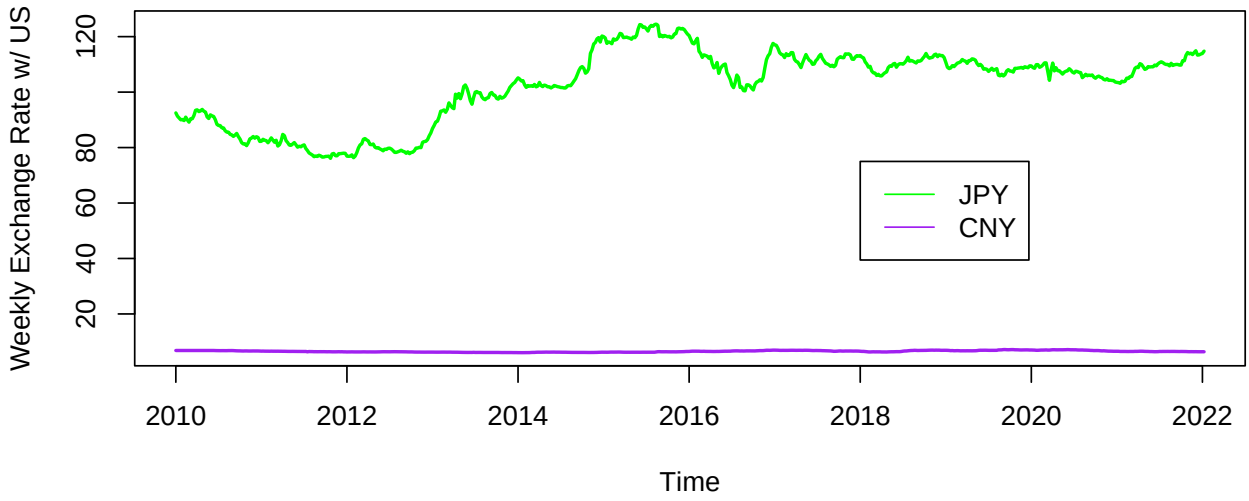
Response

I think you'd want to use weekly instead of monthly time series data in a few situations. One being you just don't have enough data for monthly in order to make a time series model and then forecast (ex. you only have 2 "months" of data vs 8 "weeks" of data). Another situation is in environments where things change frequently like the stock market. That is one example that even fluctuates hourly. Or another situation would be you are trying to build an "early" warning system so to speak (ex. are we using a lot of users on the website per day/week, are we losing sales quickly). You'd rather model weekly in order to react quicker than monthly because by that point, you may be out of business.

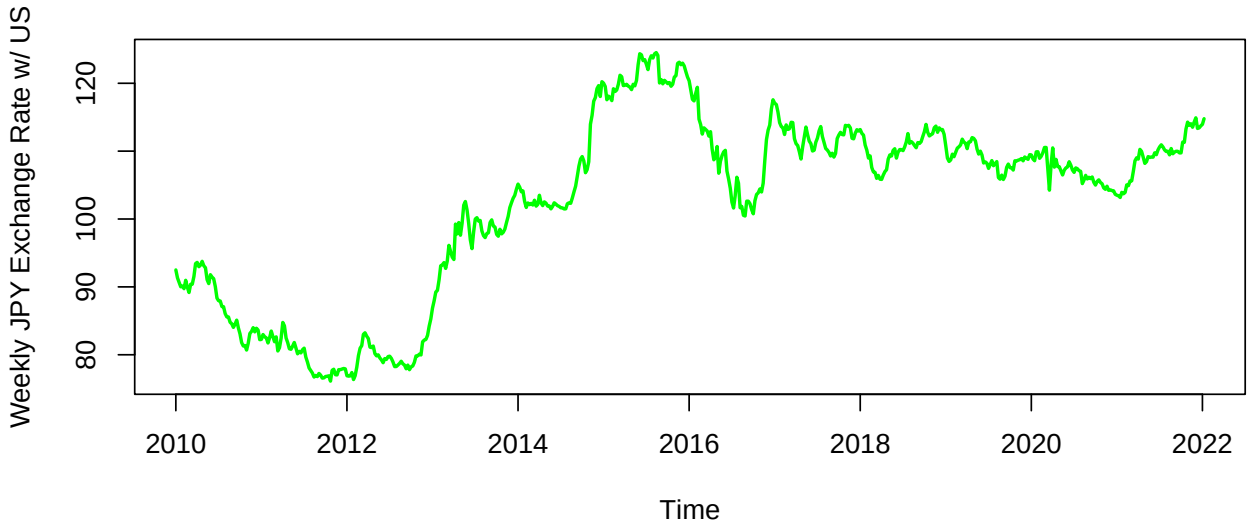
1b. Plot the time series plots for both currency exchange rates comparing weekly vs monthly data. How do the weekly and monthly time series data compare? How do the time series for the two currencies compare?

```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"
```

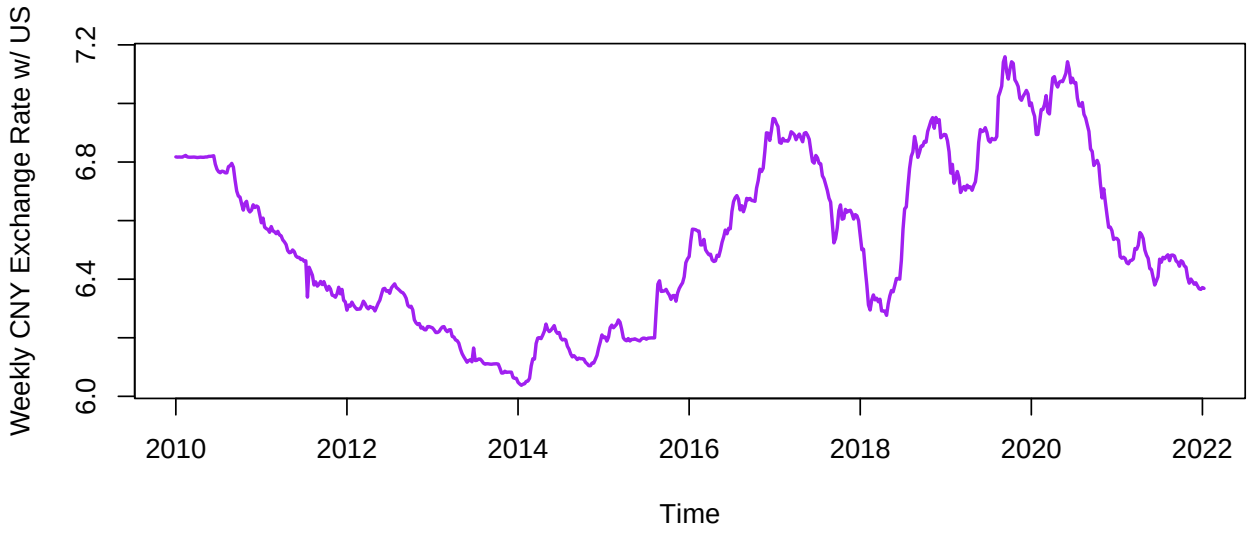
```r
weekly_jp.ts <- ts(weekly$JPY, start = 2010, freq = 52)
weekly_cny.ts <- ts(weekly$CNY, start = 2010, freq = 52)

all_val <- c(weekly_jp.ts, weekly_cny.ts)
ylim <- c(min(all_val),max(all_val))
ts.plot(weekly_jp.ts,lwd=2,col="green", ylim = ylim, ylab="Weekly Exchange Rate w/ US")
lines(weekly_cny.ts,lwd=2,col="purple")
legend(x=2018,y=75,legend=c("JPY","CNY"),lty = 1, col=c("green","purple"))
```



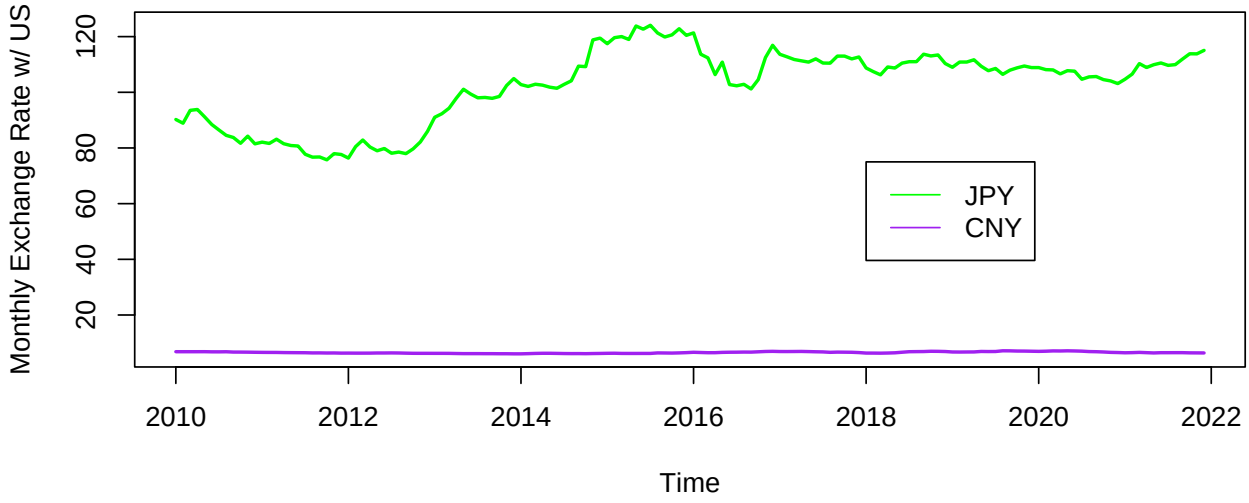

```r
ts.plot(weekly_jp.ts,lwd=2,col="green", ylab="Weekly JPY Exchange Rate w/ US")
```



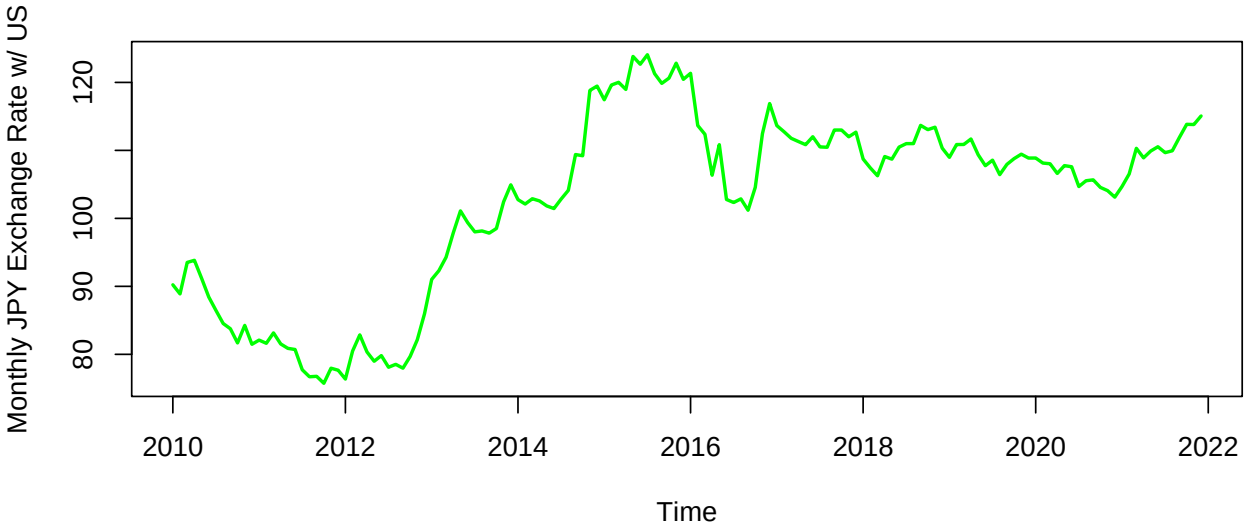

```r
ts.plot(weekly_cny.ts,lwd=2,col="purple", ylab="Weekly CNY Exchange Rate w/ US")
```



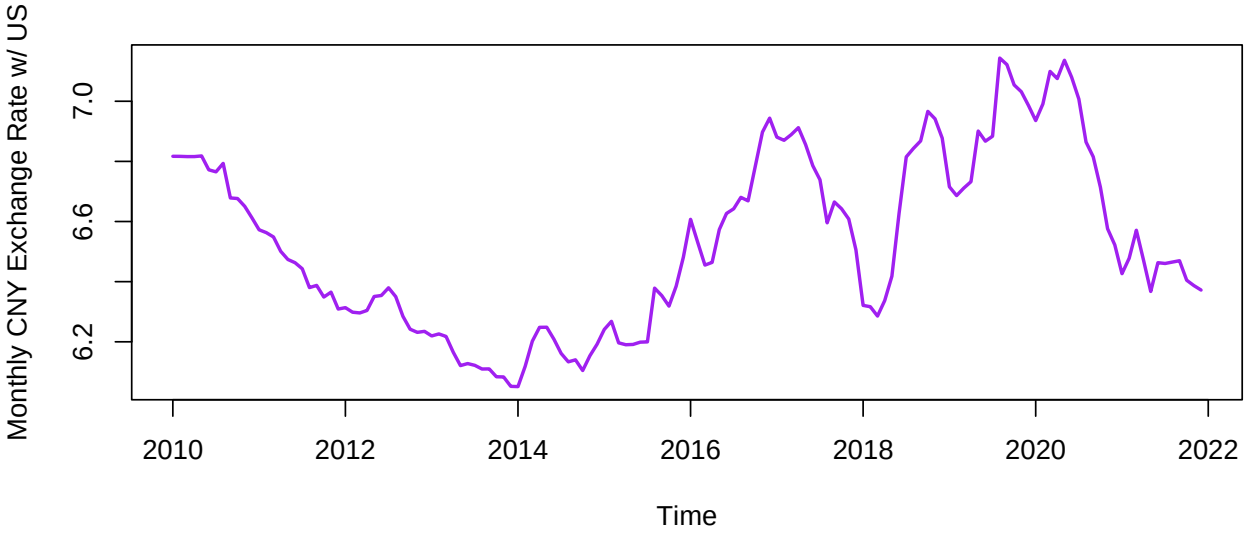

```
monthly_jp.ts <- ts(monthly$JPY, start = 2010, freq = 12)
monthly_cny.ts <- ts(monthly$CNY, start = 2010, freq = 12)

all_val <- c(monthly_jp.ts, monthly_cny.ts)
ylim <- c(min(all_val),max(all_val))
ts.plot(monthly_jp.ts,lwd=2,col="green", ylim = ylim, ylab="Monthly Exchange Rate w/ US")
lines(monthly_cny.ts,lwd=2,col="purple")
legend(x=2018,y=75,legend=c("JPY","CNY"),lty = 1, col=c("green","purple"))
```




```
ts.plot(monthly_jp.ts,lwd=2,col="green", ylab="Monthly JPY Exchange Rate w/ US")
```




```
ts.plot(monthly_cny.ts,lwd=2,col="purple", ylab="Monthly CNY Exchange Rate w/ US")
```



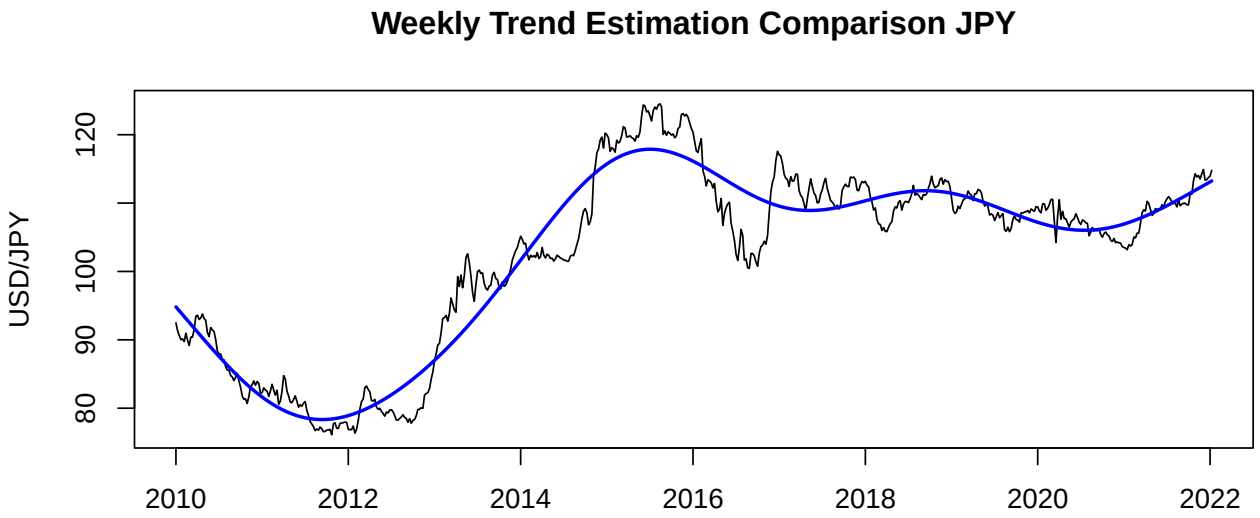
Response: Weekly vs Monthly Time Series data comparison

Weekly vs Monthly: When looking at the weekly graphs and monthly graphs side by side, they look rather similar. They have the same overall shape with each other. The main difference I can see is that with the weekly data shows the smaller peaks and valleys that occur throughout time whereas monthly is a little smoother. This goes for both JPN and CNY.

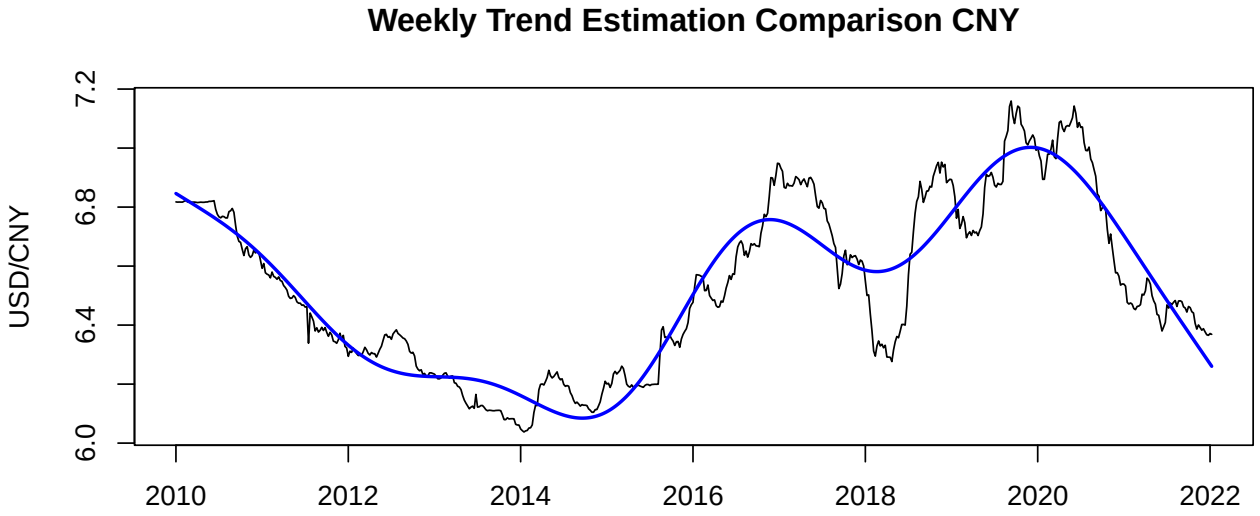
JPN vs CNY: These are pretty different from one another. JPN has a much higher USD currency conversion rate than CNY. JPN fluctuates between 75 and 120. CNY on the other hand fluctuates between only 6 and 8. When you plot them both on a graph together, CNY looks almost like a flat line when compared to JPN. The shapes of the graphs also vary between each other. For JPN, we see a low valley around 2012 and then the exchange rate begins to climb back up between 2012-2016. After 2012 it starts to level out.For CNY, the exchange rate drops and reaches its low in 2014, then starts to rise through 2017, drops again in 2018, rises again through 2020 and then after 2020, it drops.

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
## weekly
# convert X axis to 0-1 scale
points <- 1:length(weekly_jp.ts)
points <- (points - min(points)) / max(points)
# splines
gam.model <- gam(weekly_jp.ts ~ s(points))
gam.fit1 <- ts(fitted(gam.model), start = 2010, frequency = 52)
ts.plot(weekly_jp.ts, xlab = "", ylab = "USD/JPY", main = "Weekly Trend Estimation Comparison JPY")
lines(gam.fit1, lwd = 2,col = "blue")
```

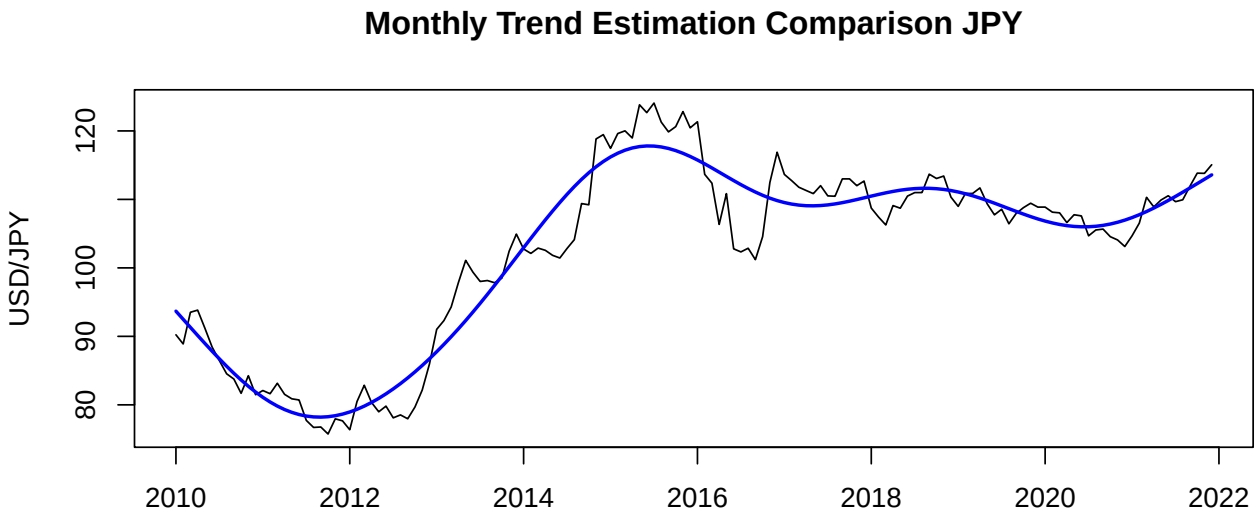



```r
gam.model <- gam(weekly_cny.ts ~ s(points))
gam.fit2 <- ts(fitted(gam.model), start = 2010, frequency = 52)
ts.plot(weekly_cny.ts, xlab = "", ylab = "USD/CNY", main = "Weekly Trend Estimation Comparison CNY")
lines(gam.fit2, lwd = 2,col = "blue")
```

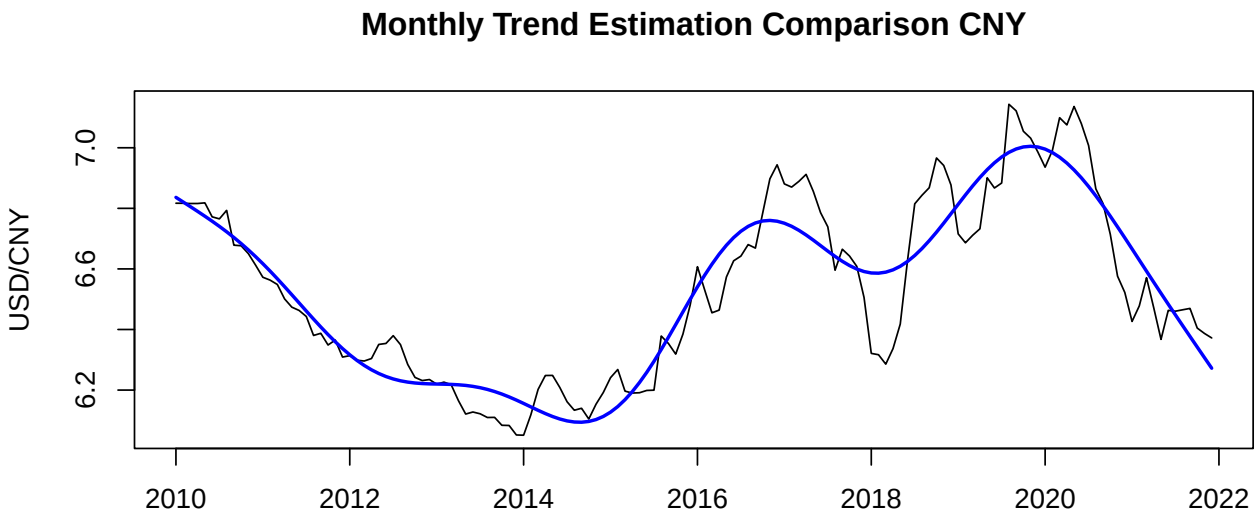



```r
## monthly
# convert X axis to 0-1 scale
points <- 1:length(monthly_jp.ts)
points <- (points - min(points)) / max(points)
# splines
gam.model <- gam(monthly_jp.ts ~ s(points))
```

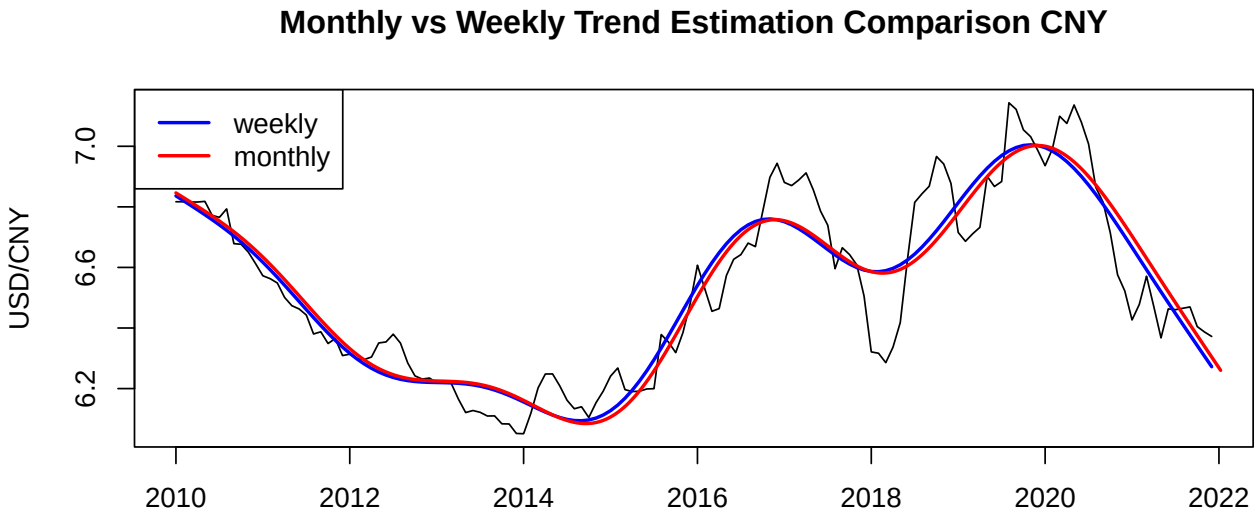
```
gam.fit3 <- ts(fitted(gam.model), start = 2010, frequency = 12)
ts.plot(monthly_jp.ts, xlab = "", ylab = "USD/JPY", main = "Monthly Trend Estimation Comparison JPY")
lines(gam.fit3, lwd = 2,col = "blue")
```



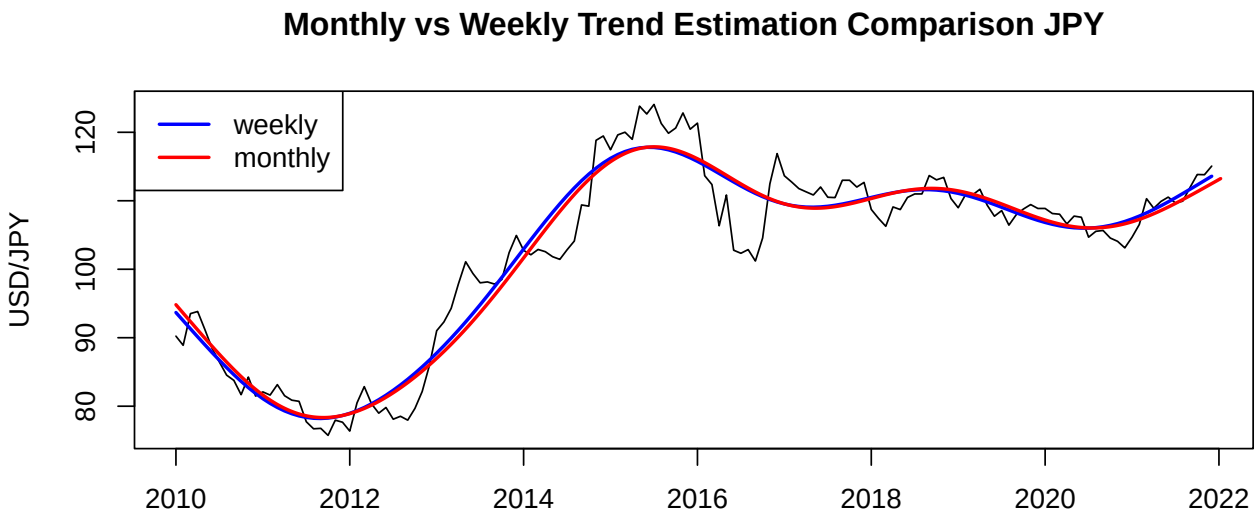

```
gam.model <- gam(monthly_cny.ts ~ s(points))
gam.fit4 <- ts(fitted(gam.model), start = 2010, frequency = 12)
ts.plot(monthly_cny.ts, xlab = "", ylab = "USD/CNY", main = "Monthly Trend Estimation Comparison CNY")
lines(gam.fit4, lwd = 2,col = "blue")
```




```
#Monthly vs weekly per currency
gam.model <- gam(monthly_cny.ts ~ s(points))
gam.fit4 <- ts(fitted(gam.model), start = 2010, frequency = 12)
ts.plot(monthly_cny.ts, xlab = "", ylab = "USD/CNY", main = "Monthly vs Weekly Trend Estimation Compari
lines(gam.fit4, lwd = 2,col = "blue")
lines(gam.fit2, lwd = 2,col = "red")
legend("topleft", legend = c("weekly", "monthly"),
col = c("blue", "red"), lwd = 2)
```




```
gam.model <- gam(monthly_jp.ts ~ s(points))
gam.fit3 <- ts(fitted(gam.model), start = 2010, frequency = 12)
ts.plot(monthly_jp.ts, xlab = "", ylab = "USD/JPY", main = "Monthly vs Weekly Trend Estimation Comparis
lines(gam.fit3, lwd = 2,col = "blue")
lines(gam.fit1, lwd = 2,col = "red")
legend("topleft", legend = c("weekly", "monthly"),
col = c("blue", "red"), lwd = 2)
```

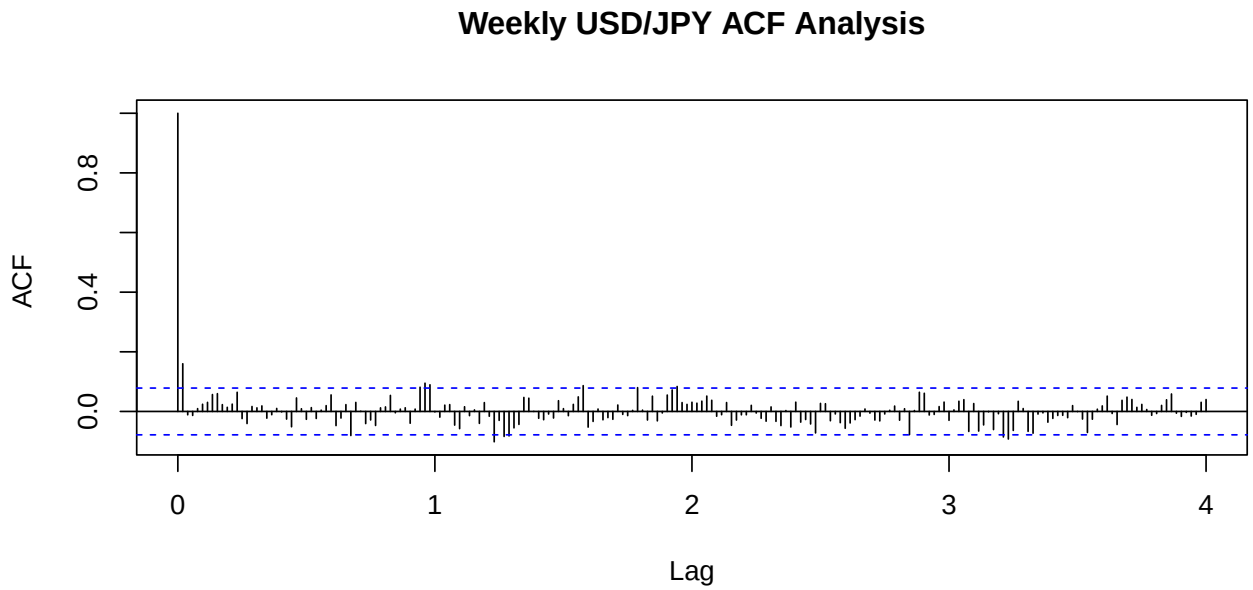


Response: Comparing Trend Estimation using weekly vs Monthly Data **Weekly vs Monthly**: The trends between weekly vs monthly for JPY look incredibly similar. They don't overlap 100%, but are pretty close. The biggest deviation is towards the end where start to split off somewhat. Overall though, they follow the same rises and falls. The trends between weekly vs monthly for CNY are the same story; they look incredibly similar to one another. Don't overlap 100%, but are pretty close.

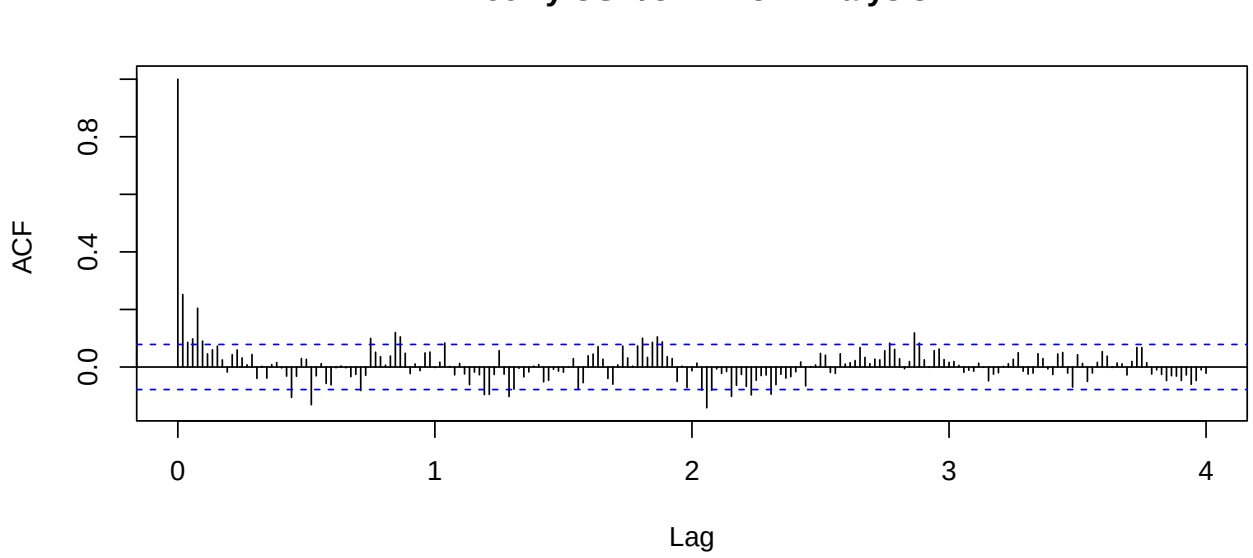
JPY vs CNY: When looking at JPY vs CNY, JPY exchange rate with US decreases from 2010 to approx. 2012, hitting its lowest value. It then increases from 2012 until 2015. It then slowly decreases until 2021 and then starts to rise back up again. For CNY, it drops from 2010 until 2015. Then the exchange rate rises back up from 2015 to 2017. It drops slightly from 2017 to 2018. Then it increases from 2018 to 2020, hitting its peak. After 2020, it drops harshly.

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?

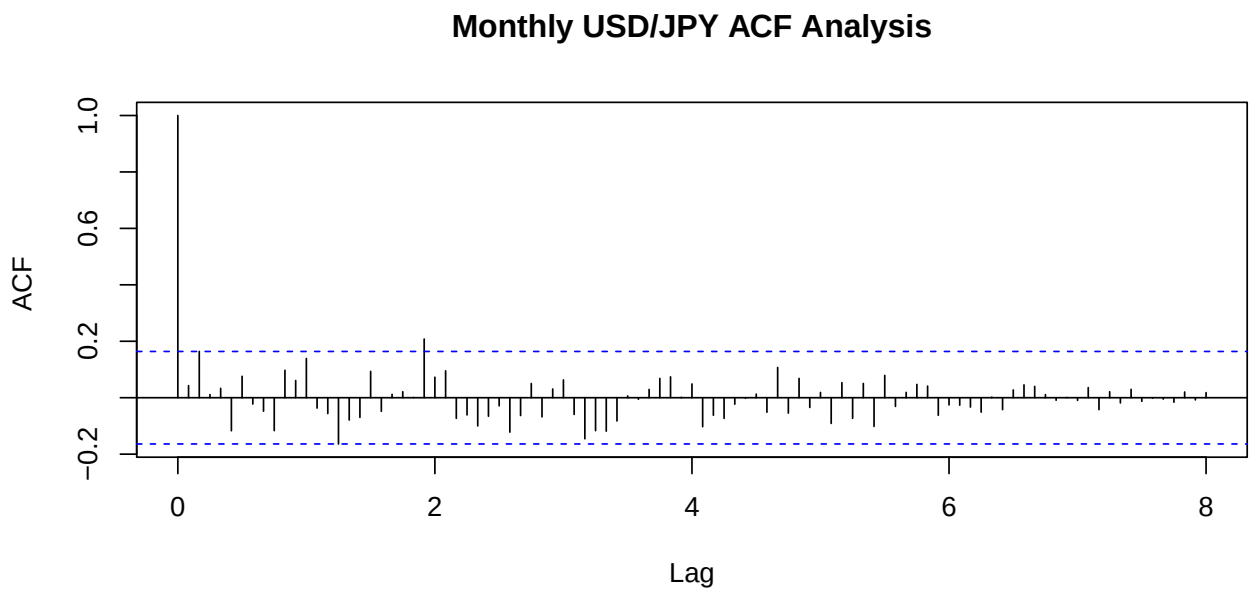
```r
# weekly
acf(diff(weekly_jp.ts), lag.max = 52 * 4, xlab = "Lag",
    ylab = "ACF ", main = "Weekly USD/JPY ACF Analysis")
```



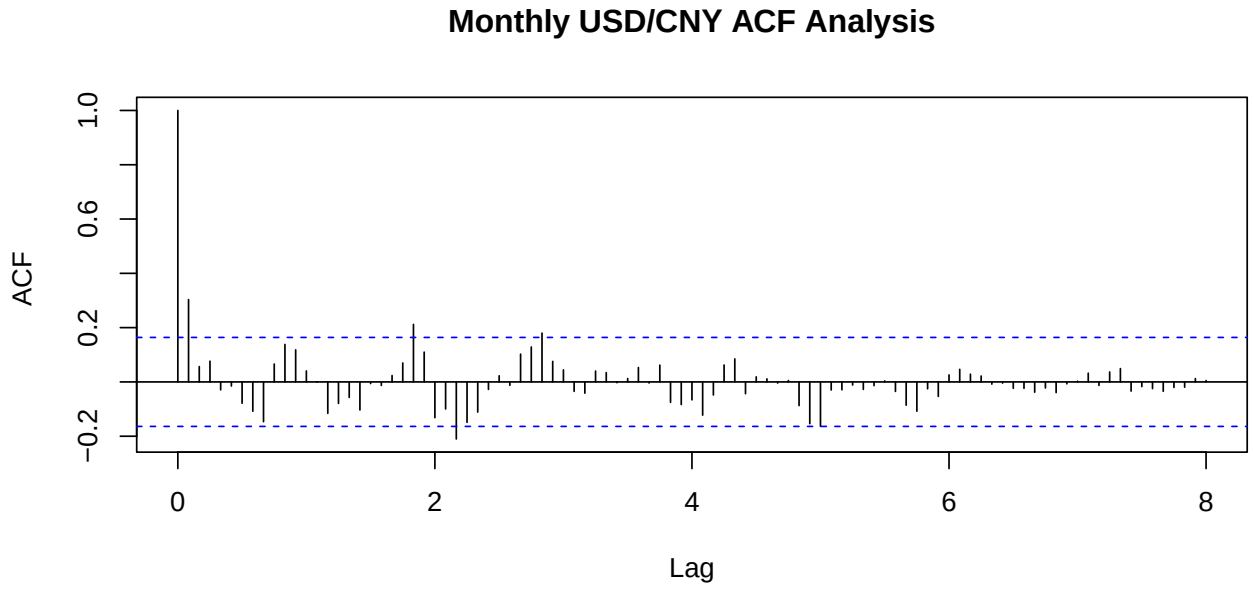

```r
acf(diff(weekly_cny.ts), lag.max = 52 * 4, xlab = "Lag",
    ylab = "ACF ", main = "Weekly USD/CNY ACF Analysis")
```




```r
# monthly
acf(diff(monthly_jp.ts), xlab = "Lag", lag.max = 12 *8,
    ylab = "ACF ", main = "Monthly USD/JPY ACF Analysis")
```




```
acf(diff(monthly_cny.ts), lag.max = 12 * 8, xlab = "Lag",
    ylab = "ACF ", main = "Monthly USD/CNY ACF Analysis")
```



Response: Exploratory Analysis of 1st Order Difference Data

Weekly vs Monthly: When looking at monthly alone, I feel like its easy to perceive patterns, even if they aren't there... especially if you don't increase the lag.max enough. Whereas with weekly, you can definitely see if a pattern is present because of the increase in the amount of data in general. Both monthly's look like they show non-stationarity (especially if you don't increase the lag.max enough) whereas only one of the weekly's looks like it shows stationarity with differencing and the other still looks non-stationary.

JPY vs. CNY: When looking at JPY vs CNY, it looks like JPY's has seasonality when you look at a monthly view, but when you look at the weekly view, there doesn't appear to be seasonality present (spikes fluctuate in a wave-pattern around the 0 axis). However, when looking at CNY's, seasonality is much more prevalent than JPY's in both monthly and weekly. CNY's ACF also seems to have more spikes that are significant compared to JPY's.

Overall: I think the differencing helped JPY's data to become stationary (like we saw last week), but not

necessarily CNY's.

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

```
# jp
library(MuMIn) # this will get us AICc
training_data_weekly_jp <- weekly[1:618, c(1,3)]
test_data_weekly_jp <- weekly[619:626, c(1,3)]

#cny
training_data_weekly_cny <- weekly[1:618, c(1,2)]
test_data_weekly_cny <- weekly[619:626, c(1,2)]

### jp first
weekly_jp.ts <- ts(training_data_weekly_jp$JPY, start = 2010, freq = 52)
# matrix in r is row, column
max_order <- 8
aic_results_d1 <- matrix(0,9,9)

# only d = 1
for(p_val in 0:max_order){
   for(q_val in 0:max_order){
    arima_model <- arima(weekly_jp.ts,order = c(p_val,1,q_val), method='ML')
    aic_results_d1[p_val+1, q_val+1] <- AICc(arima_model)
   }
 }



# only d = 2
aic_results_d2 <- matrix(0,9,9)
for(p_val in 0:max_order){
   for(q_val in 0:max_order){
    if(p_val == 5){
      if(q_val == 8) {
        print("skipping due to error in optim")}
      else {
      arima_model <- arima(weekly_jp.ts,order = c(p_val,2,q_val), method='ML')
      aic_results_d2[p_val+1, q_val+1] <- AICc(arima_model)}}
    else {
      arima_model <- arima(weekly_jp.ts,order = c(p_val,2,q_val), method='ML')
      aic_results_d2[p_val+1, q_val+1] <- AICc(arima_model)}
    }
```

```
 }
```

```
## [1] "skipping due to error in optim"
```

```
print(aic_results_d1)
```

```
##           [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
##  [1,] 1799.885 1784.492 1786.505 1788.385 1790.359 1792.254 1794.111 1794.762
##  [2,] 1785.184 1786.506 1788.516 1790.396 1790.106 1791.629 1793.191 1794.375
##  [3,] 1786.415 1788.434 1790.370 1792.285 1788.438 1792.937 1795.654 1796.388
##  [4,] 1788.427 1790.422 1789.945 1791.877 1792.124 1793.847 1795.690 1798.443
##  [5,] 1790.286 1790.028 1792.072 1791.896 1790.511 1783.964 1792.069 1800.533
##  [6,] 1792.180 1791.747 1794.046 1792.992 1784.992 1790.154 1790.723 1797.590
##  [7,] 1793.735 1793.138 1794.833 1793.089 1791.933 1791.514 1797.634 1786.875
##  [8,] 1793.929 1794.531 1796.583 1798.639 1795.452 1791.101 1785.538 1786.631
##  [9,] 1794.865 1796.577 1798.507 1800.566 1793.675 1795.787 1791.056 1794.893
##           [,9]
##  [1,] 1795.119
##  [2,] 1796.363
##  [3,] 1798.277
##  [4,] 1800.358
##  [5,] 1793.305
##  [6,] 1795.128
##  [7,] 1786.632
##  [8,] 1787.365
##  [9,] 1787.602
```

```
print(aic_results_d2)
```

```
##           [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
##  [1,] 2114.230 1804.596 1790.219 1792.241 1794.140 1796.113 1798.004 1799.854
##  [2,] 2010.534 1790.891 1792.242 1794.264 1796.155 1795.723 1797.141 1798.712
##  [3,] 1951.808 1792.160 1794.188 1796.130 1795.431 1792.552 1799.710 1801.163
##  [4,] 1913.666 1794.182 1796.184 1797.213 1800.194 1800.612 1801.732 1803.477
##  [5,] 1890.715 1796.031 1795.530 1798.629 1794.917 1802.070 1804.088 1800.186
##  [6,] 1873.061 1797.916 1797.258 1799.643 1794.090 1792.902 1806.610 1799.714
##  [7,] 1856.085 1799.445 1798.650 1801.279 1803.254 1799.742 1800.637 1803.771
##  [8,] 1845.978 1799.582 1800.061 1802.123 1804.882 1804.642 1802.469 1796.101
##  [9,] 1842.194 1800.473 1802.118 1804.062 1806.118 1799.224 1792.240 1794.059
##           [,9]
##  [1,] 1800.471
##  [2,] 1804.228
##  [3,] 1801.960
##  [4,] 1798.843
##  [5,] 1804.901
##  [6,]    0.000
##  [7,] 1795.644
##  [8,] 1794.081
##  [9,] 1796.030
```

```
# using example from lecture here
aic_values <-  as.vector(aic_results_d1)
p_0_8 <- rep(c(1:9),9)
q_0_8 <- rep(c(1:9),each=9)
aic_matrix <- which(aic_values == min(aic_values))
final_p <- p_0_8[aic_matrix] - 1
```

```r
final_q <- q_0_8[aic_matrix] - 1
final_model_jp <- arima(weekly_jp.ts,order = c(final_p,1,final_q), method='ML')

final_model_jp
```

```
##
## Call:
## arima(x = weekly_jp.ts, order = c(final_p, 1, final_q), 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
```

```r
summary(final_model_jp)
```

```
##           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
```

```r
final_model_jp$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
```

```r
final_model_jp$sigma2
```

```
## [1] 1.013479
```

```r
### CNY next
weekly_cny.ts <- ts(training_data_weekly_cny$CNY, start = 2010, freq = 52)
# matrix in r is row, column
max_order <- 8
aic_results_d1 <- matrix(0,9,9)

# only d = 1
for(p_val in 0:max_order){
```

```
   for(q_val in 0:max_order){
    arima_model <- arima(weekly_cny.ts,order = c(p_val,1,q_val), method='ML')
    aic_results_d1[p_val+1, q_val+1] <- AICc(arima_model)
   }
 }


# only d = 2
aic_results_d2 <- matrix(0,9,9)
for(p_val in 0:max_order){
   for(q_val in 0:max_order){
    arima_model <- arima(weekly_cny.ts,order = c(p_val,2,q_val), method='ML')
    aic_results_d2[p_val+1, q_val+1] <- AICc(arima_model)
   }
 }

print(aic_results_d1)
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
##  [1,] -2804.388 -2839.707 -2840.367 -2838.342 -2854.110 -2854.977 -2853.701
##  [2,] -2843.144 -2845.463 -2849.637 -2848.707 -2856.526 -2855.100 -2853.272
##  [3,] -2841.476 -2848.458 -2847.957 -2846.720 -2854.737 -2852.544 -2851.168
##  [4,] -2842.831 -2850.685 -2850.904 -2856.901 -2853.735 -2851.699 -2849.123
##  [5,] -2859.615 -2857.585 -2855.556 -2853.812 -2852.475 -2850.490 -2859.980
##  [6,] -2857.584 -2855.540 -2853.487 -2851.976 -2850.463 -2848.412 -2857.543
##  [7,] -2855.562 -2853.491 -2851.431 -2852.106 -2859.095 -2858.005 -2854.714
##  [8,] -2853.964 -2851.966 -2856.973 -2860.856 -2853.167 -2851.206 -2851.610
##  [9,] -2852.064 -2850.022 -2859.336 -2852.962 -2851.140 -2849.143 -2847.476
##           [,8]      [,9]
##  [1,] -2852.022 -2851.145
##  [2,] -2851.226 -2849.200
##  [3,] -2849.146 -2853.507
##  [4,] -2859.483 -2851.494
##  [5,] -2857.415 -2850.074
##  [6,] -2855.343 -2852.906
##  [7,] -2847.790 -2851.406
##  [8,] -2861.072 -2854.804
##  [9,] -2865.374 -2849.557
```

```
print(aic_results_d2)
```

```
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
##  [1,] -2551.753 -2811.901 -2828.752 -2828.380 -2826.365 -2841.185 -2841.963
##  [2,] -2650.260 -2830.890 -2832.726 -2828.952 -2827.087 -2843.650 -2842.359
##  [3,] -2713.454 -2829.007 -2827.081 -2827.618 -2828.383 -2841.903 -2839.601
##  [4,] -2780.459 -2829.884 -2825.726 -2830.220 -2838.297 -2840.281 -2837.891
##  [5,] -2786.106 -2846.830 -2844.788 -2842.764 -2841.019 -2839.754 -2837.673
##  [6,] -2791.435 -2844.788 -2842.732 -2840.680 -2838.796 -2837.739 -2836.011
##  [7,] -2797.871 -2842.771 -2840.678 -2838.629 -2848.091 -2847.184 -2844.848
##  [8,] -2801.429 -2841.212 -2838.887 -2837.488 -2846.867 -2840.344 -2843.149
##  [9,] -2801.423 -2839.335 -2837.276 -2846.533 -2833.871 -2831.525 -2837.322
##           [,8]      [,9]
##  [1,] -2840.732 -2839.080
##  [2,] -2840.556 -2838.492
```

```
##  [3,] -2838.632 -2836.611
##  [4,] -2835.468 -2835.226
##  [5,] -2846.052 -2844.532
##  [6,] -2833.930 -2842.596
##  [7,] -2841.668 -2842.711
##  [8,] -2844.914 -2844.373
##  [9,] -2835.634 -2837.789
```

```
# using example from lecture here
aic_values <-  as.vector(aic_results_d1)
p_0_8 <- rep(c(1:9),9)
q_0_8 <- rep(c(1:9),each=9)
aic_matrix <- which(aic_values == min(aic_values))
final_p <- p_0_8[aic_matrix] - 1
final_q <- q_0_8[aic_matrix] - 1
final_model_cny <- arima(weekly_cny.ts,order = c(final_p,1,final_q), method='ML', optim.control = list(
```

```
final_model_cny
```

```
##
## Call:
## arima(x = weekly_cny.ts, order = c(final_p, 1, final_q), method = "ML", optim.control = list(maxit =
##
## Coefficients:
##           ar1      ar2      ar3     ar4      ar5      ar6     ar7      ar8
##       -0.1573  -0.1582  -0.4595  0.1444  -0.1977  -0.0223  0.8070  -0.0633
## s.e.   0.1203   0.0834   0.0786  0.1137   0.0660   0.0794  0.0648   0.0607
##          ma1     ma2     ma3     ma4     ma5     ma6      ma7
##       0.3945  0.2955  0.5878  0.1113  0.3304  0.2060  -0.7154
## s.e.  0.1120  0.1241  0.1218  0.1530  0.1219  0.1242   0.1119
##
## sigma^2 estimated as 0.0005211:  log likelihood = 1449.46,  aic = -2866.92
```

```
summary(final_model_cny)
```

```
##           Length Class  Mode
## coef       15    -none- numeric
## sigma2      1    -none- numeric
## var.coef  225    -none- numeric
## mask       15    -none- logical
## loglik      1    -none- numeric
## aic         1    -none- numeric
## arma        7    -none- numeric
## residuals 618    ts     numeric
## call        5    -none- call
## series      1    -none- character
## code        1    -none- numeric
## n.cond      1    -none- numeric
## nobs        1    -none- numeric
## model      10    -none- list
```

```
final_model_cny$coef
```

```
##         ar1         ar2         ar3         ar4         ar5         ar6
## -0.15731932 -0.15821869 -0.45949698  0.14443518 -0.19769122 -0.02230005
##         ar7         ar8         ma1         ma2         ma3         ma4
```

```
##  0.80703376 -0.06333756  0.39448492  0.29548347  0.58783891  0.11130607
##         ma5         ma6         ma7
##  0.33042603  0.20596928 -0.71539535
```

```
final_model_cny$sigma2
```

```
## [1] 0.0005210837
```

Response: Analysis of the ARIMA Fit for the Weekly and Monthly Data Even though the header for this response says "monthly" in it, I am only focusing on weekly since that is what 2a asked to examine.

For JPY: For this process, I iterated through 0-8 for both the p value and q value of ARIMA, along with 1 and 2 for the d value. For each iteration, I examined the AICc (found a great package that would let you calculate this!) value to see if it increased or decreased. I noticed that every combination that included d=2 had a higher AIC value than its counterpart using d=1. Additionally, I had a lot more issues with d=2 as well (getting errors in the optimization math under the hood in some cases, specifically for p=5 and q=8, needed to skip around that combo and move on to the next numbers). Though in either case, most values fell between 1700 and 1800. Comparing the two matrices, d=1 seemed to be the way to go. After numerous iterations and without looking at statistical significance at this point, it looked like arima(4,1,5) produced the best results in terms of AICc (1783.964) for JPY. Coefficients are shown above along with sigma2.

For CNY: For this process, I iterated through 0-8 for both the p value and q value of ARIMA, along with 1 and 2 for the d value. For each iteration, I examined the AICc (found a great package that would let you calculate this!) value to see if it increased or decreased. CNY was able to use d=2 for all the values. This time around for both d=1 and d=2, the AICc values were all negative, mainly between -2900 and -2800. After numerous iterations and without looking at statistical inference at this point, it looked like arima(8,1,7) produced the best results in terms of AICc (-2848.091) for CNY. Coefficients are shown above along with sigma2.

```
## 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 (from professor)
tv_jpy_weekly <-as.numeric(final_model_jp$coef)/as.numeric(sqrt(diag(final_model_jp$var.coef)))
tv_cny_weekly <-as.numeric(final_model_cny$coef)/as.numeric(sqrt(diag(final_model_cny$var.coef)))

## get p-vals
library(lmtest)
coeftest(final_model_jp)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
## ar1  0.175818   0.012476  14.0929 < 2.2e-16 ***
## ar2  1.305782   0.014983  87.1503 < 2.2e-16 ***
## ar3  0.203681   0.011745  17.3414 < 2.2e-16 ***
## ar4 -0.964743   0.014000 -68.9092 < 2.2e-16 ***
## ma1 -0.012556   0.041906  -0.2996    0.7645
## ma2 -1.359486   0.017598 -77.2537 < 2.2e-16 ***
## ma3 -0.408874   0.055901  -7.3143 2.588e-13 ***
## ma4  0.953252   0.017431  54.6881 < 2.2e-16 ***
## ma5  0.180930   0.040646   4.4514 8.532e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coeftest(final_model_cny)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1 -0.157319   0.120330 -1.3074 0.1910779
## ar2 -0.158219   0.083406 -1.8970 0.0578324 .
## ar3 -0.459497   0.078641 -5.8430 5.128e-09 ***
## ar4  0.144435   0.113744  1.2698 0.2041478
## ar5 -0.197691   0.066043 -2.9934 0.0027591 **
## ar6 -0.022300   0.079363 -0.2810 0.7787196
## ar7  0.807034   0.064849 12.4448 < 2.2e-16 ***
## ar8 -0.063338   0.060684 -1.0437 0.2966095
## ma1  0.394485   0.112030  3.5213 0.0004295 ***
## ma2  0.295483   0.124117  2.3807 0.0172801 *
## ma3  0.587839   0.121818  4.8256 1.396e-06 ***
## ma4  0.111306   0.152989  0.7275 0.4668921
## ma5  0.330426   0.121893  2.7108 0.0067122 **
## ma6  0.205969   0.124160  1.6589 0.0971353 .
## ma7 -0.715395   0.111891 -6.3937 1.620e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

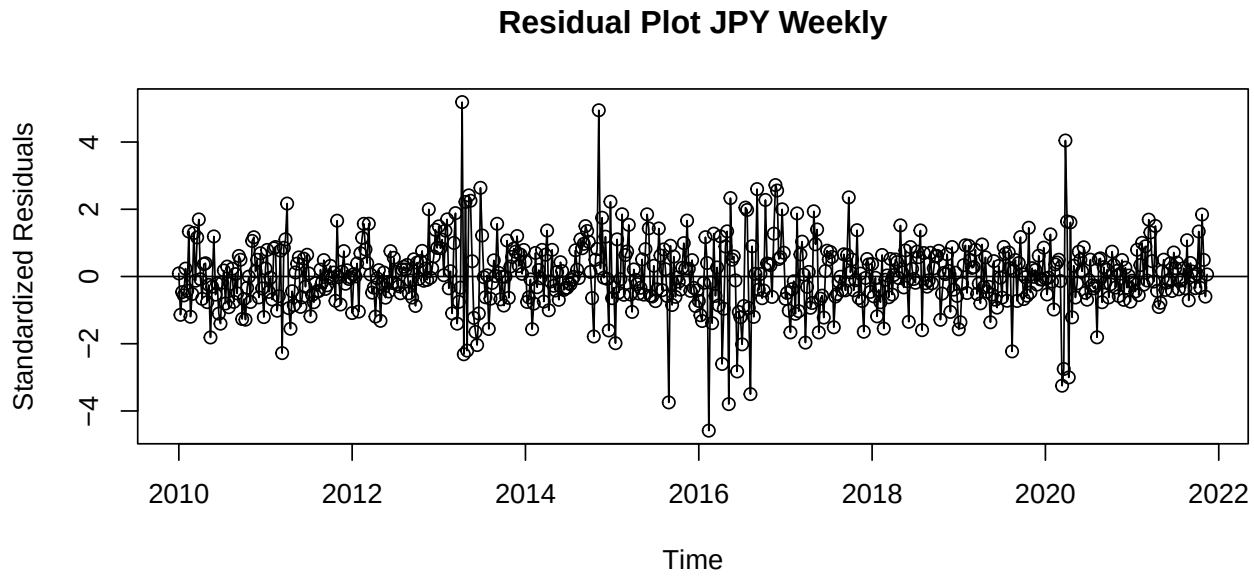
Response: Statistical Significance

For JPY Almost all the coefficients selected (AR(1) through AR(4) and MA(1) through MA(5)) are statistically significant at the 95% level. The one that is not significant at that level is MA(1). That being said, we don't necessarily just want to select the the coefficients with the best p-value. That would be "p-hacking" and definitely not the best way to select coefficients. Sometimes, those with higher p-values do help to improve the model. Though if we did only look at statistical significance (again, not really recommended) of the coefficients, I don't think a lower order model would be needed here, maybe only need to add a seasonal argument in the arima function.

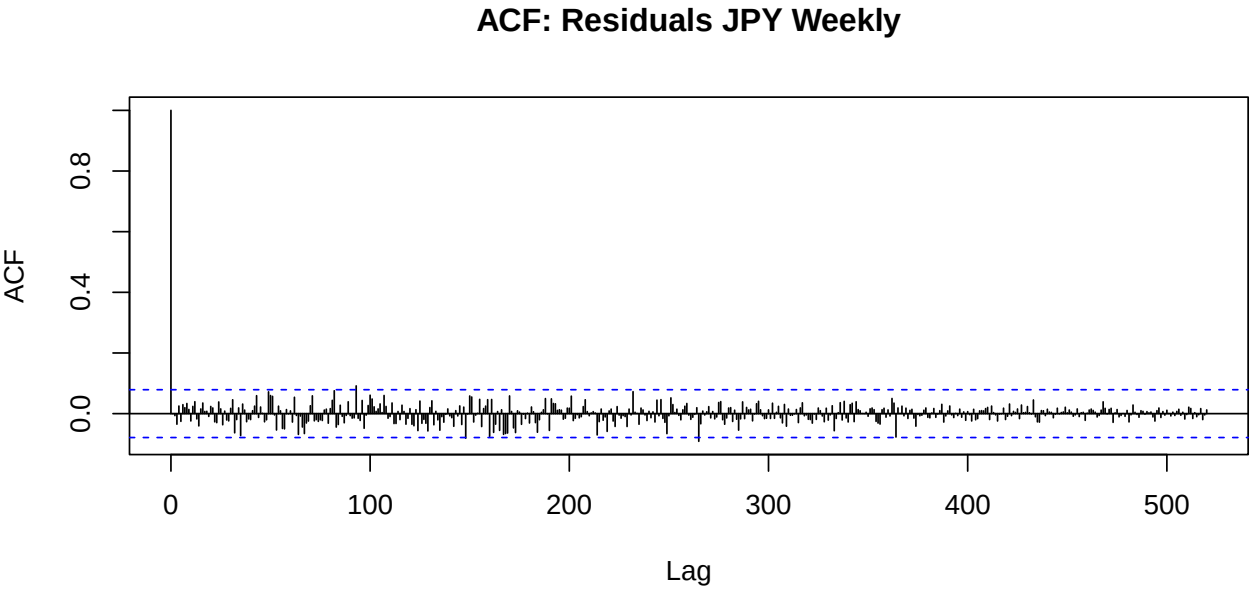
For CNY Some of the coefficients selected (AR(1) through AR(8) and MA(1) through MA(7)) are statistically significant at the 95% level. The ones that aren't significant are: AR(1), AR(4), AR(6), AR(8), MA(4), MA(6). That being said, we don't necessarily just want to select the the coefficients with the best p-value. That would be "p-hacking" and definitely not the best way to select coefficients. Sometimes, those with higher p-values do help to improve the model. Also though, this model could be over-parameterized. Though if we did only look at statistical significance (again, not really recommended) of the coefficients, maybe a lower MA and AR order. Could also add in a seasonal argument to the arima function.

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.

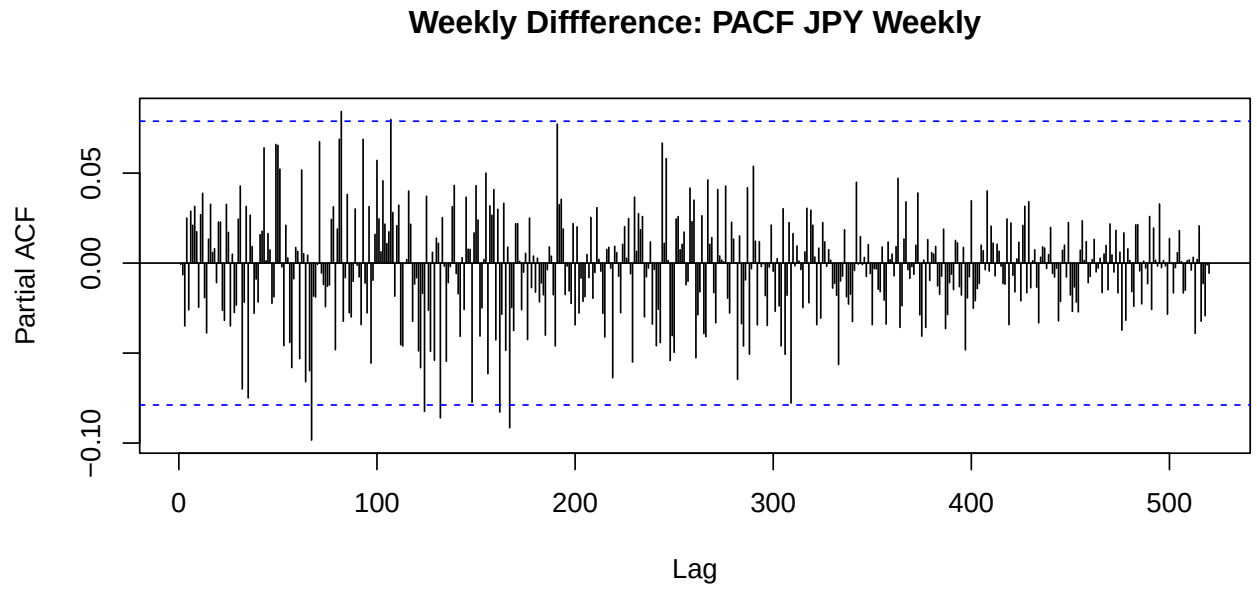
```
# jp
plot(resid(final_model_jp), ylab='Standardized Residuals',type='o',
     main="Residual Plot JPY Weekly")
abline(h=0)
```



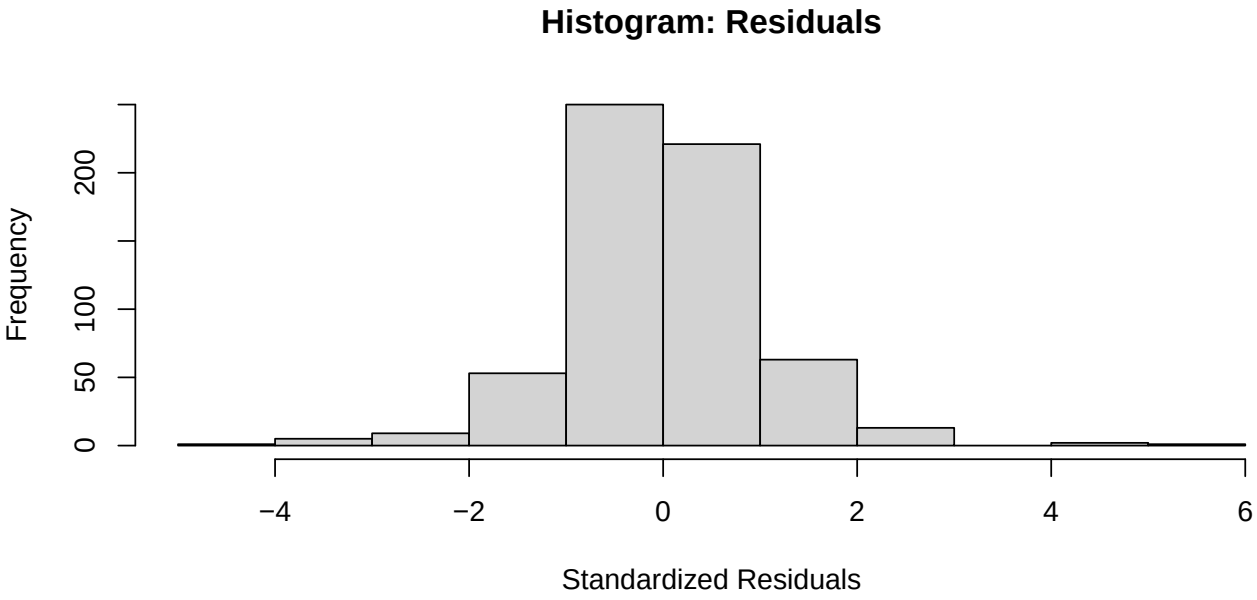

```
acf(as.vector(resid(final_model_jp)),lag.max=52*10,main="ACF: Residuals JPY Weekly")
```



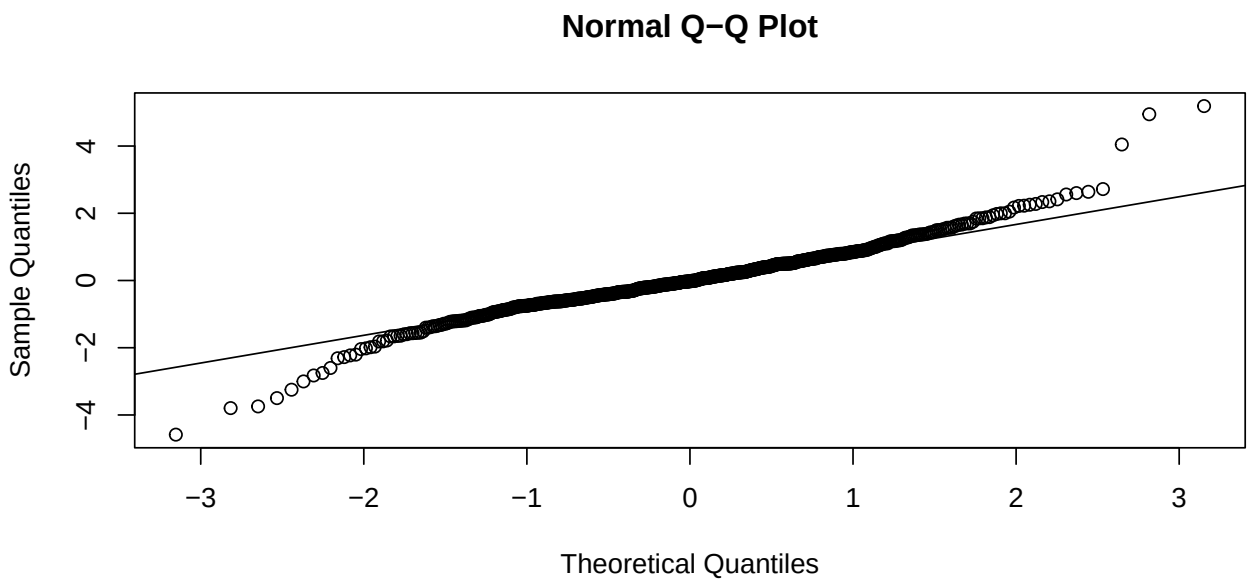

```
acf(as.vector(resid(final_model_jp)),type="partial",
    main='Weekly Diffference: PACF JPY Weekly', lag.max =  52*10)
```



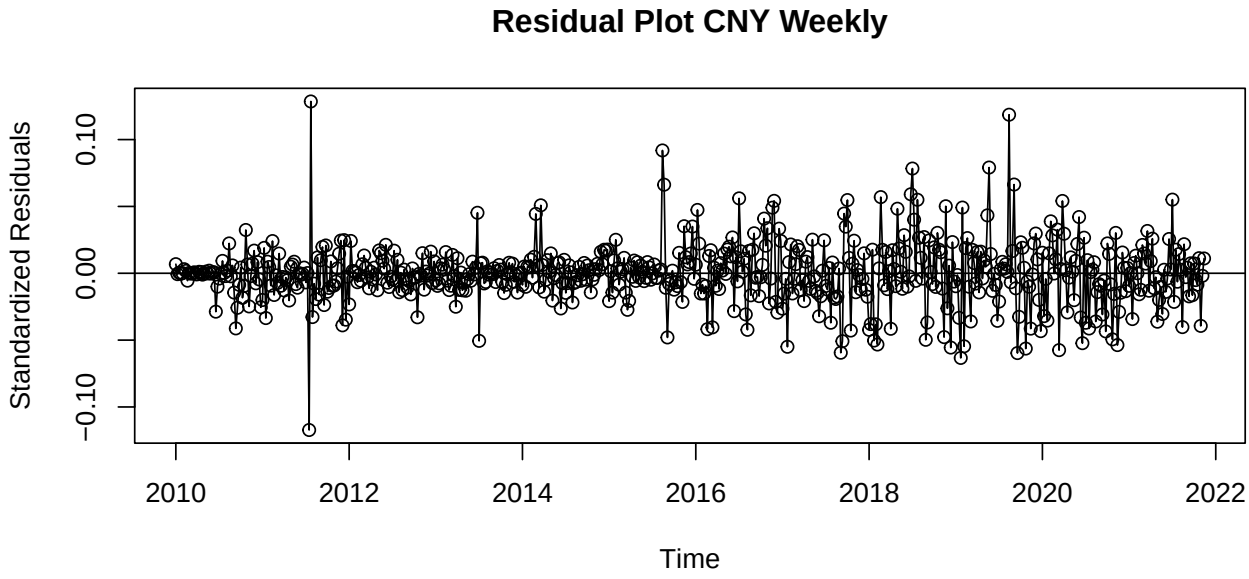

```
hist(resid(final_model_jp),xlab='Standardized Residuals',
     main='Histogram: Residuals')
```



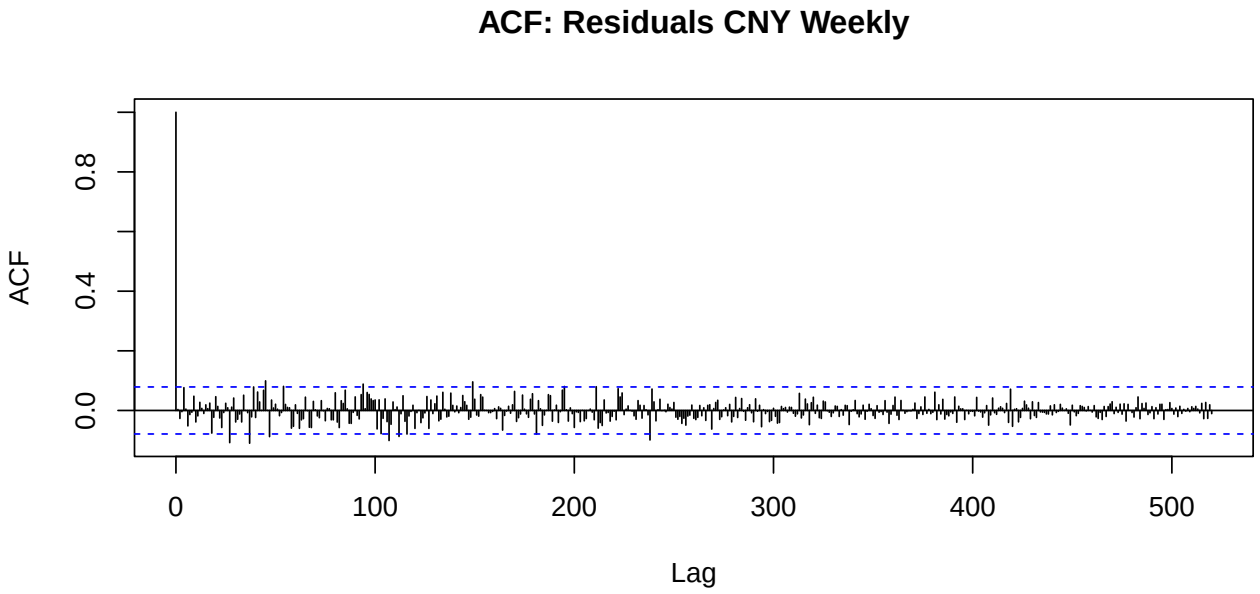

```
qqnorm(resid(final_model_jp))
qqline(resid(final_model_jp))
```



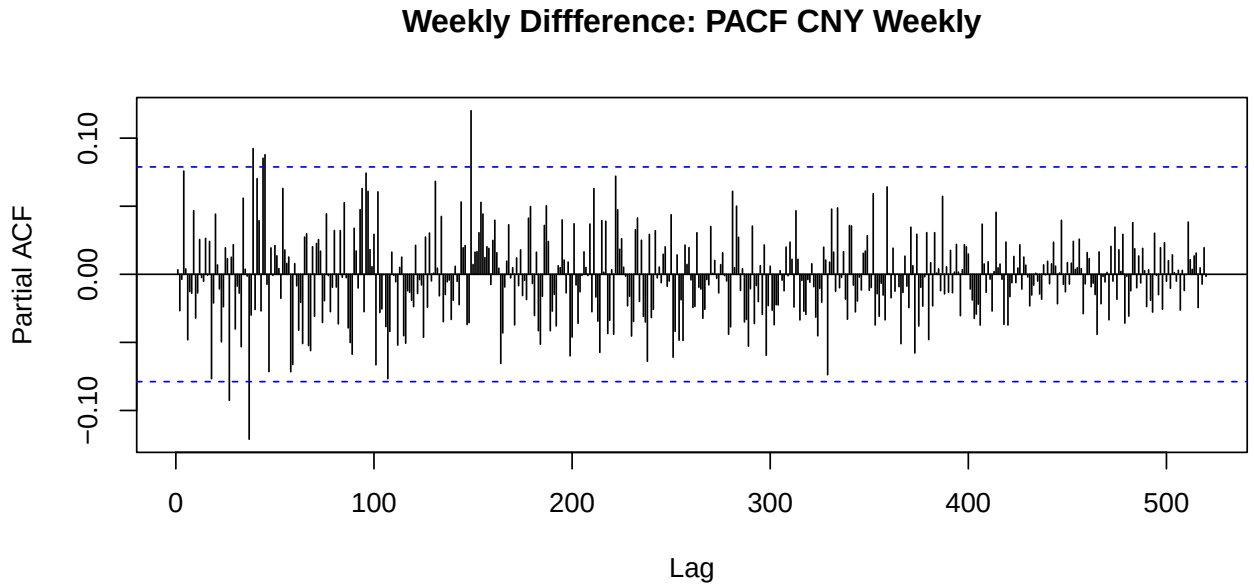

```
# cny
plot(resid(final_model_cny), ylab='Standardized Residuals',type='o',
     main="Residual Plot CNY Weekly")
abline(h=0)
```



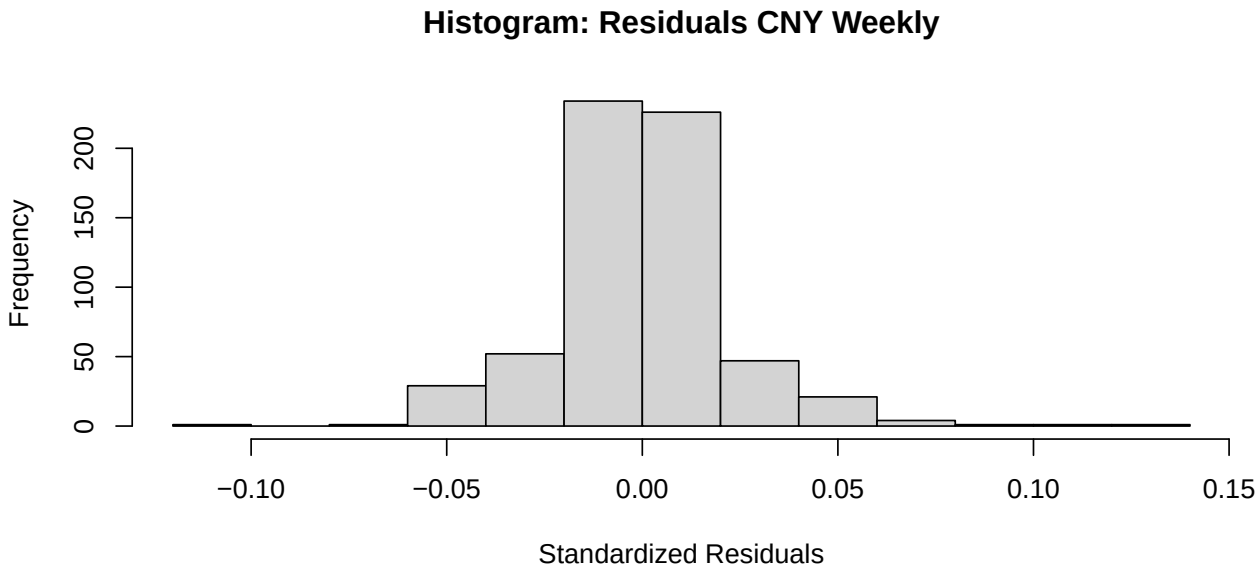

```
acf(as.vector(resid(final_model_cny)),lag.max=52*10,main="ACF: Residuals CNY Weekly")
```



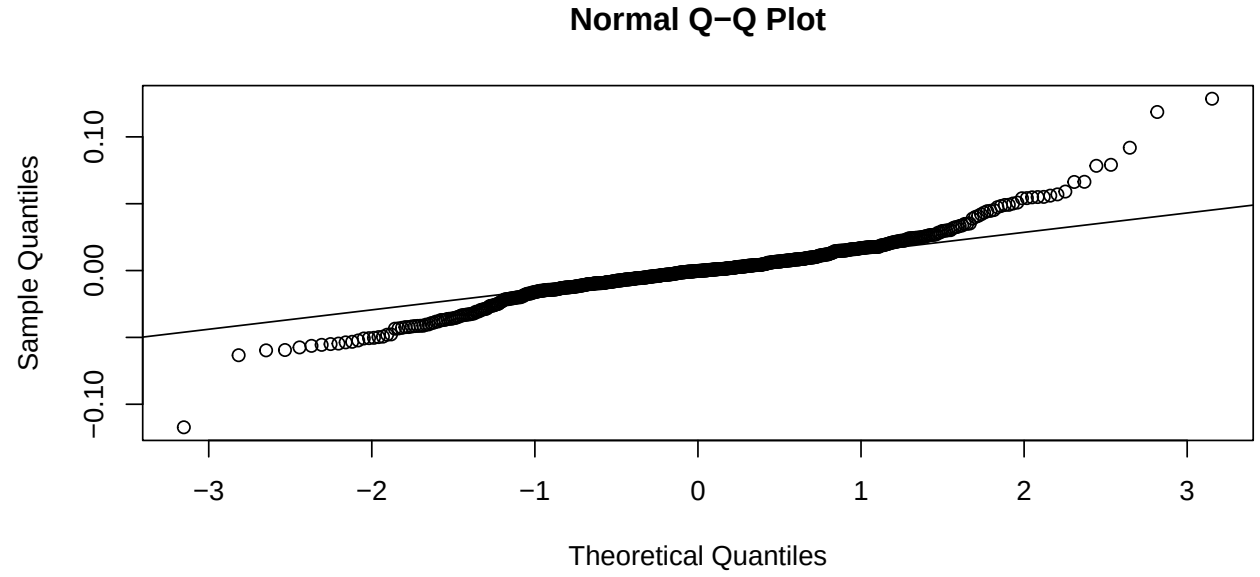

```
acf(as.vector(resid(final_model_cny)),type="partial",
    main='Weekly Diffference: PACF CNY Weekly', lag.max =  52*10)
```




```
hist(resid(final_model_cny),xlab='Standardized Residuals',main='Histogram: Residuals CNY Weekly')
```




```
qqnorm(resid(final_model_cny))
qqline(resid(final_model_cny))
```




```
# definition: serial correlation means the errors that are associated with a given time period carry ov

# let's use ljung box
Box.test(final_model_jp$residuals, lag = (4+5+1), type = "Ljung-Box", fitdf = (4+5))
```

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

```
Box.test(final_model_cny$residuals, lag = (8+7+1), type = "Ljung-Box", fitdf = (8+7))
```

```
##
##  Box-Ljung test
```

```
##
## data:  final_model_cny$residuals
## X-squared = 9.3321, df = 1, p-value = 0.002252
```

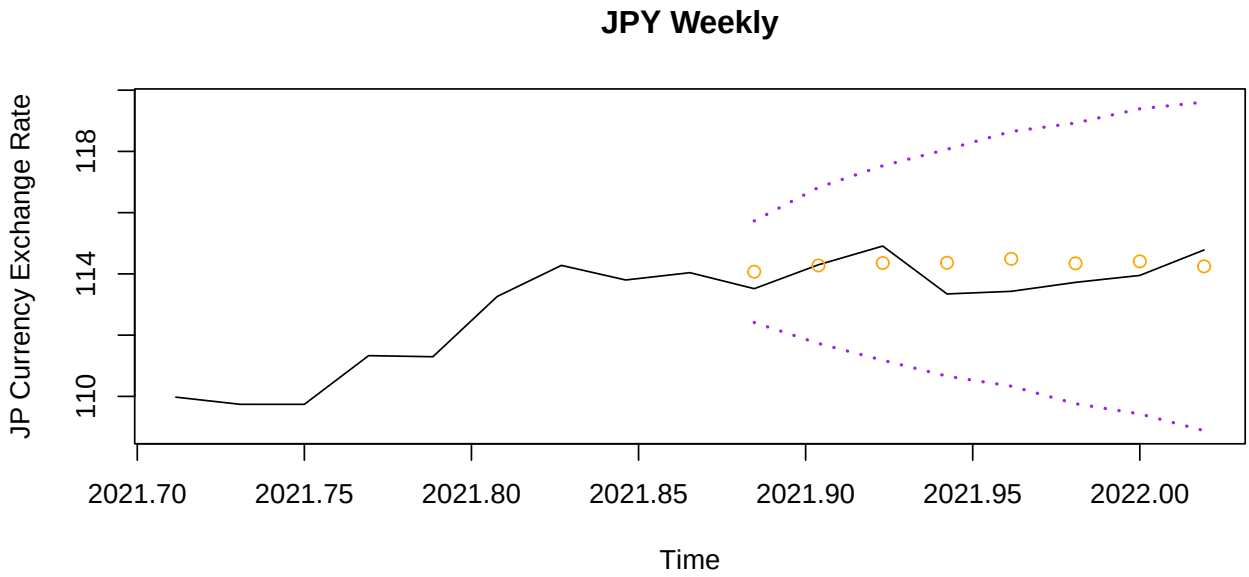
Response: Residual Analysis

For JP: When looking at the ACF and PACF plots, they do appear to be improved over what we saw earlier on. We don't see significant spikes like we did. Everything seems to fall w/in the blue dotted boundary lines. We also don't see a "wave-like" pattern that we saw before either. When looking at the histogram (and I also included a q-q plot as well), It does appear to be relatively normal, but the tails do start to deviate at the end. Maybe a transformation is needed? Finally, when looking at the hypothesis tests for serial correlation, the p-value of the Ljung-Box test is not significant... indicating that we don't have serial correlation present and uncorrelated residuals is plausible. I think this model fits reasonably well.

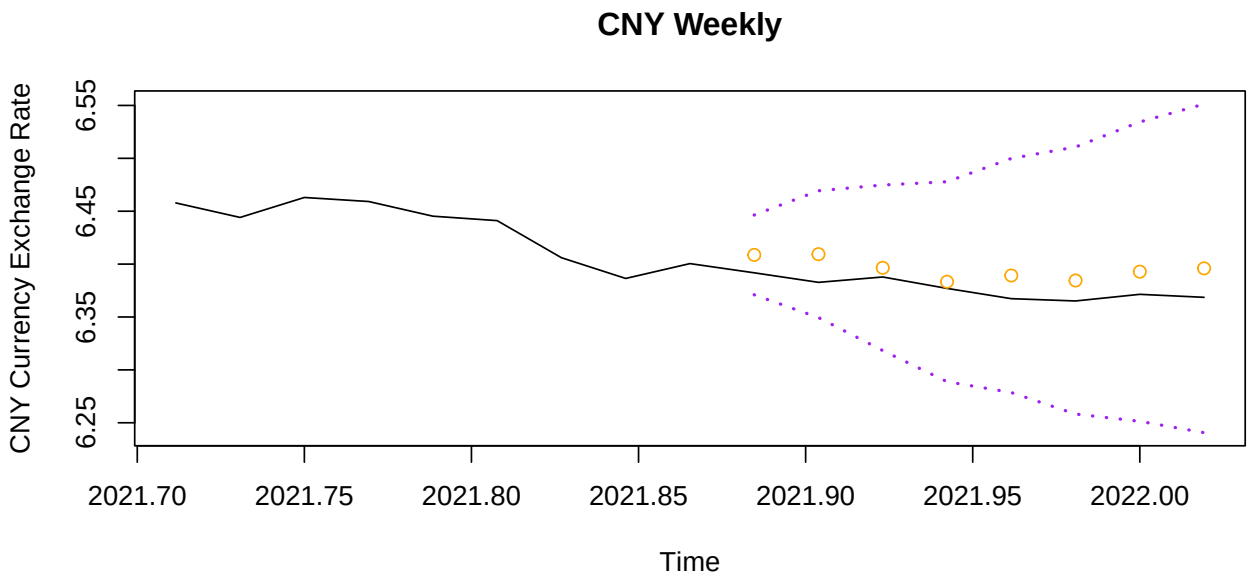
For CNY: When looking at the ACF and PACF plots, they do appear to be improved over what we saw earlier on. We do see a couple significant spikes, but not nearly as many as we saw earlier. Additionally, we also don't see a "wave-like" pattern that we saw before either. When looking at the histogram (and I also included a q-q plot as well), the tails do deviate off more than the JP model. I think transformation or additional improvement to the model is needed to help with this. Finally, when looking at the hypothesis tests for serial correlation, the p-value of the Ljung-Box test is significant... indicating that we may have serial correlation present. I think the model's fit could be improved here, but definitely better than what it was before.

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.

```
# JP first
# need to remake my og time series b/c I wrote over it earlier lol
weekly_jp.ts <- ts(weekly$JPY, start = 2010, freq = 52)
total_series_length <- length(weekly_jp.ts)
without_8 <- total_series_length-8
timevol <- time(weekly_jp.ts,0) # the 0 signifies the start of the week

# predict 8 weeks out
out_pred_jp <- as.vector(predict(final_model_jp,n.ahead=8))
out_pred_jp_w <- as.vector(predict(final_model_jp,n.ahead=8))
# make 90% interval
upper_bound <- out_pred_jp$pred+1.645*out_pred_jp$se
lower_bound <- out_pred_jp$pred-1.645*out_pred_jp$se
y_axis_min <- min(lower_bound)
y_axis_max <- max(upper_bound)

# plot interval
plot(timevol[(total_series_length-16):total_series_length],
     weekly_jp.ts[(total_series_length-16):total_series_length],type="l",
     ylim=c(y_axis_min,y_axis_max), xlab="Time",
     ylab="JP Currency Exchange Rate", main = "JPY Weekly")

# its spooky season soon... so orange and purple it will be
points(timevol[(without_8+1):total_series_length],out_pred_jp$pred,col="orange")
lines(timevol[(without_8+1):total_series_length],upper_bound,lty=3,lwd= 2, col="purple")
lines(timevol[(without_8+1):total_series_length],lower_bound,lty=3,lwd= 2, col="purple")
```




```
# now CNY
# need to remake my og time series b/c I wrote over it earlier lol
weekly_cny.ts <- ts(weekly$CNY, start = 2010, freq = 52)
total_series_length <- length(weekly_cny.ts)
without_8 <- total_series_length-8
timevol <- time(weekly_cny.ts,0) # the 0 signifies the start of the week

# predict 8 weeks out
out_pred_cny <- as.vector(predict(final_model_cny,n.ahead=8))
out_pred_cny_w <- as.vector(predict(final_model_cny,n.ahead=8))
# make 90% interval
upper_bound <- out_pred_cny$pred+1.645*out_pred_cny$se
lower_bound <- out_pred_cny$pred-1.645*out_pred_cny$se
y_axis_min <- min(lower_bound)
y_axis_max <- max(upper_bound)

# plot interval
plot(timevol[(total_series_length-16):total_series_length],
     weekly_cny.ts[(total_series_length-16):total_series_length],type="l",
     ylim=c(y_axis_min,y_axis_max), xlab="Time",
     ylab="CNY Currency Exchange Rate", main = "CNY Weekly")

# its spooky season soon... so orange and purple it will be
points(timevol[(without_8+1):total_series_length],out_pred_cny$pred,col="orange")
lines(timevol[(without_8+1):total_series_length],upper_bound,lty=3,lwd= 2, col="purple")
lines(timevol[(without_8+1):total_series_length],lower_bound,lty=3,lwd= 2, col="purple")
```



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.

```
# MAPE
## jp
mean(abs(out_pred_jp$pred - test_data_weekly_jp$JPY)/test_data_weekly_jp$JPY) *100
```

```
## [1] 0.5296186
```

```
##cny
mean(abs(out_pred_cny$pred - test_data_weekly_cny$CNY)/test_data_weekly_cny$CNY) *100
```

```
## [1] 0.2915171
```

```
# precision measure
## jp
pm_jp <- sum((test_data_weekly_jp$JPY -
out_pred_jp$pred)^2)/sum((test_data_weekly_jp$JPY-mean(test_data_weekly_jp$JPY))^2)

print(pm_jp)
```

```
## [1] 1.412865
```

```
## cny
pm_cny <- sum((test_data_weekly_cny$CNY -
out_pred_cny$pred)^2)/sum((test_data_weekly_cny$CNY - mean(test_data_weekly_cny$CNY))^2)
print(pm_cny)
```

```
## [1] 4.525448
```

Response: Prediction Accuracy **Prediction Bands**: All observations for JPY and CNY are within their respective 90% prediction bands.

MAPE: Additionally, both of their MAPEs are both under 1% meaning my average absolute difference between the predicted value I calculated and the actual value given is under 1%. JPY's is a little higher than CNY's but not by much. Regardless, because these values are low, it indicates that these models have good accuracy based off of the mean absolute prediction error.

PM: Recalled from lecture, this is the ratio b/w mean squared prediction error and the sum of square

differences (the response and mean of responses). The way we can interpret this is that it is a proportion of the variability in our prediction and the variability of new data we are looking at. The smaller the value, the better our prediction is! JPY's is pretty good coming in at 1.412, but CNY's is okay . . . coming in at 4.52.

Now, if we wanted to go one step further, by subtracting the PM value from 1, we'd get Rsquared.

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.

```
# jp
library(MuMIn) # this will get us AICc
training_data_monthly_jp <- monthly[1:142, c(1,3)]
test_data_monthly_jp <- monthly[143:144, c(1,3)]

#cny
training_data_monthly_cny <- monthly[1:142, c(1,2)]
test_data_monthly_cny <- monthly[143:144, c(1,2)]

### jp first
monthly_jp.ts <- ts(training_data_monthly_jp$JPY, start = 2010, freq = 12)
# matrix in r is row, column
max_order <- 8
aic_results_d1 <- matrix(0,9,9)

# only d = 1
for(p_val in 0:max_order){
   for(q_val in 0:max_order){
    arima_model <- arima(monthly_jp.ts,order = c(p_val,1,q_val), method='ML')
    aic_results_d1[p_val+1, q_val+1] <- AICc(arima_model)
   }
 }

# only d = 2
aic_results_d2 <- matrix(0,9,9)
for(p_val in 0:max_order){
   for(q_val in 0:max_order){
    arima_model <- arima(monthly_jp.ts,order = c(p_val,2,q_val), method='ML', optim.control = list(maxi
    aic_results_d2[p_val+1, q_val+1] <- AICc(arima_model)
   }
 }

print(aic_results_d1)
```

```
##           [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
##  [1,] 653.5126 655.3260 653.5927 655.3681 657.5162 656.7851 657.5167 659.2115
##  [2,] 655.2457 655.9477 655.4252 657.5183 657.2470 658.2537 659.5272 661.4648
##  [3,] 653.4438 655.5614 652.9955 654.4886 654.9929 657.1286 658.8694 660.7792
##  [4,] 655.5616 657.7117 654.3044 656.4035 658.5294 657.2872 655.5402 661.7949
```

```
##  [5,] 657.7069 657.9292 655.0391 658.5254 657.2067 659.5188 661.9285 663.9390
##  [6,] 657.8095 659.2677 656.8359 657.2145 659.5679 661.8630 657.8526 666.3611
##  [7,] 659.0033 661.2416 658.7439 659.4957 661.8824 658.1704 659.6065 663.8931
##  [8,] 661.2234 663.4676 660.4726 661.8078 664.1770 666.6112 663.8140 669.5948
##  [9,] 662.8608 664.8270 657.9733 660.1331 662.5469 661.0326 661.9314 666.7792
##           [,9]
##  [1,] 661.2660
##  [2,] 662.6427
##  [3,] 656.2701
##  [4,] 658.3082
##  [5,] 666.3721
##  [6,] 662.5632
##  [7,] 662.0165
##  [8,] 666.2193
##  [9,] 670.3262
```

```
print(aic_results_d2)
```

```
##           [,1]     [,2]     [,3]     [,4]     [,5]     [,6]     [,7]     [,8]
##  [1,] 739.5673 656.2345 658.0531 656.1877 657.9183 660.0936 659.5704 660.2253
##  [2,] 688.2152 657.9614 658.2202 657.9804 660.1011 659.8189 661.0384 662.2468
##  [3,] 681.2945 655.9728 658.3382 658.8878 660.6865 657.7177 659.9277 661.6265
##  [4,] 677.5679 658.1236 658.8321 660.8436 659.2525 658.0244 660.1231 662.2833
##  [5,] 679.1900 660.2931 660.5069 662.4864 657.7605 660.0000 662.3457 664.6688
##  [6,] 672.8260 660.5414 661.9671 659.6291 660.0180 662.3493 664.7376 664.2595
##  [7,] 672.3646 661.6949 663.9641 662.4478 662.3119 662.2942 663.7602 669.5988
##  [8,] 674.2636 663.9379 666.2180 658.5217 664.6801 667.1132 669.5834 665.2809
##  [9,] 676.5306 665.6766 667.7164 660.9124 667.1122 669.5834 668.6412 671.1935
##           [,9]
##  [1,] 661.9183
##  [2,] 664.2116
##  [3,] 663.6605
##  [4,] 664.6690
##  [5,] 667.1078
##  [6,] 666.9237
##  [7,] 672.0835
##  [8,] 667.8237
##  [9,] 676.0585
```

```
# using example from lecture here
aic_values <-  as.vector(aic_results_d1)
p_0_8 <- rep(c(1:9),9)
q_0_8 <- rep(c(1:9),each=9)
aic_matrix <- which(aic_values == min(aic_values))
final_p <- p_0_8[aic_matrix] - 1
final_q <- q_0_8[aic_matrix] - 1
final_model_jp <- arima(monthly_jp.ts,order = c(final_p,1,final_q), method='ML')

final_model_jp
```

```
##
## Call:
## arima(x = monthly_jp.ts, order = c(final_p, 1, final_q), 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_model_jp)
```

```
##           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_model_jp$coef
```

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

```
final_model_jp$sigma2
```

```
## [1] 5.424235
```

```
### CNY next
monthly_cny.ts <- ts(training_data_monthly_cny$CNY, start = 2010, freq = 12)
# matrix in r is row, column
max_order <- 8
aic_results_d1 <- matrix(0,9,9)

# only d = 1
for(p_val in 0:max_order){
   for(q_val in 0:max_order){
    arima_model <- arima(monthly_cny.ts,order = c(p_val,1,q_val), method='ML')
    aic_results_d1[p_val+1, q_val+1] <- AICc(arima_model)
   }
 }


# only d = 2
aic_results_d2 <- matrix(0,9,9)
for(p_val in 0:max_order){
   for(q_val in 0:max_order){
    arima_model <- arima(monthly_cny.ts,order = c(p_val,2,q_val), method='ML')
    aic_results_d2[p_val+1, q_val+1] <- AICc(arima_model)
   }
 }
```

```
print(aic_results_d1)
```

```
##            [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
##  [1,] -352.8851 -364.9531 -362.8679 -362.7795 -361.4121 -360.1159 -358.5801
##  [2,] -364.3905 -362.8774 -360.9151 -364.0733 -362.4056 -359.8869 -357.5251
##  [3,] -362.5142 -366.4897 -364.2215 -362.0462 -359.6763 -359.6260 -357.2385
##  [4,] -361.2760 -364.3853 -362.2164 -366.1774 -367.3933 -367.2314 -360.7733
##  [5,] -360.0396 -362.0474 -361.5871 -369.7656 -359.6295 -357.4066 -358.6774
##  [6,] -357.9200 -360.2358 -358.4262 -364.3060 -358.7658 -362.1443 -359.2350
##  [7,] -356.9209 -355.1366 -356.2342 -360.8027 -358.5780 -358.7949 -356.3155
##  [8,] -355.1045 -353.0405 -354.9154 -366.6154 -364.6739 -358.1658 -362.0579
##  [9,] -354.5736 -356.2998 -352.7868 -364.9570 -362.9550 -361.4509 -359.6436
##            [,8]      [,9]
##  [1,] -356.3342 -361.1915
##  [2,] -355.6863 -358.8998
##  [3,] -355.4376 -364.8533
##  [4,] -358.4661 -363.7871
##  [5,] -363.7340 -361.3494
##  [6,] -356.8267 -359.3214
##  [7,] -353.8762 -362.7272
##  [8,] -360.8337 -360.9808
##  [9,] -357.9087 -355.9526
```

```
print(aic_results_d2)
```

```
##            [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
##  [1,] -303.4844 -343.4744 -355.0399 -352.9201 -352.9871 -351.4465 -350.3344
##  [2,] -316.5596 -354.6208 -352.9203 -355.7916 -354.1308 -352.4955 -349.9829
##  [3,] -330.4297 -352.6515 -356.6015 -354.3658 -352.1529 -350.2614 -347.9901
##  [4,] -330.3686 -351.5216 -354.3586 -352.1517 -349.8994 -347.9827 -357.2062
##  [5,] -333.7114 -350.1197 -352.4257 -349.8916 -351.0380 -349.4648 -355.8785
##  [6,] -332.1595 -348.0049 -350.2786 -348.2623 -357.7906 -347.1155 -348.4240
##  [7,] -331.4575 -346.8125 -347.8466 -345.8456 -347.8669 -345.4743 -354.7311
##  [8,] -329.5235 -344.8665 -346.0380 -344.1210 -356.3343 -354.2607 -352.2626
##  [9,] -341.3888 -344.9853 -345.9163 -343.4750 -354.5259 -352.4940 -351.1581
##            [,8]      [,9]
##  [1,] -348.6251 -346.3398
##  [2,] -347.6989 -345.4675
##  [3,] -345.5632 -343.4041
##  [4,] -357.3742 -346.0951
##  [5,] -354.9820 -355.5539
##  [6,] -352.9856 -352.3142
##  [7,] -352.2616 -350.0030
##  [8,] -349.8190 -343.9828
##  [9,] -349.2625 -346.7742
```

```
# using example from lecture here
aic_values <-  as.vector(aic_results_d1)
p_0_8 <- rep(c(1:9),9)
q_0_8 <- rep(c(1:9),each=9)
aic_matrix <- which(aic_values == min(aic_values))
final_p <- p_0_8[aic_matrix] - 1
final_q <- q_0_8[aic_matrix] - 1
final_model_cny <- arima(monthly_cny.ts,order = c(final_p,1,final_q), method='ML', optim.control = list
```

```
final_model_cny
```

```
##
## Call:
## arima(x = monthly_cny.ts, order = c(final_p, 1, final_q), method = "ML", optim.control = list(maxit =
##
## 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_model_cny)
```

```
##           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        5    -none- call
## series      1    -none- character
## code        1    -none- numeric
## n.cond      1    -none- numeric
## nobs        1    -none- numeric
## model      10    -none- list
```

```
final_model_cny$coef
```

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

```
final_model_cny$sigma2
```

```
## [1] 0.003588273
```

```
## 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 (from professor)
tv_jpy_monthly <-as.numeric(final_model_jp$coef)/as.numeric(sqrt(diag(final_model_jp$var.coef)))
tv_cny_monthly <-as.numeric(final_model_cny$coef)/as.numeric(sqrt(diag(final_model_cny$var.coef)))

## Apply the pvalue.coef function

library(lmtest)
coeftest(final_model_jp)
```

```
##
```

```
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
## ar1  0.168054   0.065203   2.5774 0.0099552 **
## ar2 -0.834766   0.050327 -16.5867 < 2.2e-16 ***
## ma1 -0.199817   0.040933  -4.8816 1.052e-06 ***
## ma2  0.999958   0.297369   3.3627 0.0007719 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coeftest(final_model_cny)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
## ar1  0.371372   0.082790   4.4857 7.266e-06 ***
## ar2 -0.069097   0.071675  -0.9640 0.3350346
## ar3 -0.737362   0.065374 -11.2791 < 2.2e-16 ***
## ar4  0.294609   0.081336   3.6221 0.0002922 ***
## ma1 -0.053495   0.046017  -1.1625 0.2450232
## ma2 -0.053493   0.046746  -1.1443 0.2524896
## ma3  0.999996   0.043215  23.1398 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Response: Analysis of the ARIMA Fit for the Weekly and Monthly Data **Note** I am only analyzing the Monthly data for this one since that is what the question asked for.

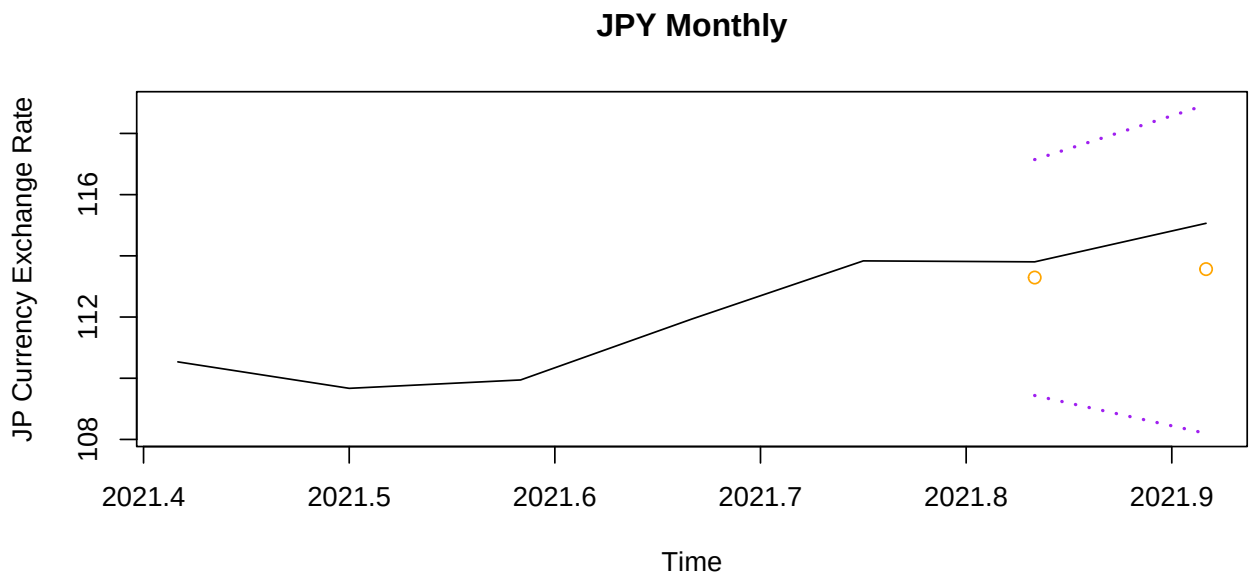
For JPY: For this process, I iterated through 0-8 for both the p value and q value of ARIMA, along with 1 and 2 for the d value. For each iteration, I examined the AICc value to see if it increased or decreased. Unlike last time, I didn't have any issues with d=2. I noticed that every combination that included d=2 had a higher AIC value than its counterpart using d=1. Though in either case, most values fell between 600 and 700. Comparing the two matrices, d=1 seemed to be the way to go. After numerous iterations and without looking at statistical significance at this point, it looked like arima(2,1,2) produced the best results in terms of AICc (652.99) for JPY. Coefficients are shown above along with sigma2. All are statistically significant.

For CNY: For this process, I iterated through 0-8 for both the p value and q value of ARIMA, along with 1 and 2 for the d value. For each iteration, I examined the AICc value to see if it increased or decreased. Just like last time, for both d=1 and d=2, the AICc values were all negative. Though in this version, the values fell w/in the -300 to -400 range. After numerous iterations and without looking at statistical inference at this point, it looked like arima(4,1,3) produced the best results in terms of AICc (-369.76) for CNY. Coefficients are shown above along with sigma2. AR(1), AR(3), AR(4), and MA(3) were all statistically significant coefficients.

Response: Monthly vs Weekly Data Right off the bat, we can see that for both JPY and CNY, the monthly data model has less coefficients than the weekly data model. For the monthly JPY model, it uses ARIMA(2,1,2)... whereas in its weekly model, it used ARIMA(4,1,5). For CNY, its monthly model used ARIMA(4,1,3)... whereas in its weekly model, it used ARIMA(8,1,7). For JPY monthly, all coefficients were statistically significant compared to weekly. For CNY, about half of the coefficients for monthly and for weekly were significant.

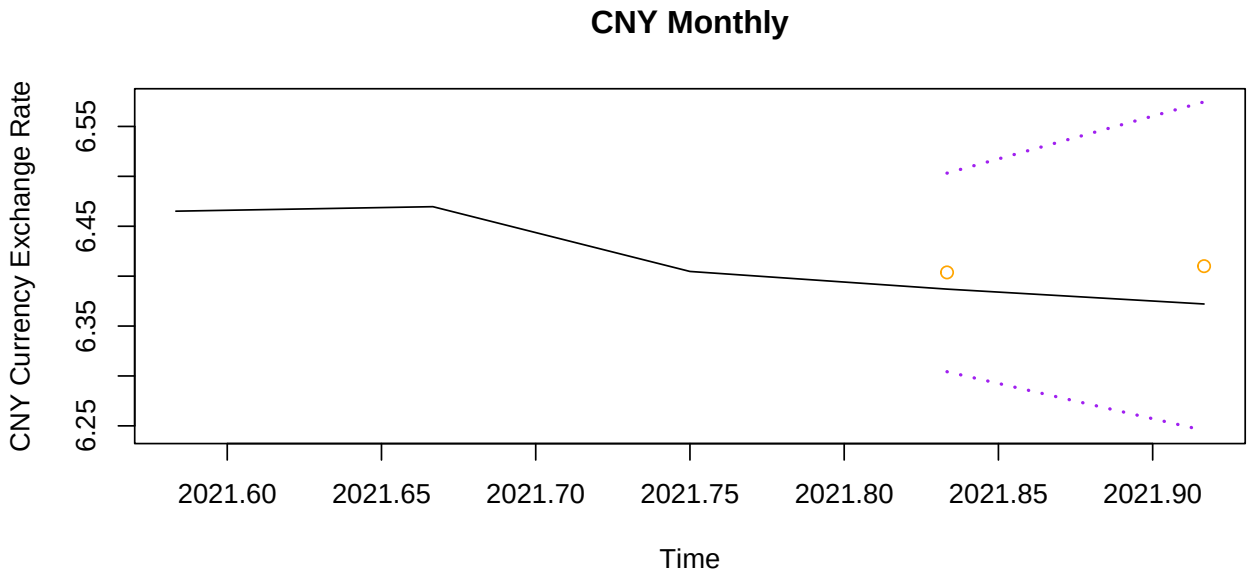
3b. For each currency exchange, apply the model identified in (3a) and forecast the last two months of data. Plot the predicted data to compare the predicted values to the actual observed ones. Include 90% confidence intervals for the forecasts in the corresponding plots.

```r
# JP first
#final_model_jp_bu <- final_model_jp
#final_model_jp <- arima(monthly_jp.ts, order = c(2,1,2), method='ML')
# need to remake my og time series b/c I wrote over it earlier lol
monthly_jp.ts <- ts(monthly$JPY, start = 2010, freq = 12)
total_series_length <- length(monthly_jp.ts)
without_2 <- total_series_length-2
timevol <- time(monthly_jp.ts,0) # the 0 signifies the start of the week

# predict 8 weeks out
out_pred_jp <- as.vector(predict(final_model_jp,n.ahead=2))
out_pred_jp_m <- as.vector(predict(final_model_jp,n.ahead=2))
# make 90% interval
upper_bound <- out_pred_jp$pred+1.645*out_pred_jp$se
lower_bound <- out_pred_jp$pred-1.645*out_pred_jp$se
y_axis_min <- min(lower_bound)
y_axis_max <- max(upper_bound)

# plot interval
plot(timevol[(total_series_length-6):total_series_length],
     monthly_jp.ts[(total_series_length-6):total_series_length],type="l",
     ylim=c(y_axis_min,y_axis_max), xlab="Time",
     ylab="JP Currency Exchange Rate", main = "JPY Monthly")

# its spooky season soon... so orange and purple it will be
points(timevol[(without_2+1):total_series_length],out_pred_jp$pred,col="orange")
lines(timevol[(without_2+1):total_series_length],upper_bound,lty=3,lwd= 2, col="purple")
lines(timevol[(without_2+1):total_series_length],lower_bound,lty=3,lwd= 2, col="purple")
```




```r
# now CNY
# need to remake my og time series b/c I wrote over it earlier lol
monthly_cny.ts <- ts(monthly$CNY, start = 2010, freq = 12)
total_series_length <- length(monthly_cny.ts)
without_2 <- total_series_length-2
timevol <- time(monthly_cny.ts,0) # the 0 signifies the start of the week
```

```
# predict 8 weeks out
out_pred_cny <- as.vector(predict(final_model_cny,n.ahead=2))
out_pred_cny_m <- as.vector(predict(final_model_cny,n.ahead=2))
# make 90% interval
upper_bound <- out_pred_cny$pred+1.645*out_pred_cny$se
lower_bound <- out_pred_cny$pred-1.645*out_pred_cny$se
y_axis_min <- min(lower_bound)
y_axis_max <- max(upper_bound)

# plot interval
plot(timevol[(total_series_length-4):total_series_length],
     monthly_cny.ts[(total_series_length-4):total_series_length],type="l",
     ylim=c(y_axis_min,y_axis_max), xlab="Time",
     ylab="CNY Currency Exchange Rate", main = "CNY Monthly")

# its spooky season soon... so orange and purple it will be
points(timevol[(without_2+1):total_series_length],out_pred_cny$pred,col="orange")
lines(timevol[(without_2+1):total_series_length],upper_bound,lty=3,lwd= 2, col="purple")
lines(timevol[(without_2+1):total_series_length],lower_bound,lty=3,lwd= 2, col="purple")
```



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.

```
# MAPE
## jp
mean(abs(out_pred_jp$pred - test_data_monthly_jp$JPY)/test_data_monthly_jp$JPY) *100
```

```
## [1] 0.8758606
```

```
##cny
mean(abs(out_pred_cny$pred - test_data_monthly_cny$CNY)/test_data_monthly_cny$CNY) *100
```

```
## [1] 0.4284442
```

```
# precision measure
## jp
```

```
pm_jp <- sum((test_data_monthly_jp$JPY -
out_pred_jp$pred)^2)/sum((test_data_monthly_jp$JPY-mean(test_data_monthly_jp$JPY))^2)

print(pm_jp)
```

```
## [1] 3.159275
```

```
## cny
pm_cny <- sum((test_data_monthly_cny$CNY -
out_pred_cny$pred)^2)/sum((test_data_monthly_cny$CNY - mean(test_data_monthly_cny$CNY))^2)
print(pm_cny)
```

```
## [1] 15.4665
```

Response: Predictions

MAPE: For mean absolute percent error for JPY, the value was 0.875%. CNY's was lower at 0.428%, meaning CNY's prediction was slightly more accurate than JPY's. However, because both have super low values in general, both JPY's and CNY's are pretty accurate when looking at MAPE. (With MAPE, the lower the better!) In comparison to the weekly results, JPY's and CNY's monthly prediction did slightly worse than weekly (0.529% and 0.292% respectively). But because all of these values are so low, regardless if they were monthly or weekly, I'd say both monthly and weekly predictions did well.

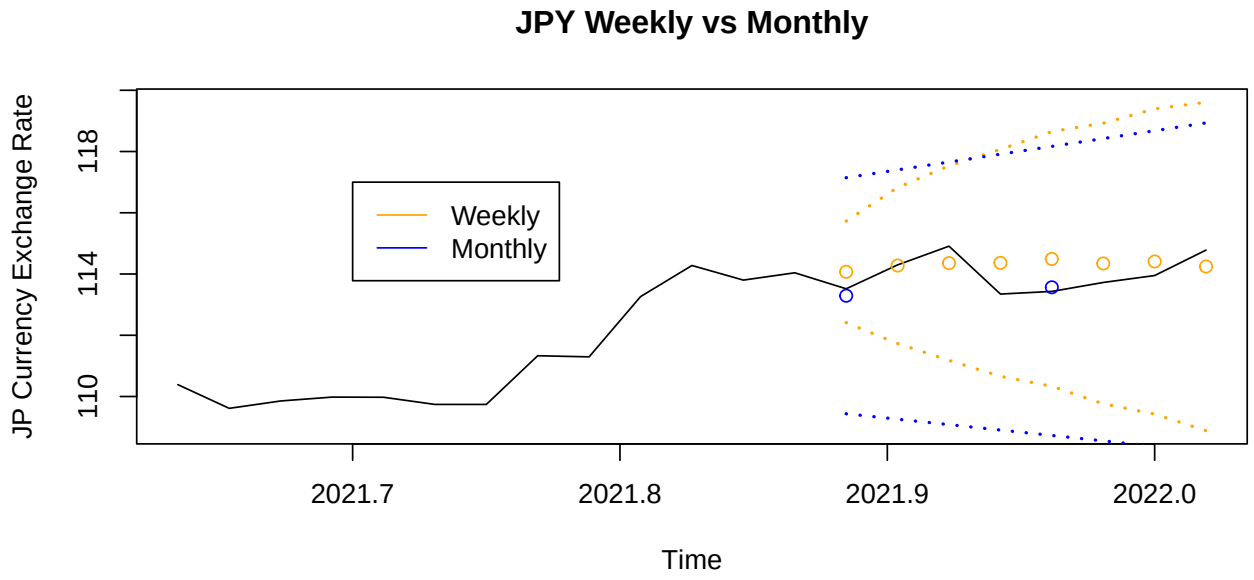
PM: Recalled from lecture, this is the ratio b/w mean squared prediction error and the sum of square differences (the response and mean of responses). The way we can interpret this is that it is a proportion of the variability in our prediction and the variability of new data we are looking at. The smaller the value, the better our prediction is! JPY's is okay coming in at 3.16, but CNY's is not so good. . . coming in at 15.47..meaning it didn't do such a great job here. The weekly values did better in comparison.

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?

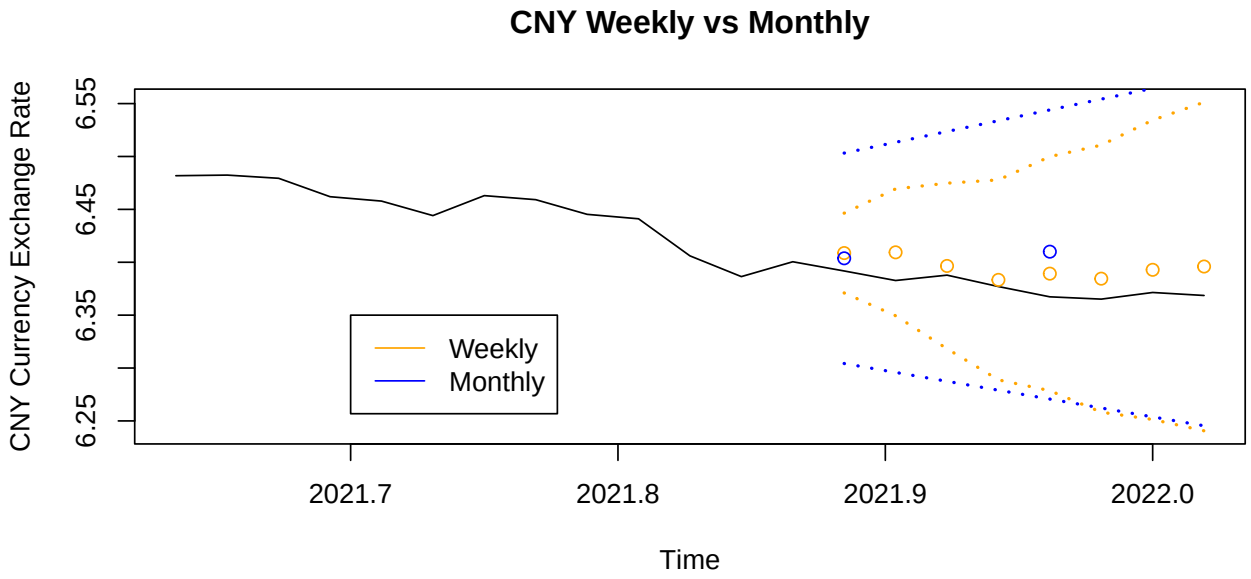
```
#JP
weekly_jp.ts <- ts(weekly$JPY, start = 2010, freq = 52)
total_series_length <- length(weekly_jp.ts)
timevol <- time(weekly_jp.ts,0) # the 0 signifies the start of the week

monthly_jp.ts <- ts(monthly$JPY, start = 2010, freq = 12)
timevol2 <- time(monthly_jp.ts,0)
total_series_length2 <- length(monthly_jp.ts)
# make 90% interval
upper_bound_w <- out_pred_jp_w$pred+1.645*out_pred_jp_w$se
lower_bound_w <- out_pred_jp_w$pred-1.645*out_pred_jp_w$se
y_axis_min_w <- min(lower_bound_w)
y_axis_max_w <- max(upper_bound_w)
upper_bound_m <- out_pred_jp_m$pred+1.645*out_pred_jp_m$se
lower_bound_m <- out_pred_jp_m$pred-1.645*out_pred_jp_m$se

# plot interval
plot(timevol[(total_series_length-20):total_series_length],
     weekly_jp.ts[(total_series_length-20):total_series_length],type="l",
     ylim=c(y_axis_min_w,y_axis_max_w), xlab="Time",
     ylab="JP Currency Exchange Rate", main = "JPY Weekly vs Monthly")
```

```
points(timevol[(without_8+1):total_series_length],out_pred_jp_w$pred,col="orange")
lines(timevol[(without_8+1):total_series_length],upper_bound_w,lty=3,lwd= 2, col="orange")
lines(timevol[(without_8+1):total_series_length],lower_bound_w,lty=3,lwd= 2, col="orange")
points(timevol[c(619,623)],out_pred_jp_m$pred,col="blue")
lines(timevol[c(619,626)],upper_bound_m,lty=3,lwd= 2, col="blue")
lines(timevol[c(619,626)],lower_bound_m,lty=3,lwd= 2, col="blue")
legend(x=2021.7,y=117,legend=c("Weekly","Monthly"),lty = 1, col=c("orange","blue"))
```




```
#CNY
weekly_cny.ts <- ts(weekly$CNY, start = 2010, freq = 52)
total_series_length <- length(weekly_cny.ts)
timevol <- time(weekly_cny.ts,0) # the 0 signifies the start of the week

monthly_cny.ts <- ts(monthly$CNY, start = 2010, freq = 12)
timevol2 <- time(monthly_cny.ts,0)
total_series_length2 <- length(monthly_cny.ts)
# make 90% interval
upper_bound_w <- out_pred_cny_w$pred+1.645*out_pred_cny_w$se
lower_bound_w <- out_pred_cny_w$pred-1.645*out_pred_cny_w$se
y_axis_min_w <- min(lower_bound_w)
y_axis_max_w <- max(upper_bound_w)
upper_bound_m <- out_pred_cny_m$pred+1.645*out_pred_cny_m$se
lower_bound_m <- out_pred_cny_m$pred-1.645*out_pred_cny_m$se

# plot interval
plot(timevol[(total_series_length-20):total_series_length],
     weekly_cny.ts[(total_series_length-20):total_series_length],type="l",
     ylim=c(y_axis_min_w,y_axis_max_w), xlab="Time",
     ylab="CNY Currency Exchange Rate", main = "CNY Weekly vs Monthly")


points(timevol[(without_8+1):total_series_length],out_pred_cny_w$pred,col="orange")
lines(timevol[(without_8+1):total_series_length],upper_bound_w,lty=3,lwd= 2, col="orange")
lines(timevol[(without_8+1):total_series_length],lower_bound_w,lty=3,lwd= 2, col="orange")
```

```
points(timevol[c(619,623)],out_pred_cny_m$pred,col="blue")
lines(timevol[c(619,626)],upper_bound_m,lty=3,lwd= 2, col="blue")
lines(timevol[c(619,626)],lower_bound_m,lty=3,lwd= 2, col="blue")
legend(x=2021.7,y=6.35,legend=c("Weekly","Monthly"),lty = 1, col=c("orange","blue"))
```



Response: Prediction Comparison **JPY**: Both the monthly (in blue) and weekly (in orange) have a slight upward increase. The monthly 90% confidence interval is wider than the weekly interval on the lower portion. On the upper portion, they seem to overlap.

CNY: The monthly values (in blue) seems to have an upward "trend" whereas the weekly values (in orange) seems to have a slightly downward "trend" then a little upward trend at the very end. The monthly 90% confidence interval is wider than the weekly interval.

Overall: Weekly provides a more granular view in regards to prediction compared to monthly. There seems to be a time and place for weekly data. For example, if you were trying to project sales and only relied on monthly prediction, you may perceive an upward trend. But in reality, you may miss a decreasing trend a long the way and could be too late to fix it . Whereas with the weekly trend, you may be able to catch that decrease and a) know that its coming and/or b) try to correct if necessary. Or, as another example, if you need to determine demand/product stock needed for your store. Probably better to know and predict per week, or risk running out by the end of the month. However, that goes to say that there is also a place for monthly data. One example could be you run a business and need to budget for business costs such as software and payroll. It seems silly to try to track this on a daily level since some days may have no costs at all and some may have a whole lot. The same could be said for weekly too. . . maybe your company pays the same 2 weeks every month. If you do monthly, you see everything aggregated all at once and can see how costs increase/decrease over time easily.

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 Overall, I think ARIMA was effective for this data set. The only thing I

think that would have been needed is some kind of transformation on the response variable for CNY. I think that would have helped improve the model. I'd use ARIMA when you need to address non-stationarity. We saw from the last homework that you can resolve that in multiple steps. But with ARIMA, you can resolve it all in one step. I'd recommend using it for supply chains (ex. supply/demand forecasting) or other financial based data sets like the one we just worked with (ex. maybe stock market data). I'd recommend against using it if you already have stationary data since then you could just use and AR, MA, or ARMA model instead. I'd also make sure you aren't already working with differenced data, because it's easy to difference it again using ARIMA since you can just type in a d value easily. Also when using "ML" in the model, be cognizant of the fact that not all p,d,q values will work nicely with each other. You may encounter some combinations where the optimization problem won't be solved under the hood.