Macro Homework

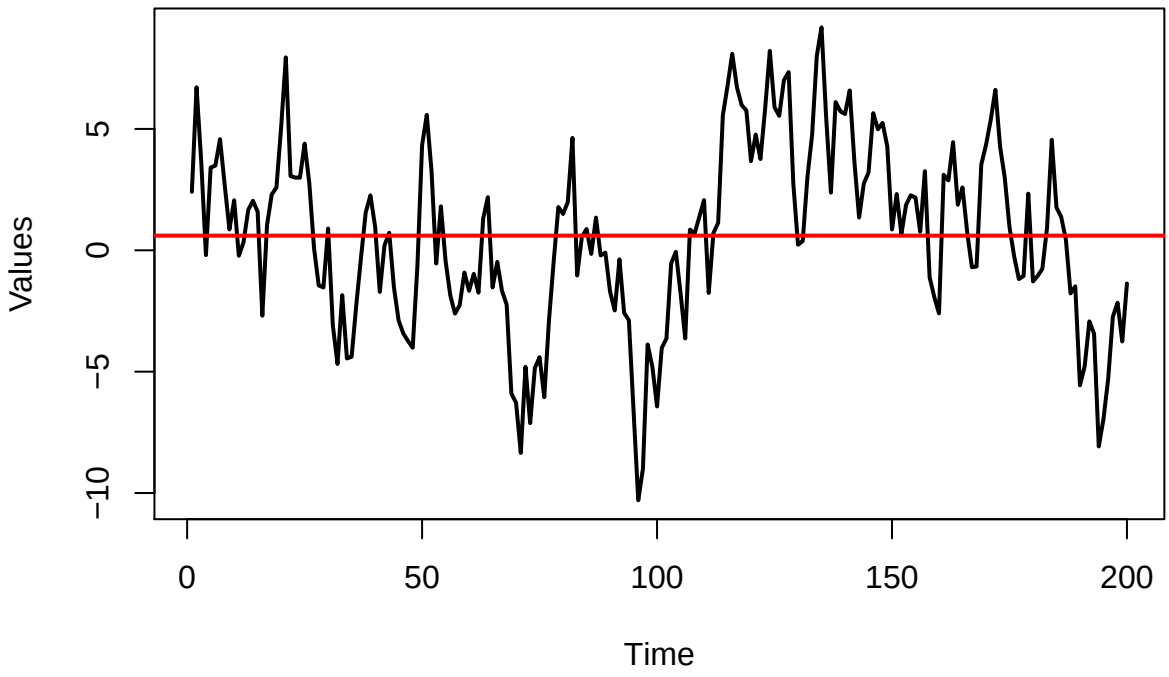
David Contento

November 26, 2018

Question 1

Part A) Plotting data and determining whether it is stationary

Figure 1



From looking at the plot we can tell that it is likely stationary because there are no upward or downward trends and no indication of seasonality. We went on to confirm whether the data was stationary byh looking at whether the means and variances were constant over time, looking at the lagged differences of the data, and by performing a augmented Dicky-Fuller stationary test.

```
#Determining whether the data is stationary

##looking at means and variances
x1=data[1:100,]
x2=data[101:200,]
mean(x1)
```

```
## [1] -0.5915845
```

```r
mean(x2)
```

```
## [1] 1.799254
```

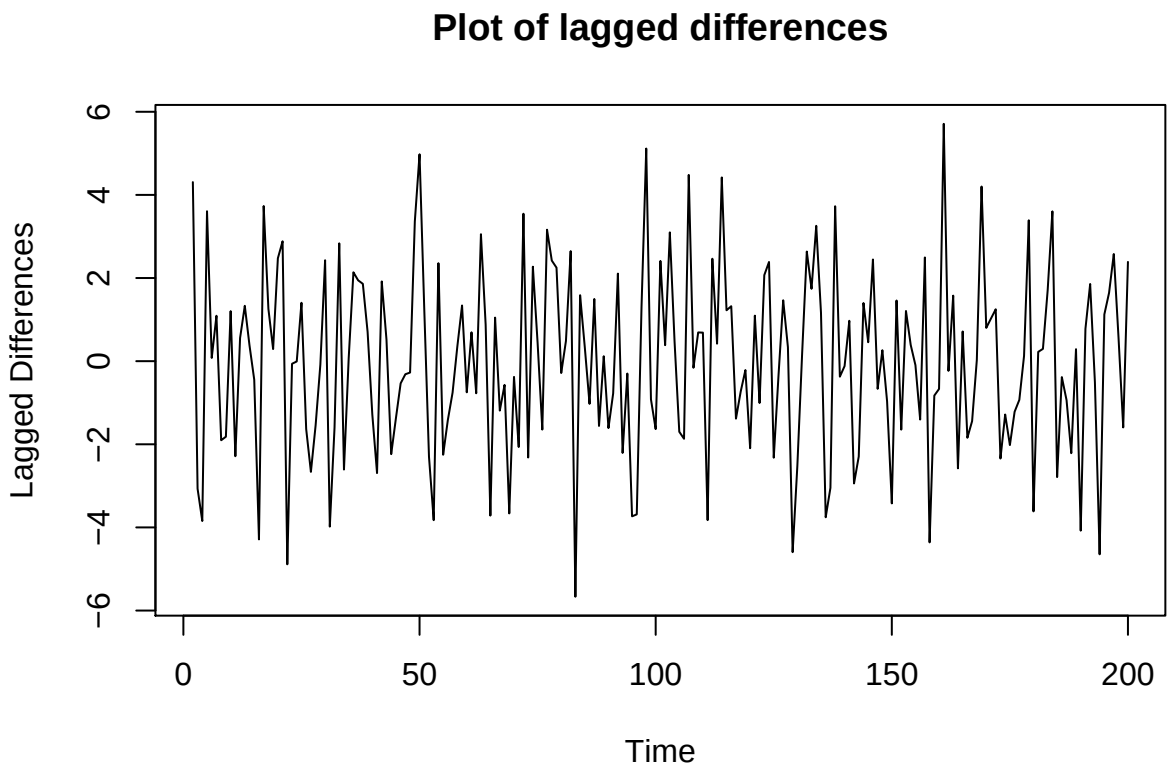
```r
var(x1)
```

```
## [1] 11.99315
```

```r
var(x2)
```

```
## [1] 13.39794
```

```r
#looking at lagged differences
dif = diff(data)
plot(dif, main="Plot of lagged differences", ylab="Lagged Differences")
```




```r
#performing augmented dicky-fuller stationary test
adf.test(data,alternative = "stationary",k=0)
```

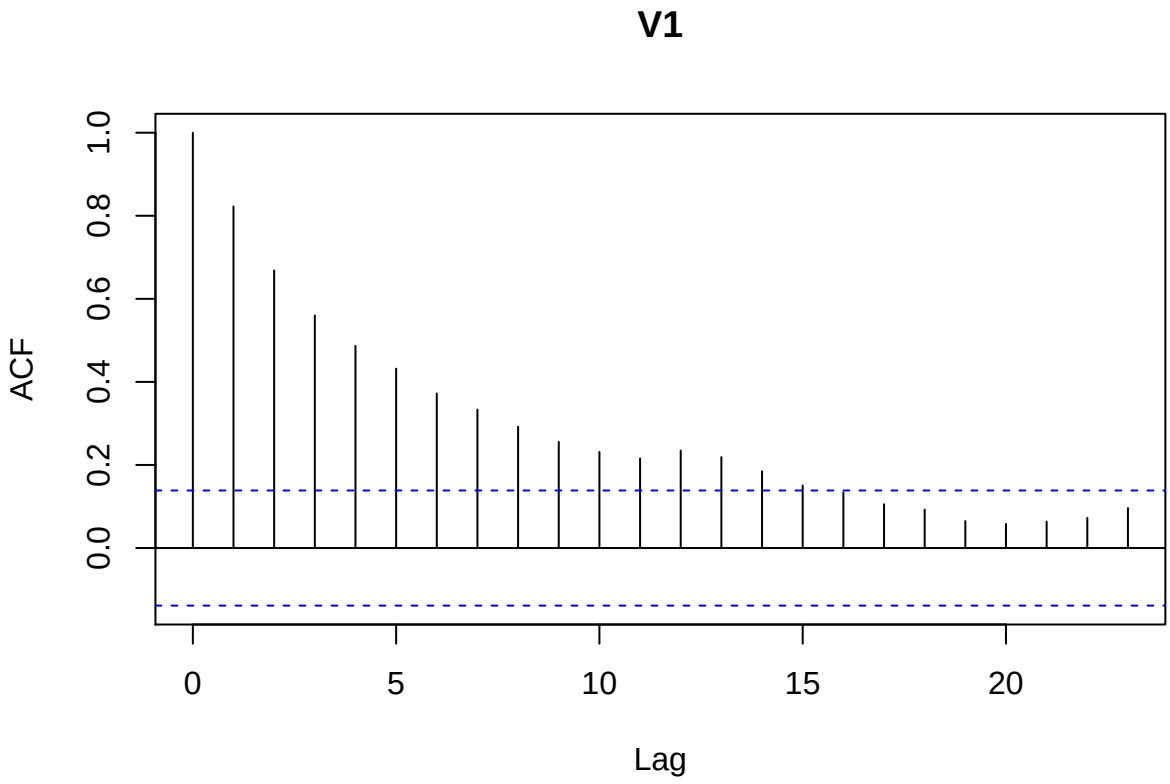
```
## Warning in adf.test(data, alternative = "stationary", k = 0): p-value
## smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  data
## Dickey-Fuller = -4.358, Lag order = 0, p-value = 0.01
## alternative hypothesis: stationary
```

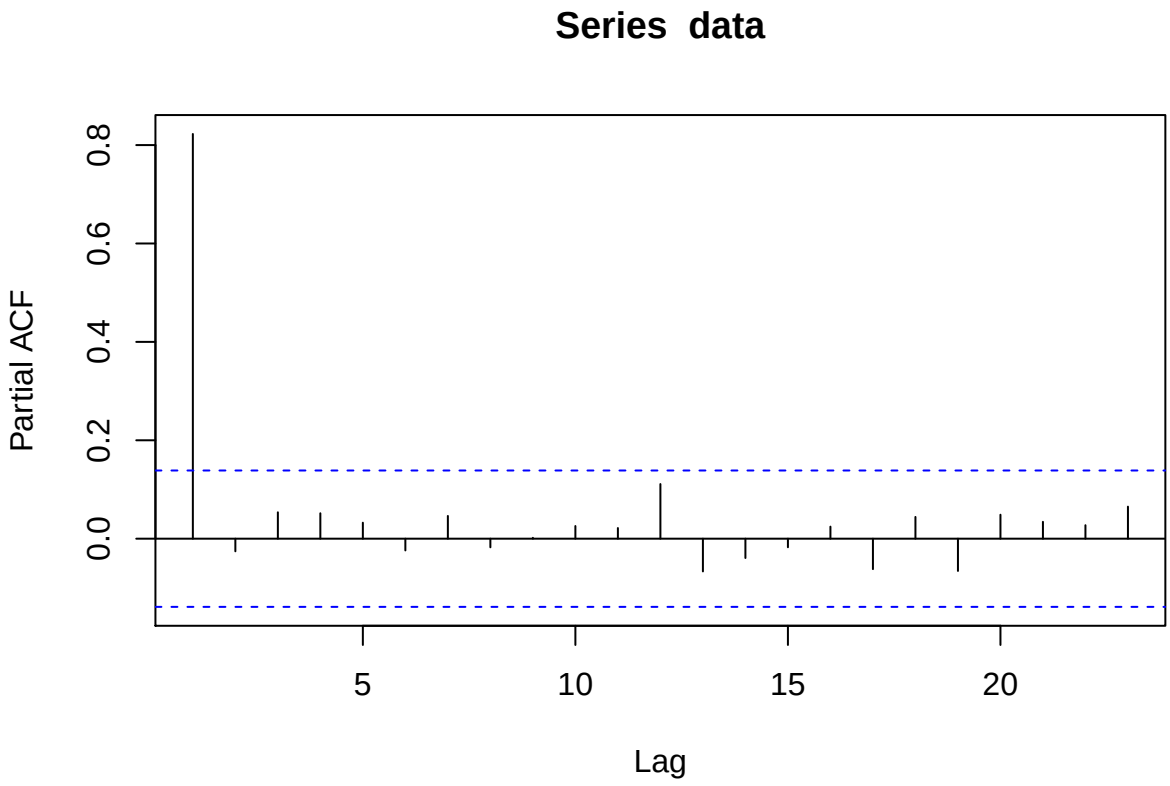
```
#reject the null, therefore stationarity
```

From the output above you can see that the mean and variances of the data remained fairly constant over time. The mean for the first half of the data and the second half differed only by 1.2 and the variances differed by less than 1.5 . From the graph above you can also see that the lagged differences remained fairly constant over time and close to zero. This suggests that the data is stationary. Lastly, you can see from the output above that the augmented Dicky-Fuller test also determined that the data was stationary at the 1% critical level. As a result we believe it is safe to assume the data is stationary.

```
acf(data)
```



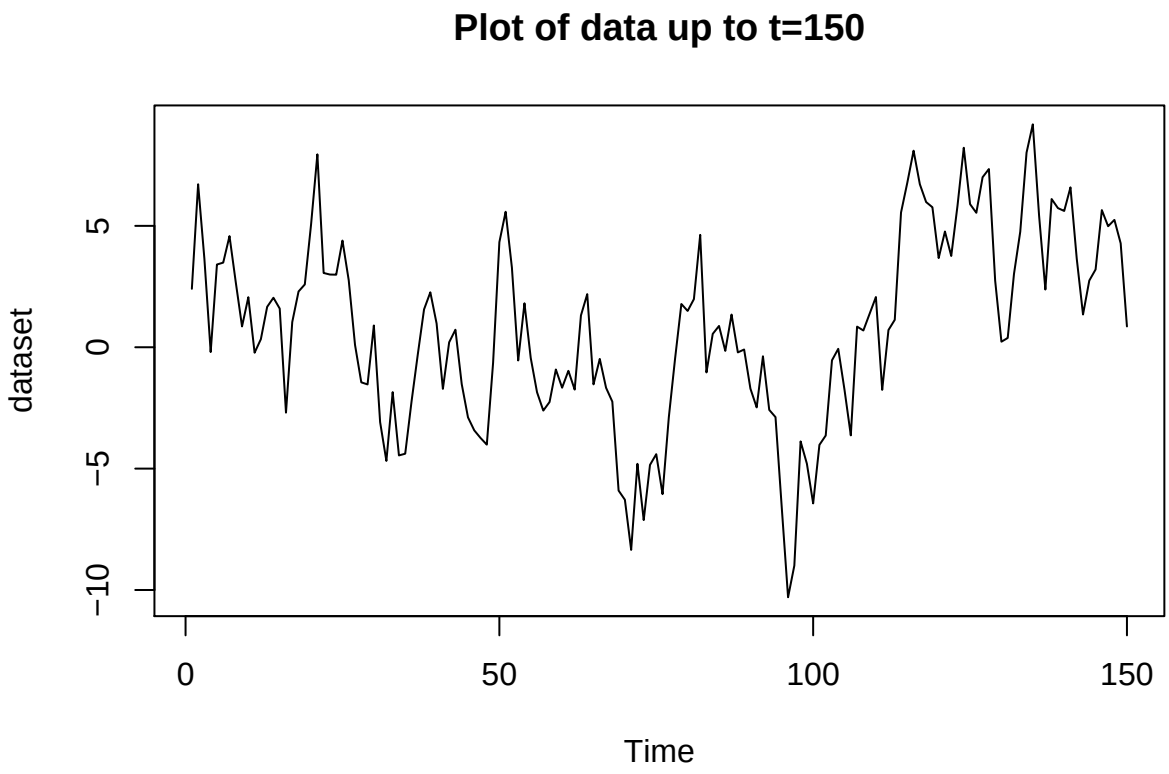
```r
pacf(data)
```



We also looked at the ACF and PCF plots of the data to determine whether the data is stationary. The tapering ACF values as we reach higher order lags, and the consistently low PCF values both indicate and confirm our assumption that the data is stationary.

Part B) Estimating AR(1) Model

```r
#Subsetting data for modeling
data_set=data[1:150,]

dataset=ts(data_set)
plot(dataset,type="l", main="Plot of data up to t=150")
```




```r
#Generating AR(1) model
ar=arima(dataset,order=c(1,0,0))
print(ar)
```

```
##
## Call:
## arima(x = dataset, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.8293     0.8210
## s.e.  0.0442     0.9953
##
## sigma^2 estimated as 4.608:  log likelihood = -328,  aic = 662.01
```

In order to generate our AR(1) model we used the arima function, which uses MLE to find the optimal parameters. From the output above you can see that the estimate for the AR(1) beta coefficient is .8293 with a standard error of .0442 . We can also see that the parameter estimate for the intercept coefficient is .8210 with a standard error of .9953. Below we tested whether these coefficients where significant using their estimates, standard errors, and a student t-distribution.

```r
#testing significance of AR(1) Coefficients

#AR(1) Beta coefficient Significance
t=0.8293/0.0442
2*pt(-abs(t),df=150) #this is significant
```

```
## [1] 3.47698e-41
```

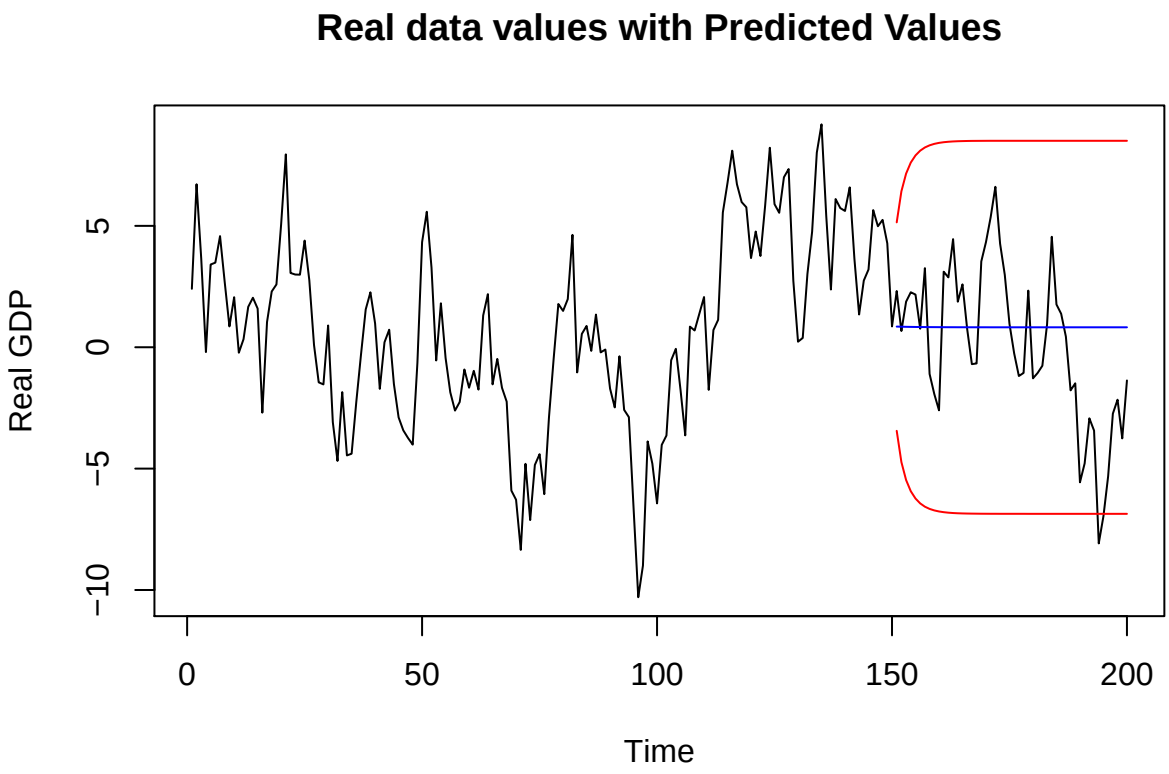
```
#intercept signifcance
t=0.8211/0.9953
2*pt(-abs(t),df=150) #intercept is not significant
```

```
## [1] 0.4106944
```

Above you'll find the p-value of our AR(1) beta coefficient and the intercept respectively. When testing the significance of each coefficient we first created a t-statistic using the coefficients estimate and standard error. We then used the pt function in order to determine whether the t-statistic was significant when compared to the t-critical from a student t-distribution with 150 degrees of freedom. We determined that the AR(1) beta coefficient was significant with a p.value of 3.48e-41. The intercept coefficient was not significant at the 5% critical level since it had a p.value of .4107 .

Part C) Creating a one period forecast

```
#fitting AR(1) model to the data up to t=150
arfit1 = arima(dataset,order=c(1,0,0))

#create a one period forecast for the remaing 50 observations
pred1 = predict(arfit1,n.ahead = 50)

#plotting original data with predicted values
plot(data, main="Real data values with Predicted Values", ylab="Real GDP")
lines(pred1$pred,col="blue")
lines(pred1$pred+2*pred1$se,col="red")
lines(pred1$pred-2*pred1$se,col="red")
```



The plot above shows the real values of our data along with our predicted values. The predicted values is the blue line that extends right from t=150. the two red lines are the 95% confidence interval for our predictions.

```
#calculating the root mean squared error
obs1=ts(data[151:200,])
pred =as.numeric(pred1$pred)
obs=as.numeric(obs1)

#calculating root mean squared error using function (method 1)
rmse(pred,obs)
```

```
## [1] 3.272672
```

```
#calculating root mean squared error manually (method 2)
rmse1 = sqrt(mean((pred - obs)^2, na.rm = TRUE))
rmse1
```

```
## [1] 3.272672
```

When it came to calculating the root mean squared error for our predictions we did it in two ways. First, we used the rmse function which takes our predicted values and actual values and calculates the root mean squared error for us. We also manually calculated the root mean squared error by subtracting the difference between the predicted values and the actual values, squaring the results, then calculating the mean, and square rooting it. We got the same result for both methods and found that root mean squared error to be approximately 3.161 .

Part D) Repeating processes with an AR(2) Model

First we fit and AR(2) model to the data using the ARIMA function.

```
#Fitting an AR(2) model to the data up to t=150
ar=arima(dataset,order=c(2,0,0))
print(ar)
```

```
##
## Call:
## arima(x = dataset, order = c(2, 0, 0))
##
## Coefficients:
##          ar1      ar2  intercept
##       0.8552  -0.0310     0.8089
## s.e.  0.0823   0.0831     0.9686
##
## sigma^2 estimated as 4.603:  log likelihood = -327.93,  aic = 663.87
```

```
#Testing significance of AR(2) coefficients

##AR(1) Beta significance
t=0.8552/0.0823
2*pt(-abs(t),df=150) #this is significant
```

```
## [1] 2.170406e-19
```

```
##AR(2) Beta significance
t=-0.0310/0.0831
2*pt(-abs(t),df=150) #this is not significant
```

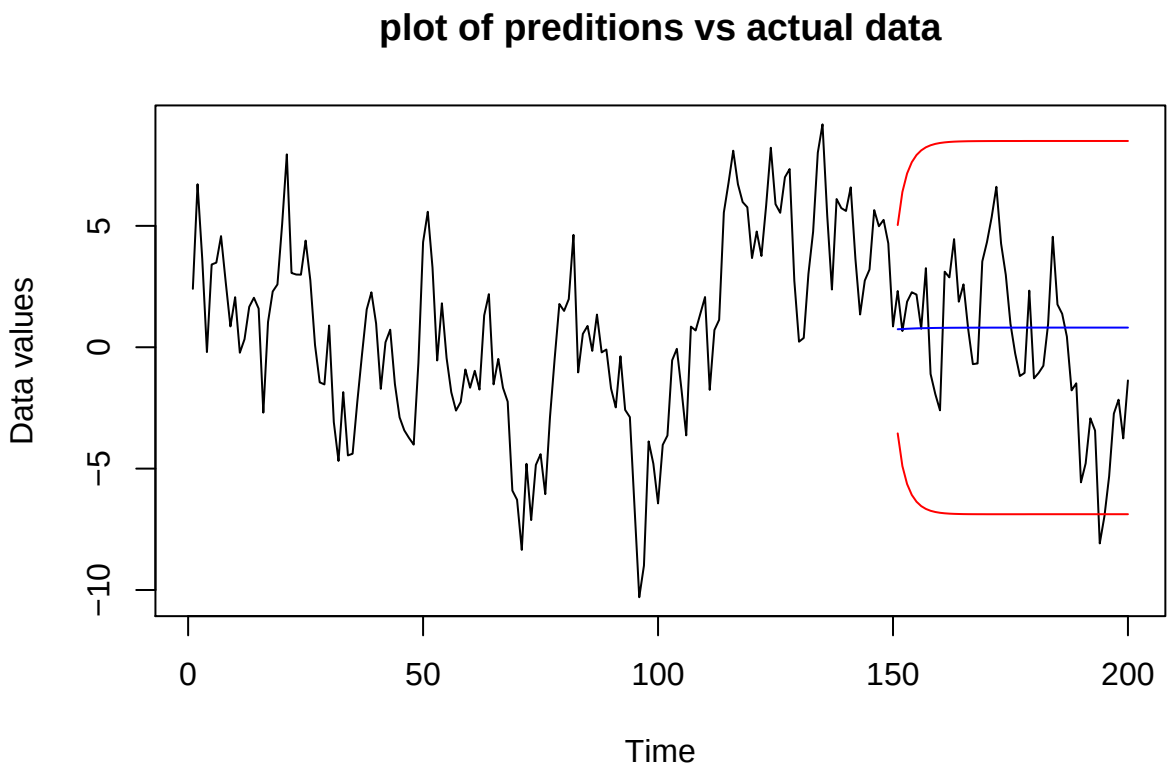
```
## [1] 0.7096419
```

```
##AR(2) Intercept significance
t=-0.8089/0.9868
2*pt(-abs(t),df=150) #this is not significant
```

```
## [1] 0.4136762
```

Above you can see the p.value for the AR(1) beta coefficient, AR(2) beta coefficient, and the intercept respectively. We first calculated each coefficients t-statistic using their respective estimate and standard error. We then compared the t-statistic's to the t-critical using the pt function and found that only the AR(1) coefficient was significant at the 5% critical level.

```
#Creating predictions of data with AR(2) model
arfit2 = arima(dataset,order=c(2,0,0))
pred2 = predict(arfit2,n.ahead = 50)

#Plotting predictions
plot(data, main="plot of preditions vs actual data", ylab="Data values")
lines(pred2$pred,col="blue")
lines(pred2$pred+2*pred2$se,col="red")
lines(pred2$pred-2*pred2$se,col="red")
```




```
pred =as.numeric(pred2$pred)

#calculating root mean squared error using function
rmse(pred,obs)
```

```
## [1] 3.272521
```

The output above is the root mean squared error of our predictions using the AR(2) model. Our root mean squared error for the AR(2) model is slightly smaller than our AR(1). This would suggest that the AR(2) generates better predictions that are closer to the actual values than our AR(1) model. We could therefore argue that the AR(2) model is better at generating forecasts in terms of root mean squared error.

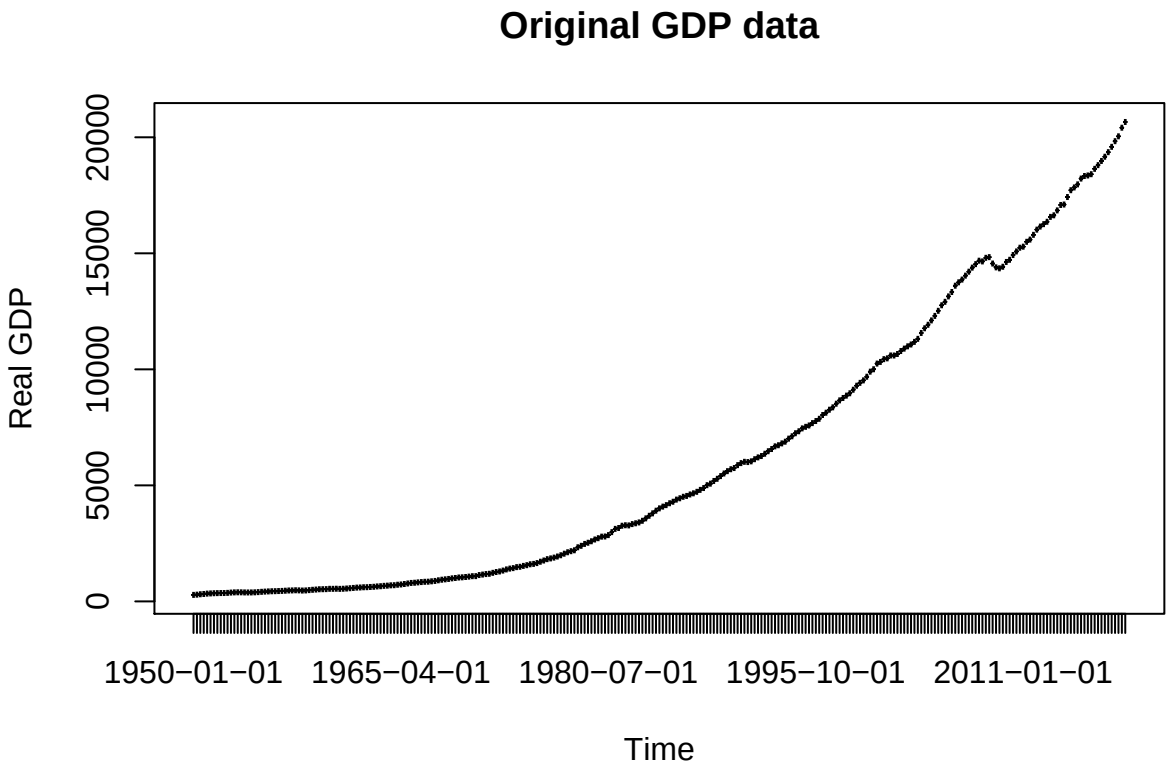
Question 2

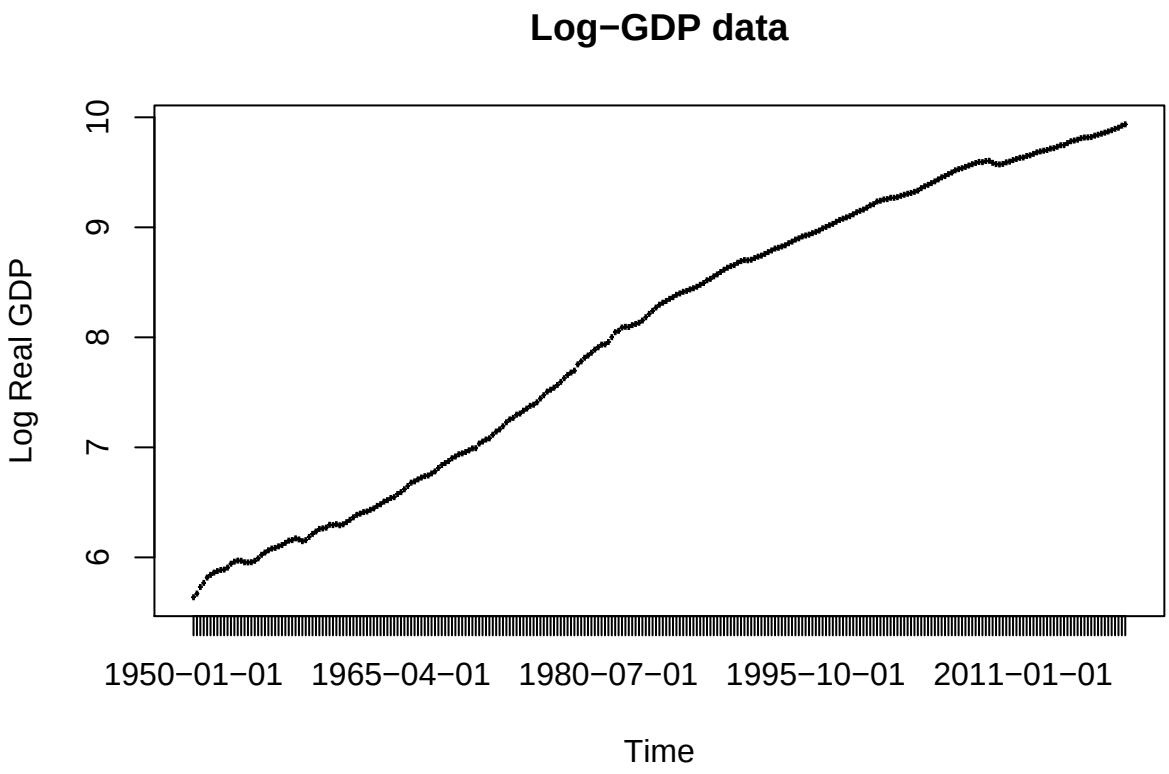
Part A) Estimate AR model, forecast, and plot

First we took the log of our GDP data and then plotted both our original GDP and the Logged GDP data.

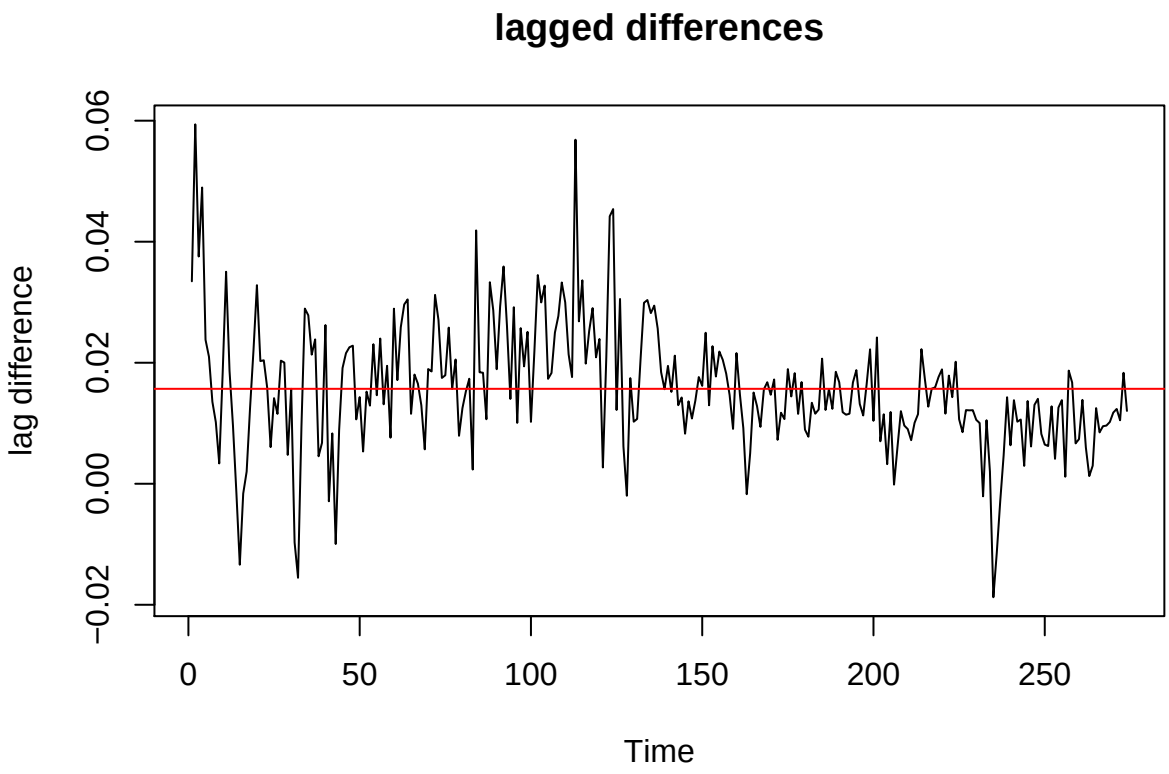
```
#gdp = gdp
#lgdp = log(gdp)
#diff.gdp = log differenced
#sub.gdp = subsetted to 2010:4
#gdp_forecast = forecasted for real GDP
```

```r
#gdp_for1 = forecasted for real GDP & actual GDP values
#trend.gdp = detrended log GDP

#Taking log of data and plotting
lgdp = gdp
lgdp$GDP =log(lgdp$GDP)
```






```
#Differencing data
diff.gdp =  diff(log(gdp$GDP))
diff.gdp = ts(diff.gdp)

#plotting data and determining whether it is stationary
plot(diff.gdp, main="lagged differences", ylab="lag difference")
abline(h=mean(diff.gdp),col="red")
```

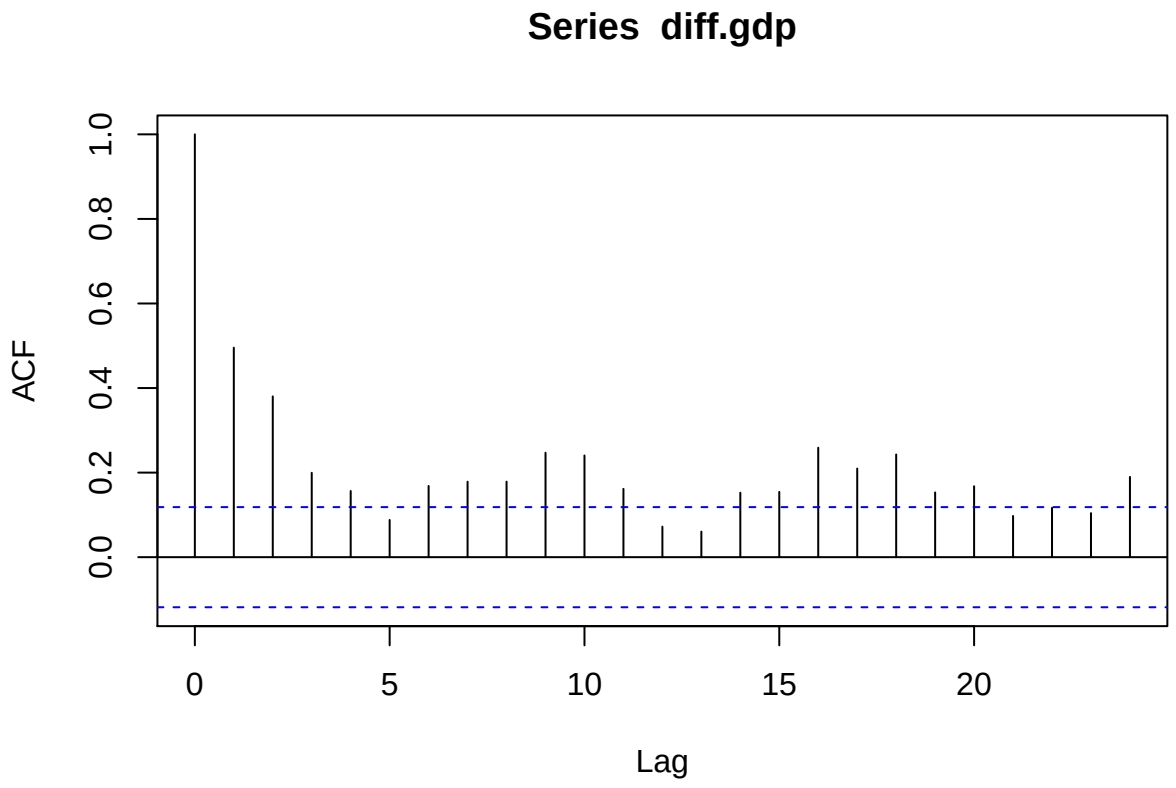



```
#Augmented Dicky-Fuller test
adf.test(diff.gdp,alternative = "stationary",k=0)
```

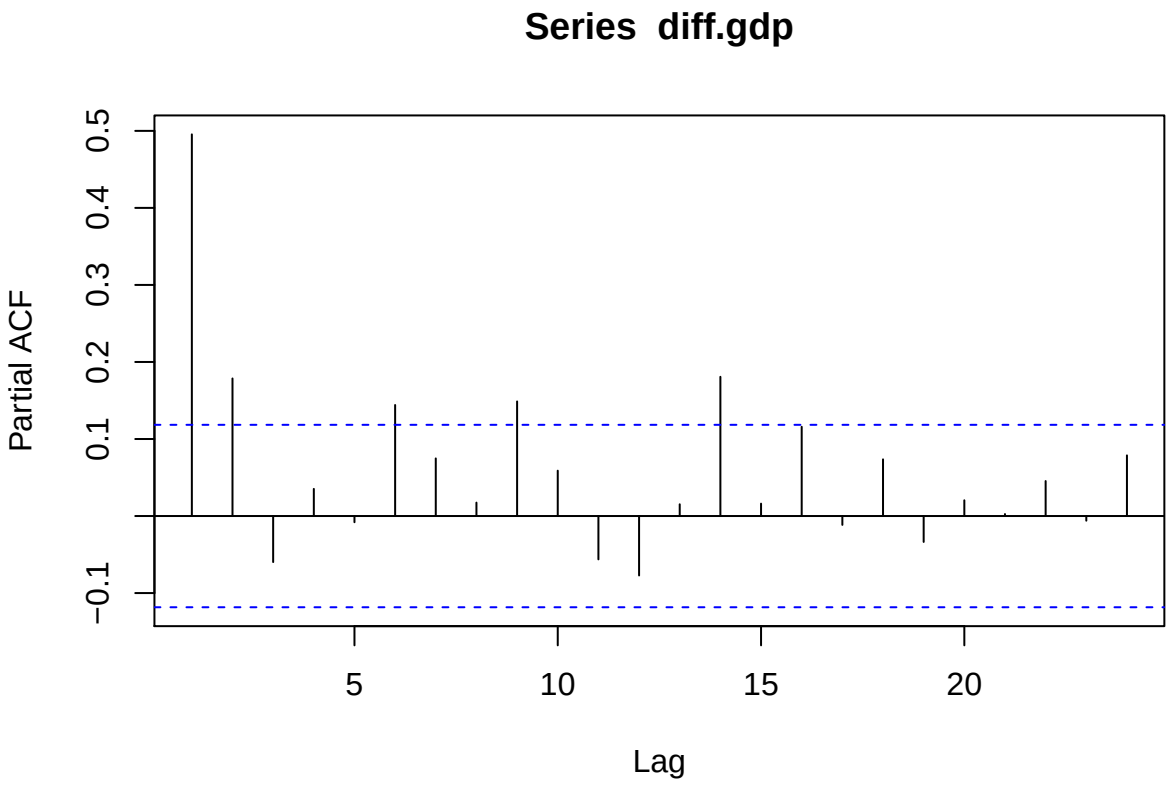
```
## Warning in adf.test(diff.gdp, alternative = "stationary", k = 0): p-value
## smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff.gdp
## Dickey-Fuller = -10.155, Lag order = 0, p-value = 0.01
## alternative hypothesis: stationary
```

```
#ACF and PCF plots
acf(diff.gdp)
```




```
pacf(diff.gdp)
```



When determining whether the data was stationary we looked at a plot of the data, the ACF/PCF plots, and we ran a augmented Dicky-Fuller stationary test. From the plot we can see that data does not deviate very far from the mean at any point. We can also see that the variation in the data remains fairly constant throughout, and has no upward/downward trends or indication of seasonality. This all suggests the data is stationary. Looking at the ACF plot we can see that the lags start to taper off the higher order you go, and the pcf plot also has consistently low values after the first lag. This also suggests that the data is stationary. Lastly, we ran an augmented Dicky-Fuller test which determined that the data was stationary at the 1% critical level. For these reasons we can assume the data is stationary.

```
#subsetting data
sub.gdp = diff.gdp[1:243]
sub.gdp = ts(sub.gdp)

#determining which AR model to use no constant
ar1 = arima(sub.gdp,order=c(1,0,0))
ar2 = arima(sub.gdp,order=c(2,0,0))
ar3 = arima(sub.gdp,order=c(3,0,0))

pred1 = predict(ar1,n.ahead = 31)
obs1 = diff.gdp[244:274]

pred1=as.numeric(pred1$pred)
obs1 = as.numeric(obs1)
rmse1 = rmse(pred1,obs1)

pred2 = predict(ar2,n.ahead = 31)
```

```
pred2=as.numeric(pred2$pred)
rmse2 = rmse(pred2,obs1)

pred3 = predict(ar3,n.ahead = 31)
pred3=as.numeric(pred3$pred)
rmse3 = rmse(pred3,obs1)

rmse1 #Root mean squared error of AR(1) model
```

```
## [1] 0.007920465
```

```
rmse2 #Root mean squared error of AR(2) model
```

```
## [1] 0.007856889
```

```
rmse3 #Root mean squared error of AR(3) model
```

```
## [1] 0.007880436
```

We decided that an AR(1) model would best fit and predict the data not because it had the lowest root mean squared error among the three models we tested (very close to other models, but not lowest). But because we know that when all else is equal (or almost equal) it is best to choose the lower AR model. Thus, despite it having a slightly larger root mean square error it is still favorable to choose an AR(1) model because is it the lowest order model.

```
#fitted AR(1) model
ar1
```

```
##
## Call:
## arima(x = sub.gdp, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.4911     0.0165
## s.e.  0.0560     0.0012
##
## sigma^2 estimated as 8.756e-05:  log likelihood = 790.25,  aic = -1574.5
```

The output above gives us our AR(1) beta coefficient and intercept for the model. We used this information to calculate the significance of each term below.

```
#testing significance of AR(1) Coefficients

##AR(1) Beta coefficients
t=0.4911/0.0560
2*pt(-abs(t),df=240) #this is significant
```

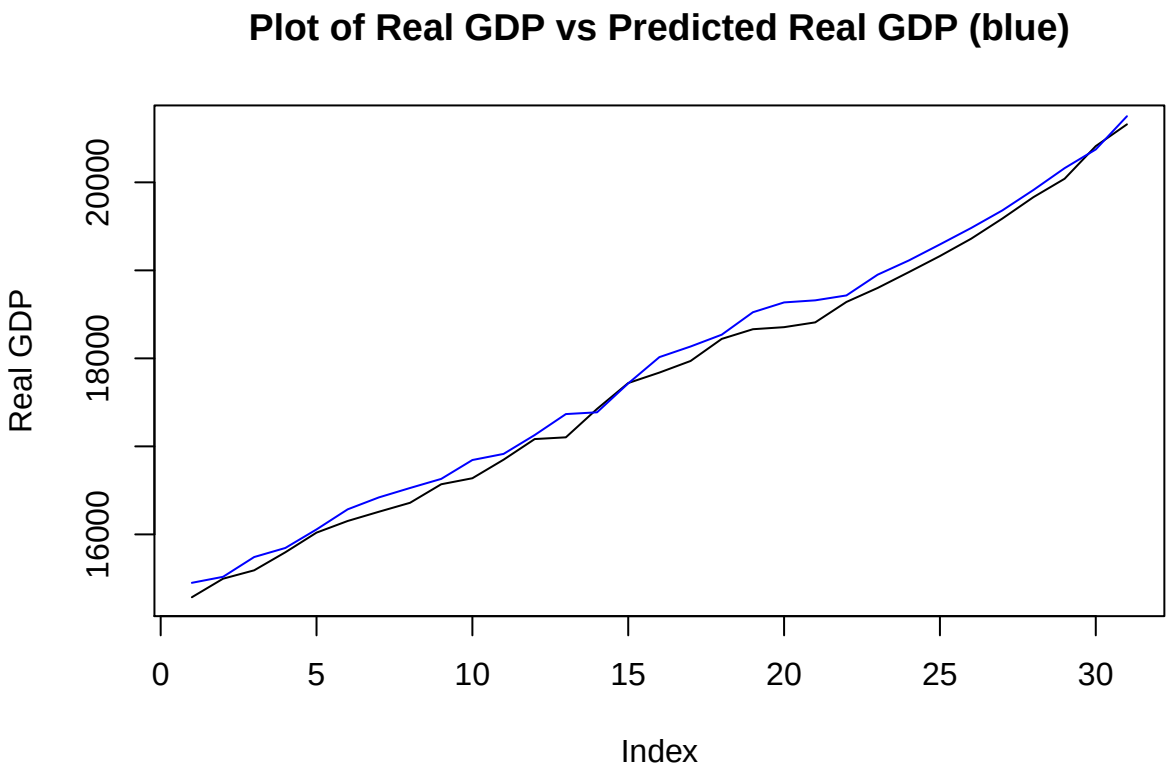
```
## [1] 3.365294e-16
```

```r
##AR(1) Intercept
t=0.0165/0.0012
2*pt(-abs(t),df=240) #intercept is significant
```

```
## [1] 4.073543e-32
```

The output above is the p-value for the AR(1) beta coefficient and the intercept respectively. We calculated each coefficients t-statistic using their respective estimate and standard error. We then compared the t-statistic's to the t-critical using the pt function and found that both the AR(1) beta coefficient and the intercept was significant at the 5% critical level.

```r
#getting predicted values for real GDP
newgdp=NULL

for(i in 1:31){
  newgdp[i]=exp(lgdp$GDP[i+243]+pred1[i])
  }

gdp_for1 = NULL

for (i in 1:244){
  gdp_for1[i]=gdp$GDP[i]
}

for (i in 245:275){
  gdp_for1[i]=newgdp[i-244]
}

gdp_for1=as.numeric(gdp_for1)

#plotting predicted values vs real values
plot(gdp$GDP[245:275],type="l", main="Plot of Real GDP vs Predicted Real GDP (blue)", ylab="Real GDP")
lines(gdp_for1[245:275],col="blue")
```




```
#calculating root mean squared error
rmse = rmse(gdp_for1[245:275],gdp$GDP[245:275])

rmse #root mean squared error of predicted values
```

```
## [1] 139.9915
```

From the graph above you can clearly see that our predictions follow the data closely. This confirms our assumption that the AR(1) model is accurate in predicting future periods. We also calculated the root mean squared error of our data and using the rsme functions and found it be 139.9915 .

Part B) Repeat Analysis with linearly detrended data

First we detrended the data by running a regression with a constant term and time term, and taking the residuals.

```
#detrending logged data with constant term and time
time=c(1:275)
detrended=lm(log(gdp$GDP)~time)
summary(detrended)
```

```
##
## Call:
## lm(formula = log(gdp$GDP) ~ time)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.37978 -0.13524 -0.00422  0.15565  0.27207
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 5.7294214  0.0209461   273.5   <2e-16 ***
## time        0.0166773  0.0001316   126.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1732 on 273 degrees of freedom
## Multiple R-squared:  0.9833, Adjusted R-squared:  0.9832
## F-statistic: 1.607e+04 on 1 and 273 DF,  p-value: < 2.2e-16
```

```
#taking residuals
detrendeddata=log(gdp$GDP)-detrended$fitted.values
newdata=data.frame(ID=c(1:275), detrendedGDP=detrendeddata)
```

We then subsetted the data and fit different AR models to see which one was the best predictor.

```
#subsetting data
subtrend.gdp = subset(newdata, ID<=244)

#determining which AR model to use (no constant)
ar1 = arima(subtrend.gdp$detrendedGDP,order=c(1,0,0))
ar2 = arima(subtrend.gdp$detrendedGDP,order=c(2,0,0))
ar3 = arima(subtrend.gdp$detrendedGDP,order=c(3,0,0))

pred1 = predict(ar1,n.ahead = 31)
obs1 = subset(newdata, ID>244)

pred1=as.numeric(pred1$pred)
obs1 = as.numeric(obs1$detrendedGDP)
rmse1 = rmse(pred1,obs1)

pred2 = predict(ar2,n.ahead = 31)
pred2=as.numeric(pred2$pred)
rmse2 = rmse(pred2,obs1)

pred3 = predict(ar3,n.ahead = 31)
pred3=as.numeric(pred3$pred)
rmse3 = rmse(pred3,obs1)

rmse1 #root mean squared error for AR(1)
```

```
## [1] 0.1379456
```

```
rmse2 #root mean squared error for AR(2)
```

```
## [1] 0.1420958
```

```
rmse3 #root mean squared error for AR(3)
```

```
## [1] 0.141975
```

The output above is the root mean squared error for the AR(1), AR(2), and AR(3) models respectively. Since the AR(1) model has the lowest root mean squared error and because it is the lowest order AR, which is always preferred, we chose to use an AR(1) model for the data.

```
ar1 #AR(1) Model
```

```
##
## Call:
## arima(x = subtrend.gdp$detrendedGDP, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.9968    -0.0834
## s.e.  0.0029     0.1169
##
## sigma^2 estimated as 0.0001149:  log likelihood = 757.92,  aic = -1509.83
```

The output above gives us our AR(1) beta and intercept coefficients along with their respective standard errors. We calculated the significance of these coefficients below.

```
#testing significance of AR(1) coefficients

#Significance of AR(1) Beta coefficient
t=0.9968/0.0029
2*pt(-abs(t),df=240) #this is significant
```

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

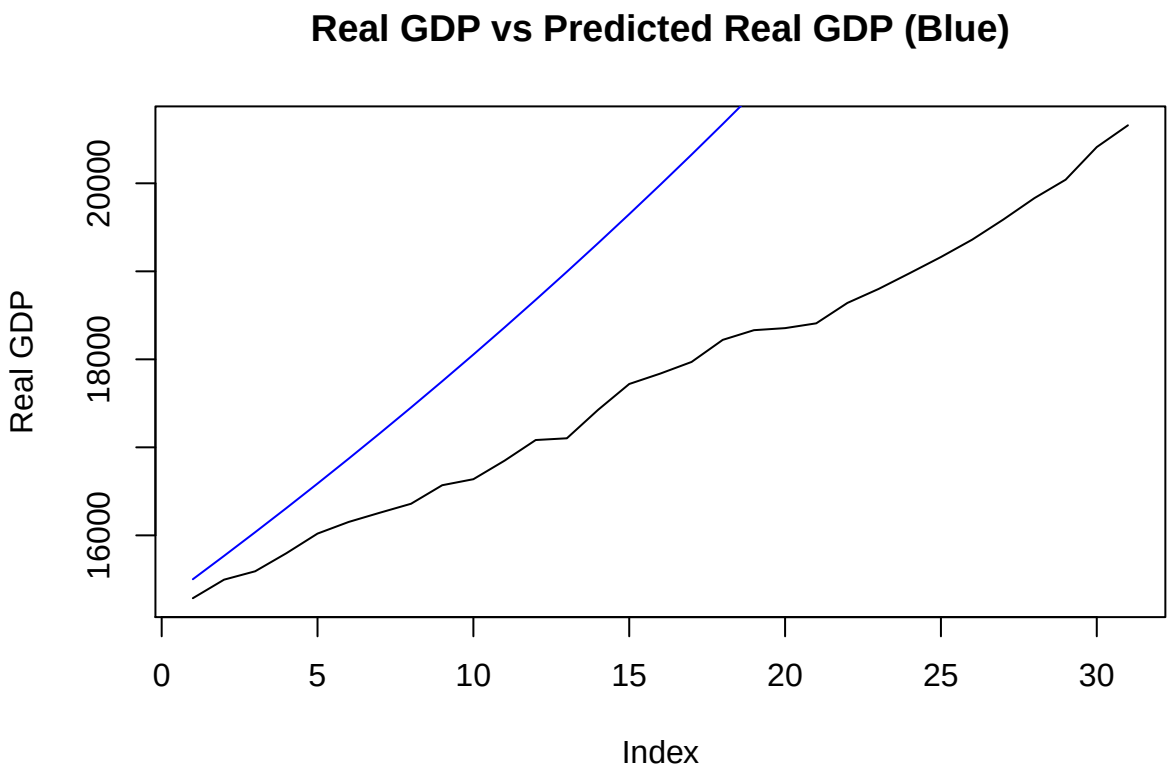
```
#significance of Intercept
t=-0.0834/0.1169
2*pt(-abs(t),df=240) #intercept is not significant
```

```
## [1] 0.4762727
```

The output above is the p-value for the AR(1) beta coefficient and the intercept respectively. to find each coefficients p-value We first calculated each their respective t-statistic using their estimate and standard error. We then compared the each t-statistic to the t-critical using the pt function and found that only the AR(1) beta coefficient was significant at the 5% critical level. The intercept in this case is not statistically significant at the 5% level. We then transformed our predictions into real GDP and plotted the results below.

```
#reversing transformation
realpred=NULL

for(i in 1:31){
realpred[i]=exp(detrended$coefficients[1]+detrended$coefficients[2]*time[i+244]+pred1[i])
}
```

```r
#plotting data
plot(gdp$GDP[245:275],type="l", main="Real GDP vs Predicted Real GDP (Blue)", ylab="Real GDP")
lines(realpred,col="blue",type="l")
```




```r
#root mean squared error
rmse=rmse(realpred,gdp$GDP[245:275])
rmse
```

```
## [1] 2887.773
```

The output above is the root mean squared error of out AR(1) model. We can clearly see that the model makes bad predicitons when using linearly detrended data.

Part C) Comparing Methods

Based on our results it is clear that the log-difference transformation is better for forecasting GDP. This is clearly illustrated by the plot of each methods predicitons. The log-difference predicitons followed the data very closely, where as the linearly detrended predicitons were not very accurate at all. As a result the log-differenced data also had a much lower root mean squared error which confirmed our assumptions.