Groupwork Assignment Submission 1

Group 3A

6/23/2020

Name of Group Members

- Abimbola Olawale Victor (abimbolaolawale41@gmail.com)
- Ogbekile Chukwudi Samuel (Ogbekilechukwudi@gmail.com)
- Aboubacar SAÏDOU BOUREIMA (aboubacarsaidouboureima145@gmail.com)
- Michael Segun Olanipekun (molanipekun10@gmail.com)
- Abiodun Jimoh (suliman.jumah@gmail.com)

Introduction

This submission document the Linear regression analysis of JP Morgan, S&P 500 stock prices and the ARIMA/ARMA forecast for the Case-Shiller Index from 1987 till date. The JP Morgan was first evaluated with the basic statistics computation in terms of returns and annualized volatility. The document also cover the linear regression analysis of JP Morgan and S&P 500 with the former as the explained variable and the latter the explanatory variable. The ARIMA forecast of the Case-Shiller are also explain along within the codeblocks.

1.0 Univariate Basic Statistics

Importing the JP Morgan data downloaded

```
library(readr)
library(tseries)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
JPM_sheet <- read_csv("JPM(1).csv",col_types = cols(Date = col_date(format = "%m/%d/%Y")))
```

The average value of the JP Morgan Stock

```
average_stock <- mean(JPM_sheet$`Adj Close`)
average_stock
```

```
## [1] 104.5844
```

Get the Adjusted Close Price

```
price<-JPM_sheet$`Adj Close`
n<- length(price)
```

Calculate the daily return of the JPM Adjusted Closed Price

```
daily_return_log<-log(price[-n]/price[-1])
daily_return_log
```

```
##   [1]  2.241076e-02  4.914034e-02 -2.996931e-02 -6.756147e-03  4.521726e-02
##   [6] -1.982452e-02 -1.533076e-02 -6.156187e-03 -2.286190e-02 -4.164370e-03
##  [11]  7.211563e-03 -2.616398e-04 -4.175749e-03  1.824781e-03 -2.006173e-02
##  [16] -1.236897e-02  1.194274e-02  1.597563e-02  1.808454e-02  9.702707e-04
##  [21] -1.523792e-02 -8.688031e-04  3.740904e-03 -8.721076e-05 -2.835477e-02
##  [26]  3.224460e-03  1.205602e-02  1.124585e-02 -2.519664e-03 -1.734058e-03
##  [31]  7.914103e-03 -9.599596e-04 -8.719846e-04  4.264311e-02  2.710336e-02
##  [36] -3.037229e-02  1.959059e-02  1.572734e-03 -1.807644e-02  1.946633e-02
##  [41] -1.362954e-02 -1.506918e-02 -1.304502e-02  2.525369e-02 -1.193693e-02
##  [46] -1.893200e-02  1.694122e-02 -2.455597e-02  2.745293e-02  8.163086e-04
##  [51]  0.000000e+00  8.108281e-03 -2.171644e-02  2.240211e-03  4.856266e-03
##  [56]  4.698589e-03  3.811288e-03 -9.996723e-04  6.378176e-03  5.683519e-03
##  [61]  0.000000e+00  7.937183e-03  6.320927e-03 -1.103548e-02 -8.631851e-03
##  [66] -1.470340e-02 -2.156965e-02 -7.729407e-03  3.769440e-03 -3.513017e-04
##  [71]  7.667751e-03 -2.738832e-03  3.358202e-03  1.633312e-02 -9.136501e-03
##  [76] -7.639105e-03  4.612051e-03  1.126407e-02  5.137716e-03  4.368403e-02
##  [81] -2.258828e-02  1.244442e-02 -1.290581e-02 -4.610219e-04  5.640525e-03
##  [86] -2.309920e-02 -3.978973e-03 -2.794021e-03  2.523274e-03  5.791232e-03
##  [91]  1.998632e-03  1.779876e-02  1.204014e-03 -2.591626e-03  5.840659e-03
##  [96]  9.309510e-05  2.788830e-04  1.650606e-02  9.119491e-03 -6.677952e-04
## [101]  1.556974e-02 -1.623709e-02  6.981464e-03 -8.409890e-03  1.408811e-02
## [106] -6.480688e-03 -3.272738e-03 -3.047455e-02  6.171111e-03  2.159472e-03
## [111] -4.314222e-03  4.596264e-03 -3.890973e-02  7.237226e-04 -9.277984e-03
## [116]  1.481363e-02 -1.256951e-02 -1.843095e-02 -7.032948e-03 -8.982856e-03
## [121]  2.869258e-03 -1.022184e-02 -6.014774e-03  1.536639e-02 -6.157634e-03
## [126] -4.227595e-03 -8.060468e-03 -2.562624e-04 -3.664612e-03 -2.039745e-03
## [131]  7.755846e-03  9.887713e-03  1.602693e-02 -6.651093e-03  8.320713e-03
## [136] -9.366695e-03  0.000000e+00  1.307766e-03 -6.088540e-03  3.039604e-03
## [141]  2.089755e-03  4.358840e-04 -1.754650e-02  4.895793e-03  3.277253e-03
## [146]  4.936134e-03  5.309662e-03 -4.962348e-03  4.875095e-03  4.285207e-03
## [151] -1.926247e-03  5.350252e-03 -6.312023e-03  1.186773e-02 -3.883365e-03
## [156]  1.761594e-04 -2.991091e-03 -4.032660e-03 -2.863259e-02 -8.550250e-03
## [161]  6.596828e-03  9.634798e-03  2.831203e-03  1.184060e-02  4.356599e-03
## [166]  1.477856e-02 -5.832001e-03 -4.132379e-03 -9.344747e-03 -8.975586e-03
## [171]  5.654891e-03 -6.088649e-03  6.961452e-03  2.699395e-02  3.042130e-02
## [176]  1.097286e-02  5.719907e-03 -2.121403e-02 -1.107827e-02  1.596962e-02
## [181]  1.666531e-03  1.446800e-02  1.049115e-02  1.879790e-02 -1.508560e-02
## [186]  1.382773e-02 -1.373229e-02 -1.749041e-02 -2.151020e-02  3.669136e-04
## [191]  5.520859e-03 -6.529679e-03 -4.664084e-03 -1.700785e-02 -8.040818e-03
## [196]  9.746554e-03  2.125036e-02 -5.856915e-03  2.083776e-02 -2.520830e-02
## [201]  7.271103e-04 -7.608057e-03  2.170830e-02  7.496827e-03  9.239804e-03
## [206] -2.417790e-02 -4.201208e-03 -1.105781e-02  7.963799e-03 -1.021485e-02
## [211] -9.398869e-03  4.566333e-02  1.920778e-02  1.822771e-02  1.886192e-02
## [216]  9.815263e-03 -6.455084e-03 -9.895398e-04  8.242021e-03  1.284504e-02
## [221]  4.758445e-03  1.276632e-02  8.671474e-03  2.392307e-02  2.179244e-02
## [226] -4.062212e-02 -1.119197e-02  2.166424e-03
```

Calculate the Volatility of Stock

```
volatility<- sd(daily_return_log)*sqrt(252)*100
volatility
```

```
## [1] 22.88891
```

2.0 Linear Regression

Import the JPM dataset

```
JPM_sheet <- read_csv("JPM(1).csv",
                      col_types = cols(Date = col_date(format = "%m/%d/%Y")))
```

Import the S & P dataset

```
SandP500_sheet <- read_csv("^GSPC(1).csv")
```

```
## Parsed with column specification:
## cols(
##   Date = col_date(format = ""),
##   Open = col_double(),
##   High = col_double(),
##   Low = col_double(),
##   Close = col_double(),
##   `Adj Close` = col_double(),
##   Volume = col_double()
## )
```

Extracting the explained and explanatory variable from the two different datasets.

```
JPM_price<-JPM_sheet$`Adj Close`
SandP500_price <-SandP500_sheet$`Adj Close`
```

Applying linear regression model

```
linear_regression = lm(JPM_price~SandP500_price)
linear_regression
```

```
##
## Call:
## lm(formula = JPM_price ~ SandP500_price)
```

```
##
## Coefficients:
##   (Intercept)  SandP500_price
##      13.41551         0.03323
```

Summary of Linear Regression

```r
summary(linear_regression)
```

```
##
## Call:
## lm(formula = JPM_price ~ SandP500_price)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.5902 -2.3388  0.4717  2.3256  5.5104
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    13.415509   5.165471   2.597     0.01 *
## SandP500_price  0.033234   0.001882  17.662   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.898 on 227 degrees of freedom
## Multiple R-squared:  0.5788, Adjusted R-squared:  0.5769
## F-statistic: 311.9 on 1 and 227 DF,  p-value: < 2.2e-16
```

Based on the Linear regression, we have:

JPM_Price = 13.415509 + 0.033234SandP500_price. A unit increase in SandP500_price will cause about <2e-16S in JPM_Price. Also the Pvalue ofthe explanatory variable is signiificant which means that it is contributing significantly to the model. The Mutiple R-Squared is about 57.88% while the Adjusted R-Squared is about 57.68%. The "Estimate" represent the coefficient of the explanatory variable S&P price and the value of the intercept (constant). Thus, y= mx + c The Std. Error represent the standard errors of the estimated coefficient. The standard error of the estimated coefficient is much less than the residual standard error, thus, the hypothesis at the true values will not be rejected. Only the intercept has a standard error greater than the standard error but can be overlook as its not one of the explain variable. The t-values for the individual significances define the ratio of the estimated values to the Std. Error of the coefficients. F-Statistics evaluates the joint statistics significance of all explanatory variables. The F-Statistics is similar to the t-value and in this model the same values as with the explanatory variable t-value (i.e there is only one explanatory variable used). The p-value of the F-statistics is approximately zero and thus the hypothesis can be accepted.

Based on the result from the linear regression it can be deduced that we can explain only about 57% of the variation among the explanatory variables.

3.0 Univariate Time Series

Importing the Data

```r
chdata <- read.csv("CSUSHPISA.csv")
```

Noticed the date data is not well formated.

```
chdata$DATE= as.Date(chdata$DATE)
```

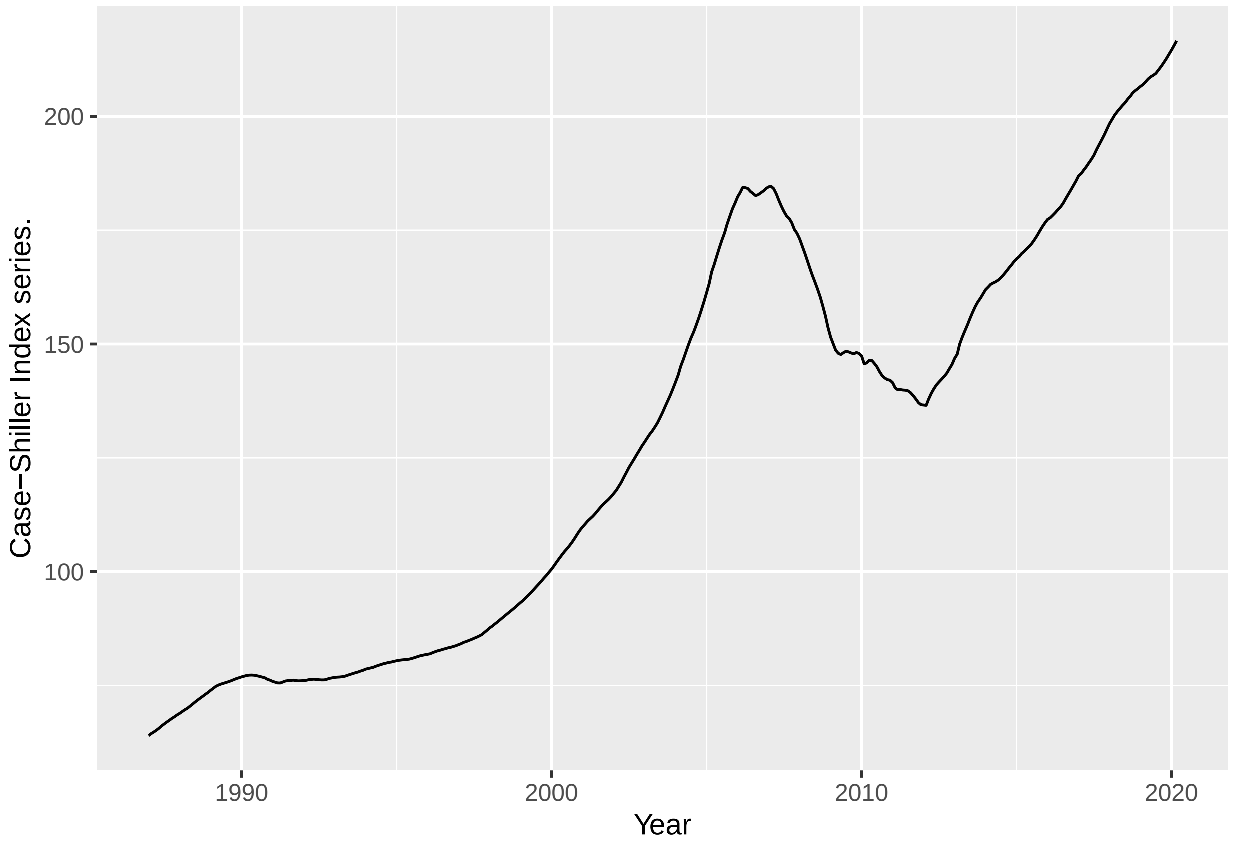
```
data.ts<-ts(chdata$CSUSHPISA,frequency=12, start=c(1987,1,1))
```

Fitting and Forcasting ARMA model

```
library(ggfortify)
```

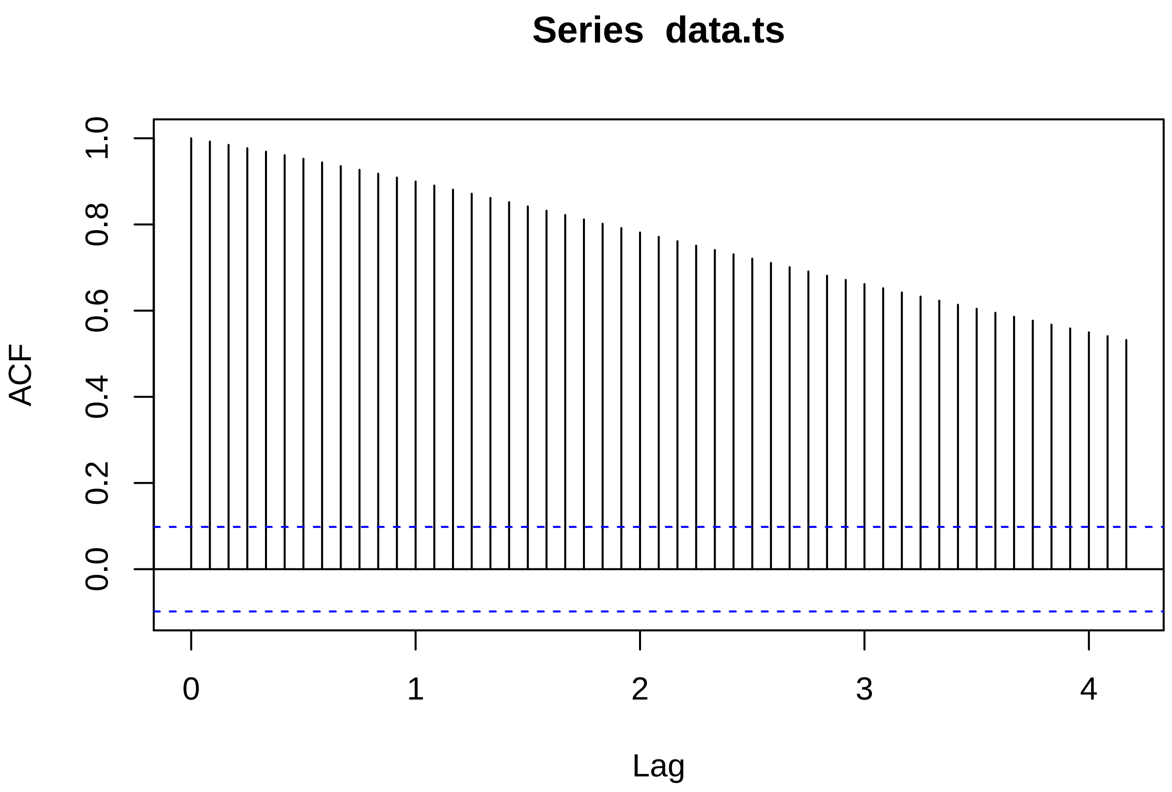
```
## Loading required package: ggplot2
```

```
autoplot(data.ts) +
  xlab("Year") + ylab("Case-Shiller Index series.")
```



### Plotting the ACF plot of the series

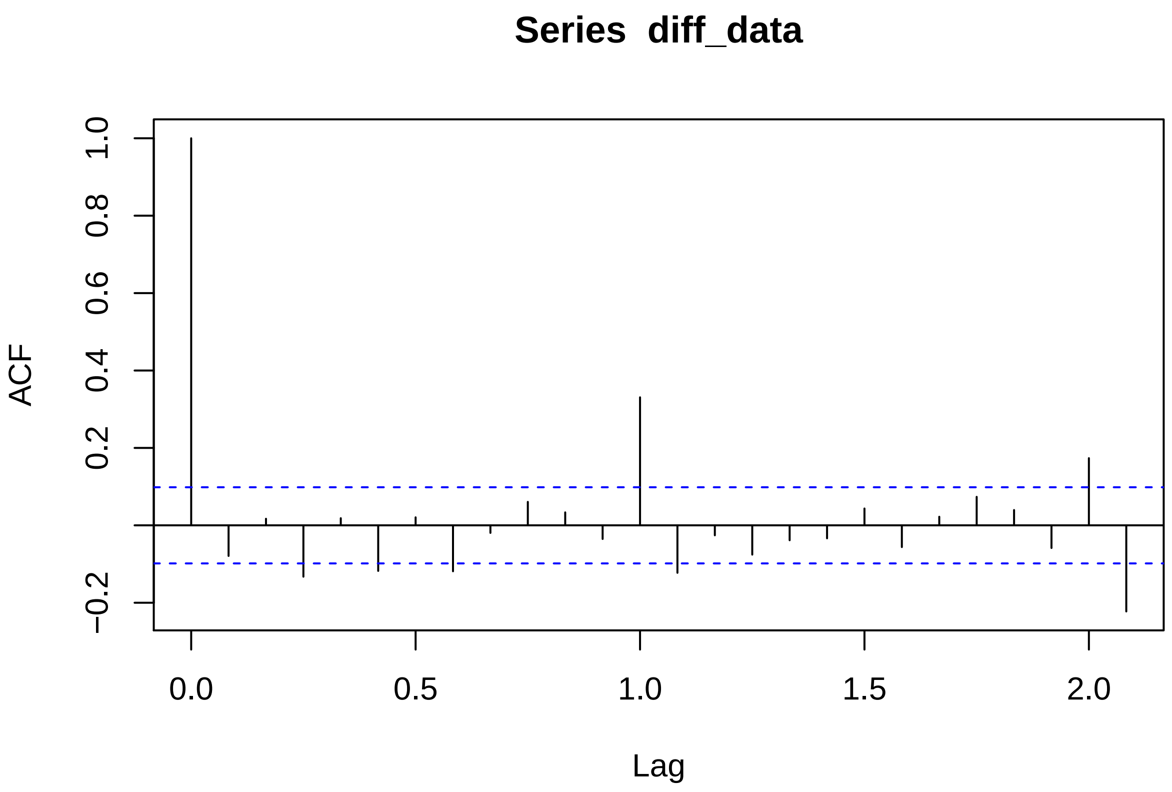
```
data.acf <- acf(data.ts, lag.max = 50)
```



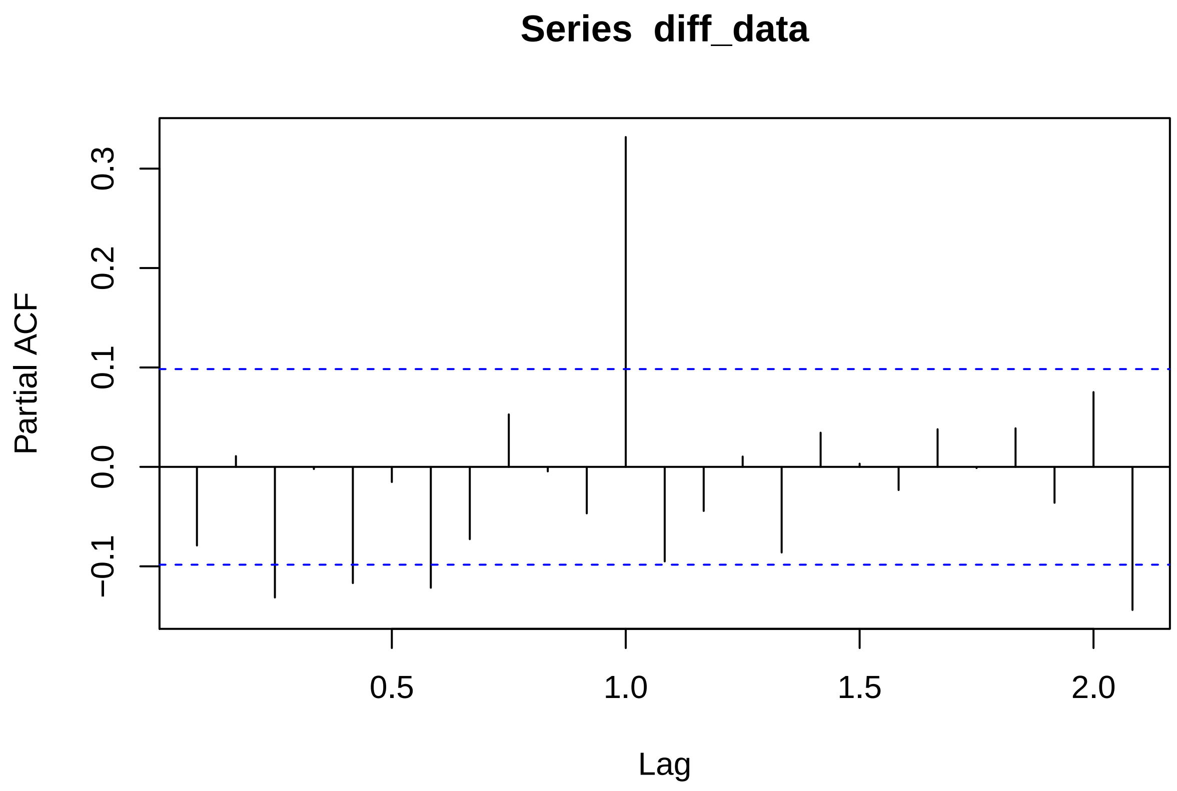
Looking at the ACF plot it is obvious that the data is finding it difficult to decay to zero which indicate that the data is not stationary. So we decided to difference the data.

```
diff_data <- diff(data.ts, differences = 2)
```

```
acf(diff_data)
```



```
pacf(diff_data)
```



Based on the ACF and PACF plot used, we strongly suggest to fit an ARMA model of order (p,q) = (15,1)

```r
library(tseries)
adf.test(diff_data)
```

```
## Warning in adf.test(diff_data): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff_data
## Dickey-Fuller = -9.3017, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

In other to confirm the stationarity level of the differennced data we used Augumented Dicky fuller test which the result is as above.

Fitting the ARMA model

```r
library(forecast)
```

```
## Registered S3 methods overwritten by 'forecast':
##   method                 from
##   autoplot.Arima         ggfortify
##   autoplot.acf           ggfortify
##   autoplot.ar            ggfortify
##   autoplot.bats          ggfortify
##   autoplot.decomposed.ts ggfortify
##   autoplot.ets           ggfortify
##   autoplot.forecast      ggfortify
##   autoplot.stl           ggfortify
##   autoplot.ts            ggfortify
##   fitted.ar              ggfortify
##   fortify.ts             ggfortify
##   residuals.ar           ggfortify
```

```r
ARMAmodel<- arima(diff_data, order=c(15,0,1))
```

Model Summary

```r
ARMAmodel
```

```
##
## Call:
## arima(x = diff_data, order = c(15, 0, 1))
##
## Coefficients:
##           ar1      ar2      ar3      ar4      ar5      ar6      ar7      ar8
##       -0.6192  -0.0267  -0.1806  -0.0884  -0.0846  -0.0827  -0.1017  -0.1206
## s.e.   0.2906   0.0602   0.0586   0.0738   0.0570   0.0618   0.0578   0.0629
##          ar9    ar10     ar11    ar12    ar13     ar14    ar15     ma1
##       0.0517  0.0408  -0.0363  0.3092  0.0894  -0.0972  0.0158  0.5799
## s.e.  0.0610  0.0625   0.0570  0.0573  0.1076   0.0643  0.0547  0.2864
##       intercept
##          0.0014
## s.e.     0.0094
##
## sigma^2 estimated as 0.05274:  log likelihood = 19.84,  aic = -3.68
```

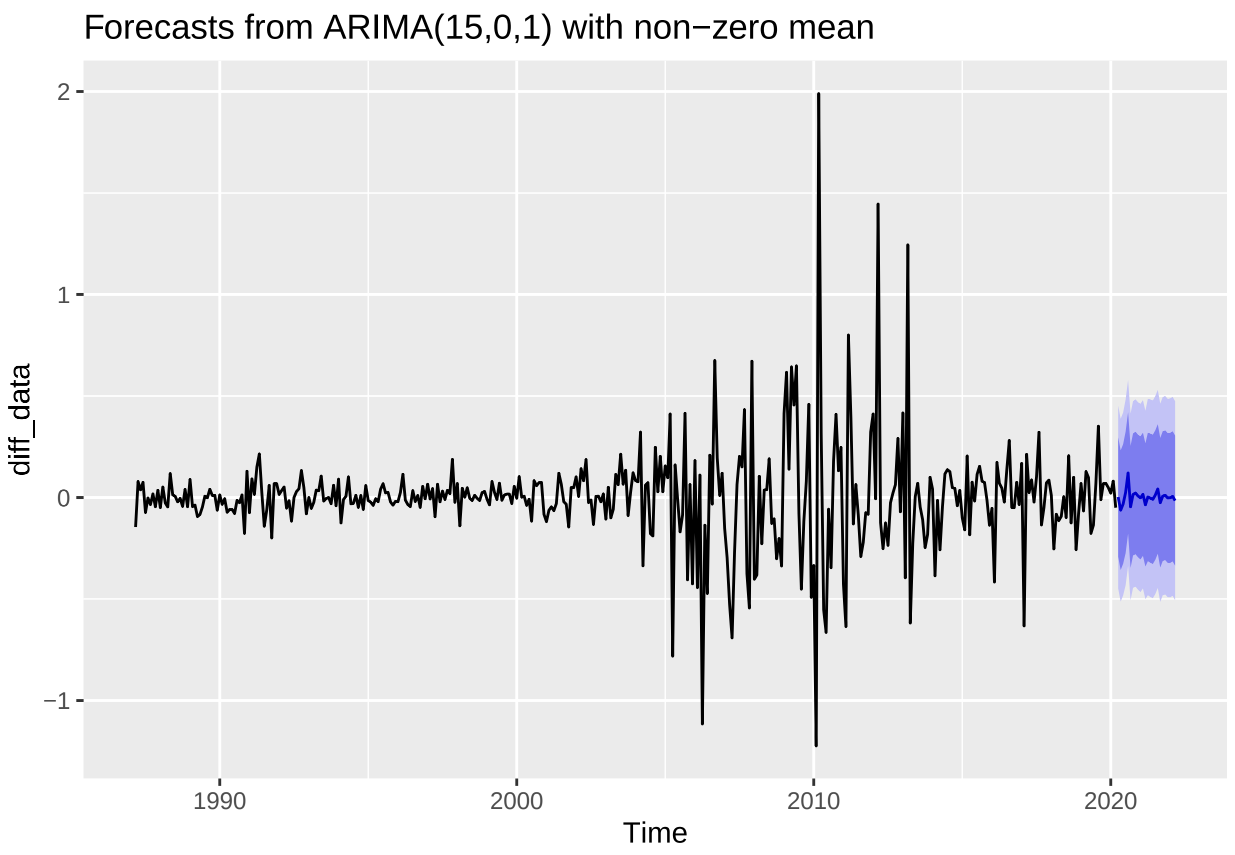
Predicting ARMA Model

```
forcast <- forecast(ARMAmodel)
forcast
```

```
##          Point Forecast      Lo 80     Hi 80      Lo 95     Hi 95
## Apr 2020    0.002189350 -0.2921134 0.2964921 -0.4479078 0.4522865
## May 2020   -0.062698936 -0.3572296 0.2318317 -0.5131446 0.3877468
## Jun 2020   -0.031904959 -0.3264364 0.2626264 -0.4823519 0.4185419
## Jul 2020    0.025070754 -0.2740887 0.3242302 -0.4324541 0.4825956
## Aug 2020    0.121116330 -0.1781650 0.4203976 -0.3365949 0.5788276
## Sep 2020   -0.046471129 -0.3470255 0.2540832 -0.5061294 0.4131871
## Oct 2020    0.014540984 -0.2860288 0.3151108 -0.4451409 0.4742228
## Nov 2020    0.022243178 -0.2795343 0.3240206 -0.4392856 0.4837719
## Dec 2020    0.007505923 -0.2944050 0.3094168 -0.4542270 0.4692388
## Jan 2021   -0.002213210 -0.3054633 0.3010369 -0.4659943 0.4615678
## Feb 2021    0.015780434 -0.2875226 0.3190835 -0.4480815 0.4796424
## Mar 2021   -0.036761684 -0.3400906 0.2665672 -0.5006632 0.4271398
## Apr 2021    0.002819179 -0.3137073 0.3193457 -0.4812664 0.4869047
## May 2021   -0.003599610 -0.3217773 0.3145781 -0.4902104 0.4830112
## Jun 2021   -0.009124037 -0.3274743 0.3092262 -0.4959987 0.4777506
## Jul 2021    0.012157401 -0.3067631 0.3310779 -0.4755895 0.4999043
## Aug 2021    0.042243876 -0.2767250 0.3612128 -0.4455770 0.5300647
## Sep 2021   -0.025994367 -0.3452651 0.2932763 -0.5142768 0.4622880
## Oct 2021    0.006610000 -0.3127307 0.3259507 -0.4817795 0.4949995
## Nov 2021    0.010952610 -0.3086028 0.3305080 -0.4777652 0.4996704
## Dec 2021   -0.002490085 -0.3220593 0.3170792 -0.4912291 0.4862489
## Jan 2022   -0.001959486 -0.3223721 0.3184532 -0.4919884 0.4880694
## Feb 2022    0.005625784 -0.3147929 0.3260445 -0.4844123 0.4956639
## Mar 2022   -0.015691282 -0.3361386 0.3047560 -0.5057732 0.4743906
```

Plot Forcast

```
autoplot(forecast(ARMAmodel))
```



Augumented Dicky Fuller (ADF) Test

```r
adf.test(data.ts)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  data.ts
## Dickey-Fuller = -3.0809, Lag order = 7, p-value = 0.1207
## alternative hypothesis: stationary
```

Interpretation of ADF

Based on the result of the ADF test, it was obvious that the pvalue of the test is 0.1207 which is less than the 0.05. According tp the selection Criterion we accepted the null hyputhesis.

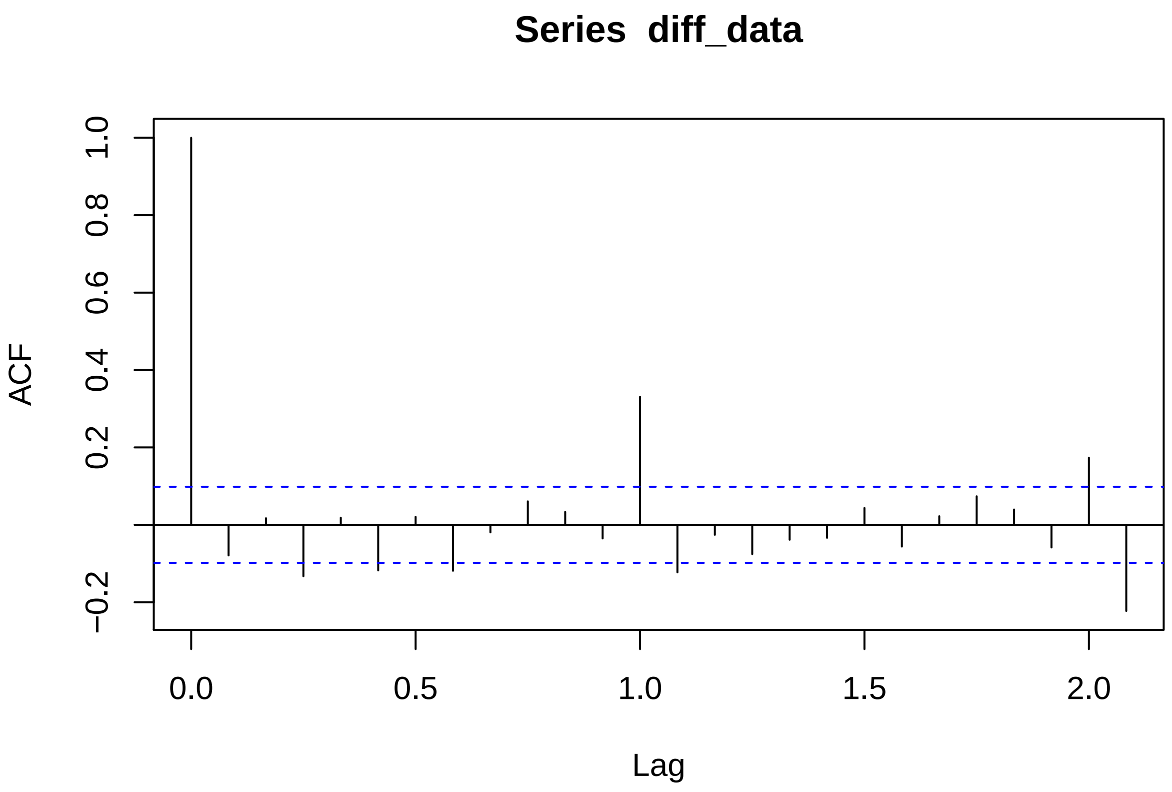
Decision Rule

The Cash Shriller Index is not Stationary and it will be of our best interest to find a way to stationarize it before fitting the model.

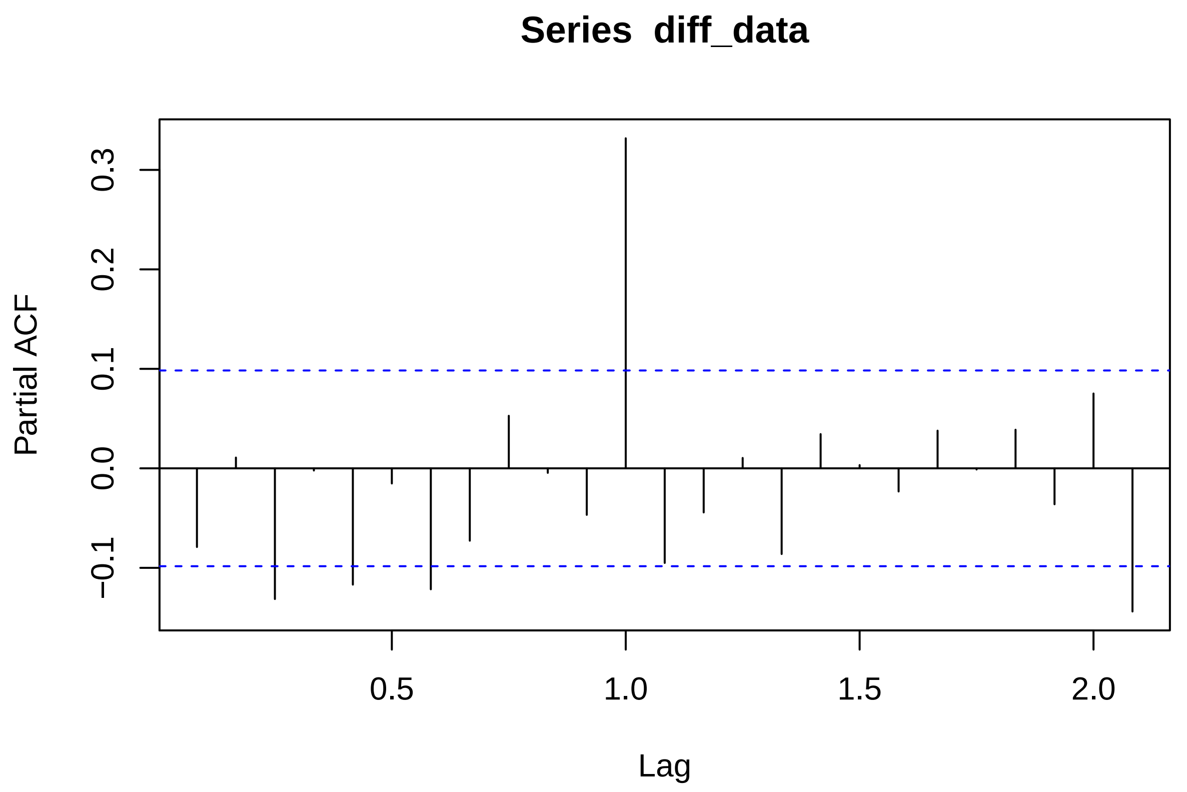
Implimenting Arima Model

Implimenting Box Jenkins for model order selection

```r
acf(diff_data)
```



```r
pacf(diff_data)
```



We used the ACF and PACF plot to determine the order of the arima model. Note that the order of differencing of the model i.e "d" is 2. this was based on the fact that we differenced the data twice to make it stationary before fitting the model.

The ACF plot shows after the first spike the model started to decay to zero. This shows the MA of order 1.

Also, we observed that there was no spike that crosses the line until the 3rd lag while other decays to zero.Therefore we decided to select an AR of order 3.

The order to be used for this model is (3,2,1).

```
library(forecast)
ARIMAmodel<-arima(diff_data, order=c(3,2,1))
```

```
ARIMAmodel
```

```
##
## Call:
## arima(x = diff_data, order = c(3, 2, 1))
##
## Coefficients:
##           ar1      ar2      ar3      ma1
##       -0.7626  -0.4666  -0.3050  -1.0000
## s.e.   0.0479   0.0566   0.0477   0.0063
##
## sigma^2 estimated as 0.08249:  log likelihood = -72,  aic = 154.01
```

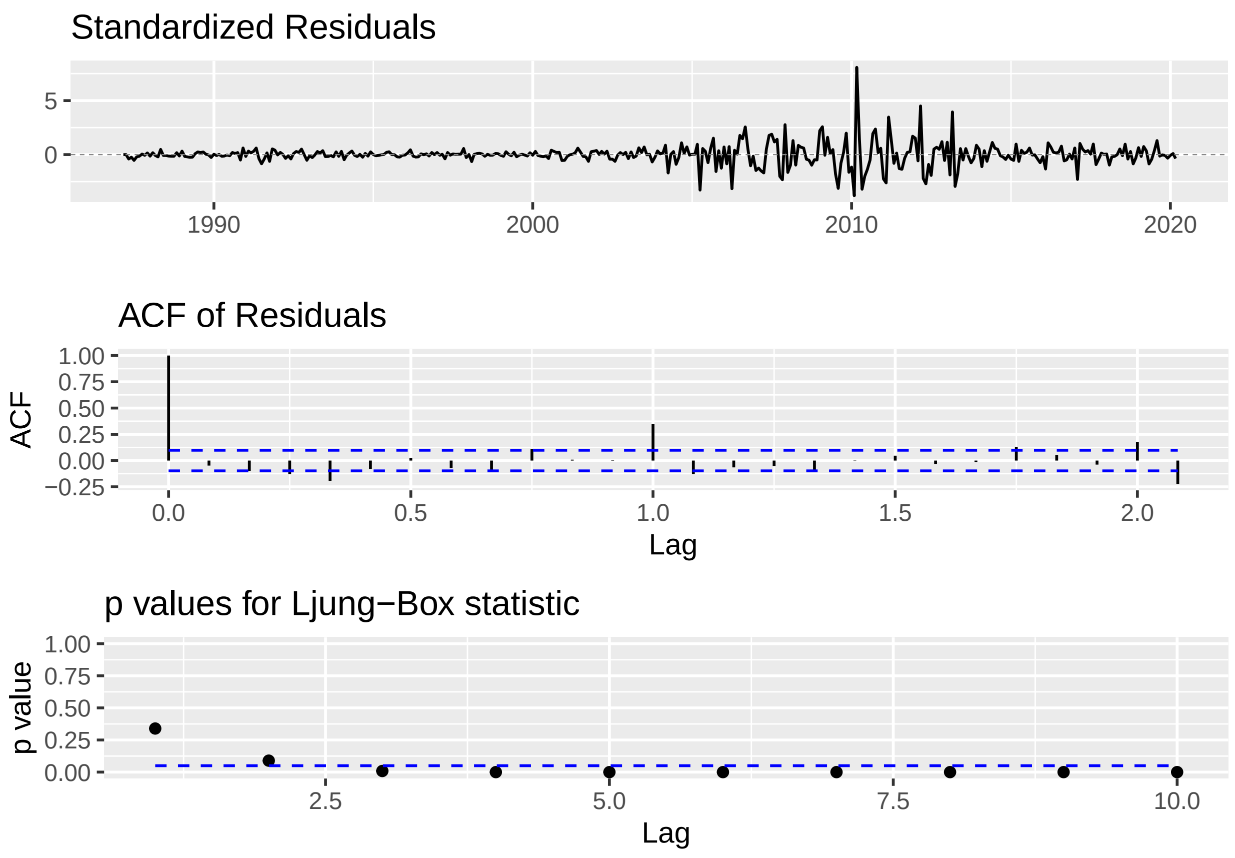
```
cast <- forecast(ARIMAmodel)
cast
```

```
##          Point Forecast      Lo 80     Hi 80      Lo 95     Hi 95
## Apr 2020   0.0306460195 -0.3378887 0.3991807 -0.5329791 0.5942712
## May 2020   0.0121811189 -0.3668137 0.3911760 -0.5674415 0.5918037
## Jun 2020   0.0287960509 -0.3721951 0.4297872 -0.5844670 0.6420591
## Jul 2020   0.0003474638 -0.4175742 0.4182691 -0.6388085 0.6395034
## Aug 2020   0.0201270132 -0.4407782 0.4810323 -0.6847667 0.7250207
## Sep 2020   0.0134566339 -0.4654454 0.4923587 -0.7189608 0.7458741
## Oct 2020   0.0181973701 -0.4831405 0.5195353 -0.7485328 0.7849275
## Nov 2020   0.0118671245 -0.5091768 0.5329110 -0.7850008 0.8087350
## Dec 2020   0.0167228995 -0.5279854 0.5614312 -0.8163366 0.8497824
## Jan 2021   0.0147337768 -0.5487534 0.5782209 -0.8470455 0.8765130
## Feb 2021   0.0161216710 -0.5670748 0.5993182 -0.8758004 0.9080437
## Mar 2021   0.0147162058 -0.5870292 0.6164616 -0.9055740 0.9350064
## Apr 2021   0.0159530418 -0.6047793 0.6366854 -0.9333751 0.9652812
## May 2021   0.0154482802 -0.6229747 0.6538712 -0.9609354 0.9918319
## Jun 2021   0.0158907072 -0.6402019 0.6719833 -0.9875164 1.0192978
## Jul 2021   0.0156175123 -0.6575870 0.6888221 -1.0139600 1.0451950
## Aug 2021   0.0159793027 -0.6741841 0.7061428 -1.0395346 1.0714932
## Sep 2021   0.0159018728 -0.6907153 0.7225190 -1.0647758 1.0965795
## Oct 2021   0.0160813671 -0.7067821 0.7389448 -1.0894429 1.1216056
## Nov 2021   0.0160761967 -0.7227017 0.7548541 -1.1137871 1.1459395
## Dec 2021   0.0162259378 -0.7382453 0.7706972 -1.1376382 1.1700901
## Jan 2022   0.0162653460 -0.7536106 0.7861413 -1.1611584 1.1936891
## Feb 2022   0.0163729341 -0.7686965 0.8014423 -1.1842871 1.2170330
## Mar 2022   0.0164327608 -0.7835961 0.8164616 -1.2071058 1.2399713
```

Checking for the model Adequacy

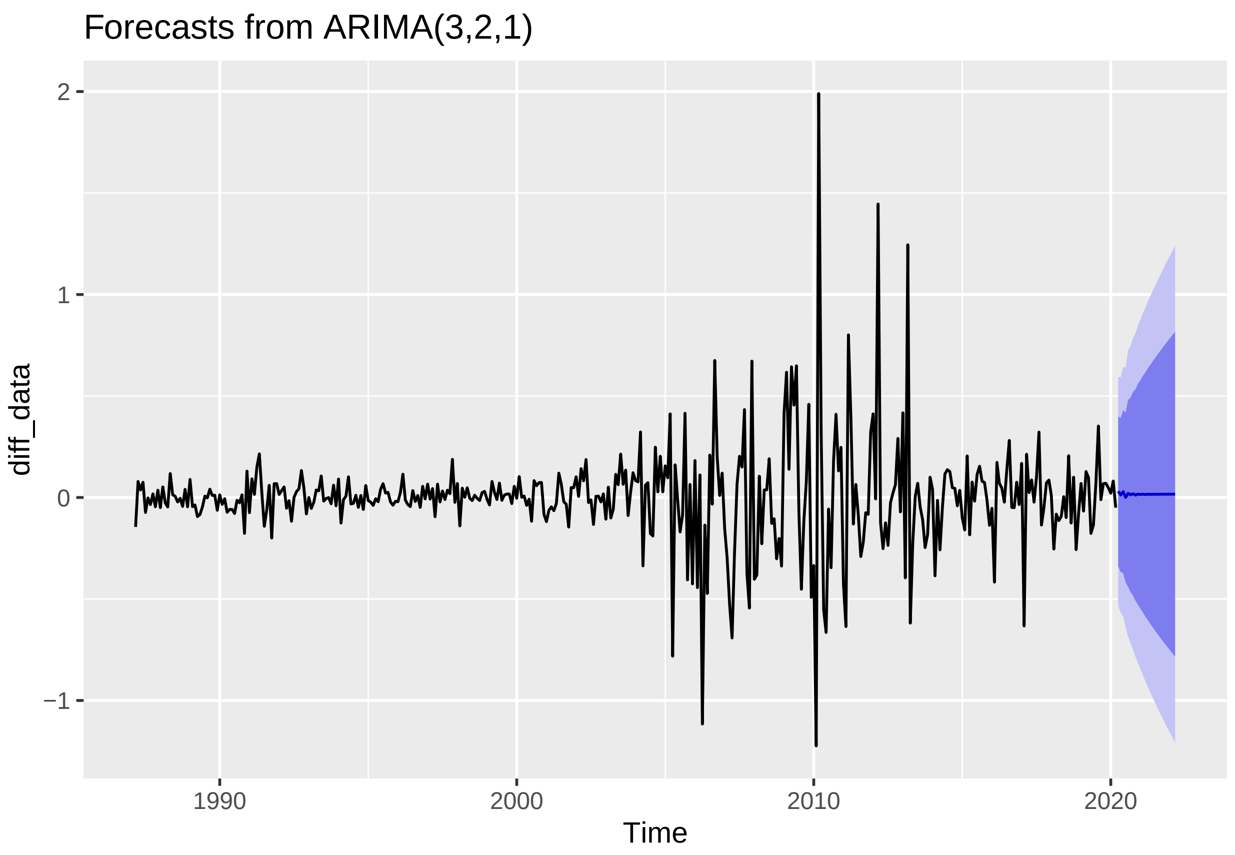
```
ggtsdiag(ARIMAmodel)
```

```
## Warning: `mutate_()` is deprecated as of dplyr 0.7.0.
## Please use `mutate()` instead.
## See vignette('programming') for more help
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```



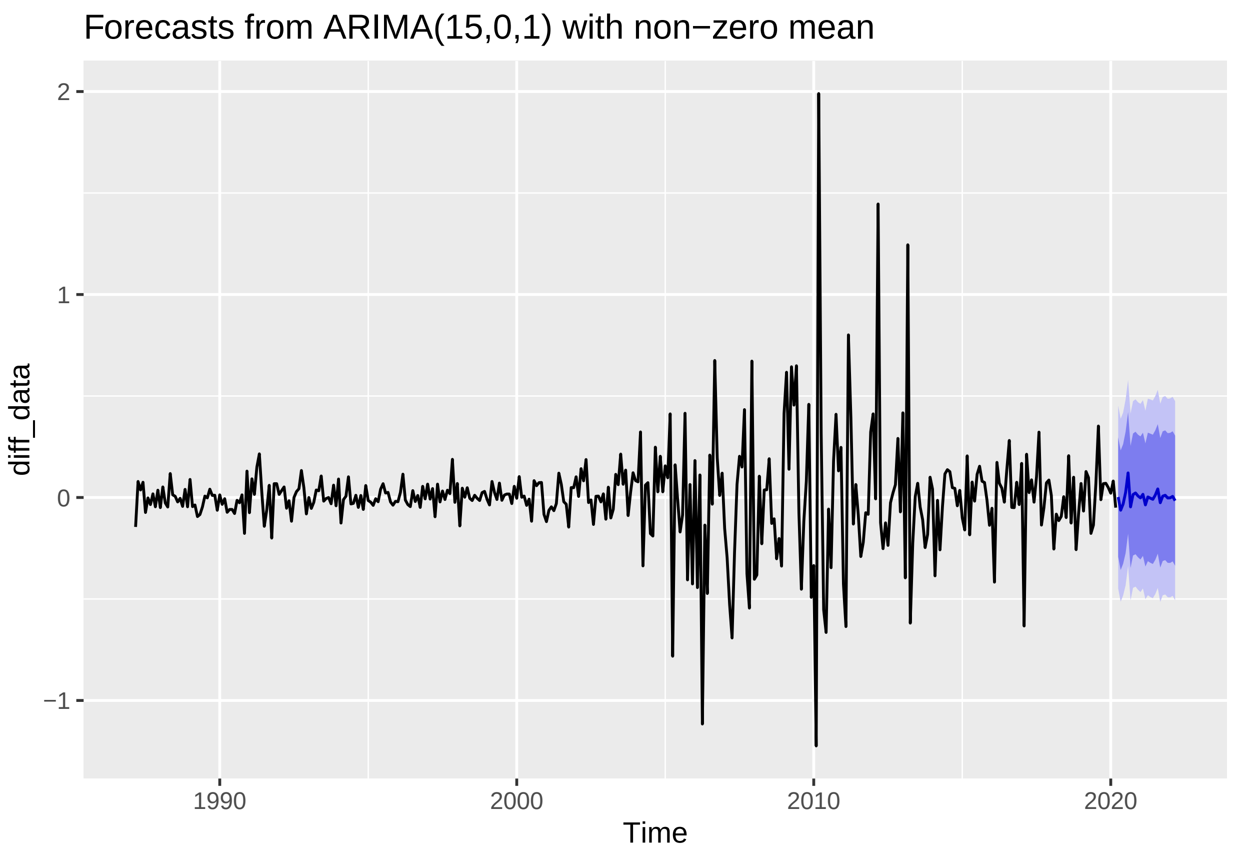
Forcasting the Arima Model

```
arima_fut <-autoplot(forecast(ARIMAmodel))
arima_fut
```



Predicting ARMA model for the next 3 years i.e future evolution

```
pred_fut <-autoplot(forecast(ARMAmodel))
pred_fut
```


Forecasts from ARIMA(15,0,1) with non−zero mean
diff_data
2
1
0
−1
1990
2000
2010
2020
Time