MATH1318 Time Series Analysis Assignment 2

Egg depositions (in millions) of age-3 Lake Huron Bloaters between years 1981 and 1996

Student Details

Student Name： Shuai Gao (s3596156)

Introduction

This assignment's task is to analyze the egg depositions of Lake Huron Bloasters by using the analysis methods and choose the best model among a set of possible models for this data-set and give forecasts of egg depositions for the next 5 years. The data-set is collect from FSAdata package, we will directly use the eggs data provide by this data-set.

Load Packages

This assignment is using TSA, forecast and lmtest package to construct the time series model, parameter estimate and to give predictions. The data is collect from FSAdata package in BloaterLH data-set. (package can be install by install.packages(‘’)).

```
library(TSA)
library(forecast)
library(FSAdata)
library(lmtest)
```

Data Input

Firstly, we input data-set BloaterLH from FSAdata package.

```
data("BloaterLH")
```

The egg depositions data are available in eggs column. We need to check the data frame firstly.

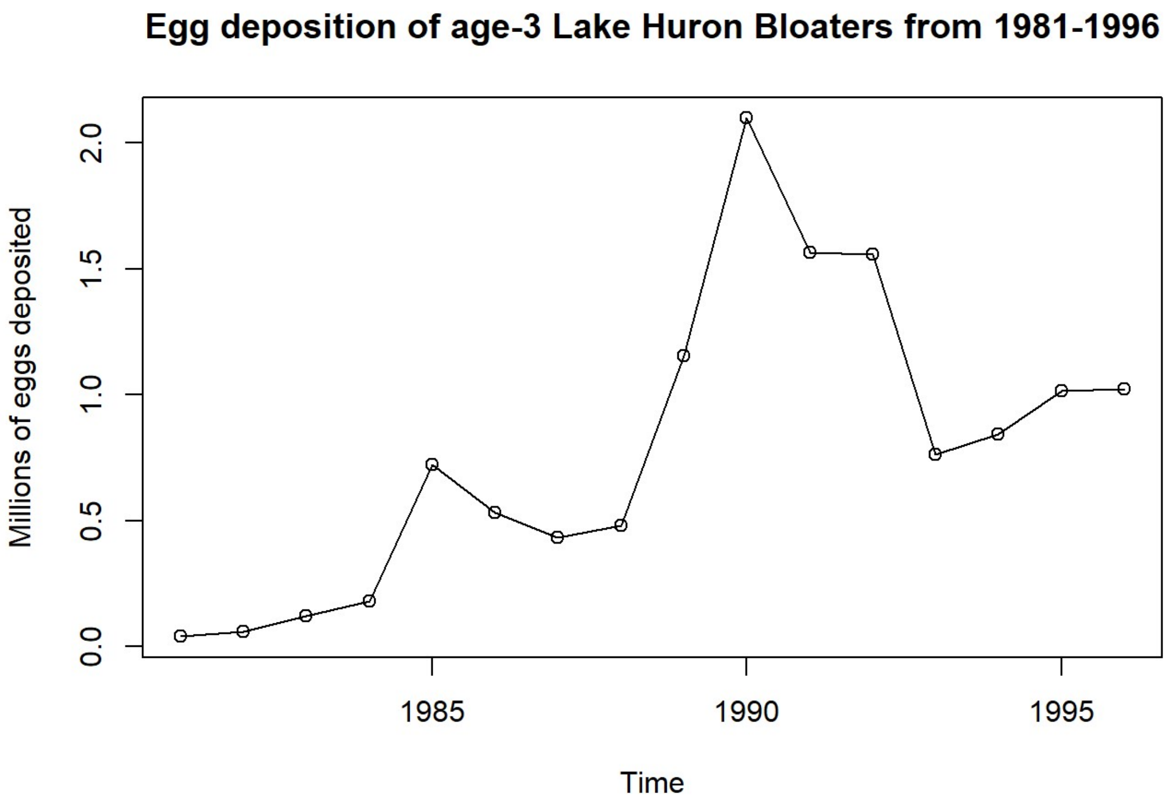
```
class(BloaterLH$eggs)
```

```
## [1] "numeric"
```

According to the output, we should cover the data to time series object by using ts() function. Then, the time series plot can be shown.

Time series plot

```
fish <-ts(BloaterLH$eggs,start = 1981) # Convert selected column which is eggs to t
he TS object!
plot(fish,type='o',ylab='Millions of eggs deposited',main="Egg deposition of age-3
Lake Huron Bloaters from 1981-1996") ## plot the time series plot
```



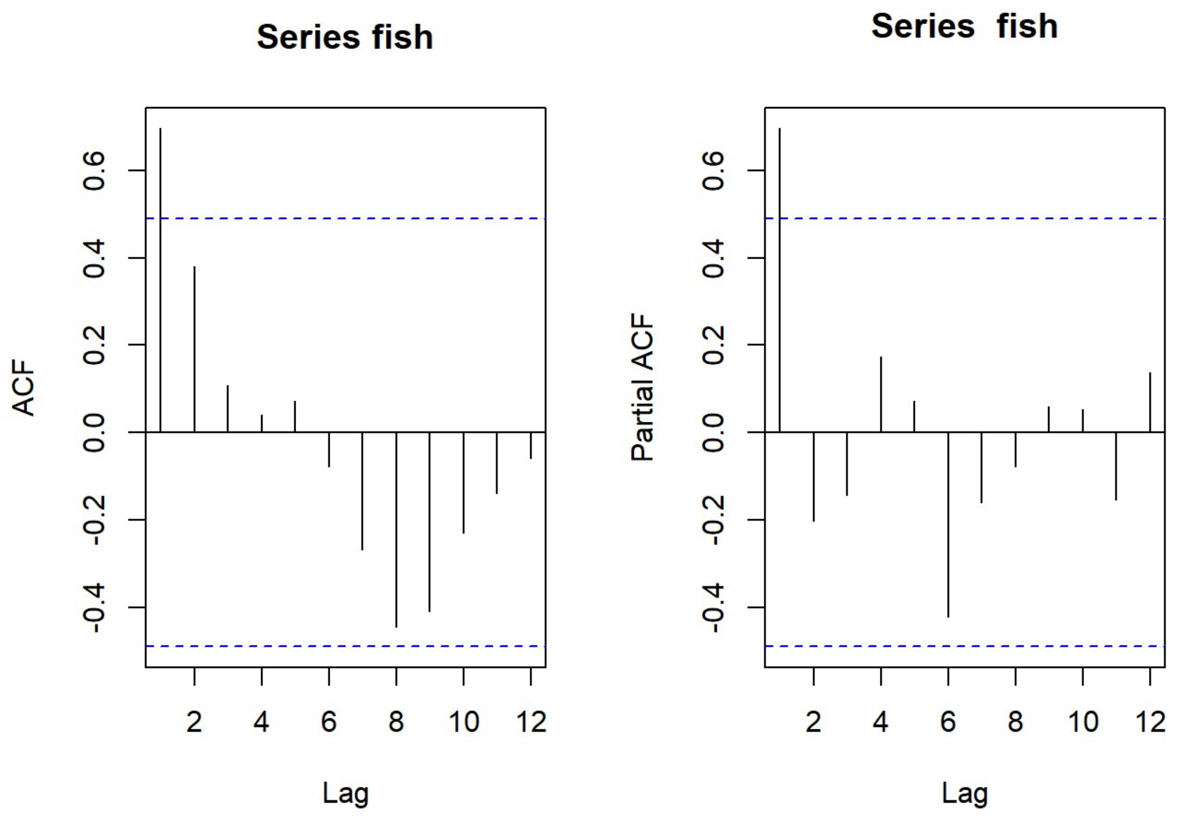
By the figure above, it is clearly to see there is trend. Although there are only 16 observations, two peaks appeared during this period and the latter is obviously higher than the former, there seems is changing variance. The succeeding observations imply the existence of auto-regressive. It would be a tricky task.

Data Prepare

Trend and Stationary Checking

ACF and PACF can conduct the preliminary verification of our hypothesis.

```
par(mfrow=c(1,2))
acf(fish)
pacf(fish)
```



Slowly decaying pattern in ACF and very high first correlation in PACF implies the existence of trend and non-stationary Which means we need do transformation and differencing to make series stationary.

Transformation

To prepare data, it is always right to make the data stationary first. Although the non-stationary can be seen by ACF and PACF, adf.test can gives another vertifacation. Then, the normality can be check by shapiro.test.

```
adf.test(fish)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  fish
## Dickey-Fuller = -2.0669, Lag order = 2, p-value = 0.5469
## alternative hypothesis: stationary
```

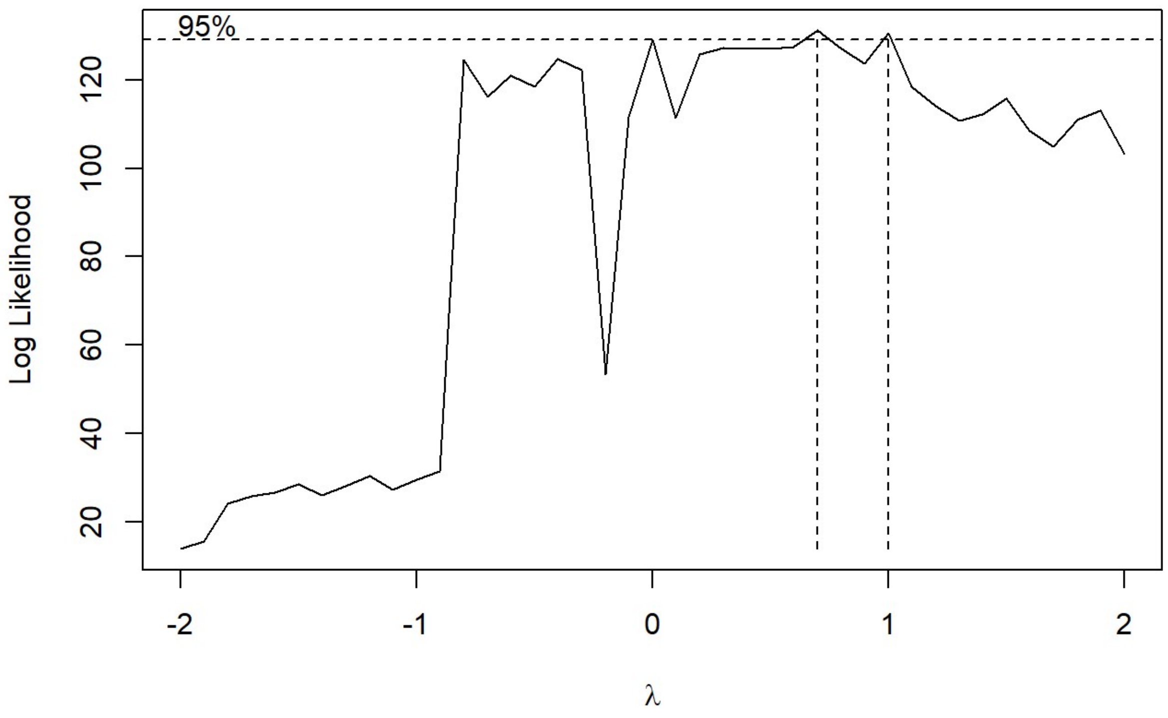
```
shapiro.test(fish)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  fish
## W = 0.94201, p-value = 0.3744
```

The p-value of Dickey-Fuller Test is not small enough to reject null hypothesis, which implies nonstationary here. The p-value of Shapiro-Wilk normality test is greater than 0.05 which means we cannot reject the null hypothesis of it is normality.

Before do differencing, if the transformations need or not should be check, the order does matter. Box-Cox transformations can give some advice.

```
p=BoxCox.ar(fish)
```

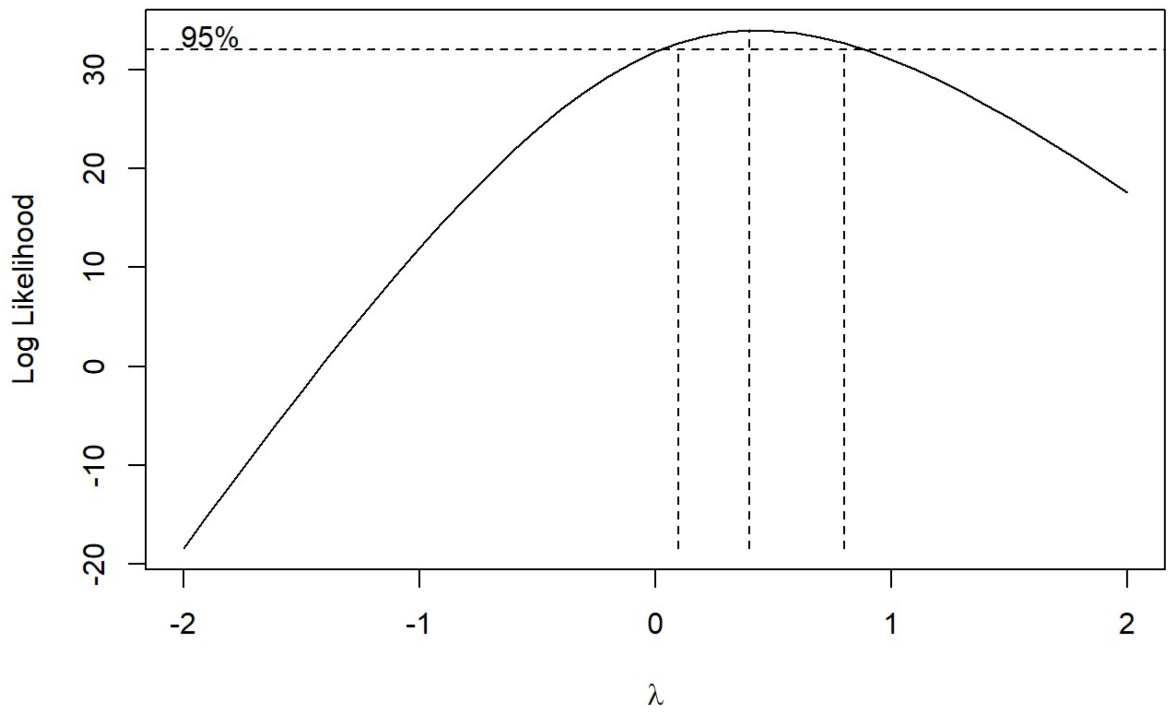


There are two peak in the figure and 1 is contained in λ. The advice is to use original data. We can also use the method to least squares or Method of Moments.

```
p=BoxCox.ar(fish,method = "yule-walker")
```

```
## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1

## Warning in arima0(x, order = c(i, 0L, 0L), include.mean = demean): possible
## convergence problem: optim gave code = 1
```



```
p$ci
```

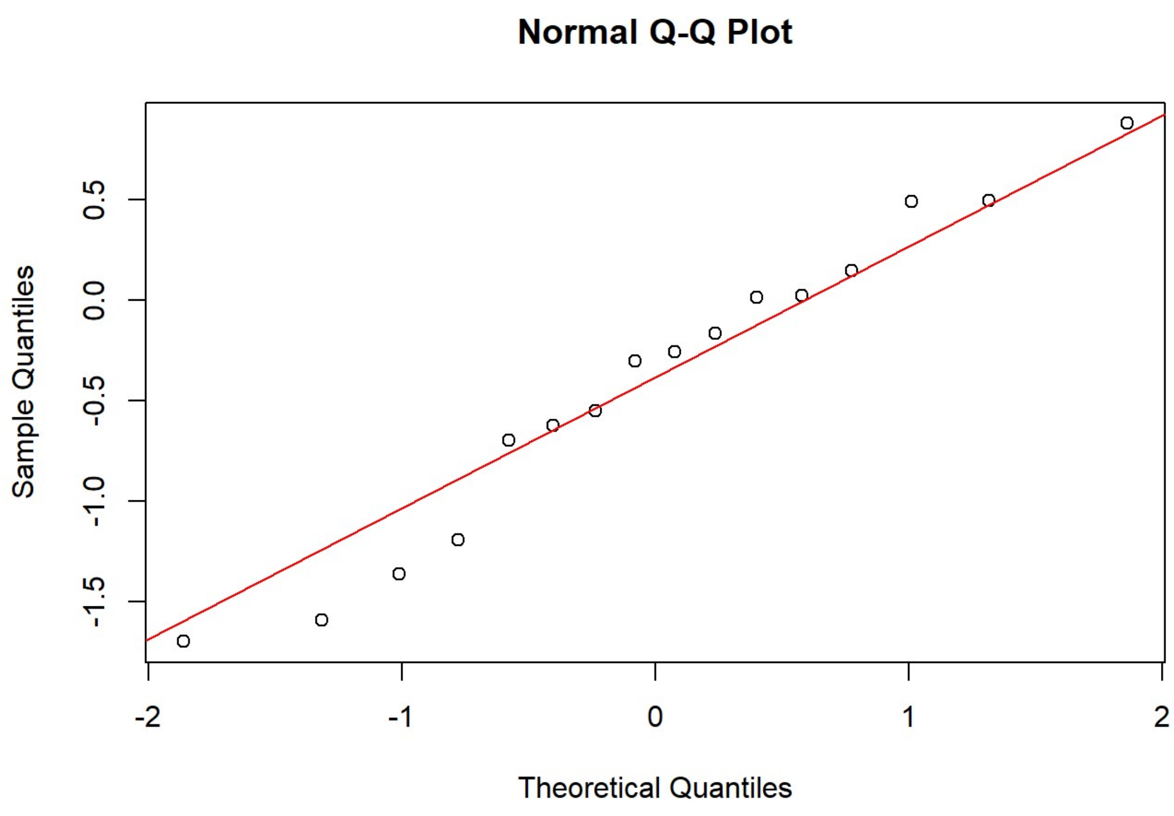
```
## [1] 0.1 0.8
```

The 95% confidence interval for λ contains the values of λ between 0.1 and nearly 0.8. The center of confidence interval corresponds to approximately 0.45, we will use it for Box-Cox transformation.

```
lambda=0.45
BC.fish = (fish^lambda-1)/lambda
```

Let’s check normality, after Box-Cox transformation.

```
qqnorm(BC.fish)
qqline(BC.fish,col=2)
```



```
shapiro.test(BC.fish)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  BC.fish
## W = 0.96269, p-value = 0.7107
```

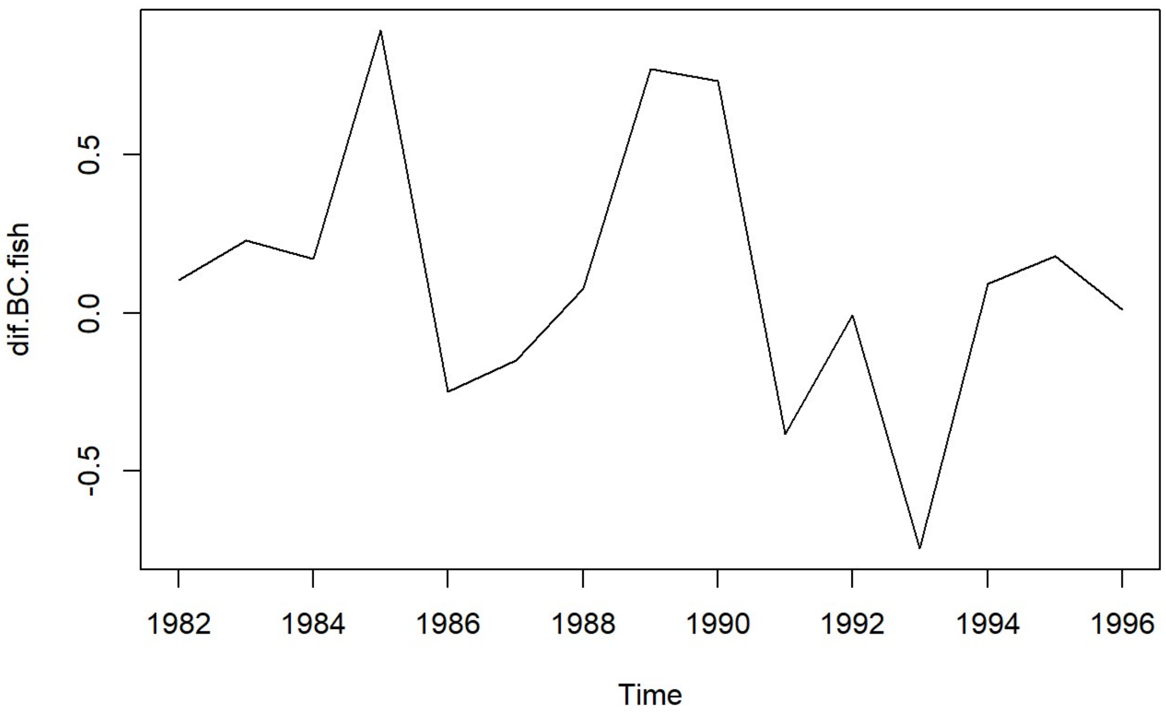
Box-Cox transformation helps to improve the series normality, since the dots are aligned around the red line and the p-value of Shapiro-Wilk normality test increased from 0.37 to 0.71.

Differencing

First Difference

After transformation, we can take first differences.

```
dif.BC.fish=diff(BC.fish)
plot(dif.BC.fish)
```



The trend still can be seen by the figure. Let’s check the stationary.

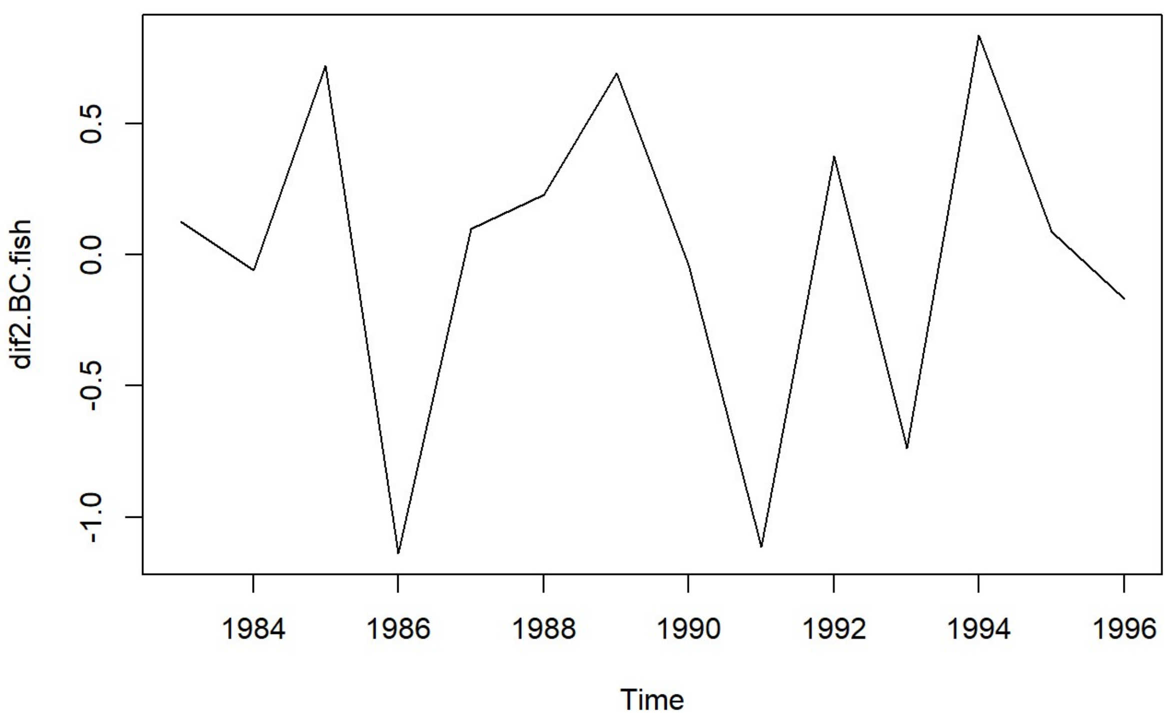
```
adf.test(dif.BC.fish,alternative = c("stationary"))
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  dif.BC.fish
## Dickey-Fuller = -3.6798, Lag order = 2, p-value = 0.0443
## alternative hypothesis: stationary
```

The p-value is smaller than 0.05 here which means we can reject non stationary hypothesis. However, the trend in the figure push us to try second difference.

Second Difference

```
dif2.BC.fish=diff(BC.fish,differences = 2)
plot(dif2.BC.fish)
```



The trend here is disappear, the series seems stationary here. Let us check stationary by adf.test.

```
adf.test(dif2.BC.fish,alternative = c("stationary"))
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  dif2.BC.fish
## Dickey-Fuller = -3.1733, Lag order = 2, p-value = 0.1254
## alternative hypothesis: stationary
```

The p-value is increased by second difference, it is uncommon. The trend has disappeared here, although the p-value is not small enough, consider the trend, we will use second difference.

Modeling

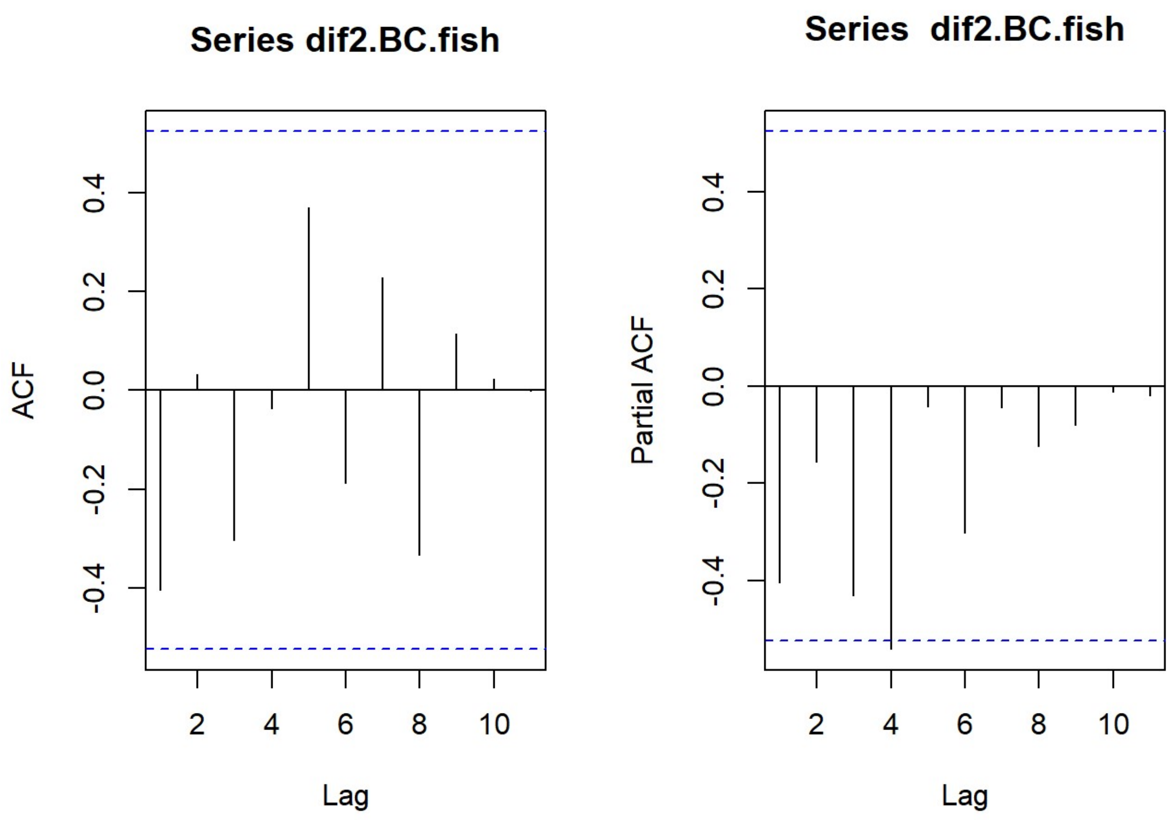
After getting rid of the changing variance and trend to make data stationary, we will do modeling next.

Model Specification

First of all, we would specify model by several approaches.

The Autocorrelation Function and The Partial Autocorrelation Function

```
par(mfrow=c(1,2))
acf(dif2.BC.fish)
pacf(dif2.BC.fish)
```



There are no significant legs in ACF, and we got one significant leg in pacf plot. We will consider ARIMA(1,2,0) model here.

The Extended Autocorrelation Function

We can also use EACF method to identify the order of autoregressive and moving average components of ARMA models.

```
eacf(dif2.BC.fish,ma.max = 3,ar.max = 3)
```

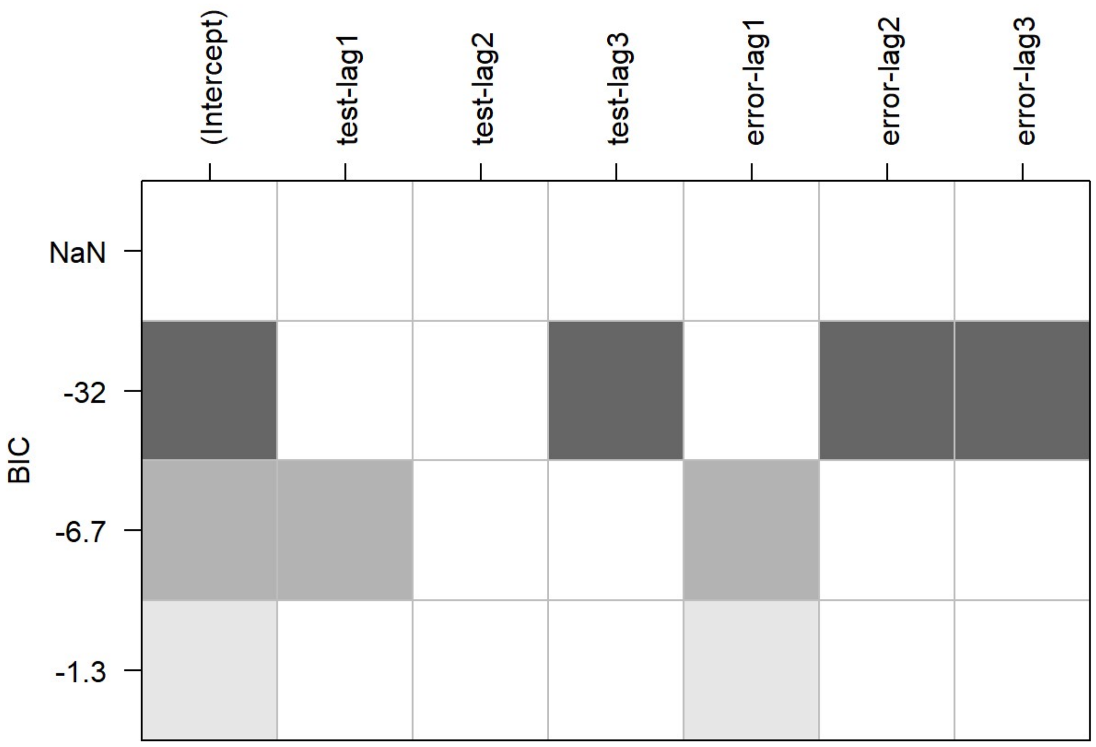
```
## AR/MA
##   0 1 2 3
## 0 o o o o
## 1 o o o o
## 2 o o o o
## 3 o o o o
```

The extended autocorrelation (EACF) method also implies existence of white noise behavior because the upper left point is (0,0). However, I will include neighbor point which are ARIMA(0,2,1) and ARIMA(1,2,0)(this one is same as the one we get from ACF and PACF).

Bayesian Information Cariterion

BIC function can give some subset ARMA models for us.

```
res=armasubsets(y=dif2.BC.fish,nar=3,nma=3,y.name='test', ar.method='ols')
plot(res)
```



In the BIC table shaded columns correspond to AR(1) and AR(3), for MA there are MA(1) MA(2). So, from here we can consider ARIMA(1,2,1),ARIMA(1,2,2), ARIMA(3,2,1) and ARIMA(3,2,2) models.

In conclusion the set of candidate models is ARIMA(1,2,0), ARIMA(0,2,1), ARIMA(1,2,1),ARIMA (1,2,2), ARIMA(3,2,1) and ARIMA(3,2,2).

Parameter Estimation

For parameter estimation, we will apply both maximum likelihood estimation and least squares estimates and then select the best model based on AIC and BIC.

- *ARIMA(1,2,0)*

```
model_120_css = arima(BC.fish,order=c(1,2,0),method='CSS')
coeftest(model_120_css)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ar1 -0.40609    0.24460 -1.6602  0.09687 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_120_ml = arima(BC.fish,order=c(1,2,0),method='ML')
coeftest(model_120_ml)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ar1 -0.38056    0.23493 -1.6199   0.1053
```

AR(1) coefficient insignificant in both CSS and ML method.

- *ARIMA(0,2,1)*

```
model_021_css = arima(BC.fish,order=c(0,2,1),method='CSS')
coeftest(model_021_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ma1 -1.093338   0.056532  -19.34 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_021_ml = arima(BC.fish,order=c(0,2,1),method='ML')
coeftest(model_021_ml)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ma1 -0.99994    0.66759 -1.4978   0.1342
```

MA(1) coefficient significant according to CSS and insignidicant according to ML.

- *ARIMA(1,2,1)*

```
model_121_css = arima(BC.fish,order=c(1,2,1),method='CSS')
coeftest(model_121_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1  0.030947   0.290460  0.1065    0.9152
## ma1 -1.021947   0.201926 -5.0610 4.171e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_121_ml = arima(BC.fish,order=c(1,2,1),method='ML')
coeftest(model_121_ml)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ar1  0.11474    0.26992  0.4251 0.670766
## ma1 -1.00000    0.33204 -3.0116 0.002598 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

AR(1) component is insignificant in both ML and CSS method.

- *ARIMA(1,2,2)*

```
model_122_css = arima(BC.fish,order=c(1,2,2),method='CSS')
coeftest(model_122_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
## ar1  0.329019   0.010368  31.7337 < 2.2e-16 ***
## ma1 -2.494136   0.107804 -23.1359 < 2.2e-16 ***
## ma2  1.189979   0.159874   7.4432 9.825e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_122_ml = arima(BC.fish,order=c(1,2,2),method='ML')
coeftest(model_122_ml)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value Pr(>|z|)
## ar1  0.022040   0.984320  0.0224   0.9821
## ma1 -0.904710   0.996567 -0.9078   0.3640
## ma2 -0.095272   0.937345 -0.1016   0.9190
```

All component in this model significant with CSS method and insignificant with ML method.

- *ARIMA(3,2,1)*

```
model_321_css = arima(BC.fish,order=c(3,2,1),method='CSS')
coeftest(model_321_css)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ar1 -0.35630    0.27597 -1.2911   0.1967
## ar2 -0.36505    0.23983 -1.5221   0.1280
## ar3 -0.58811    0.23679 -2.4837   0.0130 *
## ma1 -0.33814    0.34765 -0.9727   0.3307
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_321_ml = arima(BC.fish,order=c(3,2,1),method='ML')
coeftest(model_321_ml)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ar1 -0.13108    0.29378 -0.4462  0.65546
## ar2 -0.19970    0.24973 -0.7997  0.42389
## ar3 -0.41663    0.24785 -1.6810  0.09276 .
## ma1 -0.65245    0.30355 -2.1494  0.03160 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Only ma(1) significant in both methods.

- *ARIMA(3,2,2)*

```
model_322_css = arima(BC.fish,order=c(3,2,2),method='CSS')
coeftest(model_322_css)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error z value Pr(>|z|)
## ar1 -0.3542497  0.3078409 -1.1508  0.24983
## ar2 -0.3663173  0.2556207 -1.4331  0.15184
## ar3 -0.5904878  0.2859045 -2.0653  0.03889 *
## ma1 -0.3396930  0.3655742 -0.9292  0.35278
## ma2  0.0053446  0.3677907  0.0145  0.98841
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_322_ml = arima(BC.fish,order=c(3,2,2),method='ML')
coeftest(model_322_ml)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ar1  0.33699    0.29058  1.1597  0.24617
## ar2 -0.40949    0.20143 -2.0329  0.04206 *
## ar3 -0.53148    0.26218 -2.0272  0.04264 *
## ma1 -1.31664    0.80882 -1.6278  0.10356
## ma2  0.99930    1.13702  0.8789  0.37947
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

AR(3) is significant in CSS method, AR(2) and AR(3) are significant in ML method.

The models with all coefficients significant is ARIMA(1,2,2)-CSS.

Sort by AIC and BIC

```
sort.score <- function(x, score = c("bic", "aic")){
  if (score == "aic"){
    x[with(x, order(AIC)),]
  } else if (score == "bic") {
    x[with(x, order(BIC)),]
  } else {
    warning('score = "x" only accepts valid arguments ("aic","bic")')
  }
}
```

This is a function to sort model by their AIC and BIC, by this function we can easily get the best model supported by BIC and AIC.

```
sort.score(AIC(model_120_ml,model_021_ml,model_121_ml,model_122_ml,model_321_ml,mod
el_322_ml), score = "aic")
```

	df <dbl>	AIC <dbl>
model_021_ml	2	23.12446
model_121_ml	3	24.94140
model_321_ml	5	26.92808
model_122_ml	4	26.93108
model_120_ml	2	27.05798
model_322_ml	6	28.01509

6 rows

```
sort.score(BIC(model_120_ml,model_021_ml,model_121_ml,model_122_ml,model_321_ml,mod
el_322_ml), score = "bic" )
```

	df <dbl>	BIC <dbl>
model_021_ml	2	24.40257
model_121_ml	3	26.85857
model_120_ml	2	28.33609
model_122_ml	4	29.48731
model_321_ml	5	30.12337
model_322_ml	6	31.84943

6 rows

Both AIC and BIC select ARIMA(0,2,1) model for this series.

Overfitting

ARIMA(1,2,1) is an overfitting model for ARIMA(0,2,1) and we find insignificant AR(1) according to MLE and CSS method.

Another overfitting model is ARIMA(0,2,2), which is fitted below to check the overfitting:

- *ARIMA(0,2,2)*

```
model_022_css = arima(BC.fish,order=c(0,2,2),method='CSS')
coeftest(model_022_css)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ma1 -1.017026   0.274754 -3.7016 0.0002143 ***
## ma2 -0.087178   0.308221 -0.2828 0.7772974
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
model_022_ml = arima(BC.fish,order=c(0,2,2),method='ML')
coeftest(model_022_ml)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error z value Pr(>|z|)
## ma1 -0.88464    0.42637 -2.0748   0.0380 *
## ma2 -0.11535    0.25484 -0.4526   0.6508
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Select the Best Model

```
sort.score(AIC(model_120_ml,model_121_ml,model_122_ml,model_321_ml,model_322_ml,mod
el_021_ml,model_022_ml), score = "aic")
```

	df <dbl>	AIC <dbl>
model_021_ml	2	23.12446
model_022_ml	3	24.93160
model_121_ml	3	24.94140
model_321_ml	5	26.92808
model_122_ml	4	26.93108
model_120_ml	2	27.05798
model_322_ml	6	28.01509

7 rows

```
sort.score(BIC(model_120_ml,model_121_ml,model_122_ml,model_321_ml,model_322_ml,mod
el_021_ml,model_022_ml), score = "bic" )
```

	df <dbl>	BIC <dbl>

	df <dbl>	BIC <dbl>
model_021_ml	2	24.40257
model_022_ml	3	26.84877
model_121_ml	3	26.85857
model_120_ml	2	28.33609
model_122_ml	4	29.48731
model_321_ml	5	30.12337
model_322_ml	6	31.84943

7 rows

Both AIC and BIC choose ARIMA(0,2,1) as the best model, we can confirm it is a suitable model for forecasting.

Model Diagnostics

For diagnostics, only for best selected model first, if it does not work well with the residuals, we will change to another model. we will analysis the behaviour of Standardised residuals based on normality and autocorrelation. We will do the Ljung-Box test as well. Then, consider the test results to verify that if the model is feasible.

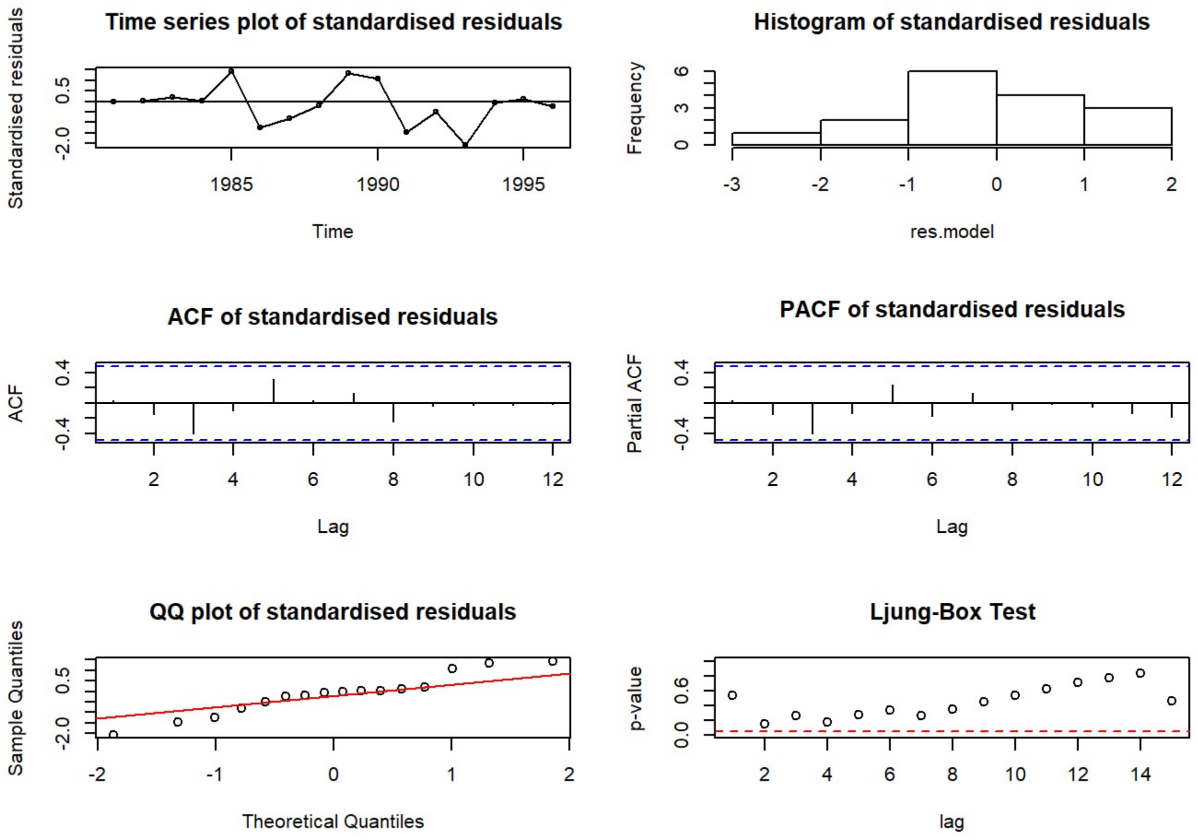
```r
residual.analysis <- function(model, std = TRUE,start = 2, class = c("ARIMA","GARC
H","ARMA-GARCH")[1]){
  # If you have an output from arima() function use class = "ARIMA"
  # If you have an output from garch() function use class = "GARCH"
  # If you have an output from ugarchfit() function use class = "ARMA-GARCH"
  library(TSA)
  library(FitAR)
  if (class == "ARIMA"){
    if (std == TRUE){
      res.model = rstandard(model)
    }else{
      res.model = residuals(model)
    }
  }else if (class == "GARCH"){
    res.model = model$residuals[start:model$n.used]
  }else if (class == "ARMA-GARCH"){
      res.model = model@fit$residuals
  }else {
    stop("The argument 'class' must be either 'ARIMA' or 'GARCH' ")
  }
  par(mfrow=c(3,2))
  plot(res.model,type='o',ylab='Standardised residuals', main="Time series plot of
standardised residuals")
  abline(h=0)
  hist(res.model,main="Histogram of standardised residuals")
  acf(res.model,main="ACF of standardised residuals")
  pacf(res.model,main="PACF of standardised residuals")
  qqnorm(res.model,main="QQ plot of standardised residuals")
  qqline(res.model, col = 2)
  print(shapiro.test(res.model))
  k=0
  LBQPlot(res.model, lag.max = length(model$residuals)-1 , StartLag = k + 1, k =
0, SquaredQ = FALSE)

}
```

The function above is for model diagnostics by residual analysis.

```r
residual.analysis(model = model_021_ml,class = "ARIMA")
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res.model
## W = 0.94991, p-value = 0.4883
```



-The first figure above shows the standardised residuals plot, there are no trend, no changing variance, which supports ARIMA(0,2,1) model.

-The second figure is the histogram of standardised residuals, the distribution here is similar as normal distribution, we cannot reject normality here.

-The third one is qq plot of standardised residuals, the majority points are aligned around the red line with few of them on the two side seems suspect. However the dataset have a small number of observations and p-value of Shapiro-Wilk normality test is 0.4883, which means we would not reject normality here.

-The fourth figure shows autocorrelation of the residuals, there is no evidence of autocorrelation in the residuals can be seen which can be treat implied white noise and support our model.

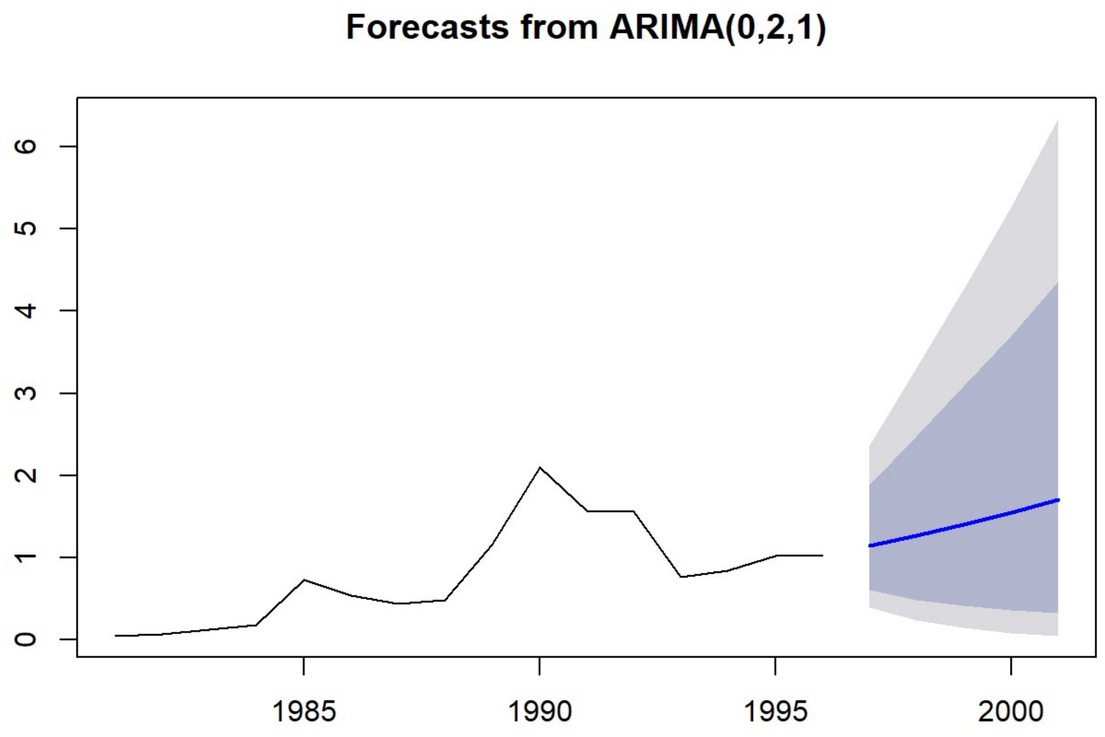
-The last one shows Ljung-Box test, according to the figure, all point here allocated above the red dashed line at 5% which support our model as well.

In conclusion, the model diagnostics result shows ARIMA(0,2,1) model seems work quite well with egg depositions (in millions) of age-3 Lake Huron Bloaters between data in different years.

Forecast

In this section, we will fit ARIMA(0,2,1) to forecast next 5 years' egg depositions.

```
fit = Arima(fish,c(0,2,1), lambda = 0.45)
plot(forecast(fit,h=5))
```



From the figure above, the next 5 years' egg depositions will float in blue region(80%) or gray region (95%).

Summary

In this assignment, in order to choose the best model to predict next 5 years' egg depositions, transforming and differentiating were used to make series stationary. Then, by using ACR, PACF, EACF and BIC for model specification, we find that both CSS and ML menthods were used for parameter estimation and selection of best model by aic and bic score. Finally, after model diagnostics by analyse residual, we choose ARIMA(0,2,1) for forecasting and showing the egg depositions of the next 5 years.