# Homework8

*Megan Jasek, Rohan Thakur, Charles Kekeh*

*Tuesday, April 5, 2016*

## Part 1 - Examining and Visualizing the Series

**Key Takeaways:**

1. The series has 372 units of time - possibly some kind of monthly data for 31 years.
2. Time series plot shows that the series is very persistent, strongly trending upwards.
3. Histogram shows no clear distribution - does not give us much information with the time component left out
4. ACF of the series very strongly resembles that of a random walk with drift - with correlations at around 0.8 for almost 25 lags (if this is indeed monthly data, that is almost 2 years!)
5. PACF drops off immediately after first lag

At initial glance, the series strongly resembles a random walk with drift.

```
series = ts(read.csv("hw08_series.csv", header = TRUE))
# removing extra column

series = series[, c("x")]

# Describing Series
str(series)
```

```
##  Time-Series [1:372] from 1 to 372: 40.6 41.1 40.5 40.1 40.4 41.2 39.3 41.6 42.3 43.2 ...
```

```
summary(series)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   36.00   57.38   76.45   84.83  111.50  152.60
```

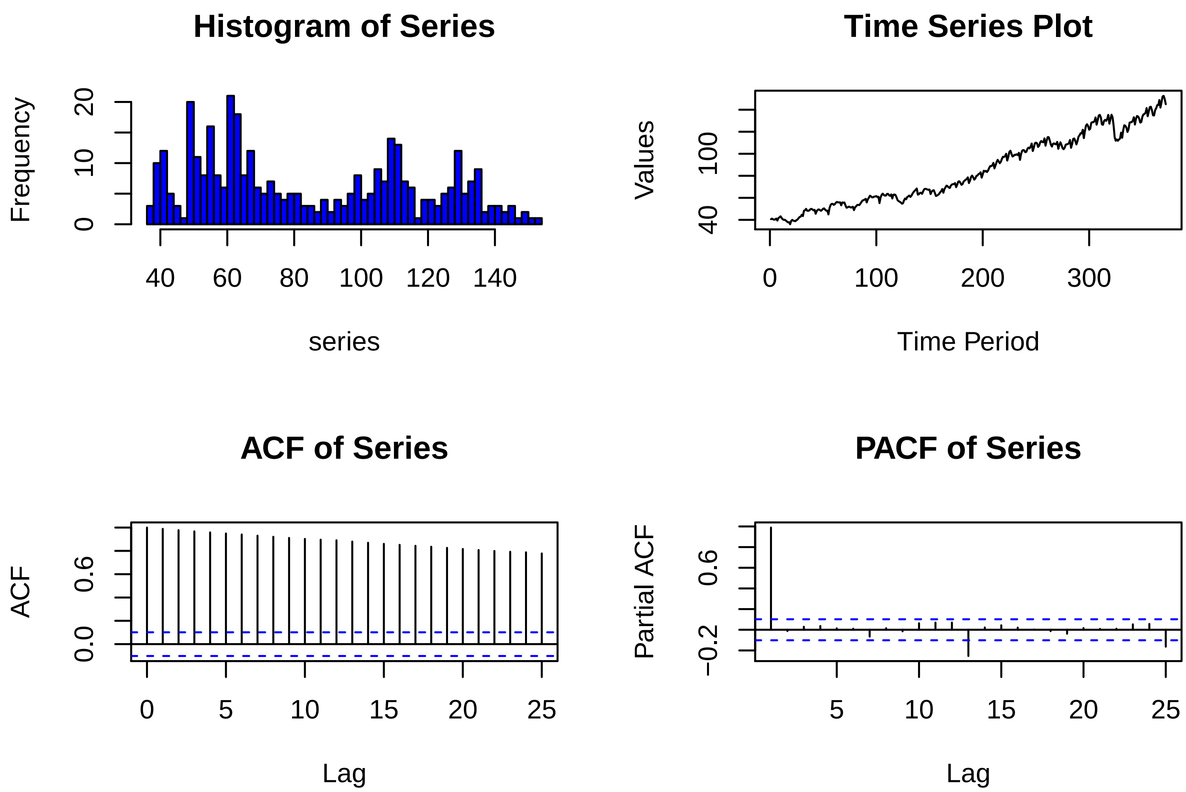
```
cbind(head(series), tail(series))
```

```
##      [,1]  [,2]
## [1,] 40.6 141.9
## [2,] 41.1 146.9
## [3,] 40.5 152.0
## [4,] 40.1 152.6
## [5,] 40.4 149.7
## [6,] 41.2 145.0
```

```
quantile(as.numeric(series), c(0.01, 0.05, 0.1, 0.25, 0.5, 0.75,
    0.9, 0.95, 0.99))
```

```
##      1%      5%     10%     25%     50%     75%     90%     95%     99%
##  38.713  40.555  48.300  57.375  76.450 111.525 130.750 135.590 147.451
```

```r
# Plot in order to see what is going on
par(mfrow = c(2, 2))
hist(series, breaks = 60, col = "blue", main = "Histogram of Series")
plot.ts(series, main = "Time Series Plot", ylab = "Values", xlab = "Time Period")
acf(series, main = "ACF of Series")
pacf(series, main = "PACF of Series")
```




## Part 2 - Estimating Models and Examining Residuals

**Key Takeaways:** We estimated various AR and ARMA models, and chose AR(1) as best representing the series according to AIC value and independence of residuals.

We tested 2 assumptions for this model:

1. Independence of Residuals: We can take this assumption to hold, as upon running the Ljung Box Test, we were unable to reject the null at the 5% level, that residuals are independently distributed.
2. Stationarity: The process is not stationary, since its root = 1. This is also evident from visual inspection of the graph, as we can see that it persistently trends upwards, and so the mean cannot be stationary.

**Detailed Results** Since there is no reversion to the mean, we decided to ignore pure MA models, and go ahead with tests for AR and ARMA models. The following 4 models were estimated:

1. AR(12): Estimation using the ar() function using MLE gave us an AR model of order 12. Aside from having higher AIC than other models, this model had a large coefficient for the first lag term but very

small coefficients for subsequent lag terms. Therefore, we choose to pursure parsimony and ignore this model.

2. AR(1): This model gave us the lowest AIC, along with a Ljung Box test that failed to reject the null hypothesis at the 5% level, that residuals are independently distributed. It will be our choice moving forward. The fitted value versus residuals plot did not show any clear trend, although there did seem to be increasing variance along with the passage of time.
3. ARMA(1,1) and ARMA(2,2): Both models showed higher AIC values than the AR(1). For both models, upon running the Ljung Box Test, we were able to reject the null hypothesis that the residuals are independent, at the 5% level. Therefore, we do not continue with these models.

```r
# Since there is no reversion to the mean, an MA model is
# probably not a great fit for this series. We will attempt
# to fit AR and ARMA models to the series.

# Try fitting the model to an AR
series.model = ar(series, method = "mle")
series.model
```

```
##
## Call:
## ar(x = series, method = "mle")
##
## Coefficients:
##       1        2        3        4        5        6        7        8
##  0.7795   0.0091   0.1034   0.1859   0.0578  -0.0459   0.0271  -0.1118
##       9       10       11       12
## -0.1795  -0.1012  -0.0123   0.2872
##
## Order selected 12  sigma^2 estimated as  5.548
```

```r
series.model$aic
```

```
##          0          1          2          3          4          5
## 1906.84529   69.65890   70.17796   57.66203   49.70815   51.68971
##          6          7          8          9         10         11
##   84.98875   54.76859   48.52034   49.06466   46.57030   28.58301
##         12
##    0.00000
```

```r
sqrt(series.model$asy.var)
```

```
## Warning in sqrt(series.model$asy.var): NaNs produced
```

```
##               [,1]        [,2]        [,3]         [,4]        [,5]
##  [1,] 0.0262515619         NaN 0.005187624 0.0007261853         NaN
##  [2,]          NaN 0.037049330         NaN 0.0054784706 0.003833045
##  [3,] 0.0051876235         NaN 0.037063087          NaN 0.005435983
##  [4,] 0.0007261853 0.005478471         NaN 0.0370102530         NaN
##  [5,]          NaN 0.003833045 0.005435983          NaN 0.037005208
##  [6,] 0.0015707431         NaN 0.003795177 0.0050249925         NaN
##  [7,]          NaN 0.007115888         NaN 0.0042747376 0.004974349
##  [8,] 0.0075547501         NaN 0.007269663          NaN 0.004128981
```

```
##  [9,]          NaN 0.008855783         NaN 0.0073497387         NaN
## [10,] 0.0072066281         NaN 0.008969362          NaN 0.007269663
## [11,] 0.0021566815 0.007012286         NaN 0.0088557833         NaN
## [12,]          NaN 0.002156681 0.007206628          NaN 0.007554750
##              [,6]        [,7]        [,8]         [,9]       [,10]
##  [1,] 0.001570743         NaN 0.007554750          NaN 0.007206628
##  [2,]         NaN 0.007115888         NaN 0.0088557833         NaN
##  [3,] 0.003795177         NaN 0.007269663          NaN 0.008969362
##  [4,] 0.005024992 0.004274738         NaN 0.0073497387         NaN
##  [5,]         NaN 0.004974349 0.004128981          NaN 0.007269663
##  [6,] 0.036941406         NaN 0.004974349 0.0042747376         NaN
##  [7,]         NaN 0.036941406         NaN 0.0050249925 0.003795177
##  [8,] 0.004974349         NaN 0.037005208          NaN 0.005435983
##  [9,] 0.004274738 0.005024992         NaN 0.0370102530         NaN
## [10,]         NaN 0.003795177 0.005435983          NaN 0.037063087
## [11,] 0.007115888         NaN 0.003833045 0.0054784706         NaN
## [12,]         NaN 0.001570743         NaN 0.0007261853 0.005187624
##             [,11]        [,12]
##  [1,] 0.002156681          NaN
##  [2,] 0.007012286 0.0021566815
##  [3,]         NaN 0.0072066281
##  [4,] 0.008855783          NaN
##  [5,]         NaN 0.0075547501
##  [6,] 0.007115888          NaN
##  [7,]         NaN 0.0015707431
##  [8,] 0.003833045          NaN
##  [9,] 0.005478471 0.0007261853
## [10,]         NaN 0.0051876235
## [11,] 0.037049330          NaN
## [12,]         NaN 0.0262515619
```

```r
# We get a model of order 12. Seems like this is overfitting
# the data, and we could easily do with fewer parameters.
series.model2 = arima(series, order = c(1, 0, 0))
series.model2
```

```
##
## Call:
## arima(x = series, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.9982    90.6882
## s.e.  0.0021    39.1616
##
## sigma^2 estimated as 7.145:  log likelihood = -896.41,  aic = 1798.83
```

```r
# We try and fit an ARMA model to check for a MA piece
series.model3 <- arima(series, order = c(1, 0, 1))
```

```
## Warning in arima(series, order = c(1, 0, 1)): possible convergence problem:
## optim gave code = 1
```

```r
series.model3
```

```
##
## Call:
## arima(x = series, order = c(1, 0, 1))
##
## Coefficients:
##          ar1     ma1  intercept
##       0.9982  0.0745    97.1995
## s.e.  0.0025  0.0622    43.7234
##
## sigma^2 estimated as 7.249:  log likelihood = -899.18,  aic = 1806.36
```

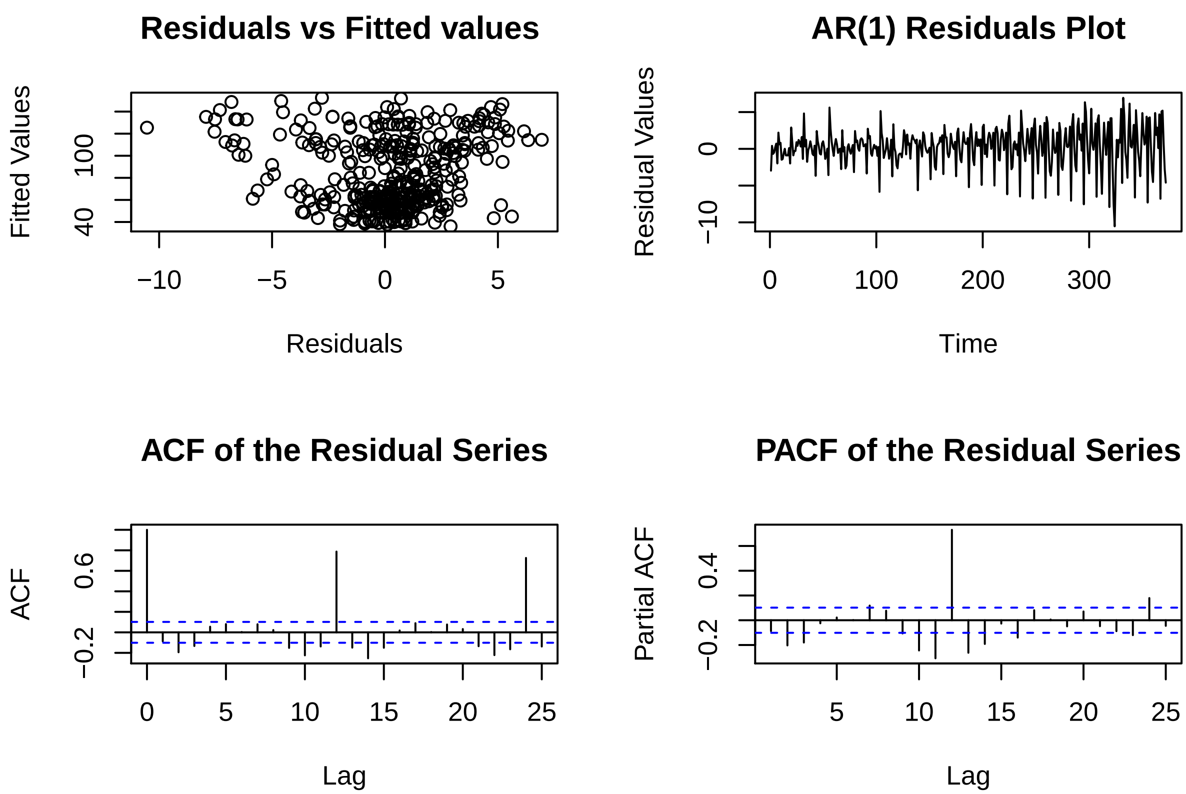
```r
# We try and fit an ARMA model to check for a MA piece
series.model4 = arima(series, order = c(2, 0, 2), method = "ML")
series.model4
```

```
##
## Call:
## arima(x = series, order = c(2, 0, 2), method = "ML")
##
## Coefficients:
##         ar1     ar2     ma1     ma2   intercept
##       1e-04  0.9999  1.0563  0.0593     84.9329
## s.e.  0e+00  0.0000  0.0143  0.0145  12666.4139
##
## sigma^2 estimated as 6.716:  log likelihood = -883.77,  aic = 1779.54
```

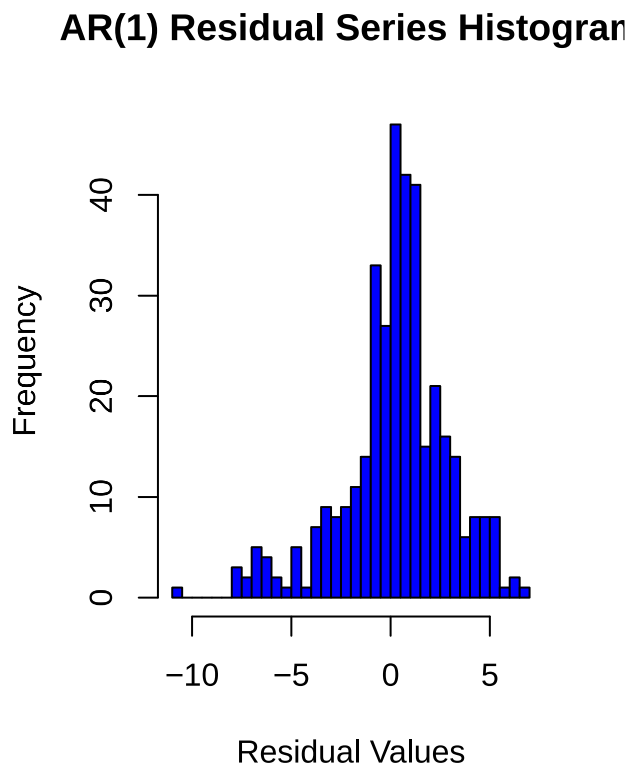
```r
# Now, examining Residuals

# AR(1)
head(series.model2$resid)
```

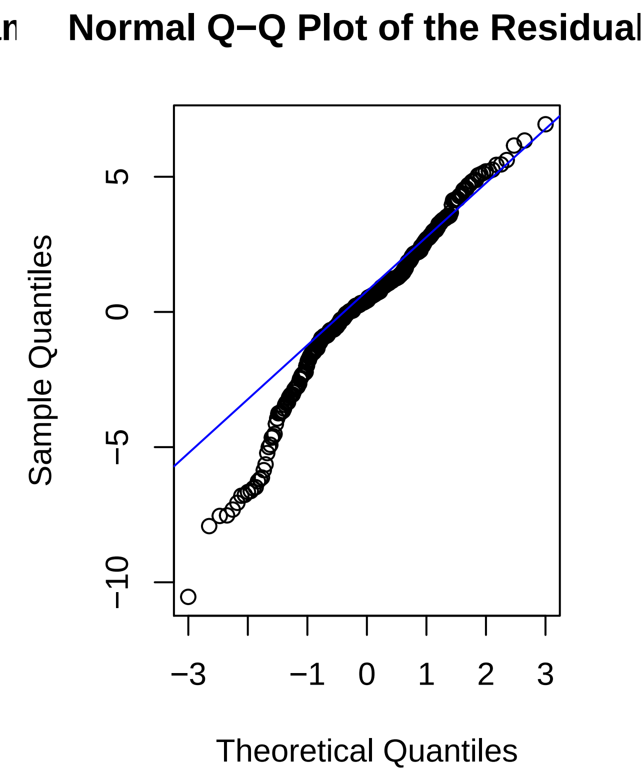
```
## [1] -2.9704552  0.4118418 -0.6872782 -0.4883342  0.2109618  0.7114898
```

```r
par(mfrow = c(2, 2))
plot(series.model2$resid, fitted(series.model2), main = "Residuals vs Fitted values ",
    xlab = "Residuals", ylab = "Fitted Values")
plot(series.model2$resid, type = "l", main = "AR(1) Residuals Plot",
    xlab = "Time", ylab = "Residual Values")
acf(series.model2$resid, main = "ACF of the Residual Series")
pacf(series.model2$resid, main = "PACF of the Residual Series")
```




```r
par(mfrow = c(1, 2))
hist(series.model2$resid, breaks = 60, col = "blue", main = "AR(1) Residual Series Histogram",
    xlab = "Residual Values")
qqnorm(series.model2$resid, main = "Normal Q-Q Plot of the Residuals",
    type = "p")
qqline(series.model$resid[-1], col = "blue")
```

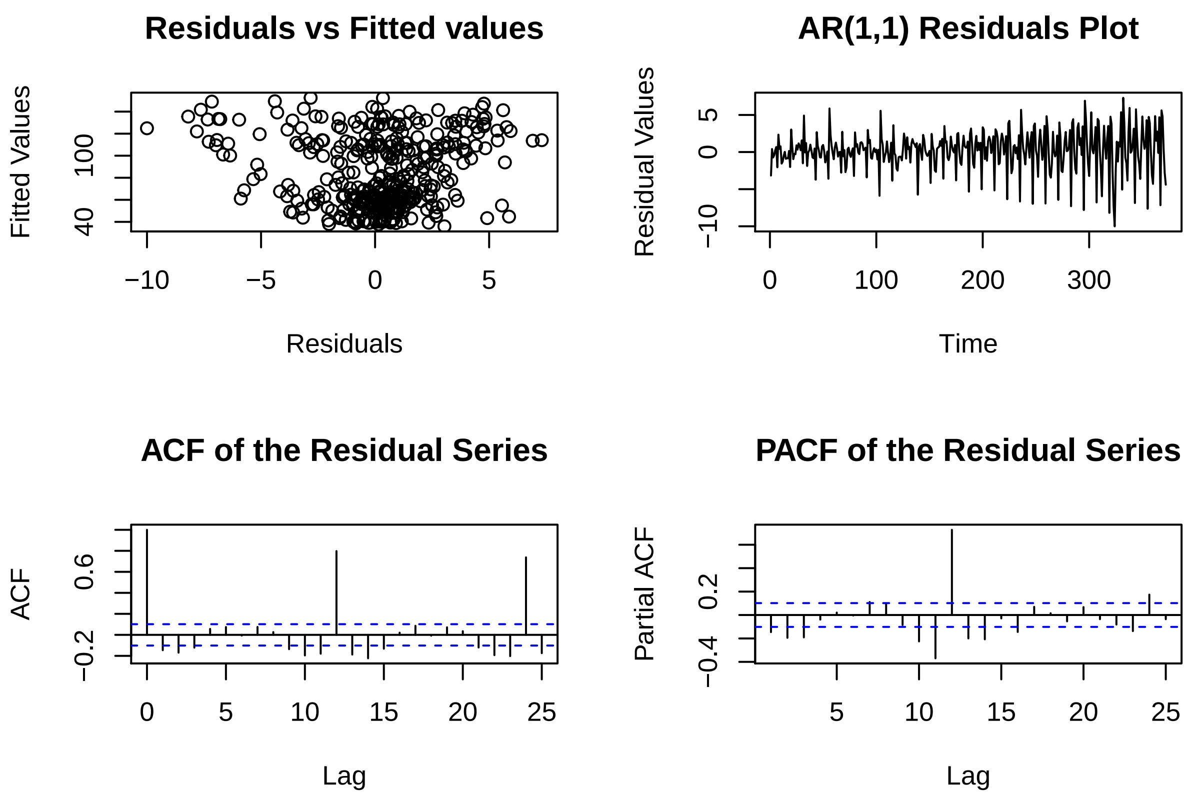


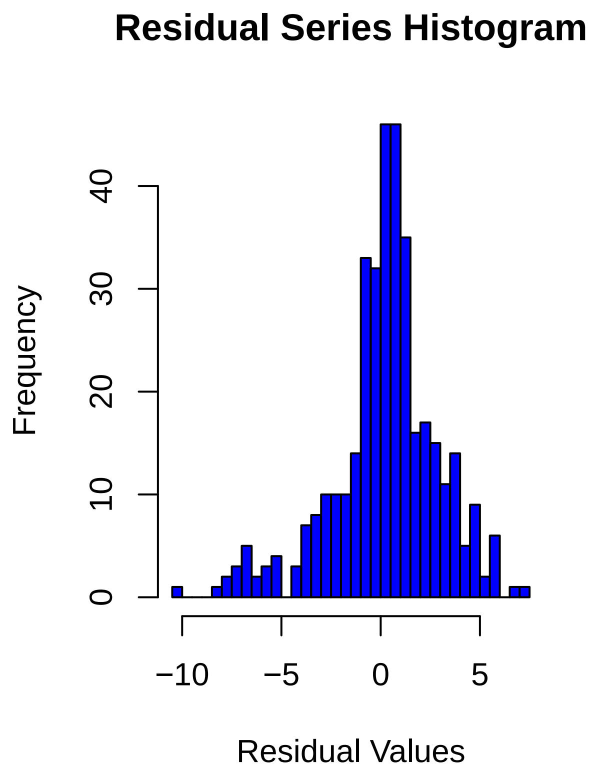

```r
# ARMA(1,1)
head(series.model3$resid)
```

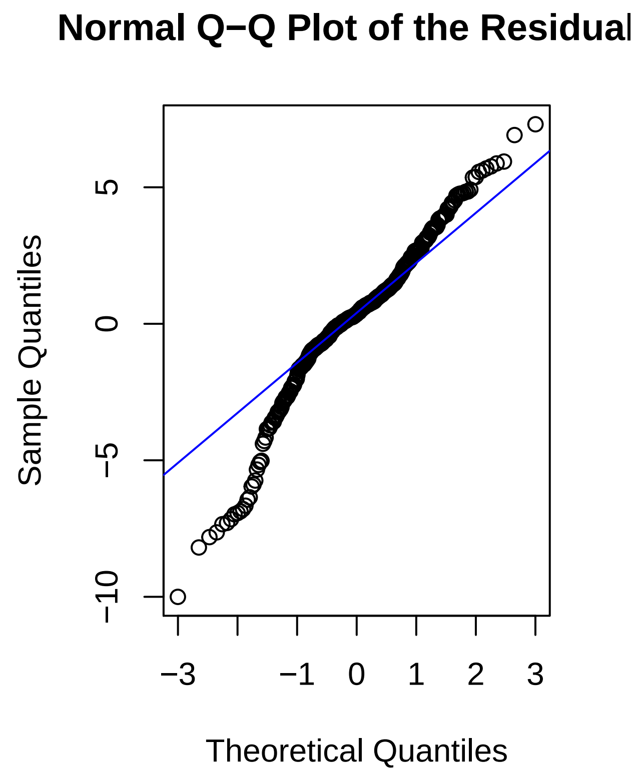
```
## [1] -3.1646126  0.4098372 -0.7317536 -0.4478718  0.2302542  0.6802679
```

```r
par(mfrow = c(2, 2))
plot(series.model3$resid, fitted(series.model3), main = "Residuals vs Fitted values",
    xlab = "Residuals", ylab = "Fitted Values")
plot(series.model3$resid, type = "l", main = "AR(1,1) Residuals Plot",
    xlab = "Time", ylab = "Residual Values")
acf(series.model3$resid, main = "ACF of the Residual Series")
pacf(series.model3$resid, main = "PACF of the Residual Series")
```




```
par(mfrow = c(1, 2))
hist(series.model3$resid, breaks = 60, col = "blue", main = "Residual Series Histogram",
    xlab = "Residual Values")
qqnorm(series.model3$resid, main = "Normal Q-Q Plot of the Residuals",
    type = "p")
qqline(series.model3$resid[-1], col = "blue")
```

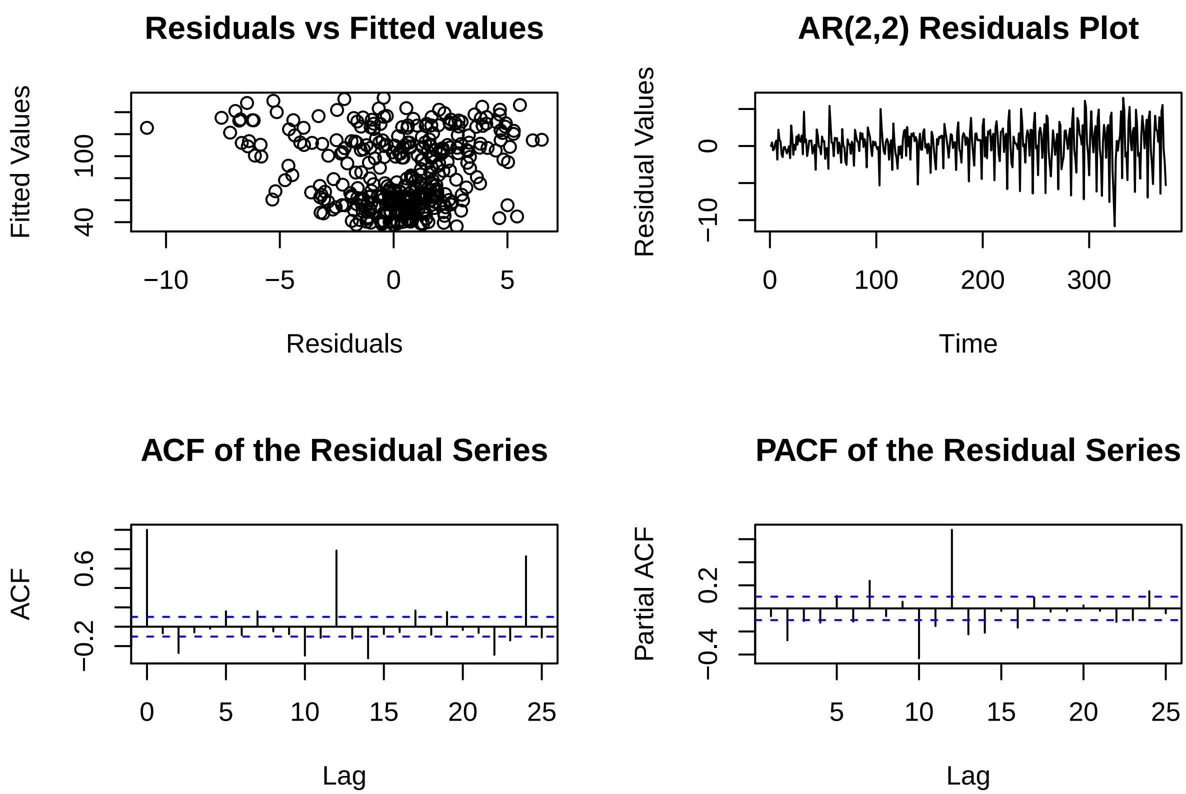


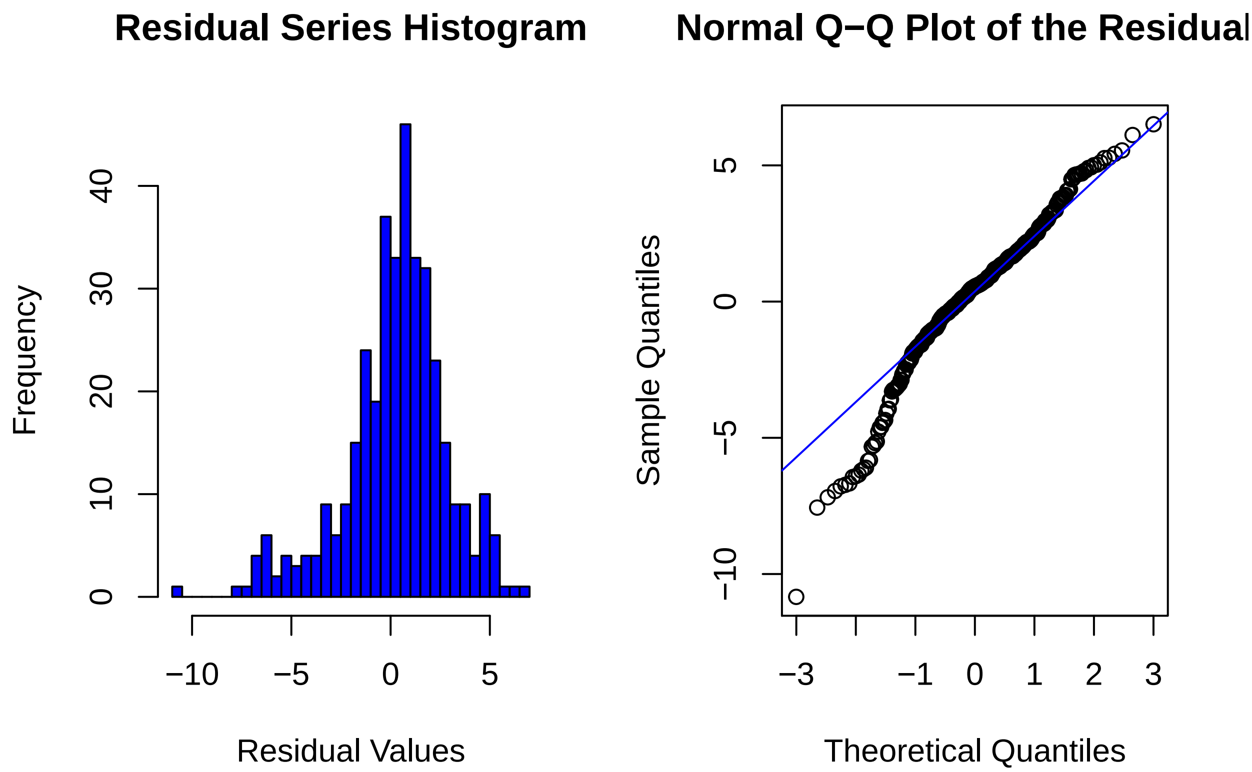

```r
# ARMA(2,2)
head(series.model4$resid)
```

```
## [1] -0.0003330845  0.4846433935 -0.5794333572 -0.4200433930  0.3556912769
## [6]  0.7381059461
```

```r
par(mfrow = c(2, 2))
plot(series.model4$resid, fitted(series.model4), main = "Residuals vs Fitted values",
    xlab = "Residuals", ylab = "Fitted Values")
plot(series.model4$resid, type = "l", main = "AR(2,2) Residuals Plot",
    xlab = "Time", ylab = "Residual Values")
acf(series.model4$resid, main = "ACF of the Residual Series")
pacf(series.model4$resid, main = "PACF of the Residual Series")
```




```
par(mfrow = c(1, 2))
hist(series.model4$resid, breaks = 60, col = "blue", main = "Residual Series Histogram",
    xlab = "Residual Values")
qqnorm(series.model4$resid, main = "Normal Q-Q Plot of the Residuals",
    type = "p")
qqline(series.model4$resid[-1], col = "blue")
```




```r
# Running the box test to see independence of the residuals
Box.test(series.model2$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  series.model2$resid
## X-squared = 2.88, df = 1, p-value = 0.08968
```

```r
Box.test(series.model3$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  series.model3$resid
## X-squared = 8.0155, df = 1, p-value = 0.004638
```

```r
Box.test(series.model4$resid, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  series.model4$resid
## X-squared = 1.7237, df = 1, p-value = 0.1892
```

```r
# Since we have chosen to move forward with the AR(1), we
# test assumptions
roots = polyroot(c(1, -series.model2$coef["ar1"]))
Mod(roots[1])
```

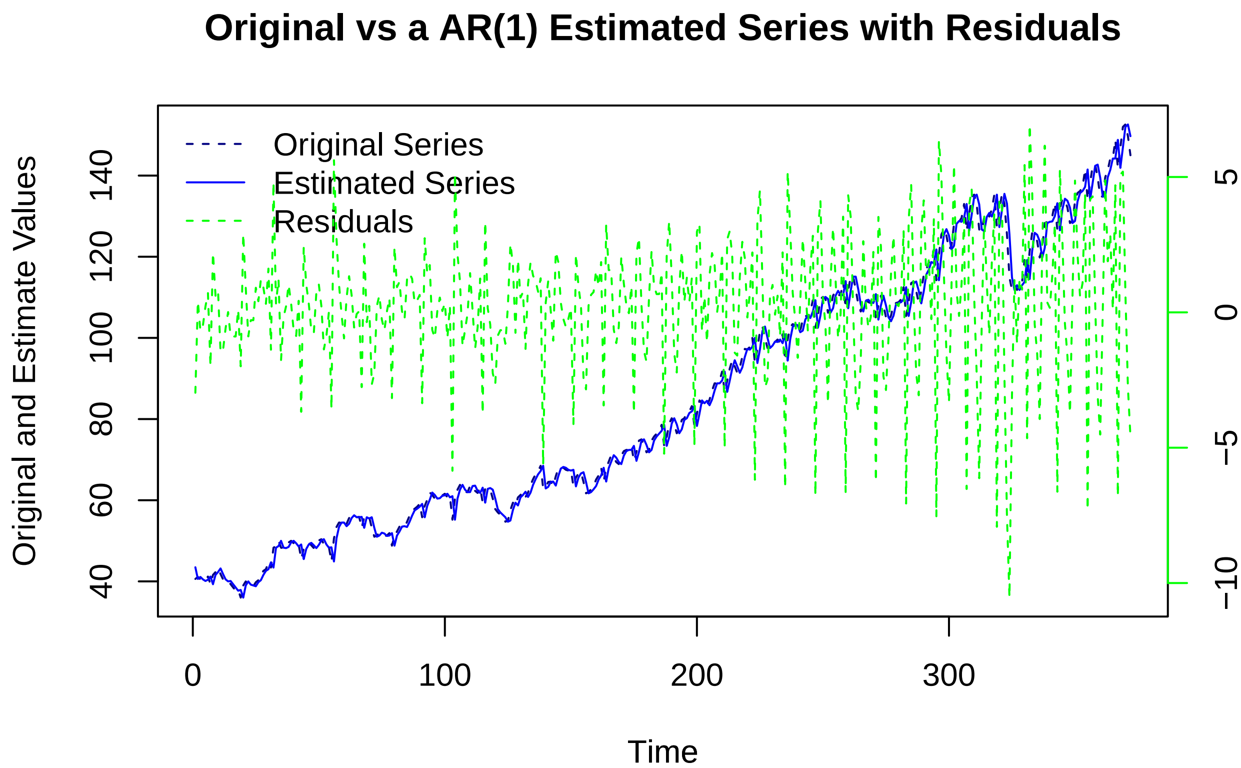
```
## [1] 1.001763
```

```r
# Not stationary
```

## Part 3 - In-Sample Fit

The AR(1) fits the series extremely well, the only cause for concern being the increasing amplitude of the residuals with time.

```r
par(mfrow = c(1, 1))
plot.ts(series, col = "navy", lty = 2, main = "Original vs a AR(1) Estimated Series with Residuals",
    ylab = "Original and Estimate Values")
par(new = T)
plot(fitted(series.model2), col = "blue", axes = F, ylab = "")
leg.txt <- c("Original Series", "Estimated Series", "Residuals")
legend("topleft", legend = leg.txt, lty = c(2, 1, 2), col = c("navy",
    "blue", "green"), bty = "n", cex = 1)
par(new = T)
plot.ts(series.model2$resid, axes = F, xlab = "", ylab = "",
    col = "green", pch = 1, lty = 2)
axis(side = 4, col = "green")
mtext("Residuals", side = 4, line = 2, col = "green")
```



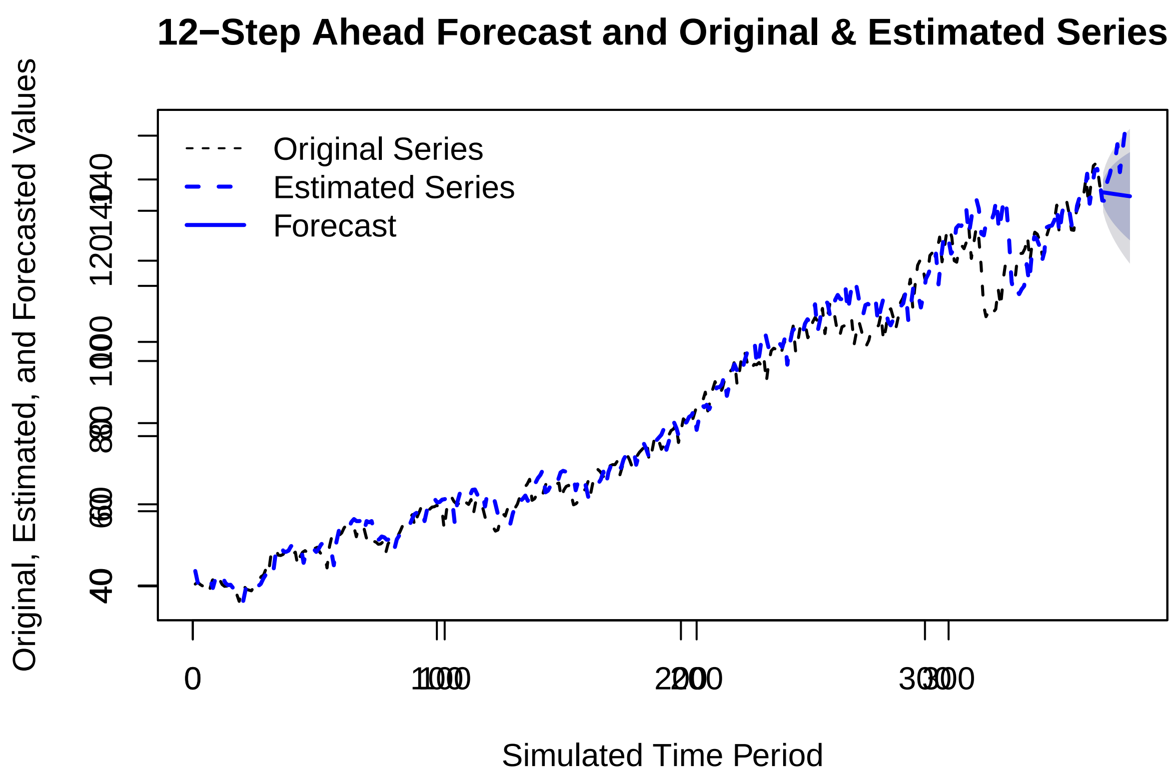

## Part 4 - 12 Step Ahead Forecast

**Key Takeaways:** 1.

```
series.model.fcast <- forecast.Arima(series.model2, h = 12)

length(series.model.fcast$mean)
```

```
## [1] 12
```

```
plot(series.model.fcast, main = "12-Step Ahead Forecast and Original & Estimated Series",
    xlab = "Simulated Time Period", ylab = "Original, Estimated, and Forecasted Values",
    lty = 2, lwd = 1.5)
par(new = T)
plot.ts(fitted(series.model2), col = "blue", lty = 2, lwd = 2,
    xlab = "", ylab = "")
leg.txt <- c("Original Series", "Estimated Series", "Forecast")
legend("topleft", legend = leg.txt, lty = c(2, 2, 1), lwd = c(1,
    2, 2), col = c("black", "blue", "blue"), bty = "n", cex = 1)
```




## Part 5 - Backtesting

**Key Takeaways:** 1. The backtesting model does a good job of forecasting, with the series falling within the forecast.

```
series.model.b <- Arima(series[1:(length(series) - 37)], order = c(1,
    0, 0))
summary(series.model.b)
```

```
## Series: series[1:(length(series) - 37)]
## ARIMA(1,0,0) with non-zero mean
##
## Coefficients:
##          ar1  intercept
##       0.9976    79.4191
## s.e.  0.0028    30.6340
##
## sigma^2 estimated as 6.377:  log likelihood=-788.34
## AIC=1582.67   AICc=1582.74   BIC=1594.11
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE    MAPE     MASE
## Training set 0.239139 2.525272 1.843797 0.2311798 2.33672 1.000371
##                     ACF1
```

```
## Training set -0.08404489
```

```r
length(fitted(series.model.b))
```

```
## [1] 335
```

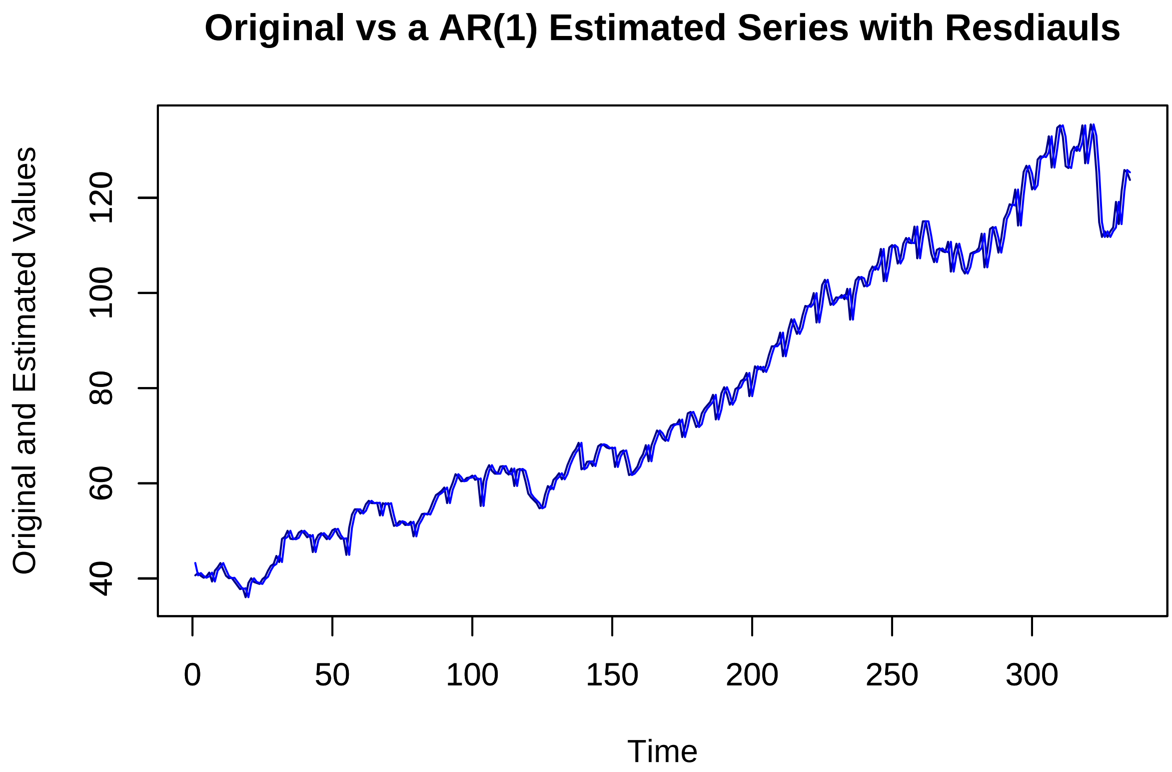
```r
length(series.model.b$resid)
```

```
## [1] 335
```

```r
df = cbind(series[1:(length(series) - 37)], fitted(series.model.b),
    series.model.b$resid)
colnames(df) = c("orig_series", "fitted_vals", "resid")
head(df)
```

```
##      orig_series fitted_vals     resid
## [1,]        40.6    43.30457 -2.704571
## [2,]        41.1    40.69433  0.405670
## [3,]        40.5    41.19311 -0.693115
## [4,]        40.1    40.59457 -0.494573
## [5,]        40.4    40.19554  0.204455
## [6,]        41.2    40.49482  0.705184
```

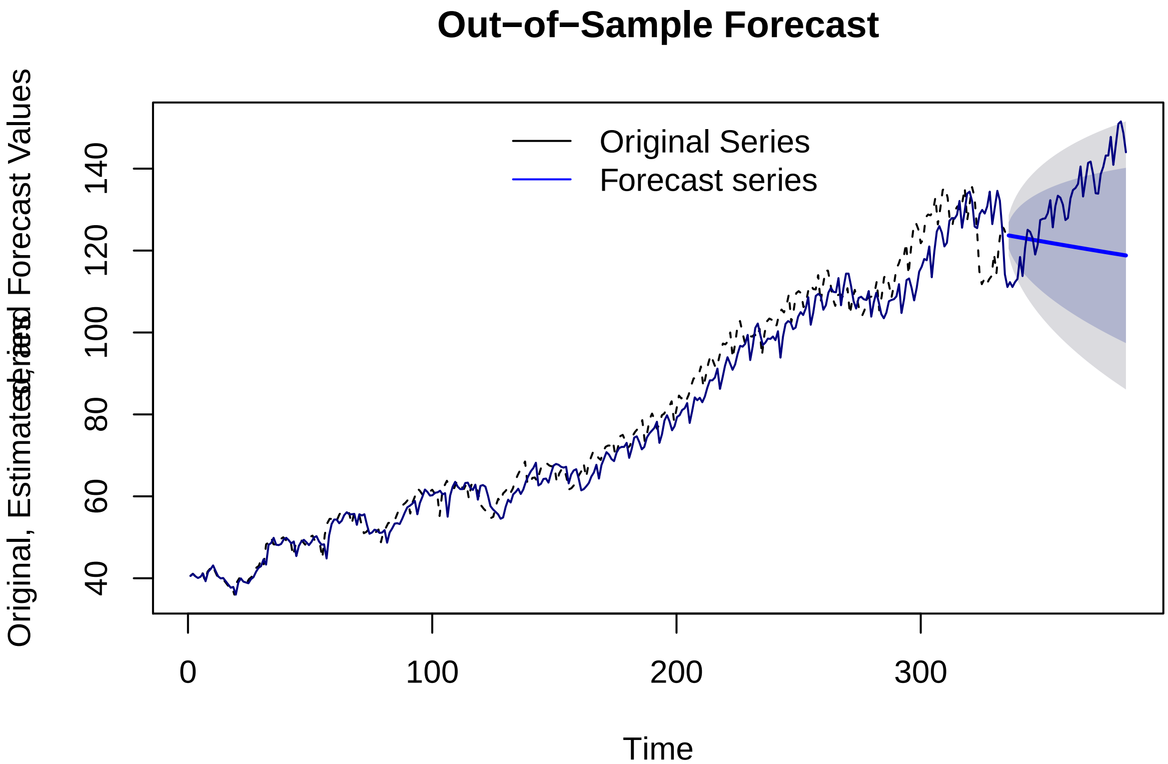
```r
# Plot the original and estimate series
par(mfrow = c(1, 1))
plot.ts(df[, "orig_series"], col = "navy", main = "Original vs a AR(1) Estimated Series with Resdiauls"
    ylab = "Original and Estimated Values")
par(new = T)
plot.ts(df[, "fitted_vals"], col = "blue", axes = T, xlab = "",
    ylab = "")
```




```r
# Step 2: Out of sample forecast
series.model.b.fcast <- forecast.Arima(series.model.b, h = 49)
length(series.model.b.fcast$mean)
```

```
## [1] 49
```

```r
par(mfrow = c(1, 1))
plot(series.model.b.fcast, lty = 2, main = "Out-of-Sample Forecast",
    ylab = "Original, Estimated, and Forecast Values")
par(new = T)
plot.ts(series, col = "navy", axes = F, lty = 1)
leg.txt <- c("Original Series", "Forecast series")
legend("top", legend = leg.txt, lty = 1, col = c("black", "blue"),
    bty = "n", cex = 1)
```


Out−of−Sample Forecast
Original Series
Forecast series
Original, Estimated series, and Forecast Values
40
60
80
100
120
140
0
100
200
300
Time