W271 Lab 3 Spring 2016

Megan Jasek, Rohan Thakur, Charles Kekeh

Sunday, April 24, 2016

```
# Functions for Parts 2, 3, 4
get.best.arima <- function(x.ts, maxord = c(1, 1, 1)) {
    best.aic <- 1e+08
    all.aics <- vector()
    all.models <- vector()
    n <- length(x.ts)
    for (p in 0:maxord[1]) for (d in 0:maxord[2]) for (q in 0:maxord[3]) {
        fit <- arima(x.ts, order = c(p, d, q), method = "ML")
        fit.aic <- -2 * fit$loglik + (log(n) + 1) * length(fit$coef)
        if (fit.aic < best.aic) {
            best.aic <- fit.aic
            best.fit <- fit
            best.model <- c(p, d, q)
        }
        all.aics <- c(all.aics, fit.aic)
        all.models <- c(all.models, sprintf("(%d, %d, %d)", p,
            d, q))
    }
    list(best = list(best.aic, best.fit, best.model), others = data.frame(aics = all.aics,
        models = all.models))
}

get.best.sarima <- function(x.ts, maxord = c(1, 1, 1, 1, 1, 1),
    freq) {
    best.aic <- 1e+08
    all.aics <- vector()
    all.models <- vector()
    n <- length(x.ts)
    for (p in 0:maxord[1]) for (d in 0:maxord[2]) for (q in 0:maxord[3]) for (P in 0:maxord[3]) for (D
        fit <- arima(x.ts, order = c(p, d, q), seasonal = list(order = c(P,
            D, Q), freq), method = "CSS", optim.control = list(maxit = 10000))
        fit.aic <- -2 * fit$loglik + (log(n) + 1) * length(fit$coef)
        if (fit.aic < best.aic) {
            best.aic <- fit.aic
            best.fit <- fit
            best.model <- c(p, d, q, P, D, Q)
        }
        all.aics <- c(all.aics, fit.aic)
        all.models <- c(all.models, sprintf("(%d, %d, %d, %d, %d, %d)",
            p, d, q, P, D, Q))
    }
    list(best = list(best.aic, best.fit, best.model), others = data.frame(aics = all.aics,
        models = all.models))
}

plot.time.series <- function(x.ts, bins = 30, name) {
    str(x.ts)
```

```
    par(mfrow = c(2, 2))
    hist(x.ts, bins, main = paste("Histogram of", name, sep = " "),
        xlab = "Values")
    plot(x.ts, main = paste("Plot of", name, sep = " "), ylab = "Values",
        xlab = "Time Period")
    acf(x.ts, main = paste("ACF of", name, sep = " "))
    pacf(x.ts, main = paste("PACF of", name, sep = " "))
}

plot.residuals.ts <- function(x.mod, model_name) {
    par(mfrow = c(1, 1))
    hist(x.mod$residuals, 30, main = paste("Histogram of", model_name,
        "Residuals", sep = " "), xlab = "Values")
    par(mfrow = c(2, 2))
    plot(x.mod$residuals, fitted(x.mod), main = paste(model_name,
        "Fitted vs. Residuals", sep = " "), ylab = "Fitted Values",
        xlab = "Residuals")
    plot(x.mod$residuals, main = paste(model_name, "Residuals",
        sep = " "), ylab = paste("Residuals", sep = " "))
    acf(x.mod$residuals, main = paste("ACF of", model_name, sep = " "))
    pacf(x.mod$residuals, main = paste("PACF of", model_name,
        sep = " "))
    Box.test(x.mod$residuals, type = "Ljung-Box")
}

estimate.ar <- function(x.ts) {
    x.ar = ar(x.ts)
    print("Difference in AICs")
    print(x.ar$aic)
    print("AR parameters")
    print(x.ar$ar)
    print("AR order")
    print(x.ar$order)
    return(x.ar)
}

plot.orig.model.resid <- function(x.ts, x.mod, model_name, xlim,
    ylim) {
    df <- data.frame(cbind(x.ts, fitted(x.mod), x.mod$residuals))
    class(df)
    stargazer(df, type = "text", title = "Descriptive Stat",
        digits = 1)

    summary(x.ts)
    summary(x.mod$residuals)
    par(mfrow = c(1, 1))
    plot.ts(x.ts, col = "red", main = paste("Orivinal vs Estimated",
        model_name, "Series with Resdiauls", sep = " "), ylab = "Original and Estimated Values",
        xlim = xlim, ylim = ylim, pch = 1, lty = 2)
    par(new = T)
    plot.ts(fitted(x.mod), col = "blue", axes = T, xlab = "",
        ylab = "", xlim = xlim, ylim = ylim, lty = 1)
    leg.txt <- c("Original Series", "Estimated Series", "Residuals")
```

```
    legend("topleft", legend = leg.txt, lty = c(2, 1, 2), col = c("red",
        "blue", "green"), bty = "n", cex = 1)
    par(new = T)
    plot.ts(x.mod$residuals, axes = F, xlab = "", ylab = "",
        col = "green", xlim = xlim, ylim = ylim, lty = 2, pch = 1,
        col.axis = "green")
    axis(side = 4, col = "green")
    mtext("Residuals", side = 4, line = 2, col = "green")
}

plot.model.forecast <- function(x.mod, mod.fcast, num_steps,
    x, y) {
    par(mfrow = c(1, 1))
    plot(mod.fcast, main = paste(num_steps, "-Step Ahead Forecast and Original & Estimated Series",
        sep = ""), xlab = "Simulated Time Period", ylab = "Original, Estimated, and Forecasted Values",
        xlim = x, ylim = y, lty = 2, lwd = 1.5)
    par(new = T)
    plot.ts(fitted(x.mod), col = "blue", lty = 2, lwd = 2, xlab = "",
        ylab = "", xlim = x, ylim = y)
    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 2 (25 points): Modeling and Forecasting a Real-World Macroeconomic / Financial time series

The series appears to be a time series of financial data, presumably one of a daily closing price of some financial instrumemt or index.

We observe that the time series is non-stationary in the mean. That observation, is coupled with the fact that the PACF of the time series indicates a correlation at lag 1 that ressembles and AR(1) series. Therefore we can attempt to diff the time series to see if the resulting series is stationary in the mean.

The plot of the original financial series does not indicate any amount of seasonality. The observation is confirmed by the ACF which doesn't display any significant amount of correlation at any lag. We will therefore not try to fit a model that includes a seasonal component to the data. We now proceed to analyze the first difference of the time series.

We get confirmation from the 1st difference series obtained that the original financial series likely has an AR(1) component as the first difference series is stationary in the mean. We also observe that after the first difference is taken, the ACF of the series, just like the PACF do not show noticeble dynamic in the mean. We have no reason to suspect an additional MA component in the original series. But the first differential series does show clustered volatility in the variance.

```
# Load the data and describe it
data <- read.csv(file.path("lab3_series02.csv"))

# Describing Series
str(data)
```

```
## 'data.frame':    2332 obs. of  2 variables:
##  $ X         : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
##  $ DXCM.Close: num  9.88 9.79 9.68 9.64 9.42 9.47 9.16 8.99 8.6 8.81 ...
```

```r
summary(data)
```

```
##        X            DXCM.Close
##  Min.   :   1.0   Min.   :  1.390
##  1st Qu.: 583.8   1st Qu.:  8.188
##  Median :1166.5   Median : 12.355
##  Mean   :1166.5   Mean   : 23.210
##  3rd Qu.:1749.2   3rd Qu.: 32.565
##  Max.   :2332.0   Max.   :101.910
```

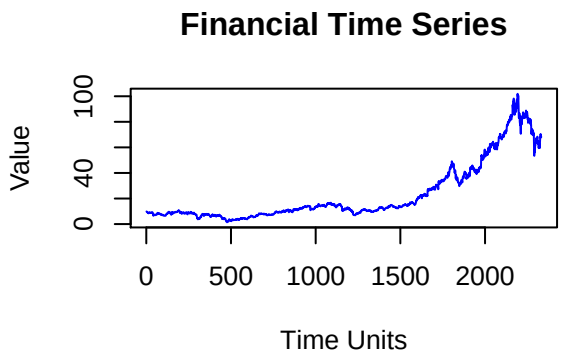
```r
cbind(head(data), tail(data))
```

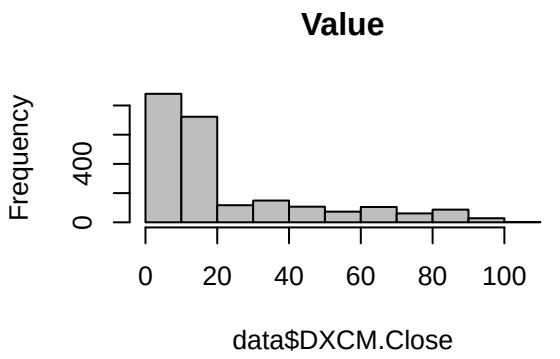
```
##   X DXCM.Close    X DXCM.Close
## 1 1       9.88 2327      67.63
## 2 2       9.79 2328      70.49
## 3 3       9.68 2329      67.79
## 4 4       9.64 2330      68.72
## 5 5       9.42 2331      68.43
## 6 6       9.47 2332      68.08
```

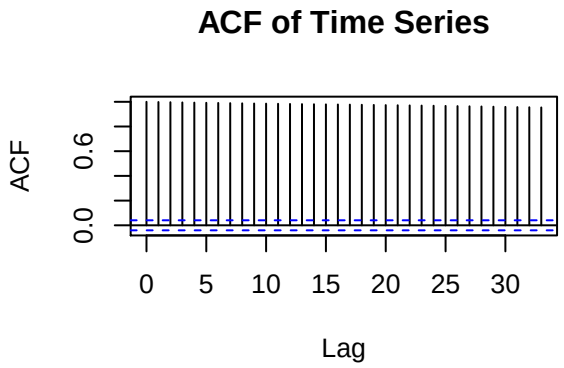
```r
quantile(as.numeric(data$DXCM.Close), c(0, 0.01, 0.05, 0.1, 0.25,
    0.5, 0.75, 0.9, 0.95, 0.99, 1))
```

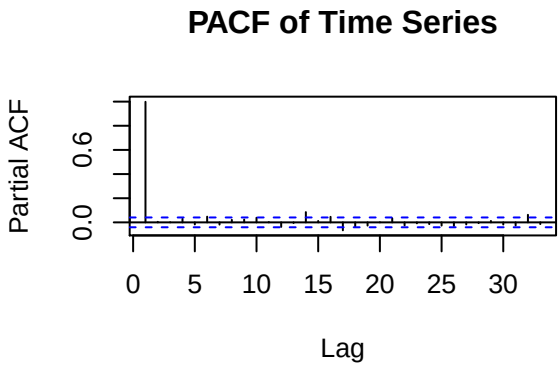
```
##       0%       1%       5%      10%      25%      50%      75%      90%
##   1.3900   2.7531   4.1700   6.1630   8.1875  12.3550  32.5650  63.5210
##      95%      99%     100%
##  80.0430  91.6887 101.9100
```

```r
# EDA on series
par(mfrow = c(2, 2))
plot.ts(data$DXCM.Close, main = "Financial Time Series", ylab = "Value",
    xlab = "Time Units", col = "blue")
hist(data$DXCM.Close, col = "gray", main = "Value")
acf(data$DXCM.Close, main = "ACF of Time Series")
pacf(data$DXCM.Close, main = "PACF of Time Series")
```

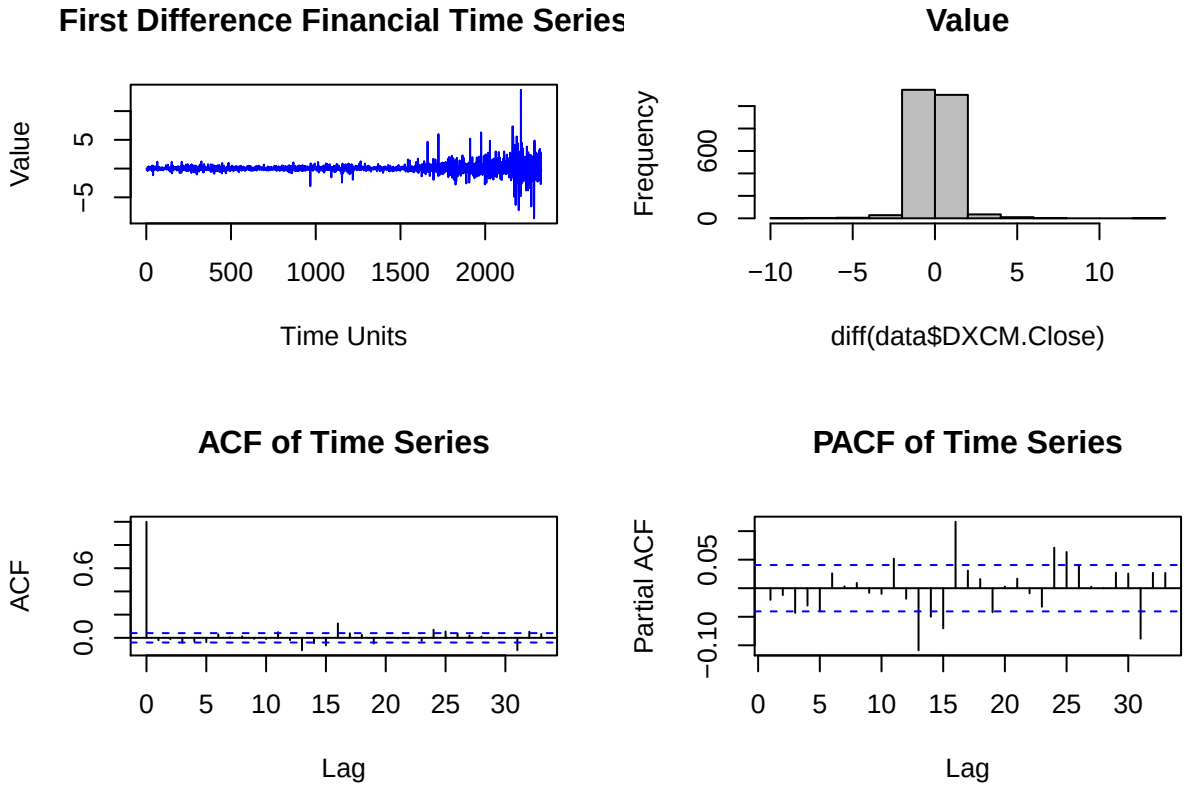









```r
# EDA on the first difference series
par(mfrow = c(2, 2))
plot.ts(diff(data$DXCM.Close), main = "First Difference Financial Time Series",
    ylab = "Value", xlab = "Time Units", col = "blue")
hist(diff(data$DXCM.Close), col = "gray", main = "Value")
acf(diff(data$DXCM.Close), main = "ACF of Time Series")
pacf(diff(data$DXCM.Close), main = "PACF of Time Series")
```



We now try to determine a best SARIMA model based on the best AIC for that series. Interestingly, the best model based on AIC is a seasonal model with (p,d,q,P,D,Q) of values (0, 0, 0, 0, 1, 0). If we assume a frequency of one, that model is the same as the second best model, which has orders (p,d,q,P,D,Q) of (0, 1, 0, 0, 0, 0). The results confirm the observation we made of a lag 1 correlation looking at the PACF of the financial time series We therefore decide to use (p,d,q,P,D,Q)=(0, 1, 0, 0, 0, 0) as our fitted model going forward.

```
# data.best <- get.best.sarima(data$DXCM, maxord=rep(3,6),1)
# data.best$best
# data.best$others[order(data.best$others$aics)[1:20],]
```

The fitted model is simply a first differential series, and therefore has no parameters which need 95 confidence intervals that we need to validate.

```
# Fitting a first difference series to our model
data.fit <- Arima(data$DXCM, order = c(1, 0, 0), seasonal = list(order = c(0,
    0, 0)), method = "CSS-ML")
data.res <- data.fit$resid
quantile(as.numeric(data.res), c(0, 0.01, 0.05, 0.1, 0.25, 0.5,
    0.75, 0.9, 0.95, 0.99, 1))
```

```
##           0%           1%           5%          10%          25%
## -8.624611999 -2.432495919 -1.092944118 -0.561843680 -0.235516419
##          50%          75%          90%          95%          99%
## -0.008677486  0.247368331  0.661566155  1.151658704  2.879518631
##         100%
```

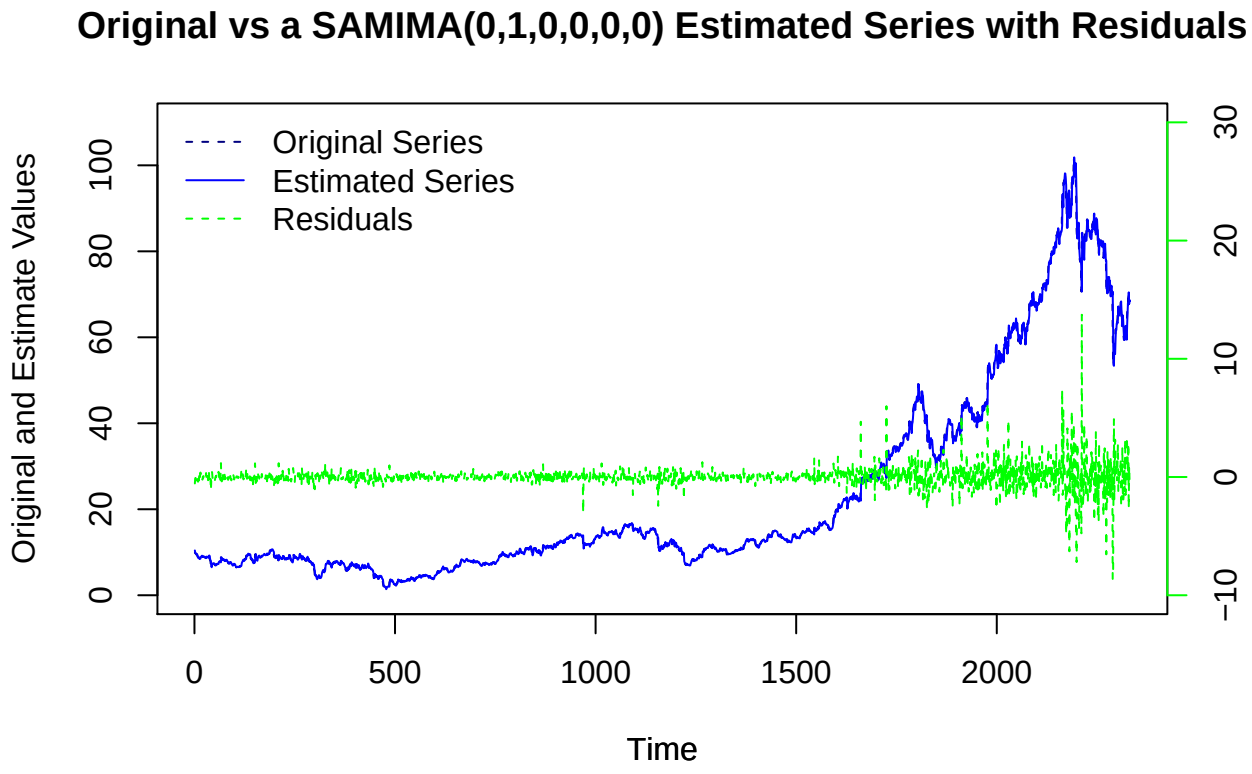
```
## 13.725399392
```

```
t(confint(data.fit))
```

```
##              ar1 intercept
## 2.5 %  0.9982756 -17.00436
## 97.5 % 1.0005853  69.11511
```

We now perform in-sample fit using the fitted series to assess our fitted model. The fitted series models the original series very well and the model selection seems appropriat based on in-sample fit.

```
# Performing in-sample fit using our fitted series
par(mfrow = c(1, 1))
plot.ts(data$DXCM, col = "navy", lty = 2, main = "Original vs a SAMIMA(0,1,0,0,0,0) Estimated Series wi
    ylab = "Original and Estimate Values", ylim = c(0, 110))
par(new = T)
plot(fitted(data.fit), col = "blue", axes = F, ylab = "", ylim = c(0,
    110))
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(data.res, axes = F, xlab = "", ylab = "", col = "green",
    ylim = c(-10, 30), pch = 1, lty = 2)
axis(side = 4, col = "green")
mtext("Residuals", side = 4, line = 2, col = "green")
```



To further assess the fitted series, We now perform 36 steps backtesting. What the out of sample forecast shows is that the original series falls for the most part within the 80% confidence interval and almost totally within the 95% confidence interval of the prediction. However, we can see from the plot of the financial time series that the volatility of the series seems to increase over time. Clustered volatility in the variance likely explain this dynamic. We next turn our eyes to the residuals of the fitted series and to analysis of the dynamics of its variance.

```
# Performing 36 steps backtesting using our fitted series
data.fit.back <- Arima(data$DXCM[1:(length(data$DXCM) - 36)],
    order = c(0, 1, 0), seasonal = list(order = c(0, 0, 0)),
    method = "CSS-ML")
summary(data.fit.back)
```

```
## Series: data$DXCM[1:(length(data$DXCM) - 36)]
## ARIMA(0,1,0)
##
## sigma^2 estimated as 0.8223:  log likelihood=-3031.91
## AIC=6065.83   AICc=6065.83   BIC=6071.57
##
## Training set error measures:
##                       ME      RMSE       MAE         MPE     MAPE      MASE
## Training set 0.02262625 0.9065937 0.4574782 0.007157606 2.477381 0.9995739
##                      ACF1
## Training set -0.01924297
```

```
length(fitted(data.fit.back))
```

```
## [1] 2296
```

```
length(data.fit.back$resid)
```

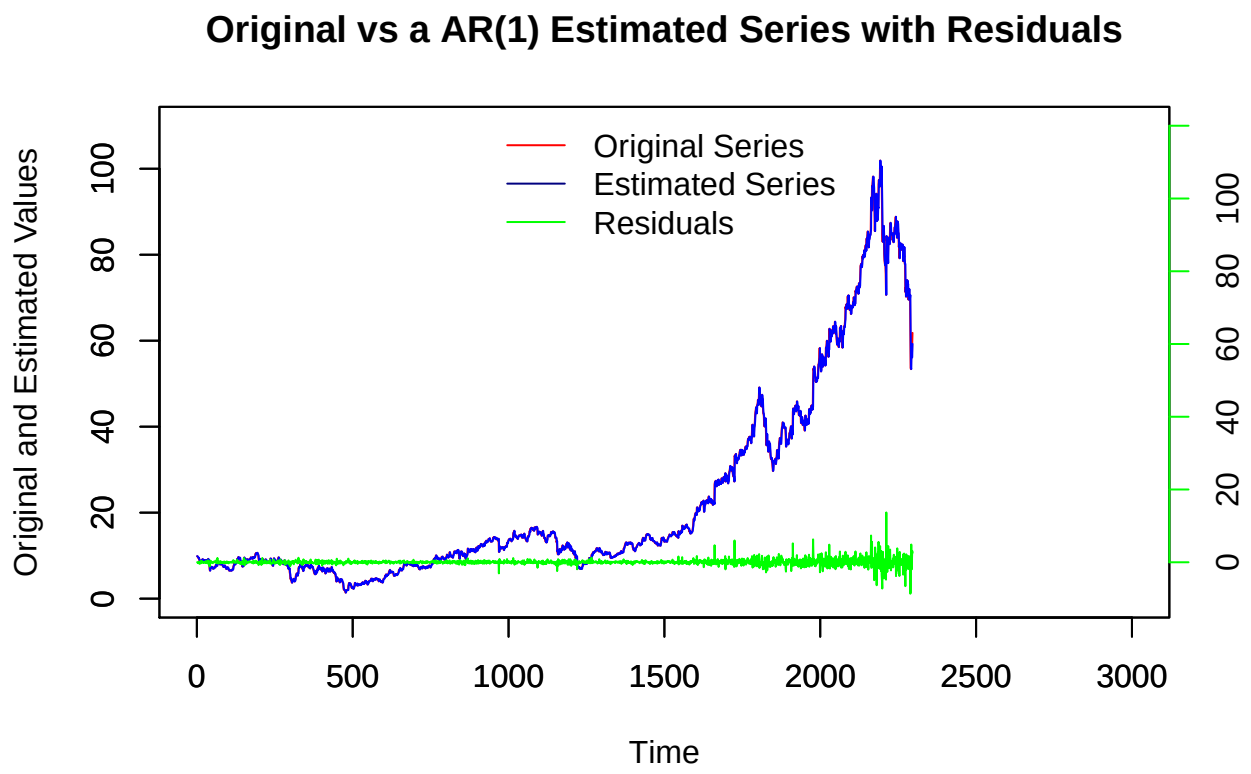
```
## [1] 2296
```

```
df = cbind(data$DXCM[1:(length(data$DXCM) - 36)], fitted(data.fit.back),
    data.fit.back$resid)
colnames(df) = c("orig_series", "fitted_vals", "resid")
head(df)
```

```
##      orig_series fitted_vals        resid
## [1,]        9.88     9.87012  0.009879995
## [2,]        9.79     9.88000 -0.090000000
## [3,]        9.68     9.79000 -0.110000000
## [4,]        9.64     9.68000 -0.040000000
## [5,]        9.42     9.64000 -0.220000000
## [6,]        9.47     9.42000  0.050000000
```

```
# Step 1: Plot the original and estimate series
par(mfrow = c(1, 1))
plot.ts(df[, "orig_series"], col = "red", main = "Original vs a AR(1) Estimated Series with Residuals",
    ylab = "Original and Estimated Values", xlim = c(0, 3000),
    ylim = c(0, 110))
```

```
par(new = T)
plot.ts(df[, "fitted_vals"], col = "blue", axes = T, xlab = "",
    ylab = "", xlim = c(0, 3000), ylim = c(0, 110))
leg.txt <- c("Original Series", "Estimated Series", "Residuals")
legend("top", legend = leg.txt, lty = 1, col = c("red", "navy",
    "green"), bty = "n", cex = 1)
par(new = T)
plot.ts(df[, "resid"], axes = F, xlab = "", ylab = "", col = "green",
    xlim = c(0, 3000), ylim = c(-10, 120), pch = 1)
axis(side = 4, col = "green")
mtext("Residuals", side = 4, line = 2, col = "green")
```

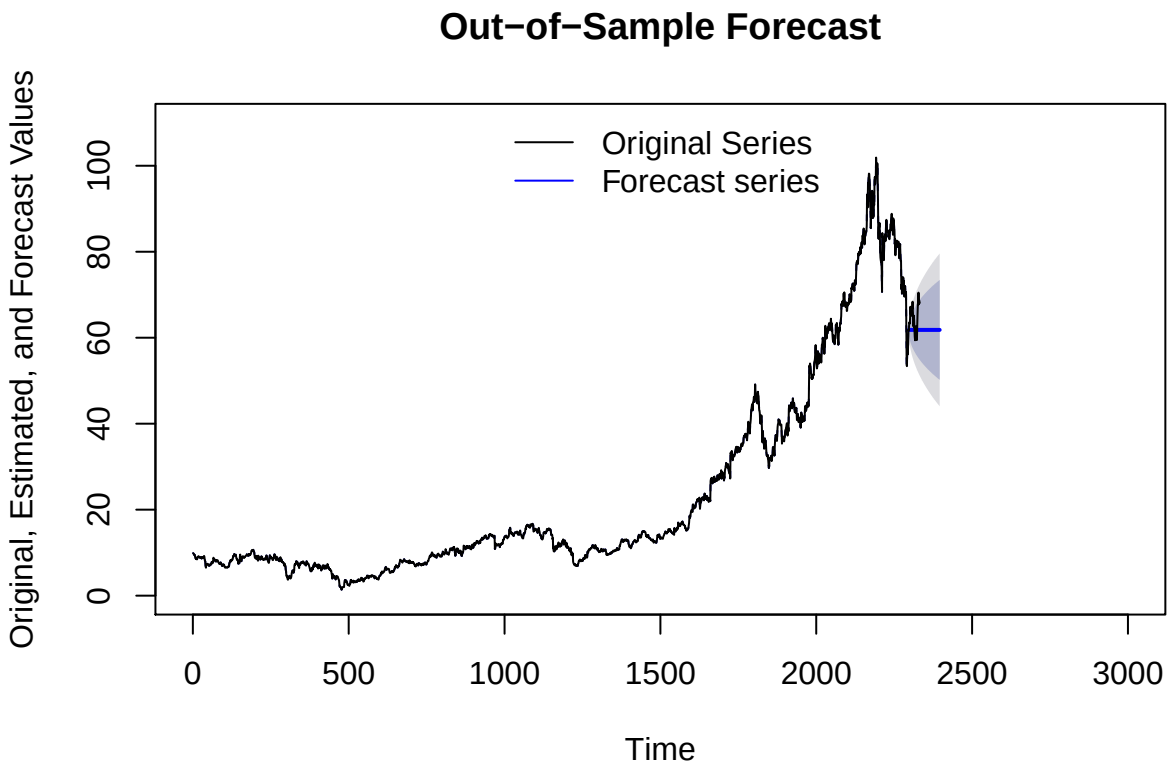



```
# Step 2: Out of sample forecast
data.fit.back.fcast <- forecast.Arima(data.fit.back, h = 100)
length(data.fit.back.fcast$mean)
```

```
## [1] 100
```

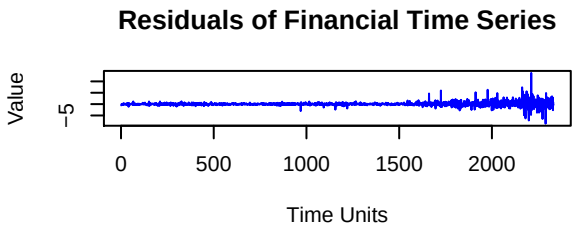
```
par(mfrow = c(1, 1))
plot(data.fit.back.fcast, lty = 2, col = "navy", main = "Out-of-Sample Forecast",
    ylab = "Original, Estimated, and Forecast Values", xlim = c(0,
        3000), ylim = c(0, 110))
par(new = T)
plot.ts(data$DXCM, axes = F, lty = 1, xlim = c(0, 3000), ylim = c(0,
```

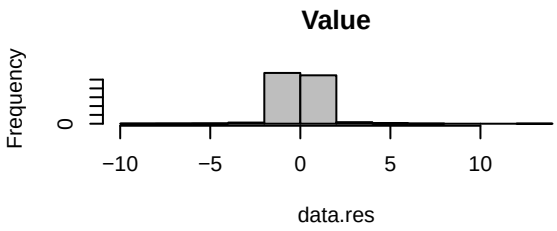
```
    110), ylab = "")
leg.txt <- c("Original Series", "Forecast series")
legend("top", legend = leg.txt, lty = 1, col = c("black", "blue"),
    bty = "n", cex = 1)
```

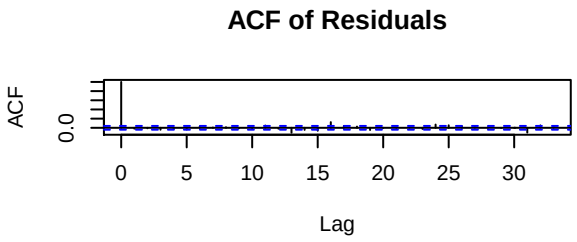


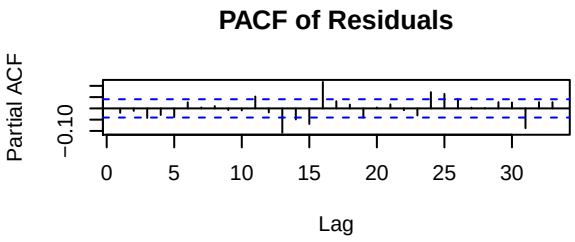
We observe from the residual time series that the variance of the series is non-stationary. The series exhibits volatility with a variance changing in a regular way. It exhibits conditional heteroskedasticity. An observation of the PACF of the squared residuals series provides confirmation of the variance dynamics. Therefore, we decide to model its residuals using GARCH

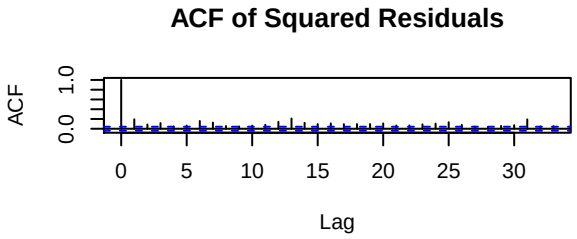
```
# Plot the residuals time series
par(mfrow = c(3, 2))
plot.ts(data.res, main = "Residuals of Financial Time Series",
    ylab = "Value", xlab = "Time Units", col = "blue")
hist(data.res, col = "gray", main = "Value")
acf(data.res, main = "ACF of Residuals")
pacf(data.res, main = "PACF of Residuals")
acf(data.res^2, main = "ACF of Squared Residuals")
```











We chose the default (p, q) = (1, 1) parameters of the function for our GARCH model. The parameters of the model are all significant based on a 95% confidence interval.

We observe from the ACF of the residuals of the GARCH fitted series that they have the charactersitics of white noise with mostly non-significant correlations at all lags of the ACF. What the GARCH model of the residuals tells is that we can expect more or less volatility through the forecast of the point series that invalidate the confidence interval of our predictions since those were made with the assumption of a stationary variance.

```
# Model the residuala of the financila time series using
# GARCH
data.garch <- garch(data.res, trace = F)
```

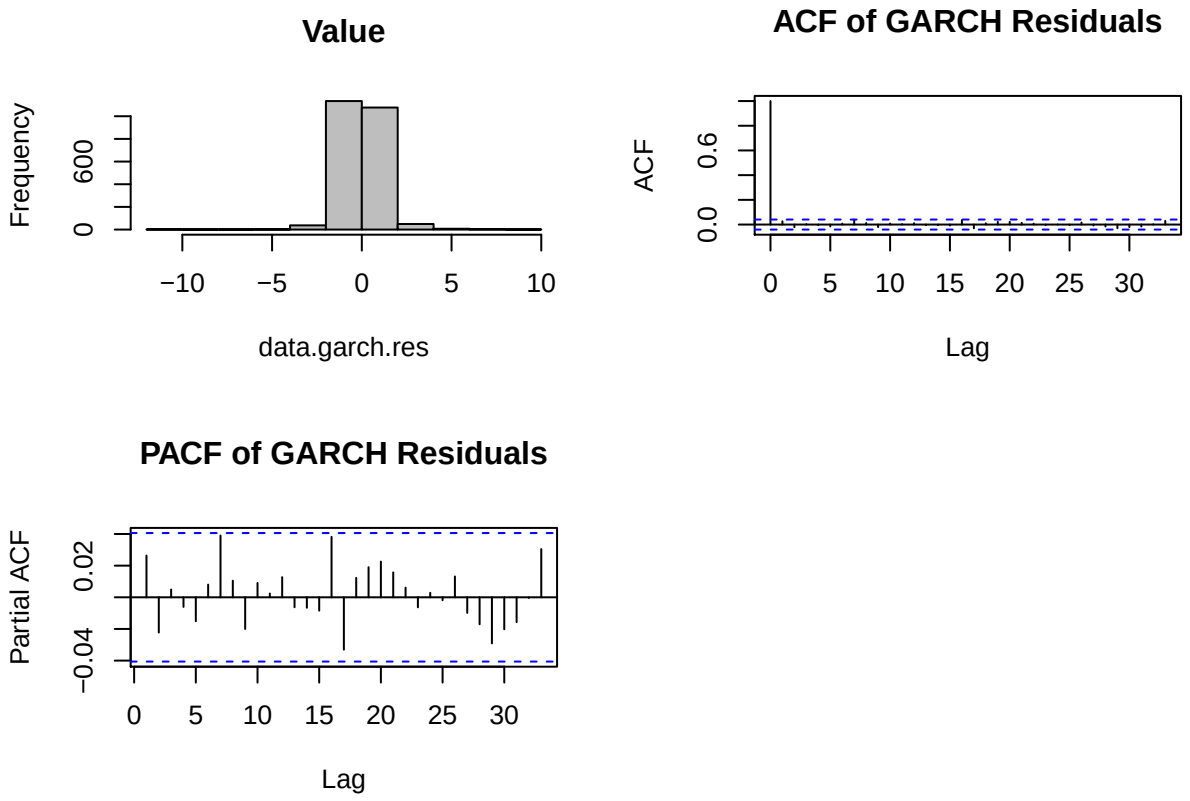
```
## Warning in sqrt(pred$e): NaNs produced
```

```
t(confint(data.garch))
```

```
##                   a0         a1        b1
## 2.5 %  5.785885e-05 0.03064265 0.9652906
## 97.5 % 4.276349e-04 0.03947031 0.9732850
```

```
data.garch.res <- resid(data.garch)[-1]

# Plot a histogram, ACF and PACF of the residuals after
# fitting a GARCH model
```

```
par(mfrow = c(2, 2))
hist(data.garch.res, col = "gray", main = "Value")
acf(data.garch.res, na.action = na.pass, main = "ACF of GARCH Residuals")
pacf(data.garch.res, na.action = na.pass, main = "PACF of GARCH Residuals")
```



With the previous observations in mind, we still use the fitted series to predict 36 steps ahead. We will later adjust the confidence intervals of the predictions using our fitted GARCH model.

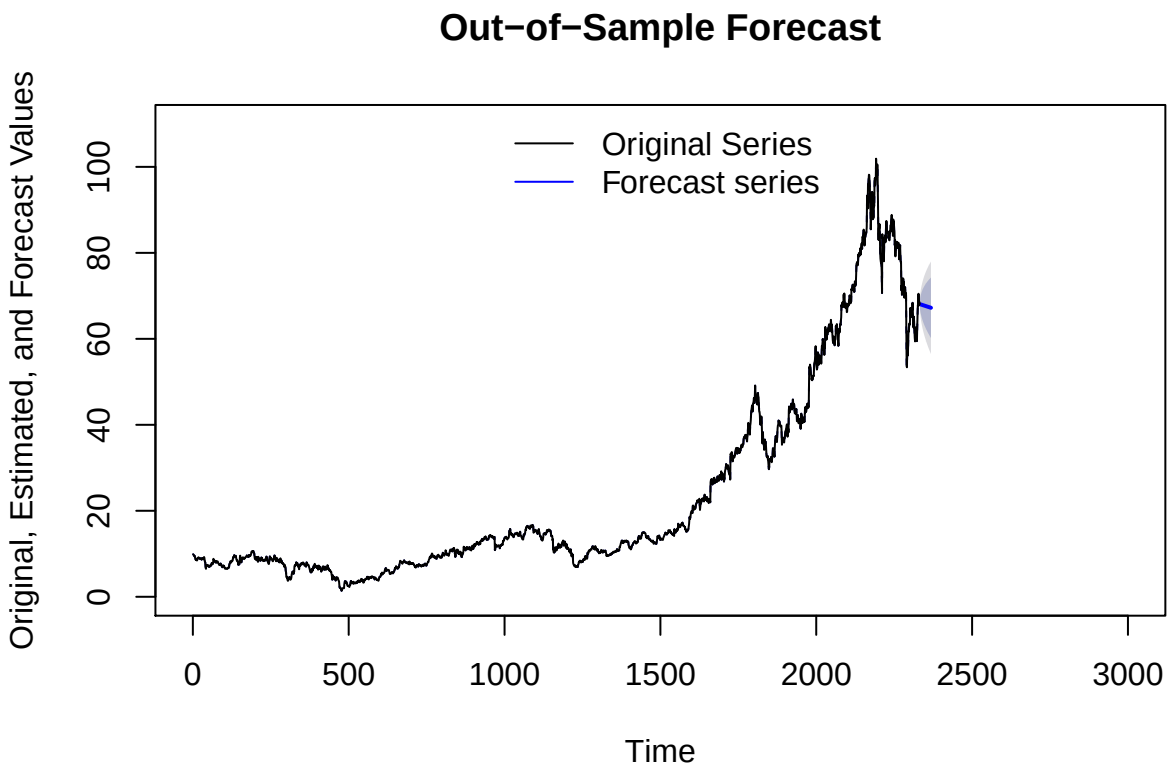
To perform a 36 steps ahead forecast of our series, we use the original point estimate of the series, that being the (0,1,0,0,0,0) SARIMA model initially estimated. The GARCH model of the residuals will additionally be used to forcast the variance of the series, and help us adjust the confidence interval of the prediction.

```
# 36 steps ahead sample forecast of the financial time series
data.fit.ahead.fcast <- forecast.Arima(data.fit, h = 36)
length(data.fit.ahead.fcast$mean)
```

```
## [1] 36
```

```
par(mfrow = c(1, 1))
plot(data.fit.ahead.fcast, lty = 2, col = "navy", main = "Out-of-Sample Forecast",
    ylab = "Original, Estimated, and Forecast Values", xlim = c(0,
        3000), ylim = c(0, 110))
par(new = T)
plot.ts(data$DXCM, axes = F, lty = 1, xlim = c(0, 3000), ylim = c(0,
    110), ylab = "")
```

```
leg.txt <- c("Original Series", "Forecast series")
legend("top", legend = leg.txt, lty = 1, col = c("black", "blue"),
    bty = "n", cex = 1)
```



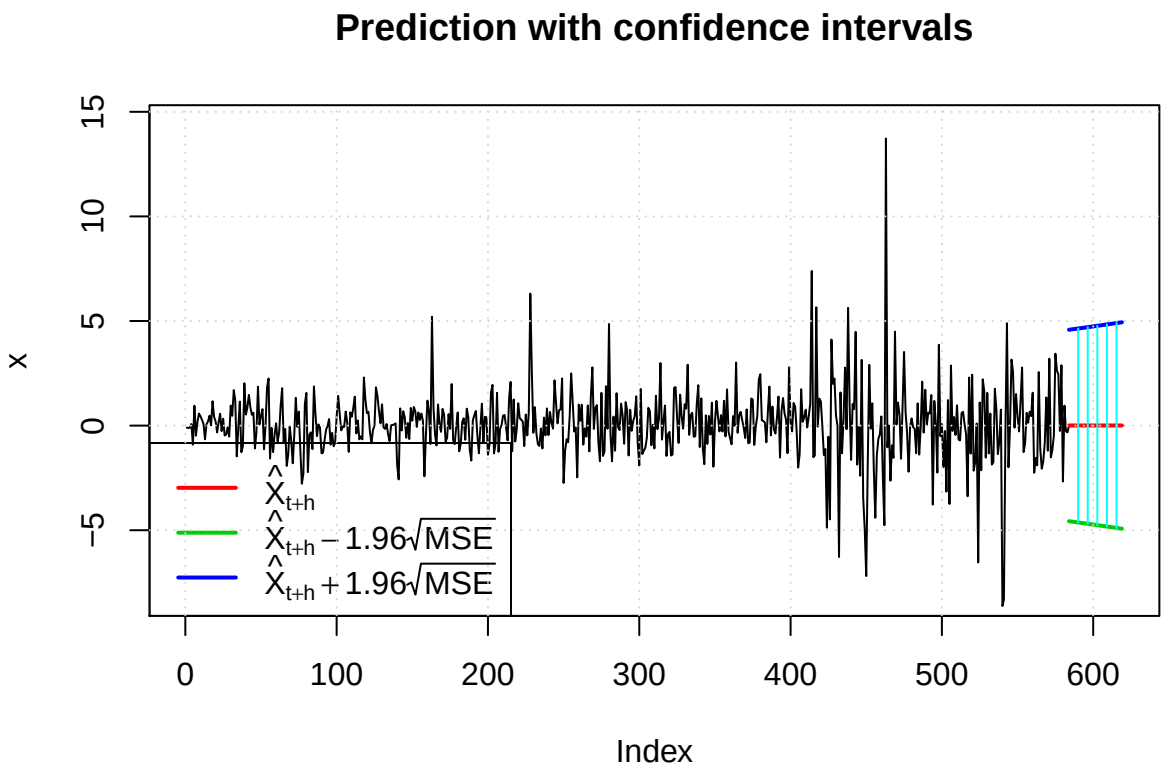
Having acknowledged the confidence interval problem on the prediction caused by the non-stationary variance of the financial search time series, we want to use our fitted GARCH model to predict the mean and variance of the residuals of the series.

The results of this prediction are a better estimate of the 95%confidence interval of the residuals of the global warming time series after modeling with our SAMIMA (1,1,1,0,0,0) model. We note that the volatility is predicted by the GARCH model to be in the range of -5 to 5, far wider than the predicted by the SARIMA model but consistent with the volatility observed towards the end of the original time series.

```
data.garch.fit = garchFit(~garch(1, 1), data = data.res, trace = FALSE)
```

```
## Warning in sqrt(diag(fit$cvar)): NaNs produced
```

```
gw.garch.pred <- predict(data.garch.fit, n.ahead = 36, plot = TRUE)
```



Using our GARCH model fitted on the residuals, we now plot the predicted confidence intervals obtained with GARCH modeling over the original fitted SARIMA(0,1,0,0,0,0) series. Replace the 95% confidence interval of the fitted SARIMA with the GARCH confidence interval The mean series of the 36 steps ahead predctions obtained from the fitted first difference model are:

```
data.fit.ahead.fcast$mean
```

```
## Time Series:
## Start = 2333
## End = 2368
## Frequency = 1
##  [1] 68.05606 68.03214 68.00823 67.98434 67.96046 67.93659 67.91274
##  [8] 67.88890 67.86507 67.84126 67.81746 67.79367 67.76990 67.74614
## [15] 67.72239 67.69866 67.67494 67.65124 67.62755 67.60387 67.58020
## [22] 67.55655 67.53292 67.50929 67.48568 67.46208 67.43850 67.41493
## [29] 67.39137 67.36783 67.34430 67.32078 67.29728 67.27379 67.25031
## [36] 67.22685
```

The corresponding lower and upper confidence intervals obtained from teh GARCH model are:

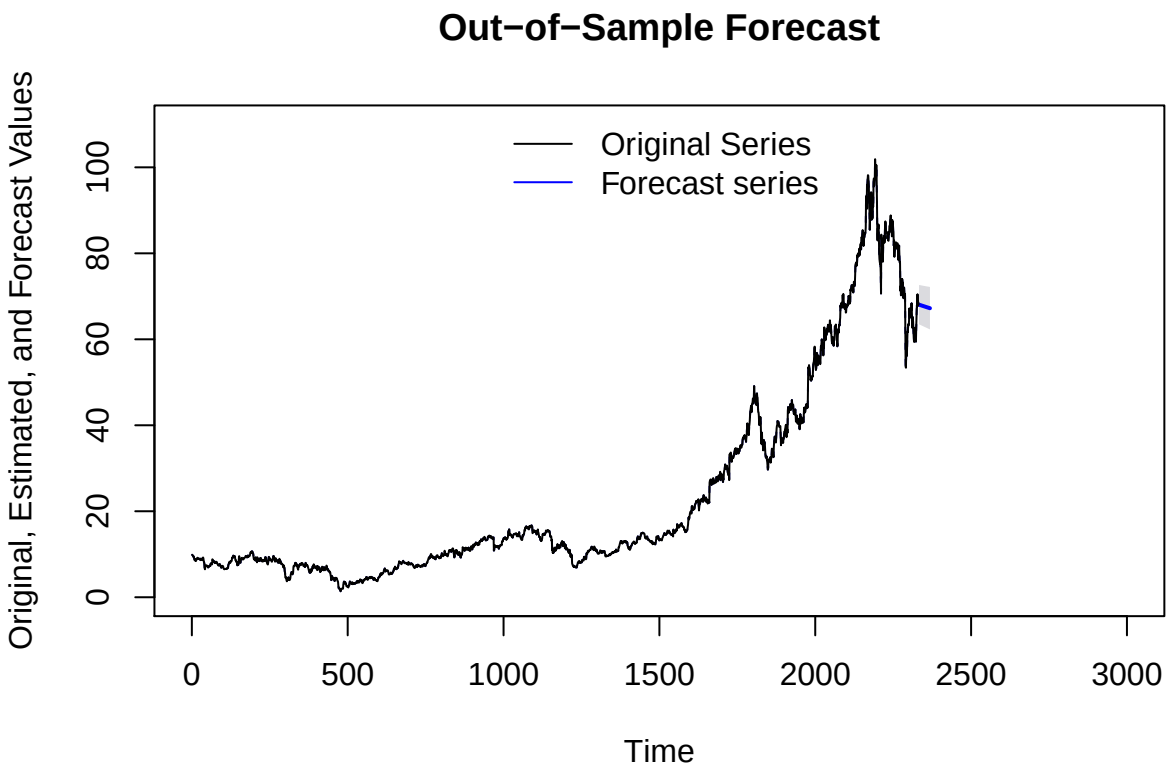
```
gw.garch.pred$lowerInterval
```

```
##  [1] -4.569388 -4.579264 -4.589162 -4.599080 -4.609019 -4.618979 -4.628961
##  [8] -4.638963 -4.648986 -4.659031 -4.669097 -4.679184 -4.689293 -4.699422
## [15] -4.709574 -4.719746 -4.729940 -4.740156 -4.750393 -4.760652 -4.770933
## [22] -4.781235 -4.791560 -4.801906 -4.812273 -4.822663 -4.833075 -4.843509
## [29] -4.853965 -4.864442 -4.874943 -4.885465 -4.896009 -4.906576 -4.917166
## [36] -4.927777
```

```
gw.garch.pred$upperInterval
```

```
##  [1] 4.581534 4.591411 4.601308 4.611226 4.621165 4.631126 4.641107
##  [8] 4.651109 4.661133 4.671177 4.681243 4.691330 4.701439 4.711568
## [15] 4.721720 4.731892 4.742086 4.752302 4.762539 4.772798 4.783079
## [22] 4.793381 4.803706 4.814052 4.824419 4.834809 4.845221 4.855655
## [29] 4.866111 4.876589 4.887089 4.897611 4.908156 4.918722 4.929312
## [36] 4.939923
```

```
data.fit.ahead.fcast$lower[, 1] <- data.fit.ahead.fcast$mean
data.fit.ahead.fcast$upper[, 1] <- data.fit.ahead.fcast$mean
data.fit.ahead.fcast$lower[, 2] <- gw.garch.pred$lowerInterval +
    data.fit.ahead.fcast$mean
data.fit.ahead.fcast$upper[, 2] <- gw.garch.pred$upperInterval +
    data.fit.ahead.fcast$mean
par(mfrow = c(1, 1))
plot(data.fit.ahead.fcast, lty = 2, col = "navy", main = "Out-of-Sample Forecast",
    ylab = "Original, Estimated, and Forecast Values", xlim = c(0,
        3000), ylim = c(0, 110))
par(new = T)
plot.ts(data$DXCM, axes = F, lty = 1, xlim = c(0, 3000), ylim = c(0,
    110), ylab = "")
leg.txt <- c("Original Series", "Forecast series")
legend("top", legend = leg.txt, lty = 1, col = c("black", "blue"),
    bty = "n", cex = 1)
```




```
par(new = T)
```

Part 3 (25 points): Forecast the Web Search Activity for global Warming

Data Analysis

1. The time series has weekly 630 values starting at 1/4/04 and ending at 1/24/16. The minimum value is -0.551 and the maximum value is 4.104.
2. Time series plot shows that the series is very persistent, The series is basically flat from 2004 to 2012. After 2012, there is a sharp trend upward. There is more volatility after 2012. There are spikes and dips which could be seasonal with a yearly frequency. The series is not stationary.
3. Histogram shows is heavily positively skewed with most values between -0.551 and -0.3.
4. ACF of the series has correlations at around 0.75 for almost 25 lags.
5. PACF drops off immediately after first lag. There are 4 points that fall outside the 95% confidence interval (blue lines) at lags 3, 5, 11 and 14.

Model Selection Process

1. **Try AR models.** Use the ar() command in R to find AR(p) models or order p that potentially fit the time series. This command output a model or order 15, but looking at the difference in AICs, the AIC for the AR(1) model is not that different from the AIC of the AR(15), so for parsimony we will try using that one. Check if the residuals look like white noise.

- Histogram: Yes. This looks like a normal distribution.
- Fitted vs. Residuals: No. The plot does not look like an evenly distributed cloud.
- Plot: No. The plot does not look random, there is a lot of volatility on the right hand side of the graph.
- ACF: No. The ACF drops off after lag 0, but has only a few lags where the correlation comes out of the 95% CI.
- PACF: No. The PACF shows correlation with several values outside of the 9%% CI. In summary, the residuals for this model do not look like white noise, so there is more variation that could be explained by our model.
  The In-Sample fit of this estimated model matches the original model very well as evidenced in the plot.

2. **Try ARIMA models.** Use the get.best.arima() function which will try models with c(p,d,q) where p=0-4, d=0-2 and q=0-2. And then we can print out a list in ascending order by AIC of the 20 models with the lowest AIC. Inspecty these models for parsimony and select one with a good AIC and a small number of parameters. The best model output from the function had an AIC of -1058.794 with parameters = c(1, 2, 2). For parsimony a model of c(1,1,1) was chosen with an AIC of -1032.364 which is not that different from the best AIC. Check if the residuals look like white noise. No, the residuals do not look like white noise. They exhibit the same characteristics as the AR(1) model from step 1.
   The In-Sample fit of this estimated model matches the original model very well as evidenced in the plot.

3. **Try SARIMA models.** From the plot of the original series, it looks like this series has a seasonal component with a 52-week periodicity. Use the get.best.sarima() function with parameters c(2,2,2,2,2,2). The best AIC output is -1276.817 with a model of c(1, 2, 2, 1, 0, 2). For parsimony try running get.best.sarima() with c(1,1,1,1,1,1). A parsimonious model from this output is c(0, 1, 1, 1, 0, 1) with AIC -1246.412 which is very close to the AIC output from c(2,2,2,2,2,2). For parsimony we will choose c(0, 1, 1, 1, 0, 1) and check the residuals. No, the residuals do not look like white noise. They exhibit the same characteristics as the AR(1) model from step 1. The residuals exhibit evidence of time
   The In-Sample fit of this estimated model matches the original model very well as evidenced in the plot.

4. **Backtesting.**

5. **Forecast the model.** Using the SARIMA model, we will make the requested 12-step ahead forecast of the model. The forecast looks like it captures the seasonality of the model as it matches the upward trend and the seasonal volatility.

6. **GARCH.** Since the residuals look like they have time-varying volatility, we will use GARCH to model the residuals. The ACF of the squared residuals shows no obvious patterns or significant values. The model has reached and acceptable fit to the original time series.

7. **Update the forecast with GARCH.** To add the GARCH information to the forecast, we create a GARCH forecast. We then take the forecast from the SARIMA model and update it with the GARCH confidence intervals.

Conclusion

This time series is satisfactorily modeled with a SARMIMA(x,x,x,x,x,x) model to hande trends and seasonality and a GARCH model to handle the time-varying volatility. We observe that the forecasts of the model are consisent with the seasonality of the original series and the forecasts are within the 95% confidence interval produced by GARCH. This gives us confidence that this model will give us decent forecasts for the original time series.

```
# Read in the time series data
glob.warm = read.csv("globalWarming.csv", header = TRUE)
glob.warm.ts = ts(glob.warm$data.science, start = 2004, frequency = 52)
# glob.warm.ts = ts(glob.warm$data.science) Print descriptive
# statistics
str(glob.warm.ts)
```

```
##  Time-Series [1:630] from 2004 to 2016: -0.44 -0.474 -0.423 -0.551 -0.486 -0.551 -0.453 -0.462 -0.55
```

```
summary(glob.warm.ts)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.551000 -0.506000 -0.485000  0.000038 -0.200000  4.104000
```

```
cbind(head(glob.warm.ts), tail(glob.warm.ts))
```

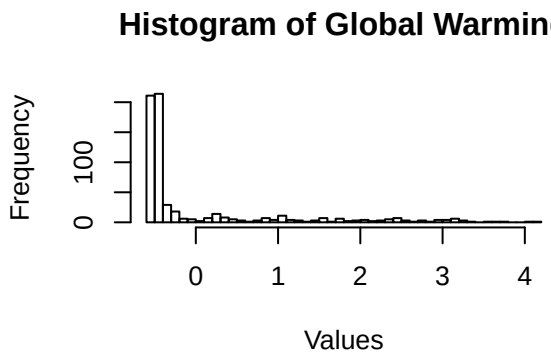
```
##        [,1]  [,2]
## [1,] -0.440 2.227
## [2,] -0.474 2.360
## [3,] -0.423 3.662
## [4,] -0.551 3.721
## [5,] -0.486 4.087
## [6,] -0.551 4.104
```

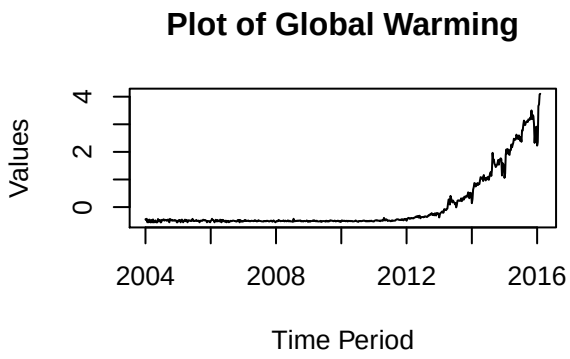
```
quantile(as.numeric(glob.warm.ts), c(0.01, 0.05, 0.1, 0.25, 0.5,
    0.75, 0.9, 0.95, 0.99))
```

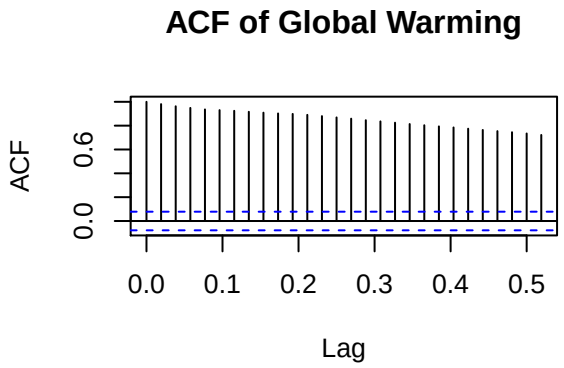
```
##       1%       5%      10%      25%      50%      75%      90%      95%
## -0.55100 -0.53220 -0.51900 -0.50600 -0.48500 -0.20000  1.68410  2.48055
##      99%
##  3.28021
```

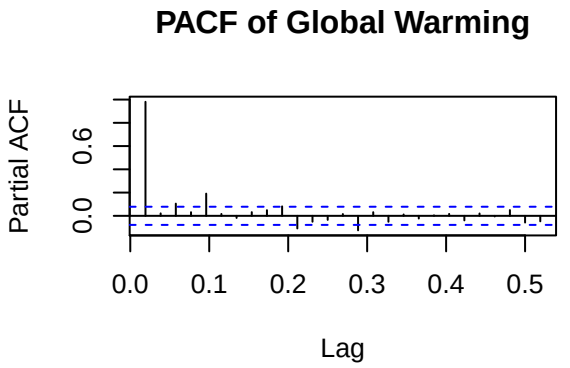
```
# Plot the time series
plot.time.series(glob.warm.ts, 50, "Global Warming")
```

```
##  Time-Series [1:630] from 2004 to 2016: -0.44 -0.474 -0.423 -0.551 -0.486 -0.551 -0.453 -0.462 -0.55
```





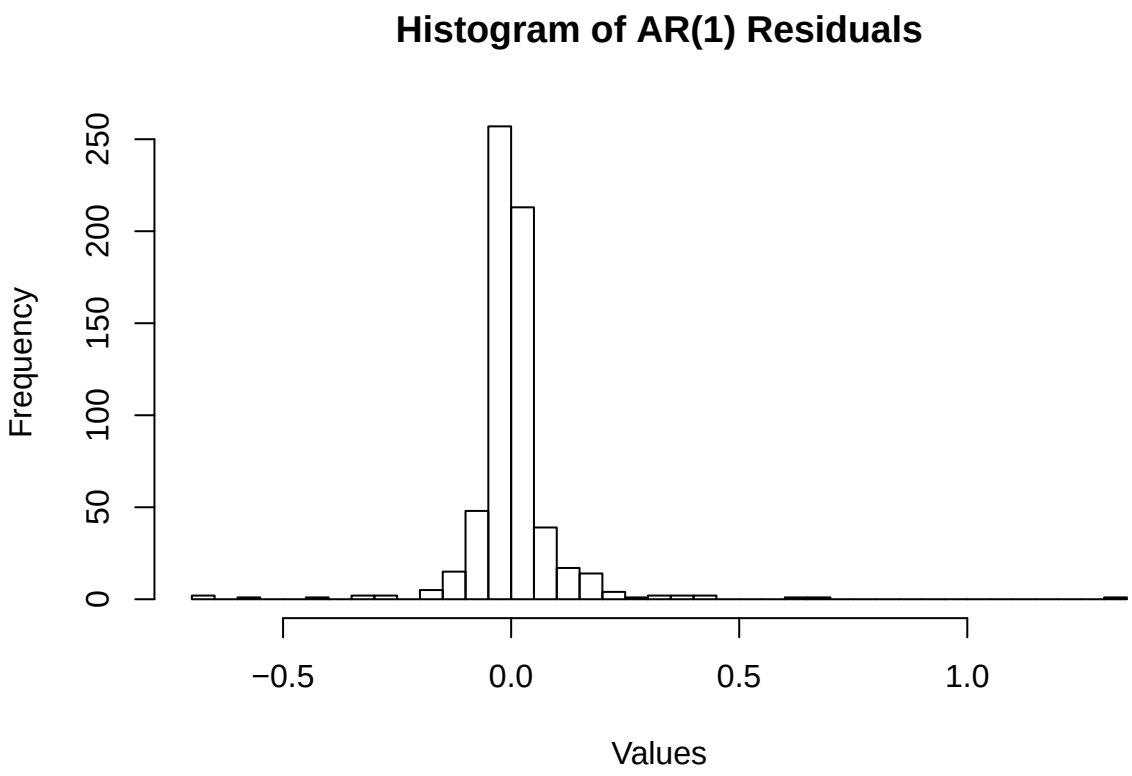


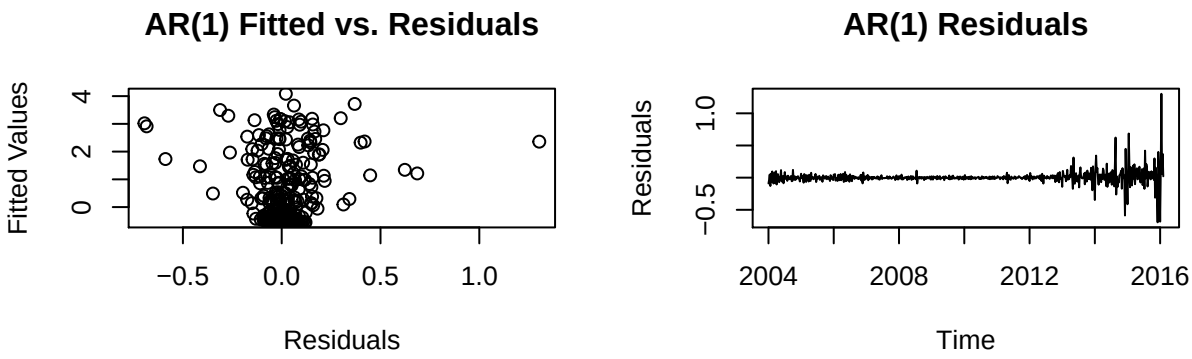


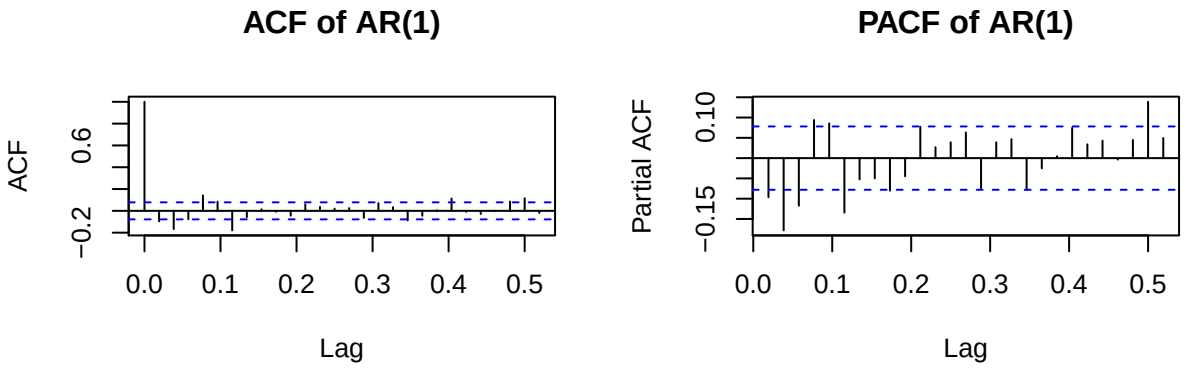

```
### 1. Try AR models
glob.warm.ar = estimate.ar(glob.warm.ts)
```

```
## [1] "Difference in AICs"
##           0           1           2           3           4           5
## 2084.447812   29.847743   31.560248   26.579621   27.960796    6.553854
##           6           7           8           9          10          11
##    8.386263   10.176681   11.540473   12.035569    9.848063    4.382889
##          12          13          14          15          16          17
##    4.754476    5.996066    7.842039    0.000000    1.380591    1.728222
##          18          19          20          21          22          23
##    3.638626    5.291781    7.280008    9.104280   10.136039   11.875658
##          24          25          26          27
##   13.856501   14.302766   14.191716   14.781132
## [1] "AR parameters"
##  [1]  0.944522755 -0.084770519  0.084153344 -0.171500315  0.188422207
##  [6]  0.058499722 -0.055671998 -0.008980095 -0.033122819  0.204945468
## [11] -0.084654024 -0.006099723 -0.059202741  0.132988289 -0.124502569
## [1] "AR order"
## [1] 15
```

```
glob.warm.ar1 = arima(glob.warm.ts, order = c(1, 0, 0))
# Plot the residuals
plot.residuals.ts(glob.warm.ar1, "AR(1)")
```


Histogram of AR(1) Residuals
Frequency
0
50
100
150
200
250
−0.5
0.0
0.5
1.0
Values

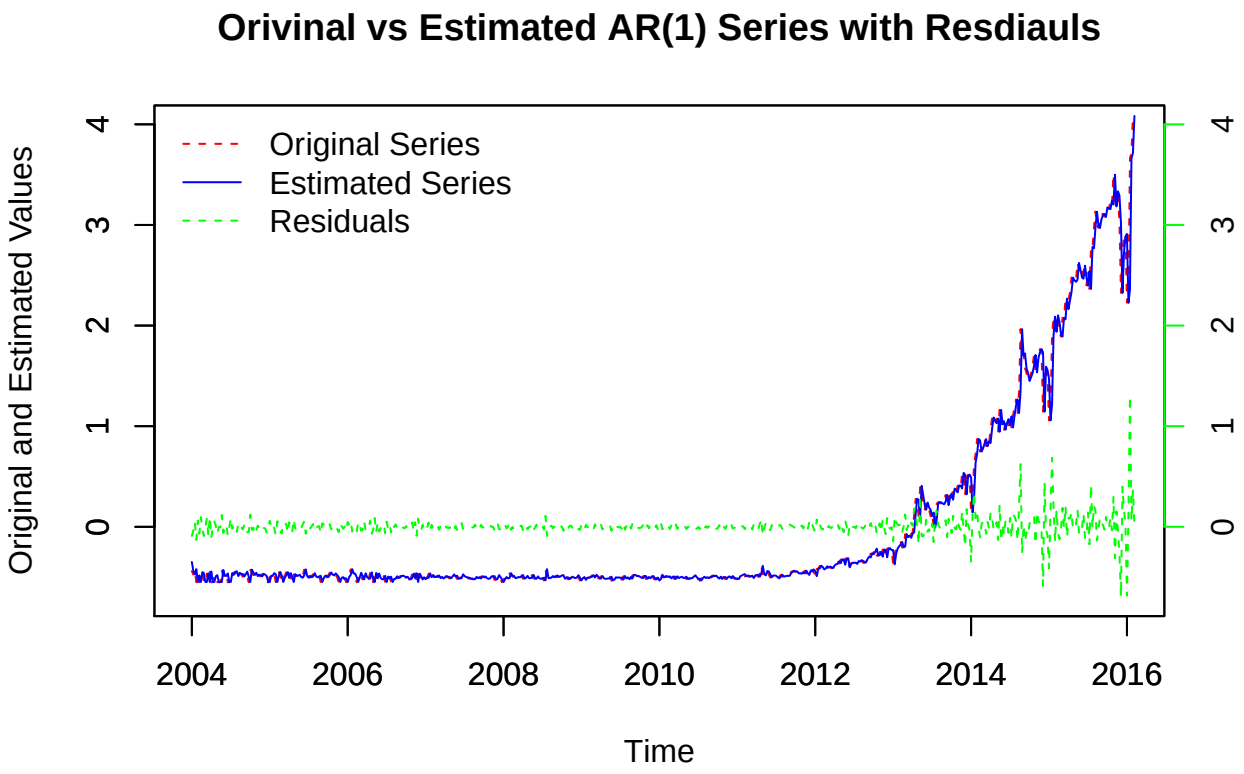




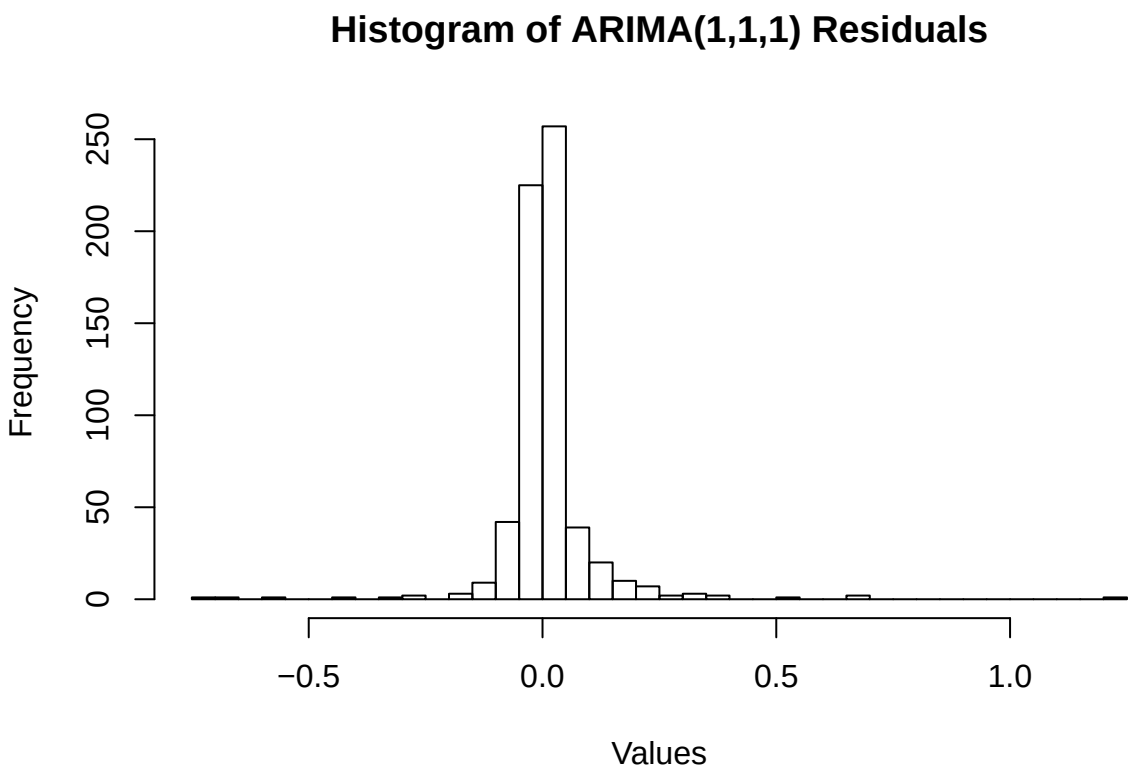

```
##
##  Box-Ljung test
##
## data:  x.mod$residuals
## X-squared = 5.8789, df = 1, p-value = 0.01532
```

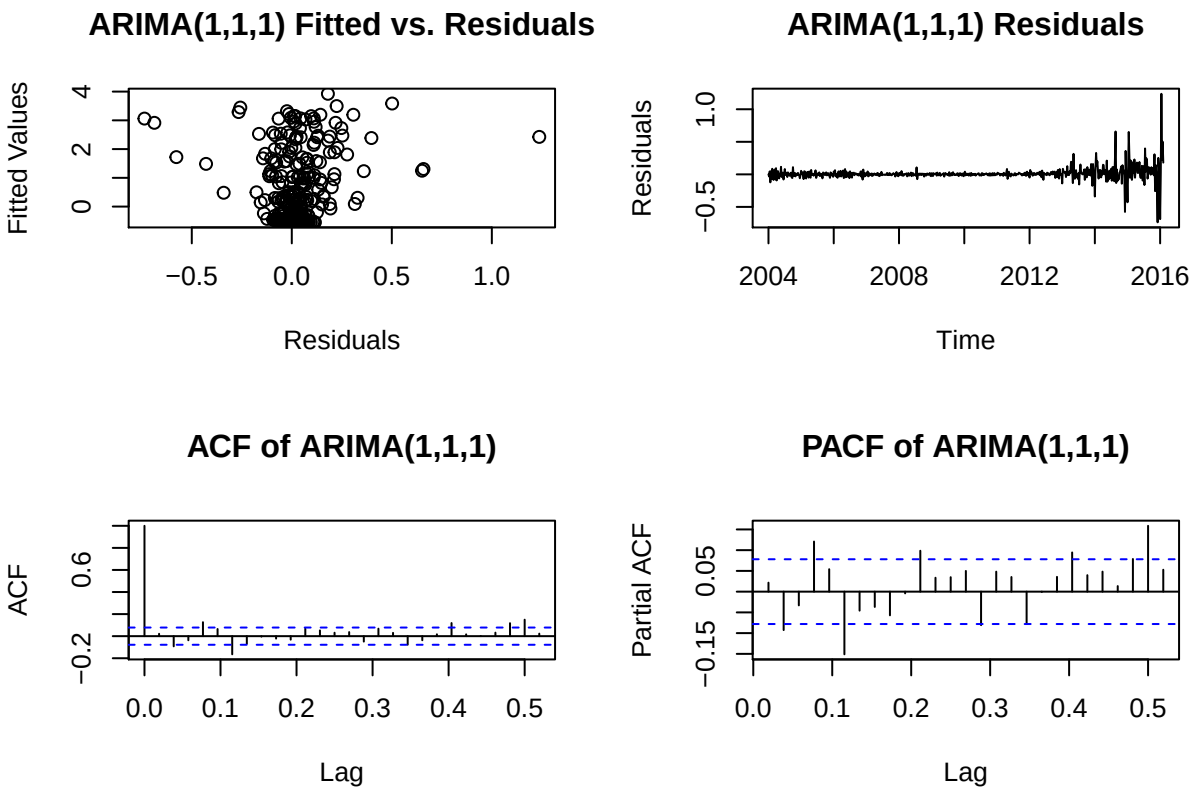
```
# Plot the In-sample fit
plot.orig.model.resid(glob.warm.ts, glob.warm.ar1, "AR(1)", c(2004,
    2016), c(-0.7, 4))
```

```
##
## Descriptive Stat
## ========================================
## Statistic        N  Mean  St. Dev. Min  Max
## ----------------------------------------
## x.ts            630 0.000   1.0    -0.6 4.1
## fitted.x.mod.   630 -0.01   1.0    -0.5 4.1
## x.mod.residuals 630 0.01    0.1    -0.7 1.3
## ----------------------------------------
```




```
### 2. Try ARIMA models gw.arima.best <-
### get.best.arima(glob.warm.ts, maxord=c(4,2,2)) Print the top
### 20 best models based on AIC
### gw.arima.best$others[order(gw.arima.best$others$aics)[1:20],]
glob.warm.arima = arima(glob.warm.ts, order = c(1, 1, 1))
# Plot the residuals
plot.residuals.ts(glob.warm.arima, "ARIMA(1,1,1)")
```

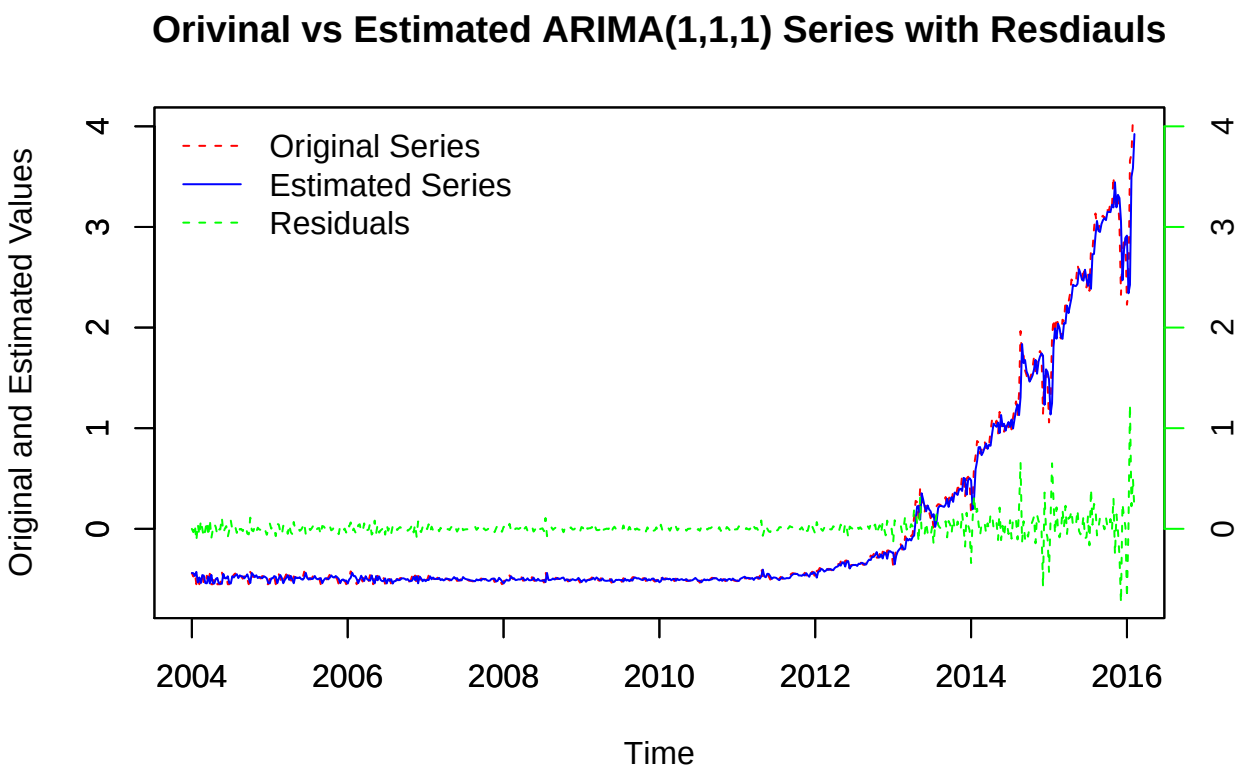

Histogram of ARIMA(1,1,1) Residuals
Frequency
0
50
100
150
200
250
−0.5
0.0
0.5
1.0
Values



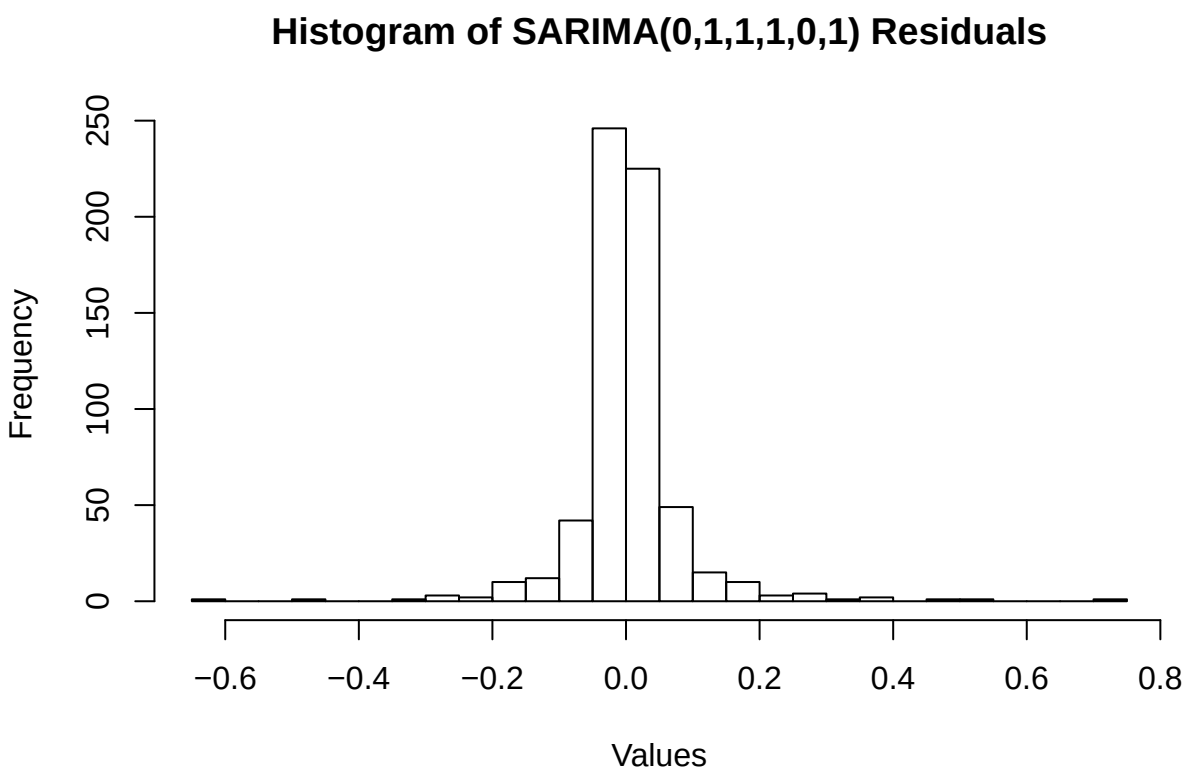

```
##
##  Box-Ljung test
##
## data:  x.mod$residuals
## X-squared = 0.29725, df = 1, p-value = 0.5856
```

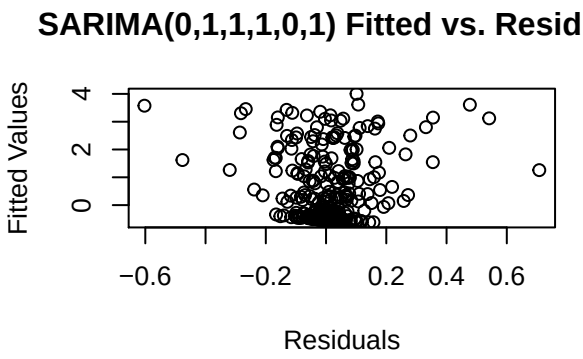
```
# Plot the In-sample fit
plot.orig.model.resid(glob.warm.ts, glob.warm.arima, "ARIMA(1,1,1)",
    c(2004, 2016), c(-0.7, 4))
```

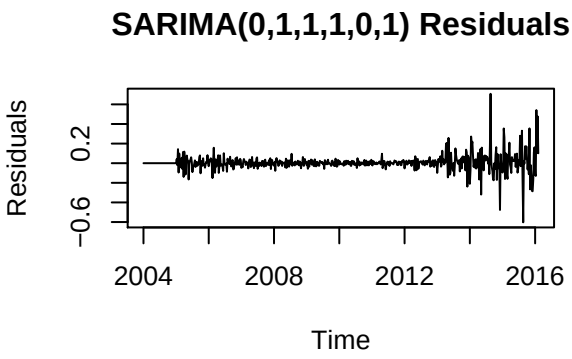
```
##
## Descriptive Stat
## ========================================
## Statistic        N  Mean  St. Dev. Min  Max
## ----------------------------------------
## x.ts            630 0.000   1.0    -0.6 4.1
## fitted.x.mod.   630 -0.01   1.0    -0.5 3.9
## x.mod.residuals 630 0.01    0.1    -0.7 1.2
## ----------------------------------------
```

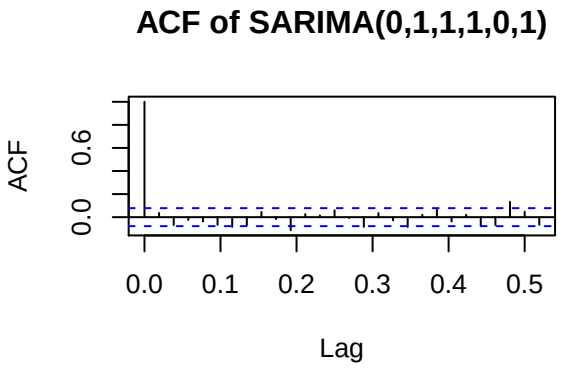


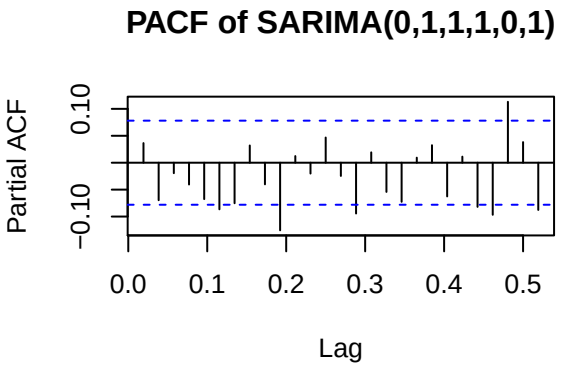

```
### 3. Try SARIMA models gw.seas.best <-
### get.best.sarima(glob.warm.ts, maxord=c(2,2,2,2,2,2), 52)
### Print the top 20 best models based on AIC
### gw.seas.best$others[order(gw.seas.best$others$aics)[1:20],]

# gw.seas.best1 <- get.best.sarima(glob.warm.ts,
# maxord=c(1,1,1,1,1,1), 52) Print the top 20 best models
# based on AIC
# gw.seas.best1$others[order(gw.seas.best1$others$aics)[1:20],]
glob.warm.arima.seas = arima(glob.warm.ts, order = c(0, 1, 1),
    seas = list(order = c(1, 0, 1), 52), method = "CSS")
# Plot the residuals
glob.warm.arima.seas.res = glob.warm.arima.seas$residuals
plot.residuals.ts(glob.warm.arima.seas, "SARIMA(0,1,1,1,0,1)")
```


Histogram of SARIMA(0,1,1,1,0,1) Residuals
Frequency
0
50
100
150
200
250
−0.6
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8
Values



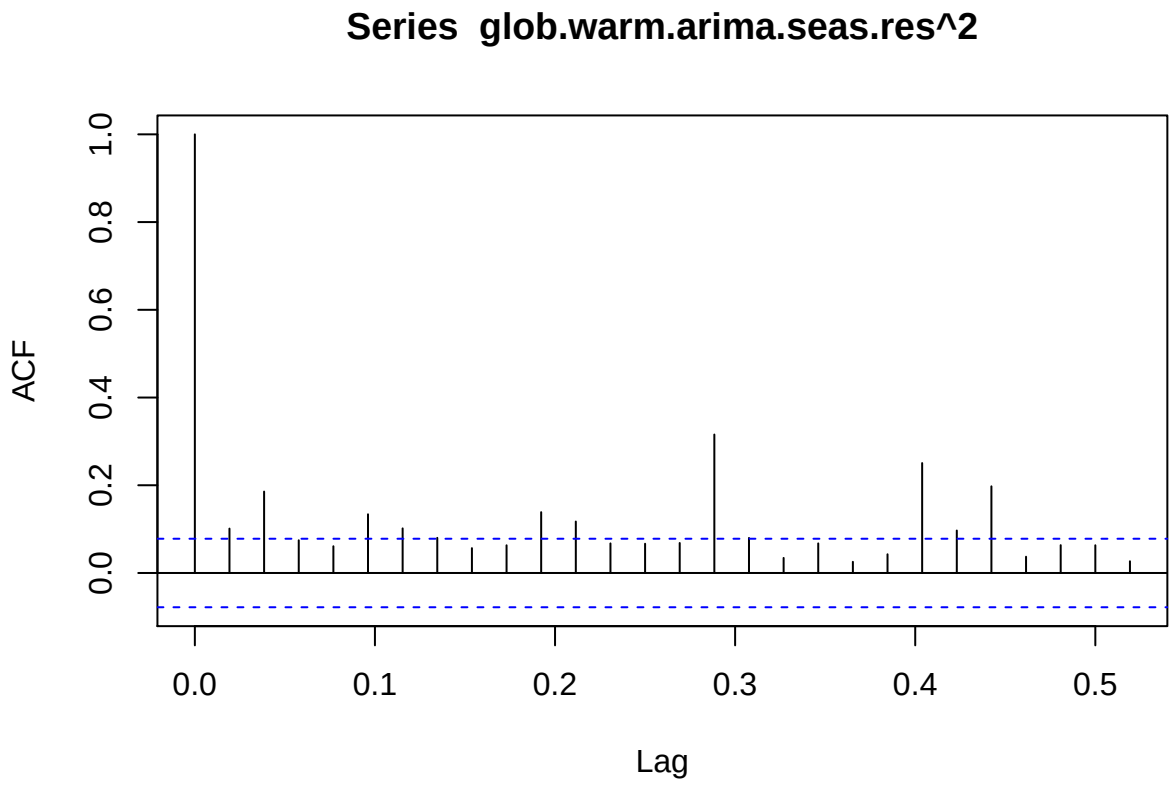






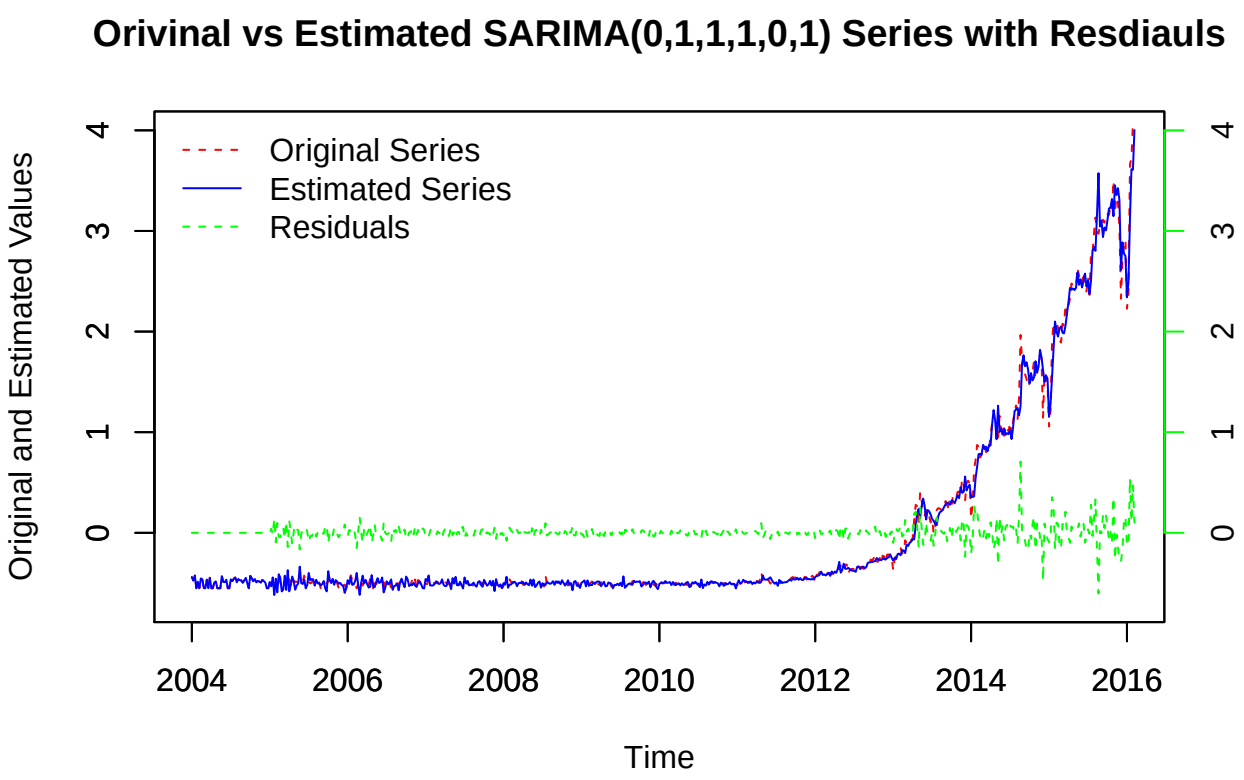

```
##
##  Box-Ljung test
##
## data:  x.mod$residuals
## X-squared = 0.8408, df = 1, p-value = 0.3592
```

```
par(mfrow = c(1, 1))
acf(glob.warm.arima.seas.res^2)
```




```
# Plot the In-sample fit
plot.orig.model.resid(glob.warm.ts, glob.warm.arima.seas, "SARIMA(0,1,1,1,0,1)",
    c(2004, 2016), c(-0.7, 4))
```

```
##
## Descriptive Stat
## ==========================================
## Statistic        N  Mean  St. Dev. Min  Max
## ------------------------------------------
## x.ts            630 0.000   1.0    -0.6 4.1
## fitted.x.mod.   630 -0.01   1.0    -0.6 4.0
## x.mod.residuals 630 0.01    0.1    -0.6 0.7
## ------------------------------------------
```

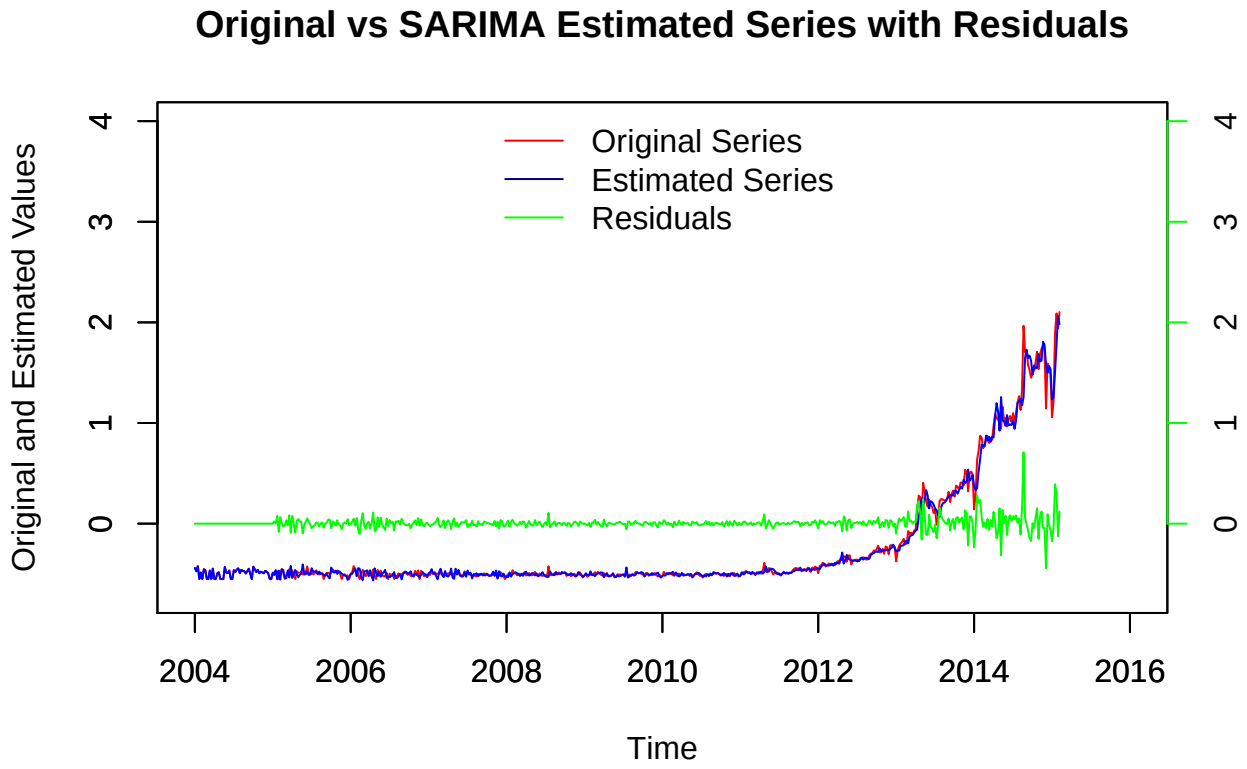



```
### 4. Backtesting
glob.warm.bt.ts = ts(glob.warm.ts[1:(length(glob.warm.ts) - 52)],
    start = 2004, frequency = 52)
glob.warm.arima.seas.bt = arima(glob.warm.bt.ts, order = c(0,
    1, 1), seas = list(order = c(1, 0, 1), 52), method = "CSS")
df = cbind(glob.warm.bt.ts, fitted(glob.warm.arima.seas.bt),
    glob.warm.arima.seas.bt$resid)
colnames(df) = c("orig_series", "fitted_vals", "resid")
head(df)
```

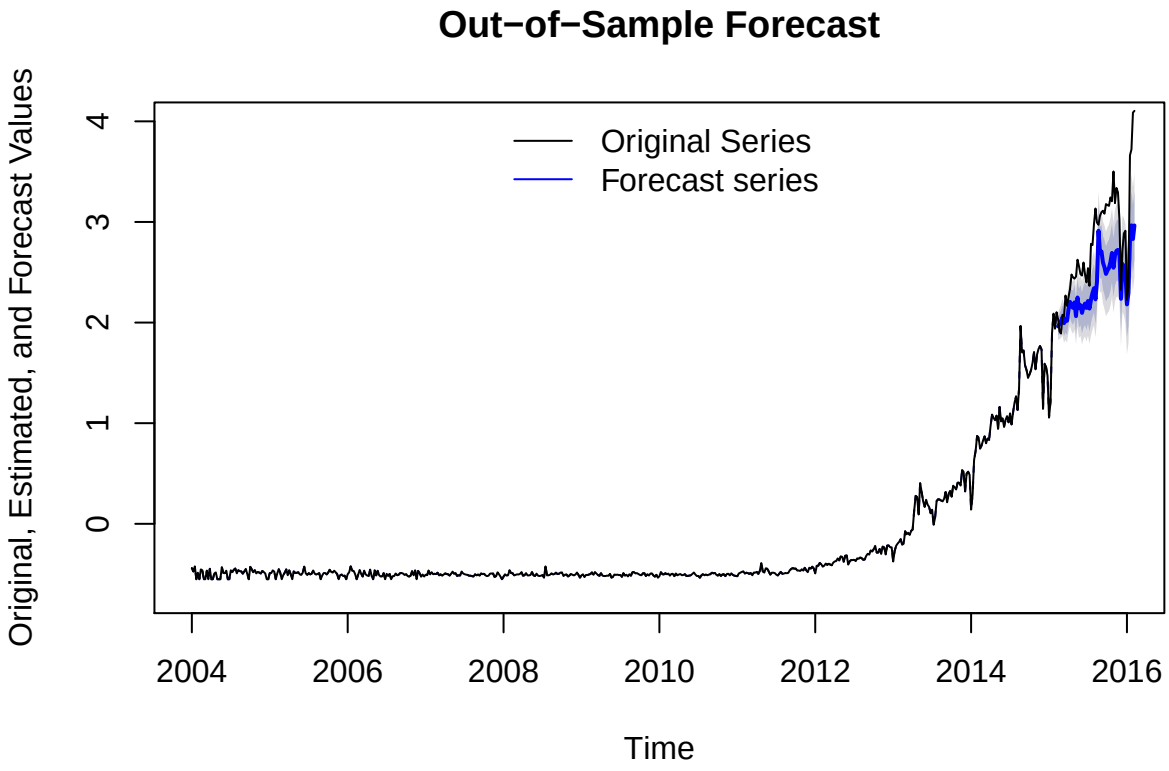
```
##      orig_series fitted_vals resid
## [1,]      -0.440      -0.440     0
## [2,]      -0.474      -0.474     0
## [3,]      -0.423      -0.423     0
## [4,]      -0.551      -0.551     0
## [5,]      -0.486      -0.486     0
## [6,]      -0.551      -0.551     0
```

```
# Plot the original and estimate series
par(mfrow = c(1, 1))
plot.ts(df[, "orig_series"], col = "red", main = "Original vs SARIMA Estimated Series with Residuals",
    ylab = "Original and Estimated Values", xlim = c(2004, 2016),
    ylim = c(-0.7, 4))
par(new = T)
plot.ts(df[, "fitted_vals"], col = "blue", axes = T, xlab = "",
```

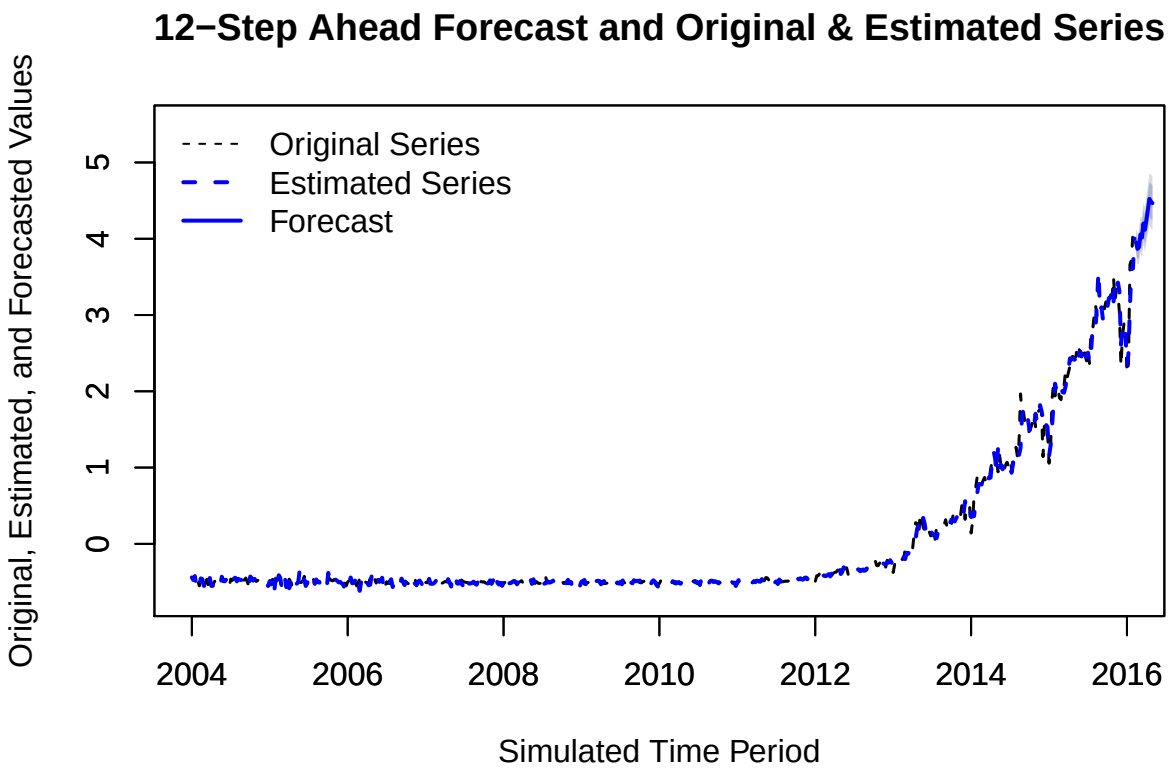
```
    ylab = "", xlim = c(2004, 2016), ylim = c(-0.7, 4))
leg.txt <- c("Original Series", "Estimated Series", "Residuals")
legend("top", legend = leg.txt, lty = 1, col = c("red", "navy",
    "green"), bty = "n", cex = 1)
par(new = T)
plot.ts(df[, "resid"], axes = F, xlab = "", ylab = "", col = "green",
    xlim = c(2004, 2016), ylim = c(-0.7, 4), pch = 1)
axis(side = 4, col = "green")
mtext("Residuals", side = 4, line = 2, col = "green")
```



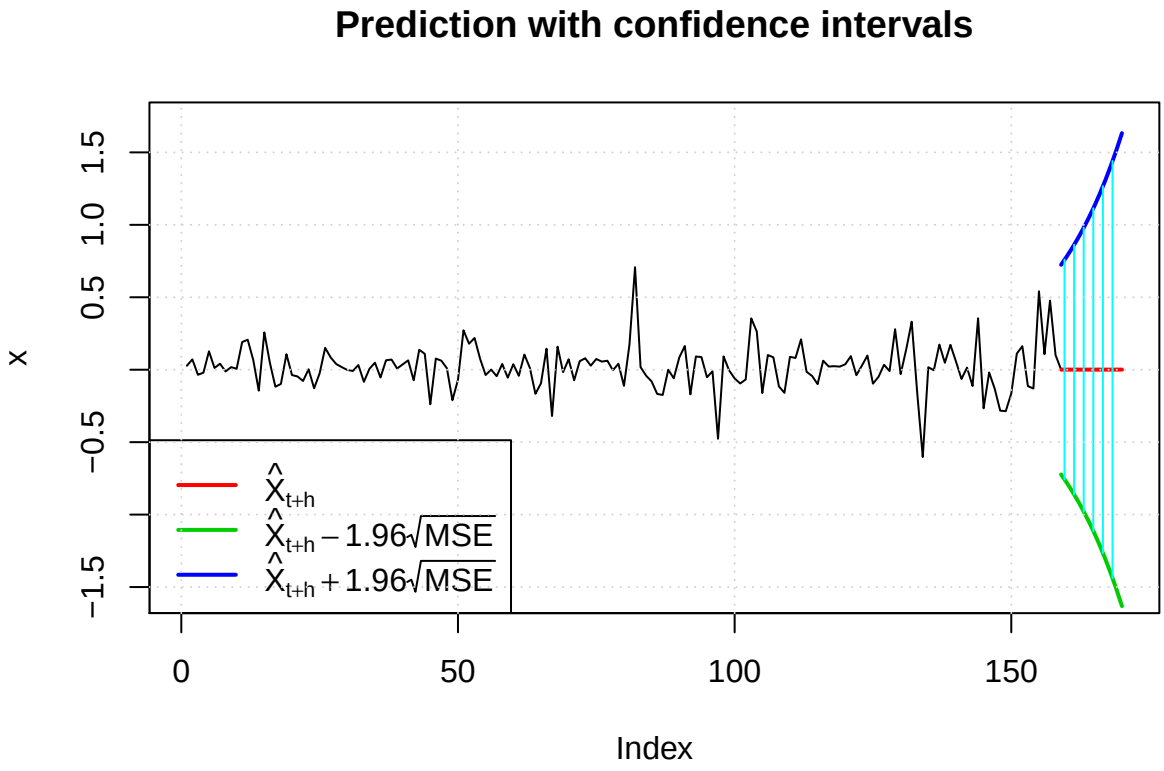

```
glob.warm.arima.bt.fcast = forecast.Arima(glob.warm.arima.seas.bt,
    h = 52)
par(mfrow = c(1, 1))
plot(glob.warm.arima.bt.fcast, lty = 2, col = "navy", main = "Out-of-Sample Forecast",
    ylab = "Original, Estimated, and Forecast Values", xlim = c(2004,
        2016), ylim = c(-0.7, 4))
par(new = T)
plot.ts(glob.warm.ts, axes = F, lty = 1, col = "black", xlim = c(2004,
    2016), ylim = c(-0.7, 4), ylab = "")
leg.txt <- c("Original Series", "Forecast series")
legend("top", legend = leg.txt, lty = 1, col = c("black", "blue"),
    bty = "n", cex = 1)
```



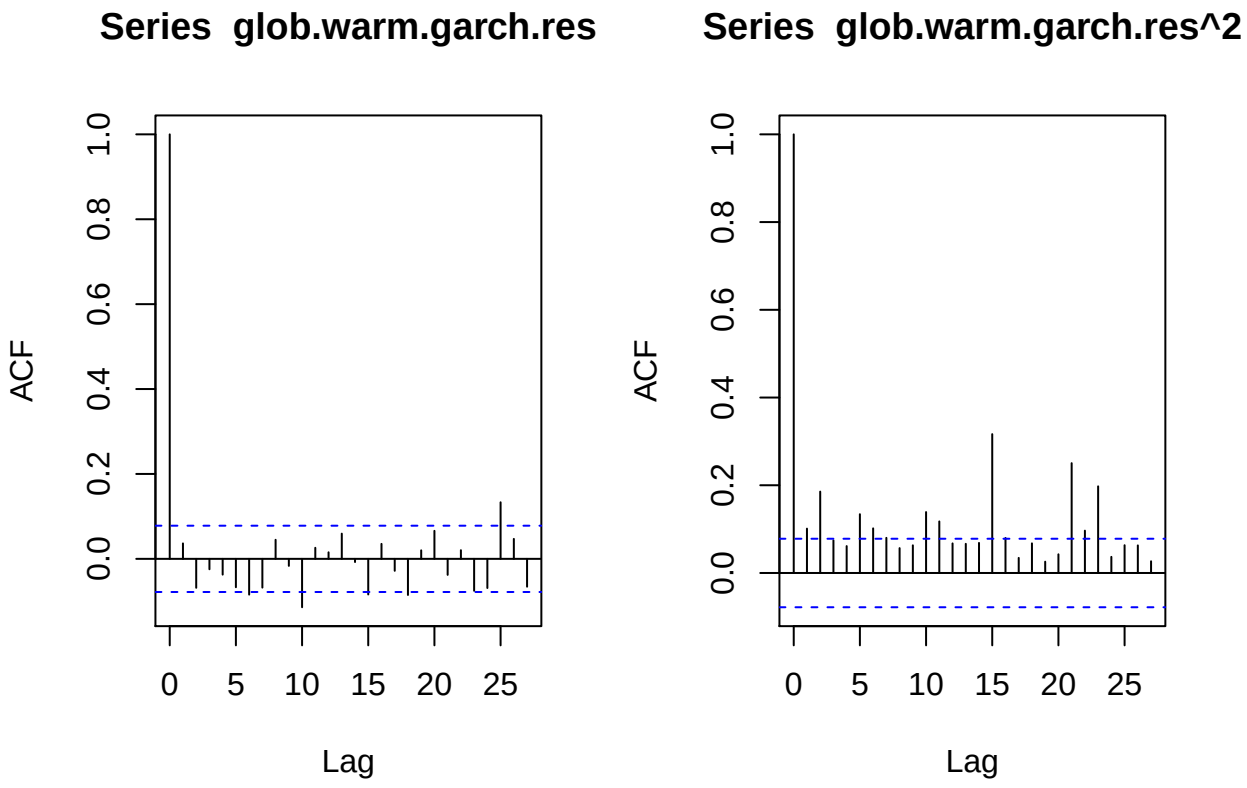

```
### 5. Forecast the model
glob.warm.arima.fcast = forecast.Arima(glob.warm.arima.seas,
    h = 12)
plot.model.forecast(glob.warm.arima.seas, glob.warm.arima.fcast,
    "12", c(2004, 2016), c(-0.7, 5.5))
```



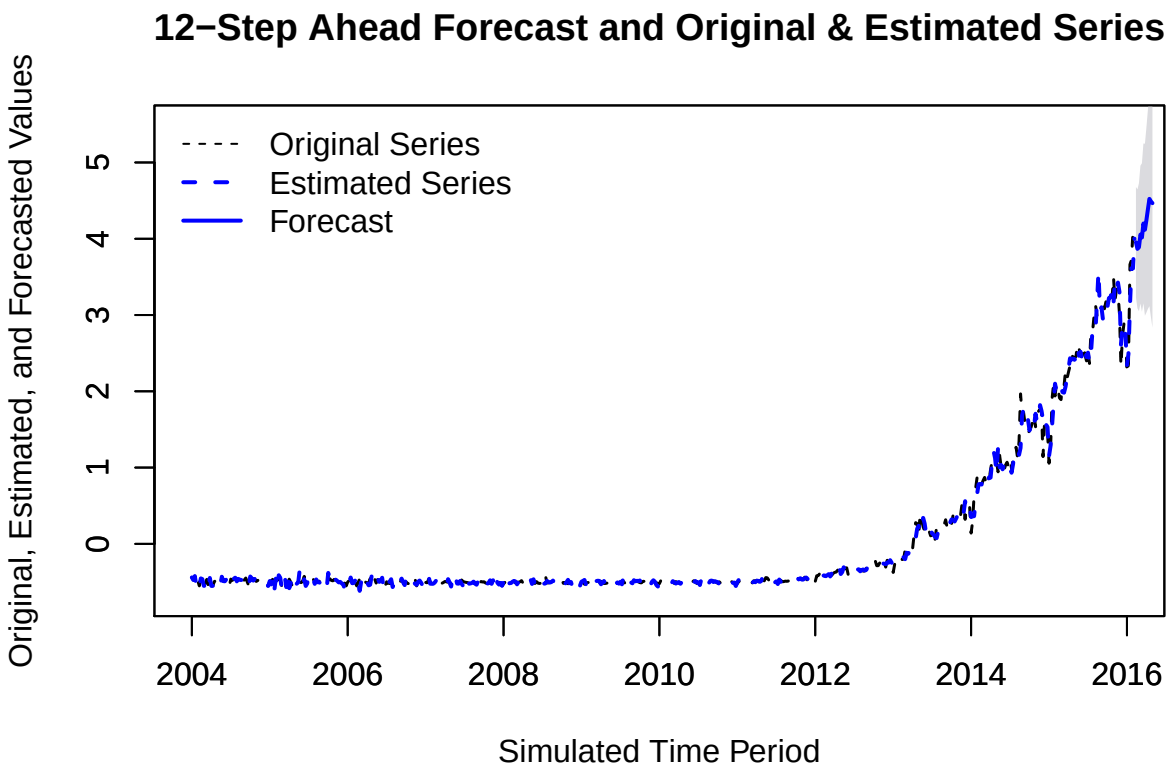

```
### 6. GARCH
glob.warm.garch.fit = garchFit(~garch(1, 1), data = glob.warm.arima.seas.res,
    trace = FALSE)
gw.garch.pred <- predict(glob.warm.garch.fit, n.ahead = 12, plot = TRUE)
```




```
glob.warm.garch.res <- glob.warm.garch.fit@residuals
par(mfrow = c(1, 2))
acf(glob.warm.garch.res)
acf(glob.warm.garch.res^2)
```




```
### 7. Update the forecast with GARCH Clear the 80% confident
### interval of the fitted ARIMA by setting all values to the
### predicted mean of it And set the values of the 95%
### confidence interval to those that came out of the GARCH
### model
glob.warm.arima.fcast$lower[, 1] <- glob.warm.arima.fcast$mean
glob.warm.arima.fcast$upper[, 1] <- glob.warm.arima.fcast$mean
glob.warm.arima.fcast$lower[, 2] <- gw.garch.pred$lowerInterval +
    glob.warm.arima.fcast$mean
glob.warm.arima.fcast$upper[, 2] <- gw.garch.pred$upperInterval +
    glob.warm.arima.fcast$mean

plot.model.forecast(glob.warm.arima.seas, glob.warm.arima.fcast,
    "12", c(2004, 2016), c(-0.7, 5.5))
```


12−Step Ahead Forecast and Original & Estimated Series
Original, Estimated, and Forecasted Values
0
1
2
3
4
5
Original Series
Estimated Series
Forecast
2004
2006
2008
2010
2012
2014
2016
Simulated Time Period