

W271 Lab 3 Spring 2016

Megan Jasek, Rohan Thakur, Charles Kekeh

Sunday, April 24, 2016

Below, we define some functions we will be using in the problem set:

```
# 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 = frequency(x.ts)) {
  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 in 0:maxord[3]) {
    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("Original vs Estimated",
    model_name, "Series with Residuals", 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 1 (25 points): Modeling House Values

Step 1 - Univariate Analysis

1. **Crime Rate** - This variable is positively skewed, with 90% of datapoints having a crime rate below 11.2%, but outliers above that going upto 89%. We take the log to create a new variable before proceeding.
2. **nonRetailBusiness** - Has a suspiciously high mode at 0.18, which may indicate that a lot of the data points come from the same neighbourhood, which would explain the high number of occurrences of a single value.
3. **withWater** - This is a categorical variable. 6.75% of homes in the given sample are in neighbourhoods within 5 miles of a water body.
4. **ageHouse** - This value is in percentage terms and not in strict proportion like other variables in the dataset. Over 50% of the houses in the dataset are in neighbourhoods with a proportion of houses older than 1950 that is greater than 78%
5. **distanceToCity** - 75% of the houses are less than 15 miles away from a city, and 90% are less than 25 miles away. However, the variable has a large outlier, which is almost 55 miles away from a city. We take a log of the variable before proceeding, in order to make it more evenly distributed.
6. **distanceToHighway** - Definition of the variable is not provided in the dictionary. We see that there are 104 datapoints, exactly 24 miles away from the highway, so we assume this variable measures distance of a neighbourhood from the highway. This is exactly the same number of points for which nonRetailBusiness has a value of 0.18, so it further strengthens the argument that a lot of the datapoints seem to be for houses in the same or very close neighbourhood.
7. **pupilTeacherRatio** - We find another variable with a high modal value of exactly 23.2 pupils per teacher. Further the above argument that a large part of the sample is taken from a single neighbourhood.
8. **pctLowIncome** - 90% of the homes come from neighbourhoods with less than 30% households being low-income, however we do have values going up to 49% in the dataset.
9. **homeValue** - The distribution has 95% of houses valued at well below \$1 million, however, there are outliers above that value upto \$1.125 million. We take the log of the variable to make it closer to a normal distribution.
10. **pollutionIndex** - Has a scattered distribution with a median of 38.8, and a large outlier at 72.1.
11. **nBedRooms** - Close to normal distribution, with the mean and median around 4.25 bedrooms on average for a single family home, however there are small, as well as large outliers in the distribution.

```
q1.dataset = read.csv("houseValueData.csv")
```

```
str(q1.dataset)
```

```
## 'data.frame': 400 obs. of 11 variables:
## $ crimeRate_pc : num 37.6619 0.5783 0.0429 22.5971 0.0664 ...
## $ nonRetailBusiness: num 0.181 0.0397 0.1504 0.181 0.0405 ...
## $ withWater : int 0 0 0 0 0 0 0 0 0 0 ...
## $ ageHouse : num 78.7 67 77.3 89.5 74.4 71.3 68.2 97.3 92.2 96.2 ...
## $ distanceToCity : num 2.71 4.12 7.82 1.95 5.54 ...
## $ distanceToHighway: int 24 5 4 24 5 5 5 5 3 5 ...
## $ pupilTeacherRatio: num 23.2 16 21.2 23.2 19.6 23.9 22.2 17.7 20.8 17.7 ...
## $ pctLowIncome : int 18 9 13 41 8 9 12 18 5 4 ...
## $ homeValue : int 245250 1125000 463500 166500 672750 596250 425250 483750 852750 1125000 .
## $ pollutionIndex : num 52.9 42.5 31.4 55 36 37 34.9 72.1 33.8 45.5 ...
## $ nBedRooms : num 4.2 6.3 4.25 3 4.86 ...
```

```
summary(q1.dataset)
```

```
##      crimeRate_pc      nonRetailBusiness      withWater      ageHouse
## Min.   : 0.00632    Min.   :0.0074    Min.   :0.0000    Min.   : 2.90
## 1st Qu.: 0.08260    1st Qu.:0.0513    1st Qu.:0.0000    1st Qu.: 45.67
## Median : 0.26600    Median :0.0969    Median :0.0000    Median : 77.95
## Mean   : 3.76256    Mean   :0.1115    Mean   :0.0675    Mean   : 68.93
## 3rd Qu.: 3.67481    3rd Qu.:0.1810    3rd Qu.:0.0000    3rd Qu.: 94.15
## Max.   :88.97620    Max.   :0.2774    Max.   :1.0000    Max.   :100.00
## distanceToCity      distanceToHighway      pupilTeacherRatio      pctLowIncome
## Min.   : 1.228    Min.   : 1.000    Min.   :15.60    Min.   : 2.00
## 1st Qu.: 3.240    1st Qu.: 4.000    1st Qu.:19.90    1st Qu.: 8.00
## Median : 6.115    Median : 5.000    Median :21.90    Median :14.00
## Mean   : 9.638    Mean   : 9.582    Mean   :21.39    Mean   :15.79
## 3rd Qu.:13.628    3rd Qu.:24.000    3rd Qu.:23.20    3rd Qu.:21.00
## Max.   :54.197    Max.   :24.000    Max.   :25.00    Max.   :49.00
##      homeValue      pollutionIndex      nBedRooms
## Min.   : 112500    Min.   :23.50    Min.   :1.561
## 1st Qu.: 384188    1st Qu.:29.88    1st Qu.:3.883
## Median : 477000    Median :38.80    Median :4.193
## Mean   : 499584    Mean   :40.61    Mean   :4.266
## 3rd Qu.: 558000    3rd Qu.:47.58    3rd Qu.:4.582
## Max.   :1125000    Max.   :72.10    Max.   :6.780
```

```
# Performing univariate analysis Crime Rate
```

```
summary(q1.dataset$crimeRate_pc)
```

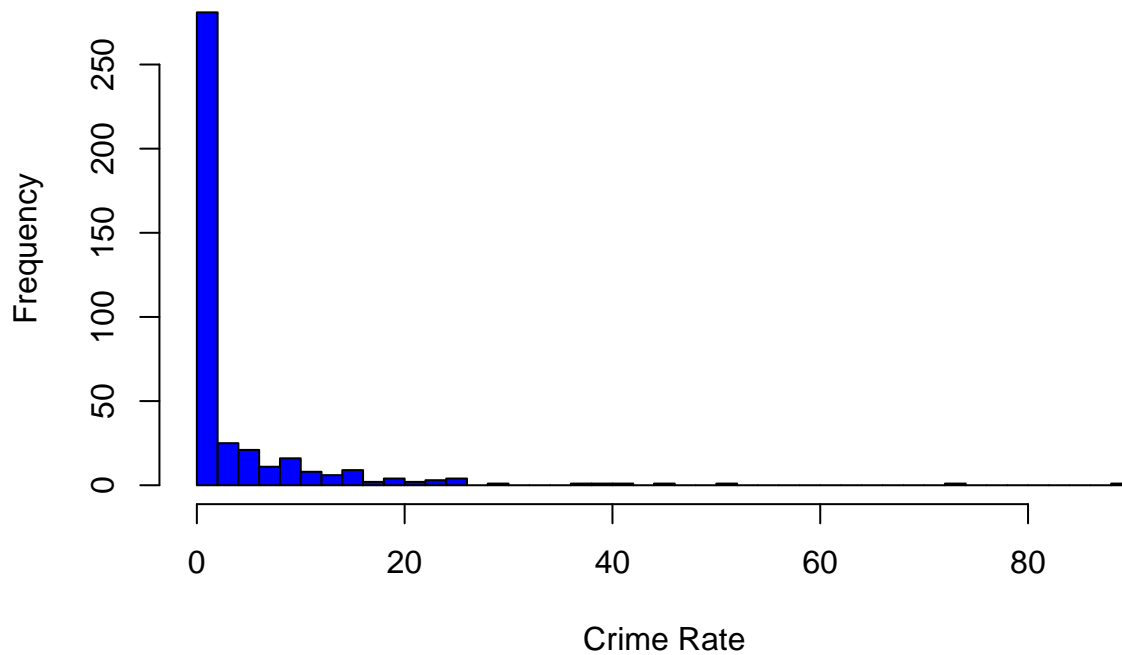
```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00632 0.08260 0.26600 3.76300 3.67500 88.98000
```

```
quantile(q1.dataset$crimeRate_pc, probs = c(0.01, 0.05, 0.1, 0.25, 0.5,
0.75, 0.9, 0.95, 0.99, 1))
```

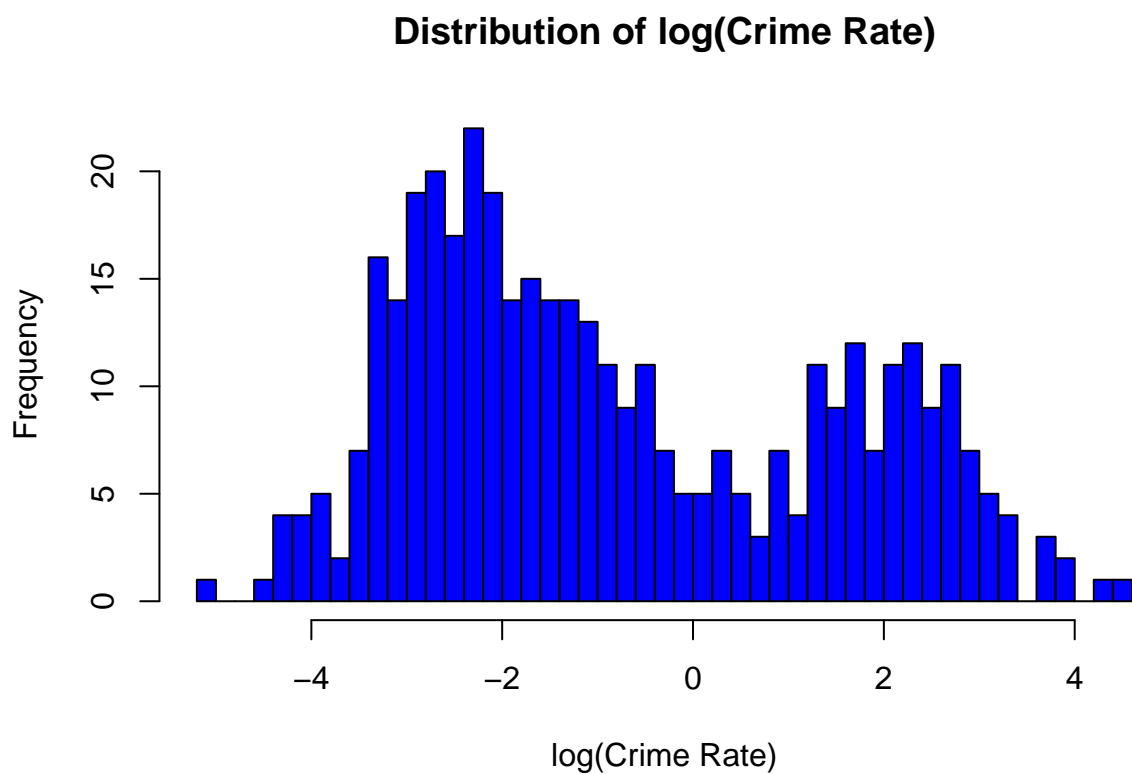
```
##      1%      5%      10%      25%      50%      75%
## 0.0143128 0.0310980 0.0410280 0.0825975 0.2660050 3.6748075
##      90%      95%      99%      100%
## 11.2021500 18.1052800 41.5713690 88.9762000
```

```
hist(q1.dataset$crimeRate_pc, breaks = 60, col = "blue", main = "Distribution of Crime Rate",
xlab = "Crime Rate")
```

Distribution of Crime Rate



```
# This variable is extremely +vely skewed, so we take the log
q1.dataset$logCrimeRate_pc = log(q1.dataset$crimeRate_pc)
hist(q1.dataset$logCrimeRate_pc, breaks = 60, col = "blue", main = "Distribution of log(Crime Rate)",
     xlab = "log(Crime Rate)")
```



```
# nonRetailBusiness
summary(q1.dataset$nonRetailBusiness)
```

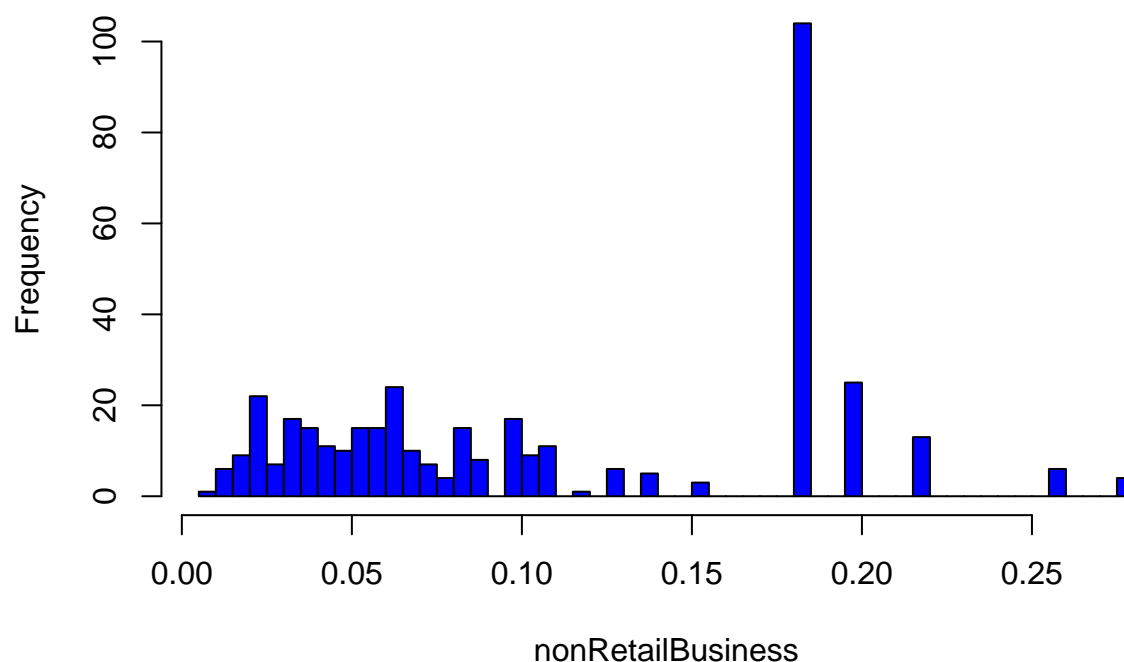
```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.0074  0.0513  0.0969  0.1115  0.1810  0.2774
```

```
quantile(q1.dataset$nonRetailBusiness, probs = c(0.01, 0.05, 0.1, 0.25,
  0.5, 0.75, 0.9, 0.95, 0.99, 1))
```

```
##      1%      5%     10%     25%     50%     75%     90%     95%
## 0.013794 0.021725 0.028900 0.051300 0.096900 0.181000 0.195800 0.218900
##      99%     100%
## 0.256709 0.277400
```

```
hist(q1.dataset$nonRetailBusiness, breaks = 60, col = "blue", main = "Distribution of nonRetailBusiness",
  xlab = "nonRetailBusiness")
```

Distribution of nonRetailBusiness



```
head(q1.dataset[order(q1.dataset$nonRetailBusiness, decreasing = TRUE),
  c("nonRetailBusiness")], n = 50)
```

```
## [1] 0.2774 0.2774 0.2774 0.2774 0.2565 0.2565 0.2565 0.2565 0.2565 0.2565
## [11] 0.2189 0.2189 0.2189 0.2189 0.2189 0.2189 0.2189 0.2189 0.2189 0.2189
## [21] 0.2189 0.2189 0.2189 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958
## [31] 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958
## [41] 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958 0.1958 0.1810 0.1810
```

```
tail(sort(table(q1.dataset$nonRetailBusiness)), 5)
```

```
##
## 0.2189 0.062 0.0814 0.1958 0.181
## 13 14 15 25 104
```

```
# suspicious that this has such a large modal value. May be some coded
# val
```

```
# withWater
summary(q1.dataset$withWater)
```

```
## Min. 1st Qu. Median Mean 3rd Qu. Max.
## 0.0000 0.0000 0.0000 0.0675 0.0000 1.0000
```



```
# 7% have water
```

```
# ageHouse
```

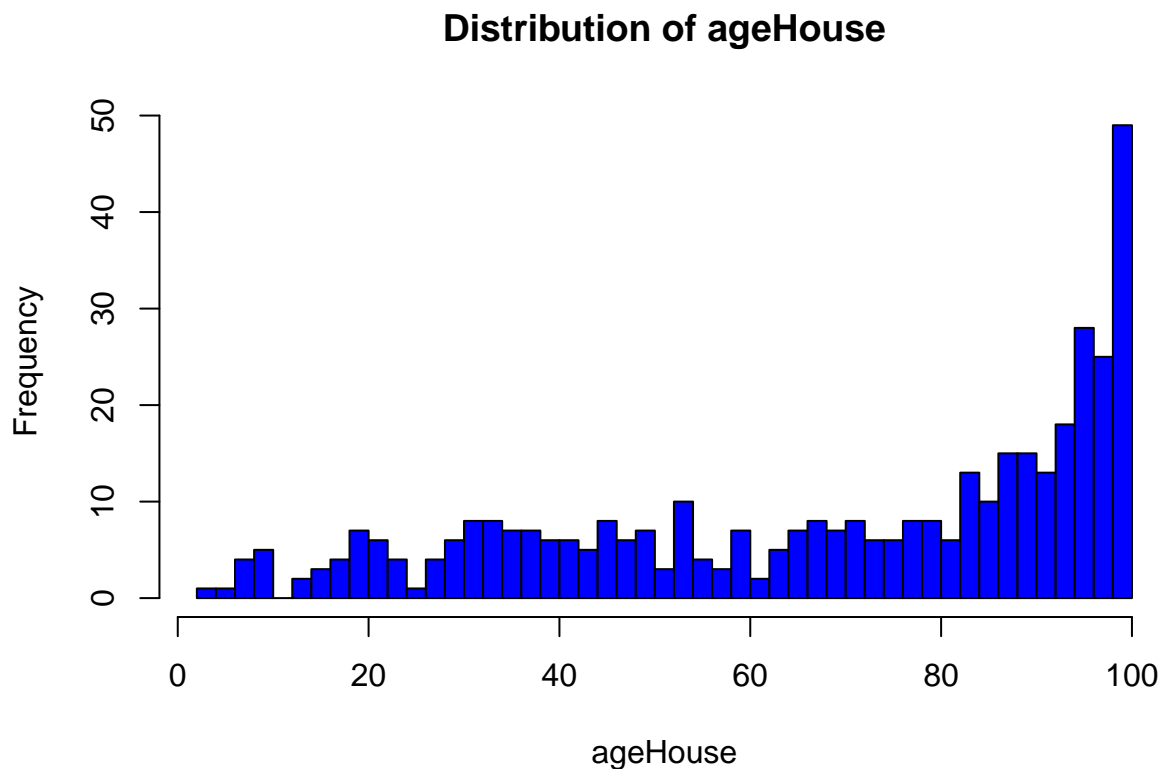
```
summary(q1.dataset$ageHouse)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      2.90  45.68   77.95   68.93   94.15  100.00
```

```
quantile(q1.dataset$ageHouse, probs = c(0.01, 0.05, 0.1, 0.25, 0.5, 0.75,
    0.9, 0.95, 0.99, 1))
```

```
##      1%      5%     10%     25%     50%     75%     90%     95%     99%
##      7.788 18.370 27.690 45.675 77.950 94.150 98.410 100.000 100.000
##      100%
## 100.000
```

```
hist(q1.dataset$ageHouse, breaks = 60, col = "blue", main = "Distribution of ageHouse",
    xlab = "ageHouse")
```



```
# Looks like a % value. May require a power transformation
```

```
# disttocity
```

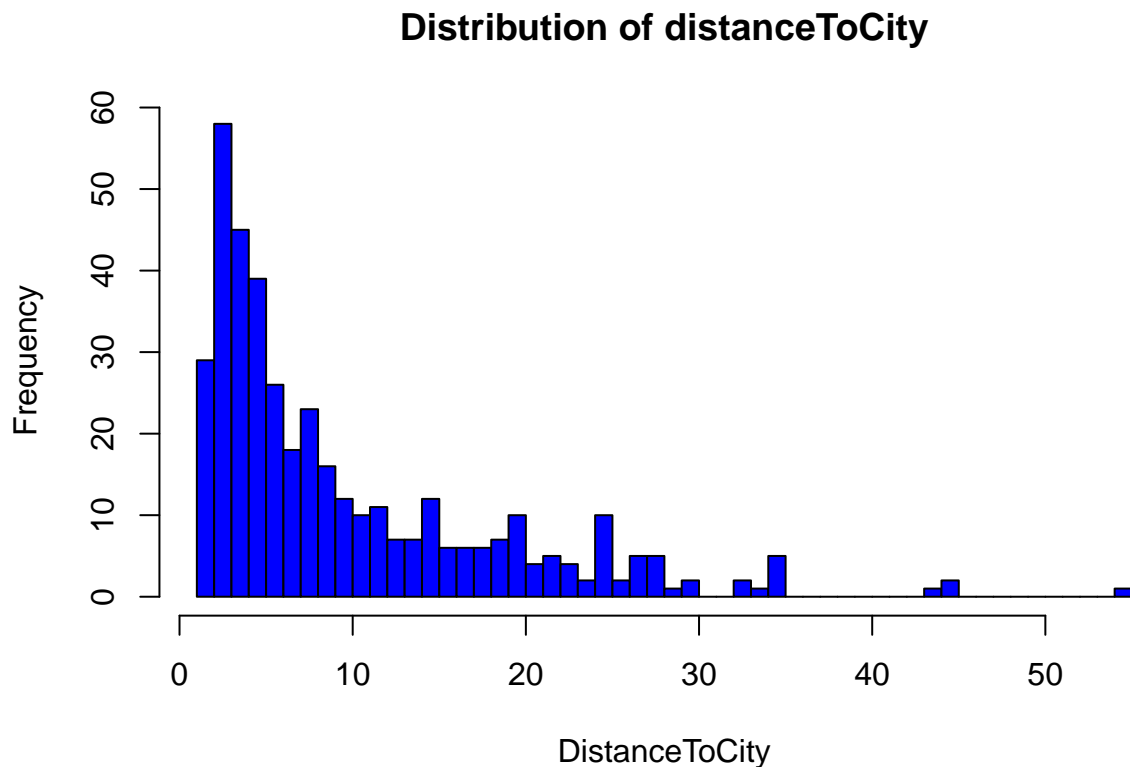
```
summary(q1.dataset$distanceToCity)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      1.228   3.240   6.115   9.638  13.630  54.200
```

```
quantile(q1.dataset$distanceToCity, probs = c(0.01, 0.05, 0.1, 0.25, 0.5,
0.75, 0.9, 0.95, 0.99, 1))
```

```
##          1%          5%          10%          25%          50%          75%          90%
##  1.342576  1.889692  2.158538  3.239878  6.114617 13.627873 22.682747
##          95%          99%         100%
## 26.939533 35.063729 54.197188
```

```
hist(q1.dataset$distanceToCity, breaks = 60, col = "blue", main = "Distribution of distanceToCity",
xlab = "DistanceToCity")
```



```
q1.dataset$logDistanceToCity = log(q1.dataset$distanceToCity)
# skewed with a large outlier at the end. Keep in mind while running
# model

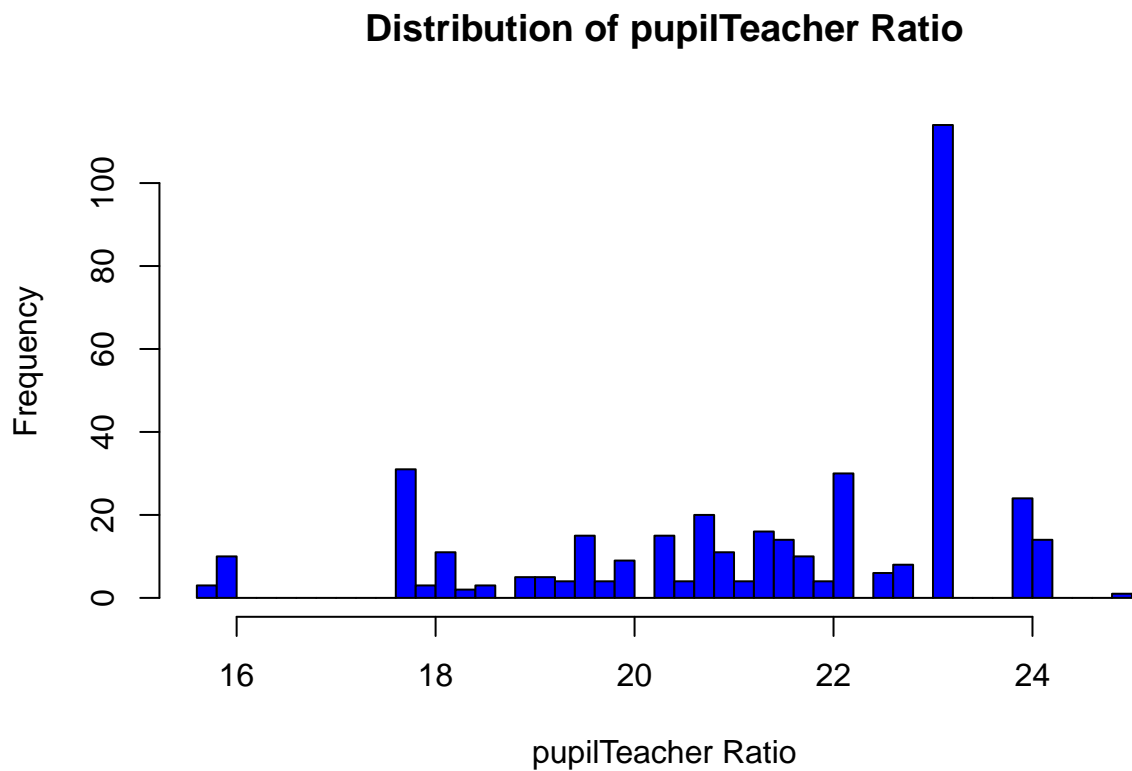
# pupilTeacher
summary(q1.dataset$pupilTeacherRatio)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      15.60  19.90  21.90  21.39  23.20  25.00
```

```
quantile(q1.dataset$pupilTeacherRatio, probs = c(0.01, 0.05, 0.1, 0.25,
  0.5, 0.75, 0.9, 0.95, 0.99, 1))
```

```
##    1%    5%   10%   25%   50%   75%   90%   95%   99%  100%
## 16.0 17.7 17.7 19.9 21.9 23.2 23.2 24.0 24.2 25.0
```

```
hist(q1.dataset$pupilTeacherRatio, breaks = 60, col = "blue", main = "Distribution of pupilTeacher Ratio",
  xlab = "pupilTeacher Ratio")
```



```
tail(sort(table(q1.dataset$pupilTeacherRatio)), 5)
```

```
##
##    24 22.2 20.8 17.7 23.2
##    16   17   20   28  110
```

High mode at 23.2, suspicious

dist to highway

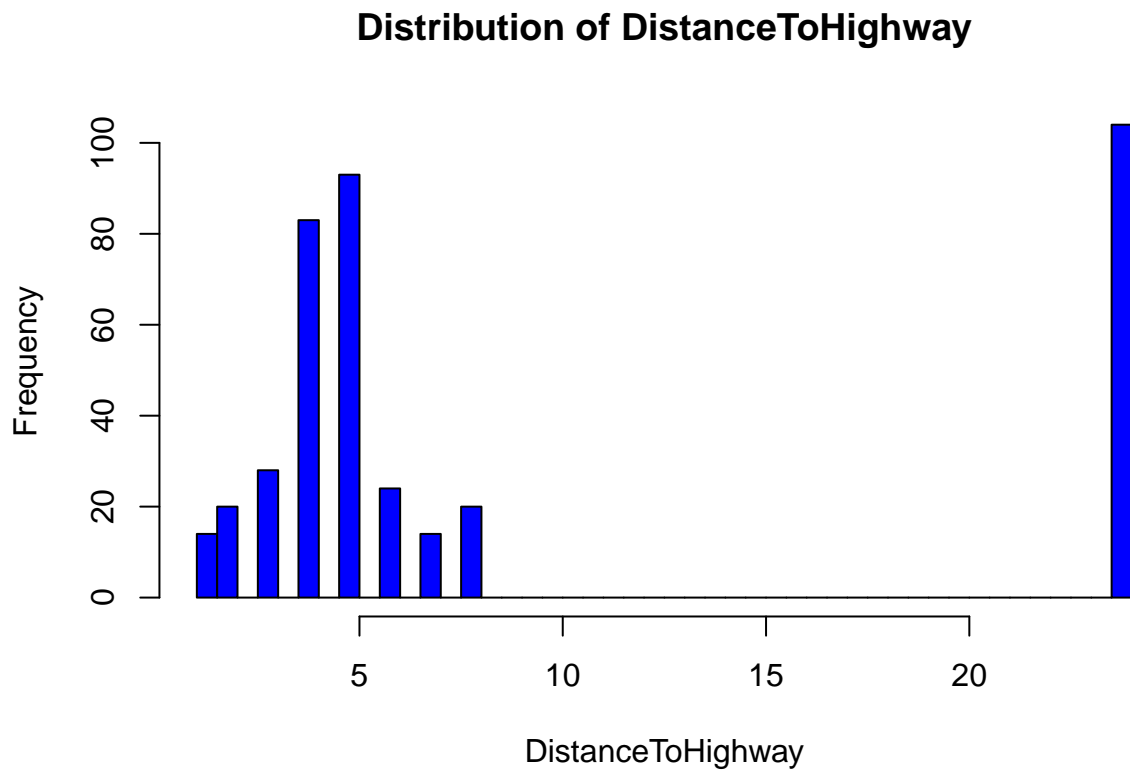
```
summary(q1.dataset$distanceToHighway)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.000   4.000   5.000   9.582  24.000  24.000
```

```
quantile(q1.dataset$distanceToHighway, probs = c(0.01, 0.05, 0.1, 0.25,
  0.5, 0.75, 0.9, 0.95, 0.99, 1))
```

```
##    1%    5%   10%   25%   50%   75%   90%   95%   99%  100%
##     1     2     3     4     5    24    24    24    24    24
```

```
hist(q1.dataset$distanceToHighway, breaks = 60, col = "blue", main = "Distribution of DistanceToHighway",
  xlab = "DistanceToHighway")
```



```
tail(sort(table(q1.dataset$distanceToHighway)), 5)
```

```
##
##    6    3    4    5   24
##   24   28   83   93  104
```

*# Very strange that so many values are exactly 24. May not be best
thing for regression.*

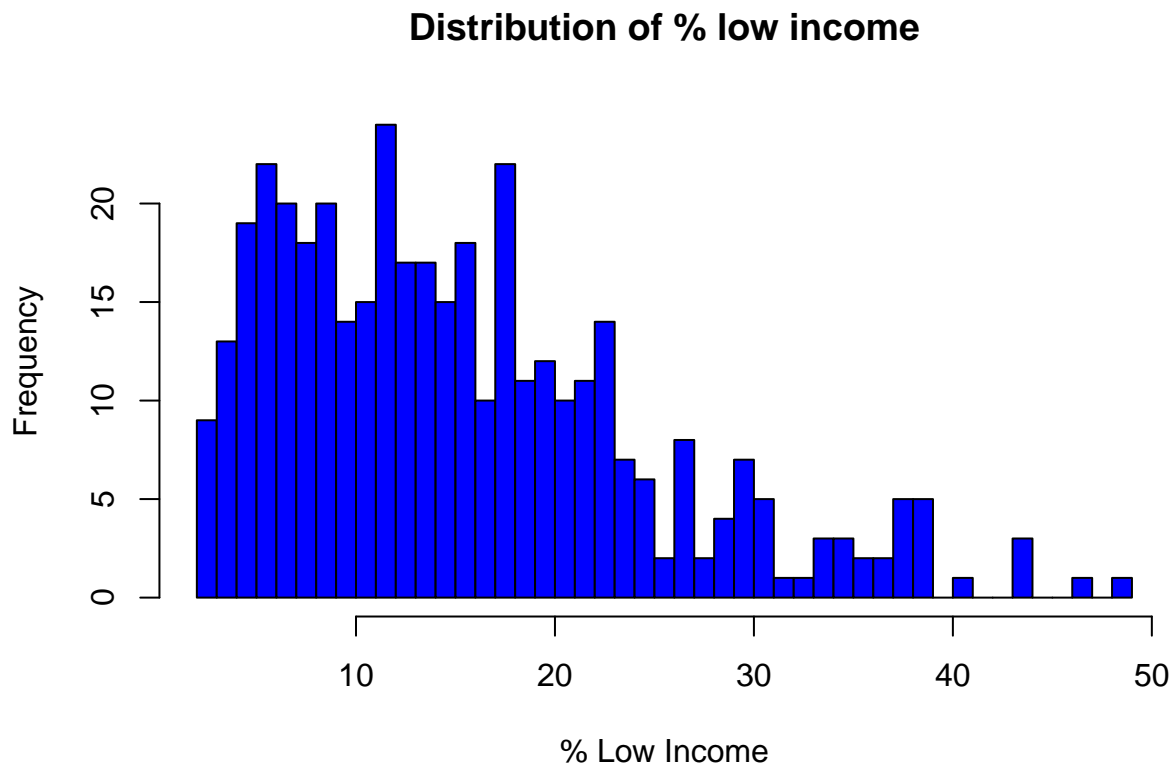
```
# pctlowincome
summary(q1.dataset$pctLowIncome)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     2.0     8.0    14.0    15.8    21.0    49.0
```

```
quantile(q1.dataset$pctLowIncome, probs = c(0.01, 0.05, 0.1, 0.25, 0.5,
      0.75, 0.9, 0.95, 0.99, 1))
```

```
##      1%      5%     10%     25%     50%     75%     90%     95%     99%    100%
##    3.00    4.00    5.00    8.00   14.00   21.00   29.10   35.05   44.00   49.00
```

```
hist(q1.dataset$pctLowIncome, breaks = 60, col = "blue", main = "Distribution of % low income",
     xlab = "% Low Income")
```



```
tail(sort(table(q1.dataset$pctLowIncome)), 5)
```

```
##
##  7  9  6 18 12
## 20 20 22 22 24
```

```
# slight neg skew
```

```
# Home value
```

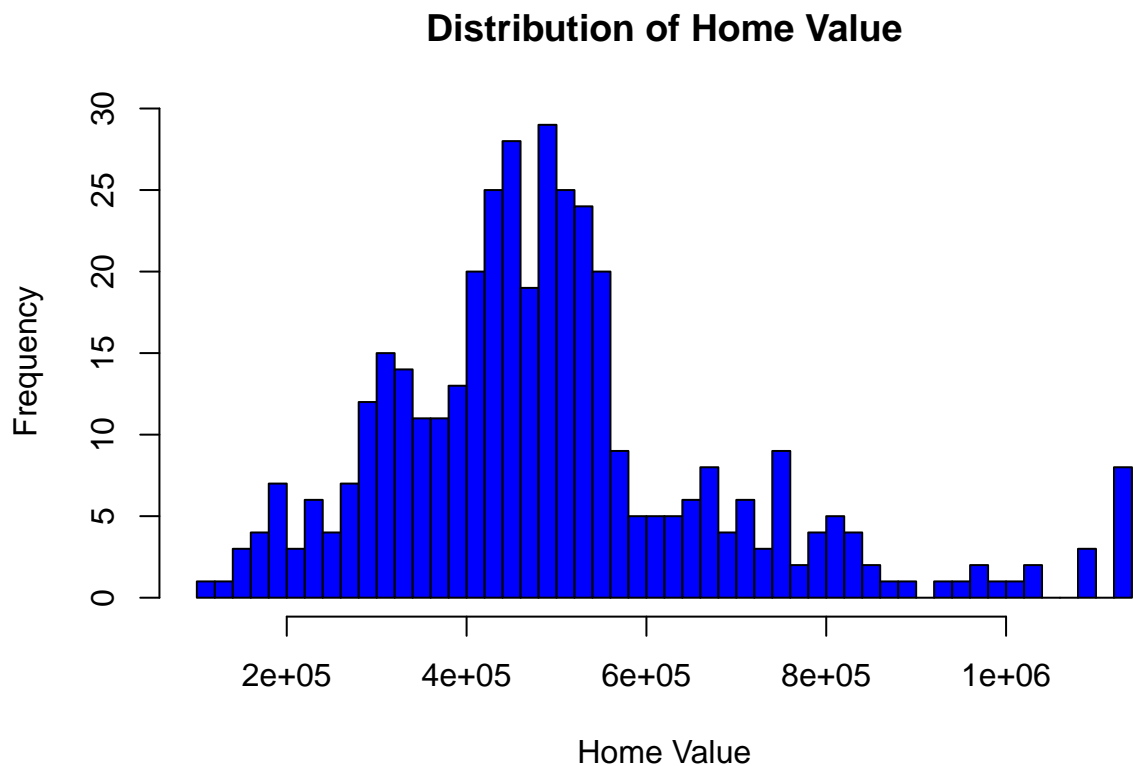
```
summary(q1.dataset$homeValue)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   112500   384200   477000   499600   558000  1125000
```

```
quantile(q1.dataset$homeValue, probs = c(0.01, 0.05, 0.1, 0.25, 0.5, 0.75,
0.9, 0.95, 0.99, 1))
```

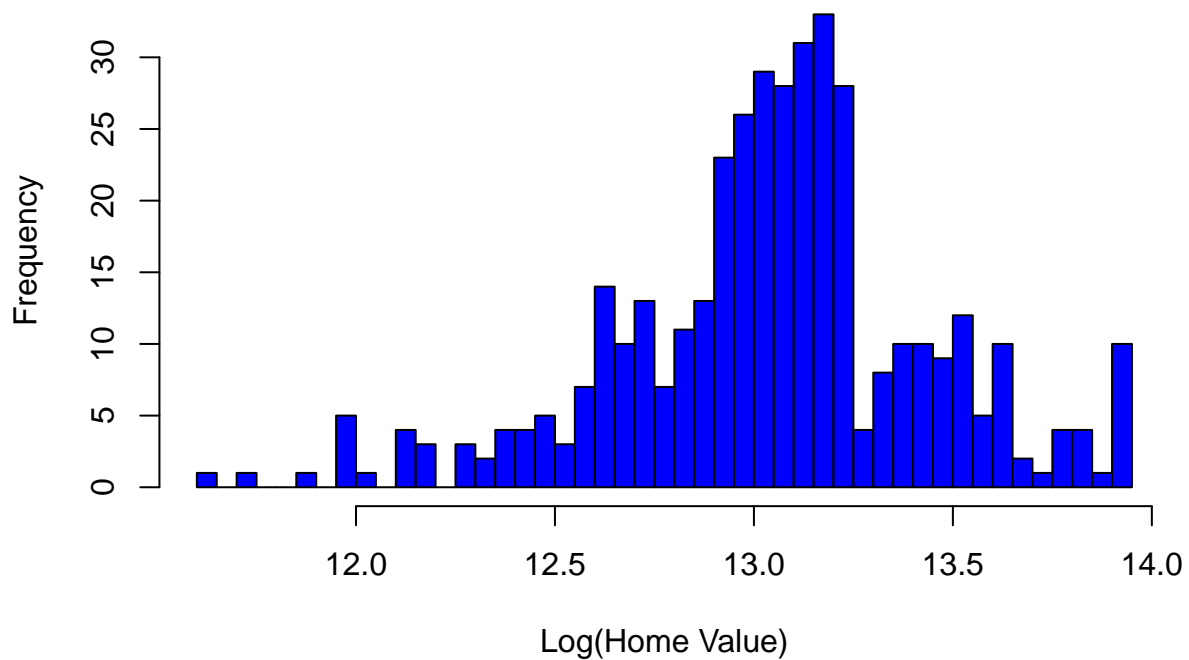
```
##          1%          5%          10%          25%          50%          75%          90%
## 157500.0 229500.0 291825.0 384187.5 477000.0 558000.0 749475.0
##          95%          99%         100%
## 871987.5 1125000.0 1125000.0
```

```
hist(q1.dataset$homeValue, breaks = 60, col = "blue", main = "Distribution of Home Value",
xlab = "Home Value")
```



```
q1.dataset$logHomeValue = log(q1.dataset$homeValue)
hist(q1.dataset$logHomeValue, breaks = 60, col = "blue", main = "Distribution of Log(Home Value)",
xlab = "Log(Home Value)")
```

Distribution of Log(Home Value)



```
# Pretty normal
```

```
# poll Index
```

```
summary(q1.dataset$pollutionIndex)
```

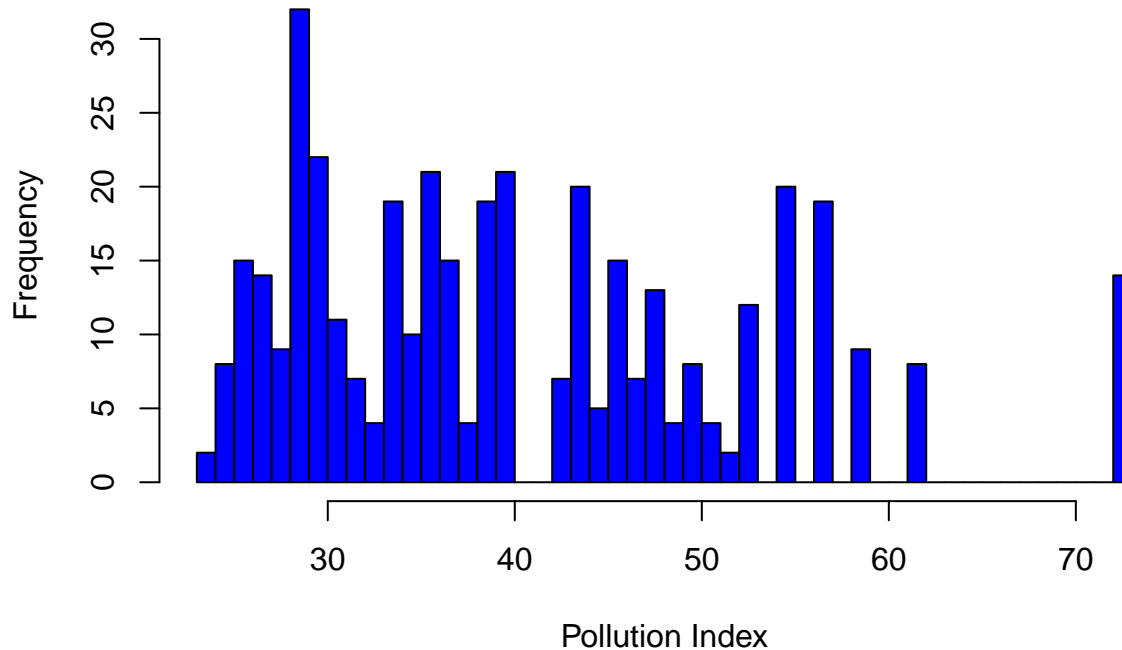
```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  23.50  29.87   38.80   40.61  47.58   72.10
```

```
quantile(q1.dataset$pollutionIndex, probs = c(0.01, 0.05, 0.1, 0.25, 0.5,
  0.75, 0.9, 0.95, 0.99, 1))
```

```
##      1%      5%     10%     25%     50%     75%     90%     95%     99%    100%
## 24.398 25.880 27.600 29.875 38.800 47.575 56.300 62.000 72.100 72.100
```

```
hist(q1.dataset$pollutionIndex, breaks = 60, col = "blue", main = "Distribution of Pollution Index",
  xlab = "Pollution Index")
```

Distribution of Pollution Index



```
# scattered dist, one high outlier at 72
```

```
# nbedrooms
```

```
summary(q1.dataset$nBedRooms)
```

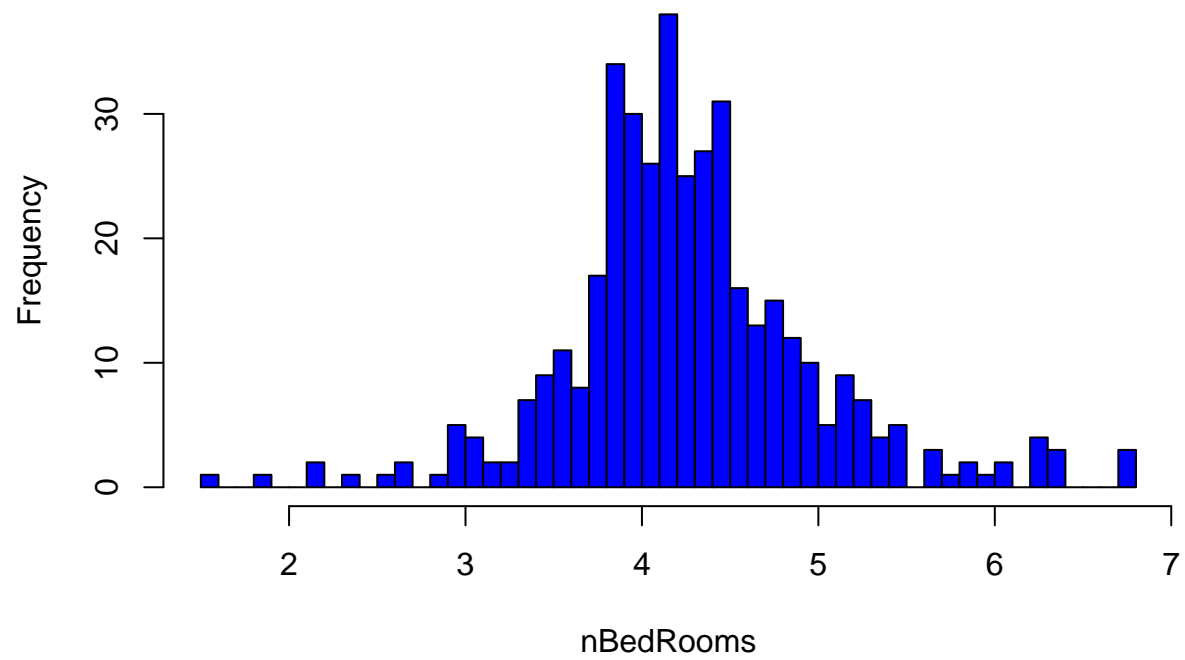
```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  1.561   3.883   4.193   4.266   4.582   6.780
```

```
quantile(q1.dataset$nBedRooms, probs = c(0.01, 0.05, 0.1, 0.25, 0.5, 0.75,
0.9, 0.95, 0.99, 1))
```

```
##      1%      5%     10%     25%     50%     75%     90%     95%     99%
## 2.36570 3.26770 3.53550 3.88300 4.19300 4.58175 5.14710 5.45480 6.37523
##    100%
## 6.78000
```

```
hist(q1.dataset$nBedRooms, breaks = 60, col = "blue", main = "Distribution of nBedRooms",
xlab = "nBedRooms")
```


Distribution of nBedRooms



Pretty normal

Step 2 - Bivariate Analysis

We examine bivariate correlations and scatterplots for all (transformed) variables in the dataset.

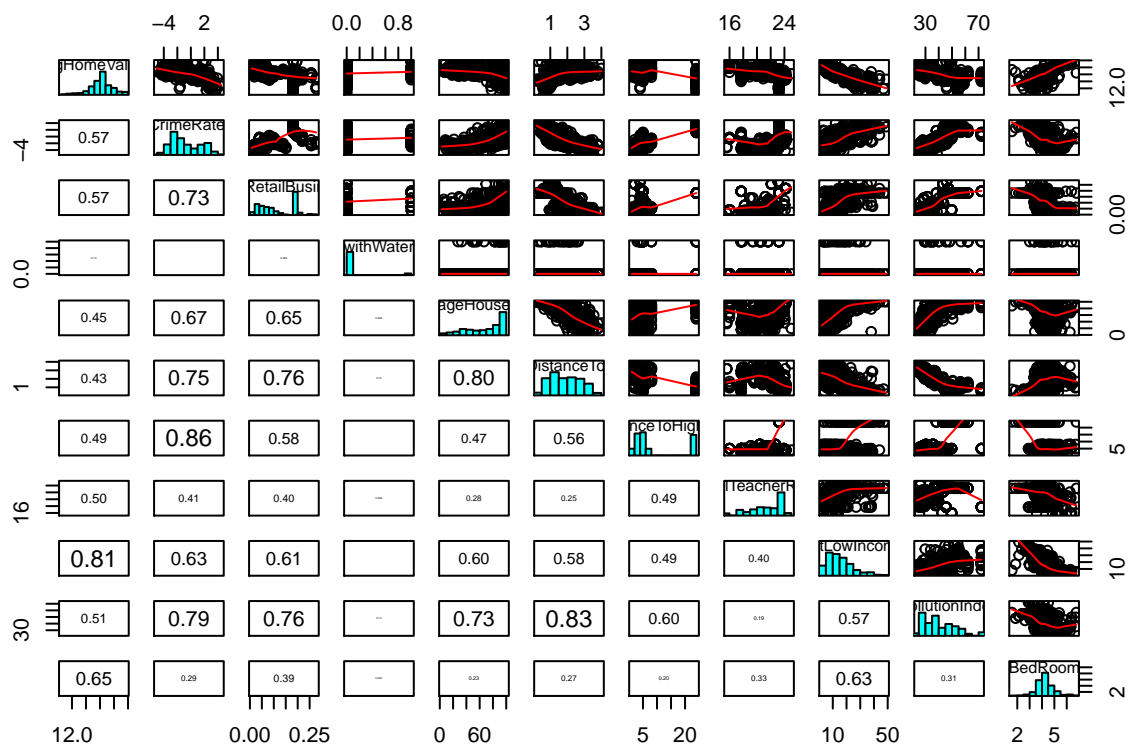
Conclusions

1. A lot of variables show strong correlations with each other in the dataset (absolute val of correlation > 0.7). We must be wary of multicollinearity when including these variables together in our regression models which would make our model coefficients lose precision. At the same time, it is important to include necessary variables in order to prevent any omitted variable bias.
- logCrimeRate shows a strong correlation with logdistanceToCity, distanceToHighway and pollutionIndex
 - nonRetailBusiness shows a strong correlation with logdistanceToCity and pollutionIndex
 - ageHouse also shows a strong correlation with logdistanceToCity and pollutionIndex
 - logDistancetoCity shows a strong correlation with pollutionIndex

```
panel.hist <- function(x, ...) {
  usr <- par("usr")
  on.exit(par(usr))
  par(usr = c(usr[1:2], 0, 1.5))
  h <- hist(x, plot = FALSE)
  breaks <- h$breaks
  nB <- length(breaks)
  y <- h$counts
  y <- y/max(y)
  rect(breaks[-nB], 0, breaks[-1], y, col = "cyan", ...)
}

panel.cor <- function(x, y, digits = 2, prefix = "", cex.cor, ...) {
  usr <- par("usr")
  on.exit(par(usr))
  par(usr = c(0, 1, 0, 1))
  r <- abs(cor(x, y))
  txt <- format(c(r, 0.123456789), digits = digits)[1]
  txt <- paste0(prefix, txt)
  if (missing(cex.cor))
    cex.cor <- 0.8/strwidth(txt)
  text(0.5, 0.5, txt, cex = cex.cor * r)
}

pairs(logHomeValue ~ logCrimeRate_pc + nonRetailBusiness + withWater +
  ageHouse + logDistanceToCity + distanceToHighway + pupilTeacherRatio +
  pctLowIncome + pollutionIndex + nBedRooms, data = q1.dataset, upper.panel = panel.smooth,
  lower.panel = panel.cor, diag.panel = panel.hist)
```



Step 3 - Model Estimation

We start off with a naive approach, including all variables in the regression to observe results.

```
model.1 = lm(logHomeValue ~ logCrimeRate_pc + nonRetailBusiness + withWater +
  ageHouse + logDistanceToCity + distanceToHighway + pupilTeacherRatio +
  pctLowIncome + pollutionIndex + nBedRooms, data = q1.dataset)
summary(model.1)
```

```
##
## Call:
## lm(formula = logHomeValue ~ logCrimeRate_pc + nonRetailBusiness +
##     withWater + ageHouse + logDistanceToCity + distanceToHighway +
##     pupilTeacherRatio + pctLowIncome + pollutionIndex + nBedRooms,
##     data = q1.dataset)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73040 -0.09641 -0.00502  0.09332  0.78653
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    14.2334735   0.2107218   67.546 < 2e-16 ***
## logCrimeRate_pc  -0.0107344   0.0129755   -0.827  0.408585
## nonRetailBusiness -0.4128304   0.2645968   -1.560  0.119520
## withWater        0.1411053   0.0409450    3.446  0.000631 ***
## ageHouse         0.0001650   0.0006556    0.252  0.801461
## logDistanceToCity -0.1288004   0.0254045   -5.070  6.17e-07 ***
## distanceToHighway -0.0010709   0.0025088   -0.427  0.669728
## pupilTeacherRatio -0.0303139   0.0060208   -5.035  7.33e-07 ***
## pctLowIncome     -0.0238385   0.0018412  -12.947 < 2e-16 ***
## pollutionIndex   -0.0081282   0.0018766   -4.331  1.89e-05 ***
## nBedRooms        0.1028429   0.0192037    5.355  1.46e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1999 on 389 degrees of freedom
## Multiple R-squared:  0.7529, Adjusted R-squared:  0.7465
## F-statistic: 118.5 on 10 and 389 DF,  p-value: < 2.2e-16
```

```
AIC(model.1)
```

```
## [1] -140.1337
```

```
BIC(model.1)
```

```
## [1] -92.23611
```

In this model, we see that several variables do not have statistical significance. We see that the coefficients lack precision, having extremely high standard errors.

Before we move on to more parsimonious models, we will examine interaction variables to see if they add any explanatory power to our model. We have two categorical variables in our dataset: withWater and

distanceToHighway (though a numerical variable, it has only nine distinct values, effectively functioning as a categorical). We add all possible interactions with these variables to see if we obtain any noteworthy result.

```
model.2 = lm(logHomeValue ~ logCrimeRate_pc + nonRetailBusiness + withWater +
  ageHouse + logDistanceToCity + distanceToHighway + pupilTeacherRatio +
  pctLowIncome + pollutionIndex + nBedRooms + distanceToHighway:logCrimeRate_pc +
  distanceToHighway:nonRetailBusiness + distanceToHighway:ageHouse +
  distanceToHighway:pupilTeacherRatio + distanceToHighway:pctLowIncome +
  distanceToHighway:pollutionIndex + distanceToHighway:nBedRooms + withWater:pollutionIndex +
  withWater:logCrimeRate_pc + withWater:nonRetailBusiness + withWater:ageHouse +
  withWater:logDistanceToCity + withWater:pupilTeacherRatio + withWater:pctLowIncome +
  withWater:nBedRooms, data = q1.dataset)
summary(model.2)
```

```
##
## Call:
## lm(formula = logHomeValue ~ logCrimeRate_pc + nonRetailBusiness +
##   withWater + ageHouse + logDistanceToCity + distanceToHighway +
##   pupilTeacherRatio + pctLowIncome + pollutionIndex + nBedRooms +
##   distanceToHighway:logCrimeRate_pc + distanceToHighway:nonRetailBusiness +
##   distanceToHighway:ageHouse + distanceToHighway:pupilTeacherRatio +
##   distanceToHighway:pctLowIncome + distanceToHighway:pollutionIndex +
##   distanceToHighway:nBedRooms + withWater:pollutionIndex +
##   withWater:logCrimeRate_pc + withWater:nonRetailBusiness +
##   withWater:ageHouse + withWater:logDistanceToCity + withWater:pupilTeacherRatio +
##   withWater:pctLowIncome + withWater:nBedRooms, data = q1.dataset)
##
## Residuals:
```

	Min	1Q	Median	3Q	Max
	-0.66948	-0.07603	-0.00873	0.06591	0.68353

```
##
## Coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.298e+01	4.468e-01	29.055	< 2e-16
logCrimeRate_pc	4.846e-02	1.559e-02	3.108	0.002027
nonRetailBusiness	-1.489e+00	4.294e-01	-3.468	0.000585
withWater	1.801e-01	8.372e-01	0.215	0.829771
ageHouse	-3.679e-03	7.789e-04	-4.723	3.30e-06
logDistanceToCity	-1.238e-01	2.274e-02	-5.445	9.40e-08
distanceToHighway	1.881e-02	7.910e-02	0.238	0.812165
pupilTeacherRatio	-3.658e-02	1.793e-02	-2.040	0.042057
pctLowIncome	-1.443e-03	2.692e-03	-0.536	0.592234
pollutionIndex	-3.252e-04	2.705e-03	-0.120	0.904352
nBedRooms	3.753e-01	2.812e-02	13.346	< 2e-16
logCrimeRate_pc:distanceToHighway	-1.020e-02	1.447e-03	-7.052	8.58e-12
nonRetailBusiness:distanceToHighway	2.281e-01	8.831e-02	2.583	0.010177
ageHouse:distanceToHighway	2.572e-04	8.876e-05	2.898	0.003976
distanceToHighway:pupilTeacherRatio	2.983e-03	3.840e-03	0.777	0.437728
distanceToHighway:pctLowIncome	-1.189e-03	1.600e-04	-7.429	7.50e-13
distanceToHighway:pollutionIndex	-7.424e-04	1.844e-04	-4.026	6.88e-05
distanceToHighway:nBedRooms	-1.827e-02	1.668e-03	-10.949	< 2e-16
withWater:pollutionIndex	-1.191e-02	5.861e-03	-2.032	0.042887
logCrimeRate_pc:withWater	6.485e-02	6.100e-02	1.063	0.288479
nonRetailBusiness:withWater	1.332e+00	1.408e+00	0.946	0.344643

```
## withWater:ageHouse          1.852e-03  2.952e-03   0.627 0.530828
## withWater:logDistanceToCity  8.978e-02  1.229e-01   0.730 0.465649
## withWater:pupilTeacherRatio  2.894e-02  2.386e-02   1.213 0.225988
## withWater:pctLowIncome      -6.662e-03  6.682e-03  -0.997 0.319416
## withWater:nBedRooms         -1.047e-01  5.846e-02  -1.790 0.074245
##
## (Intercept)                ***
## logCrimeRate_pc             **
## nonRetailBusiness           ***
## withWater
## ageHouse                    ***
## logDistanceToCity           ***
## distanceToHighway
## pupilTeacherRatio           *
## pctLowIncome
## pollutionIndex
## nBedRooms                   ***
## logCrimeRate_pc:distanceToHighway ***
## nonRetailBusiness:distanceToHighway *
## ageHouse:distanceToHighway   **
## distanceToHighway:pupilTeacherRatio
## distanceToHighway:pctLowIncome ***
## distanceToHighway:pollutionIndex ***
## distanceToHighway:nBedRooms  ***
## withWater:pollutionIndex    *
## logCrimeRate_pc:withWater
## nonRetailBusiness:withWater
## withWater:ageHouse
## withWater:logDistanceToCity
## withWater:pupilTeacherRatio
## withWater:pctLowIncome
## withWater:nBedRooms         .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1584 on 374 degrees of freedom
## Multiple R-squared:  0.8508, Adjusted R-squared:  0.8408
## F-statistic: 85.29 on 25 and 374 DF, p-value: < 2.2e-16
```

```
AIC(model.2)
```

```
## [1] -311.9024
```

```
BIC(model.2)
```

```
## [1] -204.1328
```

```
waldtest(model.1, model.2)
```

```
## Wald test
##
## Model 1: logHomeValue ~ logCrimeRate_pc + nonRetailBusiness + withWater +
```

```
##      ageHouse + logDistanceToCity + distanceToHighway + pupilTeacherRatio +
##      pctLowIncome + pollutionIndex + nBedRooms
## Model 2: logHomeValue ~ logCrimeRate_pc + nonRetailBusiness + withWater +
##      ageHouse + logDistanceToCity + distanceToHighway + pupilTeacherRatio +
##      pctLowIncome + pollutionIndex + nBedRooms + distanceToHighway:logCrimeRate_pc +
##      distanceToHighway:nonRetailBusiness + distanceToHighway:ageHouse +
##      distanceToHighway:pupilTeacherRatio + distanceToHighway:pctLowIncome +
##      distanceToHighway:pollutionIndex + distanceToHighway:nBedRooms +
##      withWater:pollutionIndex + withWater:logCrimeRate_pc + withWater:nonRetailBusiness +
##      withWater:ageHouse + withWater:logDistanceToCity + withWater:pupilTeacherRatio +
##      withWater:pctLowIncome + withWater:nBedRooms
## Res.Df Df      F      Pr(>F)
## 1      389
## 2      374 15 16.357 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We do see some additional explanatory power through the addition of the interaction terms as we obtain a model with higher R square, lower AIC and lower BIC, as well as a significant Wald Test p value. However, apart from having a model which is extremely difficult to interpret, we also notice that most of the coefficient estimates are extremely small, having very little practical significance.

Now, we want to narrow down our model to include only variables that really add to the explanatory power of the model, that reduce multicollinearity, that provide some valuable practical significance, while meeting the think-tank's specific ask of desirable neighbourhood features and environmental features' relation to home values.

We remove the following variables:

1. nonRetailBusiness: It has a high correlation with several of the variables in the dataset, and is not of direct interest to answering the question asked.
2. ageHouse: It is not of direct consequence to the question asked.
3. DistanceToCity has a high correlation with pollutionIndex, a variable we are definitely interested in, so we remove it to reduce the loss of precision that comes with multicollinearity
4. nBedRooms: It is not of direct consequence to the question asked.
5. distanceToHighway interactions except the interaction with log crime: This is the only interaction with a variable still in the model which has statistical and practical significance.
6. withWater interactions except the interaction with pollutionIndex: This interaction seems to have some practical as well as statistical significance.

```
model.3 = lm(logHomeValue ~ logCrimeRate_pc + withWater + distanceToHighway +
  pctLowIncome + pollutionIndex + distanceToHighway:logCrimeRate_pc +
  withWater:pollutionIndex, data = q1.dataset)
summary(model.3)
```

```
##
## Call:
## lm(formula = logHomeValue ~ logCrimeRate_pc + withWater + distanceToHighway +
##      pctLowIncome + pollutionIndex + distanceToHighway:logCrimeRate_pc +
##      withWater:pollutionIndex, data = q1.dataset)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -0.66018 -0.14040 -0.02863 0.10239 0.86916
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)    13.643386   0.086436 157.843 < 2e-16
## logCrimeRate_pc 0.044515   0.015860   2.807 0.00525
## withWater      0.476923   0.144643   3.297 0.00107
## distanceToHighway 0.004164   0.003324   1.253 0.21110
## pctLowIncome  -0.029823   0.001600 -18.640 < 2e-16
## pollutionIndex -0.002229   0.001718  -1.297 0.19540
## logCrimeRate_pc:distanceToHighway -0.005868   0.001226  -4.786 2.41e-06
## withWater:pollutionIndex -0.007204   0.003038  -2.371 0.01820
##
## (Intercept)          ***
## logCrimeRate_pc      **
## withWater            **
## distanceToHighway
## pctLowIncome          ***
## pollutionIndex
## logCrimeRate_pc:distanceToHighway ***
## withWater:pollutionIndex *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2189 on 392 degrees of freedom
## Multiple R-squared:  0.7013, Adjusted R-squared:  0.696
## F-statistic: 131.5 on 7 and 392 DF, p-value: < 2.2e-16
```

```
AIC(model.3)
```

```
## [1] -70.30455
```

```
BIC(model.3)
```

```
## [1] -34.38137
```

We see that the distanceToHighway and pollutionIndex variables don't seem to have statistical significance. We will remove the distanceToHighway variable since it does not have direct consequence to the question asked. Further, we remove the logCrimeRate variable, since it has a high correlation with pollutionIndex, a variable of importance to us. We do this to observe if its removal increases the precision of the pollutionIndex variable. Note that this also means that we remove the interaction of distanceToHighway and logCrimeRate

```
model.4 = lm(logHomeValue ~ withWater + pctLowIncome + pollutionIndex +
  withWater:pollutionIndex, data = q1.dataset)
summary(model.4)
```

```
##
## Call:
## lm(formula = logHomeValue ~ withWater + pctLowIncome + pollutionIndex +
##     withWater:pollutionIndex, data = q1.dataset)
##
## Residuals:
```



```
##      Min      1Q   Median      3Q      Max
## -0.67726 -0.14224 -0.02364  0.10665  0.85551
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      13.639280   0.043563 313.090 < 2e-16 ***
## withWater         0.474642   0.148998   3.186 0.00156 **
## pctLowIncome     -0.032366   0.001525 -21.218 < 2e-16 ***
## pollutionIndex   -0.002305   0.001286  -1.792 0.07382 .
## withWater:pollutionIndex -0.006498   0.003124  -2.080 0.03820 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2273 on 395 degrees of freedom
## Multiple R-squared:  0.6754, Adjusted R-squared:  0.6721
## F-statistic: 205.4 on 4 and 395 DF, p-value: < 2.2e-16
```

```
AIC(model.4)
```

```
## [1] -43.00319
```

```
BIC(model.4)
```

```
## [1] -19.0544
```

We still obtain no significance for our `pollutionIndex` variable, and are unable to make any confident claims about this variable's impact on `homeValue`. Perhaps this variable, which was significant in earlier models with more variables, is losing precision due to an omitted variable bias.

We try adding back the `pupilTeacherRatio` variable to see if that makes any difference to the model.

```
model.5 = lm(logHomeValue ~ withWater + pctLowIncome + pupilTeacherRatio +
  pollutionIndex + withWater:pollutionIndex, data = q1.dataset)
summary(model.5)
```

```
##
## Call:
## lm(formula = logHomeValue ~ withWater + pctLowIncome + pupilTeacherRatio +
##     pollutionIndex + withWater:pollutionIndex, data = q1.dataset)
##
## Residuals:
##      Min      1Q   Median      3Q      Max
## -0.66742 -0.12448 -0.01229  0.10960  0.87460
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)      14.391272   0.117353 122.632 < 2e-16 ***
## withWater         0.493635   0.141065   3.499 0.00052 ***
## pctLowIncome     -0.028714   0.001539 -18.654 < 2e-16 ***
## pupilTeacherRatio -0.037394   0.005463  -6.844 2.95e-11 ***
## pollutionIndex   -0.002505   0.001217  -2.057 0.04033 *
## withWater:pollutionIndex -0.007437   0.002961  -2.512 0.01240 *
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2152 on 394 degrees of freedom
## Multiple R-squared:  0.7099, Adjusted R-squared:  0.7062
## F-statistic: 192.8 on 5 and 394 DF,  p-value: < 2.2e-16
```

```
AIC(model.5)
```

```
## [1] -85.94111
```

```
BIC(model.5)
```

```
## [1] -58.00086
```

We obtain statistical significance for all our coefficients, while maintaining a high R square value. While AIC and BIC values are not the lowest, we prefer the parsimony of this model, and choose this as our final model to present to the think-tank.

We do not consider introducing instrument variables to the model for the following reasons:

1. None of our predictors seems to have any correlation with the error term, making them all exogenous (shown above).
2. While some variables do meet the criteria for instrument relevance, they are correlated with multiple variables in the model, so they do not make good overall candidates for instruments as they would introduce multicollinearity to the prediction of the variable for which they would serve as instruments.

```
cor(q1.dataset$pctLowIncome, model.5$residuals)
```

```
## [1] -9.991986e-18
```

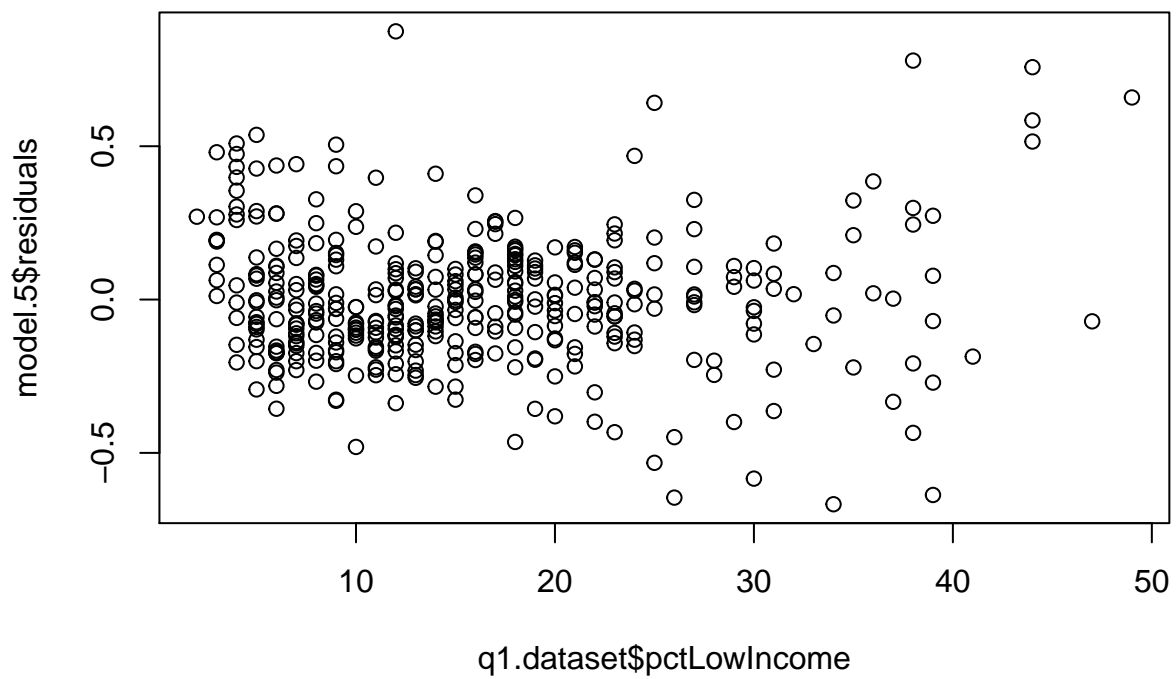
```
cor(q1.dataset$pollutionIndex, model.5$residuals)
```

```
## [1] 5.264698e-16
```

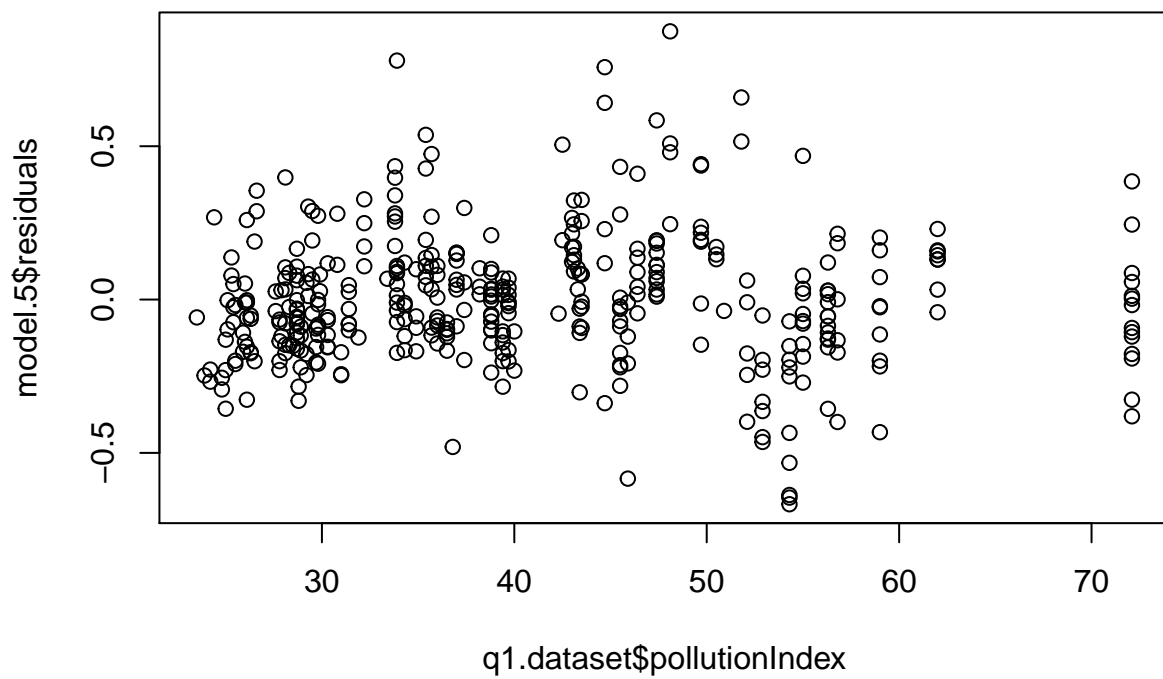
```
cor(q1.dataset$pupilTeacherRatio, model.5$residuals)
```

```
## [1] 2.849975e-15
```

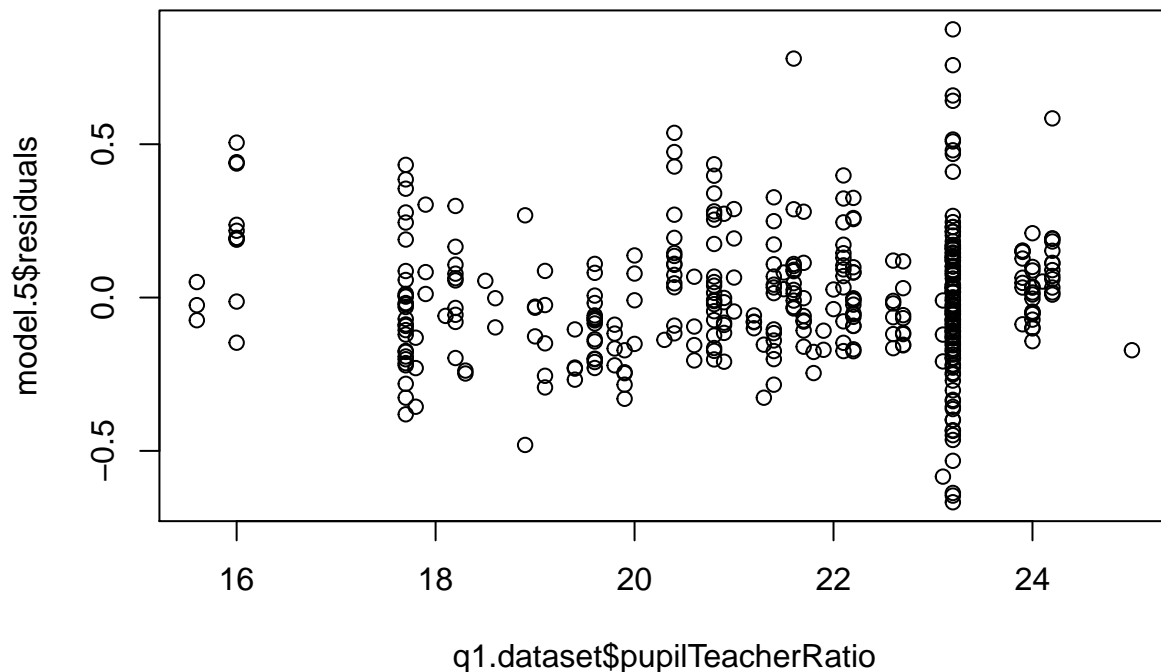
```
plot(q1.dataset$pctLowIncome, model.5$residuals)
```



```
plot(q1.dataset$pollutionIndex, model.5$residuals)
```



```
plot(q1.dataset$pupilTeacherRatio, model.5$residuals)
```



Now, we look at diagnostics for our final model.

1. The residuals vs fitted plot shows a very slight upward trend. The slope is negligible, so we assume zero conditional mean to hold.
2. Errors follow a close to normal distribution. In either case, we have 400 observations, enabling us to rely on OLS asymptotics
3. The scale-location plot shows some trend which is not a significant cause for concern. It shows some heteroskedasticity, which we account for by taking robust standard errors below. Variables in our model remain significant. The Wald-Test shows that the model also remains significant.
4. The Residuals vs Leverage plot shows some outliers but no major cause for concern.

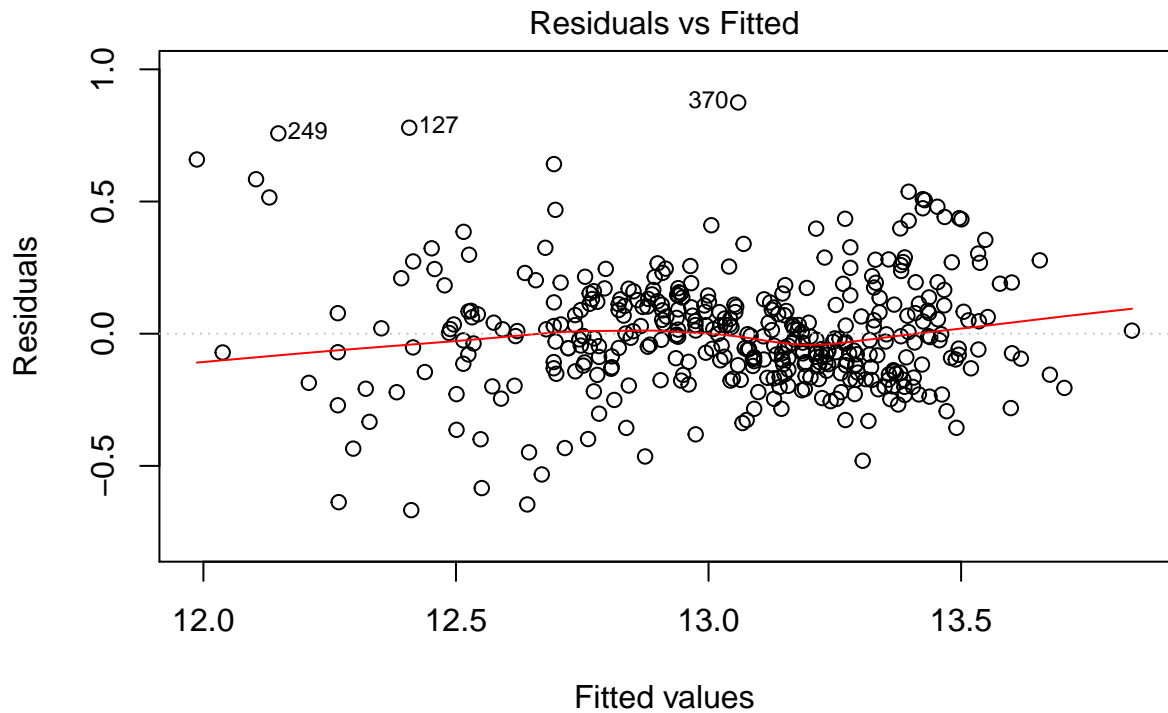
```
coeftest(model.5)
```

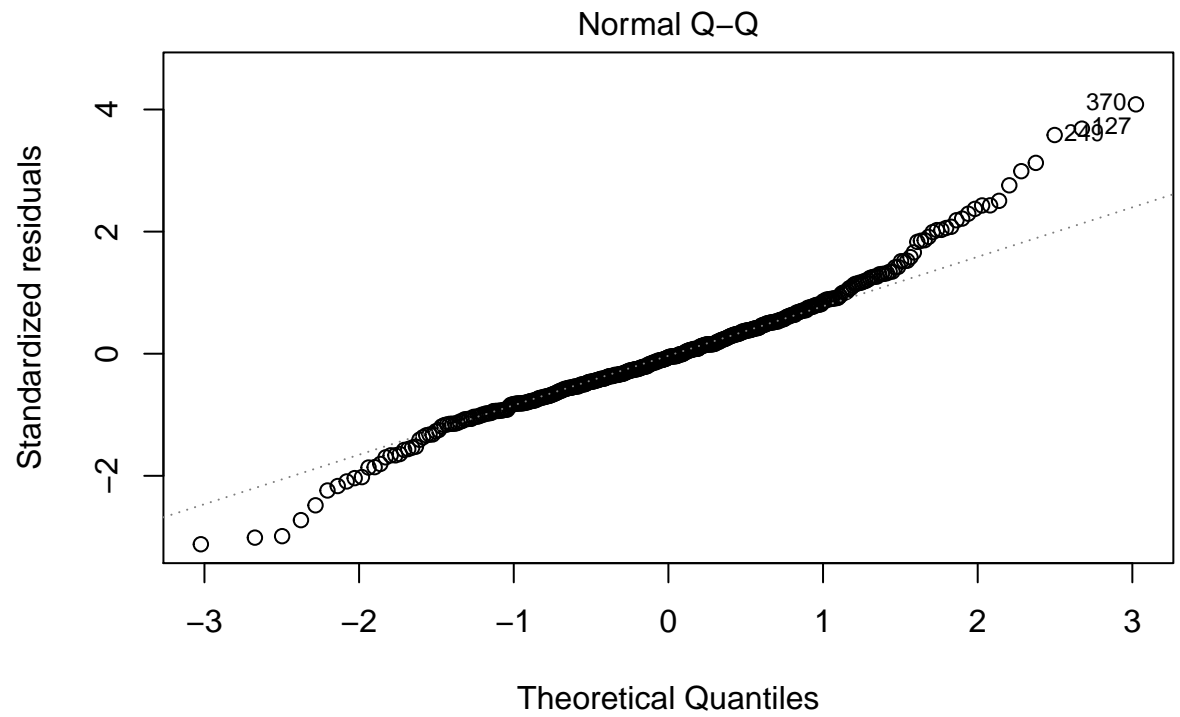
```
##
## t test of coefficients:
##
##               Estimate Std. Error  t value  Pr(>|t|)
## (Intercept)    14.3912717  0.1173529  122.6325 < 2.2e-16 ***
## withWater       0.4936346  0.1410650   3.4993 0.0005197 ***
## pctLowIncome   -0.0287143  0.0015393 -18.6535 < 2.2e-16 ***
## pupilTeacherRatio -0.0373937  0.0054634  -6.8444 2.952e-11 ***
## pollutionIndex  -0.0025045  0.0012175  -2.0572 0.0403278 *
## withWater:pollutionIndex -0.0074373  0.0029605  -2.5122 0.0123980 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
waldtest(model.5, vcov = vcovHC)
```

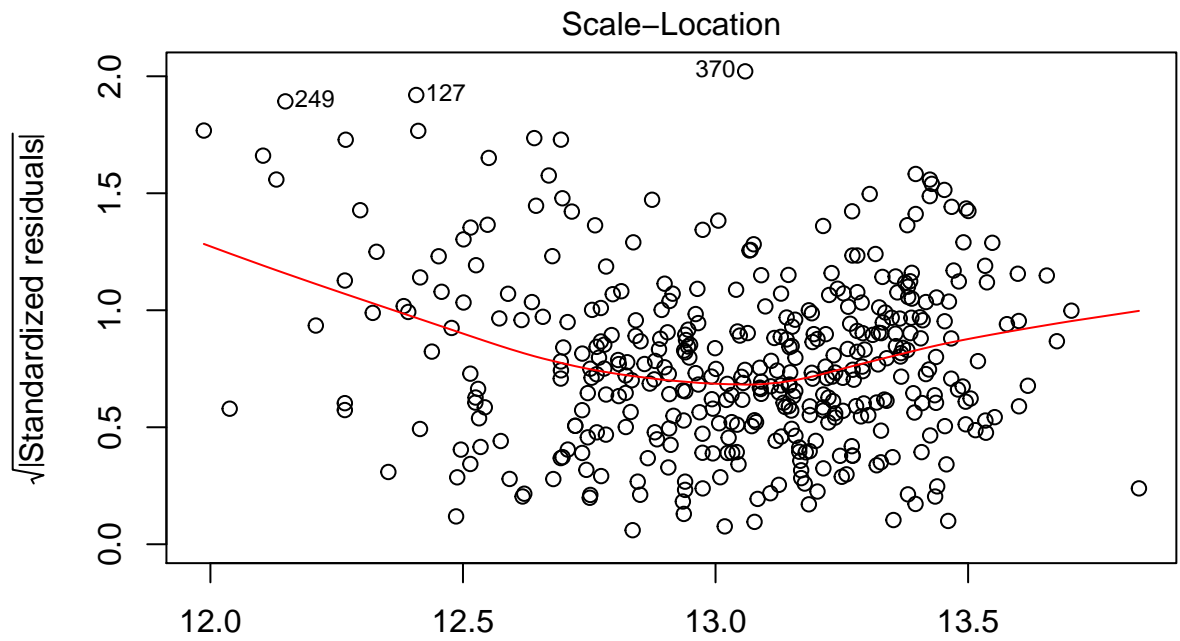
```
## Wald test
##
## Model 1: logHomeValue ~ withWater + pctLowIncome + pupilTeacherRatio +
##   pollutionIndex + withWater:pollutionIndex
## Model 2: logHomeValue ~ 1
##   Res.Df Df      F    Pr(>F)
## 1      394
## 2      399 -5 123.26 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
plot(model.5)
```

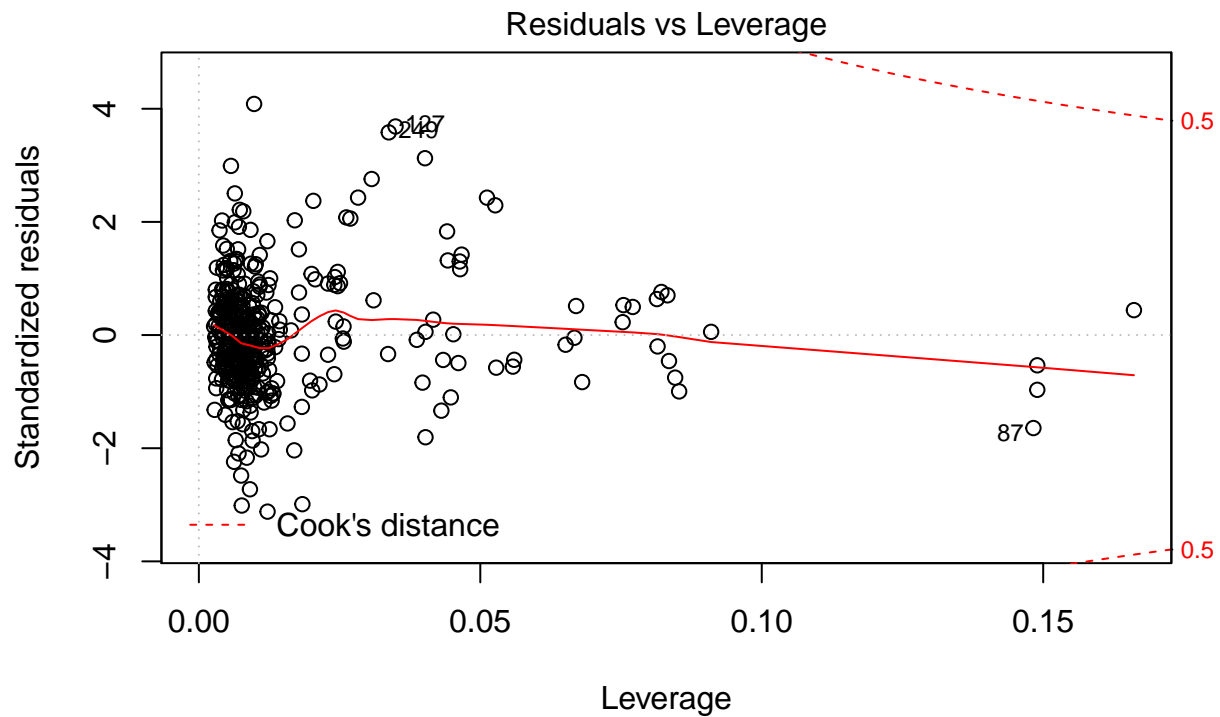




lm(logHomeValue ~ withWater + pctLowIncome + pupilTeacherRatio + pollutionI ...



Fitted values
 $\text{lm}(\log\text{HomeValue} \sim \text{withWater} + \text{pctLowIncome} + \text{pupilTeacherRatio} + \text{pollutionI} \dots$



lm(logHomeValue ~ withWater + pctLowIncome + pupilTeacherRatio + pollutionI ...

Step 4 - Final Model Conclusions

Conclusion

Below we present a comparison of initial and the final models run. We see that despite removing 6 variables, we see a reduction of only around 4% in R square, implying that we have preserved most of the explanatory power of the model, while sticking with parsimony.

```
stargazer(model.1, model.5, type = "text")
```

```
##
## =====
##                               Dependent variable:
##                               -----
##                               logHomeValue
##                               (1)                (2)
## -----
## logCrimeRate_pc                -0.011
##                               (0.013)
##
## nonRetailBusiness              -0.413
##                               (0.265)
##
## withWater                      0.141***
##                               (0.041)                0.494***
##                               (0.141)
##
## ageHouse                       0.0002
##                               (0.001)
##
## logDistanceToCity              -0.129***
##                               (0.025)
##
## distanceToHighway              -0.001
##                               (0.003)
##
## pupilTeacherRatio              -0.030***
##                               (0.006)                -0.037***
##                               (0.005)
##
## pctLowIncome                   -0.024***
##                               (0.002)                -0.029***
##                               (0.002)
##
## pollutionIndex                 -0.008***
##                               (0.002)                -0.003**
##                               (0.001)
##
## nBedRooms                      0.103***
##                               (0.019)
##
## withWater:pollutionIndex       -0.007**
##                               (0.003)
##
## Constant                      14.233***
##                               (0.211)                14.391***
##                               (0.117)
##
```

```
## -----
## Observations                400                400
## R2                          0.753                0.710
## Adjusted R2                 0.747                0.706
## Residual Std. Error         0.200 (df = 389)        0.215 (df = 394)
## F Statistic                 118.510*** (df = 10; 389) 192.789*** (df = 5; 394)
## =====
## Note:                       *p<0.1; **p<0.05; ***p<0.01
```

Explanation of the Model

The final results from the model we present to the think-tank are as follows:

1. Home values are 50% greater for homes located within 5 miles of water.
2. For every unit percentage increase in low-income households, home values are close to 3% lower.
3. For every one unit increase in the pupilTeacherRatio in a neighbourhood, home values are close to 4% lower.
4. For an additional unit on the pollutionIndex, we expect to see a decrease in home value of 0.2% if it is in a neighbourhood not within 5 miles of a water body. However, if it is within 5 miles of a water body, we expect an almost 1% decrease in home value.

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 instrument or index.

We observe that the time series is non-stationary in the mean. Therefore we can attempt to diff the time series to see if the resulting series is stationary in the mean. We also observe that the PACF of the time series indicates a correlation at lag 1 that resembles an AR(1) series.

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 other than the first. 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.

The 1st 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 suggest white noise dynamics. 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 ...
```

```
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
```

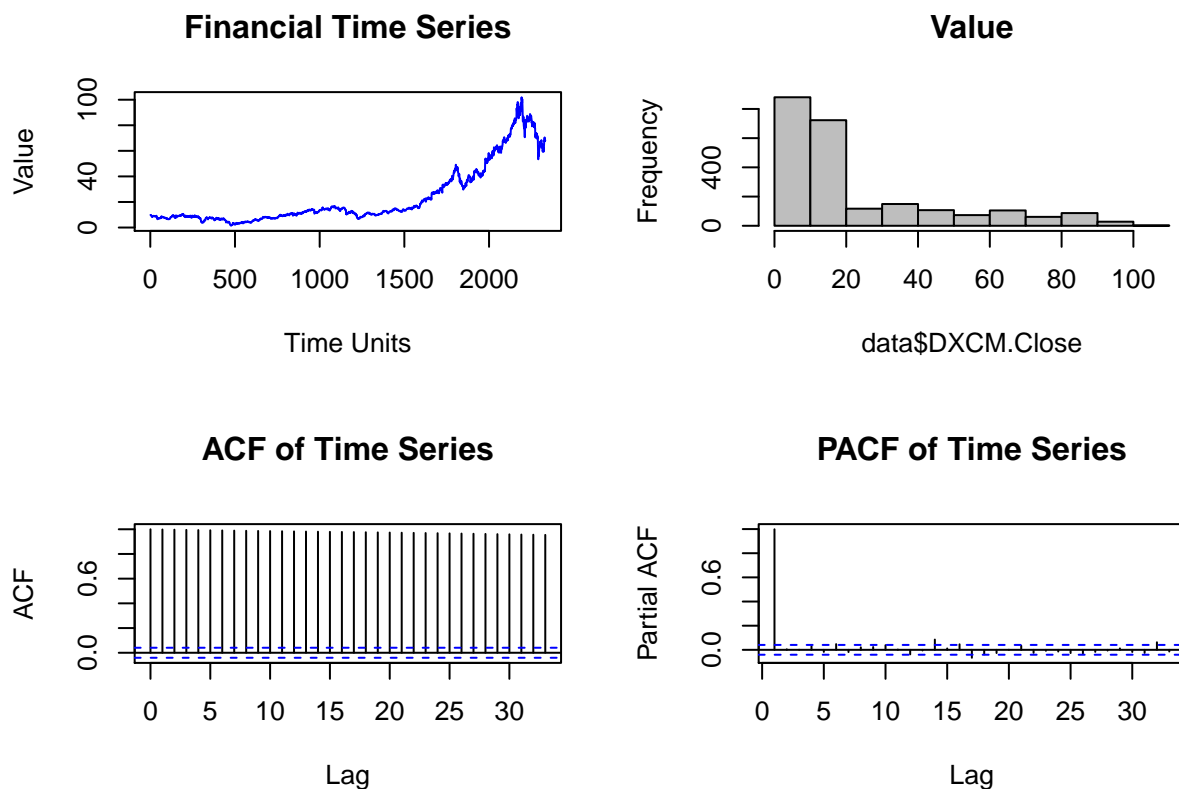
```
cbind(head(data), tail(data))
```

```
##   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
```

```
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))
```

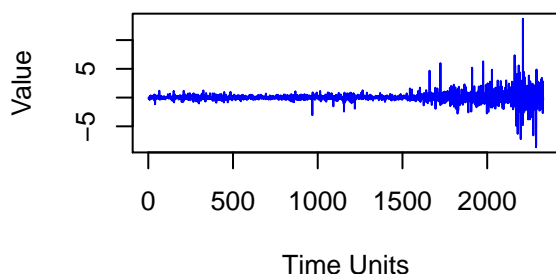
##	0%	1%	5%	10%	25%	50%	75%	90%
##	1.3900	2.7531	4.1700	6.1630	8.1875	12.3550	32.5650	63.5210
##	95%	99%	100%					
##	80.0430	91.6887	101.9100					

```
# 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")
```

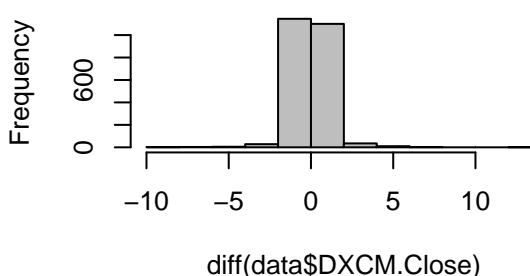


```
# EDA on the first difference series
par(mfrow = c(2, 2))
plot.ts(diff(data$DXCM.Close), main = "First Difference Financial Time Series",
        ylab = "Value", xlab = "Time Units", col = "blue")
hist(diff(data$DXCM.Close), col = "gray", main = "Value")
acf(diff(data$DXCM.Close), main = "ACF of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

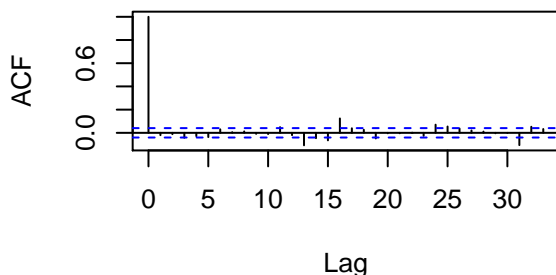
First Difference Financial Time Series



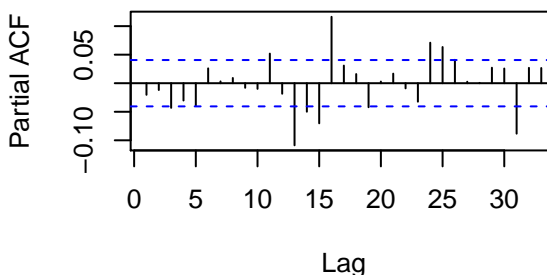
Value



ACF of Time Series



PACF of Time Series



We now try to determine a best SARIMA model based on the best AIC for that series. Interestingly enough, the best model based on AIC is a seasonal model with (p,d,q,P,D,Q) of values $(0, 0, 0, 1, 0, 0)$. Since we're assuming a frequency of one, that model is the same as the second best model, which has orders (p,d,q,P,D,Q) of $(0, 1, 0, 0, 0, 0)$. 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 one with a differential component of value one, and has no other parameters for which we need to validate 95 confidence intervals.

```
# Fitting a first difference series to our model
data.fit <- Arima(data$DXCM, order = c(0, 1, 0), seasonal = list(order = c(0,
  0, 0)), method = "CSS-ML")
data.res <- data.fit$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%      50%      75%      90%      95%
## -8.6500 -2.4369 -1.1100 -0.5600 -0.2300  0.0000  0.2500  0.6400  1.1500
##      99%     100%
##  2.8538 13.7000
```

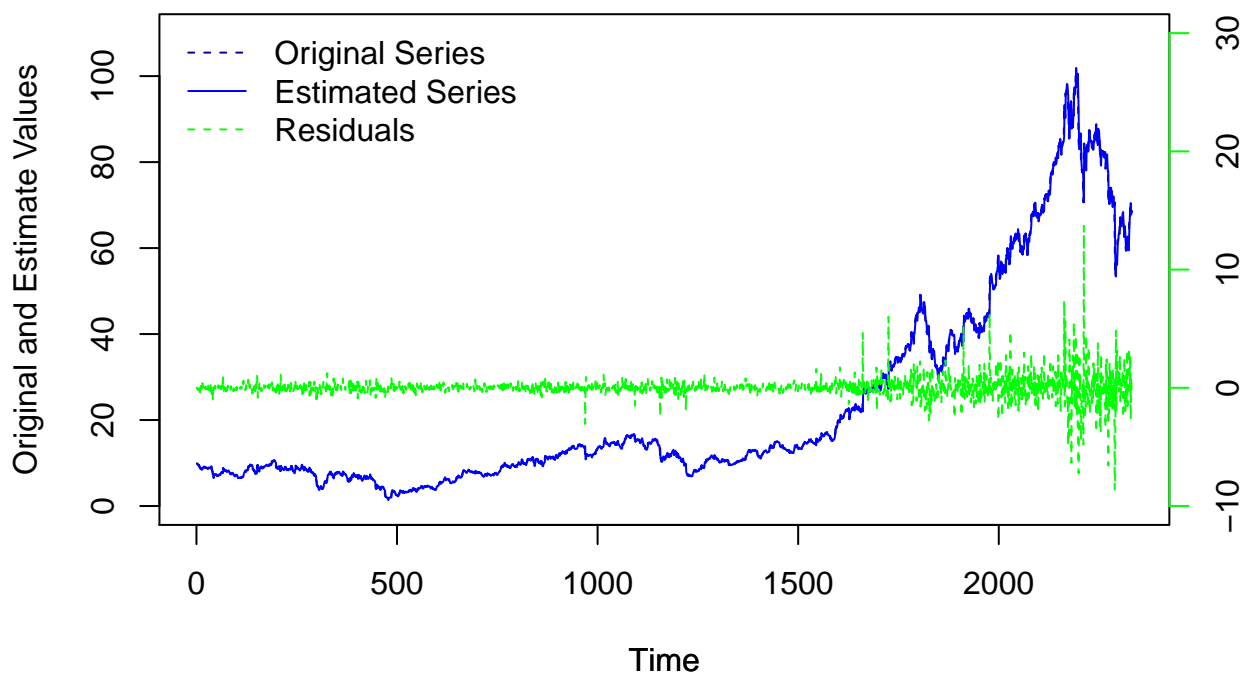
```
t(confint(data.fit))
```

```
##
## 2.5 %
## 97.5 %
```

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 appropriate based on in-sample fit.

```
# Performing in-sample fit using our fitted series
par(mfrow = c(1, 1))
plot.ts(data$DXCM, col = "navy", lty = 2, main = "Original vs a SAMIMA(0,1,0,0,0,0) Estimated Series with Residuals",
        ylab = "Original and Estimate Values", ylim = c(0, 110))
par(new = T)
plot(fitted(data.fit), col = "blue", axes = F, ylab = "", ylim = c(0, 110))
leg.txt <- c("Original Series", "Estimated Series", "Residuals")
legend("topleft", legend = leg.txt, lty = c(2, 1, 2), col = c("navy", "blue", "green"), bty = "n", cex = 1)
par(new = T)
plot.ts(data$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")
```

Original vs a SAMIMA(0,1,0,0,0,0) Estimated Series with Residuals



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 likely explains this dynamic. We next turn our eyes to the residuals of the fitted series and to the 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))
```

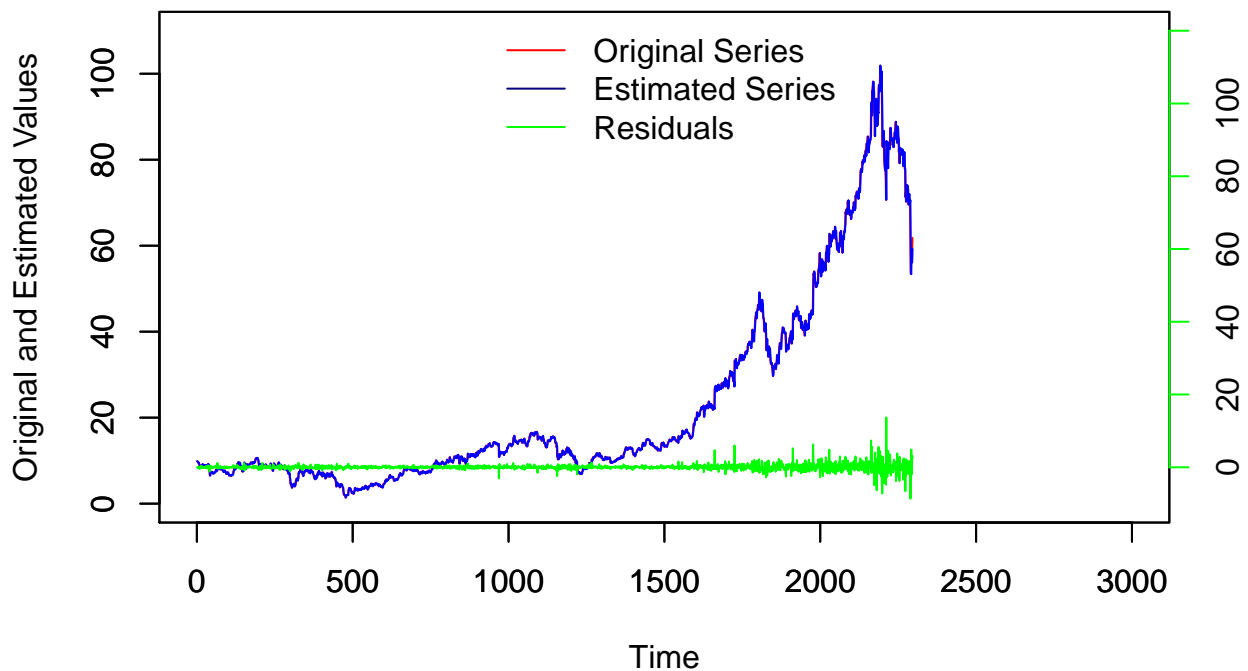


```

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")

```

Original vs a AR(1) Estimated Series with Residuals



```

# 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
```

```

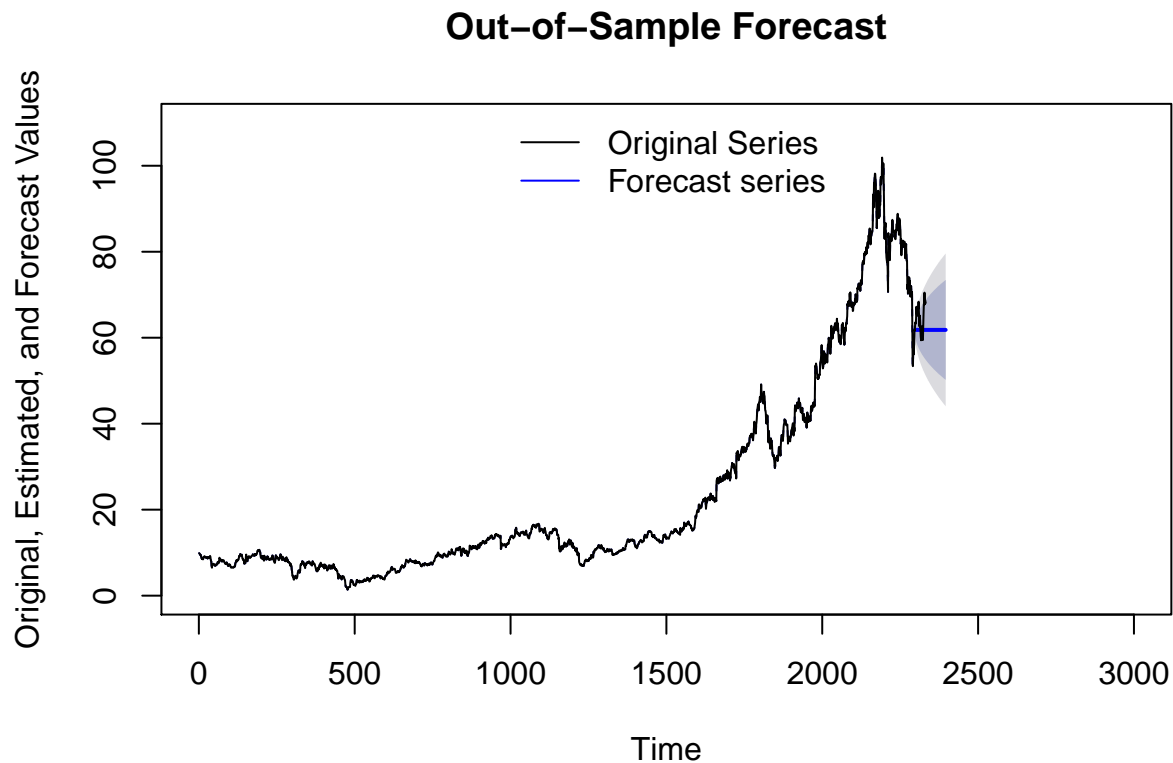
par(mfrow = c(1, 1))
plot(data.fit.back.fcast, lty = 2, col = "navy", main = "Out-of-Sample Forecast",
     ylab = "Original, Estimated, and Forecast Values", xlim = c(0,
     3000), ylim = c(0, 110))
par(new = T)
plot.ts(data$DXCM, axes = F, lty = 1, xlim = c(0, 3000), ylim = c(0,

```

```

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

```

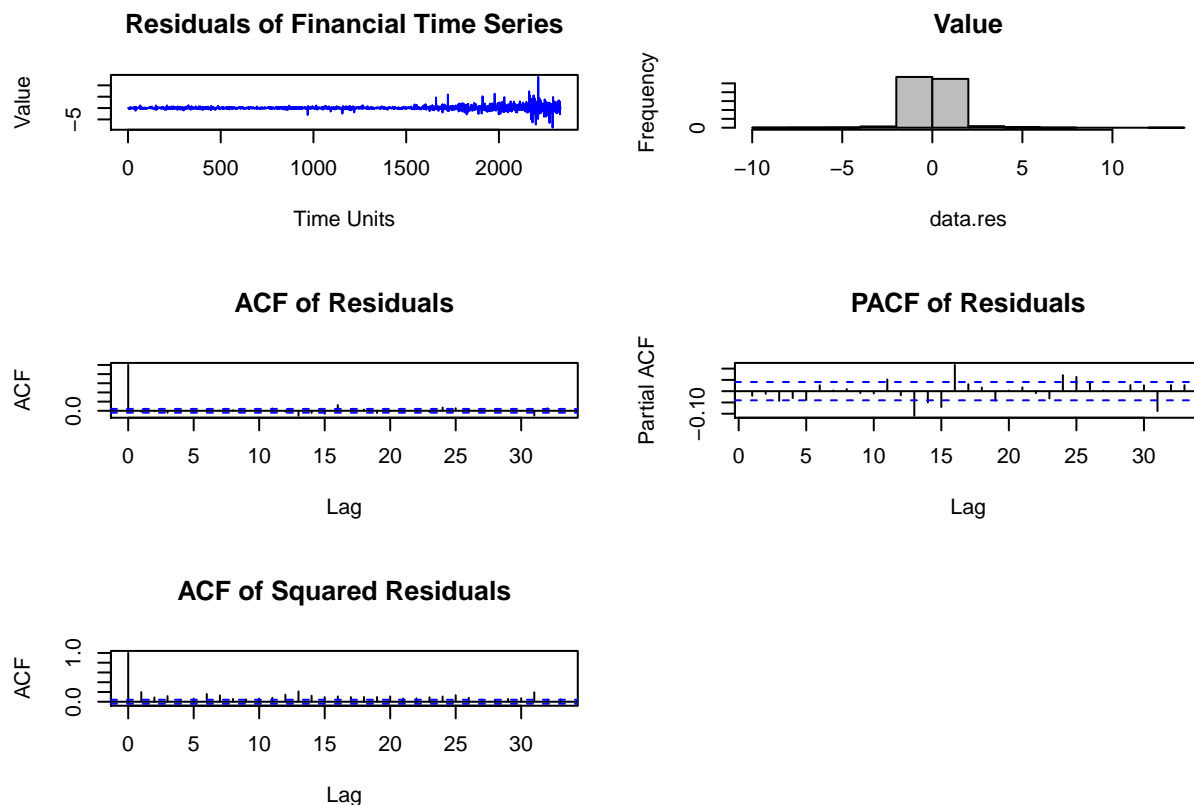


We observe from the residual time series that the variance of the series is non-stationary. The series exhibits volatility with a variance changing in a regular way. It exhibits conditional heteroskedasticity. An observation of the PACF of the squared residuals series provides confirmation of the variance 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 characteristics 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 residuals of the financial time series using
# GARCH
data.garch <- garch(data.res, trace = F)
```

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

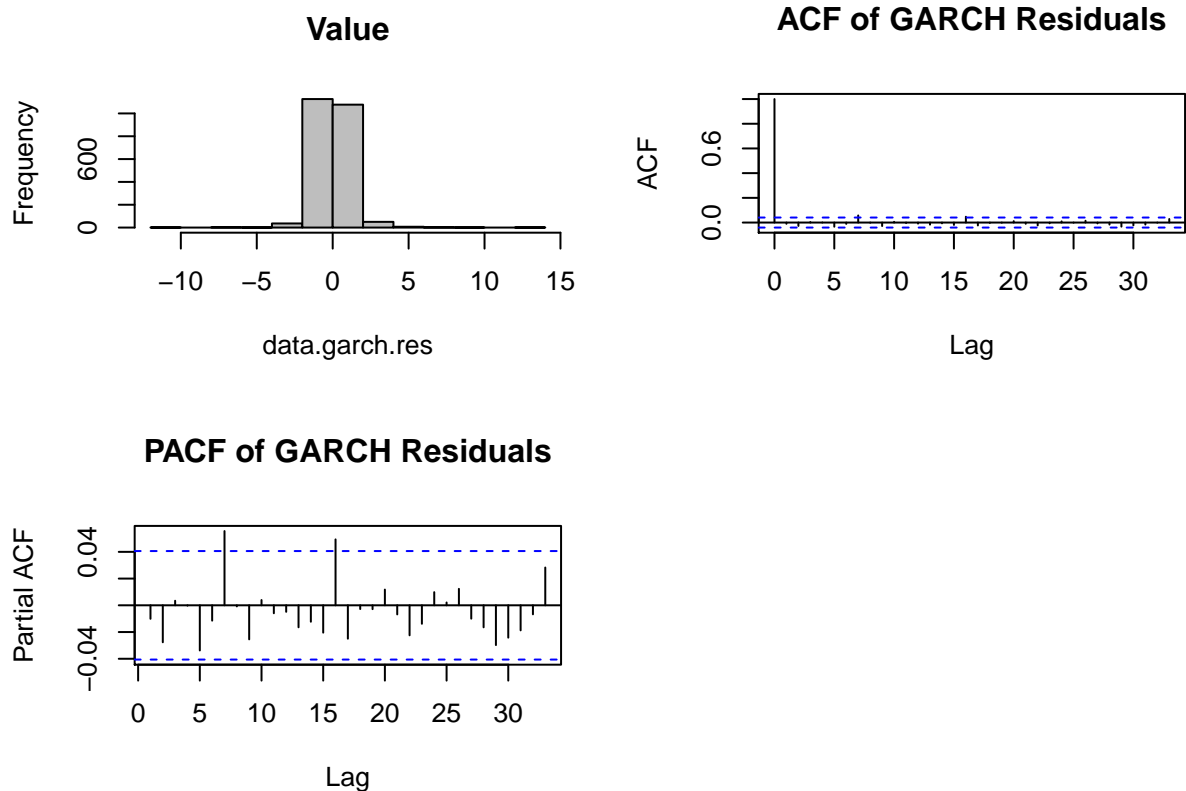
```
t(confint(data.garch))
```

```
##           a0           a1           b1
## 2.5 % 5.480144e-05 0.03050862 0.9655033
## 97.5 % 4.229287e-04 0.03925722 0.9734176
```

```
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 about volatility 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.

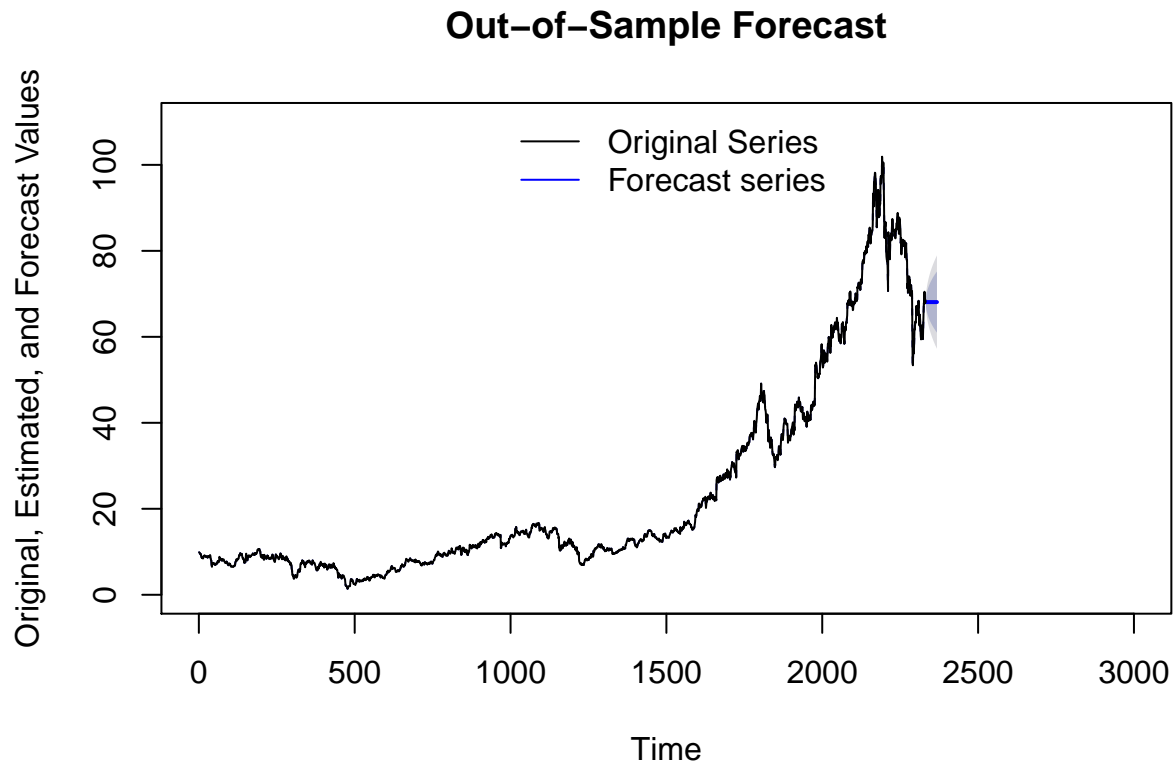
To perform a 36 steps ahead forecast of our series, we use the original point estimate of the series, that being the (0,1,0,0,0) SARIMA model initially estimated. The GARCH model of the residuals will additionally be used to forecast the variance of the series, and help us adjust the confidence interval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

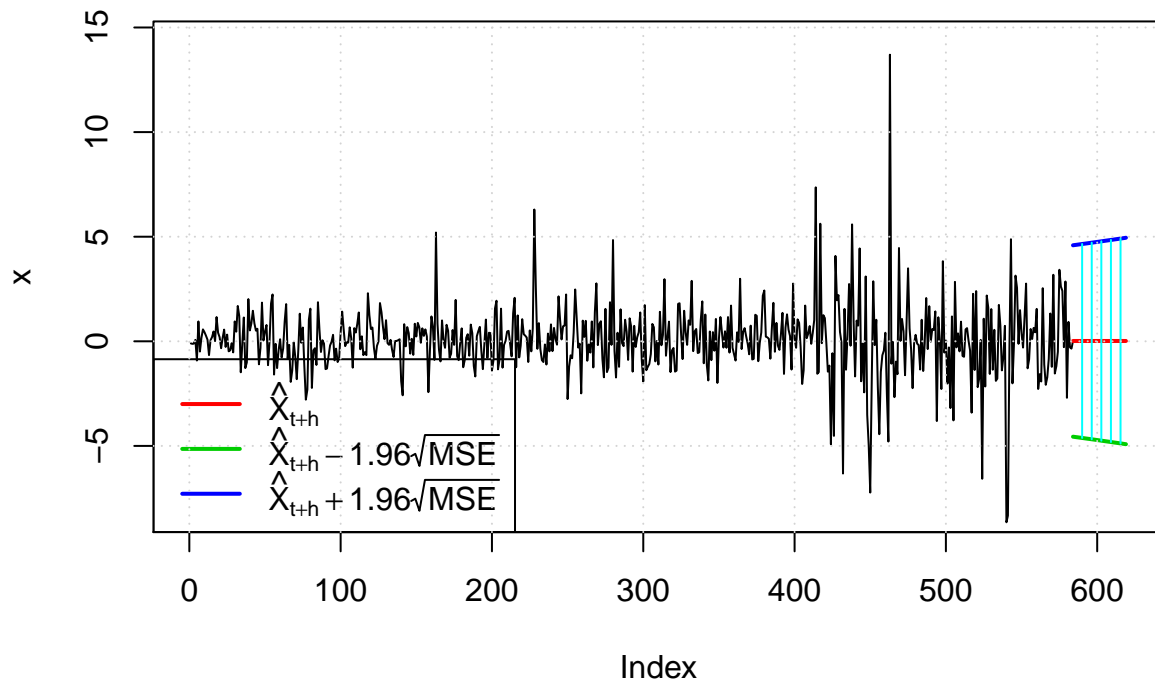


Having acknowledged the confidence interval problem on the prediction caused by the non-stationary variance of the financial search time series, we want to use our fitted GARCH model to predict the mean and variance of the residuals of the series.

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 (0,1,0,0,0) model. We note that the volatility is predicted by the GARCH model to be in the range of -5 to 5, wider than the range 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)
data.garch.pred <- predict(data.garch.fit, n.ahead = 36, plot = TRUE)
```

Prediction with confidence intervals



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) series.

The mean series of the 36 steps ahead predictions obtained from the fitted SARIMA(0,1,0,0,0) model and the corresponding lower and upper confidence intervals after substituting for the conditional variance obtained from GARCH model are:

```
# Reset the 95% confidence interval using the conditional
# variance from GARCH
data.fit.ahead.fcast$lower[, 2] <- -(((data.fit.ahead.fcast$mean -
  data.fit.ahead.fcast$lower[, 2])/(1.96 * sqrt(data.fit.ahead.fcast$model$sigma2))) *
  data.garch.pred$standardDeviation) + data.fit.ahead.fcast$mean

data.fit.ahead.fcast$upper[, 2] <- (((data.fit.ahead.fcast$upper[,
  2] - data.fit.ahead.fcast$mean)/(1.96 * sqrt(data.fit.ahead.fcast$model$sigma2))) *
  data.garch.pred$standardDeviation) + data.fit.ahead.fcast$mean

df <- data.frame(cbind(data.fit.ahead.fcast$mean, data.fit.ahead.fcast$lower[,
  2], data.fit.ahead.fcast$upper[, 2]))
names(df) <- c("Mean", "Lower 95% conf", "Upper 95% conf")
stargazer(df, type = "text", summary = FALSE)
```

```
##
## =====
##      Mean  Lower 95% conf  Upper 95% conf
## -----
```

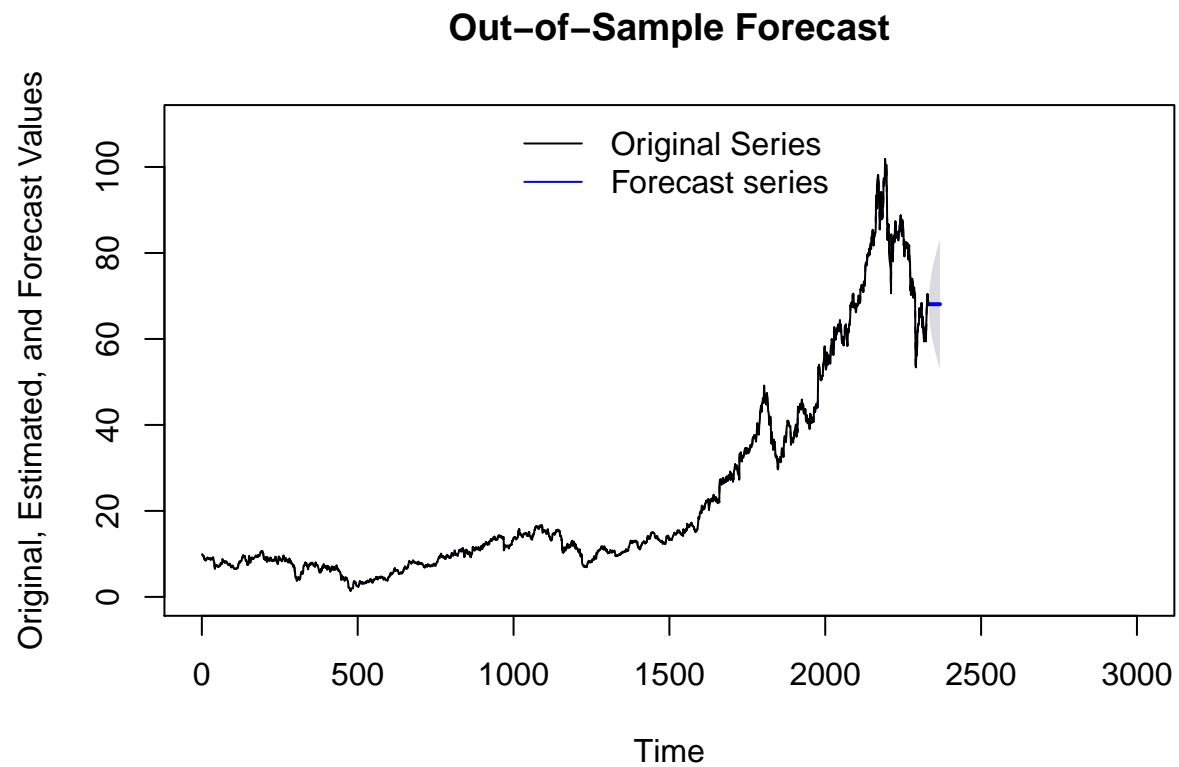
```
## 1 68.080      65.746      70.414
## 2 68.080      64.772      71.388
## 3 68.080      64.020      72.140
## 4 68.080      63.382      72.778
## 5 68.080      62.816      73.344
## 6 68.080      62.301      73.859
## 7 68.080      61.824      74.336
## 8 68.080      61.378      74.782
## 9 68.080      60.956      75.204
## 10 68.080     60.555      75.605
## 11 68.080     60.170      75.990
## 12 68.080     59.801      76.359
## 13 68.080     59.444      76.716
## 14 68.080     59.098      77.062
## 15 68.080     58.763      77.397
## 16 68.080     58.437      77.723
## 17 68.080     58.118      78.042
## 18 68.080     57.807      78.353
## 19 68.080     57.503      78.657
## 20 68.080     57.205      78.955
## 21 68.080     56.912      79.248
## 22 68.080     56.625      79.535
## 23 68.080     56.342      79.818
## 24 68.080     56.064      80.096
## 25 68.080     55.789      80.371
## 26 68.080     55.519      80.641
## 27 68.080     55.252      80.908
## 28 68.080     54.988      81.172
## 29 68.080     54.728      81.432
## 30 68.080     54.470      81.690
## 31 68.080     54.215      81.945
## 32 68.080     53.963      82.197
## 33 68.080     53.713      82.447
## 34 68.080     53.465      82.695
## 35 68.080     53.220      82.940
## 36 68.080     52.977      83.183
## -----
```

Having computed 95% confidence intervals based on GARCH, we now plot the predicted point series with the 95% confidence interval updated.

```
# Clear the 80% confidence interval
data.fit.ahead.fcast$lower[, 1] <- data.fit.ahead.fcast$mean
data.fit.ahead.fcast$upper[, 1] <- data.fit.ahead.fcast$mean

# Plot the forecast with the updated 95% confidence interval
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 95% 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 an 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) model to handle trends and seasonality and a GARCH model to handle the time-varying volatility. We observe that the forecasts of the model are consistent 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))
```

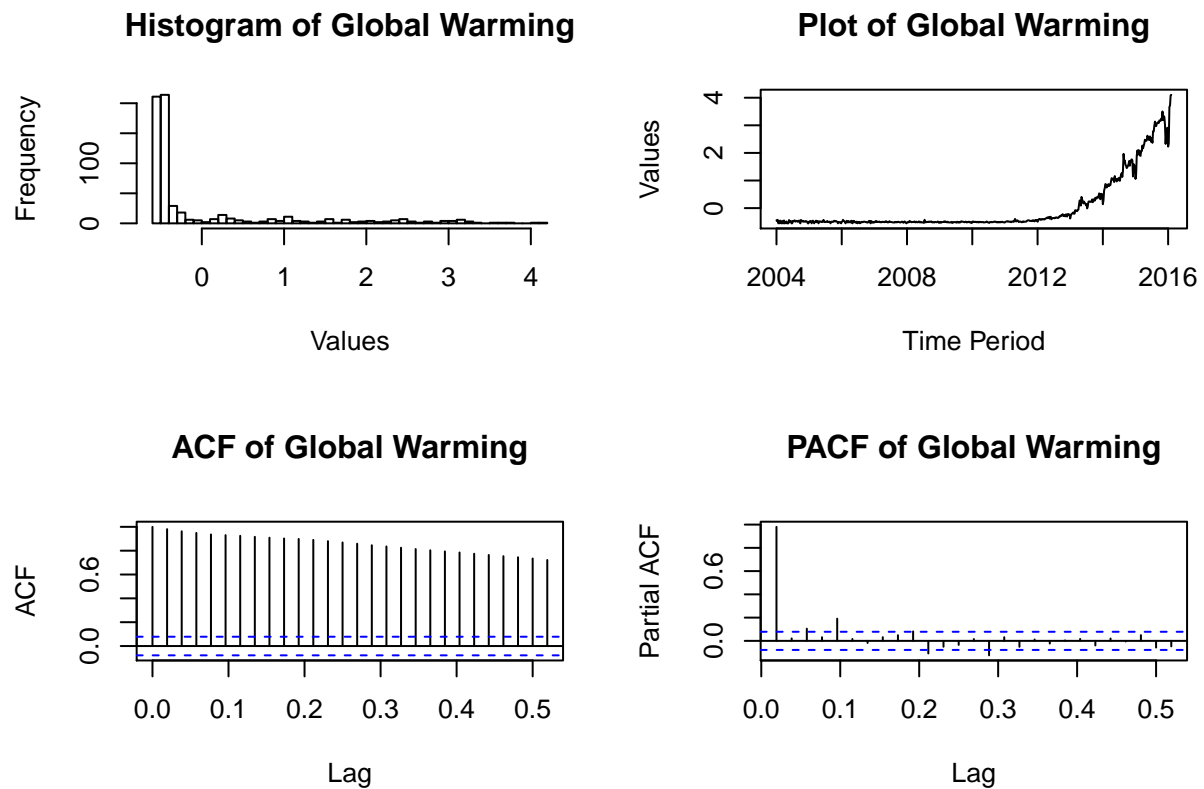
```
##      [,1] [,2]
## [1,] -0.440 2.227
## [2,] -0.474 2.360
## [3,] -0.423 3.662
## [4,] -0.551 3.721
## [5,] -0.486 4.087
## [6,] -0.551 4.104
```

```
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")
```

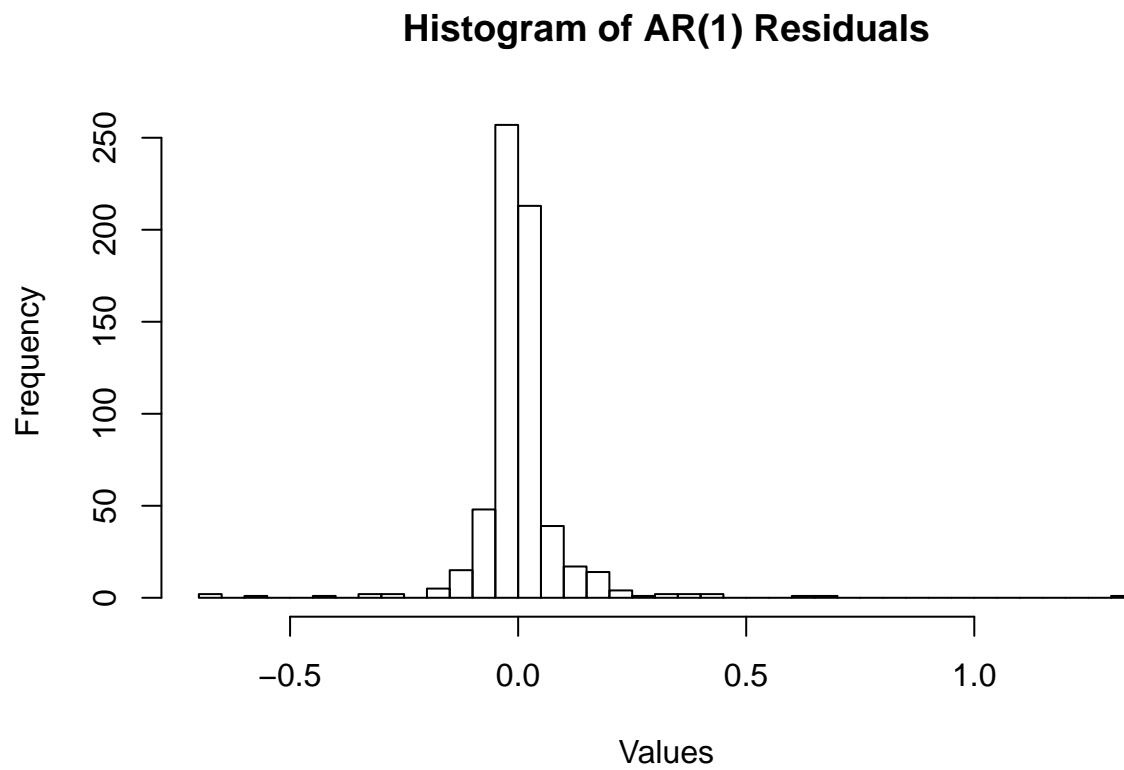
```
## Time-Series [1:630] from 2004 to 2016: -0.44 -0.474 -0.423 -0.551 -0.486 -0.551 -0.453 -0.462 -0.55
```

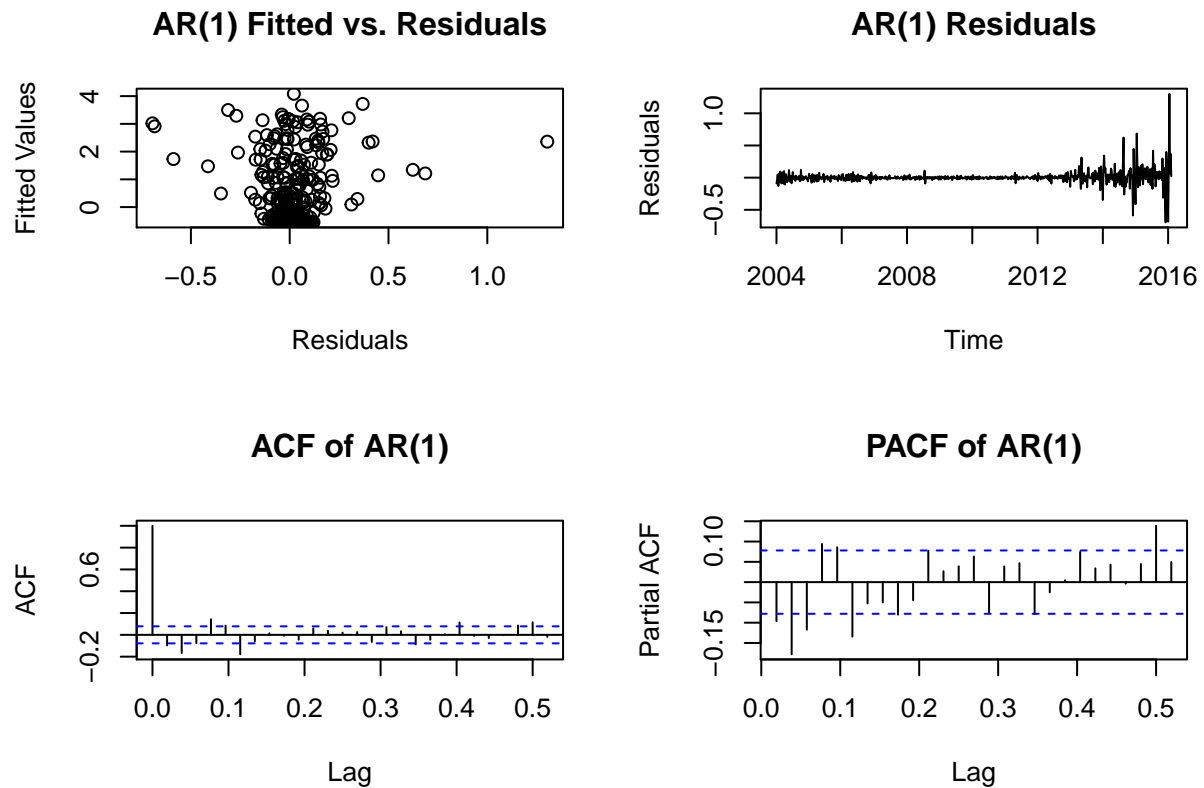


```
### 1. Try AR models
glob.warm.ar = estimate.ar(glob.warm.ts)
```

```
## [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)")
```



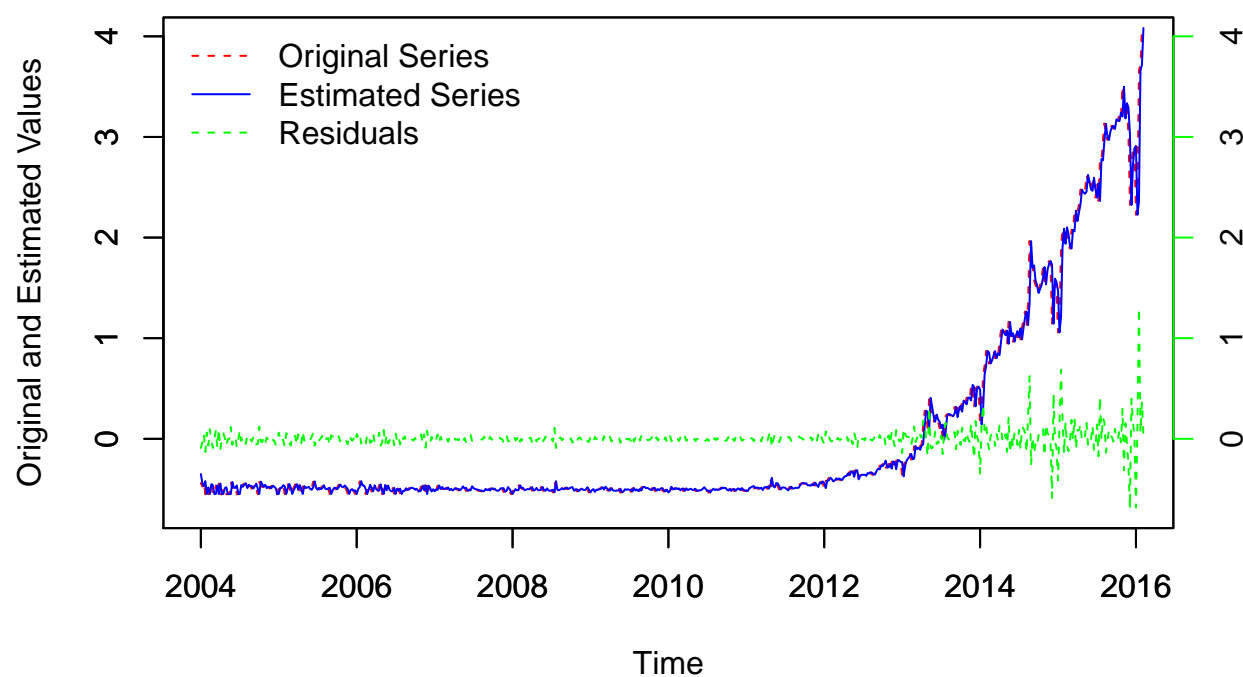


```
##
## Box-Ljung test
##
## data: x.mod$residuals
## X-squared = 5.8789, df = 1, p-value = 0.01532
```

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

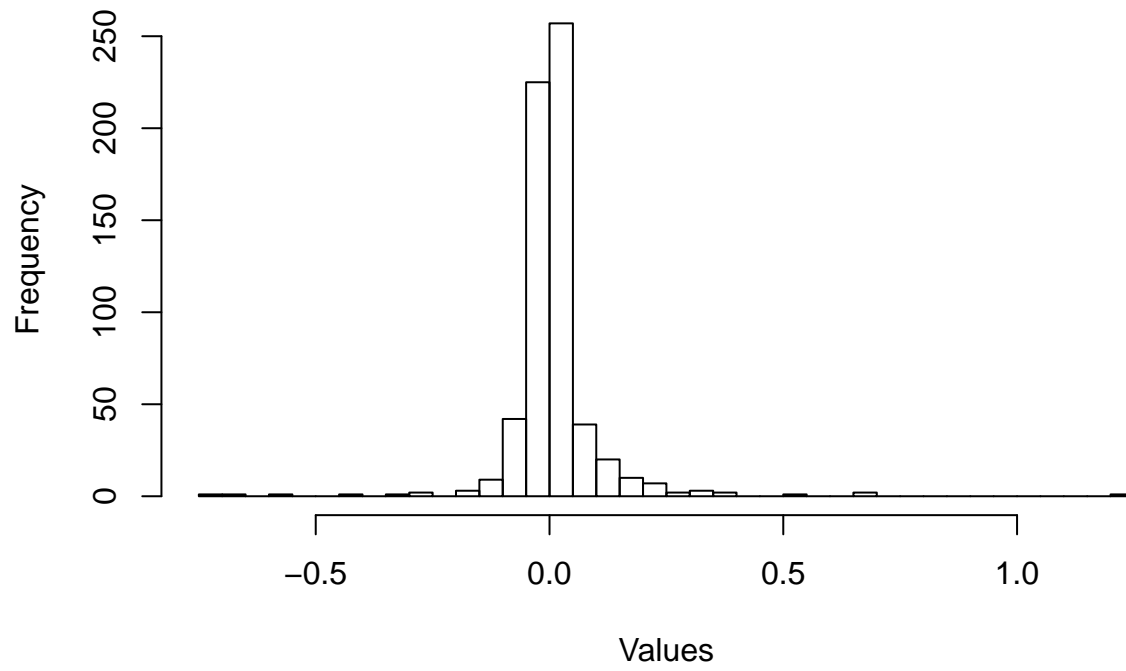
```
##
## Descriptive Stat
## =====
## Statistic      N Mean St. Dev. Min Max
## -----
## x.ts           630 0.000  1.0   -0.6 4.1
## fitted.x.mod.   630 -0.01  1.0   -0.5 4.1
## x.mod.residuals 630 0.01   0.1   -0.7 1.3
## -----
```

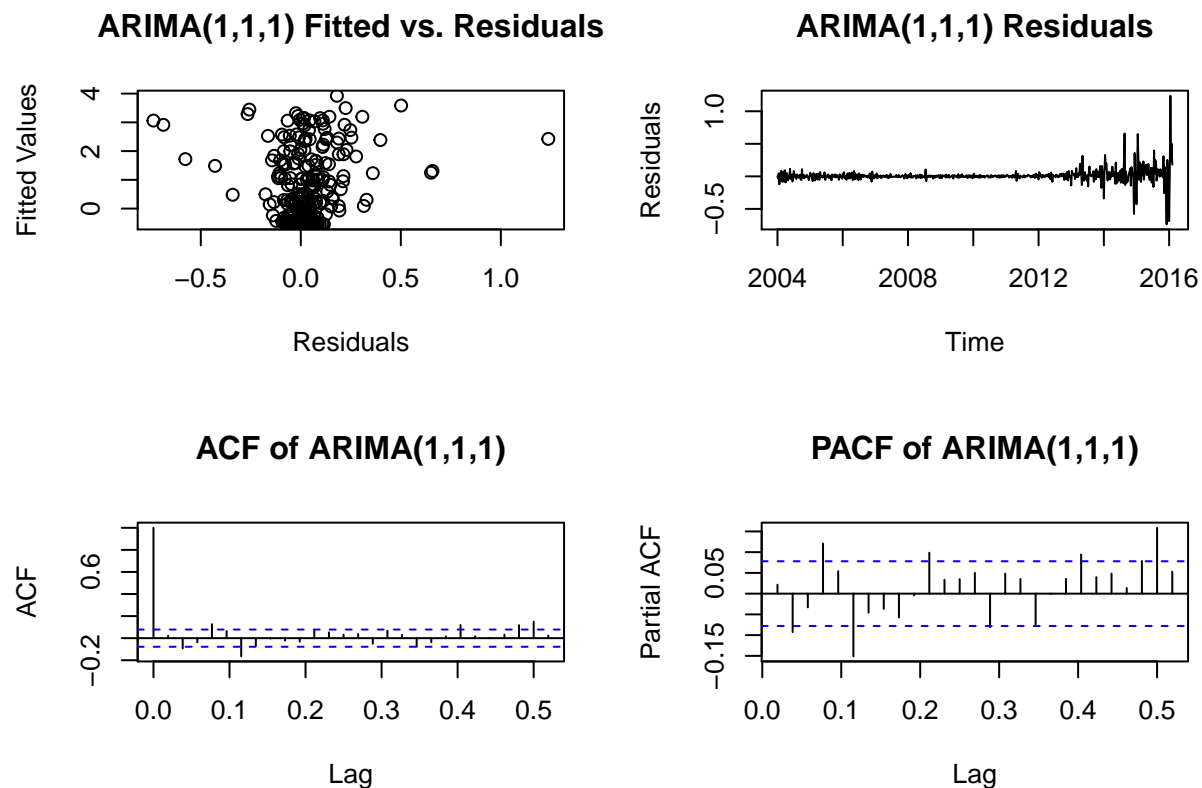
Orivinal vs Estimated AR(1) Series with Resdialuls



```
### 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



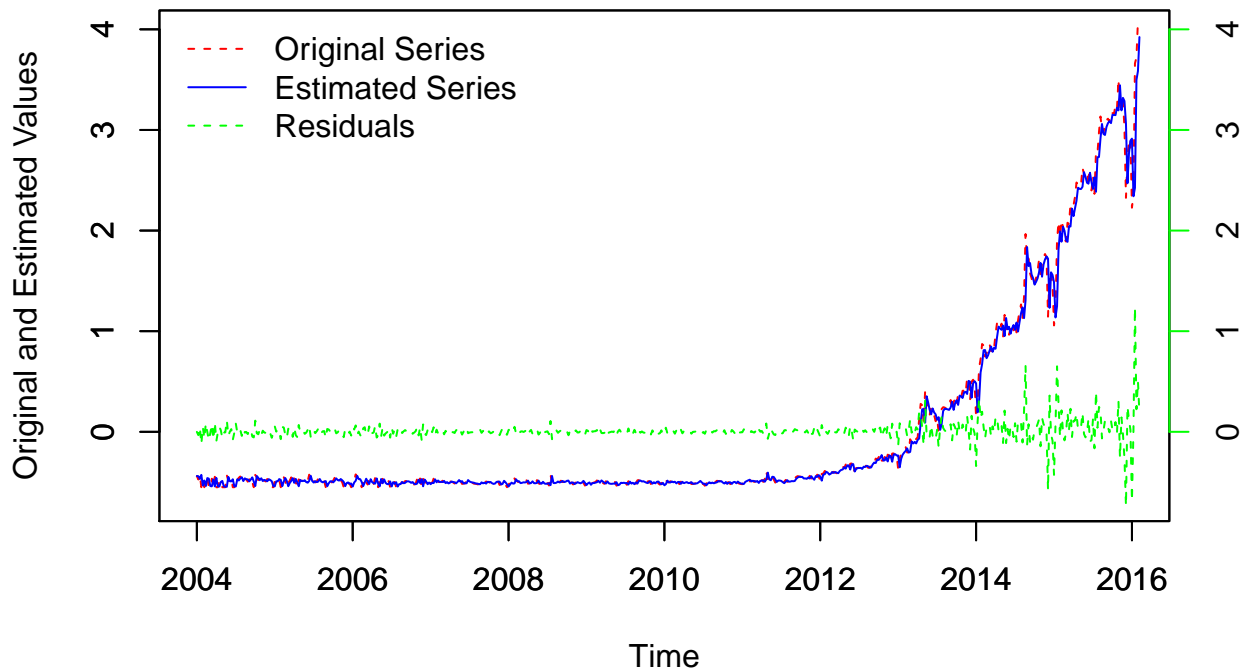


```
##
## Box-Ljung test
##
## data: x.mod$residuals
## X-squared = 0.29725, df = 1, p-value = 0.5856
```

```
# 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
## -----
```

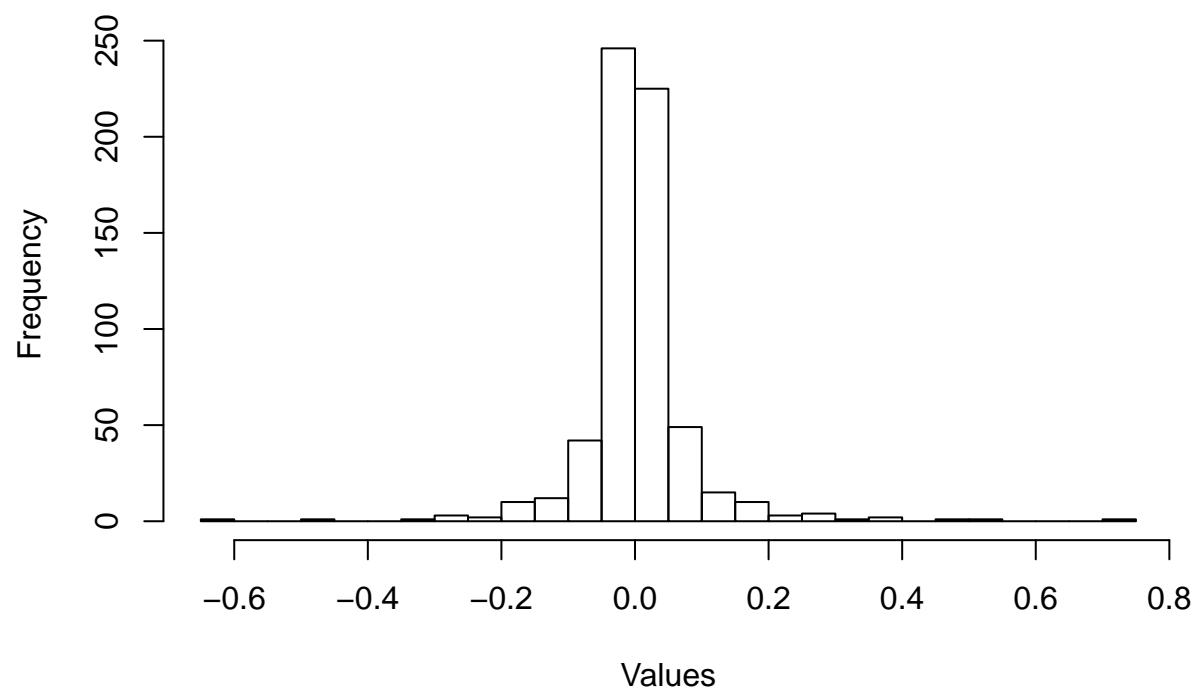

Orivinal vs Estimated ARIMA(1,1,1) Series with Resdialus



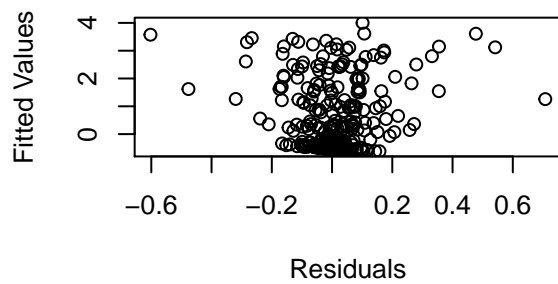
```
### 3. Try SARIMA models gw.seas.best <-
### get.best.sarima(glob.warm.ts, maxord=c(2,2,2,2,2,2), 52)
### Print the top 20 best models based on AIC
### gw.seas.best$others[order(gw.seas.best$others$aics)[1:20],]

# 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)")
```

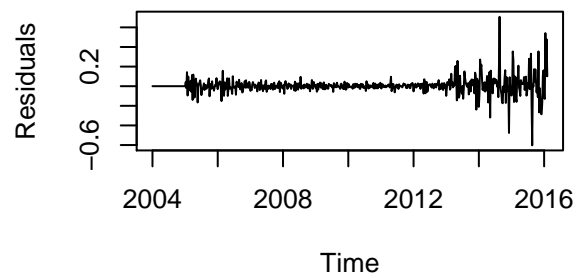
Histogram of SARIMA(0,1,1,1,0,1) Residuals



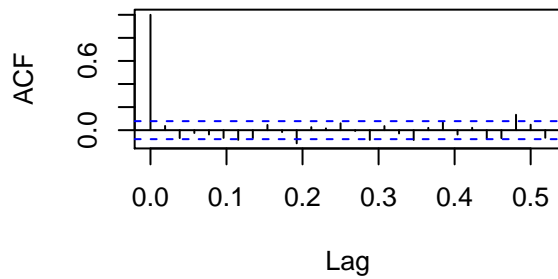
SARIMA(0,1,1,1,0,1) Fitted vs. Residual



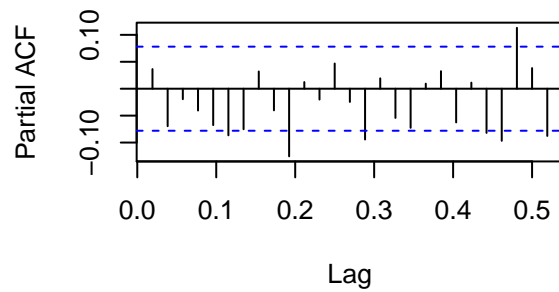
SARIMA(0,1,1,1,0,1) Residuals



ACF of SARIMA(0,1,1,1,0,1)



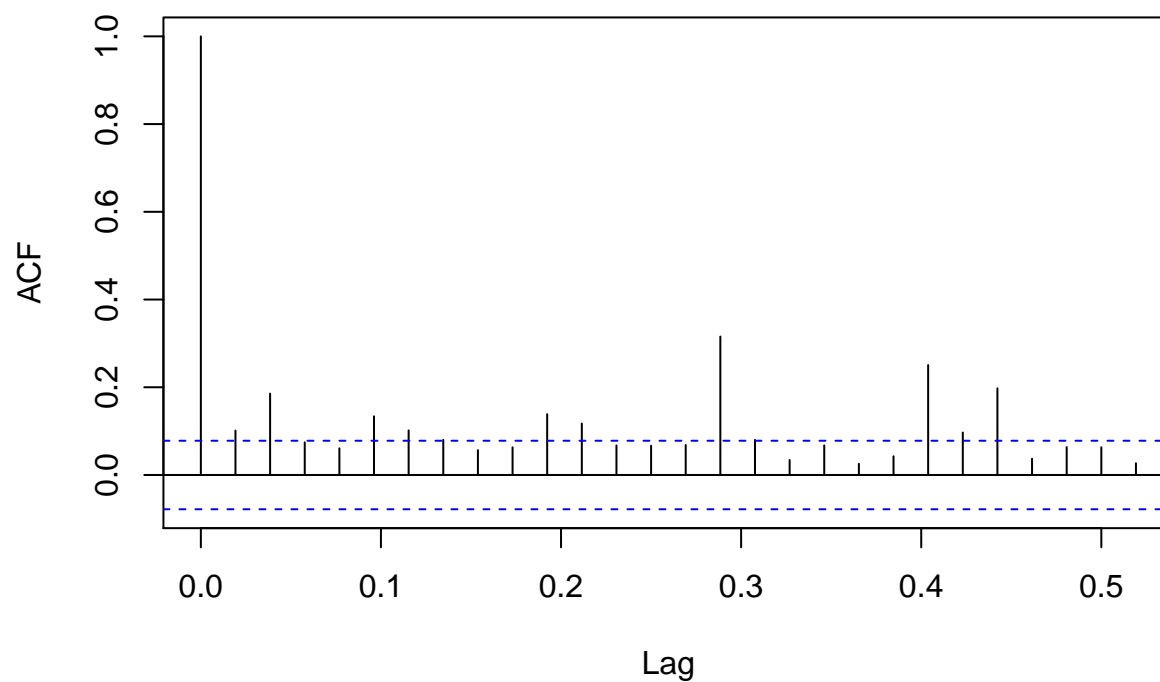
PACF of SARIMA(0,1,1,1,0,1)



```
##  
## 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)
```

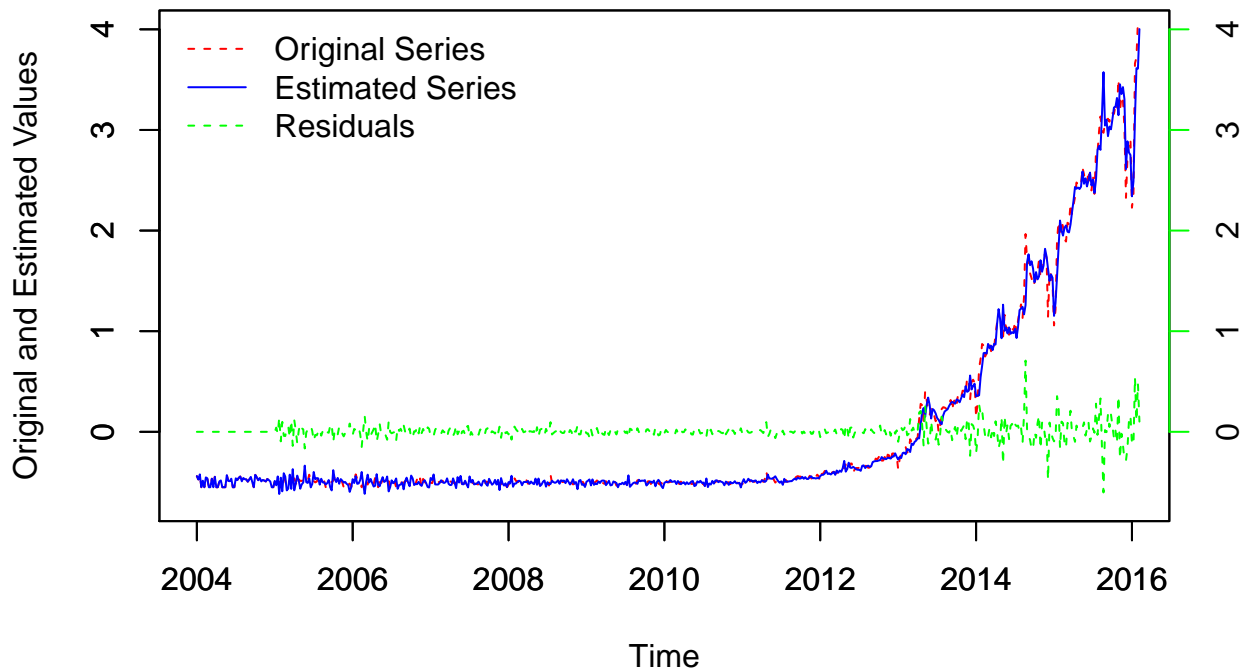
Series glob.warm.arima.seas.res^2



```
# 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
## -----
```

Orivinal vs Estimated SARIMA(0,1,1,1,0,1) Series with Resdialus



```
### 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
```

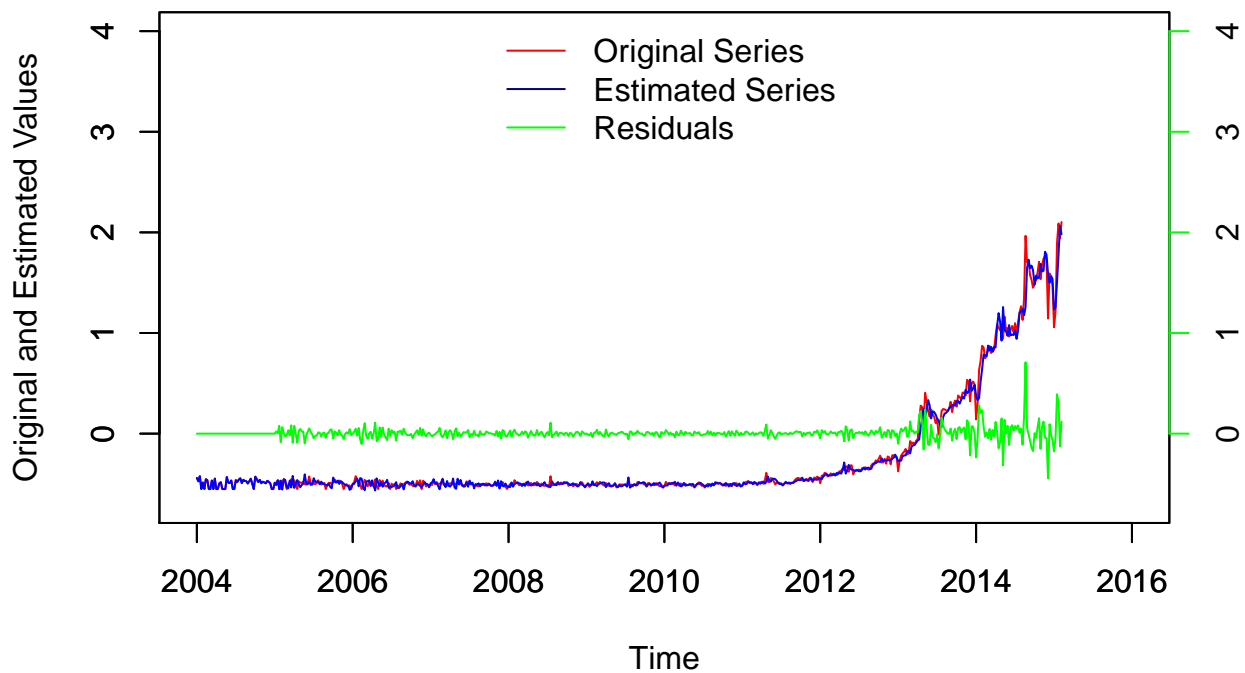
```
# Plot the original and estimate series
par(mfrow = c(1, 1))
plot.ts(df[, "orig_series"], col = "red", main = "Original vs SARIMA Estimated Series with Residuals",
  ylab = "Original and Estimated Values", xlim = c(2004, 2016),
  ylim = c(-0.7, 4))
par(new = T)
plot.ts(df[, "fitted_vals"], col =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

```

Original vs SARIMA Estimated Series with Residuals

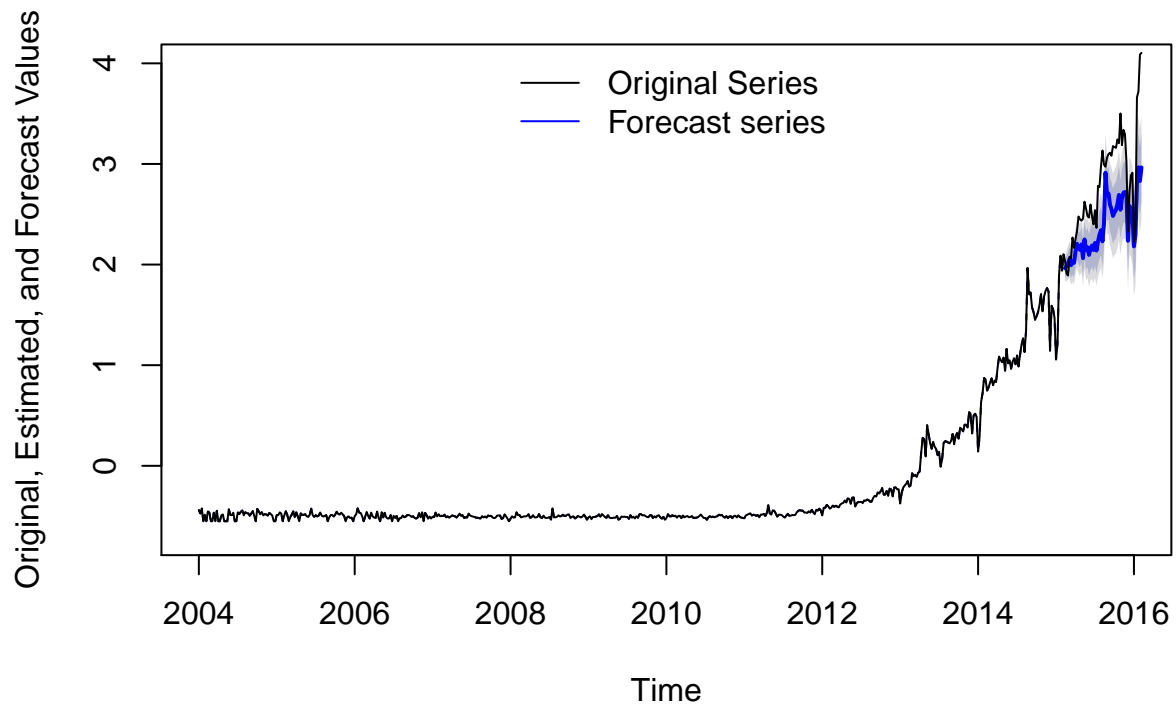


```

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)

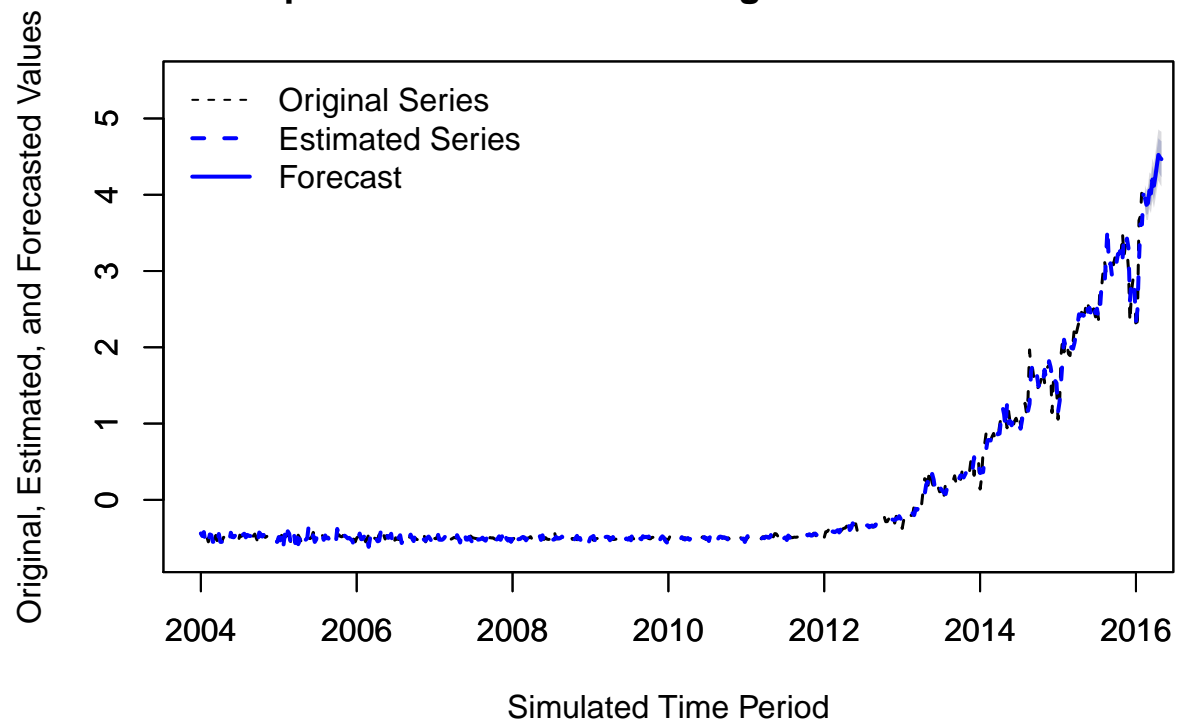
```

Out-of-Sample Forecast



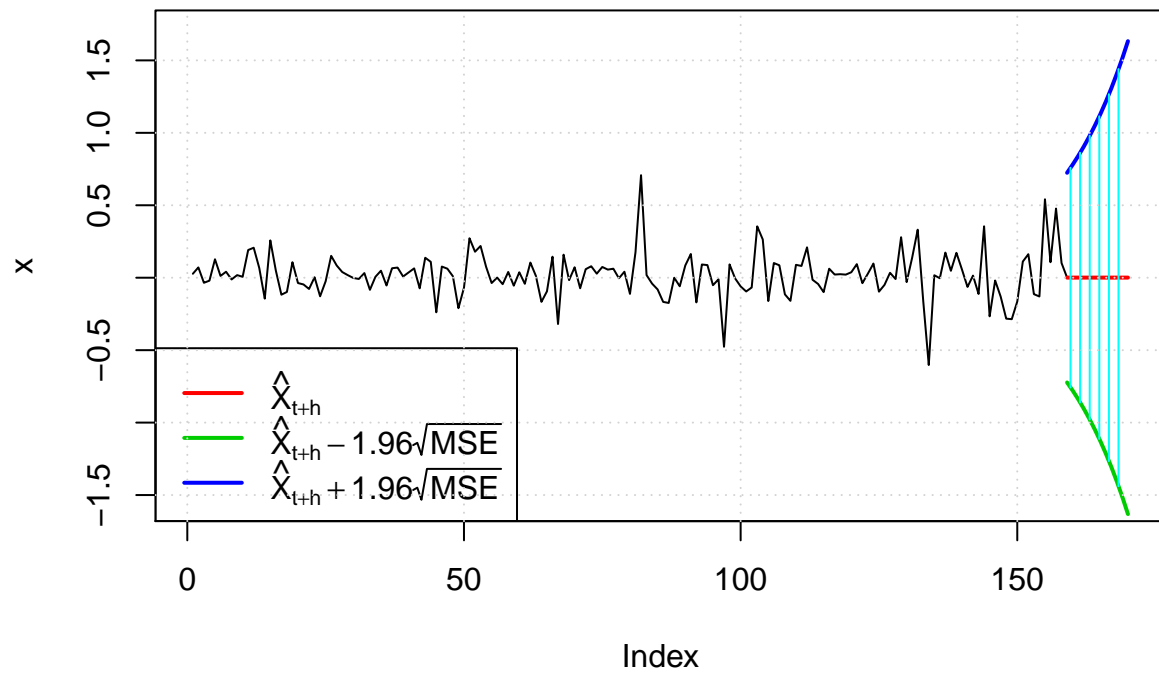
```
### 5. Forecast the model
glob.warm.arima.fcast = forecast.Arima(glob.warm.arima.seas,
  h = 12)
plot.model.forecast(glob.warm.arima.seas, glob.warm.arima.fcast,
  "12", c(2004, 2016), c(-0.7, 5.5))
```

12-Step Ahead Forecast and Original & Estimated Series



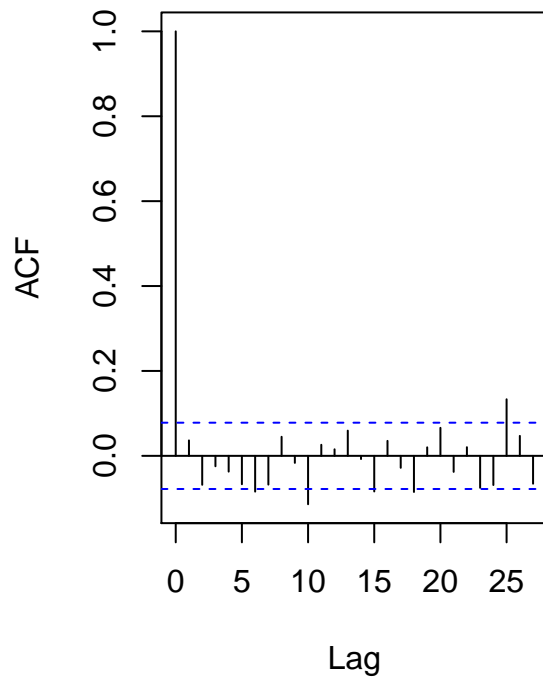
```
### 6. GARCH
glob.warm.garch.fit = garchFit(~garch(1, 1), data = glob.warm.arima.seas.res,
  trace = FALSE)
gw.garch.pred <- predict(glob.warm.garch.fit, n.ahead = 12, plot = TRUE)
```


Prediction with confidence intervals

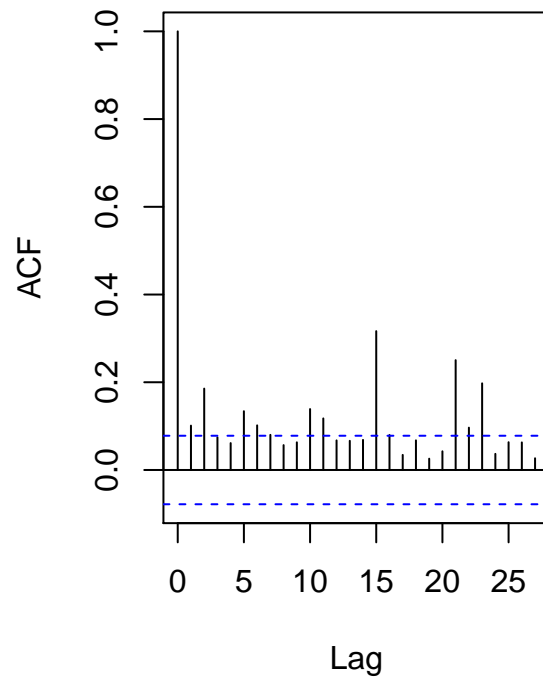


```
glob.warm.garch.res <- glob.warm.garch.fit@residuals
par(mfrow = c(1, 2))
acf(glob.warm.garch.res)
acf(glob.warm.garch.res^2)
```

Series glob.warm.garch.res



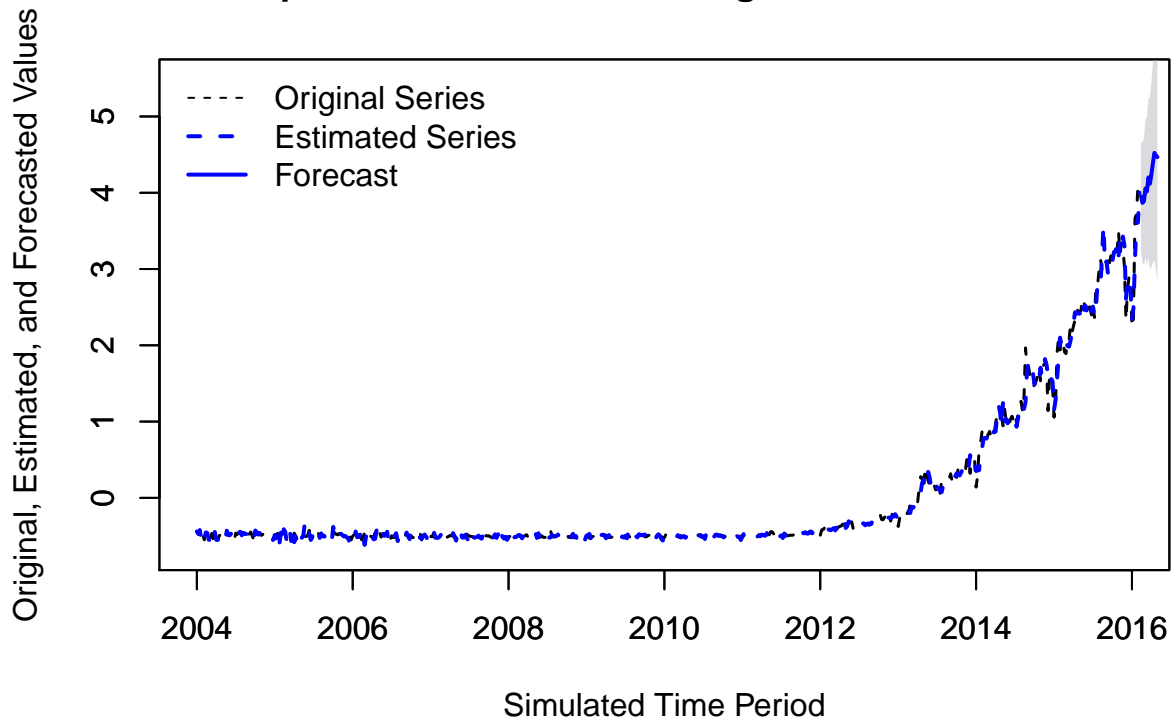
Series glob.warm.garch.res^2



```
### 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



Part 4 (25 points): Forecast Inflation-Adjusted Gas Price

The dataframe contains three variables. Date, Production and Price. It consists of 410 observations of those variables. The Date variable indicates that the data ranges from January 01 1978 to February 01 2012. We now perform some exploratory data analysis of those variables.

1 Variable Production The histogram shows a data distribution of the variable that appears to be multimodal. We can certainly not assume that the underlying data comes from a normal distribution. However, there are no outliers or singularities in the data that would require that we investigate further or that we remove them from the data set.

2 Variable Price The histogram shows a data distribution that appears to be positively skewed. There are no indications from the histogram that the data would follow a normal distribution. However, there are no outliers or singularities in the data that would require that we investigate further or that we remove them from the data set. Our side by side plot of both series shows trends up and down and volatility. The series appear to be non-stationary in the mean.

```
# Load the data
gas.data <- load(file.path("gasOil.Rdata"))

# Sumamry information about the data
str(gasOil)
```

```
## 'data.frame':   410 obs. of  3 variables:
## $ Date       : chr  "1978-01-01" "1978-02-01" "1978-03-01" "1978-04-01" ...
```

```
## $ Production: num 259 235 270 265 274 ...
## $ Price      : num 2.46 2.44 2.43 2.41 2.41 ...
```

```
summary(gasOil)
```

```
##      Date           Production      Price
## Length:410      Min.       :119.4   Min.       :1.329
## Class :character 1st Qu.:173.0     1st Qu.:1.823
## Mode  :character Median    :201.4     Median    :2.096
##                               Mean     :210.0     Mean     :2.391
##                               3rd Qu.:255.8     3rd Qu.:2.909
##                               Max.      :283.2     Max.      :4.432
```

```
cbind(head(gasOil$Date), head(gasOil$Price), head(gasOil$Production),
      tail(gasOil$Date), tail(gasOil$Price), tail(gasOil$Production))
```

```
##      [,1]      [,2]      [,3]      [,4]
## [1,] "1978-01-01" "2.45669201913876" "259.15" "2011-09-01"
## [2,] "1978-02-01" "2.44122034761905" "234.544" "2011-10-01"
## [3,] "1978-03-01" "2.42581832649842" "270.324" "2011-11-01"
## [4,] "1978-04-01" "2.41427695305164" "264.526" "2011-12-01"
## [5,] "1978-05-01" "2.41393090697674" "273.583" "2012-01-01"
## [6,] "1978-06-01" "2.42461854"      "264.974" "2012-02-01"
##      [,5]      [,6]
## [1,] "3.78699964541901" "166.849"
## [2,] "3.61365255477027" "181.493"
## [3,] "3.54091412492241" "179.099"
## [4,] "3.41761395265139" "185.712"
## [5,] "3.52764117334551" "190.358"
## [6,] "3.72698725376175" "180.969"
```

```
# EDA for variable Production
```

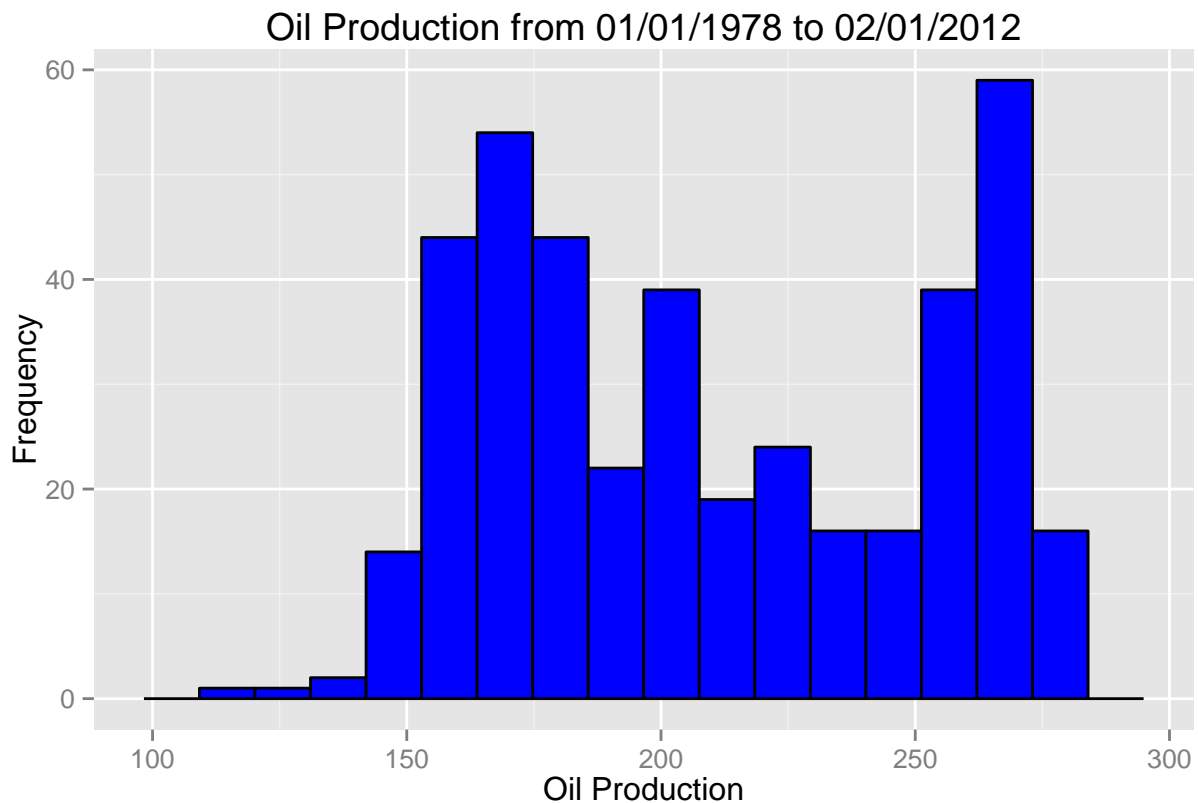
```
print(quantile(gasOil$Production, probs = c(0.01, 0.05, 0.1,
      0.25, 0.5, 0.75, 0.9, 0.95, 0.99, 1)))
```

```
##      1%      5%      10%      25%      50%      75%      90%      95%
## 143.5285 154.5767 159.3485 173.0135 201.4405 255.7722 267.6471 271.1311
##      99%     100%
## 279.2919 283.2480
```

```
# Plot the histogram of at 15 bins
```

```
gasOil.prod.hist <- ggplot(gasOil, aes(Production)) + theme(legend.position = "none") +
  geom_histogram(fill = "Blue", colour = "Black", binwidth = (range(gasOil$Production)[2] -
    range(gasOil$Production)[1])/15) + labs(title = "Oil Production from 01/01/1978 to 02/01/2012",
    x = "Oil Production", y = "Frequency")

plot(gasOil.prod.hist)
```



```
# EDA for variable Price
```

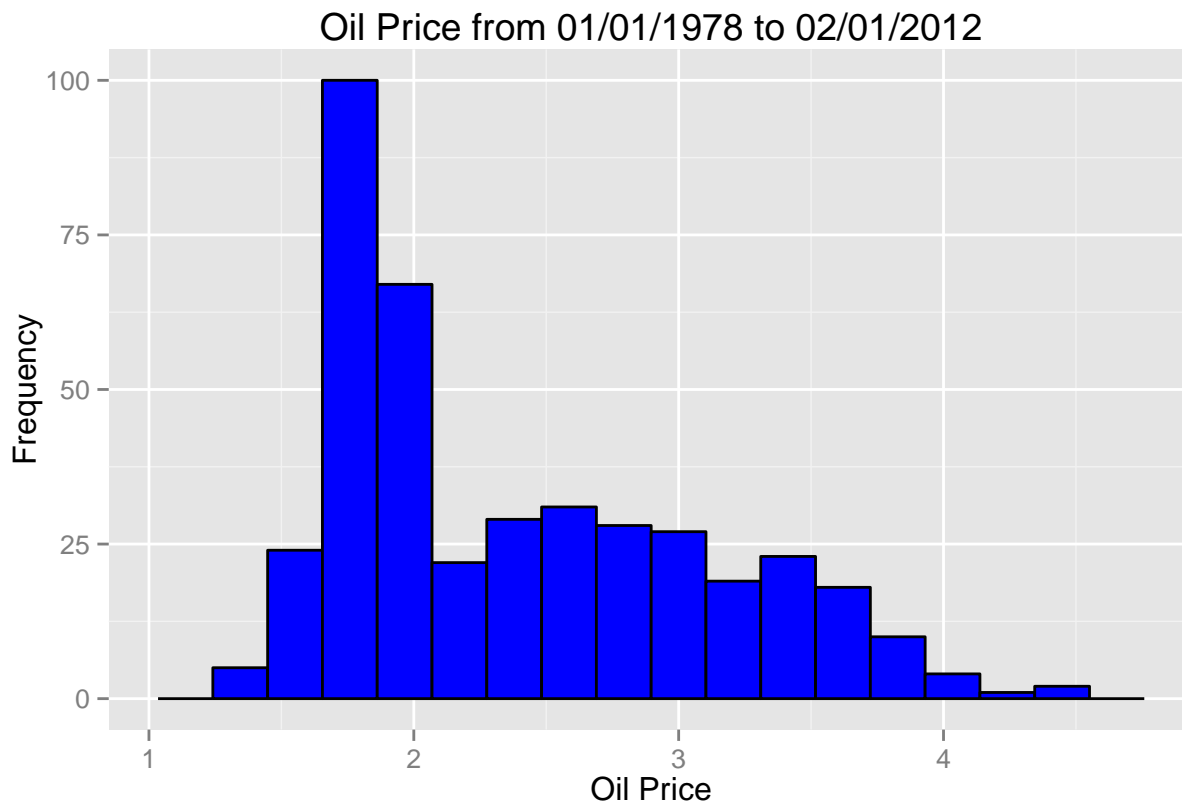
```
print(quantile(gasOil$Price, probs = c(0.01, 0.05, 0.1, 0.25,
    0.5, 0.75, 0.9, 0.95, 0.99, 1)))
```

```
##      1%      5%      10%      25%      50%      75%      90%      95%
## 1.443087 1.619687 1.709716 1.823093 2.096003 2.908782 3.471567 3.671866
##      99%     100%
## 4.100232 4.431625
```

```
# Plot the histogram of at 15 bins
```

```
gasOil.price.hist <- ggplot(gasOil, aes(Price)) + theme(legend.position = "none") +
  geom_histogram(fill = "Blue", colour = "Black", binwidth = (range(gasOil$Price)[2] -
    range(gasOil$Price)[1])/15) + labs(title = "Oil Price from 01/01/1978 to 02/01/2012",
    x = "Oil Price", y = "Frequency")
```

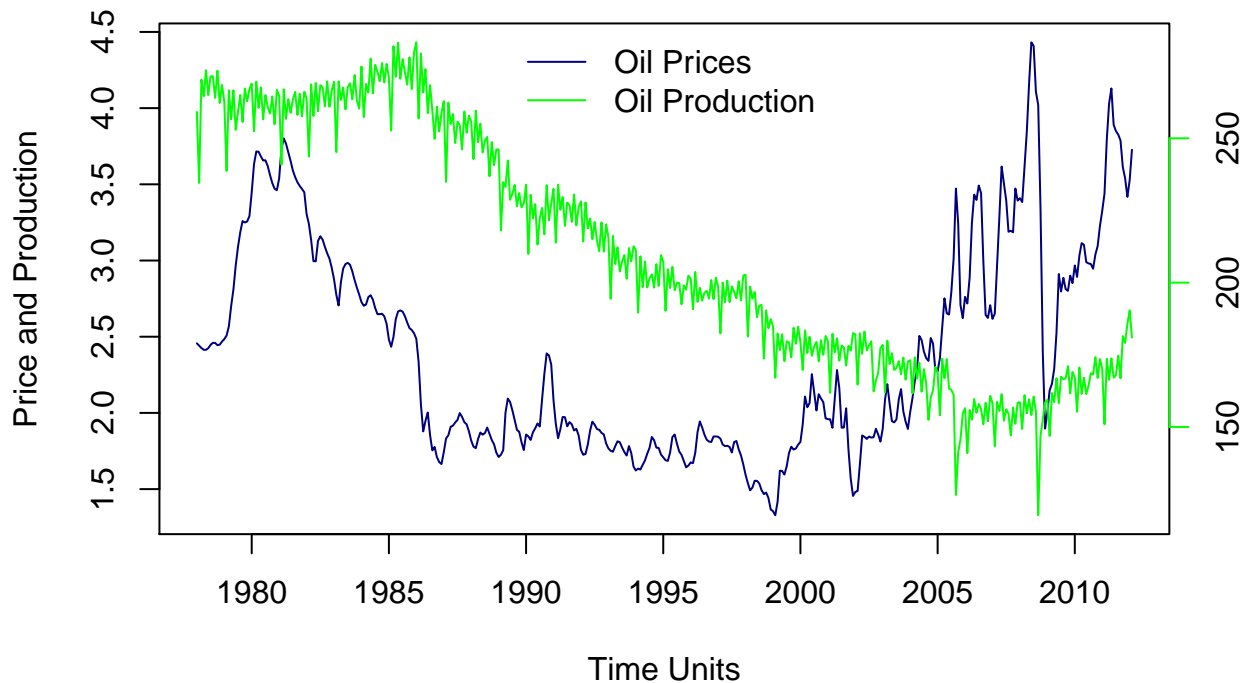
```
plot(gasOil.price.hist)
```



```
# Side by side dynamics of Price and Production time series
# First convert the price data to a time series:
price.ts <- ts(gasOil$Price, start = c(1978, 1), end = c(2012,
  2), frequency = 12)
# Convert the production data to a time series
production.ts <- ts(gasOil$Production, start = c(1978, 1), end = c(2012,
  2), frequency = 12)

# Plot the two time series
par(mfrow = c(1, 1))
plot.ts(price.ts, main = "Oil Prices and Production From 1978 to 2012",
  ylab = "Price and Production", xlab = "Time Units", col = "navy")
par(new = T)
plot.ts(production.ts, ylab = "", xlab = "", col = "green", axes = F)
axis(side = 4, col = "green")
leg.txt <- c("Oil Prices", "Oil Production")
legend("top", legend = leg.txt, lty = 1, col = c("navy", "green"),
  bty = "n", cex = 1)
```

Oil Prices and Production From 1978 to 2012



Task1

We can assume that the AP tested the correlation of the time series of Oil Price and Oil Production. We can replicate the calculation of the reported p-value with a test of the correlation of the variables Price and Production. The test reports a p-value of 0.5752, which is non-significant. The reported 95% confidence interval for the correlation is $[-0.06927648 \ 0.12427029]$. Since the p-value for the test is non-significant, the confidence interval non-surprisingly spans the zero value.

When computing the correlation of two sets of observations of data, we assume that the data is from random samples drawn from the same population with a distribution that has a constant mean and variance. We know that our data is from a time series. We have seen from the side by side plots that it's non-stationary in the mean. Therefore the assumption of constant mean in the calculation of the correlation does not hold and the calculated p-value is likely flawed. Another assumption made with correlations is the assumption of independence of variables in the samples. As stated before, we usually assume that the samples are random draws from a population. For a time series the assumption of independence between the data observations must be rejected. For times series, the observation of x_t is dependent on previous observations of x_{t-1} , x_{t-2}, \dots . That dependency is captured in a joint probability distribution which is unavailable to us, as the time series represents the single instance of the realisation of a stochastic process that we are unable to observe.

We next turn to studying the time series of gas prices.

```
cor.test(gasOil$Price, gasOil$Production)
```

```
##
```

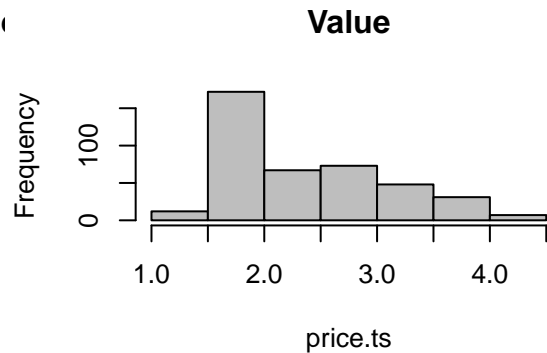
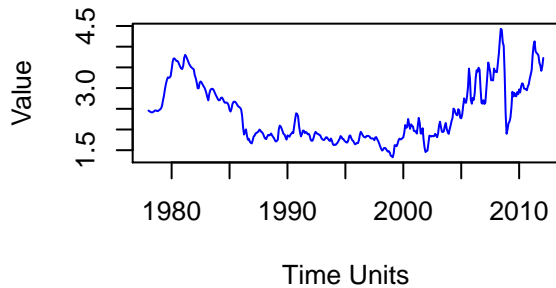
```
## Pearson's product-moment correlation
##
## data: gasOil$Price and gasOil$Production
## t = 0.56088, df = 408, p-value = 0.5752
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
## -0.06927648 0.12427029
## sample estimates:
## cor
## 0.02775705
```

Task 2

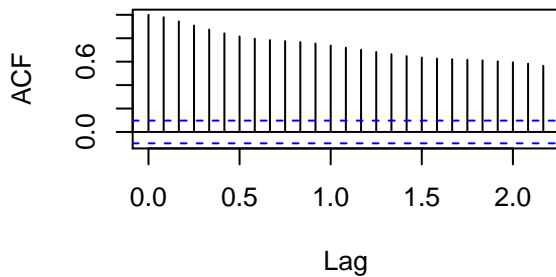
The series exhibits trends up and down and a lot of volatility. It appears to be non-stationary in the mean. We can also see that ACF is gradually descending, indicating a possible ARIMA or SARIMA dynamics. Knowing the series to be that of oil prices, we can speculate that it incorporates seasonality as we'd expect prices to follow the seasons of the year. We would expect yearly seasonality. The PACF shows significant correlations at lags 1 and possibly 2, suggesting that the series might have characteristics of an AR(1) or AR(2) component. To verify our observations, we next study 1 and 2 order differences of the series.

```
# EDA on series
par(mfrow = c(2, 2))
plot.ts(price.ts, main = "Oil Prices From 1978 to 2011 Time Series",
        ylab = "Value", xlab = "Time Units", col = "blue")
hist(price.ts, col = "gray", main = "Value")
acf(price.ts, main = "ACF of Time Series")
pacf(price.ts, main = "PACF of Time Series")
```

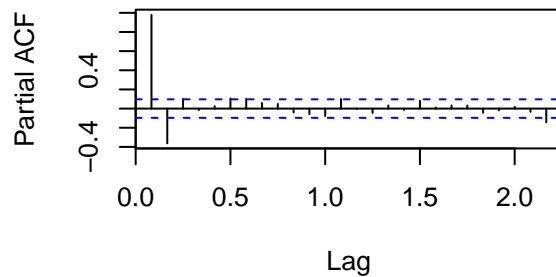

Oil Prices From 1978 to 2011 Time Series



ACF of Time Series



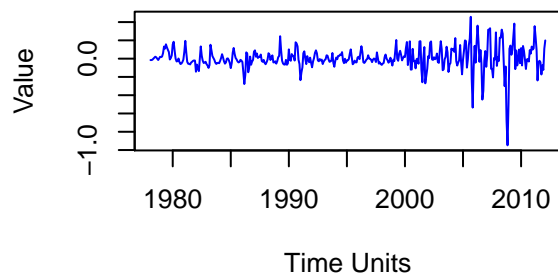
PACF of Time Series



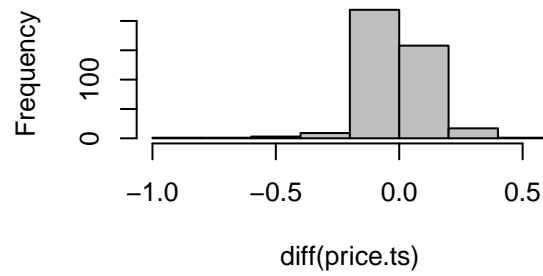
As expected, the lag(1) and lag(2) series are stationary in the mean. They appear to show conditional volatility that will need to be analyzed with an GARCH model. We can also observe patterns that appear to be seasonal patterns of repetition on yearly basis in the series. That observation would support the intuition that weather cycles and corresponding consumption changes may affect gas prices. Because the first difference series is stationary in the mean, we will not need the second order differencing of the series when we study it further. We can also see that the ACF of the first and second order series drop sharply after lag 1 indicating the presence of an MA(1) component in the series. Similarly, the PACF of the first and second difference series have a single significant correlation at lag 1, indicating the possible presence of an AR(1) component in the series. We next perform a systematic search of the best model fit for the series based on the AIC.

```
par(mfrow = c(2, 2))
plot.ts(diff(price.ts), main = "First order difference Oil Prices series",
        ylab = "Value", xlab = "Time Units", col = "blue")
hist(diff(price.ts), col = "gray", main = "Value")
acf(diff(price.ts), main = "ACF of Time Series")
pacf(diff(price.ts), main = "PACF of Time Series")
```

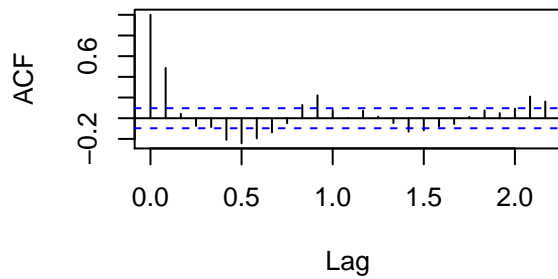
First order difference Oil Prices series



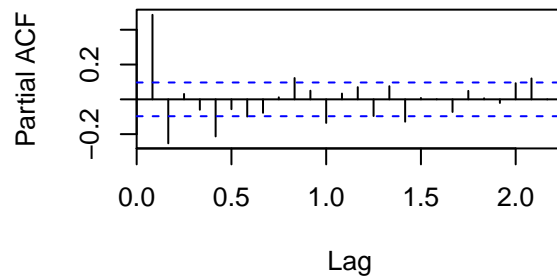
Value



ACF of Time Series

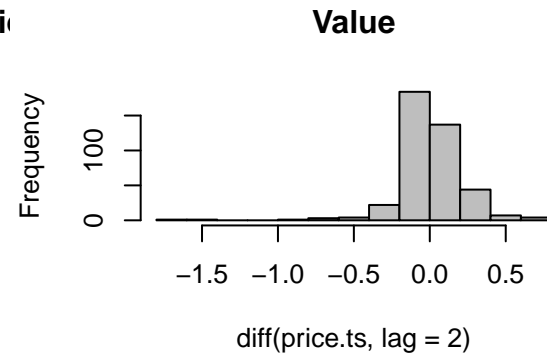
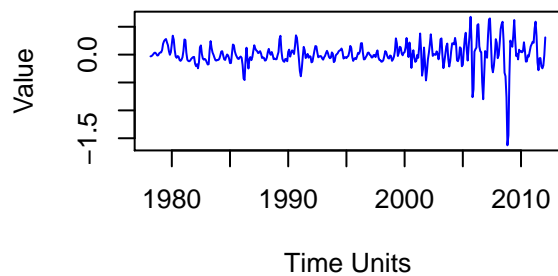


PACF of Time Series

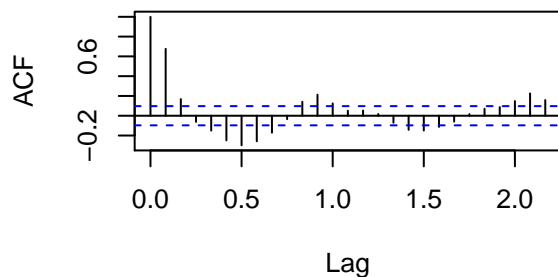


```
par(mfrow = c(2, 2))
plot.ts(diff(price.ts, lag = 2), main = "Second order difference Oil Prices series",
        ylab = "Value", xlab = "Time Units", col = "blue")
hist(diff(price.ts, lag = 2), col = "gray", main = "Value")
acf(diff(price.ts, lag = 2), main = "ACF of Time Series")
pacf(diff(price.ts, lag = 2), main = "PACF of Time Series")
```

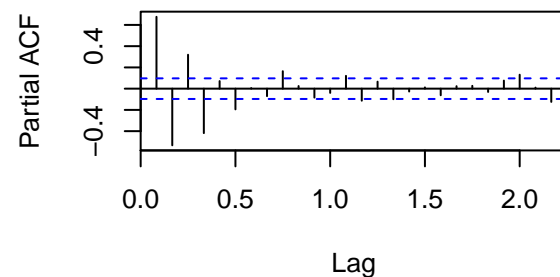
Second order difference Oil Prices series



ACF of Time Series



PACF of Time Series



The best SARIMA model fitted is a (p,d,q,P,D,Q) of $(0, 1, 2, 1, 0, 3)$ with an AIC of -672.50. We do take note of the fact that immediately following that series, is a SARIMA of parameters $(0, 1, 1, 1, 0, 3)$ with an AIC of -672.19. With parsimony in mind, we select that model to fit our data and proceed to assess the in-sample fit of that model. It matches our earlier observations of a stationary first difference series, and of the possible presence of AR(1) and MA(1) components in the time series, along with that of a seasonal component. An inspection of the confidence intervals of the parameters of this model indicate that the parameter for the second MA component of the seasonal model is not significant. But the third parameter of the same seasonal component is significant. We next perform in-sample fit using the fitted series to assess our fitted model.

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

The fitted series models the original series very well and the model selection seems appropriate based on in-sample fit. To further assess the fitted series, We next perform 48 steps backtesting.

```
# Fit the selected ARIMA model to the time series data.
price.fit <- Arima(price.ts, order = c(0, 1, 1), seasonal = list(order = c(1,
  0, 3), period = 12), method = "CSS-ML")
price.res <- price.fit$resid
t(confint(price.fit))
```

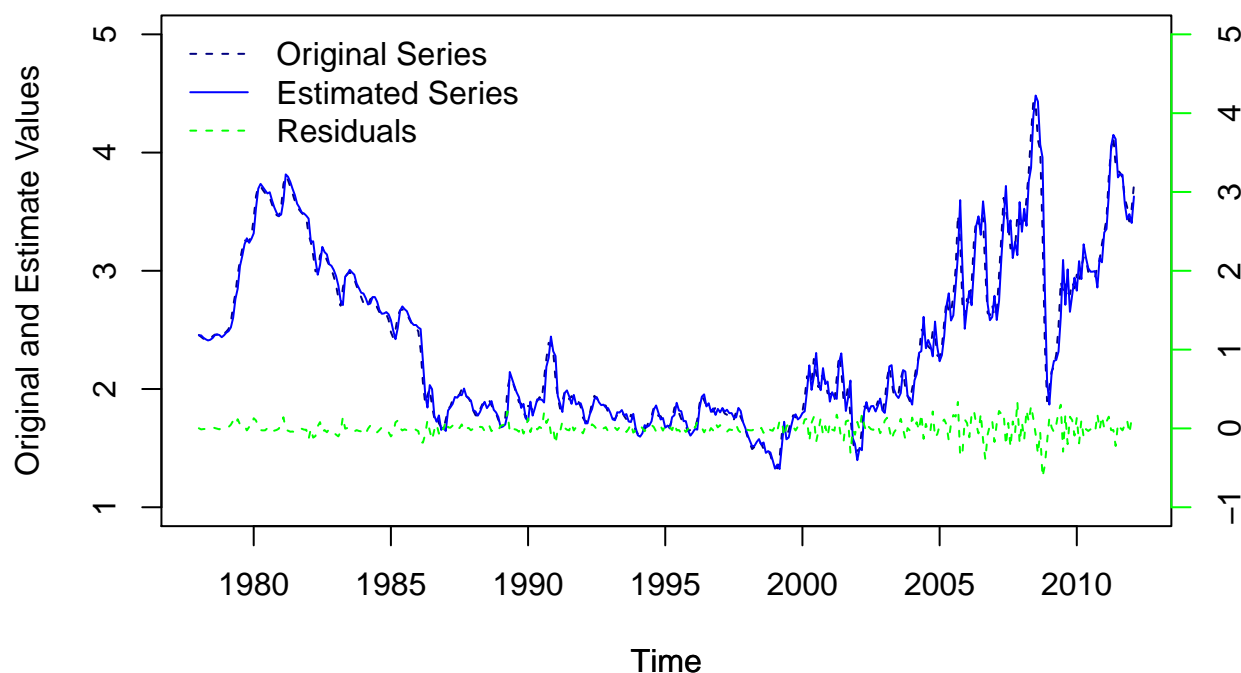
```
##          ma1          sar1          sma1          sma2          sma3
## 2.5 %  0.4570524 0.855634 -1.1133820 -0.1289343 0.1123691
## 97.5 % 0.5999814 0.992224 -0.8704157  0.1411505 0.3294568
```

```
quantile(as.numeric(price.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%
## -0.616577523 -0.323314392 -0.164782375 -0.083391460 -0.030337106
##           50%           75%           90%           95%           99%
## -0.003213021  0.044032156  0.114419059  0.163404708  0.231930102
##           100%
##  0.340313336
```

```
# We now perform in-sample fit using the fitted series to
# assess our fitted model.
par(mfrow = c(1, 1))
plot.ts(price.ts, col = "navy", lty = 2, main = "Original vs a SAMIMA(0, 1, 1, 1, 0, 3) Estimated Series",
  ylab = "Original and Estimate Values", ylim = c(1, 5))
par(new = T)
plot(fitted(price.fit), col = "blue", axes = F, ylab = "", ylim = c(1,
  5))
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(price.res, axes = F, xlab = "", ylab = "", col = "green",
  ylim = c(-1, 5), pch = 1, lty = 2)
axis(side = 4, col = "green")
mtext("Residuals", side = 4, line = 2, col = "green")
```

Original vs a SAMIMA(0, 1, 1, 1, 0, 3) Estimated Series with Residual



Our out of sample forecast appears to be reasonable. The 80 and 95 % confidence intervals of the forecast partially include the actual values of the series. We had previously observed that the variance of the residuals seemed to show some volatility that we know is not modeled by the fitted point series. Therefore, we next turn our eyes to the residuals of the fitted series and to the analysis of the dynamics of its variance.

```
price.fit.back <- Arima(price.ts[1:(length(price.ts) - 48)],
  order = c(0, 1, 1), seasonal = list(order = c(1, 0, 3), period = 12),
  method = "CSS-ML")
summary(price.fit.back)
```

```
## Series: price.ts[1:(length(price.ts) - 48)]
## ARIMA(0,1,1)(1,0,3)[12]
##
## Coefficients:
##          ma1      sar1      sma1      sma2      sma3
##          0.5370  0.8933  -0.9756  0.0840  0.1450
## s.e.    0.0408  0.0564   0.0799  0.0794  0.0673
##
## sigma^2 estimated as 0.00724:  log likelihood=374.29
## AIC=-736.58  AICc=-736.34  BIC=-713.25
##
## Training set error measures:
##              ME      RMSE      MAE      MPE      MAPE
## Training set 0.001656362 0.08496843 0.05633002 0.06153449 2.437364
##              MASE      ACF1
## Training set 0.8325864 0.01216812
```

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

```
## [1] 362
```

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

```
## [1] 362
```

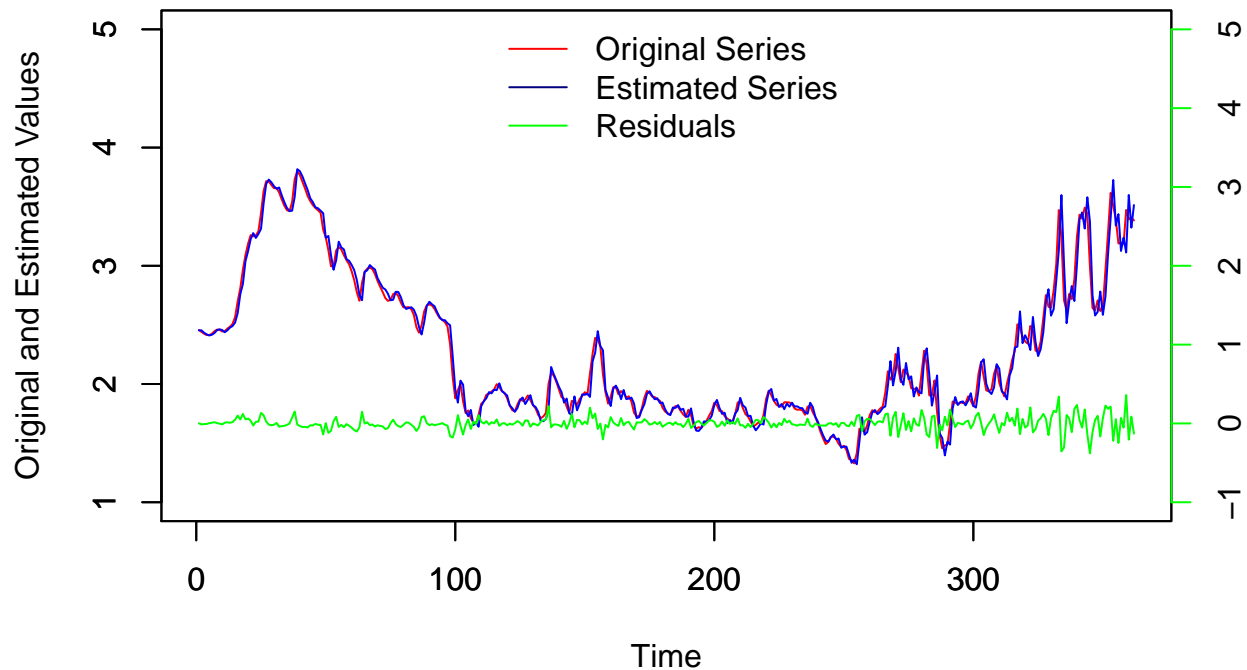
```
df = cbind(price.ts[1:(length(price.ts) - 48)], fitted(price.fit.back),  
            price.fit.back$resid)  
colnames(df) = c("orig_series", "fitted_vals", "resid")  
head(df)
```

```
##      orig_series fitted_vals      resid  
## [1,]    2.456692    2.454235  0.002456690  
## [2,]    2.441220    2.454073 -0.012852734  
## [3,]    2.425818    2.433999 -0.008180921  
## [4,]    2.414277    2.420843 -0.006566487  
## [5,]    2.413931    2.410769  0.003161880  
## [6,]    2.424619    2.416242  0.008377038
```

```
# 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", ylim = c(1, 5))  
par(new = T)  
plot.ts(df[, "fitted_vals"], col = "blue", axes = T, xlab = "",  
        ylab = "", ylim = c(1, 5))  
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",  
        ylim = c(-1, 5), pch = 1)  
axis(side = 4, col = "green")  
mtext("Residuals", side = 4, line = 2, col = "green")
```

Original vs a AR(1) Estimated Series with Residuals

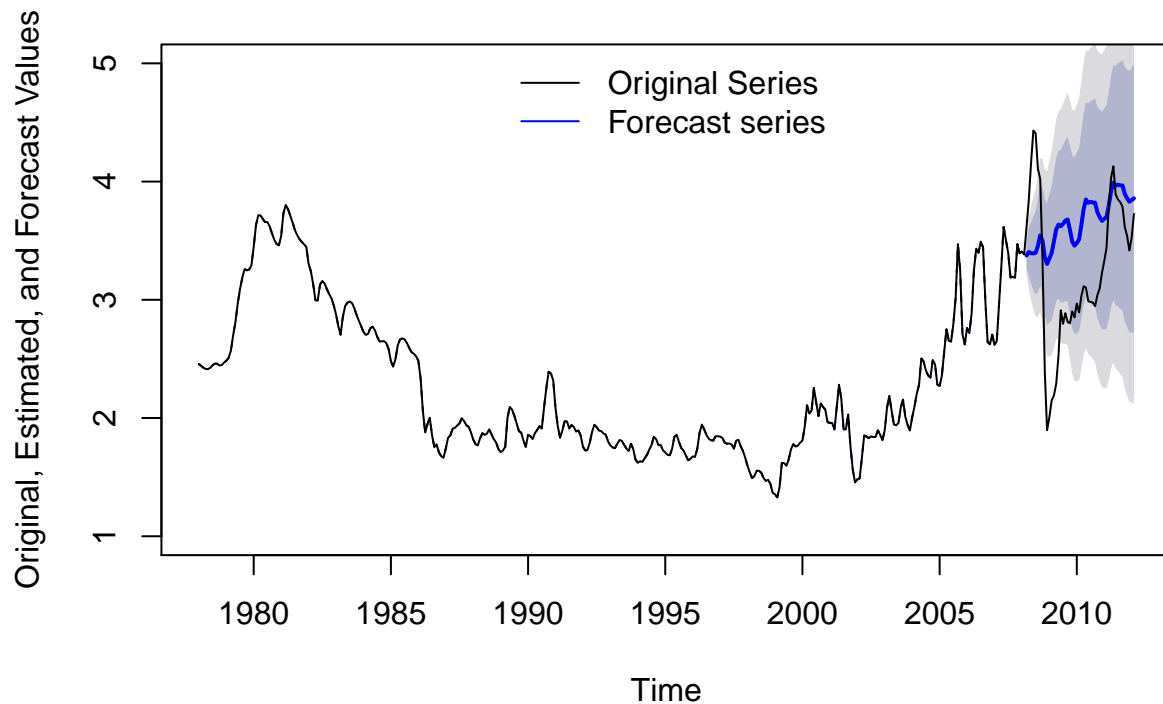


```
# Step 2: Out of sample forecast
price.fit.back.fcast <- forecast.Arima(price.fit.back, h = 48)
length(price.fit.back.fcast$mean)
```

```
## [1] 48
```

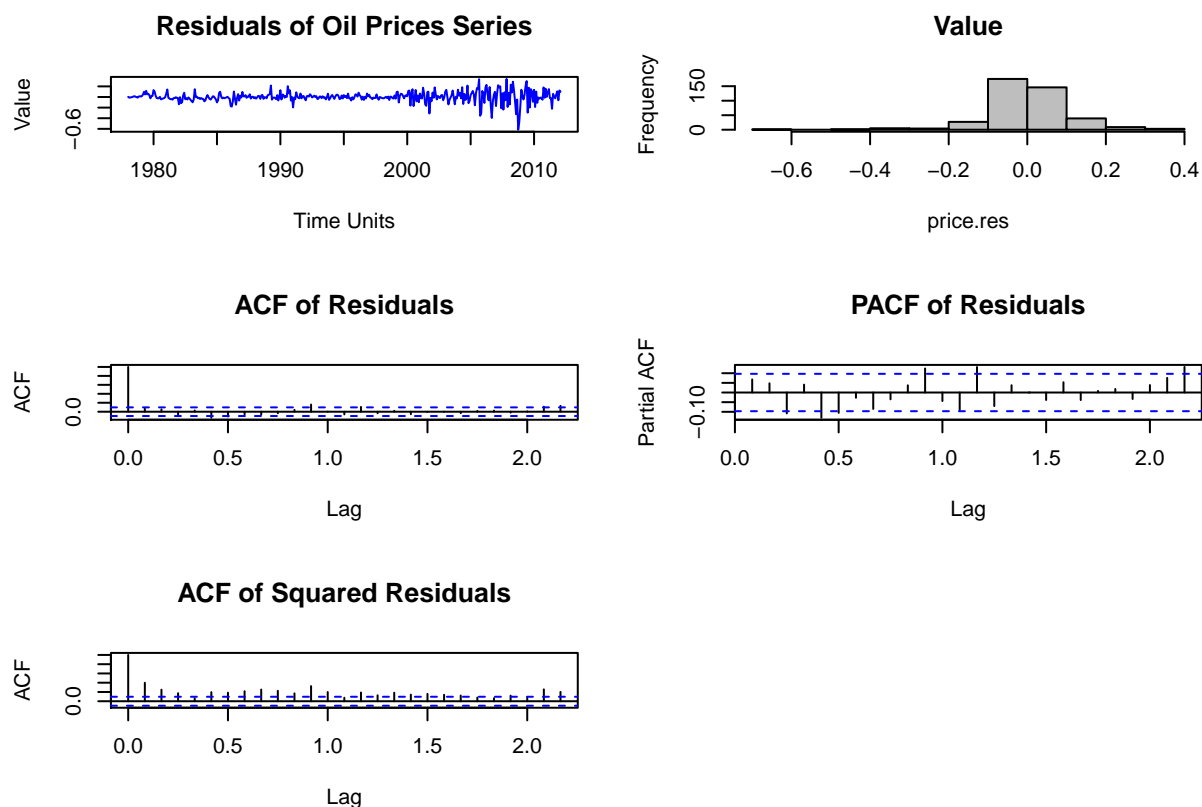
```
par(mfrow = c(1, 1))
plot(price.fit.back.fcast, lty = 2, col = "navy", main = "Out-of-Sample Forecast",
     ylab = "Original, Estimated, and Forecast Values", ylim = c(1,
     5), , axes = F)
par(new = T)
plot.ts(price.ts, axes = T, lty = 1, ylim = c(1, 5), ylab = "")
leg.txt <- c("Original Series", "Forecast series")
legend("top", legend = leg.txt, lty = 1, col = c("black", "blue"),
     bty = "n", cex = 1)
```

Out-of-Sample Forecast



The ACF and PACF of the residual series resemble those of a white noise series. But we observe from the squared residuals 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 behavior. Therefore, we will model its residuals using GARCH. With this in mind, we proceed to an initial 48 steps ahead forecast of the gas price series.

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

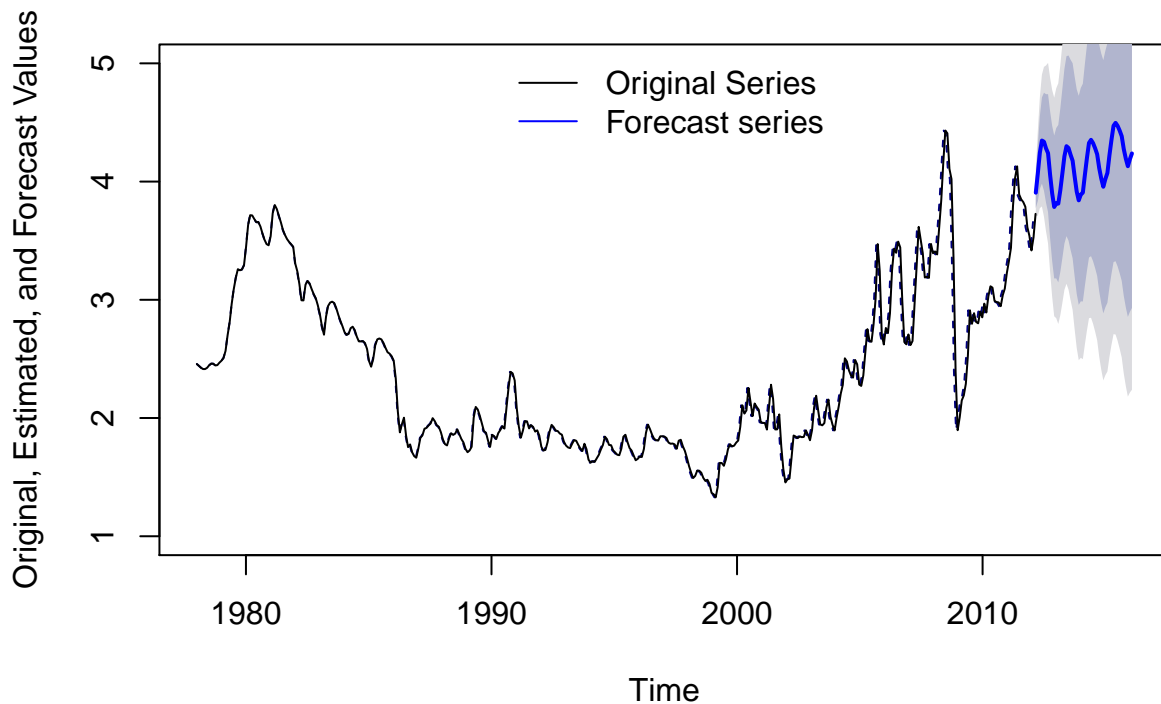
The point estimates of our forecasts look reasonable. We are aware that the model used for this point forecast assumes a stationary mean and variance. We can assess a stationary mean, but not a variance. The consequence of the non-stationary variance is that the confidence intervals around our estimates are biased. Having acknowledged the confidence interval problem on the prediction caused by the non-stationary variance of the financial time series, we want to use our fitted GARCH model to predict the mean and variance of the residuals of the point series.

```
# 2012-2016 steps ahead sample forecast
price.fit.ahead.fcast <- forecast.Arima(price.fit, h = 48)
length(price.fit.ahead.fcast$mean)
```

```
## [1] 48
```

```
par(mfrow = c(1, 1))
plot(price.fit.ahead.fcast, lty = 2, col = "navy", main = "Out-of-Sample Forecast",
     ylab = "Original, Estimated, and Forecast Values", ylim = c(1,
     5))
par(new = T)
plot.ts(price.ts, axes = F, lty = 1, ylim = c(1, 5), xlim = c(1978,
     2016), ylab = "")
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



We observe from the ACF of the residuals of the GARCH fitted series that they have characteristics 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 volatility affects the confidence intervals of the estimates of our SARIMA model as previously observed with the backtesting. Using our fitted GARCH model, we can now better predict the variance of the point estimates from 2012 to 2016.

```
# Fit a GARCH model to the residuals of the fitted time
# series
price.garch <- garch(price.res, order = c(1, 1), trace = F)
t(confint(price.garch))
```

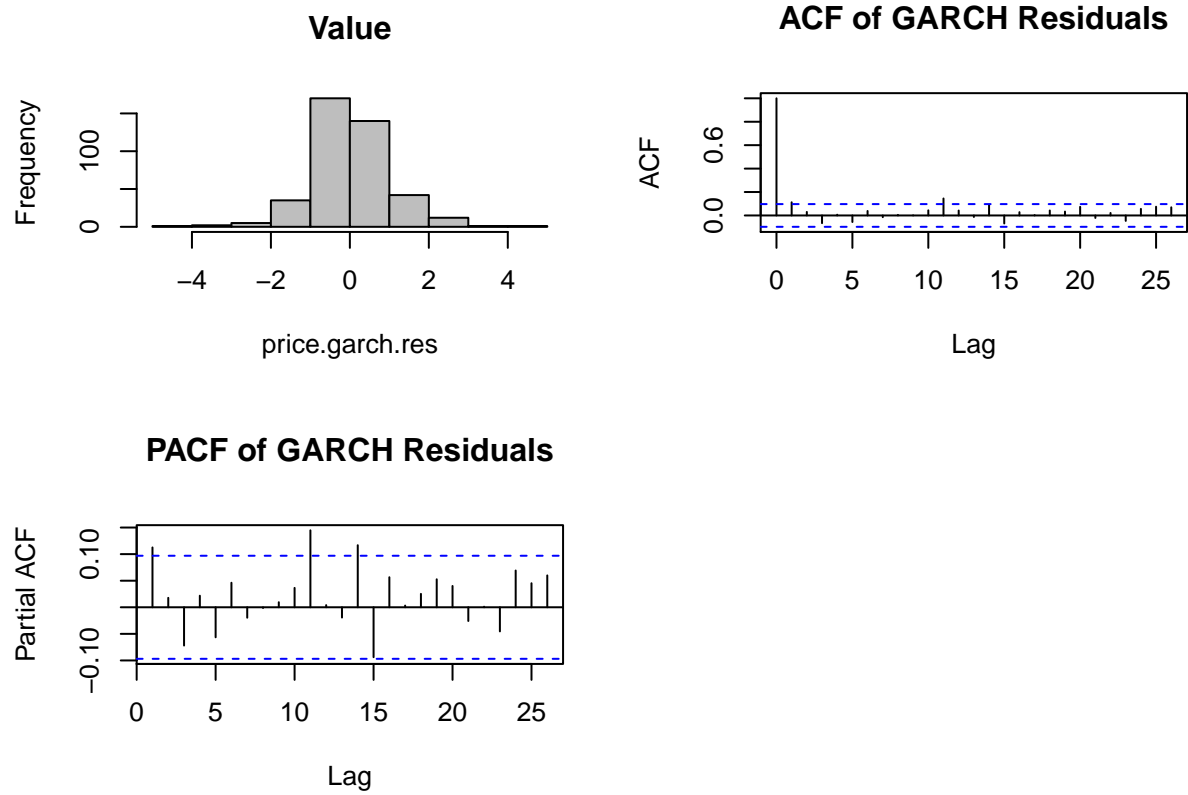
```
##              a0              a1              b1
## 2.5 %  9.844544e-05  0.1557701  0.7131228
## 97.5 %  3.329228e-04  0.3048951  0.8252643
```

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

# Perform EDA on residuals of fit
par(mfrow = c(2, 2))
hist(price.garch.res, col = "gray", main = "Value")
acf(price.garch.res, na.action = na.pass, main = "ACF of GARCH Residuals")
pacf(price.garch.res, na.action = na.pass, main = "PACF of GARCH Residuals")

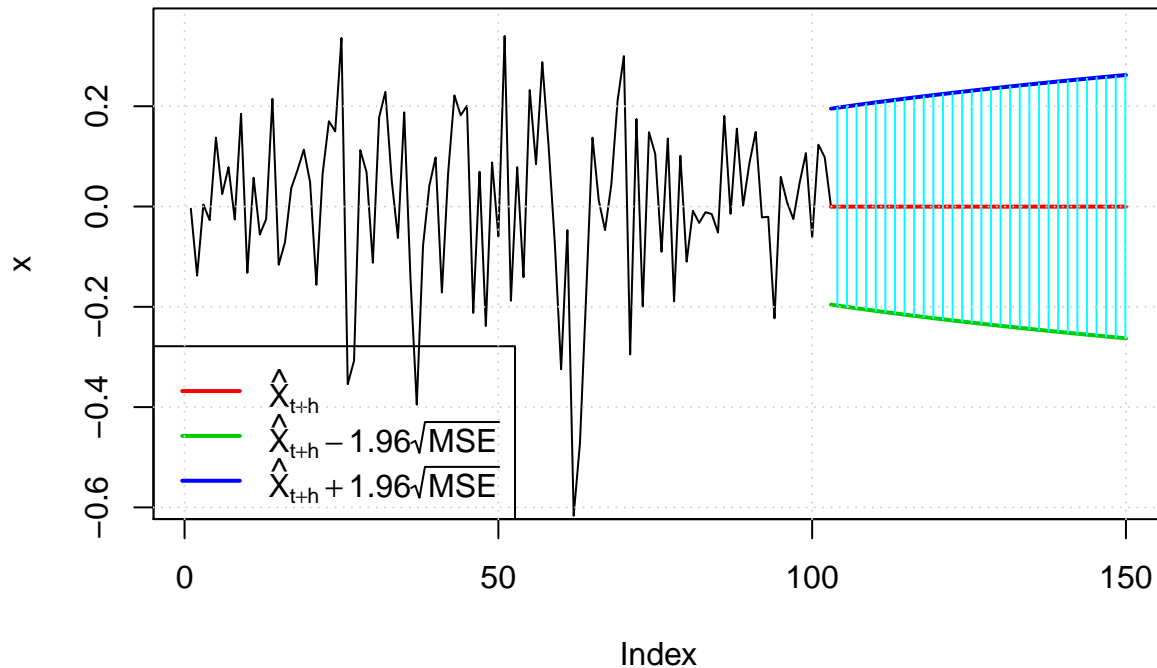
# Predict the residuals variance 4 years ahead
```

```
price.garch.fit <- garchFit(~garch(1, 1), data = price.res, trace = FALSE)
par(mfrow = c(1, 1))
```



```
price.garch.pred <- predict(price.garch.fit, n.ahead = 48, plot = TRUE)
```

Prediction with confidence intervals



Having better estimates of the variance with GARCH, we now update our estimated SARIMA model's confidence interval with those.

```
# Recompute the lower and upper 95% confidence intervals
# based on GARCH variance
price.fit.ahead.fcast$lower[, 1] <- price.fit.ahead.fcast$mean
price.fit.ahead.fcast$upper[, 1] <- price.fit.ahead.fcast$mean

price.fit.ahead.fcast$lower[, 2] <- -(((price.fit.ahead.fcast$mean -
  price.fit.ahead.fcast$lower[, 2])/(1.96 * sqrt(price.fit.ahead.fcast$model$sigma2))) *
  price.garch.pred$standardDeviation) + price.fit.ahead.fcast$mean
price.fit.ahead.fcast$lower[, 2]

## [1] 3.806050 3.887047 4.002978 4.057141 4.004508 3.909445 3.833124
## [8] 3.615745 3.426595 3.279992 3.281559 3.248886 3.337532 3.453060
## [15] 3.574412 3.635127 3.595281 3.515403 3.441744 3.283567 3.134890
## [22] 3.038464 3.064686 3.059834 3.188892 3.311426 3.419429 3.426570
## [29] 3.379268 3.316536 3.246087 3.109404 3.002920 2.914366 2.960236
## [36] 2.993406 3.120855 3.228182 3.322145 3.322978 3.273564 3.209945
## [43] 3.139246 3.007399 2.903499 2.816209 2.853158 2.878413

price.fit.ahead.fcast$upper[, 2] <- (((price.fit.ahead.fcast$upper[,
  2] - price.fit.ahead.fcast$mean)/(1.96 * sqrt(price.fit.ahead.fcast$model$sigma2))) *
  price.garch.pred$standardDeviation) + price.fit.ahead.fcast$mean
price.fit.ahead.fcast$upper[, 2]
```

```
## [1] 4.005486 4.254692 4.486695 4.636981 4.669221 4.651636 4.647502
## [8] 4.498307 4.374195 4.290082 4.352028 4.377956 4.519524 4.684832
## [15] 4.855118 4.964009 4.971655 4.938648 4.911292 4.798901 4.695531
## [22] 4.643969 4.714645 4.753864 4.924048 5.086123 5.233437 5.279671
## [29] 5.271252 5.247203 5.215247 5.116875 5.048526 4.997939 5.081614
## [36] 5.152431 5.324431 5.480062 5.621993 5.670475 5.668407 5.651847
## [43] 5.627930 5.542603 5.484970 5.443704 5.526445 5.597268
```

We can now plot our 2012 to 2016 point estimates with the proper 95% confidence interval.

```
# 2012-2016 steps ahead sample forecast
par(mfrow = c(1, 1))
plot(price.fit.ahead.fcast, lty = 2, col = "navy", main = "Out-of-Sample Forecast",
     ylab = "Original, Estimated, and Forecast Values", ylim = c(1,
     5))
par(new = T)
plot.ts(price.ts, axes = F, lty = 1, ylim = c(1, 5), xlim = c(1978,
     2016), ylab = "")
leg.txt <- c("Original Series", "Forecast series")
legend("top", legend = leg.txt, lty = 1, col = c("black", "blue"),
     bty = "n", cex = 1)
```

