W271-2 – Spring 2016 – **Lab 3**

Juanjo Carin, Kevin Davis, Ashley Levato, Minghu Song

April 22, 2016

Contents

Instructions 1

Part 1 **2**

Modeling House Values 2

Part 2 **3**

Modeling and Forecasting a Real-World Macroeconomic / Financial time series 3

Part 3 **4**

Forecast the Web Search Activity for global Warming 4

Part 4 **8**

Forecast Inflation-Adjusted Gas Price 8

1st task 8

2nd task 18

Instructions

- Thoroughly analyze the given dataset or data series. Detect any anomalies in each of the variables. Examine if any of the variables that may appear to be top- or bottom-coded.
- Your report needs to include a comprehensive graphical analysis
- Your analysis needs to be accompanied by detailed narrative. Just printing a bunch of graphs and econometric results will likely receive a very low score.
- Your analysis needs to show that your models are valid (in statistical sense).
- Your rationale of using certian metrics to choose models need to be provided. Explain the validity / pros / cons of the metric you use to choose your "best" model.
- Your rationale of any decisions made in your modeling needs to be explained and supported with empirical evidence.
- All the steps to arrive at your final model need to be shown and explained clearly.
- All of the assumptions of your final model need to be thoroughly tested and explained and shown to be valid. Don't just write something like, "the plot looks reasonable", or "the plot looks good", as different people interpret vague terms like "reasonable" or "good" differently.

Part 1

Modeling House Values

In Part 1, you will use the data set `houseValue.csv` to build a linear regression model, which includes the possible use of the instrumental variable approach, to answer a set of questions interested by a philanthropist group. You will also need to test hypotheses using these questions.

The philanthropist group hires a think tank to examine the relationship between the house values and neighborhood characteristics. For instance, they are interested in the extent to which houses in neighbhorhood with desirable features command higher values. They are specifically interested in environmental features, such as proximity to water body (i.e. lake, river, or ocean) or air quality of a region.

The think tank has collected information from tens of thousands of neighborhoods throughout the United States. They hire your group as contractors, and you are given a small sample and selected variables of the original data set collected to conduct an initial, proof-of-concept analysis. Many variables, in their original form or transfomed forms, that can explain the house values are included in the dataset. Analyze each of these variables as well as different combinations of them very carefully and use them (or a subset of them), in its original or transformed version, to build a linear regression model and test hypotheses to address the questions. Also address potential (statistical) issues that may be casued by omitted variables.

Part 2

Modeling and Forecasting a Real-World Macroeconomic / Financial time series

Build a time-series model for the series in `lab3_series02.csv`, which is extracted from a real-world macroeconomic/financial time series, and use it to perform a 36-step ahead forecast. The periodicity of the series is purposely not provided. Possible models include AR, MA, ARMA, ARIMA, Seasonal ARIMA, GARCH, ARIMA-GARCH, or Seasonal ARIMA-GARCH models.

Part 3

Forecast the Web Search Activity for global Warming

Imagine that you group is part of a data science team in an apparel company. One of its recent products is Global-Warming T-shirts. The marketing director expects that the demand for the t-shirts tends to increase when global warming issues are reported in the news. As such, the director asks your group to forecast the level of interest in global warming in the news. The dataset given to your group captures the relative web search activity for the phrase, "global warming" over time. For the purpose of this exercise, ignore the units reported in the data as they are unimportant and irrelevant. Your task is to produce the weekly forecast for the *next 3 months* for the relative web search activity for global warming. For the purpose of this exercise, treat it as a *12-step ahead forecast*.

The dataset for this exercise is provided in `globalWarming.csv`. Use only models and techniques covered in the course (up to lecture 13). Note that one of the modeling issues you may have to consider is whether or not to use the entire series provided in the data set. Your choice will have to be clearly explained and supported with empirical evidence. As in other parts of the lab, the general instructions in the *Instruction Section* apply.

We start loading the data and inspecting (and transforming[1]) the resulting dataframe.

```
GW <- read.csv('globalWarming.csv', header = TRUE)
rbind(head(GW,4 ), tail(GW, 4))
```

```
##        Date data.science
## 1    1/4/04       -0.440
## 2   1/11/04       -0.474
## 3   1/18/04       -0.423
## 4   1/25/04       -0.551
## 627  1/3/16        3.662
## 628 1/10/16        3.721
## 629 1/17/16        4.087
## 630 1/24/16        4.104
```

```
GW$Date <- as.Date(as.character(GW$Date), '%m/%d/%y')
# Day of week of 1st observation
as.character(wday(GW$Date[1], label = TRUE, abbr = FALSE))
```

```
## [1] "Sunday"
```

```
# Check that all observations correspond to same day of the week
all(wday(GW$Date, label = TRUE) == wday(GW$Date[1], label = TRUE))
```

```
## [1] TRUE
```

```
# Check that all weeks between start and end dates appear in the dataset
identical(GW$Date, seq(min(GW$Date), max(GW$Date), by=7))
```

```
## [1] TRUE
```

[1] Converting the dates from factors to dates, shortening the names of the dataframe and variables, etc.

```
names(GW)[2] <- "DS"
summary(GW)
```

```
##       Date                  DS
##  Min.   :2004-01-04   Min.   :-0.551000
##  1st Qu.:2007-01-08   1st Qu.:-0.506000
##  Median :2010-01-13   Median :-0.485000
##  Mean   :2010-01-13   Mean   : 0.000038
##  3rd Qu.:2013-01-18   3rd Qu.:-0.200000
##  Max.   :2016-01-24   Max.   : 4.104000
```

```
round(stat.desc(as.data.frame(GW$DS), desc = TRUE, norm = TRUE), 2)
```

```
##                  GW$DS
## nbr.val         630.00
## nbr.null          0.00
## nbr.na            0.00
## min              -0.55
## max               4.10
## range             4.66
## sum               0.02
## median           -0.48
## mean              0.00
## SE.mean           0.04
## CI.mean.0.95      0.08
## var               1.00
## std.dev           1.00
## coef.var      26249.94
## skewness          2.12
## skew.2SE         10.88
## kurtosis          3.47
## kurt.2SE          8.93
## normtest.W        0.59
## normtest.p        0.00
```

It contains 630 observations of a numeric variables (`data.science`, shortened to `DS`), with no missing values, from

The dataset contains 630 observations of a numeric variable (`data.science`, which we shortened to `DS`), from 2004-01-04 to 2016-01-24 (i.e., approximately 12.1 years). All dates correspond to Sundays (so we have weekly observations), and there are no missing values for any Sunday, and all Sundays between the start and end date are available in the dataset. The numeric variable must have been standardized (i.e., rescaled by subtracting its mean and dividing by its standard deviation), and that's the possible reason it has zero mean and unit variance. In any case, it does not resemble a normal distribution at all, as it is very right-skewed.

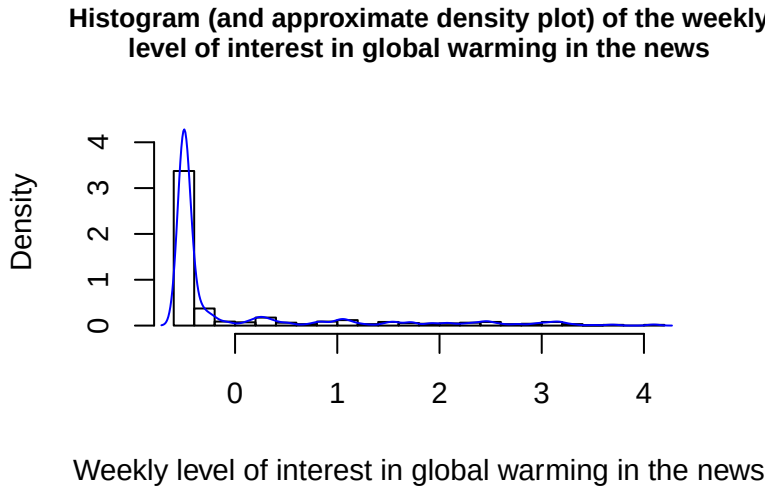

Figure 1: Histogram (and approximate density plot) of the weekly level of interest in global warming in the news

But the density distribution of a time series tells us nothing about its dynamics: we have to look at the time series plot to know that the value of the (standardized) variable was almost flat until 2012, when it started growing almost linearly (with some shocks up and down, the most prominent ones a fall at the end of 2015 and a peak right after that).

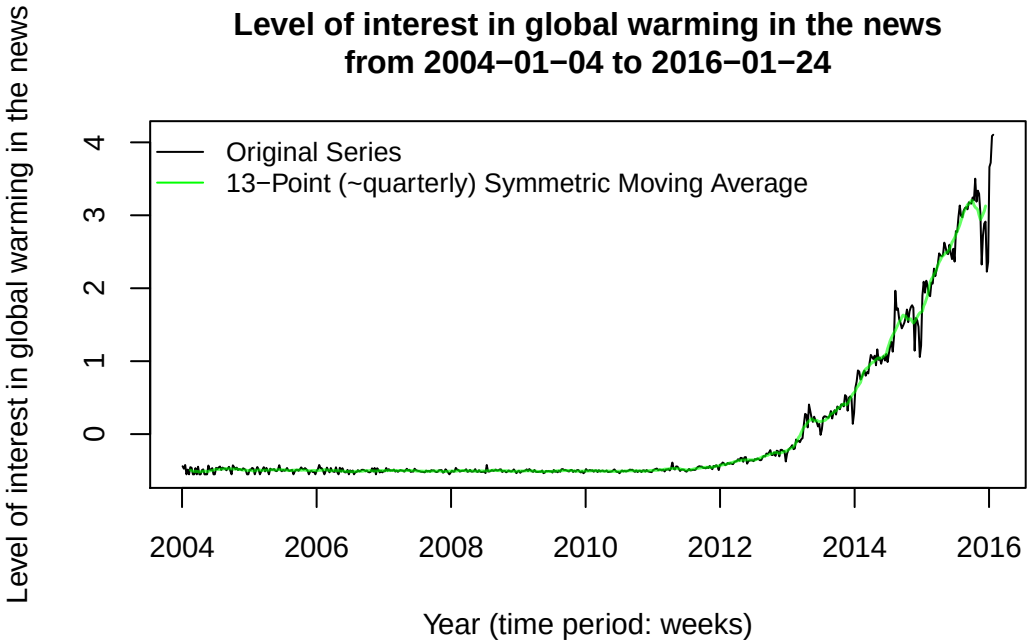

Figure 2: Level of interest in global warming in the news from 2004 to 2015

```
GW.whole <- GW
GW.last <- GW[which(year(GW$Date) >= 2013), ]
arima.whole.fit <- auto.arima(GW.whole$DS, seasonal = TRUE)
arima.last.fit <- auto.arima(GW.last$DS, seasonal = TRUE)
GW.whole.train <- GW.whole[1:(dim(GW.whole)[1]-16), ]
GW.last.train <- GW.last[1:(dim(GW.last)[1]-16), ]
GW.test <- GW[(dim(GW)[1]-16):dim(GW)[1], ]
```

```
arima.whole.oos.fit <- Arima(GW.whole.train$DS,
                             order = arima.whole.fit$arma[c(1, 2, 6)],
                             seas = list(order = arima.whole.fit$arma[c(3, 4,
                                                                        7)],
                                         freq = arima.whole.fit$arma[5]))
arima.whole.oos.fit.fcast <- forecast.Arima(arima.whole.oos.fit, h = 16)
arima.last.oos.fit <- Arima(GW.last.train$DS,
                            order = arima.last.fit$arma[c(1, 2, 6)],
                            seas = list(order = arima.last.fit$arma[c(3, 4,
                                                                      7)],
                                        freq = arima.whole.fit$arma[5]))
arima.last.oos.fit.fcast <- forecast.Arima(arima.last.oos.fit, h = 16)
```

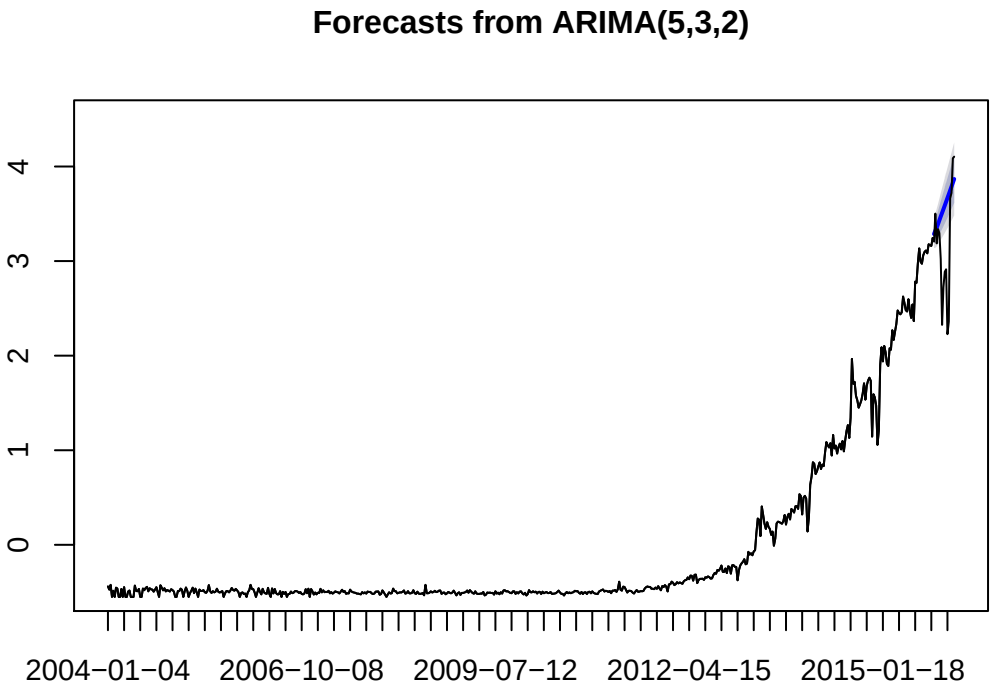

Figure 3: xxxx

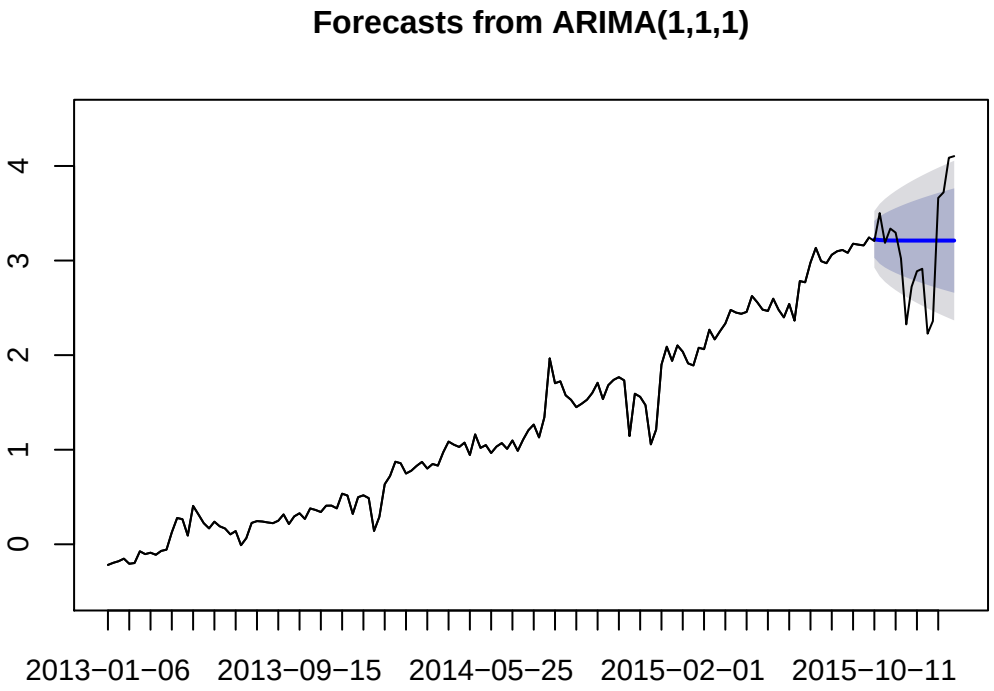

Figure 4: xxxx

Part 4

Forecast Inflation-Adjusted Gas Price

During 2013 amid high gas prices, the Associated Press (AP) published an article about the U.S. inflation-adjusted price of gasoline and U.S. oil production. The article claims that there is "*evidence of no statistical correlation*" between oil production and gas prices. The data was not made publicly available, but comparable data was created using data from the Energy Information Administration. The workspace and data frame `gasOil.Rdata` contains the U.S. oil production (in millions of barrels of oil) and the inflation-adjusted average gas prices (in dollars) over the date range the article indicates.

In support of their conclusion, the AP reported a single p-value. You have two tasks for this exericse, and both tasks need the use of the data set `gasOil.Rdata`.

1st task

Your first task is to recreate the analysis that the AP likely used to reach their conclusion. Thoroughly discuss all of the errors the AP made in their analysis and conclusion.

It would seem reasonable (and that's probably what interested parties tell about the benefits of drilling) that the more oil is produced in the U.S. (mainly because of that new technique), the lower the price of gasoline would be. In other words, we might expect a highly significant **negative** correlation between domestic oil production and (inflation-adjusted) prices[2].

[2] At first sight, that could seem coherent with the *law of supply and demand*: if the supply increases, the price should go down... **assuming the demand is constant** (let's not forget that assumption). That might not be the case, and an increase in both production and demand can result in higher prices. In any case, that so-called law is just an economic model of price determination in a market; as all models, it can fit or not the reality.

Possible sources for the mentioned article might be this one or this other one (though none of them reproduced the phrase "*evidence of no statistical correlation*"). That phrase would be the first error made by the AP (in case they used it): in **hypothesis testing**, we can talk of *no evidence of correlation* (or any other fact) but not of *evidence of no correlation* (in general, of evidence of a fact not occurring). Remember that, whatever a **null hypothesis** is (in this case, that the **correlation** is—**not statistically significantly different from**—**zero**), we can never prove or confirm it, just claim that we have evidence or not to reject it. To put an example, if we toss a coin N times and get heads N times, we do not have evidence that the coin is fair (i.e., $\Pr(heads) = 0.5$); but we should not claim that we have evidence that the opposite is true (i.e., the coin is unfair or biased); the most we can say, strictly speaking, is that we are quite confident (the more confident the greater the number of tosses).

Let's continue by loading and exploring the data frame we are given:

```
load('gasOil.Rdata')
rbind(head(gasOil,4 ), tail(gasOil, 4))
```

```
##           Date Production    Price
## 1   1978-01-01    259.150 2.456692
## 2   1978-02-01    234.544 2.441220
## 3   1978-03-01    270.324 2.425818
## 4   1978-04-01    264.526 2.414277
## 407 2011-11-01    179.099 3.540914
## 408 2011-12-01    185.712 3.417614
## 409 2012-01-01    190.358 3.527641
## 410 2012-02-01    180.969 3.726987
```

```
gasOil$Date <- as.Date(as.character(gasOil$Date), '%Y-%m-%d')
summary(gasOil)
```

```
##       Date              Production        Price
##  Min.   :1978-01-01   Min.   :119.4   Min.   :1.329
##  1st Qu.:1986-07-08   1st Qu.:173.0   1st Qu.:1.823
##  Median :1995-01-16   Median :201.4   Median :2.096
##  Mean   :1995-01-15   Mean   :210.0   Mean   :2.391
##  3rd Qu.:2003-07-24   3rd Qu.:255.8   3rd Qu.:2.909
##  Max.   :2012-02-01   Max.   :283.2   Max.   :4.432
```

```
round(stat.desc(gasOil[, 2:3], desc = TRUE, norm = TRUE), 2)
```

```
##              Production  Price
## nbr.val          410.00 410.00
## nbr.null           0.00   0.00
## nbr.na             0.00   0.00
## min              119.41   1.33
## max              283.25   4.43
## range            163.84   3.10
## sum            86102.96 980.50
## median           201.44   2.10
## mean             210.01   2.39
## SE.mean            2.07   0.03
## CI.mean.0.95       4.07   0.07
## var             1753.57   0.49
```

```
## std.dev           41.88   0.70
## coef.var           0.20   0.29
## skewness           0.17   0.71
## skew.2SE           0.71   2.95
## kurtosis          -1.38  -0.59
## kurt.2SE          -2.87  -1.23
## normtest.W         0.92   0.91
## normtest.p         0.00   0.00
```

```
# Check that all months between start and end dates appear in the dataset
identical(gasOil$Date, seq(min(gasOil$Date), max(gasOil$Date), by='month'))
```

```
## [1] TRUE
```

The dataset contains 410 observations of 3 variables: the first one corresponds to dates (in character format), the second to U.S. oil production (in millions of barrels of oil, ranging from 119.4 to 283.2 millions of barrels), and the third one to inflation-adjusted average gas prices (in U.S. dollars, ranging from 1.33 to 4.43 USD). All dates correspond to the first day of the month (i.e., we have monthly observations of production and price), from January 1978 until February 2012 (i.e., 34 years—from 1978 to 2011—and 2 months—the first 2 moths of 2012). There are no missing values for any month, and all months between the start and end date are available in the dataset.

```
Production <- ts(data = gasOil$Production, start = year(gasOil$Date[1]),
                 frequency = 12)
Price <- ts(data = gasOil$Price, start = year(gasOil$Date[1]), frequency = 12)
```

Oil production was relatively flat (it was actually U-shaped, but it decreased and then increased at a much lower rate than it did afterwards) from 1978 to 1985, then it had a declining trend until 2009 or so, and it has increased (at a lower rate) since then (probably due to the introduction of driling).

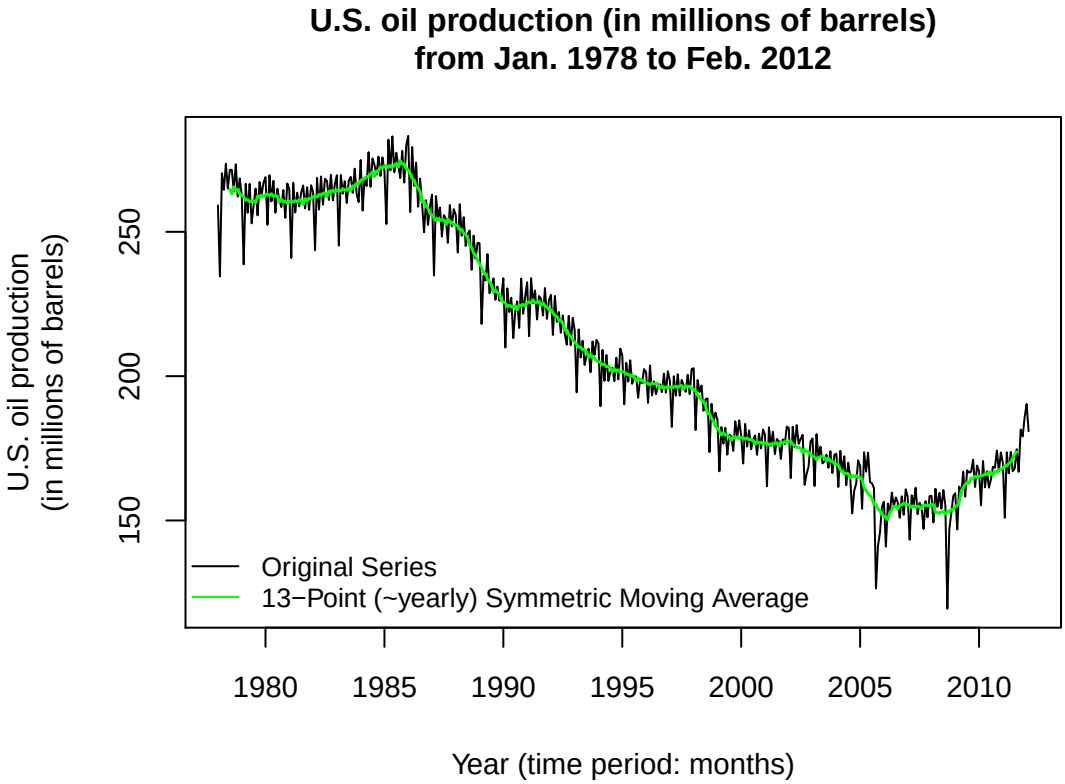

Figure 5: Time series plots of the U.S. oil production (in millions of barrels) from January 1978 to February 2012

As for the inflation-adjusted average gas prices, their dynamics are quite different: they increased a lot (more than \$1, almost a 50% increase) from 1978 to 1981 or so, then decreased until 1986-1987 (to levels below the previous ones), remained relatively flat (or even decreased a bit more) unti 1999, and has kept increasing since then, except for a sharp fall at the end of 2008 (that's approximately when the domestic oil production began to increase again; maybe the drop in production was due to the hype about drilling?).

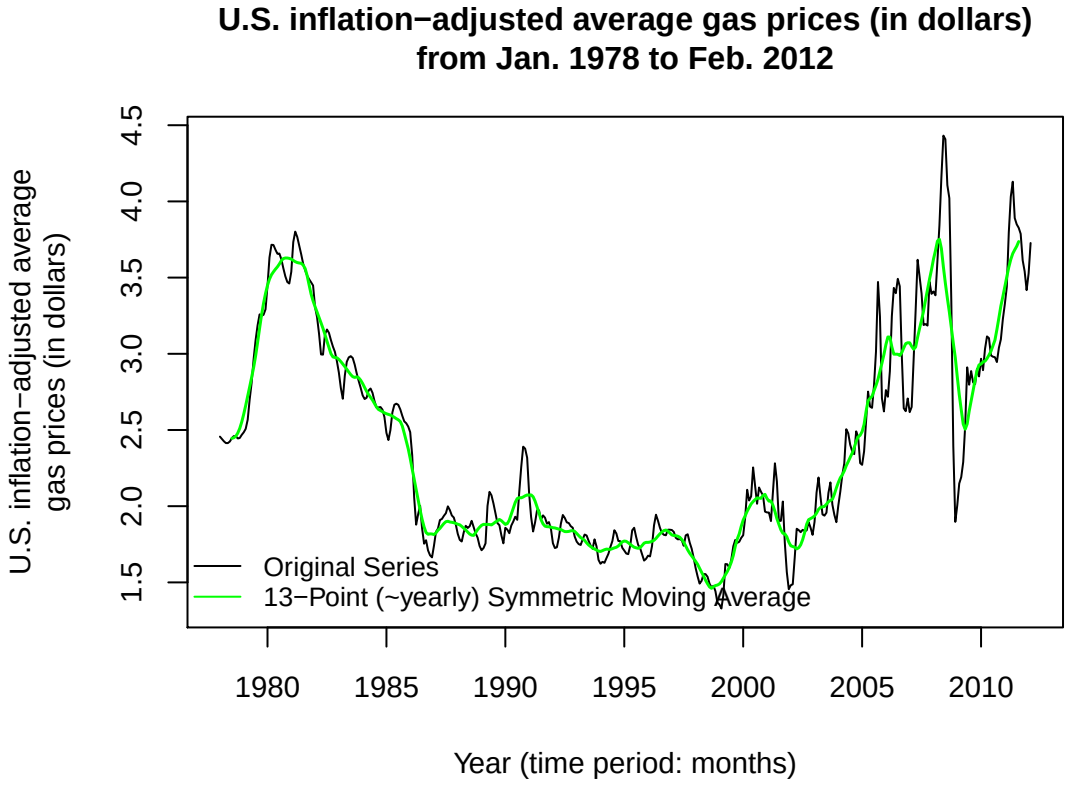

Figure 6: Time series plot of the U.S. inflation-adjusted average gas prices (in dollars) from January 1978 to February 2012

Another thing to notice is that the oil production has much more variability (it fluctuates a lot around its moving average), while the gas price is more persistent. Neither has a clear increasing or decreasing trend but the trend varies over time. Finally, the production seems to have a yearly seasonal component (this is later confirmed when plotting its PACF), while the price does not.

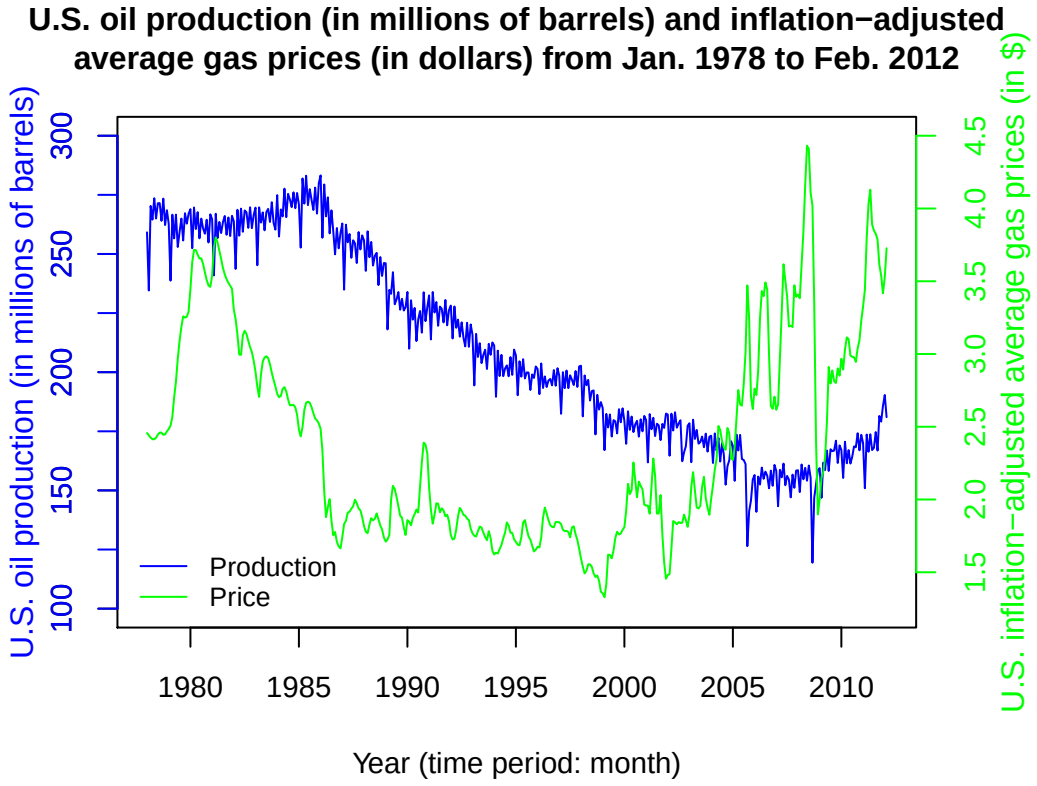

Figure 7: Combined time series plot of the U.S. inflation-adjusted average gas prices (in dollars) and U.S. inflation-adjusted average gas prices (in dollars) from January 1978 to February 2012

The Figure in the next page shows the (approximate) density plots of both time series (one is bimodal and the other is very right-skewed; anyway, density plots tell us nothing about the dynamics of a time series), as well as the correlation (close to zero) and the scatterplot (U-shaped instead of a diagonal line).

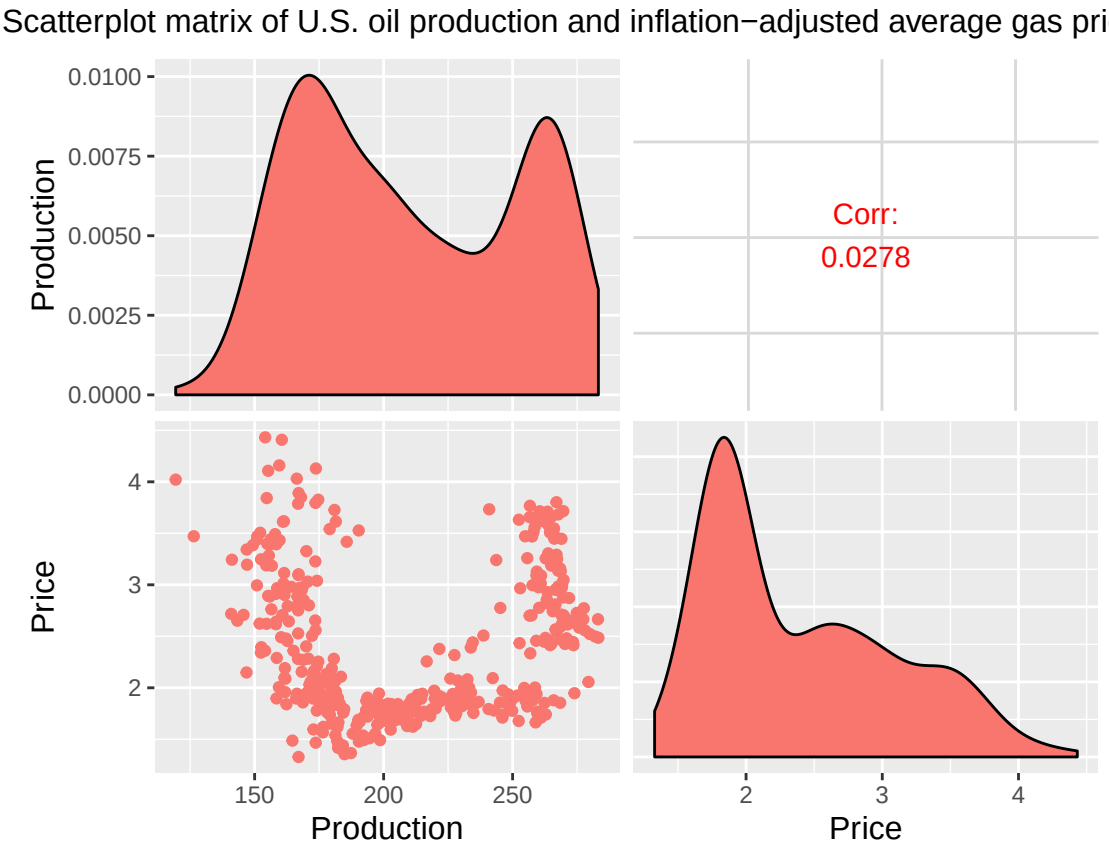

Figure 8: Matrix of the density plots, correlation, and scatterplot of the U.S. oil production and inflation-adjusted average gas prices, from January 1978 to February 2012

Let's now run a Pearson's correlation test to estimate the correlation between both time series, as the AP did, as well the p-value and standard error.

```
(ProdPrice.cor <- cor.test(Production, Price))
```

```
##
##  Pearson's product-moment correlation
##
## data:  Production and Price
## 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
```

The estimated correlation, as the previous Figure already showed, is about 0.028, not significantly different from zero ($p = 0.575$, and the confidence interval includes zero). That is consistent with the claims from the AP. So the mathematical result, so to speak, is true.

Another way to estimate the correlation is running a linear regression with one of the time series as the regressand and the other as the regressor. The squared root of the R-squared value of the regression is the correlation between both.

```
(linReg <- summary(lm(Price ~ Production)))
```

```
##
## Call:
## lm(formula = Price ~ Production)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0430 -0.5683 -0.2762  0.5287  2.0660
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.2943109  0.1765964  12.992   <2e-16 ***
## Production  0.0004626  0.0008247   0.561    0.575
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6984 on 408 degrees of freedom
## Multiple R-squared:  0.0007705,  Adjusted R-squared:  -0.001679
## F-statistic: 0.3146 on 1 and 408 DF,  p-value: 0.5752
```

```
(ProdPrice.cor2 <- sqrt(linReg$r.squared))
```

```
## [1] 0.02775705
```

What is erroneous (the second error) is the implications from that result (and the use of correlation, to begin with): correlation is not a good measure of the dependency of two time series. Same way that two independent time series can show a high **spurious correlation** (the correlation between both time series is driven by some underlying common driver or it is merely "coincidental"; there are multiple examples), the opposite can happen (though to noise or other factors that may also drive the time series and "mask" their dependence; that's might be the case here: even if U.S. prices depend on domestic production, it's world production—and other possible factors—what drives them).

This goes down to the **definitions of correlation and time series** models. The correlation is defined as the quotient of the covariance of two random variables divided by the product of their respective standard deviations (the square root of their variances). (The sample estimates[3] of) These two parameters—variance and covariance—depend on the value of the individual observations and their (sample) mean, which is constant (and that is not the case in time series!). Now let's revisit what a stochastic process (which is how we model time series) is: it is **a collection of random variables** representing the evolution of some sytem of random values over time. That is (in the case of discrete time series like the ones we are working with), a sequence of random variables, that may be completely different at the different times (and dependent... or not); the only requirement is that those random variables all take values in the same space. To put it simply, it makes no sense to define the mean of $\{x_t\}$, $t = 1, \ldots, n$[4] because $x_1, x_2, \ldots, x_n$ **are not observations from a single random variable**, **but** from a realization of a stochastic process, i.e., **from** n **different random variables, not necessarily i.i.d.**

[3] Which is what we are able to estimate (we are often not able to estimate the population estimates).
[4] The same can be said of the correlation, if we extend this idea to two time series, $\{x_t\}$ and $\{y_t\}$.

What could have been done different? Two time series that are independent and contain unit roots (i.e., they exhibit **stochastic trends**) may show an apparent linear relationship, due to chance similarity of the random walks over the period of the time series. However, it is also possible that those time series are actually related / **cointegrated** (if **a linear combination of them is stationary**). Hence, **we can check if production and price are cointegrated. If we don't have evidence that supports that hypothesis, we also have no evidence of a (linear) relationship. At the same time, the analysis of the inflation-adjusted gas prices will serve us a first step in the creation of a model for the 2nd task.**

As a first step, we plot the ACF and PACF of both time series. They do not suggest that any of the two series is a random walk, since the PACF does not fall sharply after the 1st lag. That would have been the simplest case, but it's always worth exploring it (we could also have plotted the ACF and PACF of the first difference.)

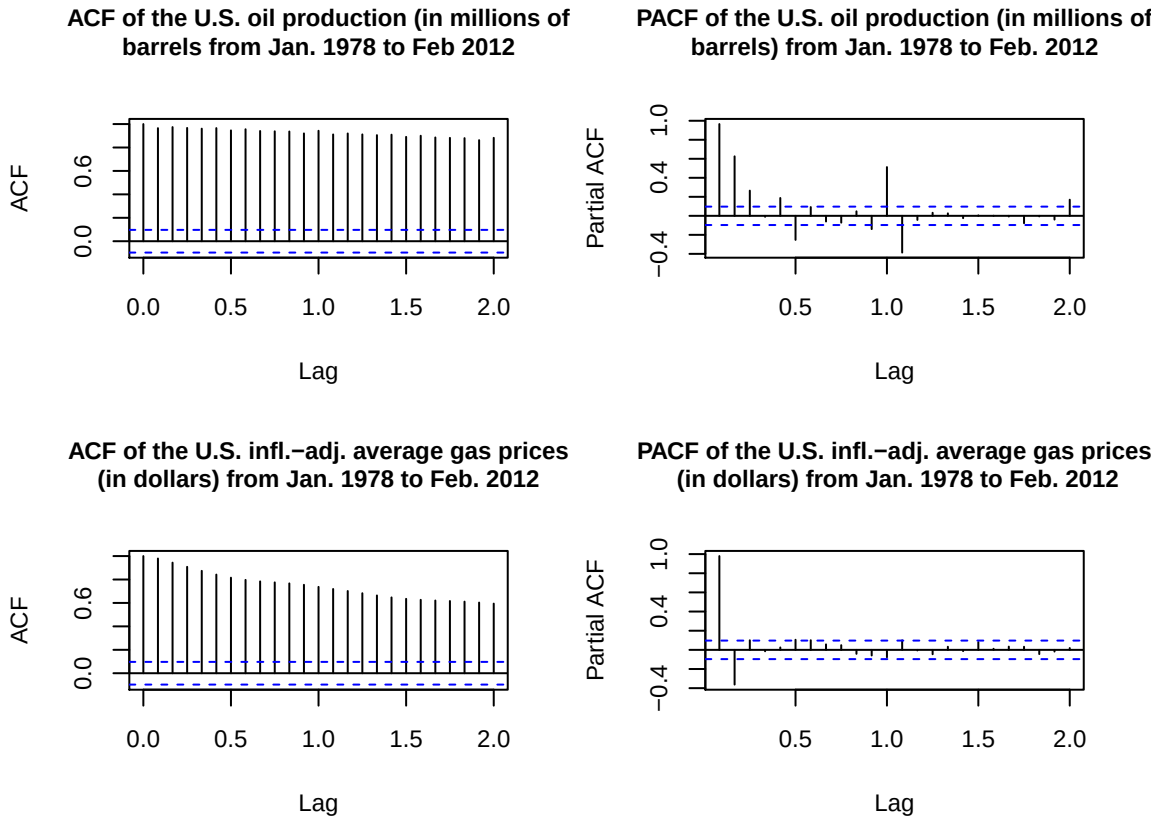

Figure 9: ACF and PACF of the U.S. oil production and inflation-adjusted average gas prices, from January 1978 to February 2012

To test for unit roots, we use the augmented Dickey-Fuller and the Phillips-Perron tests. Based on the p-values of both tests, there is no evidence to reject the unit root hypothesis in the price time series; interestingly, the results of both tests are completely different for the production time series.

```
# Augmented Dickey-Fuller Test
adf.test(Production)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Production
```

```
## Dickey-Fuller = -0.10686, Lag order = 7, p-value = 0.99
## alternative hypothesis: stationary
```

```
adf.test(Price)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Price
## Dickey-Fuller = -1.0162, Lag order = 7, p-value = 0.9355
## alternative hypothesis: stationary
```

```
# Phillips-Perron Unit Root Test
pp.test(Production)
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  Production
## Dickey-Fuller Z(alpha) = -124.32, Truncation lag parameter = 5,
## p-value = 0.01
## alternative hypothesis: stationary
```

```
pp.test(Price)
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  Price
## Dickey-Fuller Z(alpha) = -9.5647, Truncation lag parameter = 5,
## p-value = 0.5752
## alternative hypothesis: stationary
```

Now we run a Phillips-Ouliaris test to test the cointegration of both time series. And we find no evidence of cointegration between U.S. oil production and inflation-adjusted average gas prices, from January 1978 to February 2012 (the p-value is 0.15 so we cannot reject the null hypothesis that the two series are **not** cointegrated).

```
po.test(gasOil[, 2:3])
```

```
##
##  Phillips-Ouliaris Cointegration Test
##
## data:  gasOil[, 2:3]
## Phillips-Ouliaris demeaned = -3.5509, Truncation lag parameter =
## 4, p-value = 0.15
```

2nd task

Your second task is to create a more statistically-sound model that can be used to predict/forecast inflation-adjusted gas prices. Use your model to forecast the inflation-adjusted gas prices from 2012 to 2016.

In the previous task we already found that the price series is likely to have a unit root, no seasonal component, and possibly an AR component of order 2 (because its PACF falls sharply after that lag).

First we explore ARIMA possible models based on their AIC and BIC values, re-using part of the code we used in **HW 8** with the following changes:

- This time we include integrated series of order d.
- But not SARIMA models since it seems the series has no seasonal component.
- We limit the maximum order of p, d, or q to 3. As we know there is a unit root, the minimum order of d will be 1.

> Same results are found if we include $d = 0$. The same goes for SARIMA models (anyway, including the seasonal components (P, D, and Q) makes this "brute-force" approach take a very long time).

```
max_coef <- 3
orders <- data.frame(permutations(n = max_coef + 1, r = 3, v = 0:max_coef,
                                  set = FALSE, repeats.allowed = TRUE))
dim(orders)[1] # Number of models up to max_coef
```

```
## [1] 64
```

```
colnames(orders) <- c("p", "d", "q")
orders <- orders %>% dplyr::filter(d >= 1)
dim(orders)[1] # Number of models considered
```

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

```
orders %>% sample_n(10) # A 10-sample of the possible orders
```

```
##    p d q
## 6  0 2 1
## 30 2 2 1
## 29 2 2 0
## 46 3 3 1
## 38 3 1 1
## 28 2 1 3
## 1  0 1 0
## 10 0 3 1
## 27 2 1 2
## 21 1 3 0
```

```
model_list <- orders %>% rowwise() %>%
  mutate(aic = try_default(AIC(Arima(Price, order = c(p, d, q))), default = NA,
                           quiet = TRUE))
model_list <- model_list %>% dplyr::filter(!is.na(aic))
dim(model_list)[1] # Number of models estimated
```

```
## [1] 47
```

```
model_list <- model_list %>%
  mutate(bic = BIC(Arima(Price, order = c(p, d, q))))
```

Table 1: Top 5 models based on the (lowest) AIC value

p	d	q	aic	bic
1	1	3	-675.6	-655.6
2	1	3	-675.2	-651.1
3	1	3	-674.1	-646.0
1	1	2	-670.8	-654.7
0	1	3	-670.4	-654.3

Table 2: Top 5 models based on the (lowest) BIC value

p	d	q	aic	bic
1	1	3	-675.6	-655.6
0	1	2	-667.4	-655.4
2	1	0	-667.3	-655.2
1	1	2	-670.8	-654.7
0	1	3	-670.4	-654.3

The ARIMA model with the lowest AIC and BIC values is ARIMA(1,1,3). The 2nd and 3rd best models, based on their AIC value (very close to the former), are ARIMA(2,1,3) and ARIMA(3,1,3); the increased number of parameters is penalized by the BIC, so those do not appear in the Top 5 models based on the BIC value: that criterion selects ARIMA(0,1,2) and ARIMA(2,1,0) as the 2nd and 3rd best models, respectively. These 5 models will be our potential candidates.

```
orders_AIC <- model_list %>% arrange(aic) %>% top_n(-3, aic) %>% select(p, d, q)
orders_BIC <- model_list %>% arrange(bic) %>% top_n(-3, bic) %>% select(p, d, q)
orders <- rbind_list(orders_AIC, orders_BIC) %>% unique()
models <- apply(orders, 1, function(arima_order)
  Arima(Price, order = c(arima_order[1], arima_order[2], arima_order[3])))
```

If we use the `auto.arima()` function instead of our own loop the best model based on the AIC value is still the ARIMA(1,1,3)... even if we include the seasonal components (so our decision of excluding them seems correct). The best model based on the BIC value is the ARIMA(0,1,2) (possibly because the function uses other method by default different from `Arima()`).

```
auto.arima(Price, seasonal = TRUE, ic = "aic") # same result using AICc
```

```
## Series: Price
## ARIMA(1,1,3)
##
## Coefficients:
##          ar1      ma1      ma2      ma3
##       0.7578  -0.1455  -0.3948  -0.2355
## s.e.  0.0947   0.1039   0.0565   0.0508
```

```
##
## sigma^2 estimated as 0.01104:  log likelihood=342.82
## AIC=-675.65   AICc=-675.5   BIC=-655.58
```

```
auto.arima(Price, seasonal = TRUE, ic = "bic")
```

```
## Series: Price
## ARIMA(0,1,2)
##
## Coefficients:
##          ma1     ma2
##       0.6453  0.1767
## s.e.  0.0531  0.0516
##
## sigma^2 estimated as 0.01133:  log likelihood=336.7
## AIC=-667.41   AICc=-667.35   BIC=-655.37
```

As we know, the lowest AIC or BIC value may not necessarily involve the best model (with highest explanatory power), especially when we're interested in forecasting. That criterion should be combined with others, such as how much the residuals of the model resemble a white noise (and then selecting the simplest model among those). So next we examine the ACFs and PACFs of the residuals of all these models.

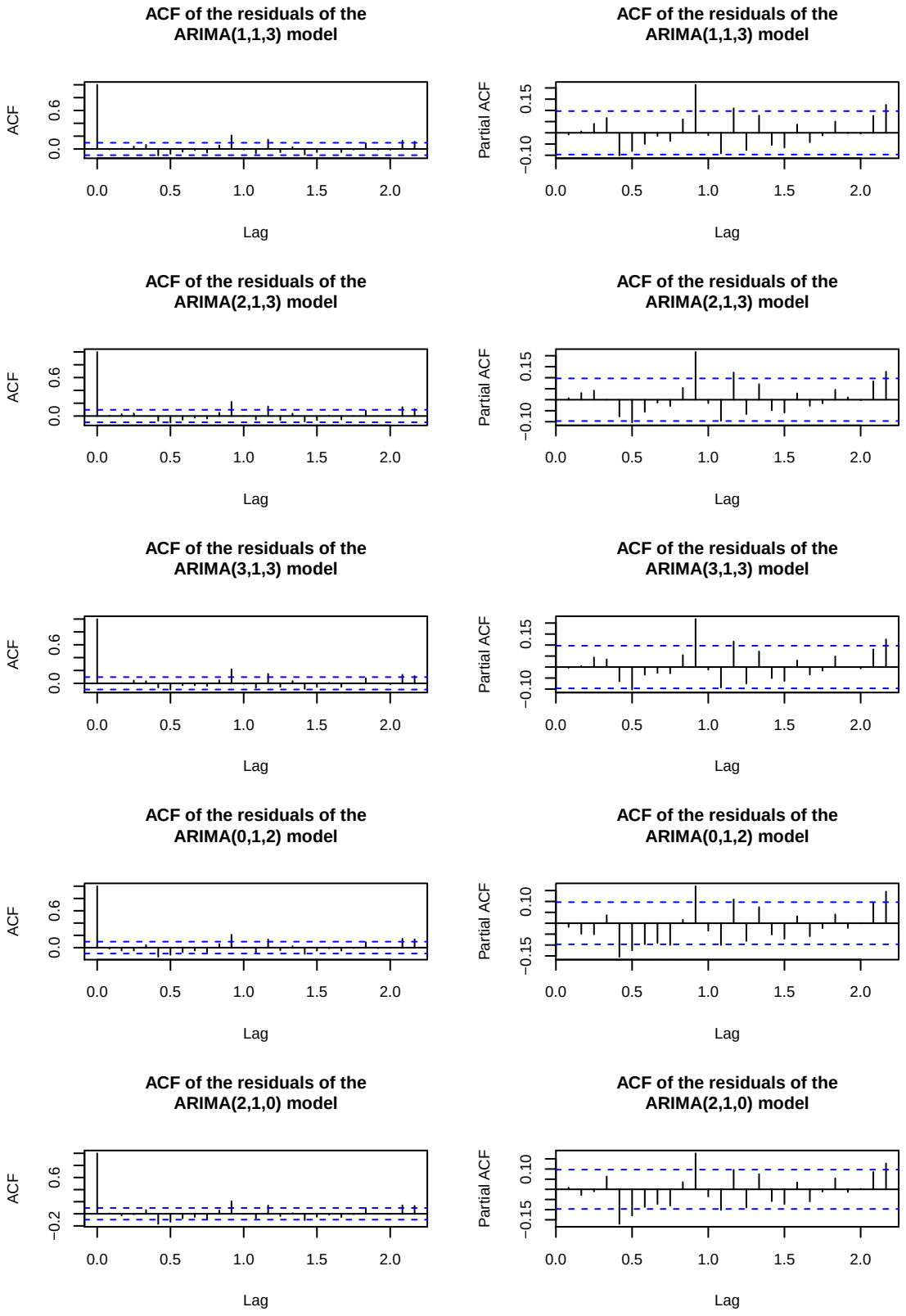

Figure 10: ACF and PACF of the 5 candidates models for the U.S. inflation-adjusted average gas prices, from January 1978 to February 2012

If the residuals were a white noise, only 5% of the auto-correlations (or partial auto-correlations), on average, would be significant (by mere chance). That would correspond to 1 (or 2 at the most) significant auto-correlations (or partial auto-correlations) at the 24 lags we have plotted, but the previous Figure show that 3 or 4 (up to 5 or 6 for the last 2 models, ARIMA(0,1,2) and ARIMA(2,1,0)) are significant. **Anyway, the significant auto-correlations** (which, for the best 3 models based on the AIC value, always occur from lag 9 on) **have a relatively low value (0.2 or so)**, so we'll assume the residuals of those 3 models approximately ensemble a white noise.

As it happened in **HW8**, most of the plots do not differ too much between one another, so selecting the best is difficult after a visual inspection. We complement that with our previous approach: explore the sum of the absolute value of all the auto-correlations (and partial auto-correlations) that are significant (i.e., that exceed $2/\sqrt{n}$, the variance of the lag k autocorrelation—ρ_k—of a white noise, in absolute value).

```
sum_acf <- function(model) {
  # Get the ACFs of first 24 lags
  ACF <- stats::acf(model$residuals, plot = FALSE, lag.max = 24)$acf
  # Exclude (assign 0) to those not significant
  significant_ACF <- ifelse(abs(ACF) < qnorm(.975) / sqrt(model$nobs), 0,
                            abs(ACF))
  # Sum absolute values (exluding lag 0)
  return(sum(significant_ACF[-1]))
}
sum_pacf <- function(model) {
  # Get the PACFs of first 24 lags
  PACF <- pacf(model$residuals, plot = FALSE, lag.max = 24)$acf
  # Exclude (assign 0) to those not significant
  significant_PACF <- ifelse(abs(PACF) < qnorm(.975) / sqrt(model$nobs), 0,
                             abs(PACF))
  # Sum absolute values
  return(sum(significant_PACF))
}
model_list <- join(orders, model_list, by=c("p","d","q"), type="inner") %>%
  rowwise() %>%
  mutate(ACF = sum_acf(Arima(Price, order = c(p, d, q))),
         PACF = sum_pacf(Arima(Price, order = c(p, d, q))))
```

Table 3: Top 5 models based on the (lowest) sum of the absolute value of their (significant) auto-correlations

p	d	q	aic	bic	ACF	PACF
1	1	3	-675.6	-655.6	0.5	0.4
3	1	3	-674.1	-646.0	0.5	0.4
2	1	3	-675.2	-651.1	0.5	0.4
0	1	2	-667.4	-655.4	0.7	0.8
2	1	0	-667.3	-655.2	0.8	0.6

The Table above confirms our visual inspection of the plots in the Figure of the previous page: the 3 models with the lowest AIC value are similar in terms of the ACF and PACF of their residuals, and the other 2 models (2nd and 3rd best models based on the BIC value) are worse in terms of their residuals.

So the ARIMA(0,1,2) and ARIMA(2,1,0) models have similar BIC values than the ARIMA(1,1,3) model, and they are less complex (in terms of the number of coefficients (2 vs. 4), but their AIC values are higher (and hence worse; though this is not a critical issue; we are interested in the best predictions of future values,

not the best fitting of the past ones) and (more importantly) their residuals do not resemble white noise so well. As for the ARIMA(2,1,3) and ARIMA(3,1,3) models, their AIC values are almost equal to that of the ARIMA(1,1,3) model and their residuals look almost the same, but their BIC values are much higher and they are more complex (5 and 6 coefficients vs. 4). Summarizing, we select the ARIMA(1,1,3) model as the best candidate; we'll also try the (much simpler) ARIMA(0,1,2) model.

To select between these 2 models we will also analyze their out-of-sample fit (we'll omit the in-sample fit for the whole time period; the results for the training set used in the out-of-sample fit look pretty similar). To train the models we will use approximately 90% of the original observations, 31 years, leaving out the last 38 months.

```
models <- models[c(1,4)]
orders <- orders[c(1,4), ]
Price.train <- window(Price, start = 1978, end=c(2008, 12))
Price.test <- window(Price, start = 2009)
(arima113.oos.fit <- Arima(Price.train, order = as.numeric(orders[1, ])))
```

```
## Series: Price.train
## ARIMA(1,1,3)
##
## Coefficients:
##           ar1     ma1     ma2     ma3
##       -0.7445  1.4761  0.6657  0.0159
## s.e.   0.1290  0.1387  0.1446  0.0737
##
## sigma^2 estimated as 0.009951:  log likelihood=330.35
## AIC=-650.71   AICc=-650.54   BIC=-631.13
```

Time	Original series	Estimated series	Residuals
Jan 1978	2.46	2.45	0.00
Feb 1978	2.44	2.45	-0.01
Mar 1978	2.43	2.43	-0.01
Apr 1978	2.41	2.42	-0.01
May 1978	2.41	2.41	0.00
Jun 1978	2.42	2.42	0.01

```
arima113.oos.fit.fcast <- forecast.Arima(arima113.oos.fit, h = 38)
```

Table 5: Goodness-of-fit parameters for the training and test sets (ARIMA(1,1,3)

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	-0.0009231	0.0990829	0.0620179	-0.0565951	2.659808	0.226098	0.0017451
Test set	1.3127492	1.4127668	1.3127492	40.0112148	40.011215	4.785874	0.8837268

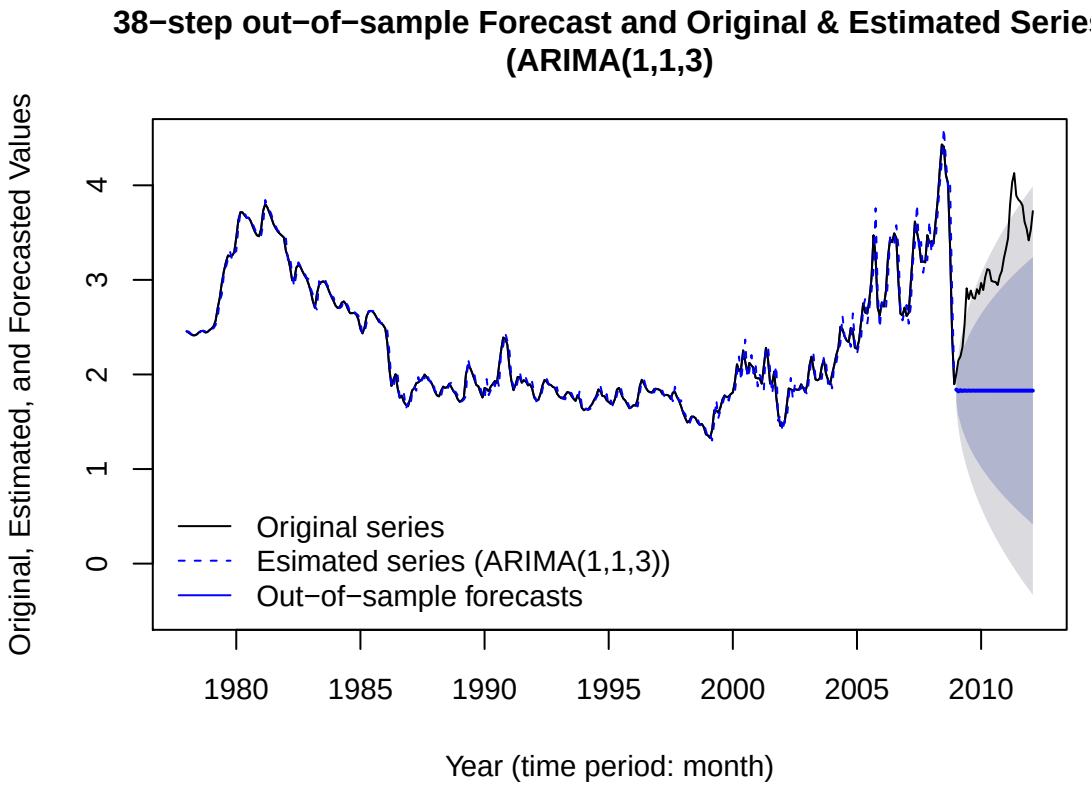

Figure 11: Out-of-sample fit of the ARIMA(1,1,3) model to the U.S. inflation-adjusted average gas prices (in dollars)

```
(arima012.oos.fit <- Arima(Price.train, order = as.numeric(orders[2, ])))
```

```
## Series: Price.train
## ARIMA(0,1,2)
##
## Coefficients:
##          ma1     ma2
##       0.7215  0.1895
## s.e.  0.0568  0.0511
##
## sigma^2 estimated as 0.0103:  log likelihood=323.15
## AIC=-640.3   AICc=-640.23   BIC=-628.55
```

Time	Original series	Estimated series	Residuals
Jan 1978	2.46	2.45	0.00
Feb 1978	2.44	2.45	-0.01
Mar 1978	2.43	2.43	-0.01
Apr 1978	2.41	2.42	-0.00
May 1978	2.41	2.41	0.00
Jun 1978	2.42	2.42	0.01

```
arima012.oos.fit.fcast <- forecast.Arima(arima012.oos.fit, h = 38)
```

Table 7: Goodness-of-fit parameters for the training and test sets (ARIMA(0,1,2)

	ME	RMSE	MAE	MPE	MAPE	MASE	ACF1
Training set	-0.0009427	0.1010619	0.0625097	-0.0504004	2.655063	0.2278907	-0.0192977
Test set	1.3622718	1.4588620	1.3622718	41.6361560	41.636156	4.9664178	0.8845666

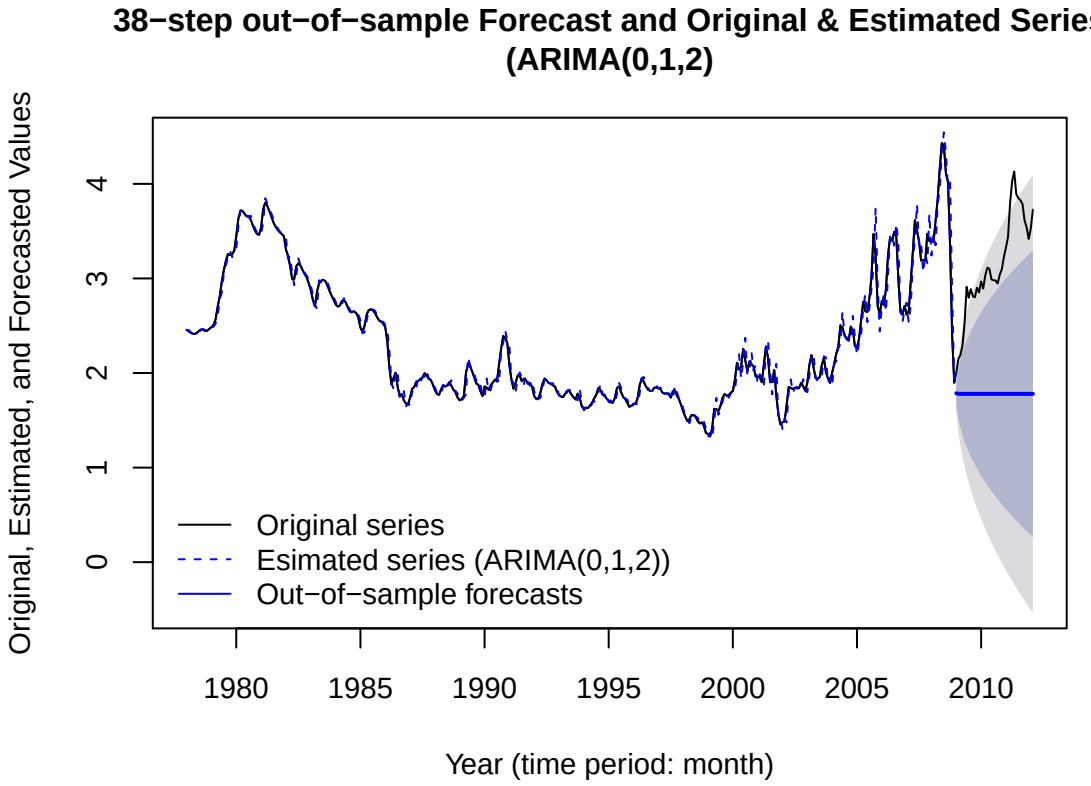

Figure 12: Out-of-sample fit of the ARIMA(0,1,2) model to the U.S. inflation-adjusted average gas prices (in dollars)

The last 2 Figures look very similar. For both models:

- the in-sample fit (of the training set) is very good,
- the mean value of the forecasts is (almost) constant, equal to the last value in the training set, and
- the "real" values in the test set fall within the confidence region of the forecasts, except for the peak in the middle of 2011.

We have to **look at the RMSE, MAE, and other goodness-of-fit parameters in the Tables previously shown** to see that the ARIMA(1,1,3) is a better fit (though the differences with the ARIMA(0,1,2) model are small). Hence, that's the model we'll use to forecast the inflation-adjusted gas prices from 2012 to 2016.

```
(arima113.fit <- models[[1]])
```

```
## Series: Price
## ARIMA(1,1,3)
##
## Coefficients:
##          ar1      ma1      ma2      ma3
##       0.7578  -0.1455  -0.3948  -0.2355
## s.e.  0.0947   0.1039   0.0565   0.0508
##
## sigma^2 estimated as 0.01104:  log likelihood=342.82
## AIC=-675.65   AICc=-675.5   BIC=-655.58
```

Table 8: Coefficients, SEs, and 95% CIs of the estimated ARIMA(1,1,3) model

	Coefficient	SE	95% CI lower	95% CI upper
ar1	0.7578	0.0947	0.5685	0.9472
ma1	-0.1455	0.1039	-0.3533	0.0623
ma2	-0.3948	0.0565	-0.5078	-0.2818
ma3	-0.2355	0.0508	-0.3371	-0.1339

```
arima113.fit.fcast <- forecast.Arima(arima113.fit, h = 58)
pander(predict(arima113.fit, n.ahead = 58)$pred)
```

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
2012	NA	NA	3.823	3.815	3.784	3.761	3.744	3.731	3.721	3.713	3.708	3.703
2013	3.7	3.697	3.696	3.694	3.693	3.692	3.692	3.691	3.691	3.69	3.69	3.69
2014	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69
2015	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69
2016	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69	3.69

I.e, our model, ARIMA(1,1,3), for $\{x_t\}$ (where x_t is the U.S. inflation-adjusted average gas prices (in dollars) at time t) is:

$$\Theta_1(B)(1-B)^1 x_t = \Phi_3(B)\omega_t$$

$$(1 - B - 0.758B)(1 - B)x_t = (1 + -0.145B + -0.395B^2 + -0.236B^3)\omega_t$$

$$x_t = 1.758x_{t-1} - 0.758x_{t-2} + \omega_t - 0.145\omega_{t-1} - 0.395\omega_{t-2} - 0.236\omega_{t-3}$$

where $\{w_t\}$ is a white noise series with mean zero and variance σ^2.

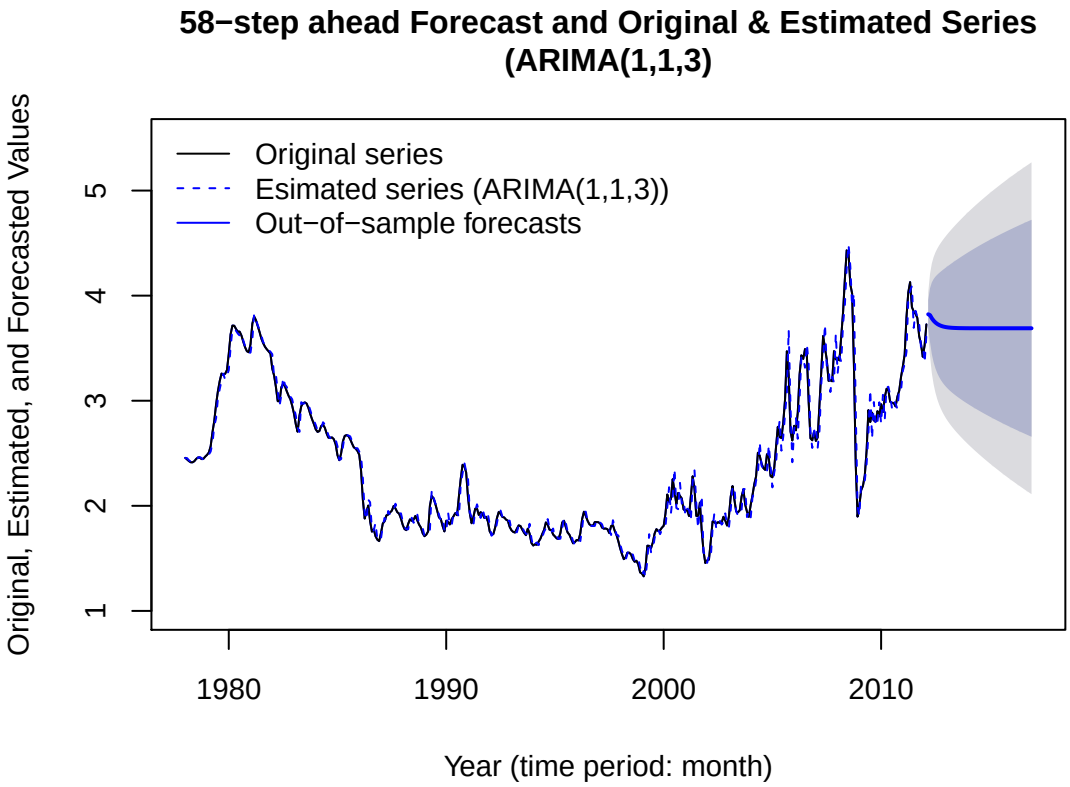

Figure 13: 58-step ahead forecasts (from March 2012 to December 2016) of the U.S. inflation-adjusted average gas prices (in dollars) based on an ARIMA(1,1,3) model fitted to data from January 1978 to February 2012

As shown above, the confidence region of the forecasts is quite wide. But we haven't checked for conditional heteroskedasticity (volatility) yet, and enhancing our model may allow us to narrow down those confidence intervals. After checking the auto-correlations of the squared residuals of our model (see the 1st Figure in the following page) we find that many of them are significant, which indicates volatility, so we start applying a GARCH(1,1) model.

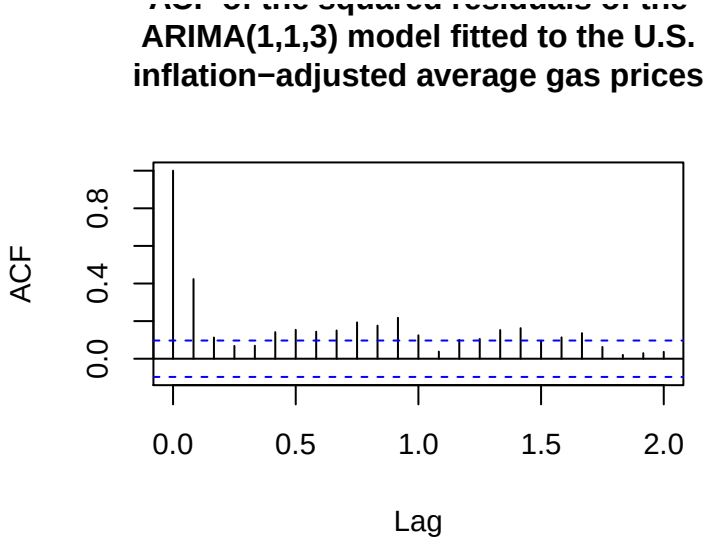

Figure 14: ACF of the squared residuals of the ARIMA(1,1,3) model fitted to the U.S. inflation-adjusted average gas prices

```
(Price.garch11 <- garch(resid(arima113.fit), trace = FALSE))
```

```
##
## Call:
## garch(x = resid(arima113.fit), trace = FALSE)
##
## Coefficient(s):
##        a0         a1         b1
## 0.0002877  0.2158131  0.7753565
```

```
Price.garch11.res <- Price.garch11$res[-1]
t(confint(Price.garch11))
```

```
##                  a0        a1        b1
## 2.5 %  0.0001442508 0.1219635 0.6961534
## 97.5 % 0.0004312333 0.3096627 0.8545597
```

As shown below, the residuals of that GARCH(1,1) model fairly resemble a white noise (the auto-correlation at lag 16 is significant, but about 5% of them could be, just due to chance), so we can use this model GARCH(1,1).

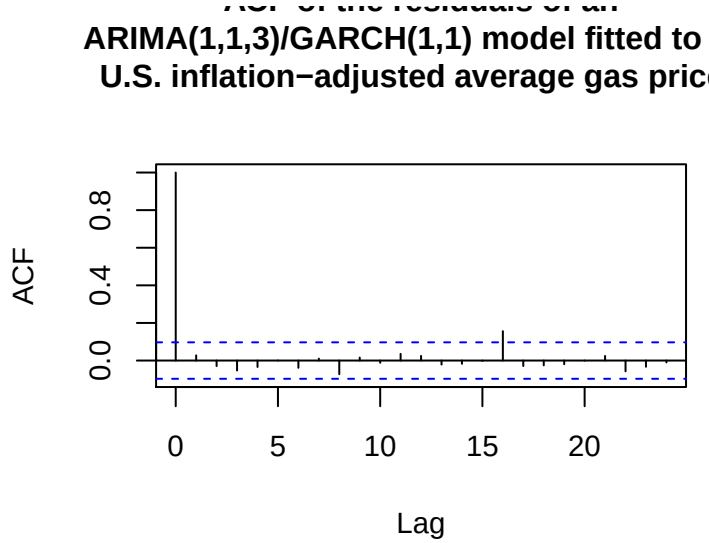

Figure 15: ACF of the residuals of an ARIMA(1,1,3)/GARCH(1,1) model fitted to the U.S. inflation-adjusted average gas prices

Hence, our complete model, ARIMA(1,1,3)/GARCH(1,1) is:

$$x_t = 1.758x_{t-1} - 0.758x_{t-2} + \epsilon_t - 0.145\epsilon_{t-1} - 0.395\epsilon_{t-2} - 0.236\epsilon_{t-3}$$

where

$$\epsilon_t = \omega_t\sqrt{h_t}$$

and

$$h_t = \alpha_0 + \alpha_1\epsilon_{t-1}^2 + \beta_1 h_{t-1} = 0.000288 + 0.215813\epsilon_{t-1}^2 + 0.775357h_{t-1}$$

($\{\omega_t\}$ is again a white noise wiht zero mean, but now with unit variance; the variance of the error term—now called ϵ_t—at each moment is h_t.)

Next we estimate the conditional variance of the series ($h_t = \sigma_t^2$), and confirm how it changes with time (especially after 2000).

```
ht <- Price.garch11$fit[,1]^2 # conditional variance
```

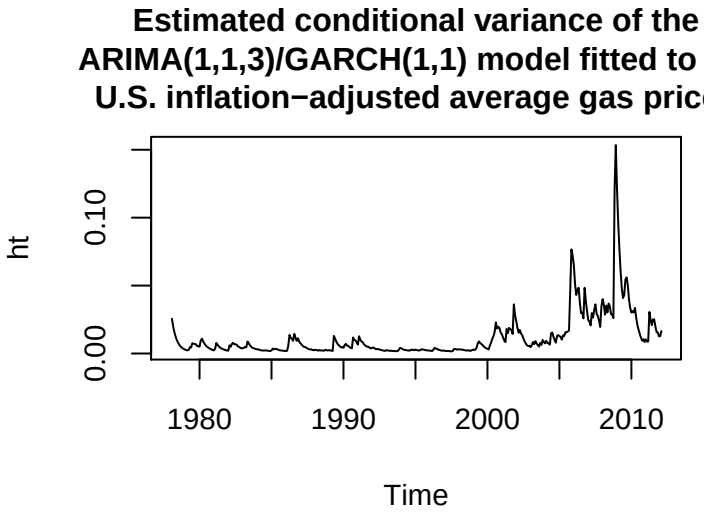

Figure 16: Estimated conditional variance of the ARIMA(1,1,3)/GARCH(1,1) model fitted to the U.S. inflation-adjusted average gas prices

Finally, we have to predict the variance for the 58 months until the end of 2016.

```
res.CI.halfwidth <- qnorm(.975) * sqrt(ht) # CI of epsilon_t
# Variation of Price during observation period
Price.lower <- fitted.values(arima113.fit) - res.CI.halfwidth
Price.upper <- fitted.values(arima113.fit) + res.CI.halfwidth
# Forecasts
# Initialize h_t (cond. variance) and epsilon_t (residuals or error term)
  # 58 elements (as many as forecasts)
ht.fcst <- res.fcst <- rep(0, 58)
for (i in 1:58) {
  if (i == 1) { # use last observation
    ht.fcst[i] <- Price.garch11$coef[1] +
      Price.garch11$coef[2] * resid(arima113.fit)[length(Price)]^2 +
      Price.garch11$coef[3] * ht[length(Price)]
  } else { # use previous predictions
    ht.fcst[i] <- Price.garch11$coef[1] +
      Price.garch11$coef[2] * res.fcst[i-1]^2 +
      Price.garch11$coef[3] * ht.fcst[i-1]
  }
  res.fcst[i] <- sqrt(ht.fcst[i]) # epsilon_t = omega_t * sqrt(h_t)
  #### SHOULDN'T I USE THIS LINE BELOW INSTEAD???
  # res.fcst[i] <- rnorm(1) * sqrt(ht.fcst[i])
  #### THAT GWN(0,1) COMPONENT MAKES h_t FLUCTUATES MUCH MORE
  #### PROBABLY NOT APPROPRIATE FOR FORECASTING
}
# Lower & upper limits of the Price forecasts CI
Price.fcst.lower <- as.numeric(arima113.fit.fcast$mean) -
```

```
  qnorm(.975) * sqrt(ht.fcst)
Price.fcst.upper <- as.numeric(arima113.fit.fcast$mean) +
  qnorm(.975) * sqrt(ht.fcst)
```

And now we can plot the original series and the forecasts until 2016, with the 95% confidence intervals for both periods.

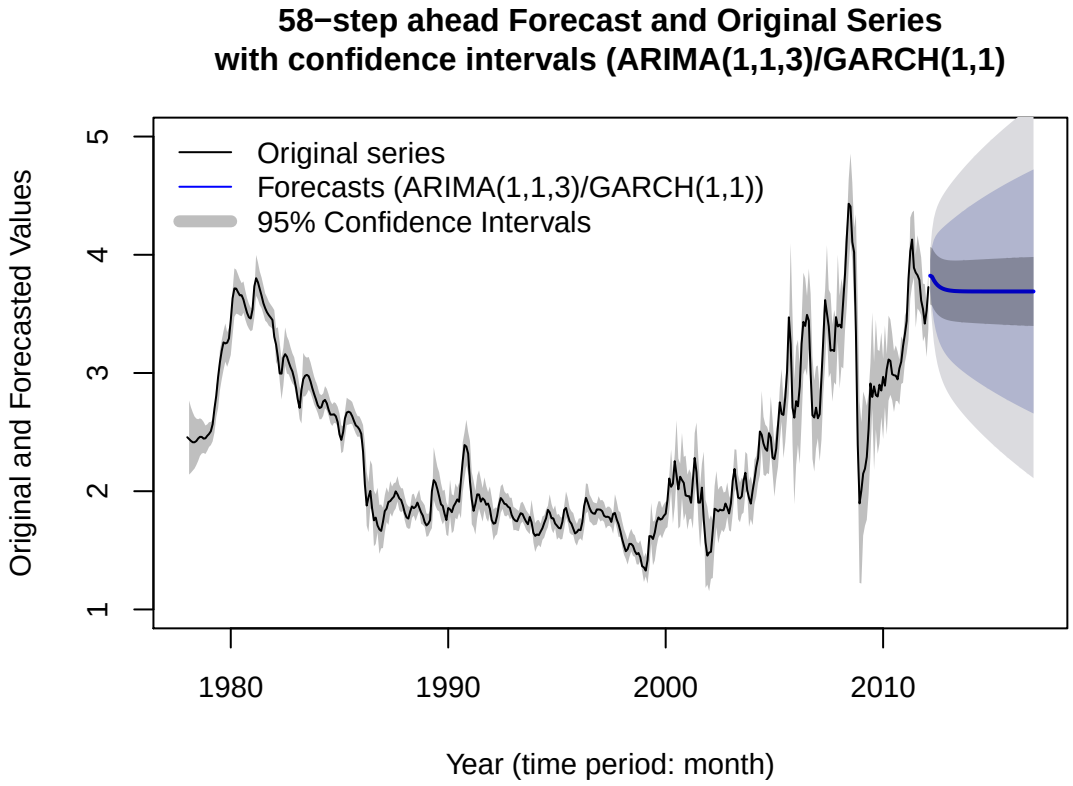

Figure 17: 58-step ahead forecasts (from March 2012 to December 2016) of the U.S. inflation-adjusted average gas prices (in dollars) based on an ARIMA(1,1,3)/GARCH(1,1) model fitted to data from January 1978 to February 2012. Gray area shows the 95% confidence region

Do note how much we've been able to narrow down the confidence intervals (the 80% and 95% CIs for the ARIMA model only are also plotted for comparison). Though it's hard to distinguish from the Figure above, the CIs still increase over time (because the variance of the residuals do), but not as much as when we did not include the GARCH(1,1) model.

The `fGarch` package also allows to make predictions of GARCH models, so we start using it on the residuals of our original ARIMA(1,1,3) model:

```
(Price.garch11.2 <- garchFit(~ garch(1,1), data = resid(arima113.fit),
                             trace = FALSE))
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~garch(1, 1), data = resid(arima113.fit),
##     trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ garch(1, 1)
## <environment: 0xea202f0>
##  [data = resid(arima113.fit)]
##
## Conditional Distribution:
##  norm
##
## Coefficient(s):
##          mu        omega       alpha1        beta1
## -0.00167114   0.00029377   0.21332844   0.77490094
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##          Estimate  Std. Error  t value Pr(>|t|)
## mu     -0.0016711   0.0032399   -0.516  0.60599
## omega   0.0002938   0.0001272    2.310  0.02090 *
## alpha1  0.2133284   0.0753299    2.832  0.00463 **
## beta1   0.7749009   0.0699617   11.076  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  454.3815    normalized:  1.108247
##
## Description:
##  Sun Apr 17 15:45:58 2016 by user:
```

```
# res.fcst2 <- predict(Price.garch11.2, n.ahead=58, plot = TRUE, conf = .95)
res.fcst.2 <- predict(Price.garch11.2, n.ahead=58, conf = .95)
res.fcst.lower.2 <- res.fcst.2$meanForecast -
  qnorm(.975) * res.fcst.2$standardDeviation
res.fcst.upper.2 <- res.fcst.2$meanForecast +
  qnorm(.975) * res.fcst.2$standardDeviation
Price.fcst.lower.2 <- as.numeric(forecast(arima113.fit, 58)$mean) +
  res.fcst.lower.2
Price.fcst.upper.2 <- as.numeric(forecast(arima113.fit, 58)$mean) +
  res.fcst.upper.2
```

The 95% confidence intervals of the forecasts are quite similar (i.e., we've been able to replicate the predictions of 'fGarch' with our own code :).

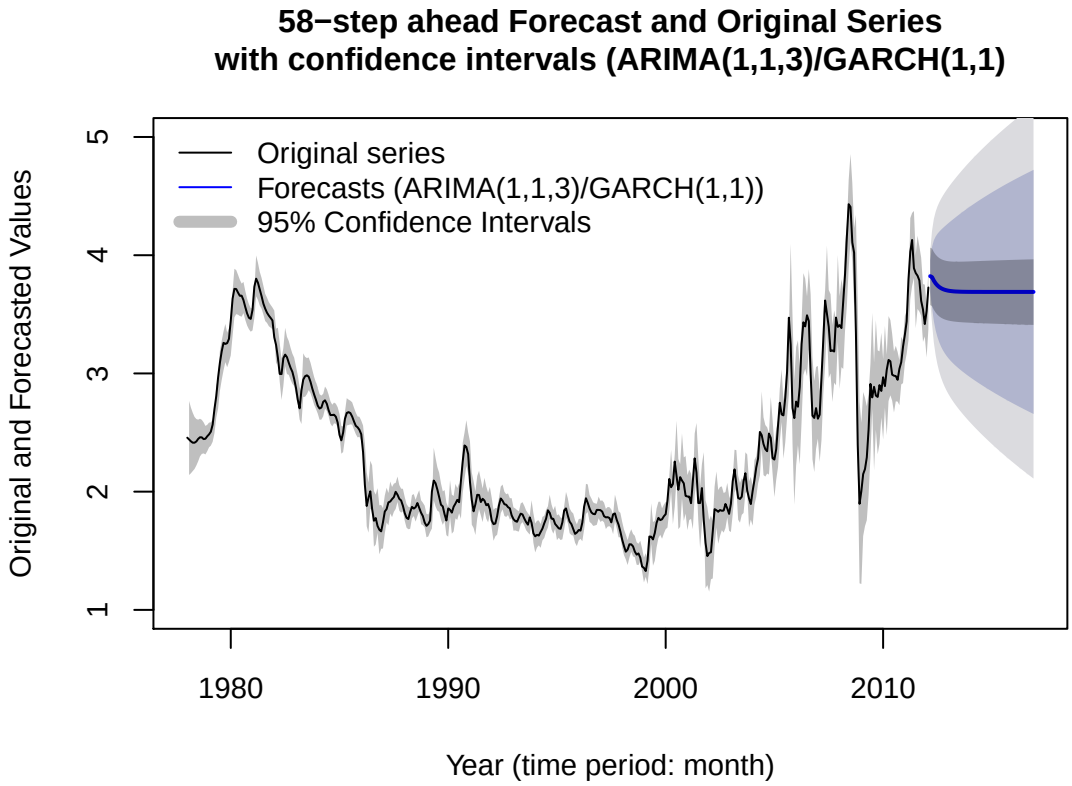

Figure 18: (2nd version of) 58-step ahead forecasts (from March 2012 to December 2016) of the U.S. inflation-adjusted average gas prices (in dollars) based on an ARIMA(1,1,3)/GARCH(1,1) model fitted to data from January 1978 to February 2012. Gray area shows the 95% confidence region

We can also apply an ARMA(1,3)/GARCH(1,1) on the original series differentiated (`fGarch` only allows to combine ARMA and GARCH models, not ARIMA).

```
(Price.garch11.3 <- garchFit(~ arma(1,3) + garch(1,1), data = diff(Price),
                             trace = FALSE))
```

```
##
## Title:
##  GARCH Modelling
##
## Call:
##  garchFit(formula = ~arma(1, 3) + garch(1, 1), data = diff(Price),
##     trace = FALSE)
##
## Mean and Variance Equation:
##  data ~ arma(1, 3) + garch(1, 1)
## <environment: 0x72c5df0>
##  [data = diff(Price)]
##
## Conditional Distribution:
##  norm
```

```
##
## Coefficient(s):
##          mu          ar1          ma1          ma2          ma3
## -0.00011153   0.71547907  -0.05543106  -0.38365694  -0.18357647
##       omega       alpha1        beta1
##  0.00023501   0.17377016   0.81366803
##
## Std. Errors:
##  based on Hessian
##
## Error Analysis:
##          Estimate  Std. Error  t value Pr(>|t|)
## mu     -0.0001115   0.0012419   -0.090 0.928442
## ar1     0.7154791   0.1404980    5.092 3.53e-07 ***
## ma1    -0.0554311   0.1470199   -0.377 0.706151
## ma2    -0.3836569   0.0905220   -4.238 2.25e-05 ***
## ma3    -0.1835765   0.0541993   -3.387 0.000706 ***
## omega   0.0002350   0.0001119    2.101 0.035662 *
## alpha1  0.1737702   0.0640187    2.714 0.006640 **
## beta1   0.8136680   0.0625801   13.002  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log Likelihood:
##  454.048    normalized:  1.110142
##
## Description:
##  Sun Apr 17 15:45:59 2016 by user:
```

```
res.fcst.3 <- predict(Price.garch11.3, n.ahead=58, conf = .95)
res.fcst.lower.3 <- res.fcst.3$meanForecast -
  qnorm(.975) * res.fcst.3$standardDeviation
res.fcst.upper.3 <- res.fcst.3$meanForecast +
  qnorm(.975) * res.fcst.3$standardDeviation
Price.fcst.lower.3 <- as.numeric(forecast(arima113.fit, 58)$mean) +
  res.fcst.lower.3
Price.fcst.upper.3 <- as.numeric(forecast(arima113.fit, 58)$mean) +
  res.fcst.upper.3
```

As expected, the results are quite similar than in the 2 previous cases.

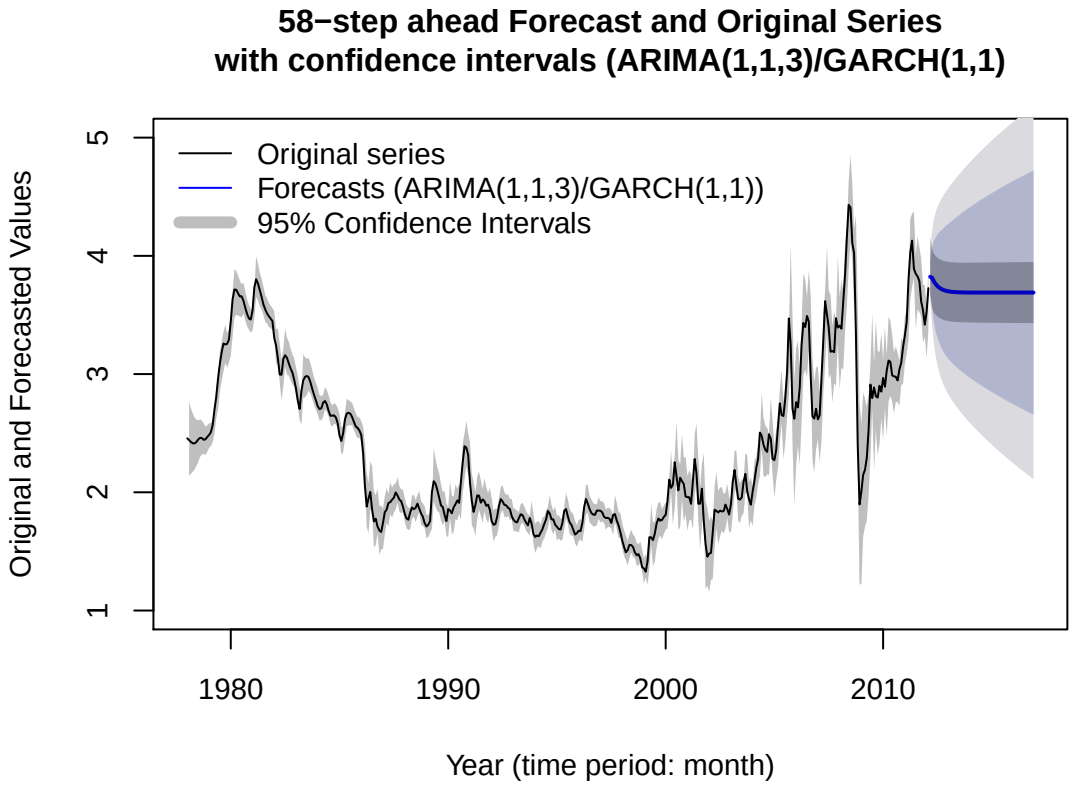

Figure 19: (3rd version of) 58-step ahead forecasts (from March 2012 to December 2016) of the U.S. inflation-adjusted average gas prices (in dollars) based on an ARIMA(1,1,3)/GARCH(1,1) model fitted to data from January 1978 to February 2012. Gray area shows the 95% confidence region

For comparison, let's just finish plotting the standard deviation of the forecasts for the 3 approaches (our own and two using the `fGarch` library):

```
head(cbind(sqrt(ht.fcst), res.fcst.2$standardDeviation,
           res.fcst.3$standardDeviation))
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.1239134 0.1237363 0.1264679
## [2,] 0.1245258 0.1241943 0.1266026
## [3,] 0.1251299 0.1246453 0.1267355
## [4,] 0.1257258 0.1250894 0.1268666
## [5,] 0.1263137 0.1255267 0.1269959
## [6,] 0.1268936 0.1259574 0.1271235
```

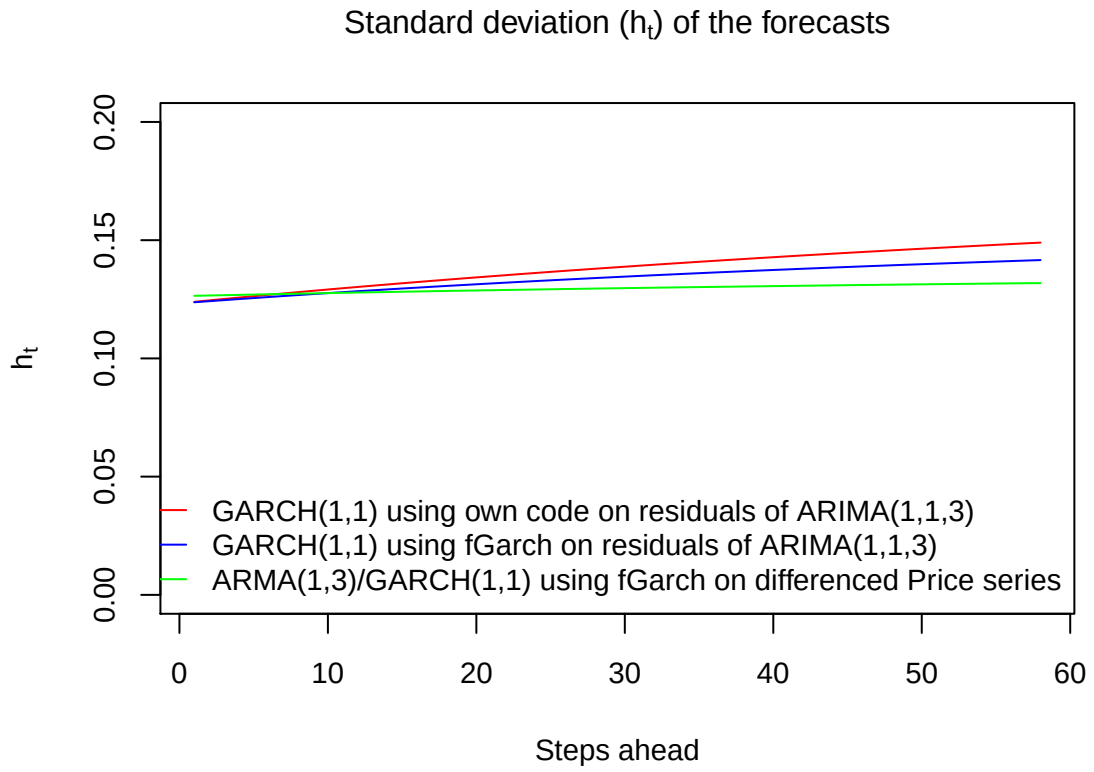

Figure 20: Standard deviation (h_t) of the U.S. inflation-adjusted average gas price forecasts for the 3 methods used