DS809 - Industrial Inputs Price Index Modelling

Nate Thomas

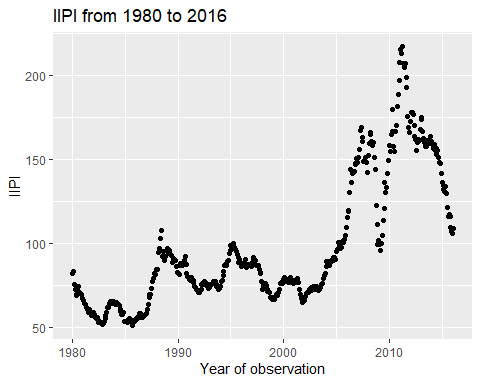
2022-03-05

# Introduction and Overview

This project endeavors to provide example implementation of the various methods outlined in DS809 Time Series. The commodity index Industrial Inputs Price Index is modelled from 1980 through 2016. The Industrial Input Price Index, hereafter IIPI, is provided by the International Monetary Fund (IMF) in the IMF Primary Commodity Prices Report (*IMF Primary Commodity Prices*, 2021). As a break-out index, it does not include energy commodities (Coal, Natural Gas, Spot Crude, Propane) but does include agricultural raw materials (Cotton, Hides, Rubber, Timber, Wool), metals (Aluminum, Cobalt, Copper, Iron Ore, Lead, Molybdenum, Nickel, Tin, Uranium, and Zinc), and precious metals (Gold, Palladium, Platinum, Silver). The index uses weighted average prices as representative of the global market, corrected for volume, as determined by the largest import markets for each commodity constituent. The purpose of the index is to provide a relative value indicator (here indexed to 2005 = 100) for assessment of the overall commodities’ market in question.

The first task performed for this project was to create the time series dataset using *Predictive Dynamics in Commodity Prices* (Gargano & Timmerman, 2012) as a loose framework for select variables therein. These were collected and organized for subsequent analysis. The following is the list of variables collected and their sources:

A. IIPI (Aliyev, 2020), Dependent Variable and Date index. tags: **IIPI, Date\_form**. This dataset defined the time frame of interest (1980-02-01 to 2016-02-01). All other datasets were selected contingent upon their inclusion of this timeframe.



*Plot of IIPI as a function of time*

B. 13 Week Treasury Bill (*US Trade - Statistics & Facts*, 2021), Independent Variable. tag: **TBillClose**.

C. US Real GDP (*Real Gross Domestic Product (GDPC1)* | *FRED* | *St. Louis Fed*, n.d.). tag: **GDP**

D. US Inflation (*US Inflation Rate by Month*, n.d.). tag: **InfRate**

E. Industrial Production (*Industrial Production: Total Index (INDPRO)*| *FRED* | *St. Louis Fed*, n.d.). tag: **IndPro**

F. Money Supply M2, (*BOARD OF GOVERNORS of the FEDERAL RESERVE SYSTEM*, 2021). tag: **M2**

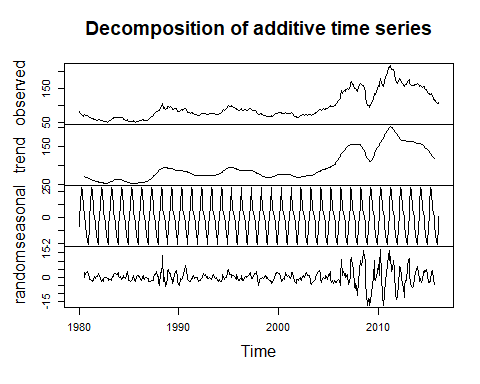
G. US Unemployment, (*Employment and Unemployment*). tag: **UnempRate**

H. Recession, (*Dates of U.S. Recessions as Inferred by GDP-based Recession Indicator*). tag : **Recession**

IIPI is indicative of global market but this project uses United States specific independent variable values as analog facsimiles for global extrapolation. This is a recognized short-hand performed here to allow for timely and comprehensive analysis. United States’ trade accounts for the largest percentage (13.5%) of total import trade worldwide (*US Trade - Statistics & Facts*, 2021), and therefore while this is not perfect US specific data is used for the purposes of this project.

## Visual Inspection of the Data

The data was decomposed using native R *stats* package to visually assess the potential time series decomposition of IIPI.



*Time Series Decomposition*

The scale of the seasonal component is misleading, and is found to be insignificant in subsequent analysis to follow. There is also very clearly non-constant variance in the series as seen in the randomness plot.

## Project Goal

The goal of any time series analysis is to achieve white noise residuals. By achieving white noise residuals one can conclude that a model has captured the non-stochastic behavior of the modeled series.

## Excluding White Noise Criteria

The first task presented in this project is to exclude IIPI as a purely white noise process. If it were white noise then modelling it would be trivial and effectively complete. A white noise process is defined by the following criteria:

1. Constant Mean: for all
2. Constant Variance: for all
3. Autocorrelation is not found at any lags: for all

To show then that a series is not a white noise process one must simply prove at least one of these constraints to be false.

### IIPI White Noise Rejection

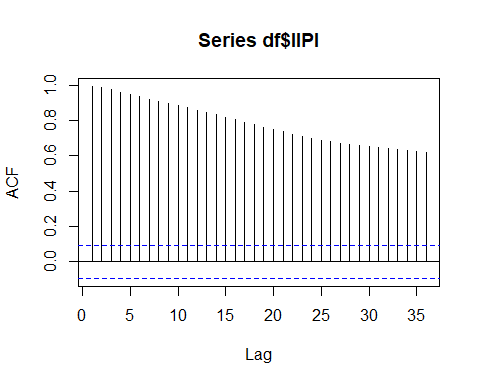
The Box-Pierce Test is used to confirm the existence of autocorrelation up to a defined maximum lag value (here 36 months):

##   
## Box-Pierce test  
##   
## data: df$IIPI  
## X-squared = 9809, df = 36, p-value < 2.2e-16

: All autocorrelctions are zero (0), : At least one (1) autocorrelation is not zero (0)

The test indicates one can safely reject the null hypothesis () - such that at least one (1) autocorrleation is not zero.

This is also seen through visual inspection of the autocorrelation function (ACF):



The series is non-stationary as indicated by the slow decay in autocorrelation.

Based on this, the series is not a white noise process.

### IIPI First Difference White Noise Process Rejection

The first difference is also reviewed to ensure it is not white noise.

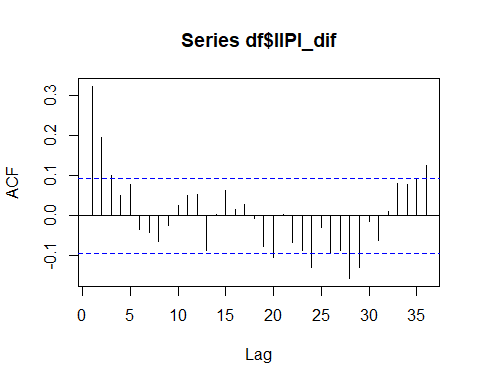
Again, the Box-Pierce Test is used to confirm the existence of autocorrelation up to a defined maximum lag value (here 36 months) for the first difference:

##   
## Box-Pierce test  
##   
## data: df$IIPI\_dif  
## X-squared = 143.33, df = 36, p-value = 9.548e-15

: All autocorrelctions are zero (0), : At least one (1) autocorrelation is not zero (0)

The test indicates one can safely reject the null hypothesis (H0) - such that at least one (1) autocorrleation is not zero in the first difference of the series.

This is also seen through visual inspection of the ACF chart:



The first difference of IIPI is stationary as can be seen in the fast decay in autocorrelation. However, it is not white noise due to a variety of significant autocorrelations (those seen beyond the dotted blue line).

# Regression Model

A multiple linear regression is estimated:

Model reduction is performed iteratively, and the VIFs of each iterations is shown below. Ten (10) is the typically accepted subjective value at which a variance inflation factor is considered high. Only one independent variable is removed at each iteration. First GDP then IndPro are removed:

Iteration 1:

|  | x |
| --- | --- |
| TBillClose | 6.449018 |
| GDP | 204.551481 |
| InfRate | 2.595684 |
| IndPro | 106.137107 |
| M2 | 27.685682 |
| UnempRate | 3.117667 |
| Recession | 1.214842 |

Iteration 2:

|  | x |
| --- | --- |
| TBillClose | 5.516964 |
| InfRate | 2.590142 |
| IndPro | 11.576634 |
| M2 | 9.732960 |
| UnempRate | 2.745397 |
| Recession | 1.195094 |

Iteration 3. The VIFs of the variable reduced model are shown below. This model is retained for subsequent analysis:

|  | x |
| --- | --- |
| TBillClose | 5.201603 |
| InfRate | 2.590000 |
| M2 | 3.246820 |
| UnempRate | 1.034536 |
| Recession | 1.186641 |

Due to the multicollinearities the regression model has been reduced to the following:

### Model Fit Plot

The model is fit to the first 423 observations, then ten test predictions are estimated (after the blue dotted line):

Graphical user interface, chart

Description automatically generated

*Regression Model Fit*

### Residual Assumptions

Before proceeding with model interpretation the regression residual assumptions must be verified:

1. (zero mean assumption)
2. for all (constant variance assumption)
3. for all (normality assumption)
4. for (error terms are not autocorrelated along the time dimension)

#### Residual Normality

The Shapiro-Wilk Test is performed to verify residual normality:

##   
## Shapiro-Wilk normality test  
##   
## data: rstandard(mlm.ts2)  
## W = 0.97691, p-value = 2.931e-06

: Series are normally distributed. vs. : Series are not normally distributed

The resulting p-value is less than 0.05, therefore is rejected, and one must conclude that the series are not normally distributed.

The residuals are therefore not white noise. The other assumptions are verified to ensure fully rigorous analysis, however are technically unnecessary given this result.

#### Residual Autocorrelation

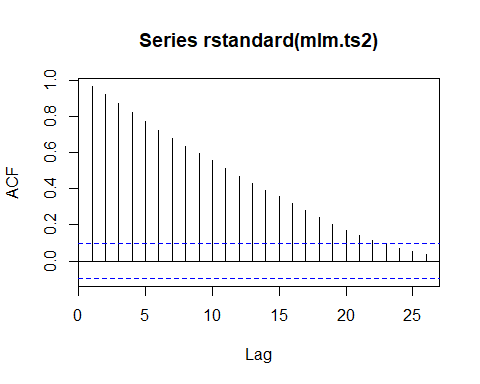
The Box-Pierce test is performed to identify if the residuals are autocorrelated:

##   
## Box-Pierce test  
##   
## data: rstandard(mlm.ts2)  
## X-squared = 3032.8, df = 36, p-value < 2.2e-16

: (all autocorrelations are zero). vs. : at least one (at least one autocorrealtion is not zero)

The residuals are highly autocorrelated and non-stationary. This model can be improved greatly, as there still appears to be time series information not captured in this model.

The resulting p-value is less than 0.05, therefore is rejected, and one must conclude that at least one lag value returns a statistically significant autocorrelation, as seen below:



#### Constant Variance

White’s test for heteroskedasticity is performed to confirm constant variance:

## # A tibble: 1 x 5  
## statistic p.value parameter method alternative  
## <dbl> <dbl> <dbl> <chr> <chr>   
## 1 127. 1.27e-17 20 White's Test greater

: There is no Heteroskedasticity (constant variance) vs. : There is Heteroskedasticity (no constant variance)

The resulting p-value is less than 0.05, therefore is rejected, and one must conclude that there exists non-constant variance within the residuals.

### Regression Residual Summary

All the residual assumptions failed for this initial regression. The residuals are not white noise. Subsequent methods and analysis will be required to extract inherent non-stochastic information presented in the IIPI series.

### Model Summary

##   
## Call:  
## lm(formula = IIPI ~ ., data = df.ts.modelset.train %>% select(-c(Date\_form,   
## GDP, IndPro)))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -46.572 -12.036 -2.934 11.558 64.760   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.568e-01 5.909e+00 -0.027 0.979   
## TBillClose 3.528e-01 5.830e-01 0.605 0.545   
## InfRate 3.607e+00 5.887e-01 6.128 2.06e-09 \*\*\*  
## M2 1.384e-02 5.912e-04 23.416 < 2e-16 \*\*\*  
## UnempRate 2.254e+00 5.697e-01 3.956 8.95e-05 \*\*\*  
## Recession -1.210e+01 2.933e+00 -4.124 4.49e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 18.77 on 417 degrees of freedom  
## Multiple R-squared: 0.7733, Adjusted R-squared: 0.7706   
## F-statistic: 284.5 on 5 and 417 DF, p-value: < 2.2e-16

If the residual assumptions were found to be true one could interpret the coefficients displayed above. All p-values indicate significance, except the intercept and TBillClose (though the residual distribution assumptions fail making interpretation dubious).

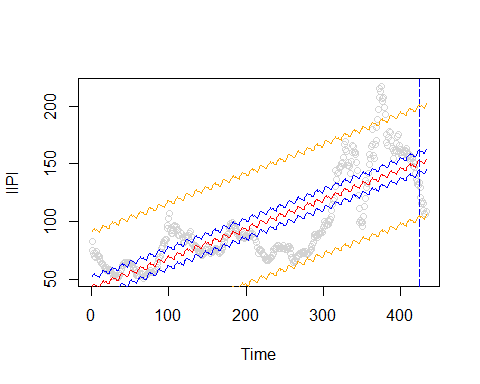
* The expected value of IIPI at time zero (Feb, 1980) is -0.1567672.
* The expected change in the value of IIPI, ceteris paribus, for each unit change in TBillClose is 0.3527733.
* The expected change in the value of IIPI, ceteris paribus, for each unit change in InfRate is 3.6073808.
* The expected change in the value of IIPI, ceteris paribus, for each unit change in M2 is 0.0138443.
* The expected change in the value of IIPI, ceteris paribus, for each unit change in UnempRate is 2.2538474.
* The expected change in the value of IIPI, ceteris paribus, for each unit change in Recession is -12.095765.

# Deterministic Models

## Indicator Variable Regression Model with Trend

A month indicator variable linear regression is estimated, correcting for linear time trend:

### Model Fit Plot



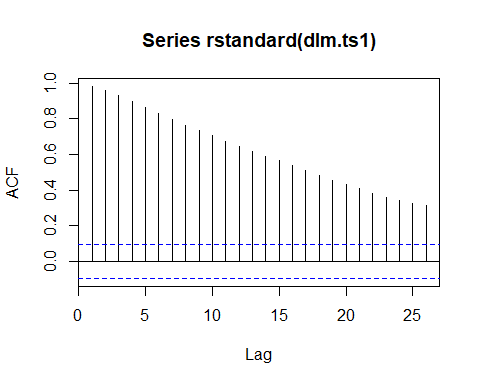
*Indicator Variable Regression*

### Residual Autocorrelation

The Box-Pierce test is performed to identify if the residuals are autocorrelated (see hypothesis definition above):

##   
## Box-Pierce test  
##   
## data: rstandard(dlm.ts1)  
## X-squared = 4374.3, df = 20, p-value < 2.2e-16

Again, the residuals are highly autoccorrelated and non-stationary.



### Model Summary

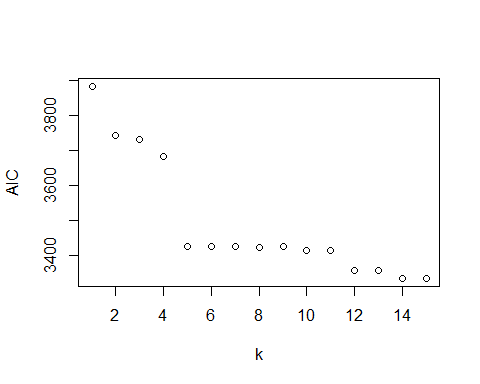
##   
## Call:  
## lm(formula = IIPI ~ index + Month, data = df.ts.modelset.deterministic.train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -44.781 -15.685 0.234 16.182 77.000   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 41.489568 4.549852 9.119 <2e-16 \*\*\*  
## index 0.255417 0.009554 26.735 <2e-16 \*\*\*  
## Month2 1.782036 5.694805 0.313 0.754   
## Month3 2.215359 5.694733 0.389 0.697   
## Month4 3.329823 5.694677 0.585 0.559   
## Month5 3.414111 5.735076 0.595 0.552   
## Month6 1.947270 5.734957 0.340 0.734   
## Month7 1.790995 5.734854 0.312 0.755   
## Month8 1.596455 5.734766 0.278 0.781   
## Month9 0.603795 5.734694 0.105 0.916   
## Month10 -0.405964 5.734639 -0.071 0.944   
## Month11 -0.857836 5.734599 -0.150 0.881   
## Month12 -1.290996 5.734575 -0.225 0.822   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 23.99 on 410 degrees of freedom  
## Multiple R-squared: 0.6358, Adjusted R-squared: 0.6251   
## F-statistic: 59.64 on 12 and 410 DF, p-value: < 2.2e-16

None of the month indicator variables are significant.

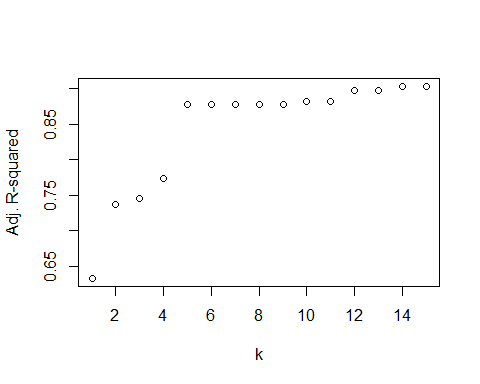
## Polynomial Model

A polynomial linear regression is estimated:

was found by assessment of the adjusted R-squared values and AIC values of prospective models:



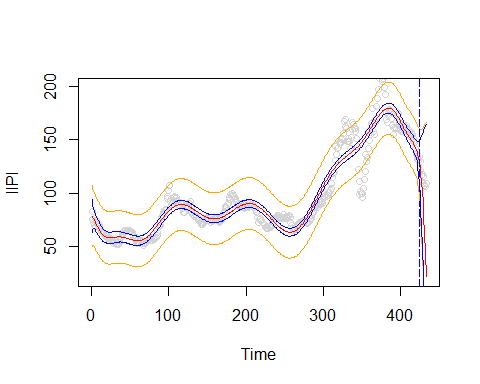
*Polynomial k Search - AIC*



*Polynomial k Search - Adj. R-squared*

The minimum AIC is found to be at .

### Model Fit Plot



*Polynomial Fit, k=14*

The model fits the data well in-sample, but performes poorly within the prediction region.

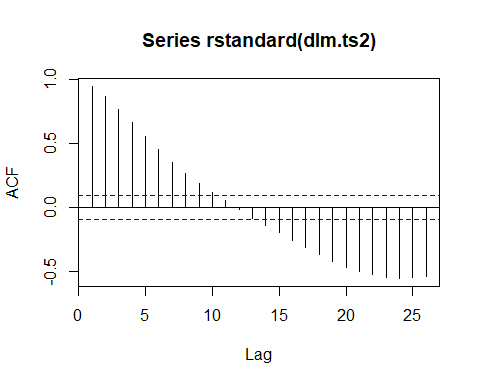
### Residual Assumptions

#### Residual Autocorrelation

The Box-Pierce test is performed to identify if the residuals are autocorrelated (hypothesis outlined above):

##   
## Box-Pierce test  
##   
## data: rstandard(dlm.ts2)  
## X-squared = 1778.1, df = 20, p-value < 2.2e-16

is rejected. Autocorrelation is present in the series.



Again, the residuals are highly autoccorrelated and non-stationary. More work is required to better capture the inherent information contained within the IIPI series.

### Model Summary

##   
## Call:  
## lm(formula = IIPI ~ poly(index, 14), data = df.ts.modelset.deterministic.train)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -52.714 -5.879 -0.781 5.149 41.701   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 96.8240 0.5942 162.948 < 2e-16 \*\*\*  
## poly(index, 14)1 641.0526 12.2209 52.455 < 2e-16 \*\*\*  
## poly(index, 14)2 259.1117 12.2209 21.202 < 2e-16 \*\*\*  
## poly(index, 14)3 76.8761 12.2209 6.291 8.16e-10 \*\*\*  
## poly(index, 14)4 -136.9970 12.2209 -11.210 < 2e-16 \*\*\*  
## poly(index, 14)5 -257.6309 12.2209 -21.081 < 2e-16 \*\*\*  
## poly(index, 14)6 -13.6410 12.2209 -1.116 0.2650   
## poly(index, 14)7 22.0385 12.2209 1.803 0.0721 .   
## poly(index, 14)8 24.5687 12.2209 2.010 0.0450 \*   
## poly(index, 14)9 -3.9927 12.2209 -0.327 0.7441   
## poly(index, 14)10 -49.4098 12.2209 -4.043 6.31e-05 \*\*\*  
## poly(index, 14)11 15.4381 12.2209 1.263 0.2072   
## poly(index, 14)12 99.8887 12.2209 8.174 3.81e-15 \*\*\*  
## poly(index, 14)13 -11.7975 12.2209 -0.965 0.3349   
## poly(index, 14)14 -60.6462 12.2209 -4.963 1.02e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 12.22 on 408 degrees of freedom  
## Multiple R-squared: 0.9059, Adjusted R-squared: 0.9027   
## F-statistic: 280.7 on 14 and 408 DF, p-value: < 2.2e-16

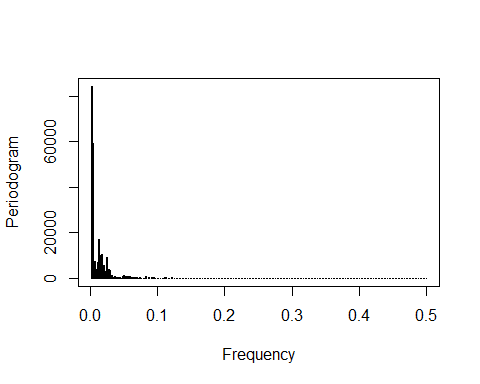
This model fits the in-sample data well but fails in the prediction region. While many of the coefficients in the model would be significant, the residuals are not white noise, therefore their interpretation is not rigorously permitted, and is foregone.

## Harmonic Model

A de-trended harmonic model is estimated:

### Periodogram

To determine the harmonics, a periodogram is reviewed:



*Periodogram*

Unfortunately, the periodogram has such high density at lower harmonics the number of peaks is not readily countable through visual inspection. Therefore, a function is used to create the model harmionic values using the top 100 harmonics identified in the Periodogram. While this is expected to be an over estimate, the model harmonic terms are subsequently removed by evaluating that the sine and cosine pair are both not significant in the model:

listgen <- function(t,j,h,s){  
 sin\_df <- tibble(rep(NA,length(t)))  
 cos\_df <- tibble(rep(NA,length(t)))  
 for (i in 1:j){  
 sinx = sin(2\*pi\*t\*h[i]/length(s))  
 cosx = cos(2\*pi\*t\*h[i]/length(s))  
 sin\_df <- cbind(sin\_df,sinx)  
 cos\_df <- cbind(cos\_df,cosx)  
 }  
 colnames(sin\_df) = c('temp',paste('sin',1:j,sep=""))  
 colnames(cos\_df) = c('temp',paste('cos',1:j,sep=""))  
 sin\_df <- sin\_df %>% select(-temp)  
 cos\_df <- cos\_df %>% select(-temp)  
 df <- cbind(sin\_df,cos\_df)  
 return(df)  
}  
trigf <- listgen(df.ts.modelset.harmonic$index,length(harmonics),harmonics,df.ts.modelset.harmonic$IIPI)

A linear model is estimated with all 200 harmonic terms (one sine and one cosine per harmonic), and those harmonics which have an insignificant (p is greater than 0.05) coefficient terms are displayed:

## [1] "index" "sin1" "sin6" "sin7" "sin3" "sin5" "sin4" "sin14"   
## [9] "sin23" "sin26" "sin17" "sin15" "sin40" "sin57" "sin82" "sin76"   
## [17] "sin55" "sin73" "sin74" "sin47" "sin99" "sin143" "sin93" "sin105"  
## [25] "sin117" "sin100" "sin118" "sin66" "sin95" "sin144" "sin63" "sin87"   
## [33] "sin148" "sin178" "sin103" "sin155" "sin185" "sin188" "sin147" "sin96"   
## [41] "sin160" "sin169" "sin190" "sin189" "cos12" "cos16" "cos18" "cos30"   
## [49] "cos28" "cos17" "cos15" "cos53" "cos44" "cos45" "cos54" "cos61"   
## [57] "cos46" "cos62" "cos72" "cos85" "cos51" "cos50" "cos34" "cos92"   
## [65] "cos98" "cos43" "cos93" "cos71" "cos142" "cos66" "cos116" "cos144"  
## [73] "cos148" "cos178" "cos110" "cos103" "cos78" "cos155" "cos188" "cos147"  
## [81] "cos96" "cos160" "cos169" "cos190" "cos119"

Then those terms which have the same harmonic numeric indicator are removed - ensuring that the full harmonic (both terms) are not significant:

termsforremoval

## [1] "sin17" "sin15" "sin93" "sin66" "sin144" "sin148" "sin178" "sin103"  
## [9] "sin155" "sin188" "sin147" "sin96" "sin160" "sin169" "sin190" "cos17"   
## [17] "cos15" "cos93" "cos66" "cos144" "cos148" "cos178" "cos103" "cos155"  
## [25] "cos188" "cos147" "cos96" "cos160" "cos169" "cos190"

### Model Summary

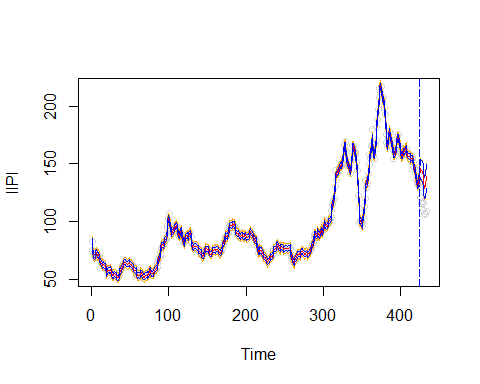
The names of the terms are saved to a vector and subsequently removed from the model. The model is then estimated and dispalyed below:

##   
## Call:  
## lm(formula = IIPI ~ ., data = df.ts.modelset.harmonic.train %>%   
## select(-all\_of(termsforremoval)))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.5927 -0.8850 -0.0845 0.7391 6.6099   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 69.733614 2.993315 23.296 < 2e-16 \*\*\*  
## index 0.129345 0.014112 9.166 < 2e-16 \*\*\*  
## sin1 -14.029071 1.941380 -7.226 5.95e-12 \*\*\*  
## sin2 -23.979111 0.965428 -24.838 < 2e-16 \*\*\*  
## sin6 1.680226 0.308924 5.439 1.27e-07 \*\*\*  
## sin8 -1.833573 0.229273 -7.997 4.68e-14 \*\*\*  
## sin7 2.800089 0.262767 10.656 < 2e-16 \*\*\*  
## sin11 3.327416 0.172984 19.235 < 2e-16 \*\*\*  
## sin3 -3.967231 0.638198 -6.216 2.11e-09 \*\*\*  
## sin5 1.823923 0.374617 4.869 1.99e-06 \*\*\*  
## sin9 -4.298880 0.204598 -21.011 < 2e-16 \*\*\*  
## sin4 -3.358971 0.473620 -7.092 1.34e-11 \*\*\*  
## sin12 2.868092 0.163282 17.565 < 2e-16 \*\*\*  
## sin13 2.343456 0.156362 14.987 < 2e-16 \*\*\*  
## sin10 -2.444362 0.186368 -13.116 < 2e-16 \*\*\*  
## sin14 -0.461284 0.151511 -3.045 0.002578 \*\*   
## sin22 1.586996 0.140019 11.334 < 2e-16 \*\*\*  
## sin24 -1.450585 0.138500 -10.474 < 2e-16 \*\*\*  
## sin25 -2.292941 0.137720 -16.649 < 2e-16 \*\*\*  
## sin23 0.033867 0.139270 0.243 0.808070   
## sin36 1.593236 0.134458 11.849 < 2e-16 \*\*\*  
## sin26 -0.273138 0.136957 -1.994 0.047197 \*   
## sin16 -2.547570 0.145862 -17.466 < 2e-16 \*\*\*  
## sin21 1.048587 0.140743 7.450 1.50e-12 \*\*\*  
## sin18 0.736184 0.143120 5.144 5.43e-07 \*\*\*  
## sin19 -0.803460 0.142229 -5.649 4.36e-08 \*\*\*  
## sin29 -0.435802 0.135028 -3.227 0.001415 \*\*   
## sin30 -1.712003 0.134586 -12.720 < 2e-16 \*\*\*  
## sin28 0.918456 0.135584 6.774 8.85e-11 \*\*\*  
## sin27 0.840485 0.136235 6.169 2.73e-09 \*\*\*  
## sin32 0.541408 0.134087 4.038 7.17e-05 \*\*\*  
## sin41 0.784835 0.135674 5.785 2.15e-08 \*\*\*  
## sin38 -0.931774 0.135006 -6.902 4.18e-11 \*\*\*  
## sin48 0.525152 0.134214 3.913 0.000118 \*\*\*  
## sin49 0.259170 0.133654 1.939 0.053609 .   
## sin40 -0.260521 0.135506 -1.923 0.055664 .   
## sin53 0.878337 0.131095 6.700 1.36e-10 \*\*\*  
## sin39 -0.933828 0.135275 -6.903 4.14e-11 \*\*\*  
## sin31 -0.638533 0.134271 -4.756 3.34e-06 \*\*\*  
## sin44 0.688065 0.135645 5.073 7.64e-07 \*\*\*  
## sin45 0.566032 0.135432 4.179 4.04e-05 \*\*\*  
## sin35 0.524884 0.134235 3.910 0.000119 \*\*\*  
## sin56 0.271782 0.129321 2.102 0.036584 \*   
## sin37 -0.433864 0.134724 -3.220 0.001449 \*\*   
## sin54 0.550190 0.130465 4.217 3.46e-05 \*\*\*  
## sin20 -0.586267 0.141464 -4.144 4.66e-05 \*\*\*  
## sin61 0.622944 0.127400 4.890 1.80e-06 \*\*\*  
## sin42 0.331910 0.135761 2.445 0.015181 \*   
## sin46 0.339141 0.135116 2.510 0.012703 \*   
## sin57 0.086031 0.128824 0.668 0.504863   
## sin86 0.453122 0.125291 3.617 0.000361 \*\*\*  
## sin82 -0.003802 0.125139 -0.030 0.975787   
## sin62 0.411830 0.127168 3.238 0.001364 \*\*   
## sin81 0.416918 0.125121 3.332 0.000992 \*\*\*  
## sin72 0.360850 0.125748 2.870 0.004460 \*\*   
## sin80 0.273314 0.125119 2.184 0.029856 \*   
## sin76 0.076864 0.125304 0.613 0.540154   
## sin55 0.119708 0.129870 0.922 0.357543   
## sin85 0.431349 0.125250 3.444 0.000672 \*\*\*  
## sin51 -0.545172 0.132401 -4.118 5.20e-05 \*\*\*  
## sin84 -0.264671 0.125208 -2.114 0.035515 \*   
## sin50 -0.738959 0.133044 -5.554 7.09e-08 \*\*\*  
## sin73 -0.204430 0.125623 -1.627 0.104921   
## sin34 -0.624113 0.134084 -4.655 5.26e-06 \*\*\*  
## sin74 -0.199346 0.125504 -1.588 0.113464   
## sin92 0.355486 0.125310 2.837 0.004928 \*\*   
## sin47 0.138432 0.134707 1.028 0.305100   
## sin98 0.279521 0.124875 2.238 0.026072 \*   
## sin99 0.124988 0.124793 1.002 0.317519   
## sin43 0.236580 0.135754 1.743 0.082609 .   
## sin143 0.084354 0.123712 0.682 0.495956   
## sin105 0.184618 0.124457 1.483 0.139226   
## sin117 0.172466 0.124073 1.390 0.165751   
## sin67 -0.284702 0.126391 -2.253 0.025152 \*   
## sin71 0.355666 0.125878 2.825 0.005101 \*\*   
## sin142 0.269450 0.123742 2.178 0.030374 \*   
## sin100 -0.071327 0.124716 -0.572 0.567890   
## sin118 -0.151727 0.124025 -1.223 0.222342   
## sin116 0.228746 0.124122 1.843 0.066522 .   
## sin95 0.122607 0.125125 0.980 0.328091   
## sin63 -0.168549 0.126972 -1.327 0.185568   
## sin87 0.022003 0.125326 0.176 0.860779   
## sin110 0.225270 0.124343 1.812 0.071231 .   
## sin78 0.274493 0.125171 2.193 0.029230 \*   
## sin185 -0.045381 0.123566 -0.367 0.713732   
## sin119 -0.323249 0.123979 -2.607 0.009673 \*\*   
## sin189 0.013263 0.123434 0.107 0.914518   
## cos1 18.713902 0.197993 94.518 < 2e-16 \*\*\*  
## cos2 -6.152902 0.197063 -31.223 < 2e-16 \*\*\*  
## cos6 -8.002657 0.187701 -42.635 < 2e-16 \*\*\*  
## cos8 6.454197 0.180326 35.792 < 2e-16 \*\*\*  
## cos7 4.244849 0.184184 23.047 < 2e-16 \*\*\*  
## cos11 -4.937828 0.167540 -29.473 < 2e-16 \*\*\*  
## cos3 -6.046799 0.195533 -30.925 < 2e-16 \*\*\*  
## cos5 -3.244000 0.190805 -17.002 < 2e-16 \*\*\*  
## cos9 3.662915 0.176208 20.788 < 2e-16 \*\*\*  
## cos4 3.598936 0.193433 18.606 < 2e-16 \*\*\*  
## cos12 0.720405 0.163176 4.415 1.50e-05 \*\*\*  
## cos13 1.462370 0.158917 9.202 < 2e-16 \*\*\*  
## cos10 -4.185963 0.171915 -24.349 < 2e-16 \*\*\*  
## cos14 2.135403 0.154851 13.790 < 2e-16 \*\*\*  
## cos22 0.765701 0.136058 5.628 4.87e-08 \*\*\*  
## cos24 1.582953 0.135313 11.698 < 2e-16 \*\*\*  
## cos25 -1.246895 0.135290 -9.216 < 2e-16 \*\*\*  
## cos23 1.800301 0.135547 13.282 < 2e-16 \*\*\*  
## cos36 -0.372101 0.134527 -2.766 0.006097 \*\*   
## cos26 -2.084147 0.135409 -15.391 < 2e-16 \*\*\*  
## cos16 -0.381194 0.147619 -2.582 0.010382 \*   
## cos21 -1.028466 0.136911 -7.512 1.02e-12 \*\*\*  
## cos18 0.380282 0.141981 2.678 0.007886 \*\*   
## cos19 1.154559 0.139841 8.256 8.63e-15 \*\*\*  
## cos29 1.383411 0.136011 10.171 < 2e-16 \*\*\*  
## cos30 0.072733 0.136130 0.534 0.593616   
## cos28 0.104043 0.135823 0.766 0.444385   
## cos27 -0.425611 0.135606 -3.139 0.001901 \*\*   
## cos32 -0.868523 0.136061 -6.383 8.32e-10 \*\*\*  
## cos41 -0.595277 0.131237 -4.536 8.89e-06 \*\*\*  
## cos38 0.609112 0.133253 4.571 7.62e-06 \*\*\*  
## cos48 0.637318 0.128595 4.956 1.32e-06 \*\*\*  
## cos49 0.633520 0.128507 4.930 1.50e-06 \*\*\*  
## cos40 -0.932727 0.131889 -7.072 1.51e-11 \*\*\*  
## cos53 -0.068806 0.128441 -0.536 0.592639   
## cos39 -0.566425 0.132570 -4.273 2.74e-05 \*\*\*  
## cos31 -0.834834 0.136153 -6.132 3.36e-09 \*\*\*  
## cos44 0.125896 0.129632 0.971 0.332394   
## cos45 -0.033233 0.129253 -0.257 0.797303   
## cos35 -0.469877 0.135067 -3.479 0.000594 \*\*\*  
## cos56 0.613718 0.128305 4.783 2.94e-06 \*\*\*  
## cos37 0.777359 0.133914 5.805 1.94e-08 \*\*\*  
## cos54 0.084090 0.128422 0.655 0.513199   
## cos20 -0.971437 0.138159 -7.031 1.93e-11 \*\*\*  
## cos61 0.234238 0.127436 1.838 0.067233 .   
## cos42 -0.332994 0.130633 -2.549 0.011397 \*   
## cos46 -0.119955 0.128958 -0.930 0.353167   
## cos57 0.439586 0.128196 3.429 0.000708 \*\*\*  
## cos86 0.255807 0.125083 2.045 0.041888 \*   
## cos82 0.612796 0.125364 4.888 1.82e-06 \*\*\*  
## cos62 0.081872 0.127193 0.644 0.520370   
## cos81 0.297556 0.125418 2.373 0.018422 \*   
## cos72 0.089035 0.125566 0.709 0.478939   
## cos80 -0.255766 0.125460 -2.039 0.042536 \*   
## cos76 -0.559377 0.125523 -4.456 1.26e-05 \*\*\*  
## cos55 0.508469 0.128379 3.961 9.74e-05 \*\*\*  
## cos85 -0.152116 0.125157 -1.215 0.225353   
## cos51 -0.082759 0.128448 -0.644 0.519970   
## cos84 -0.416111 0.125230 -3.323 0.001024 \*\*   
## cos50 -0.035626 0.128463 -0.277 0.781762   
## cos73 0.559856 0.125539 4.460 1.24e-05 \*\*\*  
## cos34 -0.104045 0.135512 -0.768 0.443333   
## cos74 0.385497 0.125528 3.071 0.002368 \*\*   
## cos92 0.027097 0.124790 0.217 0.828276   
## cos47 0.427921 0.128741 3.324 0.001020 \*\*   
## cos98 -0.060032 0.124783 -0.481 0.630872   
## cos99 0.283235 0.124779 2.270 0.024062 \*   
## cos43 -0.137472 0.130094 -1.057 0.291659   
## cos143 0.237812 0.123967 1.918 0.056202 .   
## cos105 0.241217 0.124576 1.936 0.053952 .   
## cos117 0.290486 0.124163 2.340 0.020092 \*   
## cos67 0.308068 0.126068 2.444 0.015228 \*   
## cos71 0.040926 0.125611 0.326 0.744838   
## cos142 0.161283 0.123960 1.301 0.194423   
## cos100 0.227429 0.124767 1.823 0.069519 .   
## cos118 0.403875 0.124173 3.253 0.001301 \*\*   
## cos116 -0.063375 0.124157 -0.510 0.610189   
## cos95 0.230928 0.124773 1.851 0.065376 .   
## cos63 0.327125 0.126945 2.577 0.010541 \*   
## cos87 0.248062 0.125012 1.984 0.048311 \*   
## cos110 -0.105289 0.124286 -0.847 0.397719   
## cos78 0.151073 0.125511 1.204 0.229852   
## cos185 0.340628 0.123933 2.748 0.006422 \*\*   
## cos119 0.166942 0.124185 1.344 0.180062   
## cos189 0.250463 0.124037 2.019 0.044523 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.798 on 251 degrees of freedom  
## Multiple R-squared: 0.9987, Adjusted R-squared: 0.9979   
## F-statistic: 1171 on 171 and 251 DF, p-value: < 2.2e-16

The model contains 85 harmonics, all sine cosine pairs are significant (due in many cases to the other respective term being significant).

### Model Fit Plot

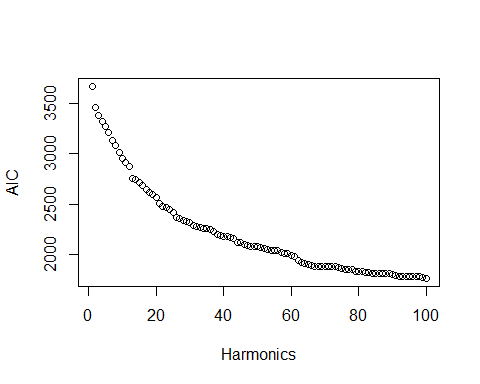
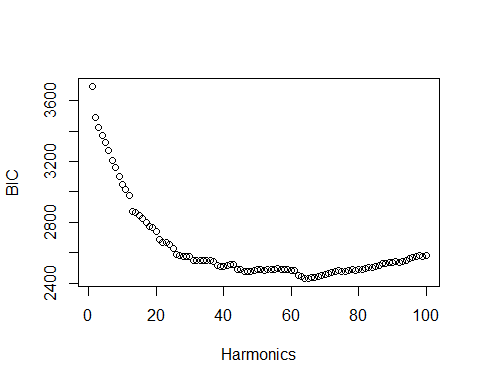
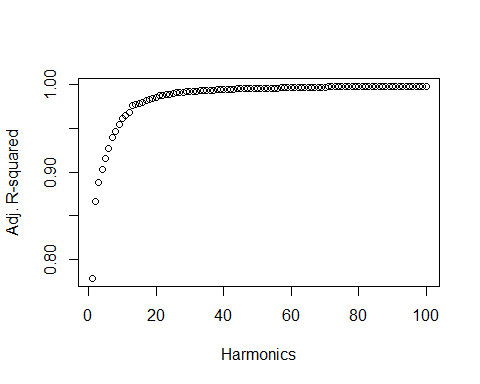
This initial model with 85 harmonics significantly overfits the data.



*Harmonic Model with 85 harmonics*

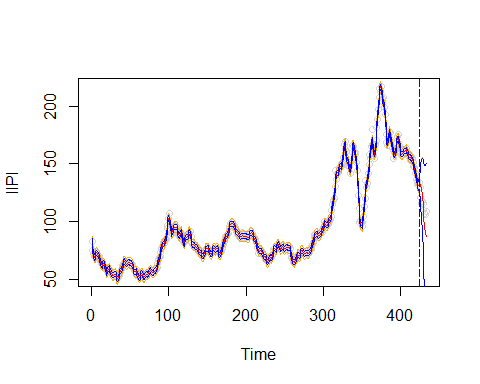
The model with 85 terms fits the in-sample data nearly perfectly (Adjusted R-squared: 0.9979). However, due to failure in the prediction region harmonic term reduction is investigated to assess if the predictive utility can be improved.

The three in-sample metrics are displayed as a function of increasing harmonics:



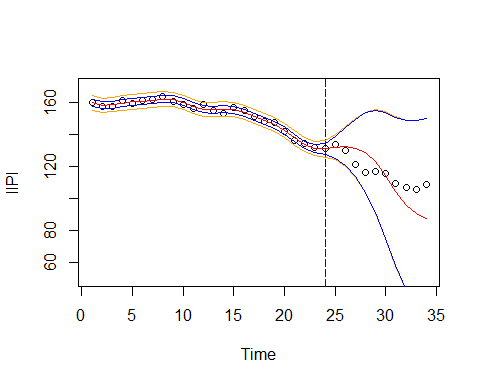
*Harmonics vs. Fit Metrics*

AIC does not minimize in this region and adjusted R-squared continues to increase. BIC has a minimum at 65 harmonics. The following is the fit with 65 harmonics:



*Harmonic Model with 65 harmonics*

And the prediction region zoomed in for clarity:



*Harmonic Model with 65 harmonics*

The 65 harmonic model appears to still overfit the data, therefore instead in an effort to balance the bias-variance trade-off the visual elbow method is used with the adjusted R-squared metric.

Visual inspection provides that there are about 15 harmonics before there is a reduced rate of increase in adjusted R-squared. Therefore the 15 harmonic model is estimated:

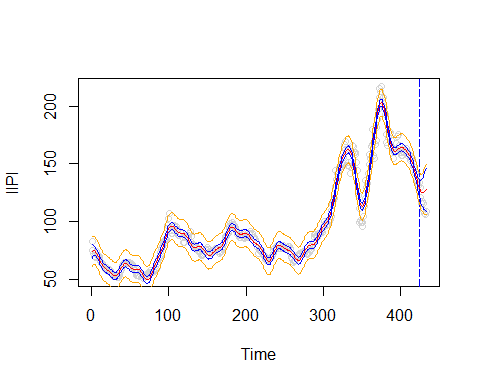
##   
## Call:  
## lm(formula = IIPI ~ ., data = df.ts.modelset.harmonic.train1)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -21.4227 -2.4675 0.0402 2.3339 22.8161   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 70.35785 5.85544 12.016 < 2e-16 \*\*\*  
## index 0.12513 0.02759 4.536 7.65e-06 \*\*\*  
## sin1 -14.56573 3.80832 -3.825 0.000152 \*\*\*  
## sin2 -24.18113 1.91348 -12.637 < 2e-16 \*\*\*  
## sin6 1.83559 0.67916 2.703 0.007177 \*\*   
## sin8 -1.58503 0.54342 -2.917 0.003741 \*\*   
## sin7 3.00514 0.59948 5.013 8.14e-07 \*\*\*  
## sin11 3.67601 0.45324 8.111 6.57e-15 \*\*\*  
## sin3 -4.02918 1.28666 -3.132 0.001870 \*\*   
## sin5 1.92168 0.79612 2.414 0.016245 \*   
## sin9 -4.01217 0.50325 -7.972 1.73e-14 \*\*\*  
## sin4 -3.33055 0.97752 -3.407 0.000725 \*\*\*  
## sin12 3.24073 0.43809 7.397 8.56e-13 \*\*\*  
## sin13 2.73564 0.42724 6.403 4.37e-10 \*\*\*  
## sin10 -2.12437 0.47423 -4.480 9.83e-06 \*\*\*  
## sin14 -0.05400 0.41951 -0.129 0.897643   
## sin22 1.96653 0.40158 4.897 1.43e-06 \*\*\*  
## cos1 18.13876 0.48234 37.606 < 2e-16 \*\*\*  
## cos2 -6.72174 0.48113 -13.971 < 2e-16 \*\*\*  
## cos6 -8.50633 0.46902 -18.136 < 2e-16 \*\*\*  
## cos8 6.00447 0.45958 13.065 < 2e-16 \*\*\*  
## cos7 3.76657 0.46451 8.109 6.66e-15 \*\*\*  
## cos11 -5.28618 0.44339 -11.922 < 2e-16 \*\*\*  
## cos3 -6.60520 0.47914 -13.786 < 2e-16 \*\*\*  
## cos5 -3.76964 0.47302 -7.969 1.77e-14 \*\*\*  
## cos9 3.24458 0.45435 7.141 4.55e-12 \*\*\*  
## cos4 3.05499 0.47641 6.412 4.13e-10 \*\*\*  
## cos12 0.40989 0.43789 0.936 0.349819   
## cos13 1.19109 0.43252 2.754 0.006165 \*\*   
## cos10 -4.57039 0.44891 -10.181 < 2e-16 \*\*\*  
## cos14 1.90433 0.42738 4.456 1.09e-05 \*\*\*  
## cos22 0.84354 0.40110 2.103 0.036097 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 5.756 on 391 degrees of freedom  
## Multiple R-squared: 0.98, Adjusted R-squared: 0.9784   
## F-statistic: 618.1 on 31 and 391 DF, p-value: < 2.2e-16

The Adjusted R-squared is still very high in-sample fit, (0.9784).

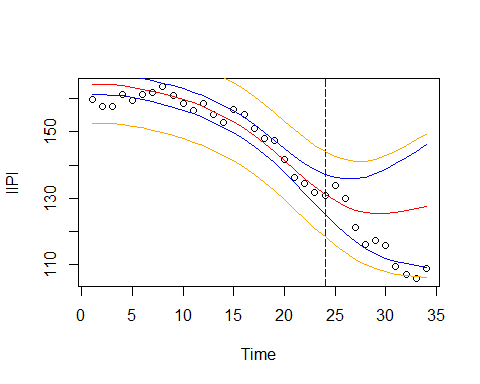
Terms for this model are assessed for pair-wise significance. Only two trigonometric terms (not of the same harmonic) are not significant. Therefore this model is retained for comparison going forward.

## [1] "sin14" "cos12"

The model with 15 harmonics is shown below:



*Harmonic Model with 15 Harmonics*



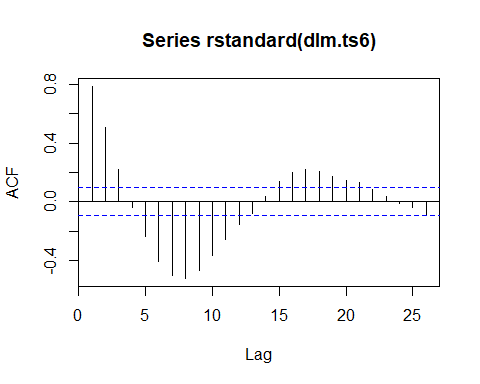
*Harmonic Model with 15 Harmonics - Prediction Region*

### Residual Assumptions (model with 15 harmonics)

#### Residual Autocorrelation

The Box-Pierce test is performed to identify if residuals are autocorrelated for the model with 15 harmonics (hypothesis defined above):

##   
## Box-Pierce test  
##   
## data: rstandard(dlm.ts6)  
## X-squared = 1074, df = 36, p-value < 2.2e-16



The residuals are stationary but not white noise, with fast sinusoidal decay.

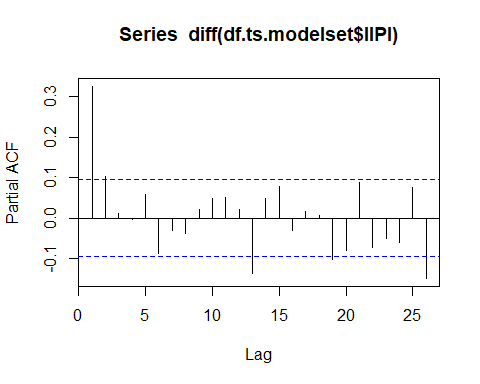
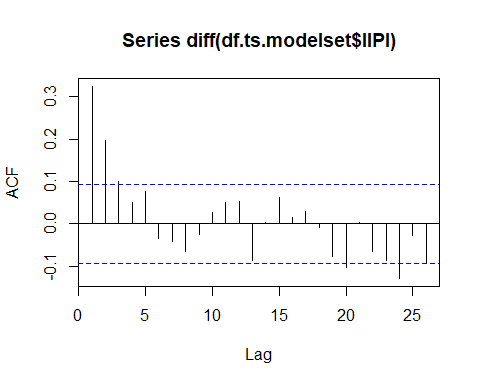
# Stochastic Models

## ARIMA Models

Next, Stochastic Models are estimated.

Before fitting a stochastic model the series must be stationary. IIPI is not stationary, but the first difference results in stationarity with sinusoidal decay, as seen above.

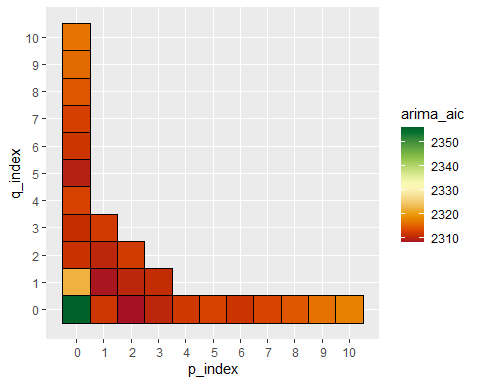
Having achieved stationarity, at the first difference, the ACF and PACF are reviewed to determine appropriate ARIMA model parameters:



In reviewing the ACF and PACF there are three proposed reasonable ARIMA models for investigation:

* ARIMA(2,1,0) due to perceived cutoff of the PACF at 2, and interpretable sinusoidal decay of the ACF.
* ARIMA(0,1,5) due to perceived cutoff of the ACF at 5, and interpretable sinusoidal decay of the PACF.
* ARIMA(x,1,y) due to the perceived exponential decay of both ACF and PACF;

This assessment is validated using a subset search of p and q values (ARIMA(p,1,q)). The resulting model AICs are displayed:



*ARIMA model AIC Heatmap*

The listed models resulted in the lowest AIC values, verifying the intuition and methodological approach:

| p\_index | q\_index | arima\_aic |
| --- | --- | --- |
| 2 | 0 | 2307.985 |
| 1 | 1 | 2308.373 |
| 0 | 5 | 2309.411 |

All three are estimated and reviewed for comparison given their comparative AICs.

##   
## Call:  
## arima(x = df.ts.modelset.train$IIPI, order = c(2, 1, 0))  
##   
## Coefficients:  
## ar1 ar2  
## 0.2874 0.1142  
## s.e. 0.0485 0.0485  
##   
## sigma^2 estimated as 13.76: log likelihood = -1151.99, aic = 2307.98

##   
## Call:  
## arima(x = df.ts.modelset.train$IIPI, order = c(1, 1, 1))  
##   
## Coefficients:  
## ar1 ma1  
## 0.5909 -0.2984  
## s.e. 0.1022 0.1192  
##   
## sigma^2 estimated as 13.77: log likelihood = -1152.19, aic = 2308.37

##   
## Call:  
## arima(x = df.ts.modelset.train$IIPI, order = c(0, 1, 5))  
##   
## Coefficients:  
## ma1 ma2 ma3 ma4 ma5  
## 0.2913 0.2027 0.1083 0.0470 0.1239  
## s.e. 0.0482 0.0508 0.0526 0.0531 0.0544  
##   
## sigma^2 estimated as 13.61: log likelihood = -1149.71, aic = 2309.41

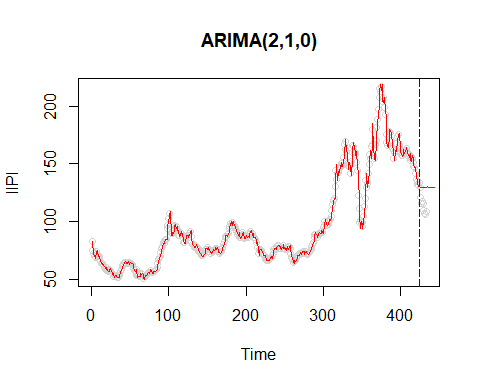
As can be seen above, all coefficients are significant for each model except and of the MA(5) model. Therefore this one is dropped for comparison going forward. It also, of the three models, has the highest AIC.

The ARIMA(2,1,0) and ARIMA(1,1,1) equations follow:

ARIMA(2,1,0):

ARIMA(1,1,1):

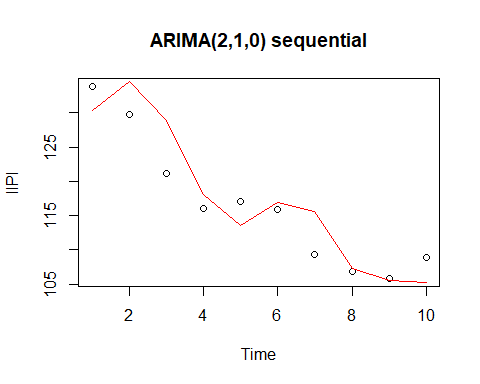
### Model Fit Plots



*ARIMA(2,1,0) Model with 20 step ahead predictions*

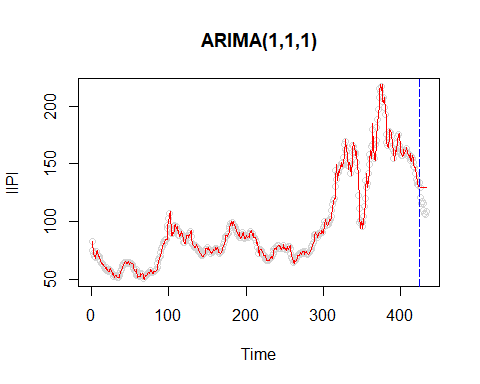
The model performs poorly when used to predict 10 values into the future.

However, when looking at sequential prediction, the model does incredibly well in the prediction region:

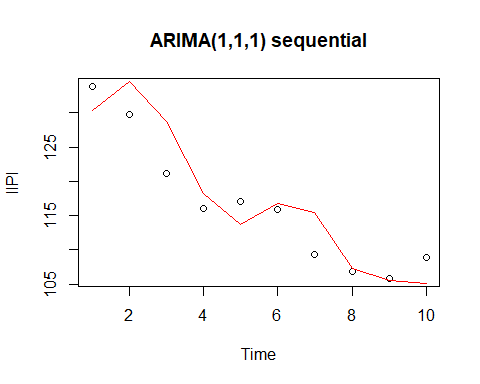


*ARIMA(2,1,0) Model with sequential prediction*

Similarly for ARIMA(1,1,1):



*ARIMA(1,1,1) Model with 20 step ahead predictions*



*ARIMA(1,1,1) Model with sequential prediction*

### Confirming White Noise

The residuals are reviewed for white noise.

##   
## Box-Pierce test  
##   
## data: arima210$residuals  
## X-squared = 28.268, df = 20, p-value = 0.1032

##   
## Box-Pierce test  
##   
## data: arima111$residuals  
## X-squared = 27.886, df = 20, p-value = 0.1121

Both are marginally white noise based on the above Box-Pierce tests. Success! But these models can be improved by stochastic residual correction outlined below.

# Stochastic Residual Corrections

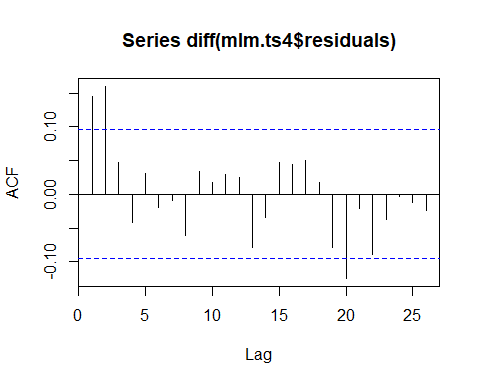
The following models are reviewed in this section to assess residual correction using ARIMA on the respective resulting residuals:

* Regression Model
* Harmonic Model with 15 harmonics

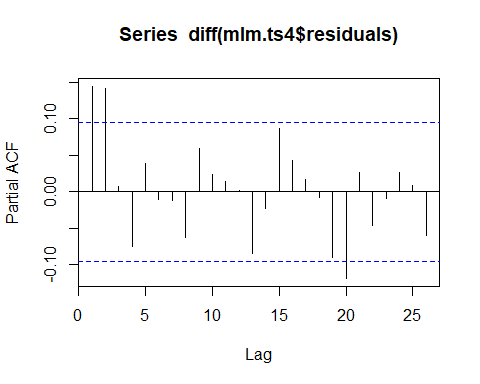
The Polynomial Model with is not included in this section due to computational restrictions in the approximation methods used to estimate ARIMA models. Reciprocals are taken of the various polynomial terms which result in computational singularities. For example at index results in a value greater that . The reciprocal is interpreted as singularity by approximation.

## Stochastic Residual Correction for Regression Model

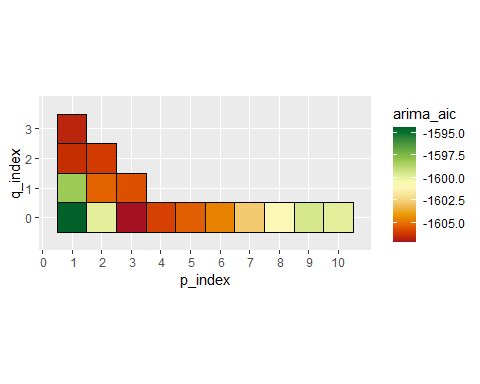
And the residuals of this model were not stationary, therefore the first difference is taken to induce stationarity:



Next the PACF is investigated for ARIMA parameter assignment:



As both appear potentially cuff-off or fast sinusoidal decay a search is again performed for reasonable and values:



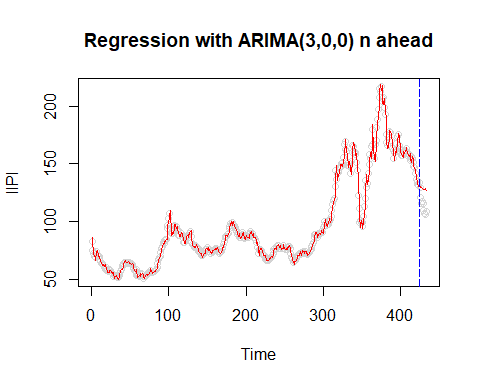
*ARIMA model AIC Heatmap*

As seen in the above chart, the minimum AIC occurs at , . Therefore an ARIMA(3,0,0) is fit to the residuals.

| p\_index | q\_index | arima\_aic |
| --- | --- | --- |
| 3 | 0 | -1607.192 |
| 1 | 3 | -1606.707 |
| 1 | 2 | -1606.460 |

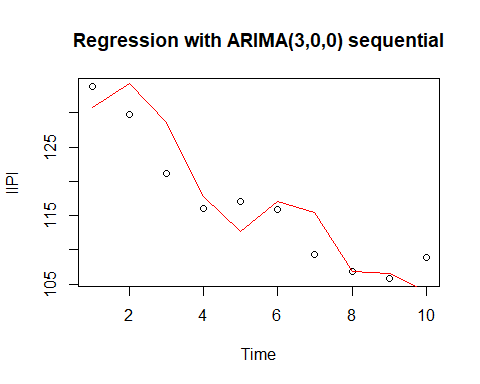
### Model Fit Plot

Again, n ahead performs poorly:



*Regression with ARIMA(3,0,0) n ahead*

However as seen before, when looking at one step ahead prediction values, the prediction clearly performs much better:



*Regression with ARIMA(3,0,0) sequential - prediction region*

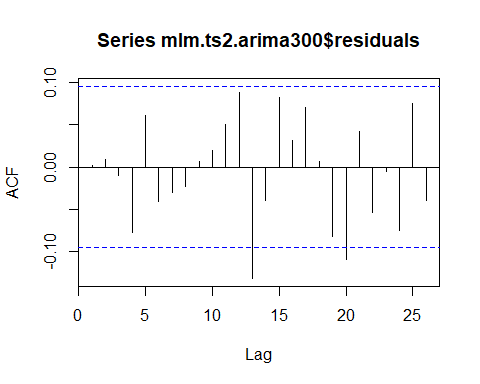
The model is:

### Model Summary

##   
## Call:  
## arima(x = df.ts.modelset.train$IIPI, order = c(3, 0, 0), xreg = df.ts.modelset.train %>%   
## select(-c("Date\_form", "IIPI", "GDP", "IndPro")))  
##   
## Coefficients:  
## ar1 ar2 ar3 intercept TBillClose InfRate M2  
## 1.2594 -0.1352 -0.1282 110.3489 -0.0243 1.9988 -0.0036  
## s.e. 0.0499 0.0802 0.0498 44.8738 0.3770 0.4747 0.0056  
## UnempRate Recession  
## -0.3204 0.7069  
## s.e. 1.0666 1.1781  
##   
## sigma^2 estimated as 13.02: log likelihood = -1145.71, aic = 2309.42

All the regression coefficients become insignificant except the intercept and InfRate.

### Confirming White Noise



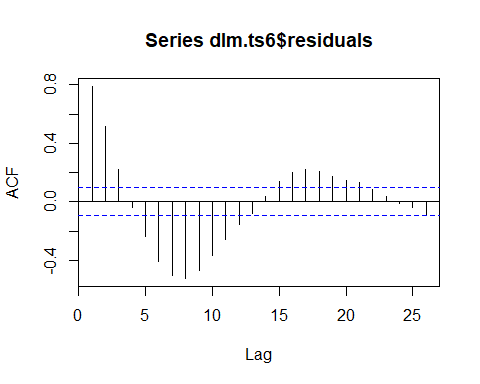
##   
## Box-Pierce test  
##   
## data: mlm.ts2.arima300$residuals  
## X-squared = 31.276, df = 20, p-value = 0.05165

The residuals of the regression model with ARMIA residual correction are white noise.

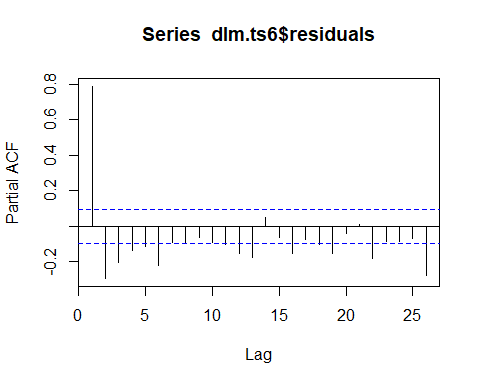
## Stochastic Residual Correction for Harmonic Model

The harmonic model is (:

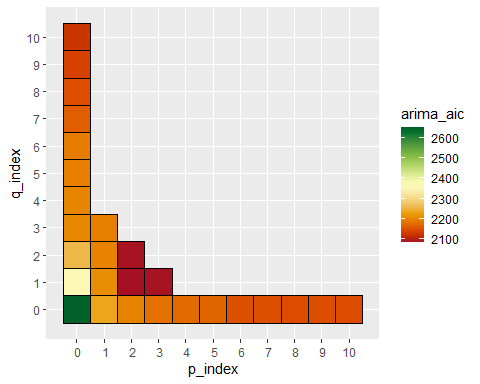
First, as a reminder the residuals were very clearly auto-correlated, but already stationary:



The PACF is investigated to determine the appropriate ARIMA model:



Both the ACF and PACF do not have a clear cut-off, and could be interpreted as showing exponential decay. Expectation is therefore that an ARMA(p,q) with , will likely be applicable for the residual correction.

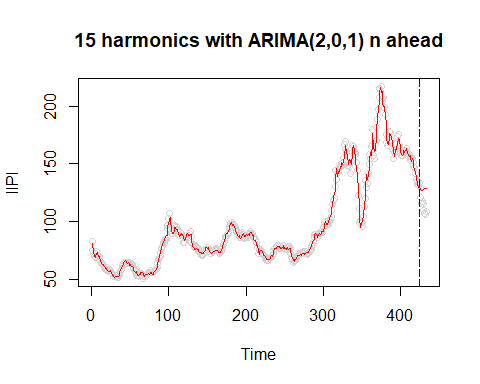


*ARIMA Model AIC Heatmap*

| p\_index | q\_index |  | arima\_aic |
| --- | --- | --- | --- |
| 2 | 1 |  | 2084.413 |
| 2 | 2 |  | 2086.360 |
| 3 | 1 |  | 2086.366 |

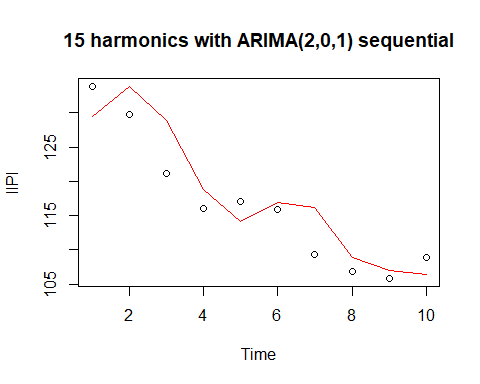
As seen in the above chart ARIMA(2,0,1) best fits the residuals.

### Model Fit Plot



*15 harmonics with ARIMA(2,0,1) n ahead*

Again, as expected the n ahead prediction performs poorly.



*15 harmonics with ARIMA(2,0,1) sequential*

… and the sequential prediction performs well.

Model:

### Model Summary

##   
## Call:  
## arima(x = df.ts.modelset.harmonic.train1$IIPI, order = c(2, 0, 1), xreg = df.ts.modelset.harmonic.train1 %>%   
## select(-c("IIPI")))  
##   
## Coefficients:  
## ar1 ar2 ma1 intercept index sin1 sin2 sin6  
## 1.7269 -0.8581 -1.0000 71.7040 0.1193 -15.3773 -24.5896 1.6908  
## s.e. 0.0247 0.0247 0.0059 7.1503 0.0336 4.6240 2.2943 0.7141  
## sin8 sin7 sin11 sin3 sin5 sin9 sin4 sin12  
## -1.6982 2.8785 3.5886 -4.3044 1.7513 -4.1149 -3.5399 3.1593  
## s.e. 0.5206 0.6013 0.4106 1.5107 0.8745 0.4644 1.1145 0.4091  
## sin13 sin10 sin14 sin22 cos1 cos2 cos6 cos8  
## 2.6596 -2.2186 -0.1261 1.8673 18.2870 -6.5736 -8.3587 6.1522  
## s.e. 0.4224 0.4284 0.4488 1.0322 0.2951 0.2968 0.3163 0.3355  
## cos7 cos11 cos3 cos5 cos9 cos4 cos12 cos13  
## 3.9142 -5.1365 -6.4572 -3.6219 3.3927 3.2029 0.5615 1.3448  
## s.e. 0.3250 0.3818 0.2997 0.3093 0.3482 0.3038 0.4038 0.4302  
## cos10 cos14 cos22  
## -4.4217 2.0611 1.0904  
## s.e. 0.3635 0.4617 1.0429  
##   
## sigma^2 estimated as 7.766: log likelihood = -1037.05, aic = 2144.1

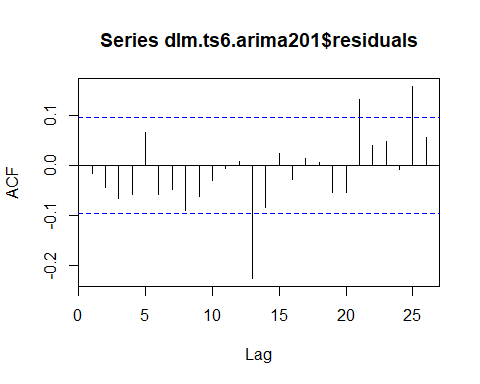
Only one pair of harmonic terms (sin22 and cos22) have been made insignificant by this addition.

The AIC has been improved by correcting the residuals:

Harmonic Model: 2713.8738679

Harmonic Model with residual ARIMA correction: 2144.1014833

### Confirming White Noise



##   
## Box-Pierce test  
##   
## data: dlm.ts6.arima201$residuals  
## X-squared = 41.873, df = 20, p-value = 0.002873

Despite the correction, residuals of the model do not result in white noise.

## Generalized Autoregressive conditionally Heteroskedastic Model (GARCH)

The following section investigates using GARCH. First, as shown before the series and its first difference are not white noise and therefore using GARCH on its own is not applicable. However, many of the previous models resulted in white noise residuals, and are investigated here.

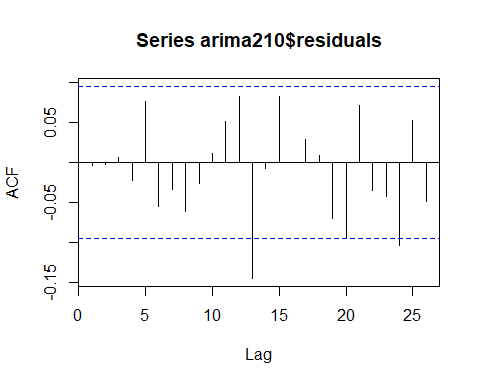
The following models thus far have resulted in white noise residuals making them candidates for assessing utilization of GARCH:

* ARIMA(2,1,0)
* ARIMA(1,1,1)
* Regression with ARIMA(3,0,0) on the residuals
* ARIMA(0,1,5) – not included here for brevity

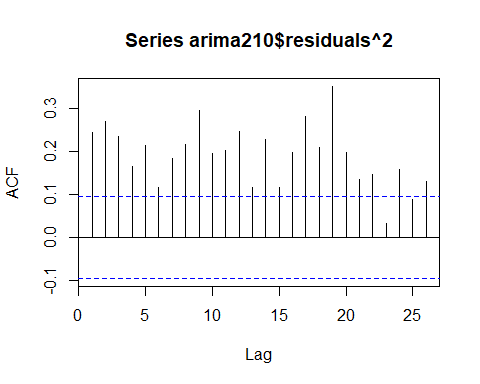
## ARIMA Models with (GARCH)

### ARIMA(2,1,0) with GARCH

Residuals are first checked for GARCH model potential:



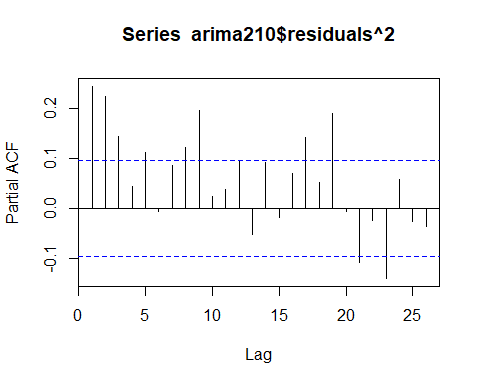
##   
## Box-Pierce test  
##   
## data: arima210$residuals  
## X-squared = 28.268, df = 20, p-value = 0.1032



##   
## Box-Pierce test  
##   
## data: arima210$residuals^2  
## X-squared = 415.36, df = 20, p-value < 2.2e-16

As can be seen the original ARIMA(2,1,0) model residuals are white noise, but the squared residuals are not. This is an excellent candidate for using GARCH!

The PACF is displayed to investigate the possible GARCH order.



### Model Summary

As a model was not immediately obvious from the ACF and PACF, a appropriate GARCH model was found by estimating a diversity of GARCH parameters using the following search code:

p\_index <- list()  
q\_index <- list()  
d <- diff(df.ts.modelset.train$IIPI)  
garch\_list\_2 <-   
 foreach(i=1:2) %:%  
 foreach(j=0:2) %do% {  
 garch\_list<-garchFit(substitute(~arma(2,0)+garch(p,q),list(p=i,q=j)),data=d, trace=FALSE, include.mean = FALSE)   
 }  
  
garch\_list\_2

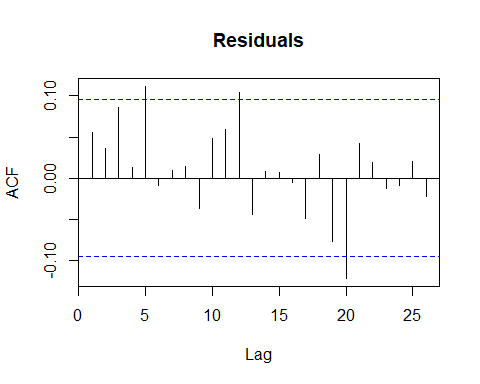
The output is not included here for brevity, but was reviewed for best AIC, resulting in: ARIMA(2,1,0) with GARCH(1,1):

##   
## Title:  
## GARCH Modelling   
##   
## Call:  
## garchFit(formula = ~arma(2, 0) + garch(1, 1), data = diff(df.ts.modelset.train$IIPI),   
## include.mean = FALSE, trace = FALSE)   
##   
## Mean and Variance Equation:  
## data ~ arma(2, 0) + garch(1, 1)  
## <environment: 0x00000000151e4d00>  
## [data = diff(df.ts.modelset.train$IIPI)]  
##   
## Conditional Distribution:  
## norm   
##   
## Coefficient(s):  
## ar1 ar2 omega alpha1 beta1   
## 0.20651 0.11658 0.18619 0.19309 0.80473   
##   
## Std. Errors:  
## based on Hessian   
##   
## Error Analysis:  
## Estimate Std. Error t value Pr(>|t|)   
## ar1 0.20651 0.05502 3.754 0.000174 \*\*\*  
## ar2 0.11658 0.05184 2.249 0.024515 \*   
## omega 0.18619 0.08213 2.267 0.023384 \*   
## alpha1 0.19309 0.04190 4.608 4.06e-06 \*\*\*  
## beta1 0.80473 0.03506 22.956 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Log Likelihood:  
## -1009.791 normalized: -2.39287   
##   
## Description:  
## Sat Mar 05 17:30:59 2022 by user: nptho

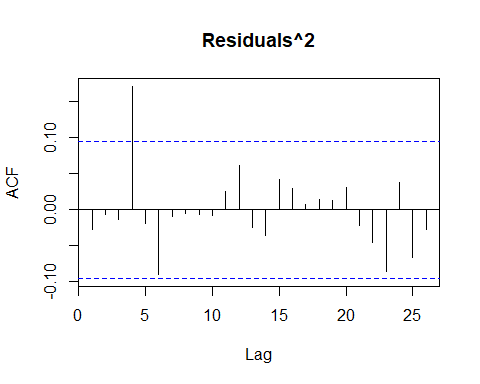
The mean is removed from the model and all model parameters are significant.

Model:

### White noise confirmation



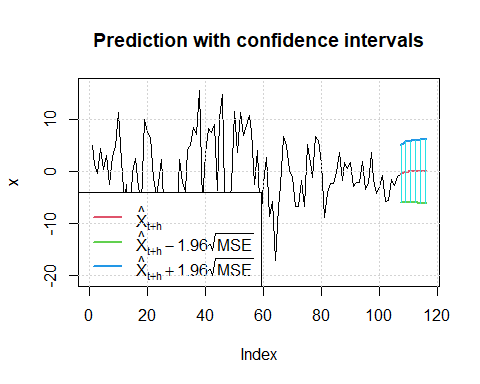
##   
## Box-Pierce test  
##   
## data: arima210.garch11@residuals/arima210.garch11@sigma.t  
## X-squared = 29.041, df = 20, p-value = 0.08696



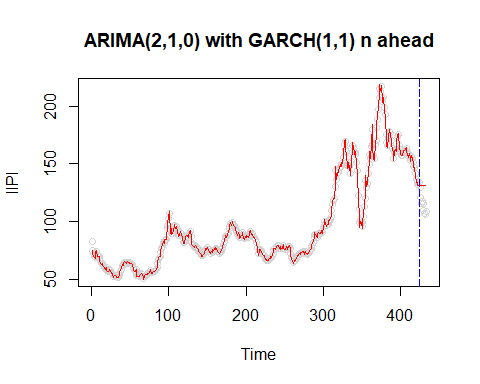
##   
## Box-Pierce test  
##   
## data: (arima210.garch11@residuals/arima210.garch11@sigma.t)^2  
## X-squared = 20.888, df = 20, p-value = 0.4038

The residuals and squared residuals of this model are white noise!

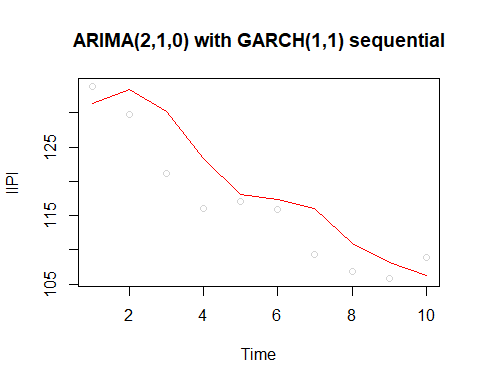
The following is the predicted conditional variance of the IIPI first-difference:



As expected the n ahead prediction values tend toward the mean asymptotically:



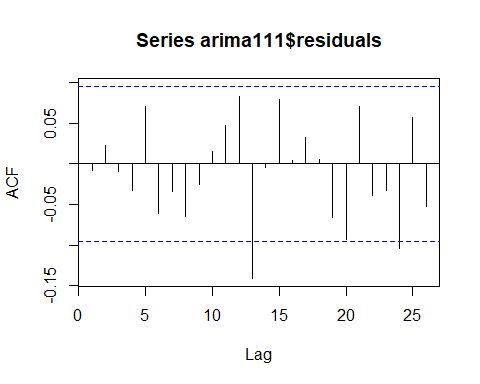
When taking the predictions 1 step ahead, the model predicts well:



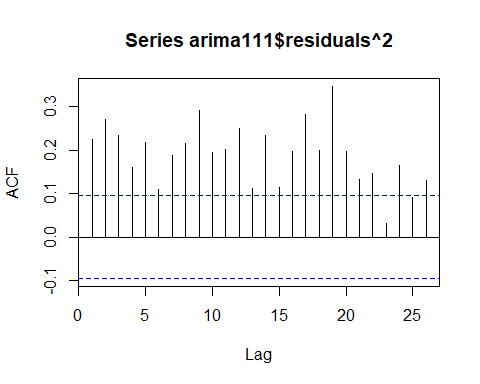
This result can be seen in the MAPE comparison summarised in the Prediction Comparison section below.

### ARIMA(1,1,1) with GARCH

Residuals are first to checked for GARCH model potential:



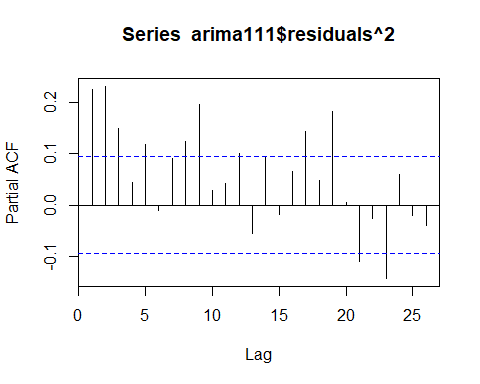
##   
## Box-Pierce test  
##   
## data: arima111$residuals  
## X-squared = 27.886, df = 20, p-value = 0.1121



##   
## Box-Pierce test  
##   
## data: arima111$residuals^2  
## X-squared = 411.27, df = 20, p-value < 2.2e-16

As can be seen, again similar to the ARIMA(2,1,0) model, the ARIMA(1,1,1) residuals are white noise, but the squared residuals are not. This is an excellent candidate for using GARCH, again!

The PACF is displayed to investigate the possible GARCH order.



### Model Summary

As a model was not immediately obvious from the ACF and PACF, a appropriate GARCH model was found by estimating a diversity of GARCH parameters using the following search code:

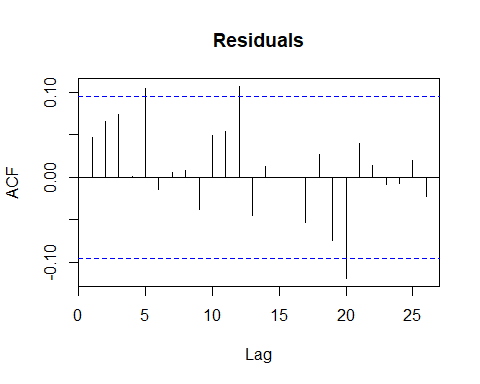
p\_index <- list()  
q\_index <- list()  
d <- diff(df.ts.modelset.train$IIPI)  
garch\_list\_2 <-   
 foreach(i=1:2) %:%  
 foreach(j=0:2) %do% {  
 garch\_list<-garchFit(substitute(~arma(1,1)+garch(p,q),list(p=i,q=j)),data=d, trace=FALSE, include.mean = FALSE)   
 }  
  
garch\_list\_2

The output is not included here for brevity, but was reviewed for best AIC, resulting in: ARIMA(1,1,1) with GARCH(1,1)

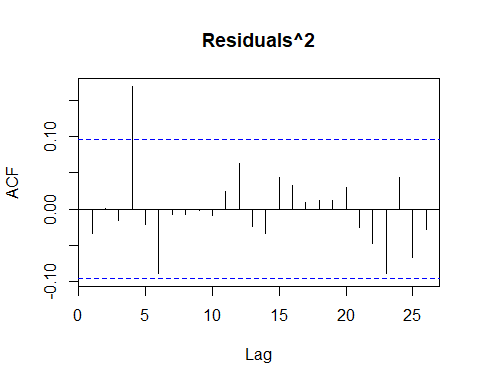
##   
## Title:  
## GARCH Modelling   
##   
## Call:  
## garchFit(formula = ~arma(1, 1) + garch(1, 1), data = diff(df.ts.modelset.train$IIPI),   
## include.mean = FALSE, trace = FALSE)   
##   
## Mean and Variance Equation:  
## data ~ arma(1, 1) + garch(1, 1)  
## <environment: 0x0000000010618060>  
## [data = diff(df.ts.modelset.train$IIPI)]  
##   
## Conditional Distribution:  
## norm   
##   
## Coefficient(s):  
## ar1 ma1 omega alpha1 beta1   
## 0.57153 -0.35250 0.18164 0.19030 0.80761   
##   
## Std. Errors:  
## based on Hessian   
##   
## Error Analysis:  
## Estimate Std. Error t value Pr(>|t|)   
## ar1 0.57153 0.11745 4.866 1.14e-06 \*\*\*  
## ma1 -0.35250 0.13286 -2.653 0.00797 \*\*   
## omega 0.18164 0.07955 2.283 0.02241 \*   
## alpha1 0.19030 0.04057 4.690 2.73e-06 \*\*\*  
## beta1 0.80761 0.03364 24.007 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Log Likelihood:  
## -1009.899 normalized: -2.393125   
##   
## Description:  
## Sat Mar 05 15:47:25 2022 by user: nptho

The insignificant mean is removed from the model and all other model parameters are significant.

### White noise confirmation



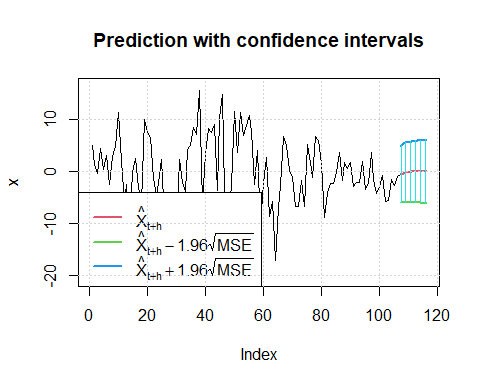
##   
## Box-Pierce test  
##   
## data: arima111.garch11@residuals/arima111.garch11@sigma.t  
## X-squared = 28.11, df = 20, p-value = 0.1068



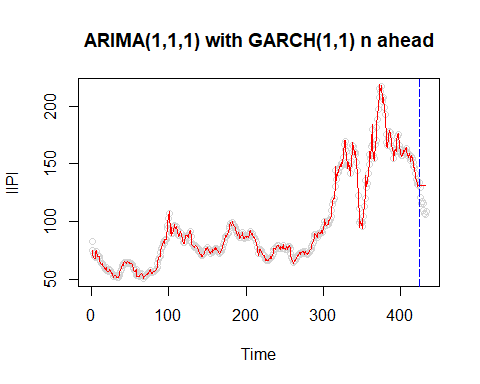
##   
## Box-Pierce test  
##   
## data: (arima111.garch11@residuals/arima111.garch11@sigma.t)^2  
## X-squared = 20.536, df = 20, p-value = 0.4249

The residuals and squared residuals of this model are white noise!

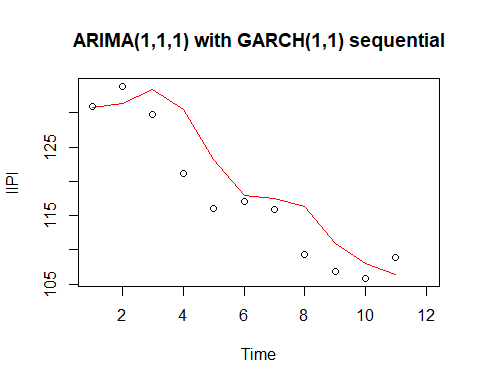
The following is the predicted conditional variance of the IIPI first-difference:



As expected the n ahead prediction value tends toward an asymptotic value:



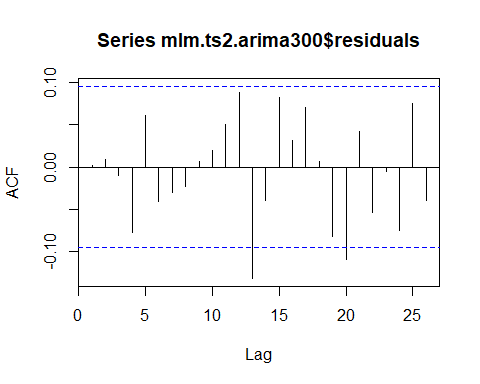
When taking the predictions 1 step ahead, the model predicts well:



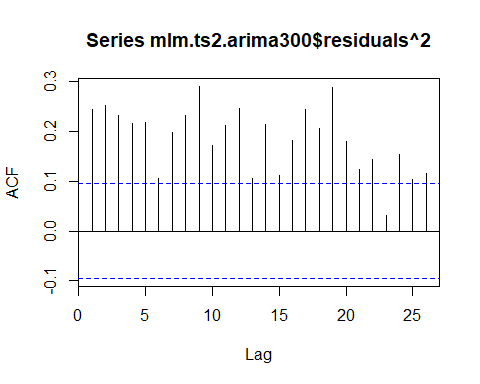
This result can be seen in the MAPE comparison summarized in the Prediction Comparison section.

### Regression with ARMIA(3,0,0) on residuals and GARCH

Residuals are first checked for GARCH model potential:



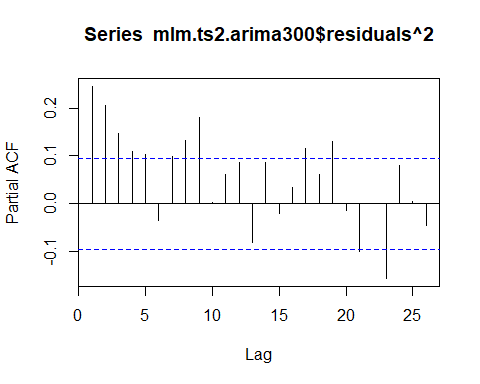
##   
## Box-Pierce test  
##   
## data: mlm.ts2.arima300$residuals  
## X-squared = 31.276, df = 20, p-value = 0.05165



##   
## Box-Pierce test  
##   
## data: mlm.ts2.arima300$residuals^2  
## X-squared = 387.34, df = 20, p-value < 2.2e-16

As can be seen, similar to the ARIMA(2,1,0) and ARIMA(1,1,1) models, the residuals of the regression with ARIMA(3,0,0) residuals are white noise, but the squared residuals are not. This is again an excellent candidate for using GARCH!

The PACF is displayed to investigate the possible GARCH order.



### Model Summary

As a model was not immediately obvious from the ACF and PACF, a appropriate GARCH model was assumed and estimated using the *rugarch* R library. NOTE: for this implementation of GARCH, *fGARCH* was not used, as it did not appear to allow for inclusion of external regression parameters:

p\_index <- list()  
q\_index <- list()  
d <- diff(df.ts.modelset.train$IIPI)   
x <- df.ts.modelset.train[2:423,] %>% select(-IIPI,-Date\_form,-GDP,-IndPro) %>% as.matrix()  
  
spec = ugarchspec(mean.model = list(armaOrder = c(3, 0), external.regressors = cbind(x)))  
fit = ugarchfit(spec = spec, data = d)

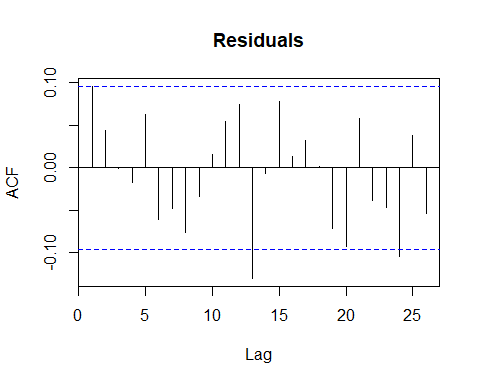
##   
## \*---------------------------------\*  
## \* GARCH Model Fit \*  
## \*---------------------------------\*  
##   
## Conditional Variance Dynamics   
## -----------------------------------  
## GARCH Model : sGARCH(1,1)  
## Mean Model : ARFIMA(3,0,0)  
## Distribution : norm   
##   
## Optimal Parameters  
## ------------------------------------  
## Estimate Std. Error t value Pr(>|t|)  
## mu 0.828213 1.044427 0.79298 0.427788  
## ar1 0.182797 0.055661 3.28411 0.001023  
## ar2 0.083830 0.055118 1.52091 0.128282  
## ar3 0.025723 0.053817 0.47797 0.632669  
## mxreg1 -0.130815 0.083281 -1.57076 0.116237  
## mxreg2 -0.041577 0.095066 -0.43735 0.661856  
## mxreg3 -0.000159 0.000117 -1.36175 0.173278  
## mxreg4 0.108487 0.100599 1.07841 0.280851  
## mxreg5 -0.438873 0.437626 -1.00285 0.315933  
## omega 0.184981 0.079906 2.31499 0.020613  
## alpha1 0.190621 0.040313 4.72852 0.000002  
## beta1 0.805855 0.034369 23.44727 0.000000  
##   
## Robust Standard Errors:  
## Estimate Std. Error t value Pr(>|t|)  
## mu 0.828213 1.248895 0.66316 0.507230  
## ar1 0.182797 0.055030 3.32174 0.000895  
## ar2 0.083830 0.060353 1.38900 0.164834  
## ar3 0.025723 0.056813 0.45277 0.650716  
## mxreg1 -0.130815 0.081678 -1.60159 0.109246  
## mxreg2 -0.041577 0.069099 -0.60171 0.547371  
## mxreg3 -0.000159 0.000136 -1.16816 0.242742  
## mxreg4 0.108487 0.112313 0.96594 0.334076  
## mxreg5 -0.438873 0.377317 -1.16314 0.244771  
## omega 0.184981 0.076488 2.41842 0.015588  
## alpha1 0.190621 0.043850 4.34712 0.000014  
## beta1 0.805855 0.030042 26.82431 0.000000  
##   
## LogLikelihood : -1007.778   
##   
## Information Criteria  
## ------------------------------------  
##   
## Akaike 4.8331  
## Bayes 4.9481  
## Shibata 4.8315  
## Hannan-Quinn 4.8785  
##   
## Weighted Ljung-Box Test on Standardized Residuals  
## ------------------------------------  
## statistic p-value  
## Lag[1] 1.565 0.210940  
## Lag[2\*(p+q)+(p+q)-1][8] 6.202 0.005627  
## Lag[4\*(p+q)+(p+q)-1][14] 9.602 0.154639  
## d.o.f=3  
## H0 : No serial correlation  
##   
## Weighted Ljung-Box Test on Standardized Squared Residuals  
## ------------------------------------  
## statistic p-value  
## Lag[1] 0.5128 0.47393  
## Lag[2\*(p+q)+(p+q)-1][5] 5.3330 0.12855  
## Lag[4\*(p+q)+(p+q)-1][9] 10.0431 0.04912  
## d.o.f=2  
##   
## Weighted ARCH LM Tests  
## ------------------------------------  
## Statistic Shape Scale P-Value  
## ARCH Lag[3] 0.133 0.500 2.000 0.715381  
## ARCH Lag[5] 9.460 1.440 1.667 0.008899  
## ARCH Lag[7] 12.041 2.315 1.543 0.006066  
##   
## Nyblom stability test  
## ------------------------------------  
## Joint Statistic: 1.8095  
## Individual Statistics:   
## mu 0.11850  
## ar1 0.22650  
## ar2 0.20195  
## ar3 0.05323  
## mxreg1 0.05848  
## mxreg2 0.07864  
## mxreg3 0.18389  
## mxreg4 0.10377  
## mxreg5 0.08529  
## omega 0.05126  
## alpha1 0.10927  
## beta1 0.13580  
##   
## Asymptotic Critical Values (10% 5% 1%)  
## Joint Statistic: 2.69 2.96 3.51  
## Individual Statistic: 0.35 0.47 0.75  
##   
## Sign Bias Test  
## ------------------------------------  
## t-value prob sig  
## Sign Bias 0.09473 0.9246   
## Negative Sign Bias 0.82757 0.4084   
## Positive Sign Bias 0.17236 0.8632   
## Joint Effect 1.18575 0.7564   
##   
##   
## Adjusted Pearson Goodness-of-Fit Test:  
## ------------------------------------  
## group statistic p-value(g-1)  
## 1 20 22.64 0.2534  
## 2 30 28.57 0.4877  
## 3 40 33.92 0.7003  
## 4 50 41.74 0.7594  
##   
##   
## Elapsed time : 0.163841

To summarize the output above:

mxreg1 is through mxreg5 correspond to TBillClose, InfRate, M2, UnempRate, Recession

It appears that with inclusion of the GARCH on the residuals and are no longer significant, and all regression coefficients are now insignificant. All GARCH coefficients are significant.

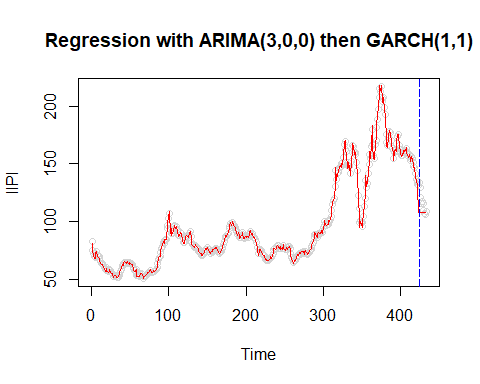
### White noise confirmation



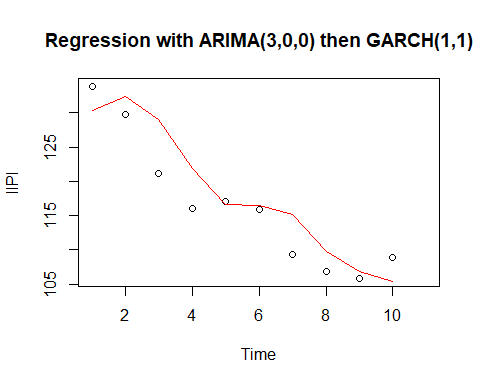
##   
## Box-Pierce test  
##   
## data: fit@fit$residuals  
## X-squared = 31.46, df = 20, p-value = 0.0494

Interestingly, residuals of this model would not be considered white noise.

As expected the n ahead prediction value tends toward an asymptotic value:



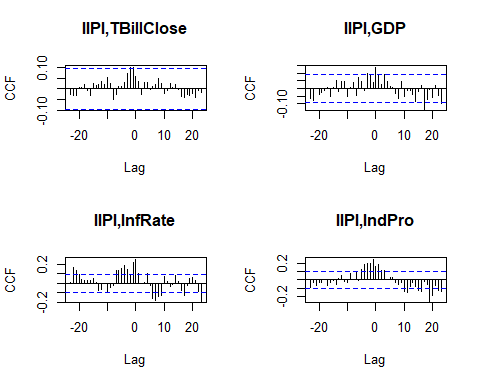
When taking the predictions 1 step ahead, the model performs much better, though not as well as the ARIMA models above:

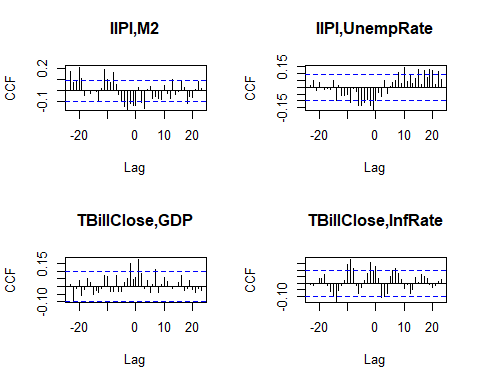


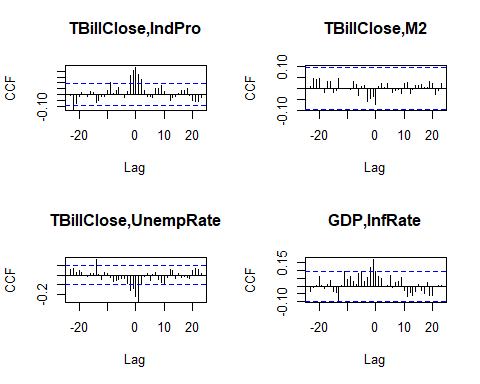
## Vector Autoregressive Moving Average

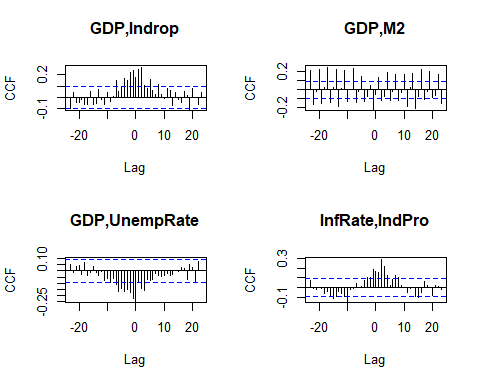
VARMA is used to estimate systems of time series data, allowing for concurent parameter estimation.

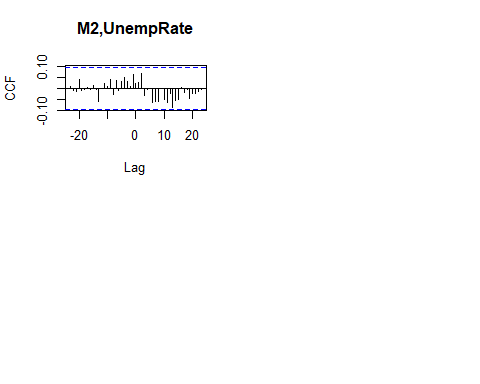
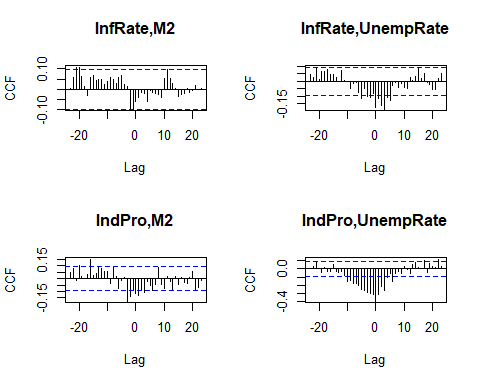
To perform VARMA, first, stationarity is induced in all variables by taking the first difference. Recession is removed from the dataset as it is an indicator variable and is therefore not applicable in VARMA modelling. All pair-wise cross-correlation functions (CCFs) are checked for bivariate white noise. If one variable is found to be bivariate white noise with all others, it would be removed from the subsequent VARMA model:











M2 would be a candidate for removal from the VARMA except that it is not bivariate white noise with GDP. All others are clearly not bivariate white noise.

In consideration of computation time only VAR(1) model is estimated, with a matrix of 49 estimates:

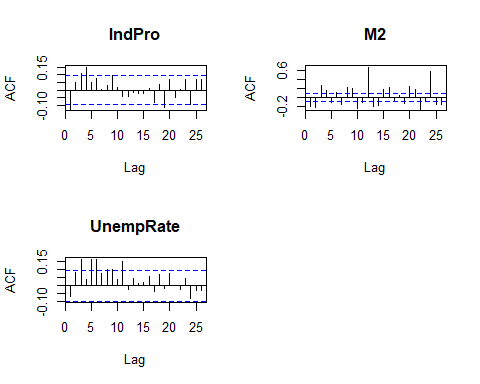
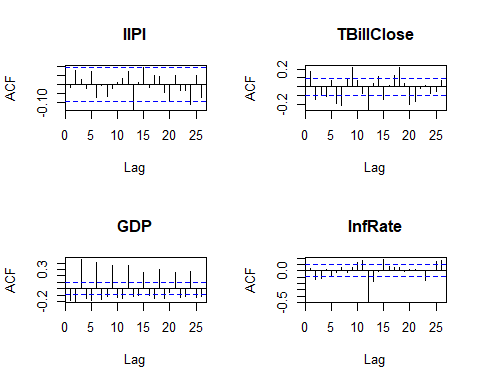
|  | IIPI | TBillClose | GDP | InfRate | IndPro | M2 | UnempRate |
| --- | --- | --- | --- | --- | --- | --- | --- |
| IIPI | 0.3005948 | 0.0062764 | 0.6331118 | 0.0097601 | 0.0216818 | -0.4756376 | -0.0035395 |
| TBillClose | 0.4149962 | 0.2898252 | 2.8811548 | 0.0365280 | 0.1319462 | -2.3916959 | -0.0287407 |
| GDP | 0.0008294 | 0.0006881 | -0.1430356 | 0.0005846 | 0.0016747 | -0.0187129 | -0.0003959 |
| InfRate | 0.1565902 | -0.0341202 | 2.8613937 | 0.3723366 | 0.0760777 | -8.6053639 | -0.0048826 |
| IndPro | 1.0920535 | -0.0029580 | 38.4951005 | 0.0275926 | 0.1319939 | 6.1424325 | -0.0769526 |
| M2 | 0.0068815 | -0.0002027 | 0.4821545 | -0.0004496 | 0.0001455 | 0.4263939 | 0.0002248 |
| UnempRate | 1.2035537 | -0.4594747 | 5.9520571 | -0.0053139 | -0.5185707 | 2.2371101 | 0.0096518 |

An interesting note: Industrial Production from the previous month has significant affect on the subsequent month’s GDP!

However, only those with a ratio of estimate to standard error > ~1.96 are significant:

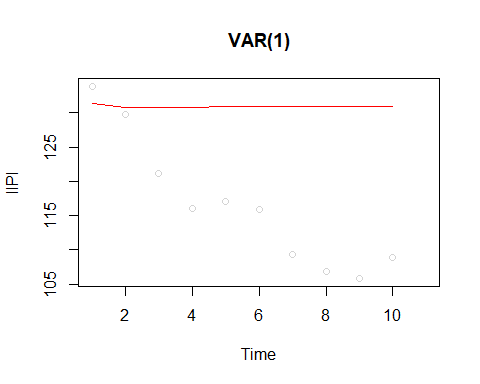
|  | IIPI | TBillClose | GDP | InfRate | IndPro | M2 | UnempRate |
| --- | --- | --- | --- | --- | --- | --- | --- |
| IIPI | 6.2954218 | 1.1255351 | 0.8625932 | 2.0722417 | 3.4912181 | 0.9092890 | 1.6826953 |
| TBillClose | 1.0736669 | 6.4293550 | 0.4851040 | 0.9591342 | 2.6274657 | 0.5655601 | 1.6898429 |
| GDP | 0.2743562 | 1.9496803 | 3.0789594 | 1.9606739 | 4.2602952 | 0.5652120 | 2.9733342 |
| InfRate | 0.3373553 | 0.6294744 | 0.4010733 | 8.1333096 | 1.2603479 | 1.6925581 | 0.2387558 |
| IndPro | 2.9307513 | 0.0678201 | 6.7208944 | 0.7505189 | 2.7234662 | 1.5046711 | 4.6869664 |
| M2 | 1.6798273 | 0.4238165 | 7.6571374 | 1.1126582 | 0.2731064 | 9.5013832 | 1.2456107 |
| UnempRate | 1.0457814 | 3.4167056 | 0.3364158 | 0.0466863 | 3.4643665 | 0.1774486 | 0.1902881 |

The vectors of residuals are not white noise:



### Model Fit

IIPI exclusively, the model fit is:



Sequential 1 step ahead is not included for comparison, as this requires re-estimating the model ten times in a loop, in consideration of computation time. However, it is expected that it would perform better than the n ahead (), as seen for the other ARIMA models. VARMA would be no different in this regard.

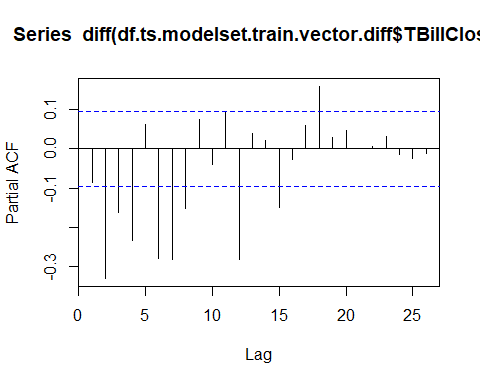
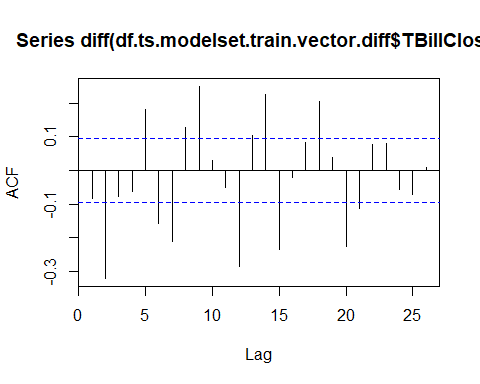
# Prediction Comparison

The various models outlined above are compared using mean absolute percent error (MAPE). The 1 step sequential predictions out perform all other methods.

| Model | MAPE |
| --- | --- |
| ARIMA(0,1,5) seq. 1 ahead | 0.0279081 |
| ARIMA(1,1,1) seq. 1 ahead | 0.0279240 |
| ARIMA(2,1,0) seq. 1 ahead | 0.0283082 |
| Regression, residual ARIMA(3,0,0), seq. 1 ahead | 0.0292654 |
| Regression, residual ARIMA(3,0,0), GARCH(1,1), seq. 1 ahead | 0.0293076 |
| Deterministic Model, 15 Harmonics, residual ARIMA, seq. 1 ahead | 0.0303350 |
| AIMAA(2,1,0) with GARCH(1,1) seq. 1 ahead | 0.0349845 |
| AIMAA(1,1,1) with GARCH(1,1) seq. 1 ahead | 0.0350633 |
| Regression, residual ARIMA(3,0,0), GARCH(1,1), n ahead | 0.0727769 |
| Deterministic Model, 65 Harmonics | 0.0787285 |
| Deterministic Model, 15 Harmonics | 0.1030518 |
| Deterministic Model, 15 Harmonics, residual ARIMA, n ahead | 0.1155852 |
| Deterministic Model, 100 Harmonics | 0.1161898 |
| Regression, residual ARIMA(3,0,0), n ahead | 0.1163019 |
| ARIMA(0,1,5) n ahead | 0.1224968 |
| ARIMA(1,1,1) n ahead | 0.1248374 |
| ARIMA(2,1,0) n ahead | 0.1269807 |
| VARMA prediction of IIPI | 0.1342517 |
| ARIMA(2,1,0) with GARCH(1,1), n ahead | 0.1347798 |
| ARIMA(1,1,1) with GARCH(1,1), n ahead | 0.1350159 |
| Deterministic Model, 85 Harmonics | 0.1993137 |
| Deterministic Model, Polynomial k=14 | 0.2574530 |
| Deterministic Model, Monthly Indicator with Trend | 0.3107488 |
| Regression Model, variable reduced | 0.5718380 |

# Transfer Function

In order to perform Transfer Function modelling, pre-whitening must be performed on all the stationary independent variables. This is accomplished using various ARIMA models on each independent variable. For each independent variable first differenced ACF and PACF are displayed, then used for assessment of potential ARIMA model, then white noise is verified using Box-Pierce:



##   
## Call:  
arima(x = df.ts.modelset.train.vector.diff$TBillClose, order = c(20, 0, 0))

Coefficients:

ar1 ar2 ar3 ar4 ar5 ar6 ar7

0.4604 -0.1726 0.1581 -0.2073 0.1360 -0.2151 0.0457

s.e. 0.0483 0.0535 0.0553 0.0553 0.0592 0.0583 0.0586

ar8 ar9 ar10 ar11 ar12 ar13 ar14

0.0023 0.2323 -0.0656 0.0863 -0.3250 0.1850 0.0959

s.e. 0.0591 0.0590 0.0601 0.0594 0.0581 0.0602 0.0619

ar15 ar16 ar17 ar18 ar19 ar20

-0.2202 0.1829 0.0284 0.1182 -0.1472 -0.2261

s.e. 0.0653 0.0685 0.0702 0.0707 0.0711 0.0619

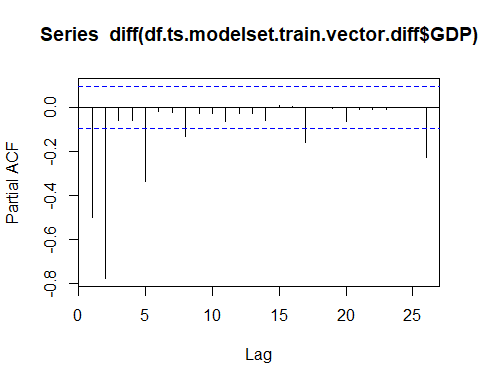
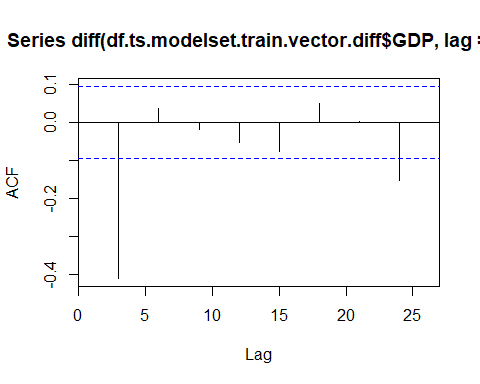
intercept

-0.0286

s.e. 0.0195

sigma^2 estimated as 0.1121: log likelihood = -143.51, aic = 331.02 ##   
## Box-Pierce test  
##   
## data: arimaTBillClose$residuals  
## X-squared = 12.156, df = 20, p-value = 0.9106

An ARIMA(20,0,0) is used to pre-whiten the first-differenced TBillClose, and the resulting residuals can be seen to be white noise based on the Box-Pierce test.



##   
## Call:  
arima(x = df.ts.modelset.train.vector.diff$GDP, order = c(5, 0, 0))

Coefficients:

ar1 ar2 ar3 ar4 ar5 intercept

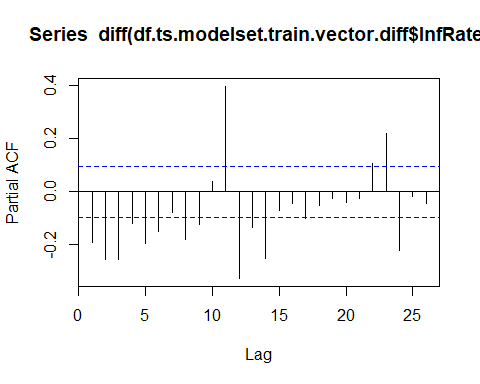
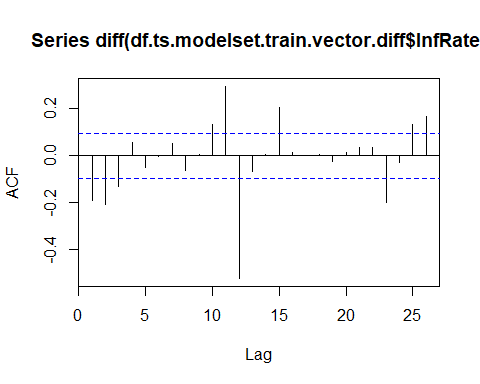
-0.0609 -0.0648 0.5648 -0.0673 -0.0648 24.6860

s.e. 0.0486 0.0491 0.0407 0.0492 0.0492 3.1892

sigma^2 estimated as 2072: log likelihood = -2210.75, aic = 4435.5

##   
## Box-Pierce test  
##   
## data: arimaGDP$residuals  
## X-squared = 30.674, df = 20, p-value = 0.05964

An ARIMA(5,0,0) is used to pre-whiten the first-differenced TBillClose, and the resulting residuals can be seen to be white noise based on the Box-Pierce test.



##   
## Call:  
arima(x = df.ts.modelset.train.vector.diff$InfRate, order = c(2, 0, 2), seasonal = list(order = c(1,

0, 1), period = 12))

Coefficients:

ar1 ar2 ma1 ma2 sar1 sma1

1.2253 -0.2341 -0.6651 -0.2327 -0.1296 -0.8561

s.e. 0.1084 0.1060 0.1057 0.0879 0.0557 0.0337

intercept

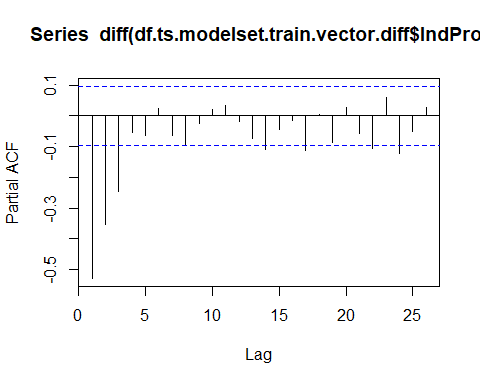
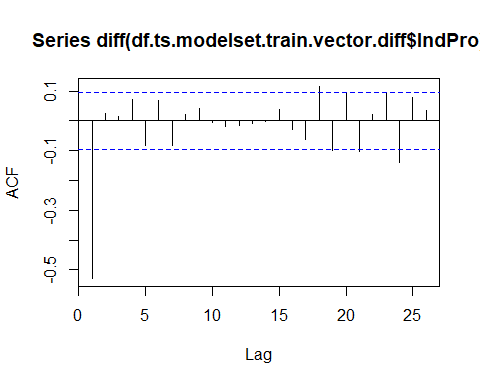
-0.0295

s.e. 0.0218

sigma^2 estimated as 0.06965: log likelihood = -45.45, aic = 106.9

##   
## Box-Pierce test  
##   
## data: arimaInfRate$residuals  
## X-squared = 26.537, df = 20, p-value = 0.1488

A SARIMA(2,0,2)(1,0,1) is used to pre-whiten the first-differenced InfRate, and the resulting residuals can be seen to be white noise based on the Box-Pierce test.



##   
## Call:  
arima(x = df.ts.modelset.train.vector.diff$IndPro, order = c(2, 0, 2))

Coefficients:

ar1 ar2 ma1 ma2 intercept

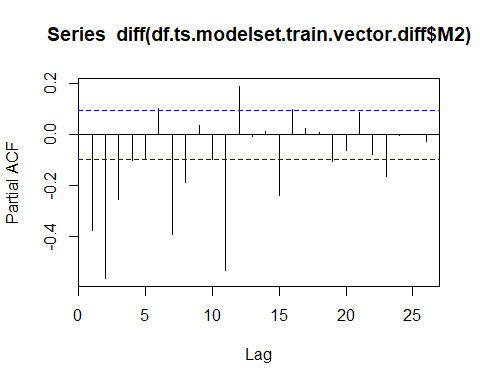
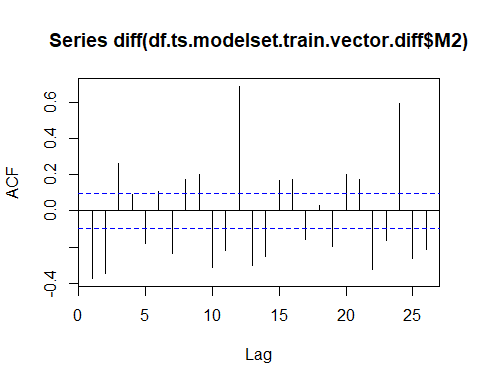
1.5389 -0.6222 -1.4154 0.6264 0.1020

s.e. 0.3547 0.3147 0.3247 0.2008 0.0582

sigma^2 estimated as 0.2239: log likelihood = -283.27, aic = 578.54

##   
## Box-Pierce test  
##   
## data: arimaIndPro$residuals  
## X-squared = 17.1, df = 20, p-value = 0.6464

An ARIMA(2,0,2) is used to pre-whiten the first-differenced IndPro, and the resulting residuals can be seen to be white noise based on the Box-Pierce test.



##   
## Call:  
arima(x = df.ts.modelset.train.vector.diff$M2, order = c(16, 0, 0), seasonal = list(order = c(1,

0, 0), period = 12))

Coefficients:

ar1 ar2 ar3 ar4 ar5 ar6

0.3010 -0.0271 0.2573 -0.1214 -0.0237 0.0045

s.e. 0.0473 0.0494 0.0495 0.0486 0.0335 0.0329

ar7 ar8 ar9 ar10 ar11 ar12 ar13

-0.0907 0.086 0.0416 0.0016 -0.0147 0.7227 -0.3110

s.e. 0.0335 0.034 0.0338 0.0337 0.0339 0.0382 0.0489

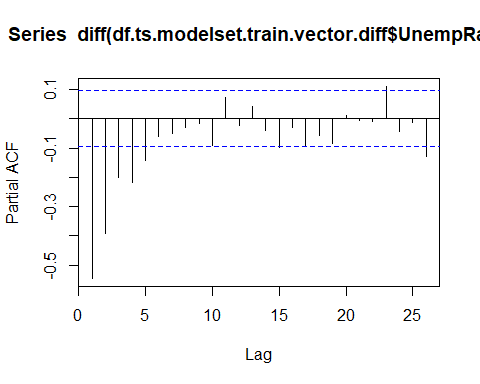
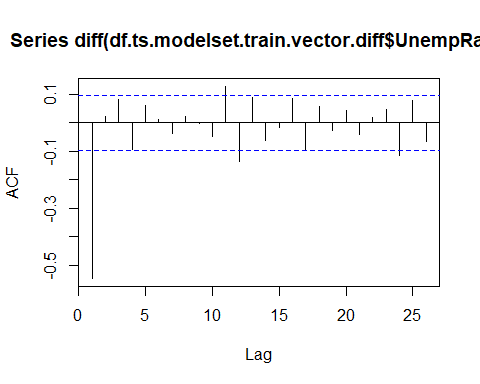
ar14 ar15 ar16 sar1 intercept

0.0177 -0.1420 0.2340 -0.2994 27.9815

s.e. 0.0502 0.0506 0.0484 0.0550 11.4516

sigma^2 estimated as 569.5: log likelihood = -1942.35, aic = 3922.7 ##   
## Box-Pierce test  
##   
## data: arimaM2$residuals  
## X-squared = 24.741, df = 20, p-value = 0.2115

A SARIMA(16,0,0)(1,0,0) is used to pre-whiten the first-differenced M2, and the resulting residuals can be seen to be white noise based on the Box-Pierce test.

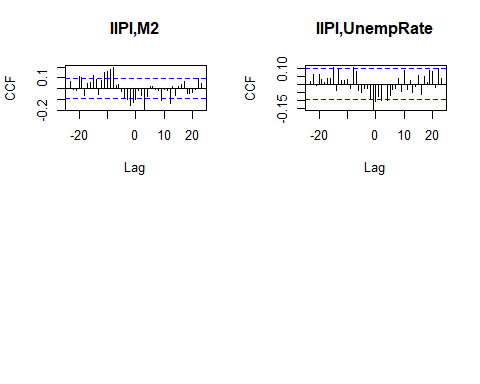
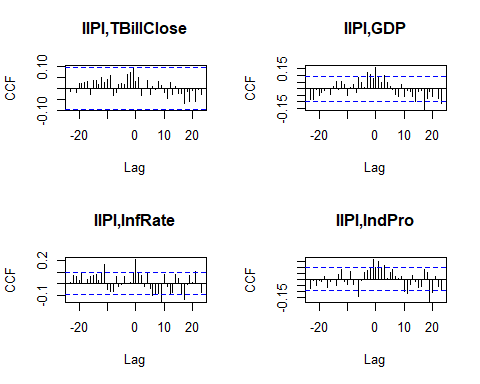


##   
## Call:  
## arima(x = df.ts.modelset.train.vector.diff$UnempRate, order = c(2, 1, 2))  
##   
## Coefficients:  
## ar1 ar2 ma1 ma2  
## -1.1607 -0.1907 0.2350 -0.7594  
## s.e. 0.0594 0.0595 0.0425 0.0426  
##   
## sigma^2 estimated as 0.02607: log likelihood = 169.17, aic = -330.34

##   
## Box-Pierce test  
##   
## data: arimaUnempRate$residuals  
## X-squared = 21.574, df = 20, p-value = 0.3641

An ARIMA(2,0,2) is used to pre-whiten the first-differenced UnempRate, and the resulting residuals can be seen to be white noise based on the Box-Pierce test.

Next the CCFs are displayed to determine , , values:



## Transfer Function on IndPro

Taking IndPro as the independent variable, and , , and .

X\_b1 <- df.ts.modelset.train.vector.prewhite$IndPro[1:422]- mean(df.ts.modelset.train.vector.prewhite$IndPro)

Y\_b1 <- df.ts.modelset.train.vector.prewhite$IIPI[2:423] - mean(df.ts.modelset.train.vector.prewhite$IIPI)

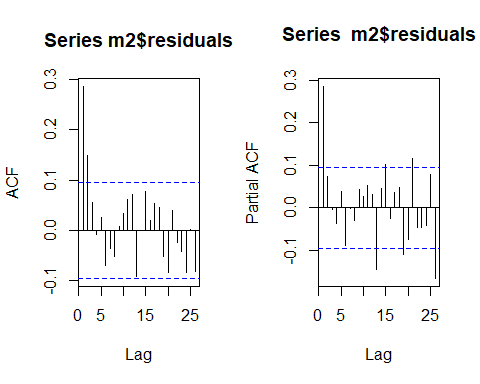
The transfer function is estimated, forcing the residual correction to zero first to investigate:

m2<-arimax(Y\_b1, order=c(1,0,0), fixed=c(0,NA,NA), xtransf=data.frame(X\_b1), transfer=list(c(1,0)), include.mean = FALSE)

##   
##   
## Call:  
## arimax(x = Y\_b1, order = c(1, 0, 0), include.mean = FALSE, fixed = c(0, NA,   
## NA), xtransf = data.frame(X\_b1), transfer = list(c(1, 0)))  
##   
## Coefficients:  
## ar1 X\_b1-AR1 X\_b1-MA0  
## 0 0.6863 1.3059  
## s.e. 0 0.0863 0.3232  
##   
## sigma^2 estimated as 14.74: log likelihood = -1163.79, aic = 2331.58

##   
## Box-Pierce test  
##   
## data: m2$residuals  
## X-squared = 65.656, df = 20, p-value = 9.172e-07

The model is then assessed for ARIMA residual correction. As the residuals of the model are not white noise and seen in the above Box-Pierce and ACF.



This is corrected using :

##   
Call:  
 arimax(x = Y\_b2, order = c(1, 0, 1), include.mean = FALSE, xtransf = data.frame(X\_b2),

transfer = list(c(1, 0)))

Coefficients:

ar1 ma1 X\_b2-AR1 X\_b2-MA0

0.5557 -0.2786 0.7135 0.5039

s.e. 0.1134 0.1278 0.1827 0.3581

sigma^2 estimated as 13.64: log likelihood = -1144.82, aic = 2297.64

##   
## Box-Pierce test  
##   
## data: m2\_arima11\_resid$residuals  
## X-squared = 26.28, df = 20, p-value = 0.1568

This has made insignificnat. The residuals are, however are white noise.

The transfer function model of IIPI using IndPro lagged by two months as predictor with ARIMA(1,0,1) correction on the residuals results in white noise. The model estimated is:

# Conclusion

Time series analysis is an exercise in extracting information from current and previous values of variables of interest and other predictors to allow explanation of the current or future values of a random variable. ARIMA takes into account previous values of the dependent variable. Transfer functions take into account previous values of independent variables. Regression models take into account the current values of independent variables.

It is my hope that this project outlines the implementation of these methods and provides clear overview of the methods and thought processes undertaken to enact each model.

The following Models resulted in white noise:

* ARIMA(2,1,0)
* ARIMA(1,1,1)
* Regression with ARMIA(3,0,0)
* ARIMA(2,1,0) with GARCH(1,1)
* ARIMA(1,1,1) with GARCH(1,1)
* TF b=1, r=1, s=0 with ARIMA(1,0,1)

Key project takeaways:

* The choice of independent variables were unfortunately lacking in predictive value.
* Harmonic models can be made to overfit in-sample, but as soon as prediction is involved, the bias-variance trade off has to be accounted for.
* Inducing white noise using ARIMA could typically be accomplished using arbitrarily many p or q values but at the cost of more parameters to estimate, and compromise of model interpretability and parsimony.
* N step ahead predictions are very poor typically beyond 1 step.
* Sequential prediction (1 step ahead) is much better.
* A transfer function with s and r equal to zero is simply a regression model. It was noted that inflation rate could have been used in transfer function modelling but the and parameters would have been zero, resulting in a trivial regression model.