W271 Live Session 9: ARIMA and SARIMA

Devesh Tiwari

7/11/2017

Main topics covered in Week 9

- Mixed Autoregressive Moving Average (ARMA) Models
  - Mathematical formulation and derivation of key properties
  - Comparing ARMA models and AR models using simulated series
  - Comparing ARMA models and AR models using an example
- An introduction to non-stationary time series model
- Random walk and integrated processes
- Autoregressive Integrated Moving Average (ARIMA) Models
  - Review the steps to build ARIMA time series model
  - Simulation
  - Modeling with simulated data using the Box-Jenkins approach
  - Estimation, model diagnostics, model identification, model selection, assumption testing, and statistical inference / forecasting, backtesting
- Seasonal ARIMA (SARIMA) Models
  - Mathematical formulation
  - An empirical example
- Putting everything together: ARIMA modeling

Readings

CM2009: Paul S.P. Cowpertwait and Andrew V. Metcalfe. *Introductory Time Series with R*. Springer. 2009.

- Ch. 4.3 – 4.7, 6, 7.1 – 7.3

SS2016: Robert H. Shumway and David S. Stoffer. *Time Series Analysis and Applications.* EZ Edition with R Examples:

- 3.7 – 3.10, review 3.1 – 3.6

HA: Rob J Hyndman and George Athanasopoulos. Forecasting: Principles and Practice:

- Ch. 8.5 – 8.9

Agenda for this week's live session:

1. Questions from last week
2. Recap
3. Breakout Session 1: ARIMA nomenclature
4. Breakout Session 2: SARIMA EDA

5. SARIMA modeling (in-class discussion / take-home)

Recap and overview

1. Last week, we were introduced to autoregressive (AR), moving average (MA), and autoregressive moving average (ARMA) models. These models are only appriorate for time-series that are weakly stationary (stationary in the mean and the variance).
2. We often are confronted with time-series that is not stationary in the mean and variance. Luckily for us, we can transform these series in order to make them stationary!
3. Here is an incomplete list of how/why time-series might not be stationary in the mean:
   a. Data has a trend
   b. Data contains a unit-root (next week)
   c. Data contains seasonal elements
4. We can take care of these problems either by detrending the data or by differencing the data. Once the data are transformed into a weakly stationary series, we can model the resulting series with an ARMA model. We call these models ARIMA models if the data do not exhibit any seasonality. If the data are seasonal, then these models are called SARIMA models.
5. Remember, here are the steps to building an ARIMA model!
   i. Conduct and EDA to determine if you need to transform the data in order to make it stationary.
   ii. Transform the data if needed.
   iii. Estimate several Arima(p,d,q) models. Remember, you set the value of d in the first step! So really, you are trying to find the apprioriate values of p and q.
   iv. Evaluate the residuals of models with the lowest AIC/BIC values and simpler models. Select the model where the residuals resemble white nose.
   v. If you still have some candidate models remaining, then conduct an out of sample test and select the model with the lowest forecasting error.
   vi. Answer your question / generate forecasts!

Breakout Session 1: Review Questions

1. Considert the following time-series process:

$$x_t = t + \alpha_1 x_{t-1} + \alpha_2 x_{t-2} + w_t$$

where t is time.

Hint:

$$x_{t-1} = t - 1 + \alpha_1 x_{t-2} + \alpha_2 x_{t-3} + w_{t-1}$$

and

$$\mathbf{B}^n(x_t) = x_{t-n}$$

and

$$\nabla = (1 - \mathbf{B})$$

(a) What does this equation mean, in words?

This is an AR(2) process with a linear trend.

(b) Take the first difference of this equation and then re-write it using the backshift operator What kind of model is it? What R commands would you use to model this?

We can rewrite this equation and transform it into an ARIMA(2,1,1) model. Note that this differencing procedure introduces a moving average term into the model, which has a coefficient of 1.

(c) How would you know if this model is stationary?

After fitting a model, I would find the roots of the second-order characteristic polynomial $1 - \alpha_1\mathbf{B} - \alpha_1\mathbf{B}^2$. If the absolute value of the roots is greater than 1, then the model is stationary.

2. Consider a time series represented by the following equation:

$$(1 - \alpha_1\mathbf{B} - \alpha_2\mathbf{B}^2)(1 - \mathbf{B})x_t = \omega_t(1 + \beta_1\mathbf{B})$$

(a) Describe this time series using the ARIMA(p,d,q) notation.

ARIMA(2,1,1)

(b) Can you tell if the time series that is generated by this model is stationary? If so, is it? If you cannot determine whether it is stationary, what additional information do you need?

Because the data require differencing, the raw series is not statioanry in the mean.

Seasonality and EDA For SARIMA models

Many time-series data exhibit a clear seasonal pattern. Consider data that are measured every month (such as monthly sales or even unemployment). There are 12 months in a year and it is very common to see a strong relationship for a month across several years. In other words, sales in January are often correlated with sales in the prior Januarys etc. In addition, sales in Janurary are also likely to be correlated with prior months' sales as well. We would say that such data are seasonal with a period of 12 and we can directly model the seasonal effect withiin the ARIMA framework.

You can detect seasonality during the EDA process. Sometimes, the seasonality is "obvious" by simply plotting the time-series. In addition to the plot, you can check to see if there is an especially high spike on the ACF and PACF charts at m (m = 12 if you have monthly data, m = 52 if you have weekly data), and you can use the monthplot (which you will do shortly).

Once you have identified seasonality, you need to model it. However, within the ARIMA framework, you need to make sure that your data are stationary. We know that we can differnce the time-series to make it stationary in the mean, but we can also take the seasonal difference to make it stationary in the mean with respect to a seasonal effect also. After you have made the data stationary (by setting the d and D values), examine *that* data: plot it and look at the ACF/PACF charts and make your initial guesses about p,q,P, and Q.

Breakout sesssion 2 Seasonality and EDA

Load the following three datasets and conduct an EDA on them. For each, come prepared to describe what you see in the initial plot of the data, what steps you took to make the data stationary in the mean, and what your initial ARIMA function would look like in R (Arima(x, order = c(p,d,q), seasonal = list(order = c(P,D,Q)), method = "ML")).

The "flight prices" exhibit an interesting pattern.

1. Data prior to 2006 seem more volatile than the rest, and it also seems as if they were generated from a different process. Should explore taking the log transformation or partitioning the data.
2. The raw time series exhibits a clear seasonal pattern. Strictly speaking, these data need to be differenced with a lag of 52.

3. The seasonal-differenced data is very peristent, particularly around 2010. It is possible that this needs to be differnced further
4. Differencing the data again produces a time-series that is clearly stationary in the mean, but not stationary in the variance.

Conclusions:

I will apply seasonal differencing and *not* difference further. However, I will examine how well a differenced model performs as well.

```
rm(list = ls())
library(knitr)
opts_chunk$set(tidy.opts=list(width.cutoff=60),tidy=TRUE)

library(forecast)
```

```
## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: timeDate

## This is forecast 7.3
```

```
library(astsa)
```

```
##
## Attaching package: 'astsa'

## The following object is masked from 'package:forecast':
##
##     gas
```

```
path <- "~/Documents/Projects/MIDS/"
setwd(path)
df <- read.csv("Summer 2016/materials for ISVC/Lab 3/correlate-flight_prices.csv")


# We can actually analyze the relative seach activity for many phrases!
df <- df[,1:2] #focus on the phrase flight prices
str(df)
```

```
## 'data.frame':    626 obs. of  2 variables:
##  $ Date         : Factor w/ 626 levels "1/1/06","1/1/12",..: 43 4 16 30 211 256 222 235 247 307 ...
##  $ flight.prices: num  5.811 2.901 1.575 0.047 2.8 ...
```

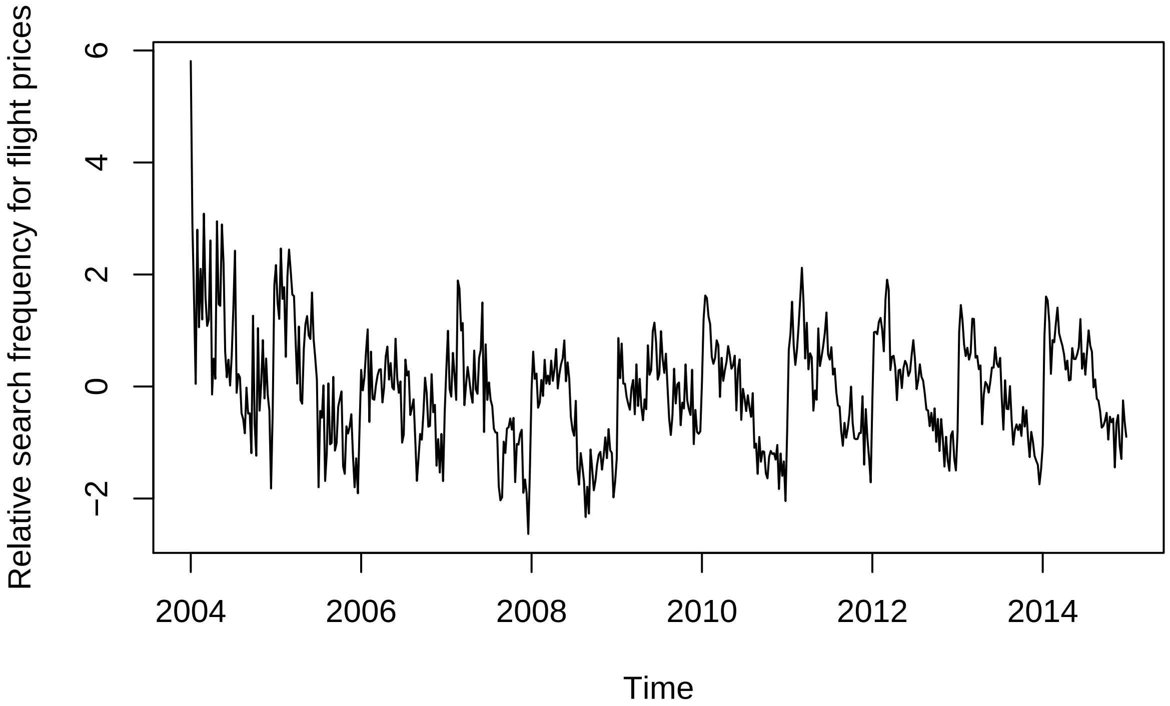
```
cbind(head(df), tail(df))
```

```
##      Date flight.prices     Date flight.prices
## 1  1/4/04         5.811 11/22/15        -1.430
## 2 1/11/04         2.901 11/29/15        -0.918
## 3 1/18/04         1.575  12/6/15        -0.952
## 4 1/25/04         0.047 12/13/15        -1.224
## 5  2/1/04         2.800 12/20/15        -0.897
## 6  2/8/04         1.058 12/27/15         0.462
```

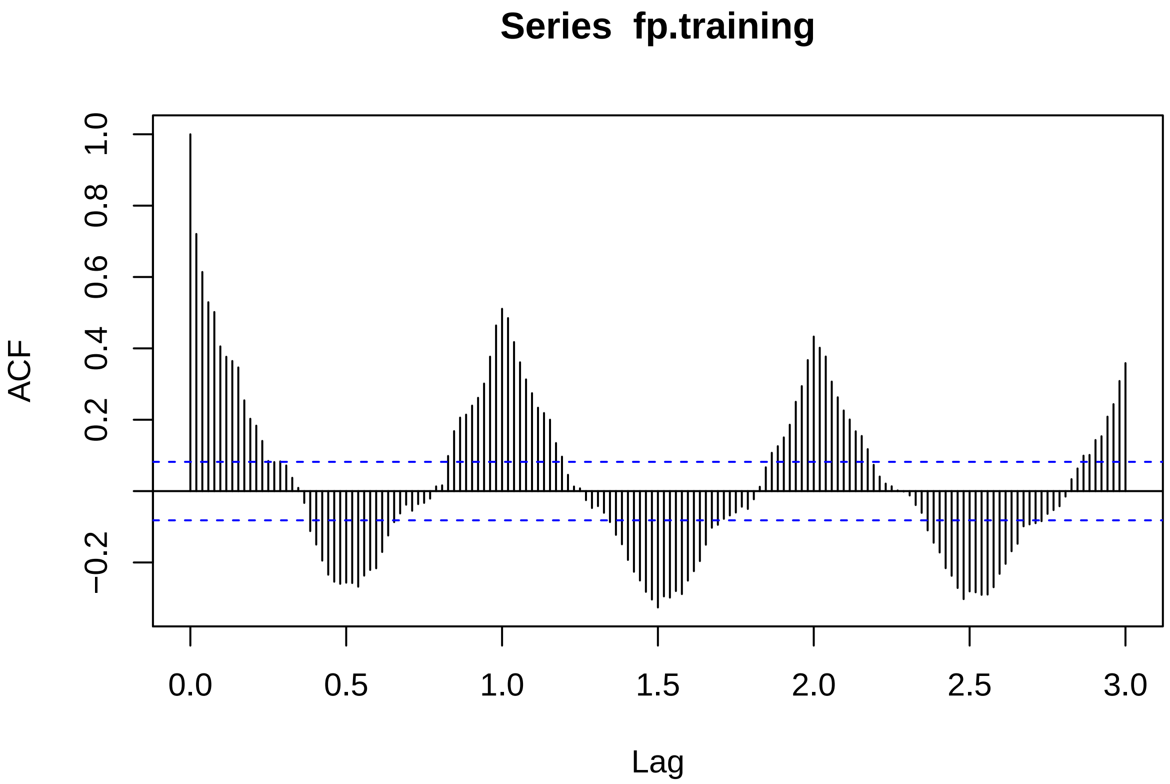
```
summary(df)
```

```
##       Date     flight.prices    
##  1/1/06 :  1   Min.   :-2.6330  
##  1/1/12 :  1   1st Qu.:-0.7480  
##  1/10/10:  1   Median : 0.0175  
##  1/11/04:  1   Mean   :-0.0123  
##  1/11/09:  1   3rd Qu.: 0.5727  
##  1/11/15:  1   Max.   : 5.8110  
##  (Other):620
```

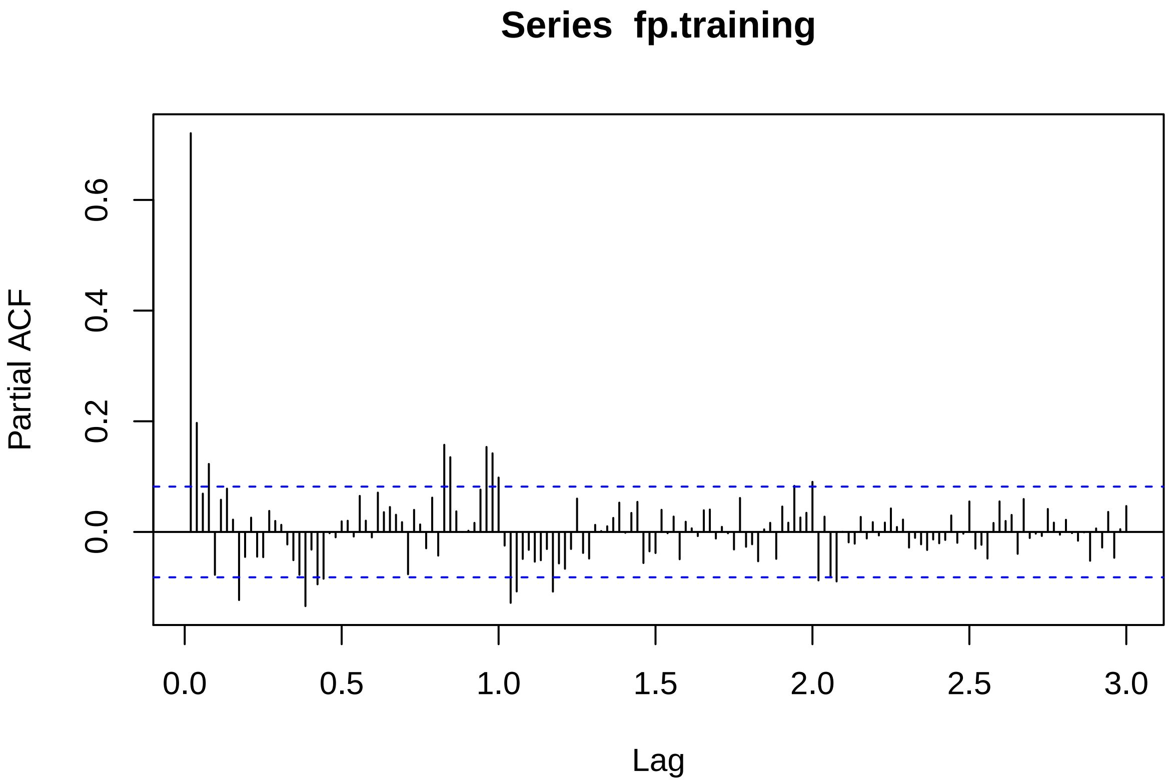
```
# Create a time-series object
fp <- ts(df$flight.prices, frequency = 52, start = c(2004,1))

fp.training <- window(fp, start = c(2004,1), end = c(2014,52))
fp.test <- window(fp, start = c(2015,1))


plot(fp.training, ylab = "Relative search frequency for flight prices")
```



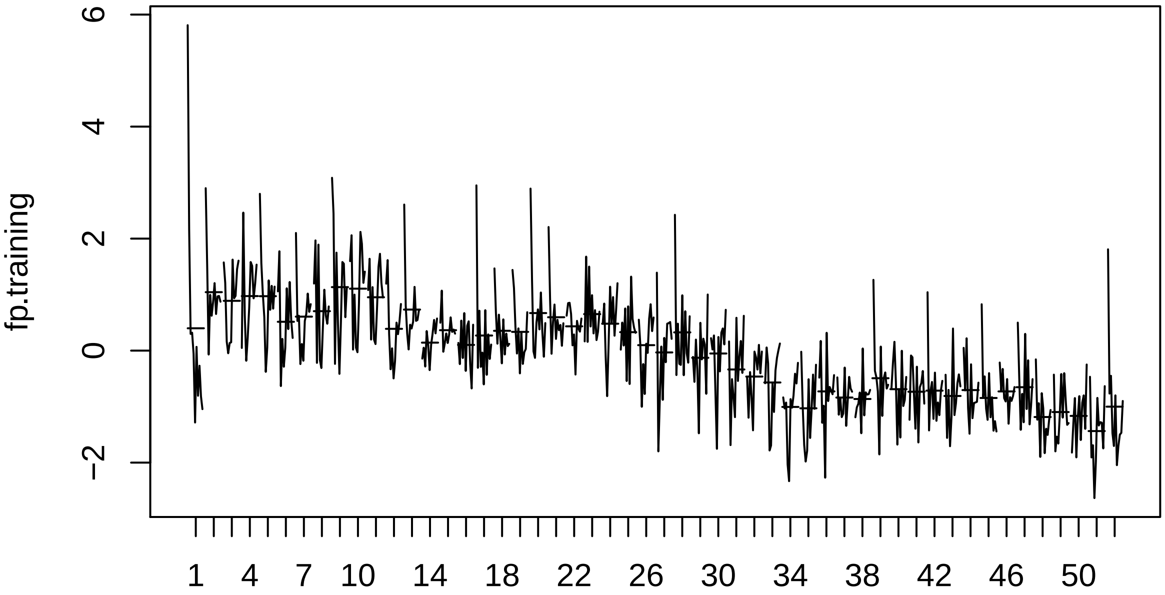
```
acf(fp.training, 156)
```



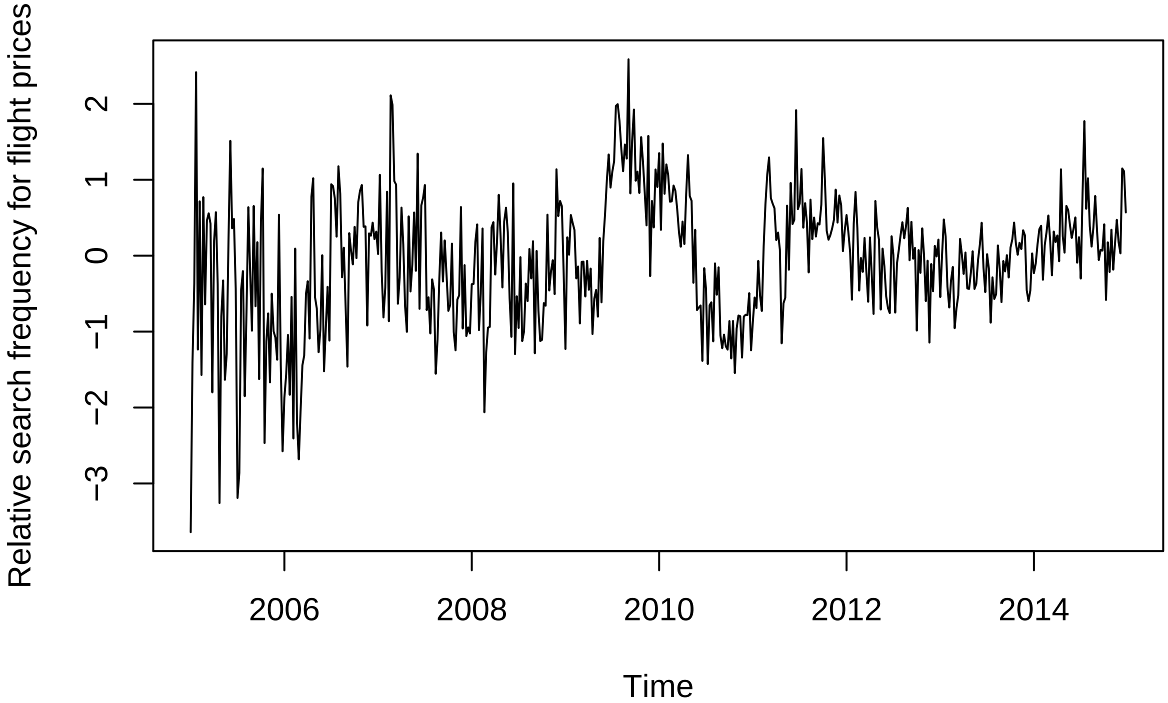
```
pacf(fp.training, 156)
```



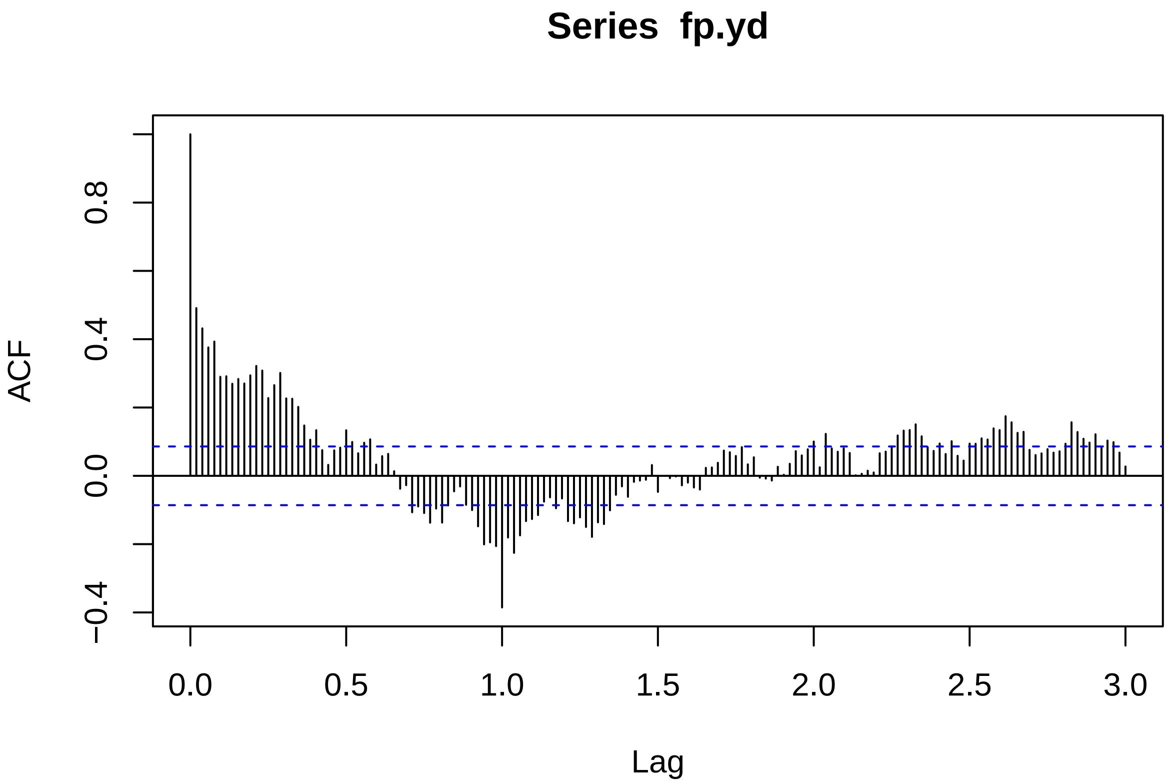
```
monthplot(fp.training)
```



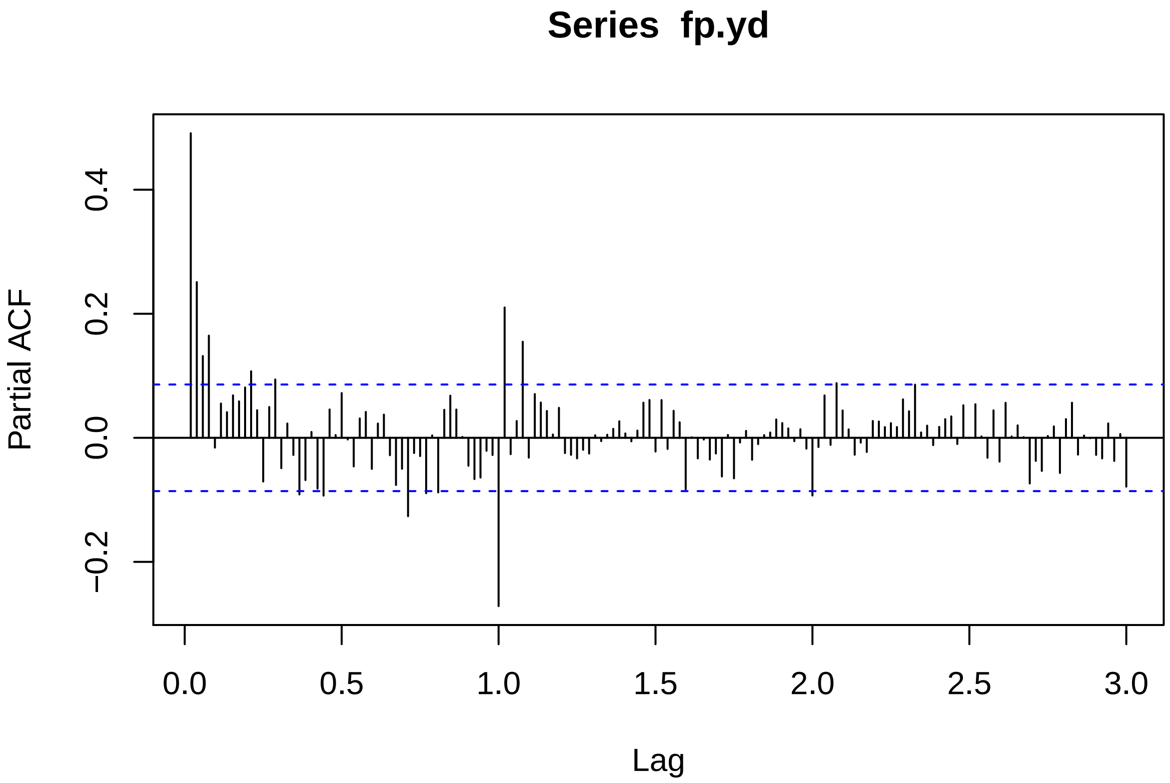
```
# Seasonal. Needs differencing

fp.yd <- diff(fp.training, lag = 52)
plot(fp.yd, ylab = "Relative search frequency for flight prices")
```



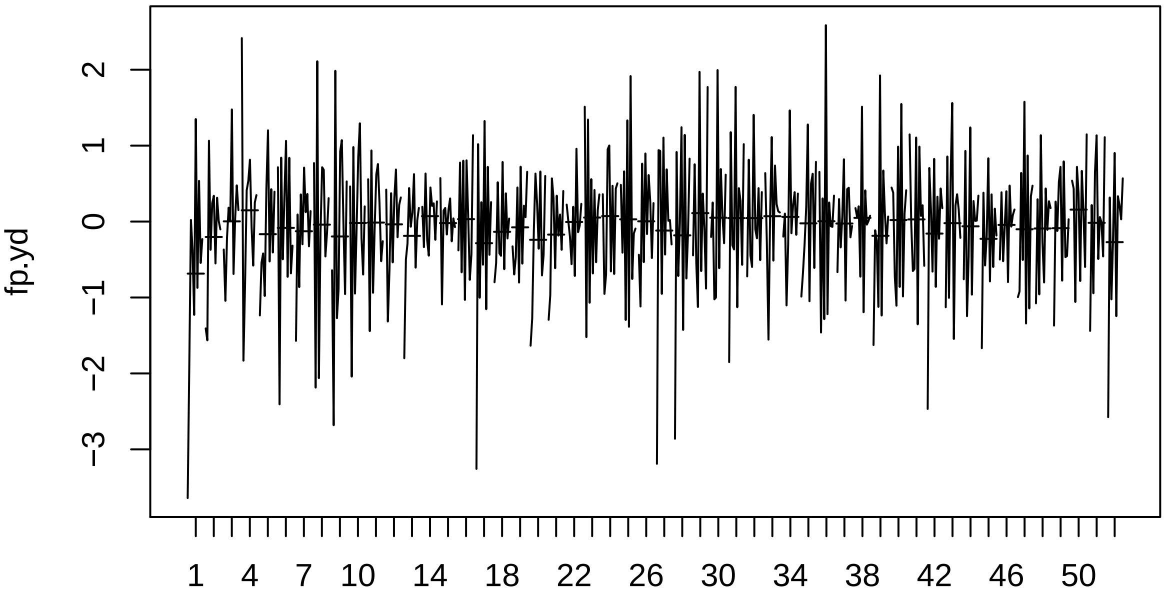
```
acf(fp.yd, 156)
```



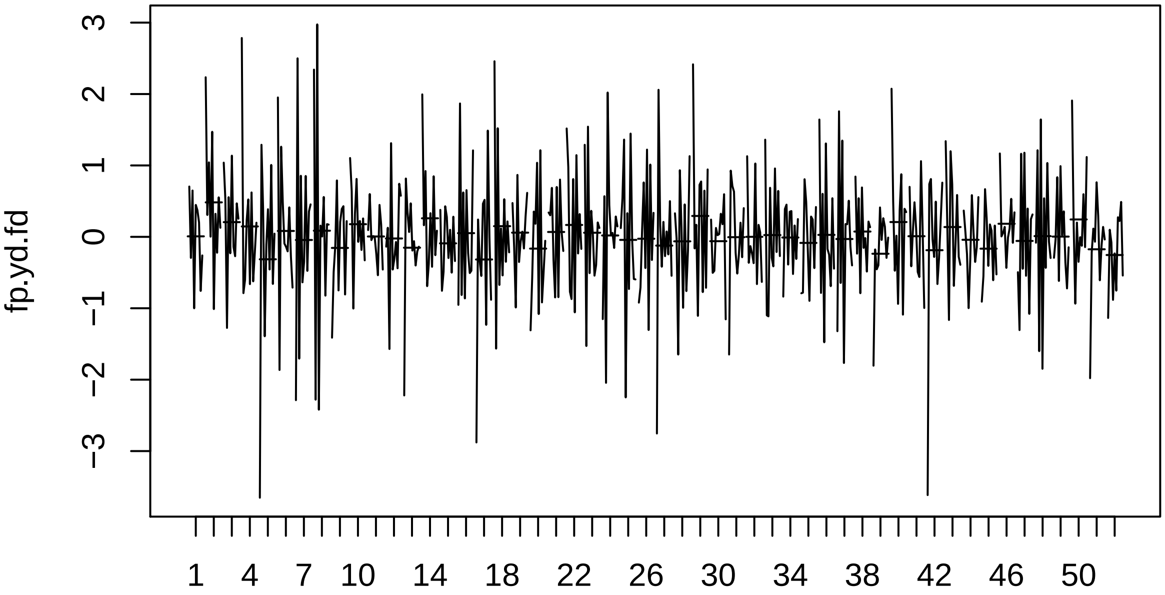
```
pacf(fp.yd, 156)
```



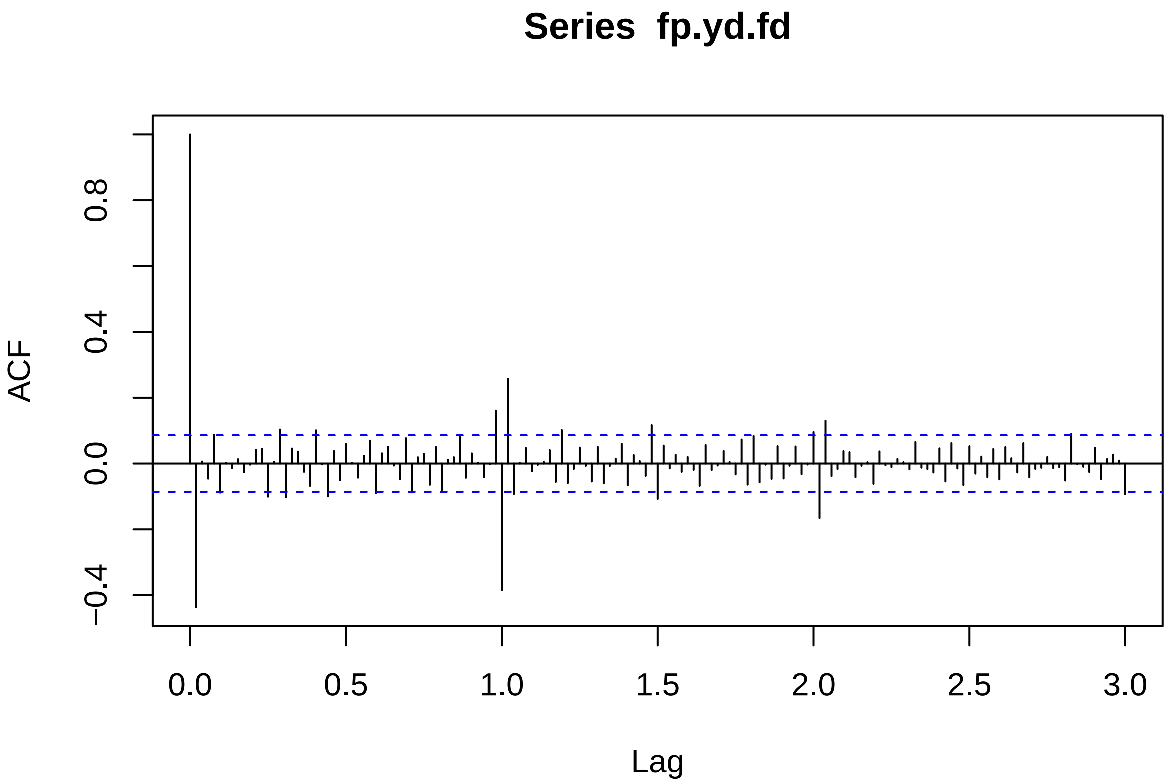
```
monthplot(fp.yd)
```



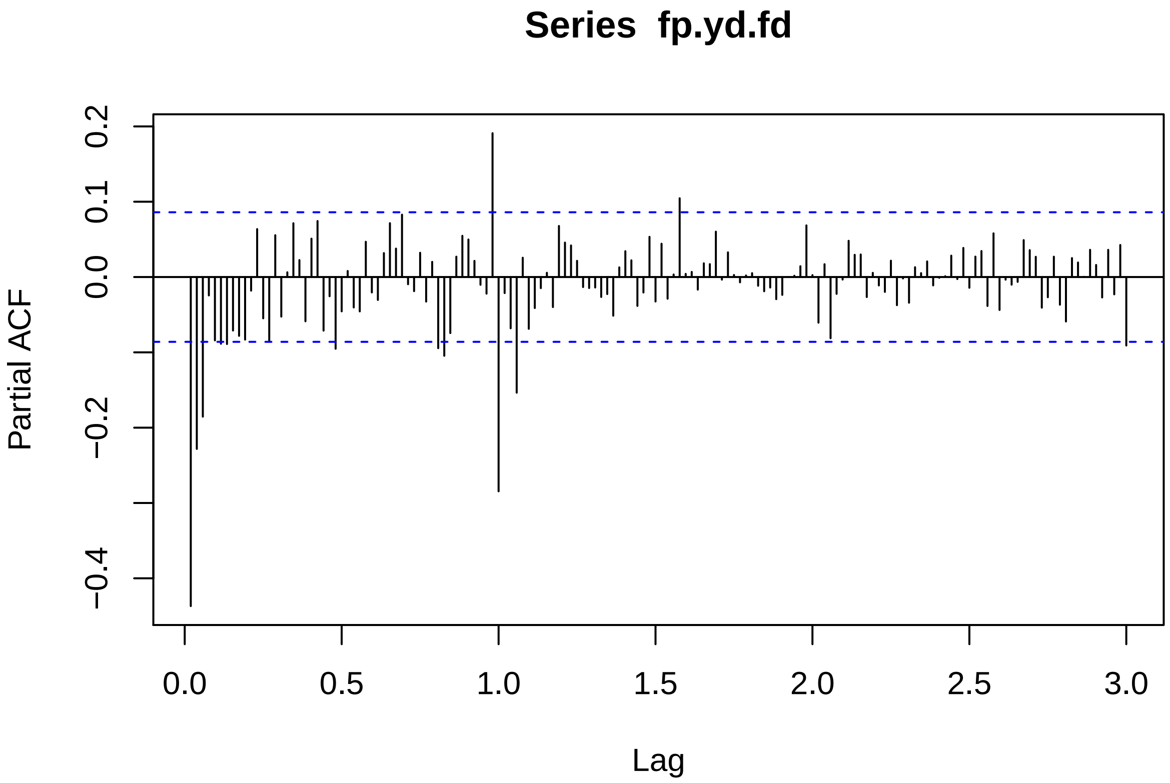
```
fp.yd.fd <- diff(fp.yd)
monthplot(fp.yd.fd)
```



```
acf(fp.yd.fd, 156)
```



```
pacf(fp.yd.fd, 156)
```



```
# conduct EDA on cmort and tempr at home
```

Putting it all together: SARIMA Flight Prices

In this live session, we are going to build a SARIMA model on the relative search activity for the phrase, “flight prices.” These data are provided by google correlate and they are weekly. For the sake of simplicity, we will focus on 2010 onward.

Remember that we can express a SARIMA model as: SARIMA(p,d,q)x(P,D,Q),m.

Step 1: Conduct EDA and determine values for d and D

Step 2: Find values for P and Q (seasonal components)

Step 3: Find values for p and q

Step 4: Fine - tuning (play around with different values of p, q, P, and Q)

Step 5: Plot fitted values against time-series

Step 6: Conduct out of sample tests

Step 7: Select model and forecast

Note: Sometimes you might not be sure about what transformations you should make. In that case, DO BOTH and compare their forecasting accuracy.

Flight price data

EDA

I am going to remove data prior to 2005. Based on the EDA, I will build models where d = 0 and D = 1. Based on the acf and pacf charts, an AR(3) may be apprioriate for the non-seasonal lags. For the seasonal component of the model, there are not any significant lags past the one year mark.

Modeling the seasonal component

It takes forever to model this data with 2 seasonal lag terms. Based on the EDA, we should not include more than 1 lag terms in the AR or MA portions of the seasonal model. Based on the AIC results, I will model the seasonal component as an ARMA(0,1), as that model’s AIC is very close to the smallest AIC.

Also, note that nothing in the ACF/PACF charts suggets that there should be more than 2 seasonal lags in the model

```
for (P in 0:2) {
    for (Q in 0:2) {
        m <- Arima(fp.training, order = c(0, 0, 0), seasonal = list(order = c(P,
            1, Q)), method = "ML")
        print(c(P, Q, m$aic))
    }
}
```

```
## [1]    0.000    0.000 1319.672
## [1]    0.000    1.000 1209.323
## [1]    0.000    2.000 1210.224
## [1]    1.000    0.000 1215.888
## [1]    1.000    1.000 1209.283
## [1]    1.000    2.000 1210.998
## [1]    2.000    0.000 1209.476
## [1]    2.000    1.000 1206.356
## [1]    2.000    2.000 1207.787
```

Non-seasonal

ARMA(2,3), ARMA(2,4), and ARMA(3,3) have AIC values around 980. More complicated models, such as the ARMA(4,4) have AIC values in the same range. The following models' residuals are worth examining:

Arima(2,0,2)x(0,1,1)

Arima(2,0,3)x(0,1,1)

Arima(2,0,4)x(0,1,1)

Arima(3,0,2)x(0,1,1)

Arima(3,0,3)x(0,1,1)

Arima(4,0,4)x(0,1,1)

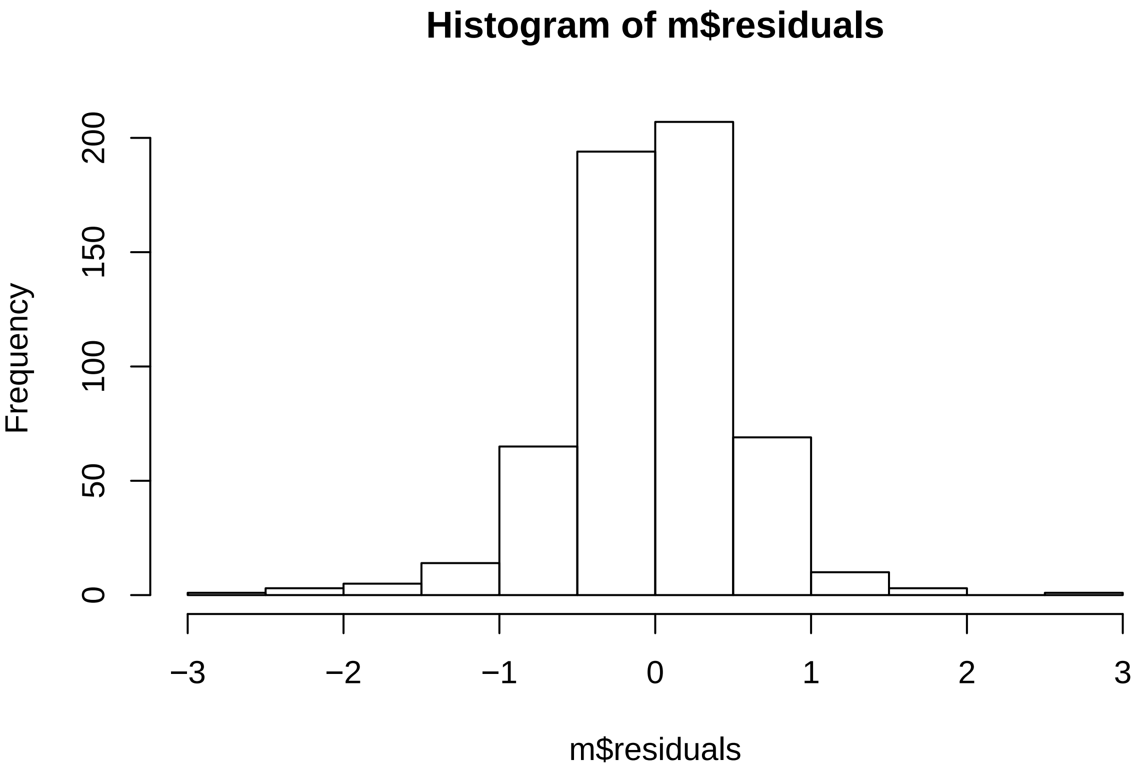
I will start by comparing the most complicated models' residuals against the simplest.

It turns out, that non of these models produce residuals that resemble white noise. That happens. We can compare these models based on their forecasting ability.

```r
for (p in 0:4) {
    for (q in 0:4) {
        m <- Arima(fp.training, order = c(p, 0, q), seasonal = list(order = c(0,
            1, 1)), method = "ML")
        print(c(p, q, m$aic))
    }
}
```

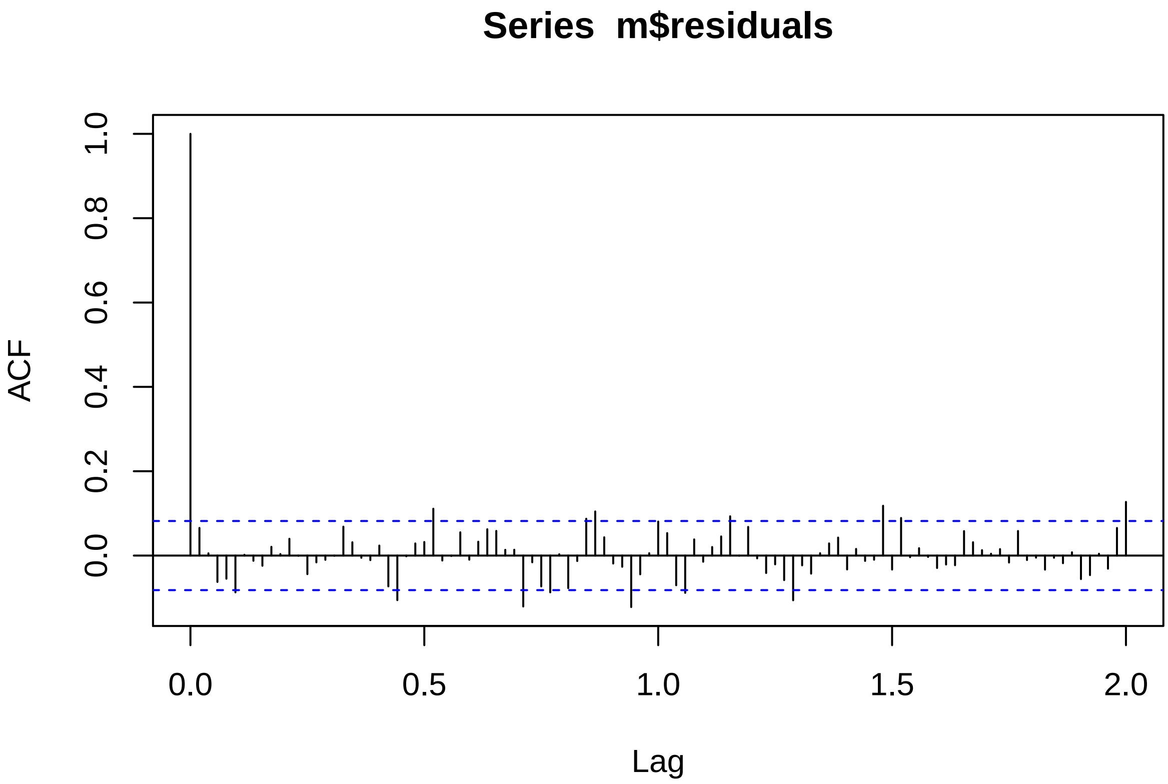
```
## [1]    0.000    0.000 1209.323
## [1]    0.000    1.000 1110.456
## [1]    0.000    2.000 1066.275
## [1]    0.000    3.000 1063.841
## [1]    0.000    4.000 1026.454
## [1]    1.000    0.000 1046.805
## [1]   1.0000   1.0000 989.8863
## [1]   1.0000   2.0000 984.6741
## [1]   1.000   3.000 985.212
## [1]   1.0000   4.0000 984.3203
## [1]    2.000    0.000 1021.523
## [1]   2.0000   1.0000 983.8524
## [1]   2.0000   2.0000 993.1502
## [1]   2.0000   3.0000 980.1366
## [1]   2.0000   4.0000 980.9286
## [1]    3.000    0.000 1014.333
## [1]   3.0000   1.0000 985.7029
## [1]   3.0000   2.0000 982.6628
## [1]   3.0000   3.0000 980.3775
## [1]   3.0000   4.0000 982.3748
## [1]   4.0000   0.0000 989.3635
## [1]   4.0   1.0 991.1
## [1]   4.0000   2.0000 982.6382
## [1]   4.0000   3.0000 982.3725
## [1]   4.0000   4.0000 981.0667
```

```r
print_resid_chart <- function(p, d, q, P, D, Q) {
    m <- Arima(fp.training, order = c(p, d, q), seasonal = list(order = c(P,
        D, Q)), method = "ML")

    print(hist(m$residuals))
    print(acf(m$residuals, 104))
    print(pacf(m$residuals, 104))
}

print_resid_chart(4, 0, 4, 0, 1, 1)
```

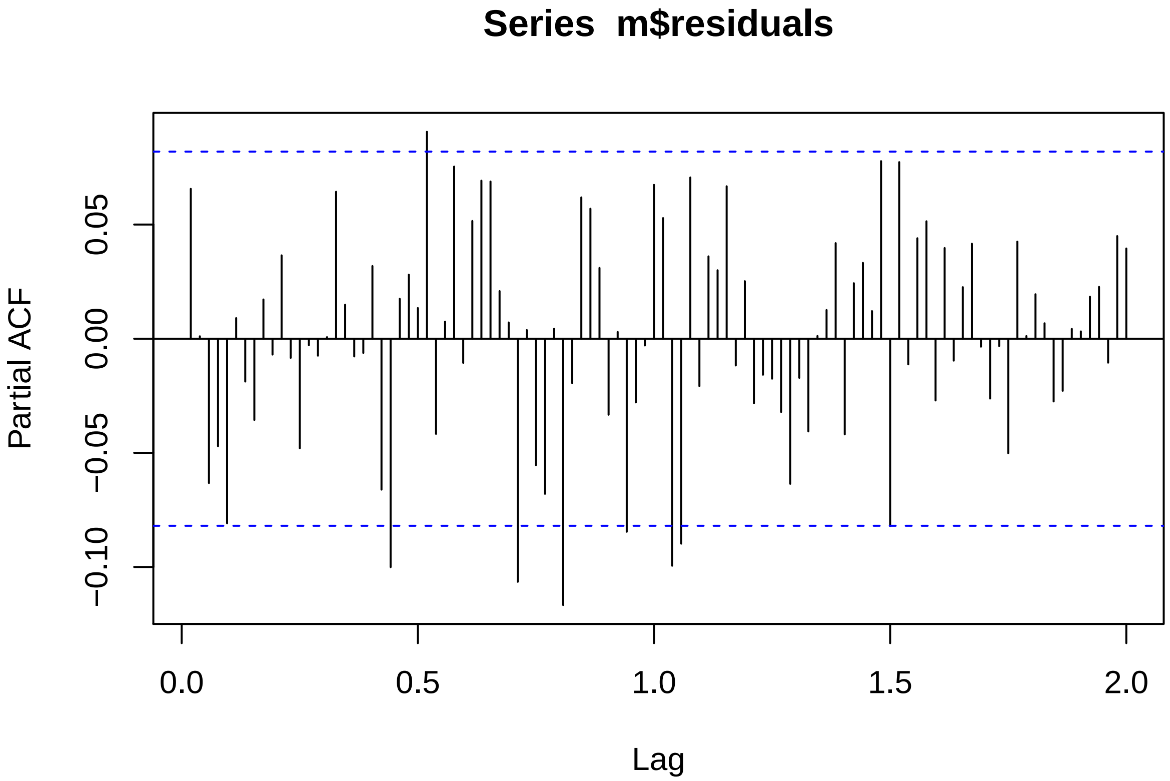


```
## $breaks
##  [1] -3.0 -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5  3.0
##
## $counts
##  [1]   1   3   5  14  65 194 207  69  10   3   0   1
##
## $density
##  [1] 0.003496503 0.010489510 0.017482517 0.048951049 0.227272727
##  [6] 0.678321678 0.723776224 0.241258741 0.034965035 0.010489510
## [11] 0.000000000 0.003496503
##
## $mids
##  [1] -2.75 -2.25 -1.75 -1.25 -0.75 -0.25  0.25  0.75  1.25  1.75  2.25
## [12]  2.75
##
## $xname
## [1] "m$residuals"
##
```

```
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

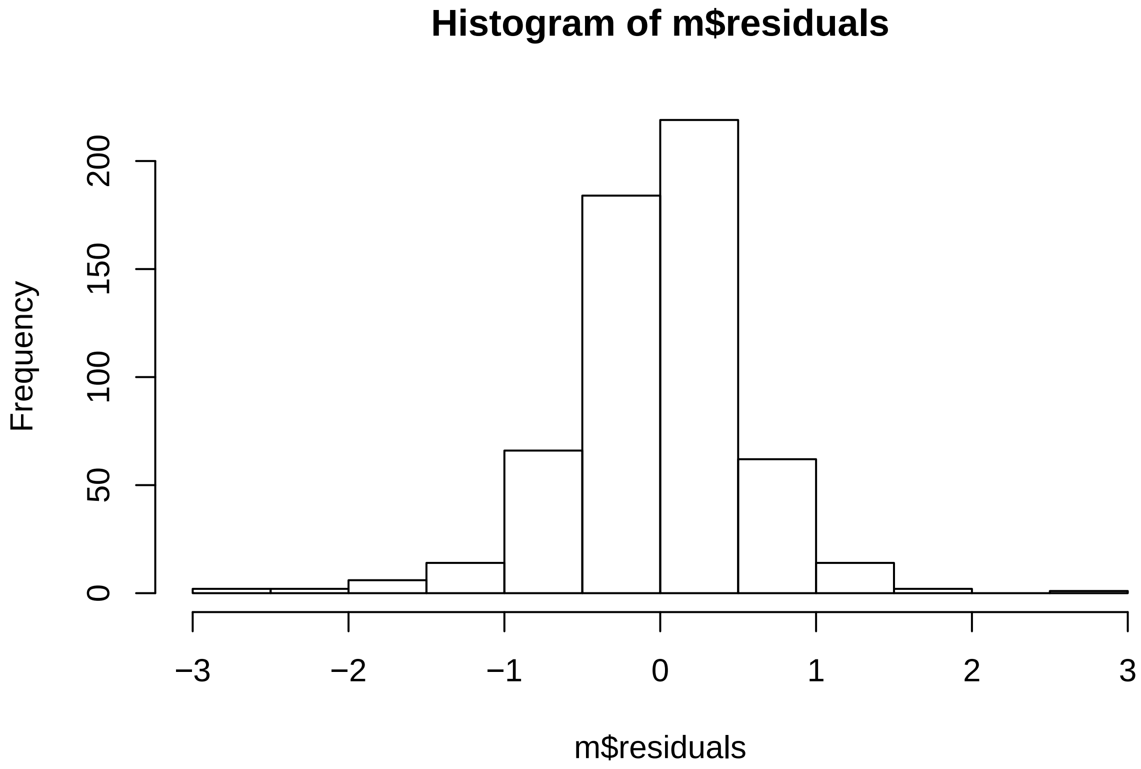


```
##
## Autocorrelations of series 'm$residuals', by lag
##
## 0.0000 0.0192 0.0385 0.0577 0.0769 0.0962 0.1154 0.1346 0.1538 0.1731
##  1.000  0.066  0.005 -0.063 -0.055 -0.087  0.002 -0.012 -0.024  0.021
## 0.1923 0.2115 0.2308 0.2500 0.2692 0.2885 0.3077 0.3269 0.3462 0.3654
##  0.004  0.040 -0.001 -0.044 -0.016 -0.010 -0.001  0.069  0.032 -0.005
## 0.3846 0.4038 0.4231 0.4423 0.4615 0.4808 0.5000 0.5192 0.5385 0.5577
## -0.011  0.024 -0.073 -0.106 -0.002  0.029  0.032  0.111 -0.012 -0.001
## 0.5769 0.5962 0.6154 0.6346 0.6538 0.6731 0.6923 0.7115 0.7308 0.7500
##  0.055 -0.010  0.033  0.062  0.058  0.014  0.014 -0.121 -0.016 -0.073
## 0.7692 0.7885 0.8077 0.8269 0.8462 0.8654 0.8846 0.9038 0.9231 0.9423
## -0.087  0.003 -0.077 -0.013  0.087  0.105  0.043 -0.019 -0.027 -0.122
## 0.9615 0.9808 1.0000 1.0192 1.0385 1.0577 1.0769 1.0962 1.1154 1.1346
## -0.044  0.006  0.081  0.053 -0.070 -0.088  0.038 -0.015  0.020  0.045
## 1.1538 1.1731 1.1923 1.2115 1.2308 1.2500 1.2692 1.2885 1.3077 1.3269
##  0.093  0.000  0.068 -0.007 -0.041 -0.021 -0.058 -0.106 -0.023 -0.043
## 1.3462 1.3654 1.3846 1.4038 1.4231 1.4423 1.4615 1.4808 1.5000 1.5192
##  0.006  0.029  0.043 -0.033  0.016 -0.013 -0.010  0.118 -0.033  0.089
## 1.5385 1.5577 1.5769 1.5962 1.6154 1.6346 1.6538 1.6731 1.6923 1.7115
## -0.004  0.018 -0.003 -0.029 -0.021 -0.023  0.058  0.032  0.013  0.004
## 1.7308 1.7500 1.7692 1.7885 1.8077 1.8269 1.8462 1.8654 1.8846 1.9038
##  0.015 -0.016  0.058 -0.011 -0.005 -0.033 -0.005 -0.018  0.008 -0.056
## 1.9231 1.9423 1.9615 1.9808 2.0000
## -0.046  0.004 -0.031  0.065  0.127
```

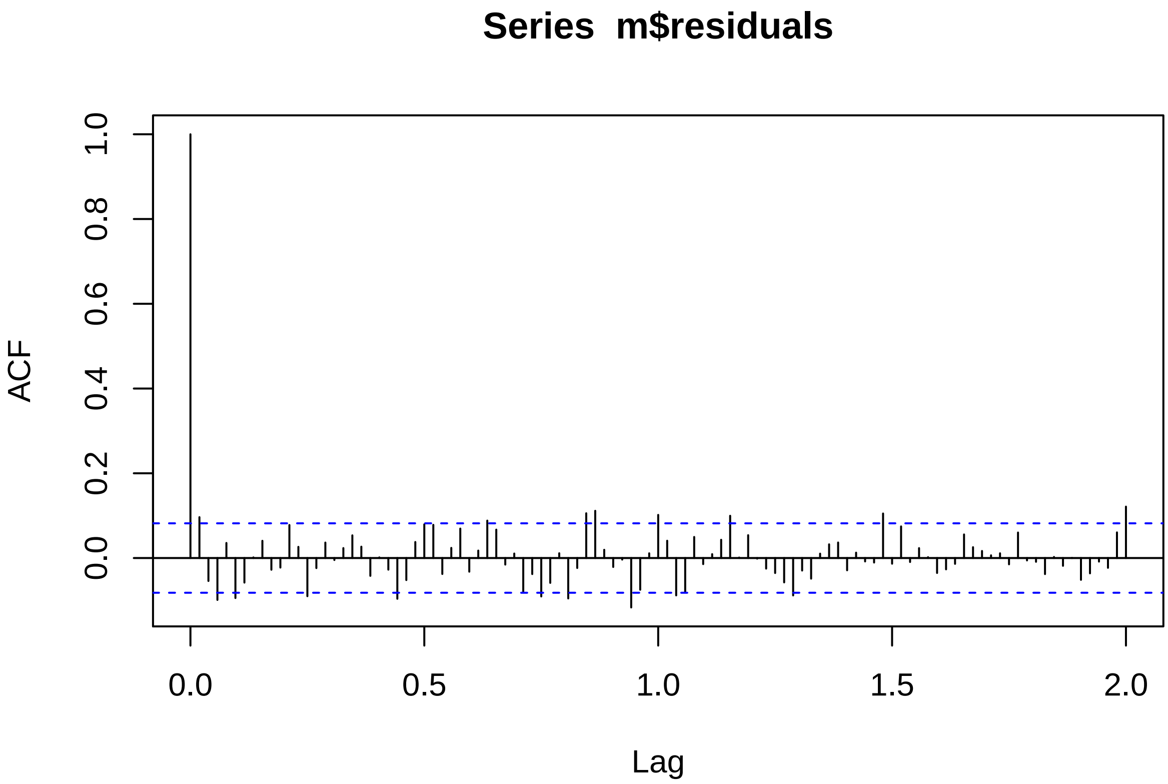


```
##
## Partial autocorrelations of series 'm$residuals', by lag
##
## 0.0192 0.0385 0.0577 0.0769 0.0962 0.1154 0.1346 0.1538 0.1731 0.1923
##  0.066  0.001 -0.063 -0.047 -0.081  0.009 -0.019 -0.036  0.017 -0.007
## 0.2115 0.2308 0.2500 0.2692 0.2885 0.3077 0.3269 0.3462 0.3654 0.3846
##  0.036 -0.008 -0.048 -0.003 -0.007  0.001  0.064  0.015 -0.008 -0.006
## 0.4038 0.4231 0.4423 0.4615 0.4808 0.5000 0.5192 0.5385 0.5577 0.5769
##  0.032 -0.066 -0.100  0.017  0.028  0.013  0.091 -0.042  0.007  0.075
## 0.5962 0.6154 0.6346 0.6538 0.6731 0.6923 0.7115 0.7308 0.7500 0.7692
## -0.011  0.052  0.069  0.069  0.021  0.007 -0.106  0.004 -0.055 -0.068
## 0.7885 0.8077 0.8269 0.8462 0.8654 0.8846 0.9038 0.9231 0.9423 0.9615
##  0.004 -0.117 -0.020  0.062  0.057  0.031 -0.033  0.003 -0.085 -0.028
## 0.9808 1.0000 1.0192 1.0385 1.0577 1.0769 1.0962 1.1154 1.1346 1.1538
## -0.003  0.067  0.053 -0.099 -0.090  0.071 -0.021  0.036  0.030  0.067
## 1.1731 1.1923 1.2115 1.2308 1.2500 1.2692 1.2885 1.3077 1.3269 1.3462
## -0.012  0.025 -0.028 -0.016 -0.018 -0.032 -0.064 -0.017 -0.041  0.001
## 1.3654 1.3846 1.4038 1.4231 1.4423 1.4615 1.4808 1.5000 1.5192 1.5385
##  0.013  0.042 -0.042  0.024  0.033  0.012  0.078 -0.082  0.077 -0.011
## 1.5577 1.5769 1.5962 1.6154 1.6346 1.6538 1.6731 1.6923 1.7115 1.7308
##  0.044  0.051 -0.027  0.040 -0.010  0.023  0.042 -0.004 -0.026 -0.003
## 1.7500 1.7692 1.7885 1.8077 1.8269 1.8462 1.8654 1.8846 1.9038 1.9231
## -0.050  0.043  0.001  0.019  0.007 -0.027 -0.023  0.004  0.003  0.018
## 1.9423 1.9615 1.9808 2.0000
##  0.023 -0.010  0.045  0.040
```

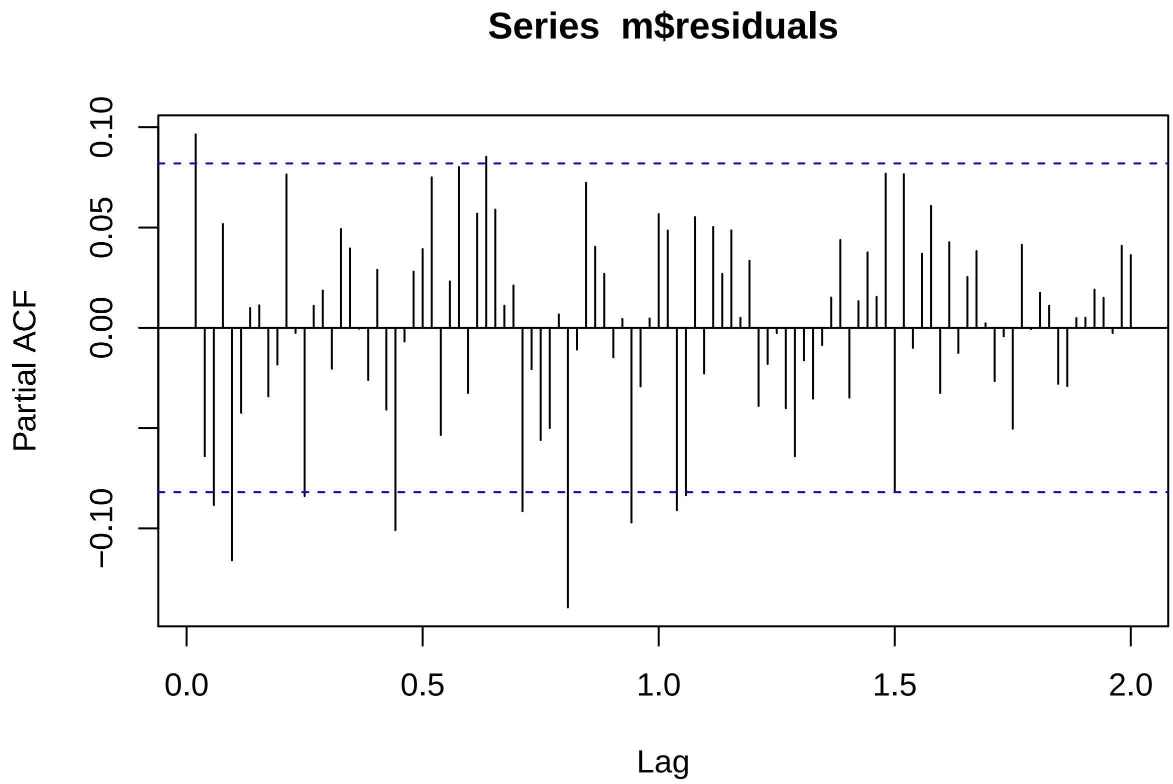
```
print_resid_chart(2, 0, 2, 0, 1, 1)
```



```
## $breaks
##  [1] -3.0 -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5  3.0
##
## $counts
##  [1]   2   2   6  14  66 184 219  62  14   2   0   1
##
## $density
##  [1] 0.006993007 0.006993007 0.020979021 0.048951049 0.230769231
##  [6] 0.643356643 0.765734266 0.216783217 0.048951049 0.006993007
## [11] 0.000000000 0.003496503
##
## $mids
##  [1] -2.75 -2.25 -1.75 -1.25 -0.75 -0.25  0.25  0.75  1.25  1.75  2.25
## [12]  2.75
##
## $xname
## [1] "m$residuals"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

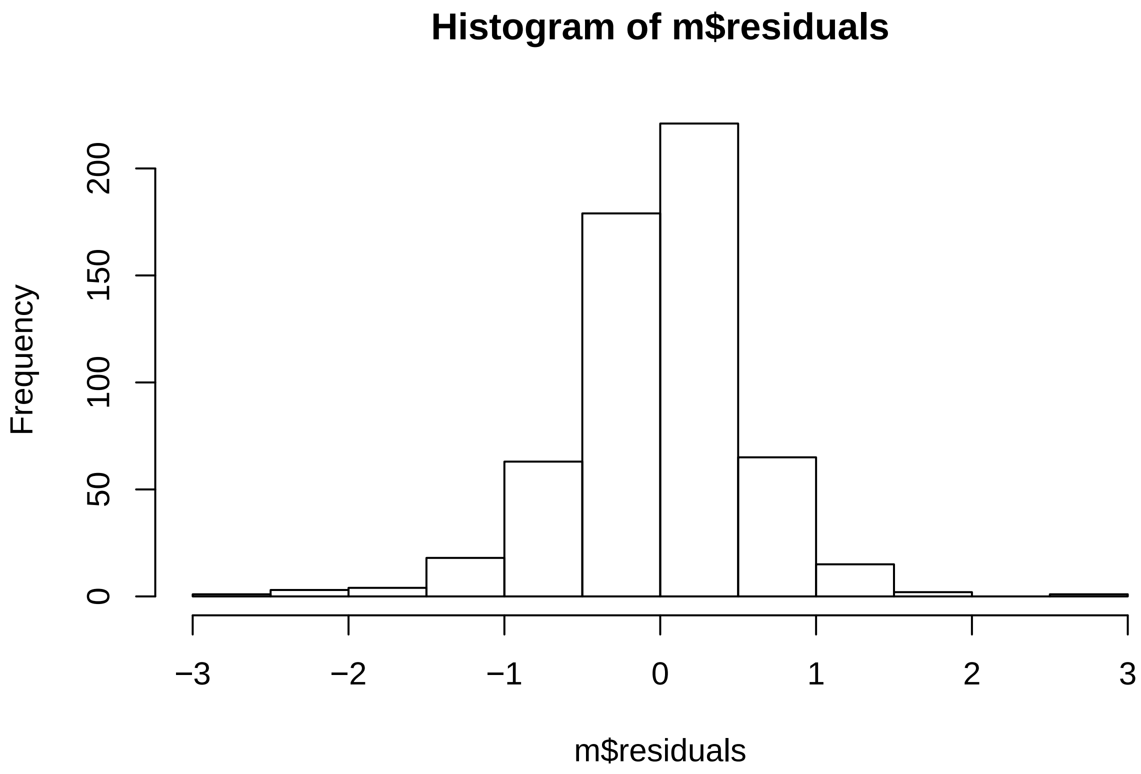


```
##
## Autocorrelations of series 'm$residuals', by lag
##
## 0.0000 0.0192 0.0385 0.0577 0.0769 0.0962 0.1154 0.1346 0.1538 0.1731
##  1.000  0.096 -0.054 -0.099  0.036 -0.094 -0.058  0.002  0.041 -0.028
## 0.1923 0.2115 0.2308 0.2500 0.2692 0.2885 0.3077 0.3269 0.3462 0.3654
## -0.023  0.078  0.026 -0.090 -0.024  0.036 -0.005  0.023  0.054  0.027
## 0.3846 0.4038 0.4231 0.4423 0.4615 0.4808 0.5000 0.5192 0.5385 0.5577
## -0.042  0.002 -0.027 -0.096 -0.052  0.038  0.080  0.078 -0.038  0.024
## 0.5769 0.5962 0.6154 0.6346 0.6538 0.6731 0.6923 0.7115 0.7308 0.7500
##  0.070 -0.032  0.018  0.088  0.067 -0.015  0.011 -0.079 -0.038 -0.091
## 0.7692 0.7885 0.8077 0.8269 0.8462 0.8654 0.8846 0.9038 0.9231 0.9423
## -0.059  0.011 -0.095 -0.023  0.106  0.111  0.019 -0.021 -0.004 -0.117
## 0.9615 0.9808 1.0000 1.0192 1.0385 1.0577 1.0769 1.0962 1.1154 1.1346
## -0.075  0.011  0.102  0.041 -0.088 -0.080  0.050 -0.014  0.009  0.043
## 1.1538 1.1731 1.1923 1.2115 1.2308 1.2500 1.2692 1.2885 1.3077 1.3269
##  0.100  0.001  0.054 -0.001 -0.025 -0.035 -0.057 -0.088 -0.030 -0.049
## 1.3462 1.3654 1.3846 1.4038 1.4231 1.4423 1.4615 1.4808 1.5000 1.5192
##  0.010  0.032  0.037 -0.029  0.013 -0.008 -0.011  0.105 -0.013  0.075
## 1.5385 1.5577 1.5769 1.5962 1.6154 1.6346 1.6538 1.6731 1.6923 1.7115
## -0.009  0.023  0.002 -0.035 -0.027 -0.014  0.056  0.026  0.017  0.006
## 1.7308 1.7500 1.7692 1.7885 1.8077 1.8269 1.8462 1.8654 1.8846 1.9038
##  0.011 -0.015  0.060 -0.006 -0.009 -0.038  0.003 -0.018  0.000 -0.051
## 1.9231 1.9423 1.9615 1.9808 2.0000
## -0.036 -0.008 -0.023  0.060  0.121
```

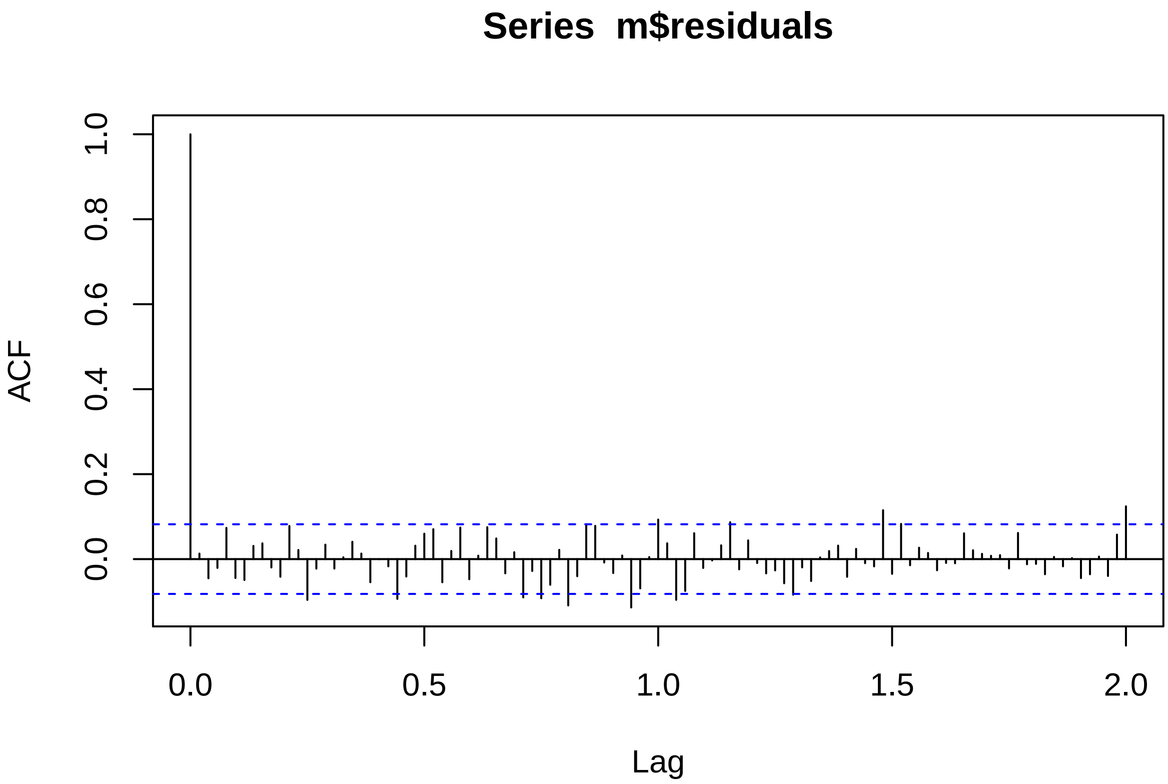


```
##
## Partial autocorrelations of series 'm$residuals', by lag
##
## 0.0192 0.0385 0.0577 0.0769 0.0962 0.1154 0.1346 0.1538 0.1731 0.1923
##  0.096 -0.064 -0.088  0.052 -0.116 -0.042  0.010  0.011 -0.034 -0.018
## 0.2115 0.2308 0.2500 0.2692 0.2885 0.3077 0.3269 0.3462 0.3654 0.3846
##  0.076 -0.003 -0.084  0.011  0.019 -0.020  0.049  0.040  0.000 -0.026
## 0.4038 0.4231 0.4423 0.4615 0.4808 0.5000 0.5192 0.5385 0.5577 0.5769
##  0.029 -0.041 -0.101 -0.007  0.028  0.039  0.075 -0.053  0.023  0.080
## 0.5962 0.6154 0.6346 0.6538 0.6731 0.6923 0.7115 0.7308 0.7500 0.7692
## -0.032  0.057  0.085  0.059  0.011  0.021 -0.091 -0.021 -0.056 -0.050
## 0.7885 0.8077 0.8269 0.8462 0.8654 0.8846 0.9038 0.9231 0.9423 0.9615
##  0.007 -0.139 -0.011  0.072  0.040  0.027 -0.015  0.004 -0.097 -0.029
## 0.9808 1.0000 1.0192 1.0385 1.0577 1.0769 1.0962 1.1154 1.1346 1.1538
##  0.005  0.057  0.049 -0.091 -0.083  0.055 -0.023  0.050  0.027  0.049
## 1.1731 1.1923 1.2115 1.2308 1.2500 1.2692 1.2885 1.3077 1.3269 1.3462
##  0.005  0.033 -0.039 -0.018 -0.003 -0.040 -0.064 -0.016 -0.035 -0.009
## 1.3654 1.3846 1.4038 1.4231 1.4423 1.4615 1.4808 1.5000 1.5192 1.5385
##  0.015  0.044 -0.035  0.013  0.038  0.015  0.077 -0.081  0.077 -0.010
## 1.5577 1.5769 1.5962 1.6154 1.6346 1.6538 1.6731 1.6923 1.7115 1.7308
##  0.037  0.061 -0.033  0.043 -0.013  0.025  0.038  0.002 -0.027 -0.004
## 1.7500 1.7692 1.7885 1.8077 1.8269 1.8462 1.8654 1.8846 1.9038 1.9231
## -0.050  0.041 -0.001  0.017  0.011 -0.028 -0.029  0.005  0.005  0.019
## 1.9423 1.9615 1.9808 2.0000
##  0.015 -0.003  0.041  0.036
```

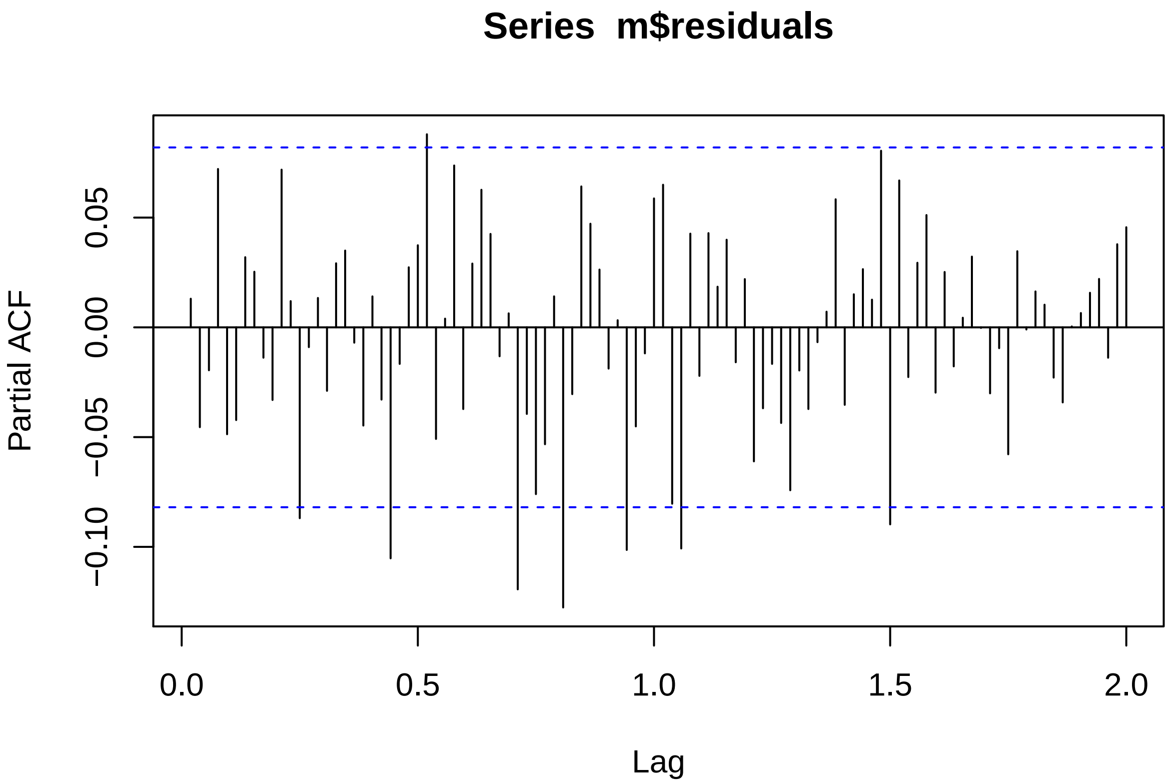
```
print_resid_chart(3, 0, 2, 0, 1, 1)
```



```
## $breaks
##  [1] -3.0 -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5  3.0
##
## $counts
##  [1]   1   3   4  18  63 179 221  65  15   2   0   1
##
## $density
##  [1] 0.003496503 0.010489510 0.013986014 0.062937063 0.220279720
##  [6] 0.625874126 0.772727273 0.227272727 0.052447552 0.006993007
## [11] 0.000000000 0.003496503
##
## $mids
##  [1] -2.75 -2.25 -1.75 -1.25 -0.75 -0.25  0.25  0.75  1.25  1.75  2.25
## [12]  2.75
##
## $xname
## [1] "m$residuals"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```

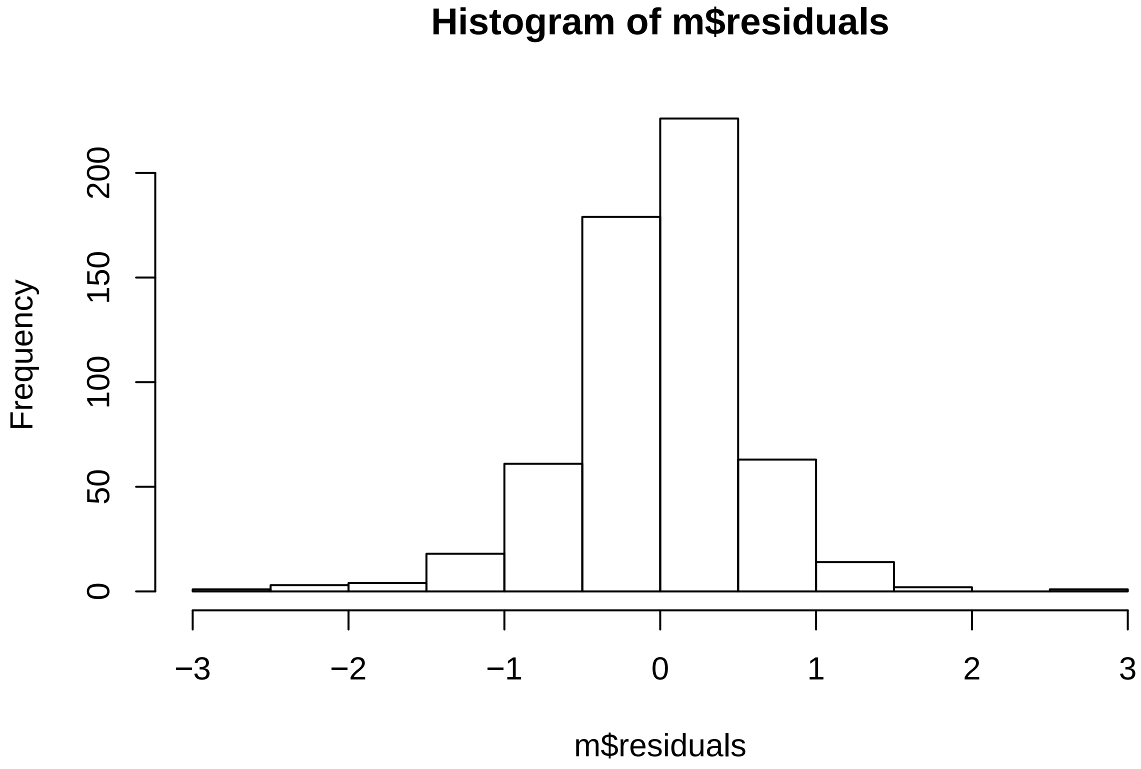


```
##
## Autocorrelations of series 'm$residuals', by lag
##
## 0.0000 0.0192 0.0385 0.0577 0.0769 0.0962 0.1154 0.1346 0.1538 0.1731
##  1.000  0.013 -0.045 -0.021  0.073 -0.045 -0.049  0.031  0.037 -0.020
## 0.1923 0.2115 0.2308 0.2500 0.2692 0.2885 0.3077 0.3269 0.3462 0.3654
## -0.042  0.078  0.021 -0.096 -0.022  0.034 -0.022  0.004  0.041  0.013
## 0.3846 0.4038 0.4231 0.4423 0.4615 0.4808 0.5000 0.5192 0.5385 0.5577
## -0.054  0.000 -0.017 -0.094 -0.041  0.032  0.060  0.070 -0.055  0.019
## 0.5769 0.5962 0.6154 0.6346 0.6538 0.6731 0.6923 0.7115 0.7308 0.7500
##  0.074 -0.047  0.008  0.075  0.049 -0.034  0.016 -0.090 -0.028 -0.092
## 0.7692 0.7885 0.8077 0.8269 0.8462 0.8654 0.8846 0.9038 0.9231 0.9423
## -0.060  0.022 -0.109 -0.040  0.080  0.078 -0.008 -0.033  0.008 -0.114
## 0.9615 0.9808 1.0000 1.0192 1.0385 1.0577 1.0769 1.0962 1.1154 1.1346
## -0.069  0.005  0.093  0.037 -0.096 -0.075  0.061 -0.021 -0.003  0.032
## 1.1538 1.1731 1.1923 1.2115 1.2308 1.2500 1.2692 1.2885 1.3077 1.3269
##  0.087 -0.024  0.044 -0.009 -0.034 -0.027 -0.057 -0.084 -0.020 -0.052
## 1.3462 1.3654 1.3846 1.4038 1.4231 1.4423 1.4615 1.4808 1.5000 1.5192
##  0.004  0.019  0.032 -0.042  0.024 -0.010 -0.017  0.115 -0.035  0.083
## 1.5385 1.5577 1.5769 1.5962 1.6154 1.6346 1.6538 1.6731 1.6923 1.7115
## -0.015  0.027  0.015 -0.026 -0.009 -0.010  0.061  0.021  0.012  0.008
## 1.7308 1.7500 1.7692 1.7885 1.8077 1.8269 1.8462 1.8654 1.8846 1.9038
##  0.009 -0.022  0.062 -0.012 -0.011 -0.036  0.005 -0.017  0.002 -0.045
## 1.9231 1.9423 1.9615 1.9808 2.0000
## -0.036  0.006 -0.040  0.058  0.124
```

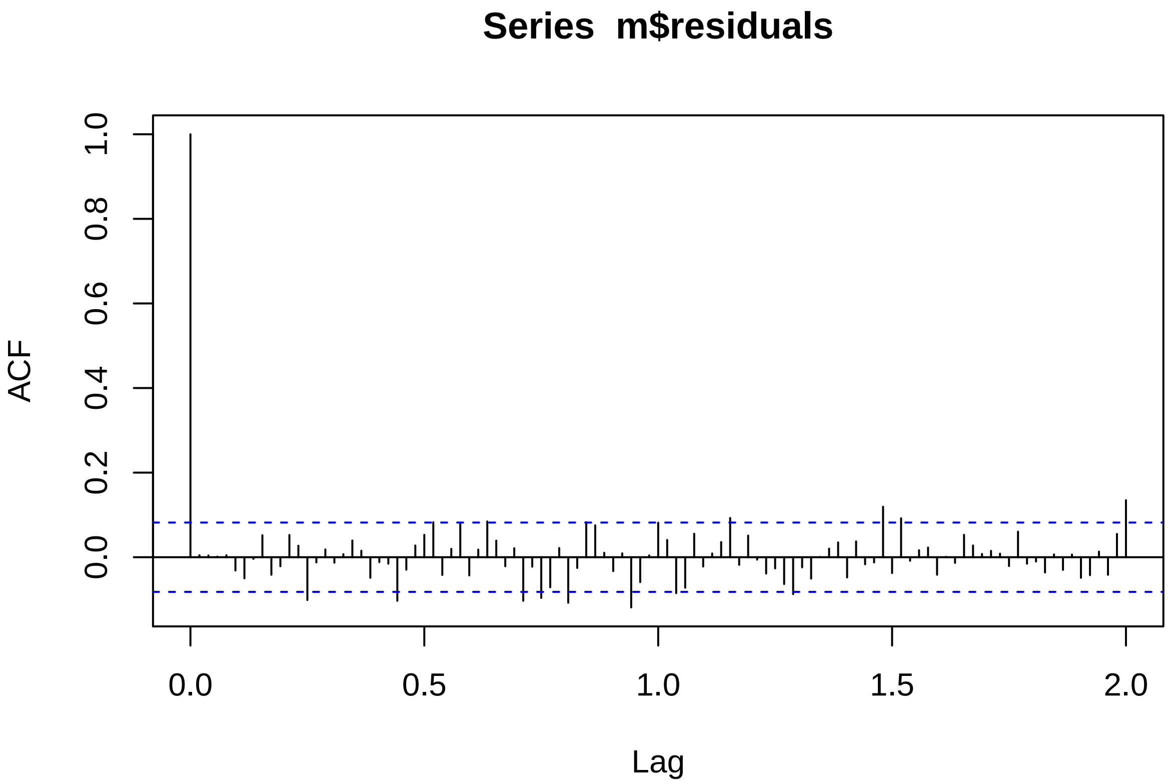


```
##
## Partial autocorrelations of series 'm$residuals', by lag
##
## 0.0192 0.0385 0.0577 0.0769 0.0962 0.1154 0.1346 0.1538 0.1731 0.1923
##  0.013 -0.045 -0.020  0.072 -0.049 -0.042  0.032  0.025 -0.014 -0.033
## 0.2115 0.2308 0.2500 0.2692 0.2885 0.3077 0.3269 0.3462 0.3654 0.3846
##  0.072  0.012 -0.087 -0.009  0.013 -0.029  0.029  0.035 -0.007 -0.045
## 0.4038 0.4231 0.4423 0.4615 0.4808 0.5000 0.5192 0.5385 0.5577 0.5769
##  0.014 -0.033 -0.105 -0.017  0.027  0.037  0.088 -0.051  0.004  0.074
## 0.5962 0.6154 0.6346 0.6538 0.6731 0.6923 0.7115 0.7308 0.7500 0.7692
## -0.037  0.029  0.063  0.043 -0.013  0.006 -0.119 -0.039 -0.076 -0.053
## 0.7885 0.8077 0.8269 0.8462 0.8654 0.8846 0.9038 0.9231 0.9423 0.9615
##  0.014 -0.128 -0.030  0.064  0.047  0.026 -0.019  0.003 -0.101 -0.045
## 0.9808 1.0000 1.0192 1.0385 1.0577 1.0769 1.0962 1.1154 1.1346 1.1538
## -0.012  0.059  0.065 -0.080 -0.101  0.043 -0.022  0.043  0.019  0.040
## 1.1731 1.1923 1.2115 1.2308 1.2500 1.2692 1.2885 1.3077 1.3269 1.3462
## -0.016  0.022 -0.061 -0.037 -0.017 -0.044 -0.074 -0.020 -0.037 -0.007
## 1.3654 1.3846 1.4038 1.4231 1.4423 1.4615 1.4808 1.5000 1.5192 1.5385
##  0.007  0.058 -0.035  0.015  0.027  0.013  0.080 -0.090  0.067 -0.023
## 1.5577 1.5769 1.5962 1.6154 1.6346 1.6538 1.6731 1.6923 1.7115 1.7308
##  0.029  0.051 -0.030  0.025 -0.018  0.004  0.032  0.000 -0.030 -0.009
## 1.7500 1.7692 1.7885 1.8077 1.8269 1.8462 1.8654 1.8846 1.9038 1.9231
## -0.058  0.035 -0.001  0.016  0.010 -0.023 -0.034  0.000  0.007  0.016
## 1.9423 1.9615 1.9808 2.0000
##  0.022 -0.014  0.038  0.046
```

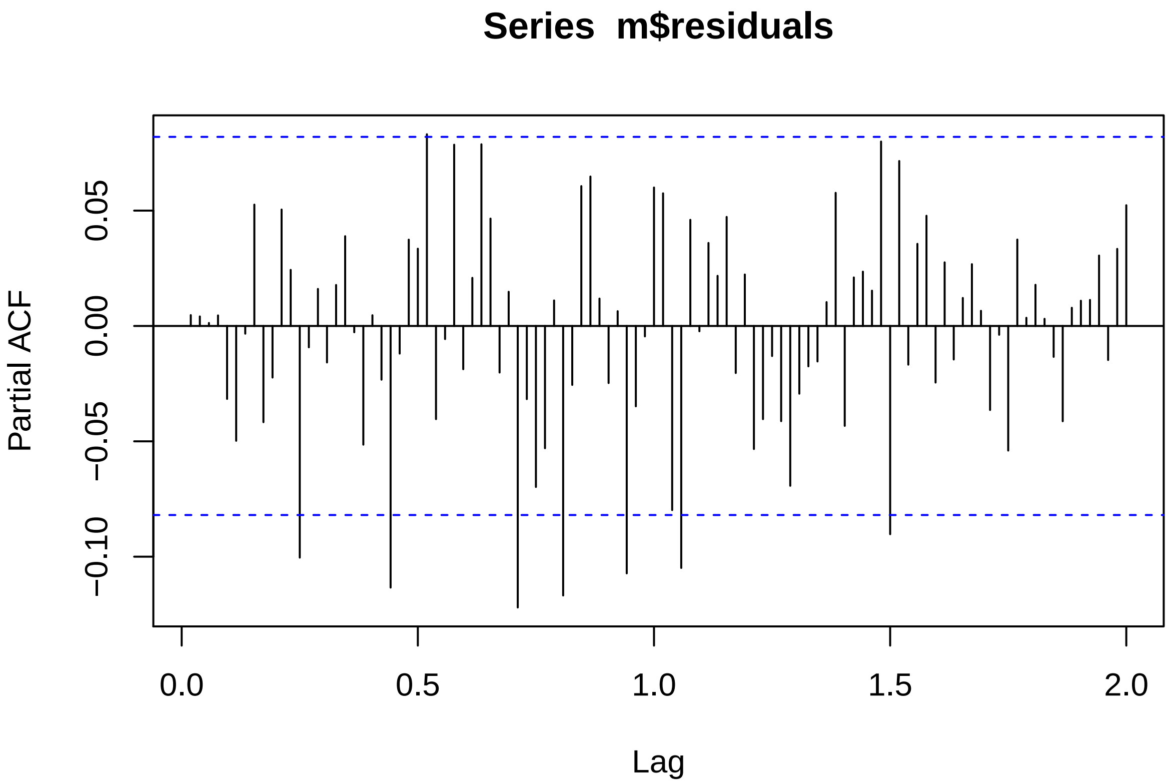
```
print_resid_chart(3, 0, 3, 0, 1, 1)
```



```
## $breaks
##  [1] -3.0 -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0  1.5  2.0  2.5  3.0
##
## $counts
##  [1]   1   3   4  18  61 179 226  63  14   2   0   1
##
## $density
##  [1] 0.003496503 0.010489510 0.013986014 0.062937063 0.213286713
##  [6] 0.625874126 0.790209790 0.220279720 0.048951049 0.006993007
## [11] 0.000000000 0.003496503
##
## $mids
##  [1] -2.75 -2.25 -1.75 -1.25 -0.75 -0.25  0.25  0.75  1.25  1.75  2.25
## [12]  2.75
##
## $xname
## [1] "m$residuals"
##
## $equidist
## [1] TRUE
##
## attr(,"class")
## [1] "histogram"
```



```
##
## Autocorrelations of series 'm$residuals', by lag
##
## 0.0000 0.0192 0.0385 0.0577 0.0769 0.0962 0.1154 0.1346 0.1538 0.1731
##  1.000  0.005  0.004  0.001  0.005 -0.032 -0.050 -0.004  0.052 -0.042
## 0.1923 0.2115 0.2308 0.2500 0.2692 0.2885 0.3077 0.3269 0.3462 0.3654
## -0.021  0.053  0.027 -0.101 -0.012  0.018 -0.013  0.007  0.040  0.015
## 0.3846 0.4038 0.4231 0.4423 0.4615 0.4808 0.5000 0.5192 0.5385 0.5577
## -0.049 -0.012 -0.015 -0.103 -0.030  0.028  0.053  0.083 -0.042  0.020
## 0.5769 0.5962 0.6154 0.6346 0.6538 0.6731 0.6923 0.7115 0.7308 0.7500
##  0.079 -0.043  0.018  0.085  0.039 -0.021  0.021 -0.103 -0.022 -0.096
## 0.7692 0.7885 0.8077 0.8269 0.8462 0.8654 0.8846 0.9038 0.9231 0.9423
## -0.071  0.022 -0.108 -0.025  0.083  0.075  0.010 -0.033  0.009 -0.119
## 0.9615 0.9808 1.0000 1.0192 1.0385 1.0577 1.0769 1.0962 1.1154 1.1346
## -0.059  0.004  0.082  0.041 -0.085 -0.072  0.056 -0.022  0.009  0.036
## 1.1538 1.1731 1.1923 1.2115 1.2308 1.2500 1.2692 1.2885 1.3077 1.3269
##  0.093 -0.018  0.051 -0.006 -0.039 -0.027 -0.063 -0.087 -0.024 -0.051
## 1.3462 1.3654 1.3846 1.4038 1.4231 1.4423 1.4615 1.4808 1.5000 1.5192
##  0.001  0.020  0.035 -0.048  0.037 -0.017 -0.013  0.119 -0.038  0.092
## 1.5385 1.5577 1.5769 1.5962 1.6154 1.6346 1.6538 1.6731 1.6923 1.7115
## -0.008  0.016  0.023 -0.042  0.001 -0.013  0.053  0.028  0.008  0.015
## 1.7308 1.7500 1.7692 1.7885 1.8077 1.8269 1.8462 1.8654 1.8846 1.9038
##  0.009 -0.021  0.060 -0.015 -0.010 -0.036  0.007 -0.030  0.006 -0.049
## 1.9231 1.9423 1.9615 1.9808 2.0000
## -0.043  0.013 -0.042  0.055  0.135
```



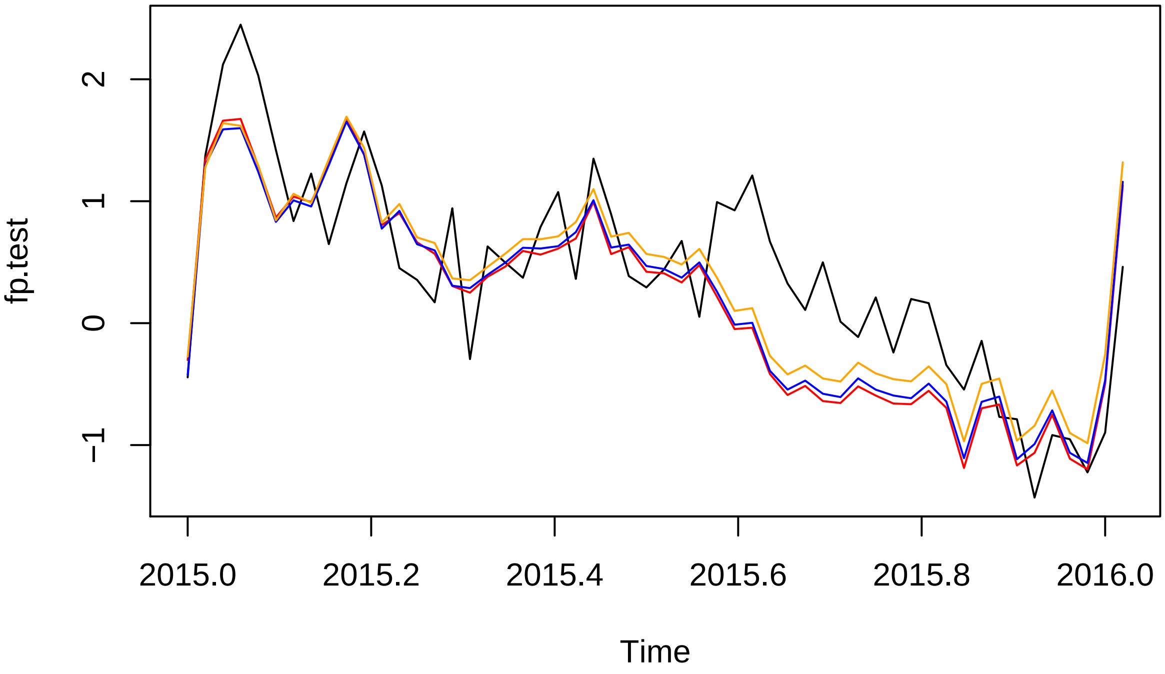
```
##
## Partial autocorrelations of series 'm$residuals', by lag
##
## 0.0192 0.0385 0.0577 0.0769 0.0962 0.1154 0.1346 0.1538 0.1731 0.1923
##  0.005  0.004  0.001  0.005 -0.032 -0.050 -0.003  0.053 -0.042 -0.022
## 0.2115 0.2308 0.2500 0.2692 0.2885 0.3077 0.3269 0.3462 0.3654 0.3846
##  0.050  0.024 -0.100 -0.009  0.016 -0.016  0.018  0.039 -0.003 -0.051
## 0.4038 0.4231 0.4423 0.4615 0.4808 0.5000 0.5192 0.5385 0.5577 0.5769
##  0.005 -0.023 -0.113 -0.012  0.037  0.033  0.083 -0.040 -0.006  0.079
## 0.5962 0.6154 0.6346 0.6538 0.6731 0.6923 0.7115 0.7308 0.7500 0.7692
## -0.019  0.021  0.079  0.047 -0.020  0.015 -0.122 -0.032 -0.070 -0.053
## 0.7885 0.8077 0.8269 0.8462 0.8654 0.8846 0.9038 0.9231 0.9423 0.9615
##  0.011 -0.117 -0.026  0.061  0.065  0.012 -0.025  0.006 -0.107 -0.035
## 0.9808 1.0000 1.0192 1.0385 1.0577 1.0769 1.0962 1.1154 1.1346 1.1538
## -0.005  0.060  0.057 -0.080 -0.105  0.046 -0.002  0.036  0.022  0.047
## 1.1731 1.1923 1.2115 1.2308 1.2500 1.2692 1.2885 1.3077 1.3269 1.3462
## -0.020  0.022 -0.053 -0.040 -0.013 -0.041 -0.069 -0.029 -0.018 -0.015
## 1.3654 1.3846 1.4038 1.4231 1.4423 1.4615 1.4808 1.5000 1.5192 1.5385
##  0.010  0.058 -0.043  0.021  0.024  0.015  0.080 -0.090  0.071 -0.017
## 1.5577 1.5769 1.5962 1.6154 1.6346 1.6538 1.6731 1.6923 1.7115 1.7308
##  0.036  0.048 -0.025  0.028 -0.015  0.012  0.027  0.007 -0.036 -0.004
## 1.7500 1.7692 1.7885 1.8077 1.8269 1.8462 1.8654 1.8846 1.9038 1.9231
## -0.054  0.037  0.004  0.018  0.003 -0.013 -0.041  0.008  0.011  0.011
## 1.9423 1.9615 1.9808 2.0000
##  0.031 -0.015  0.033  0.052
```

Forecasting

I graphed the forecasts of the simplest model, the most complicated, and also one where I differenced the non-seasonal component. The two non-differenced models seem to produce very similar forecasts, while the model with the differenced data produces a forecast that is a better fit of the data. If we were only concerned with forecasting accuracy, then I might select the one produced by the differenced data. However, if I am convinced that the non-seasonal difference was not needed, then I would select the simplest Sarima model I tested.

Note: Here, I compared the forecasts of a model with a differenced non-seasonal component without checking residuals. Strictly speaking, we should always check residuals.

```
### Check whether differencing is needed and its impact on the
### model
m1 <- Arima(fp.training, order = c(2, 0, 2), seasonal = list(order = c(0,
    1, 1)))

m2 <- Arima(fp.training, order = c(4, 0, 4), seasonal = list(order = c(0,
    1, 1)))

m3 <- Arima(fp.training, order = c(2, 1, 3), seasonal = list(order = c(0,
    1, 1)))

forecast1 <- forecast.Arima(m1, h = length(fp.test))
forecast2 <- forecast.Arima(m2, h = length(fp.test))
forecast3 <- forecast.Arima(m3, h = length(fp.test))

plot(fp.test)
lines(forecast1$mean, col = "red")
lines(forecast2$mean, col = "blue")
lines(forecast3$mean, col = "orange")
```


fp.test
Time