w271: Homework 8 (Due: Week 9)

Grace Lin

Due: By 4 p.m. Pacific Time on the Day of the Live Session in Week 9

Instructions (Please Read it Carefully!):

- **Page limit of the pdf report: None, but please be reasonable**
- Page setup:
  - Use the following font size, margin, and linespace:
    - fontsize=11pt
    - margin=1in
    - line_spacing=single
- Submission:
  - Homework needs to be completed individually; this is not a group project, but you are welcome to discuss with classmates.
  - Each student submits his/her homework to the course github repo by the deadline; submission and revision made after the deadline will not be graded
  - Submit 2 files:
    1. A pdf file that details your answers. Include all the R codes used to produce the answers. *Please do not suppress the codes in your pdf file.*
    2. R markdown file used to produce the pdf file
  - Use the following file-naming convensation; fail to do so will receive 10% reduction in the grade:
    - StudentFirstNameLastName_HWNumber.fileExtension
    - For example, if the student’s name is Kyle Cartman for homework 1, name your files as
      - KyleCartman_HW1.Rmd
      - KyleCartman_HW1.pdf
  - Although it sounds obvious, please write your name on page 1 of your pdf and Rmd files.
  - For statistical methods that we cover in this course, use only the R libraries and functions that are covered in this course. If you use libraries and functions for statistical modeling that we have not covered, you have to (1) provide an explanation of why such libraries and functions are used instead and (2) reference to the library documentation. **Lacking the explanation and reference to the documentation will result in a score of zero for the corresponding question.** For data wrangling and data visualization, you are free to use other libraries, such as dplyr, ggplot2, etc.
  - For mathematical formulae, type them in your R markdown file. **Do not write them on a piece of paper, snap a photo, and either insert the image file or sumbit the image file separately. Doing so will receive a 0 for that whole question.**
  - Students are expected to act with regards to UC Berkeley Academic Integrity.

In the last live session, I demonstrated a complete worked-out example and had you repeated the process using another series. In this week’s homework, you will continue with the exercise using another data series `series3.csv`. Your task is to (1) build a time series model using the `arima()` function and the class of ARMA model, which includes AR, MA, and ARMA models and (2) conduct a **3-step** ahead foreast.

The ARMA time series model building process, which I outlined both in the async lecture and demonstrated in the last live session (for the AR and MA models), typically includes (1) a time series EDA, which requires an examination of the stationarity of the series and a determination of whether the class of ARMA model is a “reasonable” model as a starting point, (2) model estimation (perhaps a few models need to be attempted), (3) model selection based on some metrics, say AIC, (4) model diagnostic, and (5) model forecast (after a valid model is found). You need to explain why certain AR/MA/ARMA model is chosen as a starting point based on your time series EDA.

```
# Clean up the workspace before we begin
rm(list = ls())

# Load required libraries
library(car)
library(dplyr)
library(Hmisc)
library(ggplot2)
library(ggfortify)
library(plotly)
library(astsa)
library(forecast)
library(fpp2)

# Set working directory
#wd = "YOUR DIRECTORY HERE"
# setwd(wd)
```

```
df <- read.csv("series3.csv", header = FALSE, sep=",")

# Examine the data structure
str(df)
```

```
## 'data.frame':    120 obs. of  1 variable:
##  $ V1: num  -0.877 0.43 1.173 -0.422 -2.384 ...
```

```
summary(df)
```

```
##        V1
##  Min.   :-6.7481
##  1st Qu.:-1.5763
##  Median :-0.2056
##  Mean   :-0.1509
##  3rd Qu.: 1.2307
##  Max.   : 5.0552
```

```
names(df)
```

```
## [1] "V1"
```

```
head(df)
```

	V1 <dbl>
1	-0.8773679
2	0.4300889
3	1.1726073
4	-0.4223776
5	-2.3841229
6	-3.8806242

6 rows

```
tail(df)
```

	V1 <dbl>
115	-1.1800922
116	-0.2028900
117	1.8177403
118	2.8730814
119	1.2931572
120	-0.6698399

6 rows

```
# Convert it into a time serie object
df.ts = ts(df, frequency = 12)



# Examine the converted data structure
str(df.ts)
```

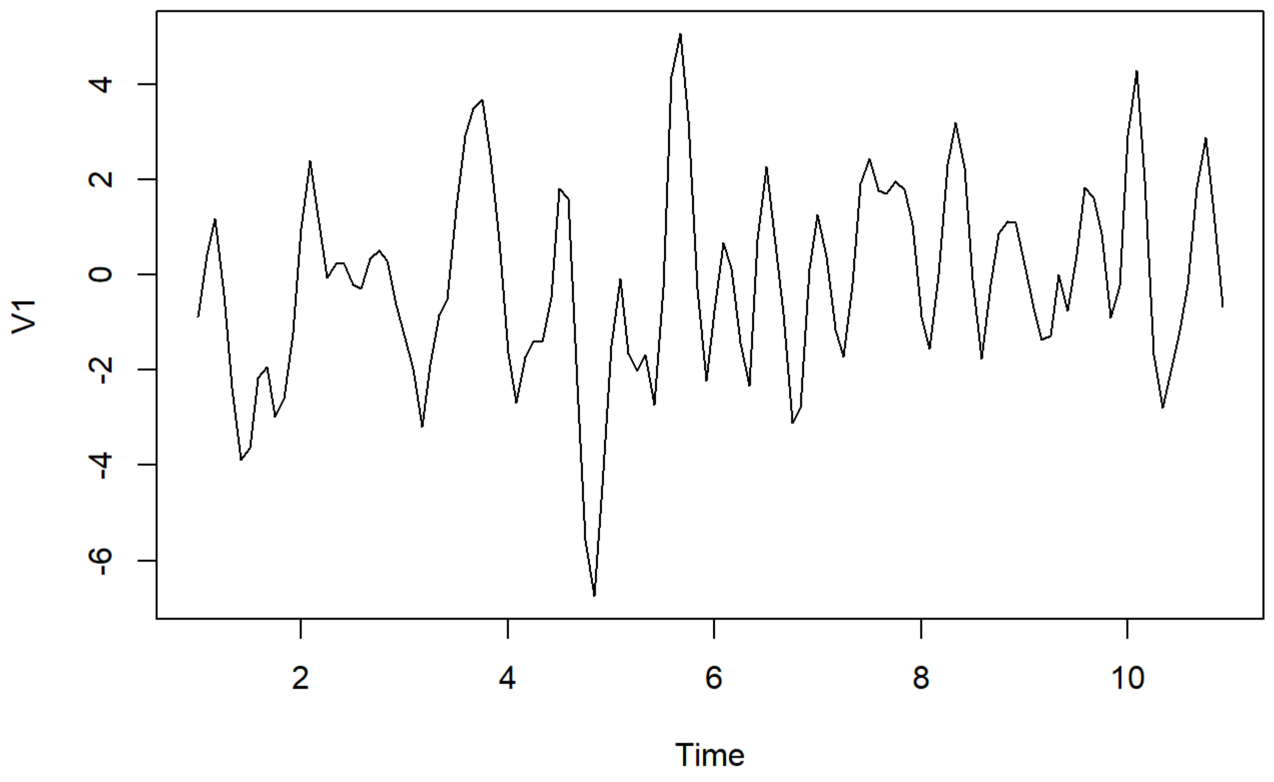
```
##  Time-Series [1:120, 1] from 1 to 10.9: -0.877 0.43 1.173 -0.422 -2.384 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr "V1"
```

```
head(cbind(time(df.ts),df.ts))
```

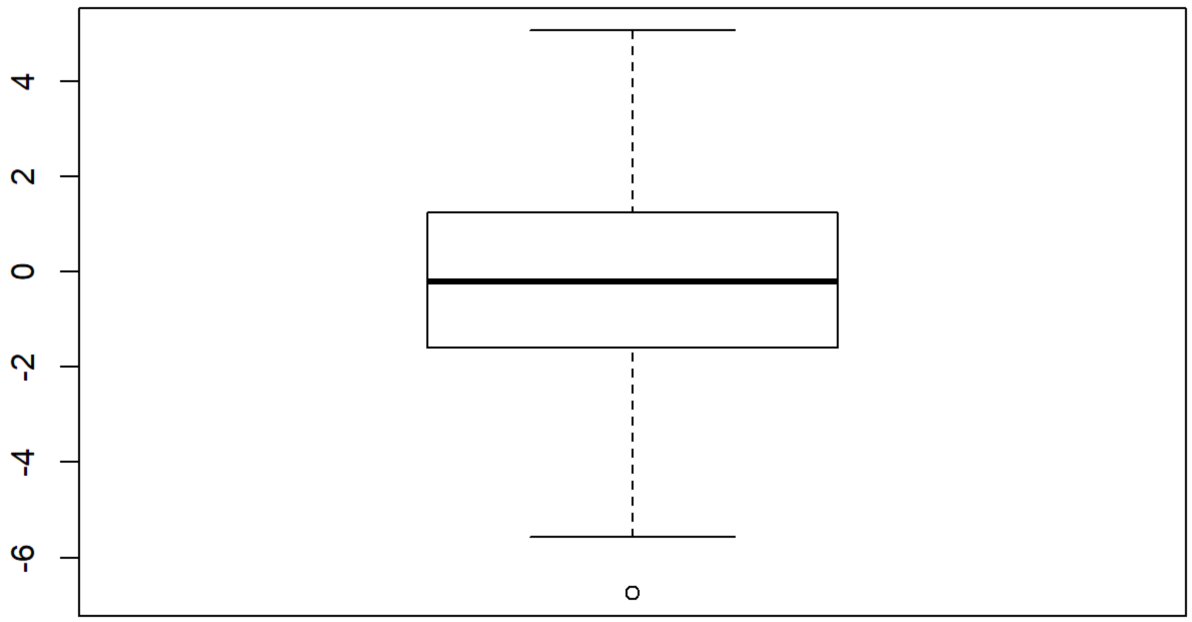
```
##       time(df.ts)      df.ts
## Jan 1    1.000000 -0.8773679
## Feb 1    1.083333  0.4300889
## Mar 1    1.166667  1.1726073
## Apr 1    1.250000 -0.4223776
## May 1    1.333333 -2.3841229
## Jun 1    1.416667 -3.8806242
```

Exploratory Time Series Data Analysis

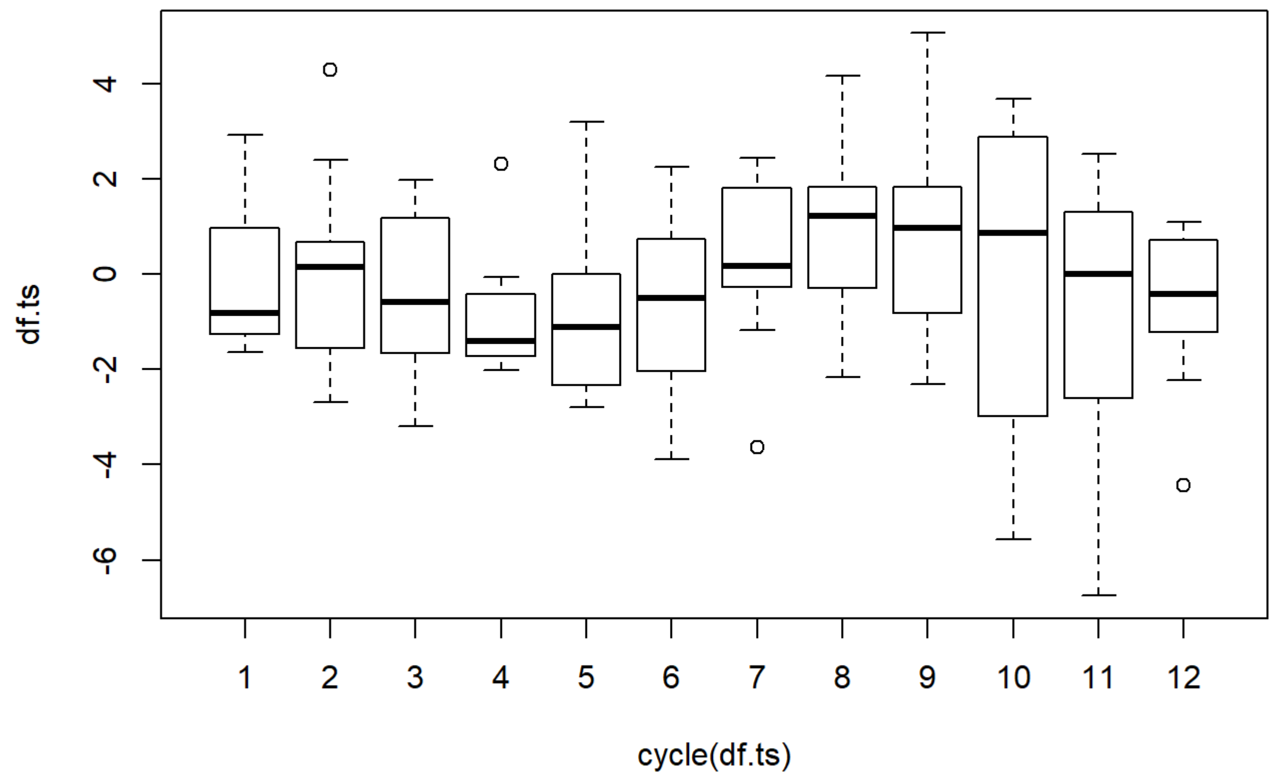
```
plot(df.ts)
```



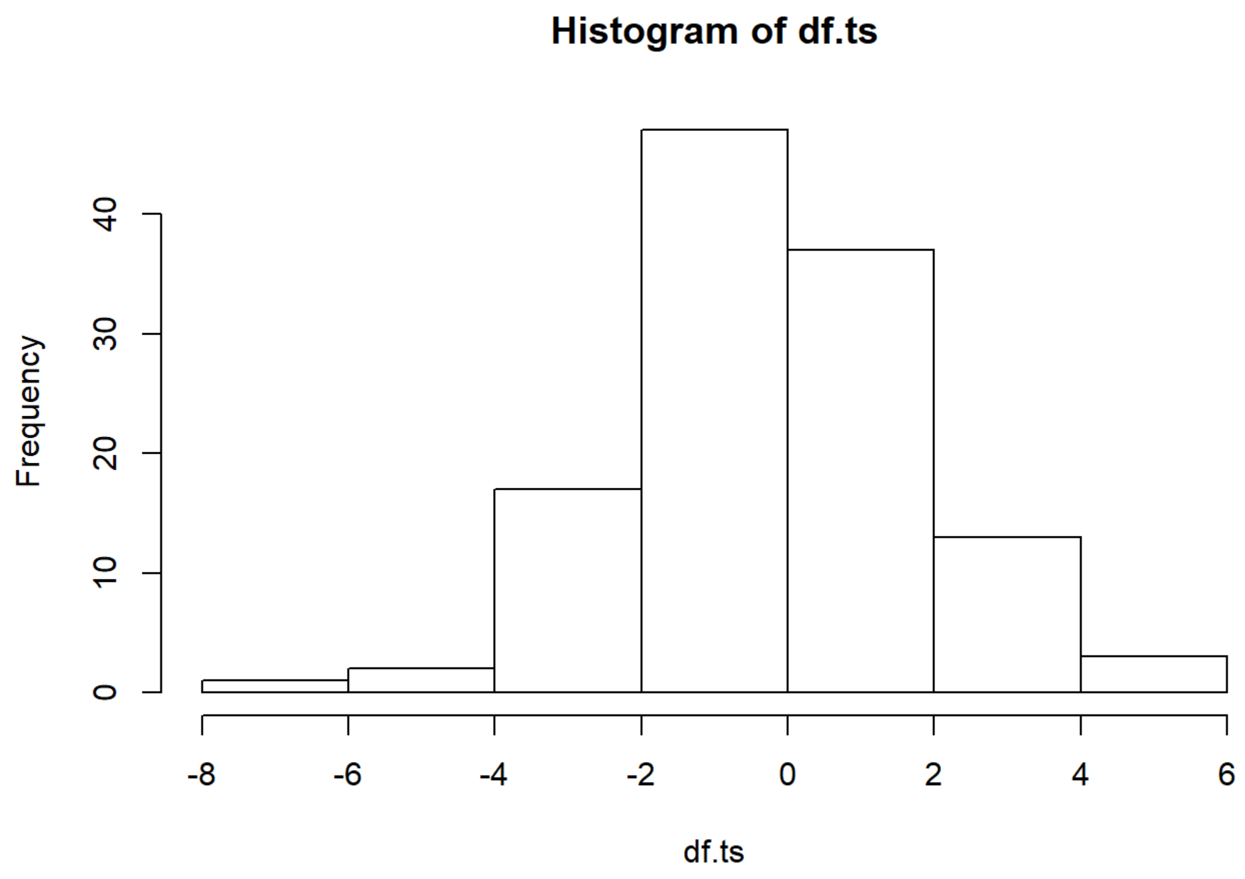
```
boxplot(df)
```



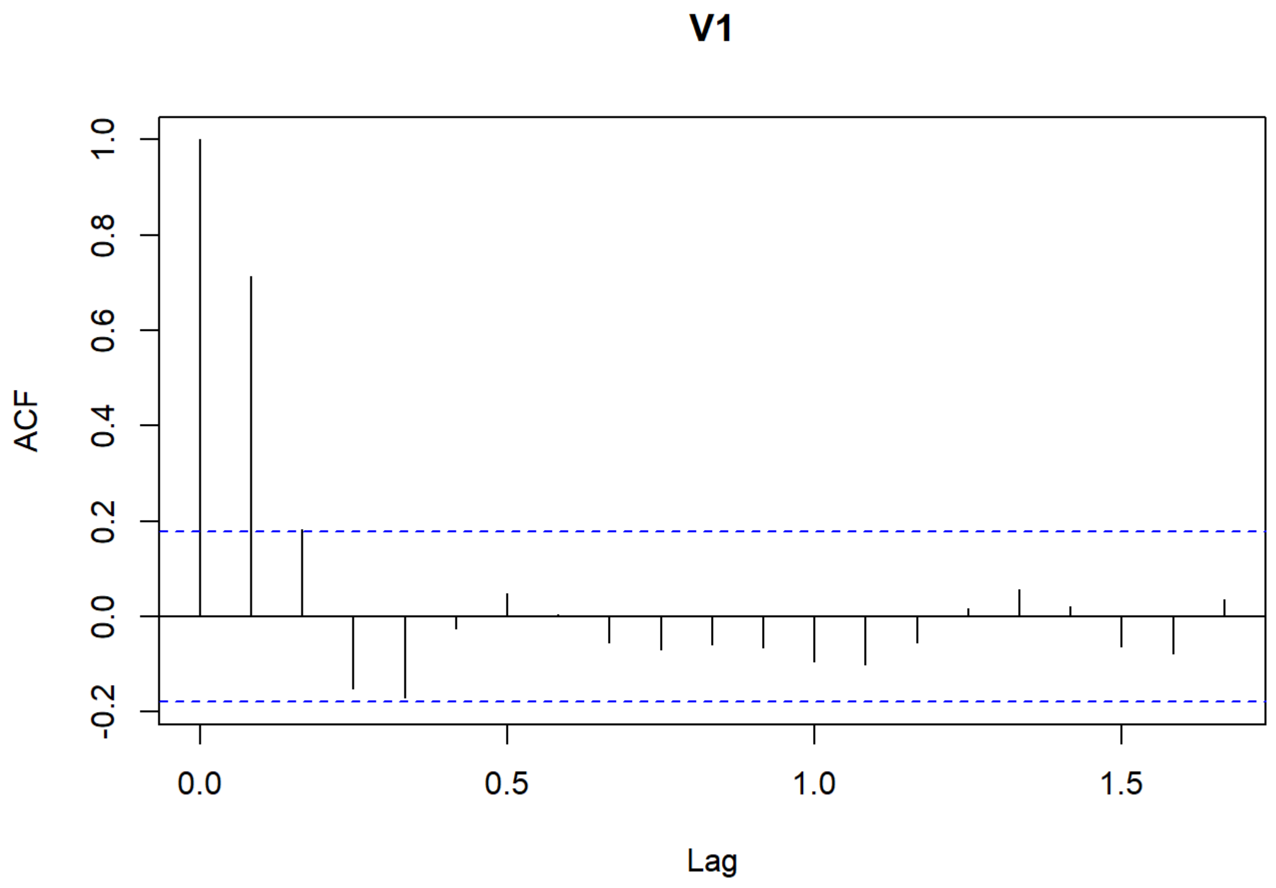
```
boxplot(df.ts~cycle(df.ts))
```



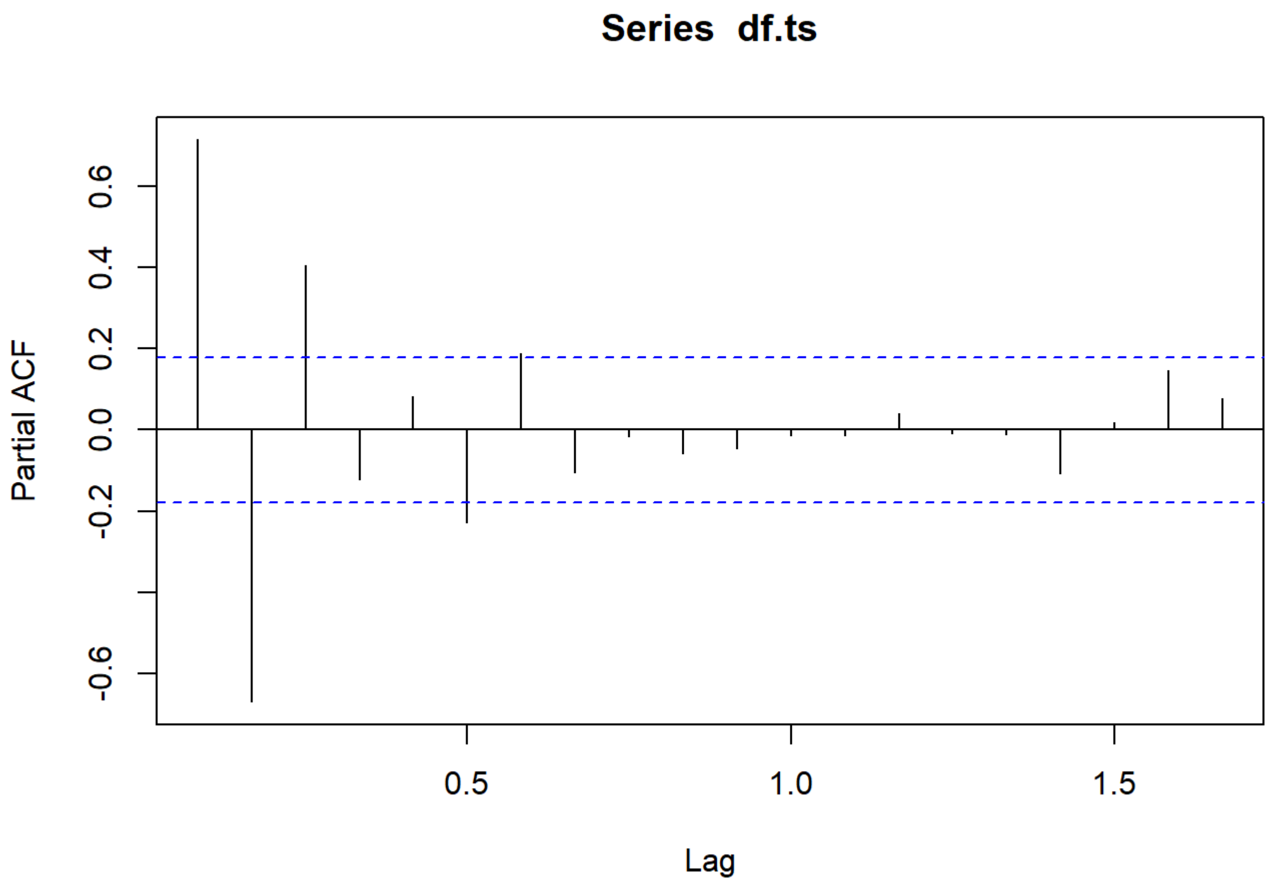
```
hist(df.ts)
```



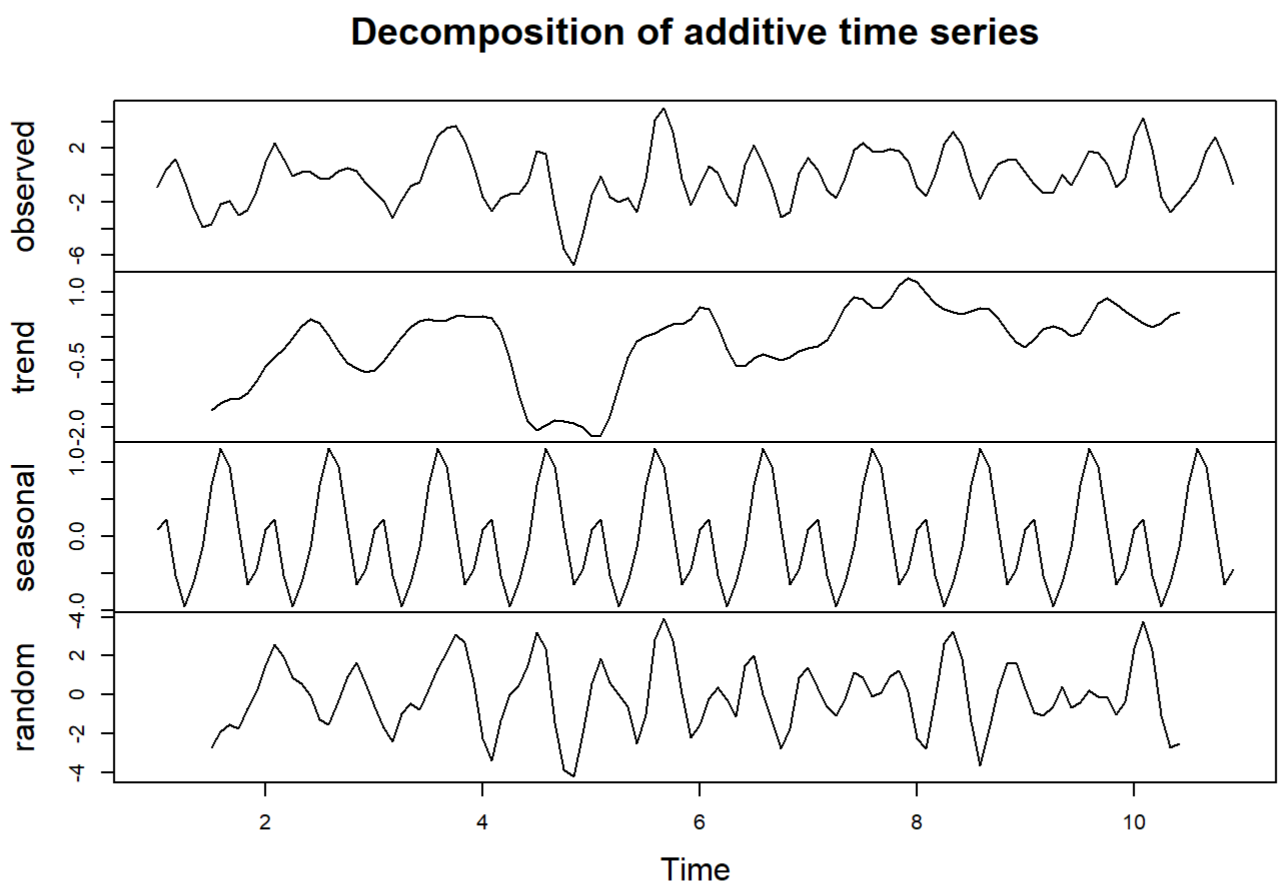
```
acf(df.ts)
```



```
pacf(df.ts)
```



```
plot(decompose(df.ts))
```



```
# Results in ACF of oscillating to 0 is indicative of AR or ARMA model
# Results in PACF that cuts off after 3 indicative of AR(3) model
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

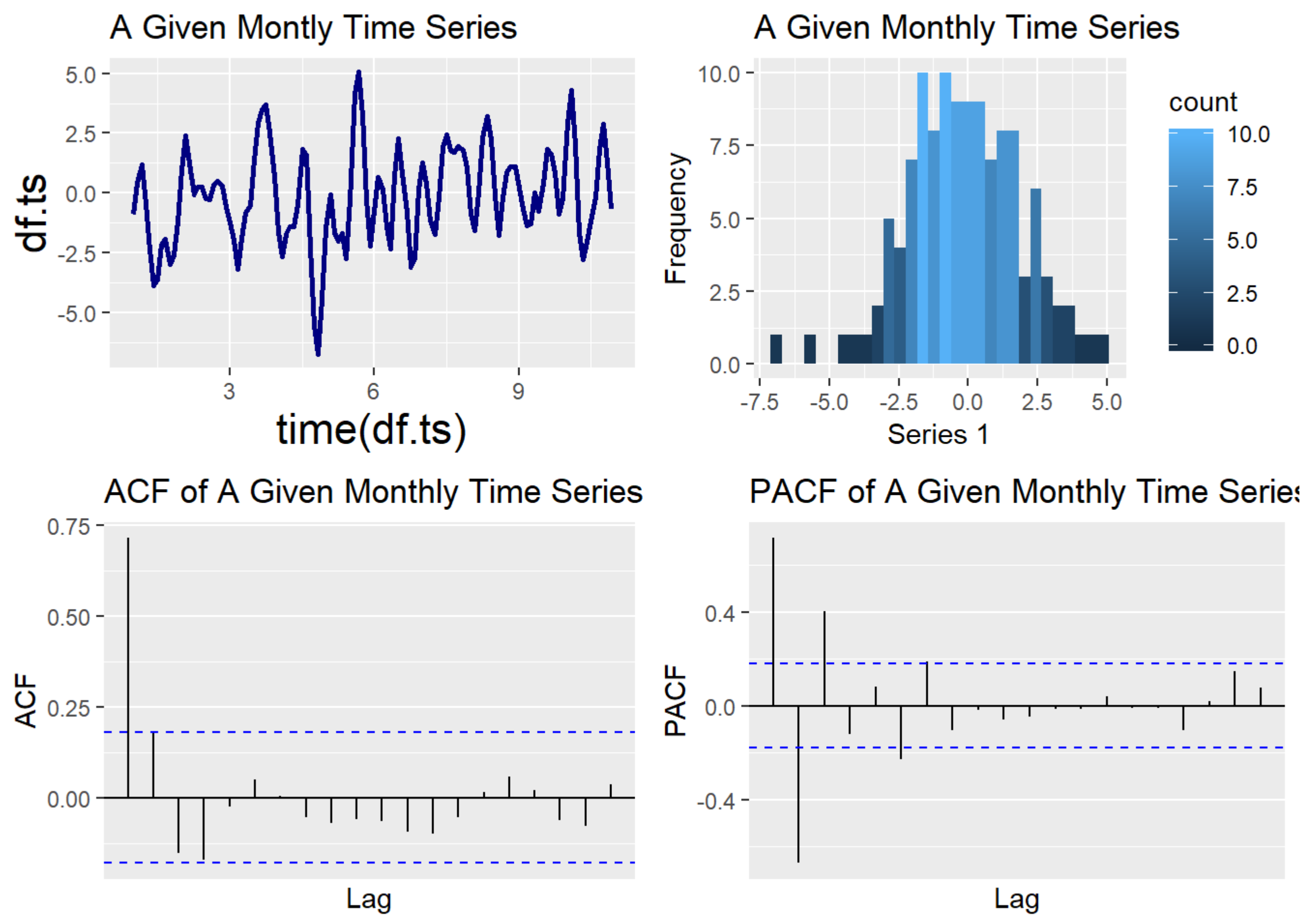
```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(grid)

#autoplot(acf(series1, plot = FALSE)) +
#  ggtitle("ACF of A Given Monthly Time Series")
#ggplotly()

p1 = ggplot(df.ts,
       aes(x=time(df.ts), y=df.ts)) +
  geom_line(colour = "navy", size = 1) +
  ggtitle("A Given Montly Time Series") +
  theme(axis.title = element_text(size = rel(1.5)))

p2 = ggplot(df, aes(x=V1)) +
  geom_histogram(aes(fill = ..count..)) +
  ggtitle("A Given Monthly Time Series") +
  xlab("Series 1") + ylab("Frequency")

p3 = autoplot(acf(df.ts, plot = FALSE)) +
  ggtitle("ACF of A Given Monthly Time Series")

p4 = autoplot(pacf(df.ts, plot = FALSE)) +
  ggtitle("PACF of A Given Monthly Time Series")

grid.arrange(p1, p2, p3, p4, ncol=2)
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Model Estimation and Diagnostics

```
model = ar(x = df.ts, order.max = 12, method="yule-walker")
model$order
```

```
## [1] 7
```

```
model$ar
```

```
## [1]  1.5835084 -1.4748837  0.9466899 -0.7080646  0.6901272 -0.5174848
## [7]  0.1879185
```

```
model$aic
```

```
##           0           1           2           3           4           5
## 177.7262571  94.0494078  24.6803796   5.3023812   5.5097764   6.7182531
##           6           7           8           9          10          11
##   2.3142337   0.0000000   0.6780434   2.6437619   4.2461707   5.9914460
##          12
##   7.9703400
```

```
# started with AR(3) based on the patterns exhibited in acf and pacf
ar.fit1 <- arima(df.ts, c(ar=3,0,0))
summary(ar.fit1)
```

```
##
## Call:
## arima(x = df.ts, order = c(ar = 3, 0, 0))
##
## Coefficients:
##          ar1      ar2     ar3  intercept
##       1.4841  -1.1800  0.4168    -0.1693
## s.e.  0.0824   0.1156  0.0824     0.3061
##
## sigma^2 estimated as 0.8943:  log likelihood = -164.83,  aic = 339.67
##
## Training set error measures:
##                       ME      RMSE       MAE       MPE     MAPE     MASE
## Training set 0.003207793 0.9456863 0.7469049 -178.5299 293.1979 0.590652
##                   ACF1
## Training set 0.0565725
```

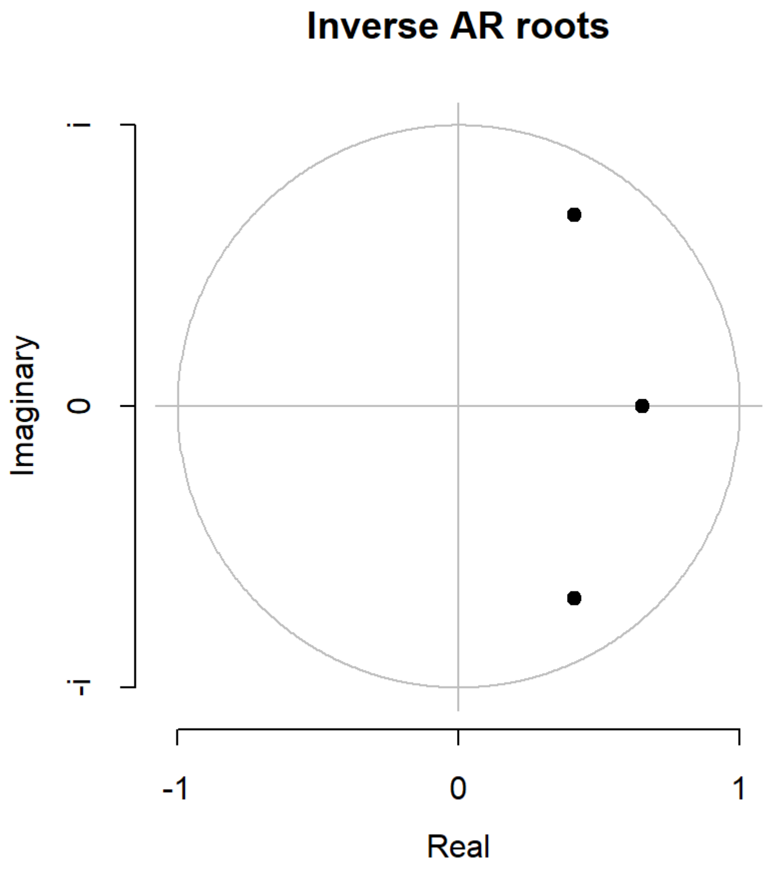
```
str(ar.fit1)
```

```
## List of 14
##  $ coef     : Named num [1:4] 1.484 -1.18 0.417 -0.169
##   ..- attr(*, "names")= chr [1:4] "ar1" "ar2" "ar3" "intercept"
##  $ sigma2   : num 0.894
##  $ var.coef : num [1:4, 1:4] 6.79e-03 -8.14e-03 4.59e-03 5.05e-05 -8.14e-03 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:4] "ar1" "ar2" "ar3" "intercept"
##   .. ..$ : chr [1:4] "ar1" "ar2" "ar3" "intercept"
##  $ mask     : logi [1:4] TRUE TRUE TRUE TRUE
##  $ loglik   : num -165
##  $ aic      : num 340
##  $ arma     : int [1:7] 3 0 0 0 12 0 0
##  $ residuals: Time-Series [1:120] from 1 to 10.9: -0.33 0.738 0.128 -1.242 -0.506 ...
##  $ call     : language arima(x = df.ts, order = c(ar = 3, 0, 0))
##  $ series   : chr "df.ts"
##  $ code     : int 0
##  $ n.cond   : int 0
##  $ nobs     : int 120
##  $ model    :List of 10
##   ..$ phi  : num [1:3] 1.484 -1.18 0.417
##   ..$ theta: num [1:2] 0 0
##   ..$ Delta: num(0)
##   ..$ Z    : num [1:3] 1 0 0
##   ..$ a    : num [1:3] -0.501 -0.457 0.61
##   ..$ P    : num [1:3, 1:3] 0 0 0 0 0 0 0 0 0
##   ..$ T    : num [1:3, 1:3] 1.484 -1.18 0.417 1 0 ...
##   ..$ V    : num [1:3, 1:3] 1 0 0 0 0 0 0 0 0
##   ..$ h    : num 0
##   ..$ Pn   : num [1:3, 1:3] 1 0 0 0 0 0 0 0 0
##  - attr(*, "class")= chr "Arima"
```

```
# Note that auto.arima(), by default, select model based on AIC
library(forecast)
auto.arima(df.ts)
```

```
## Series: df.ts
## ARIMA(2,0,1) with zero mean
##
## Coefficients:
##          ar1      ar2     ma1
##       0.8930  -0.4534  0.7386
## s.e.  0.0908   0.0889  0.0696
##
## sigma^2 estimated as 0.8546:  log likelihood=-160.84
## AIC=329.67   AICc=330.02   BIC=340.82
```

```
# Examine the roots of the Characteristics equation
plot(ar.fit1)
```



```
Mod(polyroot(c(ar.fit1$coef)))
```

```
## [1] 1.573327 2.360253 2.360253
```

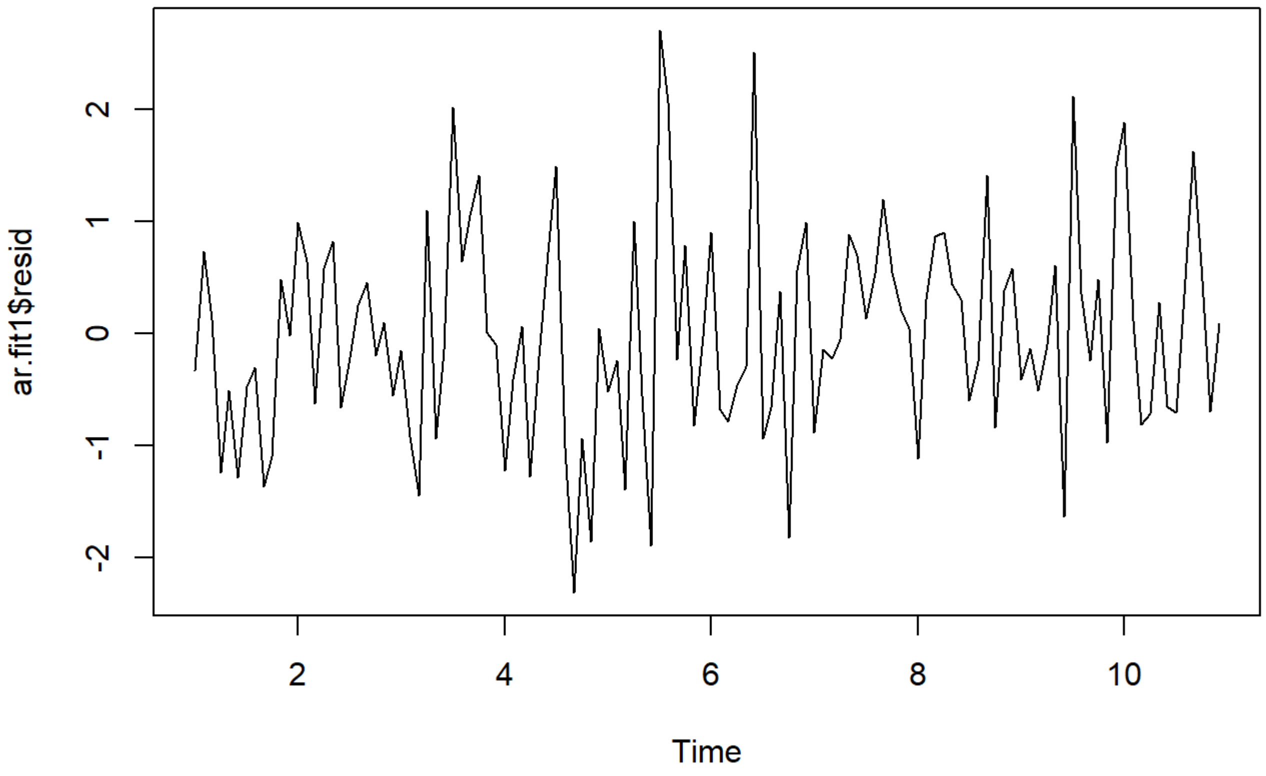
```
# Model Evaluation (Residual Diagnostic)
summary(ar.fit1$resid)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max. 
## -2.310302 -0.654685 -0.069477  0.003208  0.581047  2.702560
```

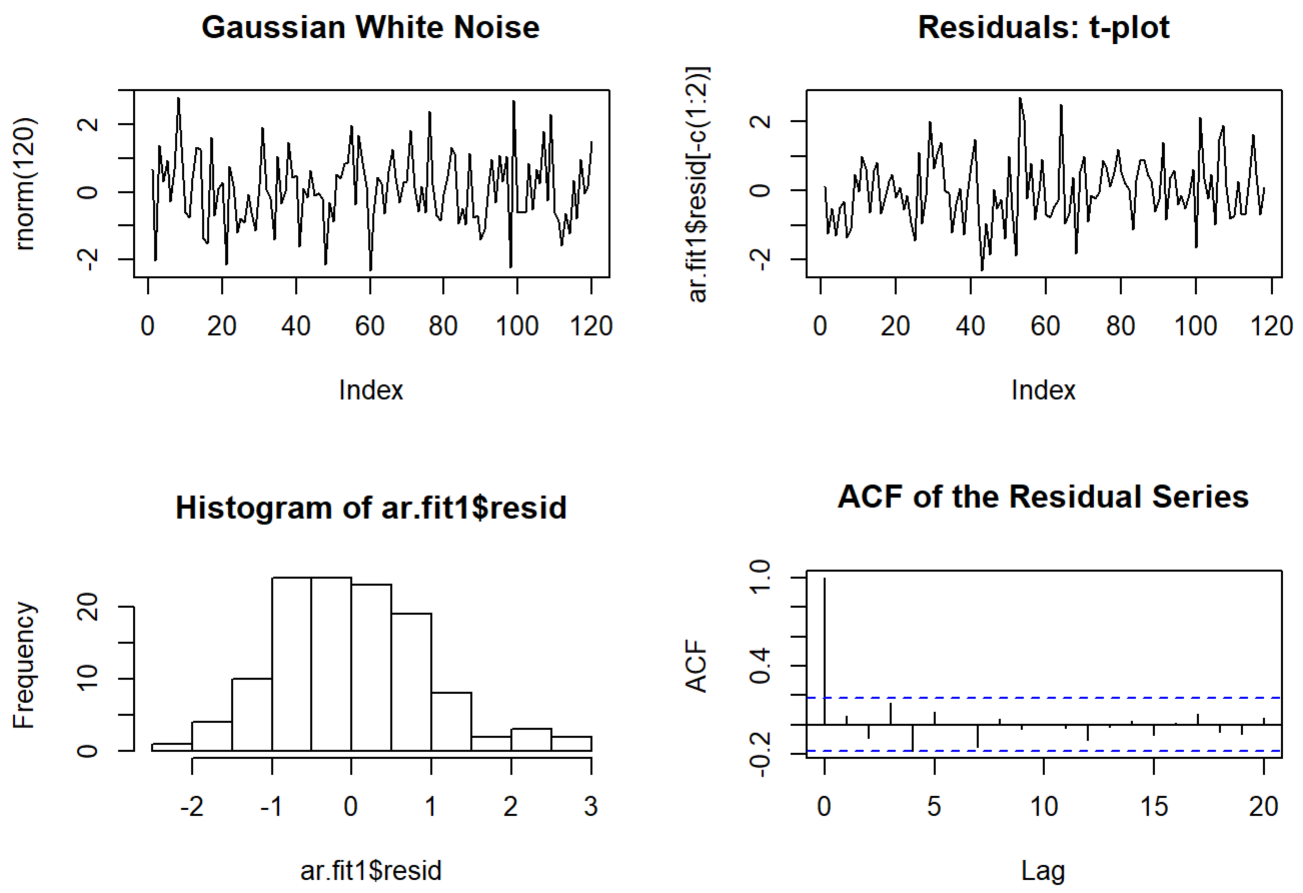
```
str(ar.fit1$resid)
```

```
##  Time-Series [1:120] from 1 to 10.9: -0.33 0.738 0.128 -1.242 -0.506 ...
```

```
plot(ar.fit1$resid)
```



```
par(mfrow=c(2,2))
plot(rnorm(120), type="l", main="Gaussian White Noise")
plot(ar.fit1$resid[-c(1:2)], type="l", main="Residuals: t-plot")
hist(ar.fit1$resid)
acf(ar.fit1$resid[-c(1:2)], main="ACF of the Residual Series")
```



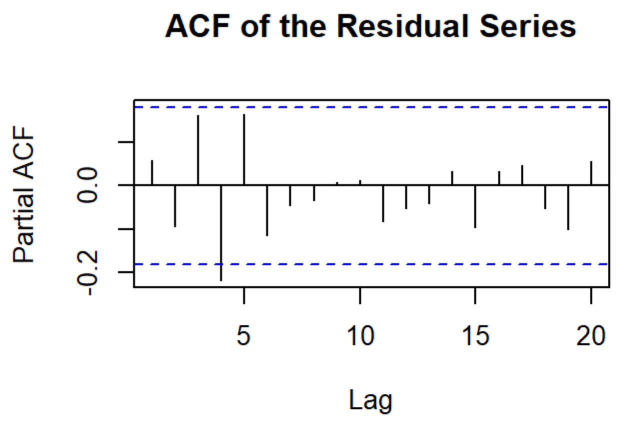
```
pacf(ar.fit1$resid[-c(1:2)], main="ACF of the Residual Series")

Box.test(residuals(ar.fit1), lag=24, type="Ljung")
```

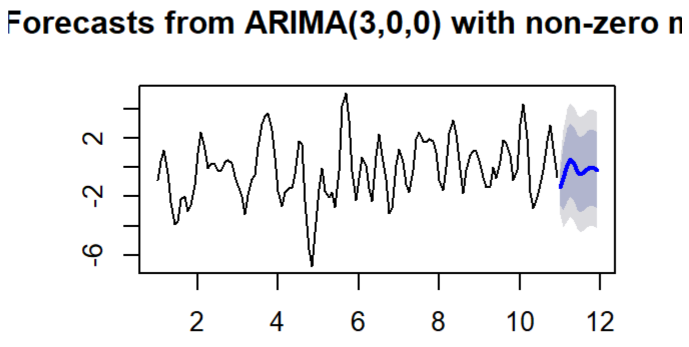
```
##
##  Box-Ljung test
##
## data:  residuals(ar.fit1)
## X-squared = 22.295, df = 24, p-value = 0.5617
```

```
tail(cbind(time(df.ts),df.ts))
```

```
##        time(df.ts)      df.ts
## Jul 10    10.50000 -1.1800922
## Aug 10    10.58333 -0.2028900
## Sep 10    10.66667  1.8177403
## Oct 10    10.75000  2.8730814
## Nov 10    10.83333  1.2931572
## Dec 10    10.91667 -0.6698399
```

```
ar.fit1.forecast = forecast(ar.fit1, h=12)
plot(ar.fit1.forecast)
```


ACF of the Residual Series
Partial ACF
0.0
-0.2
5
10
15
20
Lag


Forecasts from ARIMA(3,0,0) with non-zero n
2
-2
-6
2
4
6
8
10
12