Krysten Thompson w271: Homework 8

Professor Jeffrey Yau

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.

```r
library(car)
library(dplyr)
library(astsa)
library(forecast)
#library(fpp2)
library(ggplot2)
library(plotly)

# Insert the function to *tidy up* the code when they are printed out
library(knitr)
opts_chunk$set(tidy.opts=list(width.cutoff=60),tidy=TRUE)
```

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

```
##           V1
## 1 -0.8773679
## 2  0.4300889
## 3  1.1726073
## 4 -0.4223776
## 5 -2.3841229
## 6 -3.8806242
```

```r
str(df)
```

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

```r
tail(df)
```

```
##             V1
## 115 -1.1800922
```

```
## 116 -0.2028900
## 117  1.8177403
## 118  2.8730814
## 119  1.2931572
## 120 -0.6698399
```

```r
#changing dataframe to time series
df.ts <- ts(df)
class(df.ts)
```

```
## [1] "ts"
```

```r
#exploring
cycle(df.ts)
```

```
## Time Series:
## Start = 1
## End = 120
## Frequency = 1
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [71] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
## [106] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

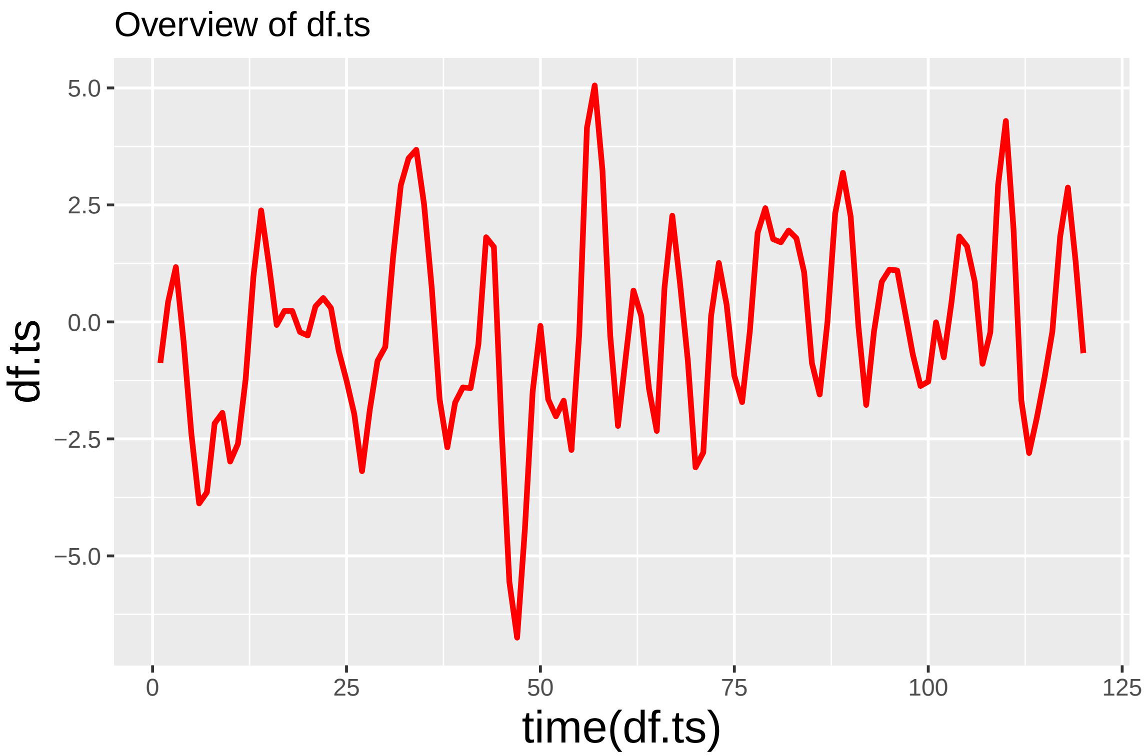
```r
summary(df.ts)
```

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

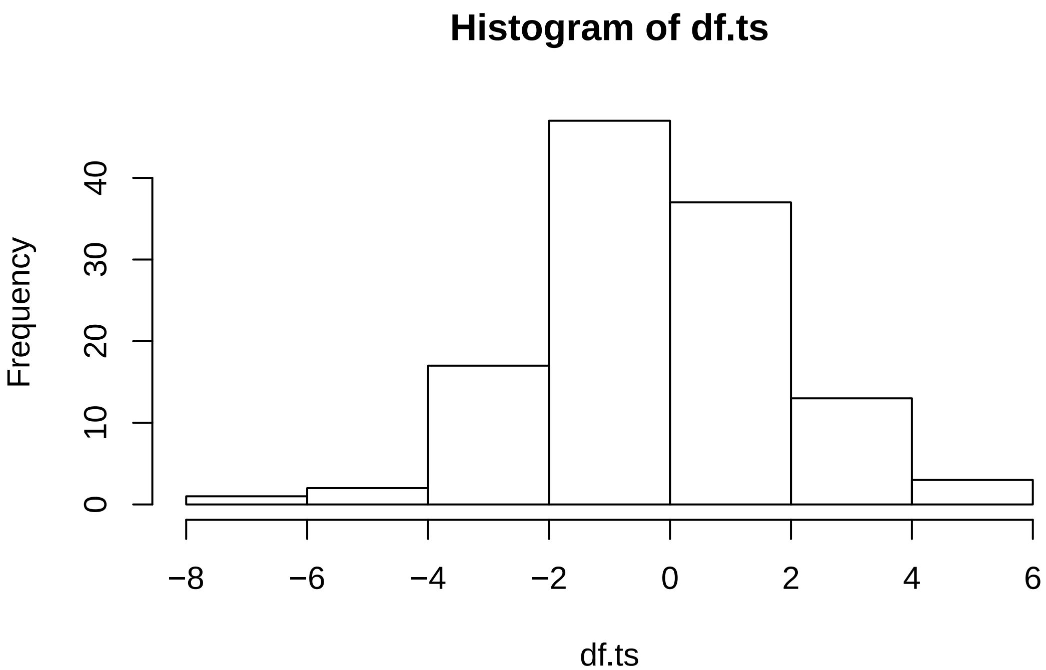
This section begins EDA

```r
#What does the data look like when plotted against time?

ggplot(df.ts, aes(x=time(df.ts), y=df.ts)) +
  geom_line(colour = "red", size = 1) +
  ggtitle("Overview of df.ts") +
  theme(axis.title = element_text(size = rel(1.5)))
```

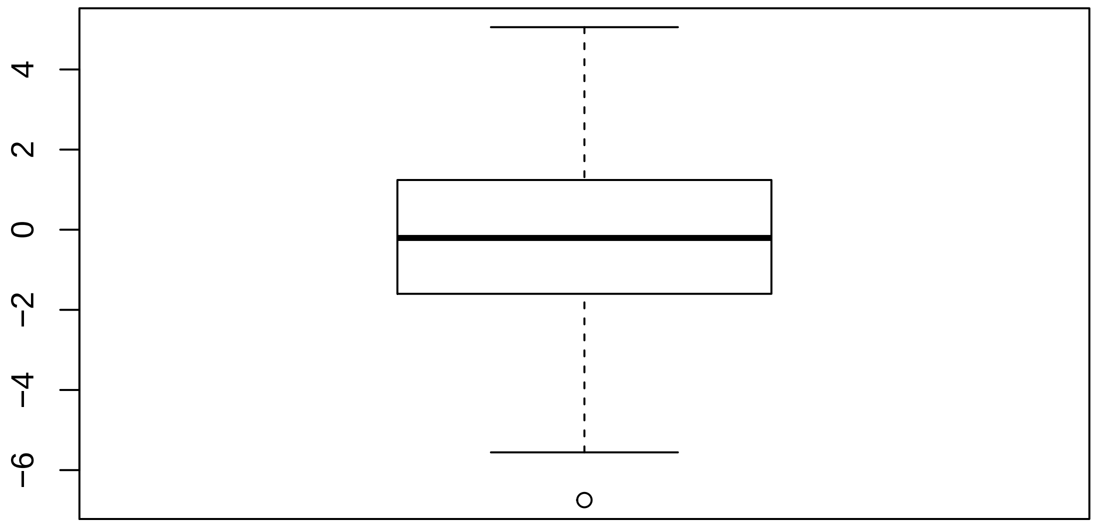
```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



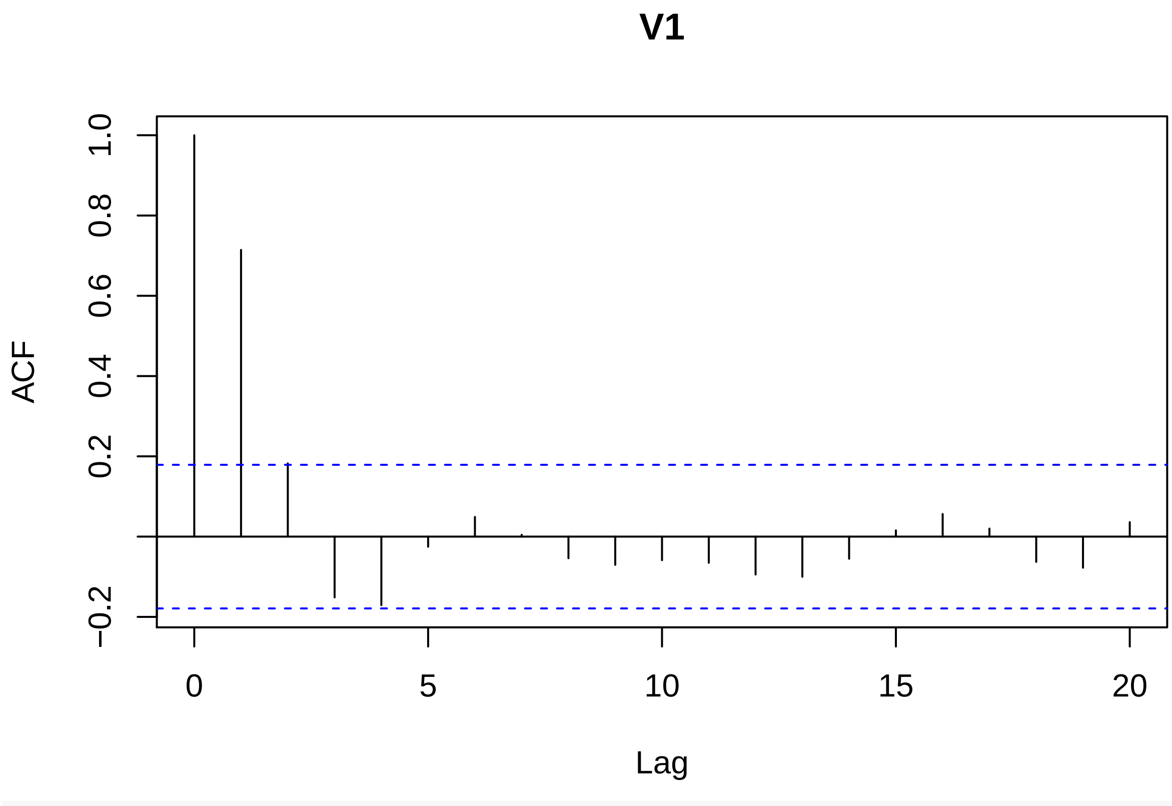

```
#Histograms aren't the most useful when exploring time series data but I still wanted to see wh

hist(df.ts)
```



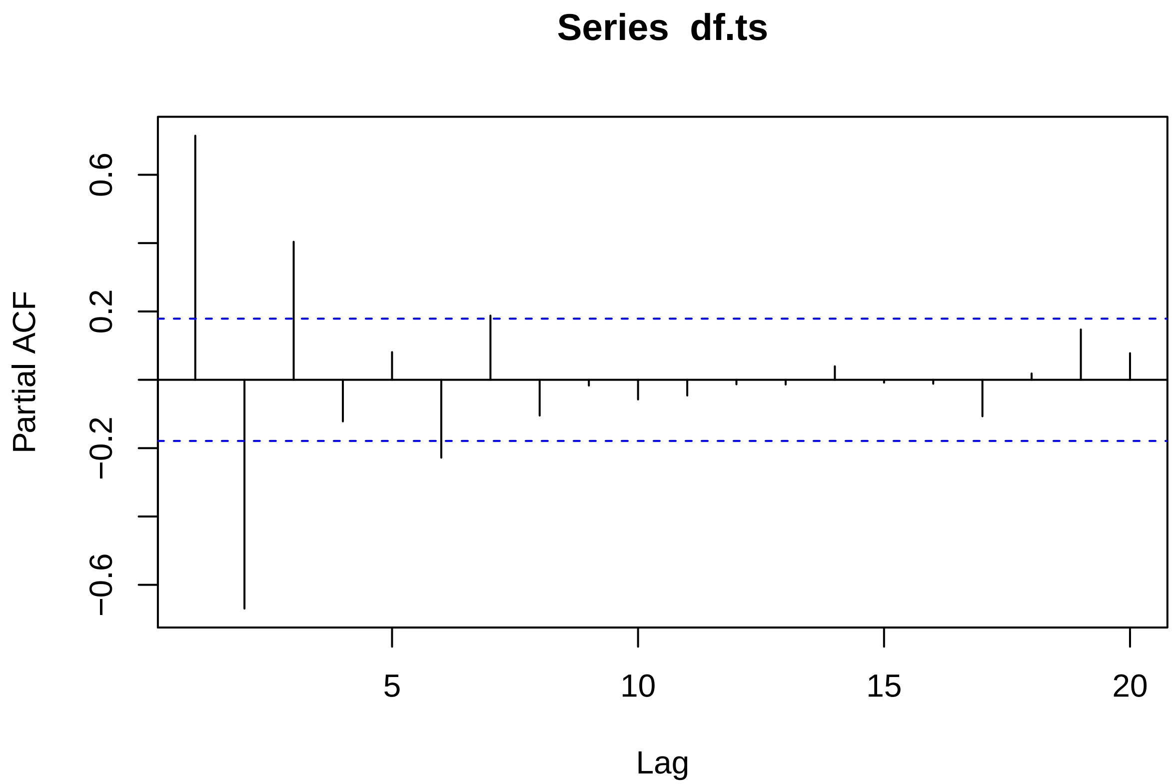

```
#I know there aren't column headers and data is not defined by specific time (e.g. week, month

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



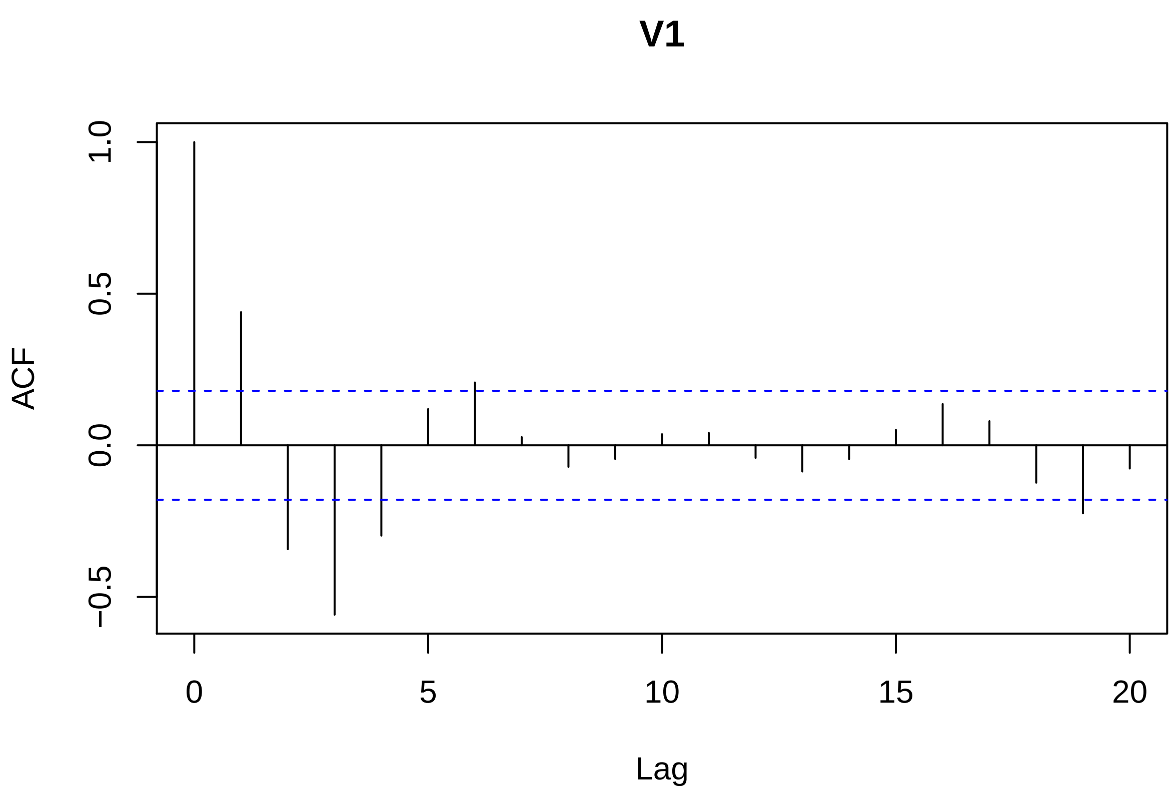

```r
#The acf plot below shows a sharp drop-off after lag 1

acf(df.ts)
```




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




```r
#Here we see first-order difference in ACF for df.ts and can see a slight
#pattern in the first 4 steps
acf(diff(df.ts))
```



This begins the modeling section

```r
#First a simple AR model with no order specifications
mod_ar <- ar(df.ts)
summary(mod_ar)
```

```
##              Length Class  Mode
## order          1    -none- numeric
## ar             7    -none- numeric
## var.pred       1    -none- numeric
## x.mean         1    -none- numeric
## aic           21    -none- numeric
## n.used         1    -none- numeric
## n.obs          1    -none- numeric
## order.max      1    -none- numeric
## partialacf    20    -none- numeric
## resid        120    ts     numeric
## method         1    -none- character
## series         1    -none- character
## frequency      1    -none- numeric
## call           2    -none- call
## asy.var.coef  49    -none- numeric
```

```r
mod_ar$ar
```

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

```r
mod_ar$order
```

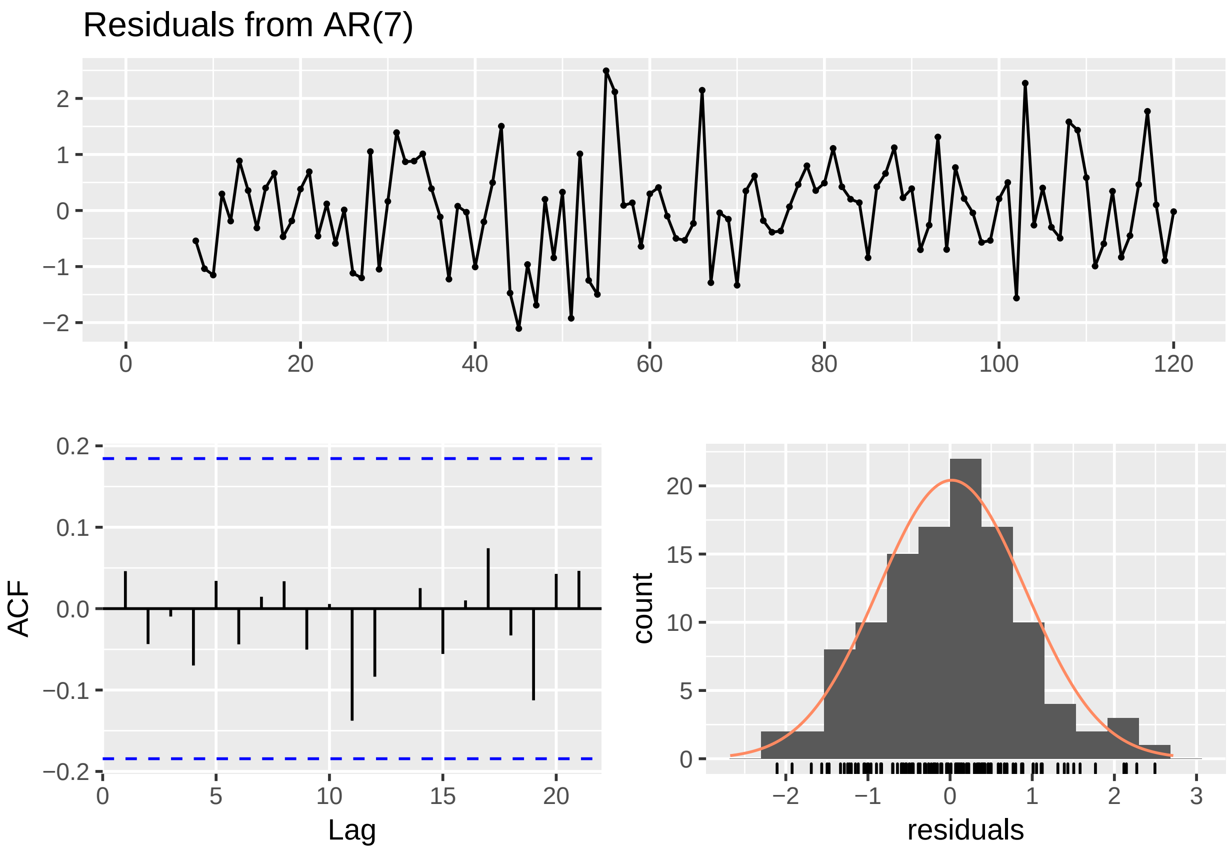
```
## [1] 7
```

```r
mod_ar$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          13          14          15          16          17
##   7.9703400   9.9472914  11.7589456  13.7507524  15.7347689  16.3566391
##          18          19          20
##  18.3150781  17.6834315  18.9515177
```

```r
checkresiduals(mod_ar)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```



Residuals plot indicates that this is not white noise.

```
#Simple AR model with order 5 (I arbitrarily chose 5)
AR.df.ts <- ar(df.ts, order.max=5)
summary(AR.df.ts)
```

```
##              Length Class  Mode
## order          1    -none- numeric
## ar             3    -none- numeric
## var.pred       1    -none- numeric
## x.mean         1    -none- numeric
## aic            6    -none- numeric
## n.used         1    -none- numeric
## n.obs          1    -none- numeric
## order.max      1    -none- numeric
## partialacf     5    -none- numeric
## resid        120    ts     numeric
## method         1    -none- character
## series         1    -none- character
## frequency      1    -none- numeric
## call           3    -none- call
## asy.var.coef   9    -none- numeric
```

```r
AR.df.ts$ar
```

```
## [1]  1.4631235 -1.1513317  0.4039593
```

```r
AR.df.ts$order
```

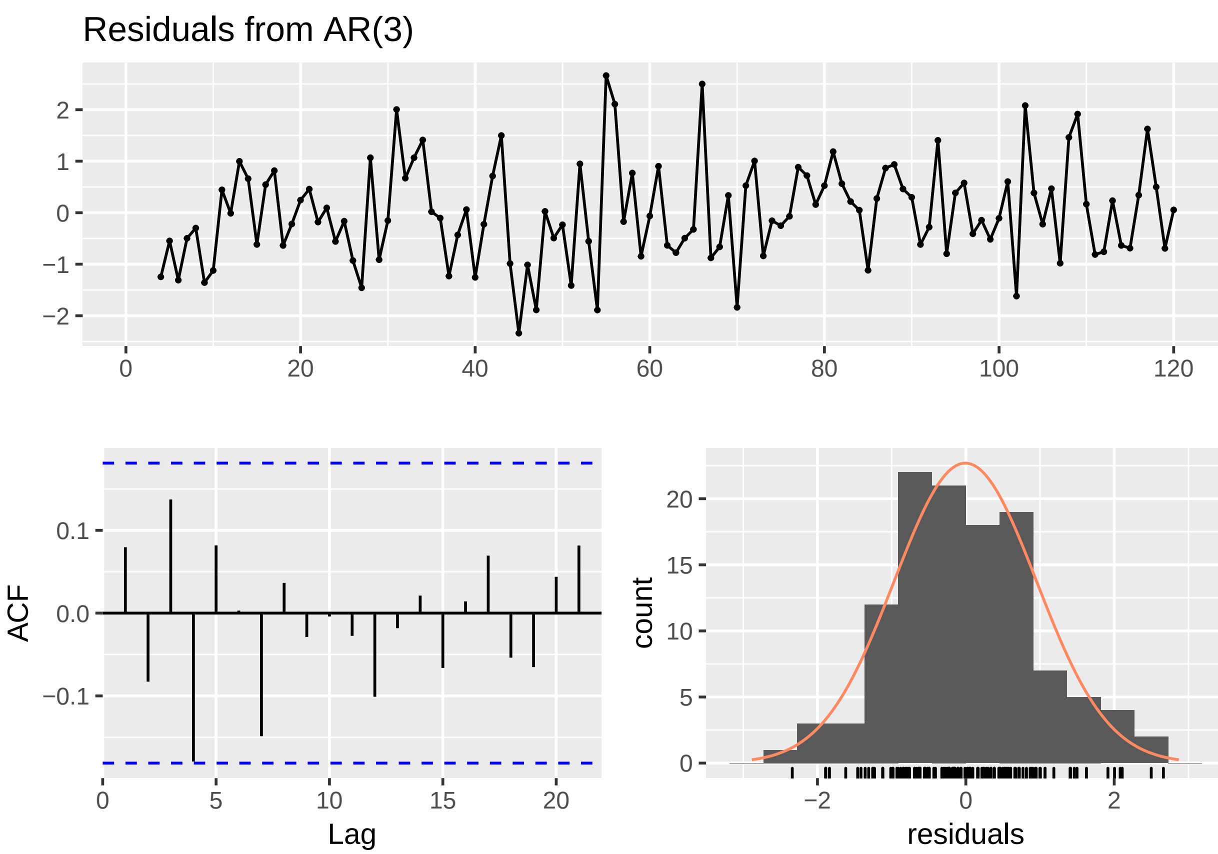
```
## [1] 3
```

```r
AR.df.ts$aic
```

```
##           0           1           2           3           4           5
## 172.4238758  88.7470266  19.3779983   0.0000000   0.2073951   1.4158719
```

```r
checkresiduals(AR.df.ts)
```

```
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```

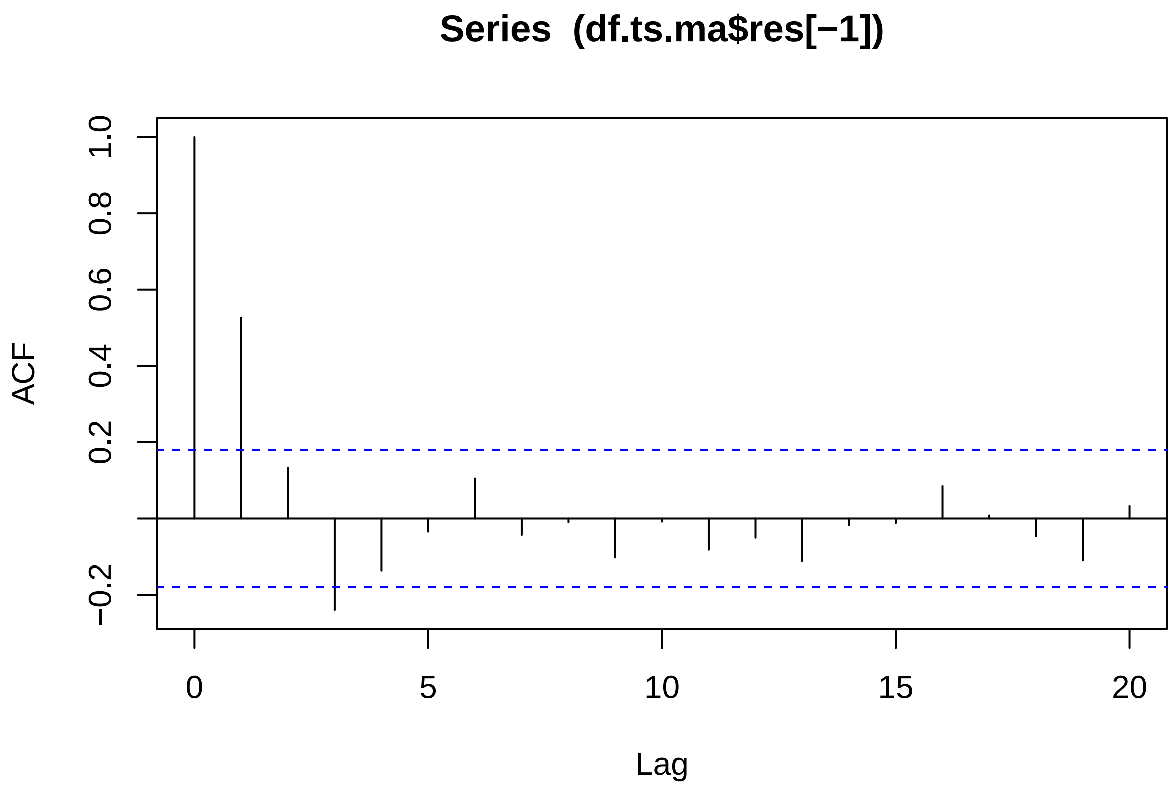



```r
df.ts.ma <- arima(df.ts, order=c(0,0,1))
df.ts.ma
```

```
##
## Call:
## arima(x = df.ts, order = c(0, 0, 1))
##
## Coefficients:
##          ma1  intercept
```

```
##        0.9353    -0.1595
## s.e.   0.0292     0.2126
##
## sigma^2 estimated as 1.46:  log likelihood = -194.02,  aic = 394.05
```

```r
acf((df.ts.ma$res[-1]))
```




```r
df.ts.ma2 <- arima(df.ts, order = c(0,0,1))
df.ts.ar2 <- arima(df.ts, order = c(1,0,0))
df.ts.arma <- arima(df.ts, order = c(1,0,1))
```

```r
#AICs for each model
AIC(df.ts.ma2)
```

```
## [1] 394.0498
```

```r
AIC(df.ts.ar2)
```

```
## [1] 432.6022
```

```r
AIC(df.ts.arma)
```

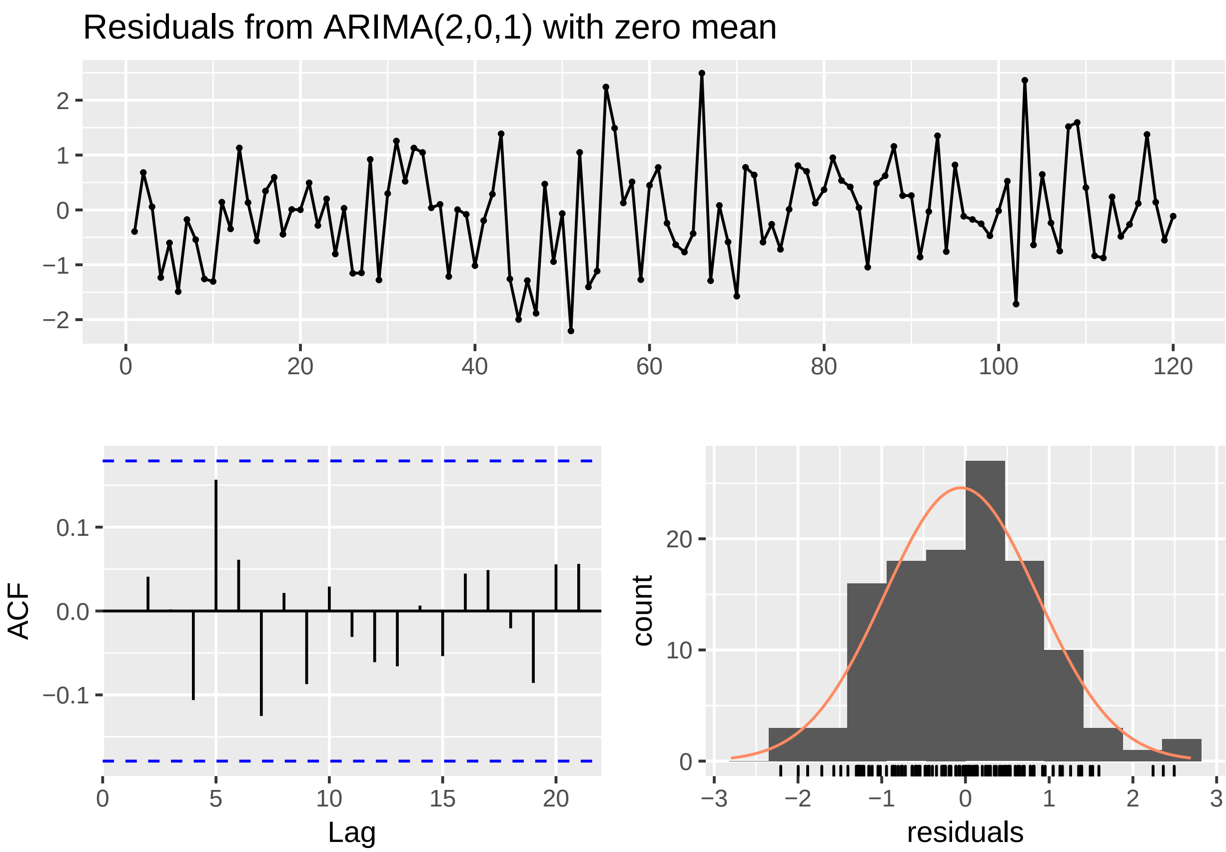
```
## [1] 351.4192
```

AICs for each model (ar, ma, arma) are very high. This is bad.

```r
#Since no seasonality was apparent in the data, seasonal = FALSE

fit <- auto.arima(df.ts, seasonal=FALSE)
fit
```

```
## 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
```

```
checkresiduals(fit)
```



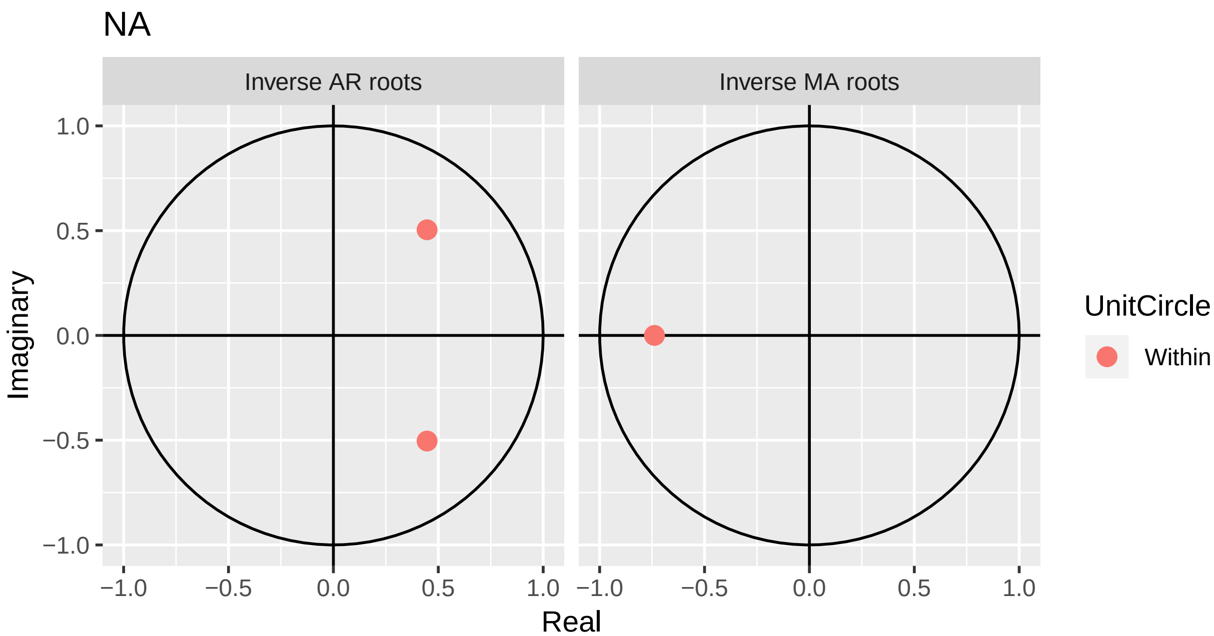

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,1) with zero mean
## Q* = 8.4324, df = 7, p-value = 0.296
##
## Model df: 3.   Total lags used: 10
```

```
Box.test(fit$residuals,  type='Ljung-Box')
```

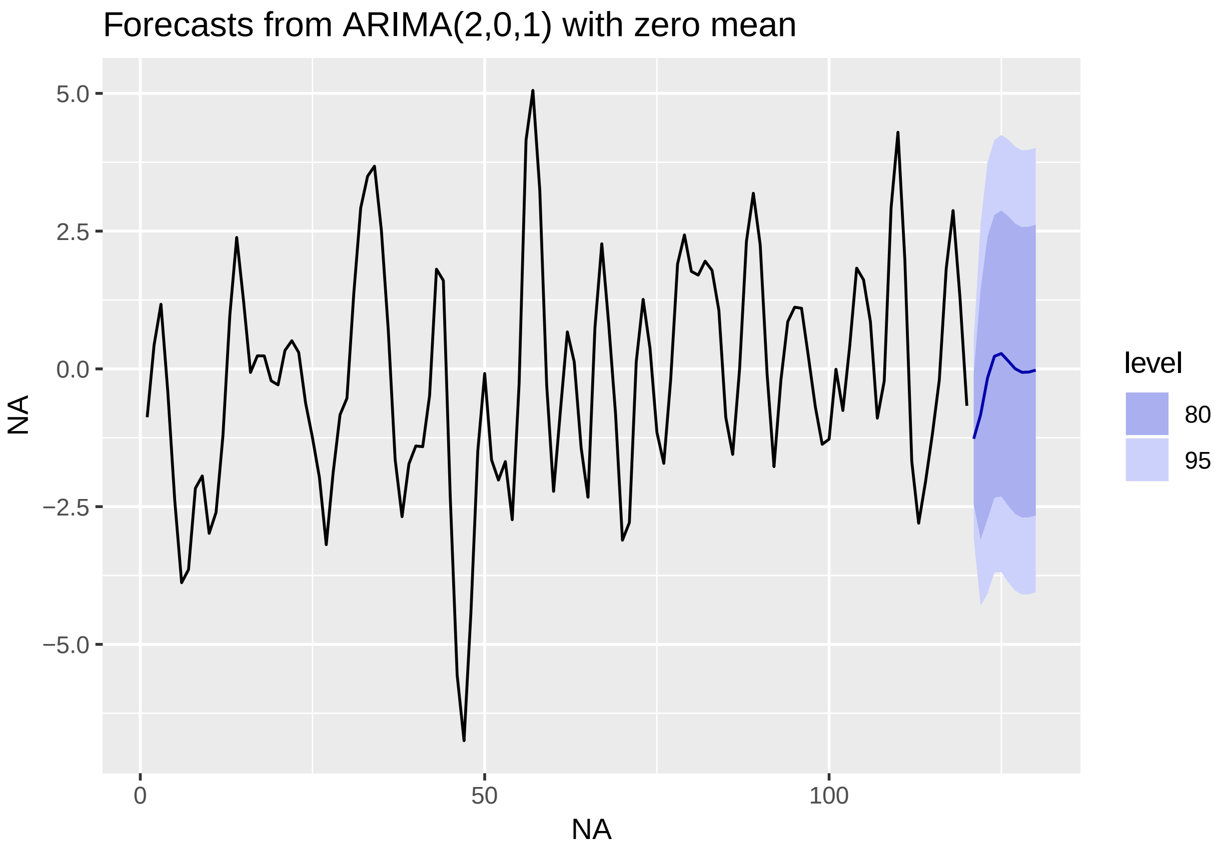
```
##
##  Box-Ljung test
```

```
##
## data:  fit$residuals
## X-squared = 8.6732e-05, df = 1, p-value = 0.9926
```

```r
#Below plots the inverse roots for p and q
#The red dots are all inside the circles which indicates the ARIMA model above is both station
autoplot(fit)
```



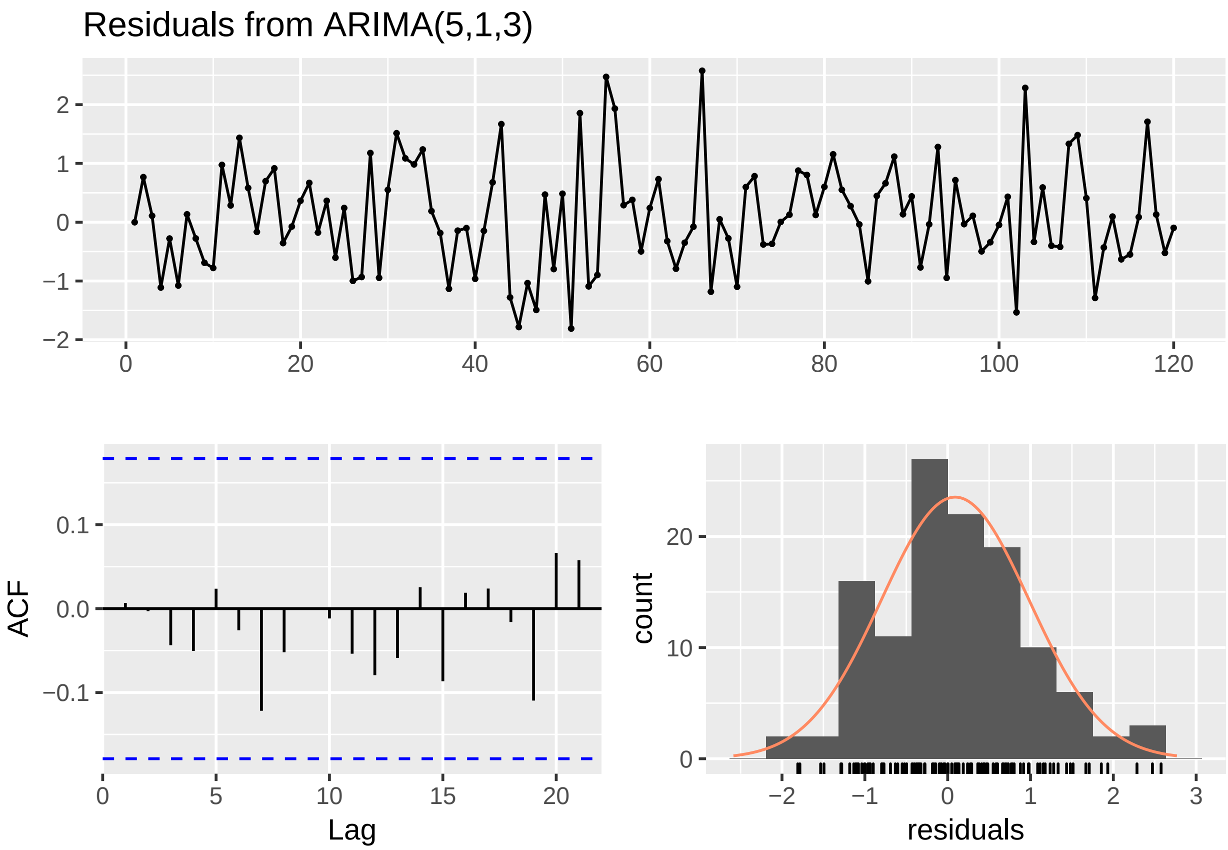

```r
autoplot(forecast(fit))
```




```
fit.custom <- Arima(df.ts, order=c(5, 1, 3))
summary(fit.custom)
```

```
## Series: df.ts
## ARIMA(5,1,3)
##
## Coefficients:
##          ar1     ar2      ar3      ar4     ar5     ma1      ma2      ma3
##       0.2033  0.1124  -0.1461  -0.2491  0.2761  0.4483  -0.7931  -0.6008
## s.e.  0.2570  0.2169   0.1543   0.1352  0.1140  0.2549   0.0912   0.2077
##
## sigma^2 estimated as 0.8625:  log likelihood=-157.92
## AIC=333.85   AICc=335.5   BIC=358.86
##
## Training set error measures:
##                      ME      RMSE       MAE       MPE     MAPE      MASE
## Training set 0.09061591 0.8931977 0.7004225 -138.0012 244.1511 0.5538937
##                     ACF1
## Training set 0.006812991
```

```
checkresiduals(fit.custom)
```

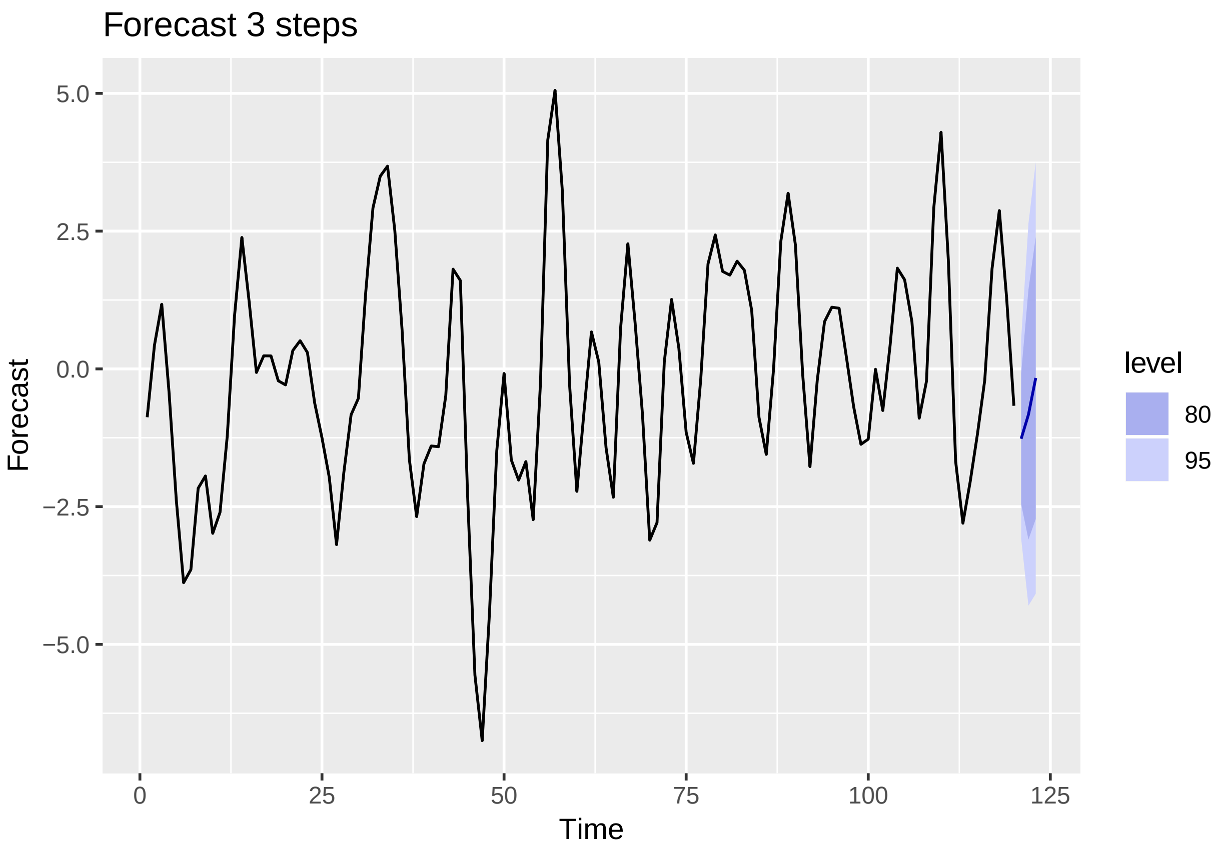



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(5,1,3)
## Q* = 3.4024, df = 3, p-value = 0.3336
##
## Model df: 8.   Total lags used: 11
```

```r
Box.test(fit.custom$residuals,  type='Ljung-Box')
```

```
##
##  Box-Ljung test
##
## data:  fit.custom$residuals
## X-squared = 0.0057104, df = 1, p-value = 0.9398
```

```r
fcast <- forecast(fit, h=3)

autoplot(fcast) + ggtitle("Forecast 3 steps") + xlab("Time") + ylab("Forecast")
```



Forecast 3 steps
5.0
2.5
0.0
−2.5
−5.0
Forecast
0
25
50
75
100
125
Time
level
80
95