Data624 - Project 1

Esteban Aramayo, Coffy Andrews-Guo, LeTicia Cancel, Joseph Connolly, Ian Costello

2022-06-21

Table of Contents

Required Libraries

library(readxl)  
library(tidyverse)  
library(ggplot2)  
library(patchwork)  
library(fpp2)  
library(caret)  
library(RANN)  
library(VIM)  
library(ggpubr)  
library(gridExtra)  
library(forecast)

# Project Summary

De-identified data was provided to conduct a series of six forecasts of different variables of a provided data set. There are two major requirements of this project:

1. This written report
2. The forecasts and error rates

df <- read\_excel("data.xls")  
head(df)

#> # A tibble: 6 x 7  
#> SeriesInd category Var01 Var02 Var03 Var05 Var07  
#> <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>  
#> 1 40669 S03 30.6 123432400 30.3 30.5 30.6  
#> 2 40669 S02 10.3 60855800 10.0 10.2 10.3  
#> 3 40669 S01 26.6 10369300 25.9 26.2 26.0  
#> 4 40669 S06 27.5 39335700 26.8 27.0 27.3  
#> 5 40669 S05 69.3 27809100 68.2 68.7 69.2  
#> 6 40669 S04 17.2 16587400 16.9 16.9 17.1

The data set does not appear to have any distinguishing labels that would indicate anything about the source or purpose of the data set. Under normal circumstances, context about data and use case is important for forecasting. Context may help identify the kinds of methods and models that would be best suited to produce an accurate result - following the “no free lunch” principle. Therefore, this dataset is completely for practice and exercising forecasting.

## Data Cleaning & Imputation

# Factoring category to get a count of the elements within dataset  
df$category <- as.factor(df$category)  
  
summary(df)

#> SeriesInd category Var01 Var02   
#> Min. :40669 S01:1762 Min. : 9.03 Min. : 1339900   
#> 1st Qu.:41303 S02:1762 1st Qu.: 23.10 1st Qu.: 12520675   
#> Median :41946 S03:1762 Median : 38.44 Median : 21086550   
#> Mean :41945 S04:1762 Mean : 46.98 Mean : 37035741   
#> 3rd Qu.:42587 S05:1762 3rd Qu.: 66.78 3rd Qu.: 42486700   
#> Max. :43221 S06:1762 Max. :195.18 Max. :480879500   
#> NA's :854 NA's :842   
#> Var03 Var05 Var07   
#> Min. : 8.82 Min. : 8.99 Min. : 8.92   
#> 1st Qu.: 22.59 1st Qu.: 22.91 1st Qu.: 22.88   
#> Median : 37.66 Median : 38.05 Median : 38.05   
#> Mean : 46.12 Mean : 46.55 Mean : 46.56   
#> 3rd Qu.: 65.88 3rd Qu.: 66.38 3rd Qu.: 66.31   
#> Max. :189.36 Max. :195.00 Max. :189.72   
#> NA's :866 NA's :866 NA's :866

* All groups are equal in length
* Columns 5-7 (Var03, Var04, Var07) all have same amount of missing values. Very close quartile and min/max values, comparable to column 3 (Var01). Column 4 (Var02) has values that are significantly larger

#### Data Structure

str(df)

#> tibble [10,572 x 7] (S3: tbl\_df/tbl/data.frame)  
#> $ SeriesInd: num [1:10572] 40669 40669 40669 40669 40669 ...  
#> $ category : Factor w/ 6 levels "S01","S02","S03",..: 3 2 1 6 5 4 3 2 1 6 ...  
#> $ Var01 : num [1:10572] 30.6 10.3 26.6 27.5 69.3 ...  
#> $ Var02 : num [1:10572] 1.23e+08 6.09e+07 1.04e+07 3.93e+07 2.78e+07 ...  
#> $ Var03 : num [1:10572] 30.3 10.1 25.9 26.8 68.2 ...  
#> $ Var05 : num [1:10572] 30.5 10.2 26.2 27 68.7 ...  
#> $ Var07 : num [1:10572] 30.6 10.3 26 27.3 69.2 ...

## Data Exploration

dim(df)

#> [1] 10572 7

summary(df)

#> SeriesInd category Var01 Var02   
#> Min. :40669 S01:1762 Min. : 9.03 Min. : 1339900   
#> 1st Qu.:41303 S02:1762 1st Qu.: 23.10 1st Qu.: 12520675   
#> Median :41946 S03:1762 Median : 38.44 Median : 21086550   
#> Mean :41945 S04:1762 Mean : 46.98 Mean : 37035741   
#> 3rd Qu.:42587 S05:1762 3rd Qu.: 66.78 3rd Qu.: 42486700   
#> Max. :43221 S06:1762 Max. :195.18 Max. :480879500   
#> NA's :854 NA's :842   
#> Var03 Var05 Var07   
#> Min. : 8.82 Min. : 8.99 Min. : 8.92   
#> 1st Qu.: 22.59 1st Qu.: 22.91 1st Qu.: 22.88   
#> Median : 37.66 Median : 38.05 Median : 38.05   
#> Mean : 46.12 Mean : 46.55 Mean : 46.56   
#> 3rd Qu.: 65.88 3rd Qu.: 66.38 3rd Qu.: 66.31   
#> Max. :189.36 Max. :195.00 Max. :189.72   
#> NA's :866 NA's :866 NA's :866

* The provided data set contains 7 columns and 10,572 rows. The first column, “SeriesInd”, is a datetime variable which can be converted to reflect a date. “category” is a classification of each row, of which there are 6 different ones, and the remaining 5 columns are variables to forecast and are the key ingredient for constructing timeseries.

##### Handling Missing Data

paste0(sum(is.na(df))," values missing from original set")

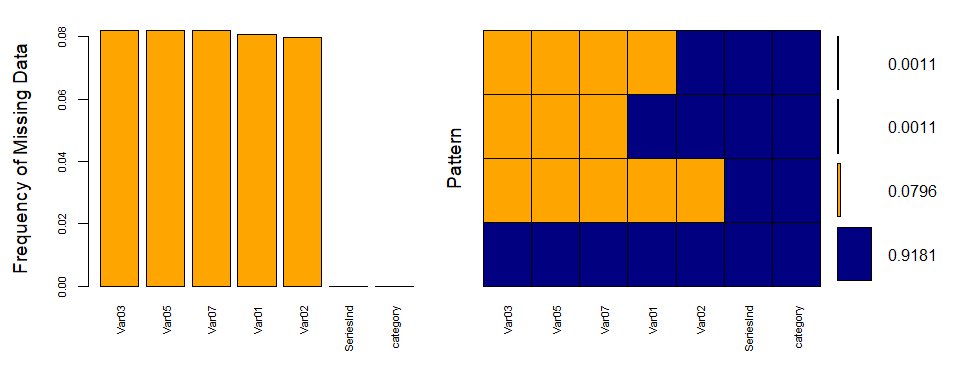
#> [1] "4294 values missing from original set"

Looking at the summary generated above, columns 3-7 each have a range of 842-866 missing values. The dilemma is to decide whether or not it is appropriate to perform an analysis via imputing missing values, or to simply delete them. According to the plot below, generated via “VIM::aggr()”, 91.81% of the data is fulfilled. Var01, Var02, Var03, Var05, and Var07 are missing about 8% of data. At first impression, this seems like an insignificant amount of data that can be omitted from the set. Further investigation is needed to confirm this impression, and deletion will be deemed appropriate if and only the data is found to be missing completely at random (MCAR).

##### Impute or Delete?

An excerpt from the following paper, , by Hyun Kang, argues when deletion is appropriate or not from the following quote: *“…if the assumption of MCAR (missing completely at random) is satisfied, a listwise deletion is known to produce unbiased estimates and conservative results. When the data do not fulfill the assumption of MCAR, listwise deletion may cause bias in the estimates of the parameters. If there is a large enough sample, where power is not an issue, and the assumption of MCAR is satisfied, the listwise deletion may be a reasonable strategy. However, when there is not a large sample, or the assumption of MCAR is not satisfied, the listwise deletion is not the optimal strategy”*

# Plots of missing values  
  
aggr\_plot <- VIM::aggr(df, col = c("navyblue", "orange"),   
 numbers = T, sortVars = T,  
 labels = names(df),  
 cex.axis = 0.7, gap = 3,  
 ylab = c("Frequency of Missing Data", "Pattern"))



#>   
#> Variables sorted by number of missings:   
#> Variable Count  
#> Var03 0.08191449  
#> Var05 0.08191449  
#> Var07 0.08191449  
#> Var01 0.08077942  
#> Var02 0.07964434  
#> SeriesInd 0.00000000  
#> category 0.00000000

By creating a shadow matrix to see a percentage (on a 0-1 scale) of missing values when all correlated among each other, this will help indicate whether or not the data is MCAR.

# Shadow Matrix: correlation of missing values from the dataset  
  
x <- as.data.frame(abs(is.na(df)))   
  
y <- x[which(sapply(x, sd) >0)] # Extracts which variables are missing/NA from the dataset  
  
cor(y) # Tendency of NA when correlated among variables

#> Var01 Var02 Var03 Var05 Var07  
#> Var01 1.0000000 0.9923369 0.9924341 0.9924341 0.9924341  
#> Var02 0.9923369 1.0000000 0.9848290 0.9848290 0.9848290  
#> Var03 0.9924341 0.9848290 1.0000000 1.0000000 1.0000000  
#> Var05 0.9924341 0.9848290 1.0000000 1.0000000 1.0000000  
#> Var07 0.9924341 0.9848290 1.0000000 1.0000000 1.0000000

Aside from considering values correlated with themselves, the following have no missing values when correlated with others:

* Var03 has no missing values when correlated with Var05 and Var07
* Var05 has no missing values when correlated with Var03 and Var07
* Var07 has no missing values when correlated with Var03 and Var05

Taking these observations into consideration, it seems there appears to be bias in the data in the context of missing values and therefore not missing completely at random. Imputing the values is an appropriate step to take.

##### Data Imputation

As stated, since the missing values are not MCAR, imputation will be utilized to represent missing values in our data set. There are numerous methods to impute data, especially in R. Within the caret package, the “preProcess()” function enables imputation, and allows for users to select a method of imputation. The method “medianImpute” was chosen because of the ease and efficiency.

# Imputation via "medianImpute" method within "preProcess()" function via the caret package  
  
preProcess\_NAdata\_model <- preProcess(as.data.frame(df), method ="medianImpute")  
  
df <- predict(preProcess\_NAdata\_model, newdata = df)  
  
paste0(sum(is.na(df))," values missing after imputation")

#> [1] "0 values missing after imputation"

* No data are missing after imputation
* No discrepencies in the data after imputation.

summary(df)

#> SeriesInd category Var01 Var02   
#> Min. :40669 S01:1762 Min. : 9.03 Min. : 1339900   
#> 1st Qu.:41303 S02:1762 1st Qu.: 25.40 1st Qu.: 13073675   
#> Median :41946 S03:1762 Median : 38.44 Median : 21086550   
#> Mean :41945 S04:1762 Mean : 46.29 Mean : 35765478   
#> 3rd Qu.:42587 S05:1762 3rd Qu.: 61.34 3rd Qu.: 39322325   
#> Max. :43221 S06:1762 Max. :195.18 Max. :480879500   
#> Var03 Var05 Var07   
#> Min. : 8.82 Min. : 8.99 Min. : 8.92   
#> 1st Qu.: 24.65 1st Qu.: 25.05 1st Qu.: 24.99   
#> Median : 37.66 Median : 38.05 Median : 38.05   
#> Mean : 45.42 Mean : 45.86 Mean : 45.86   
#> 3rd Qu.: 60.30 3rd Qu.: 60.83 3rd Qu.: 60.89   
#> Max. :189.36 Max. :195.00 Max. :189.72

##### Datetime conversion

* Datetime conversion is to be performed after imputation in order to ensure a successful imputation

# Converting Var02 to Datetime  
df$SeriesInd <- as.integer(df$SeriesInd)  
df$SeriesInd <- as.POSIXct(df$SeriesInd, origin = "1970-01-01")  
  
  
# Renaming SeriesInd to Date to clarify purpose  
  
df <- df %>% rename("Datetime" = SeriesInd)  
summary(df)

#> Datetime category Var01 Var02   
#> Min. :1970-01-01 06:17:49 S01:1762 Min. : 9.03 Min. : 1339900   
#> 1st Qu.:1970-01-01 06:28:23 S02:1762 1st Qu.: 25.40 1st Qu.: 13073675   
#> Median :1970-01-01 06:39:06 S03:1762 Median : 38.44 Median : 21086550   
#> Mean :1970-01-01 06:39:04 S04:1762 Mean : 46.29 Mean : 35765478   
#> 3rd Qu.:1970-01-01 06:49:47 S05:1762 3rd Qu.: 61.34 3rd Qu.: 39322325   
#> Max. :1970-01-01 07:00:21 S06:1762 Max. :195.18 Max. :480879500   
#> Var03 Var05 Var07   
#> Min. : 8.82 Min. : 8.99 Min. : 8.92   
#> 1st Qu.: 24.65 1st Qu.: 25.05 1st Qu.: 24.99   
#> Median : 37.66 Median : 38.05 Median : 38.05   
#> Mean : 45.42 Mean : 45.86 Mean : 45.86   
#> 3rd Qu.: 60.30 3rd Qu.: 60.83 3rd Qu.: 60.89   
#> Max. :189.36 Max. :195.00 Max. :189.72

##### Final susbets

# For forecasting later on  
  
s01 <- df %>% filter(category == "S01")  
s02 <- df %>% filter(category == "S02")  
s03 <- df %>% filter(category == "S03")  
s04 <- df %>% filter(category == "S04")  
s05 <- df %>% filter(category == "S05")  
s06 <- df %>% filter(category == "S06")

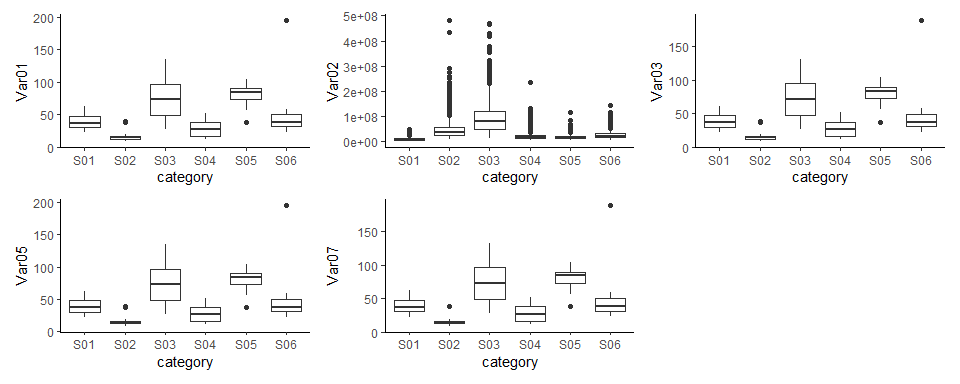
## Data Exploration & Visualizations

A time series that contain a list of numbers (the measurements), along with some information about what times those numbers were recorded (the index).

*Box plots*

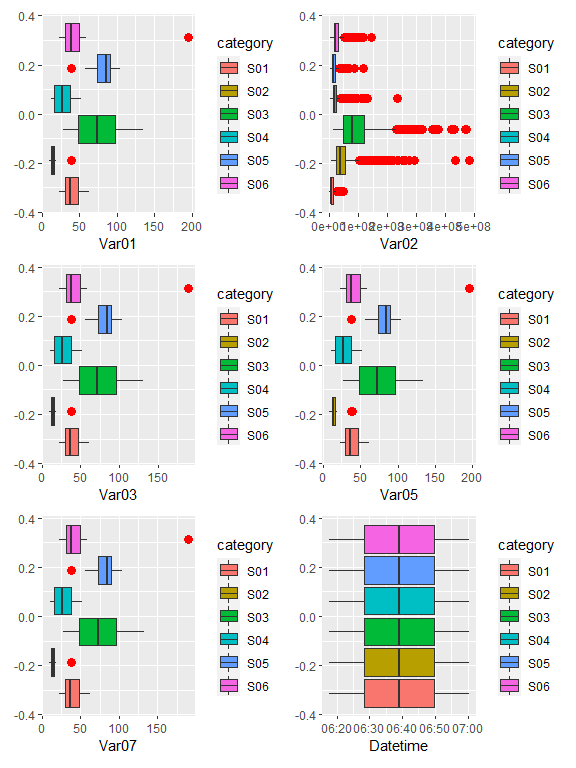
Boxplots:

# Boxplots of Predictors  
  
p1bp <- ggplot(df, aes(category, Var01)) +  
 geom\_boxplot() + theme\_classic()  
p2bp <- ggplot(df, aes(category, Var02)) +  
 geom\_boxplot() + theme\_classic()  
p3bp <- ggplot(df, aes(category, Var03)) +  
 geom\_boxplot() + theme\_classic()  
p4bp <- ggplot(df, aes(category, Var05)) +  
 geom\_boxplot() + theme\_classic()  
p5bp <- ggplot(df, aes(category, Var07)) +  
 geom\_boxplot() + theme\_classic()  
  
p1bp + p2bp + p3bp + p4bp + p5bp

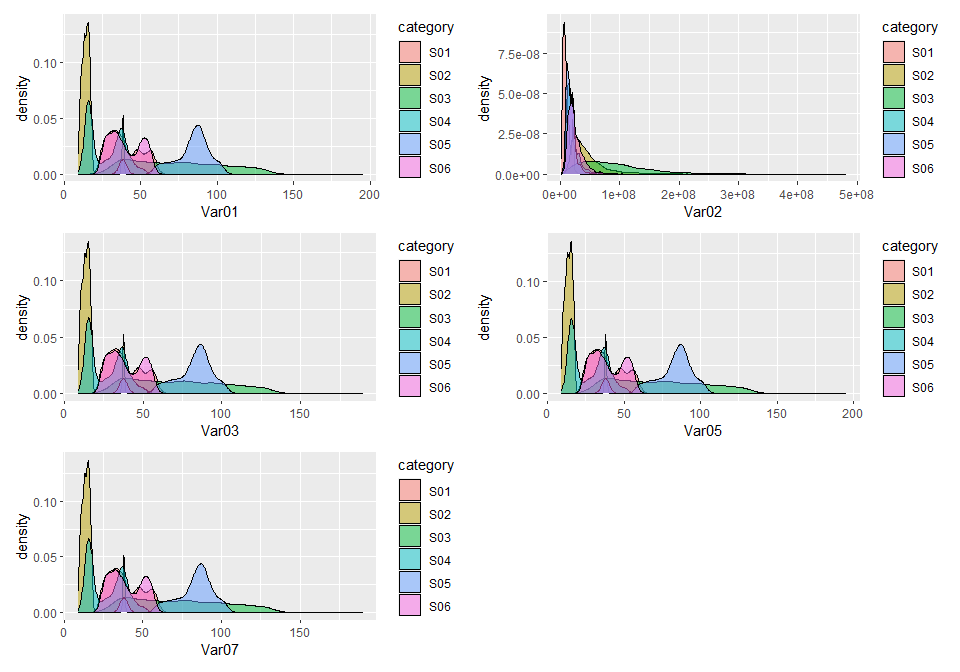


* Var02 has the most outliers. It’s also of a much larger a magnitude from the other predictors

# Other boxplot   
  
p1 <- ggplot(df, aes(Var01, fill = category)) +  
 geom\_boxplot(outlier.color = "red", outlier.size = 3)  
p2 <- ggplot(df, aes(Var02, fill = category)) +  
 geom\_boxplot(outlier.color = "red", outlier.size = 3)  
p3 <- ggplot(df, aes(Var03, fill = category)) +  
 geom\_boxplot(outlier.color = "red", outlier.size = 3)  
p4 <- ggplot(df, aes(Var05, fill = category)) +  
 geom\_boxplot(outlier.color = "red", outlier.size = 3)  
p5 <- ggplot(df, aes(Var07, fill = category)) +  
 geom\_boxplot(outlier.color = "red", outlier.size = 3)  
p6 <- ggplot(df, aes(Datetime, fill = category)) +   
 geom\_boxplot(outlier.color = "red", outlier.size = 10)  
  
(p1+p2)/(p3+p4)/(p5+p6)



# Density Plot  
  
p1 <- ggplot(df, aes(Var01, fill=category)) +  
 geom\_density(alpha = 0.5)  
p2 <- ggplot(df, aes(Var02, fill=category)) +  
 geom\_density(alpha = 0.5)  
p3 <- ggplot(df, aes(Var03, fill=category)) +  
 geom\_density(alpha = 0.5)  
p4 <- ggplot(df, aes(Var05, fill=category)) +  
 geom\_density(alpha = 0.5)  
p5 <- ggplot(df, aes(Var07, fill=category)) +  
 geom\_density(alpha = 0.5)  
  
p1+p2+p3+p4+p5+  
 plot\_layout(ncol = 2)



* According to the density plot above, all predictors are left skewed. Therefore transformations are needed for each column. By doing so, we can obtain a better forecast

## Data transformations

By using the skewness function within the moments package, we can see which predictor variable is skewed the most in quantifiable terms.

library(moments)  
paste0("Var01 skewness: ", skewness(df$Var01))

#> [1] "Var01 skewness: 0.833450080946069"

paste0("Var02 skewness: ",skewness(df$Var02))

#> [1] "Var02 skewness: 3.20209362366419"

paste0("Var03 skewness: ",skewness(df$Var03))

#> [1] "Var03 skewness: 0.834240845373782"

paste0("Var05 skewness: ",skewness(df$Var05))

#> [1] "Var05 skewness: 0.83686517614705"

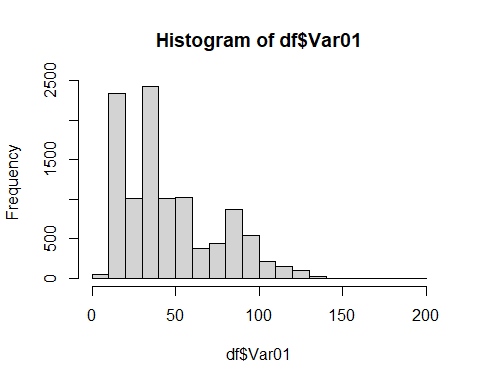
paste0("Var07 skewness: ",skewness(df$Var07))

#> [1] "Var07 skewness: 0.833896603742934"

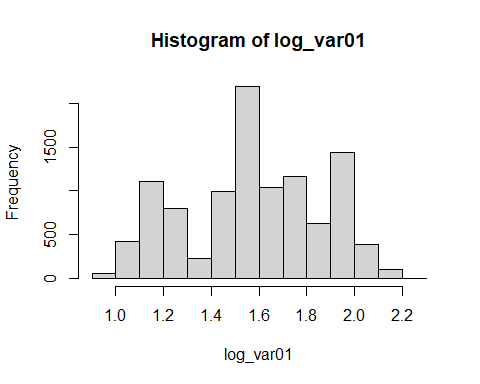
* Var02 is the most skewed

In order to improve the distribution for the predictors, we can investigate with applying 3 different transformations; log, square root, and cube root. Applying these transformations to one predictor will indicate which will provide the most normal distribution

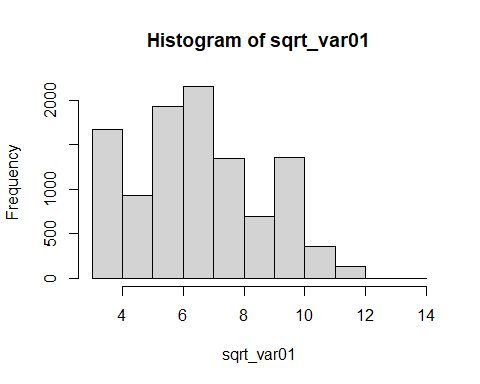
log\_var01 <- log10(df$Var01)  
sqrt\_var01 <- sqrt(df$Var01)  
cube\_var01 <- df$Var01^(1/3)  
  
hist(df$Var01)



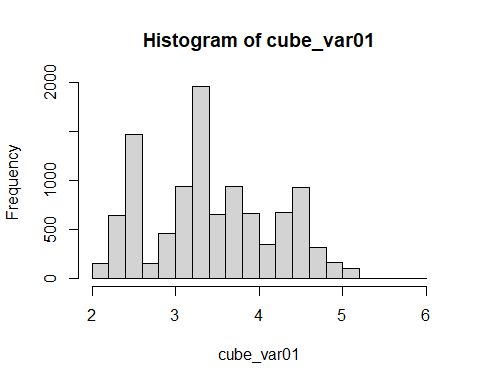
hist(log\_var01)



hist(sqrt\_var01)



hist(cube\_var01)

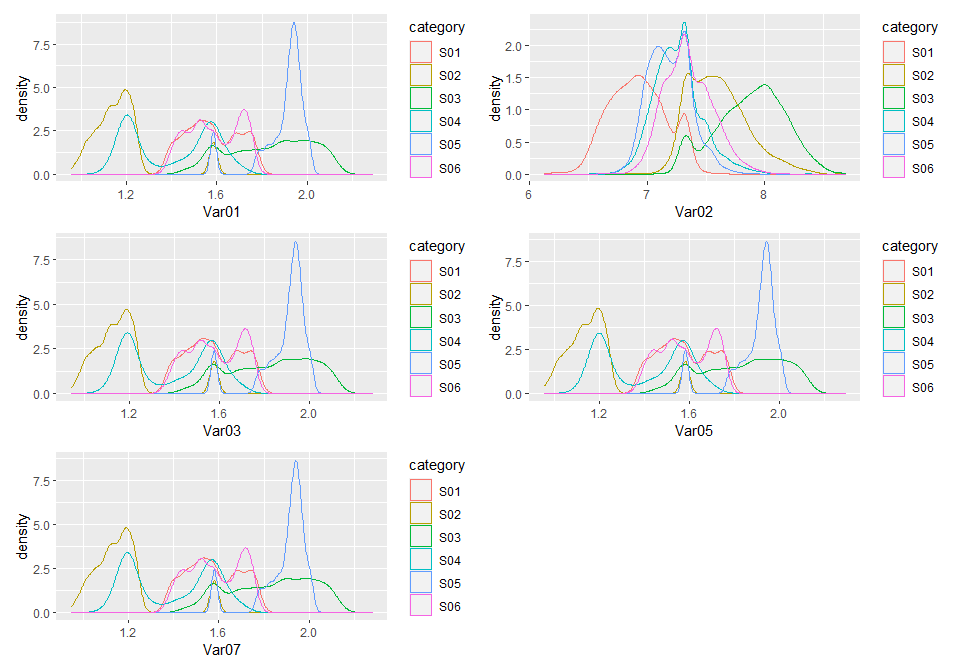


It’s revealed the log transformation appears to follow the most normal distribution. Therefore, the log transformation will be applied to all predictor variables.

df\_transformed <- df  
df\_transformed$Var01 <- log10(df$Var01)  
df\_transformed$Var02 <- log10(df$Var02)  
df\_transformed$Var03 <- log10(df$Var03)  
df\_transformed$Var05 <- log10(df$Var05)  
df\_transformed$Var07 <- log10(df$Var07)

New plots using transformed dataframe. Var02 is the most normalize so we will try different transformations with the other columns

p1 <- ggplot(df\_transformed, aes(Var01, color=category)) +  
 geom\_density()  
p2 <- ggplot(df\_transformed, aes(Var02, color=category)) +  
 geom\_density()  
p3 <- ggplot(df\_transformed, aes(Var03, color=category)) +  
 geom\_density()  
p4 <- ggplot(df\_transformed, aes(Var05, color=category)) +  
 geom\_density()  
p5 <- ggplot(df\_transformed, aes(Var07, color=category)) +  
 geom\_density()  
  
p1+p2+p3+p4+p5+  
 plot\_layout(ncol = 2)



s01 <- df\_transformed %>% dplyr::filter(category == "S01")  
s02 <- df\_transformed %>% dplyr::filter(category == "S02")  
s03 <- df\_transformed %>% dplyr::filter(category == "S03")  
s04 <- df\_transformed %>% dplyr::filter(category == "S04")  
s05 <- df\_transformed %>% dplyr::filter(category == "S05")  
s06 <- df\_transformed %>% dplyr::filter(category == "S06")

<<<<<<< Updated upstream Conversion using at.Date so each row is a date. ======= Testing conversion using at.Date so each row is a date with no time.

Stashed changes

# New time conversion (experimental?)  
  
df\_test <- read\_excel("data.xls")  
head(df\_test)

#> # A tibble: 6 x 7  
#> SeriesInd category Var01 Var02 Var03 Var05 Var07  
#> <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>  
#> 1 40669 S03 30.6 123432400 30.3 30.5 30.6  
#> 2 40669 S02 10.3 60855800 10.0 10.2 10.3  
#> 3 40669 S01 26.6 10369300 25.9 26.2 26.0  
#> 4 40669 S06 27.5 39335700 26.8 27.0 27.3  
#> 5 40669 S05 69.3 27809100 68.2 68.7 69.2  
#> 6 40669 S04 17.2 16587400 16.9 16.9 17.1

tail(df\_test)

#> # A tibble: 6 x 7  
#> SeriesInd category Var01 Var02 Var03 Var05 Var07  
#> <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl>  
#> 1 43221 S03 NA NA NA NA NA  
#> 2 43221 S02 NA NA NA NA NA  
#> 3 43221 S01 NA NA NA NA NA  
#> 4 43221 S06 NA NA NA NA NA  
#> 5 43221 S05 NA NA NA NA NA  
#> 6 43221 S04 NA NA NA NA NA

# Converting Var02 to Datetime  
df\_test$SeriesInd <- as.Date(df\_test$SeriesInd, origin = "1899-12-30")  
  
  
# Renaming SeriesInd to Date to clarify purpose  
df\_test <- df\_test %>% rename("Date" = SeriesInd)  
summary(df\_test)

#> Date category Var01 Var02   
#> Min. :2011-05-06 Length:10572 Min. : 9.03 Min. : 1339900   
#> 1st Qu.:2013-01-29 Class :character 1st Qu.: 23.10 1st Qu.: 12520675   
#> Median :2014-11-03 Mode :character Median : 38.44 Median : 21086550   
#> Mean :2014-11-01 Mean : 46.98 Mean : 37035741   
#> 3rd Qu.:2016-08-05 3rd Qu.: 66.78 3rd Qu.: 42486700   
#> Max. :2018-05-01 Max. :195.18 Max. :480879500   
#> NA's :854 NA's :842   
#> Var03 Var05 Var07   
#> Min. : 8.82 Min. : 8.99 Min. : 8.92   
#> 1st Qu.: 22.59 1st Qu.: 22.91 1st Qu.: 22.88   
#> Median : 37.66 Median : 38.05 Median : 38.05   
#> Mean : 46.12 Mean : 46.55 Mean : 46.56   
#> 3rd Qu.: 65.88 3rd Qu.: 66.38 3rd Qu.: 66.31   
#> Max. :189.36 Max. :195.00 Max. :189.72   
#> NA's :866 NA's :866 NA's :866

#new imputation  
preProcess\_NAdata\_model <- preProcess(as.data.frame(df\_test), method ="medianImpute")  
  
df\_test <- predict(preProcess\_NAdata\_model, newdata = df\_test)  
  
paste0(sum(is.na(df\_test))," values missing after imputation")

#> [1] "0 values missing after imputation"

#new subsets with data conversion  
s01\_2 <- df %>% filter(category == "S01")  
s02\_2 <- df %>% filter(category == "S02")  
s03\_2 <- df %>% filter(category == "S03")  
s04\_2 <- df %>% filter(category == "S04")  
s05\_2 <- df %>% filter(category == "S05")  
s06\_2 <- df %>% filter(category == "S06")

create time series for each subset using the dataframe with dates

s01\_ts <- ts(s01\_2[,c("Var01","Var02")], frequency = 12, start = c(2011, 5), end = c(2018, 5))  
s02\_ts <- ts(s02\_2[,c("Var02","Var03")], frequency = 12, start = c(2011, 5), end = c(2018, 5))  
s03\_ts <- ts(s03\_2[,c("Var05","Var07")], frequency = 12, start = c(2011, 5), end = c(2018, 5))  
s04\_ts <- ts(s04\_2[,c("Var01","Var02")], frequency = 12, start = c(2011, 5), end = c(2018, 5))  
s05\_ts <- ts(s05\_2[,c("Var02","Var03")], frequency = 12, start = c(2011, 5), end = c(2018, 5))  
s06\_ts <- ts(s06\_2[,c("Var05","Var07")], frequency = 12, start = c(2011, 5), end = c(2018, 5))

## Forecasting

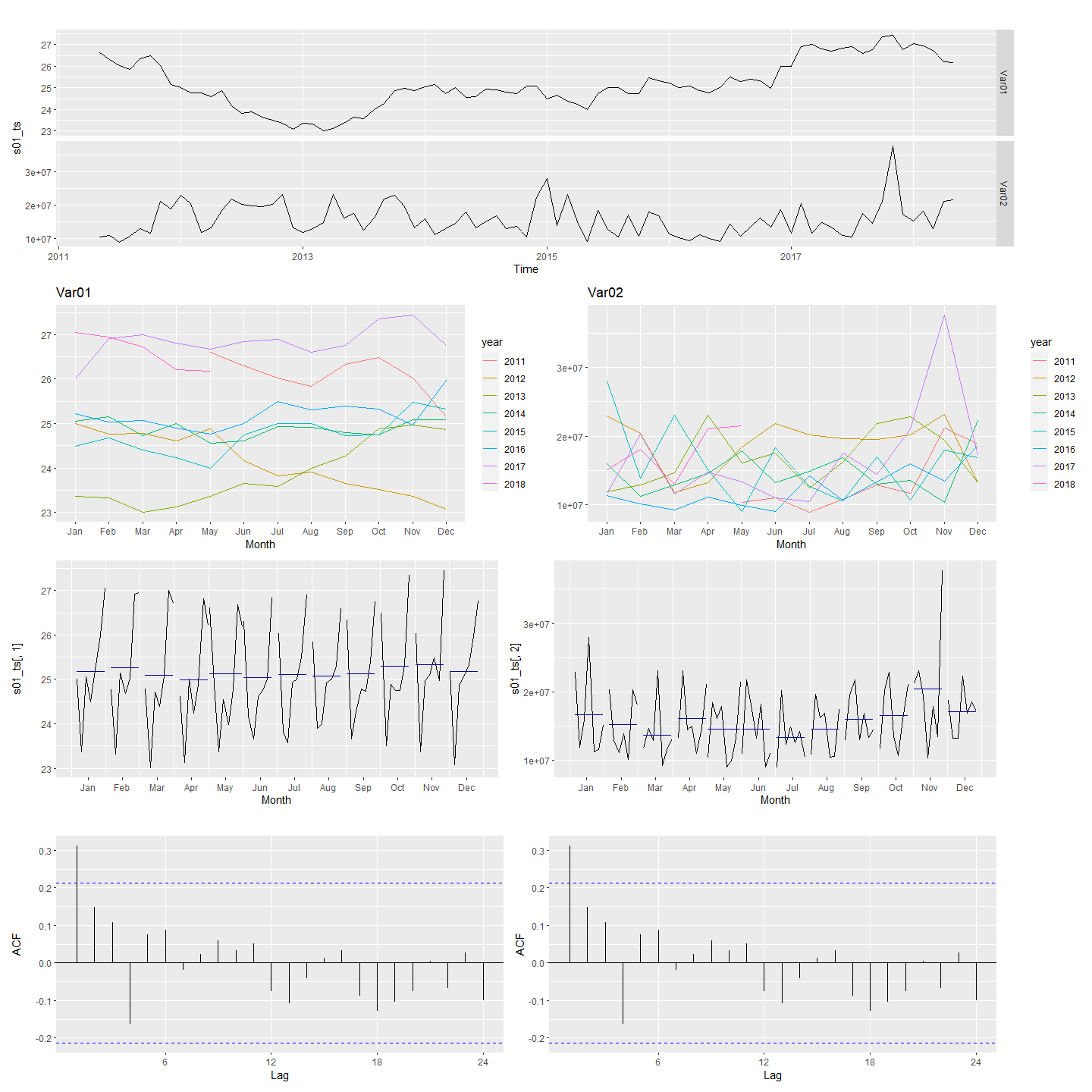
Now that the data has been transformed, we can begin forecasting

##### Auto and Seasonal Plots for S01: Var01, 02

For Var01, there are a few patterns to be pointed out. Annually, the value follows a slight valley pattern as it steadily decreases from the beginning to gradually rise towards the end. Monthly, there appears to be a pattern; toward the beginning of each month, the value decreases to a significant low only to lead up to a high-point approximately 2.5 points above the bottom, with a slight correction along the way. This timeseries is also the closest to a white noise series.

For Var02 as the years go on, the peaks get larger and more stark; particularly at 2015 and after 2017, and does not seem to follow a pattern as gradual as Var01. Monthly, there is no clear pattern in the value as it seems to invert shapes, with the exception of November. According to the lag plot, this timeseries distribution has a white noise series very similar to Var01.

(autoplot(s01\_ts, facets = 2)) /   
 (ggseasonplot(s01\_ts[,1], main = "Var01") + ggseasonplot(s01\_ts[,2], main = "Var02")) /   
 (ggsubseriesplot(s01\_ts[,1]) + ggsubseriesplot(s01\_ts[,2])) /   
 (ggAcf(s01\_ts[,2], main = "") + ggAcf(s01\_ts[,2], main = ""))

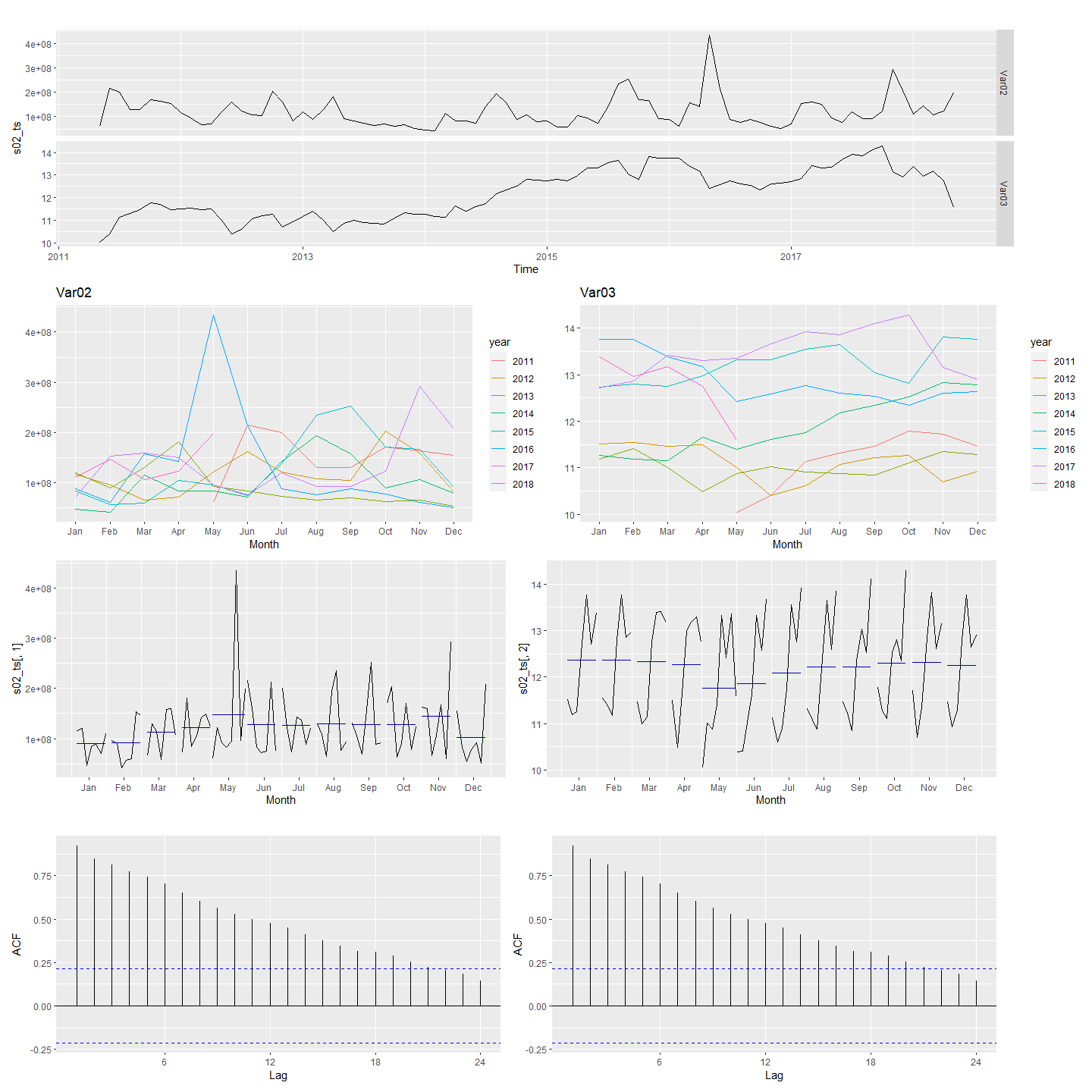


##### Auto and Seasonal Plots for S02: Var02, 03

Var02: The plots generated for this variable indicate the value has a rather dramatic cycle. During the years it seems to have no real growth, as it peaks only to fall close to the lowest value. 2016 is quite the exceptional year for this, as it peaked much more than other years indicated. Monthly, this peaks mostly near the end of each month with a few exceptions to where the peak shifts toward the middle but still favors the end. However January, June, July, and October are all exceptions as their peaks are in the beginning of the month. There is no clear seasonality with these trends. This also does not appear to be a white noise series.

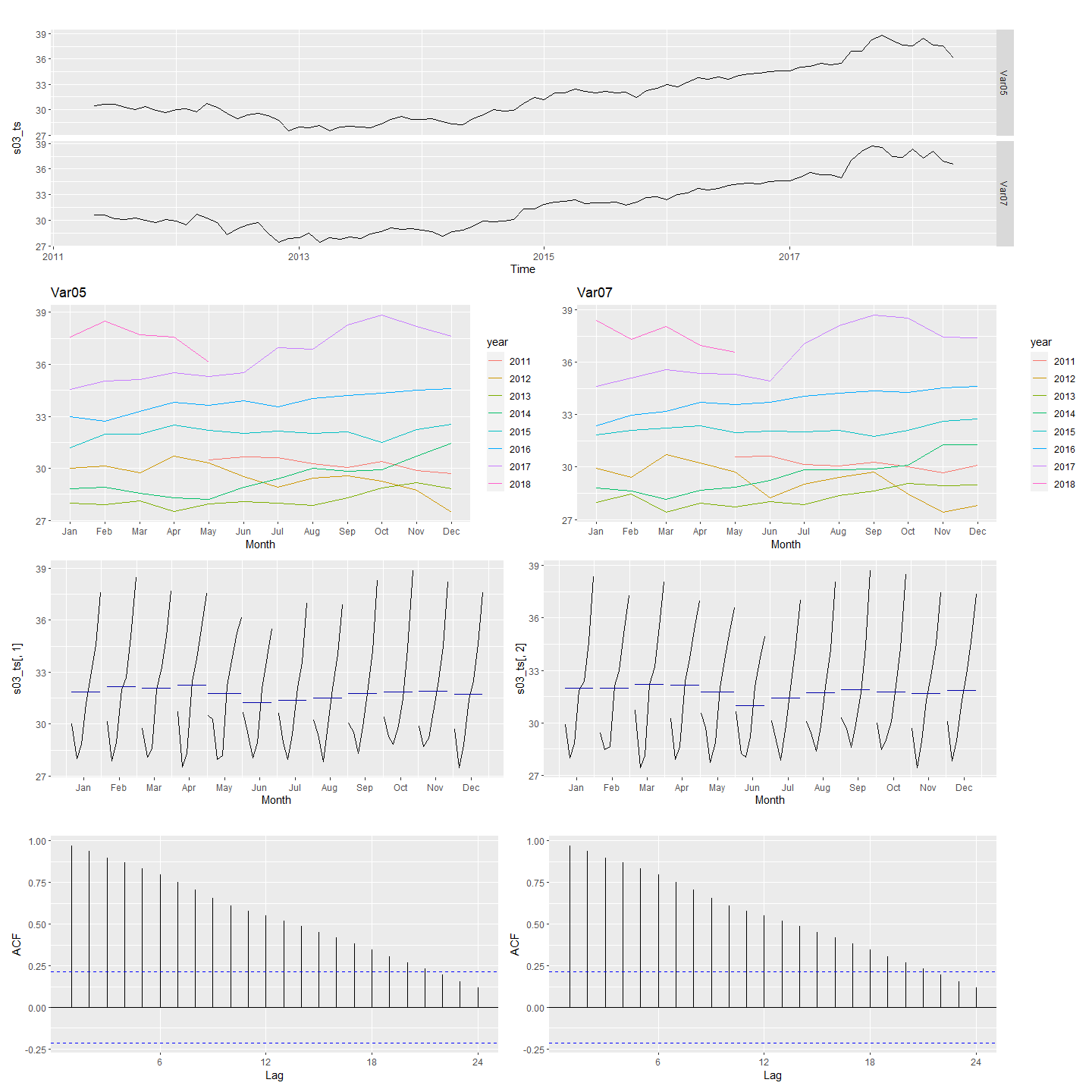
Var03: For Var03

(autoplot(s02\_ts, facets = 2)) /   
 (ggseasonplot(s02\_ts[,1], main = "Var02") + ggseasonplot(s02\_ts[,2], main = "Var03")) /   
 (ggsubseriesplot(s02\_ts[,1]) + ggsubseriesplot(s02\_ts[,2])) /   
 (ggAcf(s02\_ts[,2], main = "") + ggAcf(s02\_ts[,2], main = ""))



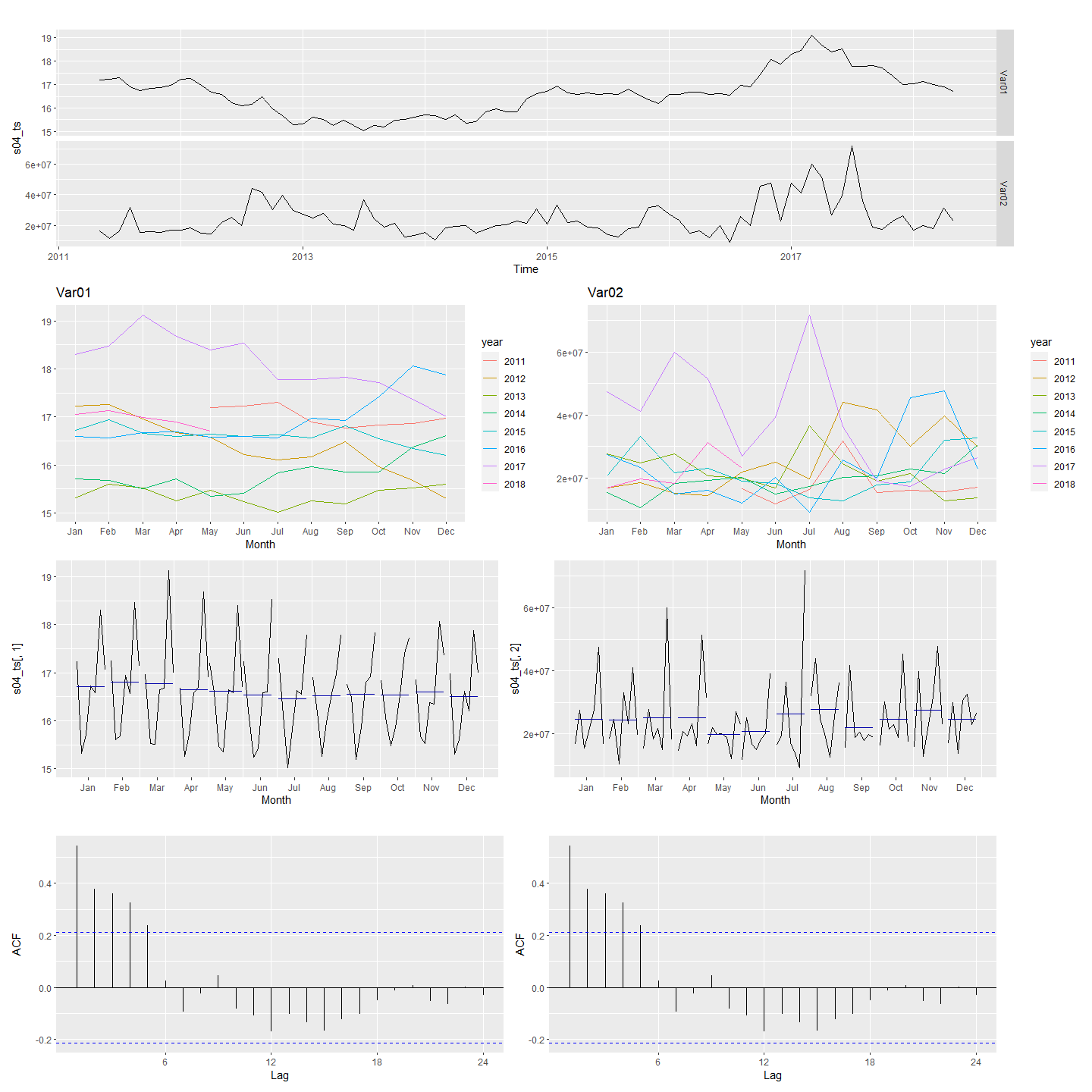
Autoplots and seasonal plots for S03

(autoplot(s03\_ts, facets = 2)) /   
 (ggseasonplot(s03\_ts[,1], main = "Var05") + ggseasonplot(s03\_ts[,2], main = "Var07")) /   
 (ggsubseriesplot(s03\_ts[,1]) + ggsubseriesplot(s03\_ts[,2])) /   
 (ggAcf(s03\_ts[,2], main = "") + ggAcf(s03\_ts[,2], main = ""))



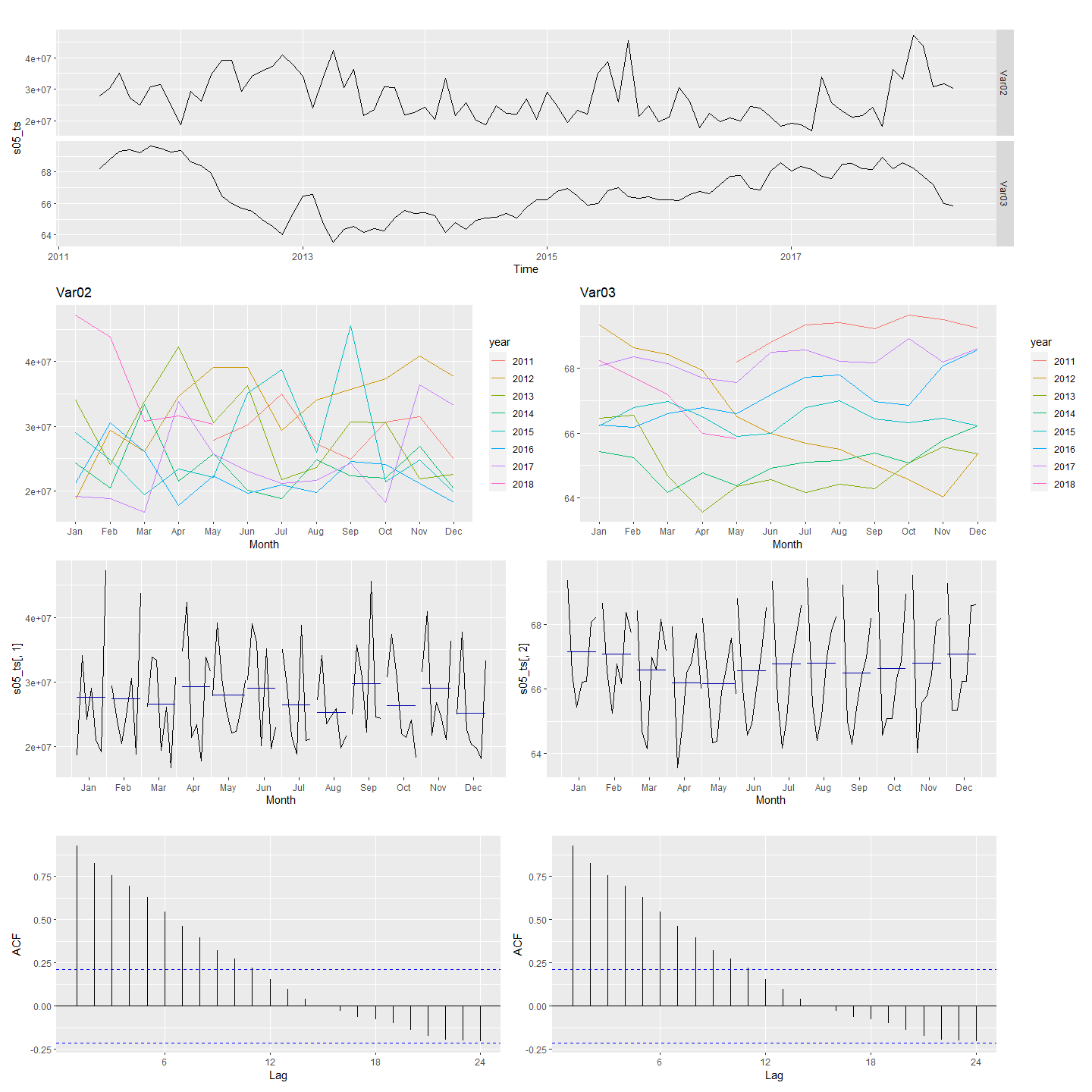
Autoplots and seasonal plots for S04

(autoplot(s04\_ts, facets = 2)) /   
 (ggseasonplot(s04\_ts[,1], main = "Var01") + ggseasonplot(s04\_ts[,2], main = "Var02")) /   
 (ggsubseriesplot(s04\_ts[,1]) + ggsubseriesplot(s04\_ts[,2])) /   
 (ggAcf(s04\_ts[,2], main = "") + ggAcf(s04\_ts[,2], main = ""))



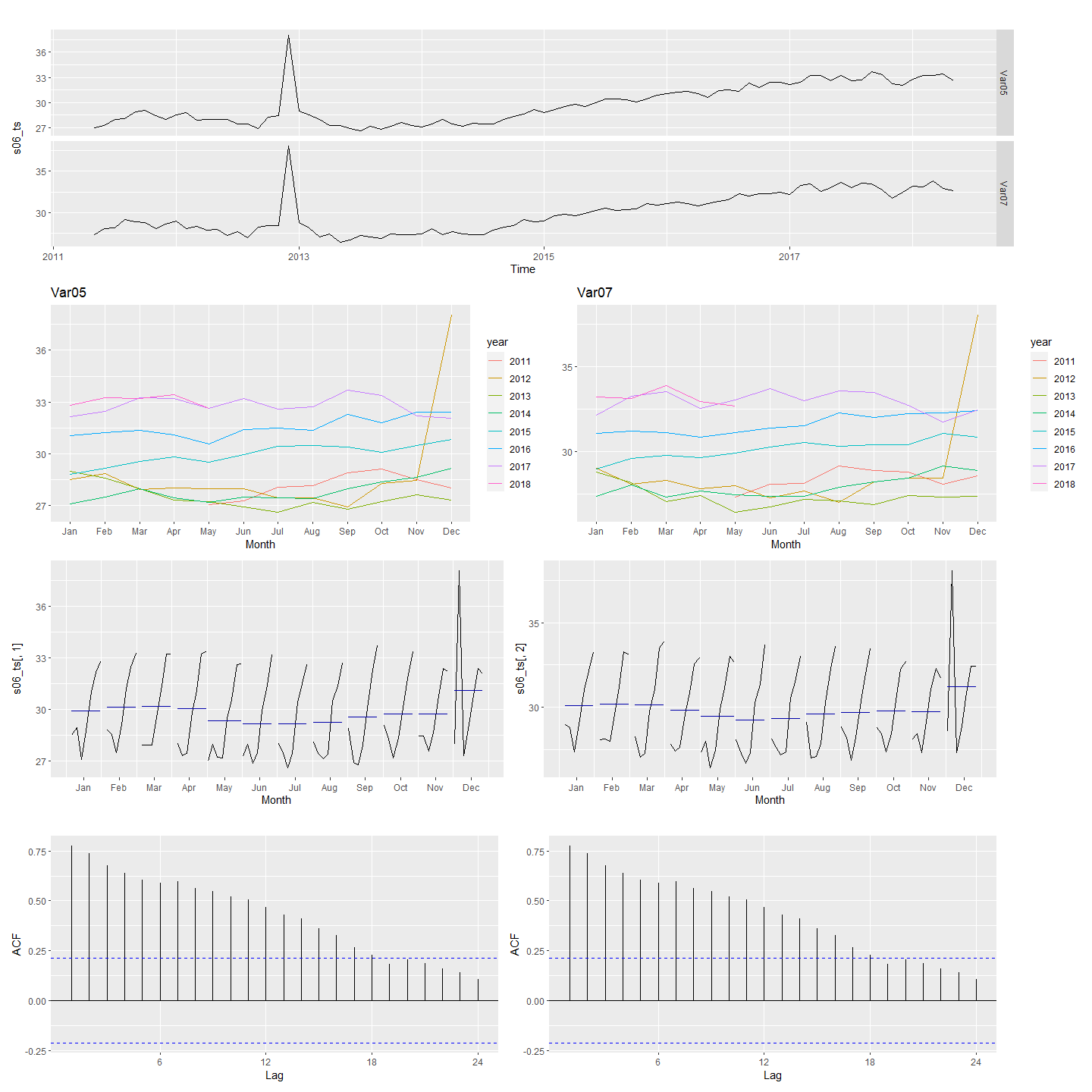
Autoplots and seasonal plots for S05

(autoplot(s05\_ts, facets = 2)) /   
 (ggseasonplot(s05\_ts[,1], main = "Var02") + ggseasonplot(s05\_ts[,2], main = "Var03")) /   
 (ggsubseriesplot(s05\_ts[,1]) + ggsubseriesplot(s05\_ts[,2])) /   
 (ggAcf(s05\_ts[,2], main = "") + ggAcf(s05\_ts[,2], main = ""))



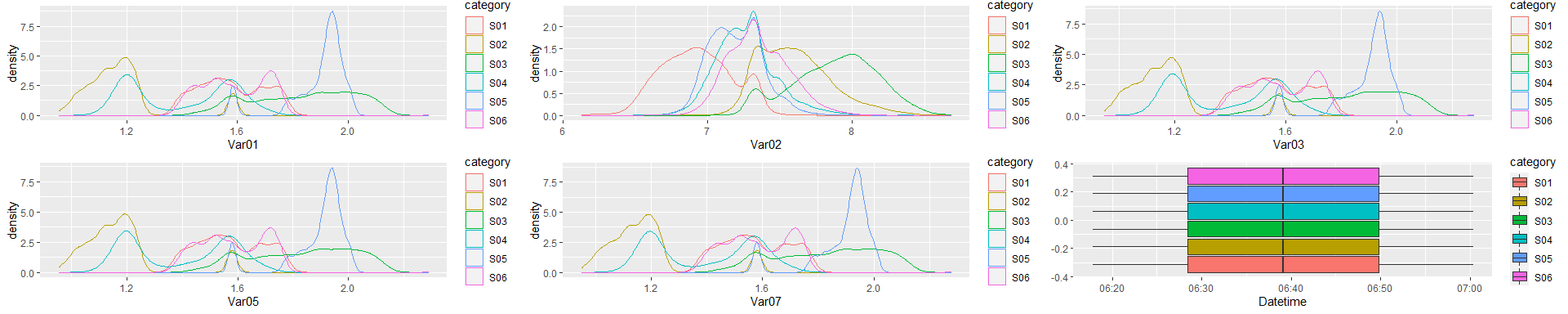
Autoplots and seasonal plots for S06

(autoplot(s06\_ts, facets = 2)) /   
 (ggseasonplot(s06\_ts[,1], main = "Var05") + ggseasonplot(s06\_ts[,2], main = "Var07")) /   
 (ggsubseriesplot(s06\_ts[,1]) + ggsubseriesplot(s06\_ts[,2])) /   
 (ggAcf(s06\_ts[,2], main = "") + ggAcf(s06\_ts[,2], main = ""))



p1 <- (gglagplot(s01\_ts[,1]) + theme(legend.position = "none") + gglagplot(s01\_ts[,2]) + theme(legend.position = "none") )  
p2 <- gglagplot(s02\_ts[,1]) + theme(legend.position = "none") + gglagplot(s02\_ts[,2]) + theme(legend.position = "none")  
p3 <- gglagplot(s03\_ts[,1]) + theme(legend.position = "none") + gglagplot(s03\_ts[,2])  
p4 <- gglagplot(s04\_ts[,1]) + theme(legend.position = "none") + gglagplot(s04\_ts[,2]) + theme(legend.position = "none")  
p5 <- gglagplot(s05\_ts[,1]) + theme(legend.position = "none") + gglagplot(s05\_ts[,2]) + theme(legend.position = "none")  
p6 <- gglagplot(s06\_ts[,1]) + theme(legend.position = "none") + gglagplot(s06\_ts[,2])

grid.arrange(p1, p2, p3, p4, p5, p6, nrow = 2)

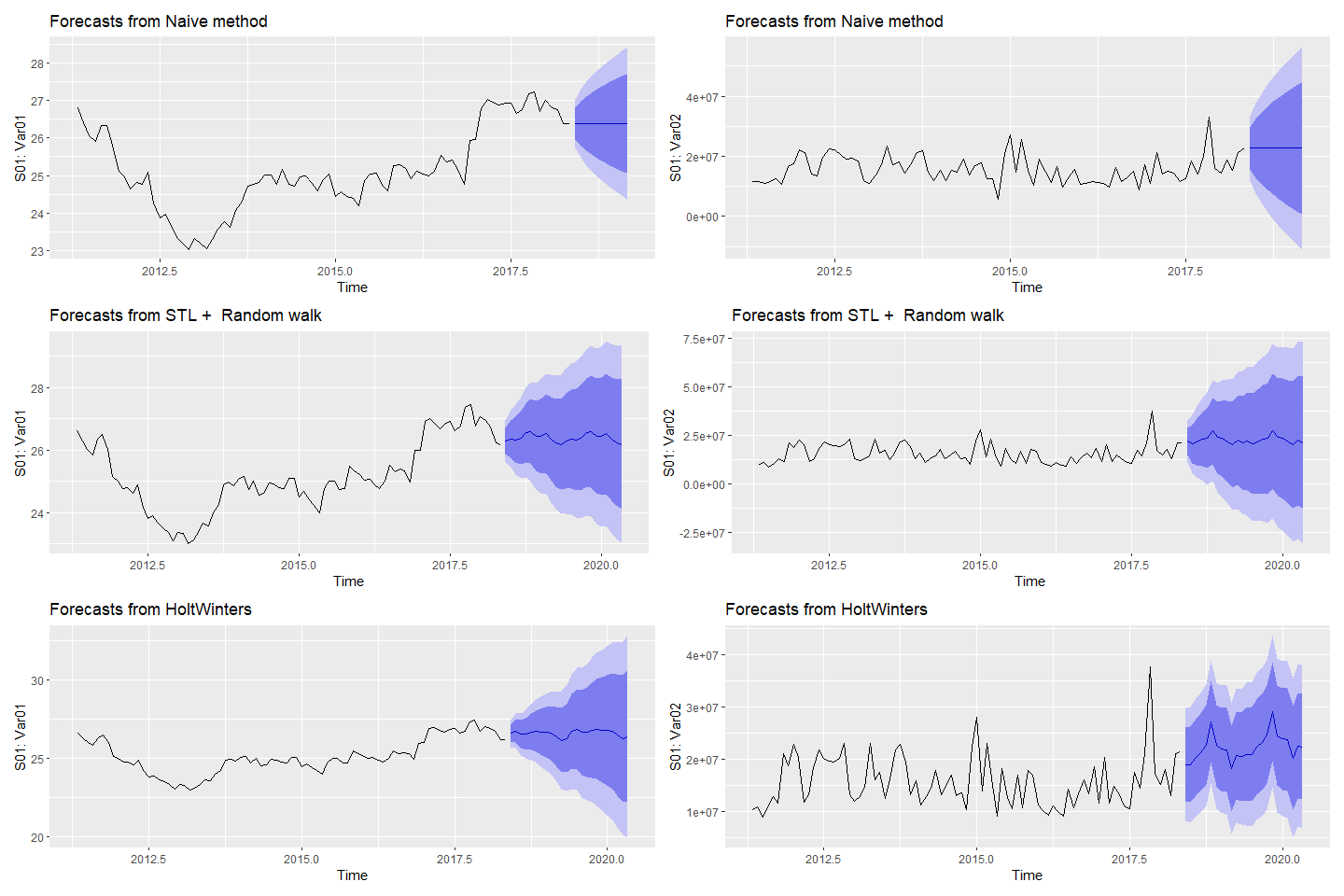


## Data Modeling and Forecasting

### Forecasting with Decomposition

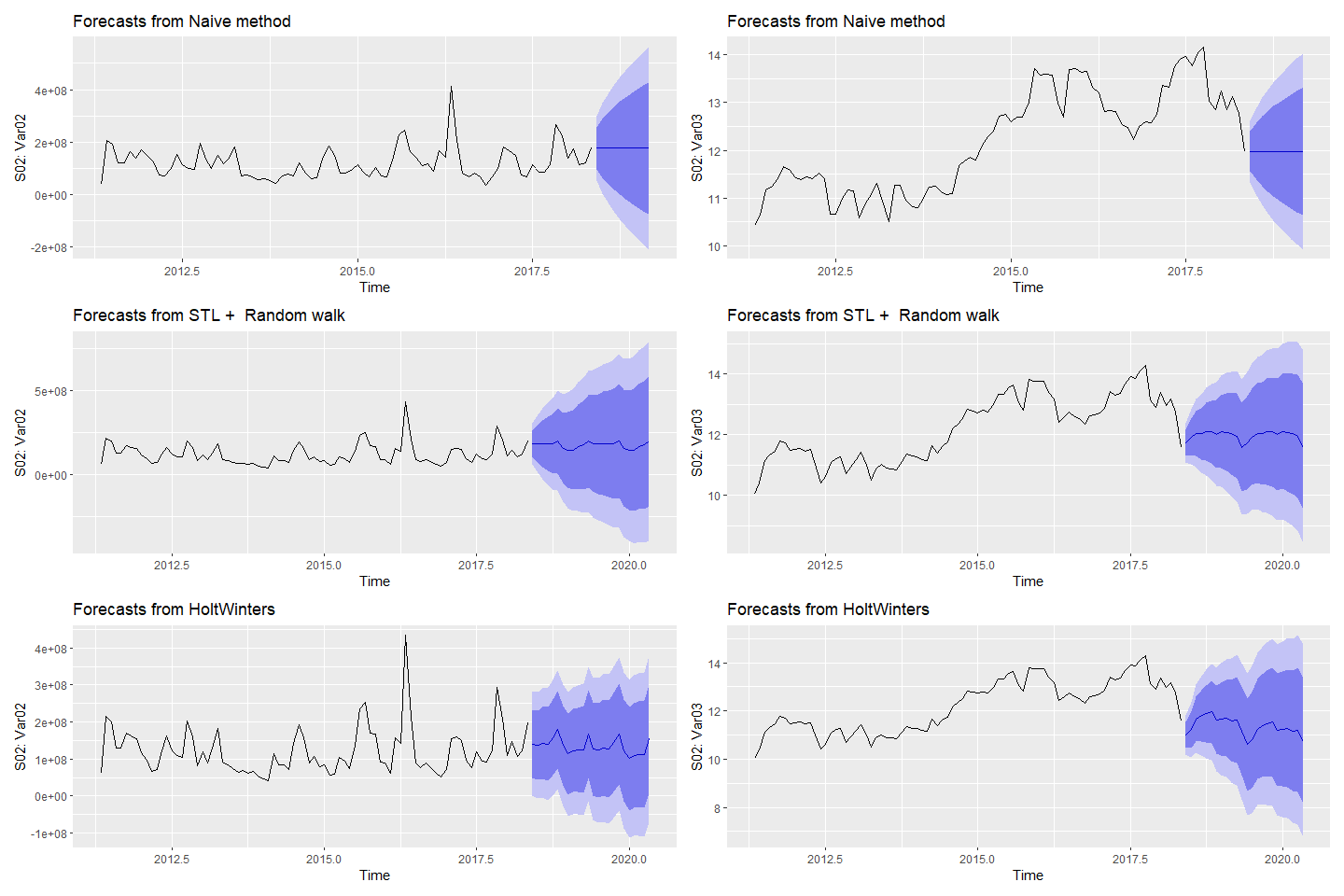
#### Forecasting S01: Var01 & Var02 with decomposition

#STL using default values  
fit\_stl\_1 <- stl(s01\_ts[,1], s.window = "periodic")  
  
#STL using default values  
fit\_stl\_2 <- stl(s01\_ts[,2], s.window = "periodic")  
  
#forecast of seasonaly adjusted data  
f1 <- fit\_stl\_1 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S01: Var01")  
  
#forecast of seasonaly adjusted data  
f2 <- fit\_stl\_2 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S01: Var02")  
  
#forecast from SLT + Random walk  
f3 <- fit\_stl\_1 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S01: Var01")  
  
#forecast from SLT + Random walk  
f4 <- fit\_stl\_2 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S01: Var02")  
  
#forecast from Holt-Winters  
hw1 <- HoltWinters(s01\_ts[,1])  
  
#forecast from Holt-Winters  
hw2 <- HoltWinters(s01\_ts[,2])  
  
f5 <- hw1 %>% forecast() %>%  
 autoplot() + ylab("S01: Var01")  
  
f6 <- hw2 %>% forecast() %>%  
 autoplot() + ylab("S01: Var02")  
  
(f1 + f2) / (f3 + f4) / (f5 + f6)



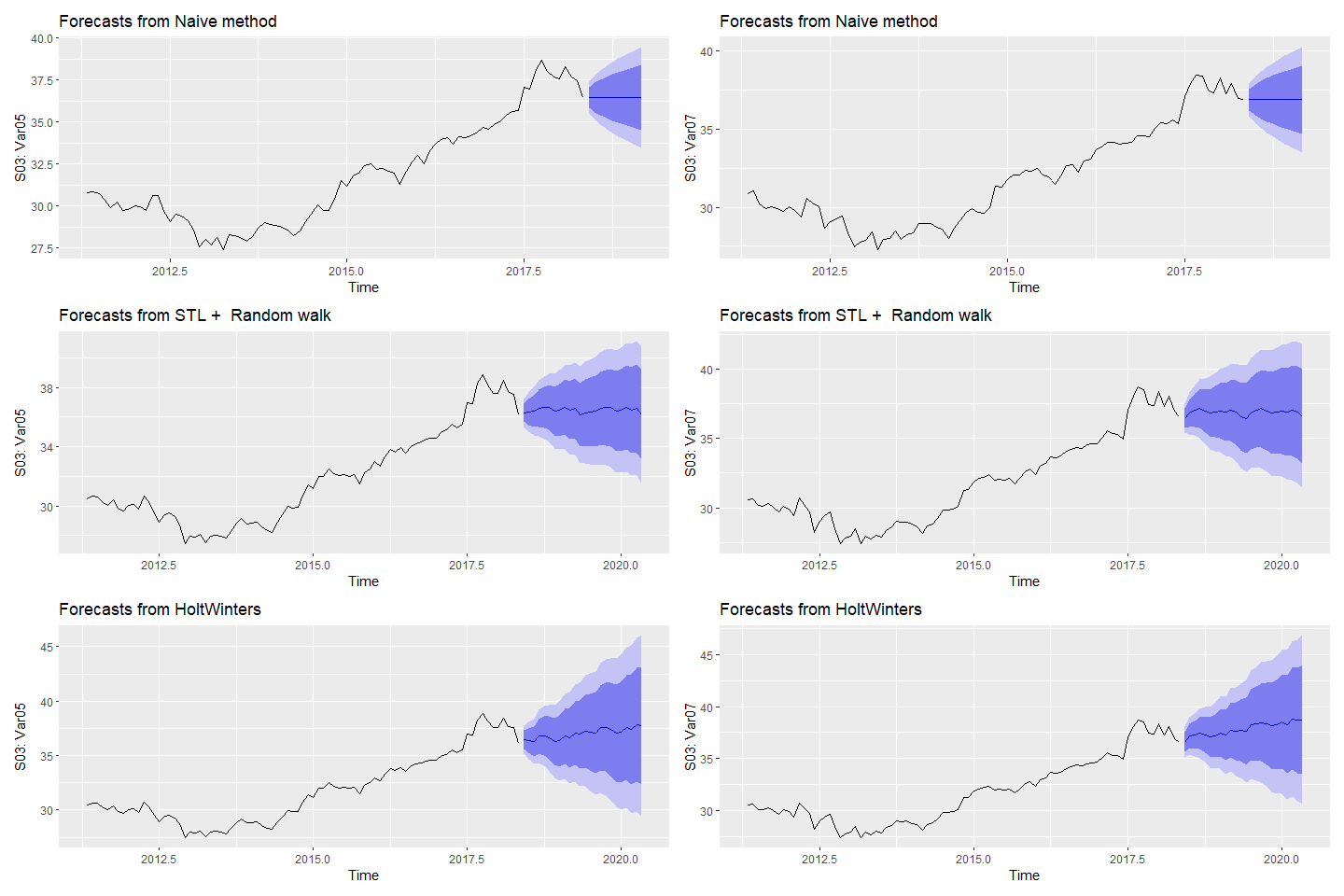
#### Forecasting S02: Var02 & Var03 with decomposition

#STL using default values  
fit\_stl\_1 <- stl(s02\_ts[,1], s.window = "periodic")  
  
#STL using default values  
fit\_stl\_2 <- stl(s02\_ts[,2], s.window = "periodic")  
  
#forecast of seasonaly adjusted data  
f1 <- fit\_stl\_1 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S02: Var02")  
  
#forecast of seasonaly adjusted data  
f2 <- fit\_stl\_2 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S02: Var03")  
  
#forecast from SLT + Random walk  
f3 <- fit\_stl\_1 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S02: Var02")  
  
#forecast from SLT + Random walk  
f4 <- fit\_stl\_2 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S02: Var03")  
  
#forecast from Holt-Winters  
hw1 <- HoltWinters(s02\_ts[,1])  
  
#forecast from Holt-Winters  
hw2 <- HoltWinters(s02\_ts[,2])  
  
f5 <- hw1 %>% forecast() %>%  
 autoplot() + ylab("S02: Var02")  
  
f6 <- hw2 %>% forecast() %>%  
 autoplot() + ylab("S02: Var03")  
  
(f1 + f2) / (f3 + f4) / (f5 + f6)



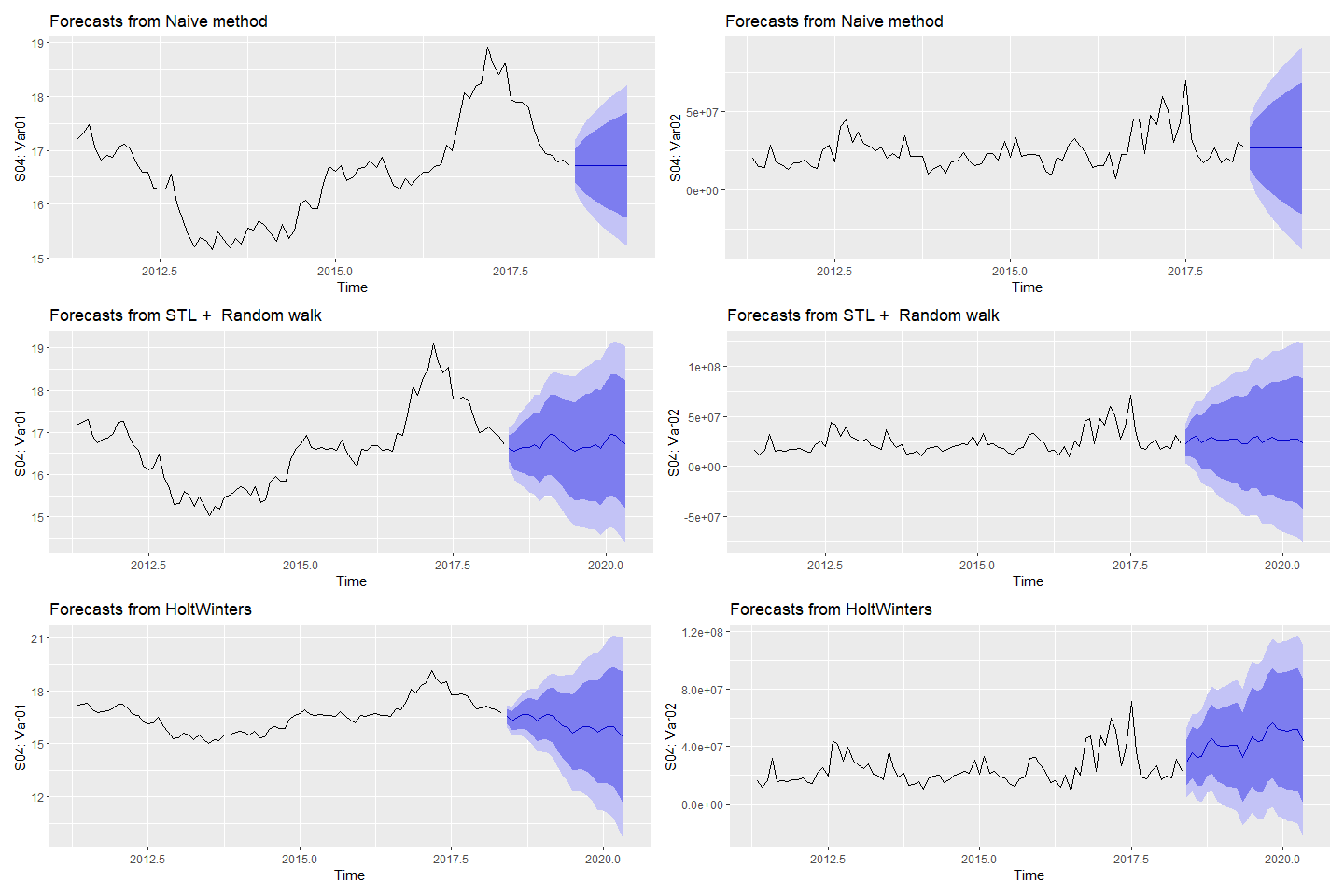
#### Forecasting S03: Var05 & Var07 with decomposition

#STL using default values  
fit\_stl\_1 <- stl(s03\_ts[,1], s.window = "periodic")  
  
#STL using default values  
fit\_stl\_2 <- stl(s03\_ts[,2], s.window = "periodic")  
  
#forecast of seasonaly adjusted data  
f1 <- fit\_stl\_1 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S03: Var05")  
  
#forecast of seasonaly adjusted data  
f2 <- fit\_stl\_2 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S03: Var07")  
  
#forecast from SLT + Random walk  
f3 <- fit\_stl\_1 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S03: Var05")  
  
#forecast from SLT + Random walk  
f4 <- fit\_stl\_2 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S03: Var07")  
  
#forecast from Holt-Winters  
hw1 <- HoltWinters(s03\_ts[,1])  
  
#forecast from Holt-Winters  
hw2 <- HoltWinters(s03\_ts[,2])  
  
f5 <- hw1 %>% forecast() %>%  
 autoplot() + ylab("S03: Var05")  
  
f6 <- hw2 %>% forecast() %>%  
 autoplot() + ylab("S03: Var07")  
  
(f1 + f2) / (f3 + f4) / (f5 + f6)



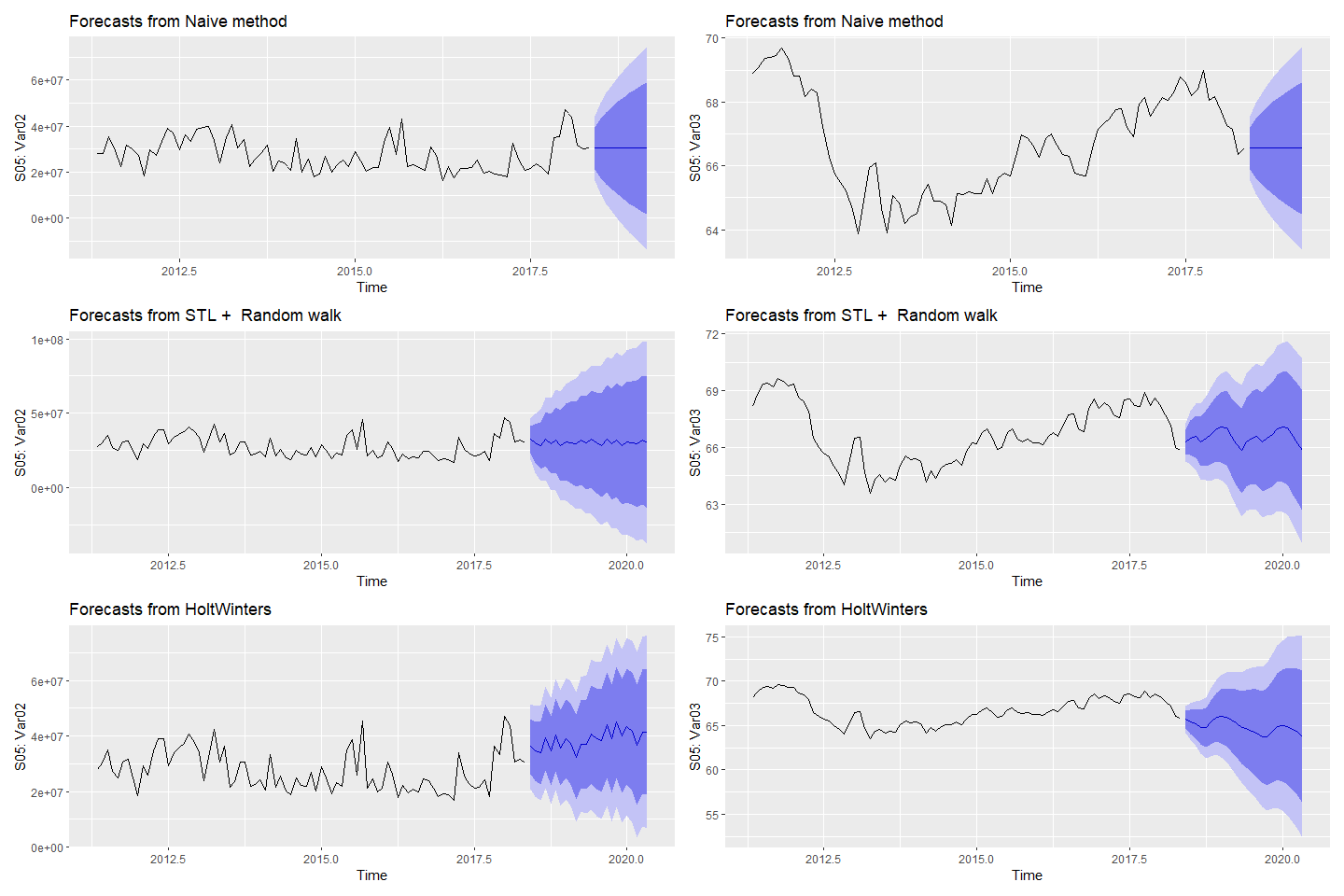
#### Forecasting S04: Var01 & Var02 with decomposition

#STL using default values  
fit\_stl\_1 <- stl(s04\_ts[,1], s.window = "periodic")  
  
#STL using default values  
fit\_stl\_2 <- stl(s04\_ts[,2], s.window = "periodic")  
  
#forecast of seasonaly adjusted data  
f1 <- fit\_stl\_1 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S04: Var01")  
  
#forecast of seasonaly adjusted data  
f2 <- fit\_stl\_2 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S04: Var02")  
  
#forecast from SLT + Random walk  
f3 <- fit\_stl\_1 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S04: Var01")  
  
#forecast from SLT + Random walk  
f4 <- fit\_stl\_2 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S04: Var02")  
  
#forecast from Holt-Winters  
hw1 <- HoltWinters(s04\_ts[,1])  
  
#forecast from Holt-Winters  
hw2 <- HoltWinters(s04\_ts[,2])  
  
f5 <- hw1 %>% forecast() %>%  
 autoplot() + ylab("S04: Var01")  
  
f6 <- hw2 %>% forecast() %>%  
 autoplot() + ylab("S04: Var02")  
  
(f1 + f2) / (f3 + f4) / (f5 + f6)



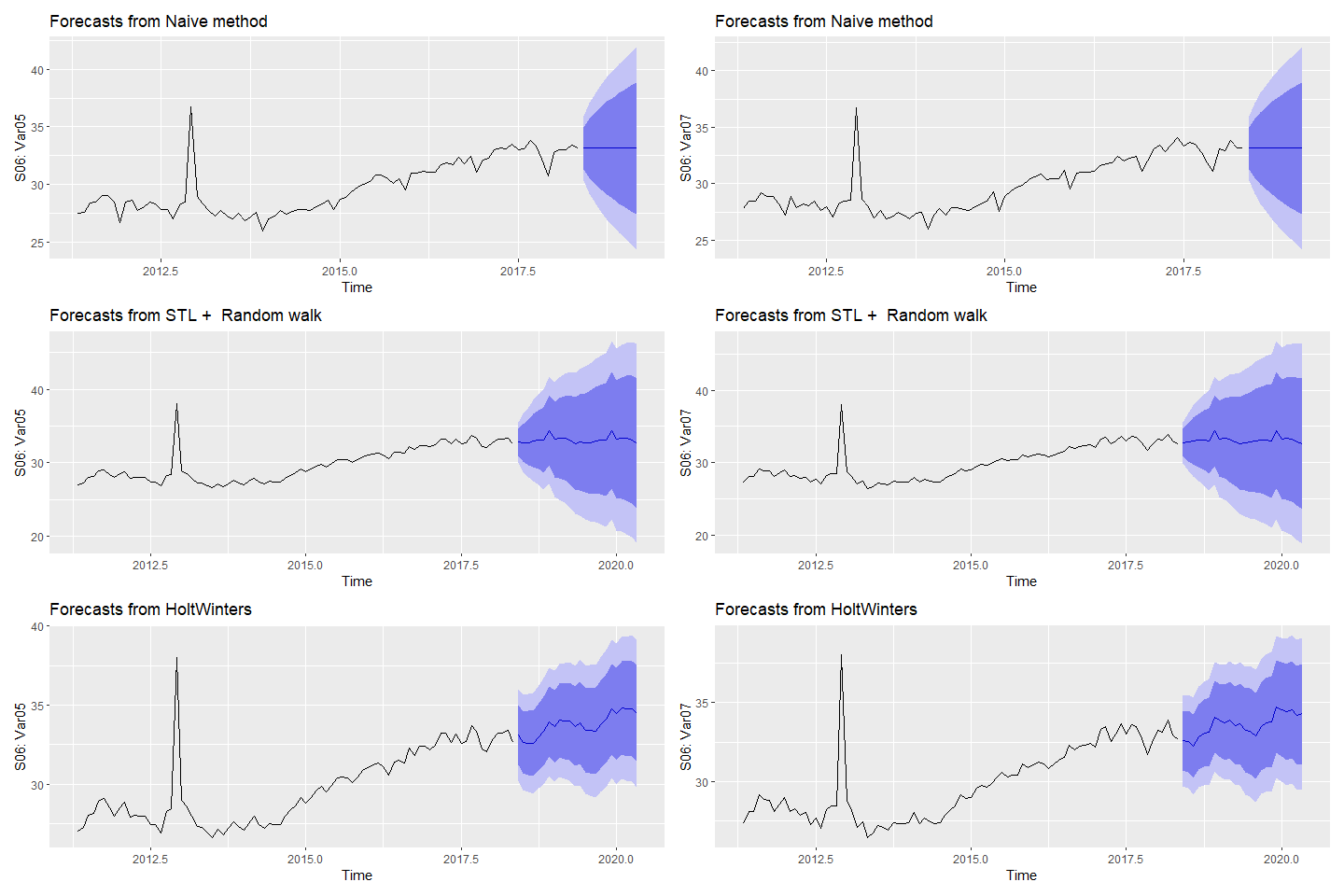
#### Forecasting S05: Var02 & Var03 with decomposition

#STL using default values  
fit\_stl\_1 <- stl(s05\_ts[,1], s.window = "periodic")  
  
#STL using default values  
fit\_stl\_2 <- stl(s05\_ts[,2], s.window = "periodic")  
  
#forecast of seasonaly adjusted data  
f1 <- fit\_stl\_1 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S05: Var02")  
  
#forecast of seasonaly adjusted data  
f2 <- fit\_stl\_2 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S05: Var03")  
  
#forecast from SLT + Random walk  
f3 <- fit\_stl\_1 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S05: Var02")  
  
#forecast from SLT + Random walk  
f4 <- fit\_stl\_2 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S05: Var03")  
  
#forecast from Holt-Winters  
hw1 <- HoltWinters(s05\_ts[,1])  
  
#forecast from Holt-Winters  
hw2 <- HoltWinters(s05\_ts[,2])  
  
f5 <- hw1 %>% forecast() %>%  
 autoplot() + ylab("S05: Var02")  
  
f6 <- hw2 %>% forecast() %>%  
 autoplot() + ylab("S05: Var03")  
  
(f1 + f2) / (f3 + f4) / (f5 + f6)



#### Forecasting S06: Var05 & Var07 with decomposition

#STL using default values  
fit\_stl\_1 <- stl(s06\_ts[,1], s.window = "periodic")  
  
#STL using default values  
fit\_stl\_2 <- stl(s06\_ts[,2], s.window = "periodic")  
  
#forecast of seasonaly adjusted data  
f1 <- fit\_stl\_1 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S06: Var05")  
  
#forecast of seasonaly adjusted data  
f2 <- fit\_stl\_2 %>% seasadj() %>% naive()%>%  
 autoplot() + ylab("S06: Var07")  
  
#forecast from SLT + Random walk  
f3 <- fit\_stl\_1 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S06: Var05")  
  
#forecast from SLT + Random walk  
f4 <- fit\_stl\_2 %>% forecast(method="naive") %>%  
 autoplot() + ylab("S06: Var07")  
  
#forecast from Holt-Winters  
hw1 <- HoltWinters(s06\_ts[,1])  
  
#forecast from Holt-Winters  
hw2 <- HoltWinters(s06\_ts[,2])  
  
f5 <- hw1 %>% forecast() %>%  
 autoplot() + ylab("S06: Var05")  
  
f6 <- hw2 %>% forecast() %>%  
 autoplot() + ylab("S06: Var07")  
  
(f1 + f2) / (f3 + f4) / (f5 + f6)

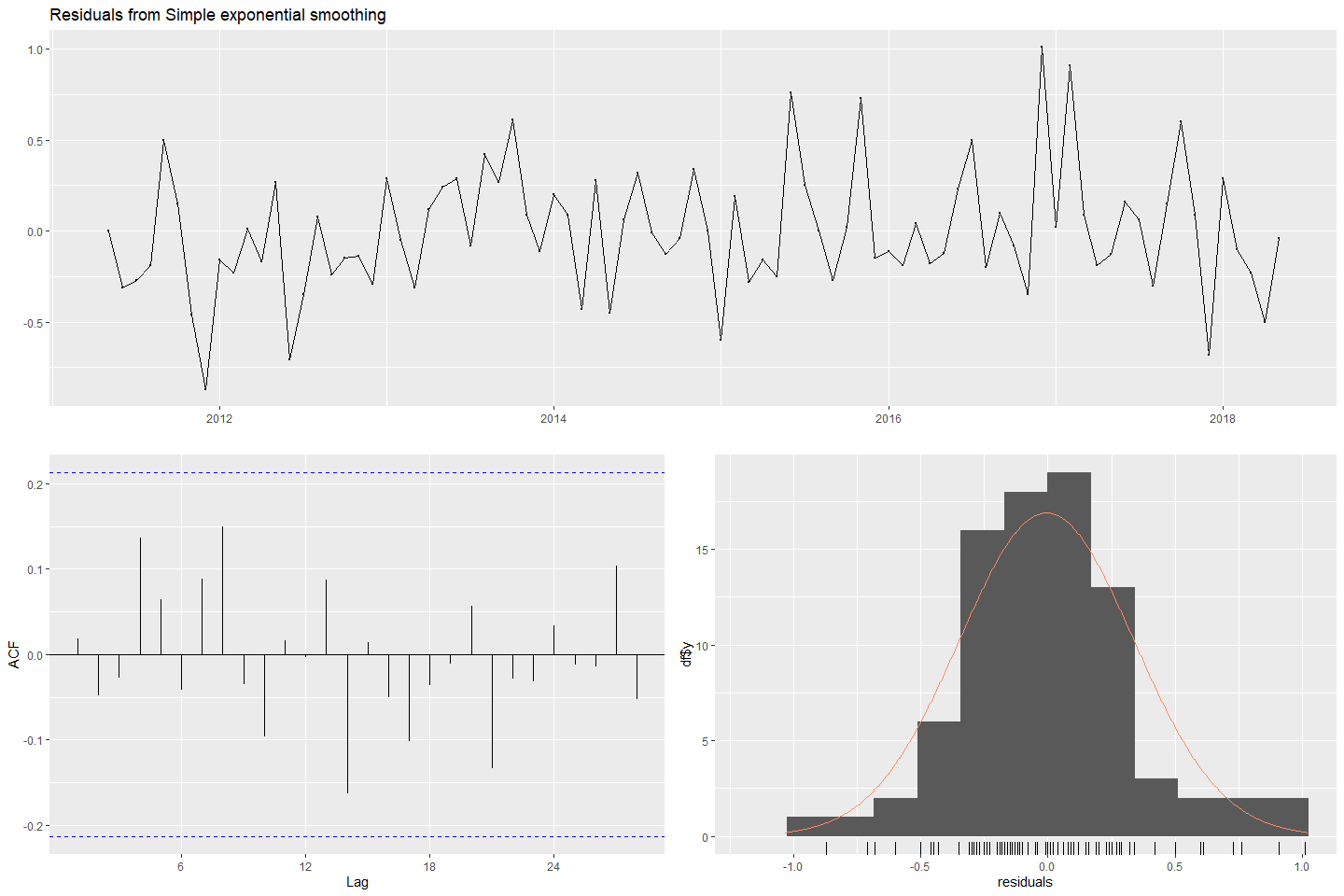


### Forecasting with Simple Exponential Smoothing (SES)

### Forecasting

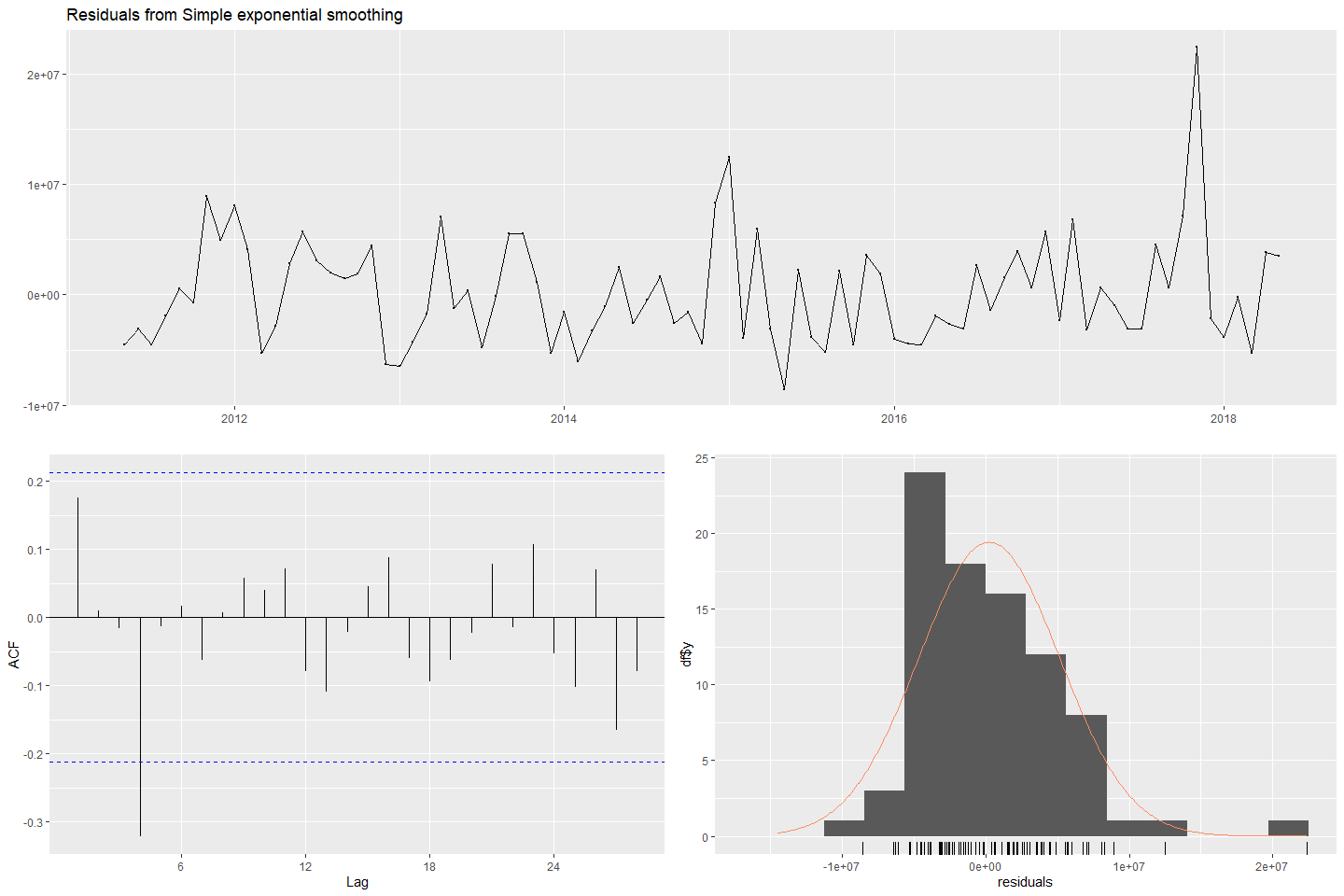
### Forecasting S01: Var01 & Var02 with ses

# simple exponential smoothing since there is no clear trend or seasonal pattern  
# Fit ses models to Var01 & Var02  
fit\_s01\_ses\_1 <- ses(s01\_ts[,1])  
fit\_s01\_ses\_2 <- ses(s01\_ts[,2])  
# check residuals  
checkresiduals(fit\_s01\_ses\_1)



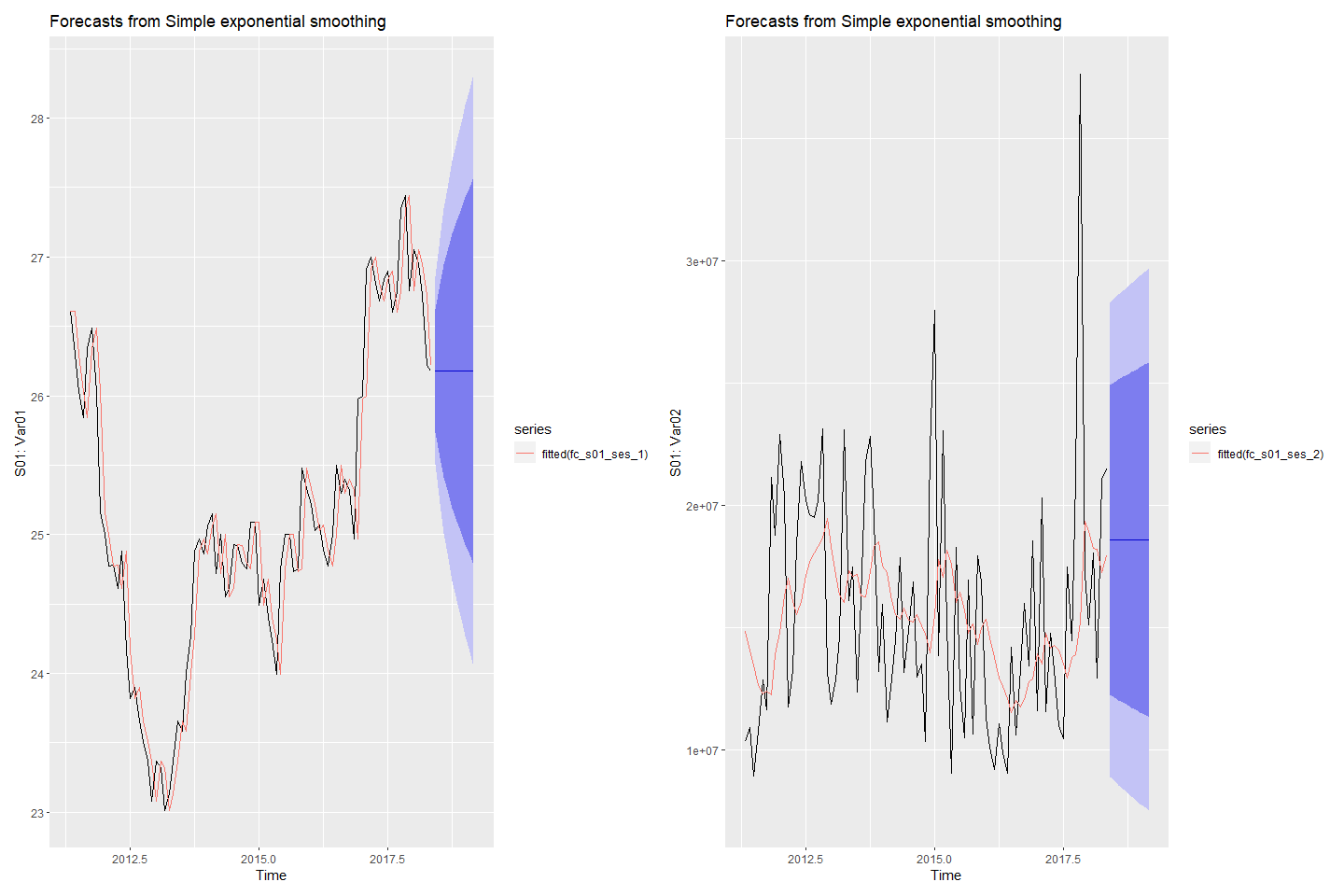
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 11.445, df = 15, p-value = 0.7205  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s01\_ses\_2)



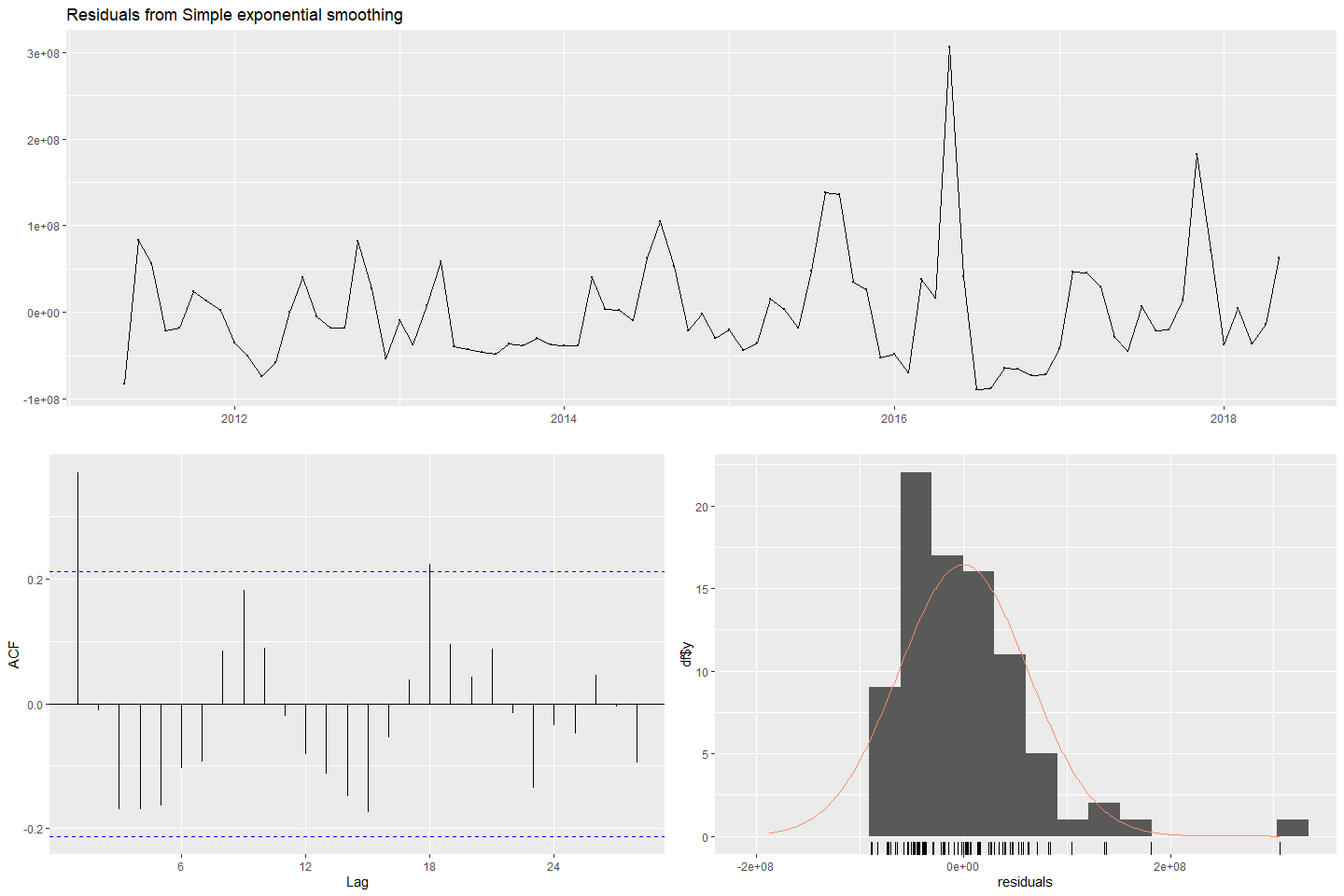
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 16.925, df = 15, p-value = 0.3234  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s01\_ses\_1 <- forecast(fit\_s01\_ses\_1)  
fc\_s01\_ses\_2 <- forecast(fit\_s01\_ses\_2)  
# plot forecasts  
fses\_S01\_1 <- autoplot(fc\_s01\_ses\_1) + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s01\_ses\_1))  
fses\_S01\_2 <- autoplot(fc\_s01\_ses\_2) + ylab("S01: Var02") +  
 autolayer(fitted(fc\_s01\_ses\_2))  
(fses\_S01\_1 + fses\_S01\_2)



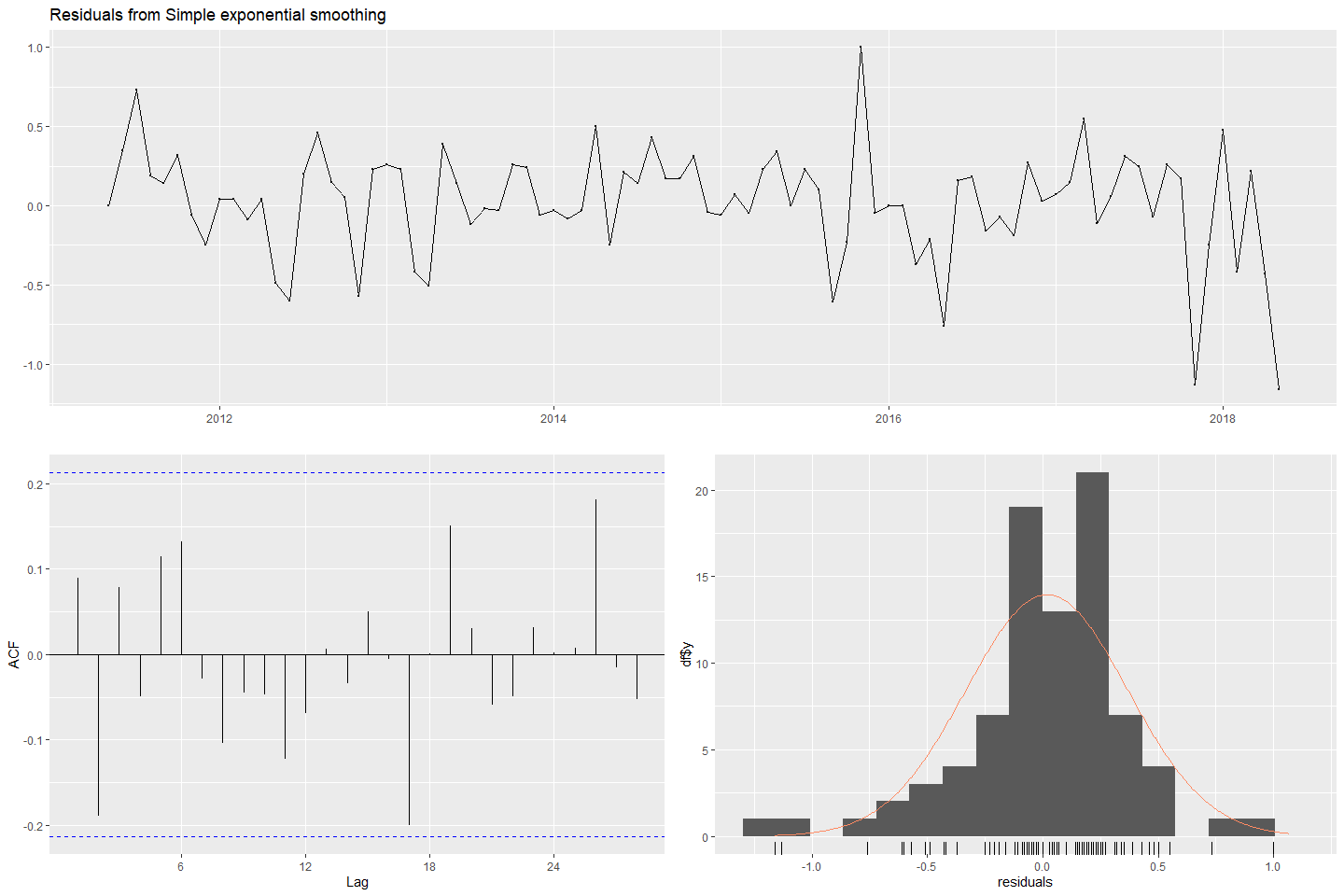
#### Forecasting S02: Var02 & Var03 with ses

# simple exponential smoothing since there is no clear trend or seasonal pattern  
# Fit ses models to Var02 & Var03  
fit\_s02\_ses\_2 <- ses(s02\_ts[,1])  
fit\_s02\_ses\_3 <- ses(s02\_ts[,2])  
# check residuals  
checkresiduals(fit\_s02\_ses\_2)



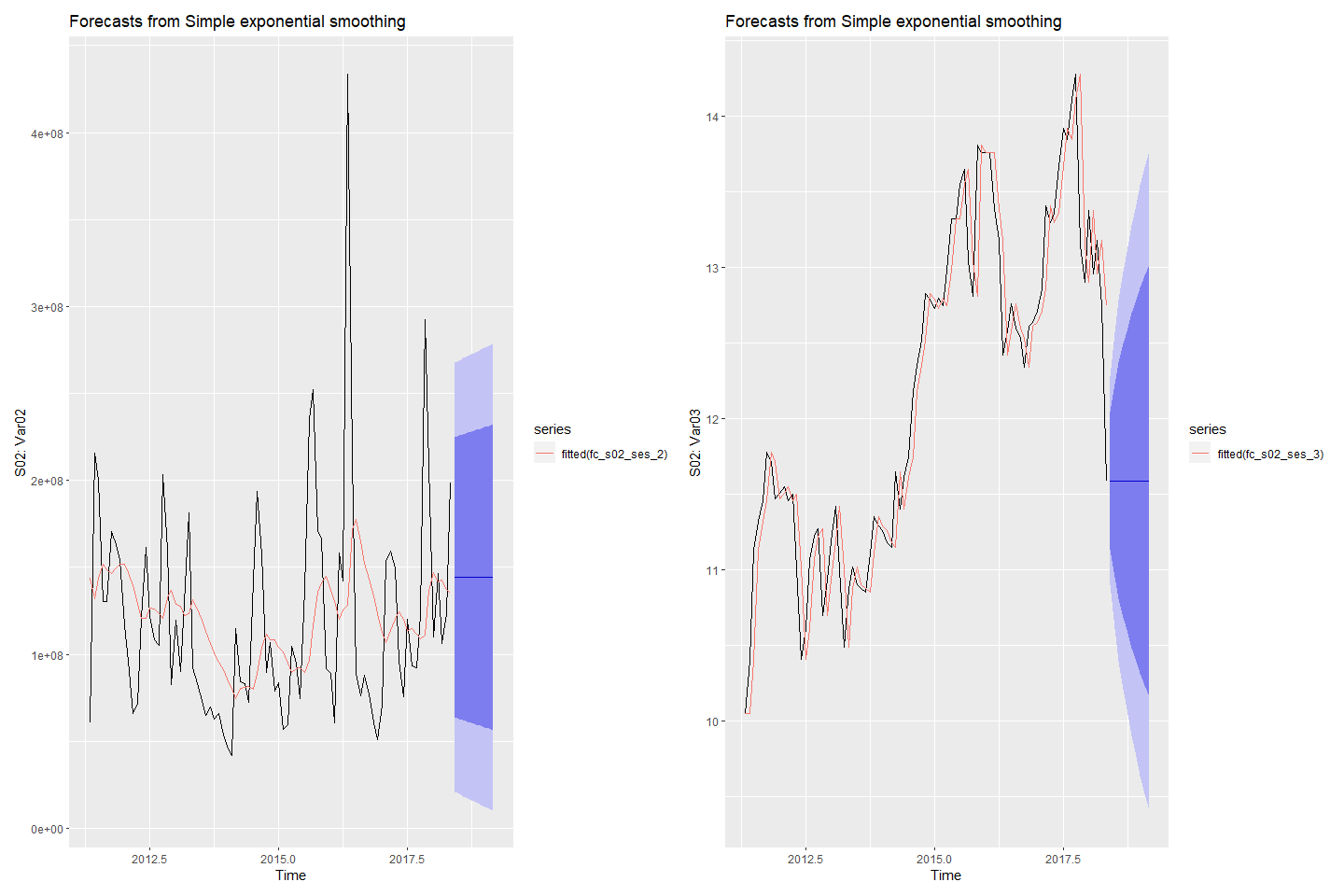
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 34.481, df = 15, p-value = 0.002914  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s02\_ses\_3)



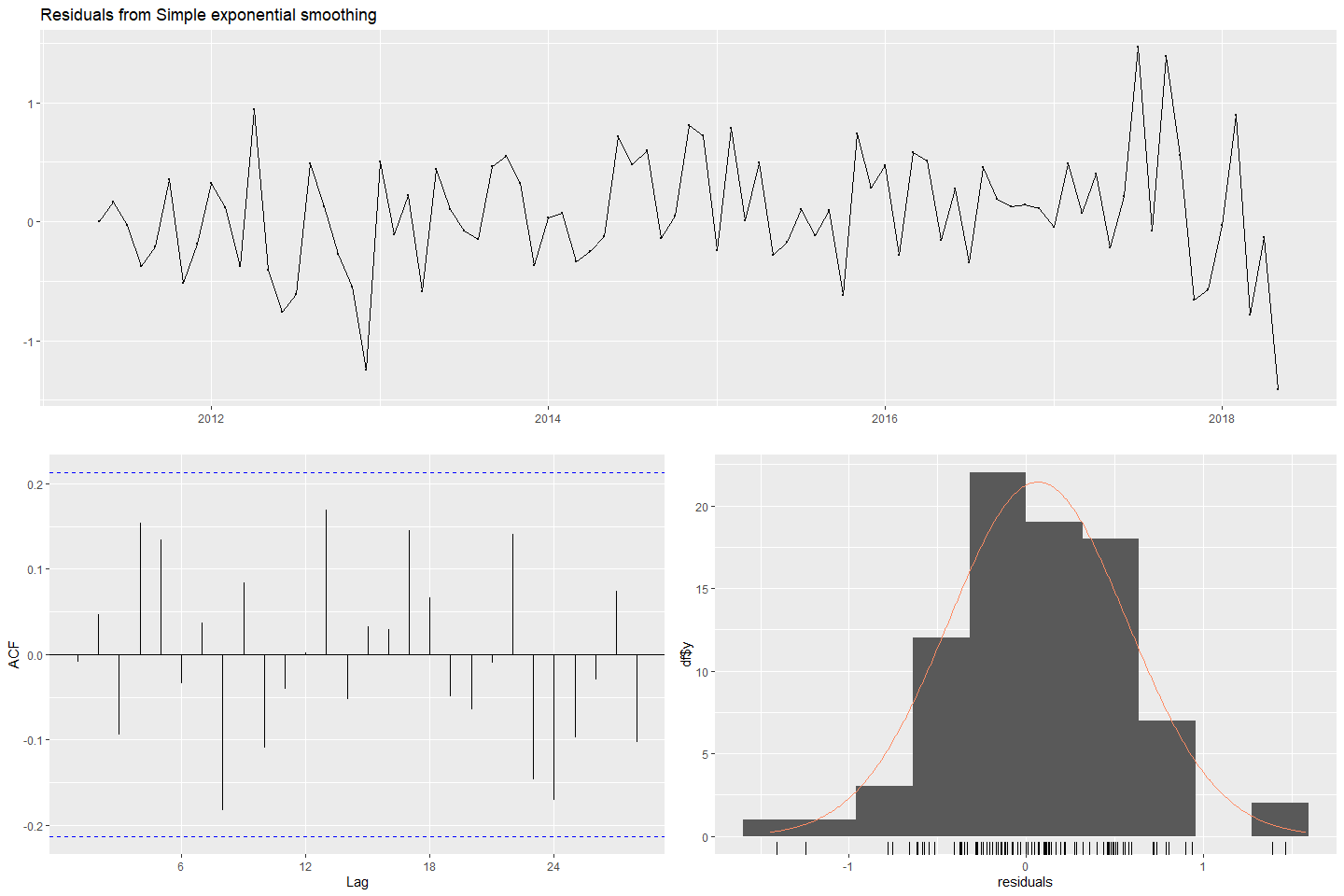
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 15.73, df = 15, p-value = 0.4002  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s02\_ses\_2 <- forecast(fit\_s02\_ses\_2)  
fc\_s02\_ses\_3 <- forecast(fit\_s02\_ses\_3)  
# plot forecasts  
fses\_S02\_2 <- autoplot(fc\_s02\_ses\_2) + ylab("S02: Var02") +  
 autolayer(fitted(fc\_s02\_ses\_2))  
fses\_S02\_3 <- autoplot(fc\_s02\_ses\_3) + ylab("S02: Var03") +  
 autolayer(fitted(fc\_s02\_ses\_3))  
(fses\_S02\_2 + fses\_S02\_3)



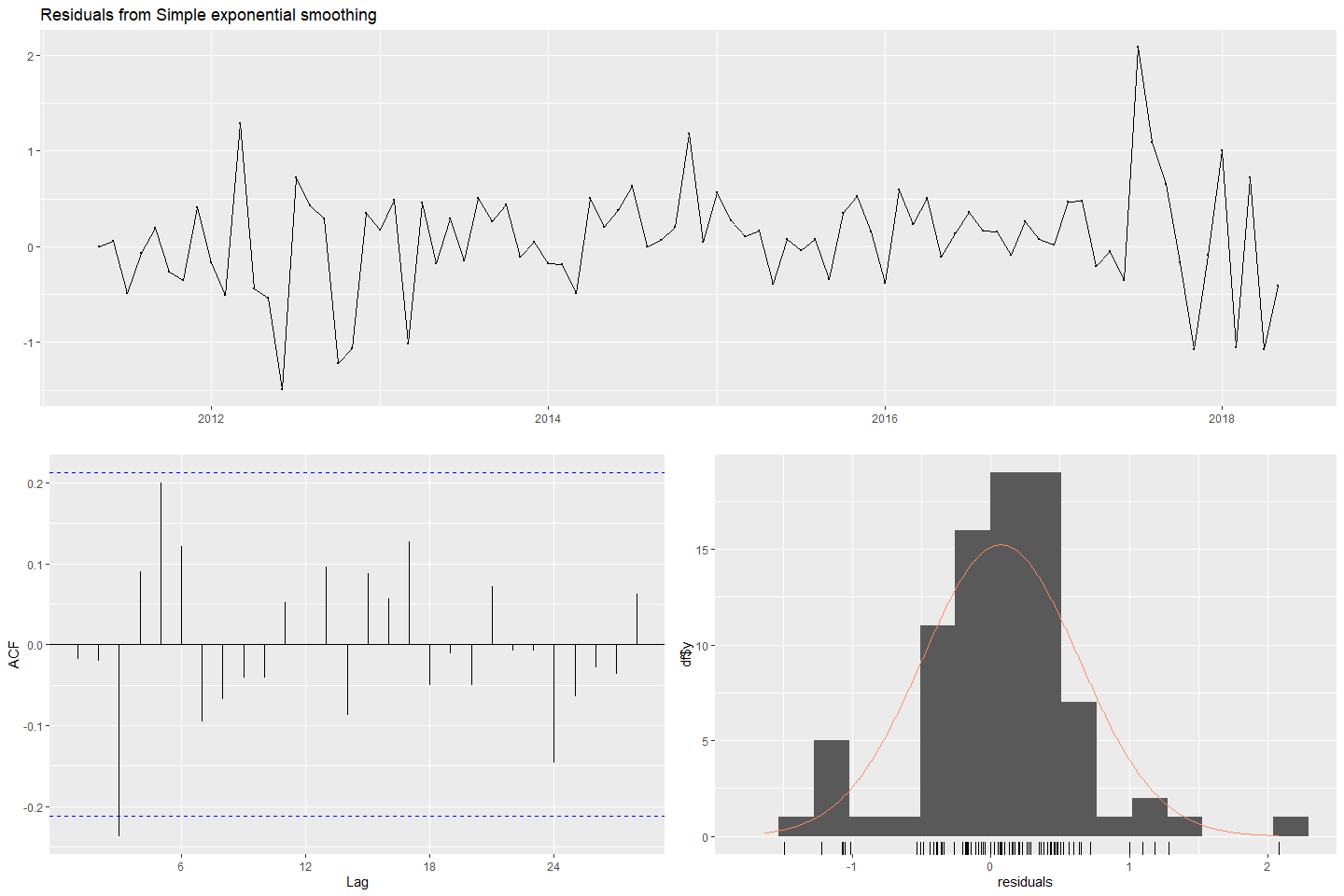
#### Forecasting S03: Var05 & Var07 with ses

# simple exponential smoothing since there is no clear trend or seasonal pattern  
# Fit ses models to Var05 & Var07  
fit\_s03\_ses\_5 <- ses(s03\_ts[,1])  
fit\_s03\_ses\_7 <- ses(s03\_ts[,2])  
# check residuals  
checkresiduals(fit\_s03\_ses\_5)



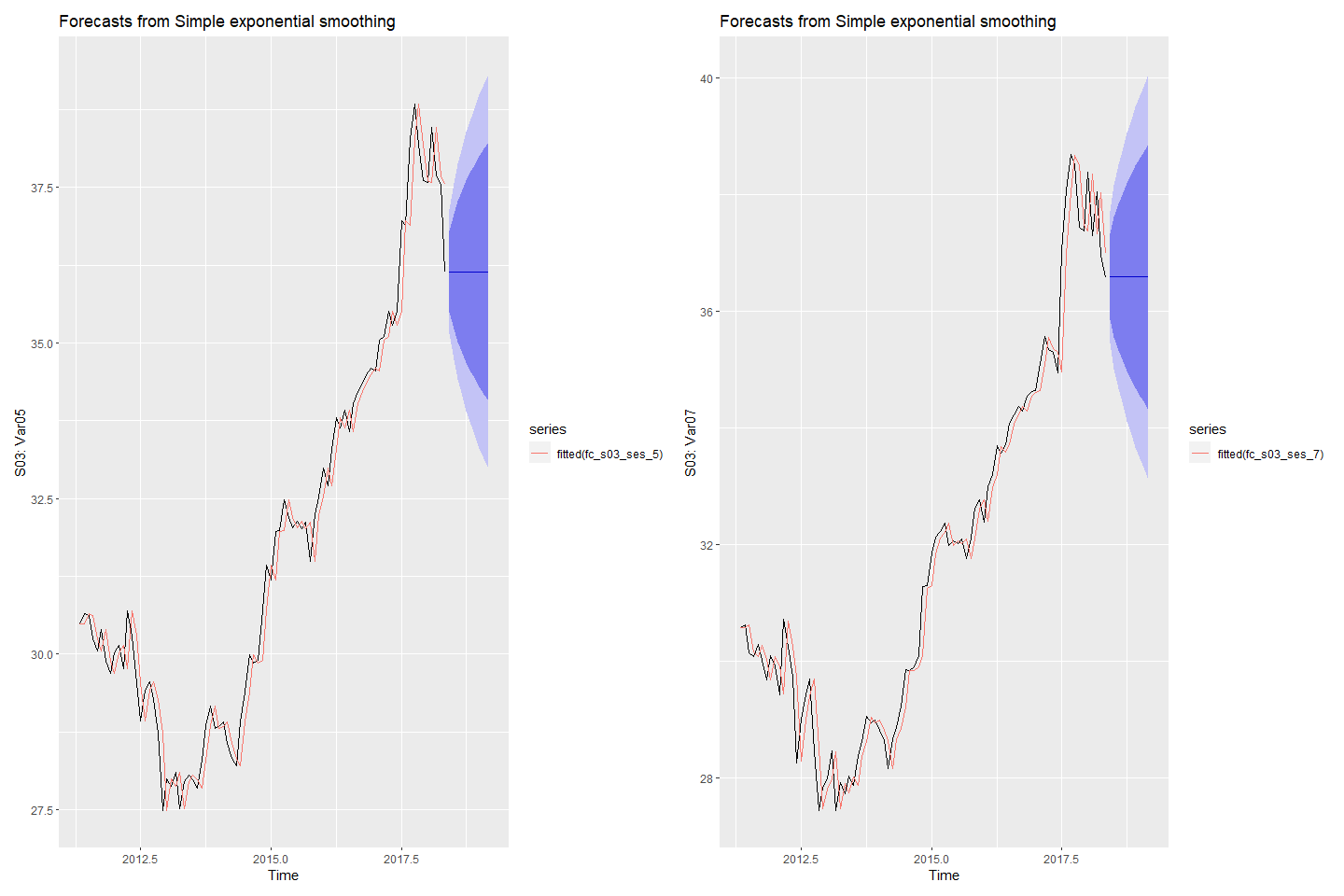
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 16.047, df = 15, p-value = 0.3789  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s03\_ses\_7)



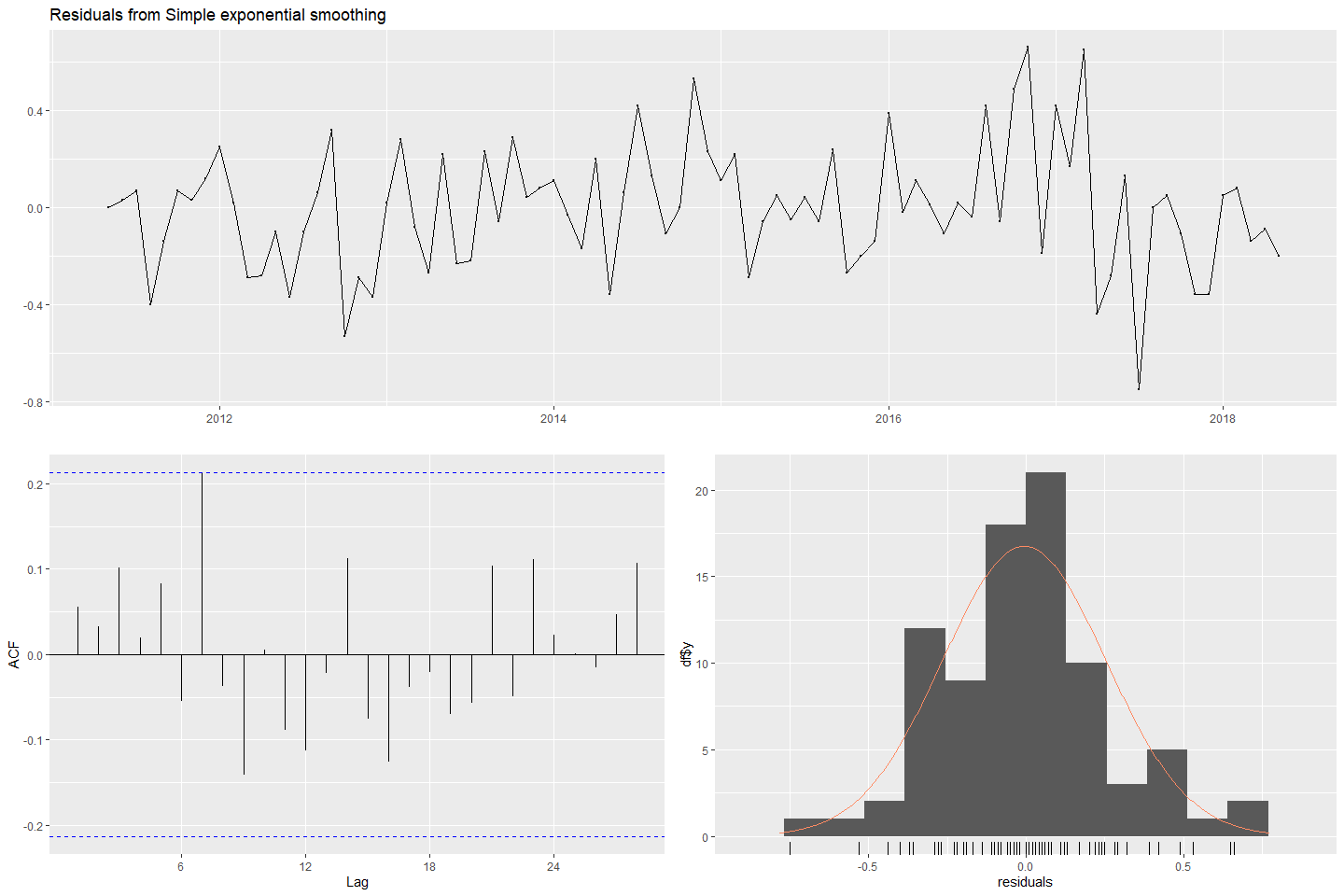
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 17.561, df = 15, p-value = 0.2864  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s03\_ses\_5 <- forecast(fit\_s03\_ses\_5)  
fc\_s03\_ses\_7 <- forecast(fit\_s03\_ses\_7)  
# plot forecasts  
fses\_S03\_5 <- autoplot(fc\_s03\_ses\_5) + ylab("S03: Var05") +  
 autolayer(fitted(fc\_s03\_ses\_5))  
fses\_S03\_7 <- autoplot(fc\_s03\_ses\_7) + ylab("S03: Var07") +  
 autolayer(fitted(fc\_s03\_ses\_7))  
(fses\_S03\_5 + fses\_S03\_7)



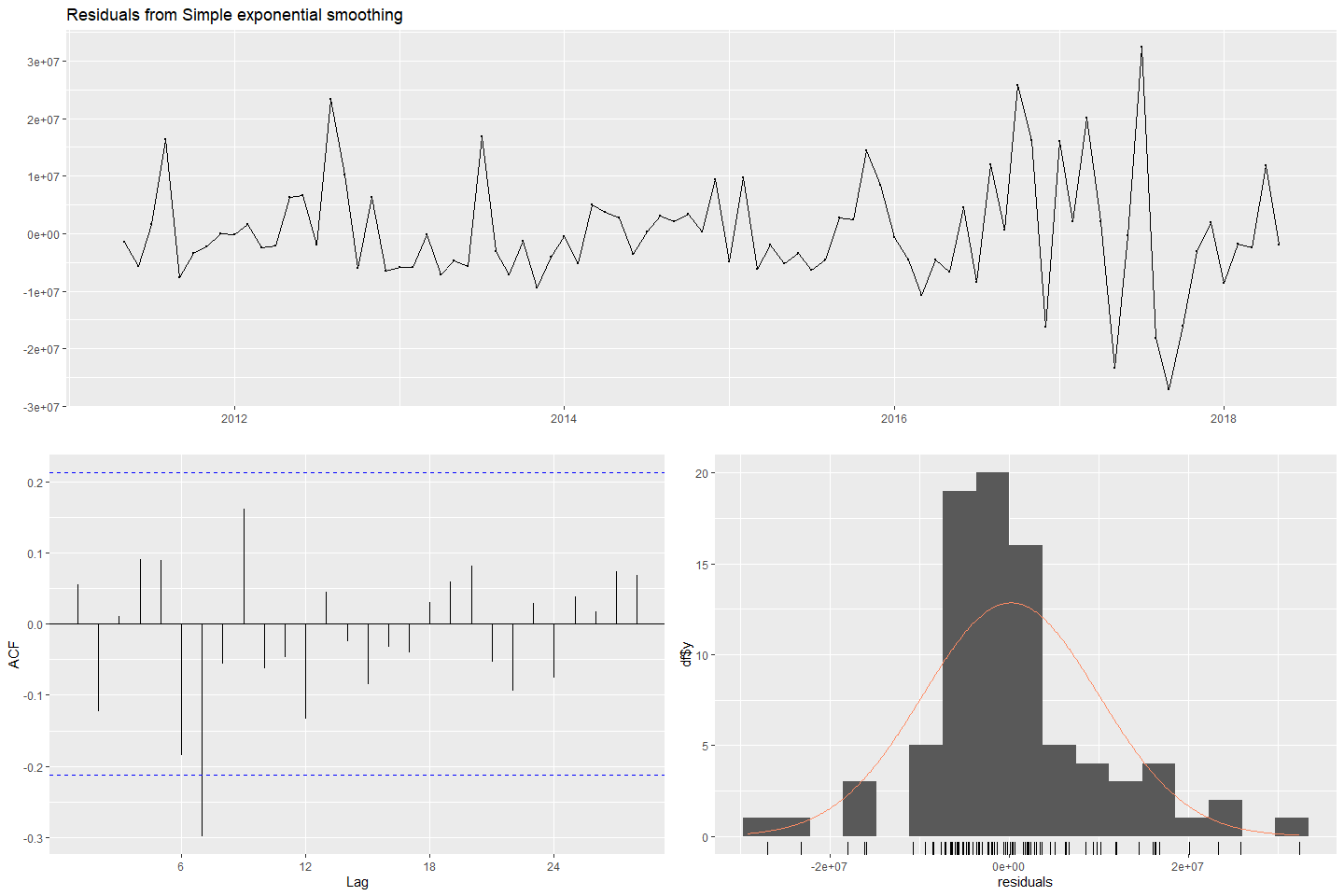
#### Forecasting S04: Var01 & Var02 with ses

# simple exponential smoothing since there is no clear trend or seasonal pattern  
# Fit ses models to Var01 & Var02  
fit\_s04\_ses\_1 <- ses(s04\_ts[,1])  
fit\_s04\_ses\_2 <- ses(s04\_ts[,2])  
# check residuals  
checkresiduals(fit\_s04\_ses\_1)



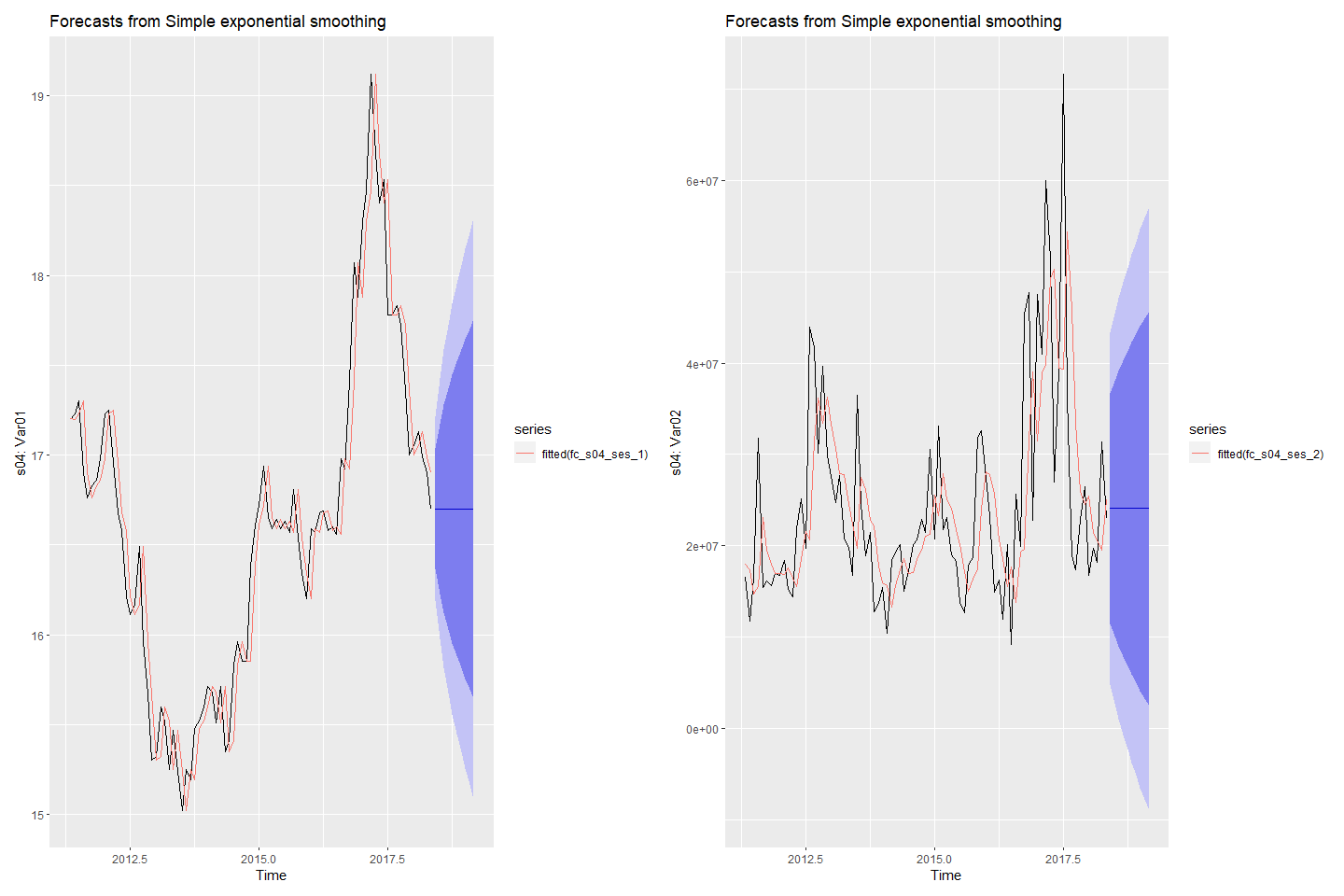
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 14.47, df = 15, p-value = 0.4902  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s04\_ses\_2)



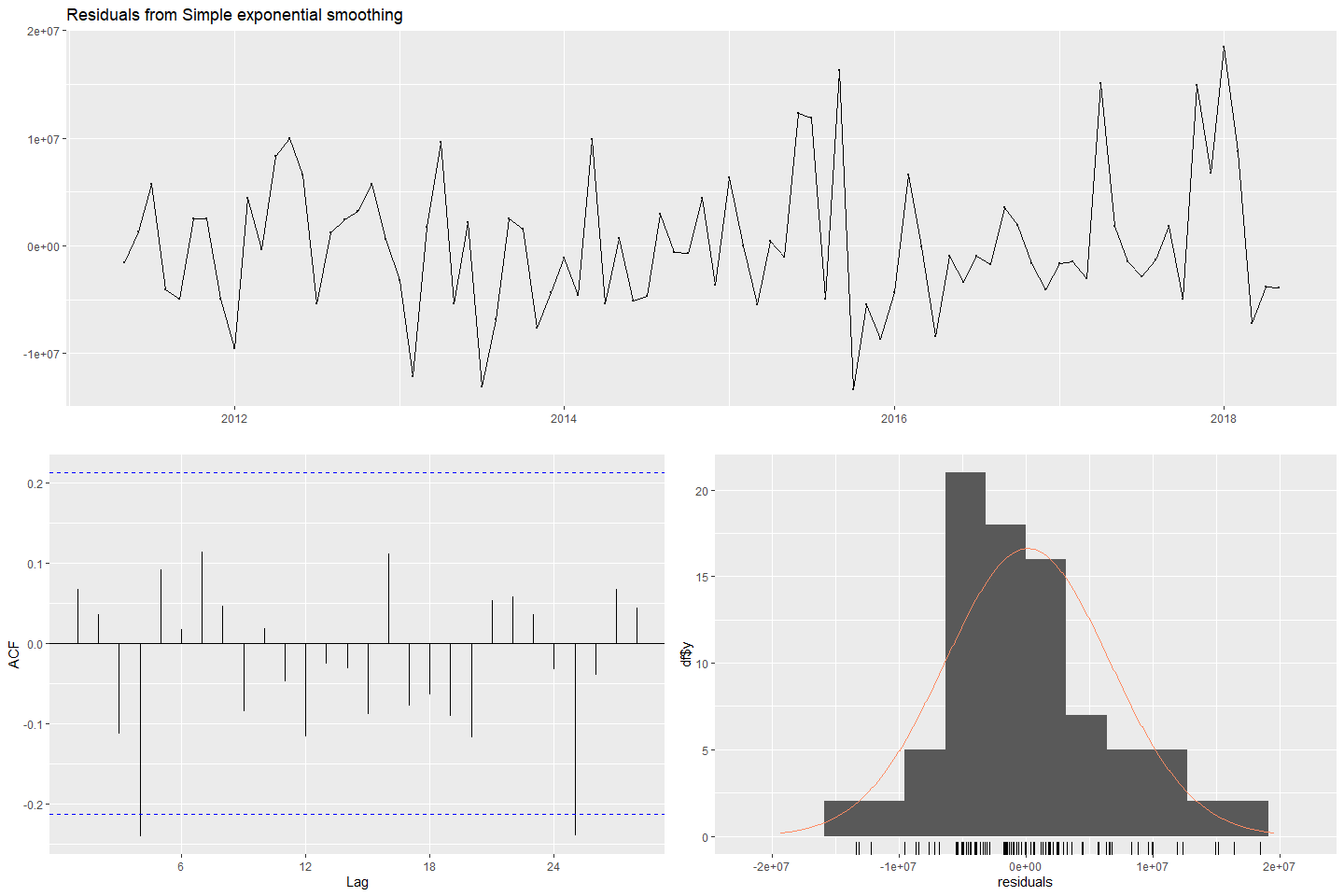
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 21.273, df = 15, p-value = 0.1283  
#>   
#> Model df: 2. Total lags used: 17

# forecast next 140 periods  
fc\_s04\_ses\_1 <- forecast(fit\_s04\_ses\_1)  
fc\_s04\_ses\_2 <- forecast(fit\_s04\_ses\_2)  
# plot forecasts  
fses\_s04\_1 <- autoplot(fc\_s04\_ses\_1) + ylab("s04: Var01") +  
 autolayer(fitted(fc\_s04\_ses\_1))  
fses\_s04\_2 <- autoplot(fc\_s04\_ses\_2) + ylab("s04: Var02") +  
 autolayer(fitted(fc\_s04\_ses\_2))  
(fses\_s04\_1 + fses\_s04\_2)



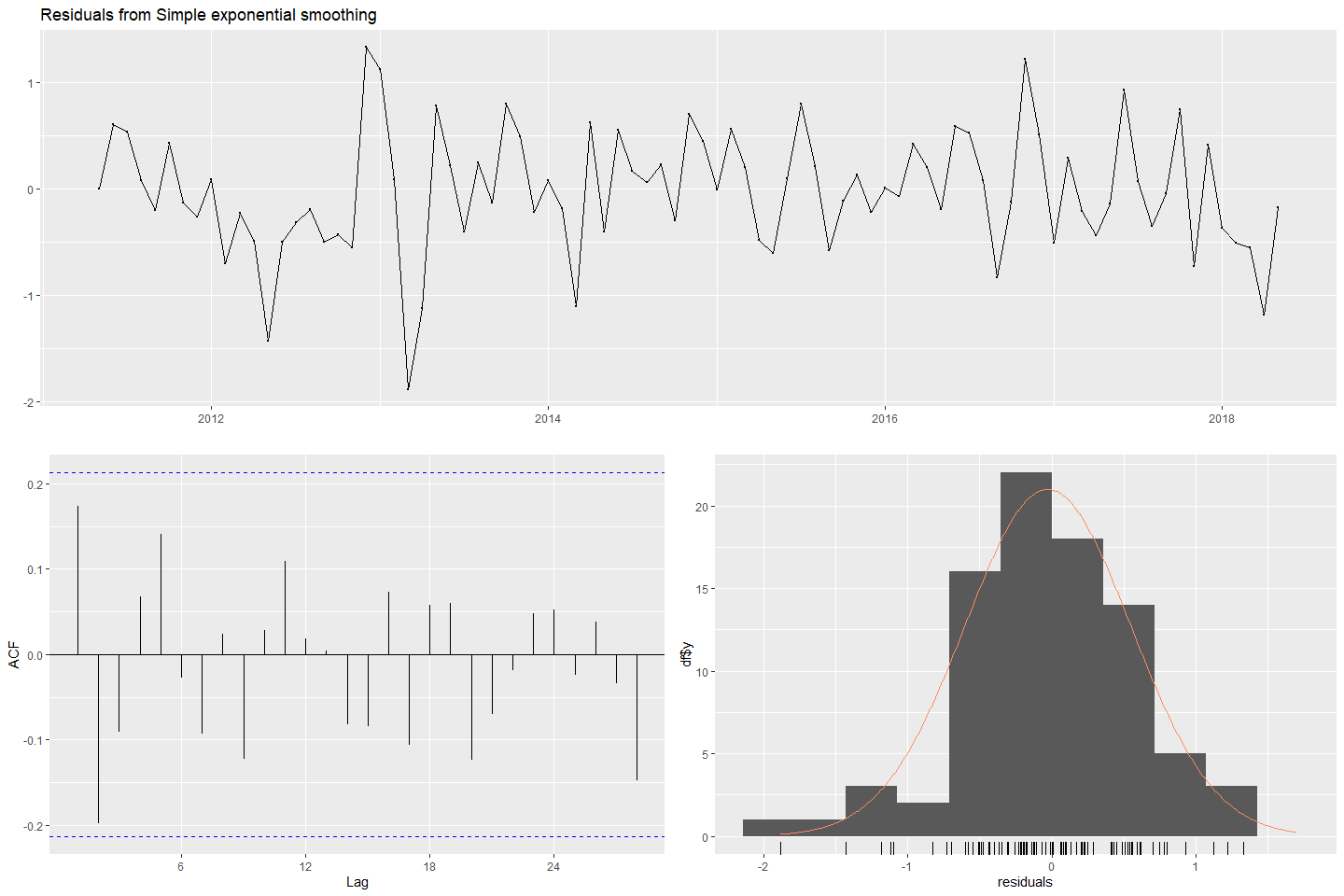
#### Forecasting S05: Var02 & Var03 with ses

# simple exponential smoothing since there is no clear trend or seasonal pattern  
# Fit ses models to Var02 & Var03  
fit\_s05\_ses\_2 <- ses(s05\_ts[,1])  
fit\_s05\_ses\_3 <- ses(s05\_ts[,2])  
# check residuals  
checkresiduals(fit\_s05\_ses\_2)



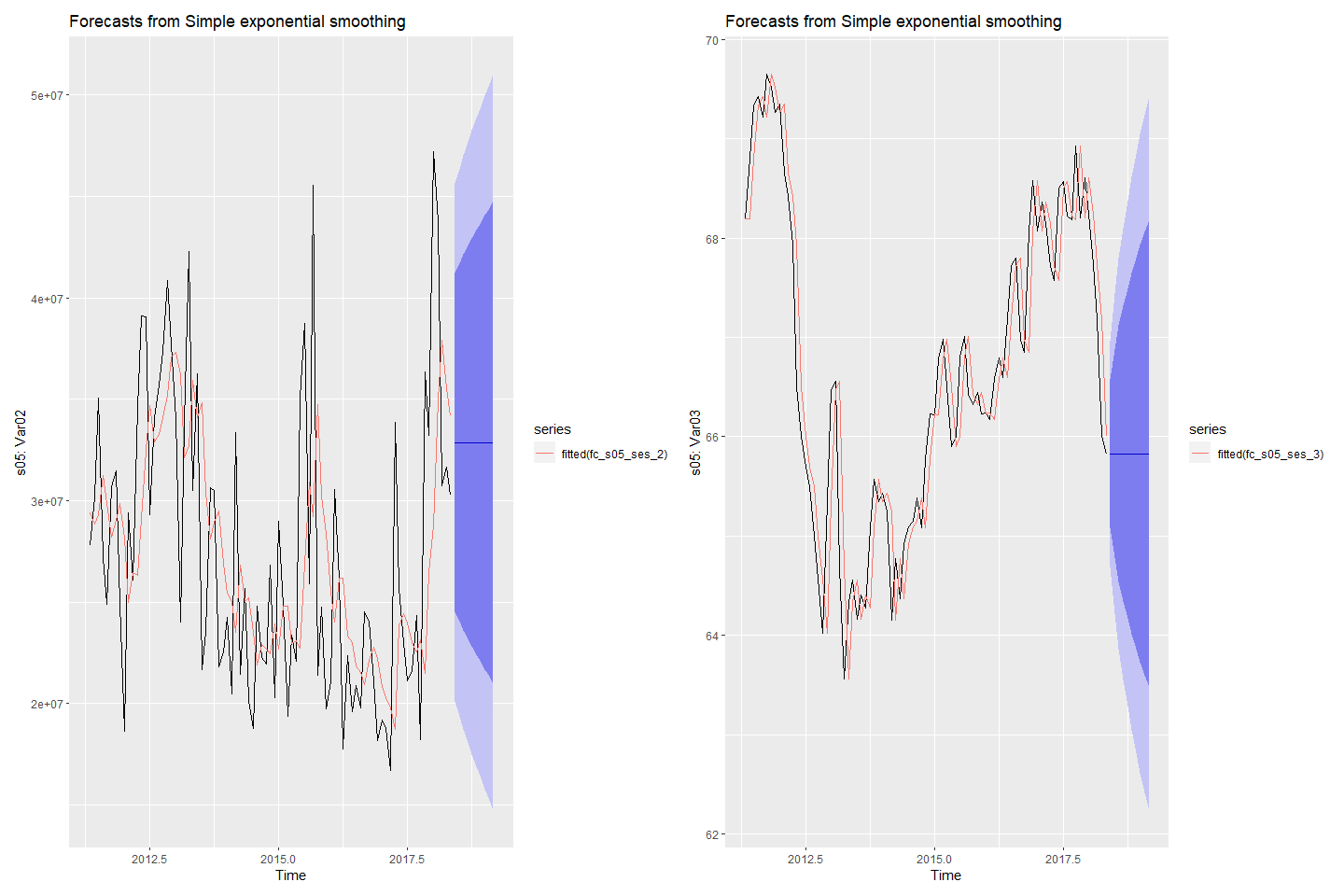
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 14.529, df = 15, p-value = 0.4859  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s05\_ses\_3)



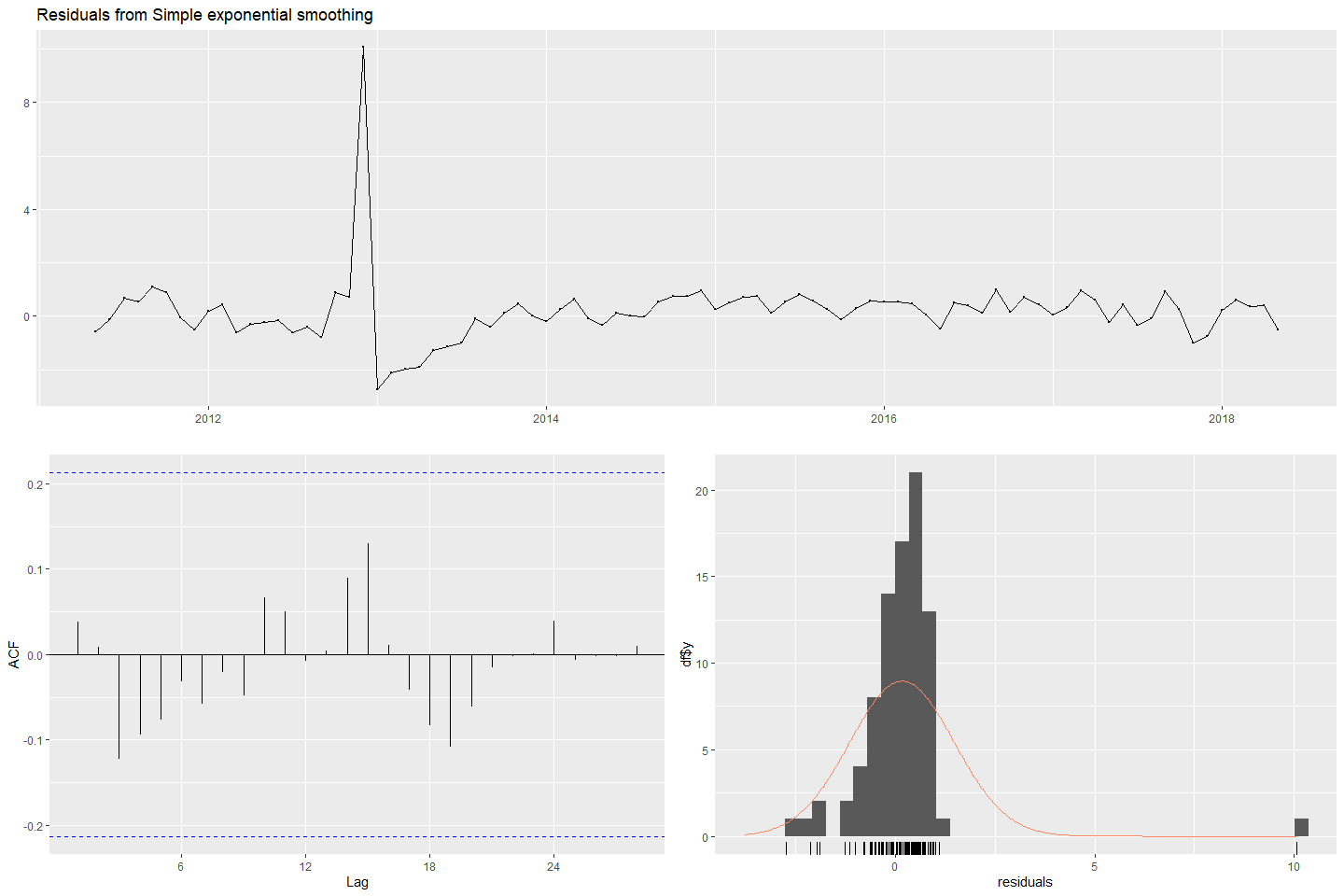
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 16.057, df = 15, p-value = 0.3783  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s05\_ses\_2 <- forecast(fit\_s05\_ses\_2)  
fc\_s05\_ses\_3 <- forecast(fit\_s05\_ses\_3)  
# plot forecasts  
fses\_s05\_2 <- autoplot(fc\_s05\_ses\_2) + ylab("s05: Var02") +  
 autolayer(fitted(fc\_s05\_ses\_2))  
fses\_s05\_3 <- autoplot(fc\_s05\_ses\_3) + ylab("s05: Var03") +  
 autolayer(fitted(fc\_s05\_ses\_3))  
(fses\_s05\_2 + fses\_s05\_3)



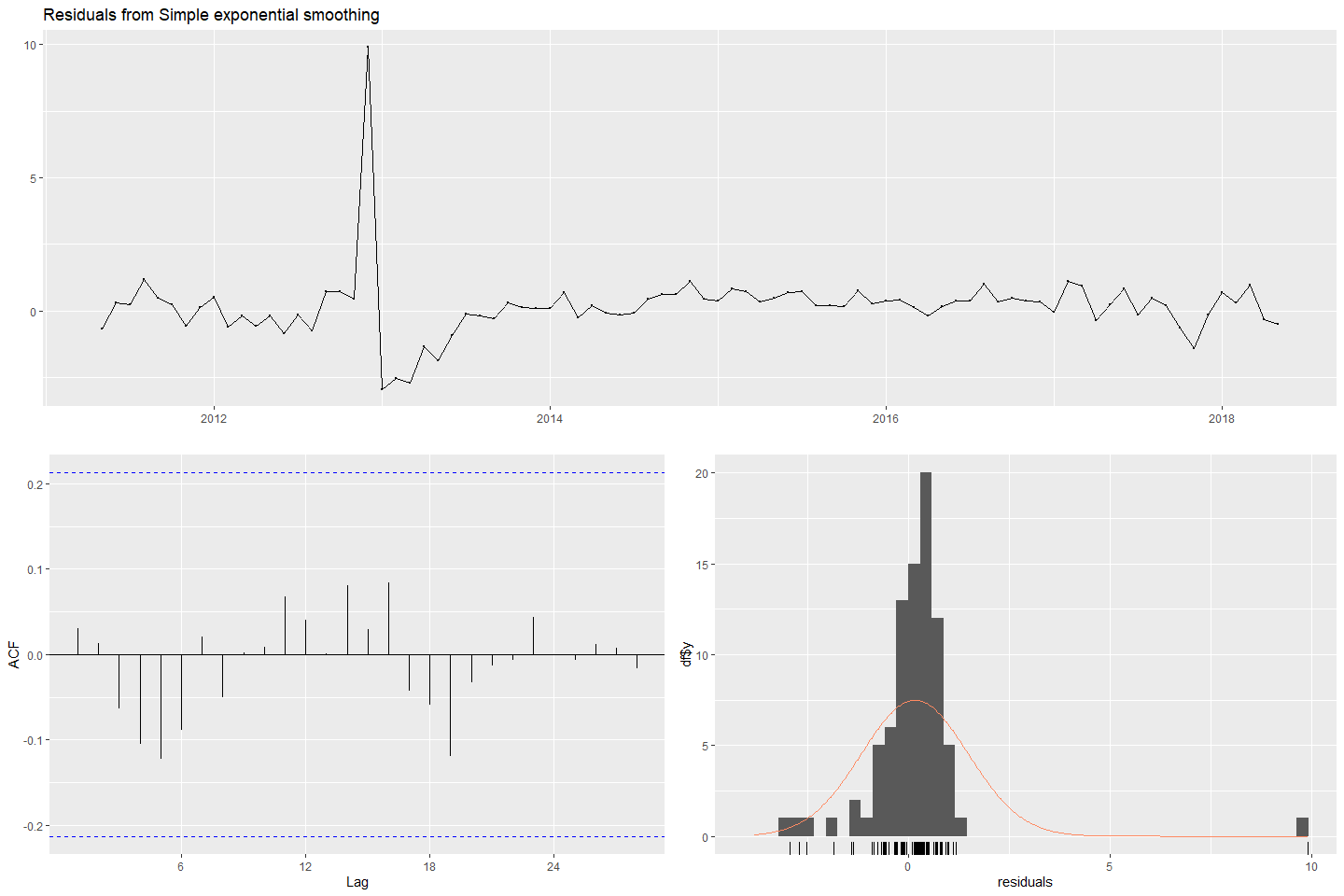
#### Forecasting S06: Var05 & Var07 with ses

# simple exponential smoothing since there is no clear trend or seasonal pattern  
# Fit ses models to Var05 & Var07  
fit\_s06\_ses\_5 <- ses(s06\_ts[,1])  
fit\_s06\_ses\_7 <- ses(s06\_ts[,2])  
# check residuals  
checkresiduals(fit\_s06\_ses\_5)



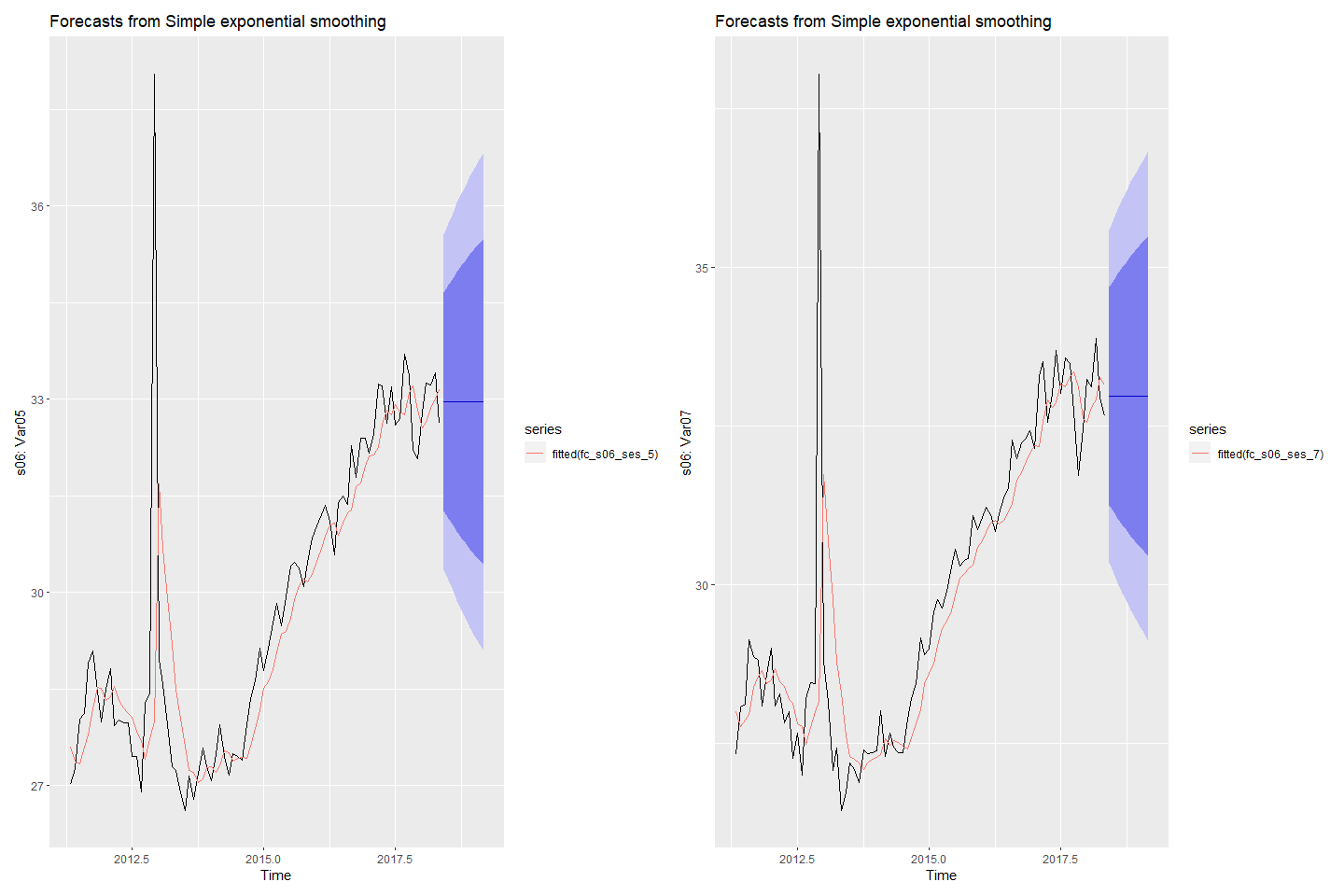
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 7.0369, df = 15, p-value = 0.9566  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s06\_ses\_7)



#>   
#> Ljung-Box test  
#>   
#> data: Residuals from Simple exponential smoothing  
#> Q\* = 6.1976, df = 15, p-value = 0.9762  
#>   
#> Model df: 2. Total lags used: 17

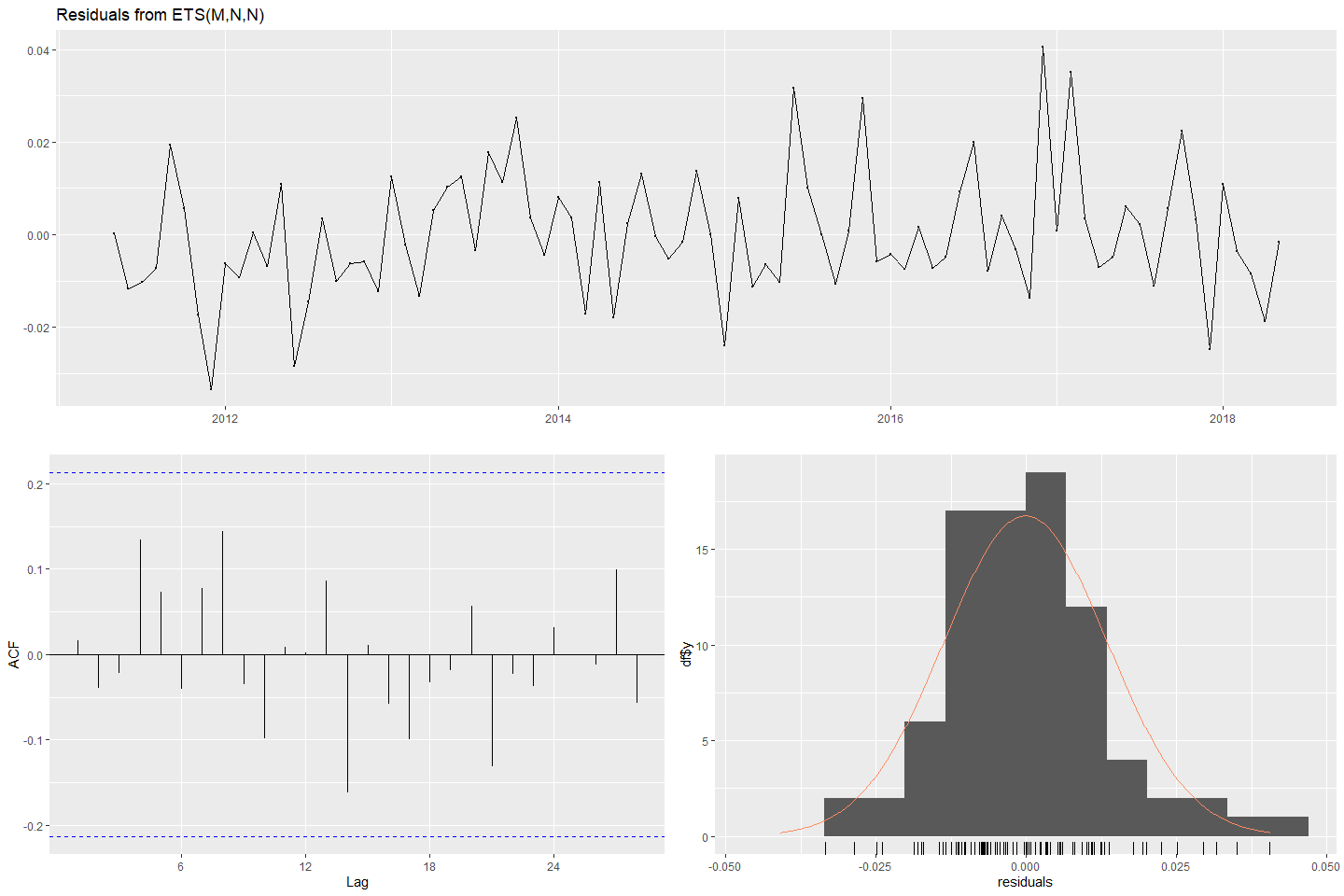
# forecast  
fc\_s06\_ses\_5 <- forecast(fit\_s06\_ses\_5)  
fc\_s06\_ses\_7 <- forecast(fit\_s06\_ses\_7)  
# plot forecasts  
fses\_s06\_5 <- autoplot(fc\_s06\_ses\_5) + ylab("s06: Var05") +  
 autolayer(fitted(fc\_s06\_ses\_5))  
fses\_s06\_7 <- autoplot(fc\_s06\_ses\_7) + ylab("s06: Var07") +  
 autolayer(fitted(fc\_s06\_ses\_7))  
(fses\_s06\_5 + fses\_s06\_7)



### Forecasting using ETS models

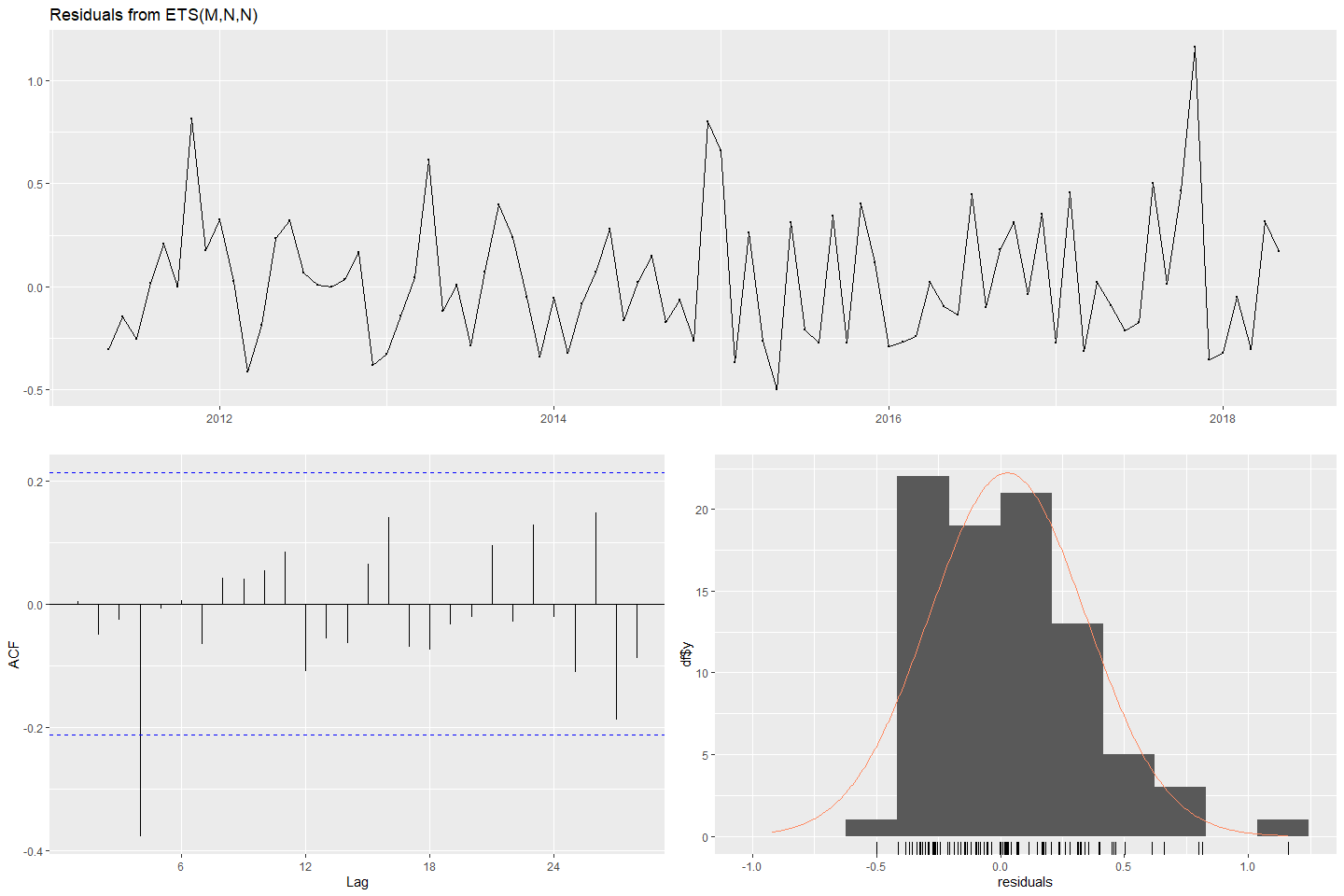
#### Forecasting S01: Var01 & Var02 with ETS

# Fit ETS models to Var01 & Var02  
fit\_s01\_ETS\_1 <- ets(s01\_ts[,1])  
fit\_s01\_ETS\_2 <- ets(s01\_ts[,2])  
# check residuals  
checkresiduals(fit\_s01\_ETS\_1)



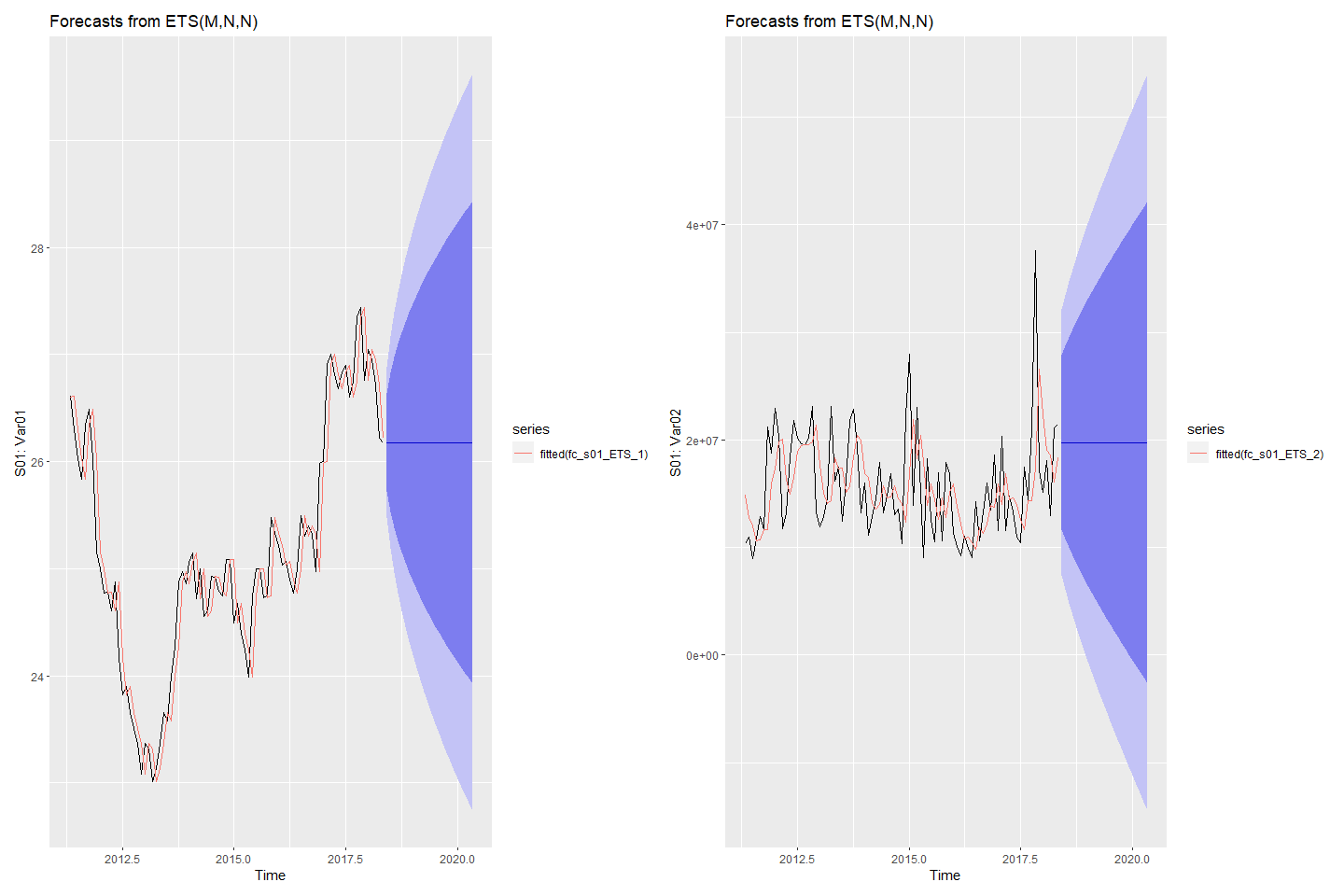
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(M,N,N)  
#> Q\* = 11.083, df = 15, p-value = 0.7467  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s01\_ETS\_2)



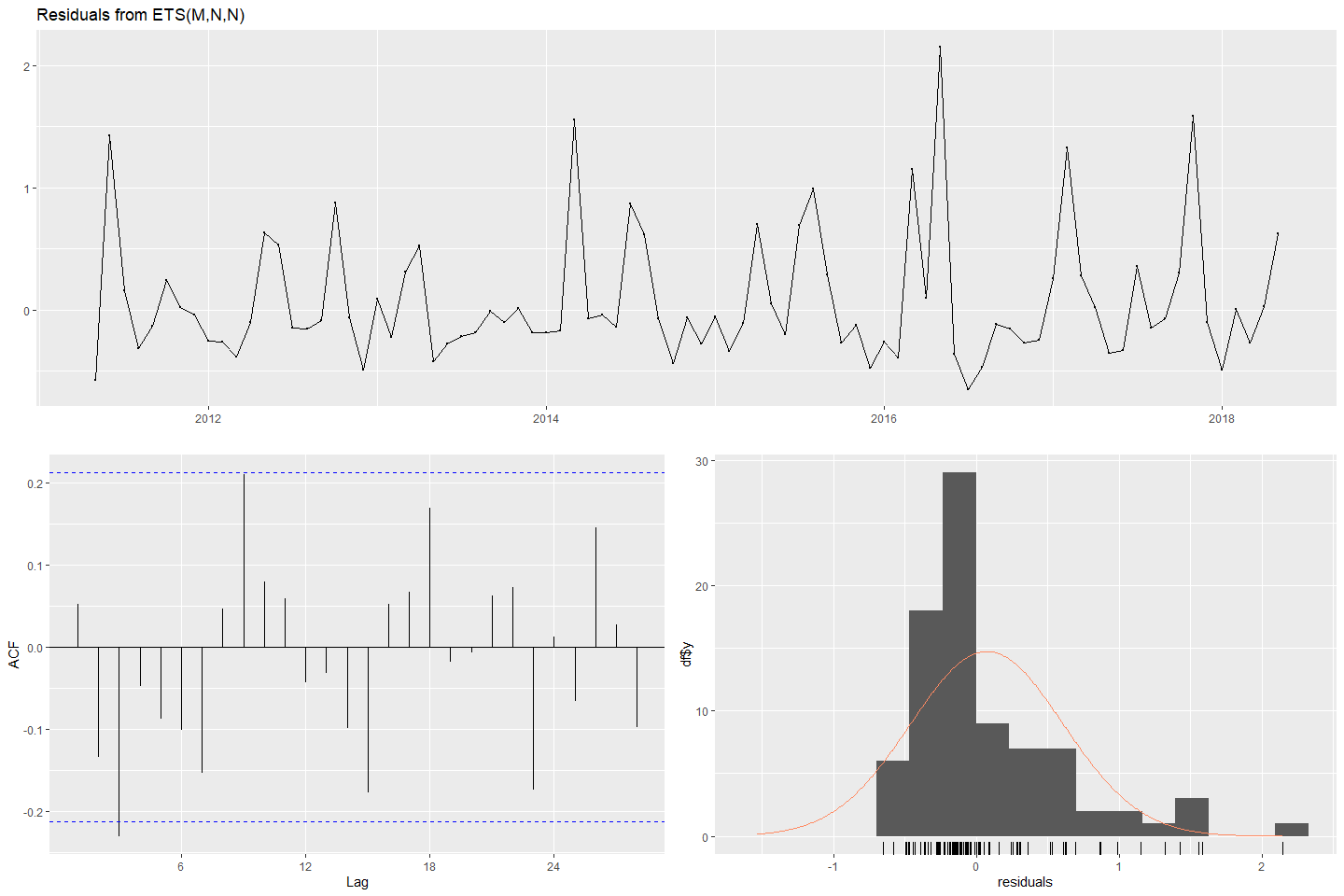
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(M,N,N)  
#> Q\* = 20.026, df = 15, p-value = 0.171  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s01\_ETS\_1 <- forecast(fit\_s01\_ETS\_1)  
fc\_s01\_ETS\_2 <- forecast(fit\_s01\_ETS\_2)  
# plot forecasts  
fets\_S01\_1 <- autoplot(fc\_s01\_ETS\_1) + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s01\_ETS\_1))  
fets\_S01\_2 <- autoplot(fc\_s01\_ETS\_2) + ylab("S01: Var02") +  
 autolayer(fitted(fc\_s01\_ETS\_2))  
(fets\_S01\_1 + fets\_S01\_2)



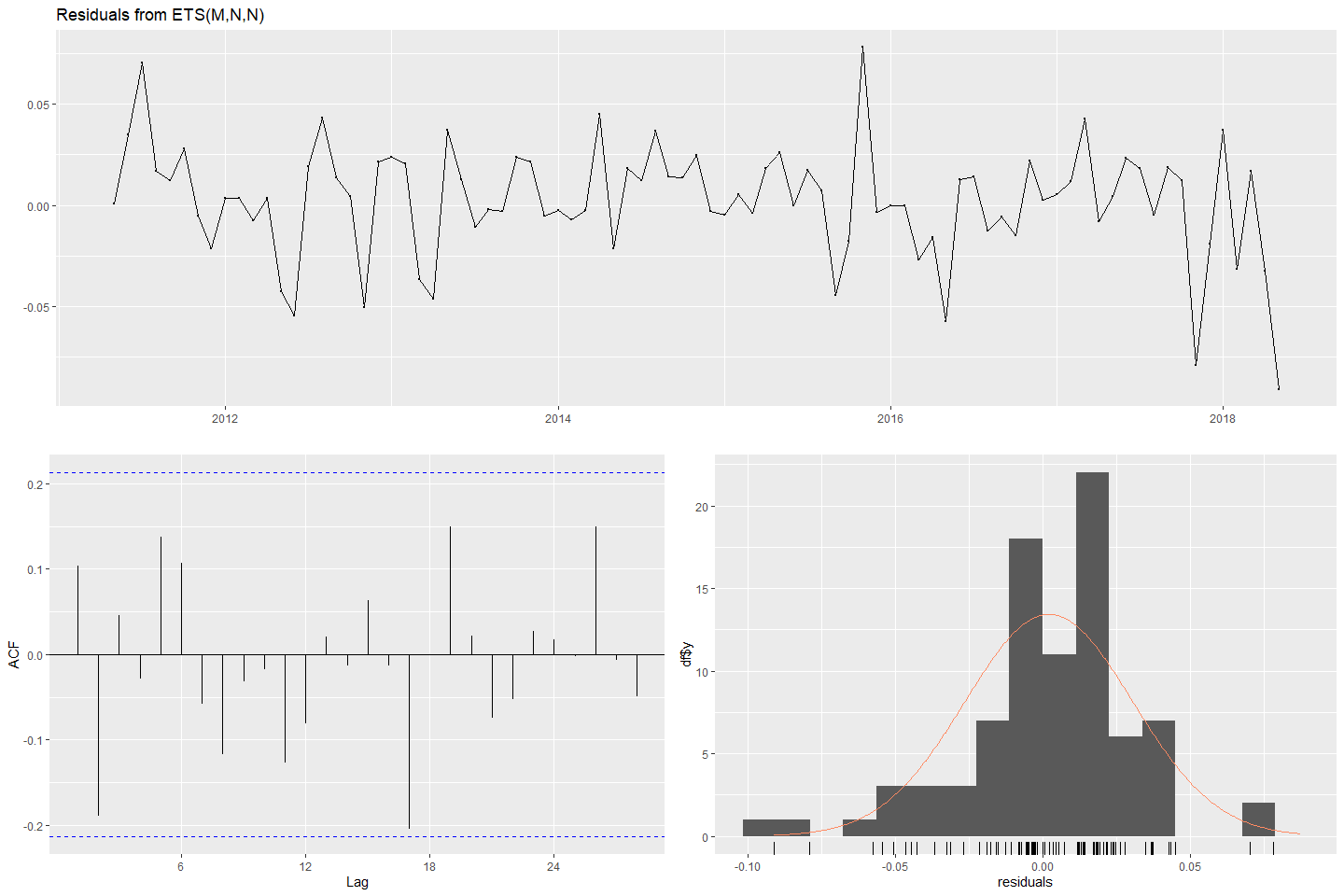
#### Forecasting S02: Var02 & Var03 with ETS

# Fit ETS models to Var02 & Var03  
fit\_s02\_ETS\_2 <- ets(s02\_ts[,1])  
fit\_s02\_ETS\_3 <- ets(s02\_ts[,2])  
# check residuals  
checkresiduals(fit\_s02\_ETS\_2)



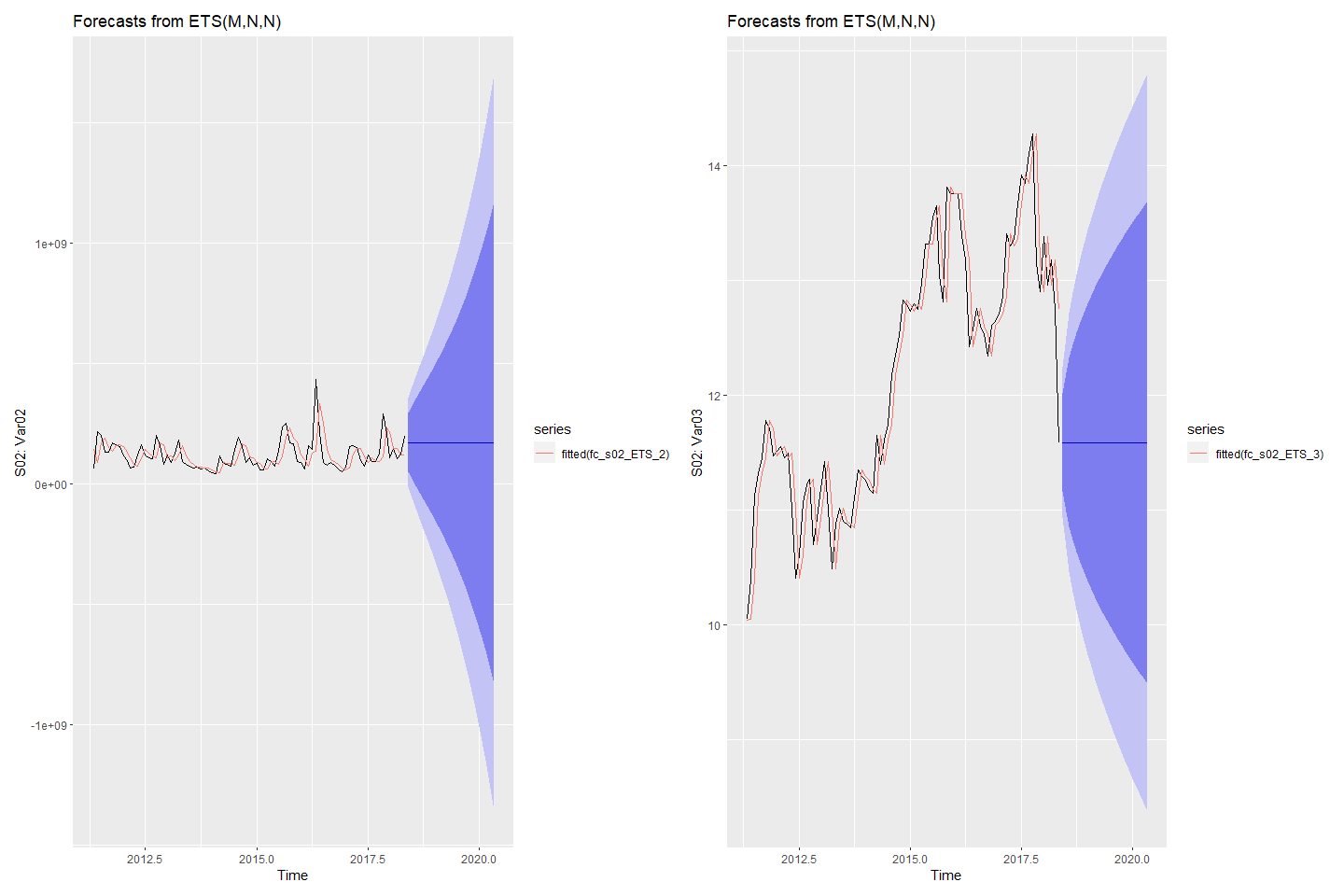
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(M,N,N)  
#> Q\* = 21.575, df = 15, p-value = 0.1194  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s02\_ETS\_3)



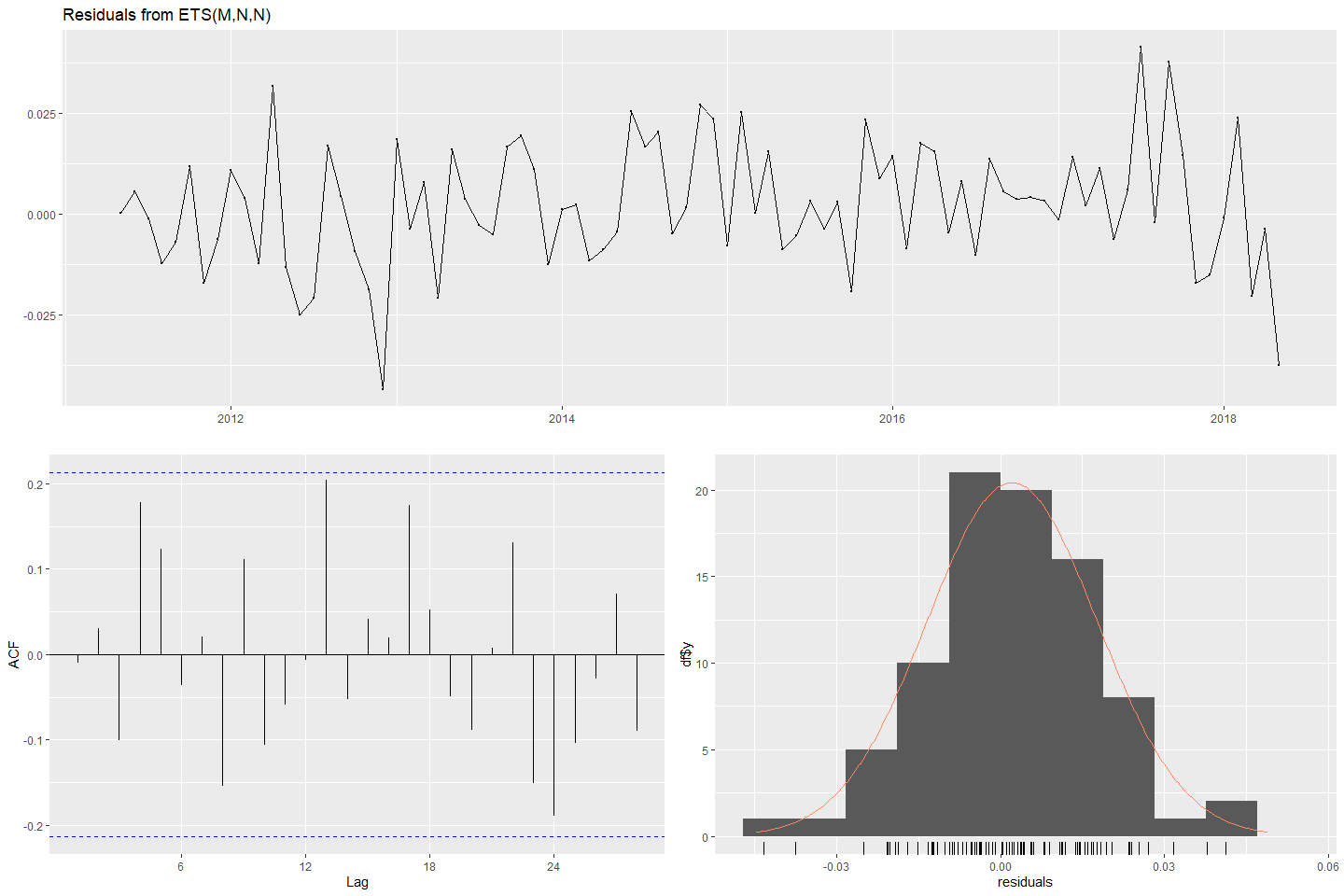
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(M,N,N)  
#> Q\* = 16.302, df = 15, p-value = 0.3623  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s02\_ETS\_2 <- forecast(fit\_s02\_ETS\_2)  
fc\_s02\_ETS\_3 <- forecast(fit\_s02\_ETS\_3)  
# plot forecasts  
fets\_S02\_2 <- autoplot(fc\_s02\_ETS\_2) + ylab("S02: Var02") +  
 autolayer(fitted(fc\_s02\_ETS\_2))  
fets\_S02\_3 <- autoplot(fc\_s02\_ETS\_3) + ylab("S02: Var03") +  
 autolayer(fitted(fc\_s02\_ETS\_3))  
(fets\_S02\_2 + fets\_S02\_3)



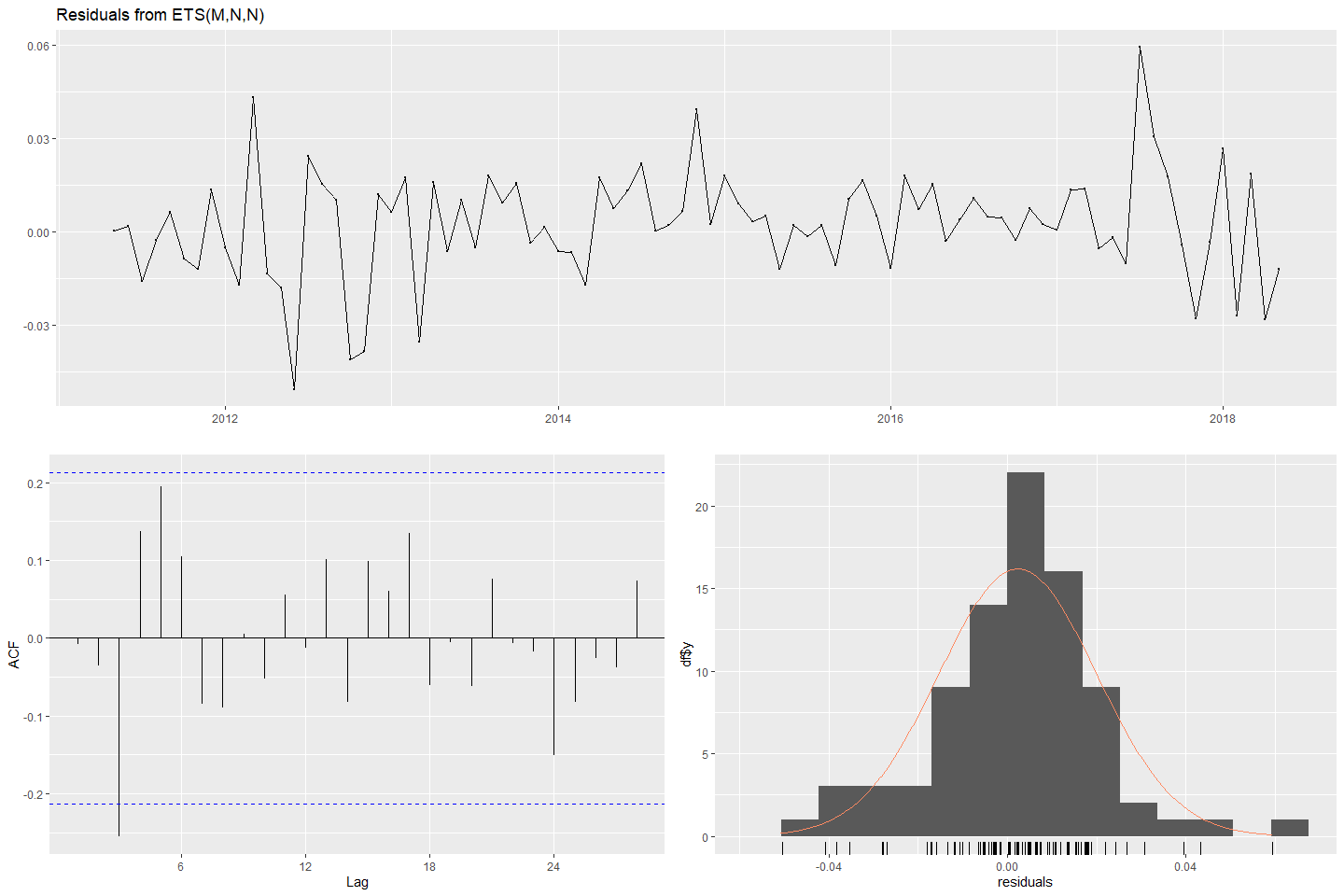
#### Forecasting S03: Var05 & Var07 with ETS

# Fit ETS models to Var05 & Var07  
fit\_s03\_ETS\_5 <- ets(s03\_ts[,1])  
fit\_s03\_ETS\_7 <- ets(s03\_ts[,2])  
# check residuals  
checkresiduals(fit\_s03\_ETS\_5)



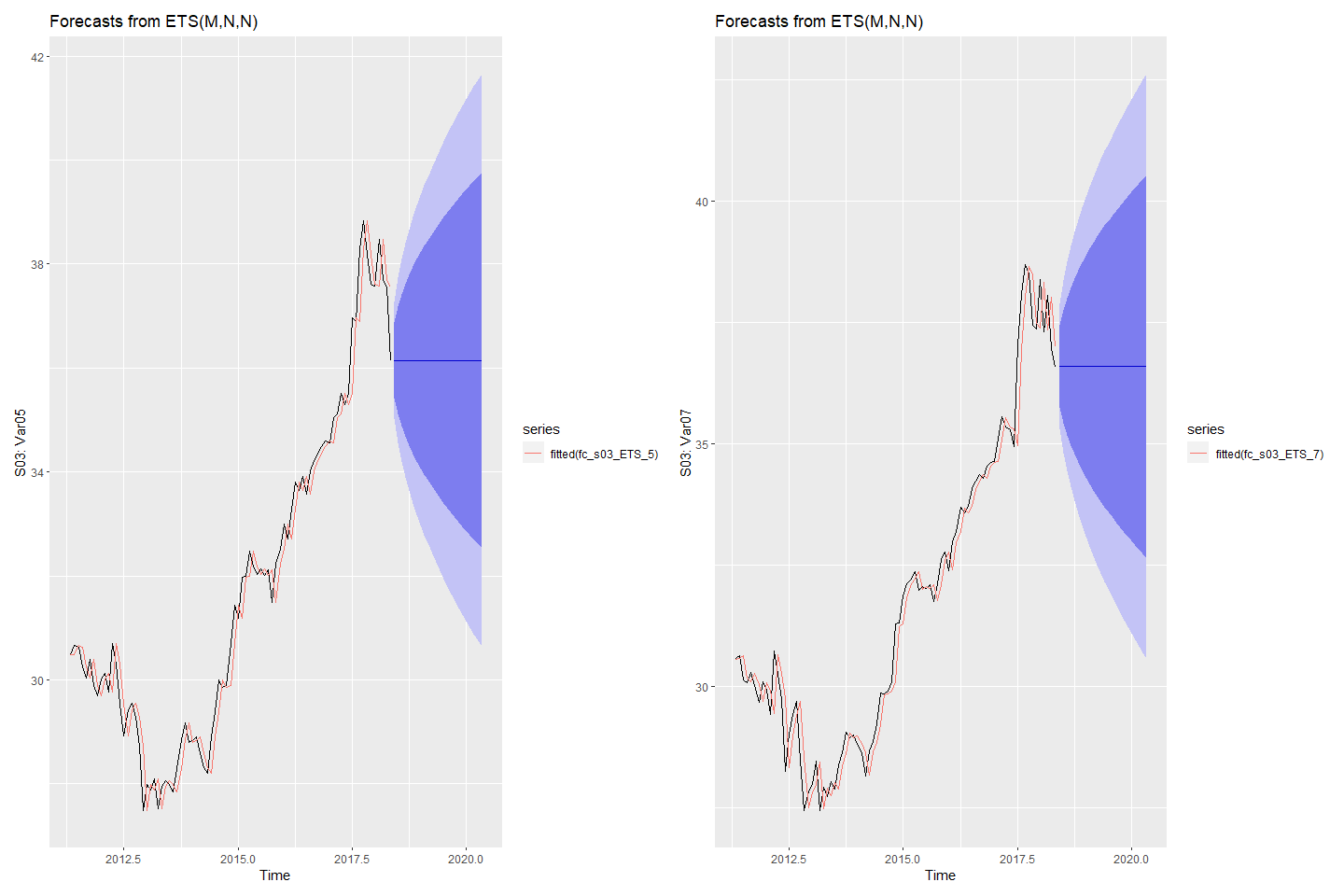
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(M,N,N)  
#> Q\* = 18.497, df = 15, p-value = 0.2375  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s03\_ETS\_7)



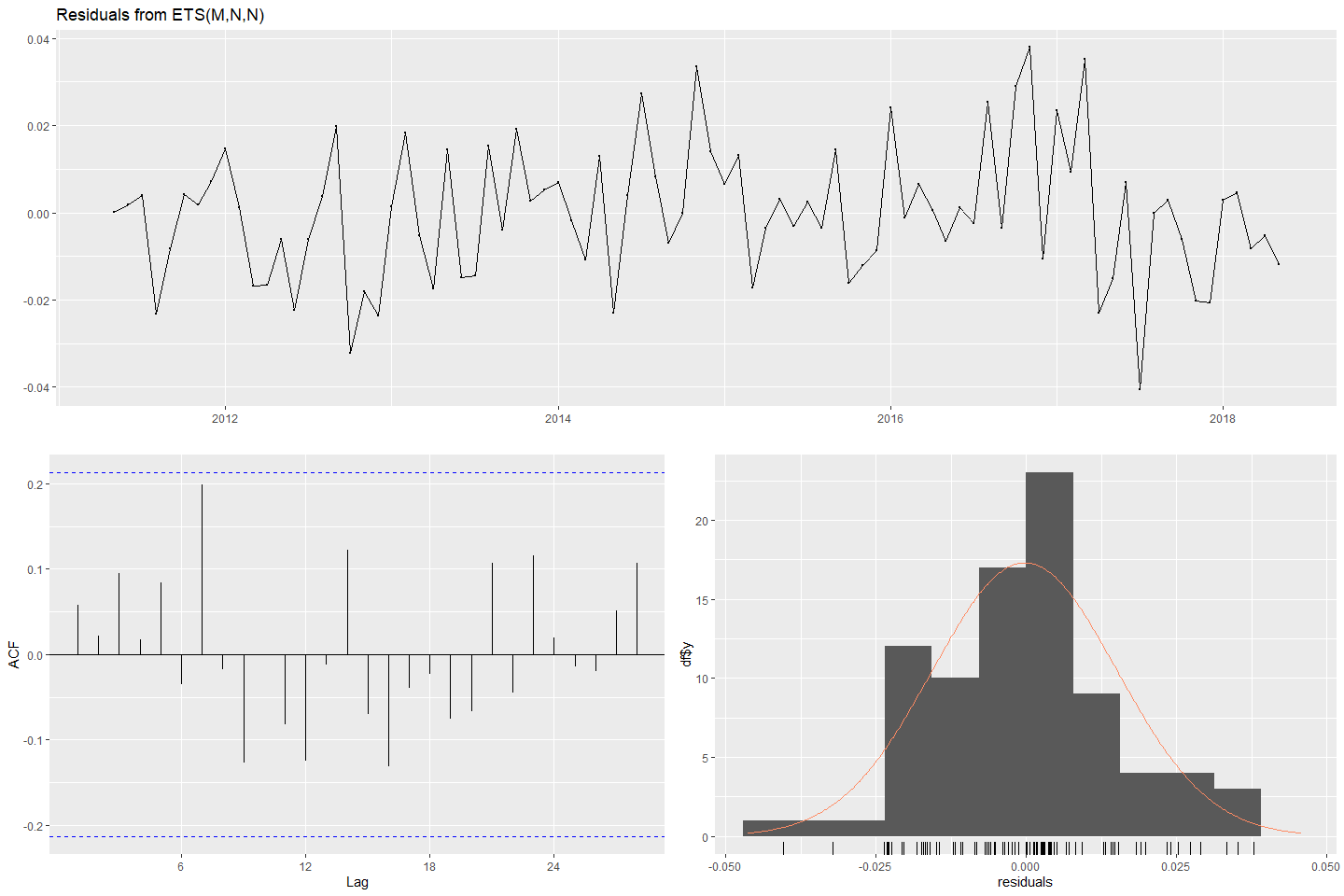
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(M,N,N)  
#> Q\* = 19.426, df = 15, p-value = 0.1951  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s03\_ETS\_5 <- forecast(fit\_s03\_ETS\_5)  
fc\_s03\_ETS\_7 <- forecast(fit\_s03\_ETS\_7)  
# plot forecasts  
fets\_S03\_5 <- autoplot(fc\_s03\_ETS\_5) + ylab("S03: Var05") +  
 autolayer(fitted(fc\_s03\_ETS\_5))  
fets\_S03\_7 <- autoplot(fc\_s03\_ETS\_7) + ylab("S03: Var07") +  
 autolayer(fitted(fc\_s03\_ETS\_7))  
(fets\_S03\_5 + fets\_S03\_7)



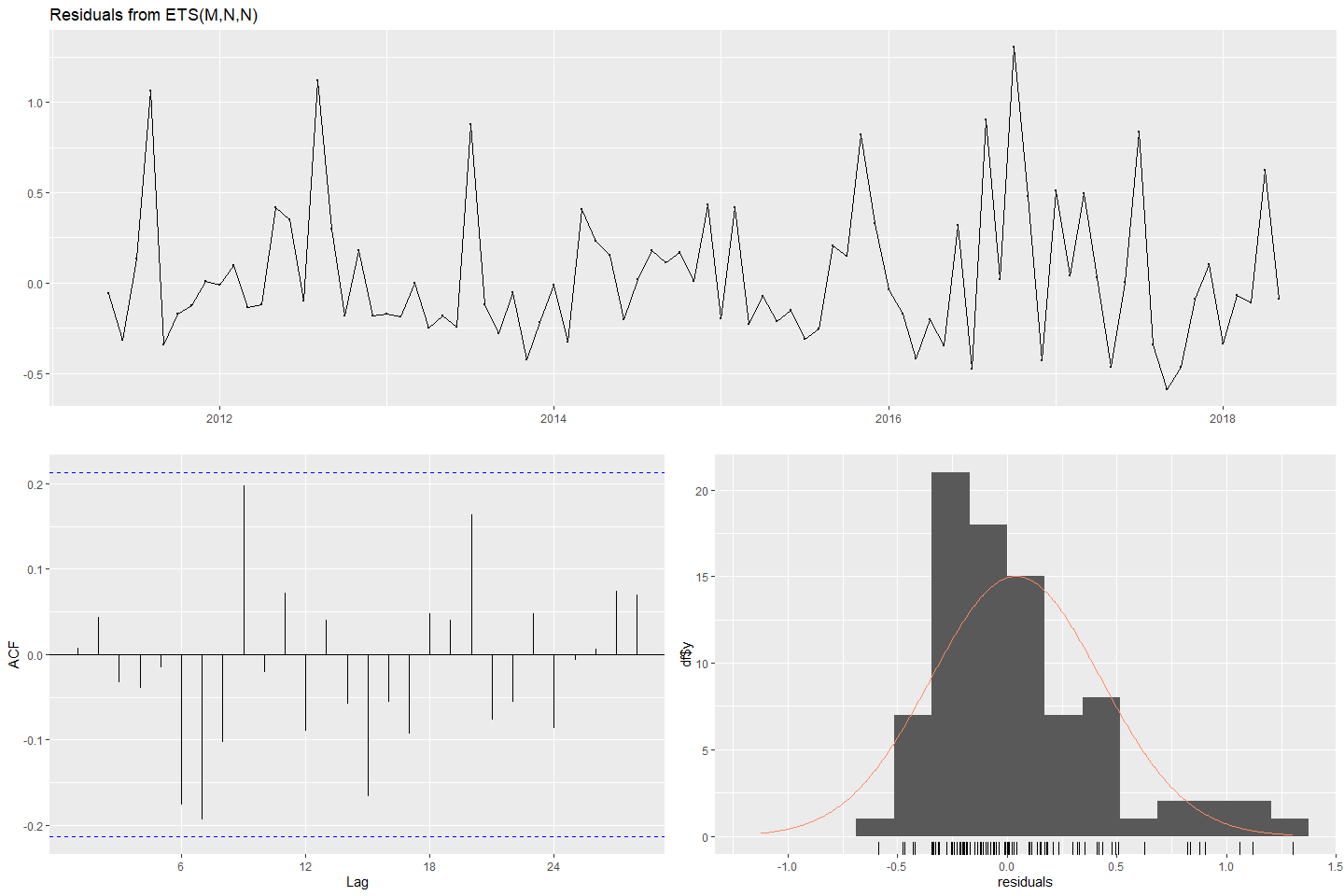
#### Forecasting S04: Var01 & Var02 with ETS

# Fit ETS models to Var01 & Var02  
fit\_s04\_ETS\_1 <- ets(s04\_ts[,1])  
fit\_s04\_ETS\_2 <- ets(s04\_ts[,2])  
# check residuals  
checkresiduals(fit\_s04\_ETS\_1)



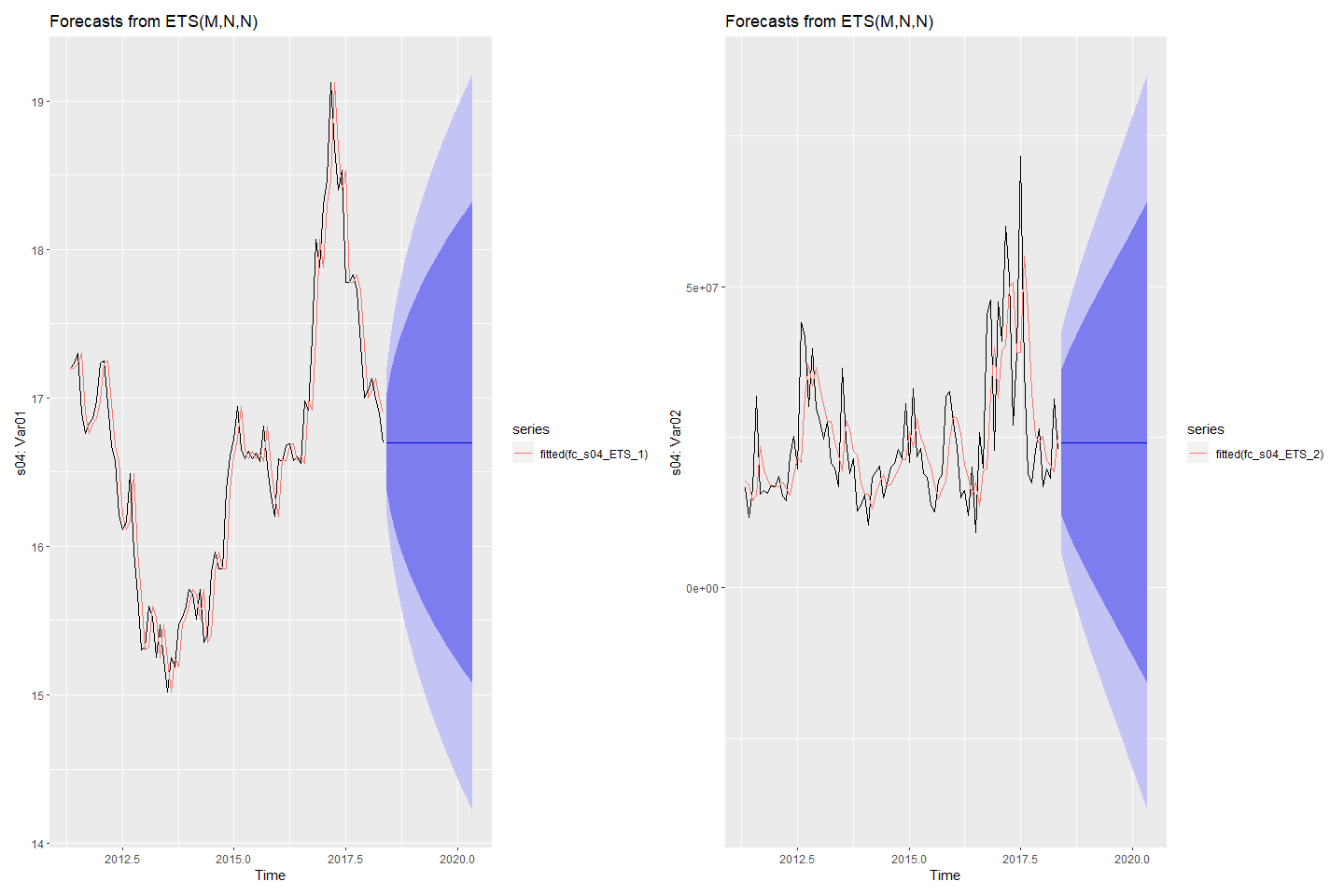
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(M,N,N)  
#> Q\* = 13.643, df = 15, p-value = 0.5527  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s04\_ETS\_2)



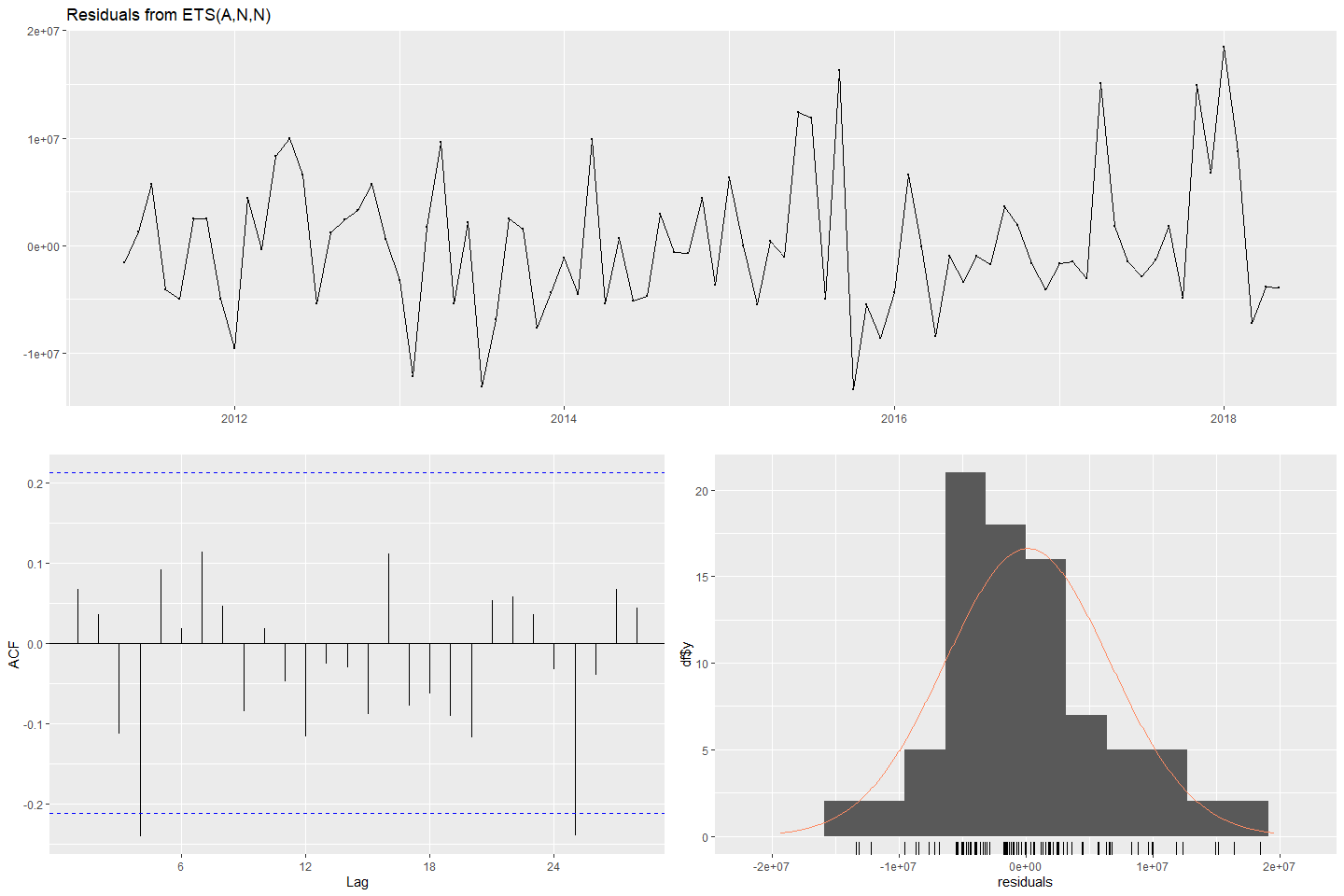
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(M,N,N)  
#> Q\* = 17.795, df = 15, p-value = 0.2736  
#>   
#> Model df: 2. Total lags used: 17

# forecast next 140 periods  
fc\_s04\_ETS\_1 <- forecast(fit\_s04\_ETS\_1)  
fc\_s04\_ETS\_2 <- forecast(fit\_s04\_ETS\_2)  
# plot forecasts  
fets\_s04\_1 <- autoplot(fc\_s04\_ETS\_1) + ylab("s04: Var01") +  
 autolayer(fitted(fc\_s04\_ETS\_1))  
fets\_s04\_2 <- autoplot(fc\_s04\_ETS\_2) + ylab("s04: Var02") +  
 autolayer(fitted(fc\_s04\_ETS\_2))  
(fets\_s04\_1 + fets\_s04\_2)



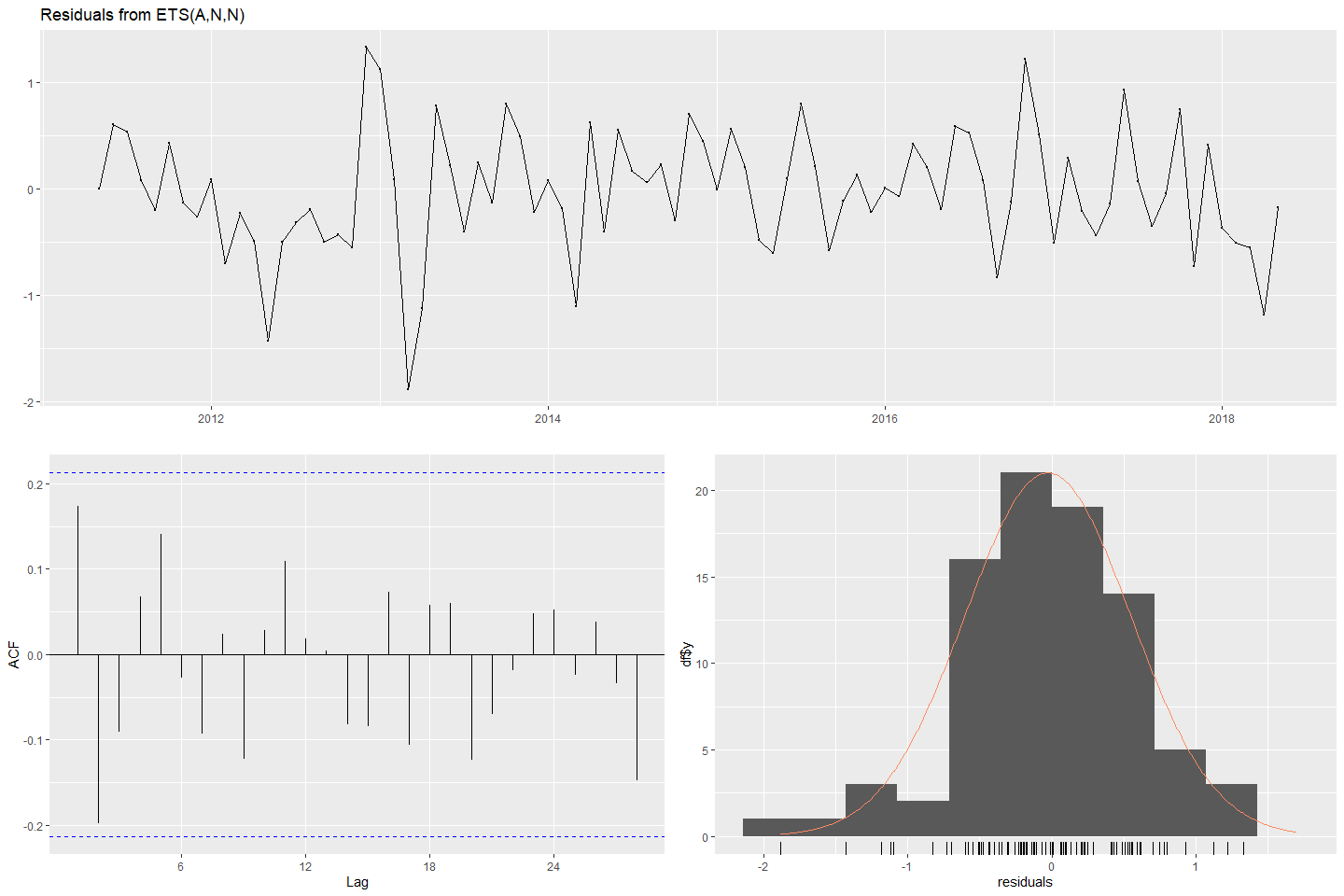
#### Forecasting S05: Var02 & Var03 with ETS

# Fit ETS models to Var02 & Var03  
fit\_s05\_ETS\_2 <- ets(s05\_ts[,1])  
fit\_s05\_ETS\_3 <- ets(s05\_ts[,2])  
# check residuals  
checkresiduals(fit\_s05\_ETS\_2)



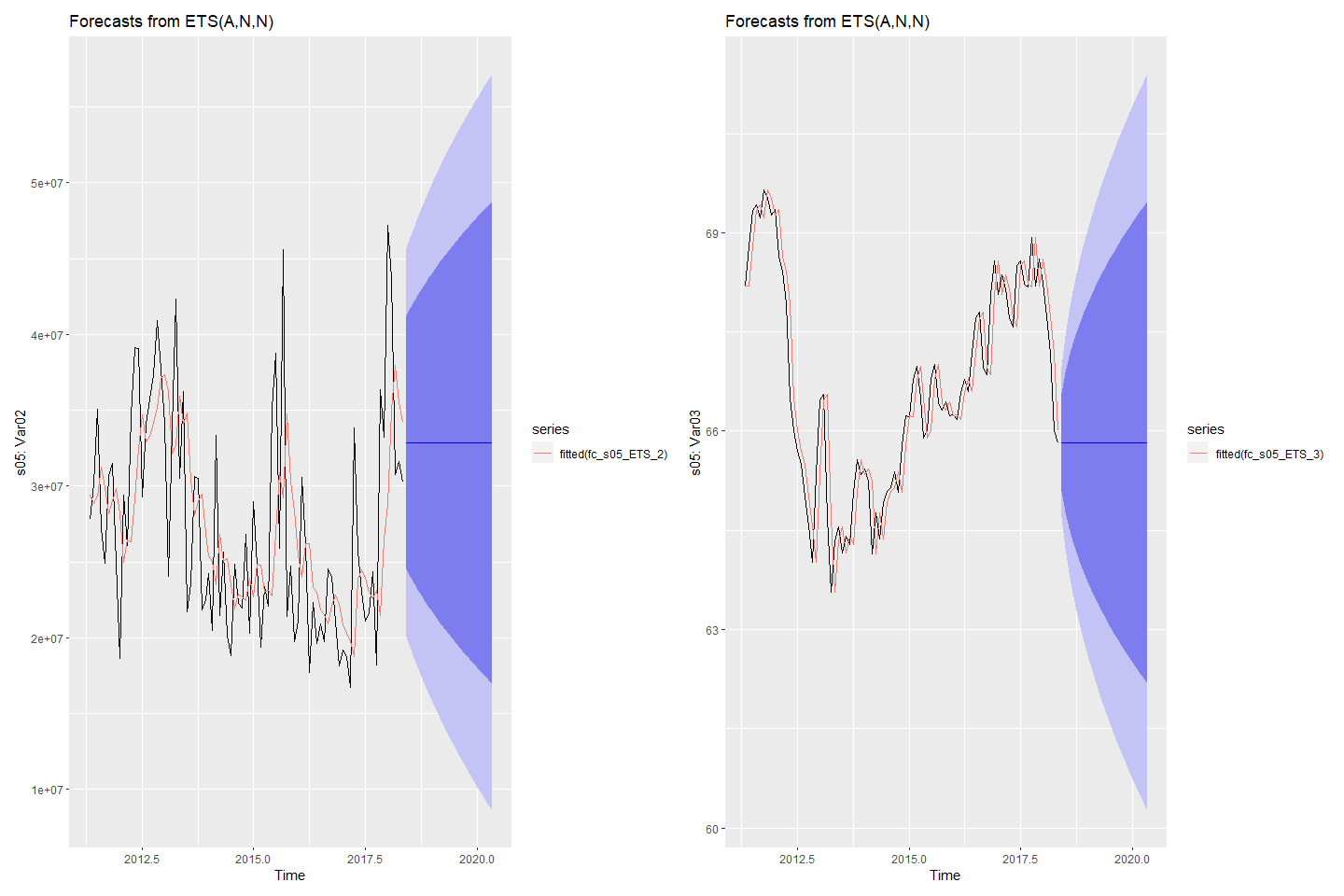
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(A,N,N)  
#> Q\* = 14.531, df = 15, p-value = 0.4857  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s05\_ETS\_3)



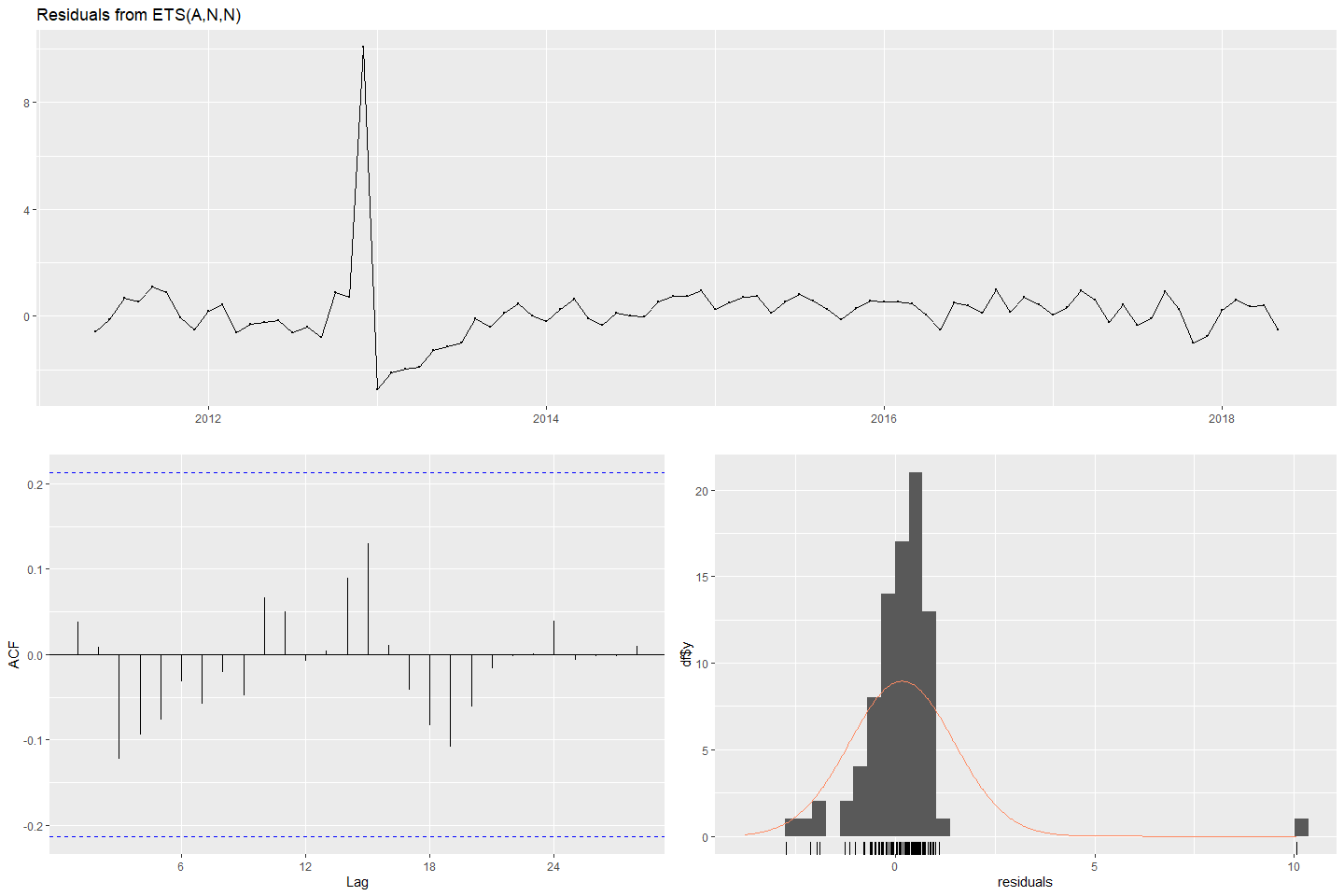
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(A,N,N)  
#> Q\* = 16.057, df = 15, p-value = 0.3782  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s05\_ETS\_2 <- forecast(fit\_s05\_ETS\_2)  
fc\_s05\_ETS\_3 <- forecast(fit\_s05\_ETS\_3)  
# plot forecasts  
fets\_s05\_2 <- autoplot(fc\_s05\_ETS\_2) + ylab("s05: Var02") +  
 autolayer(fitted(fc\_s05\_ETS\_2))  
fets\_s05\_3 <- autoplot(fc\_s05\_ETS\_3) + ylab("s05: Var03") +  
 autolayer(fitted(fc\_s05\_ETS\_3))  
(fets\_s05\_2 + fets\_s05\_3)



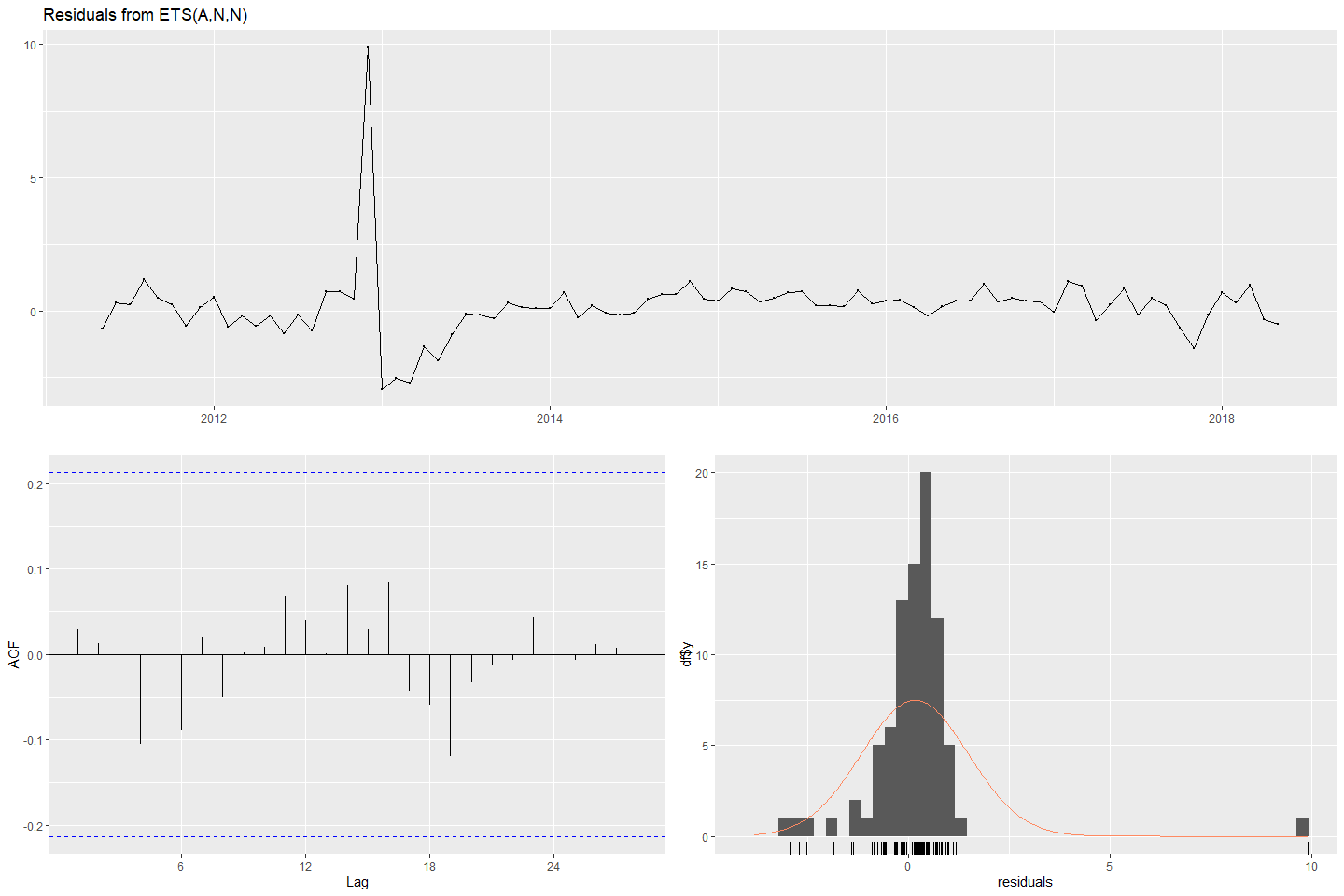
#### Forecasting S06: Var05 & Var07 with ETS

# Fit ETS models to Var05 & Var07  
fit\_s06\_ETS\_5 <- ets(s06\_ts[,1])  
fit\_s06\_ETS\_7 <- ets(s06\_ts[,2])  
# check residuals  
checkresiduals(fit\_s06\_ETS\_5)



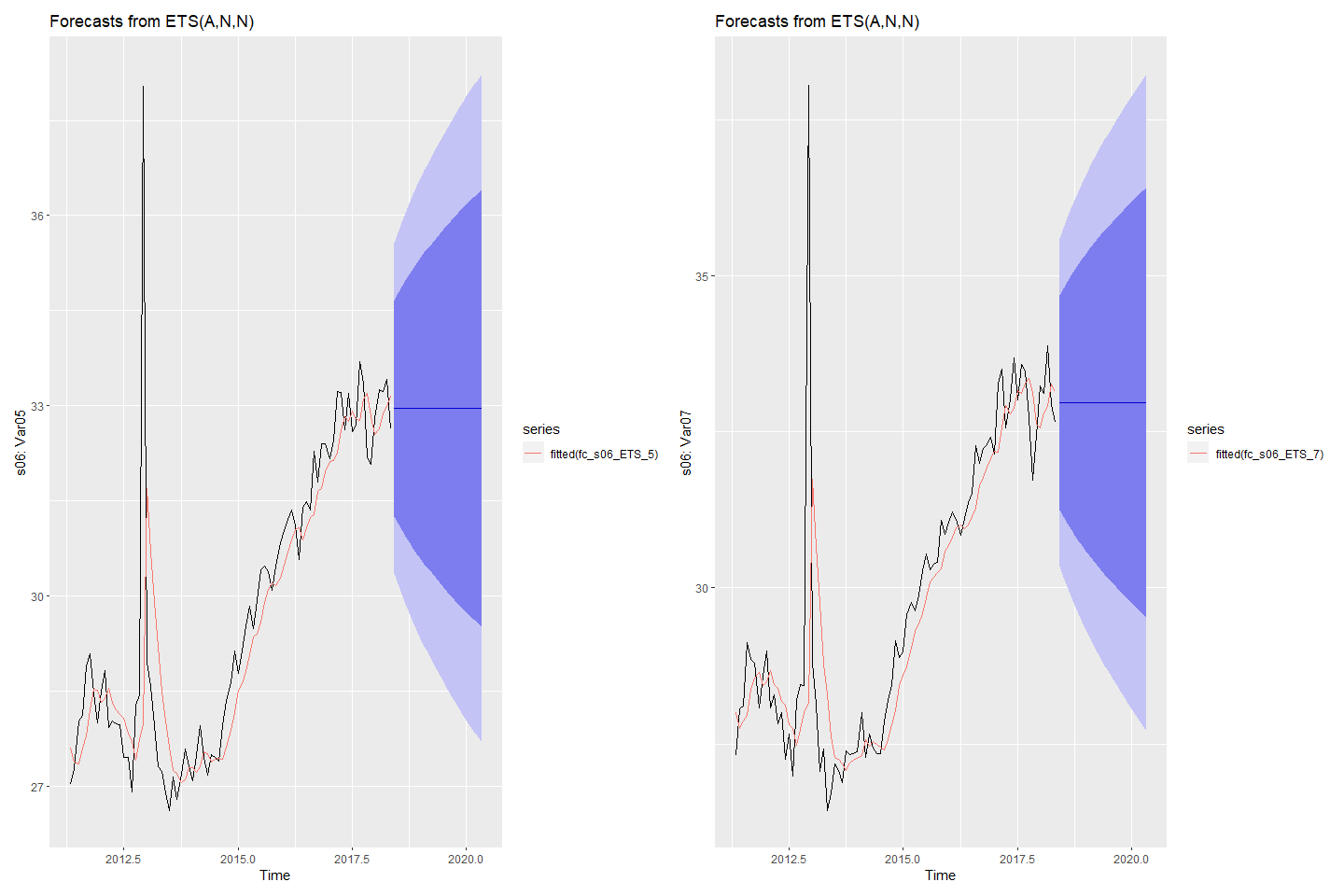
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(A,N,N)  
#> Q\* = 7.0367, df = 15, p-value = 0.9566  
#>   
#> Model df: 2. Total lags used: 17

checkresiduals(fit\_s06\_ETS\_7)



#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ETS(A,N,N)  
#> Q\* = 6.197, df = 15, p-value = 0.9762  
#>   
#> Model df: 2. Total lags used: 17

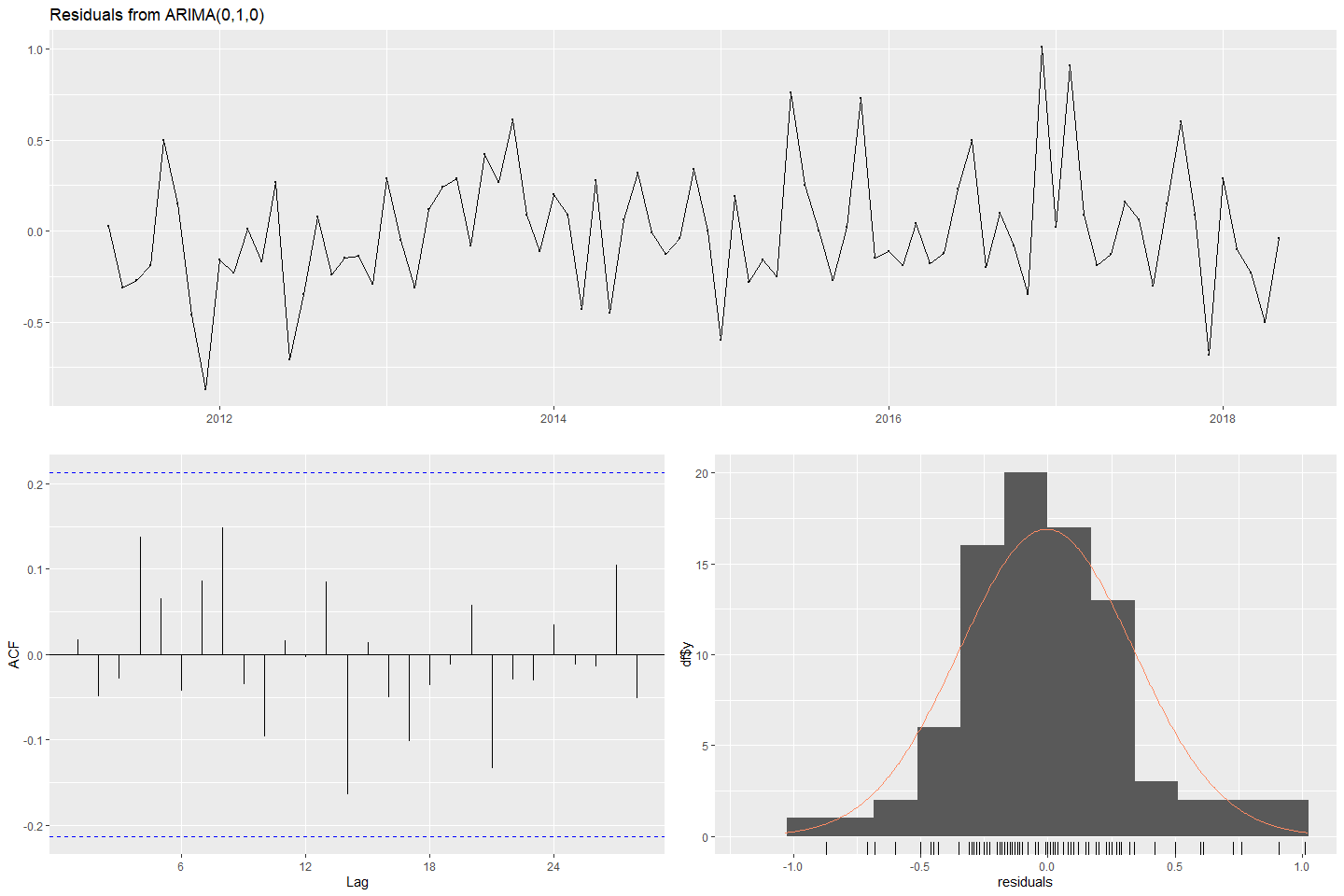
# forecast  
fc\_s06\_ETS\_5 <- forecast(fit\_s06\_ETS\_5)  
fc\_s06\_ETS\_7 <- forecast(fit\_s06\_ETS\_7)  
# plot forecasts  
fets\_s06\_5 <- autoplot(fc\_s06\_ETS\_5) + ylab("s06: Var05") +  
 autolayer(fitted(fc\_s06\_ETS\_5))  
fets\_s06\_7 <- autoplot(fc\_s06\_ETS\_7) + ylab("s06: Var07") +  
 autolayer(fitted(fc\_s06\_ETS\_7))  
(fets\_s06\_5 + fets\_s06\_7)



### Forecasting using ARIMA models

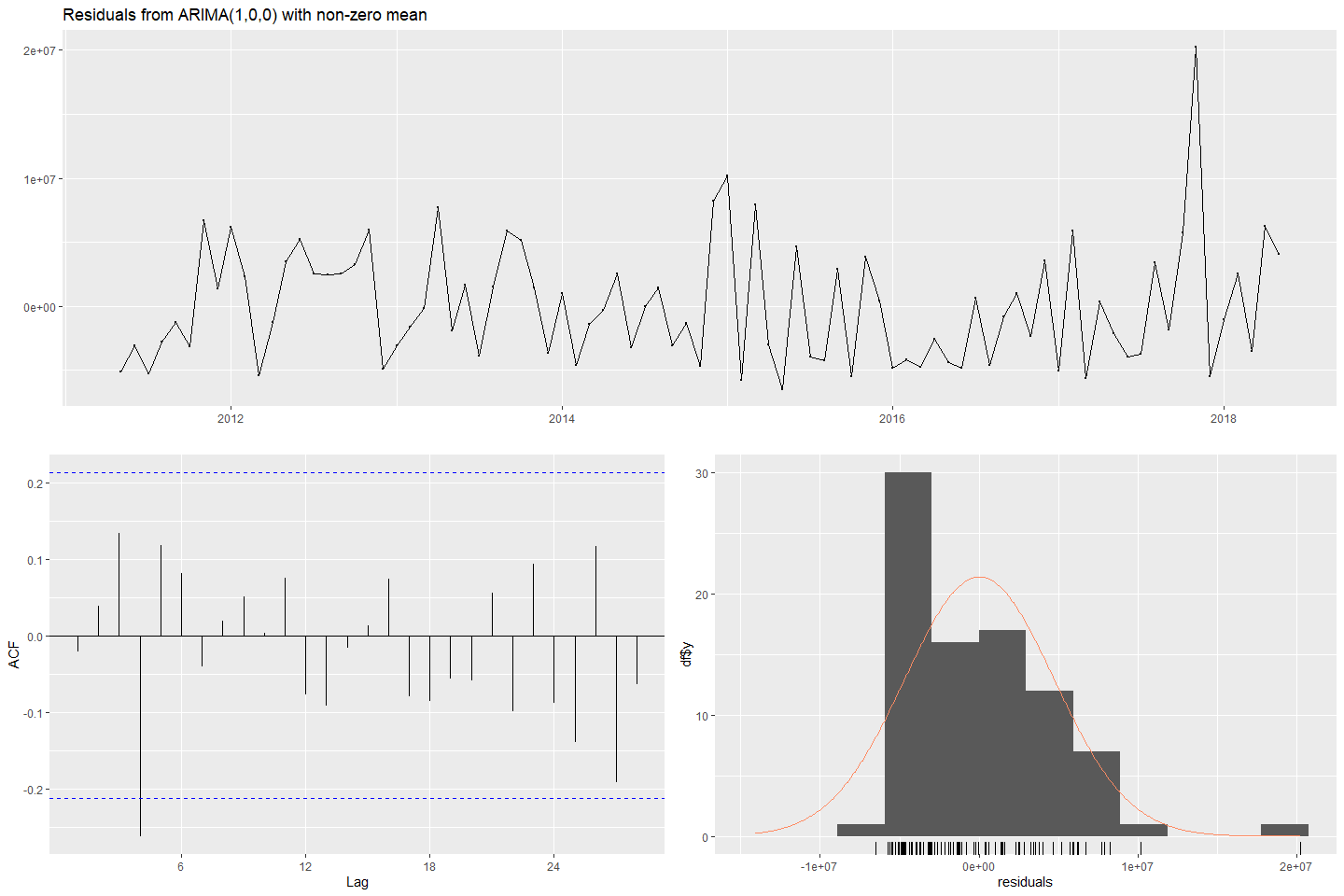
#### Forecasting S01: Var01 & Var02 with ARIMA

# Fit ARIMA models to Var01 & Var02  
fit\_s01\_ARIMA\_1 <- auto.arima(s01\_ts[,1], stepwise = TRUE)  
fit\_s01\_ARIMA\_2 <- auto.arima(s01\_ts[,2], stepwise = TRUE)  
# check residuals  
checkresiduals(fit\_s01\_ARIMA\_1)



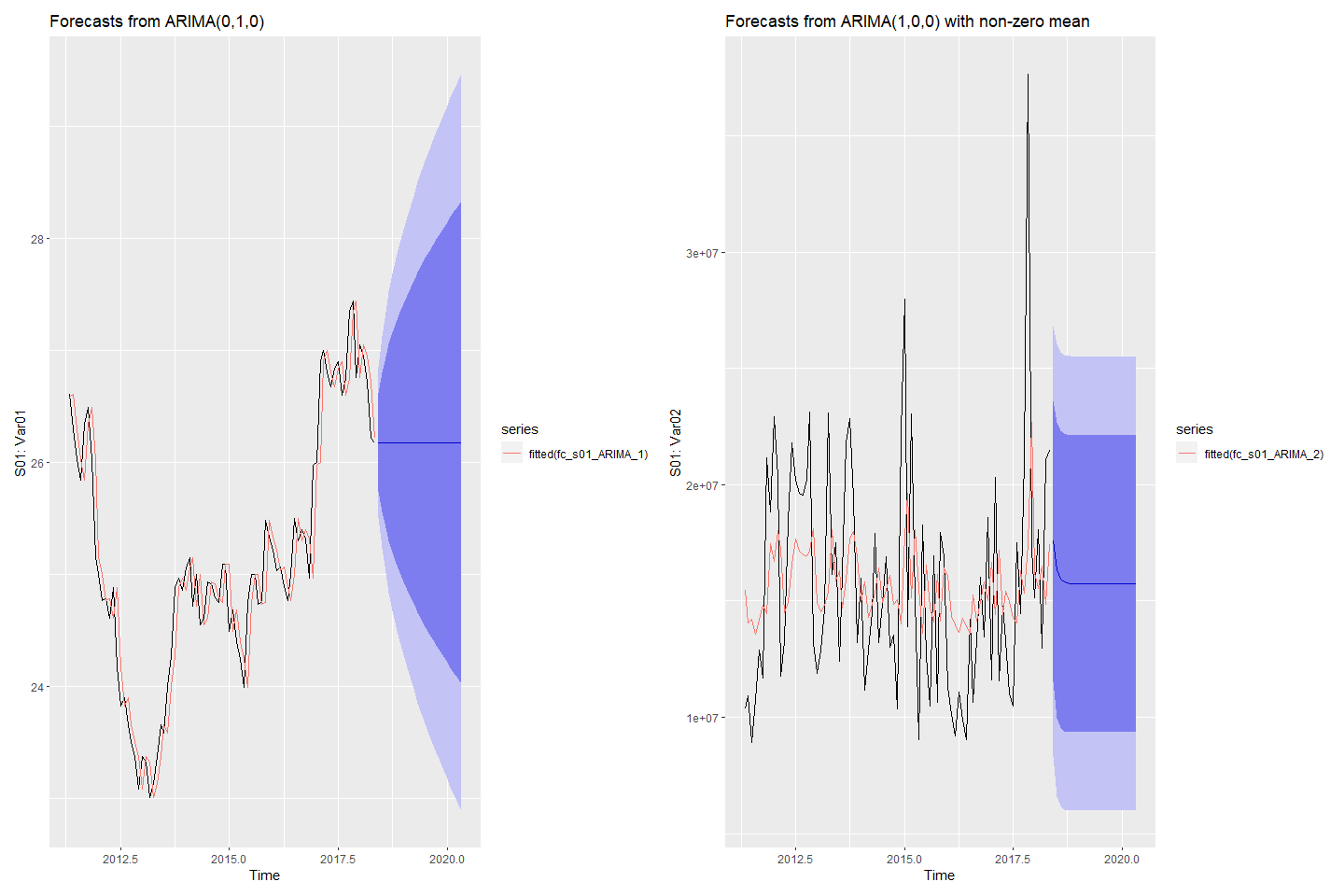
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(0,1,0)  
#> Q\* = 11.465, df = 17, p-value = 0.8314  
#>   
#> Model df: 0. Total lags used: 17

checkresiduals(fit\_s01\_ARIMA\_2)



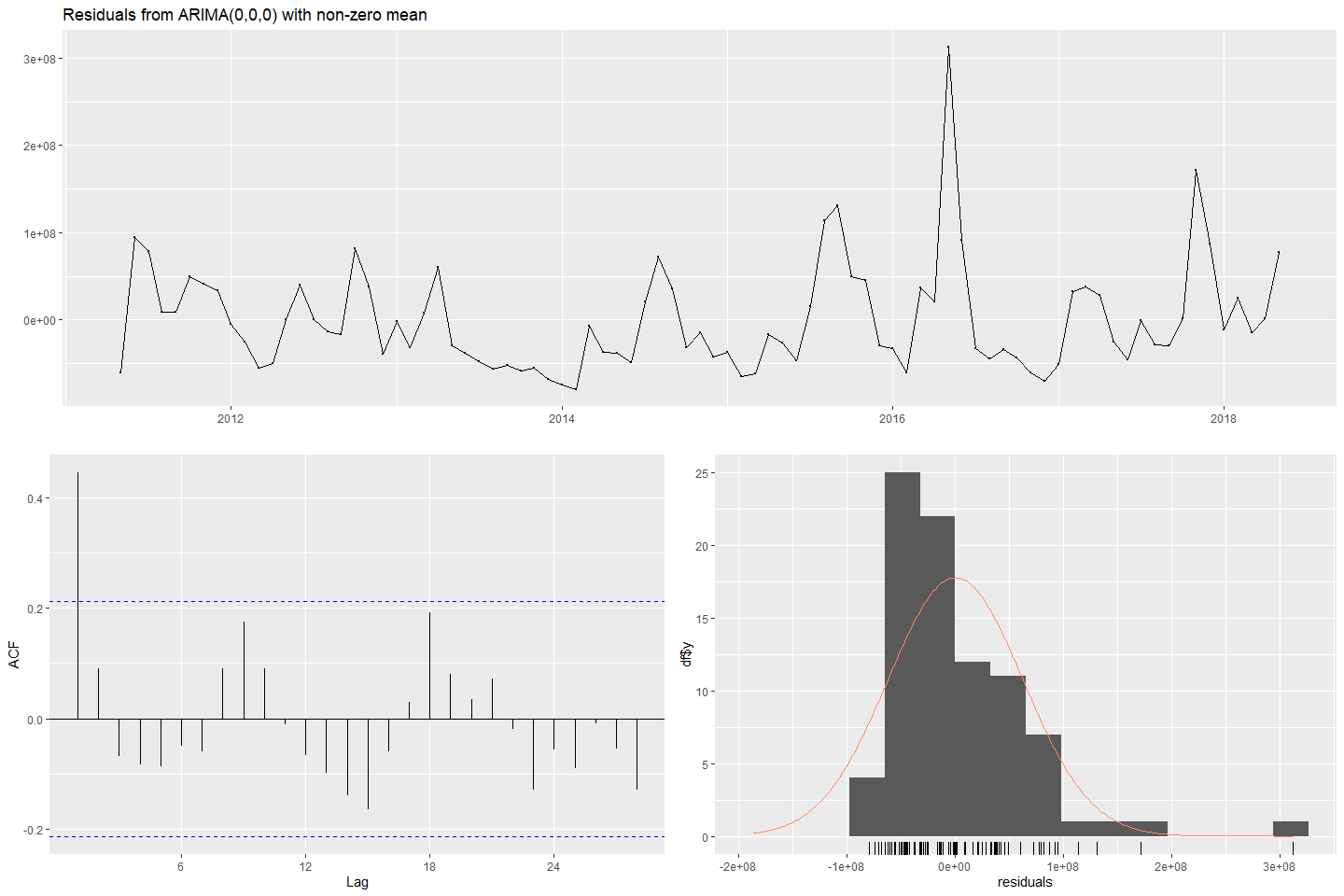
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(1,0,0) with non-zero mean  
#> Q\* = 13.722, df = 15, p-value = 0.5467  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s01\_ARIMA\_1 <- forecast(fit\_s01\_ARIMA\_1)  
fc\_s01\_ARIMA\_2 <- forecast(fit\_s01\_ARIMA\_2)  
# plot forecasts  
fa\_S01\_1 <- autoplot(fc\_s01\_ARIMA\_1) + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s01\_ARIMA\_1))  
fa\_S01\_2 <- autoplot(fc\_s01\_ARIMA\_2) + ylab("S01: Var02") +  
 autolayer(fitted(fc\_s01\_ARIMA\_2))  
(fa\_S01\_1 + fa\_S01\_2)



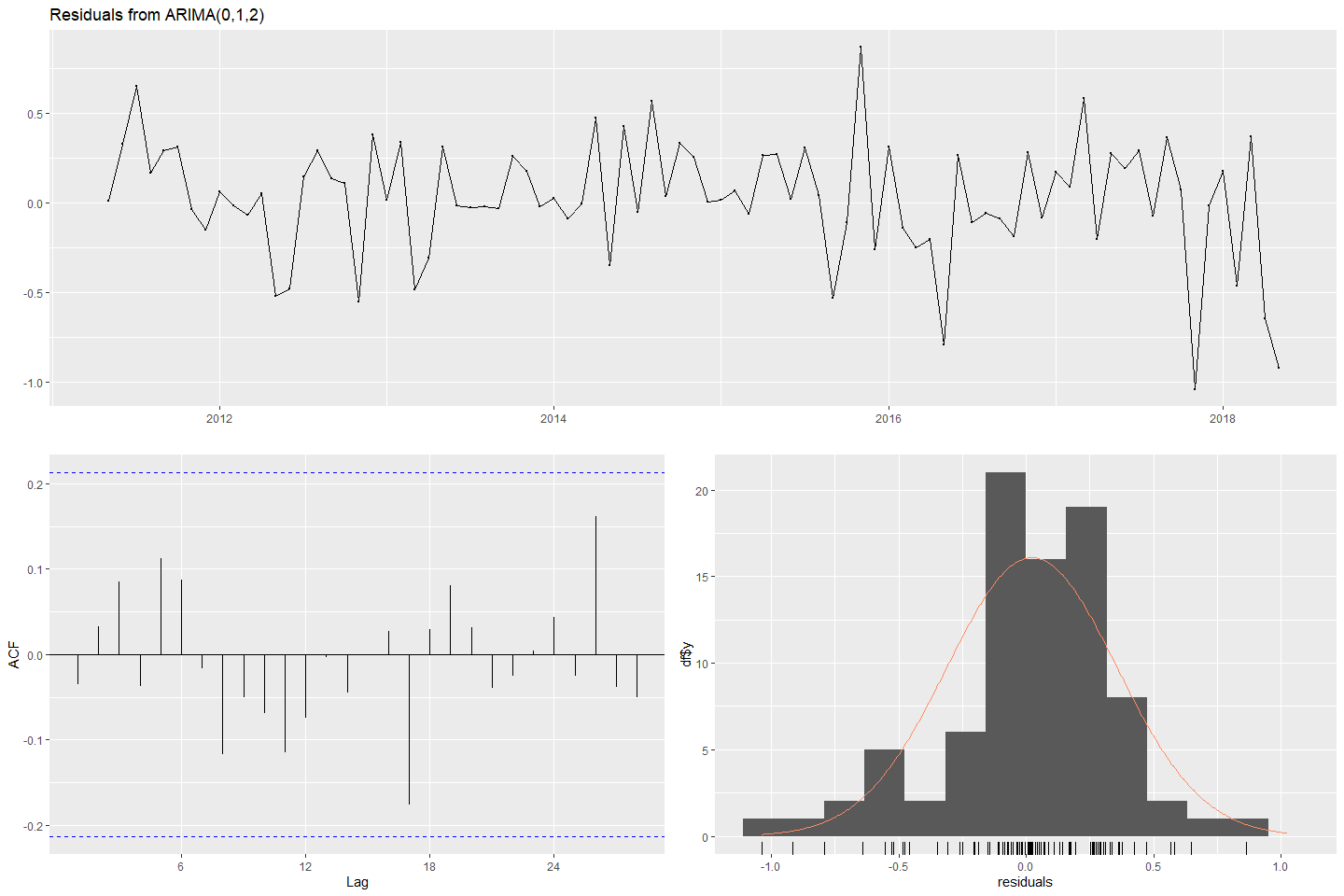
#### Forecasting S02: Var02 & Var03 with ARIMA

# Fit ARIMA models to Var02 & Var03  
fit\_s02\_ARIMA\_2 <- auto.arima(s02\_ts[,1], stepwise = TRUE)  
fit\_s02\_ARIMA\_3 <- auto.arima(s02\_ts[,2], stepwise = TRUE)  
# check residuals  
checkresiduals(fit\_s02\_ARIMA\_2)



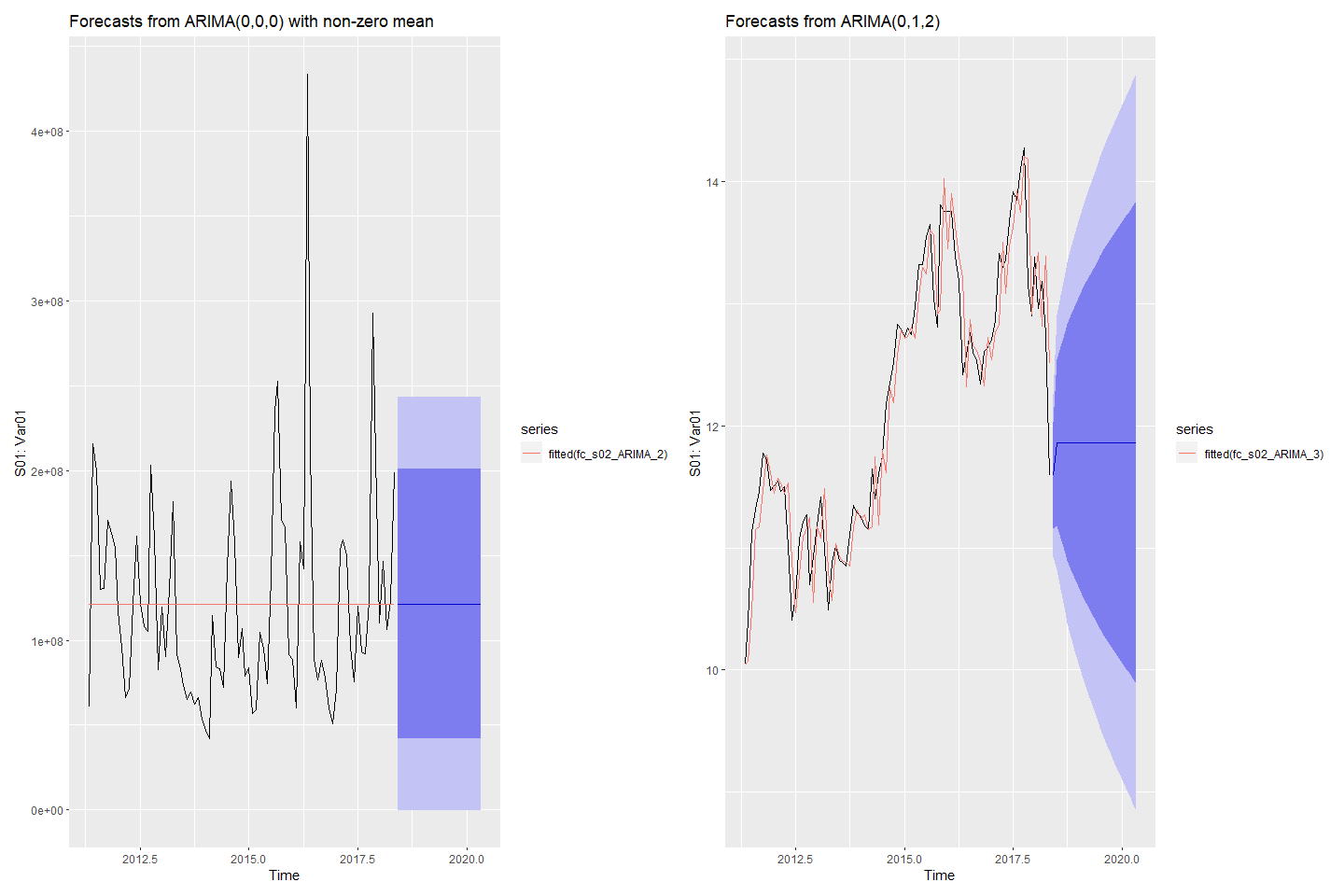
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(0,0,0) with non-zero mean  
#> Q\* = 31.775, df = 16, p-value = 0.0107  
#>   
#> Model df: 1. Total lags used: 17

checkresiduals(fit\_s02\_ARIMA\_3)



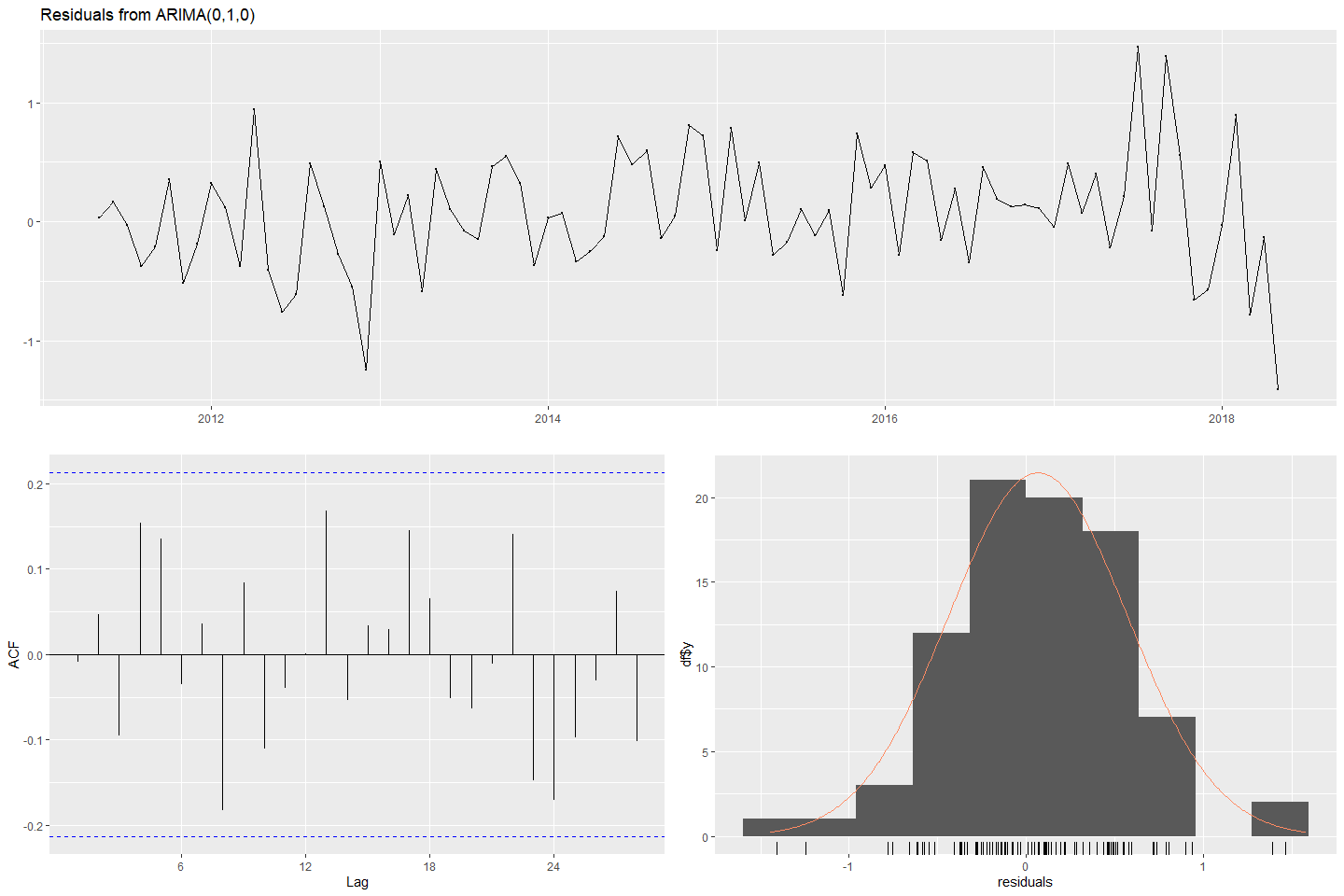
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(0,1,2)  
#> Q\* = 10.44, df = 15, p-value = 0.7912  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s02\_ARIMA\_2 <- forecast(fit\_s02\_ARIMA\_2)  
fc\_s02\_ARIMA\_3 <- forecast(fit\_s02\_ARIMA\_3)  
# plot forecasts  
fa\_S02\_2 <- autoplot(fc\_s02\_ARIMA\_2) + ylab("S02: Var02") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s02\_ARIMA\_2))  
fa\_S02\_3 <- autoplot(fc\_s02\_ARIMA\_3) + ylab("S02: Var03") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s02\_ARIMA\_3))  
(fa\_S02\_2 + fa\_S02\_3)



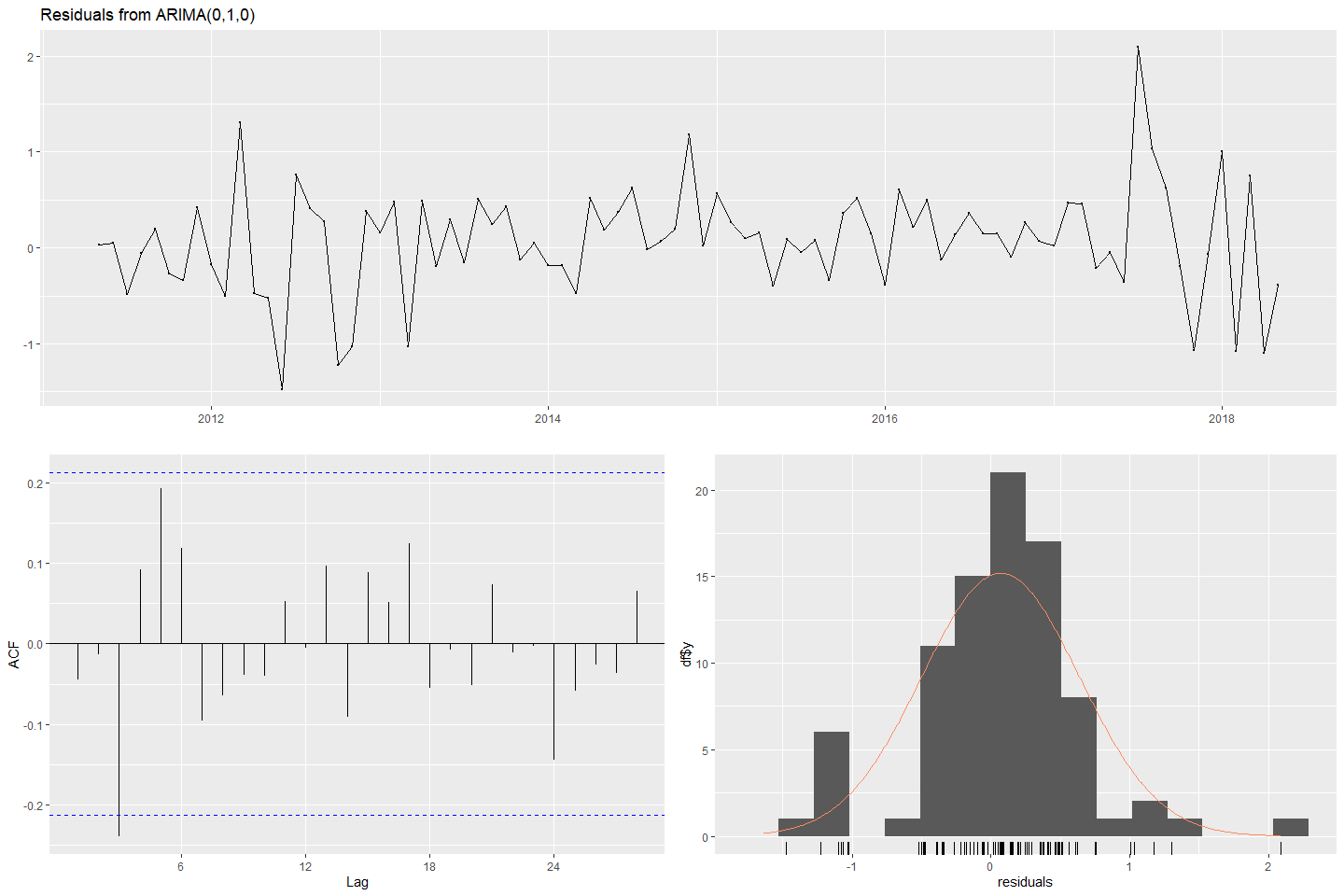
#### Forecasting S03: Var05 & Var07 with ARIMA

# Fit ARIMA models to Var05 & Var07  
fit\_s03\_ARIMA\_5 <- auto.arima(s03\_ts[,1], stepwise = TRUE)  
fit\_s03\_ARIMA\_7 <- auto.arima(s03\_ts[,2], stepwise = TRUE)  
# check residuals  
checkresiduals(fit\_s03\_ARIMA\_5)



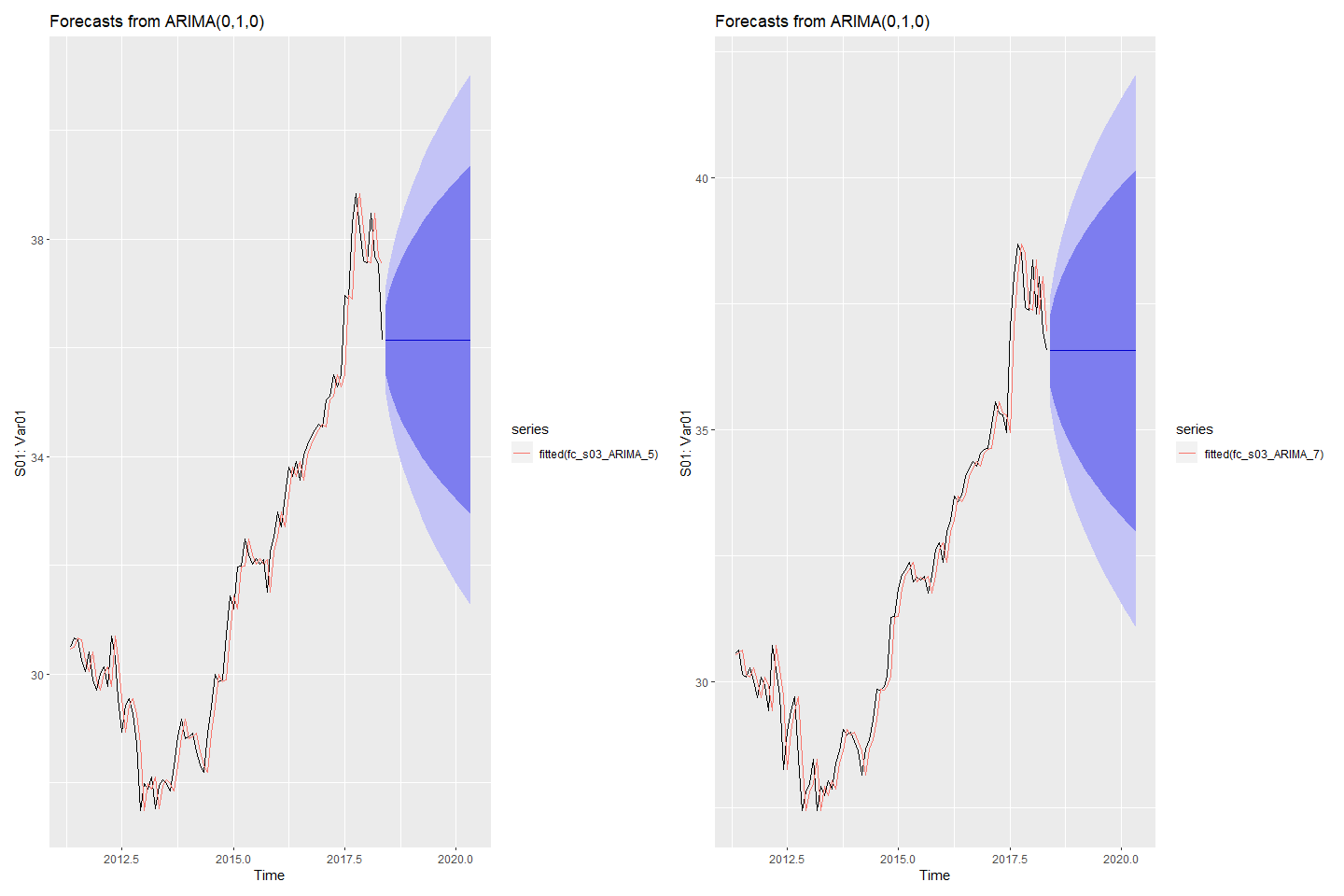
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(0,1,0)  
#> Q\* = 16.015, df = 17, p-value = 0.5228  
#>   
#> Model df: 0. Total lags used: 17

checkresiduals(fit\_s03\_ARIMA\_7)



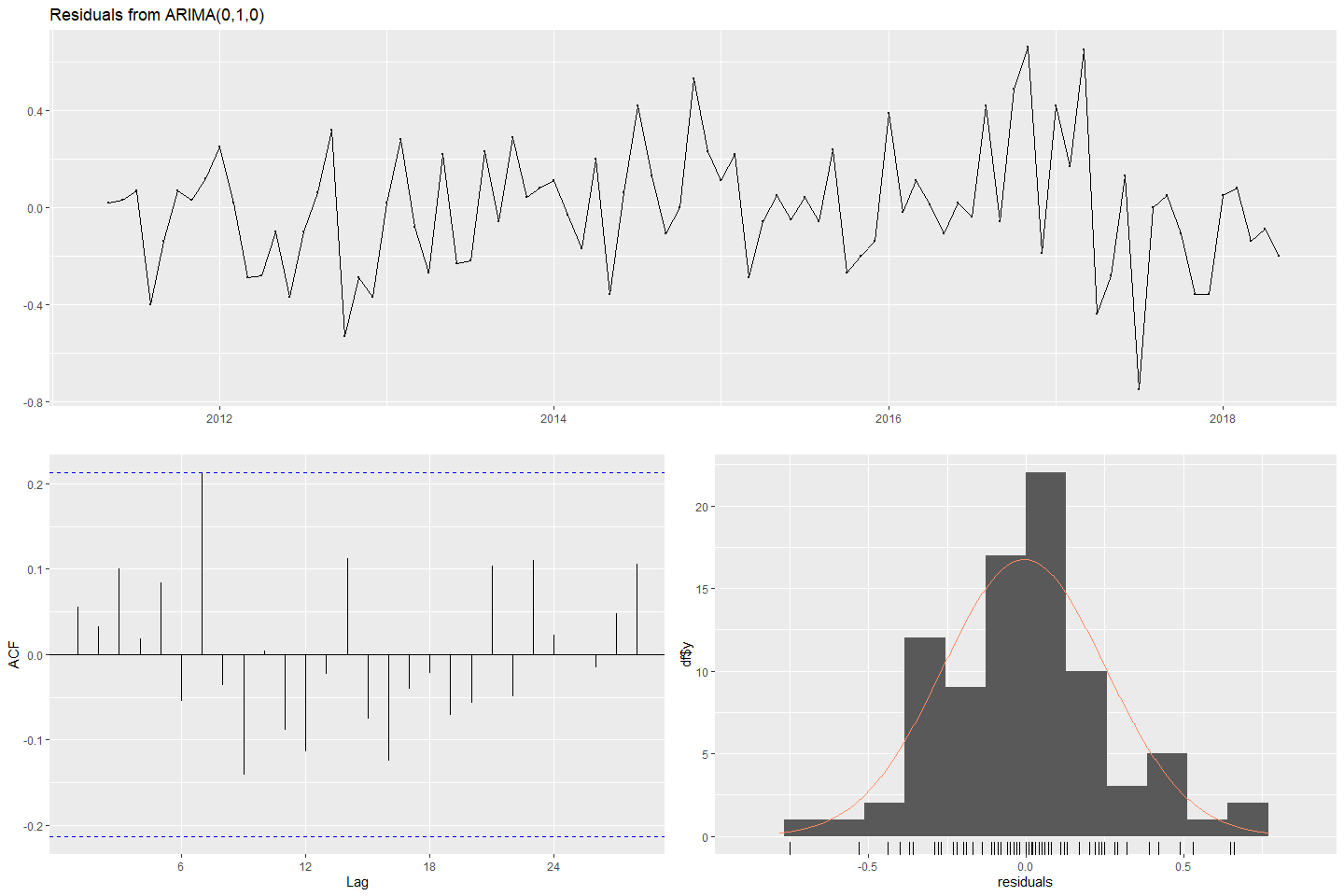
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(0,1,0)  
#> Q\* = 17.397, df = 17, p-value = 0.4278  
#>   
#> Model df: 0. Total lags used: 17

# forecast  
fc\_s03\_ARIMA\_5 <- forecast(fit\_s03\_ARIMA\_5)  
fc\_s03\_ARIMA\_7 <- forecast(fit\_s03\_ARIMA\_7)  
# plot forecasts  
fa\_S03\_5 <- autoplot(fc\_s03\_ARIMA\_5) + ylab("S03: Var05") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s03\_ARIMA\_5))  
fa\_S03\_7 <- autoplot(fc\_s03\_ARIMA\_7) + ylab("S03: Var07") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s03\_ARIMA\_7))  
(fa\_S03\_5 + fa\_S03\_7)



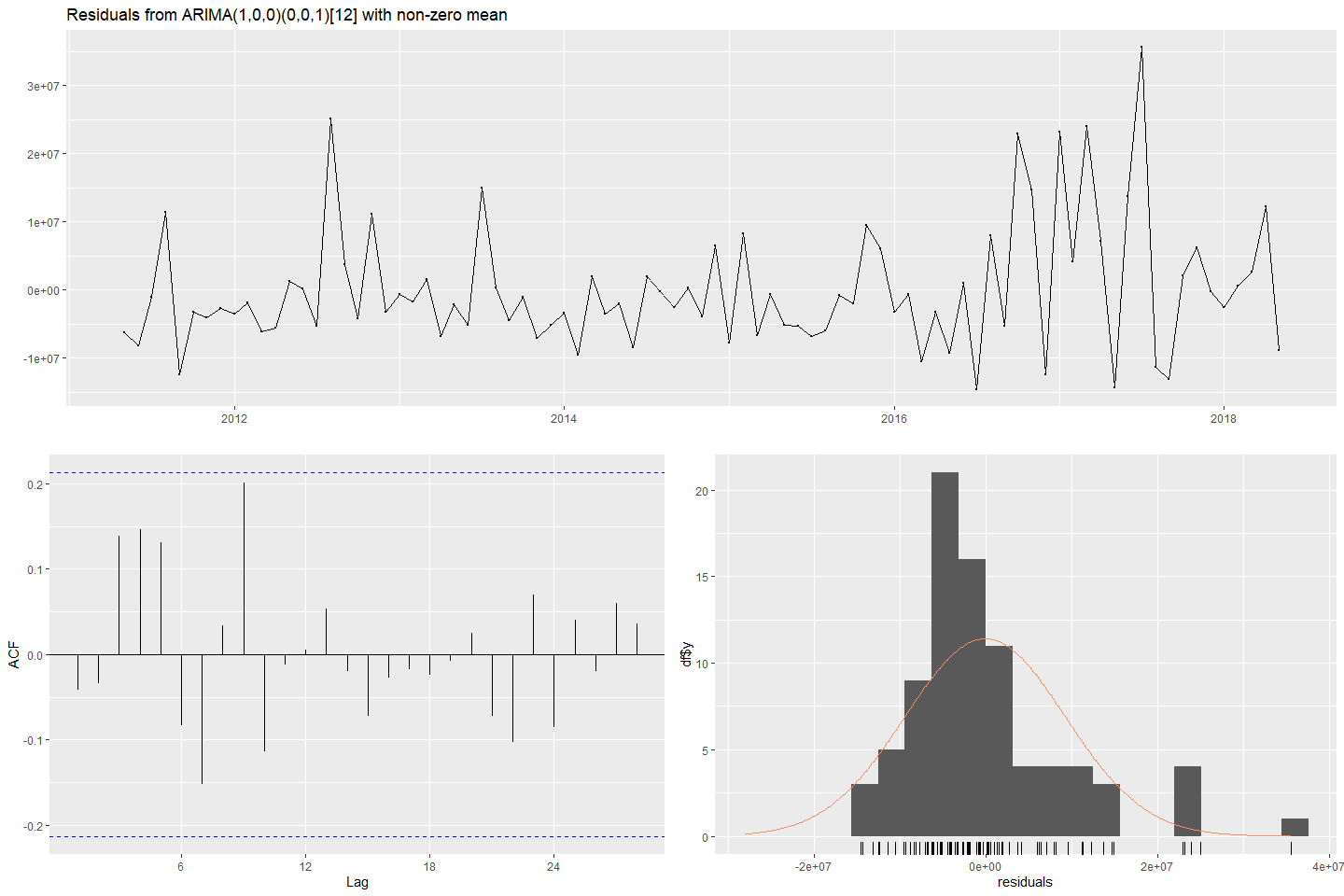
#### Forecasting S04: Var01 & Var02 with ARIMA

# Fit ARIMA models to Var01 & Var02  
fit\_s04\_ARIMA\_1 <- auto.arima(s04\_ts[,1], stepwise = TRUE)  
fit\_s04\_ARIMA\_2 <- auto.arima(s04\_ts[,2], stepwise = TRUE)  
# check residuals  
checkresiduals(fit\_s04\_ARIMA\_1)



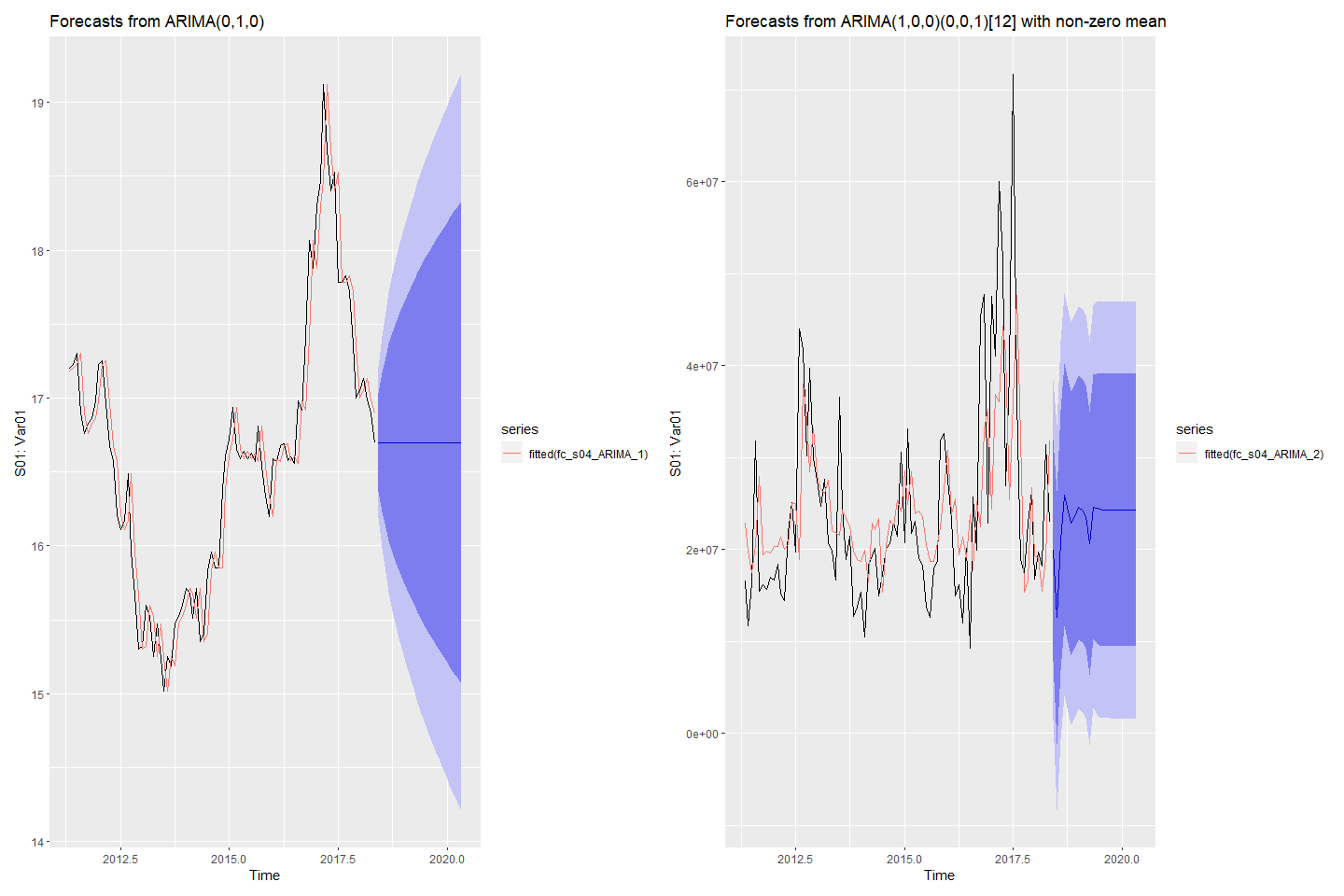
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(0,1,0)  
#> Q\* = 14.463, df = 17, p-value = 0.6341  
#>   
#> Model df: 0. Total lags used: 17

checkresiduals(fit\_s04\_ARIMA\_2)



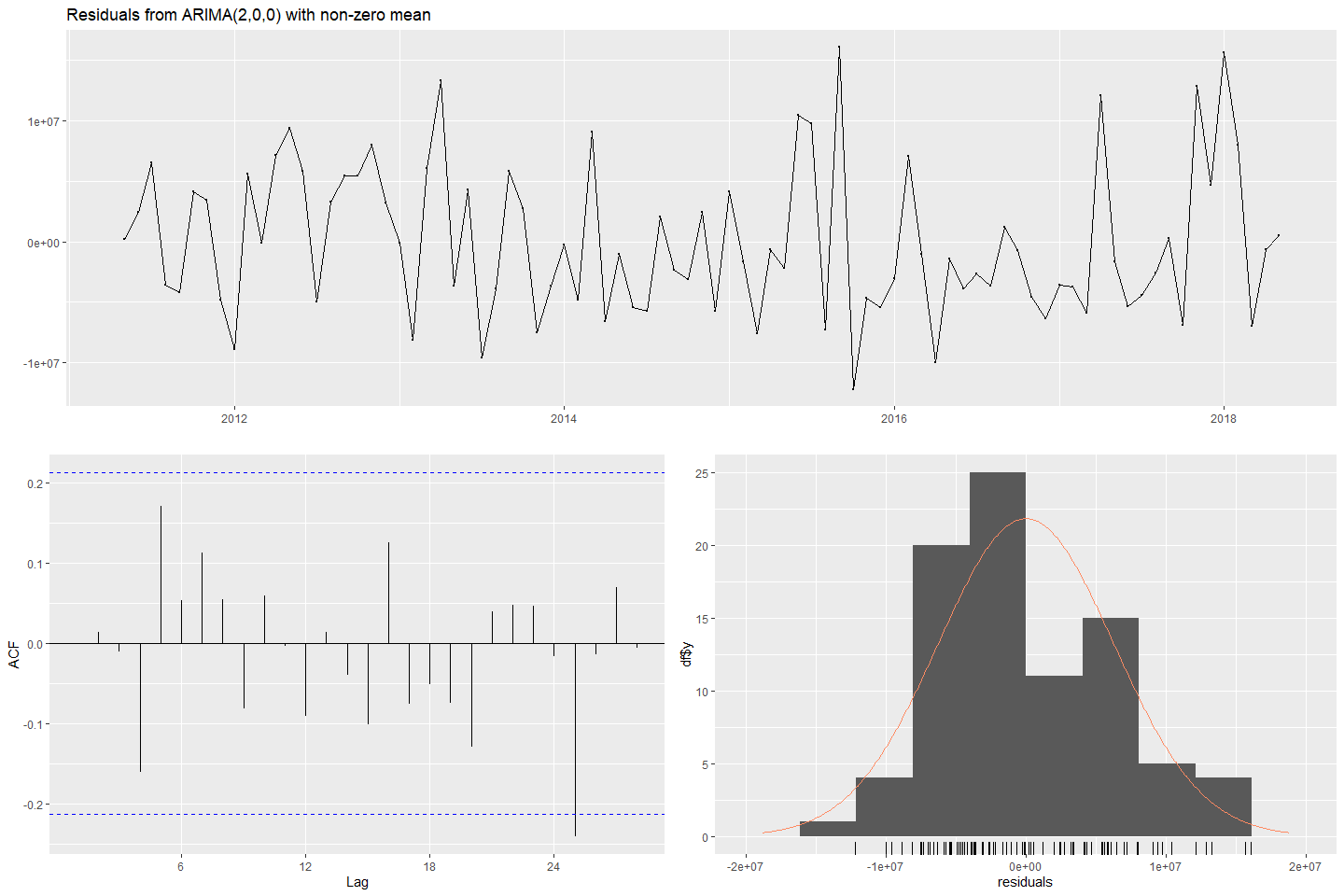
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(1,0,0)(0,0,1)[12] with non-zero mean  
#> Q\* = 14.687, df = 14, p-value = 0.3999  
#>   
#> Model df: 3. Total lags used: 17

# forecast  
fc\_s04\_ARIMA\_1 <- forecast(fit\_s04\_ARIMA\_1)  
fc\_s04\_ARIMA\_2 <- forecast(fit\_s04\_ARIMA\_2)  
# plot forecasts  
fa\_S04\_1 <- autoplot(fc\_s04\_ARIMA\_1) + ylab("S04: Var01") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s04\_ARIMA\_1))  
fa\_S04\_2 <- autoplot(fc\_s04\_ARIMA\_2) + ylab("S04: Var02") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s04\_ARIMA\_2))  
(fa\_S04\_1 + fa\_S04\_2)



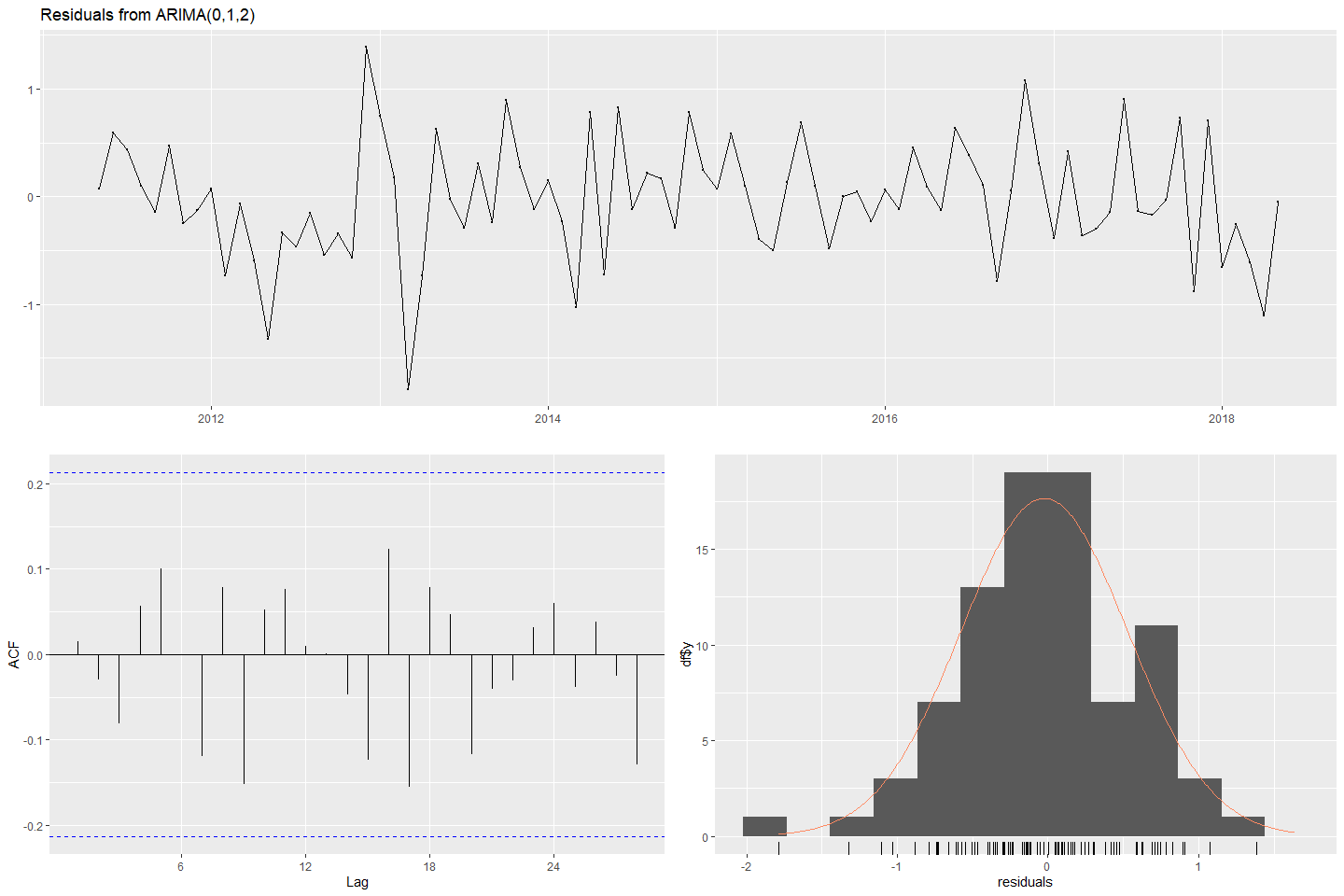
#### Forecasting S05: Var02 & Var03 with ARIMA

# Fit ARIMA models to Var02 & Var03  
fit\_s05\_ARIMA\_2 <- auto.arima(s05\_ts[,1], stepwise = TRUE)  
fit\_s05\_ARIMA\_3 <- auto.arima(s05\_ts[,2], stepwise = TRUE)  
# check residuals  
checkresiduals(fit\_s05\_ARIMA\_2)



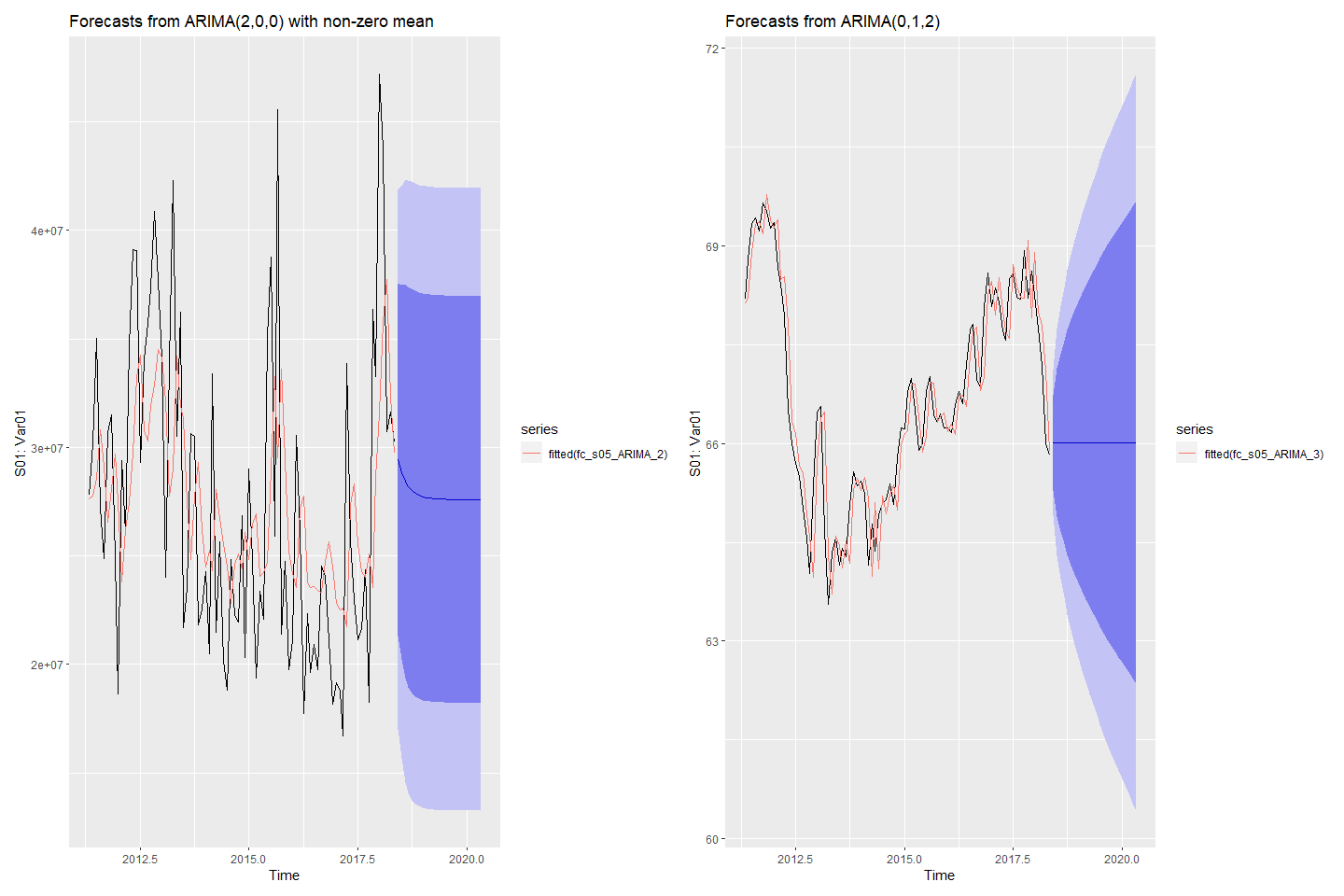
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(2,0,0) with non-zero mean  
#> Q\* = 12.202, df = 14, p-value = 0.5901  
#>   
#> Model df: 3. Total lags used: 17

checkresiduals(fit\_s05\_ARIMA\_3)



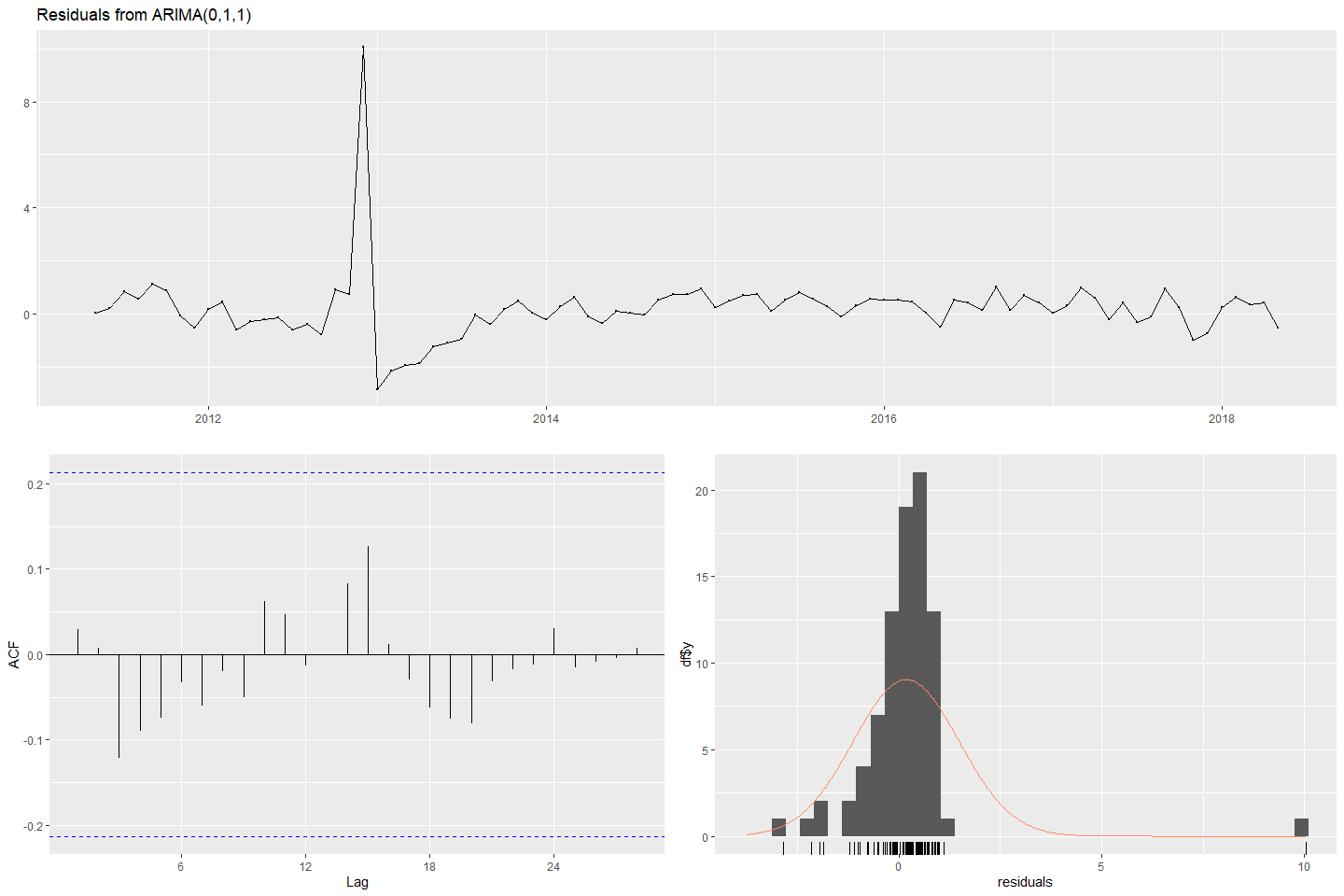
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(0,1,2)  
#> Q\* = 13.089, df = 15, p-value = 0.5954  
#>   
#> Model df: 2. Total lags used: 17

# forecast  
fc\_s05\_ARIMA\_2 <- forecast(fit\_s05\_ARIMA\_2)  
fc\_s05\_ARIMA\_3 <- forecast(fit\_s05\_ARIMA\_3)  
# plot forecasts  
fa\_S05\_2 <- autoplot(fc\_s05\_ARIMA\_2) + ylab("S05: Var02") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s05\_ARIMA\_2))  
fa\_S05\_3 <- autoplot(fc\_s05\_ARIMA\_3) + ylab("S05: Var03") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s05\_ARIMA\_3))  
(fa\_S05\_2 + fa\_S05\_3)



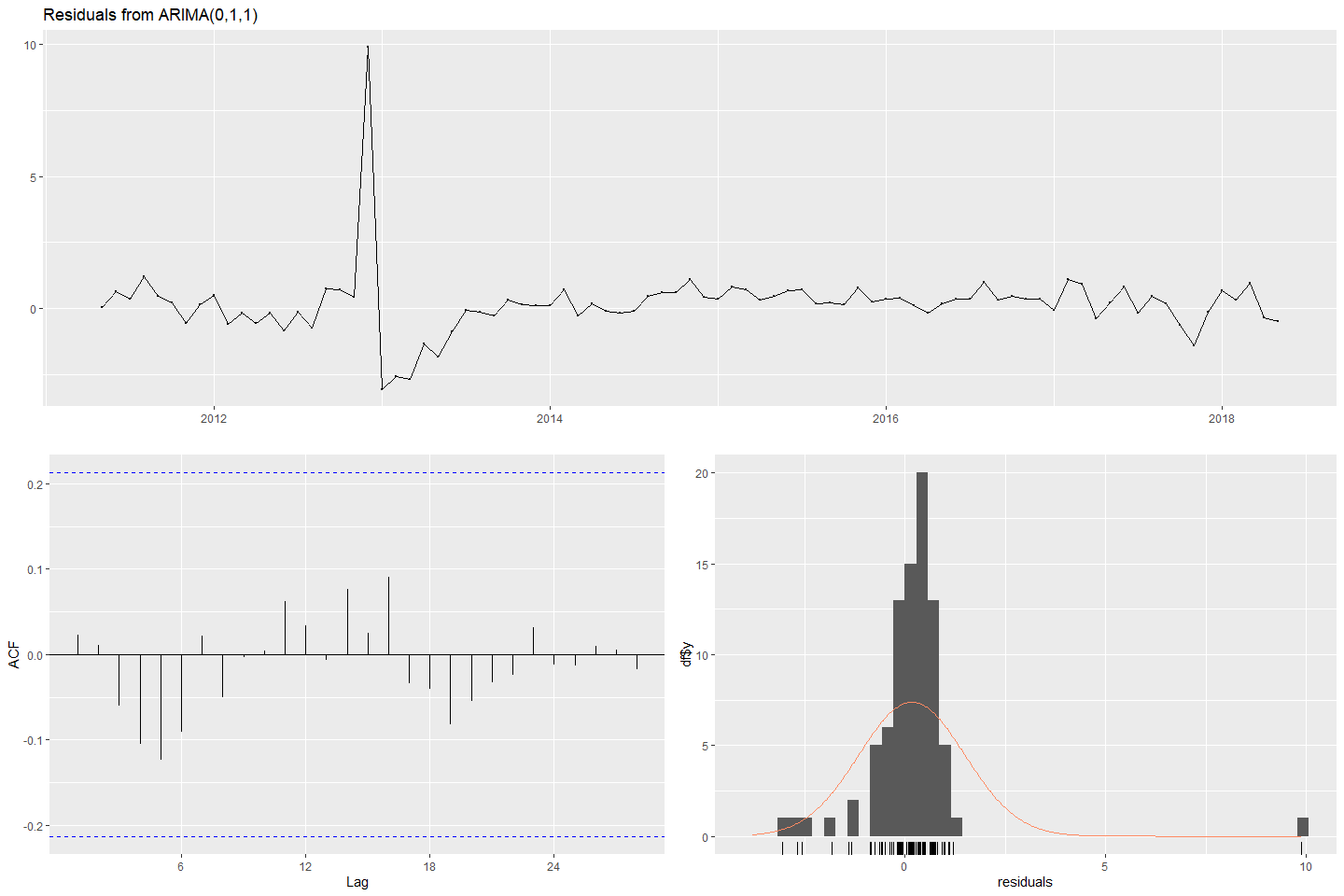
#### Forecasting S06: Var05 & Var07 with ARIMA

# Fit ARIMA models to Var05 & Var07  
fit\_s06\_ARIMA\_5 <- auto.arima(s06\_ts[,1], stepwise = TRUE)  
fit\_s06\_ARIMA\_7 <- auto.arima(s06\_ts[,2], stepwise = TRUE)  
# check residuals  
checkresiduals(fit\_s06\_ARIMA\_5)



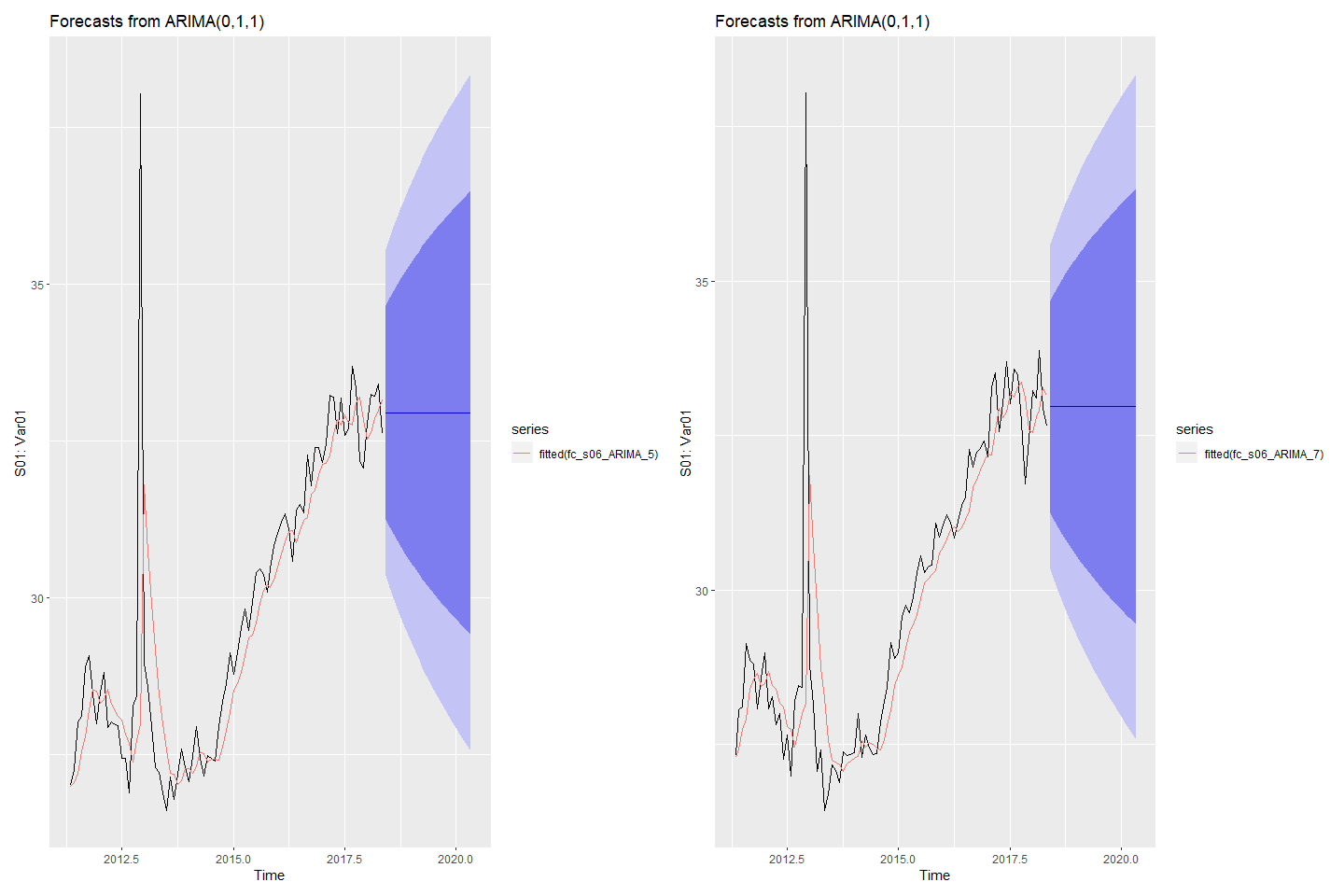
#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(0,1,1)  
#> Q\* = 6.5126, df = 16, p-value = 0.9816  
#>   
#> Model df: 1. Total lags used: 17

checkresiduals(fit\_s06\_ARIMA\_7)



#>   
#> Ljung-Box test  
#>   
#> data: Residuals from ARIMA(0,1,1)  
#> Q\* = 6.0207, df = 16, p-value = 0.9879  
#>   
#> Model df: 1. Total lags used: 17

# forecast  
fc\_s06\_ARIMA\_5 <- forecast(fit\_s06\_ARIMA\_5)  
fc\_s06\_ARIMA\_7 <- forecast(fit\_s06\_ARIMA\_7)  
# plot forecasts  
fa\_S06\_5 <- autoplot(fc\_s06\_ARIMA\_5) + ylab("S06: Var05") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s06\_ARIMA\_5))  
fa\_S06\_7 <- autoplot(fc\_s06\_ARIMA\_7) + ylab("S06: Var07") + ylab("S01: Var01") +  
 autolayer(fitted(fc\_s06\_ARIMA\_7))  
(fa\_S06\_5 + fa\_S06\_7)



## Model Selection

## Final Forecasting