

Forecasting

2023-04-11

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
#Load/install required package here
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##   date, intersect, setdiff, union
```

```
library(ggplot2)
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method              from
##   as.zoo.data.frame zoo
```

```
library(Kendall)
library(tseries)
library(outliers)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
## v dplyr   1.1.0      v stringr 1.5.0
## v forcats 1.0.0      v tibble  3.2.1
## v purrr   1.0.1      v tidyr   1.3.0
## v readr   2.1.4
```

```
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(smooth)
```

```
## Loading required package: greybox
## Package "greybox", v1.0.7 loaded.
##
##
```

```
## Attaching package: 'greybox'
##
```

```
## The following object is masked from 'package:tidyr':
```

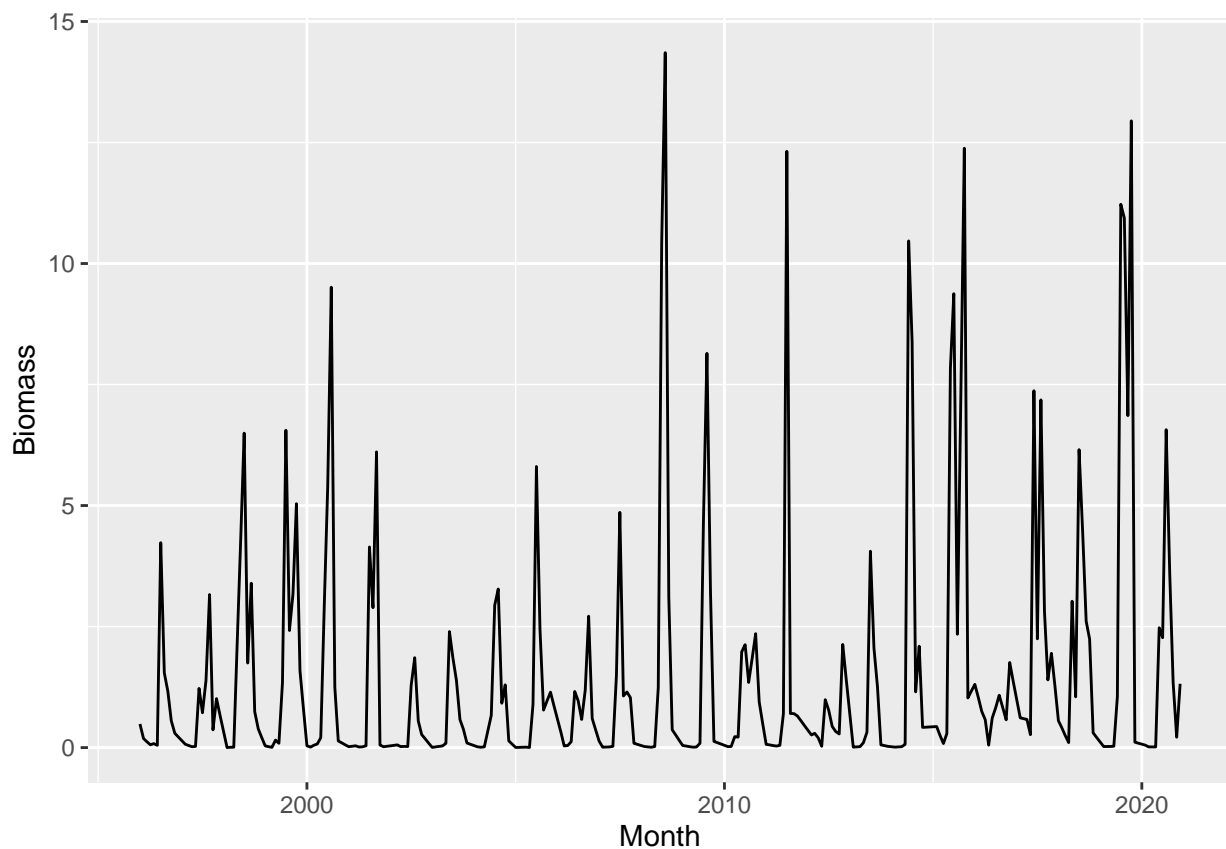
```
##
##   spread
##
## The following object is masked from 'package:lubridate':
##
##   hm
##
## This is package "smooth", v3.2.0
#New package for M9 to assist with tables
#install.packages("kableExtra")
library(kableExtra)

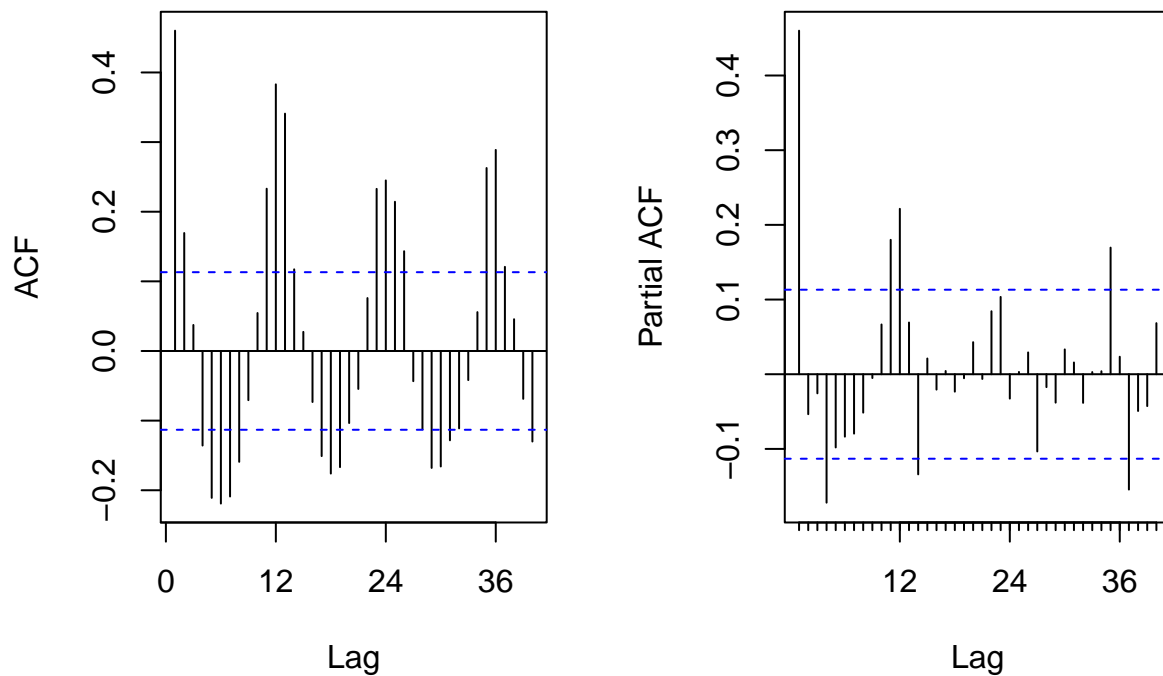
## Warning in !is.null(rmarkdown::metadata$output) && rmarkdown::metadata$output
## %in% : 'length(x) = 2 > 1' in coercion to 'logical(1)'

##
## Attaching package: 'kableExtra'
##
## The following object is masked from 'package:dplyr':
##
##   group_rows
```

(1) Auto-regression

Using the full sample from 1996 to 2020, The original plot, Autocorrelation Function (ACF) and the partial autocorrelation function (PACF) plots shown as follows:

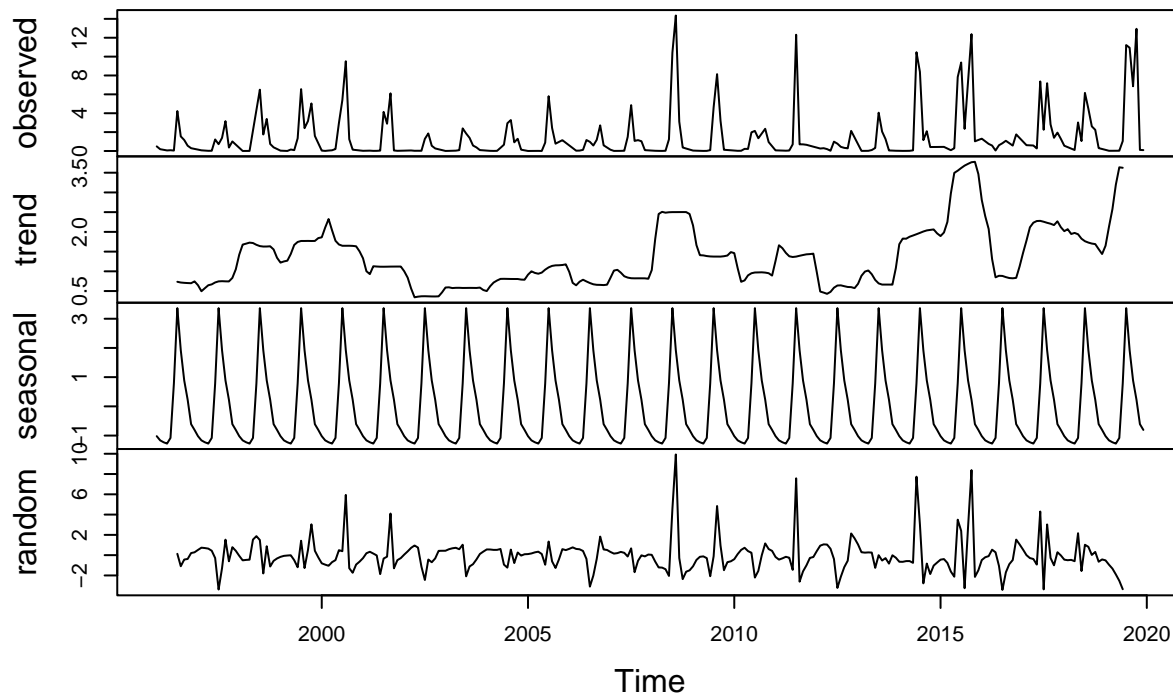




It can be seen that there are both clear trend and seasonal component in this plot. The decomposition of the time series is shown as follows:

```
#Plot ts decompose
decompose_biomass_data <- decompose(ts_biomass,"additive")
plot(decompose_biomass_data)
```

Decomposition of additive time series

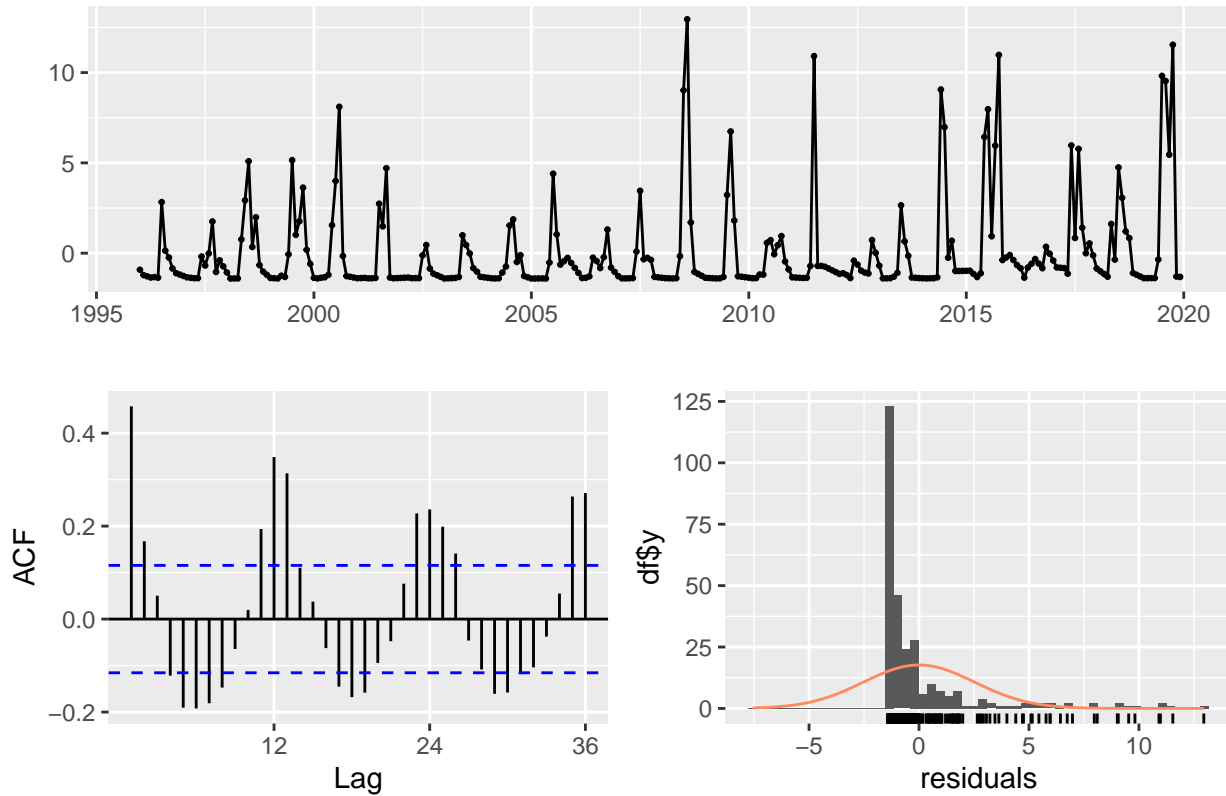


Given the complexity of the series, we further try five models to fit the model and conduct forecasting.

The method includes Arithmetic mean, Seasonal naive, SARIMA, SS Exponential smoothing, and BSM (SS with StructTS). We also present their forecast plots and residuals analysis plots individually.

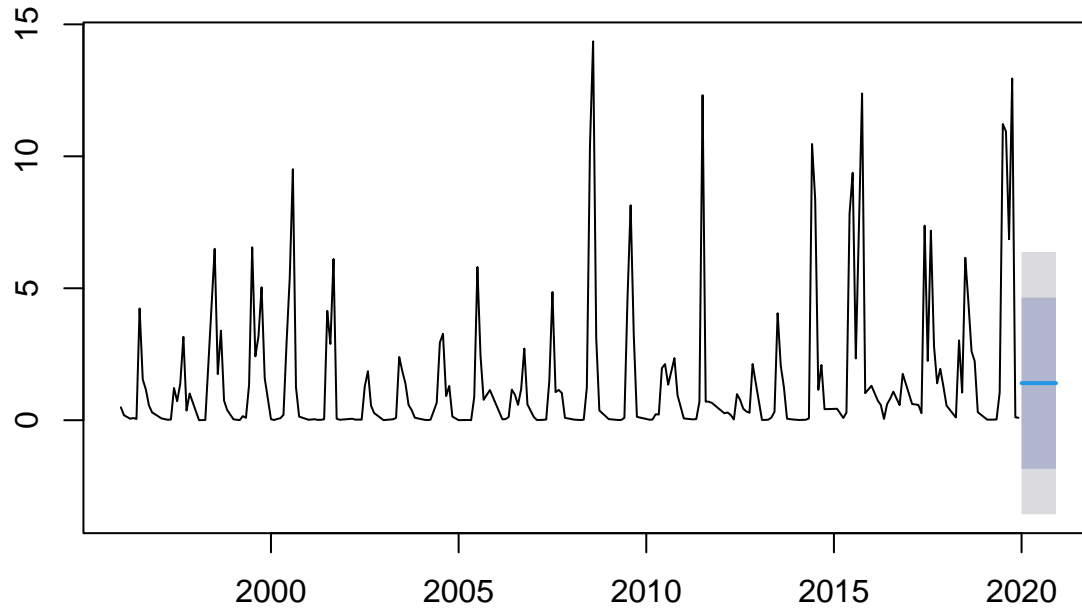
```
# Model 1: Arithmetic mean
# The meanf() has no holdout option
MEAN_seas <- meanf(y = ts_biomass, h = 12)
checkresiduals(MEAN_seas)
```

Residuals from Mean



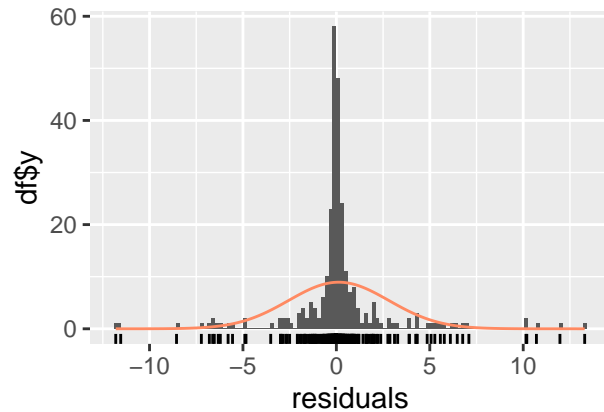
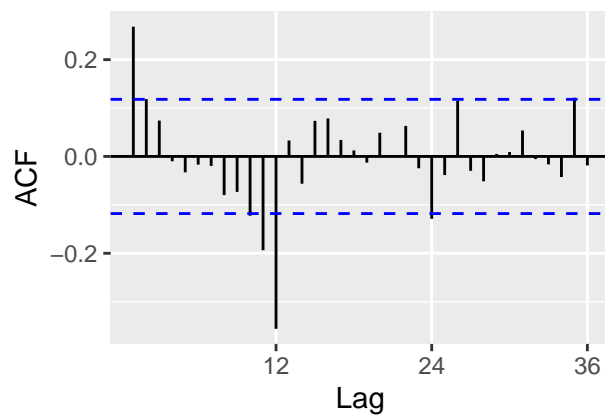
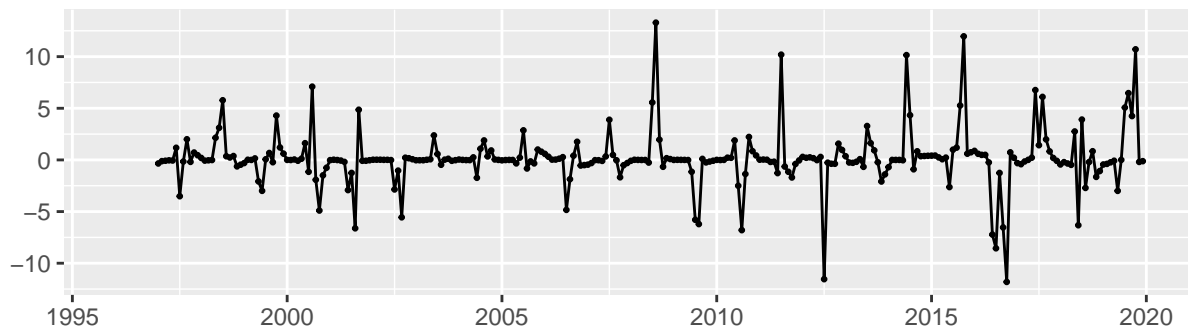
```
##
##  Ljung-Box test
##
## data:  Residuals from Mean
## Q* = 258.95, df = 23, p-value < 2.2e-16
##
## Model df: 1.    Total lags used: 24
plot(MEAN_seas)
```

Forecasts from Mean



```
# Model 2: Seasonal naive
SNAIVE_seas <- snaive(ts_biomass, h=12, holdout=FALSE)
checkresiduals(SNAIVE_seas)
```

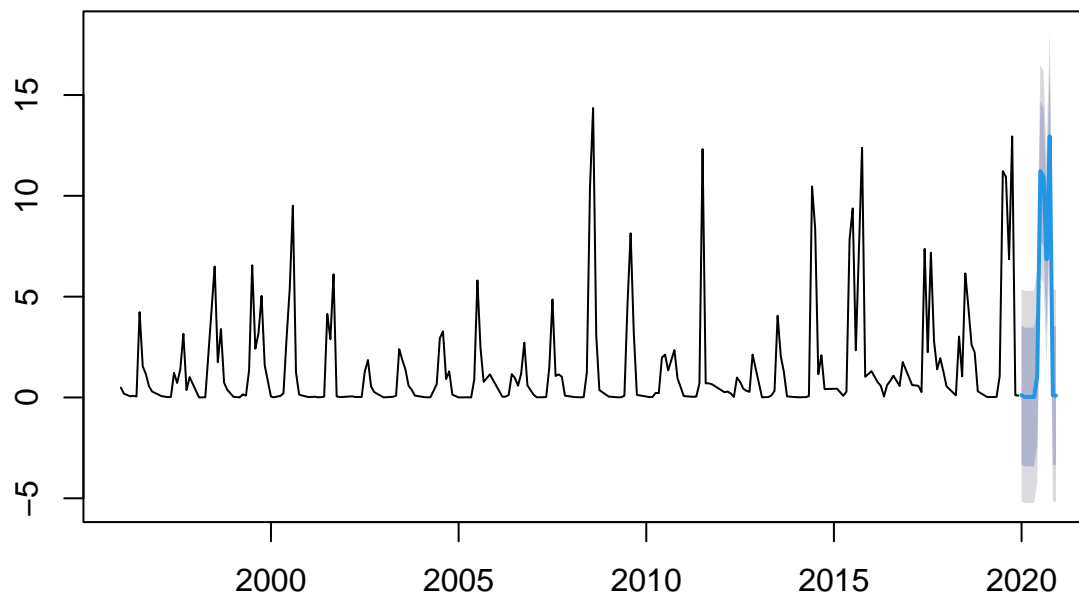
Residuals from Seasonal naive method



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

```
##
## data: Residuals from Seasonal naive method
## Q* = 93.651, df = 24, p-value = 3.556e-10
##
## Model df: 0. Total lags used: 24
plot(SNAIVE_seas)
```

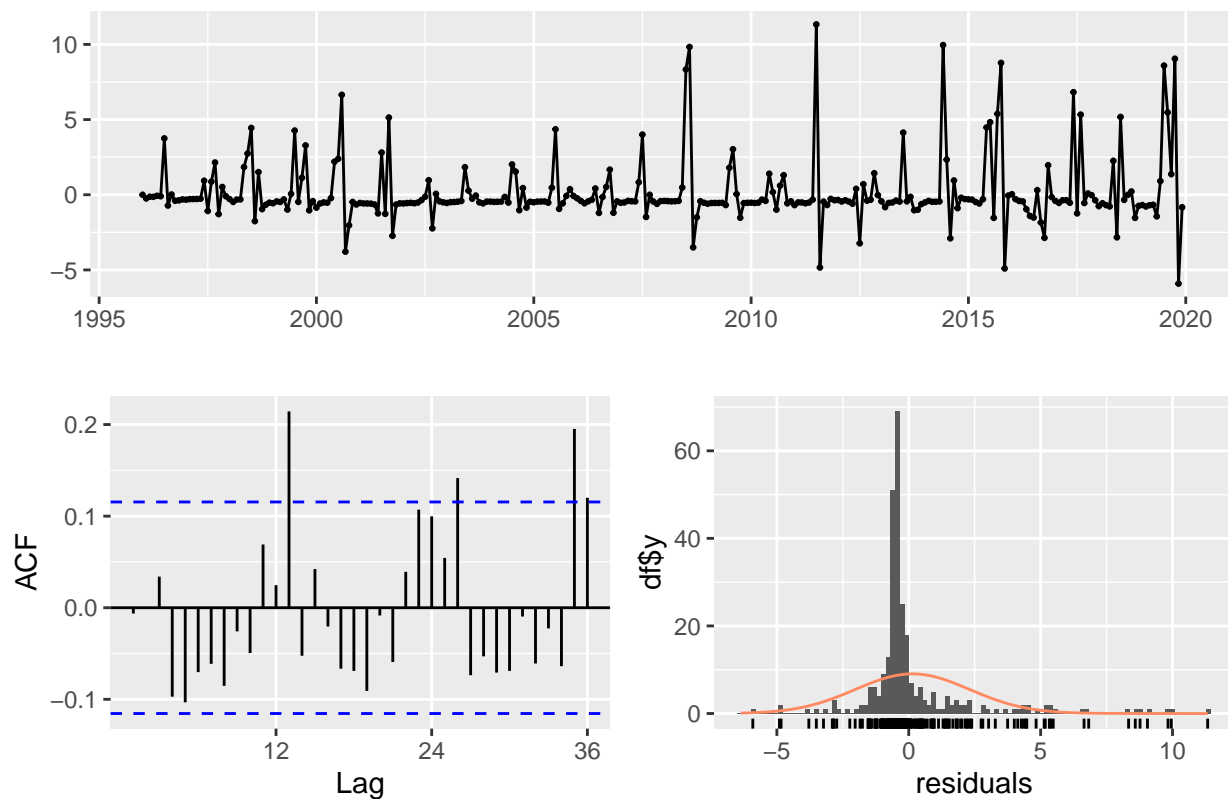
Forecasts from Seasonal naive method



```
# Model 3: SARIMA
```

```
SARIMA_autofit <- auto.arima(ts_biomass)
checkresiduals(SARIMA_autofit)
```

Residuals from ARIMA(1,1,1)(0,0,1)[12]

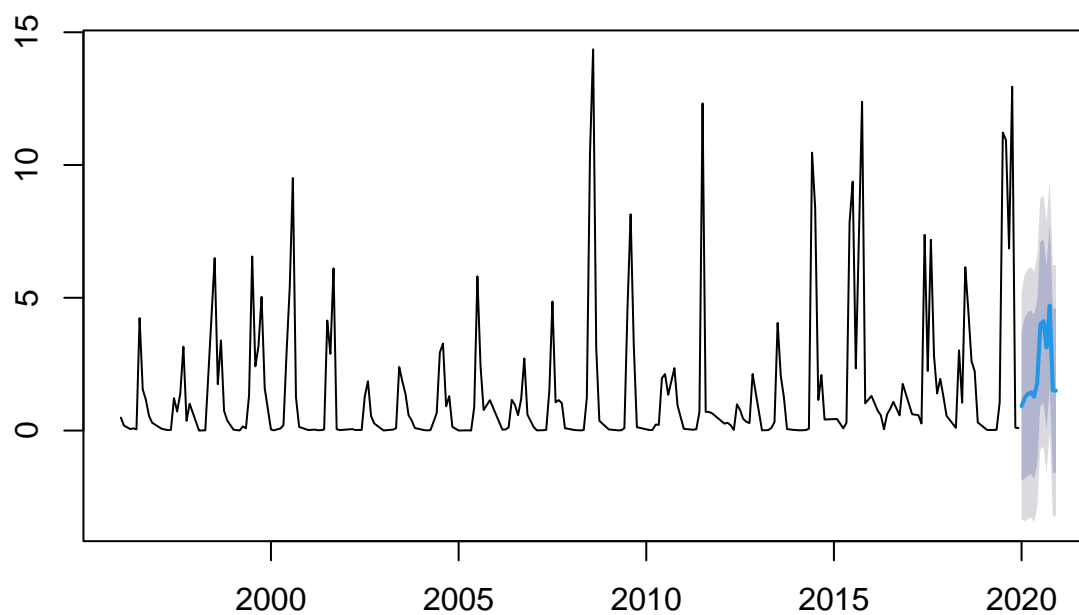


```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,1)(0,0,1)[12]
## Q* = 42.8, df = 21, p-value = 0.003333
##
## Model df: 3.   Total lags used: 24
```

#Generating forecasts

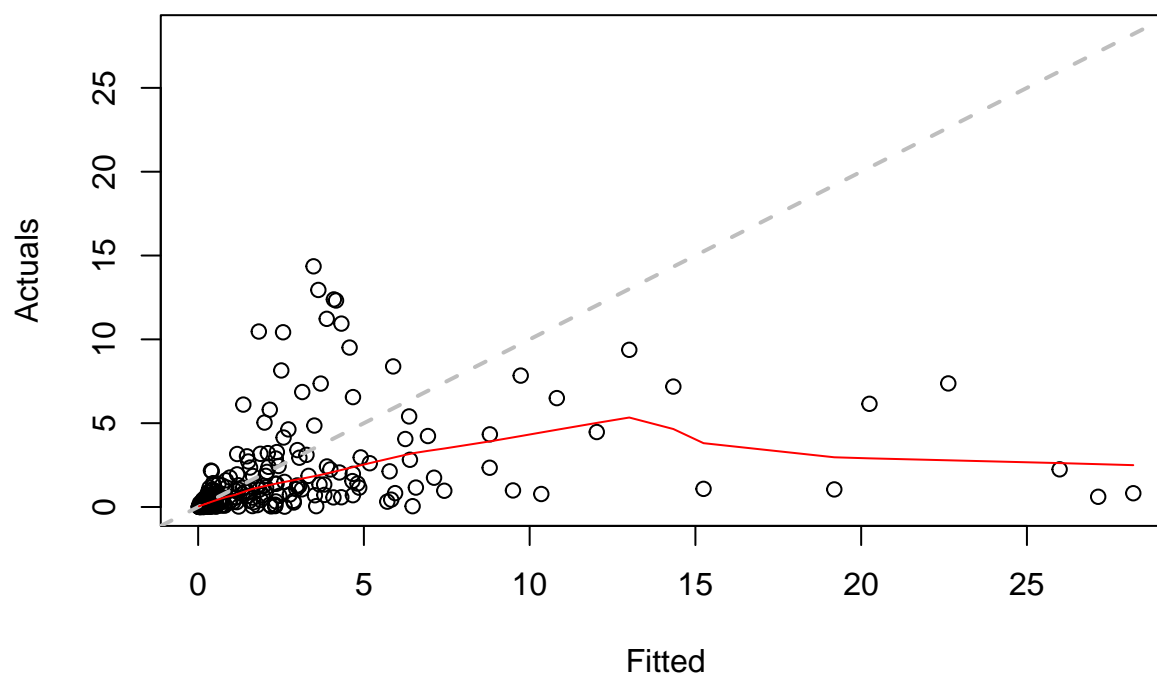
```
SARIMA_for <- forecast(SARIMA_autofit,h=12)
plot(SARIMA_for)
```

Forecasts from ARIMA(1,1,1)(0,0,1)[12]

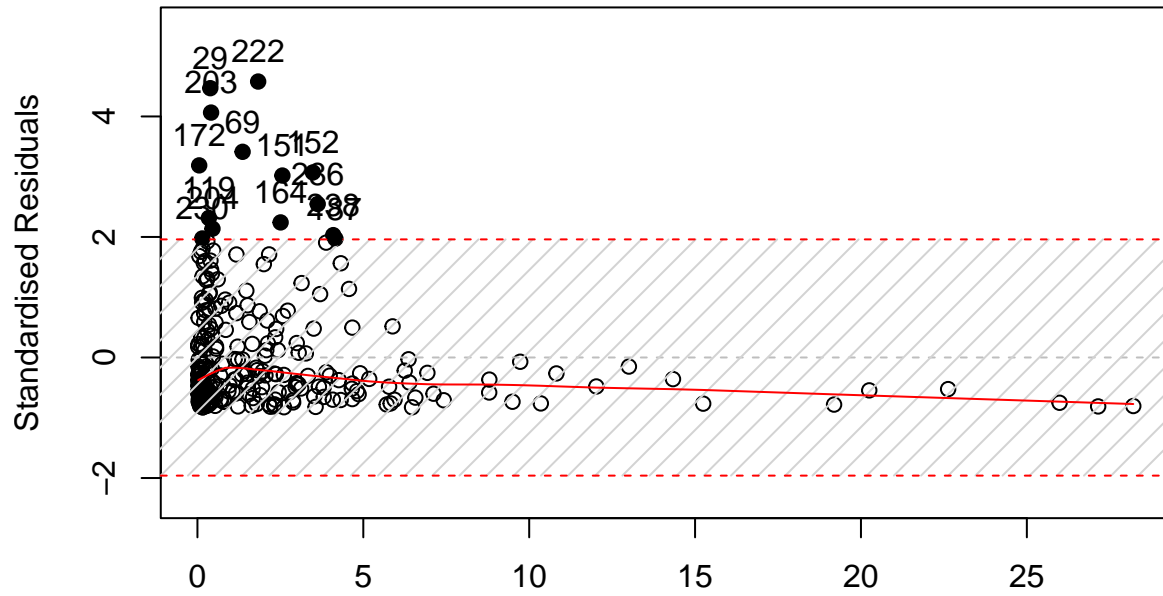


```
# Model 4: SS Exponential smoothing
SSES_seas <- es(ts_biomass,model="ZZZ",h=12,holdout=FALSE)
plot(SSES_seas)
```

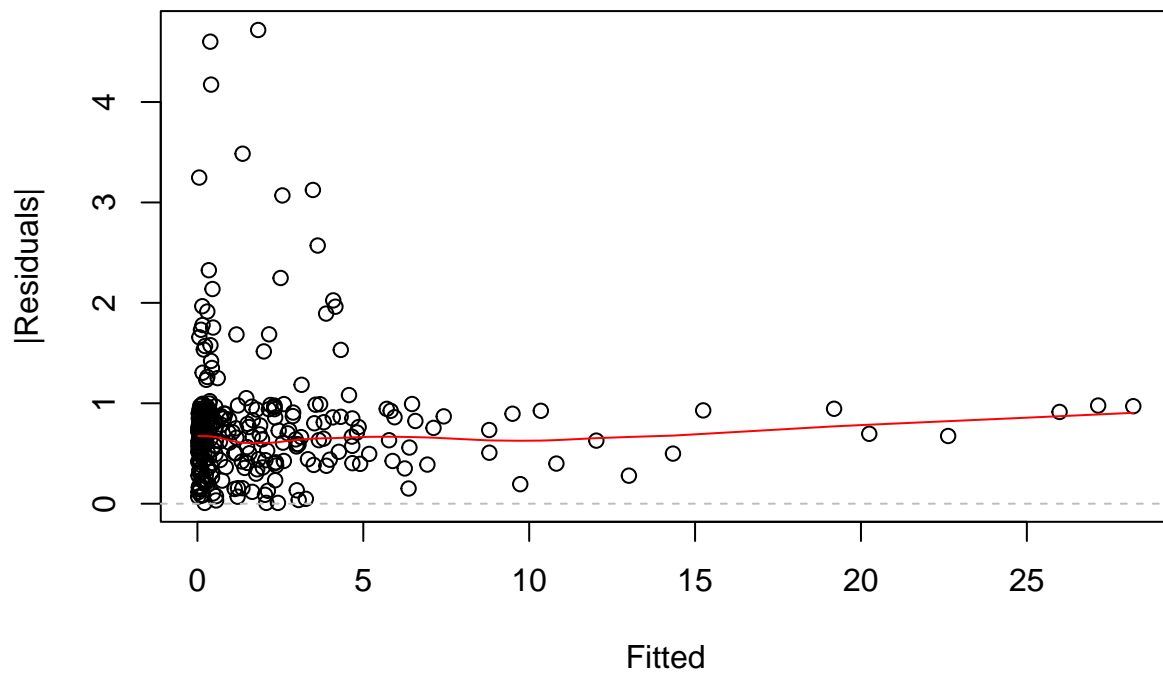
Actuals vs Fitted



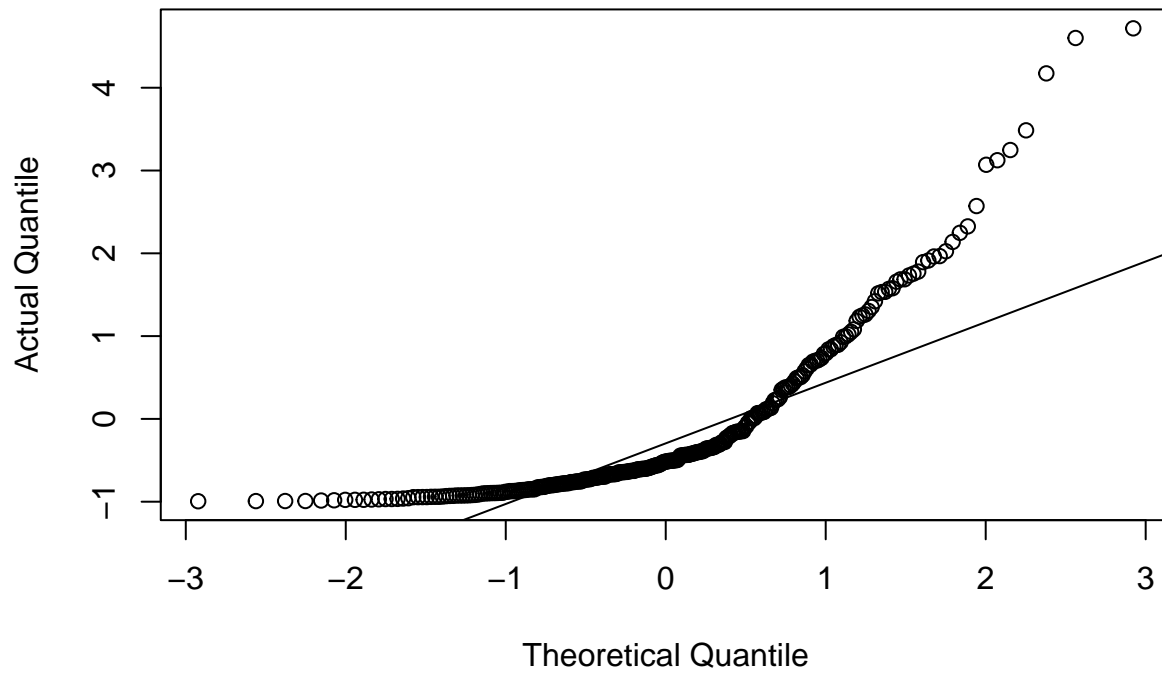
Standardised Residuals vs Fitted



|Residuals| vs Fitted

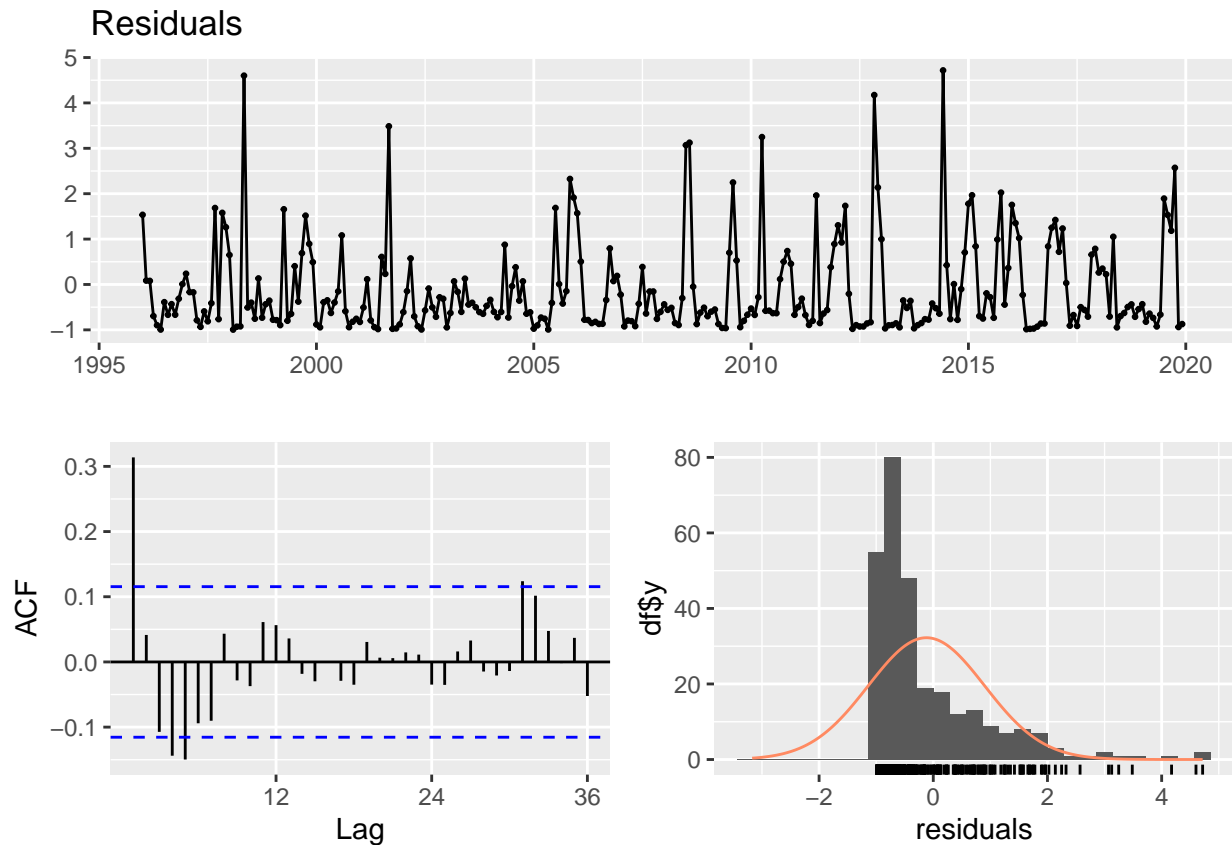


QQ plot of Normal distribution



```
checkresiduals(SSES_seas)
```

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

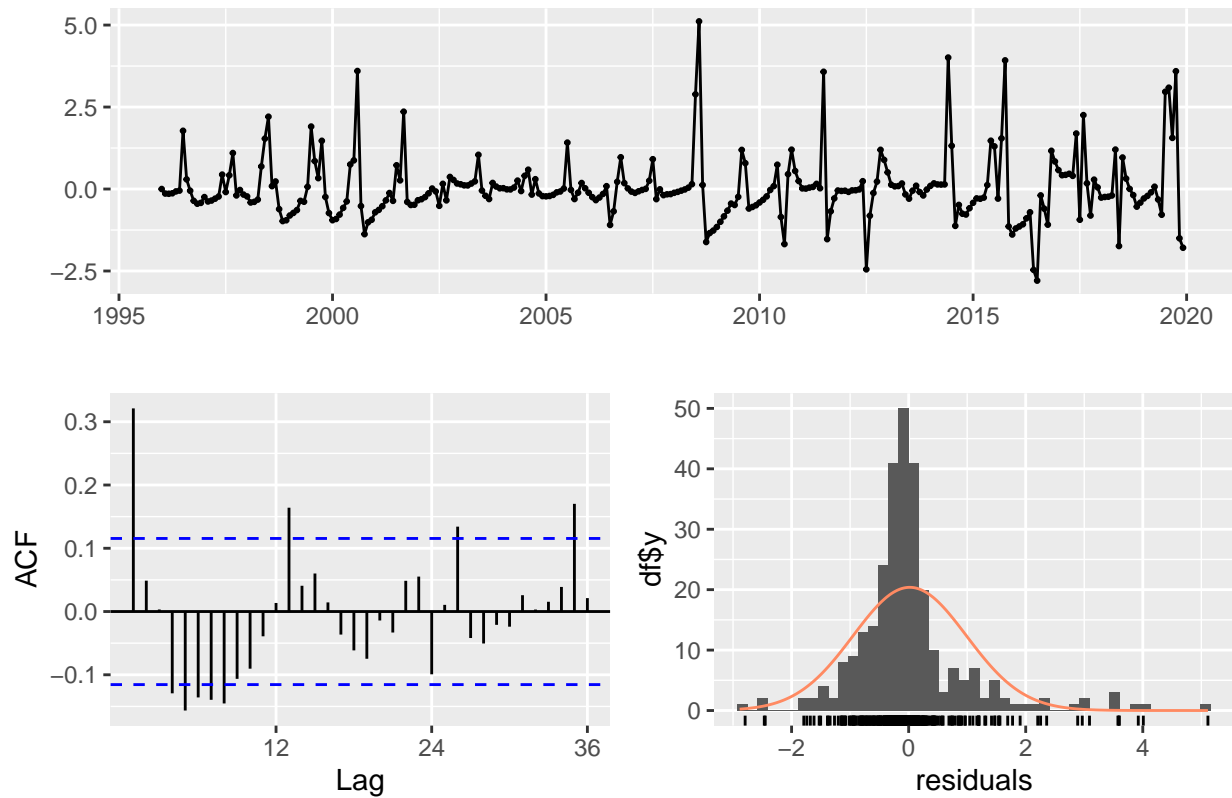


```
# Model 5: SS with StructTS()

SS_seas <- StructTS(ts_biomass,
                    type="BSM",fixed=c(0,0.001,0.3,NA)) #this function has convergence issues
checkresiduals(SS_seas)

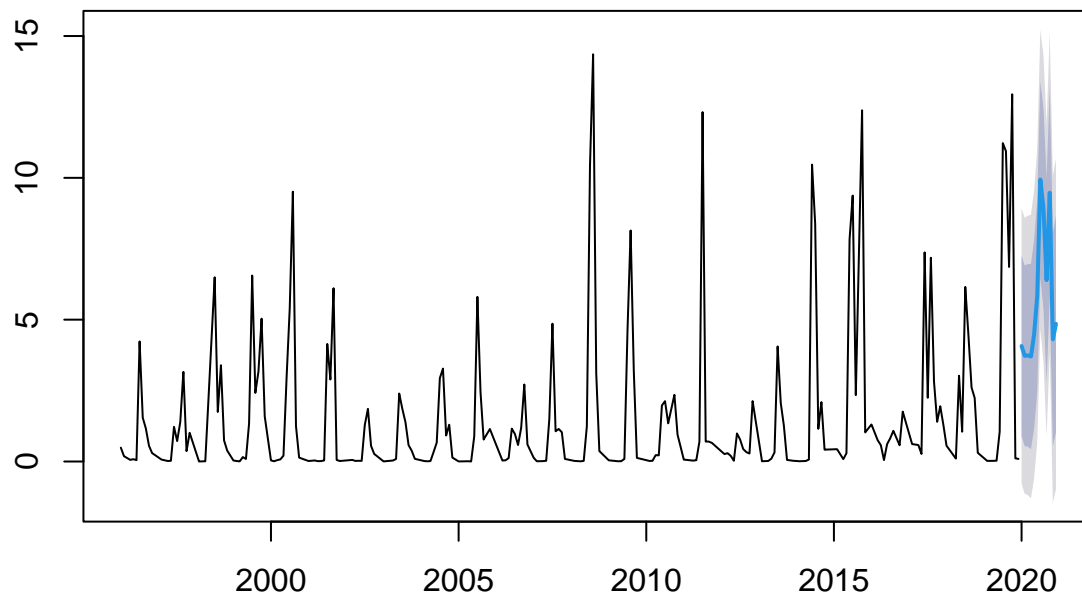
## Warning in modeldf.default(object): Could not find appropriate degrees of
## freedom for this model.
```

Residuals from StructTS



```
#Generating forecasts
SS_for <- forecast(SS_seas,h=12)
plot(SS_for)
```

Forecasts from Basic structural model



Based on the residuals plots,

Table 1: Forecast Accuracy for Seasonal Data

	ME	RMSE	MAE	MPE	MAPE
MEAN	0.09077	1.91972	1.45817	-2993.9993	3030.8195
SNAIVE	-2.12360	4.54636	2.58677	-122.4177	164.9659
SARIMA	-0.75532	1.58841	1.35418	-2867.0051	2879.9488
SSES	-2.55079	4.40694	2.64893	-998.5011	1005.9502
BSM	-4.29032	4.61610	4.29032	-8743.8441	8743.8441

```

#Model 1: Arithmetic mean
MEAN_scores <- accuracy(MEAN_seas$mean,last_obs) #store the performance metrics

#Model 2: Seasonal naive
SNAIVE_scores <- accuracy(SNAIVE_seas$mean,last_obs)

# Model 3: SARIMA
SARIMA_scores <- accuracy(SARIMA_for$mean,last_obs)

# Model 4: SSES
SSES_scores <- accuracy(SSES_seas$forecast,last_obs)

# Model 5: BSM
SS_scores <- accuracy(SS_for$mean,last_obs)

#create data frame
seas_scores <- as.data.frame(rbind(MEAN_scores, SNAIVE_scores, SARIMA_scores,SSES_scores,SS_scores))
row.names(seas_scores) <- c("MEAN", "SNAIVE", "SARIMA", "SSES", "BSM")

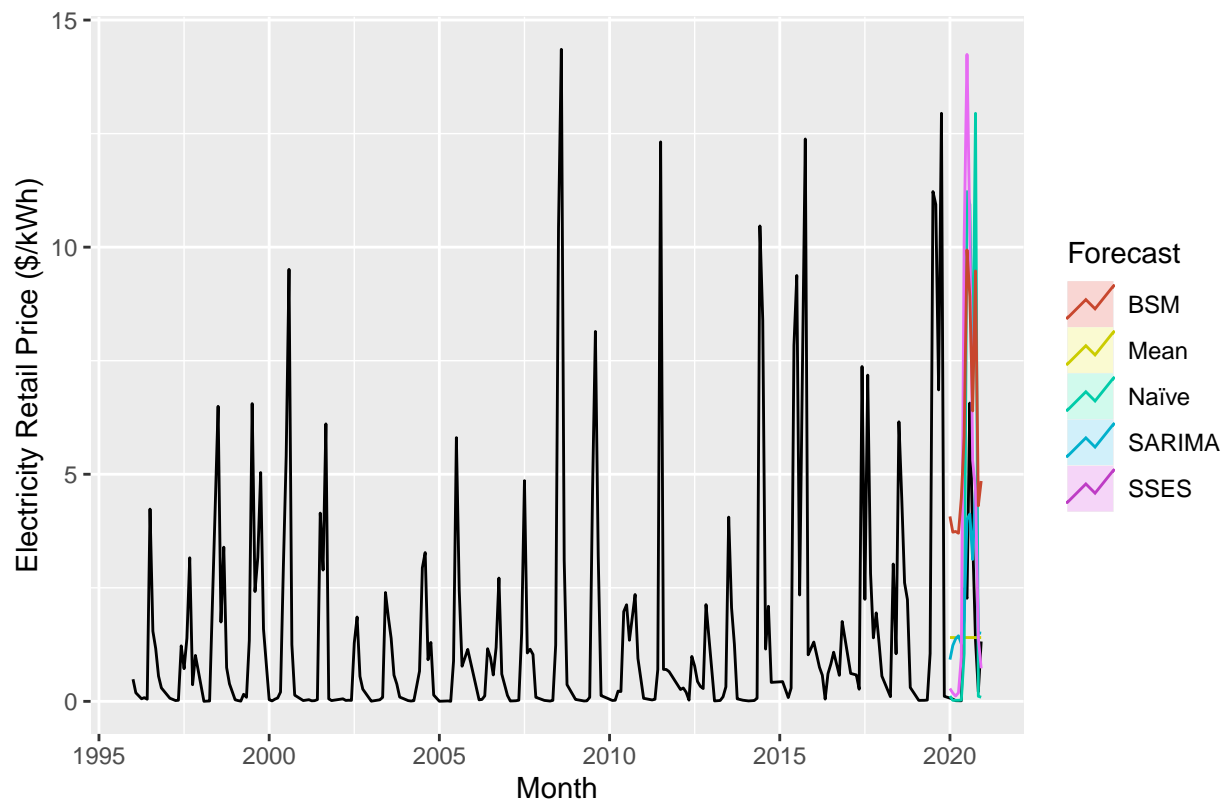
#choose model with lowest RMSE
best_model_index <- which.min(seas_scores[, "RMSE"])
cat("The best model by RMSE is:", row.names(seas_scores[best_model_index,]))

## The best model by RMSE is: SARIMA

kbl(seas_scores,
    caption = "Forecast Accuracy for Seasonal Data",
    digits = array(5,ncol(seas_scores))) %>%
    kable_styling(full_width = FALSE, position = "center") %>%
    #highlight model with lowest RMSE
    kable_styling(latex_options="striped", stripe_index = which.min(seas_scores[, "RMSE"]))

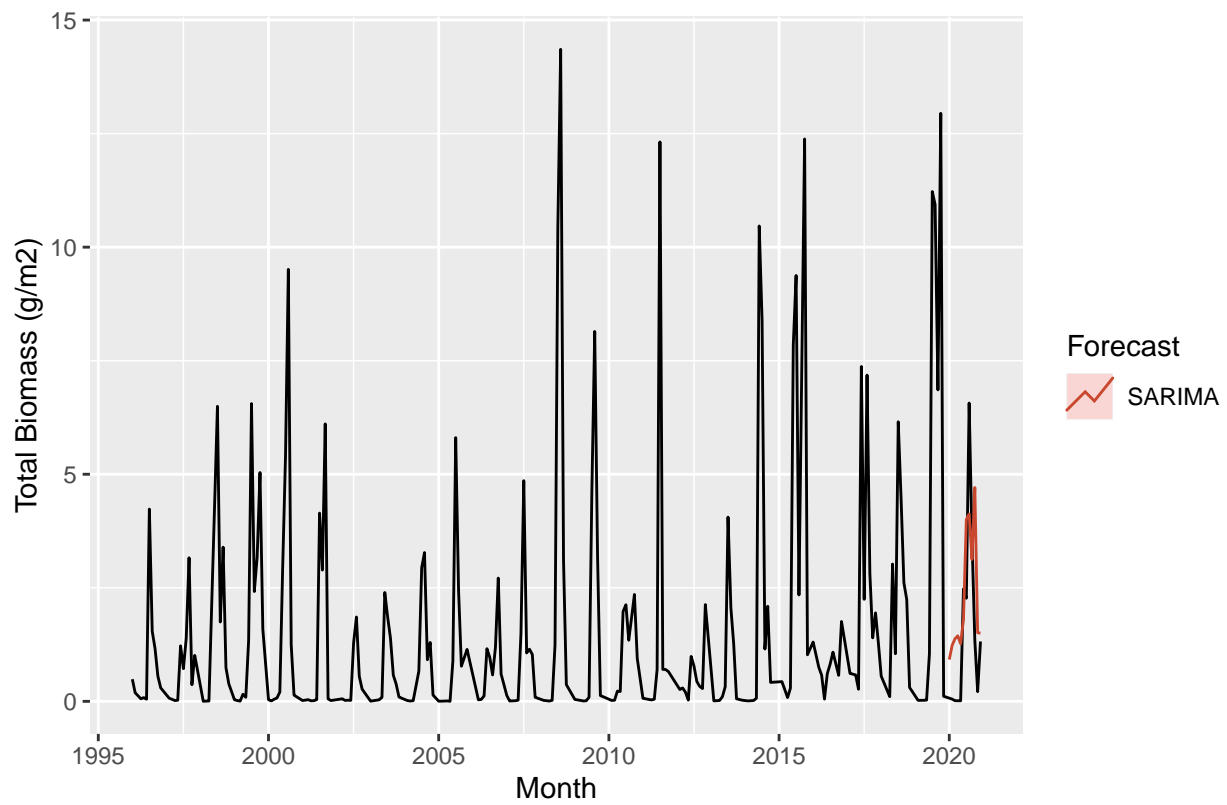
autoplot(ts_biomass_data) +
  autolayer(MEAN_seas, PI=FALSE, series="Mean") +
  autolayer(SNAIVE_seas, PI=FALSE, series="Naïve") +
  autolayer(SARIMA_for, PI=FALSE, series="SARIMA") +
  autolayer(SSES_seas$forecast, series="SSES") +
  autolayer(SS_for, PI=FALSE, series="BSM") +
  xlab("Month") + ylab("Electricity Retail Price ($/kWh)") +
  guides(colour=guide_legend(title="Forecast"))

```



```
autoplot(ts_biomass_data) +

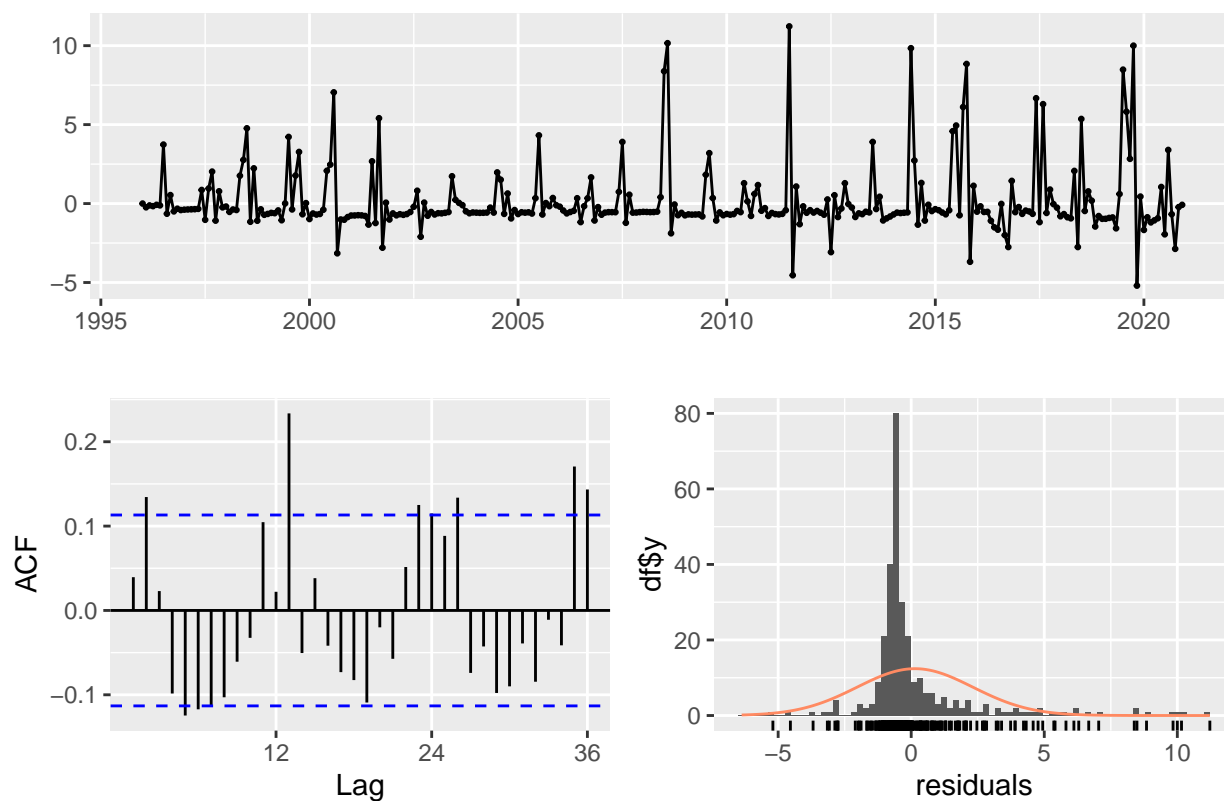
autolayer(SARIMA_for,PI=FALSE, series="SARIMA") +
  xlab("Month") + ylab("Total Biomass (g/m2)") +
  guides(colour=guide_legend(title="Forecast"))
```



Forecast

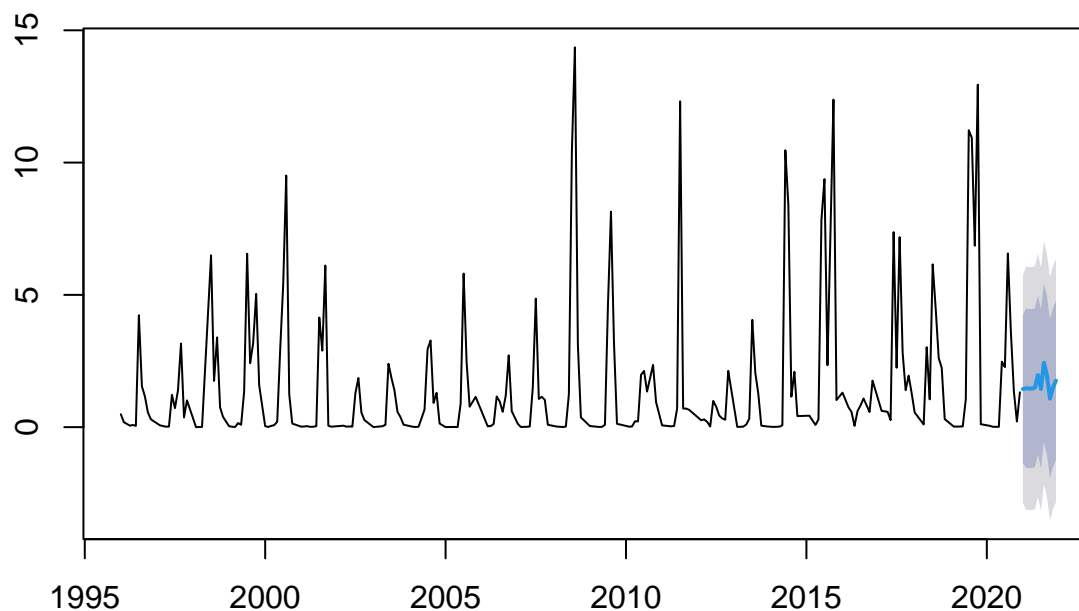
```
SARIMA_autofit_new <- auto.arima(ts_biomass_data)
checkresiduals(SARIMA_autofit_new)
```

Residuals from ARIMA(0,1,2)(0,0,1)[12]



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,2)(0,0,1)[12]
## Q* = 68.766, df = 21, p-value = 5.534e-07
##
## Model df: 3.   Total lags used: 24
SARIMA_for_new <- forecast(SARIMA_autofit_new,h=12)
plot(SARIMA_for_new)
```


Forecasts from ARIMA(0,1,2)(0,0,1)[12]



Use recent ten-year data to forecast

```
# Change the time span
```

```
# Transform to time series format
```

```
ts_biomass_data <- ts(  
  biomass_data_frame[169:300,2],  
  start=c(year(biomass_data_frame$Month[169]),month(biomass_data_frame$Month[169])),  
  frequency=12)
```

```
ts_biomass <- ts(  
  biomass_data_frame[169:288,2],  
  start=c(year(biomass_data_frame$Month[169]),month(biomass_data_frame$Month[169])),  
  frequency=12)
```

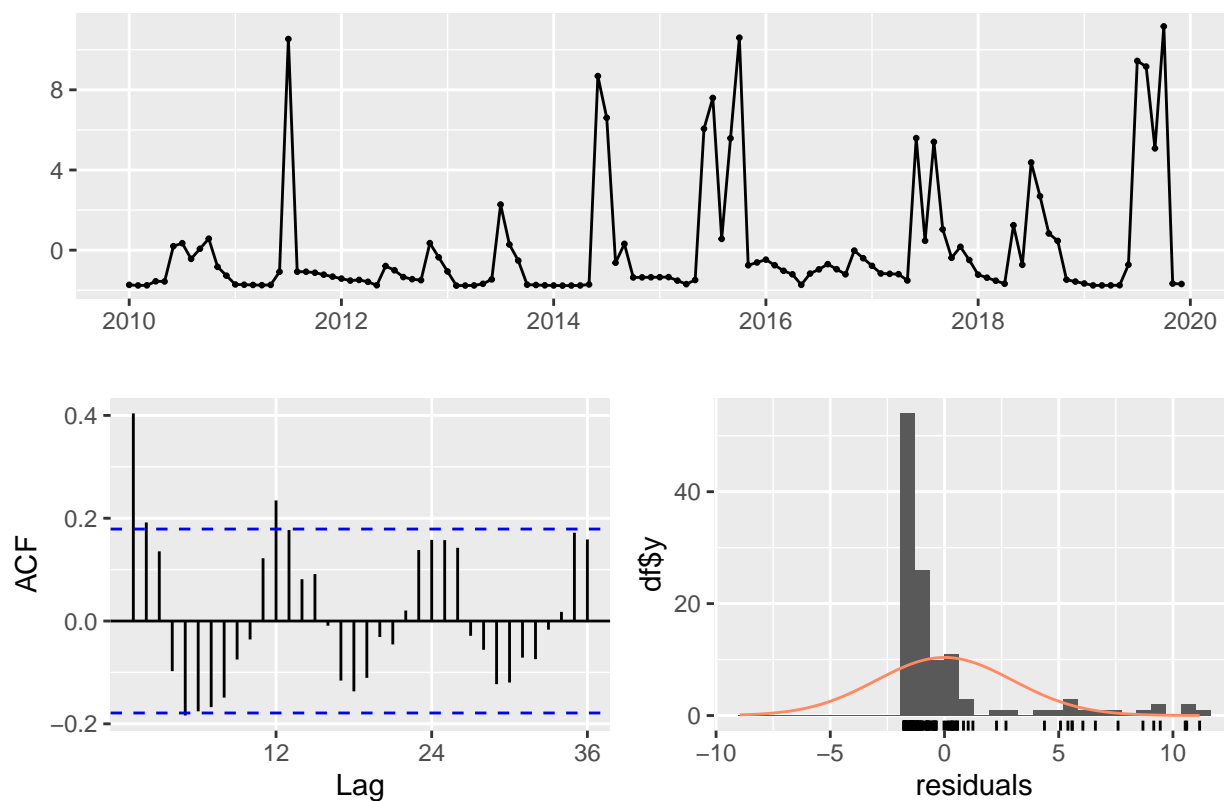
```
last_obs <- ts_biomass_data[121:132]
```

```
# Model 1: Arithmetic mean
```

```
# The meanf() has no holdout option
```

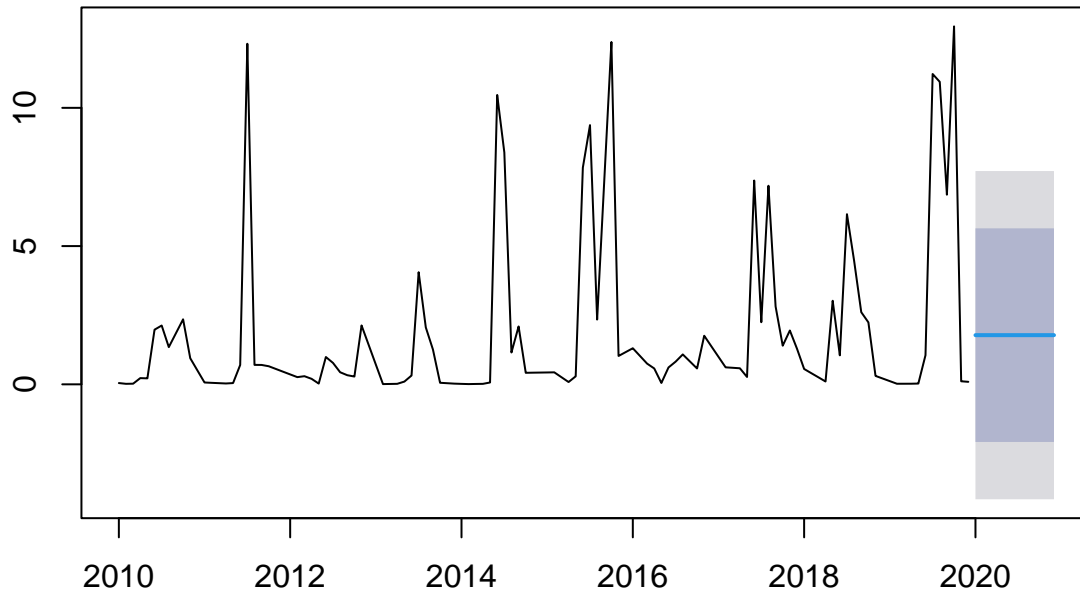
```
MEAN_seas <- meanf(y = ts_biomass, h = 12)  
checkresiduals(MEAN_seas)
```

Residuals from Mean



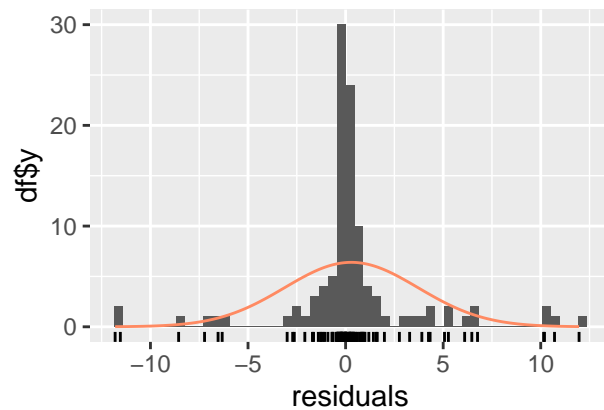
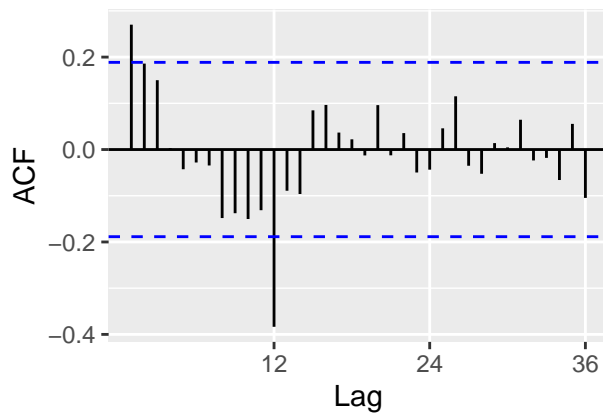
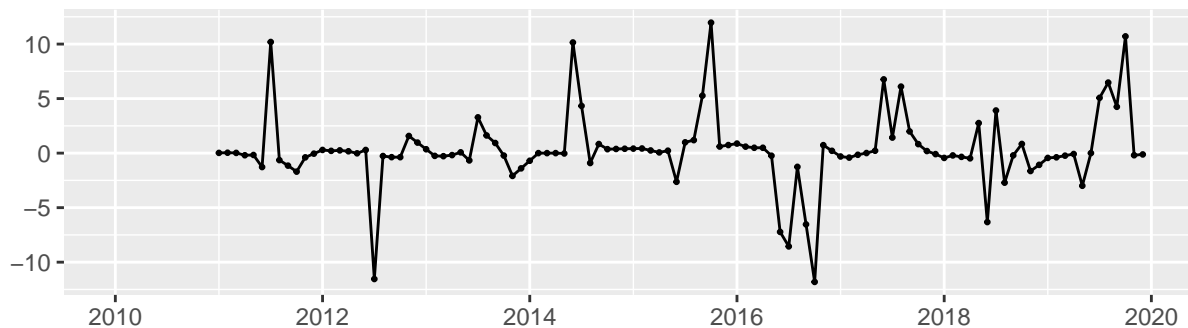
```
##
##  Ljung-Box test
##
## data:  Residuals from Mean
## Q* = 73.247, df = 23, p-value = 3.795e-07
##
## Model df: 1.   Total lags used: 24
plot(MEAN_seas)
```

Forecasts from Mean



```
# Model 2: Seasonal naive
SNAIVE_seas <- snaive(ts_biomass, h=12, holdout=FALSE)
checkresiduals(SNAIVE_seas)
```

Residuals from Seasonal naive method

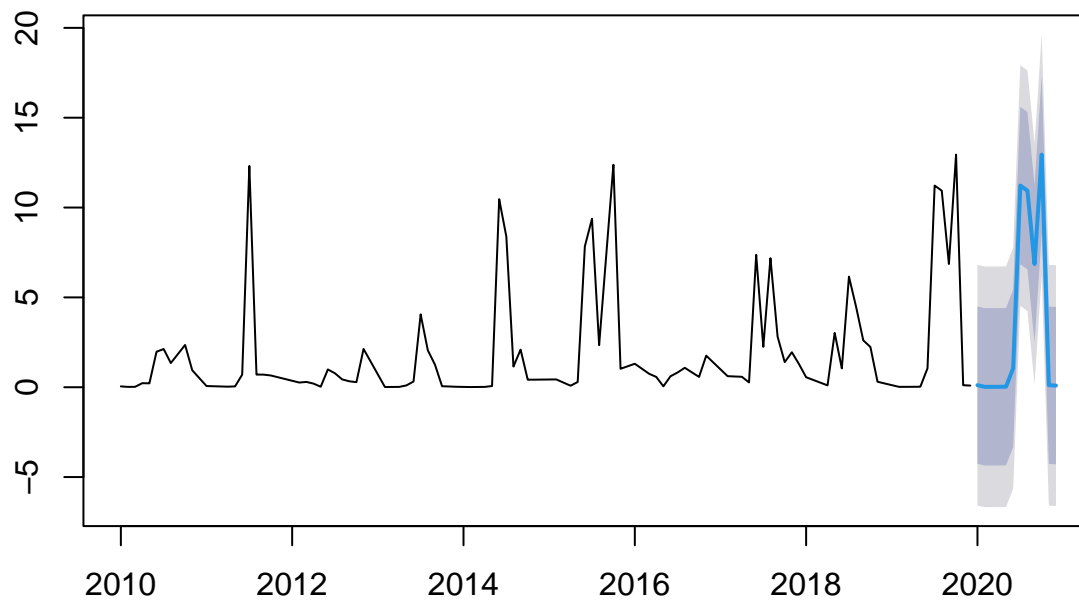


```
##
## Ljung-Box test
```

```
##
## data: Residuals from Seasonal naive method
## Q* = 49.511, df = 24, p-value = 0.001633
##
## Model df: 0. Total lags used: 24
```

```
plot(SNAIVE_seas)
```

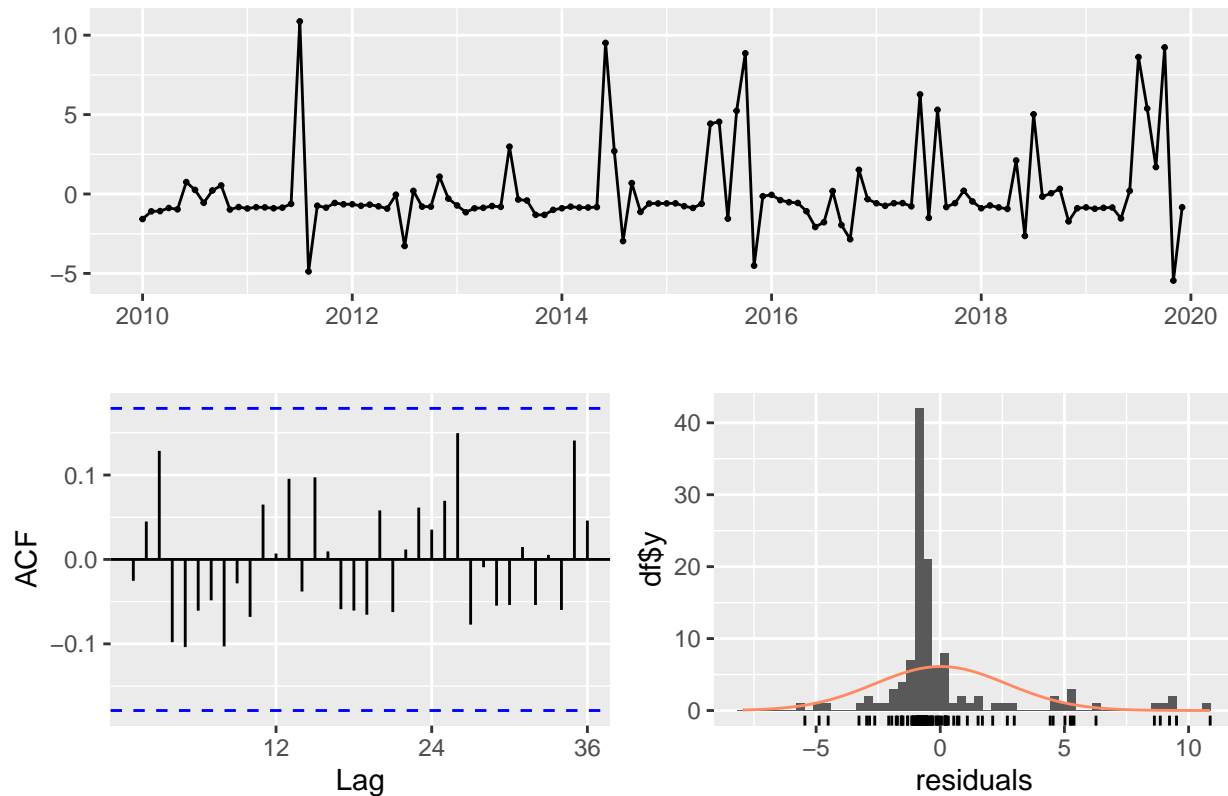
Forecasts from Seasonal naive method



```
# Model 3: SARIMA
```

```
SARIMA_autofit <- auto.arima(ts_biomass)
checkresiduals(SARIMA_autofit)
```

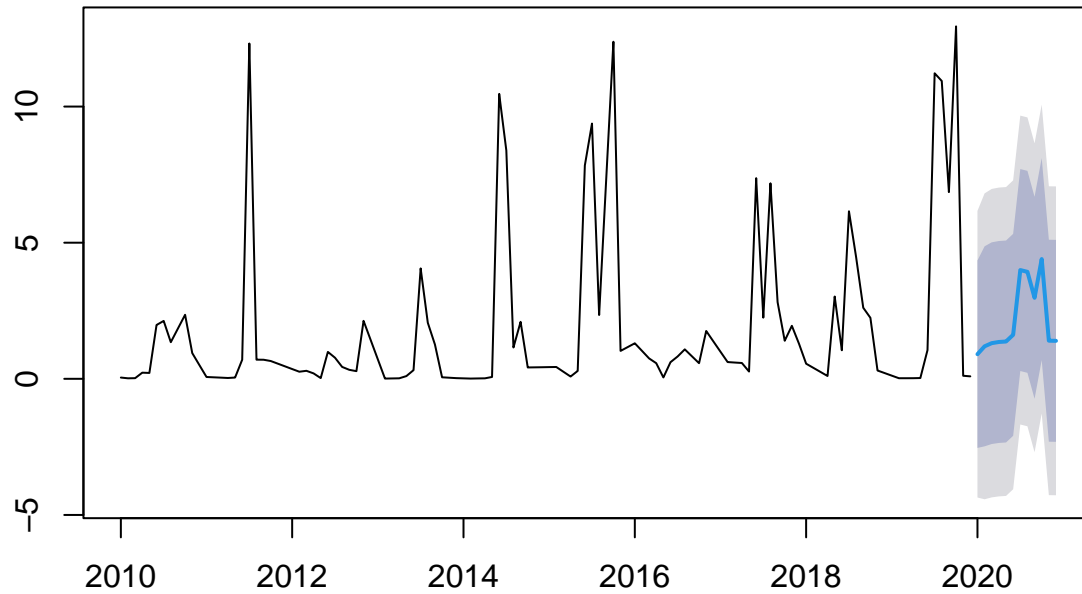
Residuals from ARIMA(1,0,0)(1,0,0)[12] with non-zero mean



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0)(1,0,0)[12] with non-zero mean
## Q* = 14.711, df = 22, p-value = 0.8743
##
## Model df: 2.   Total lags used: 24
```

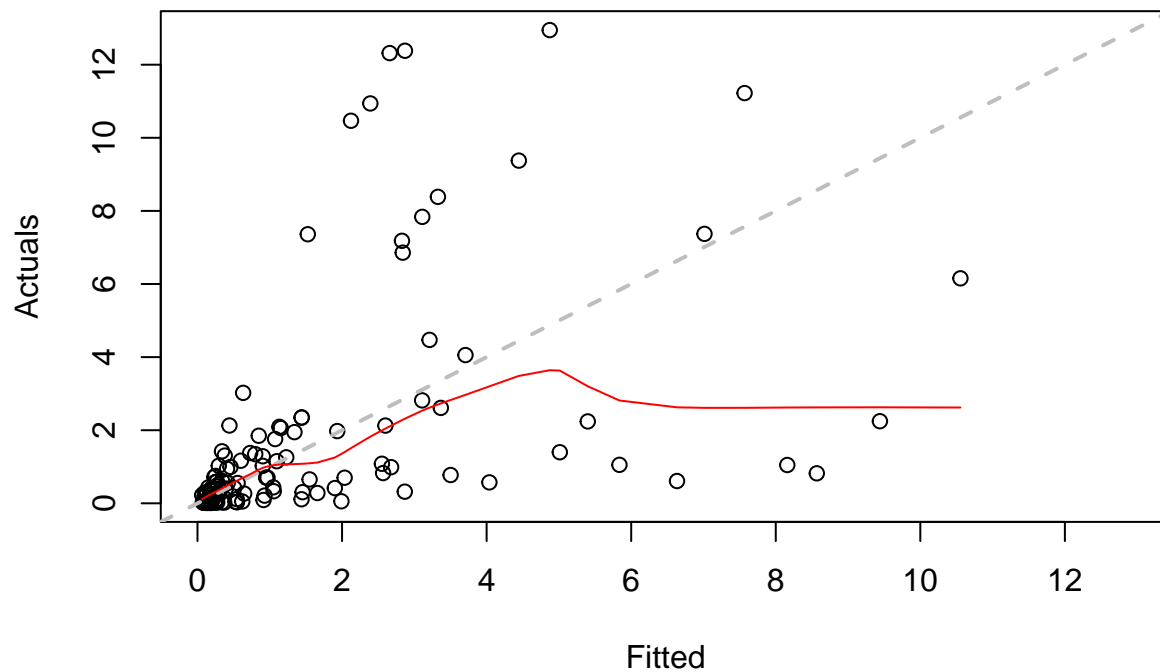
```
#Generating forecasts
#remember auto.arima does not call the forecast() internally so we need one more step
SARIMA_for <- forecast(SARIMA_autofit,h=12)
plot(SARIMA_for)
```

Forecasts from ARIMA(1,0,0)(1,0,0)[12] with non-zero mean

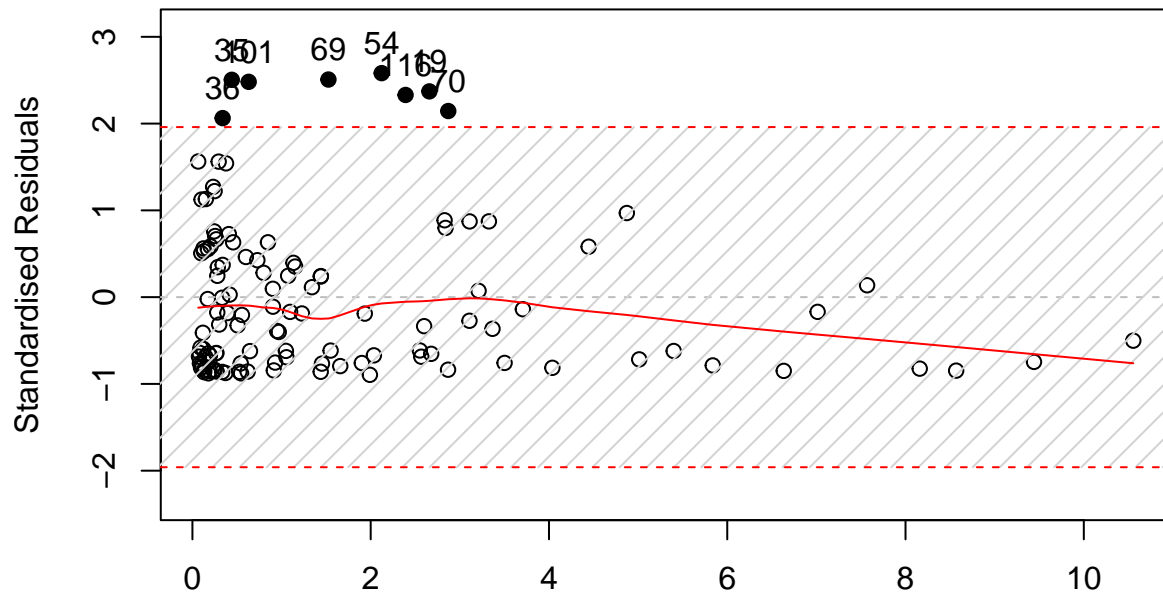


```
# Model 4: SS Exponential smoothing
SSES_seas <- es(ts_biomass,model="ZZZ",h=12,holdout=FALSE)
plot(SSES_seas)
```

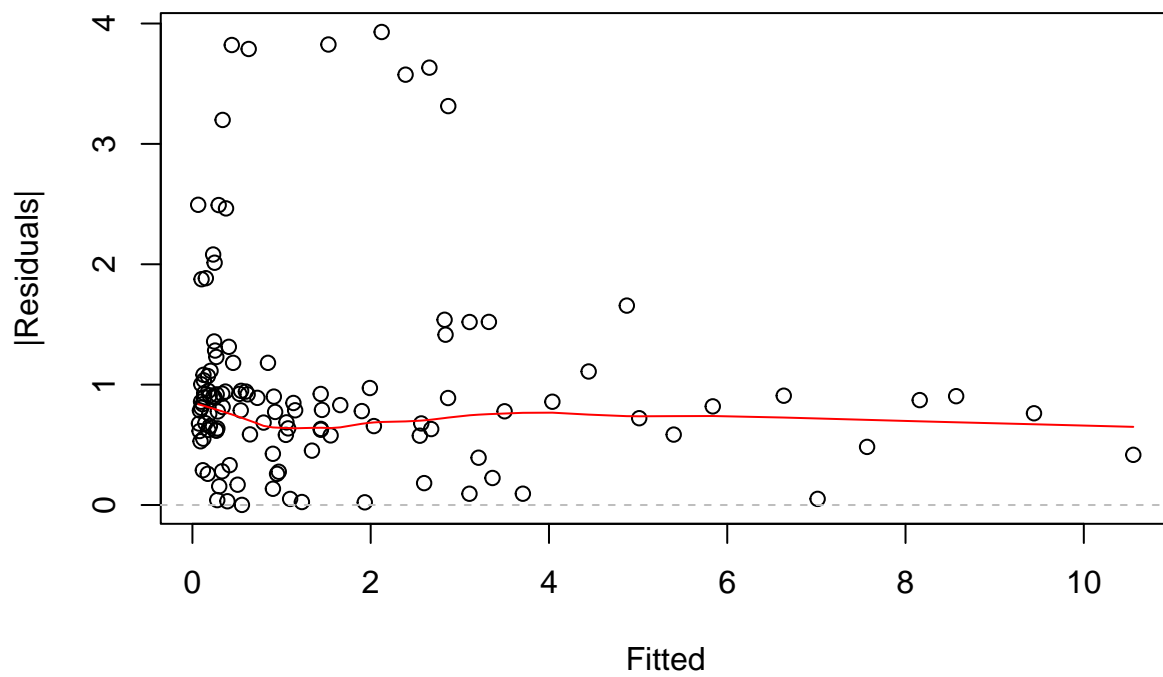
Actuals vs Fitted



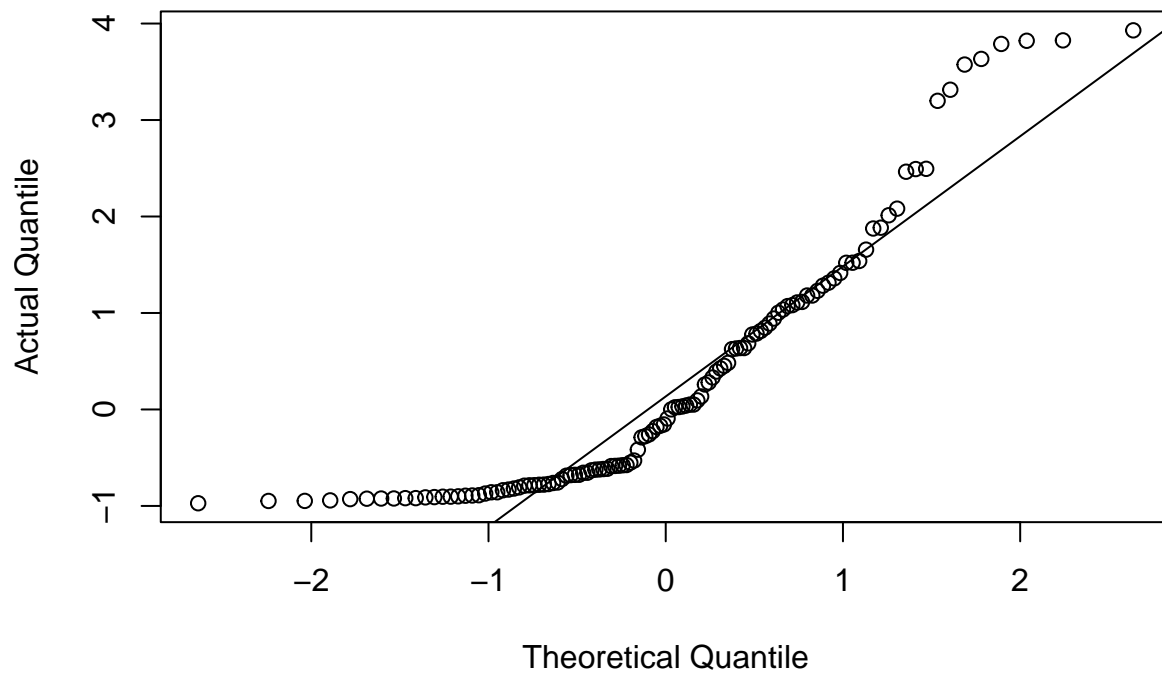
Standardised Residuals vs Fitted



|Residuals| vs Fitted

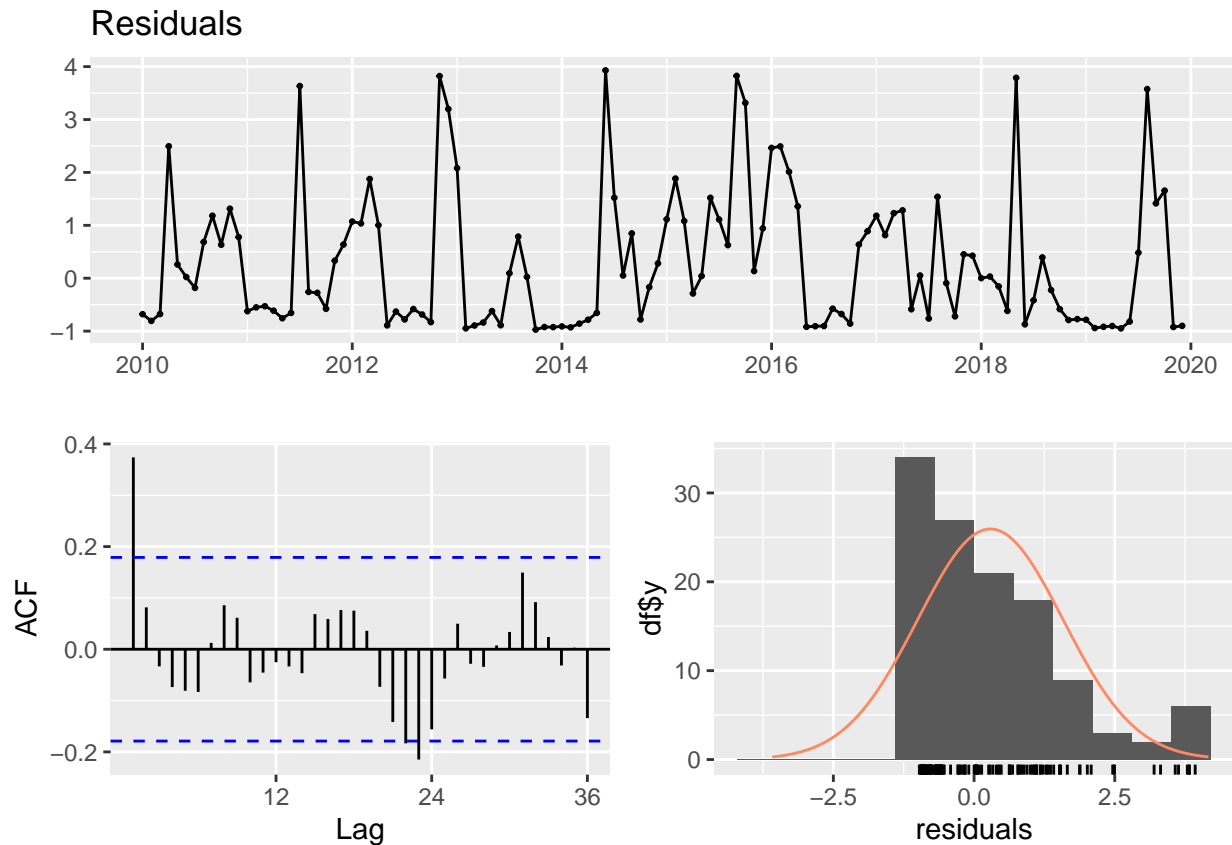


QQ plot of Normal distribution



```
checkresiduals(SSES_seas)
```

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

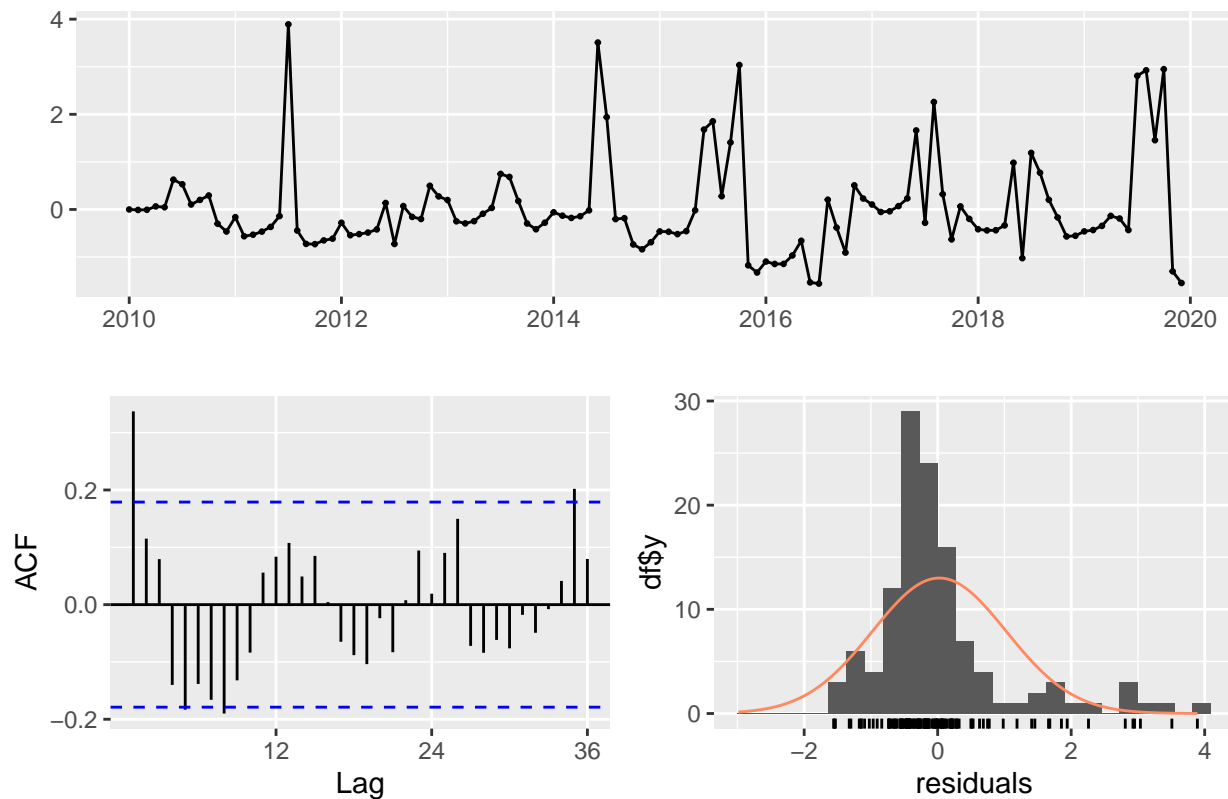



```
# Model 5: SS with StructTS()

SS_seas <- StructTS(ts_biomass,
                    type="BSM",fixed=c(0,0.001,0.3,NA)) #this function has convergence issues
checkresiduals(SS_seas)

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

Residuals from StructTS



```
#Generating forecasts
# StructTS() does not call the forecast() internally so we need one more step
SS_for <- forecast(SS_seas,h=12)
plot(SS_for)
```

Forecasts from Basic structural model

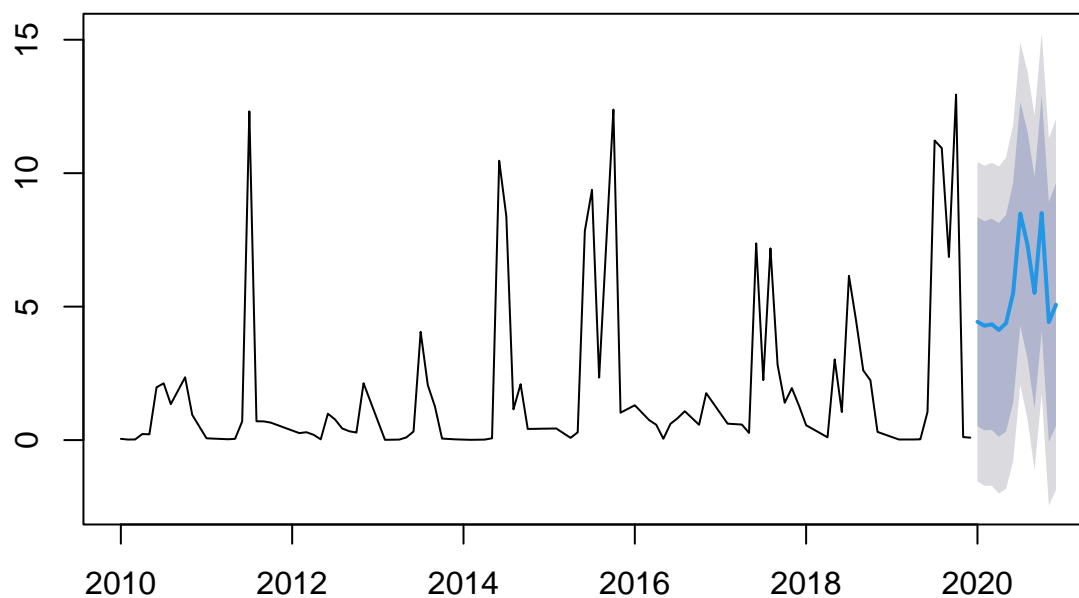


Table 2: Forecast Accuracy for Seasonal Data

	ME	RMSE	MAE	MPE	MAPE
MEAN	-0.28323	1.93838	1.58284	-3817.5199	3846.3960
SNAIVE	-2.12360	4.54636	2.58677	-122.4177	164.9659
SARIMA	-0.65751	1.55783	1.34461	-2824.6163	2839.9915
SSES	-1.10662	2.85708	1.89745	-919.7008	937.4871
BSM	-4.03626	4.34690	4.03626	-9410.6202	9410.6202

```

#Model 1: Arithmetic mean
MEAN_scores <- accuracy(MEAN_seas$mean,last_obs) #store the performance metrics

#Model 2: Seasonal naive
SNAIVE_scores <- accuracy(SNAIVE_seas$mean,last_obs)

# Model 3: SARIMA
SARIMA_scores <- accuracy(SARIMA_for$mean,last_obs)

# Model 4: SSES
SSES_scores <- accuracy(SSES_seas$forecast,last_obs)

# Model 5: BSM
SS_scores <- accuracy(SS_for$mean,last_obs)

#create data frame
seas_scores <- as.data.frame(rbind(MEAN_scores, SNAIVE_scores, SARIMA_scores,SSES_scores,SS_scores))
row.names(seas_scores) <- c("MEAN", "SNAIVE", "SARIMA", "SSES", "BSM")

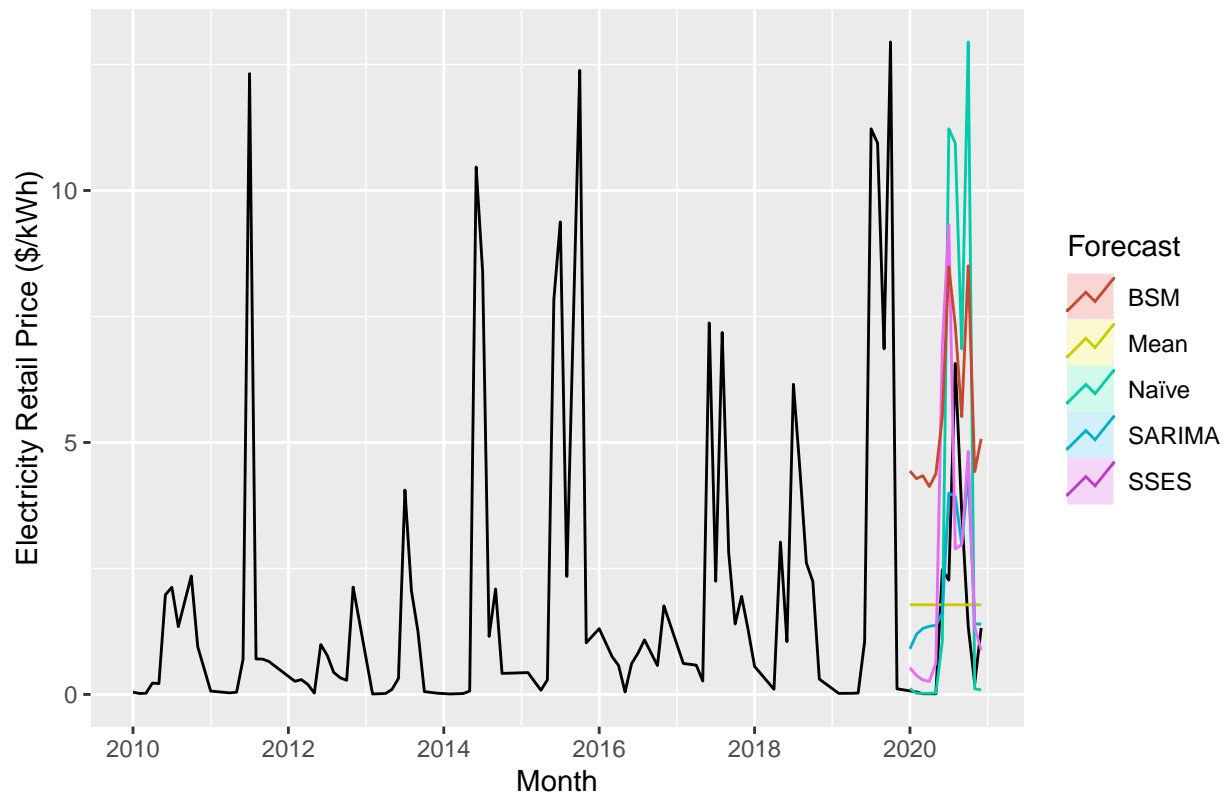
#choose model with lowest RMSE
best_model_index <- which.min(seas_scores[, "RMSE"])
cat("The best model by RMSE is:", row.names(seas_scores[best_model_index,]))

## The best model by RMSE is: SARIMA

kbl(seas_scores,
     caption = "Forecast Accuracy for Seasonal Data",
     digits = array(5,ncol(seas_scores))) %>%
  kable_styling(full_width = FALSE, position = "center") %>%
  #highlight model with lowest RMSE
  kable_styling(latex_options="striped", stripe_index = which.min(seas_scores[, "RMSE"]))

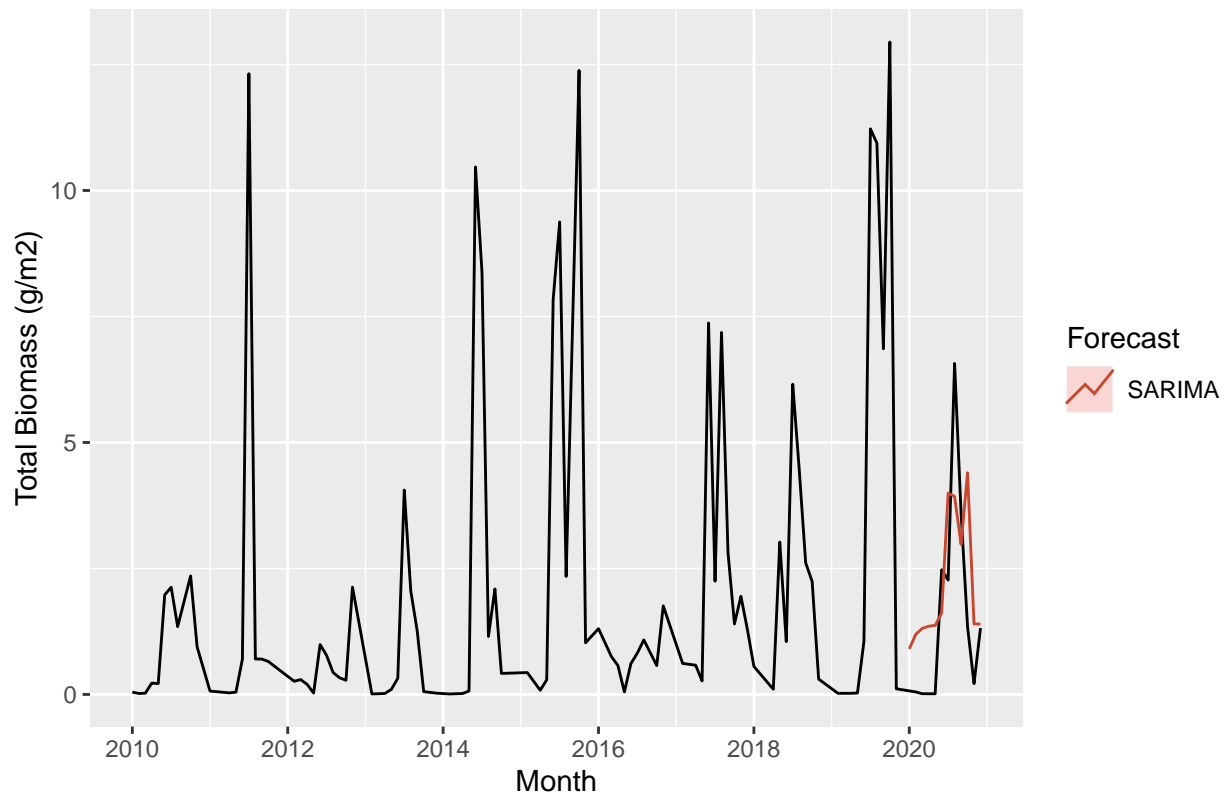
autoplot(ts_biomass_data) +
  autolayer(MEAN_seas, PI=FALSE, series="Mean") +
  autolayer(SNAIVE_seas, PI=FALSE, series="Naïve") +
  autolayer(SARIMA_for, PI=FALSE, series="SARIMA") +
  autolayer(SSES_seas$forecast, series="SSES") +
  autolayer(SS_for, PI=FALSE, series="BSM") +
  xlab("Month") + ylab("Electricity Retail Price ($/kWh)") +
  guides(colour=guide_legend(title="Forecast"))

```



```
autoplot(ts_biomass_data) +

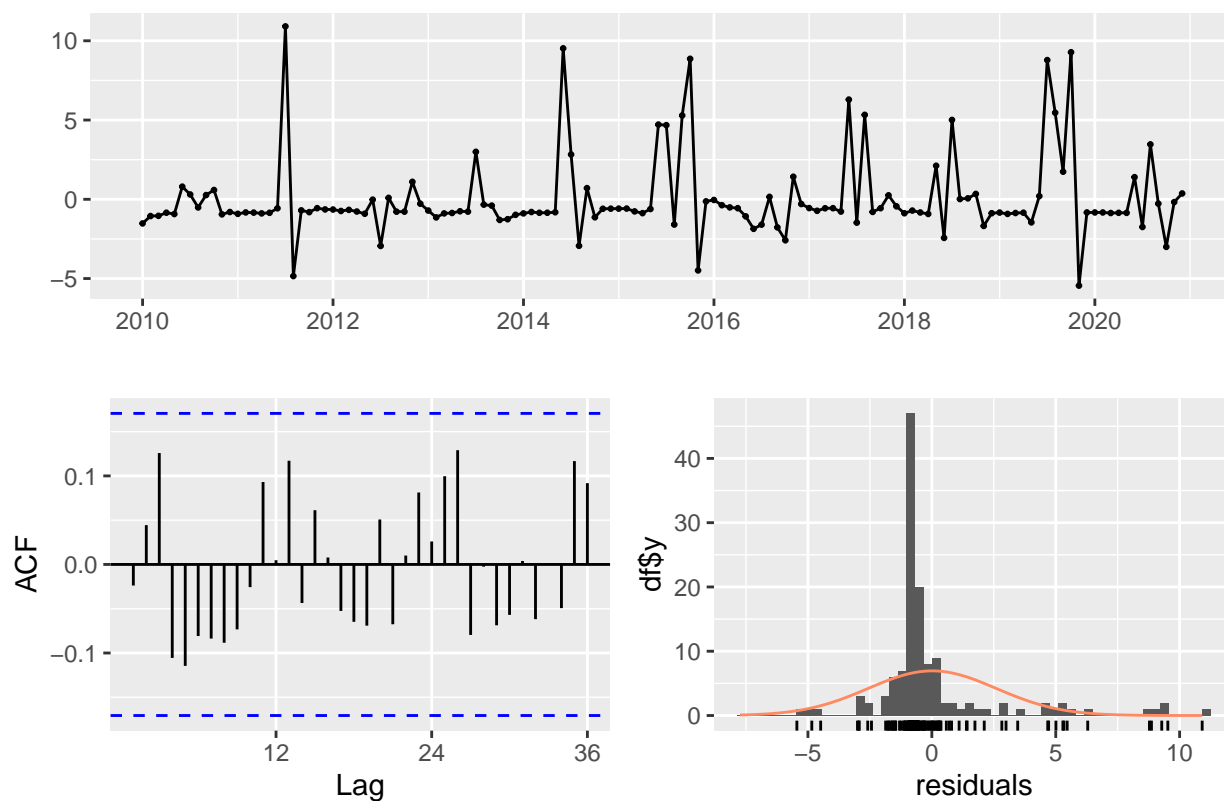
autolayer(SARIMA_for,PI=FALSE, series="SARIMA") +
  xlab("Month") + ylab("Total Biomass (g/m2)") +
  guides(colour=guide_legend(title="Forecast"))
```



```
# Forecast
```

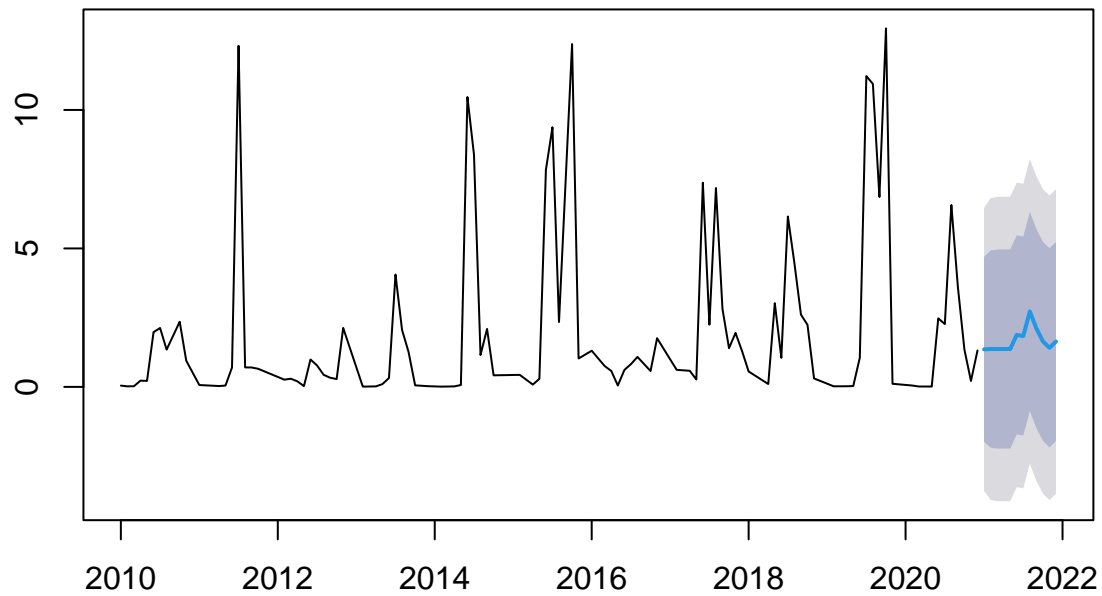
```
SARIMA_autofit_new <- auto.arima(ts_biomass_data)  
checkresiduals(SARIMA_autofit_new)
```

Residuals from ARIMA(1,0,0)(1,0,0)[12] with non-zero mean



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0)(1,0,0)[12] with non-zero mean
## Q* = 18.099, df = 22, p-value = 0.7001
##
## Model df: 2.   Total lags used: 24
SARIMA_for_new <- forecast(SARIMA_autofit_new,h=12)
plot(SARIMA_for_new)
```

Forecasts from ARIMA(1,0,0)(1,0,0)[12] with non-zero mean



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.