Fatalities in Belgium

25, July, 2019 Yen Chun, Liu

Package used

library(readxl)

library(knitr)

library(fpp2)

library(tseries)

library(portes)

Read in the data

```
data <- read_excel("DataSets.xlsx", sheet="Fatalities_m")
```

Data informaiton

The data set Fatalities_m contains the monthly number of road fatalities in Belgium from January 1995 to December 2017.

According to instruction we will split the data in a training set from January 2001 up to December 2015 and a test set from January 2016 up to December 2017. Use the training set for estimation of the methods/models, and use the test set for assessing the forecast accuracy.

```
head(data)

## # A tibble: 6 x 2
##   Date                Fatalities
##   <dttm>                   <dbl>
## 1 1995-01-01 00:00:00        108
## 2 1995-02-01 00:00:00        106
## 3 1995-03-01 00:00:00        129
## 4 1995-04-01 00:00:00        107
## 5 1995-05-01 00:00:00        134
## 6 1995-06-01 00:00:00        113

tail(data)

## # A tibble: 6 x 2
##   Date                Fatalities
##   <dttm>                   <dbl>
## 1 2017-07-01 00:00:00         53
## 2 2017-08-01 00:00:00         47
## 3 2017-09-01 00:00:00         59
## 4 2017-10-01 00:00:00         63
## 5 2017-11-01 00:00:00         61
## 6 2017-12-01 00:00:00         41
```

Cut data

Save data between Jan. 2001 to Dec. 2017

```
data <- data[c(73:276),]
```

Change to time series format

```
to <- ts(data[,2], frequency = 12, start=c(2001))
```

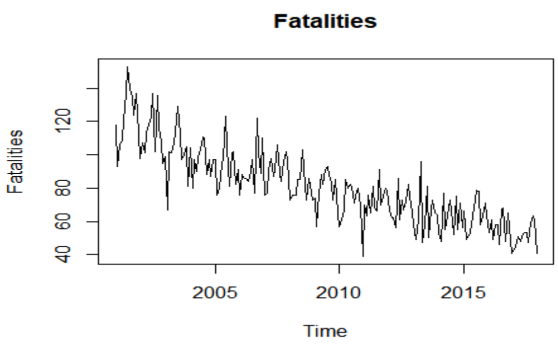
Split train and test

```
train <- window(to, start= c(2001,1), end= c(2015,12))
test <- window(to, start= c(2016,1),end= c(2017,12))
h = length(test)
```

Line plot

From the plot we can see that there's a downward trend from 2001 to 2017.The range of sudden drop and increase have decrease over years, where just by looking at the years before 2005 and after 2010. There's a little downward ladder pattern, after a sudden drop comes with a period of flatter fluctuation patterns, like after 2005 and 2011. But we can't see seasonality effect and monthly.

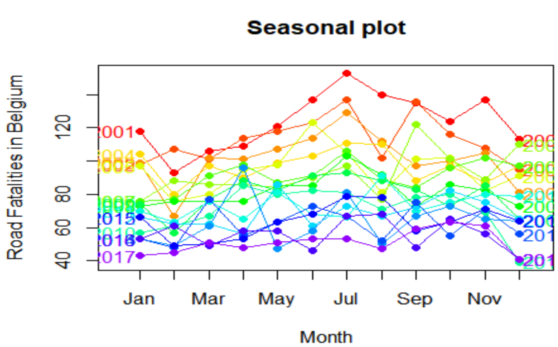
```
plot(to, main="Fatalities")
```

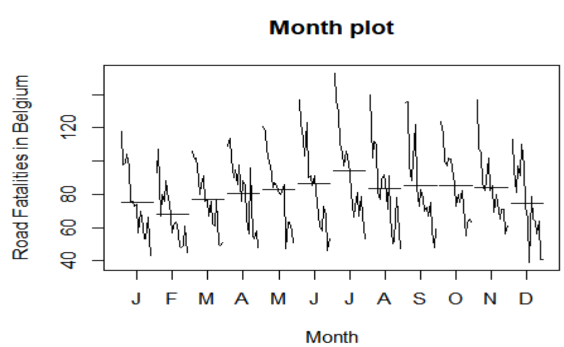


Season and month plot

From the season plot we can't tell much, before 2005 are likely over 80 vice versa. From the month plot we can see that from March starts to increase and at July reaches the highest. Feb. is the lowest.

```
seasonplot(to, year.labels=TRUE, year.labels.left=TRUE,
           main="Seasonal plot",
           ylab="Road Fatalities in Belgium",col=rainbow(21), pch=19)
```





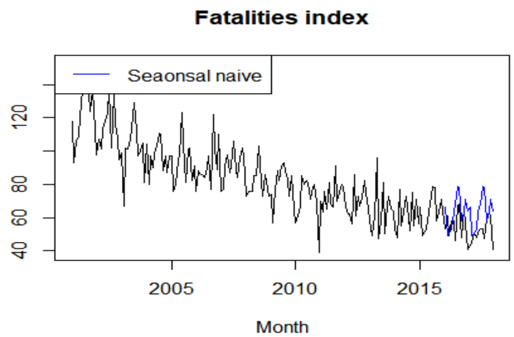

```
monthplot(to, main="Month plot", ylab = "Road Fatalities in Belgium",
          xlab="Month", type="l")
```

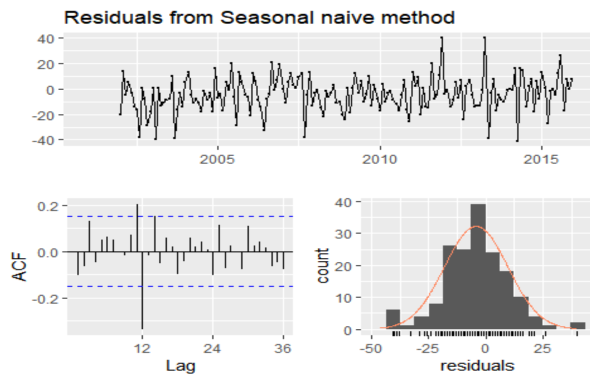
Seasonal naive method

With the plot under we can see that seasonal naive predictions does not have a down ward trend. However to judge the model performance we have to compare with other models by RMSE, MAE, MAPE and MAsE.

For white noise series, we expect each autocorrelation to be close to zero. Of course, they will not be exactly equal to zero as there is some random variation. For a white noise series, we expect 95% of the spikes in the ACF to lie within the blue dashed lines above. If one or more large spikes are outside these bounds, or if substantially more than 5% of spikes are outside these bounds, then the series is probably not white noise. If Ljung-Box test p-value is above 0.05 means accept as white noise. The residual diagnostics show that after lag 13 the residuals of this method are not white noise. There is still information not captured.

```
f1 <-snaive(train, h = h)
plot(to,main="Fatalities index", ylab="",xlab="Month")
lines(f1$mean,col=4)
legend("topleft",lty=1,col=c(4),legend=c("Seaonsal naive"))
```





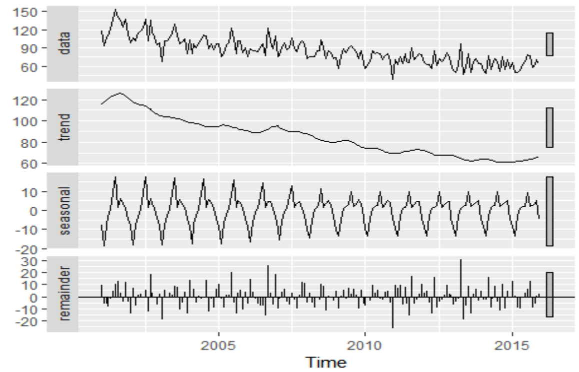

```
res <- residuals(f1)
checkresiduals(f1)
```

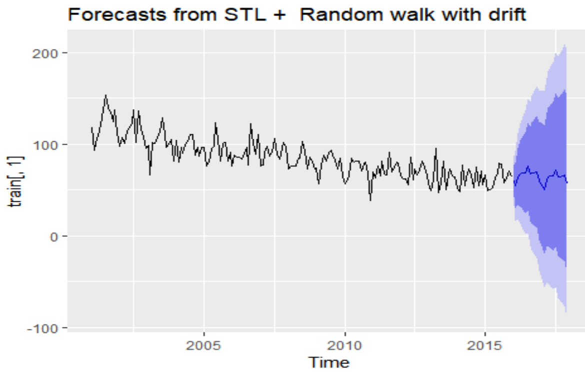
```
##
##  Ljung-Box test
##
## data:  Residuals from Seasonal naive method
## Q* = 47.498, df = 24, p-value = 0.002911
##
## Model df: 0.   Total lags used: 24

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags statistic df      p-value
##     1  1.745112  1 0.1864923966
##     5  6.230622  5 0.2844208452
##     9  7.474574  9 0.5878347664
##    13 36.641125 13 0.0004714947
##    17 42.108483 17 0.0006469117
##    21 45.064727 21 0.0016984180

accuracy(f1)[,c(2,3,5,6)]

##      RMSE      MAE     MAPE     MASE
## 14.50821 11.11905 14.50629  1.00000
```

STL decomposition

The two main parameters to be chosen when using STL are the trend-cycle window (t.window) and the seasonal window (s.window). These control how rapidly the trend-cycle and seasonal components can change. Smaller values allow for more rapid changes. Both t.window and s.window should be odd numbers; The residual diagnostics show that the residuals of this method are not white noise.

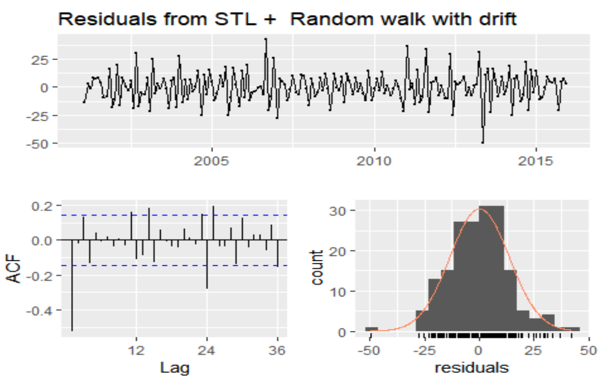
```
f2 <- forecast(stl(train[,1],t.window = 15, s.window=13), method="rwdrift",h=h)
autoplot(stl(train[,1],t.window = 15, s.window=13), method="rwdrift",h=h)
```






```r
autoplot(f2)
```

```r
res <- residuals(f2)
checkresiduals(f2)

## Warning in checkresiduals(f2): The fitted degrees of freedom is based on
## the model used for the seasonally adjusted data.
```




```
##
##  Ljung-Box test
##
## data:  Residuals from STL +  Random walk with drift
## Q* = 99.772, df = 23, p-value = 1.542e-11
##
## Model df: 1.   Total lags used: 24
```

```r
res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags statistic df      p-value
##     1  49.87563  1 1.638023e-12
##     5  56.95103  5 5.175860e-11
##     9  57.28739  9 4.451967e-09
##    13  66.34926 13 3.745482e-09
##    17  76.92900 17 1.339329e-09
##    21  78.46831 21 1.453522e-08

accuracy(f2)[,c(2,3,5,6)]

##       RMSE        MAE       MAPE       MASE
## 13.5622639 10.5673662 13.4007784  0.9503841
```

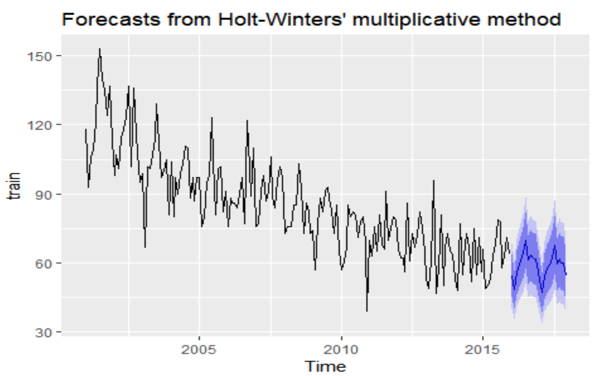
Holt-Winters method

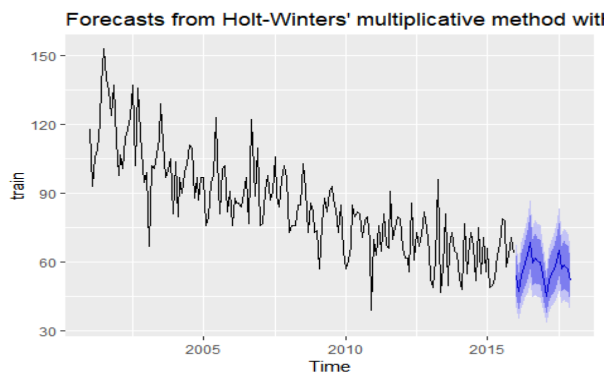
There are two variations to this method that differ in the nature of the seasonal component. The additive method is preferred when the seasonal variations are roughly constant through the series, while the multiplicative method is preferred when the seasonal variations are changing proportional to the level of the series.

We will apply Holt-Winters method with both additive and multiplicative seasonality and with/without exponential and damped or not to forecast, to see which one have the best accuracy.

The Holt-Winters multiplicative method with exponential trend have the best performance compare to others. Where it also has a acceptable result of residual diagnostics.

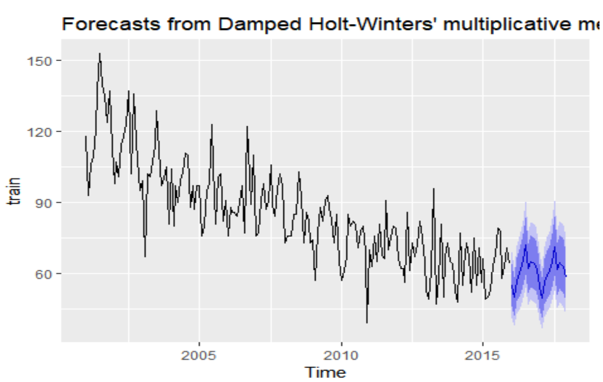
```r
f3 <- forecast(hw(train,seasonal="mult", h=h), method="rwdrift", h=h)
autoplot(f3)
```

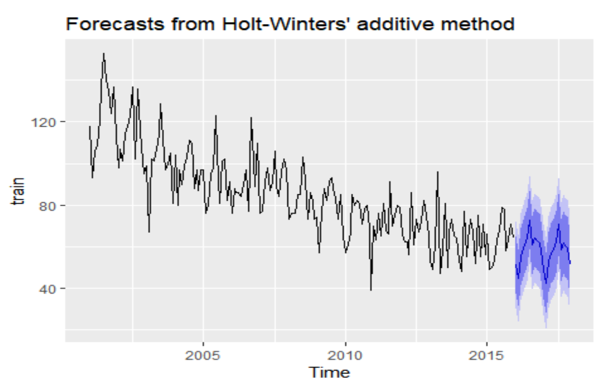





```r
f4 <- forecast(hw(train,seasonal="mult",exponential=TRUE, h=h), method="rwdrift", h=h)
autoplot(f4)
```

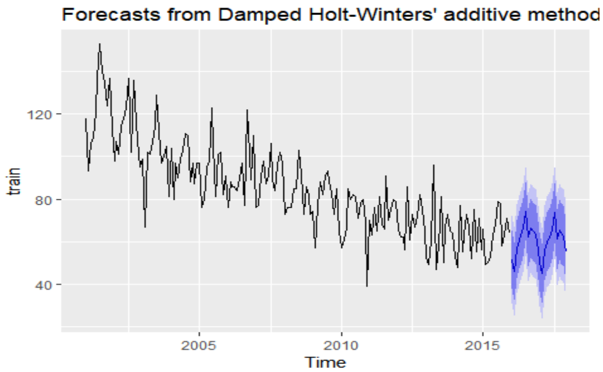
```r
f5 <- forecast(hw(train,seasonal="mult",exponential=TRUE, damped=TRUE, h=h), method="rwdrift", h=h)
autoplot(f5)
```



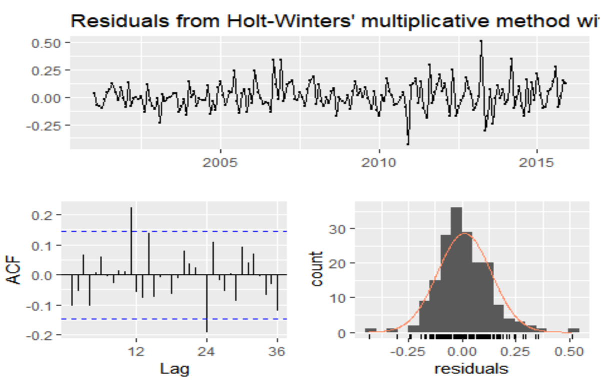



```r
f6 <- forecast(hw(train,seasonal="addi", h=h), method="rwdrift", h=h)
autoplot(f6)
```

```r
f7 <- forecast(hw(train,seasonal="addi",damped=TRUE, h=h), method="rwdrift", h=h)
autoplot(f7)
```




```r
a_fc3 <- accuracy(f3)[,c(2,3,5,6)]
a_fc4 <- accuracy(f4)[,c(2,3,5,6)]
a_fc5 <- accuracy(f5)[,c(2,3,5,6)]
a_fc6 <- accuracy(f6)[,c(2,3,5,6)]
a_fc7 <- accuracy(f7)[,c(2,3,5,6)]
acc <- rbind(a_fc3, a_fc4, a_fc5, a_fc6, a_fc7)
rownames(acc) <- c("a_fc3", "a_fc4", "a_fc5", "a_fc6", "a_fc7")
acc

##             RMSE      MAE     MAPE      MASE
## a_fc3  9.760150 7.482337 9.372578 0.6729296
## a_fc4  9.630672 7.413138 9.292348 0.6667062
## a_fc5  9.706022 7.548799 9.624697 0.6789069
## a_fc6 10.134481 7.729977 9.606757 0.6952014
## a_fc7 10.050553 7.780704 9.864922 0.6997635

res <- residuals(f4)
checkresiduals(f4)
```




```r
##
##  Ljung-Box test
##
## data:  Residuals from Holt-Winters' multiplicative method with exponential trend
## Q* = 33.023, df = 8, p-value = 6.1e-05
##
## Model df: 16.   Total lags used: 24

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags statistic df   p-value
##     1  2.048928  1 0.1523134
##     5  5.408440  5 0.3680902
##     9  6.285037  9 0.7110863
```

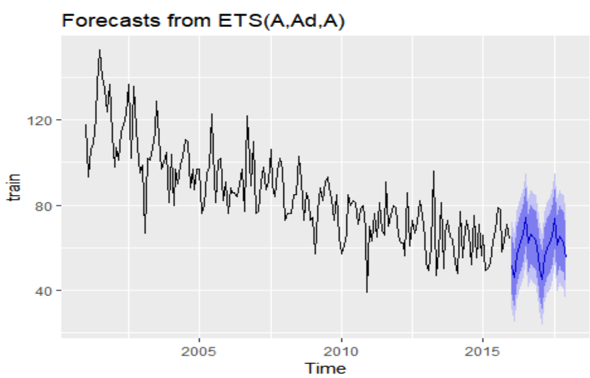
```
##    13 17.860272 13 0.1628973
##    17 22.709971 17 0.1589329
##    21 25.141153 21 0.2411024
```

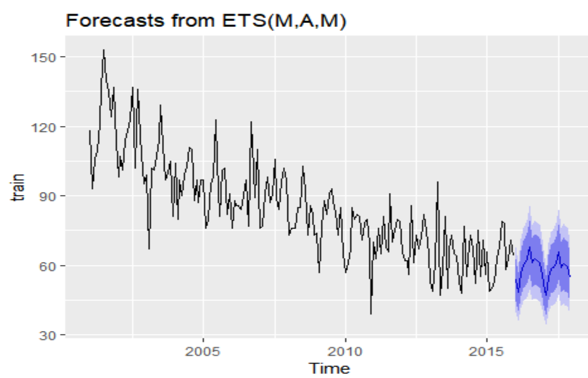
ETS

ETS (Error, Trend, Seasonal) method is an approach method for forecasting time series. Based on the properties of the data, we estimate several ETS models with a trend and a seasonal component. We consider additive and multiplicative errors, and trends with and without damping. The first letter denotes the error type ("A", "M" or "Z"); the second letter denotes the trend type ("N","A","M" or "Z"); the third letter denotes the season type ("N","A","M" or "Z"). In all cases, "N"=none, "A"=additive, "M"=multiplicative and "Z"=automatically selected.

Due to the fact that we do see a continues pattern of fluctuation with a downward trend. We will try MMM and MAM with damped and non damped, to compare with auto ets which auto ets choose among best AIC, but not accuracy. ETS MMM model have the best accuracy among ETS model for this situation. The residual diagnostics also has an acceptable result.

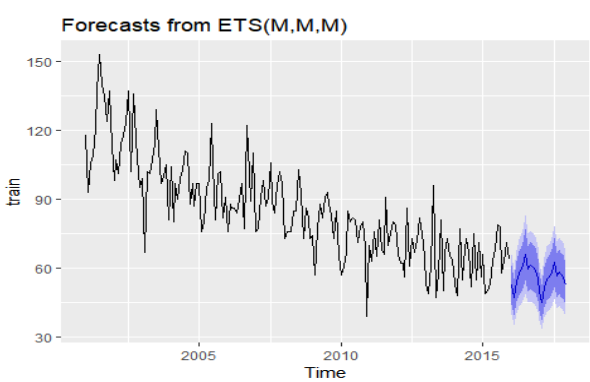
```r
f8 <- forecast(ets(train, model = "ZZZ"), method="rwdrift", h=h)
autoplot(f8)
```

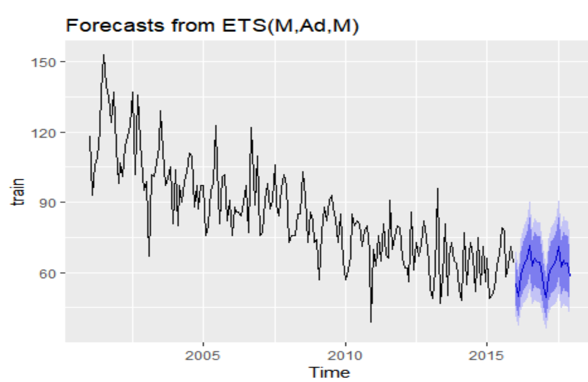





```r
f9 <- forecast(ets(train, model = "MAM"), method="rwdrift", h=h)
autoplot(f9)
```

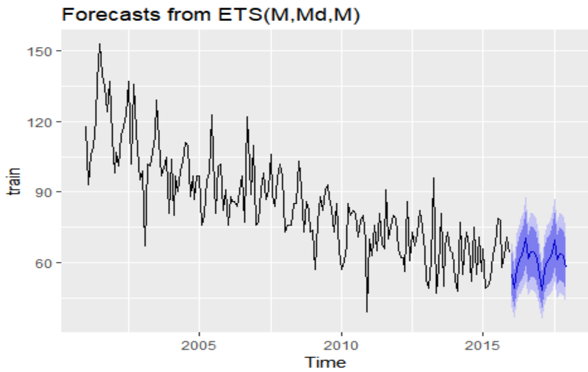
```r
f10 <- forecast(ets(train, model = "MMM"), method="rwdrift", h=h)
autoplot(f10)
```



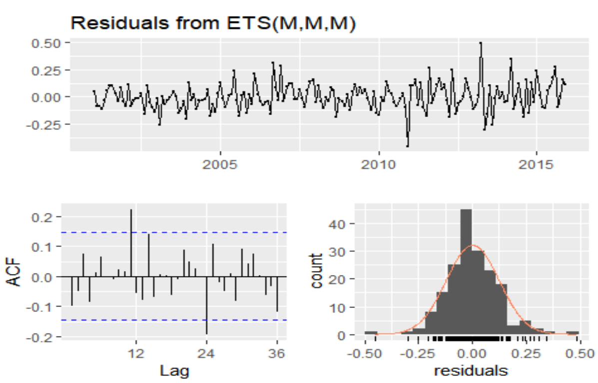



```r
f11 <- forecast(ets(train, model = "MAM",damped=TRUE), method="rwdrift", h=h)
autoplot(f11)
```

```r
f12 <- forecast(ets(train, model = "MMM",damped=TRUE), method="rwdrift", h=h)
autoplot(f12)
```




```
a_fc8 <- accuracy(f8)[,c(2,3,5,6)]
a_fc9 <- accuracy(f9)[,c(2,3,5,6)]
a_fc10 <- accuracy(f10)[,c(2,3,5,6)]
a_fc11 <- accuracy(f11)[,c(2,3,5,6)]
a_fc12 <- accuracy(f12)[,c(2,3,5,6)]
acc <- rbind(a_fc8, a_fc9, a_fc10, a_fc11, a_fc12)
rownames(acc) <- c("a_fc8", "a_fc9", "a_fc10", "a_fc11", "a_fc12")
acc

##              RMSE      MAE     MAPE      MASE
## a_fc8  10.050553 7.780704 9.864922 0.6997635
## a_fc9   9.880909 7.579046 9.414393 0.6816272
## a_fc10  9.726072 7.544467 9.471531 0.6785174
## a_fc11  9.908800 7.813165 9.927302 0.7026829
## a_fc12  9.750924 7.611987 9.693071 0.6845898

res <- residuals(f10)
checkresiduals(f10)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ETS(M,M,M)
## Q* = 32.515, df = 8, p-value = 7.53e-05
##
## Model df: 16.   Total lags used: 24

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags statistic df   p-value
##     1  1.777794  1 0.1824204
##     5  4.669414  5 0.4575412
##     9  5.576521  9 0.7814388
```

```
##    13 17.014918 13 0.1986230
##    17 21.764656 17 0.1939400
##    21 24.583257 21 0.2656642
```

ARIMA

The ACF shows that nonstationarity is mainly caused by trend, and to a lesser extent by the seasonality. The auto.arima procedure results in an ARIMA(0,1,1)(2,0,0) model. This model shows satisfactory diagnostics. We will now explore some variations starting from this model, and check model fit and forecast accuracy. The code allows us to gather the results of several models.

The first difference applyed suggest to take one difference, then the seasonal difference suggest zero differences.

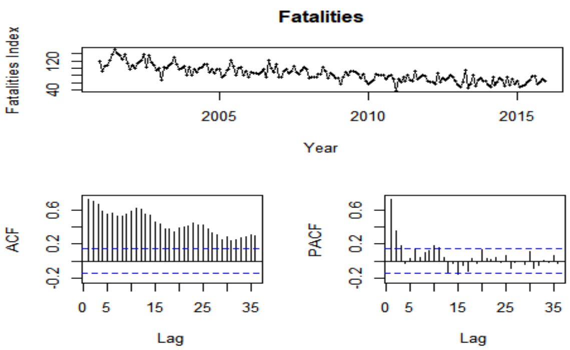
211 011 best AIC

411 112 best MASE and RMSE on the training set

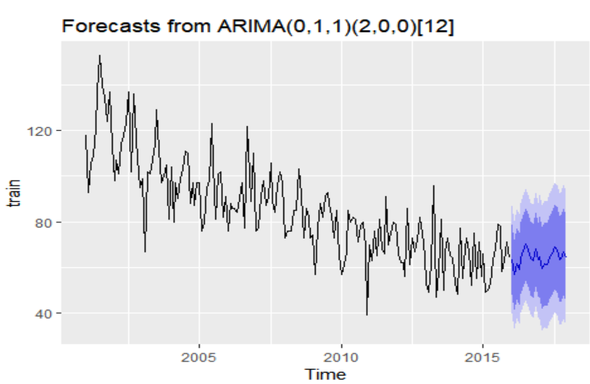
210 112 best MASE and RMSE on the testing set

Although the Arima auto does show an acceptable result of residual diagnostics, but it's accuracy are not better than Arima 411 112 and it also has residual diagnostics results that is acceptable.

```
tsdisplay(train, main="Fatalities", ylab="Fatalities Index", xlab="Year")
```

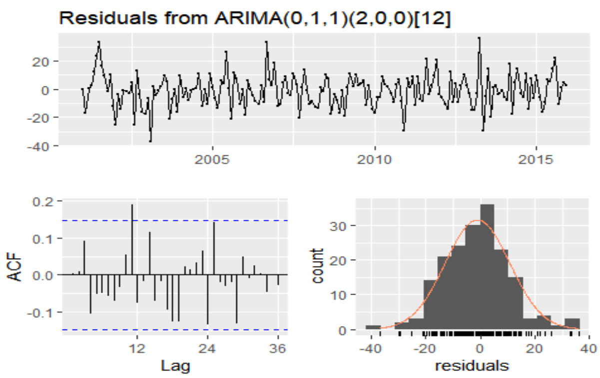



```
ndiffs(train)

## [1] 1

nsdiffs(diff(train))

## [1] 0

f13 <- forecast(auto.arima(train), h=h)
autoplot(f13)
```




```
accuracy(f13)[,c(2,3,5,6)]
```

```
##       RMSE        MAE       MAPE       MASE
## 11.8725854  9.1582208 11.7404996  0.8236516

res <- residuals(f13)
checkresiduals(f13)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,1)(2,0,0)[12]
## Q* = 31.666, df = 21, p-value = 0.06327
##
## Model df: 3.   Total lags used: 24

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags    statistic df   p-value
##     1  0.002435481  1 0.9606399
##     5  4.089328966  5 0.5366278
##     9  6.282254954  9 0.7113703
##    13 14.888676153 13 0.3143540
##    17 20.348229493 17 0.2568065
##    21 26.868978195 21 0.1752322

getinfo <- function(x,h,...)
{
  train.end <- time(x)[length(x)-h]
  test.start <- time(x)[length(x)-h+1]
  train <- window(x,end=train.end)
  test <- window(x,start=test.start)
  fit <- Arima(train,...)
  fc <- forecast(fit,h=h)
  a <- accuracy(fc,test)
  result <- matrix(NA, nrow=1, ncol=5)
  result[1,1] <- fit$aicc
  result[1,2] <- a[1,6]
  result[1,3] <- a[2,6]
  result[1,4] <- a[1,2]
  result[1,5] <- a[2,2]
  return(result)
}
mat <- matrix(NA,nrow=54, ncol=5)
modelnames <- vector(mode="character", length=54)
line <- 0
for (i in 2:4){
  for (j in 0:2){
```

```
    for (k in 0:1){
      for (l in 0:2){
        line <- line+1
        mat[line,] <- getinfo(train,h=h,order=c(i,1,j),seasonal=c(k,1,l))
        modelnames[line] <- paste0("ARIMA(",i,",1,",j,")(",k,",1,",l,")[12]")
      }
    }
  }
}
colnames(mat) <- c("AICc", "MASE_train", "MASE_test", "RMSE_train", "RMSE_test")
rownames(mat) <- modelnames

print("best AICc")

## [1] "best AICc"

mat[mat[,1]==min(mat[,1])]

## [1] 1124.8727315    0.7328616    0.7639839   10.8124114   10.5503952

which(mat[,1]==min(mat[,1]))

## ARIMA(2,1,1)(0,1,1)[12]
##                       8

print("best MASE_train")

## [1] "best MASE_train"

mat[mat[,2]==min(mat[,2])]

## [1] 1127.6497083    0.6956680    0.8033632   10.4202578   11.0225602

which(mat[,2]==min(mat[,2]))

## ARIMA(4,1,2)(0,1,2)[12]
##                      51

print("best MASE_test")

## [1] "best MASE_test"

mat[mat[,3]==min(mat[,3])]

## [1] 1149.5508524    0.7951187    0.6664983   11.6935501    8.9268098

which(mat[,3]==min(mat[,3]))

## ARIMA(2,1,0)(1,1,2)[12]
##                       6

print("best RMSE_train")

## [1] "best RMSE_train"

mat[mat[,4]==min(mat[,4])]

## [1] 1127.6497083    0.6956680    0.8033632   10.4202578   11.0225602

which(mat[,4]==min(mat[,4]))

## ARIMA(4,1,2)(0,1,2)[12]
##                      51
```

```r
print("best RMSE_test")
```

```
## [1] "best RMSE_test"
```

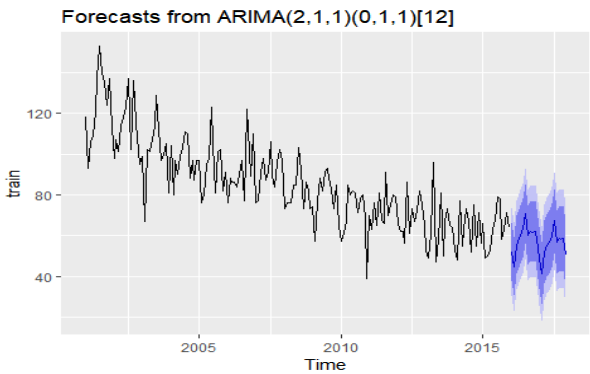
```r
mat[mat[,5]==min(mat[,5])]
```

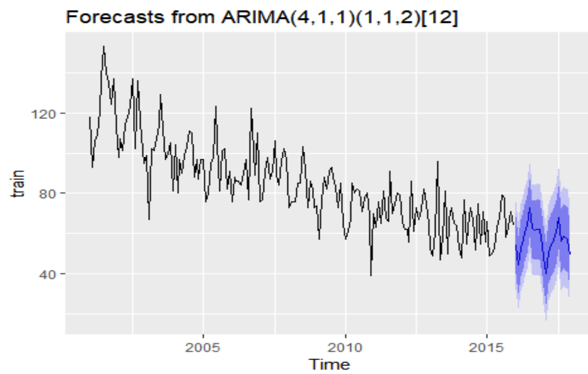
```
## [1] 1149.5508524    0.7951187    0.6664983   11.6935501    8.9268098
```

```r
which(mat[,5]==min(mat[,5]))
```

```
## ARIMA(2,1,0)(1,1,2)[12]
##                       6
```

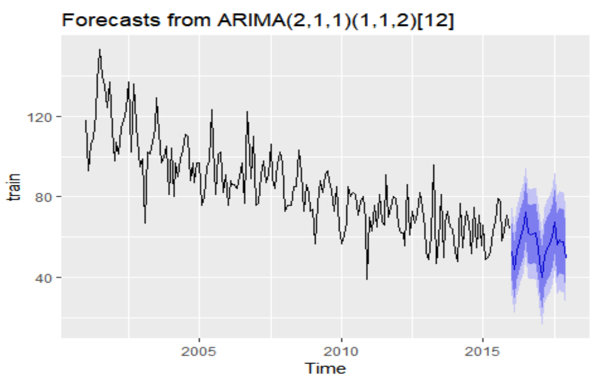
```r
f14 <- forecast(Arima(train, order=c(2,1,1), seasonal=c(0,1,1)), h=h)
autoplot(f14)
```






```r
f15 <- forecast(Arima(train, order=c(4,1,1), seasonal=c(1,1,2)), h=h)
autoplot(f15)
```

```r
f16 <- forecast(Arima(train, order=c(2,1,1), seasonal=c(1,1,2)), h=h)
autoplot(f16)
```




```r
accuracy(f13)[,c(2,3,5,6)]
```

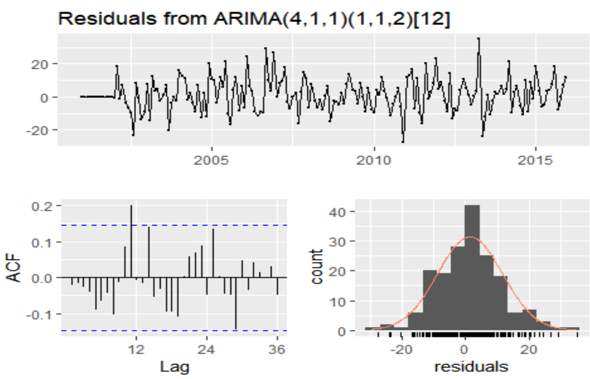
```
##       RMSE        MAE       MAPE       MASE
## 11.8725854  9.1582208 11.7404996  0.8236516
```

```r
accuracy(f14)[,c(2,3,5,6)]
```

```
##       RMSE        MAE       MAPE       MASE
## 10.5274815  8.0149662 10.2172706  0.7208321
```

```r
accuracy(f15)[,c(2,3,5,6)]
```

```
##       RMSE        MAE       MAPE       MASE 
## 10.2643277  7.8310908  9.9533502  0.7042951

accuracy(f16)[,c(2,3,5,6)]

##       RMSE        MAE       MAPE       MASE 
## 10.4057033  7.9037897 10.0840601  0.7108333

res <- residuals(f15)
checkresiduals(f15)
```




```
## 
##  Ljung-Box test
## 
## data:  Residuals from ARIMA(4,1,1)(1,1,2)[12]
## Q* = 28.987, df = 16, p-value = 0.02402
## 
## Model df: 8.   Total lags used: 24

res <- na.omit(res)
LjungBox(res, lags=seq(1,24,4), order=0)

##  lags   statistic df   p-value
##     1  0.08780311  1 0.7669888
##     5  2.05225983  5 0.8418677
##     9  5.32717771  9 0.8049058
##    13 14.48892525 13 0.3403340
##    17 20.93713334 17 0.2291150
##    21 25.89429350 21 0.2105083
```

Conclusion

With the table we can see that ETS MMM test have the best performance of RMSE, MAE, MAPE and MASE. The residual diagnostics results of ETS MMM is also acceptable. Therefore we will use ETS MMM model as final to do forecast to 2020.

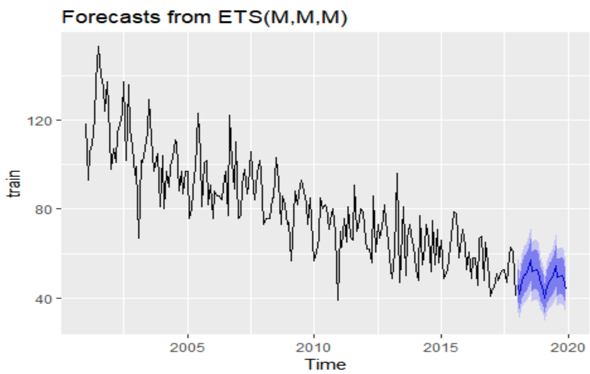
```
af1 = accuracy(f1, test)
af2 = accuracy(f2, test)
af3 = accuracy(f3, test)
af4 = accuracy(f4, test)
af5 = accuracy(f5, test)
af6 = accuracy(f6, test)
af7 = accuracy(f7, test)
af8 = accuracy(f8, test)
af9 = accuracy(f9, test)
af10 = accuracy(f10, test)
af11 = accuracy(f11, test)
af12 = accuracy(f12, test)
af13 = accuracy(f13, test)
```

```
af14 = accuracy(f14, test)
af15 = accuracy(f15, test)
af16 = accuracy(f16, test)

a.table <- rbind(af1, af2, af3, af4, af5, af6, af7, af8, af9, af10, af11, af12, af13, af14, af15, af16)
row.names(a.table)<-c("S. Naive training", 'S. Naive test',
                      'STL training', 'STL test',
                      'HW multi train','HW multi test',
                      'HW multi exponential train','HW multi exponential test',
                      'HW damped exponential train','HW damped exponential test',
                      "HW additive train", "HW additive test",
                      'HW addi damped trend train','HW addi damped trend test',
                      'ETS auto training', 'ETS auto test',
                      'ETS MAM training', 'ETS MAM test',
                      'ETS MMM training', 'ETS MMM test',
                      'ETS MAM d training', 'ETS MAM d test',
                      'ETS MMM d training', 'ETS MMM d test',
                      'ARIMA Auto training', 'ARIMA Auto test',
                      'ARIMA 211 011 training', 'ARIMA 211 011 test',
                      'ARIMA 411 112 training', 'ARIMA 411 112 test',
                      'ARIMA 211 112 training', 'ARIMA 211 112 test')
a.table <- as.data.frame(a.table)
print(kable(a.table, caption="Forecast accuracy",digits = 2 ))

## 
## 
## Table: Forecast accuracy
## 
##                                    ME    RMSE     MAE      MPE    MAPE   MASE    ACF1   Theil's U
## ----------------------------  -------  ------  ------  -------  ------  -----  ------  ----------
## S. Naive training               -4.31   14.51   11.12    -6.56   14.51   1.00   -0.10          NA
## S. Naive test                   -9.96   14.66   11.71   -20.69   23.60   1.05    0.29        1.43
## STL training                     0.00   13.56   10.57    -1.54   13.40   0.95   -0.52          NA
## STL test                       -10.94   12.96   11.50   -22.20   23.12   1.03   -0.06        1.24
## HW multi train                   0.83    9.76    7.48    -0.29    9.37   0.67   -0.08          NA
## HW multi test                   -5.44    9.25    7.61   -11.98   15.41   0.68   -0.03        0.88
## HW multi exponential train       0.56    9.63    7.41    -0.51    9.29   0.67   -0.08          NA
## HW multi exponential test       -3.60    8.22    6.57    -8.46   13.14   0.59   -0.05        0.78
## HW damped exponential train     -0.75    9.71    7.55    -2.41    9.62   0.68   -0.07          NA
## HW damped exponential test      -7.43   10.69    8.89   -15.81   18.11   0.80    0.01        1.03
## HW additive train                0.86   10.13    7.73    -0.22    9.61   0.70   -0.03          NA
## HW additive test                -4.57    9.18    7.42   -10.04   14.73   0.67    0.03        0.87
## HW addi damped trend train      -0.90   10.05    7.78    -2.66    9.86   0.70   -0.01          NA
## HW addi damped trend test       -6.92   10.79    8.80   -14.57   17.60   0.79    0.08        1.03
## ETS auto training               -0.90   10.05    7.78    -2.66    9.86   0.70   -0.01          NA
## ETS auto test                   -6.92   10.79    8.80   -14.57   17.60   0.79    0.08        1.03
## ETS MAM training                 1.16    9.88    7.58    -0.02    9.41   0.68   -0.06          NA
## ETS MAM test                    -5.06    8.89    7.29   -11.29   14.81   0.66   -0.05        0.84
## ETS MMM training                -0.26    9.73    7.54    -1.46    9.47   0.68   -0.06          NA
## ETS MMM test                    -2.87    7.80    6.15    -7.12   12.31   0.55   -0.07        0.73
## ETS MAM d training              -0.95    9.91    7.81    -2.60    9.93   0.70   -0.09          NA
## ETS MAM d test                  -8.25   11.31    9.67   -17.35   19.60   0.87    0.00        1.09
## ETS MMM d training              -0.97    9.75    7.61    -2.69    9.69   0.68   -0.06          NA
## ETS MMM d test                  -7.22   10.54    8.79   -15.43   17.91   0.79    0.00        1.01
## ARIMA Auto training             -1.15   11.87    9.16    -3.27   11.74   0.82    0.00          NA
## ARIMA Auto test                -10.59   13.31   11.18   -22.31   23.25   1.01    0.04        1.31
## ARIMA 211 011 training           1.32   10.53    8.01     0.62   10.22   0.72   -0.01          NA
## ARIMA 211 011 test              -3.48    8.39    6.77    -8.00   13.44   0.61   -0.02        0.79
## ARIMA 411 112 training           1.33   10.26    7.83     0.71    9.95   0.70   -0.02          NA
```

```
## ARIMA 411 112 test           -3.42    8.39    6.70   -7.78   13.27   0.60    0.05        0.80
## ARIMA 211 112 training        1.31   10.41    7.90    0.65   10.08   0.71   -0.01          NA
## ARIMA 211 112 test           -3.27    8.33    6.59   -7.51   13.04   0.59    0.05        0.79
```

Final model

```
train <- window(to, start=c(2001,1),end=c(2017,12))

f <- forecast(ets(train, model = "MMM"), method="rwdrift", h=24)
autoplot(f)
```




```
f$mean

##          Jan      Feb      Mar      Apr      May      Jun      Jul
## 2018 45.90558 42.00744 47.79097 50.51923 50.83330 53.16820 57.36857
## 2019 43.59661 39.89454 45.38716 47.97820 48.27647 50.49393 54.48302
##          Aug      Sep      Oct      Nov      Dec
## 2018 51.95187 52.56606 53.04963 51.95001 46.55150
## 2019 49.33878 49.92208 50.38132 49.33701 44.21003
```