Time Series Analysis Final Project

Annie Phan, Andrew Luong, Venku Buragadda

March 14, 2019

```
library(fpp)
require(rio)
library(TSA)
```

Importing Data

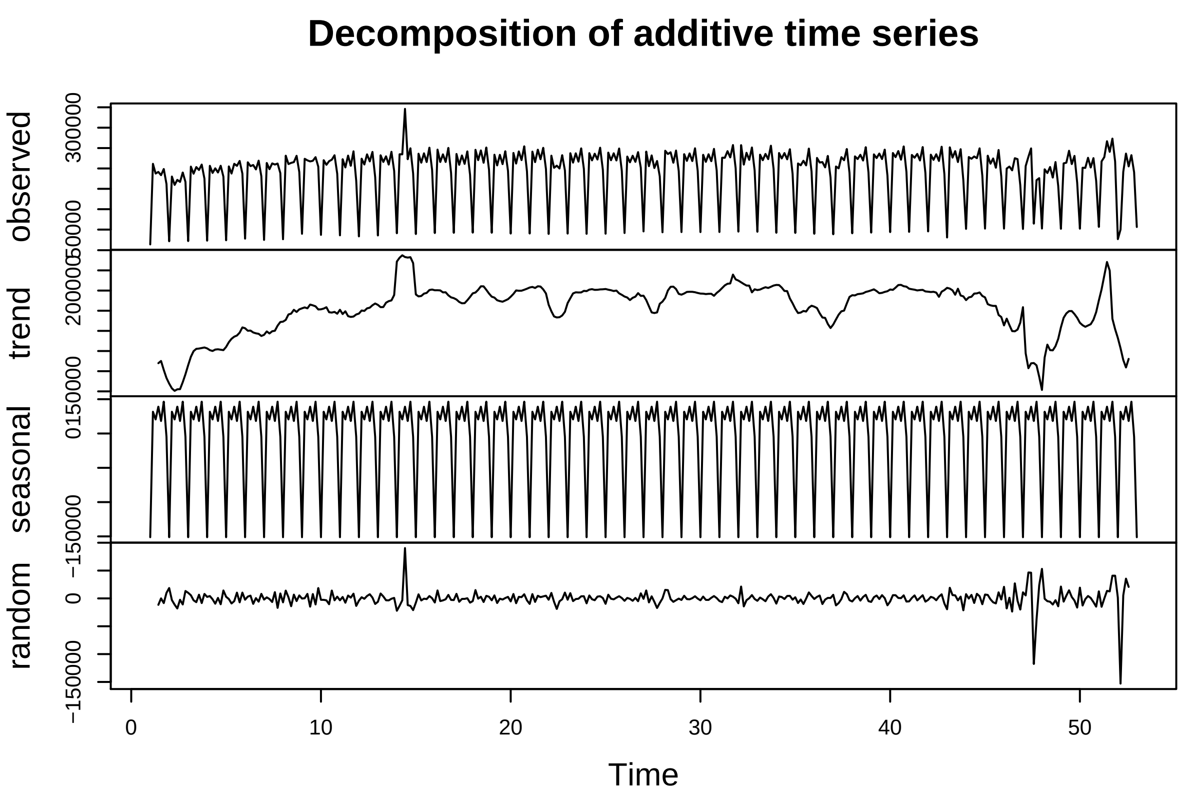
```
mas.data <- import("C:/Users/aluong/Desktop/MScA Repository/MScA Time Series/MSA TSA Assignment/daily_sl

data.case <- mas.data$`Sum of Case Shipment`
data.cost <- mas.data$`Sum of Shipment Cost`
data.dist <- mas.data$`Sum of Distance`
data.temp <- mas.data$`Avg. Temp`
```

Convering to Time Series

```
data.case.ts <- ts(data=data.case, start = 1,frequency = 7)
data.dist.ts <- ts(data=data.dist, start = 1,frequency = 7)
```

Displaying Data

```
plot(decompose(data.case.ts))
```



Training and Holdout Spliting

```
endwk <- 46

data.case.ts.train <- window(data.case.ts, end = c(endwk,5))
data.case.ts.holdout <- window(data.case.ts, start = c(endwk,6))

data.dist.ts.train <- window(data.dist.ts, end = c(endwk,5))
data.dist.ts.holdout <- window(data.dist.ts, start = c(endwk,6))

length(data.case.ts.train)
```

```
## [1] 320
```

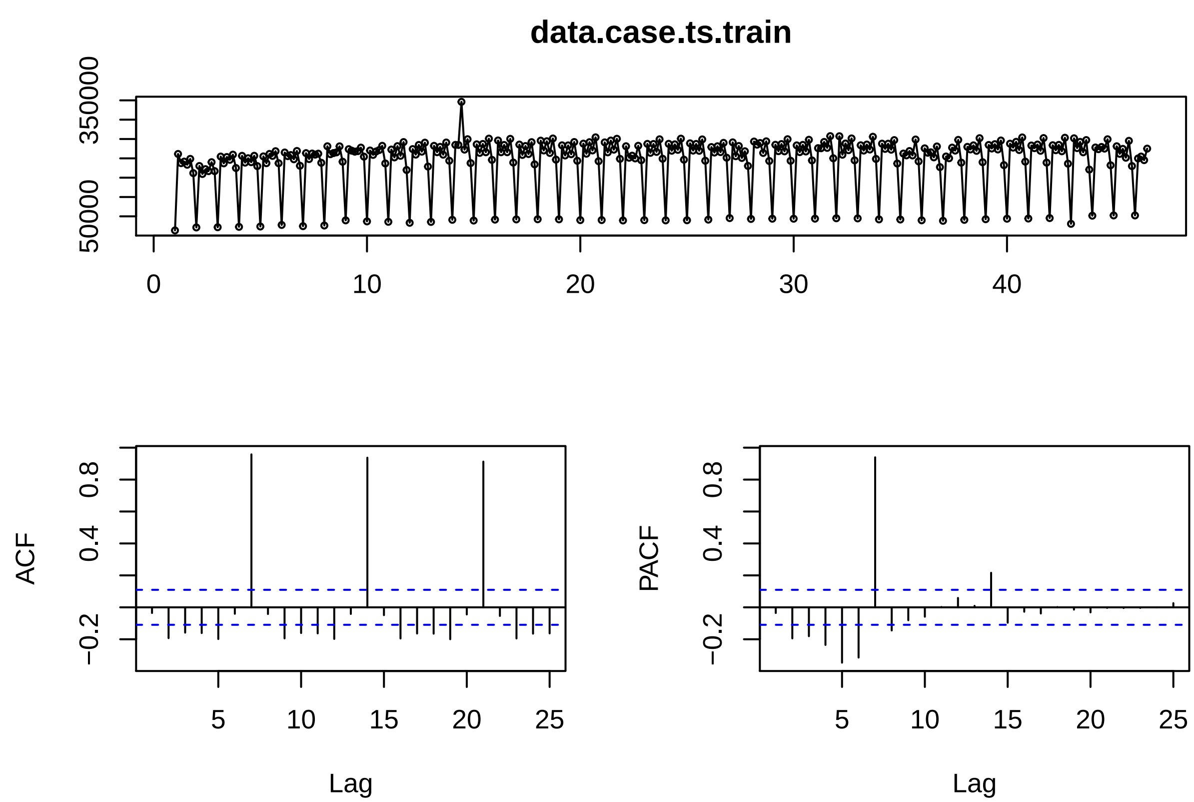
```
length(data.case.ts.holdout)
```

```
## [1] 45
```

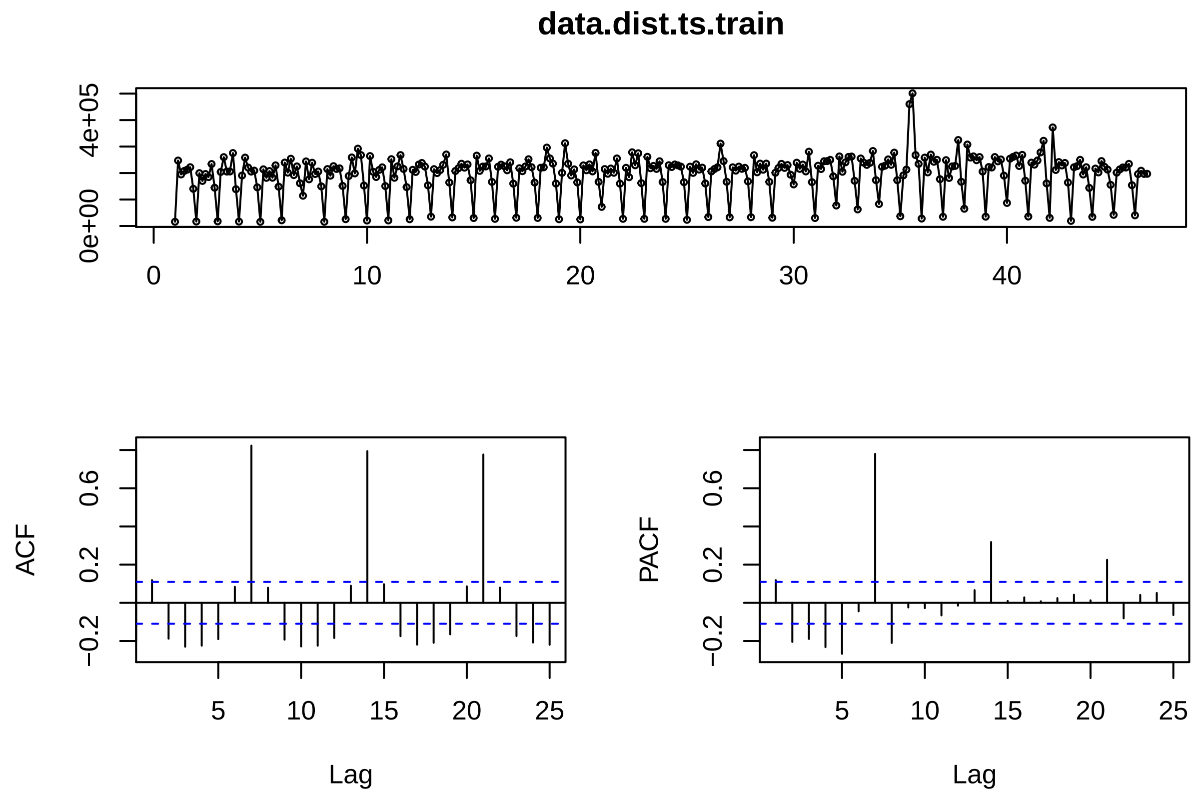
```
hf <- length(data.case.ts.holdout)
```

Displaying Training Data

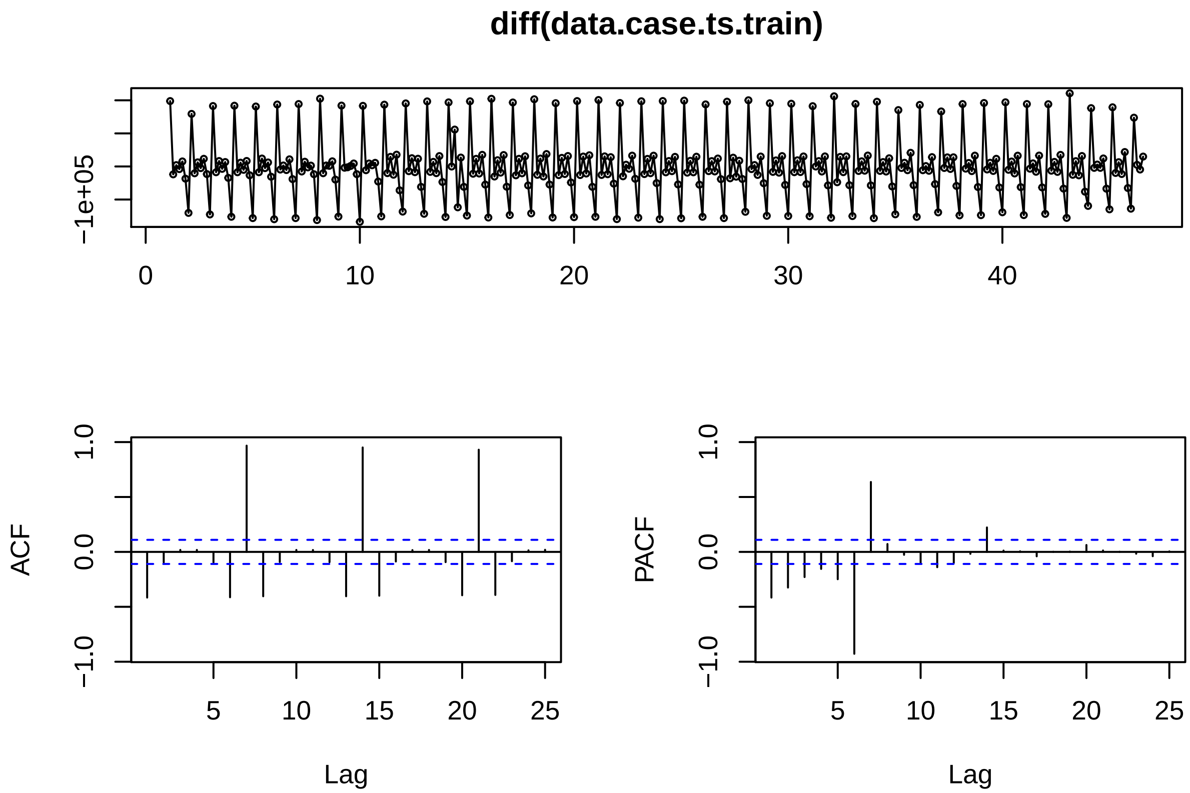
```
tsdisplay(data.case.ts.train)
```



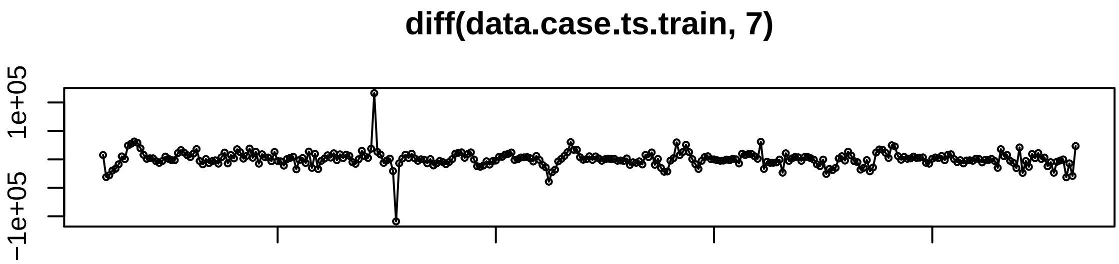

```
tsdisplay(data.dist.ts.train)
```

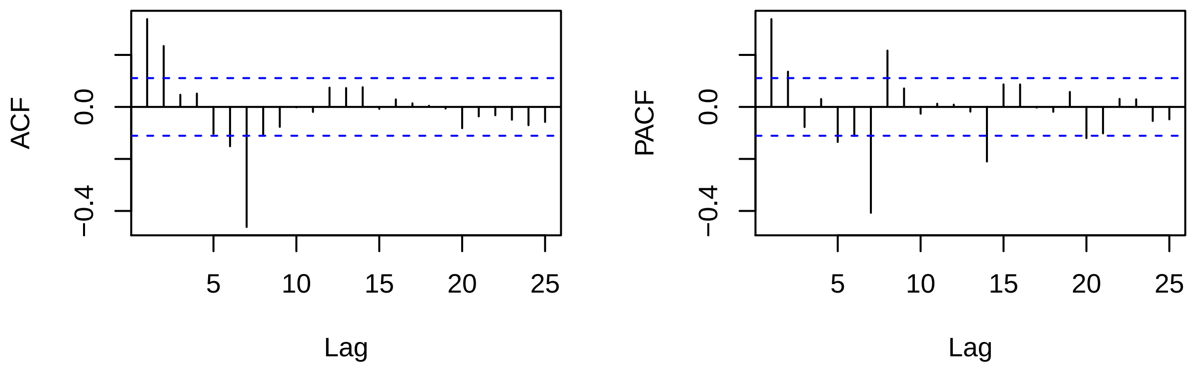



```
tsdisplay(diff(data.case.ts.train))
```




```r
tsdisplay(diff(data.case.ts.train,7))
```






```r
kpss.test(data.case.ts.train)
```

```
## Warning in kpss.test(data.case.ts.train): p-value smaller than printed p-
## value
```

```
##
##  KPSS Test for Level Stationarity
##
## data:  data.case.ts.train
## KPSS Level = 1.1234, Truncation lag parameter = 5, p-value = 0.01
```

```r
adf.test(data.case.ts.train)
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  data.case.ts.train
## Dickey-Fuller = -1.9206, Lag order = 6, p-value = 0.61
## alternative hypothesis: stationary
```

ETS Modeling

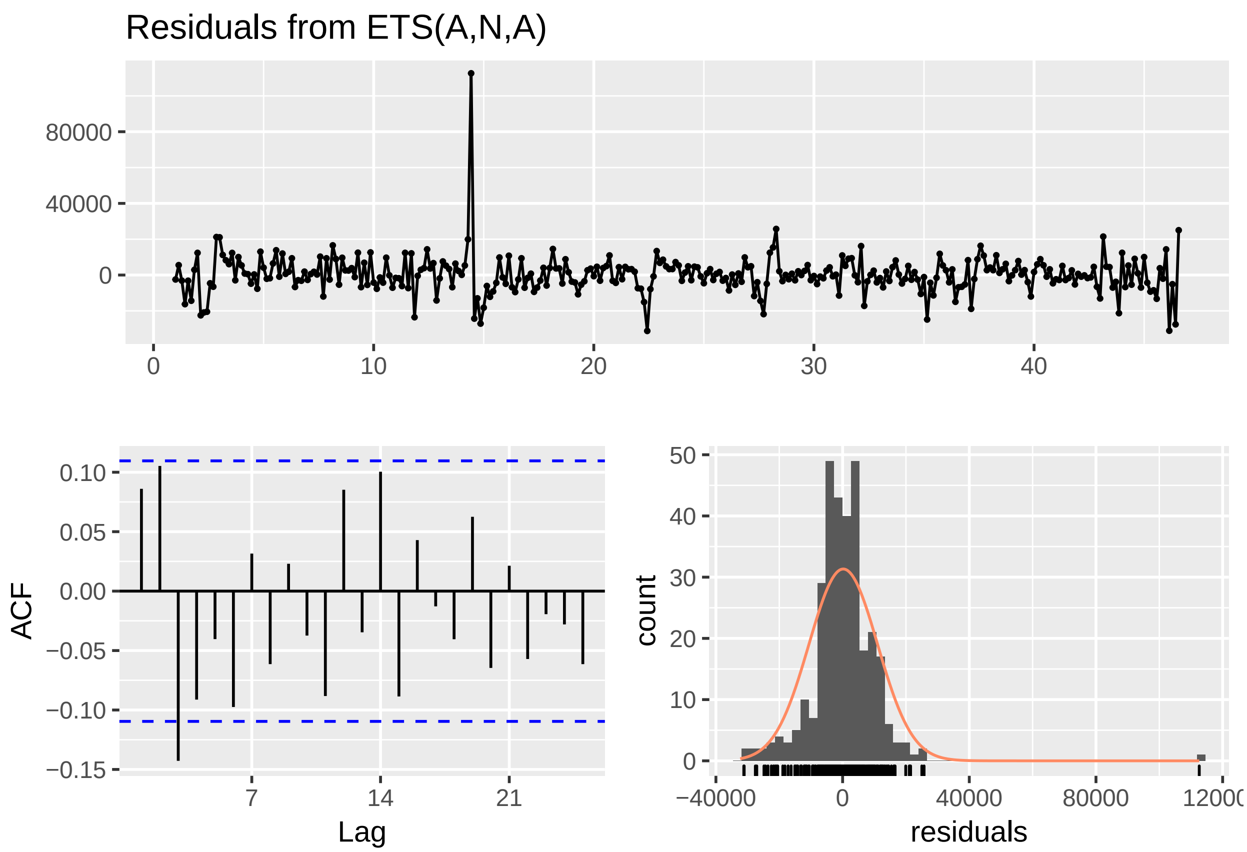
```r
case.train.ets <- ets(data.case.ts.train)
summary(case.train.ets)
```

```
## ETS(A,N,A)
##
## Call:
##  ets(y = data.case.ts.train)
```

```
##
##   Smoothing parameters:
##     alpha = 0.2236
##     gamma = 0.0903
##
##   Initial states:
##     l = 170609.3091
##     s = -4159.668 46727.48 19674.55 37195.18 19321.13 35759.82
##            -154518.5
##
##   sigma:  10991.57
##
##      AIC     AICc      BIC
## 7811.860 7812.572 7849.543
##
## Training set error measures:
##                    ME    RMSE      MAE       MPE     MAPE      MASE
## Training set 201.9829 10835.9 6874.972 0.2455703 4.882706 0.8069258
##                    ACF1
## Training set 0.08603299
```

ETS Checking Residuals

```r
checkresiduals(case.train.ets)
```

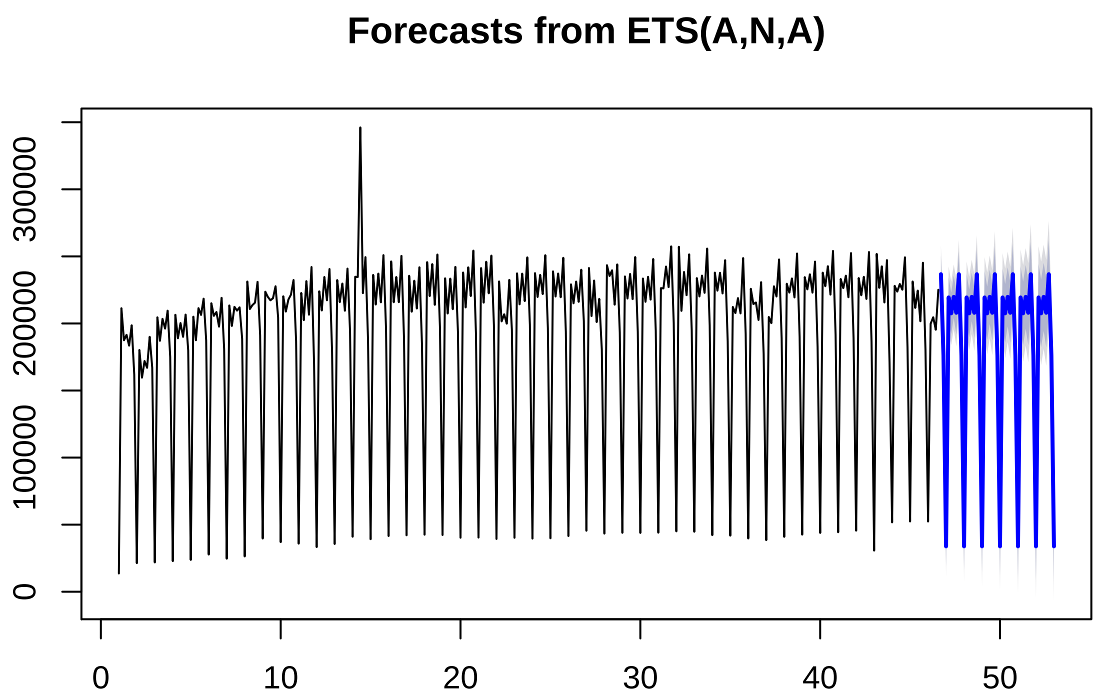



```
##
##  Ljung-Box test
##
```

```
## data:  Residuals from ETS(A,N,A)
## Q* = 30.01, df = 5, p-value = 1.468e-05
##
## Model df: 9.   Total lags used: 14
```

ETS Forecasting

```
case.train.ets.f <- forecast(case.train.ets, h = hf)
plot(case.train.ets.f)
```




```
(ets.acc <- accuracy(case.train.ets.f, data.case.ts.holdout))
```

```
##                      ME     RMSE       MAE        MPE      MAPE      MASE
## Training set   201.9829 10835.90  6874.972  0.2455703  4.882706 0.8069258
## Test set     -5244.1068 39307.98 23761.856 -9.3207187 24.656509 2.7889649
##                    ACF1 Theil's U
## Training set 0.08603299        NA
## Test set     0.20048130 0.8524236
```

SARIMA Modeling

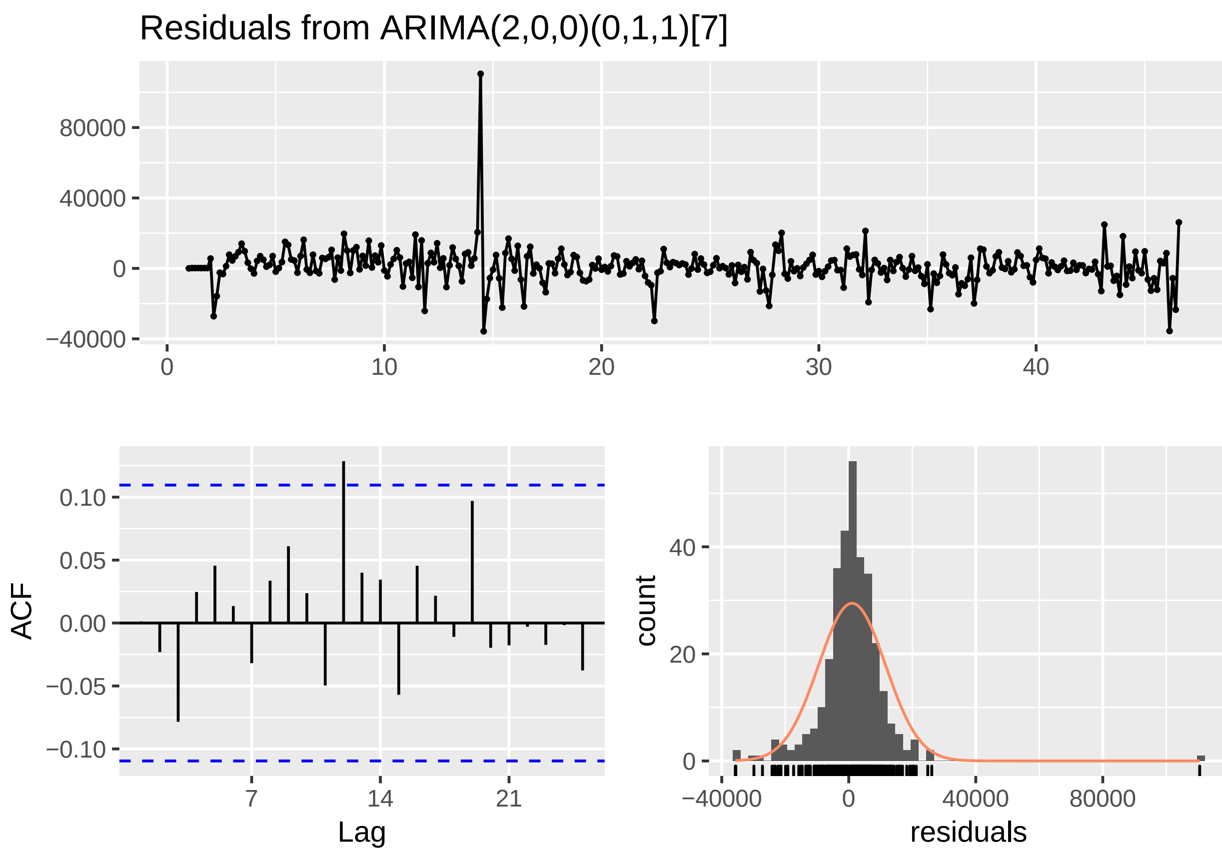
```
case.train.arima <- auto.arima(data.case.ts.train, approximation = FALSE, allowdrift = FALSE)
summary(case.train.arima)
```

```
## Series: data.case.ts.train
## ARIMA(2,0,0)(0,1,1)[7]
##
## Coefficients:
##          ar1     ar2     sma1
```

```
##        0.3360  0.2360  -0.7252
## s.e.   0.0559  0.0599   0.0501
##
## sigma^2 estimated as 1.16e+08:  log likelihood=-3351.43
## AIC=6710.86   AICc=6710.99   BIC=6725.84
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 1044.914 10601.35 6433.757 0.5549662 4.057493 0.7551397
##                     ACF1
## Training set 0.001285711
```

SARIMA Checking Residuals

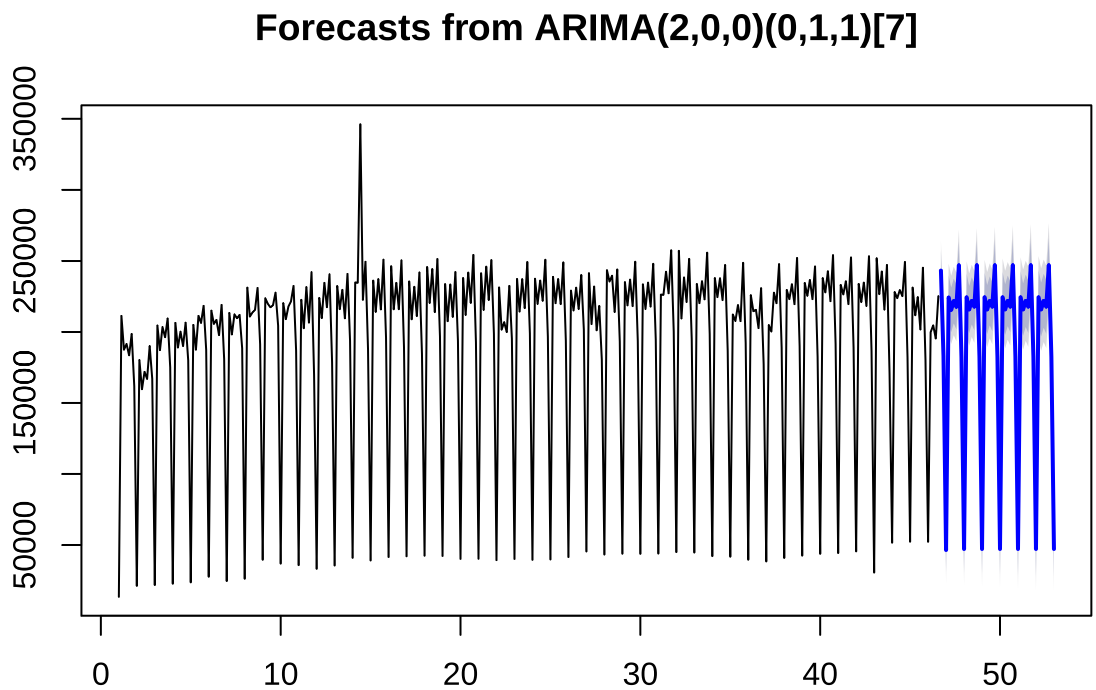
```
checkresiduals(case.train.arima)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,0)(0,1,1)[7]
## Q* = 12.529, df = 11, p-value = 0.3252
##
## Model df: 3.   Total lags used: 14
```

SARIMA Forecasting

```
case.train.arima.f <- forecast(case.train.arima, h = hf)
plot(case.train.arima.f)
```




```
(arima.acc <- accuracy(case.train.arima.f, data.case.ts.holdout))
```

```
##                       ME     RMSE       MAE         MPE      MAPE      MASE
## Training set   1044.914 10601.35  6433.757   0.5549662  4.057493 0.7551397
## Test set     -13136.172 41137.25 25063.956 -16.9960626 24.380949 2.9417943
##                     ACF1 Theil's U
## Training set 0.001285711        NA
## Test set     0.211245544 0.8837904
```

Reg with ARIMA error Modeling

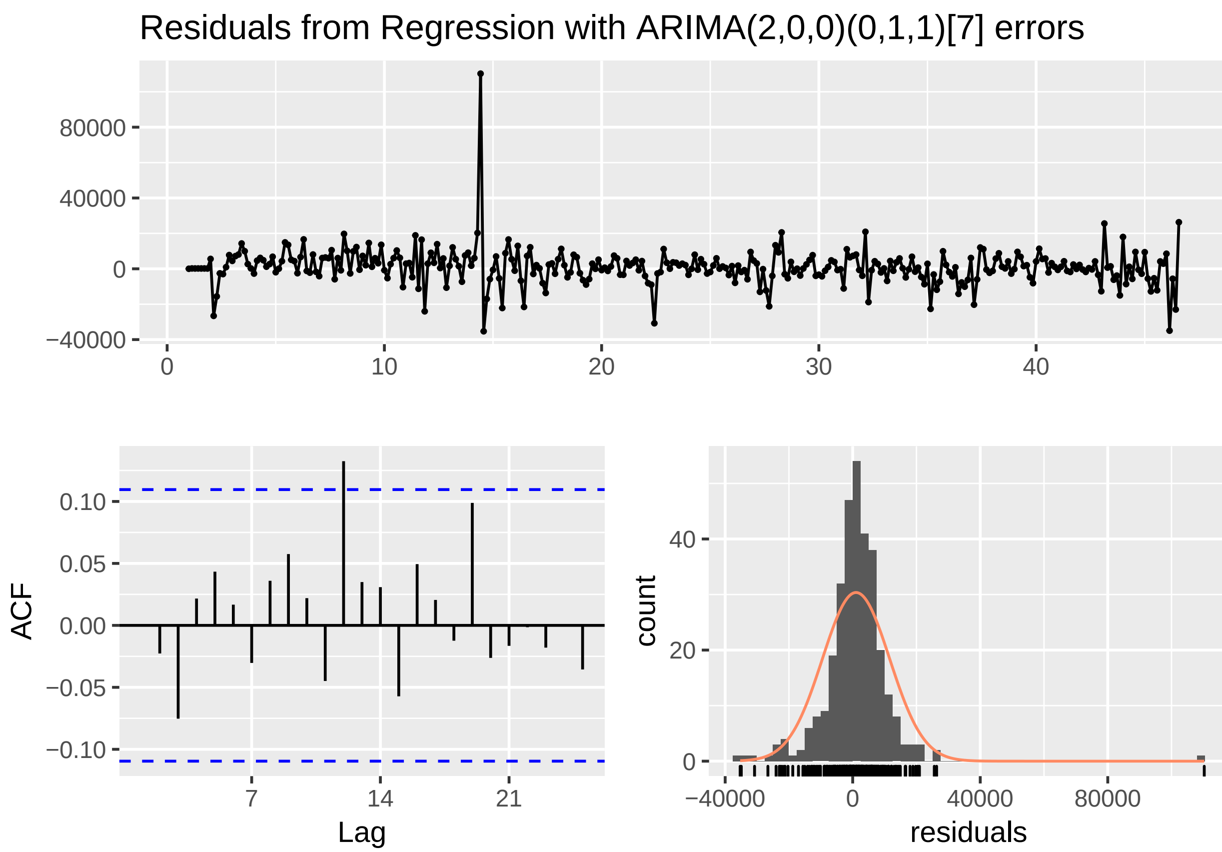
```
case.train.reg.arima <- auto.arima(data.case.ts.train, xreg = cbind(data.dist.ts.train), approximation =
summary(case.train.reg.arima)
```

```
## Series: data.case.ts.train
## Regression with ARIMA(2,0,0)(0,1,1)[7] errors
##
## Coefficients:
##          ar1     ar2     sma1    xreg
##       0.3360  0.2337  -0.7249  0.0154
## s.e.  0.0559  0.0600   0.0500  0.0175
##
## sigma^2 estimated as 116103938:  log likelihood=-3351.04
## AIC=6712.08   AICc=6712.28   BIC=6730.81
##
## Training set error measures:
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set 1044.01 10588.34 6432.697 0.5497382 4.064413 0.7550153
```

```
##                           ACF1
## Training set 0.0007195655
```

Reg with ARIMA error Checking Residuals

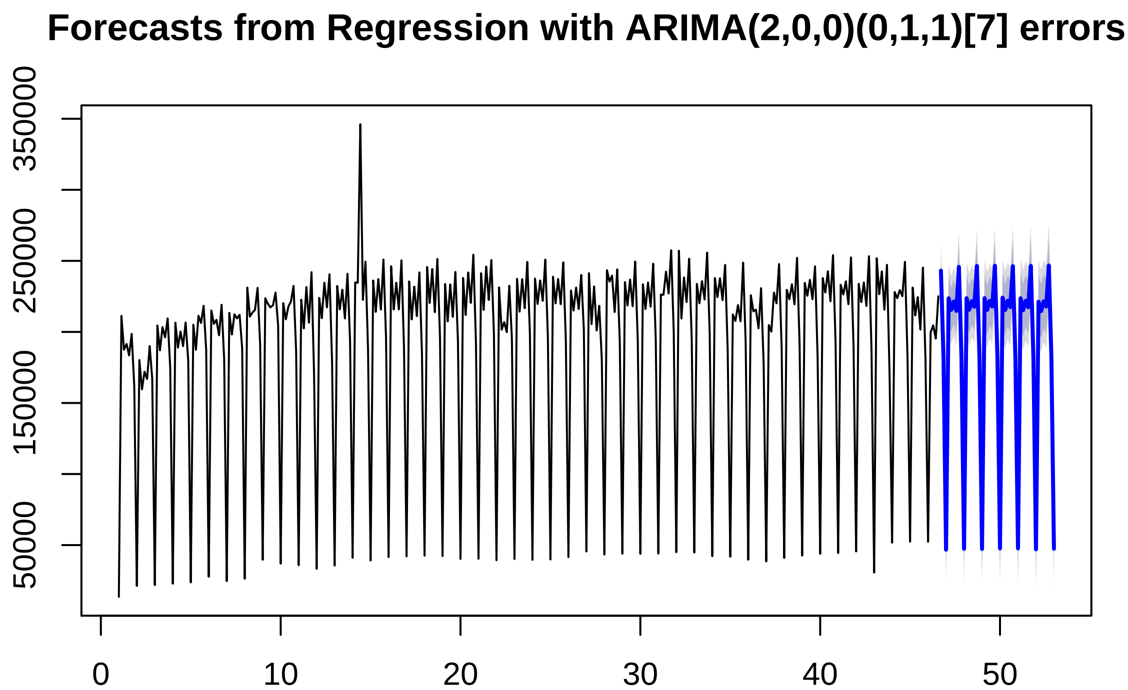
```
checkresiduals(case.train.reg.arima)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from Regression with ARIMA(2,0,0)(0,1,1)[7] errors
## Q* = 12.129, df = 10, p-value = 0.2765
##
## Model df: 4.   Total lags used: 14
```

Reg with ARIMA error Forecasting

```
case.train.reg.arima.f <- forecast(case.train.reg.arima, h =hf, xreg = cbind(data.dist.ts.holdout))
plot(case.train.reg.arima.f)
```




```r
(reg.arima.acc <- accuracy(case.train.reg.arima.f, data.case.ts.holdout))
```

```
##                      ME     RMSE       MAE         MPE      MAPE      MASE
## Training set   1044.01 10588.34  6432.697   0.5497382  4.064413 0.7550153
## Test set     -12781.73 40504.09 24748.732 -16.6725255 23.992083 2.9047960
##                      ACF1 Theil's U
## Training set 0.0007195655        NA
## Test set     0.2125687245 0.8678511
```

MAPE Table

```r
MAPE.m <- rbind(ets.acc[,"MAPE"],
arima.acc[,"MAPE"],
reg.arima.acc[,"MAPE"])
rownames(MAPE.m) <- c("ETS", "sARIMA", "Reg with ARIMA")
MAPE.m
```

```
##                Training set Test set
## ETS                4.882706 24.65651
## sARIMA             4.057493 24.38095
## Reg with ARIMA     4.064413 23.99208
```

MAE Table

```r
MAE.m <- rbind(ets.acc[,"MAE"],
arima.acc[,"MAE"],
reg.arima.acc[,"MAE"])
rownames(MAE.m) <- c("ETS", "sARIMA", "Reg with ARIMA")
MAE.m
```

```
##                 Training set Test set
## ETS                 6874.972 23761.86
## sARIMA              6433.757 25063.96
## Reg with ARIMA      6432.697 24748.73
```

RMSE Table

```
RMSE.m <- rbind(ets.acc[,"RMSE"],
arima.acc[,"RMSE"],
reg.arima.acc[,"RMSE"])
rownames(RMSE.m) <- c("ETS", "sARIMA", "Reg with ARIMA")
RMSE.m
```

```
##                 Training set Test set
## ETS                 10835.90 39307.98
## sARIMA              10601.35 41137.25
## Reg with ARIMA      10588.34 40504.09
```

Cross Validation for all three methods Data Prep

```
k <- 200 # minimum data length for fitting a model
n <- length(data.case.ts) # Number of data points

p <- 7 ### Period
H <- 7 # Forecast horizon

st <- tsp(data.case.ts)[1]+(k-2)/p #  gives the start time in time units,


mae_1 <- matrix(NA,n-k,H)
mae_2 <- matrix(NA,n-k,H)
rmse_1 <- matrix(NA,n-k,H)
rmse_2 <- matrix(NA,n-k,H)
aicc_1 <- matrix(NA,n-k,1)
aicc_2 <- matrix(NA,n-k,1)
mae_3 <- matrix(NA,n-k,H)
mae_4 <- matrix(NA,n-k,H)
rmse_3 <- matrix(NA,n-k,H)
rmse_4 <- matrix(NA,n-k,H)
aicc_3 <- matrix(NA,n-k,1)
aicc_4 <- matrix(NA,n-k,1)
mae_5 <- matrix(NA,n-k,H)
mae_6 <- matrix(NA,n-k,H)
rmse_5 <- matrix(NA,n-k,H)
rmse_6 <- matrix(NA,n-k,H)
aicc_5 <- matrix(NA,n-k,1)
aicc_6 <- matrix(NA,n-k,1)
```

Cross Validation for all three methods

```
for(i in 1:(n-k))
{


  ### One Month rolling forecasting
  # Expanding Window
  train_1 <- window(data.case.ts, end=st + i/p)  ## Window Length: k+i
```

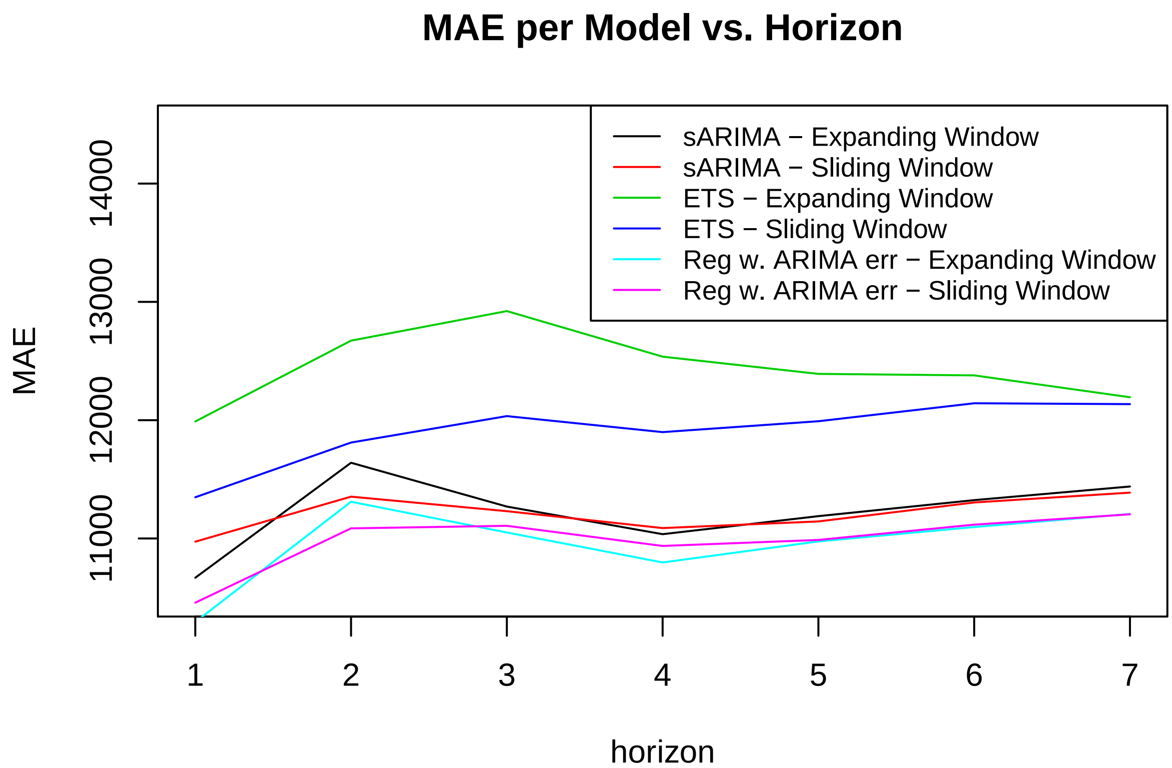
```
reg_train_1 <- window(data.dist.ts, end=st + i/p)  ## Window Length: k+i

# Sliding Window - keep the training window of fixed length.
# The training set always consists of k observations.
train_2 <- window(data.case.ts, start=st+(i-k+1)/p, end=st+i/p) ## Window Length: k
reg_train_2 <- window(data.dist.ts, start=st+(i-k+1)/p, end=st+i/p) ## Window Length: k


test <- window(data.case.ts, start=st + (i+1)/p, end=st + (i+H)/p) ## Window Length: H
reg_test <- window(data.dist.ts, start=st + (i+1)/p, end=st + (i+H)/p) ## Window Length: H

if (i<4) {
  cat(c("*** CV", i,":","len(Expanding Window):",length(train_1), "len(Sliding Window):",length(train_
  cat(c("*** TRAIN -  Expanding WIndow:",tsp(train_1)[1],'-',tsp(train_1)[2],'\n'))
  cat(c("*** TRAIN - Sliding WIndow:",tsp(train_2)[1],'-',tsp(train_2)[2],'\n'))
  cat(c("*** TEST:",tsp(test)[1],'-',tsp(test)[2],'\n'))
  cat("*************************** \n \n")
}


fit_1 <- Arima(train_1, order=c(2,0,0), seasonal=list(order=c(0,1,1), period=p))
fcast_1 <- forecast(fit_1, h=H)


fit_2 <- Arima(train_2, order=c(2,0,0), seasonal=list(order=c(0,1,1), period=p))
fcast_2 <- forecast(fit_2, h=H)

fit_3 <- ets(train_1)
fcast_3 <- forecast(fit_3, h=H)

fit_4 <- ets(train_2)
fcast_4 <- forecast(fit_4, h=H)

fit_5 <- Arima(train_1, order=c(2,0,0), seasonal=list(order=c(0,1,1), period=p), xreg = cbind(reg_trai
fcast_5 <- forecast(fit_5, xreg= cbind(reg_test), h=H)

fit_6 <- Arima(train_2, order=c(2,0,0), seasonal=list(order=c(0,1,1), period=p),  xreg = cbind(reg_tra
fcast_6 <- forecast(fit_6, xreg= cbind(reg_test), h=H)

mae_1[i,1:length(test)] <- abs(fcast_1[['mean']]-test)
mae_2[i,1:length(test)] <- abs(fcast_2[['mean']]-test)

rmse_1[i,1:length(test)] <- (fcast_1[['mean']]-test)^2
rmse_2[i,1:length(test)] <- (fcast_2[['mean']]-test)^2

aicc_1[i,1] <- fit_1$aicc
aicc_2[i,1] <- fit_2$aicc

mae_3[i,1:length(test)] <- abs(fcast_3[['mean']]-test)
mae_4[i,1:length(test)] <- abs(fcast_4[['mean']]-test)

rmse_3[i,1:length(test)] <- (fcast_3[['mean']]-test)^2
rmse_4[i,1:length(test)] <- (fcast_4[['mean']]-test)^2
```

```r
  aicc_3[i,1] <- fit_3$aicc
  aicc_4[i,1] <- fit_4$aicc

  mae_5[i,1:length(test)] <- abs(fcast_5[['mean']]-test)
  mae_6[i,1:length(test)] <- abs(fcast_6[['mean']]-test)

  rmse_5[i,1:length(test)] <- (fcast_5[['mean']]-test)^2
  rmse_6[i,1:length(test)] <- (fcast_6[['mean']]-test)^2

  aicc_5[i,1] <- fit_5$aicc
  aicc_6[i,1] <- fit_6$aicc
}
```

```
## *** CV 1 : len(Expanding Window): 200 len(Sliding Window): 200 len(Test): 7
## *** TRAIN -  Expanding WIndow: 1 - 29.4285714285714
## *** TRAIN - Sliding WIndow: 1 - 29.4285714285714
## *** TEST: 29.5714285714286 - 30.4285714285714
## ***************************
##
## *** CV 2 : len(Expanding Window): 201 len(Sliding Window): 200 len(Test): 7
## *** TRAIN -  Expanding WIndow: 1 - 29.5714285714286
## *** TRAIN - Sliding WIndow: 1.14285714285714 - 29.5714285714286
## *** TEST: 29.7142857142857 - 30.5714285714286
## ***************************
##
## *** CV 3 : len(Expanding Window): 202 len(Sliding Window): 200 len(Test): 7
## *** TRAIN -  Expanding WIndow: 1 - 29.7142857142857
## *** TRAIN - Sliding WIndow: 1.28571428571429 - 29.7142857142857
## *** TEST: 29.8571428571429 - 30.7142857142857
## ***************************
##
```

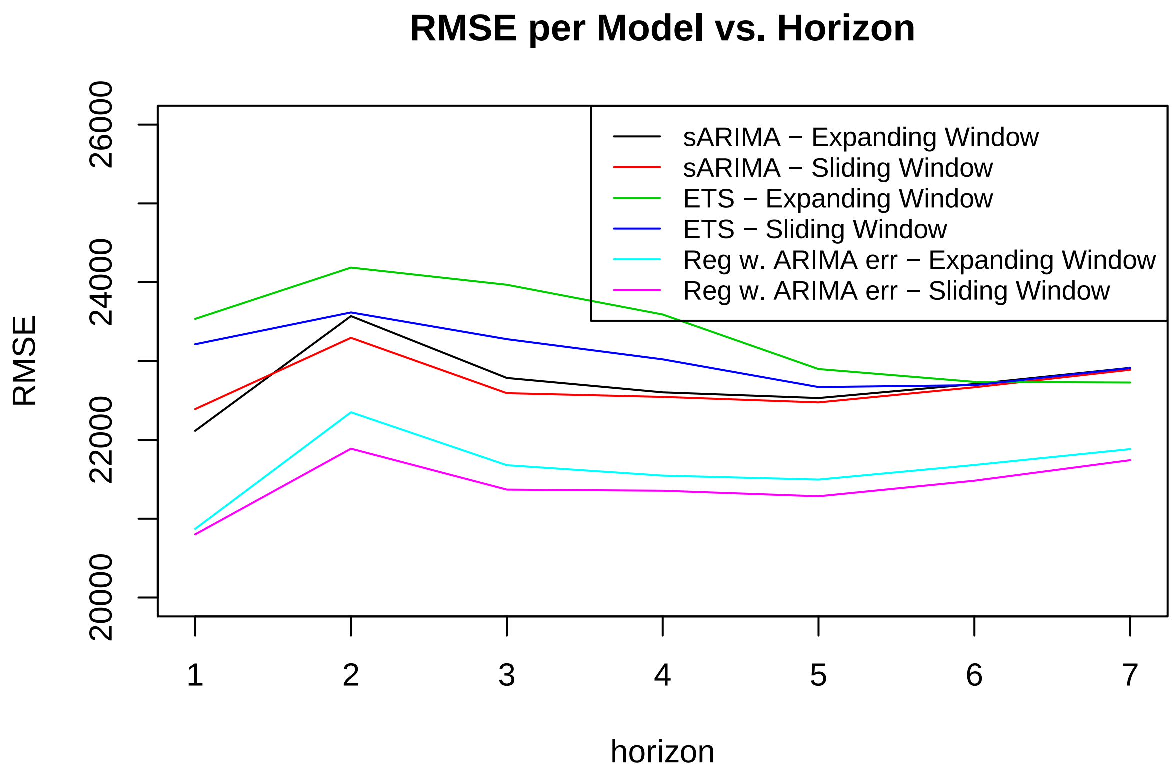
MAE Plotting

```r
#MAE PLOTTING
plot(1:H, colMeans(mae_1,na.rm=TRUE), type="l",col=1,xlab="horizon", ylab="MAE", ylim = c(10500,14500),
lines(1:H, colMeans(mae_2,na.rm=TRUE), type="l",col=2)
lines(1:H, colMeans(mae_3,na.rm=TRUE), type="l",col=3)
lines(1:H, colMeans(mae_4,na.rm=TRUE), type="l",col=4)
lines(1:H, colMeans(mae_5,na.rm=TRUE), type="l",col=5)
lines(1:H, colMeans(mae_6,na.rm=TRUE), type="l",col=6)
legend("topright",legend=c(
    "sARIMA - Expanding Window","sARIMA - Sliding Window",
    'ETS - Expanding Window ', 'ETS - Sliding Window',
    "Reg w. ARIMA err - Expanding Window","Reg w. ARIMA err - Sliding Window")
    ,col=1:6,lty=1, cex = 0.8)
```



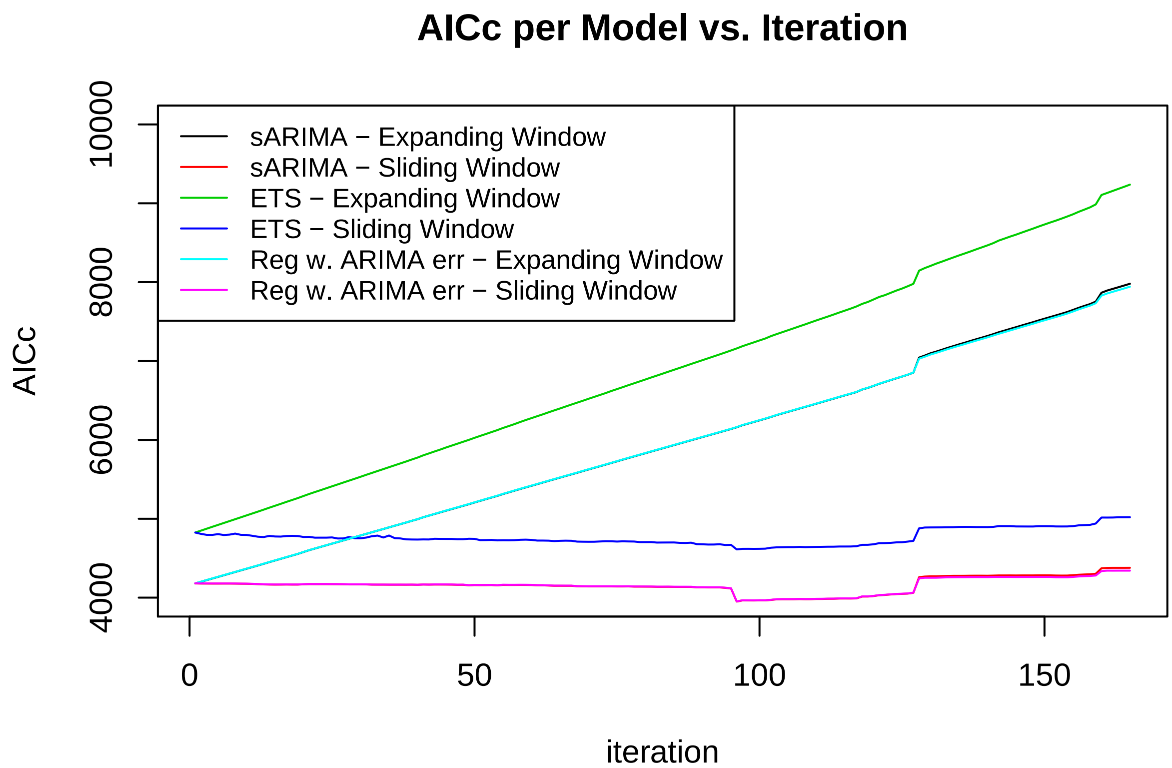
RMSE Plotting

```
#RMSE PLOTTING
plot(1:H, sqrt(colMeans(rmse_1,na.rm=TRUE)), type="l",col=1,xlab="horizon", ylab="RMSE",
     ylim = c(20000, 26000), main = "RMSE per Model vs. Horizon")
lines(1:H, sqrt(colMeans(rmse_2,na.rm=TRUE)), type="l",col=2)
lines(1:H, sqrt(colMeans(rmse_3,na.rm=TRUE)), type="l",col=3)
lines(1:H, sqrt(colMeans(rmse_4,na.rm=TRUE)), type="l",col=4)
lines(1:H, sqrt(colMeans(rmse_5,na.rm=TRUE)), type="l",col=5)
lines(1:H, sqrt(colMeans(rmse_6,na.rm=TRUE)), type="l",col=6)

legend("topright",legend=c(
    "sARIMA - Expanding Window","sARIMA - Sliding Window",
    'ETS - Expanding Window ', 'ETS - Sliding Window',
    "Reg w. ARIMA err - Expanding Window","Reg w. ARIMA err - Sliding Window")
    ,col=1:6,lty=1, cex = 0.8)
```



AICc Plotting

```
l = n-k
#AICc Graph for Both Models
plot(1:l, aicc_1[,1], type="l",col=1,xlab="iteration", ylab="AICc", ylim = c(4000,10000)
     , main = "AICc per Model vs. Iteration")
  lines(1:l, aicc_2[,1], type="l",col=2)
  lines(1:l, aicc_3[,1], type="l",col=3)
  lines(1:l, aicc_4[,1], type="l",col=4)
  lines(1:l, aicc_5[,1], type="l",col=5)
  lines(1:l, aicc_6[,1], type="l",col=6)


  legend("topleft",legend=c(
    "sARIMA - Expanding Window","sARIMA - Sliding Window",
    'ETS - Expanding Window ', 'ETS - Sliding Window',
    "Reg w. ARIMA err - Expanding Window","Reg w. ARIMA err - Sliding Window")
    ,col=1:6,lty=1, cex = 0.8)
```


AICc per Model vs. Iteration
sARIMA − Expanding Window
sARIMA − Sliding Window
ETS − Expanding Window
ETS − Sliding Window
Reg w. ARIMA err − Expanding Window
Reg w. ARIMA err − Sliding Window
AICc
4000
6000
8000
10000
0
50
100
150
iteration