Untitled

Ying Huang

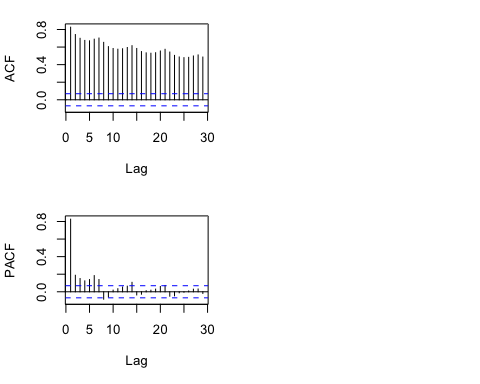
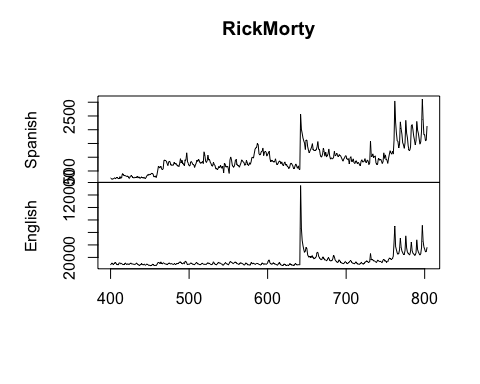
8/20/2019

## Read Rick and Morty Data

project.ts <- readRDS("~/Documents/GitHub/TimeSeries-Project/TSProject/new\_train.rds")  
project.ts <- ts(project.ts)  
rick\_morty\_es <- ts(as.numeric(project.ts[,'rick\_y\_morty']))  
rick\_morty\_es <- window(rick\_morty\_es,start=400)  
rick\_morty\_en <- ts(as.numeric(project.ts[,'rick\_and\_morty']))  
rick\_morty\_en <- window(rick\_morty\_en,start=400)

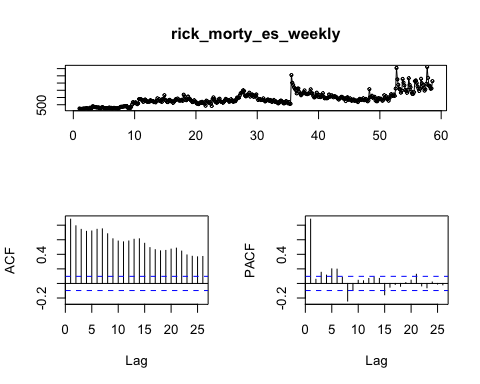
### Plot the data

RickMorty <- cbind(Spanish = rick\_morty\_es, English = rick\_morty\_en)  
tsdisplay(RickMorty)

 Since we analyse the trend of this data, we truncate the data from time 400 because it is flat before 400. We can see from the plot that Spanish page and English page almost have the same pattern, though they are on the different magnitude level. And we can see the increasement of the variance and slightly upward trending. So we can expect the future traffic of Spanish will also influnced by English page with upwarding trend and higher variance. From we ACP and PACF plot we expect frequency to be 7, and MA to be 1.

### Prepare the dataset, transform it to weekly dataset.

rick\_morty\_es\_weekly <- ts(rick\_morty\_es, frequency = 7)  
tsdisplay(rick\_morty\_es\_weekly)



### Split train and test data

es\_weekly\_train <- window(rick\_morty\_es\_weekly,end=c(55,7))  
es\_weekly\_test <- window(rick\_morty\_es\_weekly,start=c(56,1))  
h <- 19

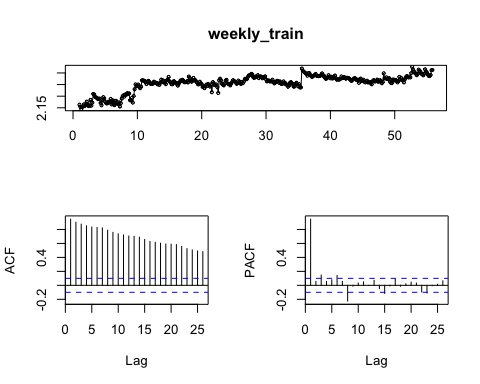
We split the train and test data here, and keep the test set untouched later.

### Data Transforming.

lambda\_weekly <- BoxCox.lambda(es\_weekly\_train)  
lambda\_weekly

## [1] -0.4145433

weekly\_train <- BoxCox(es\_weekly\_train,lambda\_weekly)  
tsdisplay(weekly\_train)



#check stationary  
kpss.test(weekly\_train)

## Warning in kpss.test(weekly\_train): p-value smaller than printed p-value

##   
## KPSS Test for Level Stationarity  
##   
## data: weekly\_train  
## KPSS Level = 3.8372, Truncation lag parameter = 5, p-value = 0.01

kpss.test(diff(weekly\_train)) #pass level

## Warning in kpss.test(diff(weekly\_train)): p-value greater than printed p-  
## value

##   
## KPSS Test for Level Stationarity  
##   
## data: diff(weekly\_train)  
## KPSS Level = 0.061879, Truncation lag parameter = 5, p-value = 0.1

kpss.test(diff(weekly\_train),null = 'Trend') #pass trend

## Warning in kpss.test(diff(weekly\_train), null = "Trend"): p-value greater  
## than printed p-value

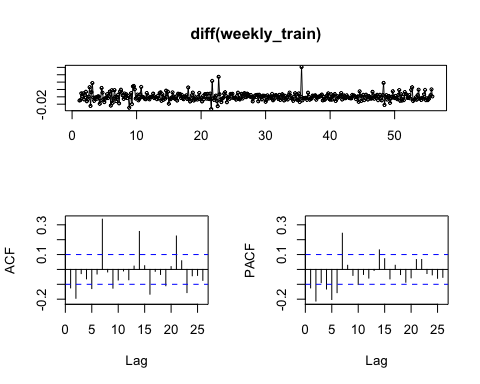
##   
## KPSS Test for Trend Stationarity  
##   
## data: diff(weekly\_train)  
## KPSS Trend = 0.038018, Truncation lag parameter = 5, p-value = 0.1

adf.test(diff(weekly\_train))

## Warning in adf.test(diff(weekly\_train)): p-value smaller than printed p-  
## value

##   
## Augmented Dickey-Fuller Test  
##   
## data: diff(weekly\_train)  
## Dickey-Fuller = -7.3718, Lag order = 7, p-value = 0.01  
## alternative hypothesis: stationary

#we need d=1  
tsdisplay(diff(weekly\_train))

 We preform BoxCox transformation and check the stationality. The data need 1 difference to become stationery.

# Fit the model

## Start with Auto.Arima.

arima.fit <- auto.arima(weekly\_train,d=1,seasonal = TRUE) #211,200  
arima.fit$aicc #-2469.01

## [1] -2469.014

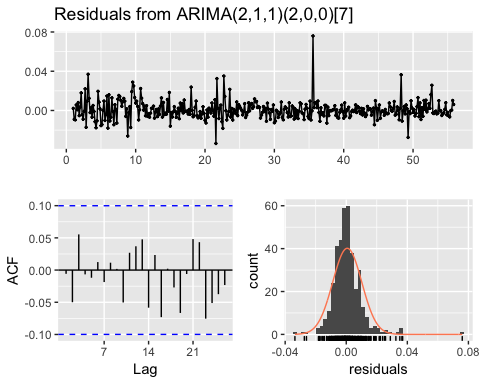
arima.fit$aic #-2469.24

## [1] -2469.237

summary(arima.fit)

## Series: weekly\_train   
## ARIMA(2,1,1)(2,0,0)[7]   
##   
## Coefficients:  
## ar1 ar2 ma1 sar1 sar2  
## 0.6893 0.0310 -0.9154 0.3043 0.1689  
## s.e. 0.0771 0.0635 0.0589 0.0534 0.0533  
##   
## sigma^2 estimated as 9.236e-05: log likelihood=1240.62  
## AIC=-2469.24 AICc=-2469.01 BIC=-2445.53  
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE  
## Training set 0.0006486813 0.009535195 0.006403116 0.02779393 0.2848847  
## MASE ACF1  
## Training set 0.5619666 -0.005803565

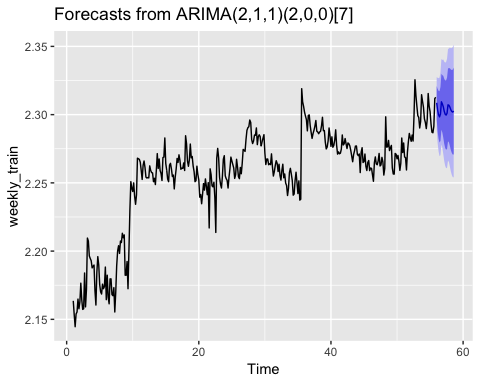
checkresiduals(arima.fit) #white noise



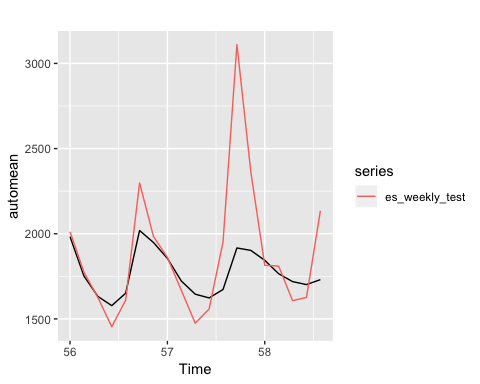
##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(2,1,1)(2,0,0)[7]  
## Q\* = 6.6207, df = 9, p-value = 0.6765  
##   
## Model df: 5. Total lags used: 14

### Prediction from auto arima

autoarima <- forecast(arima.fit,h)  
autoplot(autoarima)



automean <- InvBoxCox(autoarima$mean,lambda\_weekly)  
autoplot(automean)+autolayer(es\_weekly\_test)



sqrt(sum((es\_weekly\_test-automean)^2)/h) #327.0045

## [1] 327.0045

auto <- accuracy(automean,es\_weekly\_test)

Prediction catches some of the seasonality, but not the peak magnitude.

## Arima from eacf

eacf(weekly\_train)

## AR/MA  
## 0 1 2 3 4 5 6 7 8 9 10 11 12 13  
## 0 x x x x x x x x x x x x x x   
## 1 x x o o x o x o x o o o o x   
## 2 x x o o x o x o o o o o o x   
## 3 x x o o o o x x o o o o o x   
## 4 x o x o o o x x o x o o o x   
## 5 x o o x o x x x x o o o o o   
## 6 x x o x x x o x o o o o o o   
## 7 x x x x o x o o o o o o o o

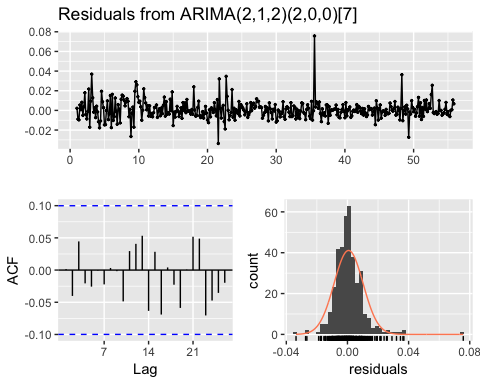
arima212 <- Arima(weekly\_train,order=c(2,1,2),seasonal = list(order=c(2,0,0)))  
arima212 #Aic -2468.07

## Series: weekly\_train   
## ARIMA(2,1,2)(2,0,0)[7]   
##   
## Coefficients:  
## ar1 ar2 ma1 ma2 sar1 sar2  
## 1.3452 -0.4161 -1.5815 0.6001 0.2963 0.1812  
## s.e. 0.4595 0.3514 0.4439 0.4143 0.0537 0.0535  
##   
## sigma^2 estimated as 9.24e-05: log likelihood=1241.04  
## AIC=-2468.07 AICc=-2467.77 BIC=-2440.42

arima312 <- Arima(weekly\_train,order=c(3,1,2),seasonal = list(order=c(2,0,0))) #Aic -2467.41  
arima312

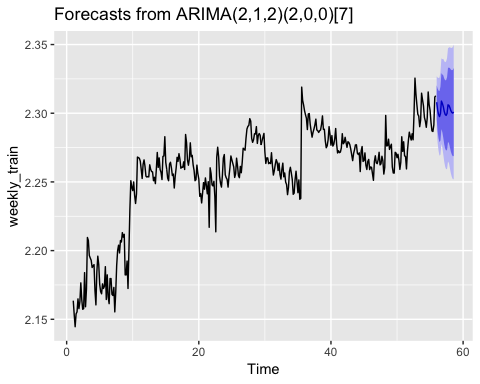
## Series: weekly\_train   
## ARIMA(3,1,2)(2,0,0)[7]   
##   
## Coefficients:  
## ar1 ar2 ar3 ma1 ma2 sar1 sar2  
## 0.3559 0.2445 0.0904 -0.5828 -0.3326 0.3058 0.1716  
## s.e. 0.4793 0.3416 0.0568 0.4803 0.4380 0.0521 0.0536  
##   
## sigma^2 estimated as 9.232e-05: log likelihood=1241.71  
## AIC=-2467.41 AICc=-2467.03 BIC=-2435.81

checkresiduals(arima212) #white noise

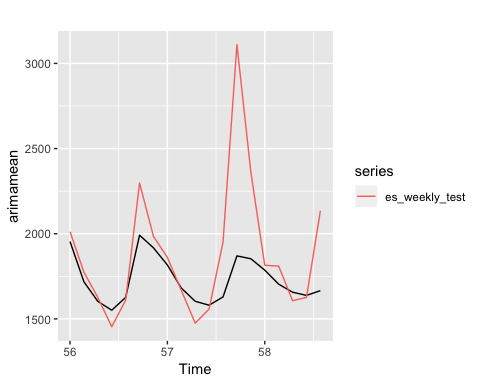


##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(2,1,2)(2,0,0)[7]  
## Q\* = 6.7237, df = 8, p-value = 0.5667  
##   
## Model df: 6. Total lags used: 14

arima212fit <- forecast(arima212,h)  
autoplot(arima212fit)



arimamean <- InvBoxCox(arima212fit$mean,lambda\_weekly)  
autoplot(arimamean)+autolayer(es\_weekly\_test)



sqrt(sum((es\_weekly\_test-arimamean)^2)/h) #345.6329

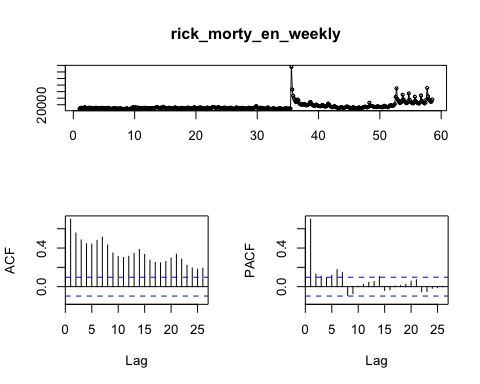
## [1] 345.6329

arima <- accuracy(arimamean,es\_weekly\_test)

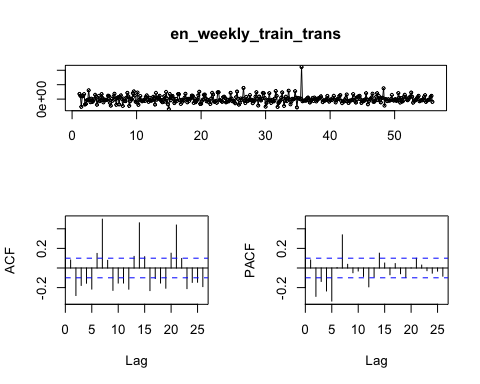
Similar result as SARIMA.

## Xreg

rick\_morty\_en\_weekly <- ts(rick\_morty\_en, frequency = 7)  
tsdisplay(rick\_morty\_en\_weekly)



en\_weekly\_train <- window(rick\_morty\_en\_weekly,end=c(55,7))  
en\_weekly\_test <- window(rick\_morty\_en\_weekly,start=c(56,1))  
  
lambda\_en\_weekly <- BoxCox.lambda(en\_weekly\_train)  
en\_weekly\_trans <- BoxCox(rick\_morty\_en\_weekly,lambda\_en\_weekly)  
en\_weekly\_trans\_diff <- diff(en\_weekly\_trans)  
en\_weekly\_train\_trans <- window(en\_weekly\_trans\_diff, end=c(55,7))  
en\_weekly\_test\_trans <- window(en\_weekly\_trans\_diff, start=c(56,1))  
  
  
tsdisplay(en\_weekly\_train\_trans)



kpss.test(en\_weekly\_train\_trans)

## Warning in kpss.test(en\_weekly\_train\_trans): p-value greater than printed  
## p-value

##   
## KPSS Test for Level Stationarity  
##   
## data: en\_weekly\_train\_trans  
## KPSS Level = 0.029665, Truncation lag parameter = 5, p-value = 0.1

kpss.test(en\_weekly\_train\_trans, null = 'Trend')

## Warning in kpss.test(en\_weekly\_train\_trans, null = "Trend"): p-value  
## greater than printed p-value

##   
## KPSS Test for Trend Stationarity  
##   
## data: en\_weekly\_train\_trans  
## KPSS Trend = 0.020651, Truncation lag parameter = 5, p-value = 0.1

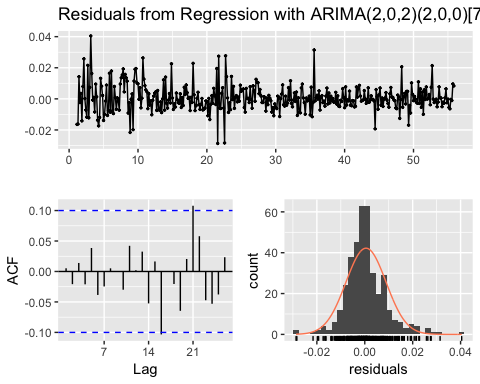
xreg.fit <- auto.arima(diff(weekly\_train),seasonal = TRUE,xreg = en\_weekly\_train\_trans)  
xreg.fit$aic #-2566.89

## [1] -2566.994

summary(xreg.fit)

## Series: diff(weekly\_train)   
## Regression with ARIMA(2,0,2)(2,0,0)[7] errors   
##   
## Coefficients:  
## ar1 ar2 ma1 ma2 sar1 sar2 xreg  
## 1.0995 -0.2919 -1.3693 0.4367 0.2303 0.1065 436.5733  
## s.e. 2.1821 1.2732 2.1867 1.8617 0.0553 0.0558 39.4843  
##   
## sigma^2 estimated as 7.134e-05: log likelihood=1291.5  
## AIC=-2566.99 AICc=-2566.61 BIC=-2535.39  
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set 0.0005194021 0.008368781 0.005887404 NaN Inf 0.7010474  
## ACF1  
## Training set 0.005242521

checkresiduals(xreg.fit) #white noise

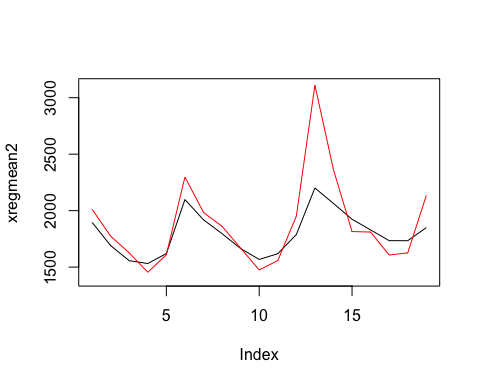


##   
## Ljung-Box test  
##   
## data: Residuals from Regression with ARIMA(2,0,2)(2,0,0)[7] errors  
## Q\* = 4.4259, df = 7, p-value = 0.7296  
##   
## Model df: 7. Total lags used: 14

For the regression with ARIMA, we need to take the time-series data of English page to the stationery as well, and we also need to take difference before split the English page data set. Pretty good result fot the residual.

### Prediction from Xreg

xreg <- forecast(xreg.fit,h,xreg=en\_weekly\_test\_trans)  
xreg2 <- append(weekly\_train[385],xreg$mean)  
xreg.cumsum <- cumsum(xreg2)  
xregmean2 <- InvBoxCox(xreg.cumsum[2:20],lambda\_weekly)  
  
plot(xregmean2,type='l',ylim=c(1400,3100)) + lines(as.numeric(es\_weekly\_test), col='red')



## integer(0)

sqrt(sum((es\_weekly\_test-xregmean2)^2)/h) #246.6643

## [1] 246.6643

xreg <- accuracy(xregmean2,es\_weekly\_test)

Now, we see the prediction is much more similar to the true data.

## VARMA model

data <- cbind(diff(weekly\_train),en\_weekly\_train\_trans)  
VARselect(data,type = 'both')

## $selection  
## AIC(n) HQ(n) SC(n) FPE(n)   
## 7 7 7 7   
##   
## $criteria  
## 1 2 3 4  
## AIC(n) -3.211489e+01 -3.219351e+01 -3.221988e+01 -3.228505e+01  
## HQ(n) -3.208156e+01 -3.214351e+01 -3.215322e+01 -3.220172e+01  
## SC(n) -3.203095e+01 -3.206760e+01 -3.205200e+01 -3.207519e+01  
## FPE(n) 1.128968e-14 1.043614e-14 1.016457e-14 9.523438e-15  
## 5 6 7 8  
## AIC(n) -3.238286e+01 -3.240557e+01 -3.251731e+01 -3.250092e+01  
## HQ(n) -3.228287e+01 -3.228892e+01 -3.238400e+01 -3.235094e+01  
## SC(n) -3.213103e+01 -3.211177e+01 -3.218155e+01 -3.212319e+01  
## FPE(n) 8.636203e-15 8.442524e-15 7.550167e-15 7.675260e-15  
## 9 10  
## AIC(n) -3.249322e+01 -3.248937e+01  
## HQ(n) -3.232658e+01 -3.230606e+01  
## SC(n) -3.207352e+01 -3.202769e+01  
## FPE(n) 7.735015e-15 7.765422e-15

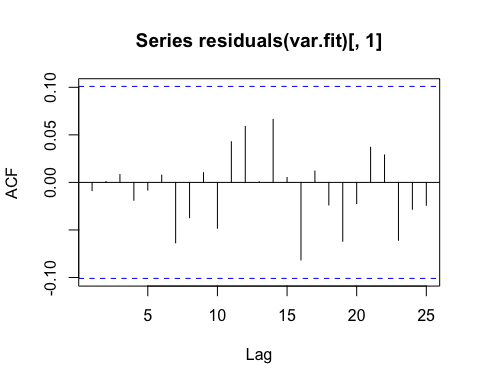
var.fit <- VAR(data,p=7,type = 'both')  
var.fit

##   
## VAR Estimation Results:  
## =======================   
##   
## Estimated coefficients for equation diff.weekly\_train.:   
## =======================================================   
## Call:  
## diff.weekly\_train. = diff.weekly\_train..l1 + en\_weekly\_train\_trans.l1 + diff.weekly\_train..l2 + en\_weekly\_train\_trans.l2 + diff.weekly\_train..l3 + en\_weekly\_train\_trans.l3 + diff.weekly\_train..l4 + en\_weekly\_train\_trans.l4 + diff.weekly\_train..l5 + en\_weekly\_train\_trans.l5 + diff.weekly\_train..l6 + en\_weekly\_train\_trans.l6 + diff.weekly\_train..l7 + en\_weekly\_train\_trans.l7 + const + trend   
##   
## diff.weekly\_train..l1 en\_weekly\_train\_trans.l1 diff.weekly\_train..l2   
## -3.671840e-01 2.565011e+02 -2.564992e-01   
## en\_weekly\_train\_trans.l2 diff.weekly\_train..l3 en\_weekly\_train\_trans.l3   
## 6.631360e+01 -1.592034e-01 8.548640e+01   
## diff.weekly\_train..l4 en\_weekly\_train\_trans.l4 diff.weekly\_train..l5   
## -1.670092e-01 8.661940e+01 -1.042715e-01   
## en\_weekly\_train\_trans.l5 diff.weekly\_train..l6 en\_weekly\_train\_trans.l6   
## -6.597137e+01 -1.440307e-01 3.350486e+01   
## diff.weekly\_train..l7 en\_weekly\_train\_trans.l7 const   
## 9.816374e-02 1.408399e+02 1.804432e-03   
## trend   
## -5.655941e-06   
##   
##   
## Estimated coefficients for equation en\_weekly\_train\_trans:   
## ==========================================================   
## Call:  
## en\_weekly\_train\_trans = diff.weekly\_train..l1 + en\_weekly\_train\_trans.l1 + diff.weekly\_train..l2 + en\_weekly\_train\_trans.l2 + diff.weekly\_train..l3 + en\_weekly\_train\_trans.l3 + diff.weekly\_train..l4 + en\_weekly\_train\_trans.l4 + diff.weekly\_train..l5 + en\_weekly\_train\_trans.l5 + diff.weekly\_train..l6 + en\_weekly\_train\_trans.l6 + diff.weekly\_train..l7 + en\_weekly\_train\_trans.l7 + const + trend   
##   
## diff.weekly\_train..l1 en\_weekly\_train\_trans.l1 diff.weekly\_train..l2   
## -2.000561e-04 5.154829e-02 -5.259392e-05   
## en\_weekly\_train\_trans.l2 diff.weekly\_train..l3 en\_weekly\_train\_trans.l3   
## -2.097410e-01 -7.453965e-06 -1.397477e-01   
## diff.weekly\_train..l4 en\_weekly\_train\_trans.l4 diff.weekly\_train..l5   
## -2.030490e-05 -1.315689e-01 -3.061543e-05   
## en\_weekly\_train\_trans.l5 diff.weekly\_train..l6 en\_weekly\_train\_trans.l6   
## -1.737779e-01 4.008594e-05 1.222125e-03   
## diff.weekly\_train..l7 en\_weekly\_train\_trans.l7 const   
## 8.080547e-06 3.177882e-01 -9.149602e-08   
## trend   
## 2.011773e-09

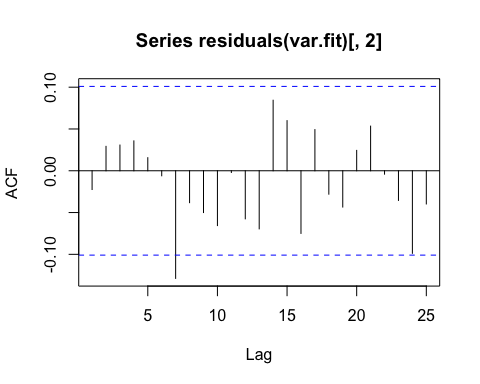
AIC(var.fit) #-10114.87

## [1] -10114.87

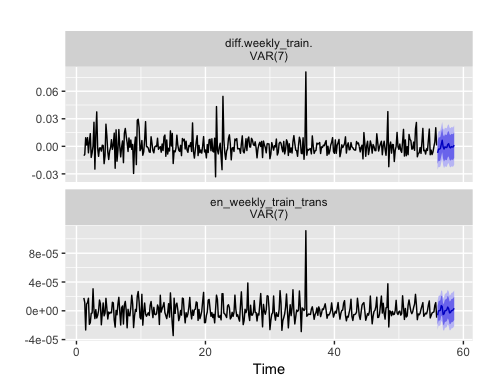
acf(residuals(var.fit)[,1])



acf(residuals(var.fit)[,2]) #english trains still have spikes.



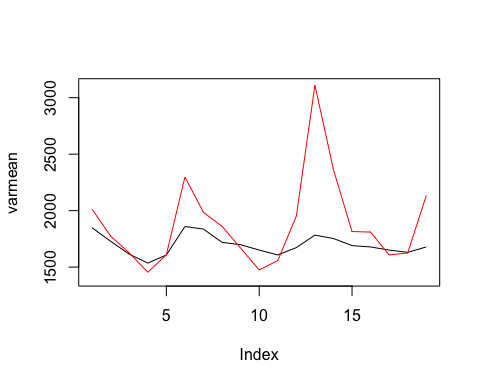
varfit <- forecast(var.fit,h)  
autoplot(varfit)



varfit.cum <- append(weekly\_train[385],varfit$forecast$diff.weekly\_train$mean)  
varfit.cumsum <- cumsum(varfit.cum)  
varfit.cumsum

## [1] 2.312734 2.305581 2.302578 2.299427 2.297059 2.299221 2.305848  
## [8] 2.305294 2.302330 2.301767 2.300465 2.299230 2.301073 2.303983  
## [15] 2.303222 2.301555 2.301231 2.300458 2.299895 2.301243

varmean <- InvBoxCox(varfit.cumsum[2:20],lambda\_weekly)  
  
plot(varmean, type='l',ylim=c(1400,3100)) + lines(as.numeric(es\_weekly\_test), col='red')



## integer(0)

sqrt(sum((es\_weekly\_test-varmean)^2)/h) #380.8135

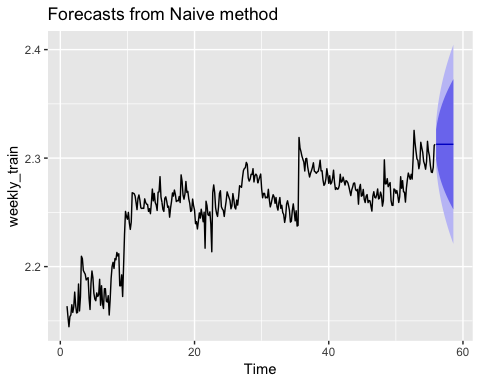
## [1] 380.8135

var <- accuracy(varmean,es\_weekly\_test)

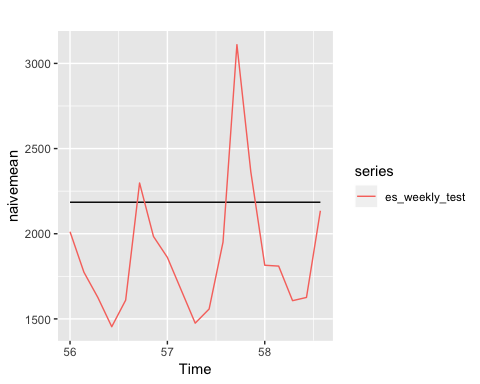
It seems like VAR predict the english page difference better than the Spanish page difference. While when we transform it back to the normal data, it seems not good as others.

## Naive method

naive <- naive(weekly\_train,h)  
autoplot(naive)



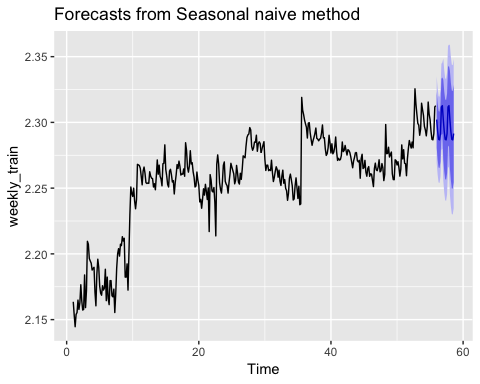
naivemean <- InvBoxCox(naive$mean,lambda\_weekly)  
autoplot(naivemean) + autolayer(es\_weekly\_test)



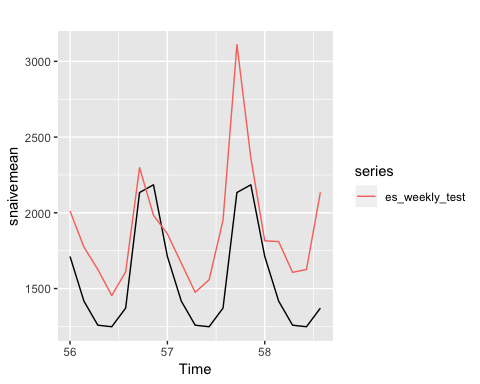
sqrt(sum((es\_weekly\_test-naivemean)^2)/h) #490.2889

## [1] 490.2889

naiveacc <- accuracy(naivemean,es\_weekly\_test)  
  
snaive <- snaive(weekly\_train,h)  
autoplot(snaive)



snaivemean <- InvBoxCox(snaive$mean,lambda\_weekly)  
autoplot(snaivemean) + autolayer(es\_weekly\_test)



sqrt(sum((es\_weekly\_test-snaivemean)^2)/h) #402.7447

## [1] 402.7447

snaiveacc <- accuracy(snaivemean,es\_weekly\_test)

## Holt-Winter

hw1 <- hw(weekly\_train,h,seasonal = 'additive', damped = TRUE)  
hw2 <- hw(weekly\_train,h,seasonal = 'multi', damped = TRUE)  
hw3 <- hw(weekly\_train,h,seasonal = 'addi', damped = FALSE)  
hw4 <- hw(weekly\_train,h,seasonal = 'multi', damped = FALSE)  
summary(hw1) #-1266.789

##   
## Forecast method: Damped Holt-Winters' additive method  
##   
## Model Information:  
## Damped Holt-Winters' additive method   
##   
## Call:  
## hw(y = weekly\_train, h = h, seasonal = "additive", damped = TRUE)   
##   
## Smoothing parameters:  
## alpha = 0.5102   
## beta = 1e-04   
## gamma = 2e-04   
## phi = 0.9701   
##   
## Initial states:  
## l = 2.1325   
## b = 0.0054   
## s = 0.0044 0.0084 -7e-04 -0.007 -0.0049 -4e-04  
## 1e-04  
##   
## sigma: 0.0097  
##   
## AIC AICc BIC   
## -1266.789 -1265.808 -1215.397   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE  
## Training set -2.313728e-05 0.009508506 0.006347621 -0.002891473 0.2826148  
## MASE ACF1  
## Training set 0.5570961 0.2212949  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 56.00000 2.303612 2.291231 2.315992 2.284678 2.322545  
## 56.14286 2.303053 2.289154 2.316952 2.281796 2.324310  
## 56.28571 2.298583 2.283315 2.313852 2.275233 2.321934  
## 56.42857 2.296439 2.279915 2.312964 2.271167 2.321712  
## 56.57143 2.302730 2.285038 2.320423 2.275672 2.329789  
## 56.71429 2.311901 2.293113 2.330690 2.283167 2.340636  
## 56.85714 2.307898 2.288074 2.327722 2.277580 2.338216  
## 57.00000 2.303633 2.282823 2.324442 2.271808 2.335458  
## 57.14286 2.303074 2.281324 2.324823 2.269811 2.336336  
## 57.28571 2.298603 2.275952 2.321254 2.263962 2.333245  
## 57.42857 2.296459 2.272941 2.319977 2.260491 2.332426  
## 57.57143 2.302749 2.278394 2.327104 2.265502 2.339996  
## 57.71429 2.311920 2.286756 2.337083 2.273435 2.350404  
## 57.85714 2.307915 2.281968 2.333863 2.268232 2.347599  
## 58.00000 2.303650 2.276940 2.330359 2.262801 2.344498  
## 58.14286 2.303090 2.275641 2.330540 2.261110 2.345071  
## 58.28571 2.298619 2.270449 2.326790 2.255536 2.341703  
## 58.42857 2.296474 2.267600 2.325348 2.252316 2.340633  
## 58.57143 2.302764 2.273204 2.332325 2.257556 2.347973

summary(hw2) #-1278.316

##   
## Forecast method: Damped Holt-Winters' multiplicative method  
##   
## Model Information:  
## Damped Holt-Winters' multiplicative method   
##   
## Call:  
## hw(y = weekly\_train, h = h, seasonal = "multi", damped = TRUE)   
##   
## Smoothing parameters:  
## alpha = 0.6485   
## beta = 1e-04   
## gamma = 0.0067   
## phi = 0.9781   
##   
## Initial states:  
## l = 2.1674   
## b = 8e-04   
## s = 1.001 1.0036 0.9995 0.9967 0.9984 1.0003  
## 1.0004  
##   
## sigma: 0.0042  
##   
## AIC AICc BIC   
## -1278.316 -1277.335 -1226.924   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE  
## Training set 0.0004121906 0.009282216 0.006055299 0.01718599 0.2693887  
## MASE ACF1  
## Training set 0.5314406 0.09310895  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 56.00000 2.306986 2.294506 2.319466 2.287899 2.326073  
## 56.14286 2.306716 2.291842 2.321589 2.283968 2.329463  
## 56.28571 2.302838 2.285933 2.319744 2.276983 2.328693  
## 56.42857 2.299214 2.280505 2.317923 2.270601 2.327827  
## 56.57143 2.305523 2.285091 2.325955 2.274275 2.336771  
## 56.71429 2.314837 2.292771 2.336903 2.281090 2.348584  
## 56.85714 2.309186 2.285727 2.332644 2.273309 2.345062  
## 57.00000 2.307014 2.282189 2.331840 2.269048 2.344981  
## 57.14286 2.306743 2.280632 2.332855 2.266809 2.346678  
## 57.28571 2.302866 2.275571 2.330160 2.261122 2.344609  
## 57.42857 2.299240 2.270816 2.327664 2.255770 2.342711  
## 57.57143 2.305550 2.275918 2.335181 2.260232 2.350867  
## 57.71429 2.314863 2.284019 2.345706 2.267691 2.362034  
## 57.85714 2.309211 2.277389 2.341032 2.260544 2.357877  
## 58.00000 2.307039 2.274208 2.339870 2.256828 2.357249  
## 58.14286 2.306767 2.272953 2.340582 2.255053 2.358482  
## 58.28571 2.302889 2.268173 2.337604 2.249796 2.355982  
## 58.42857 2.299263 2.263671 2.334855 2.244830 2.353696  
## 58.57143 2.305572 2.268972 2.342172 2.249597 2.361546

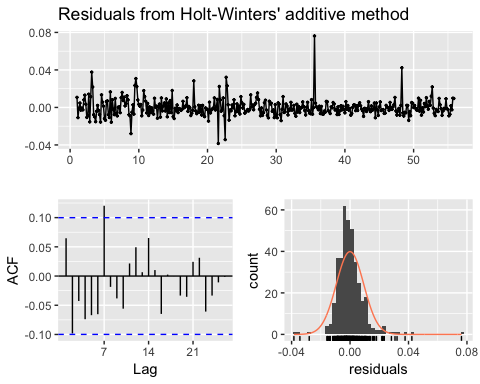
summary(hw3) #-1279.266 choose this one

##   
## Forecast method: Holt-Winters' additive method  
##   
## Model Information:  
## Holt-Winters' additive method   
##   
## Call:  
## hw(y = weekly\_train, h = h, seasonal = "addi", damped = FALSE)   
##   
## Smoothing parameters:  
## alpha = 0.6709   
## beta = 1e-04   
## gamma = 0.0442   
##   
## Initial states:  
## l = 2.1535   
## b = 4e-04   
## s = 0.0038 0.0089 2e-04 -0.0053 -0.0092 0.0028  
## -0.0011  
##   
## sigma: 0.0095  
##   
## AIC AICc BIC   
## -1279.266 -1278.428 -1231.828   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE  
## Training set -3.546339e-05 0.00938 0.006275509 -0.002370534 0.2790369  
## MASE ACF1  
## Training set 0.5507673 0.06469731  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 56.00000 2.307266 2.295069 2.319462 2.288613 2.325919  
## 56.14286 2.306207 2.291519 2.320894 2.283744 2.328669  
## 56.28571 2.304658 2.287844 2.321472 2.278943 2.330372  
## 56.42857 2.300450 2.281749 2.319150 2.271850 2.329050  
## 56.57143 2.308172 2.287758 2.328586 2.276951 2.339392  
## 56.71429 2.318289 2.296294 2.340284 2.284650 2.351927  
## 56.85714 2.312929 2.289459 2.336400 2.277035 2.348824  
## 57.00000 2.310187 2.285146 2.335229 2.271890 2.348485  
## 57.14286 2.309128 2.282781 2.335475 2.268834 2.349423  
## 57.28571 2.307579 2.279988 2.335171 2.265382 2.349777  
## 57.42857 2.303371 2.274588 2.332154 2.259352 2.347390  
## 57.57143 2.311093 2.281167 2.341020 2.265324 2.356862  
## 57.71429 2.321210 2.290181 2.352239 2.273755 2.368665  
## 57.85714 2.315851 2.283757 2.347945 2.266768 2.364934  
## 58.00000 2.313109 2.279847 2.346371 2.262239 2.363979  
## 58.14286 2.312050 2.277792 2.346308 2.259657 2.364443  
## 58.28571 2.310501 2.275275 2.345727 2.256627 2.364375  
## 58.42857 2.306293 2.270124 2.342461 2.250978 2.361608  
## 58.57143 2.314015 2.276928 2.351102 2.257295 2.370735

summary(hw4) #-1256.880

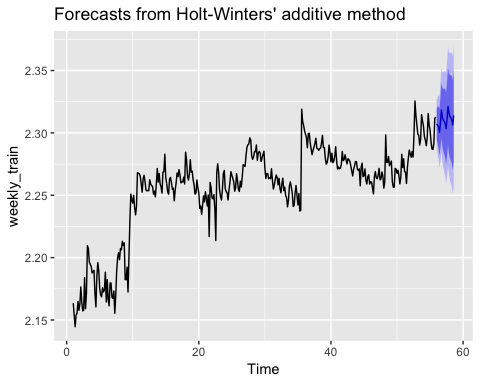
##   
## Forecast method: Holt-Winters' multiplicative method  
##   
## Model Information:  
## Holt-Winters' multiplicative method   
##   
## Call:  
## hw(y = weekly\_train, h = h, seasonal = "multi", damped = FALSE)   
##   
## Smoothing parameters:  
## alpha = 0.5942   
## beta = 1e-04   
## gamma = 0.0762   
##   
## Initial states:  
## l = 2.1573   
## b = 5e-04   
## s = 0.9997 1.0044 1.0026 0.9952 0.9957 1.0033  
## 0.9991  
##   
## sigma: 0.0043  
##   
## AIC AICc BIC   
## -1256.880 -1256.041 -1209.441   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE  
## Training set -0.0001522625 0.009581398 0.006453973 -0.007591547 0.286991  
## MASE ACF1  
## Training set 0.56643 0.129999  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 56.00000 2.305628 2.292790 2.318467 2.285994 2.325263  
## 56.14286 2.303914 2.288992 2.318836 2.281093 2.326736  
## 56.28571 2.303772 2.287017 2.320527 2.278148 2.329397  
## 56.42857 2.299169 2.280799 2.317539 2.271074 2.327264  
## 56.57143 2.307258 2.287306 2.327210 2.276744 2.337773  
## 56.71429 2.318854 2.297384 2.340323 2.286019 2.351689  
## 56.85714 2.312393 2.289658 2.335129 2.277623 2.347164  
## 57.00000 2.308906 2.284624 2.333187 2.271770 2.346041  
## 57.14286 2.307189 2.281753 2.332624 2.268289 2.346089  
## 57.28571 2.307046 2.280492 2.333599 2.266435 2.347656  
## 57.42857 2.302435 2.274862 2.330008 2.260266 2.344605  
## 57.57143 2.310535 2.281829 2.339241 2.266633 2.354437  
## 57.71429 2.322147 2.292292 2.352002 2.276487 2.367806  
## 57.85714 2.315676 2.284935 2.346417 2.268662 2.362691  
## 58.00000 2.312183 2.280299 2.344067 2.263421 2.360946  
## 58.14286 2.310463 2.277699 2.343227 2.260355 2.360571  
## 58.28571 2.310319 2.276678 2.343960 2.258870 2.361768  
## 58.42857 2.305701 2.271273 2.340130 2.253048 2.358355  
## 58.57143 2.313812 2.278426 2.349199 2.259693 2.367931

checkresiduals(hw3) #failed

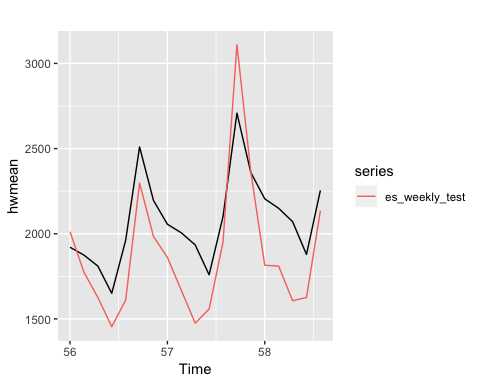


##   
## Ljung-Box test  
##   
## data: Residuals from Holt-Winters' additive method  
## Q\* = 22.199, df = 3, p-value = 5.929e-05  
##   
## Model df: 11. Total lags used: 14

autoplot(hw3)



hwmean <- InvBoxCox(hw3$mean,lambda\_weekly)  
autoplot(hwmean) + autolayer(es\_weekly\_test)



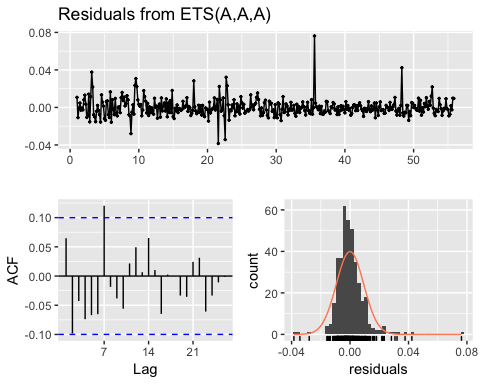
hwacc <- accuracy(hwmean,es\_weekly\_test) #276.3999

## ETS

ets <- ets(weekly\_train, model = 'ZZA') #AAA  
summary(ets) #AIC -1279.266

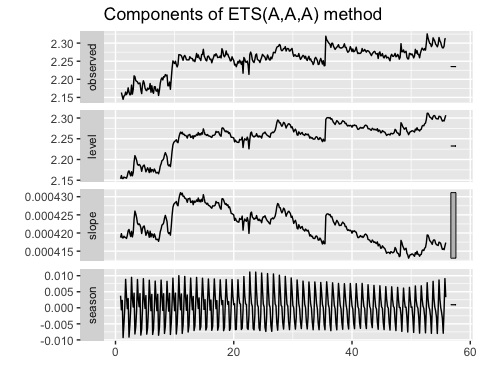
## ETS(A,A,A)   
##   
## Call:  
## ets(y = weekly\_train, model = "ZZA")   
##   
## Smoothing parameters:  
## alpha = 0.6709   
## beta = 1e-04   
## gamma = 0.0442   
##   
## Initial states:  
## l = 2.1535   
## b = 4e-04   
## s = 0.0038 0.0089 2e-04 -0.0053 -0.0092 0.0028  
## -0.0011  
##   
## sigma: 0.0095  
##   
## AIC AICc BIC   
## -1279.266 -1278.428 -1231.828   
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE  
## Training set -3.546339e-05 0.00938 0.006275509 -0.002370534 0.2790369  
## MASE ACF1  
## Training set 0.5507673 0.06469731

checkresiduals(ets) #failed

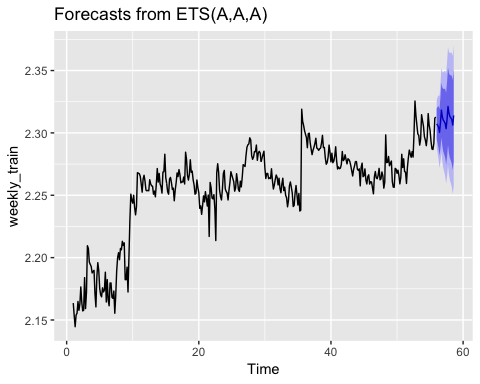


##   
## Ljung-Box test  
##   
## data: Residuals from ETS(A,A,A)  
## Q\* = 22.199, df = 3, p-value = 5.929e-05  
##   
## Model df: 11. Total lags used: 14

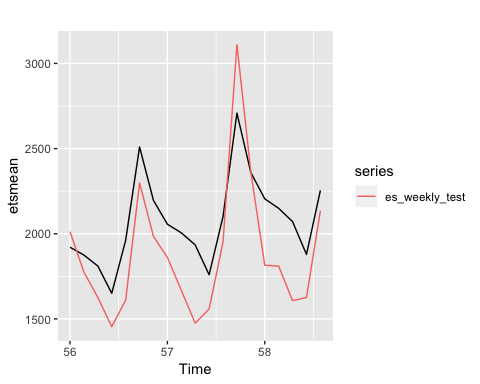
autoplot(ets)



etsfit <- forecast(ets,h)  
autoplot(etsfit)



etsmean <- InvBoxCox(etsfit$mean,lambda\_weekly)  
autoplot(etsmean)+autolayer(es\_weekly\_test)



sqrt(sum((es\_weekly\_test-etsmean)^2)/h) #276.3999

## [1] 276.3999

etsacc <- accuracy(etsmean,es\_weekly\_test)

Same as holt-winter.

## TBATS

tbats(weekly\_train) #AIC -1301.029

## TBATS(1, {1,0}, 0.981, {<7,3>})  
##   
## Call: tbats(y = weekly\_train)  
##   
## Parameters  
## Alpha: 0.2693125  
## Beta: -0.008134144  
## Damping Parameter: 0.981141  
## Gamma-1 Values: 4.997127e-05  
## Gamma-2 Values: -0.0005549377  
## AR coefficients: 0.450808  
##   
## Seed States:  
## [,1]  
## [1,] 2.1348061109  
## [2,] 0.0026189607  
## [3,] 0.0024489206  
## [4,] -0.0024074594  
## [5,] 0.0005487456  
## [6,] -0.0051943770  
## [7,] 0.0019306492  
## [8,] 0.0006130241  
## [9,] 0.0000000000  
##   
## Sigma: 0.009047829  
## AIC: -1301.029

## Fourier & harmonic

#install.packages('fpp2')  
library(fpp2)

## Loading required package: ggplot2

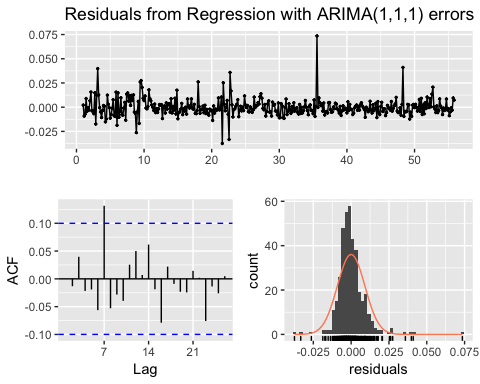
##   
## Attaching package: 'fpp2'

## The following objects are masked from 'package:fpp':  
##   
## ausair, ausbeer, austa, austourists, debitcards, departures,  
## elecequip, euretail, guinearice, oil, sunspotarea, usmelec

harmonics <- fourier(weekly\_train, K=3)  
fit <- auto.arima(weekly\_train,xreg=harmonics,seasonal = FALSE)  
summary(fit) #aic -2498.96

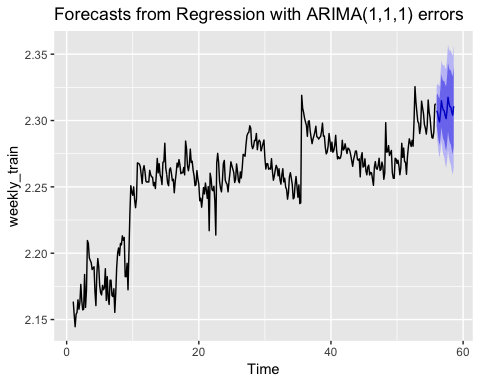
## Series: weekly\_train   
## Regression with ARIMA(1,1,1) errors   
##   
## Coefficients:  
## ar1 ma1 drift S1-7 C1-7 S2-7 C2-7 S3-7  
## 0.4493 -0.7089 4e-04 -0.0014 0.0056 -0.0028 -0.0013 -3e-04  
## s.e. 0.1206 0.0953 3e-04 0.0007 0.0007 0.0005 0.0005 4e-04  
## C3-7  
## -7e-04  
## s.e. 4e-04  
##   
## sigma^2 estimated as 8.488e-05: log likelihood=1259.48  
## AIC=-2498.96 AICc=-2498.37 BIC=-2459.46  
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE  
## Training set -6.269142e-06 0.009092815 0.005974557 -0.001159486 0.2658147  
## MASE ACF1  
## Training set 0.5243543 -0.0002398337

checkresiduals(fit)

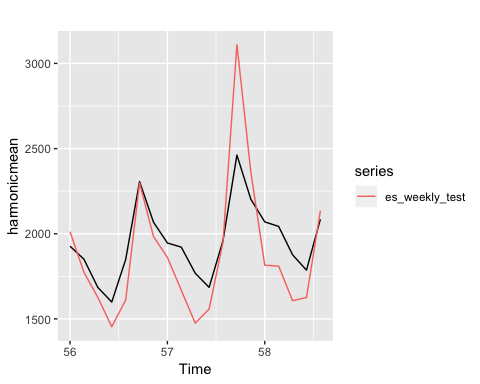


##   
## Ljung-Box test  
##   
## data: Residuals from Regression with ARIMA(1,1,1) errors  
## Q\* = 13.914, df = 5, p-value = 0.01617  
##   
## Model df: 9. Total lags used: 14

newharmonics <- fourier(weekly\_train, K=3,h)  
harmonicfit <- forecast(fit,h,xreg = newharmonics)  
autoplot(harmonicfit)



harmonicmean <- InvBoxCox(harmonicfit$mean,lambda\_weekly)  
autoplot(harmonicmean) + autolayer(es\_weekly\_test)



sqrt(sum((es\_weekly\_test-harmonicmean)^2)/h) #222.4694

## [1] 222.4694

harmonic <- accuracy(harmonicmean,es\_weekly\_test)

Fourier transformation and Dynamic Harnomic has the best prediction, though the residual is not as good as Xreg.

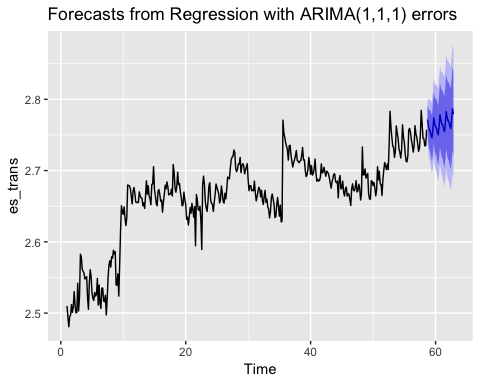
# Comparasion

comparasion <- rbind(Auto = auto, Arima = arima,Xreg = xreg, VAR = var, Naive = naiveacc, SNaive = snaiveacc, harmonic,hwacc, etsacc)  
comparasion <- as.data.frame(comparasion, row.names = c('AutoArima','Arima212','Regression with En','VAR','Naive','SNaive','Dynamic Harmonic','Holt-Winters','ETS'))  
comparasion

## ME RMSE MAE MPE MAPE  
## AutoArima 109.21835 327.0045 180.4965 3.517280 8.109754  
## Arima212 152.16145 345.6329 187.8361 5.866344 8.228730  
## Regression with En 87.02808 246.6643 150.6394 2.958780 6.933370  
## VAR 183.77833 380.8135 224.3673 7.283010 9.970731  
## Naive -304.15789 490.2889 432.0526 -20.141200 24.578435  
## SNaive 320.00000 402.7447 341.1579 16.741580 17.808006  
## Dynamic Harmonic -71.16871 222.4694 170.4067 -5.519133 9.115558  
## Holt-Winters -192.99961 276.3999 245.2026 -11.960733 13.810989  
## ETS -192.99961 276.3999 245.2026 -11.960733 13.810989  
## ACF1 Theil's U  
## AutoArima 0.3629389 0.7731742  
## Arima212 0.3683815 0.8163379  
## Regression with En 0.2761131 0.5797518  
## VAR 0.3735035 0.8865544  
## Naive 0.3700346 1.2888215  
## SNaive 0.2874601 1.0322058  
## Dynamic Harmonic 0.3416682 0.5448327  
## Holt-Winters 0.3432348 0.7080062  
## ETS 0.3432348 0.7080062

# Choose Dynamic Harmonic and Forecasting

lambda <- BoxCox.lambda(rick\_morty\_es\_weekly)  
es\_trans <- BoxCox(rick\_morty\_es\_weekly,lambda)  
harmonics\_es <- fourier(es\_trans, K=3)  
fit\_es <- auto.arima(es\_trans,xreg=harmonics\_es,seasonal = FALSE)  
  
newharmonics\_es <- fourier(es\_trans, K=3,30)  
fit30 <- forecast(fit\_es,30,xreg = newharmonics\_es)  
autoplot(fit30)

 Actually, for the all the prediction, they can predict the trend and the seasonality, but they are not good at predict the variance.