time series

hansdefarmer

8/23/2021

**R-Time series Analysis**

Any metric is measured over a regular time forms a time series.

**Time series**-Is a collection of observations made sequentially in time. Creating a Time series-ts() function is used to create a time series

E.g timeseries<-ts(data,start,end,frequency)

**Data**-Vector or matrix which contains values used in time series **Start**-start time for the first observation. **End**-end time for the last observation. **Frequency**-specifies number of observations per unit time Example ,Consider the annual snowfall details at a place starting from January 2013.

snowfall <- c(790,1170.8,860.1,1330.6,630.4,911.5,683.5,996.6,783.2,982,881.8,1021)  
timeseries <- ts(snowfall,start = c(2013,1),frequency = 12)  
timeseries

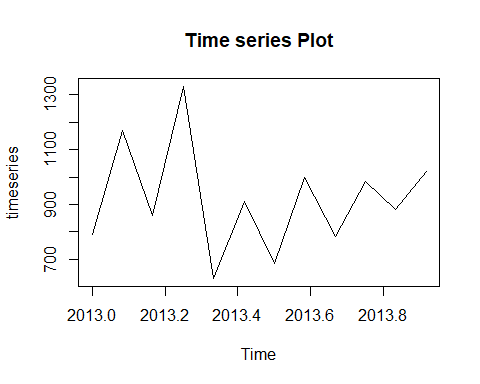
## Jan Feb Mar Apr May Jun Jul Aug Sep Oct  
## 2013 790.0 1170.8 860.1 1330.6 630.4 911.5 683.5 996.6 783.2 982.0  
## Nov Dec  
## 2013 881.8 1021.0

library(tseries)

## Warning: package 'tseries' was built under R version 4.1.3

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

plot.ts(timeseries,main="Time series Plot")

 ########################################################

**Stationary Time Series**

A series is stationary if ;

1. Mean value of series is constant over time.That implies that trend component is declared null.

2.Variance should not increase over time.

3.The seasonality effect should be minimal.

i.e it is a trendseasonal pattern .which resembles a random white noise regardless of time interval observed.

NOTE:Stationary Time Series is the one whose statistical properties like mean ,variance ,and autocorrrelation, etc are all constant over time. ######################################################

Differencing time series data to make it stationary

\*\*forex data AUD\* clearing allvariables in the Workspace

rm(list = ls())  
#loading the forecasting package  
library(fpp2)

## Warning: package 'fpp2' was built under R version 4.1.3

## -- Attaching packages ---------------------------------------------- fpp2 2.4 --

## v ggplot2 3.3.5 v fma 2.4   
## v forecast 8.16 v expsmooth 2.3

## Warning: package 'forecast' was built under R version 4.1.3

## Warning: package 'fma' was built under R version 4.1.3

## Warning: package 'expsmooth' was built under R version 4.1.3

##

Importing the data

jpy=read.csv("C:\\Users\\dennis\\Downloads\\AUD\_JPY (1 T).csv")  
jpy

## Date Time AUD.JPY  
## 1 2022-06-29 06:55:12 93.556  
## 2 2022-06-29 06:55:13 93.566  
## 3 2022-06-29 06:55:14 93.566  
## 4 2022-06-29 06:55:15 93.565  
## 5 2022-06-29 06:55:16 93.561  
## 6 2022-06-29 06:55:17 93.561  
## 7 2022-06-29 06:55:18 93.560  
## 8 2022-06-29 06:55:19 93.559  
## 9 2022-06-29 06:55:20 93.557  
## 10 2022-06-29 06:55:21 93.548  
## 11 2022-06-29 06:55:22 93.560  
## 12 2022-06-29 06:55:23 93.560  
## 13 2022-06-29 06:55:24 93.562  
## 14 2022-06-29 06:55:25 93.562  
## 15 2022-06-29 06:55:26 93.564  
## 16 2022-06-29 06:55:27 93.559  
## 17 2022-06-29 06:55:28 93.561  
## 18 2022-06-29 06:55:29 93.560  
## 19 2022-06-29 06:55:30 93.560  
## 20 2022-06-29 06:55:31 93.552  
## 21 2022-06-29 06:55:32 93.550  
## 22 2022-06-29 06:55:33 93.555  
## 23 2022-06-29 06:55:34 93.550  
## 24 2022-06-29 06:55:35 93.548  
## 25 2022-06-29 06:55:36 93.544  
## 26 2022-06-29 06:55:37 93.543  
## 27 2022-06-29 06:55:38 93.541  
## 28 2022-06-29 06:55:39 93.542  
## 29 2022-06-29 06:55:40 93.546  
## 30 2022-06-29 06:55:41 93.547  
## 31 2022-06-29 06:55:42 93.552  
## 32 2022-06-29 06:55:43 93.552  
## 33 2022-06-29 06:55:44 93.554  
## 34 2022-06-29 06:55:45 93.555  
## 35 2022-06-29 06:55:46 93.551  
## 36 2022-06-29 06:55:47 93.554  
## 37 2022-06-29 06:55:48 93.556  
## 38 2022-06-29 06:55:49 93.559  
## 39 2022-06-29 06:55:50 93.564  
## 40 2022-06-29 06:55:51 93.565  
## 41 2022-06-29 06:55:52 93.565  
## 42 2022-06-29 06:55:53 93.565  
## 43 2022-06-29 06:55:54 93.566  
## 44 2022-06-29 06:55:55 93.567  
## 45 2022-06-29 06:55:56 93.561  
## 46 2022-06-29 06:55:57 93.564  
## 47 2022-06-29 06:55:58 93.566  
## 48 2022-06-29 06:55:59 93.569  
## 49 2022-06-29 06:56:00 93.570  
## 50 2022-06-29 06:56:01 93.573  
## 51 2022-06-29 06:56:02 93.571  
## 52 2022-06-29 06:56:03 93.567  
## 53 2022-06-29 06:56:04 93.563  
## 54 2022-06-29 06:56:05 93.567  
## 55 2022-06-29 06:56:06 93.568  
## 56 2022-06-29 06:56:07 93.566  
## 57 2022-06-29 06:56:08 93.565  
## 58 2022-06-29 06:56:09 93.567  
## 59 2022-06-29 06:56:10 93.572  
## 60 2022-06-29 06:56:11 93.567  
## 61 2022-06-29 06:56:12 93.559  
## 62 2022-06-29 06:56:13 93.559  
## 63 2022-06-29 06:56:14 93.559  
## 64 2022-06-29 06:56:15 93.549  
## 65 2022-06-29 06:56:16 93.545  
## 66 2022-06-29 06:56:17 93.540  
## 67 2022-06-29 06:56:18 93.545  
## 68 2022-06-29 06:56:19 93.545  
## 69 2022-06-29 06:56:20 93.547  
## 70 2022-06-29 06:56:21 93.549  
## 71 2022-06-29 06:56:22 93.544  
## 72 2022-06-29 06:56:23 93.544  
## 73 2022-06-29 06:56:24 93.544  
## 74 2022-06-29 06:56:25 93.543  
## 75 2022-06-29 06:56:26 93.543  
## 76 2022-06-29 06:56:27 93.545  
## 77 2022-06-29 06:56:28 93.542  
## 78 2022-06-29 06:56:29 93.543  
## 79 2022-06-29 06:56:30 93.541  
## 80 2022-06-29 06:56:31 93.546  
## 81 2022-06-29 06:56:32 93.547  
## 82 2022-06-29 06:56:33 93.543  
## 83 2022-06-29 06:56:34 93.543  
## 84 2022-06-29 06:56:35 93.539  
## 85 2022-06-29 06:56:36 93.535  
## 86 2022-06-29 06:56:37 93.540  
## 87 2022-06-29 06:56:38 93.541  
## 88 2022-06-29 06:56:39 93.540  
## 89 2022-06-29 06:56:40 93.536  
## 90 2022-06-29 06:56:41 93.537  
## 91 2022-06-29 06:56:42 93.541  
## 92 2022-06-29 06:56:43 93.537  
## 93 2022-06-29 06:56:44 93.540  
## 94 2022-06-29 06:56:45 93.541  
## 95 2022-06-29 06:56:46 93.543  
## 96 2022-06-29 06:56:47 93.538  
## 97 2022-06-29 06:56:48 93.542  
## 98 2022-06-29 06:56:49 93.544  
## 99 2022-06-29 06:56:50 93.542  
## 100 2022-06-29 06:56:51 93.543

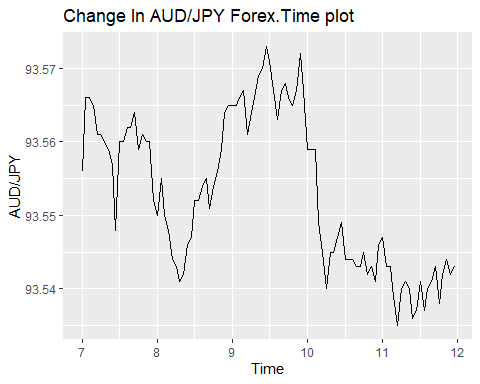
jpy=jpy[,3]  
jpy

## [1] 93.556 93.566 93.566 93.565 93.561 93.561 93.560 93.559 93.557 93.548  
## [11] 93.560 93.560 93.562 93.562 93.564 93.559 93.561 93.560 93.560 93.552  
## [21] 93.550 93.555 93.550 93.548 93.544 93.543 93.541 93.542 93.546 93.547  
## [31] 93.552 93.552 93.554 93.555 93.551 93.554 93.556 93.559 93.564 93.565  
## [41] 93.565 93.565 93.566 93.567 93.561 93.564 93.566 93.569 93.570 93.573  
## [51] 93.571 93.567 93.563 93.567 93.568 93.566 93.565 93.567 93.572 93.567  
## [61] 93.559 93.559 93.559 93.549 93.545 93.540 93.545 93.545 93.547 93.549  
## [71] 93.544 93.544 93.544 93.543 93.543 93.545 93.542 93.543 93.541 93.546  
## [81] 93.547 93.543 93.543 93.539 93.535 93.540 93.541 93.540 93.536 93.537  
## [91] 93.541 93.537 93.540 93.541 93.543 93.538 93.542 93.544 93.542 93.543

#converting to time series data. Time is in seconds i.e, changes occuring in every second  
jpyt=ts(jpy,start = c(7,1),frequency = 20)

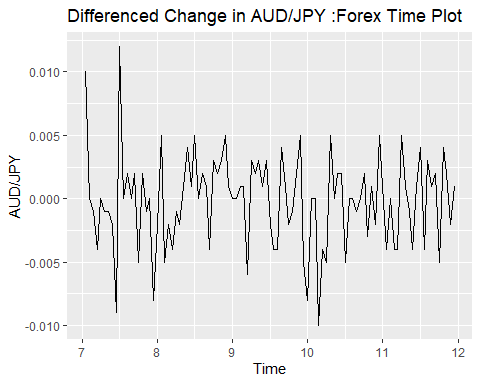
Preliminary analysis

#Time plot  
autoplot(jpyt)+ggtitle("Change In AUD/JPY Forex.Time plot")+  
 ylab("AUD/JPY")



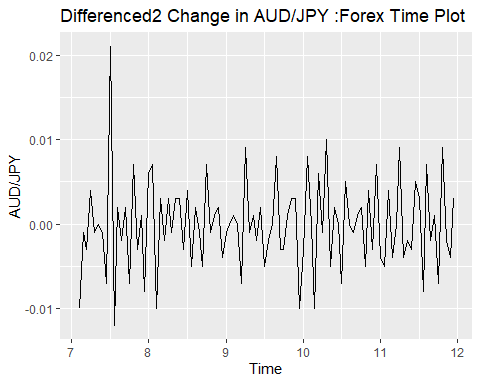
Differencing the data to remove trend

dy=diff(jpyt)  
#plotting differeced data.  
autoplot(dy)+ggtitle("Differenced Change in AUD/JPY :Forex Time Plot")+  
 ylab("AUD/JPY")

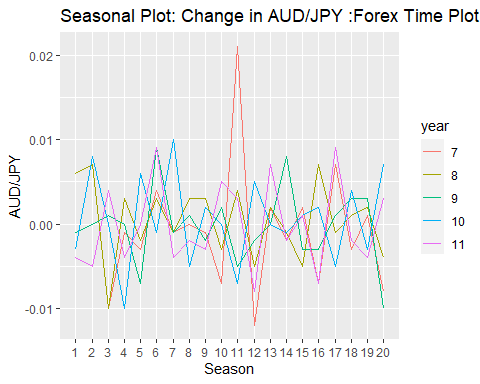


Differencing the second time to make the data stationary and remove trend

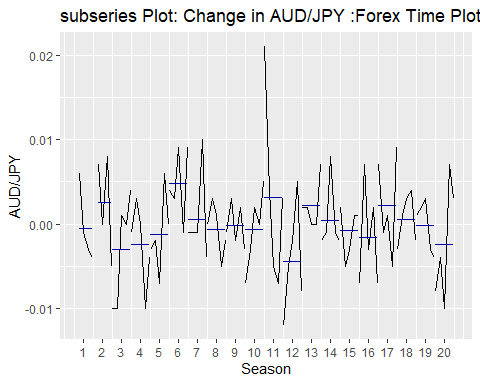
dy0=diff(dy)  
#plotting differeced data.  
autoplot(dy0)+ggtitle("Differenced2 Change in AUD/JPY :Forex Time Plot")+  
 ylab("AUD/JPY")

 series appear trend-stationary,use to investigate seasonality.

ggseasonplot(dy0)+ggtitle("Seasonal Plot: Change in AUD/JPY :Forex Time Plot")+  
 ylab("AUD/JPY")

 Another seasonal plot, the subseries plot.

ggsubseriesplot(dy0)+ggtitle("subseries Plot: Change in AUD/JPY :Forex Time Plot")+  
 ylab("AUD/JPY")



#\*\*Each season requires at least 2 observations\*\* for doing sub series plot

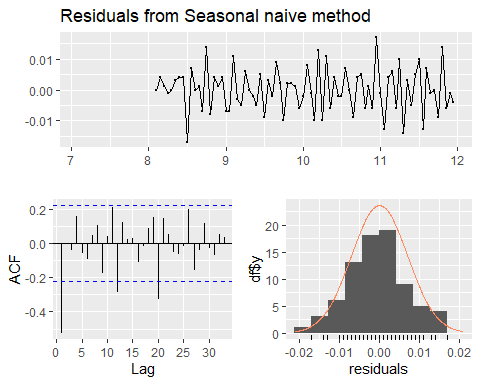
If our model has seasonality and trend we take the first difference. Forecast with various methods

use benchmark method to forecast. Using seasonal naive method as our benchmark y\_t = y\_(t-s)+e\_t

fit=snaive(dy0) #residual=0.007  
summary(fit)

##   
## Forecast method: Seasonal naive method  
##   
## Model Information:  
## Call: snaive(y = dy0)   
##   
## Residual sd: 0.007   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE ACF1  
## Training set 1.282051e-05 0.006999084 0.005551282 -Inf Inf 1 -0.5239652  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 12.00 -0.004 -0.0129696873 0.0049696873 -0.017717953 0.009717953  
## 12.05 -0.005 -0.0139696873 0.0039696873 -0.018717953 0.008717953  
## 12.10 0.004 -0.0049696873 0.0129696873 -0.009717953 0.017717953  
## 12.15 -0.004 -0.0129696873 0.0049696873 -0.017717953 0.009717953  
## 12.20 0.000 -0.0089696873 0.0089696873 -0.013717953 0.013717953  
## 12.25 0.009 0.0000303127 0.0179696873 -0.004717953 0.022717953  
## 12.30 -0.004 -0.0129696873 0.0049696873 -0.017717953 0.009717953  
## 12.35 -0.002 -0.0109696873 0.0069696873 -0.015717953 0.011717953  
## 12.40 -0.003 -0.0119696873 0.0059696873 -0.016717953 0.010717953  
## 12.45 0.005 -0.0039696873 0.0139696873 -0.008717953 0.018717953  
## 12.50 0.003 -0.0059696873 0.0119696873 -0.010717953 0.016717953  
## 12.55 -0.008 -0.0169696873 0.0009696873 -0.021717953 0.005717953  
## 12.60 0.007 -0.0019696873 0.0159696873 -0.006717953 0.020717953  
## 12.65 -0.002 -0.0109696873 0.0069696873 -0.015717953 0.011717953  
## 12.70 0.001 -0.0079696873 0.0099696873 -0.012717953 0.014717953  
## 12.75 -0.007 -0.0159696873 0.0019696873 -0.020717953 0.006717953  
## 12.80 0.009 0.0000303127 0.0179696873 -0.004717953 0.022717953  
## 12.85 -0.002 -0.0109696873 0.0069696873 -0.015717953 0.011717953  
## 12.90 -0.004 -0.0129696873 0.0049696873 -0.017717953 0.009717953  
## 12.95 0.003 -0.0059696873 0.0119696873 -0.010717953 0.016717953  
## 13.00 -0.004 -0.0166850534 0.0086850534 -0.023400115 0.015400115  
## 13.05 -0.005 -0.0176850534 0.0076850534 -0.024400115 0.014400115  
## 13.10 0.004 -0.0086850534 0.0166850534 -0.015400115 0.023400115  
## 13.15 -0.004 -0.0166850534 0.0086850534 -0.023400115 0.015400115  
## 13.20 0.000 -0.0126850534 0.0126850534 -0.019400115 0.019400115  
## 13.25 0.009 -0.0036850534 0.0216850534 -0.010400115 0.028400115  
## 13.30 -0.004 -0.0166850534 0.0086850534 -0.023400115 0.015400115  
## 13.35 -0.002 -0.0146850534 0.0106850534 -0.021400115 0.017400115  
## 13.40 -0.003 -0.0156850534 0.0096850534 -0.022400115 0.016400115  
## 13.45 0.005 -0.0076850534 0.0176850534 -0.014400115 0.024400115  
## 13.50 0.003 -0.0096850534 0.0156850534 -0.016400115 0.022400115  
## 13.55 -0.008 -0.0206850534 0.0046850534 -0.027400115 0.011400115  
## 13.60 0.007 -0.0056850534 0.0196850534 -0.012400115 0.026400115  
## 13.65 -0.002 -0.0146850534 0.0106850534 -0.021400115 0.017400115  
## 13.70 0.001 -0.0116850534 0.0136850534 -0.018400115 0.020400115  
## 13.75 -0.007 -0.0196850534 0.0056850534 -0.026400115 0.012400115  
## 13.80 0.009 -0.0036850534 0.0216850534 -0.010400115 0.028400115  
## 13.85 -0.002 -0.0146850534 0.0106850534 -0.021400115 0.017400115  
## 13.90 -0.004 -0.0166850534 0.0086850534 -0.023400115 0.015400115  
## 13.95 0.003 -0.0096850534 0.0156850534 -0.016400115 0.022400115

checkresiduals(fit)



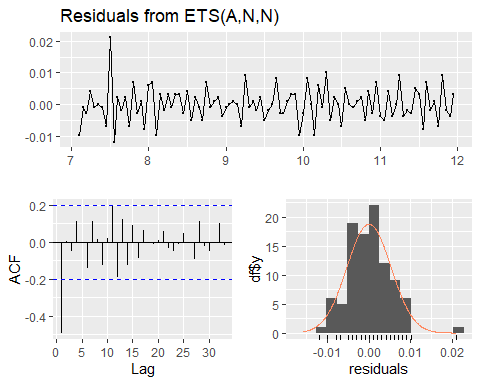
##   
## Ljung-Box test  
##   
## data: Residuals from Seasonal naive method  
## Q\* = 58.972, df = 20, p-value = 1.026e-05  
##   
## Model df: 0. Total lags used: 20

Fit exponential smoothing Fit ETS method

fit\_ets=ets(dy0) #residual= 0.0728011  
summary(fit\_ets)

## ETS(A,N,N)   
##   
## Call:  
## ets(y = dy0)   
##   
## Smoothing parameters:  
## alpha = 1e-04   
##   
## Initial states:  
## l = -1e-04   
##   
## sigma: 0.0053  
##   
## AIC AICc BIC   
## -573.3873 -573.1320 -565.6324   
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE MASE ACF1  
## Training set 1.675022e-06 0.005255177 0.004012309 Inf Inf 0.7227716 -0.4900633

checkresiduals(fit\_ets)



##   
## Ljung-Box test  
##   
## data: Residuals from ETS(A,N,N)  
## Q\* = 45.615, df = 18, p-value = 0.0003375  
##   
## Model df: 2. Total lags used: 20

fitting arima model

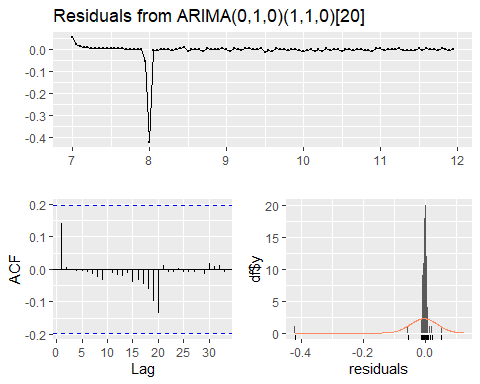
fit\_arima=auto.arima(jpyt,d=1,D=1,stepwise = FALSE,approximation = FALSE,trace = TRUE) #residual= 0.04872371

##   
## ARIMA(0,1,0)(0,1,0)[20] : -617.13  
## ARIMA(0,1,0)(0,1,1)[20] : -631.8449  
## ARIMA(0,1,0)(1,1,0)[20] : -632.9464  
## ARIMA(0,1,0)(1,1,1)[20] : -630.8486  
## ARIMA(0,1,1)(0,1,0)[20] : -615.2451  
## ARIMA(0,1,1)(0,1,1)[20] : -630.6738  
## ARIMA(0,1,1)(1,1,0)[20] : -631.6356  
## ARIMA(0,1,1)(1,1,1)[20] : -629.4158  
## ARIMA(0,1,2)(0,1,0)[20] : -613.0854  
## ARIMA(0,1,2)(0,1,1)[20] : -628.4624  
## ARIMA(0,1,2)(1,1,0)[20] : -629.4166  
## ARIMA(0,1,2)(1,1,1)[20] : -627.1362  
## ARIMA(0,1,3)(0,1,0)[20] : -611.5752  
## ARIMA(0,1,3)(0,1,1)[20] : -626.2083  
## ARIMA(0,1,3)(1,1,0)[20] : -627.3483  
## ARIMA(0,1,3)(1,1,1)[20] : -625.0265  
## ARIMA(0,1,4)(0,1,0)[20] : -612.4637  
## ARIMA(0,1,4)(0,1,1)[20] : -624.4351  
## ARIMA(0,1,4)(1,1,0)[20] : -625.4641  
## ARIMA(0,1,5)(0,1,0)[20] : -610.121  
## ARIMA(1,1,0)(0,1,0)[20] : -615.2455  
## ARIMA(1,1,0)(0,1,1)[20] : -630.6799  
## ARIMA(1,1,0)(1,1,0)[20] : -631.6431  
## ARIMA(1,1,0)(1,1,1)[20] : -629.4239  
## ARIMA(1,1,1)(0,1,0)[20] : -613.0838  
## ARIMA(1,1,1)(0,1,1)[20] : -628.461  
## ARIMA(1,1,1)(1,1,0)[20] : -629.5236  
## ARIMA(1,1,1)(1,1,1)[20] : -627.254  
## ARIMA(1,1,2)(0,1,0)[20] : -612.5305  
## ARIMA(1,1,2)(0,1,1)[20] : -626.592  
## ARIMA(1,1,2)(1,1,0)[20] : -627.2949  
## ARIMA(1,1,2)(1,1,1)[20] : -624.9536  
## ARIMA(1,1,3)(0,1,0)[20] : -611.1438  
## ARIMA(1,1,3)(0,1,1)[20] : -624.3759  
## ARIMA(1,1,3)(1,1,0)[20] : -625.2368  
## ARIMA(1,1,4)(0,1,0)[20] : -610.12  
## ARIMA(2,1,0)(0,1,0)[20] : -613.0848  
## ARIMA(2,1,0)(0,1,1)[20] : -628.4611  
## ARIMA(2,1,0)(1,1,0)[20] : -629.4244  
## ARIMA(2,1,0)(1,1,1)[20] : -627.1442  
## ARIMA(2,1,1)(0,1,0)[20] : -612.0823  
## ARIMA(2,1,1)(0,1,1)[20] : Inf  
## ARIMA(2,1,1)(1,1,0)[20] : Inf  
## ARIMA(2,1,1)(1,1,1)[20] : -624.9436  
## ARIMA(2,1,2)(0,1,0)[20] : Inf  
## ARIMA(2,1,2)(0,1,1)[20] : Inf  
## ARIMA(2,1,2)(1,1,0)[20] : Inf  
## ARIMA(2,1,3)(0,1,0)[20] : Inf  
## ARIMA(3,1,0)(0,1,0)[20] : -611.1239  
## ARIMA(3,1,0)(0,1,1)[20] : -626.1798  
## ARIMA(3,1,0)(1,1,0)[20] : -627.4687  
## ARIMA(3,1,0)(1,1,1)[20] : -625.1611  
## ARIMA(3,1,1)(0,1,0)[20] : -610.9281  
## ARIMA(3,1,1)(0,1,1)[20] : -624.2706  
## ARIMA(3,1,1)(1,1,0)[20] : -625.2446  
## ARIMA(3,1,2)(0,1,0)[20] : Inf  
## ARIMA(4,1,0)(0,1,0)[20] : -613.1485  
## ARIMA(4,1,0)(0,1,1)[20] : -624.4584  
## ARIMA(4,1,0)(1,1,0)[20] : -625.3982  
## ARIMA(4,1,1)(0,1,0)[20] : -610.8851  
## ARIMA(5,1,0)(0,1,0)[20] : -610.9229  
##   
##   
##   
## Best model: ARIMA(0,1,0)(1,1,0)[20]

summary(fit\_arima)

## Series: jpyt   
## ARIMA(0,1,0)(1,1,0)[20]   
##   
## Coefficients:  
## sar1  
## -0.5108  
## s.e. 0.1010  
##   
## sigma^2 = 0.002374: log likelihood = 318.55  
## AIC=-633.1 AICc=-632.95 BIC=-628.37  
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE  
## Training set -0.003212004 0.04302951 0.009095485 -0.00343368 0.009722312  
## MASE ACF1  
## Training set 0.694975 0.1399099

checkresiduals(fit\_arima)



##   
## Ljung-Box test  
##   
## data: Residuals from ARIMA(0,1,0)(1,1,0)[20]  
## Q\* = 7.0392, df = 19, p-value = 0.994  
##   
## Model df: 1. Total lags used: 20

forecast(fit\_arima)

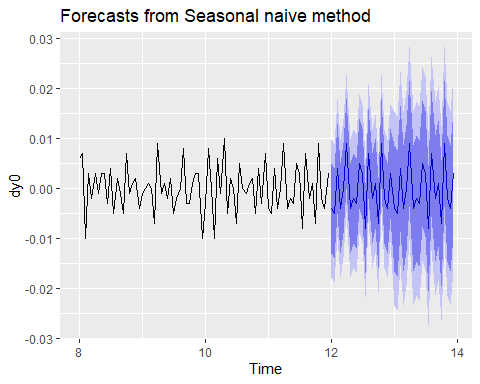
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 12.00 93.53940 93.47696 93.60184 93.44391 93.63489  
## 12.05 93.53745 93.44914 93.62575 93.40240 93.67249  
## 12.10 93.53745 93.42930 93.64559 93.37205 93.70284  
## 12.15 93.53038 93.40550 93.65526 93.33940 93.72137  
## 12.20 93.52638 93.38676 93.66600 93.31285 93.73991  
## 12.25 93.52627 93.37333 93.67922 93.29237 93.76018  
## 12.30 93.52932 93.36412 93.69451 93.27667 93.78196  
## 12.35 93.52883 93.35222 93.70543 93.25873 93.79892  
## 12.40 93.52789 93.34058 93.71521 93.24142 93.81437  
## 12.45 93.52940 93.33195 93.72685 93.22743 93.83138  
## 12.50 93.52881 93.32172 93.73589 93.21209 93.84552  
## 12.55 93.52685 93.31055 93.74314 93.19605 93.85764  
## 12.60 93.52832 93.30319 93.75344 93.18401 93.87262  
## 12.65 93.52829 93.29467 93.76192 93.17100 93.88559  
## 12.70 93.52927 93.28745 93.77110 93.15943 93.89911  
## 12.75 93.52785 93.27809 93.77760 93.14588 93.90982  
## 12.80 93.52827 93.27083 93.78572 93.13455 93.92200  
## 12.85 93.52976 93.26486 93.79467 93.12462 93.93490  
## 12.90 93.52776 93.25560 93.79993 93.11152 93.94400  
## 12.95 93.53081 93.25157 93.81004 93.10375 93.95786  
## 13.00 93.52956 93.23525 93.82387 93.07945 93.97966  
## 13.05 93.52656 93.21791 93.83521 93.05452 93.99859  
## 13.10 93.52656 93.20421 93.84891 93.03356 94.01955  
## 13.15 93.52106 93.18556 93.85655 93.00796 94.03415  
## 13.20 93.51706 93.16892 93.86520 92.98462 94.04949  
## 13.25 93.51956 93.15921 93.87990 92.96846 94.07066  
## 13.30 93.52156 93.14941 93.89371 92.95241 94.09071  
## 13.35 93.52081 93.13722 93.90440 92.93416 94.10745  
## 13.40 93.51831 93.12361 93.91300 92.91467 94.12194  
## 13.45 93.51956 93.11405 93.92506 92.89939 94.13972  
## 13.50 93.52131 93.10528 93.93733 92.88505 94.15756  
## 13.55 93.51831 93.09202 93.94460 92.86635 94.17026  
## 13.60 93.52056 93.08424 93.95687 92.85327 94.18784  
## 13.65 93.52106 93.07495 93.96717 92.83879 94.20333  
## 13.70 93.52256 93.06686 93.97826 92.82563 94.21949  
## 13.75 93.51931 93.05422 93.98440 92.80802 94.23060  
## 13.80 93.52156 93.04727 93.99585 92.79619 94.24693  
## 13.85 93.52331 93.03999 94.00663 92.78413 94.26248  
## 13.90 93.52131 93.02912 94.01349 92.76858 94.27404  
## 13.95 93.52331 93.02242 94.02420 92.75726 94.28935

Forecast using seasonal naive model

fcast=forecast(fit,h=40)  
summary(fcast)

##   
## Forecast method: Seasonal naive method  
##   
## Model Information:  
## Call: snaive(y = dy0)   
##   
## Residual sd: 0.007   
##   
## Error measures:  
## ME RMSE MAE MPE MAPE MASE ACF1  
## Training set 1.282051e-05 0.006999084 0.005551282 -Inf Inf 1 -0.5239652  
##   
## Forecasts:  
## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 12.00 -0.004 -0.0129696873 0.0049696873 -0.017717953 0.009717953  
## 12.05 -0.005 -0.0139696873 0.0039696873 -0.018717953 0.008717953  
## 12.10 0.004 -0.0049696873 0.0129696873 -0.009717953 0.017717953  
## 12.15 -0.004 -0.0129696873 0.0049696873 -0.017717953 0.009717953  
## 12.20 0.000 -0.0089696873 0.0089696873 -0.013717953 0.013717953  
## 12.25 0.009 0.0000303127 0.0179696873 -0.004717953 0.022717953  
## 12.30 -0.004 -0.0129696873 0.0049696873 -0.017717953 0.009717953  
## 12.35 -0.002 -0.0109696873 0.0069696873 -0.015717953 0.011717953  
## 12.40 -0.003 -0.0119696873 0.0059696873 -0.016717953 0.010717953  
## 12.45 0.005 -0.0039696873 0.0139696873 -0.008717953 0.018717953  
## 12.50 0.003 -0.0059696873 0.0119696873 -0.010717953 0.016717953  
## 12.55 -0.008 -0.0169696873 0.0009696873 -0.021717953 0.005717953  
## 12.60 0.007 -0.0019696873 0.0159696873 -0.006717953 0.020717953  
## 12.65 -0.002 -0.0109696873 0.0069696873 -0.015717953 0.011717953  
## 12.70 0.001 -0.0079696873 0.0099696873 -0.012717953 0.014717953  
## 12.75 -0.007 -0.0159696873 0.0019696873 -0.020717953 0.006717953  
## 12.80 0.009 0.0000303127 0.0179696873 -0.004717953 0.022717953  
## 12.85 -0.002 -0.0109696873 0.0069696873 -0.015717953 0.011717953  
## 12.90 -0.004 -0.0129696873 0.0049696873 -0.017717953 0.009717953  
## 12.95 0.003 -0.0059696873 0.0119696873 -0.010717953 0.016717953  
## 13.00 -0.004 -0.0166850534 0.0086850534 -0.023400115 0.015400115  
## 13.05 -0.005 -0.0176850534 0.0076850534 -0.024400115 0.014400115  
## 13.10 0.004 -0.0086850534 0.0166850534 -0.015400115 0.023400115  
## 13.15 -0.004 -0.0166850534 0.0086850534 -0.023400115 0.015400115  
## 13.20 0.000 -0.0126850534 0.0126850534 -0.019400115 0.019400115  
## 13.25 0.009 -0.0036850534 0.0216850534 -0.010400115 0.028400115  
## 13.30 -0.004 -0.0166850534 0.0086850534 -0.023400115 0.015400115  
## 13.35 -0.002 -0.0146850534 0.0106850534 -0.021400115 0.017400115  
## 13.40 -0.003 -0.0156850534 0.0096850534 -0.022400115 0.016400115  
## 13.45 0.005 -0.0076850534 0.0176850534 -0.014400115 0.024400115  
## 13.50 0.003 -0.0096850534 0.0156850534 -0.016400115 0.022400115  
## 13.55 -0.008 -0.0206850534 0.0046850534 -0.027400115 0.011400115  
## 13.60 0.007 -0.0056850534 0.0196850534 -0.012400115 0.026400115  
## 13.65 -0.002 -0.0146850534 0.0106850534 -0.021400115 0.017400115  
## 13.70 0.001 -0.0116850534 0.0136850534 -0.018400115 0.020400115  
## 13.75 -0.007 -0.0196850534 0.0056850534 -0.026400115 0.012400115  
## 13.80 0.009 -0.0036850534 0.0216850534 -0.010400115 0.028400115  
## 13.85 -0.002 -0.0146850534 0.0106850534 -0.021400115 0.017400115  
## 13.90 -0.004 -0.0166850534 0.0086850534 -0.023400115 0.015400115  
## 13.95 0.003 -0.0096850534 0.0156850534 -0.016400115 0.022400115

autoplot(fcast,include = 80)



Time series data from R- airpassengers.

ps=AirPassengers  
class(ps)

## [1] "ts"

boxplot(ps,notch = TRUE)

