CH5350 Course Project Fuzzy Time Series

G Amruth Raja Vardhan || MM16B003

null

Loading the required packages

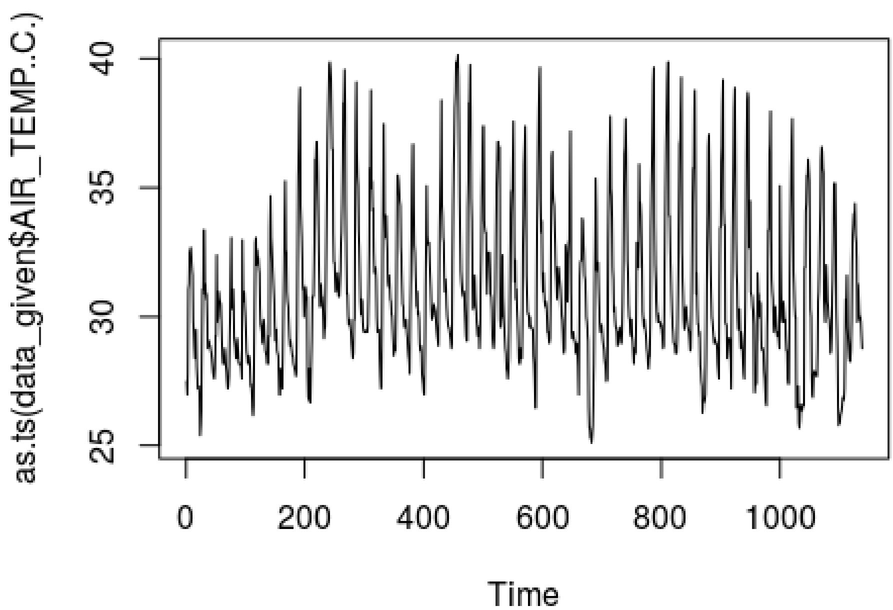
```
rm(list = ls())
library(tseries)

## Registered S3 method overwritten by 'xts':
##   method     from
##   as.zoo.xts zoo

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

library(TSA)

##
## Attaching package: 'TSA'

## The following objects are masked from 'package:stats':
##
##     acf, arima

## The following object is masked from 'package:utils':
##
##     tar

library(dplyr)

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

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

library(imputeTS)

## Registered S3 methods overwritten by 'forecast':
##   method             from
##   fitted.Arima       TSA
```

```
##   fitted.fracdiff    fracdiff
##   plot.Arima         TSA
##   residuals.fracdiff fracdiff

##
## Attaching package: 'imputeTS'

## The following object is masked from 'package:tseries':
##
##     na.remove

library(xts)

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following object is masked from 'package:imputeTS':
##
##     na.locf

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

##
## Attaching package: 'xts'

## The following objects are masked from 'package:dplyr':
##
##     first, last

library(AnalyzeTS)

## Loading required package: MASS

##
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':
##
##     select

## Loading required package: TTR

## Loading required package: urca

##
## Attaching package: 'AnalyzeTS'

## The following object is masked from 'package:base':
##
##     pmax
```

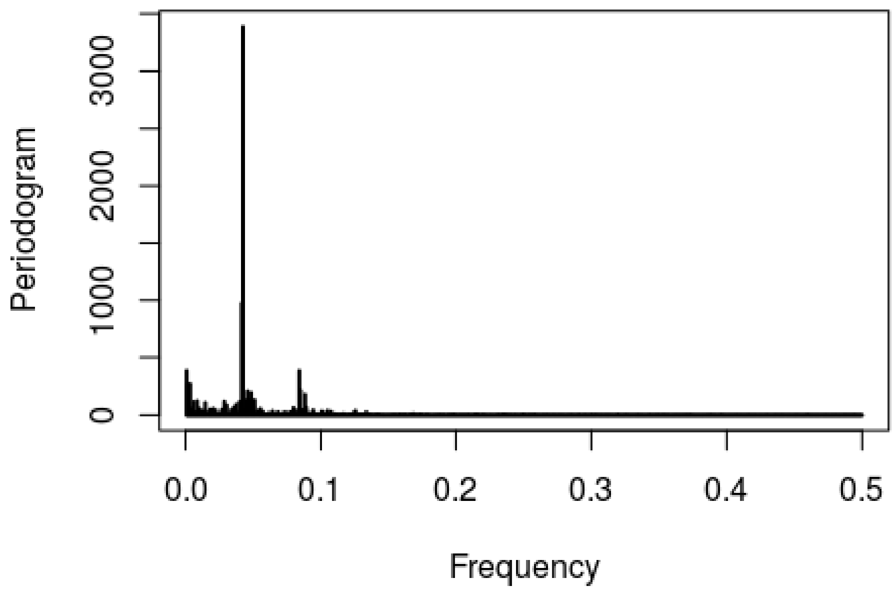
```r
library(knitr)
library(rmarkdown)
```

Loading and visualizing the given data.

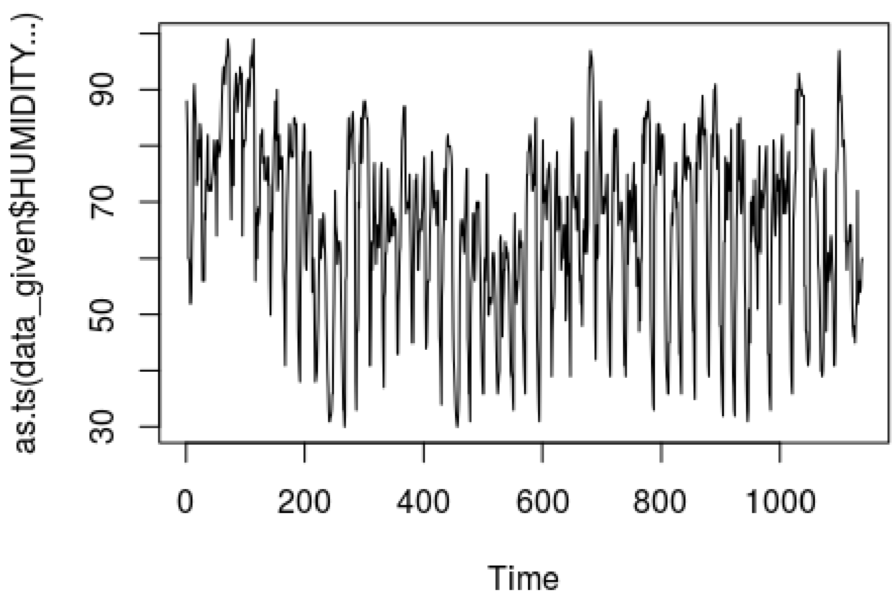
```r
data_given = read.csv("SHAR_MAY15_JULY7.csv")
columns = colnames(data_given)
data_given = data_given[-c(1:4)]
plot(as.ts(data_given$AIR_TEMP..C.))
```



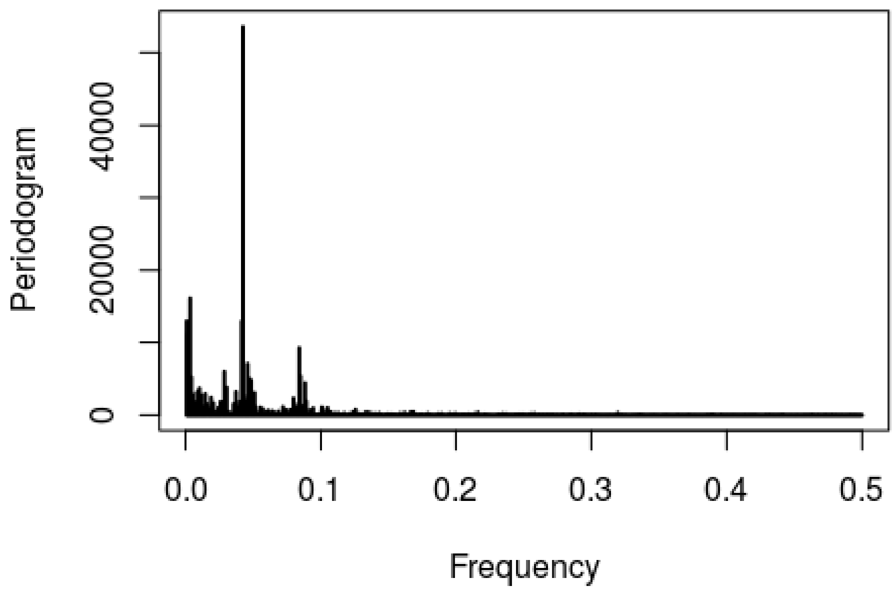

```r
periodogram(as.ts(data_given$AIR_TEMP..C.))
```




```
plot(as.ts(data_given$HUMIDITY...))
```




```
periodogram(as.ts(data_given$HUMIDITY...))
```




```
summary(data_given)

##    TIME.GMT.          DATE.GMT.      TIME.IST.         DATE.IST.
##  Min.   : 0.00   05/15/2009: 24   2:30    : 50   05/17/2009: 24
##  1st Qu.: 6.00   05/18/2009: 24   3:30    : 50   05/20/2009: 24
##  Median :12.00   05/20/2009: 24   18:30   : 49   05/21/2009: 24
##  Mean   :11.62   05/21/2009: 24   19:30   : 49   05/22/2009: 24
##  3rd Qu.:18.00   05/22/2009: 24   21:30   : 49   05/23/2009: 24
##  Max.   :23.00   05/23/2009: 24   22:30   : 49   05/24/2009: 24
##                  (Other)   :994   (Other):842   (Other)   :994
##   AIR_TEMP..C.   WIND_SPEED.m.s. WIND_DIRECTION.deg. ATMO_PRESSURE.hpa.
##  Min.   :25.10   Min.   :0.050   Min.   : 15.88      Min.   : 996.2
##  1st Qu.:28.96   1st Qu.:0.830   1st Qu.:166.91      1st Qu.:1002.1
##  Median :30.28   Median :1.910   Median :218.72      Median :1003.5
##  Mean   :31.03   Mean   :2.018   Mean   :230.30      Mean   :1003.4
##  3rd Qu.:32.67   3rd Qu.:2.980   3rd Qu.:291.54      3rd Qu.:1004.7
##  Max.   :40.15   Max.   :7.580   Max.   :358.99      Max.   :1010.6
##
##   HUMIDITY...     RAIN_FALL.mm.   SUN_SHINE.hh.mm. BATTERY_VOLTAGE.V.
##  Min.   :29.96   Min.   :894.0   0:0     : 62     Min.   :12.19
##  1st Qu.:55.96   1st Qu.:894.0   9:17    : 30     1st Qu.:12.49
##  Median :67.99   Median :894.0   10:7    : 27     Median :12.68
##  Mean   :66.10   Mean   :897.1   7:50    : 25     Mean   :12.83
##  3rd Qu.:77.96   3rd Qu.:900.0   7:51    : 16     3rd Qu.:13.07
##  Max.   :98.97   Max.   :914.0   8:41    : 16     Max.   :14.20
##                                  (Other):962
```

The above periodograms indicate seasonality in the data.

Adding a timestamp column to the dataframe. Identifying the missing values and inserting NA in their place by using full join

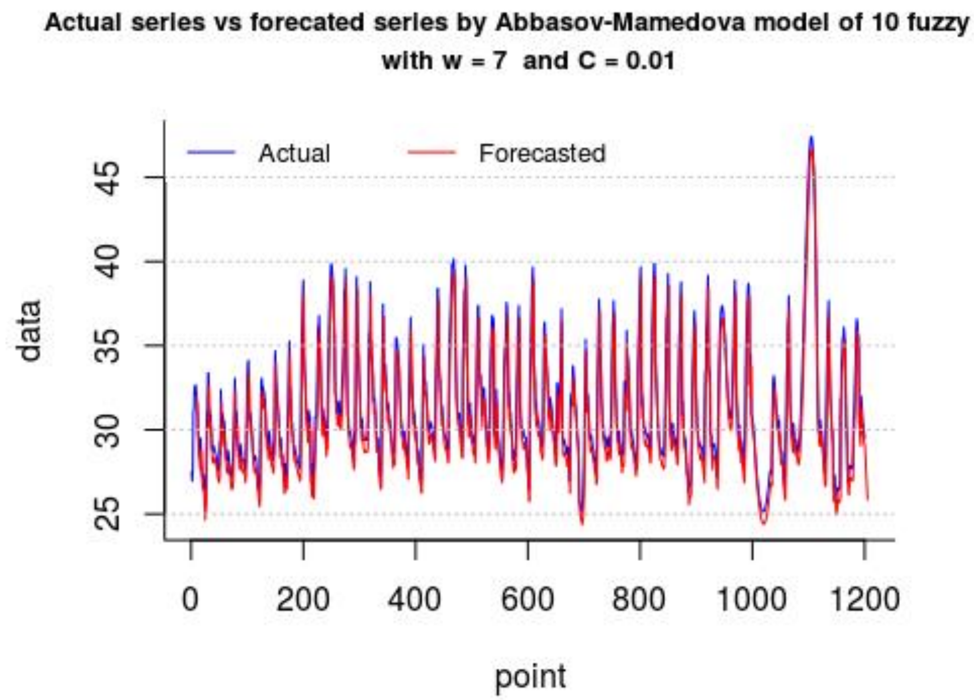
```
tstamp = paste(data_given$DATE.IST., data_given$TIME.IST., sep = " ")
tstamp = strptime(tstamp, "%m/%d/%Y %H:%M", tz = "GMT")
data_given["tstamp"] = as.POSIXct(tstamp)
begin_time = as.POSIXct(data_given$tstamp[1])
end_time = as.POSIXct(data_given$tstamp[nrow(data_given)])
# Performing full join of given data with time stamp to insert NA values
tstamp_full = seq.POSIXt(begin_time, end_time, by = "hour")
time_frame = data.frame(tstamp = tstamp_full)
data_na = full_join(time_frame, data_given) # has NA for missing values

## Joining, by = "tstamp"
```

Since the data is seasonal, it would be appropriate to use spline interpolation to impute the missing values. Accordingly, we only impute the columns corresponding to temperature and RH and then create separate time series objects for the same. Then dividing both the series into training and cross validation sets.

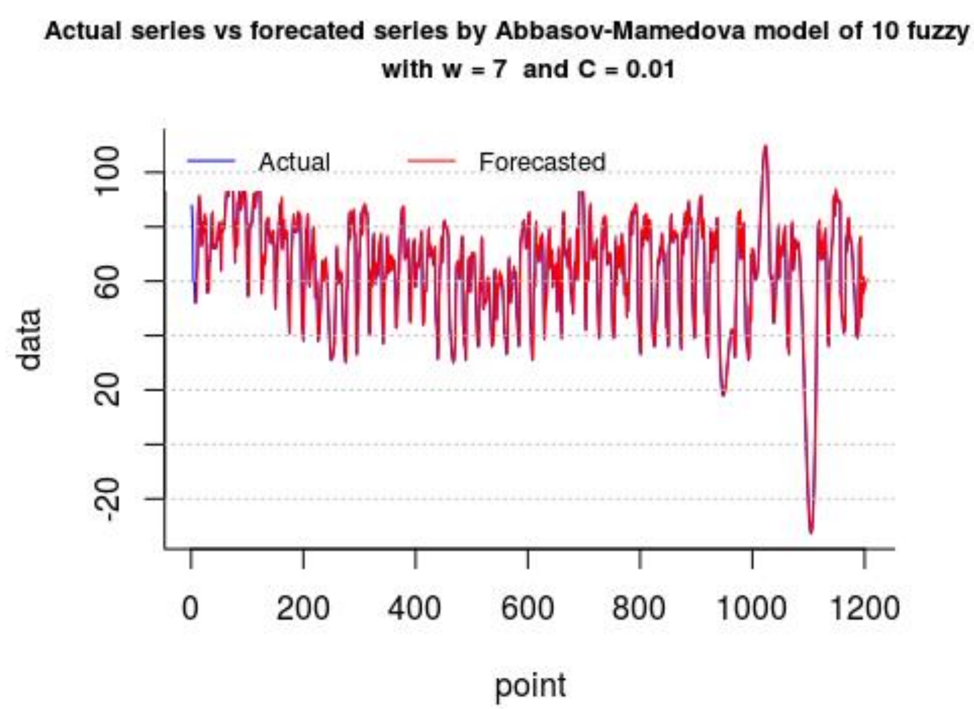
```
temper_ts = as.ts(data_na$AIR_TEMP..C.)
RH_ts = as.ts(data_na$HUMIDITY...)
# Interpolation
temper_ts = na_interpolation(temper_ts, option = "spline")
RH_ts = na_interpolation(RH_ts , option = "spline")
# Training and cross validation sets
temper_train = as.ts(temper_ts[1:1200])
temper_cval = as.ts(temper_ts[1201:1296])
RH_train = as.ts(RH_ts[1:1200])
RH_cval = as.ts(RH_ts[1201:1296])
```

Using AnalyzeTS package to build a fuzzy time series model(Abbasov-Mamedova model) for temperature and RH (Model M1) separately. The hyperparameter of the model(number of fuzzy sets n) are determined by trial and error. The models are callibrated using the RMSE with respect to the cross-validation sets.

```
temper_mod1 = na_remove(fuzzy.ts2(temper_train, n = 10, C = 0.01, trace = TRU
E, plot = TRUE,grid = TRUE))
```



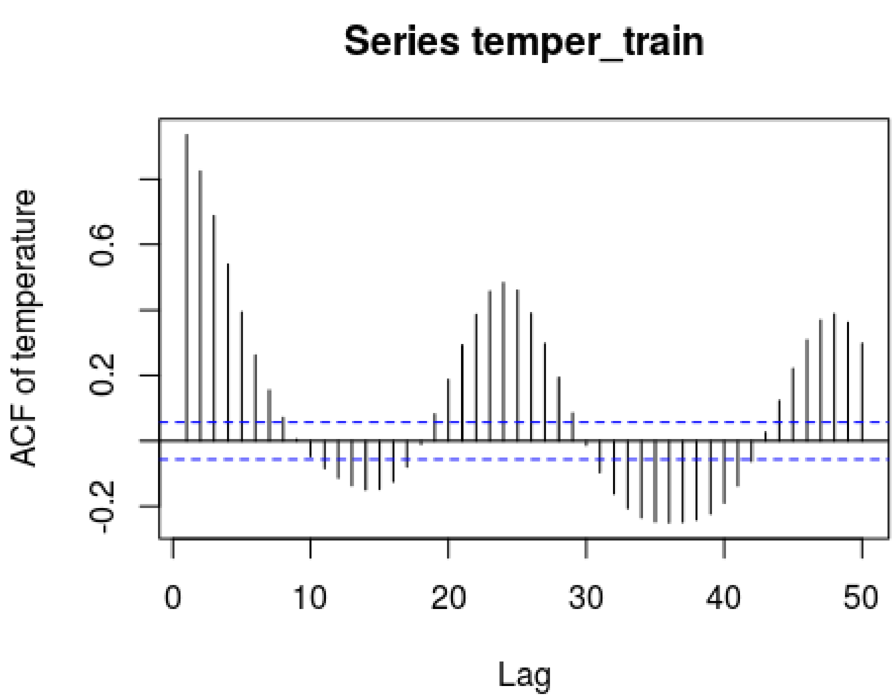

```
RH_mod1 = na_remove(fuzzy.ts2(RH_train, n = 10, C = 0.01, trace = TRUE, plot
= TRUE, grid = TRUE))
```



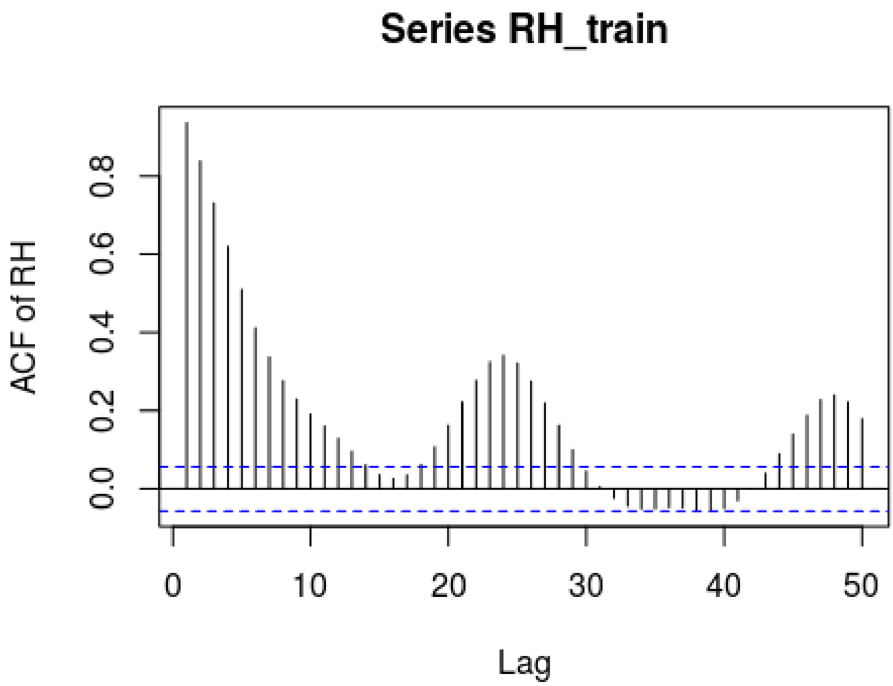
```
# Upper and lower limits of the fuzzy sets
temper_mod1$table1$dow

##  [1] -6.790 -5.578 -4.366 -3.154 -1.942 -0.730  0.482  1.694  2.906  4.118

temper_mod1$table1$up

##  [1] -5.578 -4.366 -3.154 -1.942 -0.730  0.482  1.694  2.906  4.118  5.330

# Accuracy of the models
temper_mod1$accuracy

##                      ME   MAE  MPE  MAPE   MSE RMSE        U
## Abbasov.Mamedova 0.725 1.068 2.27 3.333 2.132 1.46 1.151453

RH_mod1$accuracy

##                       ME   MAE    MPE  MAPE    MSE  RMSE         U
## Abbasov.Mamedova -0.497 4.712 -1.546 7.434 47.441 6.887 0.9880501
```

Developing SARIMA model with temperature as exogeneous input to RH (Model M2). From the periodograms (see the first chunk of code), both the series are seasonal with period 24. The CCF is also observed to be periodic. The order of the SARIMA model is chosen using the criteria of lowest AIC.

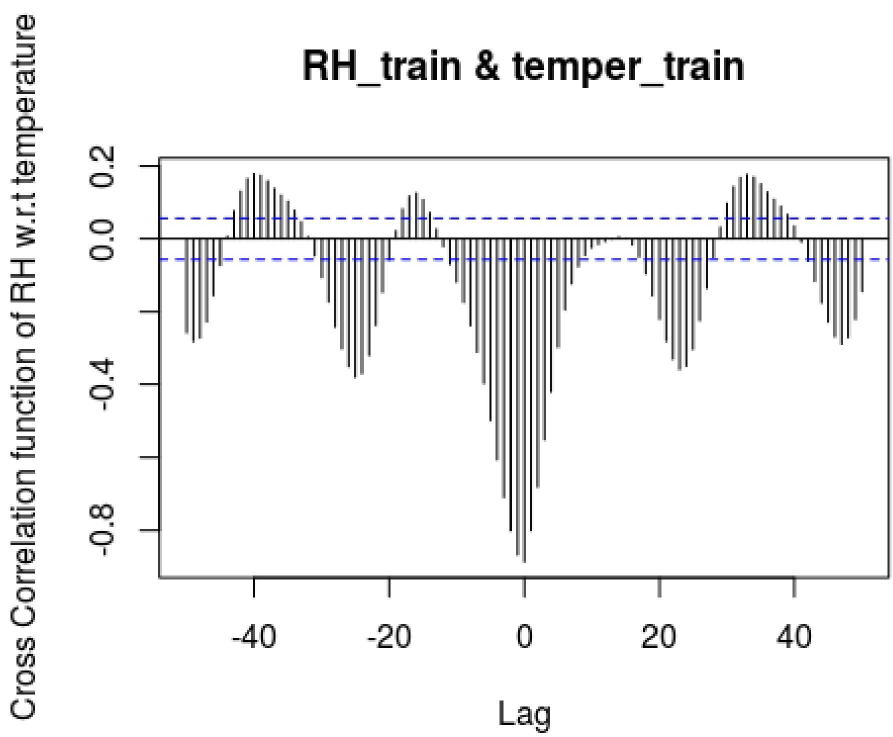
```
acf(temper_train, lag.max = 50, ylab = "ACF of temperature")
```




```
acf(RH_train, lag.max = 50, ylab = "ACF of RH")
```




```
ccf(RH_train, temper_train, lag.max = 50, ylab = "Cross Correlation function
of RH w.r.t temperature")
```



```
sarima_mod2 = arima(RH_train, order = c(1,0,2), seasonal = list(order = c(1,0
,1), period = 24), xreg = temper_train)
```

The best model(having the lowest AIC) is found to be SARIMA(1,0,2)x(1,0,1).

Choosing the best between Model M1 and Model M2: The MSE error of Model M1 (Fuzzy Time Series) for Relative Humidity is lower than the MSE error of Model M2 (SARIMA). Hence, Model M1(Fuzzy Time Series) is found to be better since it has a lower MSE.

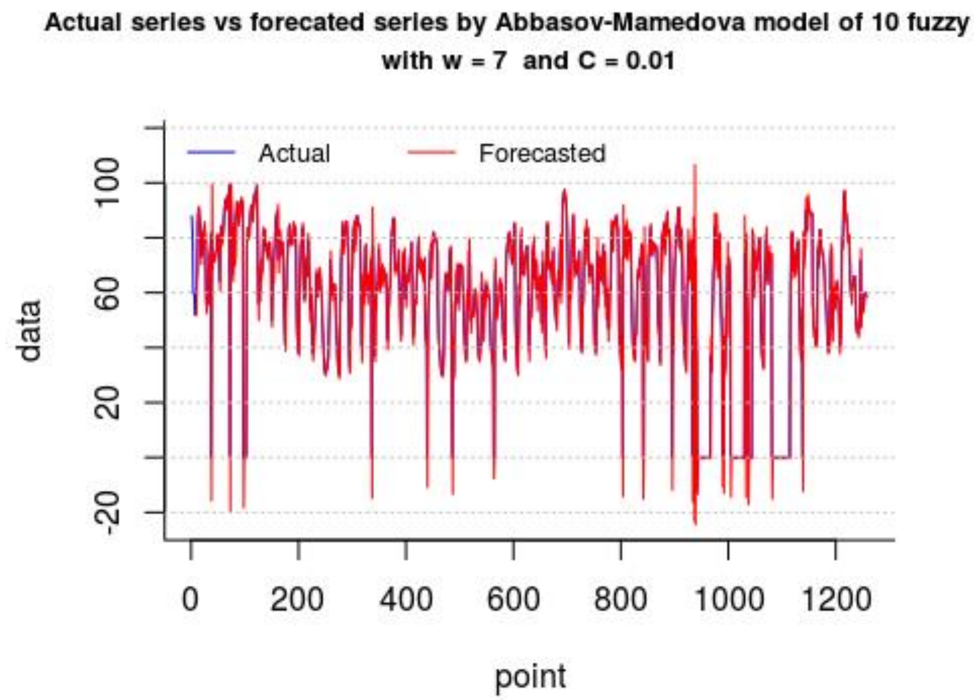
```
mse_fuzzy = var(RH_mod1$table4$diff.interpolate)
mse_sarima = sarima_mod2$sigma2
```

Replacing the imputed values using Model M1(Fuzzy Time Series).

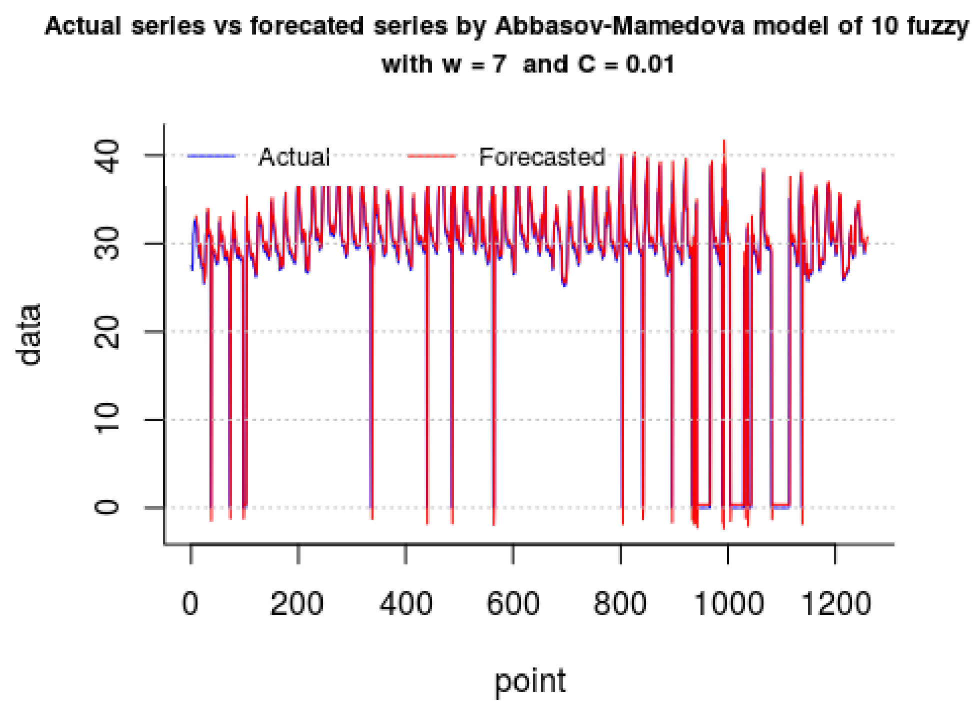
```
RH_pred = RH_mod1$table4$diff.interpolate
temper_pred = temper_mod1$table4$diff.interpolate
for(i in seq(9,1200,1))
{
  if(is.na(data_na$"HUMIDITY..."[i]))
    data_na$"HUMIDITY..."[i] = 0
  if(is.na(data_na$"AIR_TEMP..C."[i]))
    data_na$"AIR_TEMP..C."[i] = 0
}
```

Rebuilding a new fuzzy time series model where the NA data points are replaced by the predictions obtained from the first fuzzy time series (Hyperparameters of the model are same as the earlier one)

```
temper_new = as.ts(data_na$"HUMIDITY...")
RH_new = as.ts(data_na$"AIR_TEMP..C.")
temper_fuzzy2 = fuzzy.ts2(as.ts(na_remove(temper_new)), n = 10, C = 0.01, tra
ce = TRUE, plot = TRUE,grid = TRUE)
```




```
RH_fuzzy2 = fuzzy.ts2(as.ts(na_remove(RH_new)), n = 10, C = 0.01, trace = TRU
E, plot = TRUE, grid = TRUE)
```



```r
# Accuracy of the models
temper_fuzzy2$accuracy

##         ME   MAE MPE MAPE     MSE   RMSE        U
## [1,] 0.306 6.647 NaN  Inf 214.661 14.651 1.082145

RH_fuzzy2$accuracy

##          ME   MAE MPE MAPE    MSE  RMSE        U
## [1,] -0.382 2.012 NaN  Inf 37.894 6.155 1.027673
```

We can see that the model accuracy has improved as compared to the previous model. This is because the overlapping partitions in the fuzzy time series model predict the values better then simple spline interpolation.