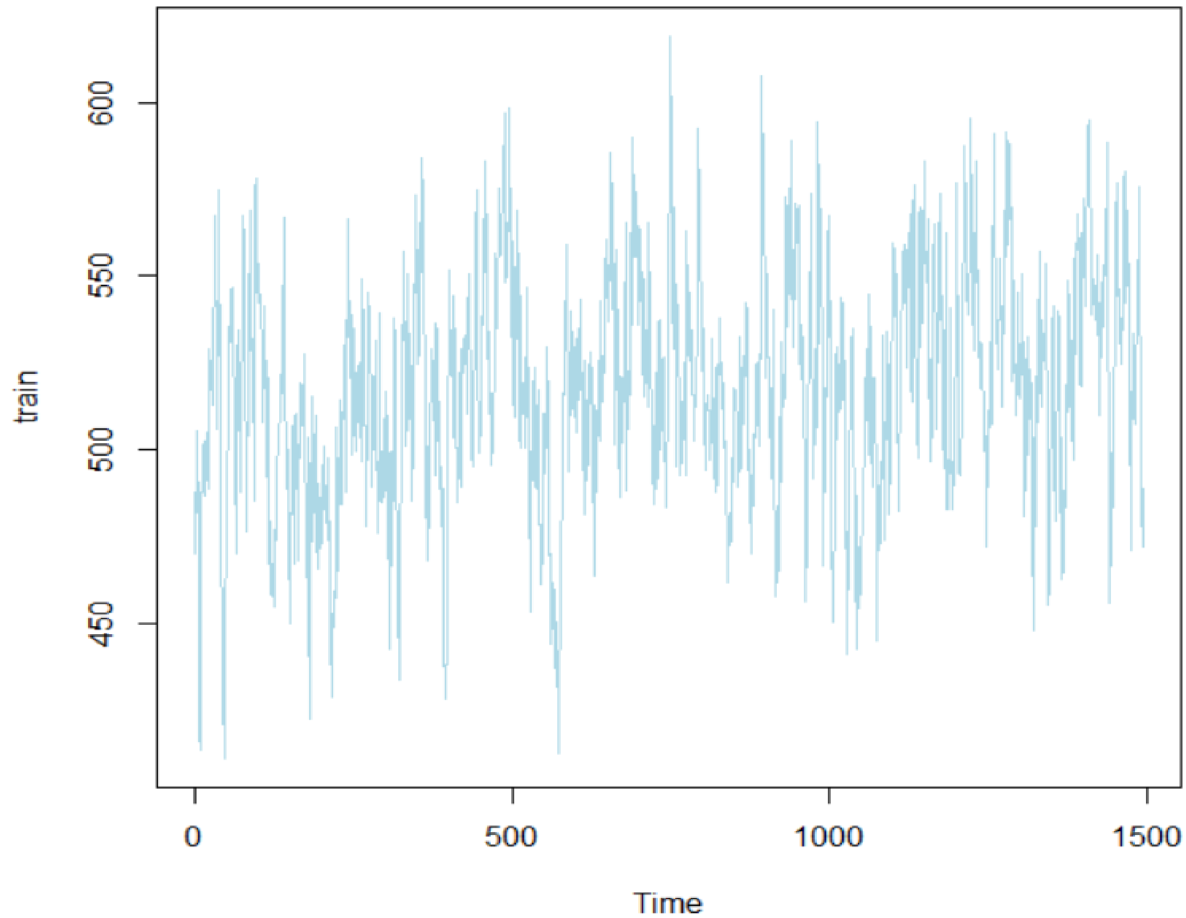
Project – 3

rpatel17

Ans. 1)

Training set – after removal of tail points in the beginning



1.1, 1.2, 1.3)

```
root_mean_square_error <- c(1:25)


for(i in 1:25){
  sma <- SimpleMovingAverage(train, i)
  root_mean_square_error[i] <- rmse(train[i+1:1494],sma[i+1:1494])
}
```

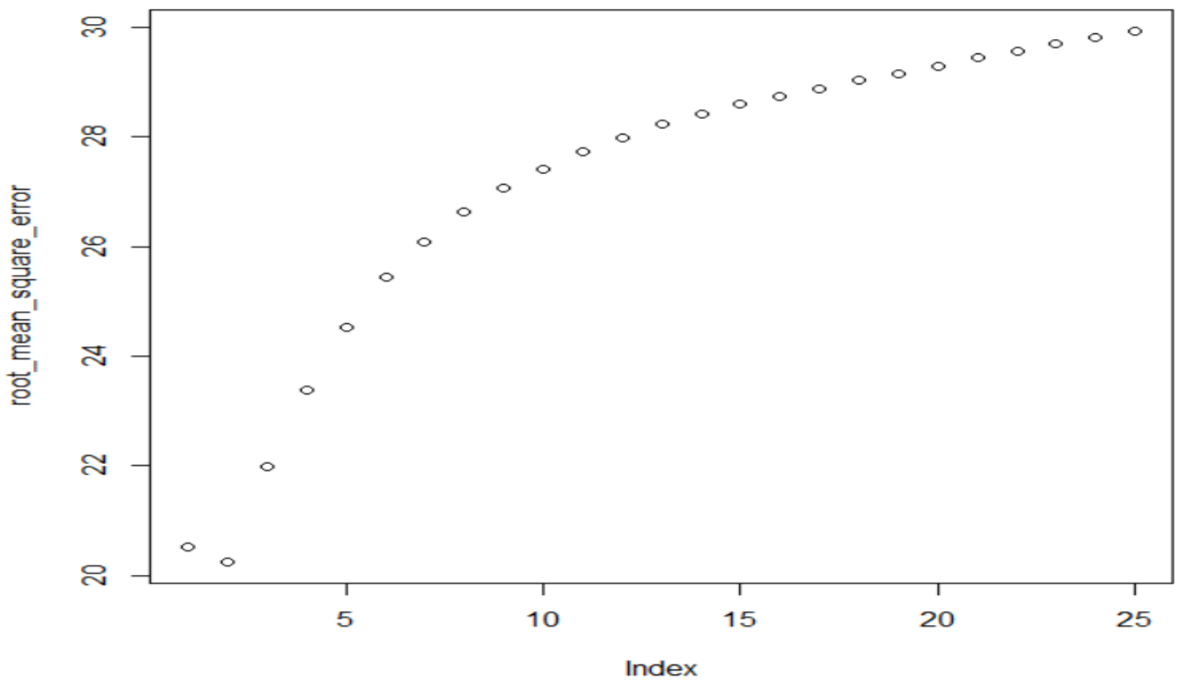
```
SimpleMovingAverage <- function(train,m) {
avg=0
ma_fit <- rep(times=m,0)

for(j in (m+1):length(train)){
        for(i in 1:m){
                avg = avg + train[j-i]
          }
   ma_fit = append(ma_fit, avg/m)
   avg = 0
  }
  return(ma_fit)
}
```

RMSE for m = 1,2,3 :

```
> sma_2 <- SimpleMovingAverage(train,2)
>
> root_mean_squa_err_2 <- rmse(train[3:1494],sma_2[3:1494])
>
> root_mean_squa_err_2
[1] 20.25715
> sma_3 <- SimpleMovingAverage(train,3)
>
> root_mean_squa_err_3 <- rmse(train[4:1494],sma_3[4:1494])
>
> root_mean_squa_err_3
[1] 21.98065
> sma_3 <- SimpleMovingAverage(train,3)
>
> root_mean_squa_err_3 <- rmse(train[4:1494],sma_3[4:1494])
>
> root_mean_squa_err_3
[1] 21.98065
> sma_1 <- SimpleMovingAverage(train,1)
>
> root_mean_squa_err_1 <- rmse(train[2:1494],sma_1[2:1494])
>
> root_mean_squa_err_1
[1] 20.51727
```

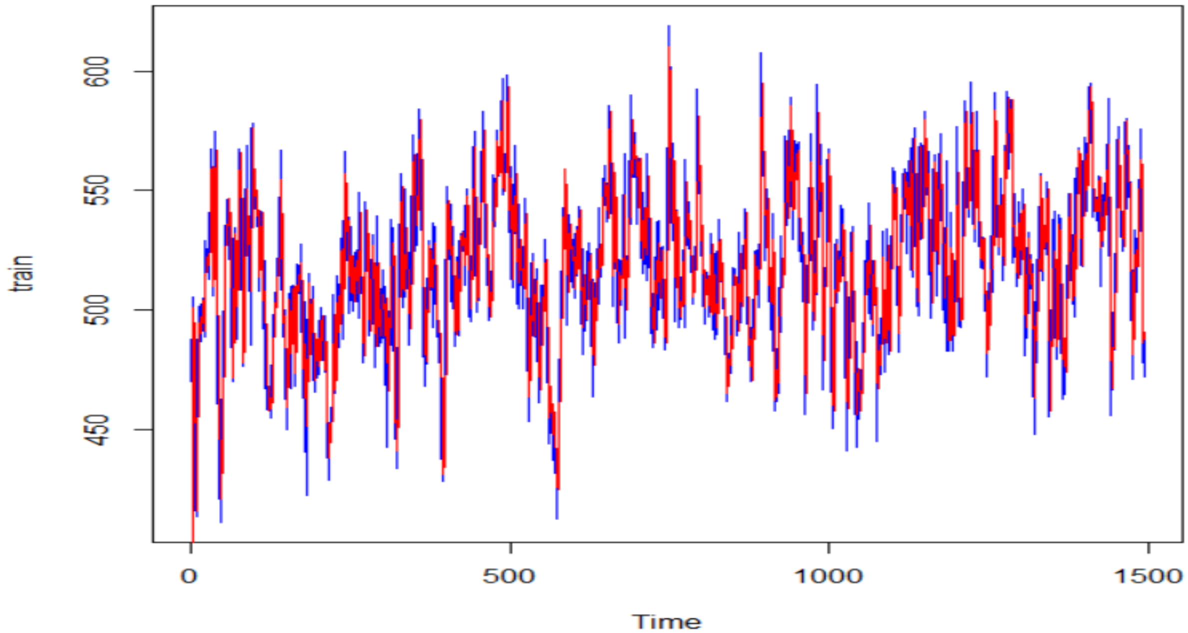
1.4)

RMSE vs m:




```
plot.ts(train, col="blue")
lines(sma_2, col="red")
```

Original values in blue and predicted value in red :



Ans. 2)

2.1, 2.2, 2.3

```
rmse_e <- c(1:11)

n<-1494

predicted_val <- c(1,1494)

k<-1

s <- seq(0, 1, by = 0.1)


for(i in s) {

        for (j in 2:1494) {

        predicted_val [j] <- (i* train[j-1]) + ((1-i)* predicted_val [j-1])

        }

error<-(train[2:1494] - predicted_val [2:1494])

sum_square_error<-sum(error^2)
```

rmse_e[k]<-sqrt(sum_square_error/n)

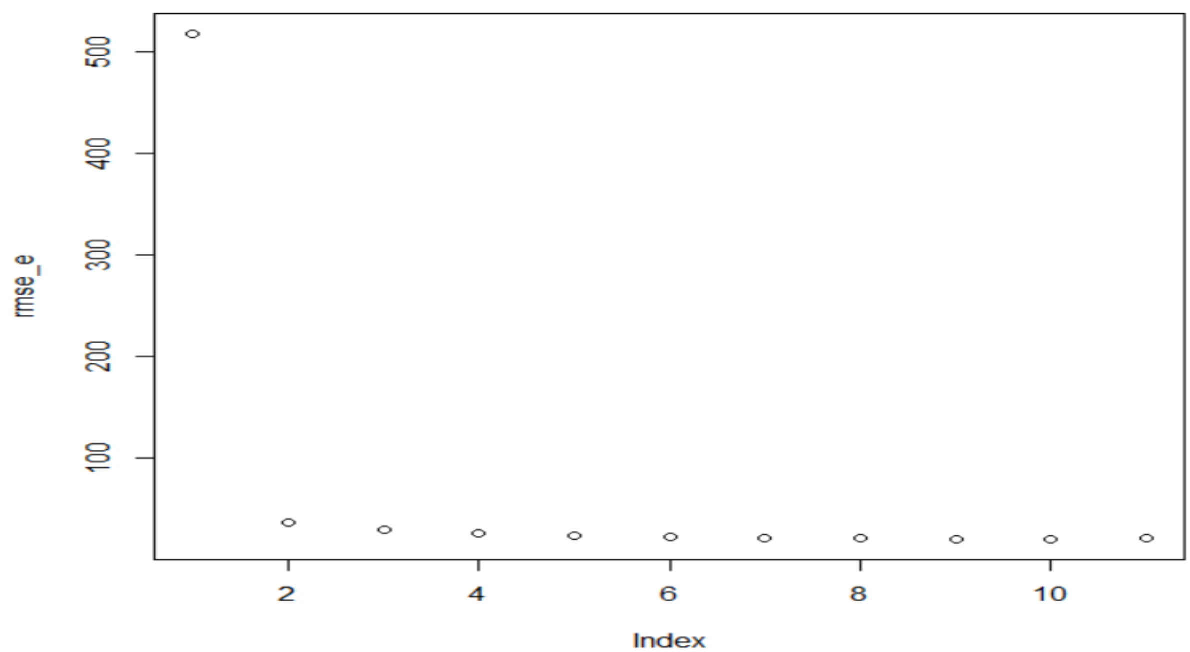
k<-k+1

}

rmse_e[1:11] #lowest value for alpha equal to 0.8

plot(rmse_e)

RMSE vs alpha : 0 to 1 (each at an increment of 0.1)



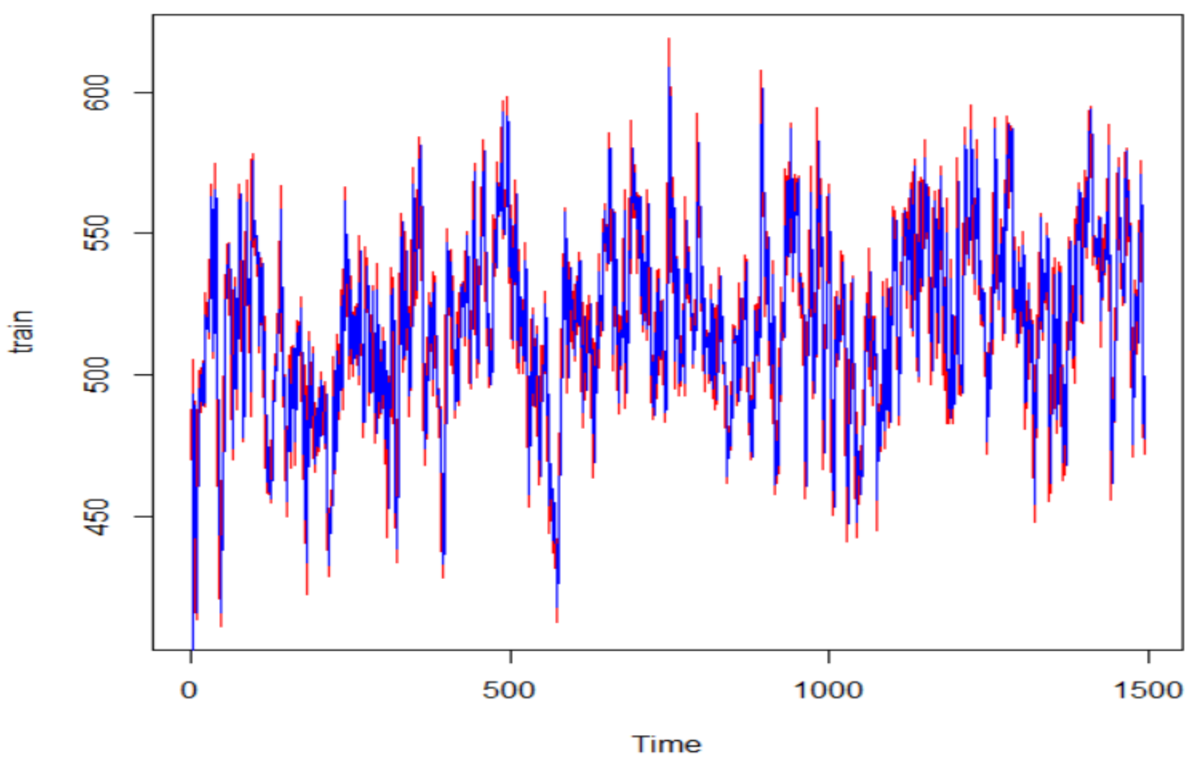

```
rmse_e[1:11] #lowest value for alpha equal to 0.8
[1] 517.74025  36.72840  28.98493  25.46766  23.31569  21.87753  20.92214
[8]  20.34631  20.09861  20.15494  20.51041
```

n<-1494

predicted_val <- c(1,1494)

```
i<-0.8

for (j in 2:1494) {
  predicted_val [j] <- (i* train[j-1]) + ((1-i)* predicted_val [j-1])
}

plot.ts(train, col="red")
lines(predicted_val, col="blue")
```

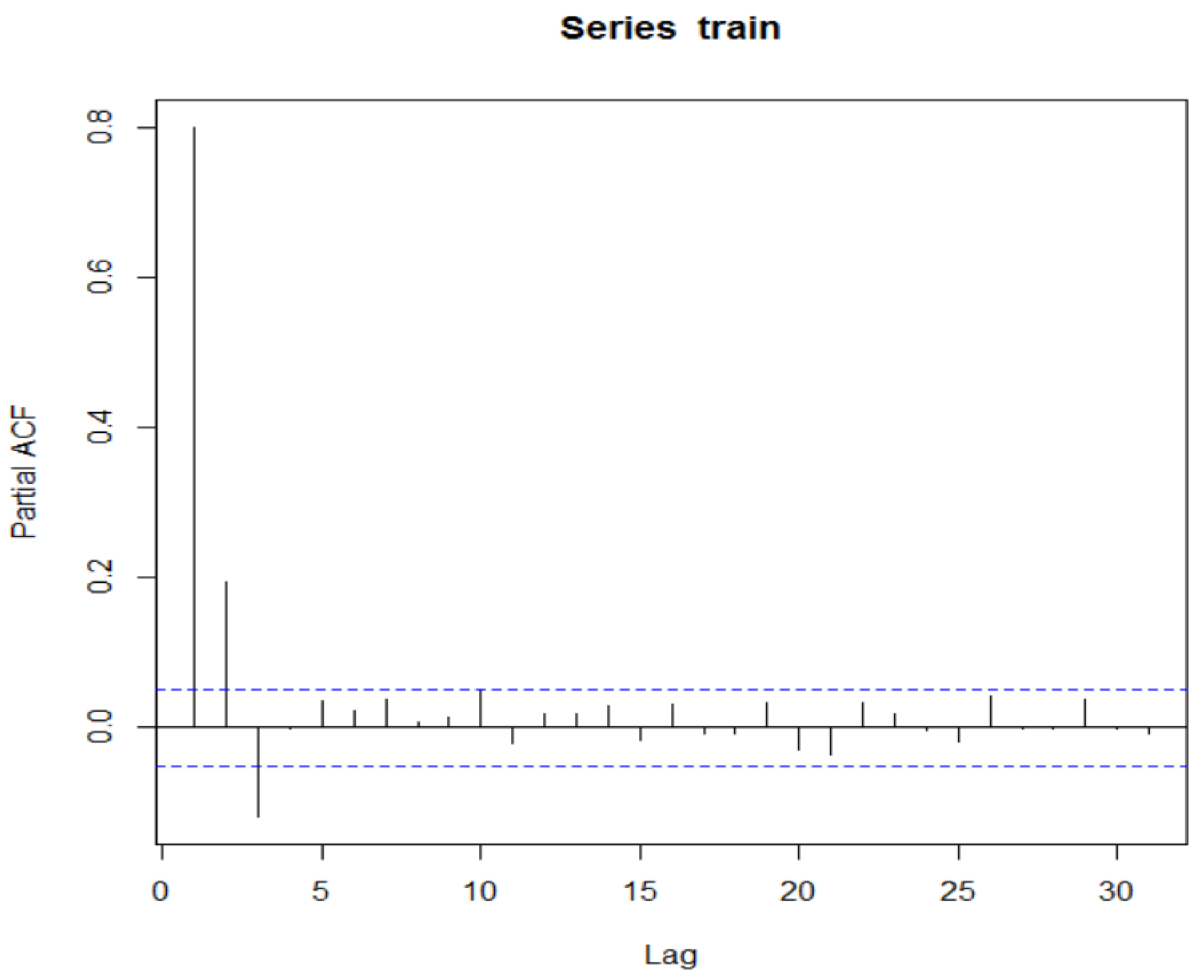
Predicted lines – blue and training data in red



Ans 3.)

3.2)

pacf(train)



3.1)

arima_fit <- auto.arima(train,d=0,max.q=0) #to check the p value

arima_fit <- auto.arima(train,d=0,max.q=0,max.p=3) #p-value of 3 is obtained since the pacf has 3 values beyond the blue lines

3.3)

arima_fit$coef

```
arima_fit$coef
      ar1         ar2         ar3   intercept 
0.6679858   0.2706990  -0.1192319 517.6127671 
```

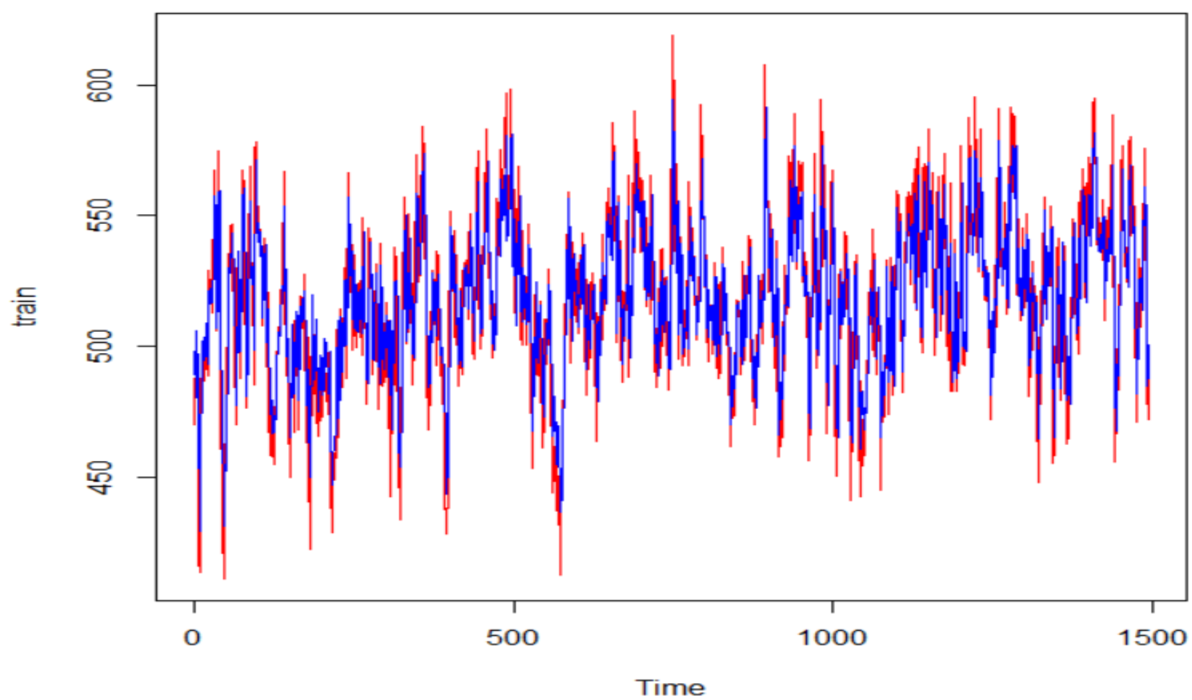
rmse(train,fitted(arima_fit))

```
> rmse(train,fitted(arima_fit))
[1] 18.96181
```

plot.ts(train, col="red")

lines(fitted(arima_fit), col="blue")

Fitted in blue and original training set in red:



Ans 4.)

```
> sma_test <- SimpleMovingAverage(test, 2)
> root_mean_sq_er <- rmse(test[3:491],sma_test[3:491])
>
> root_mean_sq_er
[1] 20.73625
>
> ##### 20.73625
>
> n<-491
>
> predicted_val <- c(1,491)
>
> i<-0.8
>
> for (j in 2:491) {
+   predicted_val[j] <- (i* test[j-1]) + ((1-i)* predicted_val[j-1])
+ }
> error<-(test[2:491] - predicted_val[2:491])
> sum_square_error<-sum(error^2)
> rmse_e<-sqrt(sum_square_error/n)
>
> rmse_e
[1] 20.66266
>
> ##### 20.66266
>
> arima_fit <- auto.arima(test,d=0,max.q=0,max.p=3)
>
> rmse(test,fitted(arima_fit))
[1] 19.13613
```

So, we have tried fit test data with all the three models and we find that AR(3) is the best model of the three models that we have tried since it has the minimum RMSE of 19.13613 amongst the three models. We used MA with m = 2, exponential smoothing with alpha value 0.8 and AR(3) for fitting our test data.