FOSSEE Summer Fellowship Report

on

FLOSS - R

submitted by

E B Benson

Cochin University of Science and Technology, Kerala

under the guidance of

Prof. Radhendushka Srivastava
Mathematics Department,
IIT Bombay

Prof. Kannan M. Moudgalya
Chemical Engineering Department,
IIT Bombay

and supervision of

Debatosh Chakraborty
R Team, FOSSEE,
IIT Bombay

July, 2024

Acknowledgment

I would like to express my sincere gratitude to Prof. Kannan M. Moudgalya from the Department of Chemical Engineering, IIT Bombay, for establishing the FOSSEE Fellowship program and providing students with this valuable opportunity. I extend my heartfelt thanks to my mentor, Prof. Radhendushka Srivastava from the Department of Mathematics, IIT Bombay, for his support, guidance, and patience throughout my research project. His expertise was invaluable in helping me understand a wide range of concepts. Additionally, I am grateful to Mr. Debatosh Chakraborty from the R FLOSS team for his insightful input and consistent guidance during the fellowship. I am truly thankful for the opportunity to learn from these esteemed individuals.

Contents

Chapter 1

Introduction

This report contains all the contributions made by me during the FOSSEE Summer Fellowship 2024 from 15th May 2024 to 15th July 2024. I did the fellowship under the guidance of the R team under the FOSSEE Project. My contributions included the R TextBook Companion Project, a Case Study on INR-USD Exchange Rate Data, and the creation of study material on Linear Time Series Models.
The FOSSEE (Free/Libre and Open Source Software for Education) project promotes the use of FLOSS tools to improve the quality of education in our country. The project aims to reduce dependency on proprietary software in educational institutions. The FOSSEE project is part of the National Mission on Education through Information and Communication Technology (ICT), Ministry of Education (MoE), Government of India.

Chapter 2

Contribution to the TextBook Companion (TBC) project

As part of the selection process for the FOSSEE Summer Fellowship, applicants are required to choose a standard textbook in areas such as Probability, Statistics, or Algebra that includes at least 80 solved examples, and to submit a TBC proposal for the R TBC project. My proposal was approved, and during the fellowship, I contributed to the R TBC project by developing a companion resource in R for the textbook listed below:

Table 2.1: Details of the textbook selected for R TBC contribution.

Textbook Name	Author	Edition
Probability And Statistics	Morris H. Degroot, Mark J. Schervish	4th

I have coded 349 solved problems of the book. My submitted TBC is available for public use on the R TBC Completed Books webpage.

Chapter 3

Case Study on Foreign Exchange Rate (INR-USD) dataset

3.1 Data Collection

The INR-USD Exchange Rate dataset consisting of the daily exchange rate of USD-INR is the dataset used for the analysis. The data are noon buying rates in New York. The data starts from January 3, 2000, to June 14, 2024. The data is recorded daily at 4 PM DST and is provided with a 7-day frequency by Federal Reserve Bank of New York.

3.2 Data Sources

The data is sourced with the following descriptions:

- **Name:** H.10 Weekly Release [1]
- **Provider:** Federal Reserve Bank of New York
- **Currency:** INR-USD exchange rate
- **Frequency:** Daily
- **Time Period:** January 2000 to June 2024

The dataset consisted of USD exchange rates with various countries,but we focused on only INR-USD exchange rate.

3.3 Data Description

The dataset is stored in the CSV format and has the dimension of 6381 rows and 2 columns. The description of headers/column names of the constructed dataset is given in the following table:

Table 3.1: Description of each header of the constructed dataset.

Attributes	Description	Data Type
Unique.Identifier..	The date of recorded exchange rate.	character
H10.H10.RXI_N.B.IN	The exchange rate of the day at noon.	character

A random subset of size 5 is selected from the constructed dataset and shown below for clarity.

Table 3.2: A random subset of the dataset.

Unique.Identifier..	H10.H10.RXI_N.B.IN
2000-01-03	43.5500
2000-02-29	43.6500
2003-12-29	45.6600
2004-02-10	45.2200
2004-02-17	45.2800

3.4 Data Preprocessing

Let us import the data and read the head of the data along with summary of the data to see, how the data is distributed.

```
df <- read.csv("FRB_H10.csv")
head(df)
summary(df)
```

Description: df [6 × 2]

	Unique.Identifier.. <chr>	H10.H10.RXI_N.B.IN <chr>
1	Time Period	RXI_N.B.IN
2	2000-01-03	43.5500
3	2000-01-04	43.5500
4	2000-01-05	43.5500
5	2000-01-06	43.5500
6	2000-01-07	43.5500

6 rows

Figure 3.1: Exchange Rates of INR against USD

```
Unique.Identifier.. H10.H10.RXI_N.B.IN
Length:6381         Length:6381
Class :character    Class :character
Mode  :character    Mode  :character
```

Figure 3.2: Summary of unprocessed data

As we can see there is an extra row stating the contents of the row, then after only does the original data starts. So we shall be removing the first row. And in the summary we can see that the both the data values are in characters datatype and not in their respective datatypes, so we shall also make them into the date and numeric format. We also convert the column names to DATE and RATE for easiness of calling them. These are done using the following commands along with a summary of the data after preprocessing using the **skim(x)** function in the skimr package of R.

```r
df <- df[-1,]
rownames(df) <- NULL
colnames(df) <- c("DATE", "RATE")
df$DATE <- as.Date(df$DATE)
df$RATE <- as.numeric(df$RATE)
skim(df)
```

```
── Data Summary ────────────────────────
                           Values
Name                       df
Number of rows             6380
Number of columns          2
_____________________
Column type frequency:
  Date                     1
  numeric                  1
_____________________
Group variables            None

── Variable type: Date ─────────────────────────────────────────────────────────
  skim_variable n_missing complete_rate min        max        median     n_unique
1 DATE                  0             1 2000-01-03 2024-06-14 2012-03-24     6380

── Variable type: numeric ──────────────────────────────────────────────────────
  skim_variable n_missing complete_rate mean   sd   p0  p25  p50  p75 p100 hist
1 RATE                248         0.961 57.3 13.2 38.5 45.8 51.9 68.1 83.6 ▇▁▁▁▁
```

Figure 3.3: Summary after cleaning

So we see that after cleaning the dataset, there are almost 248 missing values in it. We handle the missing values by applying linear interpolation using the code. And then we plot the data to see how the data looks like.

```r
df$RATE <- na.approx(df$RATE, na.rm = FALSE)
ggplot(df, aes(x = DATE, y = RATE)) +
  geom_line() +
  labs(x = "Date",
       y = "Rate") +
  theme_minimal()
```

Figure 3.4: Plot of data

From the plot we can see that there is an increasing trend in the graph providing an insight into the overall direction of the data. There is also considerable short-term fluctuations throughout the period, indicating volatility.

3.5 Data Analysis

3.5.1 ACF and PACF Plots

The Autocorrelation function[2] measures the linear relationship between lagged values of a time series. The equation is as follows:

$$r_k = \frac{\sum_{t=k+1}^{T}(x_t - \bar{x})(x_{t-k} - \bar{x})}{\sum_{t=1}^{T}(x_t - \bar{x})^2} = \frac{\text{Cov}(x_t, x_{t-k})}{\text{Var}(x_t)} \tag{3.1}$$

where:

- T is the length of the time series,
- $\bar{x}$ is the mean of the series,
- k is the lag.

The partial autocorrelation function (PACF) measures the correlation between time series observations separated by k time units, y_t and y_{t-k}, after removing the effects of all shorter lag correlations $y_{t-1}, y_{t-2}, \ldots, y_{t-(k-1)}$. The PACF plot is a graphical representation of the correlation of a time series with itself at different lags, after removing the effects of the previous lags.

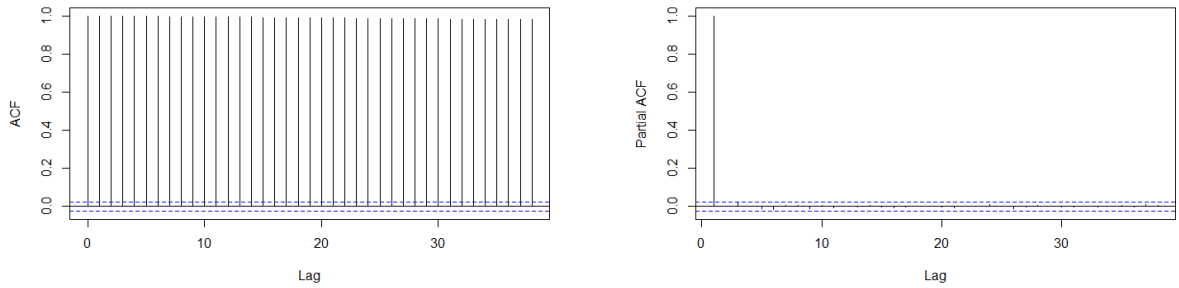

Figure 3.5: ACF and PACF plots

The ACF and PACF plots of the data is given above. As we can see the ACF plot has significant auto-correlation at many lags that does not decay quickly which are very high, that is they are close to 1. There is also a slow decay in the ACF plot, which suggests a that there is trend component. The lack of cyclical pattern in ACF states that there is no seasonality in the data. The PACF plot shows a significant spike at lag 1, and then drops off dramatically. This is a characteristic of an AR(1) process.

3.5.2 Linear Model

Let us fit a linear model to the data. A linear model is used to describe the relationship between a dependent variable and one or more independent variables by fitting a linear equation to the data.

```
lm_model <- lm(RATE ~ DATE, data = df)
df$fitted <- predict(lm_model)
ggplot(df, aes(x = DATE, y = RATE)) +
  geom_line() +
  geom_line(aes(y = fitted), color = "red") +
  labs(x = "Time", y = "RATE")
```

Figure 3.6: Linear model

In the figure the red line is the fitted values of the linear model, and the black one is the actual data. We can see from the figure itself, that the linear model is not fitting the data very well.Even though the linear model could capture the overall trend of the graph.

Now let us look at the residuals of the model [7] . We can check the residuals of the model by using the **residuals(model)** function in R. This function when we pass the model as the parameter, will provide us the residuals or error of the model. This is calculated using the difference between the actual value and fitted values.

```
df$residuals <- residuals(lm_model)
ggplot(df, aes(x = DATE, y = residuals)) +
  geom_line() +
  labs(x = "Time", y = "Residuals") +
  theme_minimal()
```

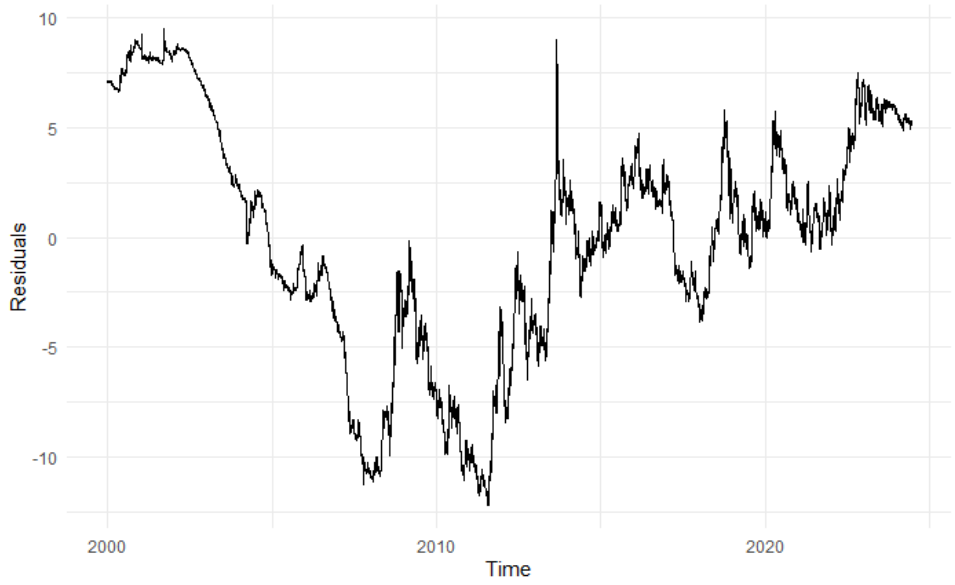

Figure 3.7: Residual plot of linear model

As we can see from the plot of the residuals [5], we can see that there is significant difference between the fitted values and the original data. The residuals keep ranging from -10 to 10. The model specifically do not capture the early part of the data. This suggests us the that the linear model is not the best fit for the data.

3.5.3 AR Model

Auto-Regressive models [2] are based on the idea that the current value of the series, y_t, can be explained as a function of p past values, $y_{t-1}, y_{t-2}, \ldots, y_{t-p}$ where p determines the number of steps into the past needed to forecast the current value. The general formula goes as follows:

$$y_t = \sum_{i=1}^{p} \phi_i y_{t-i} + \epsilon_t \tag{3.2}$$

where $\epsilon_t \sim \mathcal{N}(0, \sigma^2)$ is independent identically distributed (iid) noise with mean 0 and variance σ^2, p represents the number of lag terms, and y_{t-i} is the i-th lagged value in the data.
Let us try fitting an AR(1) model on the data. For this we will be using the **arima(x, order = c(p,0,0), method)** function, where x is the time series data, the p in the order parameter refers to the number AR coefficients and for method parameter we can use conditional sum of squares or maximum likelihood method. We here pass the data, specifying 1 AR coefficient which to be found out by maximum likelihood method[3].

```
ar1 <- arima(df$RATE, order = c(1,0,0), method = "ML")
summary(ar1)
```

```
Call:
arima(x = df$RATE, order = c(1, 0, 0), method = "ML")

Coefficients:
         ar1  intercept
      0.9999    57.3804
s.e.  0.0001    17.9925

sigma^2 estimated as 0.05216:  log likelihood = 364.29,  aic = -722.59

Training set error measures:
                      ME      RMSE       MAE          MPE      MAPE     MASE          ACF1
Training set 0.006238412 0.2283839 0.1454059 0.008944538 0.2560209 1.000476 -0.005987199
```

Figure 3.8: AR(1) model

The coefficient of the AR(1) model is 0.9999, which is very near to one. This suggests that that the series is following a unit root process, that is non-stationary and we know that it has a trend. So thus we can say that the model does not fit the best for the data due to its near-unit root behaviour.

3.5.4 ARIMA Model

An Auto-Regressive Integrated Moving Average (ARIMA) [2] model is specified by the following three parameters: (p, d, q).

Auto-Regressive (AR) part:

- The auto-regressive part involves regressing the variable on its own lagged (past) values.
- The parameter p is the number of lag observations included in the model (the number of terms in the auto-regressive part).

Integrated (I) part:

- The integrated part involves differencing the raw observations to make the time series stationary (i.e., having constant mean and variance over time).
- The parameter d is the number of times the differencing is applied to make the series stationary.

Moving Average (MA) part:

- The moving average part involves modeling the error term as a linear combination of error terms occurring contemporaneously and at various times in the past.
- The parameter q is the size of the moving average window (the number of terms in the moving average part).

Let us fit an ARIMA(1,1,0) model on our given data. We are differencing the data to remove the trend present in the data[3].

```
ar_d1 <- arima(df$RATE, order = c(1,1,0), method = "ML")
summary(ar_d1)
```

```
Call:
arima(x = df$RATE, order = c(1, 1, 0), method = "ML")

Coefficients:
          ar1
      -0.0052
s.e.   0.0125

sigma^2 estimated as 0.05221:  log likelihood = 365.16,  aic = -726.32

Training set error measures:
                      ME      RMSE       MAE          MPE      MAPE      MASE          ACF1
Training set 0.006287983 0.2284767 0.1453905 0.009443458 0.2560122 0.9997579 -0.001121566
```

Figure 3.9: ARIMA(1,1,0) model summary

As we can see, the model coefficient is -0.0052 value. This coefficient was found using the Maximum Likelihood Estimate method. Now let us check the residuals using **checkresiduals(model)** function in R, where we pass the model as a parameter.

```
checkresiduals(ar_d1)
```

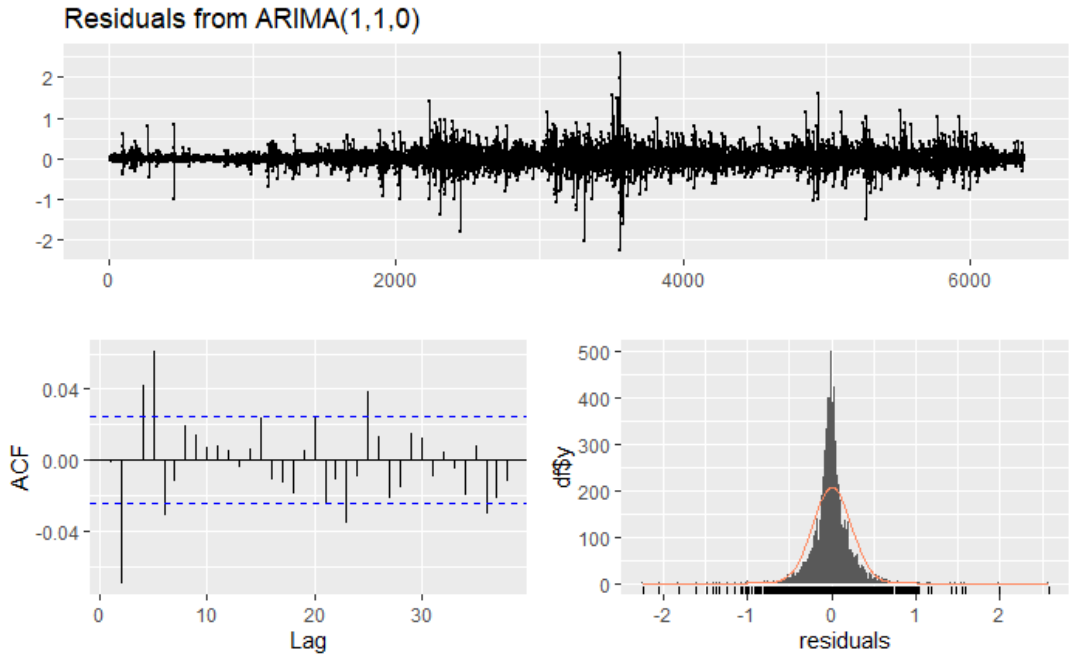

Figure 3.10: Different residual plots of ARIMA(1,1,0) model

```
        Ljung-Box test

data:  Residuals from ARIMA(1,1,0)
Q* = 76.895, df = 9, p-value = 6.669e-13

Model df: 1.   Total lags used: 10
```

Figure 3.11: Ljung Box Test of Residuals

The function gives the following results as we see above. Let us go through one-by-one. The top plot shows the plotting of the residuals. It provides us an understanding of how well the model is fitted, as we see that most residuals are clustered around zero with some occasional spikes. The ACF plot of the residuals is provided in the bottom left of the figure. As we can see most of the lags fall within the confidence interval suggesting that there is little auto-correlation. The few values exceeding the confidence interval suggest that there is some internal structure which the model could not capture very well.

Ljung-Box [5] test is a test used in time series analysis to assess whether the lagged values of the residuals from a time series model exhibit significant auto-correlation upto a specified lag (null hypothesis). Here in the above result we can assess the following:

- **Q***: It is the sum of squared auto-correlations of the residuals up to lag 10. This is the test static calculated for Ljung-Box test.
- **df**: It represents the degrees of freedom, which is the difference of number of lags used with number of parameters. It determines the number of auto-correlations that are being tested.
- **p-value**: This is the probability associated with the test statistic. Here we have a very small p-value (e.g., 6.669×10^{-13}) suggests strong evidence to reject H_0, i.e. there is significant autocorrelation in the residuals.

As there is significant autocorrelation in the residuals, we can say that the model is not the best fit for the given time series data. So we should go for nonlinear models such as TAR, STAR, etc.

3.5.5 TAR Model

Threshold Autoregressive (TAR) model [2] is a type of nonlinear time series model that incorporates a strict threshold to capture regime changes in the data. In TAR models, the regime switches abruptly when the time series crosses a certain threshold.
The Threshold Autoregressive (TAR) model is defined as:

$$X_t = \begin{cases} \phi_{1,0} + \sum_{i=1}^{p_1} \phi_{1,i} X_{t-i} + \epsilon_{1,t} & \text{if } X_{t-d} \leq \gamma \\ \phi_{2,0} + \sum_{i=1}^{p_2} \phi_{2,i} X_{t-i} + \epsilon_{2,t} & \text{if } X_{t-d} > \gamma \end{cases} \tag{3.3}$$

where:

- X_t is the time series data.

- $\phi_{1,0}, \phi_{2,0}$ are the intercept terms for the two regimes.
- p_1, p_2 are the number of lagged observations included in the model for each regime.
- $\phi_{1,i}, \phi_{2,i}$ are the autoregressive coefficients for each regime.
- γ is the threshold value.
- $\epsilon_{1,t}, \epsilon_{2,t}$ are the white noise error terms for each regime.

We can fit a TAR model for the data in R using the `setar(x, m = 2, thDelay = 1, model = "TAR")` function from the `tsDyn` package in R, where:

- `x` is the time series data,
- `m` is the embedding dimension,
- `thDelay` is the time delay for the threshold variable,
- `model` is used to represent which model to use.

```
tar_model <- setar(df$RATE, m = 2, thDelay = 1, model = "TAR")
summary(tar_model)
```

```
SETAR model ( 2 regimes)
Coefficients:
Low regime:
     const.L       phiL.1       phiL.2
-0.008769538  0.804014735  0.196357122

High regime:
     const.H       phiH.1       phiH.2
 0.000872988  1.010832919 -0.010743396

Threshold:
-Variable: Z(t) = + (0) X(t)+ (1)X(t-1)
-Value: 44.52
Proportion of points in low regime: 15.02%       High regime: 84.98%

Residuals:
       Min         1Q     Median         3Q        Max
-2.2845353 -0.0889303 -0.0052547  0.0879108  2.5319251

Fit:
residuals variance = 0.05194,  AIC = -18856, MAPE = 0.2562%

Coefficient(s):

           Estimate  Std. Error  t value  Pr(>|t|)
const.L -0.00876954  0.17692855  -0.0496    0.9605
phiL.1   0.80401473  0.04365901  18.4158 < 2.2e-16 ***
phiL.2   0.19635712  0.04378933   4.4841 7.449e-06 ***
const.H  0.00087299  0.01502571   0.0581    0.9537
phiH.1   1.01083292  0.01305200  77.4466 < 2.2e-16 ***
phiH.2  -0.01074340  0.01305536  -0.8229    0.4106
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Threshold
Variable: Z(t) = + (0) X(t) + (1) X(t-1)

Value: 44.52
```

Figure 3.12: TAR model summary

We used the **SETAR()** function to fit a TAR model[3], and we got the following model. The threshold values was found out to be 44.52, and the estimates of each regime AR coefficients are given too. As we can see in the summary only 15.02% is only in the lower regime and the rest, that is 84.98% are in the higher regime. The residuals range from -2.28 to 2.53.

Now let us look into the fitting of the model. We shall be separating the fitted values based on the regime, and plotting the fitted values along the original data.

```r
threshold <- 44.52

fitted_values <- fitted.values(tar_model)

df$Fitted_RATE <- c(rep(NA, length(df$RATE) - length(fitted_values))
   , fitted_values)

df$Regime <- ifelse(df$Fitted_RATE > threshold, 1, 2)

ggplot(df, aes(x = as.Date(DATE))) +
  geom_line(aes(y = RATE), color = "red") +
  geom_line(aes(y = Fitted_RATE, color = as.factor(Regime))) +
  scale_color_manual(values = c("1" = "blue", "2" = "green")) +
  geom_hline(yintercept = threshold, color = "black", size = 1) +
  labs(x = "Date",
       y = "RATE",
       color = "Regime") +
  theme_minimal() +
  theme(legend.position = "none")
```

Figure 3.13: Fitting of TAR model

As we can see in the above graph, the fitted values almost precisely fits the original data. Thus proves that the given model is a good fit of the data. For further understanding let us look into the ACF plot of residuals.

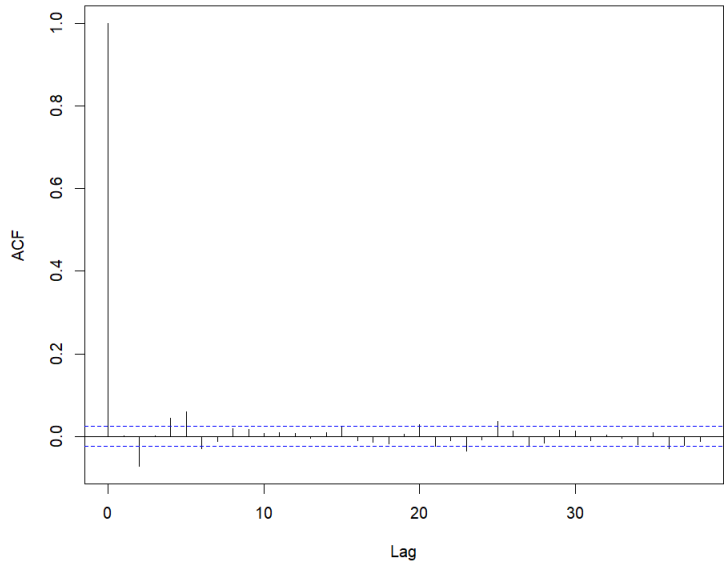

Figure 3.14: ACF of TAR model residuals

The residuals of the plot have very low correlation between them stating that the TAR model has captured the complexities of the data really well. There are only a little lags above the confidence interval. So TAR model is a good fit of the data. We shall be seeing further forecasting using the model in the results section.

3.5.6 STAR Model

Smooth Threshold Auto-Regressive (STAR) model [4] is a type of nonlinear time series model that incorporates smooth threshold functions to capture nonlinear relationships between the current observation and its lagged values. Unlike the hard threshold in Threshold Auto-Regressive (TAR) models, where the regime switches abruptly at a certain threshold, the smooth threshold function allows for gradual transitions between regimes. Common choices for S include logistic or exponential functions.
The Smooth Threshold Auto-Regressive (STAR) model is defined as:

$$X_t = \mu + \sum_{i=1}^{p} \phi_i \left(X_{t-i} - \gamma_i S(X_{t-i-1})\right) + \epsilon_t \tag{3.4}$$

where:

- X_t is the time series at time t.
- μ is the intercept term.
- p is the number of lagged observations included in the model.
- ϕ_i are the autoregressive coefficients.
- γ_i are the threshold parameters.

- $S(\cdot)$ is the smooth threshold function.
- ϵ_t is the white noise error term.

We can fit a STAR model for the data in R using the **star(x, m=2, noRegimes, d=1, thDelay = 1, sig = 0.05)** function from the tsDyn package in R, where:

- `x` is the time series datad,
- `m` is the embedding dimension,
- `noRegimes` is the maximum number of regimes,
- `d` is the time delay,
- `thDelay` is the time delay for the threshold variable,
- `sig` is the significance level.

```
star_model <- star(df$RATE, m = 3, noRegimes = 2, d = 1, thDelay =
   1, sig = 0.05)
```

```
Testing linearity...   p-Value =  4.733342e-06
The series is nonlinear. Incremental building procedure:
Building a 2 regime STAR.
Performing grid search for starting values...
Starting values fixed: gamma =  72.07692 , th =  44.50276 ; SSE =  329.5165
Optimization algorithm converged
Optimized values fixed for regime 2  : gamma =  72.07694 , th =  44.51943 ; SSE =  329.5092
Finished building a MRSTAR with 2 regimes
```

Figure 3.15: STAR model

As we can see in the STAR model the starting values are gamma value of 72.07692 which determines how sharply or smoothly the transitions between regimes occur in the model. Higher the gamma value sharper the transition between regimes. The theta value is chosen as 44.50276 which determines the point around which the transition between regime occurs.

We shall now check into the fitted values of the model. We can get the fitted values of the model using the **fitted.values(model)** function in R, by passing the model into it. After plotting the values alongside the original we can see that how well the model fits the data.

```
fitted_values <- fitted.values(star_model)

combined_df <- data.frame(
  Date = df$DATE,
  Original_RATE = df$RATE,
  Fitted_RATE = c(rep(NA, length(df$RATE) - length(fitted_values)),
     fitted_values)
)

ggplot(combined_df, aes(x = Date)) +
  geom_line(aes(y = Original_RATE, color = "Original")) +
  geom_line(aes(y = Fitted_RATE, color = "Fitted")) +
  scale_color_manual(values = c("Original" = "blue", "Fitted" = "red
     ")) +
  labs(x = "Date",
       y = "RATE",
       color = "Legend") +
  theme_minimal()
```

Figure 3.16: Fitted values of STAR model with original data

From the above graph we can see that the fitted values of the STAR model fits original data very well as the original data values are overlapped very well. So we can infer that the model is a good fit for the data.

Let us look into the ACF plot of the models residuals to further accurately look into our inference. As said before we will be using the **ACF(data)** function to plot the ACF plot.

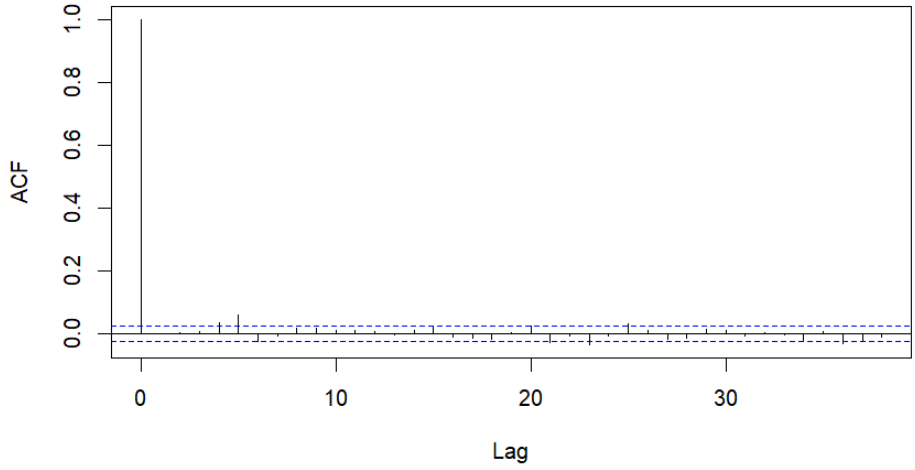

Figure 3.17: ACF of STAR model residuals

As we see from the plot above, there are less values which are crossing the confidence interval compared to the plot of the residuals of the TAR model. This depicts that it has an even better understanding of the data when compared to the TAR model. This model accurately fits the data. We shall further look into the forecasting using the model in the results section.

3.6 Results

We could see that both the TAR model and STAR model fitted the data the best compared to other linear models. Both the models fitted the data very well and the residuals of both models where very low. And the residuals of the models had very less auto-correlation in different lags too.

3.6.1 Forecast using TAR model

We use the **forecast(model, n.ahead=1)** function to predict future values, specifying the model and the prediction length with the n.ahead parameter. To visualize the forecast alongside the original data, we plot both together. Predictions for the next 7 days using both TAR and STAR models show almost similar results, suggesting that both models fit the data comparably well. Let us now examine the plot of these predicted values combined with the original data to better understand the forecast's alignment with historical trends.

```
Time Series:
Start = 6381
End = 6387
Frequency = 1
[1] 83.54857 83.55701 83.56546 83.57390 83.58235 83.59079 83.59924
```

Figure 3.18: TAR Forecast Values

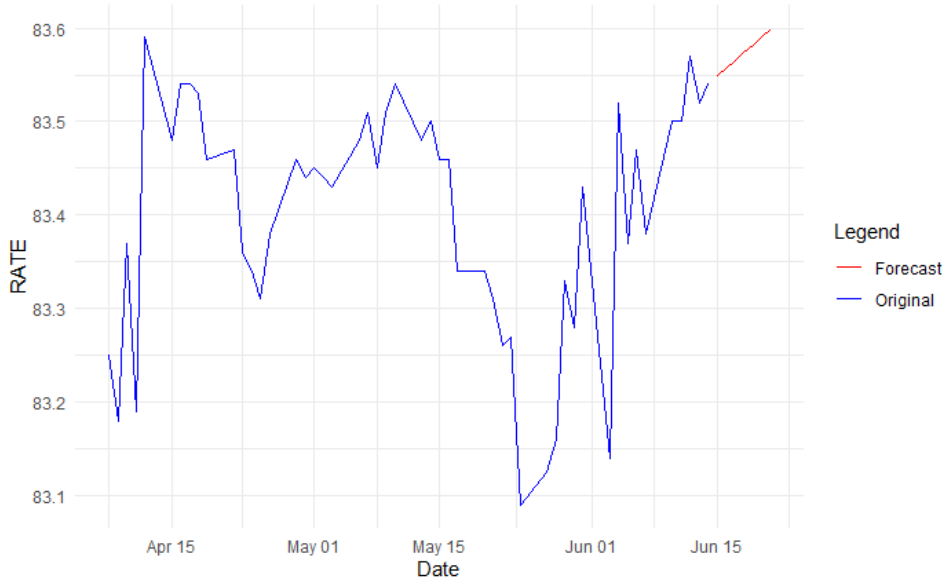

Figure 3.19: Zoomed-in TAR forecast

3.6.2 Forecast using STAR model

Similar to the above forecasting, we can do the same for the STAR model. We shall predict the next 7 forecasted values using the function and plot them. For easy visualization, we opted for only a few starting values of the original data.

```
Time Series:
Start = 6381
End = 6387
Frequency = 1
[1] 83.55358 83.56164 83.57014 83.57908 83.58799 83.59686 83.60573
```

Figure 3.20: STAR model forecast values

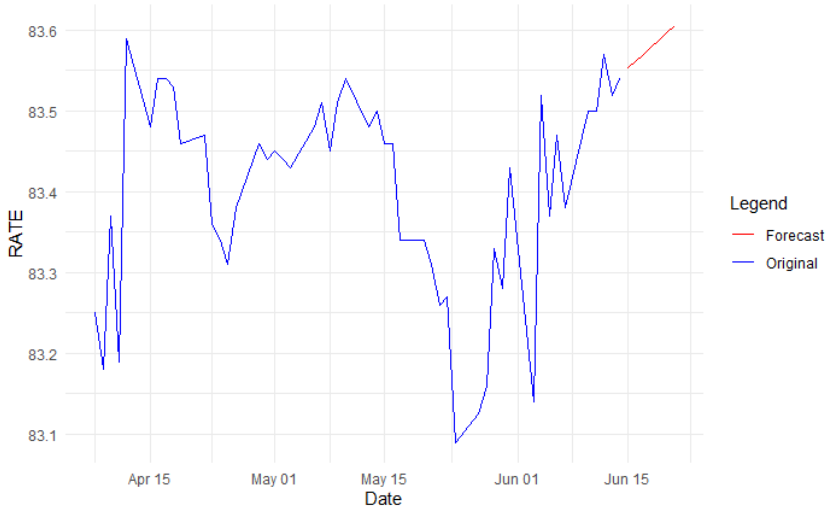

Figure 3.21: Zoomed-in STAR forecast

3.6.3 Model Evaluation

Here we can see in the results, both of them give almost similar forecasts. This shows that both the models captures the trend similarly. Thus we can say that both the models have very good fitting and forecasting of the data.

Now we can compare the models used using the Akaike Information Criterion (AIC) values of the models using the **AIC(model)** function in R. AIC is used for the evaluation of a model to know how well the model works. The lower the AIC value, the better the model is. The AIC is computed using the formula:

$$\text{AIC} = 2k - 2\ln(L) \tag{3.5}$$

where k is the number of parameters in the model, and $\ln(L)$ is the log-likelihood of the model.

```
aic_ar_d1_model <- AIC(ar_d1)
aic_tar_model <- AIC(tar_model)
aic_star_model <- AIC(star_model)

print(paste("AIC for ARIMA(1,1,0) model:", aic_ar_d1_model))
print(paste("AIC for TAR model:", aic_tar_model))
print(paste("AIC for STAR model:", aic_star_model))
```

```
[1] "AIC for ARIMA(1,1,0) model: -733.358633593421"
[1] "AIC for TAR model: -18855.7439923718"
[1] "AIC for STAR model: -18885.0205506413"
```

Figure 3.22: AIC values of the models

Now we can see that the STAR model has a lower AIC value compared to the TAR model, which stating that the STAR model is the better model over the TAR model. For further inference we can see the AIC value of ARIMA(1,1,0) model also here. We can see how much better the AIC value of the non-linear models are compared to that of the ARIMA model.

Chapter 4

Study Material Project

I was involved in the creation of study material for a workshop conducted at IIT Bombay on ”Linear Time Series”. The study material created was on time series linear models, including AutoRegressive (AR), Moving Average (MA), and AutoRegressive Moving Average (ARMA) models. Two documents were created. The first document focused on the coding aspect, i.e., the practical usage of the models, and the second one covered the underlying theoretical concepts of the code. The study aims to build and demonstrate the foundation required for the practical applications of these models in time series data forecasting. The study material included simulation of the data, model fitting, residual analysis, ACF and PACF plots, and forecasting results using each model. This resource can be accessed using the following link:

Link to Study Material: `https://r.fossee.in/resources`

Chapter 5

Conclusion

During my FOSSEE Summer Fellowship 2024, I focused on enhancing the use and understanding of R, an open-source language for statistical analysis. Key initiatives included the Textbook Companion Project, where I developed educational resources by coding textbook solutions in R. In the Foreign Exchange Rate Case Study, analyzed the INR/USD dataset to provide insights into time series analysis and currency forecasting, which will be valuable for economists and financial organizations. The creation of study material, producing resources on linear time series models shall further enrich the available educational content.

This on-campus fellowship promoted group learning and gave me useful skills that I can use in my future pursuits. This experience improved my technical proficiency and understanding of organizational dynamics. The projects increased the usability and awareness of open-source software, specifically R, and instilled in me values of dedication, hard work, and contributing to the community.

References

[1] Federal Reserve Bank. (n.d.). Foreign Exchange Rates - H.10. Retrieved from https://www.federalreserve.gov/releases/h10/current/default.htm

[2] Tsay, R., & Chen, R. (Year). *Nonlinear Time Series Analysis*.

[3] Kratofil, M. A. (Year). *Introductory Time Series with R*.

[4] van Dijk, D., Teräsvirta, T., & Franses, P. H. (Year). Smooth Transition Autoregressive Models: A Survey of Recent Developments. Econometric Institute Research Report EI2000-23/A.

[5] Hyndman, R. J., & Athanasopoulos, G. (Year). *Forecasting: Principles and Practice* (2nd ed.)

[6] Holmes, E. E., Scheuerell, M. D., & Ward, E. J. (Year). *Applied Time Series Analysis for Fisheries and Environmental Sciences*

[7] Haben, S., Voss, M., & Holderbaum, W. (2023). Time Series Forecasting: Core Concepts and Definitions. In *Core Concepts and Methods in Load Forecasting*. Springer. https://doi.org/10.1007/978-3-031-27852-5_5

[8] Predicting Exchange Rate between US Dollar (USD) and Indian Rupee (INR): An Empirical Analysis using SARIMA Model(2024).*International Journal of Research in Finance and Management*, 7(1), 1–10. https://doi.org/10.33545/26175754.2024.v7.i1a.283