Quest Journals
Journal of Research in Applied Mathematics
Volume 9 ~ Issue 5 (2023) pp: 46-50
ISSN (Online): 2394-0743 ISSN (Print): 2394-0735
www.questjournals.org

Review Paper

Univariate Time-Series Forecast Computing via R 'auto.arima()' Function

Dewi Rahardja
Statistician, U. S. Department of Defense, Fort Meade, MD 20755, USA.
(Disclaimer Statement – This research represents the author's own work and opinion. It does not reflect any policy nor represent the official position of the U.S. Department of Defense nor any other federal agency.)

ABSTRACT: *In this paper, we employ the well-known Auto-Regressive Integrated Moving Average (ARIMA) family of Time Series (TS) forecast algorithms reviewed in Rahardja (2020), as a convenient way to forecast automatically in R software, while taking into account the TS attributes in terms of parameters (p, d, q, and P, D, Q), using Spectral Decomposition algorithm. We execute such ARIMA-family univariate automatic forecasting via 'auto.arima()' function in R. Familiarity with Box-Jenkins methods (1976) is not required to forecast via such an automatic R function. For a walkthrough example, we apply such automatic R function to the famous monthly airline passenger data-series example.*
KEYWORDS: *Time Series, ARIMA, Univariate, R, Forecast.*

***R**eceived 16 May, 2023; **R**evised 28 May, 2023; **A**ccepted 31 May, 2023 © The author(s) 2023.*
P**ublished with open access at **www.questjournals.org

I. INTRODUCTION

In this scientific age, numerous organizations need to forecast their future (counts) of products or service. For instance, in various fields such as business, finance, economic, etc., they need to forecast their periodic (daily, weekly, monthly, quarterly, annually) request (counts). For instance, weekly mango sales, monthly salmon sales, yearly jewelry sales, etc.

However, with the fast-paced world, many forecasters or researchers in various fields of study, cannot afford to sort out what rigorous statistical methods and computing algorithms are available [1–15] to implement their forecasts. Previously, such statistical methods and computing algorithms are summarized in Rahardja (2020) paper [16].

As a brief recap, the Rahardja (2020) paper [16] organized the literature review into the 3-family category of Statistical Time-Series (TS) forecasting methods (see Table 1 in that paper, for the listings of each TS-method's name and its model equation). Recall that such 3-family category TS models are the Exponential Smoothing Model (ESM) family [2–15], the Auto-Regressive Integrated Moving Average (ARIMA) family models, which are a form of Box-Jenkins model [1], and the Unobserved Component Model (UCM) family, which is also called the Structural Models in the TS literature [7]. The ARIMA-family can handle much more complex models beyond the ESM-family and are beyond the scope of what Excel [17] can compute. The UCM-family can further handle what typically cannot be captured by ESM-family and/or ARIMA-family models but beyond the scope of this paper.

Among many past research [17–18] have summarized several TS-forecast implementation options. For instance, implementing the ESM-family univariate forecast via Excel [17], or executing batch forecasting via the SAS Forecast Studio automatic/drop-down menu [18]. Now, we would like to summarize how to implement univariate forecast for ARIMA models via an automatic R function, 'auto.arima()' [19–20], from its lengthy and complete source, to dive-in deeper.

In this paper, we manage the sections as follow. In Section 2 we explain the materials and methods. In Section 3, we present the results and discussion. Finally in Section 4, we conclude our paper.

II. MATERIALS AND METHODS

The materials used here are TS dataset (monthly 'Airline Passengers') and the statistics R software (free-and-downloadable). The methods used here are the famous TS-forecasting methods, the ARIMA-family models. We implement such ARIMA forecast via an automatic R function, 'auto.arima()' [19–20].

The famous Airline Passengers dataset in R provides a 144-monthly totals of a US airline passengers, from 1949 to 1960 (see Figure 1). Such dataset is available online via Google search. The dataset is also available from an inbuilt dataset of R called 'Air Passengers'. The source of the dataset is from Box and Jenkins (1976) famous book [1], "Time Series Analysis: Forecasting and Control," page 531.

	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
1949	112	118	132	129	121	135	148	148	136	119	104	118
1950	115	126	141	135	125	149	170	170	158	133	114	140
1951	145	150	178	163	172	178	199	199	184	162	146	166
1952	171	180	193	181	183	218	230	242	209	191	172	194
1953	196	196	236	235	229	243	264	272	237	211	180	201
1954	204	188	235	227	234	264	302	293	259	229	203	229
1955	242	233	267	269	270	315	364	347	312	274	237	278
1956	284	277	317	313	318	374	413	405	355	306	271	306
1957	315	301	356	348	355	422	465	467	404	347	305	336
1958	340	318	362	348	363	435	491	505	404	359	310	337
1959	360	342	406	396	420	472	548	559	463	407	362	405
1960	417	391	419	461	472	535	622	606	508	461	390	432

Figure 1: The 'Airline Passengers' dataset in R (converted as TS object).

In R, we define TS as a series of values, each associated with the timestamp also measured over regular intervals (daily, weekly, monthly, quarterly, yearly). The R software stores the TS data in the TS object and is created using the 'ts()' function as a base distribution. The syntax declaration of the TS function is given as 'ts(data, start, end, frequency)'. Here, the 'data' specify values in the TS, the 'start' specifies the first forecast observations in a TS value, the 'end' specifies the last observation value in a TS, and 'frequency' specifies periods of observations (month, quarter, annual). Before we start using the 'ts()' function, we need to load the 'forecast-Package,' in what follows (as R-code).

```
#--- Load the 'forecast-Package' ---#
install.packages('forecast')
library(forecast)
```

There are multiple pathways to enter TS dataset into R. For beginners, since TS dataset are not too big, the easiest and most common pathway is to save them as a text file format (.txt) and then copy-paste them into R, as an array of data (separated by commas). In any pathway, any TS dataset read into R software still need to be converted to a TS object/entity, which is totally different than any other type of data [21–24]. Here, although the 'Air Passengers' dataset is available already as inbuilt R dataset, we will still briefly demonstrate the easiest way for beginners, to create an array and convert it to a TS object/entity, in the R-code below.

```
#--- Example 1 (Create An Array of Data) ---#
AirPassengers <- c(112, 118, 132, 129, 121, 135, 148, 148, 136, 119, 104, 118,
115, 126, 141, 135, 125, 149, 170, 170, 158, 133, 114, 140,
145, 150, 178, 163, 172, 178, 199, 199, 184, 162, 146, 166,
171, 180, 193, 181, 183, 218, 230, 242, 209, 191, 172, 194,
196, 196, 236, 235, 229, 243, 264, 272, 237, 211, 180, 201,
204, 188, 235, 227, 234, 264, 302, 293, 259, 229, 203, 229,
242, 233, 267, 269, 270, 315, 364, 347, 312, 274, 237, 278,
284, 277, 317, 313, 318, 374, 413, 405, 355, 306, 271, 306,
315, 301, 356, 348, 355, 422, 465, 467, 404, 347, 305, 336,
340, 318, 362, 348, 363, 435, 491, 505, 404, 359, 310, 337,
360, 342, 406, 396, 420, 472, 548, 559, 463, 407, 362, 405,
417, 391, 419, 461, 472, 535, 622, 606, 508, 461, 390, 432)

#--- Convert Dataset Into TS Object ---#
# Since it is a monthly data, frequency is set to 12.
AirPassengers.TS <- ts(AirPassengers, start=c(1949,1), end=c(1960,12),frequency=12)
# Hence start date is January 1949 while the end date is December 1960.
## To plot the TS object to observe any pattern as an initial check
plot(AirPassengers.TS, ylab='Number of Passengers', main='TS plot of Airline Passengers')
```

Next, after the dataset is converted to a TS dataset, now it is ready to forecast using the automatic ARIMA forecasting function of R, the "auto.arima()". We can then proceed to the next section.

On a brief note, pretend that we did not enter/create the dataset in R, as an array. In other words, at this point, let's start R-coding from zero. Since the 'Air Passengers' dataset is already available as inbuilt R dataset, we can easily load them and do several checks on the dataset.

```
#--- Example 2 (Load An Inbuilt R Dataset) ---#
##Load the Forecast Package
install.packages('forecast')
library(forecast)
##Load the Air Passengers' Dataset and View Its Class
data("AirPassengers")
class(AirPassengers)
##Display the Dataset to see any patterns such as trends, level, seasonality
AirPassengers
##Check on date values to see the range of the dataset
start(AirPassengers)
end(AirPassengers)
#Hence start date is January 1949 while the end date is December 1960
##Find out any Missing Values
sum(is.na(AirPassengers))
##Check the Summary of the Dataset
summary(AirPassengers)
##Plot the Dataset to precheck any visually detectable pattern
plot(AirPassengers)
```

Subsequently, we can then proceed to the next section, i.e., to forecast using the automatic ARIMA forecasting function of R, the "auto.arima()".

III. RESULTS AND DISCUSSION

Here in this section, we provide the results and discussion of a walkthrough example (the 'Air Passengers' dataset) on univariate TS forecast computing via the R automatic ARIMA function "auto.arima()". To recap briefly, the "auto.arima()" function in R uses a variation of the Hyndman-Khandakar algorithm (Hyndman & Khandakar, 2008) [19–20], which combines unit root tests, minimization of the Aikaike Information Criteria (AIC) [25–26] and the maximum likelihood estimation (MLE) [27] to obtain an ARIMA [28] model. Below is a simple walkthrough R-code example (continuing from the previously ran R-code):

```
#--- Build the ARIMA Model Using auto.arima() Function ---#
mymodel <- auto.arima(AirPassengers)
mymodel
#ARIMA(211)(010)12
##Plot the Residuals (to check any obvious patterns)
plot.ts(mymodel$residuals)
##Forecast the Values for the for the next targeted several years (say 3 yrs)
myforecast <- forecast(mymodel, level=c(95), h=3*12)
plot(myforecast)
##Validate the Model by Selecting Lag Values via Ljung-Box test or any other ways
Box.test(mymodel$resid, lag=5, type="Ljung-Box")
Box.test(mymodel$resid, lag=10, type="Ljung-Box")
Box.test(mymodel$resid, lag=15, type="Ljung-Box")
```

As a recap, basically the Ljung-Box [29] test works as follows. The Ljung-Box is a 'portmanteau' test [30] that assesses the null hypothesis that a series of residuals exhibits no autocorrelation for a fixed number of lags L, against the alternative that some autocorrelation coefficient $\rho(k)$, $k = 1, ..., L$ is nonzero. In other words, the null hypothesis of Ljung-Box test is that the residuals are white noise (WN); versus the alternative hypothesis that the residuals are not WN. If the p-value is in-favor of the Null Hypothesis (i.e., the p-value is greater than the pre-specified alpha level of confidence), then stop. Meaning, your model is good (accurate) enough at such alpha level. Typically, alpha is pre-specified to be 5% level. Otherwise, when the Null

Hypothesis is rejected (i.e., in-favor of the Alternative Hypothesis), then repeat until your model is good enough, for example, by varying the lag, or any other methods which are beyond the scope of this paper.

Here, in this example, there is no obvious patterns from the residuals plot and looking at the p-values (0.7116, 0.562, 0.7104) for lags=5, 10, 15, subsequently, the Null Hypothesis cannot be rejected and we can say that our model fit is adequate and hence accurate. Therefore, we can conclude from the (automatic R function) output, the resulting ARIMA(2,1,1)(0,1,0)$_{12}$ model, with such ARIMA parameters adequately fits the data well. We can see such (automatically selected) model in Figure 2.

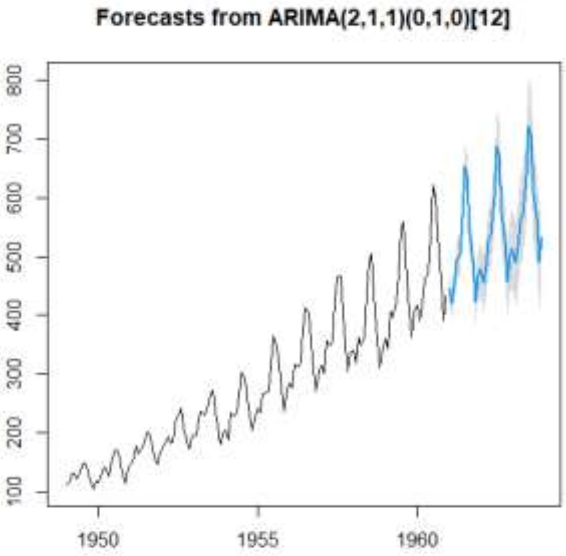

Figure 2: A forecast model resulting from the "auto.arima()" function on the 'Air Passengers' dataset.

In Figure 2, we have the 12-year (144-month) TS dataset plot from January 1949 to December 1960 (in black lines) and the subsequent 3-year (36-month) 'Air Passengers' volume of forecasts (in blue lines for points estimates and in the grey-shaded areas, corresponding to the 95% confidence intervals estimates), from January 1961 to December 1963. As we can see, there are level (intercept), trend (slope), and seasonality, resulting from the automatic ARIMA parameters output: the ARIMA(2,1,1)(0,1,0)$_{12}$ model, with the 12-month seasonality, represented by the lower-case symbol.

IV. CONCLUSION

In this paper, we have demonstrated a quick-and-easy univariate-computing option of TS-forecasting via R "auto.arima()" automatic function [17–21]. We conclude that such an automatic R "auto.arima()" function is very useful-and-convenient way to implement forecast using the famous ARIMA-family of the TS methods [2–15] reviewed in Rahardja [16], while taking into account TS attributes in terms of ARIMA parameters (p, d, q, and P, D, Q). This automatic R-pathway of TS-forecast option requires very small computing resource. Familiarity with the Box-Jenkins methods [1] is not required to forecast via such automatic R function.

Using a TS (144-month) 'Air Passengers' dataset as a walkthrough example, we have illustrated the application of "auto.arima()" function in R to forecast the future 36-month period projections of 'Air Passengers' volume. Additionally, we also have demonstrated the application of Ljung-Box test [29] to test whether the residuals are WN.

Therefore, this automatic "auto.arima()"function in R is highly recommendable for many users without any knowledge of Box-Jenkins methods [1] due to its user-friendliness, economic viability (free-downloadable software), and requires a very small computing resource. Such an automatic ARIMA function in R will select a local optimum (baseline) solution among ARIMA-family candidate models. For a starter, such baseline output model is adequate.

Moreover, there are many non-automatic ways to improve a univariate-TS forecast (still in R) beyond a baseline forecast found by the "auto.arima()" function, which cannot be captured by ARIMA-family and/or its ESM-equivalent family models (as listed in Rahardja [16] paper). For instance, via the following functions under the forecast-Package: arima(), ets(), ts(), stl(), and/or the function ucm() under the rucm-Package; or any other deterministic (non-stochastic) models. However, such non-automatic forecasting ways are beyond the scope of this paper.

DISCLAIMER STATEMENT

This research represents the author's own work and opinion. It does not reflect any policy nor represent the official position of the U.S. Department of Defense nor any other federal agency.

REFERENCES

[1]. Box, G.E.P., and Jenkins, G.M. (1976). Time Series Analysis: Forecasting and Control. John Wiley & Sons, Inc., Hoboken, New Jersey.

[2]. Brown, R.G. (1959). Statistical Forecasting for Inventory control. McGraw-Hill: New York, NY.

[3]. Brown, R.G. (1962). Smoothing, Forecasting and Prediction of Discrete Time Series. Prentice Hall: New Jersey, NJ.

[4]. De Gooijer, J.G., and Hyndman, R.J. (2006). 25 years of Time Series Forecasting. International Journal of Forecasting. **22**(3): p. 443–473.

[5]. Fomby, T.B. (2008). Exponential Smoothing Models. Class Notes Version 6. Department of Economics. Southern Methodist University, Dallas, TX, June 2008.

[6]. Gardner, E. S., and McKenzie, E. (1985). Forecasting trends in time series. Management Science, **31**(10): p. 1237–246. DOI: https://doi.org/10.1287/mnsc.31.10.1237.

[7]. Harvey, A.C. (1989). Forecasting, Structural Time Series Models, and the Kalman Filter. Cambridge University Press.

[8]. Holt, C. E. (1957). Forecasting seasonals and trends by exponentially weighted averages (O.N.R. Memorandum No. 52). Carnegie Institute of Technology, Pittsburgh, USA. DOI: https://doi.org/10.1016/j.ijforecast.2003.09.015

[9]. Hyndman, R.J., and Athanasopoulos, G. (2018). Forecasting Principles and Practices. Monash University, Australia. https://otexts.org/fpp2/.

[10]. Fried R., George A.C. (2011). Exponential and Holt-Winters smoothing. In: Lovric M. (eds) International Encyclopedia of Statistical Science. Springer, Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-642-04898-2_24.

[11]. Chatfield, C. (1978). The Holt-Winters Forecasting Procedure. Journal of the Royal Statistical Society. Series C (Applied Statistics), **27**(3): p. 264-279. DOI: https://doi.org/10.2307/2347162.

[12]. Levenberg, Kenneth (1944). A Method for the Solution of Certain Non-Linear Problems in Least Squares. Quarterly of Applied Mathematics, **2**(2): p. 164–168.

[13]. Marquardt, Donald (1963). An Algorithm for Least-Squares Estimation of Nonlinear Parameters. SIAM Journal on Applied Mathematics, **11**(2): p. 431–441. DOI: https://doi.org/10.1137/0111030.

[14]. Sypsas, P.T. (1989). Identifying Patterns in Multiple Time Series Data. Journal of Information and Optimization Sciences, **10**(3): p. 471–494.

[15]. Winters, P. R. (1960). Forecasting sales by exponentially weighted moving averages. Management Science, **6**(3): p. 324–342.

[16]. Rahardja, D. (2020). Statistical methodological review for time-series data. Journal of Statistics and Management Systems, **23**(8): p. 1445–1461. DOI: https://doi.org/10.1080/09720510.2020.1727618.

[17]. Rahardja, D. (2021). Statistical Time-Series Forecast via Microsoft Excel (FORECAST.ETS) Built-In Function. Journal of Research in Applied Mathematics, **7**(11): p. 69–73. https://questjournals.org/jram/papers/v7-i11/K07116973.pdf.

[18]. Rahardja, D. (2021). Statistical Computing for Time-Series Forecasts via SAS Forecast Studio. International Journal of Applied Science and Mathematics, **8**(4): p. 38–45. https://www.ijasm.org/index.php/archive?view=publication&task=show&id=213.

[19]. Rob J. Hyndman, R.J. and Khandakar, Y. (2008). Automatic Time Series Forecasting: The forecast Package for R. Journal of Statistical Software, **27**(3), p. 1–22. DOI: https://doi.org/10.18637/jss.v027.i03.

[20]. Rob J. Hyndman, R.J. and Khandakar, Y. (2008). Automatic Algorithms for Time Series Forecasting. The Talk: Follow Along Using R. https://robjhyndman.com/talks/Google-Oct2015-part1.pdf.

[21]. Rahardja, D., Yang, Y., and Zhang, Z. (2016). A Comprehensive Review of the Two-Sample Independent or Paired Binary Data – with or without Stratum Effects. Journal of Modern Applied Statistical Methods, **15**(2): p. 215–223. https://digitalcommons.wayne.edu/jmasm/vol15/iss2/16/.

[22]. Rahardja, D. (2017). A Review of the Multiple-Sample Tests for the Continuous-Data Type. Journal of Modern Applied Statistical Methods, **16**(1): p. 127–136. https://digitalcommons.wayne.edu/jmasm/vol16/iss1/8/.

[23]. Rahardja, D., and Wu, H. (2018). Statistical Methodological Review for Time-To-Event Data. Journal of Statistics and Management Systems, **21**(1): p. 189–199. DOI: https://doi.org/10.1080/09720510.2017.1411029.

[24]. Rahardja, D. (2021). A Note on Statistical Methods for Most-Frequently Seen 3-Type of Data. International Journal of Applied Science and Mathematics, **8**(1): p. 1–8. https://www.ijasm.org/index.php/archive?view=publication&task=show&id=208.

[25]. Akaike, H. (1981). This Week's Citation Classic (PDF). Current Contents Engineering, Technology, and Applied Sciences, **12**(51): 42. [Hirotogu Akaike comments on how he arrived at AIC].

[26]. Akaike Information Criteria (AIC), Wikipedia (2020). https://en.wikipedia.org/wiki/Akaike_information_criterion.

[27]. Maximum Likelihood Estimation (MLE), Wikipedia (2023). https://en.wikipedia.org/wiki/Maximum_likelihood_estimation.

[28]. Auto-Regressive Integrated Moving Average (ARIMA), Wikipedia (2023). https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average.

[29]. Ljung–Box test, Wikipedia (2023). https://en.wikipedia.org/wiki/Ljung-Box_test.

[30]. Portmanteau test, Wikipedia (2023). https://en.wikipedia.org/wiki/Portmanteau_test.