Time Series Analysis in R

Pranshu Kumar

**Table of Contents**

[Time Series Analysis in R 3](#_Toc31639325)

[Introduction 4](#_Toc31639326)

[Analysis 5](#_Toc31639327)

[Conclusion 15](#_Toc31639328)

[References 16](#_Toc31639329)

**Time Series Analysis in R**

Using R to apply time series on a dataset in R to predict forecasts for for making better future business decisions.

# Introduction

The project deals with applying time series data in R on a dataset for decomposing the seasonal time series, forecasting using Exponential Smoothing and using ARIMA to address issues of correlations between successive values of time series.

The project consists of two parts –

**Part A** – involves decomposing seasonal time series data and subtracting that effect from data for seasonal adjustments.

**Part B** – addresses the issues of correlations between successive values of the time series, make a better predictive model by addressing autocorrelation issues using ARIMA for irregular components.

## Analysis

**Part A – Seasonal Time Series data and adjusting**

We are using a time series data of the **number of births per month in New York city, , from January 1946 to December 1959**.

We successfully read the data in R, and stored it as a time series object.

At First, we analysed our time series data in R as well as plotted it.

We used the **scan()** function to read data into R, which assumed the data for successive time points in a single text file with a single column.

n\_birth <- scan("http://robjhyndman.com/tsdldata/data/nybirths.dat")

**## Read 168 items**

> birth\_ts <- ts(n\_birth, frequency=12, start=c(1946,1))

> birth\_ts

**## Jan Feb Mar Apr May Jun Jul Aug Sep**

**## 1946 26.663 23.598 26.931 24.740 25.806 24.364 24.477 23.901 23.175**

**## 1947 21.439 21.089 23.709 21.669 21.752 20.761 23.479 23.824 23.105**

**## 1948 21.937 20.035 23.590 21.672 22.222 22.123 23.950 23.504 22.238**

**## 1949 21.548 20.000 22.424 20.615 21.761 22.874 24.104 23.748 23.262**

**. . . . . . . .**

**. . . . . . . .**

**## 1959 26.076 25.286 27.660 25.951 26.398 25.565 28.865 30.000 29.261**

**## Oct Nov Dec**

**## 1946 23.227 21.672 21.870**

**## 1947 23.110 21.759 22.073**

**## 1948 23.142 21.059 21.573**

**. . .**

**. . .**

**## 1959 29.012 26.992 27.897**

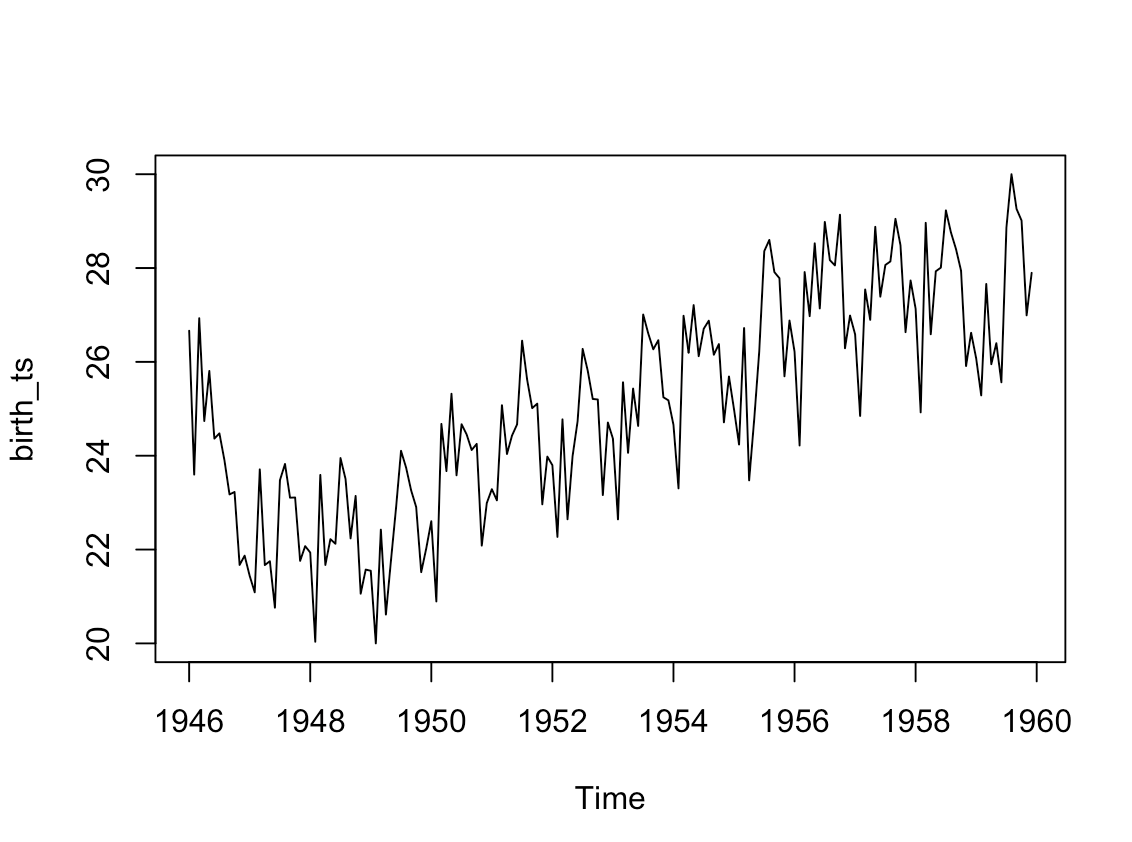
We found that some time series data that may have been collected at regular intervals is less than a year, which was true in our case.

For that, we used the **‘frequency’** parameter in **ts()** function to specify the number of times the data was collected in a year. For monthly time series data, we set it to **12.**

We also used the **‘start’** parameter in the function to specify the first year the data was collected, and its respective first interval.

The time series data was plotted using the plot.ts() function-

> plot.ts(birth\_ts)



**Figure 1: Time series plot for the number of child births in New York from 1946 to 1959.**

We observe a seasonal variation in the number of births per month. Also, it seemed that the data could have been described using an additive model, as we observed constant sized seasonal fluctuations over time with no dependence on the level of time series, and with roughly constant random fluctuations in size over time.

Decomposing a time series involves separating into the constituent components, which are the trend component, seasonal component, and an irregular component, considering a seasonal time series.

We used the ‘decompose function to estimate all the 3 components of the seasonal time series described using an additive model.

> birth\_comp <- decompose(birth\_ts)

> birth\_comp$seasonal

**## Jan Feb Mar Apr May Jun**

**## 1946 -0.6771947 -2.0829607 0.8625232 -0.8016787 0.2516514 -0.1532556**

**## 1947 -0.6771947 -2.0829607 0.8625232 -0.8016787 0.2516514 -0.1532556**

**## 1948 -0.6771947 -2.0829607 0.8625232 -0.8016787 0.2516514 -0.1532556**

**## 1949 -0.6771947 -2.0829607 0.8625232 -0.8016787 0.2516514 -0.1532556**

**## 1950 -0.6771947 -2.0829607 0.8625232 -0.8016787 0.2516514 -0.1532556**

**## 1951 -0.6771947 -2.0829607 0.8625232 -0.8016787 0.2516514 -0.1532556**

**. . . . . .**

**. . . . . .**

**. . . . . .**

**## 1959 -0.6771947 -2.0829607 0.8625232 -0.8016787 0.2516514 -0.1532556**

The estimated seasonal factors given for January-December are same for every year, having largest seasonal factor for July with a value around **1.4** and lowest for February with value around **-2.**

> birth\_comp$trend

Jan Feb Mar Apr May Jun Jul

**1946 NA NA NA NA NA NA 23.98433**

**1947 22.35350 22.30871 22.30258 22.29479 22.29354 22.30562 22.33483**

**1948 22.43038 22.43667 22.38721 22.35242 22.32458 22.27458 22.23754**

**1949 22.06375 22.08033 22.13317 22.16604 22.17542 22.21342 22.27625**

**. . . . . . . .**

**1959 26.96858 27.00512 27.09250 27.17263 27.26208 27.36033 NA**

**Aug Sep Oct Nov Dec**

**. . . . .**

**. . . . .**

**. . . . .**

> birth\_comp$random

**## Jan Feb Mar Apr May**

**## 1946 NA NA NA NA NA**

**## 1947 -0.237305288 0.863252404 0.543893429 0.175887019 -0.793193109**

**## 1948 0.183819712 -0.318705929 0.340268429 0.121262019 -0.354234776**

**## 1949 0.161444712 0.002627404 -0.571689904 -0.749362981 -0.666068109**

**## 1950 0.064569712 -0.292705929 0.479560096 1.047887019 1.561973558**

**. . . . . . . .**

**. . . . . . . .**

**## 1959 -0.215388622 0.363835737 -0.295023237 -0.419946314 -1.115734776**

**##**

**## Jun Jul Aug Sep Oct**

**## 1946 NA -0.963379006 -0.925718750 -0.939949519 -0.709369391**

**## 1947 -1.391369391 -0.311879006 0.347739583 0.150592147 0.076797276**

**## 1948 0.001672276 0.256412660 0.119531250 -0.623449519 0.289547276**

**. . . . . . . .**

**. . . . . . . .**

**. . . . . . . .**

**## 1958 0.581005609 0.282204327 0.132572917 0.290758814 -0.171994391**

**## 1959 -1.642077724 NA NA NA NA**

**Nov Dec**

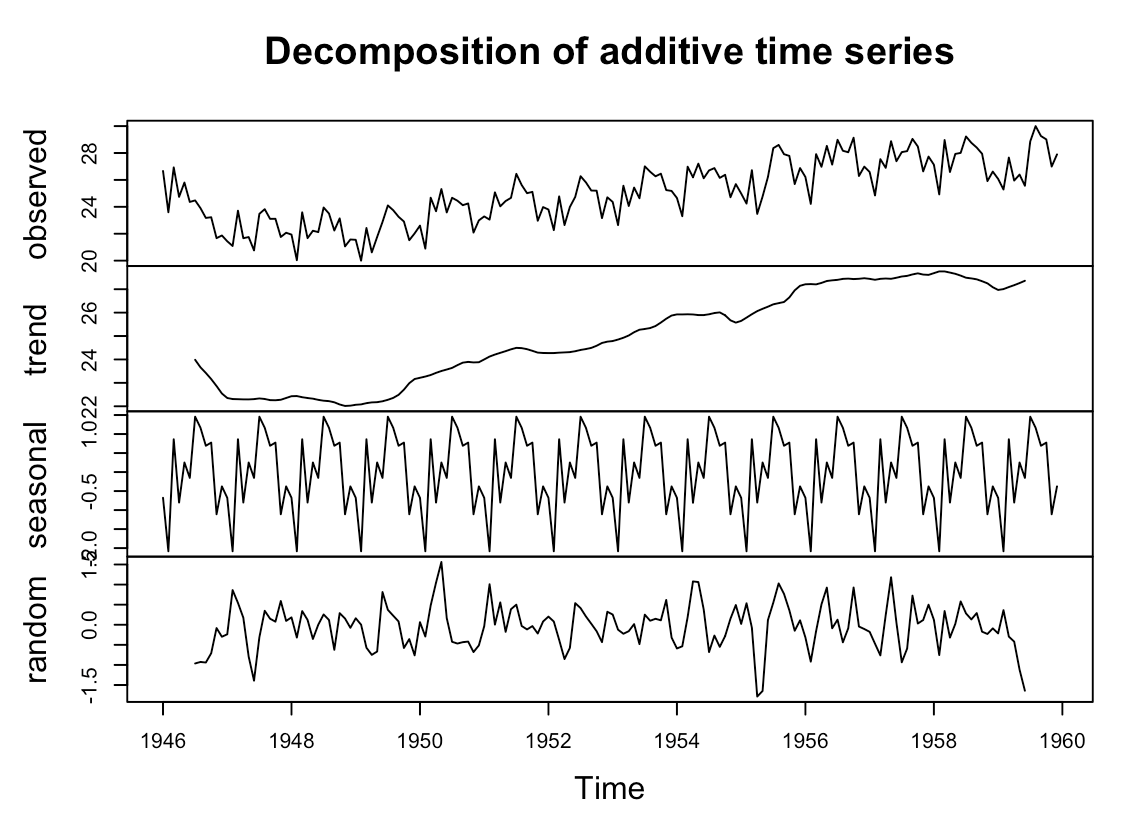
**1946 -0.082484776 -0.298388622**

**. . .**

**. . .**

**. . .**

We ploted the estimated trend, seasonal, and irregular components of the time series using the plot() function –



**Figure 2: plot indicating components of time series**

We found a small decrement from about 24 in 1947 to 22 in 1948 along with the steep increase from then on to about 27 in 1959 for the trend component estimated by the plot() function.

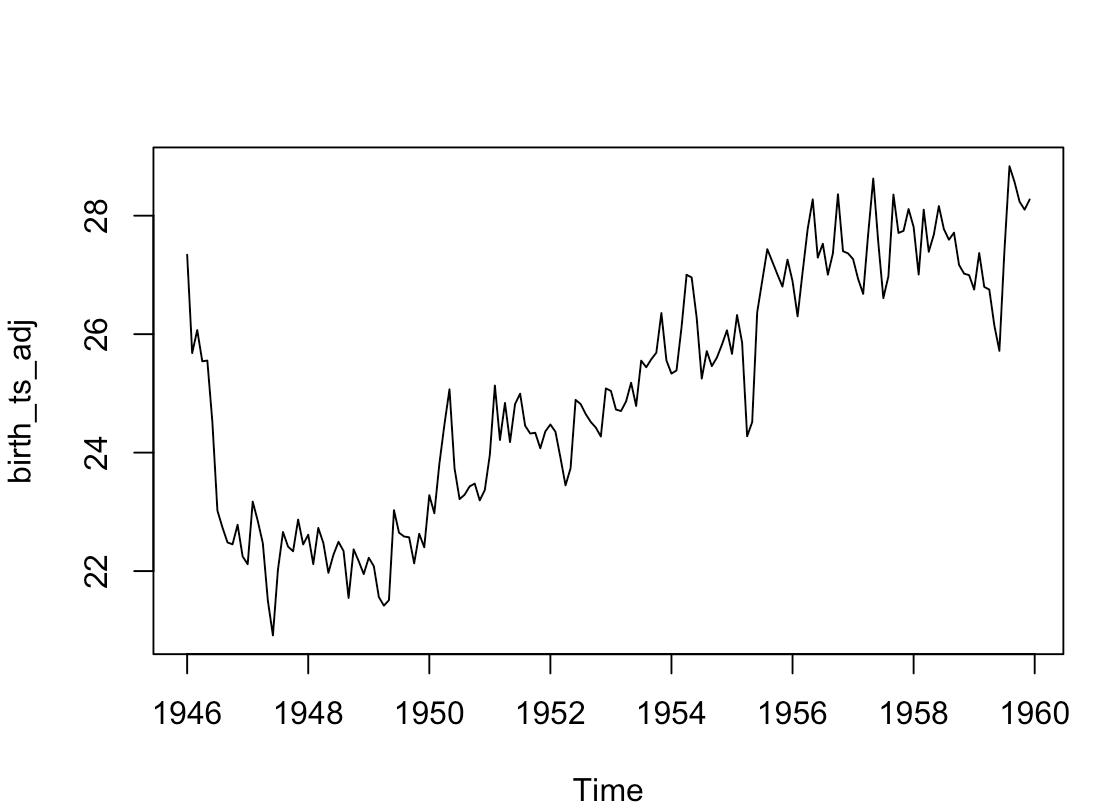
In order to achieve seasonal adjusting, it was required to estimate the seasonal component using the decompose() function, and subtract the estimated seasonal component from the main time series variable ‘birth\_comp’.

> plot(birth\_comp)

> birth\_comp<- decompose(birth\_ts)

> birth\_ts\_adj <- birth\_ts - birth\_comp$seasonal

> plot(birth\_ts\_adj)



**Figure 3: Seasonal adjusting of time series data using subtraction of seasonal component from then original time series data**

We observed the removal of seasonal variation from the seasonally adjusted time series, which has led to the presence of trend and irregular component in the seasonally adjusted time series.

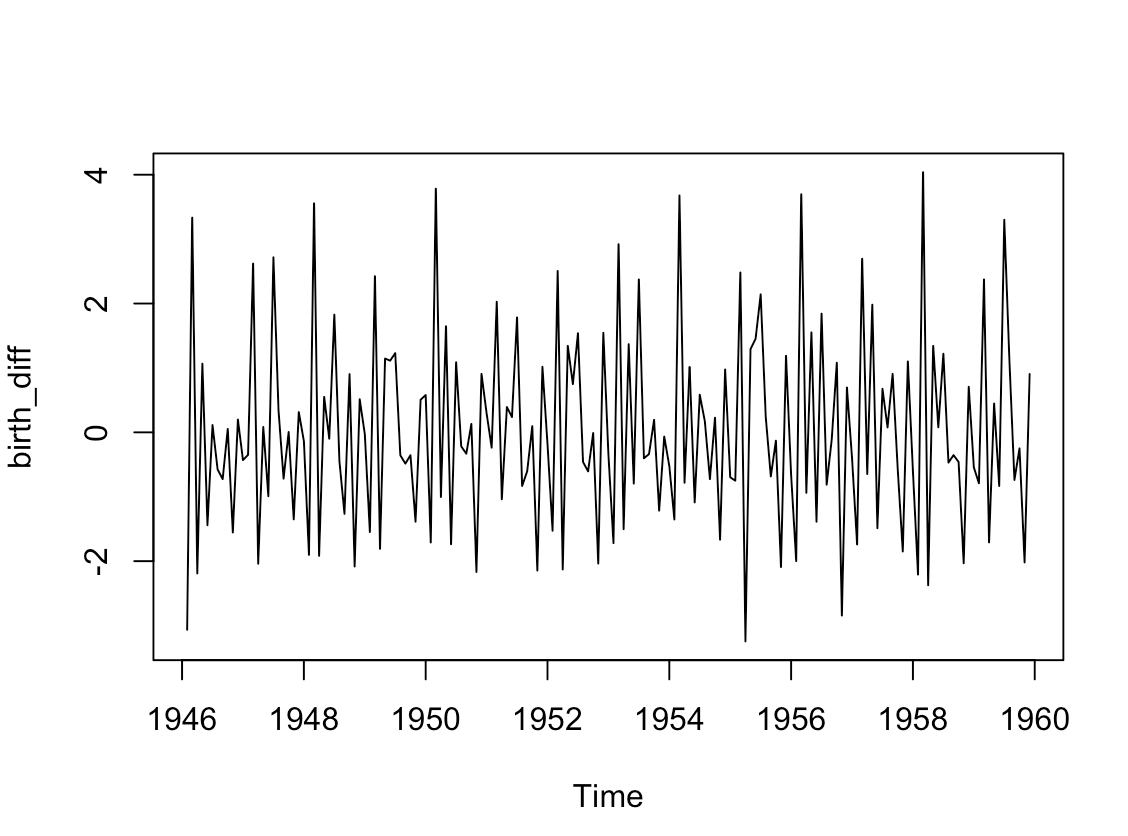
It is observed that we can make better predictive models by taking correlations in data into account.

**Part B – Autocorrelation using ARIMA**

We needed to find the difference of time series using diff() function to obtain a stationary time series as ARIMA models are defined for stationary time series.

> birth\_diff <- diff(birth\_ts, differences = 1)

> plot.ts(birth\_diff)



**Figure 4: Plot for time series data of the first differences**

As the time series of first differences looked stationary in mean and variance, so the ARIMA(p,1,q) model seems appropriate for the dataset.

By this, we removed the trend component of the time series of the dataset and examined for the corrrelations between successive terms of this component, for making a predictive model.

Now, we needed to select the appropriate ARIMA model meaning to find the most appropriate. Values of ‘p’ and ‘q’ for the model.

We plotted a **correlogram** **and partial correlogram** using the **acf()** and **pacf()** functions in R.

acf(birth\_diff, lag.max=20)

acf(birth\_diff, lag.max=20, plot = FALSE)

**## Autocorrelations of series ‘birth\_diff’, by lag**

**##**

**## 0.0000 0.0833 0.1667 0.2500 0.3333 0.4167 0.5000 0.5833 0.6667 0.7500**

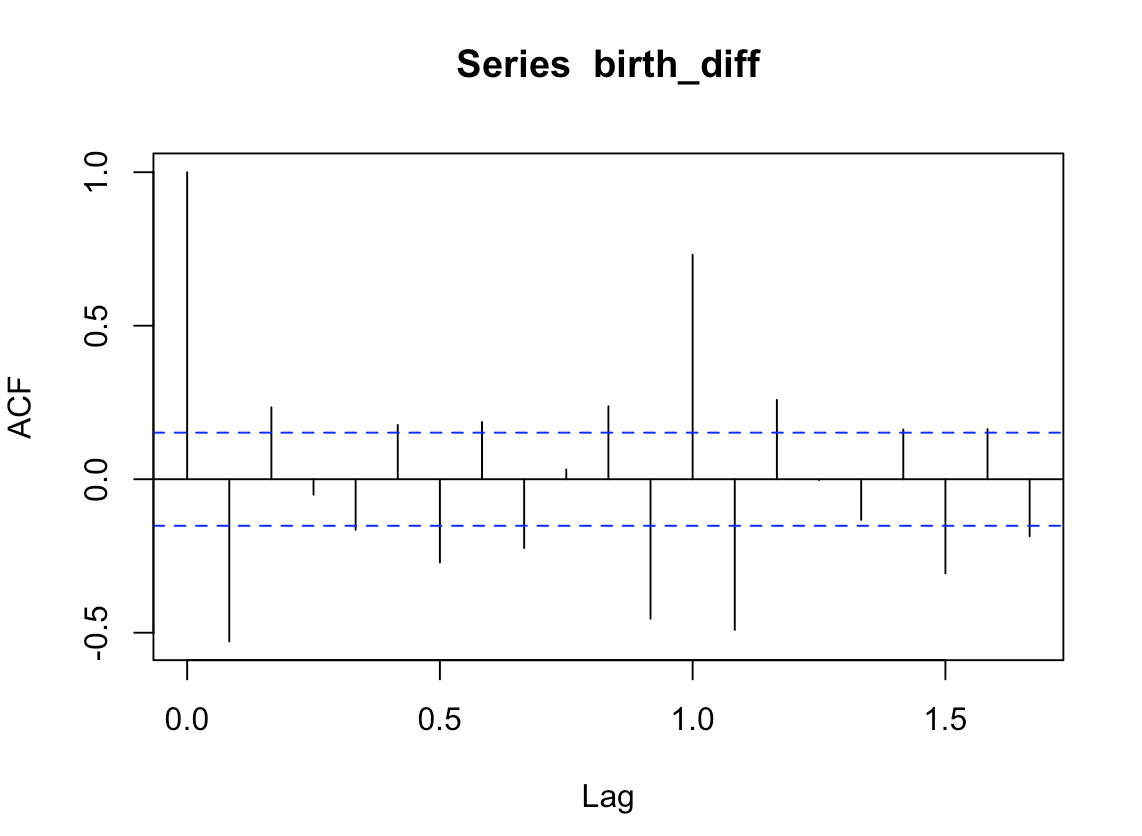
**## 1.000 -0.528 0.234 -0.050 -0.164 0.177 -0.271 0.186 -0.224 0.032**

**## 0.8333 0.9167 1.0000 1.0833 1.1667 1.2500 1.3333 1.4167 1.5000 1.5833**

**## 0.238 -0.454 0.731 -0.491 0.258 -0.003 -0.132 0.162 -0.307 0.163**

**## 1.6667**

**## -0.185**



**Figure 5: Correlogram for lags 1-20 of time series**

We found the autocorrelation at lags 0.1, 0.4, 0.5, 0.7, 0.9, 1.1 and 1.5 exceeding significance bounds.

pacf(birth\_diff, lag.max = 20)

pacf(birth\_diff, lag.max = 20, plot = FALSE)

**## Partial autocorrelations of series ‘birth\_diff’, by lag**

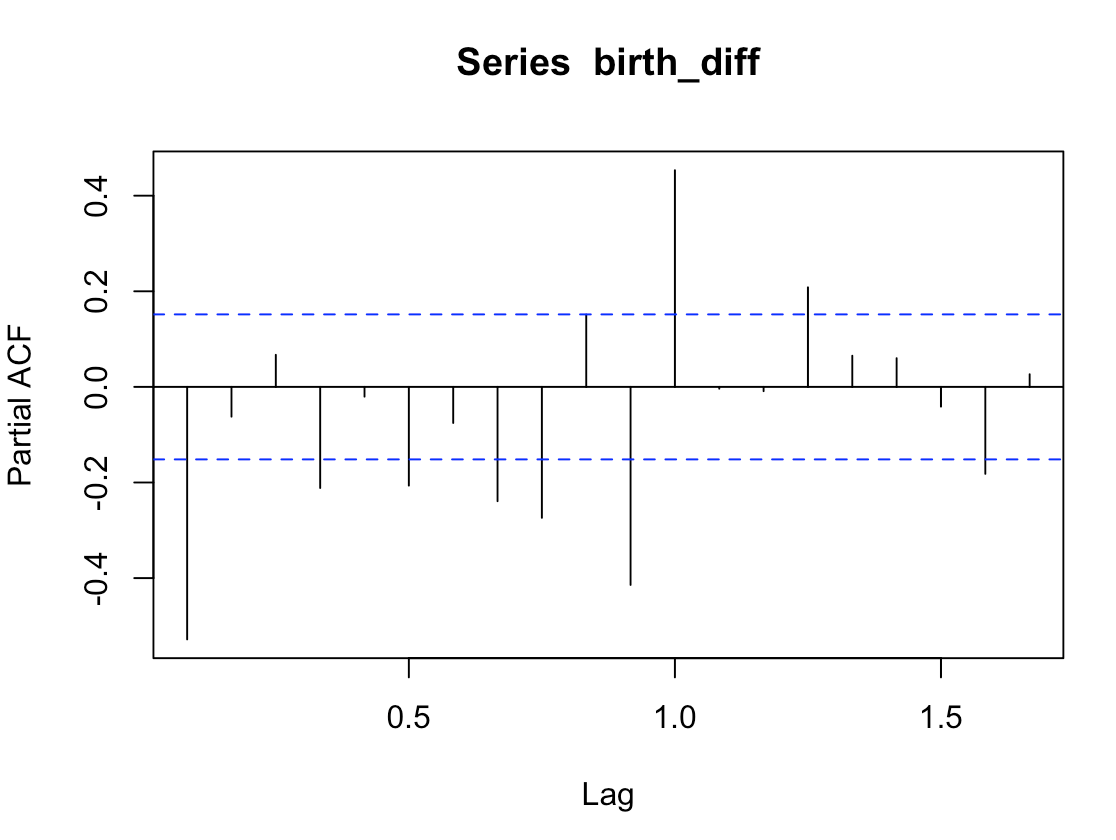
**##**

**## 0.0833 0.1667 0.2500 0.3333 0.4167 0.5000 0.5833 0.6667 0.7500 0.8333**

**## -0.528 -0.062 0.067 -0.211 -0.020 -0.207 -0.075 -0.239 -0.274 0.151**

**## 0.9167 1.0000 1.0833 1.1667 1.2500 1.3333 1.4167 1.5000 1.5833 1.6667**

**## -0.414 0.453 -0.004 -0.009 0.208 0.065 0.060 -0.041 -0.182 0.026**

****

**Figure 6: Plot for pacf() on time series data**

We found partial autocorrelation at lag 1 positive and exceeded the significance bounds, while the partial autocorrelation at several lags negative.

We used the **auto.arima()** function of the **‘forecast’** package to find an appropriate arima model.

library(“forecast”)

auto.arima(birth\_diff, ic = "bic")

**## Series: birth\_diff**

**## ARIMA(2,0,2)(1,1,1)[12]**

**##**

**## Coefficients:**

**## ar1 ar2 ma1 ma2 sar1 sma1**

**## 0.6539 -0.4541 -0.7256 0.2533 -0.2427 -0.8451**

**## s.e. 0.3004 0.2429 0.3228 0.2879 0.0985 0.0995**

**##**

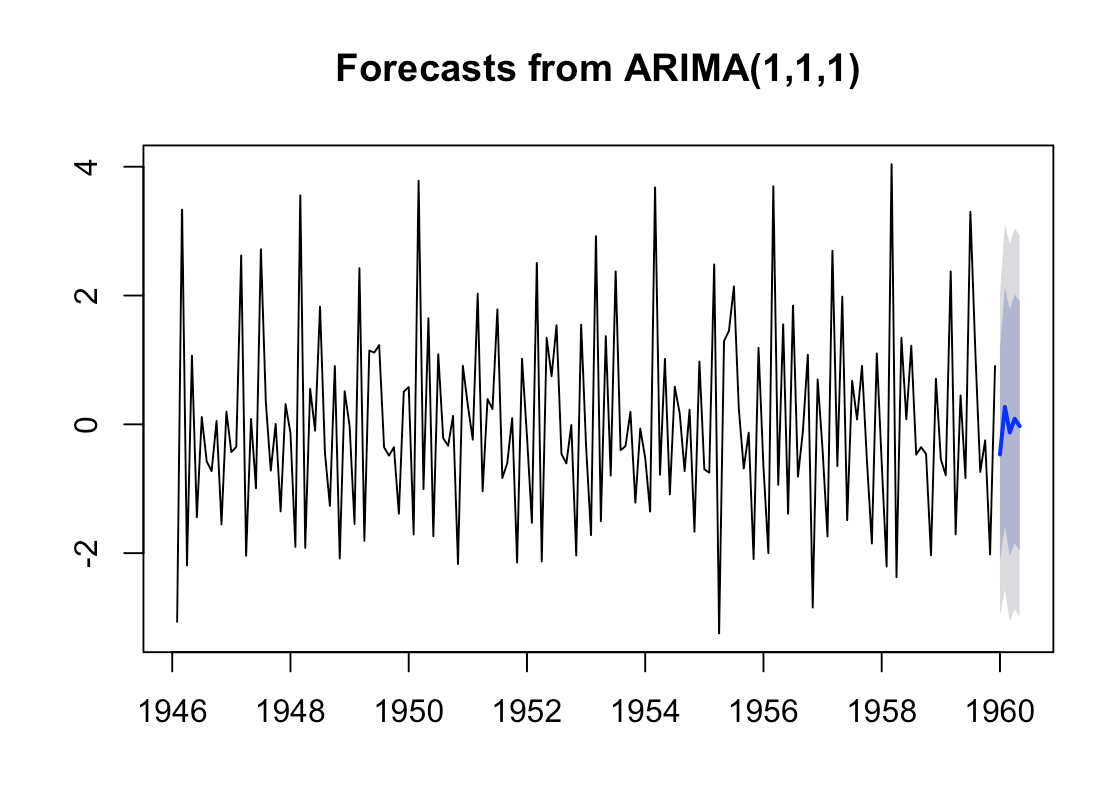
**## sigma^2 estimated as 0.4076: log likelihood=-157.46**

**## AIC=328.91 AICc=329.68 BIC=350.22**

Finally, we used the model to make forecasts for future values using forecast() function.

birth\_forecast <- forecast(birth\_arima, h = 5)

plot(birth\_forecast)



**Figure 7: Plot for forecast for next trend in child births for the dataset**

Conclusion

After successful completion of the project this week, we were successfully able to use the

time series data of the number of births per month in New York city, , from January 1946 to December 1959, decompose the seasonal time series and seasonally adjust

to subtract the seasonal components from the time series using **plot.ts()**, **decompose()** and **diff()** functions.

We successfully addressed autocorrelation issues and made a better predictive model using **acf()**, **pacf()**, **auto.arima()**, and **forecast()** functions.

We ended up forecasting the future trend for child births in New York **after 1959** successfully. This all has provided meaningful insights from data using the Time Series Analysis in R.

**References**

1. Quick-R: Time Series - statmethods.net. (n.d.). Retrieved from <https://www.statmethods.net/advstats/timeseries.html>
2. Arora, A. (2018, December 29). Step By Step Guide To Time Series Analysis In R. Retrieved from https://stepupanalytics.com/step-by-step-guide-to-time-series-analysis-in-r/
3. A Little Book of R For Time Series. (n.d.). Retrieved from https://buildmedia.readthedocs.org/media/pdf/a-little-book-of-r-for-time-series/latest/a-little-book-of-r-for-time-series.pdf
4. Time series data in R. (2014, January 28). Retrieved from https://robjhyndman.com/hyndsight/time-series-data-in-r/