Time Series Analysis in R

Last Updated : 17 May, 2025

Time series analysis is a statistical technique used to understand how data points evolve over time. In R programming, time series analysis can be efficiently performed using the `ts()` function, which helps organize data with associated time stamps. This method is widely applied in business and research to analyze trends and make forecasts, such as sales trends, inventory management, stock market prices, population growth, and more.

Creating Time Series Data in R

The core function to create time series objects in R is:

```
objectName <- ts(data, start, end, frequency)
```

Parameters:

- **data**: A numeric vector containing the observed values.
- **start**: The time of the first observation.
- **end**: The time of the last observation (optional).
- **frequency**: The number of observations per unit time (e.g., 12 for monthly data, 4 for quarterly).

Note: *For more detailed information about the `ts()` function, you can run `help("ts")` in the R console.*

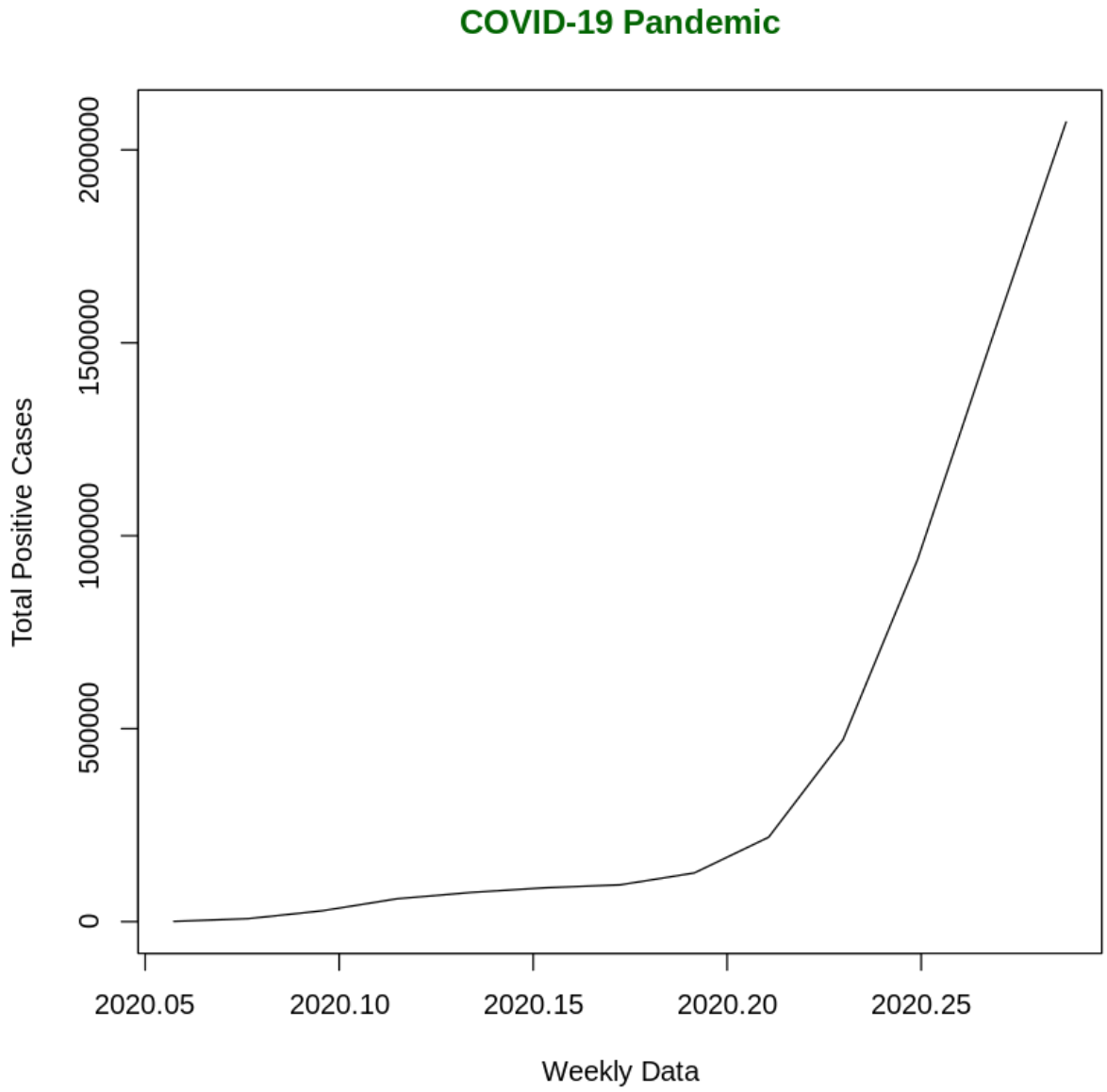
Example: COVID-19 Weekly Cases Time Series

Consider weekly COVID-19 positive cases from January 22, 2020, to April 15, 2020. The following R code creates a time series object and

plots the data: We define a vector `x` containing weekly COVID-19 positive cases. Then load the `lubridate` library to help work with date formats and ceate a time series object `mts` starting from January 22, 2020, with weekly frequency. Then we plot the time series data with appropriate axis labels and a title.

```
install.packages("lubridate")
library(lubridate)

x <- c(580, 7813, 28266, 59287, 75700,
       87820, 95314, 126214, 218843, 471497,
       936851, 1508725, 2072113)

mts <- ts(x, start = decimal_date(ymd("2020-01-22")), frequency = 365.25 /
7)

plot(mts, xlab = "Weekly Data", ylab = "Total Positive Cases",
     main = "COVID-19 Pandemic", col.main = "darkgreen")
```

Output:



Weekly Cases Time Series

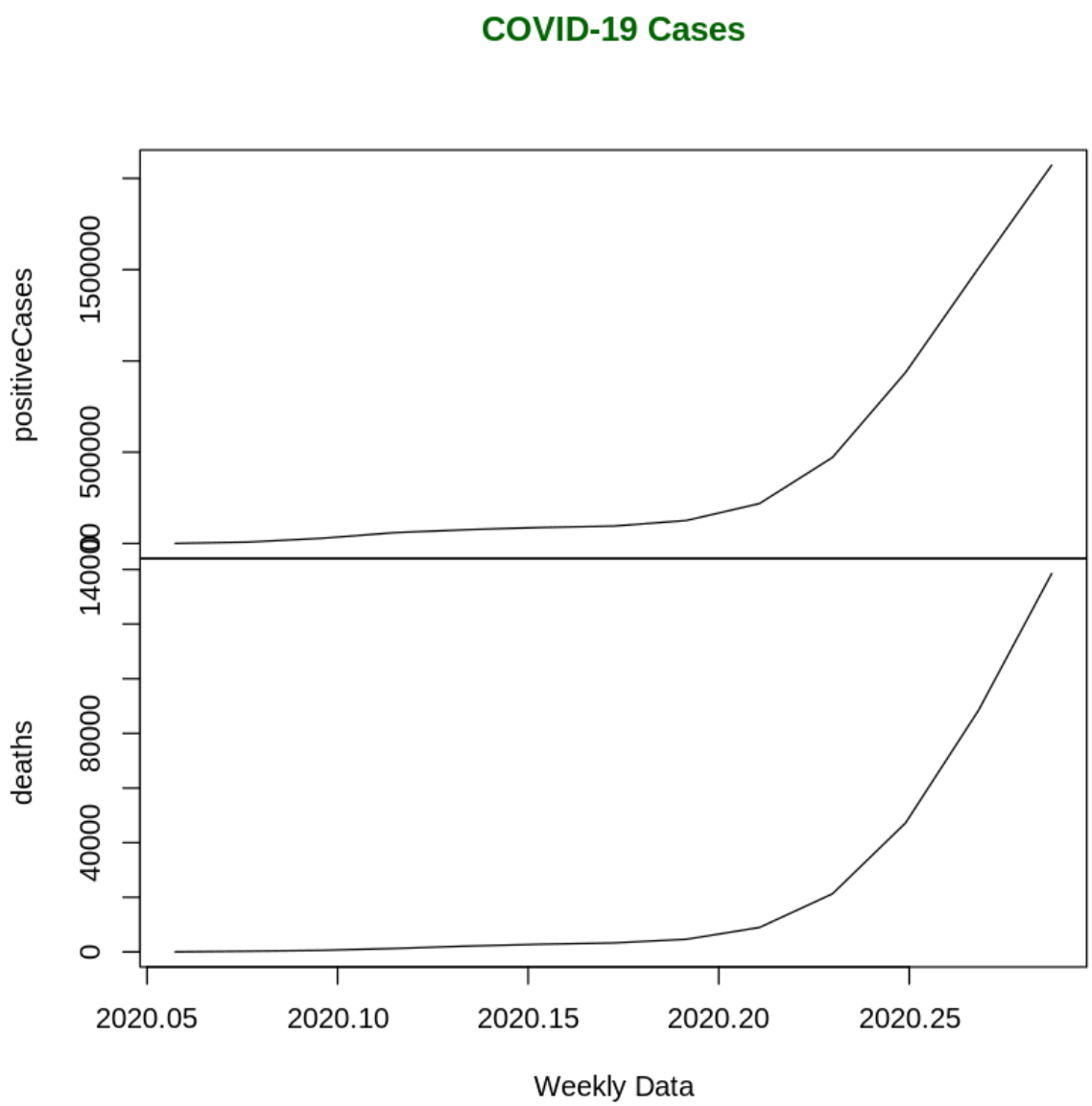
Multivariate Time Series Analysis

Multivariate time series analysis allows simultaneous visualization of multiple related time series. For example, tracking both total COVID-19 positive cases and deaths weekly , we define two numeric vectors: `positiveCases` and `deaths` representing weekly COVID-19 data. Then we load `lubridate` for date handling and combine both vectors into a multivariate time series object `mts`. Then we plot the multivariate time series with labels and title.

```
library(lubridate)

positiveCases <- c(580, 7813, 28266, 59287,
```

```
                  75700, 87820, 95314, 126214,
                  218843, 471497, 936851,
                  1508725, 2072113)

deaths <- c(17, 270, 565, 1261, 2126, 2800,
            3285, 4628, 8951, 21283, 47210,
            88480, 138475)

mts <- ts(cbind(positiveCases, deaths),
          start = decimal_date(ymd("2020-01-22")),
          frequency = 365.25 / 7)

plot(mts, xlab = "Weekly Data", main = "COVID-19 Cases", col.main =
"darkgreen")
```

Output:



Multivariate Time Series Analysis

Time Series Forecasting in R

Forecasting future values of a time series is an essential aspect of analysis. The `forecast` package in R provides tools to build predictive models. Here, we use the automated ARIMA model to forecast future COVID-19 cases . Here we load `lubridate` for date processing and `forecast` for forecasting functions. Then we create a time series object `mts` with weekly frequency starting January 22, 2020. Then we fit an ARIMA model automatically using `auto.arima()` and generate forecasts for the next 5 weeks. Then we plot both the historical and forecasted data.

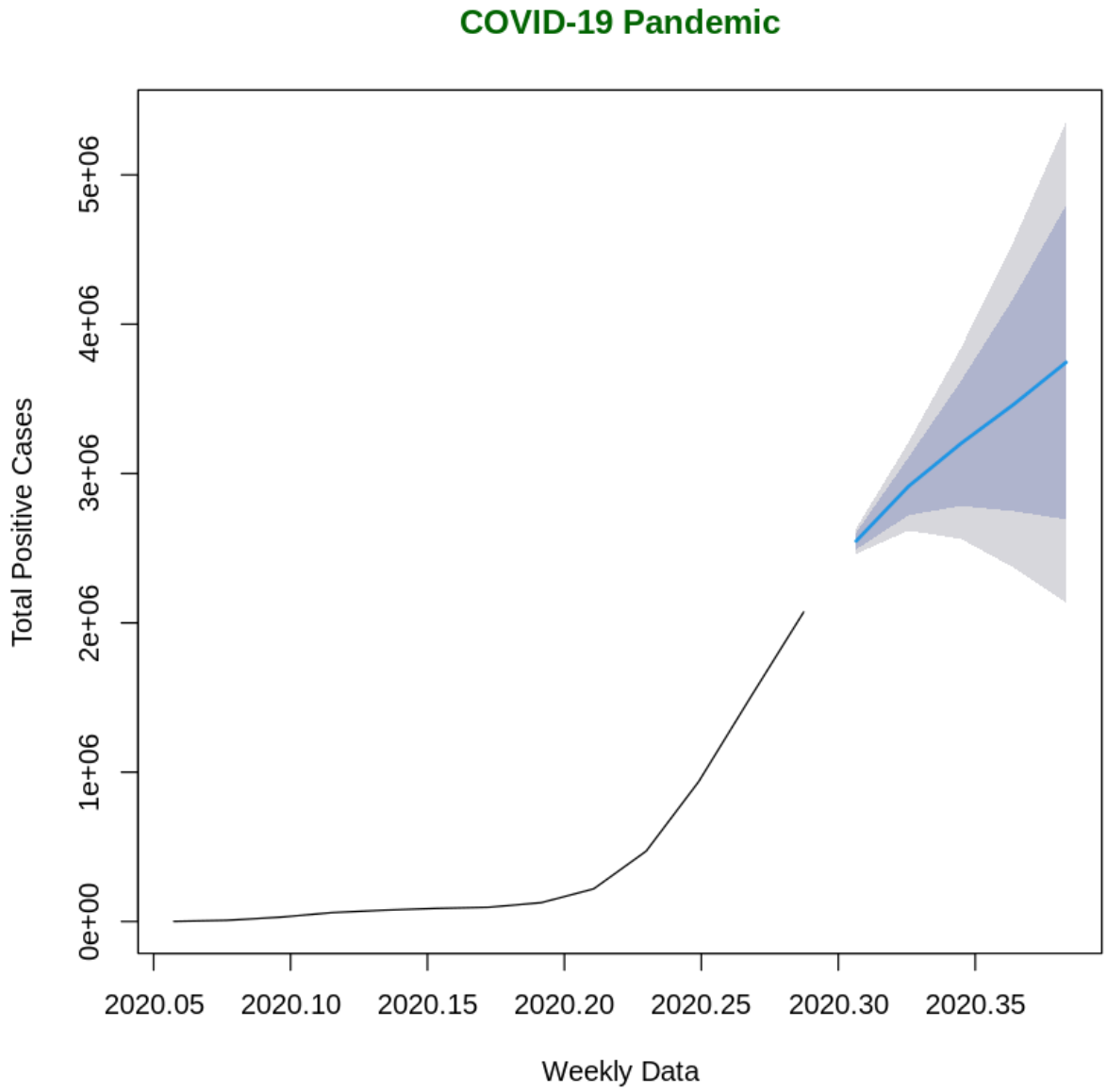
```
install.packages("forecast")
library(lubridate)
library(forecast)

x <- c(580, 7813, 28266, 59287, 75700,
       87820, 95314, 126214, 218843,
       471497, 936851, 1508725, 2072113)


mts <- ts(x, start = decimal_date(ymd("2020-01-22")), frequency = 365.25 /
7)

fit <- auto.arima(mts)

forecasted_values <- forecast(fit, 5)

print(forecasted_values)

plot(forecasted_values, xlab = "Weekly Data", ylab = "Total Positive Cases",
     main = "COVID-19 Pandemic", col.main = "darkgreen")
```

Output:

	Point Forecast	Lo 80	Hi 80	Lo 95	Hi 95
2020.307	2547989	2491957	2604020	2462296	2633682
2020.326	2915129	2721276	3108982	2618657	3211602
2020.345	3202354	2783401	3621306	2561621	3843086
2020.364	3462691	2748533	4176849	2370481	4554902
2020.383	3745055	2692886	4797223	2135901	5354208

Forecasted values

Plotting the result



Forecasted Plot

In this article, we explored how to perform time series analysis in R, including creating univariate and multivariate time series, visualizing data, and applying forecasting models using ARIMA. These techniques provide valuable insights and predictions for time-dependent data across various fields.