How can I plot multiple lines using ggplot2 in R? Can you provide an example of how to do this?”

Authored by

stats writer

June 26, 2024

GGplot2 is a powerful and widely used data visualization package in R that allows users to create visually appealing and informative graphs. One of its key features is the ability to plot multiple lines on a single graph, making it ideal for comparing trends or patterns across different groups or categories. To do this, users can use the "geom_line" function to specify the data and variables to be plotted, along with any desired aesthetic elements. An example of this can be seen by creating a line graph of average temperature over time for three different cities. By specifying the city variable as the color aesthetic, the resulting graph will display three lines, each representing a different city. This allows for easy comparison and identification of any differences or similarities in temperature trends between the cities.

Plot Multiple Lines in ggplot2 (With Example)

You can use the following basic syntax to plot multiple lines in ggplot2:

```
ggplot(df, aes(x=x_var, y=y_var)) +
geom_line(aes(color=group_var)) +
scale_color_manual(name='legend_title', labels=c('lab1', 'lab2', 'lab3'),
values=c('color1', 'color2', 'color3'))
```

This particular syntax creates a plot in ggplot2 with three lines.

This syntax assumes that your data frame is in a .

The following example shows how to plot multiple lines in ggplot2 in practice.

Example: Plot Multiple Lines in ggplot2

Suppose we have the following data frame in R that contains information on the number of sales made at three different stores on five different days:

```
#create data frame
df <- data.frame(day=c(1, 2, 3, 4, 5),
storeA=c(5, 6, 8, 8, 9),
storeB=c(3, 3, 4, 5, 7),
storeC=c(8, 10, 12, 12, 17))

#view data frame
df
```

```
day storeA storeB storeC
1 1 5 3 8
2 2 6 3 10
3 3 8 4 12
4 4 8 5 12
5 5 9 7 17
```

This data frame is currently in a wide format.

However, we can use the pivot_longer() function from

the tidyr package to quickly convert the data into a long format:

```
library(tidyr)

#convert data from wide to long format
df <- df %>% pivot_longer(cols=c('storeA', 'storeB', 'storeC'),
names_to='store',
values_to='sales')

#view updated data frame
df

# A tibble: 15 x 3
day store sales

1 1 storeA 5
2 1 storeB 3
3 1 storeC 8
4 2 storeA 6
5 2 storeB 3
6 2 storeC 10
7 3 storeA 8
8 3 storeB 4
```

9 3 storeC 12

10 4 storeA 8

11 4 storeB 5

12 4 storeC 12

13 5 storeA 9

14 5 storeB 7

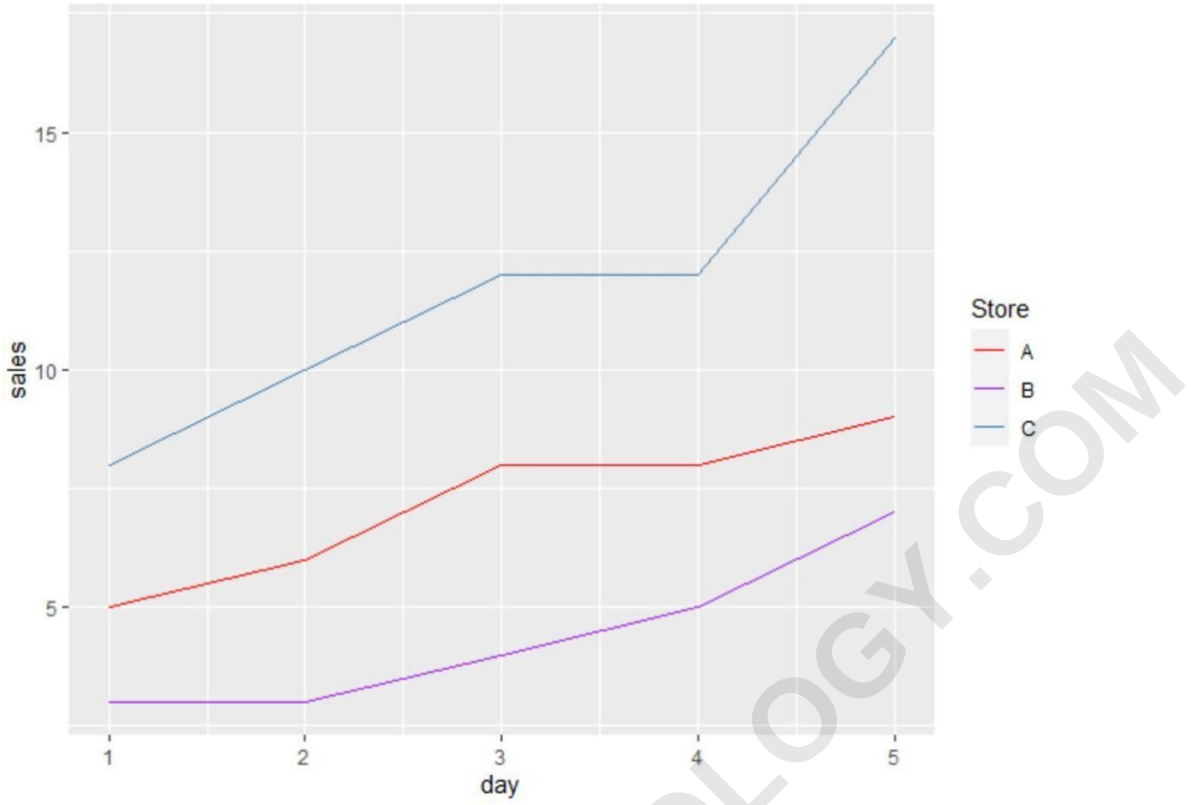
15 5 storeC 17

Related:

Now that the data frame is in a long format, we can use the following syntax with ggplot2 to plot the sales by each store:

```
library(ggplot2)

#plot sales by store
ggplot(df, aes(x=day, y=sales)) +
geom_line(aes(color=store)) +
scale_color_manual(name='Store', labels=c('A', 'B', 'C'),
values=c('red', 'purple', 'steelblue'))
```



The individual lines display the sales made at each store on each day.

Note that we used the scale_color_manual() function to create a custom legend on the right side of the plot to make the lines easier to interpret.

The following tutorials explain how to perform other common tasks in ggplot2: