

How to Add a Title to Your Plots in Base R

Authored by
stats writer

November 20, 2025

RECOMMENDED CITATION

stats writer (2025). *How to Add a Title to Your Plots in Base R*. PSYCHOLOGICAL SCALES.
Retrieved from <https://scales.arabpsychology.com/?p=98424>

Effective data visualization requires clear, informative labeling. In the Base R graphics system, adding a title to your visual outputs is a fundamental step toward ensuring your audience understands the context of the displayed data. While there are multiple ways to achieve this, the simplest initial method involves utilizing the `main` argument directly within the `plot()` function itself.

When you call the `plot()` function to generate a graph--such as a histogram, boxplot, or scatterplot--you can pass a value to the `main` parameter. This parameter accepts a character string, which is then rendered as the primary title centered above the plotting area. For example, the command `plot(x, y, main="My Plot Title")` will instantly add the title "My Plot Title" to the resulting visualization. While convenient for quick labeling, the dedicated `title()` function provides much greater control for complex aesthetic adjustments.

Harnessing the Dedicated `title()` Function

For users seeking more control over the appearance and placement of the title, the standalone `title()` function is the superior choice in Base R. Unlike the `main` argument in `plot()`, the `title()` function is typically executed immediately after the plot has been generated, allowing it to modify the existing graphic object. This separation of plotting and titling commands enhances code readability and flexibility, especially when generating complex figures or using loops.

The primary advantage of using `title()` lies in its rich set of customization arguments, which allow you to manipulate color, size, font style, and precise location. This level of detail is often critical for preparing publication-quality figures or visualizations intended for formal reports. By mastering this function, you ensure your plots are not just accurate but also aesthetically polished and compliant with specific formatting standards.

This function uses the following basic syntax structure:

```
#create scatterplot of x vs. y  
plot(df$x, df$y)
```

```
#add title  
title('This is my title')
```

Essential Customization Arguments for Titles

While the simplest execution of `title()` only requires the title character string, its power is unlocked when utilizing its optional arguments. These parameters provide detailed control over almost every visual aspect of the title text, moving beyond the default black, centered, single-size

display.

Understanding these arguments is key to creating highly customized visualizations. Here are the core arguments you can pass within the `title()` function to customize the appearance of the title:

col.main: This argument dictates the color of the title text. It accepts color names (e.g., 'red', 'blue') or hexadecimal color codes (e.g., '#FF0000').

cex.main: This is the character expansion factor for the title. It is a numerical value that specifies the size of the title relative to the default size (where 1 is the default, 2 is twice the default size, and 0.5 is half the default size).

font.main: This parameter controls the font style used for the title. Standard values correspond to: 1 = plain text, 2 = **bold**, 3 = *italic*, and 4 = **bold italic**.

adj: This specifies the horizontal alignment or justification of the title. Values range from 0 (fully left-aligned) to 1 (fully right-aligned), with 0.5 being the default centered position.

line: This argument controls the vertical location of the title. It takes a numerical value corresponding to the number of margin lines above or below the plotting area. Positive values move the title further up the margin, while negative values move the title down, potentially into the plot area itself.

Step-by-Step Example: Initial Plot Creation

To illustrate how to use the `title()` function effectively, we will walk through a practical example using a simple dataset. First, we must generate a data frame and create a basic scatterplot without any initial titles, allowing us to demonstrate the incremental addition of labeling.

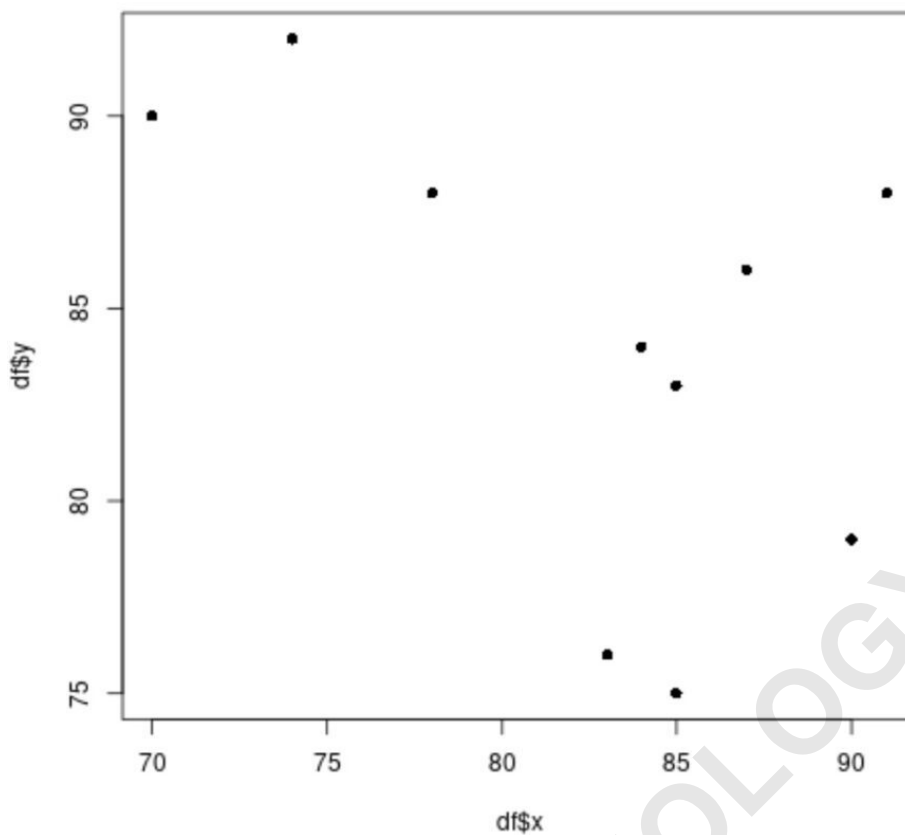
Suppose we define a small data frame, `df`, containing two variables, `x` and `y`, representing hypothetical measurements. We then use the plot() function in Base R to visualize the relationship between these two variables, utilizing `pch=16` to specify solid circular points.

#create data frame

```
df <- data.frame(x=c(70, 78, 90, 87, 84, 85, 91, 74, 83, 85),  
y=c(90, 88, 79, 86, 84, 83, 88, 92, 76, 75))
```

#create scatterplot of x vs. y

```
plot(df$x, df$y, pch=16)
```



As shown in the output above, by default, the **Base R** graphics device does not automatically add a title when only the coordinate data is provided to the `plot()` function. The plot is generated, but the context is only implicitly understood through the axis labels.

Implementing a Standard Title Using `title()`

Following the creation of the plot, we can now use the `title()` function to quickly label our graph. This is the simplest application of the function, providing only the text content. It is important to remember that `title()` must be executed immediately after the plotting command, as it operates on the active graphics window.

We rerun the plot setup code and append the `title()` command immediately afterward. This uses the default settings for font size, color (black), and position (centered on line 3 of the top margin).

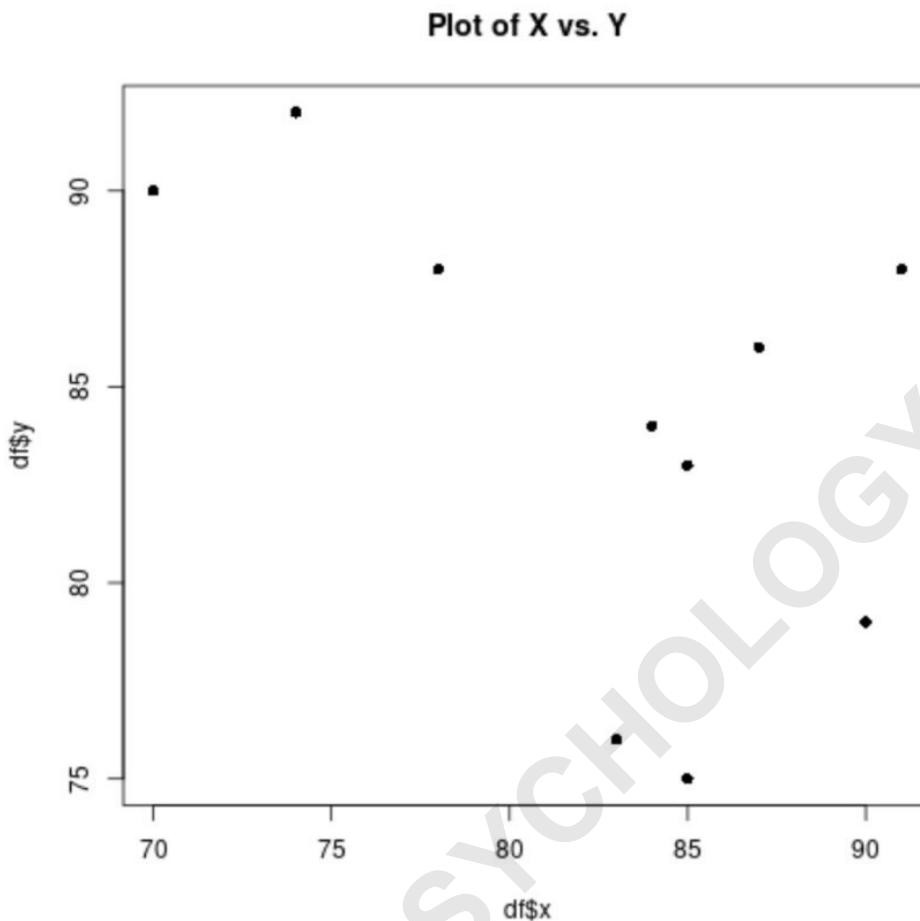
```
#create data frame
```

```
df <- data.frame(x=c(70, 78, 90, 87, 84, 85, 91, 74, 83, 85),  
y=c(90, 88, 79, 86, 84, 83, 88, 92, 76, 75))
```

```
#create scatterplot of x vs. y
```

```
plot(df$x, df$y, pch=16)
```

```
#add title  
title('Plot of X vs. Y')
```



Notice clearly that a descriptive title, "Plot of X vs. Y," has been successfully added to the top center of the scatterplot. This provides immediate context for the data visualization.

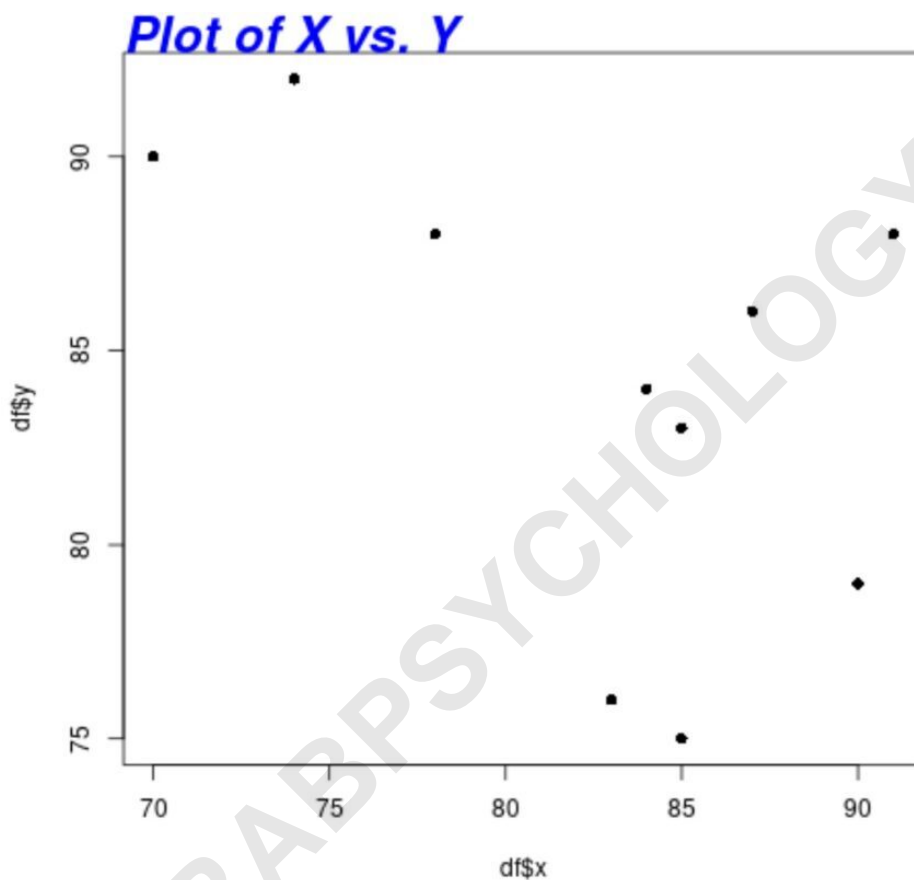
Advanced Styling and Positioning of the Title

To achieve a professional or institutionally compliant style, we often need to customize the title further. For example, we might need a larger, italicized title positioned closer to the plot area and rendered in a specific brand color. This requires leveraging the customization arguments introduced earlier within the `title()` function call.

We will customize the title appearance by setting the color to blue (`col.main`), doubling the size (`cex.main`), making the text bold italic (`font.main=4`), moving it to the far left (`adj=0`), and dropping it down to the edge of the plot border (`line=0`).

```
#create data frame
```

```
df <- data.frame(x=c(70, 78, 90, 87, 84, 85, 91, 74, 83, 85),  
y=c(90, 88, 79, 86, 84, 83, 88, 92, 76, 75))  
  
#create scatterplot of x vs. y  
plot(df$x, df$y, pch=16)  
  
#add title with custom appearance  
title('Plot of X vs. Y', col.main='blue', cex.main=2, font.main=4, adj=0, line=0)
```



Interpreting the Customization Parameters

The resulting plot demonstrates the powerful impact of combining multiple arguments within the `title()` function. Each parameter played a specific role in transforming the default title into the highly styled version seen above. Understanding the effect of each argument is essential for precise control over the final output.

Here is exactly what each argument achieved in the preceding example:

col.main: Successfully changed the title font color from the default black to a striking blue,

improving visual emphasis.

cex.main: Increased the title font size to two times the standard default size, making it much more prominent.

font.main: Changed the title font style to bold italic (value 4), enhancing its formal appearance.

adj: Moved the title entirely to the left margin (value 0), aligning it with the vertical axis label area rather than centering it.

line: By setting the line argument to 0, the title was moved vertically downward, positioning it right at the top boundary of the main plotting region.

Choosing Between `main` and `title()`

When initiating a plot, users frequently ask whether to use the `main` argument inside the `plot()` function or the standalone `title()` function. The choice generally depends on the complexity required for the title.

The `main` argument is ideal for rapid prototyping and generating plots where default title aesthetics are acceptable. It requires minimal code and is integrated directly into the plotting call. However, the `main` argument offers limited customization options; for example, changing the font size or color is cumbersome and requires modifying global graphical parameters or utilizing other functions.

In contrast, `title()` is the tool of choice for fine-tuning. It allows granular control over every aspect of the title without affecting other elements of the graphic. If your [data visualization](#) project demands specific colors, custom alignment, or precise spacing relative to the plot area, using the separate `title()` function is mandatory.

Conclusion: Mastering Plot Titles for Effective Communication

Adding a clear, aesthetically pleasing title is an indispensable step in producing high-quality charts using [Base R](#) graphics. Whether you opt for the quick simplicity of the `main` argument or the extensive customization offered by the `title()` function, mastering these techniques ensures your plots communicate their findings effectively and professionally.

We encourage users to experiment with the various arguments, particularly `adj` and `line`, to place the title exactly where required in the plot margins. Adjusting `cex.main` and `col.main` can significantly enhance the visual hierarchy of your graphical outputs, making your data more accessible and understandable to any audience.