

How to Fix “Error in plot.new() : figure margins too large

Authored by
stats writer

December 4, 2025

RECOMMENDED CITATION

stats writer (2025). *How to Fix “Error in plot.new() : figure margins too large.*

PSYCHOLOGICAL SCALES. Retrieved from <https://scales.arabpsychology.com/?p=104847>

When working with data visualization in **R**, encountering unexpected errors can halt your workflow. One common issue that often confuses beginners and even seasoned analysts is the error message generated during the setup phase of a new plot: **Error in plot.new() : figure margins too large**. This message signals a fundamental conflict between the space required by the graphical output and the space currently allocated to the plotting area. Effectively, the boundary settings--the margins--exceed the dimensions of the available drawing canvas.

This specific error is fundamentally about spatial allocation within the graphics environment, usually encountered when the output window, particularly the plot panel in **RStudio**, is constrained. To successfully render a visualization, the `plot.new()` function needs sufficient room for the plot itself, plus the required surrounding area dedicated to axes, titles, and labels. If the allocated margin space is disproportionately large relative to the overall panel size, **R** cannot initialize the drawing process, leading to the reported failure.

Fortunately, this issue is generally easy to diagnose and resolve. The solutions typically involve either physically increasing the size of the visualization container or programmatically reducing the required margin space using specific graphics parameters. This comprehensive guide details the reasons behind this error and provides three distinct, reliable methods for fixing the "figure margins too large" problem, ensuring your data visualizations render successfully every time.

The R Graphics System: Contextualizing plot.new()

Before diving into the fixes, it is crucial to understand the role of the **R** graphics system, particularly the initialization phase handled by the `plot.new()` function. In **R**, every time you call a high-level plotting function like `plot()`, `hist()`, or `boxplot()`, **R** first attempts to initialize a new graphical frame. This initialization involves calculating the exact dimensions needed for the axes, labels, and the central plotting region based on the current graphical parameters.

The core parameters defining this space are managed by the internal settings of the graphics device. These parameters include the dimensions of the plotting area itself (the region where data points are drawn) and the margins (the space reserved outside the plotting area for annotations). When **R** tries to calculate these dimensions during `plot.new()`, if the combined size of the internal plot area plus the mandated margins exceeds the physical confines of the current display device (e.g., the **RStudio** plot window), the system cannot proceed, and the error is thrown.

The most common cause is the default margin size being too restrictive for a tiny display window. **R** uses default margins that are generous enough to accommodate standard axis labels, tick marks, and titles. If the visual panel is minimized or severely constrained, these generous default settings become too large, triggering the error. Understanding this relationship between the required graphical footprint and the available viewport is key to troubleshooting the issue efficiently.

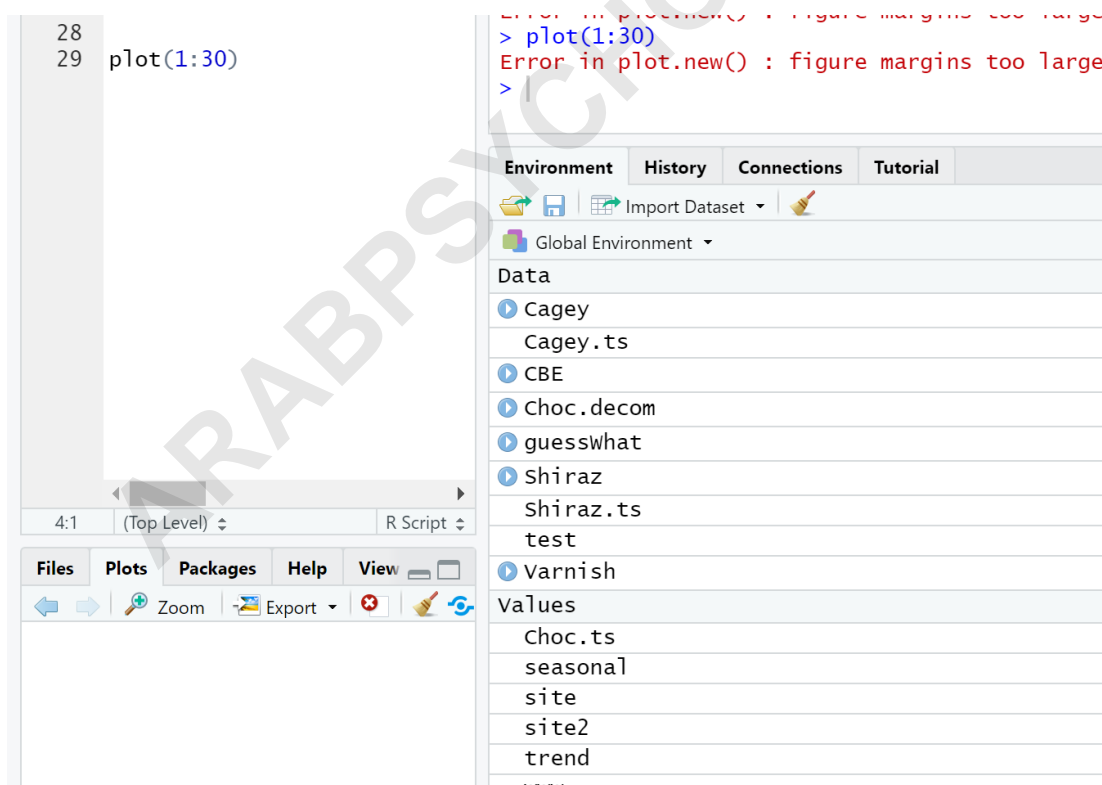
How the Figure Margins Error is Triggered (Reproduction)

To fully appreciate why this error occurs, it is helpful to reproduce it under controlled conditions. This situation most frequently arises when the dedicated plotting area in your integrated development environment (IDE) is made intentionally small, often minimized or placed in a narrow column. When the IDE's plot panel is severely reduced in size, R's automatic margin calculations fail.

Suppose we attempt to create a simple scatterplot using a standard dataset in R. Even the simplest command will fail if the display constraints are too tight. We start by executing the following code while ensuring the RStudio plot panel is significantly minimized:

```
#attempt to create scatterplot  
plot(1:30)
```

Immediately following the execution, because the plotting canvas is inadequate, R fails the initialization check and returns the following termination message, clearly indicating the spatial mismatch:



The screenshot shows the RStudio interface. The console on the left displays the following code and error message:

```
28  
29 plot(1:30)  
  
Error in plot.new() : figure margins too large  
> plot(1:30)  
Error in plot.new() : figure margins too large  
> |
```

The Environment pane on the right shows the following data and values:

Environment	History	Connections	Tutorial
Global Environment			
Data			
• Cagey			
Cagey.ts			
• CBE			
• Choc.decom			
• guesswhat			
• Shiraz			
Shiraz.ts			
test			
• Varnish			
Values			
Choc.ts			
seasonal			
site			
site2			
trend			
.....			

We receive the following error:

Error in plot.new() : figure margins too large

This error confirms that the current settings for margins (the padding around the actual data plot) cannot be accommodated within the small display area visible in the bottom left corner of the interface.

Method 1: Resolving the Error by Increasing the Size of the Plot Panel

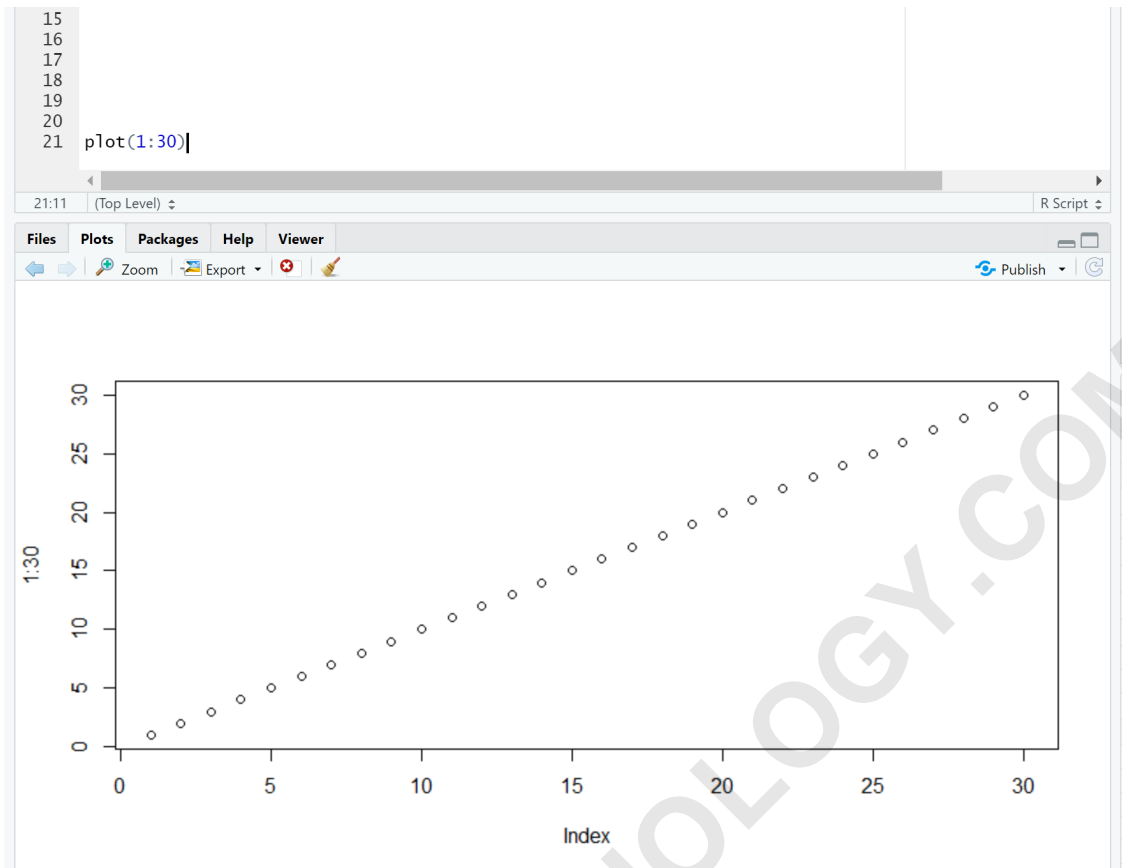
The most straightforward and immediate solution to this problem is purely spatial: simply increase the size of the window where R attempts to draw the plot. Since the error is triggered by a lack of physical space, providing more screen real estate resolves the constraint without requiring any changes to your code or graphic parameters.

In RStudio, this involves dragging the divider that separates the console and source panes from the environment and plot panes. By making the plotting pane larger, you grant R the necessary minimum dimensions to fit the default margins and the plotting region simultaneously. This is often the quickest fix, especially if you encountered the error accidentally after minimizing the plot panel.

Once the plot panel is sufficiently enlarged, re-running the exact same plotting command will succeed, as demonstrated below:

plot(1:30)

The result is a properly rendered plot, validating that the issue was merely a spatial boundary constraint, not a fundamental problem with the data or the function call itself.



As shown in the new visualization, we no longer receive the error because the graphical environment is now large enough to display the plot content along with the required margins, axes, and labels. This method is ideal when you require the default, well-spaced margins for clear visualization.

Method 2: Fine-Tuning Margins Using the `par()` Function

While resizing the panel is effective, it is not always feasible, especially when generating plots programmatically or when constrained by specific screen layouts. A more robust, programmatic solution involves explicitly reducing the default margin settings using the `par()` function. The `par()` function manages global graphics parameters, allowing fine-grained control over how plots are rendered.

The margin settings are primarily controlled by the `mar` parameter within `par()` function. The `mar` vector takes four numerical values, specifying the number of lines of margin space required on each side, ordered as: bottom, left, top, right (BLTR). The default settings in R are typically:

Bottom margin: **5.1 lines**

Left margin: **4.1 lines**

Top margin: **4.1 lines**

Right margin: **2.1 lines**

These default values ensure adequate space for axis labels and titles. However, if the plot panel is small, these five and four-line margins consume too much space. To fix the "margins too large" error without resizing the panel, we can dramatically reduce these values, for example, setting all margins to 1.

Understanding and Resetting Graphics Parameters: The `mar` and `mai` Options

When using the `par()` function, it is important to distinguish between the primary margin control options. While `mar` controls margins specified in terms of the number of line heights (the default unit), the `mai` parameter allows you to specify margins in absolute physical units, such as inches (the default unit for `mai`). Both parameters accept a vector of four values in the BLTR order.

Using `mar` is generally preferred for basic plotting adjustments because line heights scale relative to the plot's text size, maintaining visual consistency regardless of the device size. By reducing `mar` values--for instance, from the default `c(5.1, 4.1, 4.1, 2.1)` to `c(1, 1, 1, 1)`--we dramatically shrink the required surrounding space. Here is the implementation:

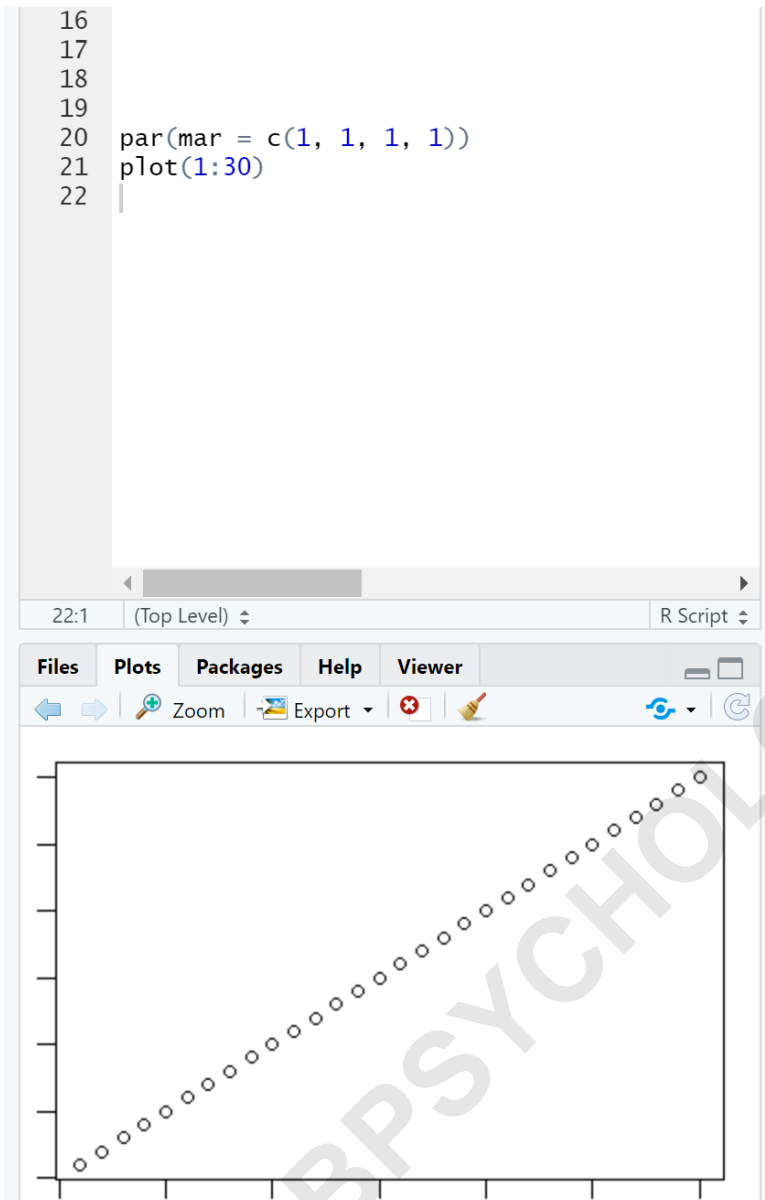
```
#adjust plot margins to 1 line on all sides
```

```
par(mar = c(1, 1, 1, 1))
```

```
#create scatterplot with reduced margins
```

```
plot(1:30)
```

Executing this code forces the plot to use minimal margins, effectively preventing the "figure margins too large" error, even if the plotting panel remains small. Note that reducing margins too severely might cut off axis labels or titles if you have customized them heavily, so always choose the smallest margins necessary to avoid the error while retaining readability.



As clearly demonstrated, the plot successfully initializes and displays in the constricted plotting panel in RStudio because we programmatically reduced the margin requirements well below the system defaults.

Method 3: Resetting the Graphics Device with `dev.off()`

In some rare but persistent cases, especially when working across multiple plots, graphical parameters may become corrupted or locked due to previous plotting operations. If resizing the panel and adjusting margins via `par()` function fail to resolve the error, the problem might stem from an actively maintained, but inaccessible, plotting device.

R manages graphical output through specialized devices (e.g., screen windows, PDF files, PNG

files). If a previous device session was opened but not properly closed, or if custom settings were applied that are now interfering with the current plot initialization, you must manually reset the system.

The ultimate solution for clearing residual device settings is to use the `dev.off()` function. This command effectively shuts down the currently active graphical device. If multiple devices are open, repeatedly calling `dev.off()` will close them sequentially until the graphical system is clean.

The syntax for shutting down the current plotting device is simple:

dev.off()

In many scenarios, executing `dev.off()` resolves the issue because it removes any potentially conflicting plot settings—including non-standard margin or size parameters—that were inherited from prior plotting attempts. After running this command, R resets the graphics environment to its default configuration, allowing subsequent calls to `plot.new()` to start afresh without being constrained by outdated parameters.

Summary and Further Plotting Resources

The "figure margins too large" error in R is a common hurdle related to spatial management within the graphics environment. Whether you choose to physically expand the plot panel, programmatically reduce the required margins using `par(mar=...)`, or forcefully reset the device using `dev.off()`, the solution lies in ensuring that the reserved space for axes and labels is small enough to fit within the visible plotting area.

Mastering graphical parameters is essential for producing high-quality visualizations. Understanding how functions like `par()` control the aesthetic layout of your figures allows for intricate customization beyond simply resolving errors. Always remember that adjusting margins is a trade-off: while smaller margins fix initialization errors, they might compromise the legibility of highly detailed axis labels.

The following resources offer detailed information on other essential plotting techniques and functions in R, further enhancing your data visualization capabilities: