# Learning to Plot Non-Parametric Distributions in R Using plotMP()

Authored by

**Mohammed looti**

November 13, 2025

## Visualizing Complex Two-Dimensional Distributions in R

When conducting advanced statistical analysis in R, researchers frequently face the complex task of graphically representing intricate data structures. A particularly challenging scenario arises when visualizing a two-dimensional non-parametric distribution. Standard two-dimensional plots, such as basic scatter plots or histograms, are inherently inadequate for this purpose because they fail to convey the crucial information: the **joint probability mass or density** across the bivariate plane. Since a non-parametric approach is necessary when the underlying distribution shape is unknown or cannot be reliably assumed to adhere to a standard parametric form (like the Normal distribution), specialized visualization is essential. This requirement necessitates a plotting tool capable of rendering a three-dimensional surface that accurately maps coordinate pairs to their corresponding probability values.

In the context of two dimensions, a non-parametric distribution fundamentally describes the relationship between two variables without imposing rigid mathematical assumptions regarding their underlying structure. This type of visualization is absolutely essential for interpreting the results of complex statistical procedures, such as mixture models, analyzing discrete multivariate data, or simply examining empirical distributions where the joint probability mass function (PMF) requires unambiguous representation. Unlike continuous distributions, which can often be depicted using contours or density heatmaps, discrete 2D non-parametric data demands a clear perspective plot that distinctly visualizes each individual point and its associated height, which represents the probability mass.

The limitations of conventional plotting functions in handling this specific visualization requirement lead us directly to seek out specialized packages within the R ecosystem. Fortunately, the vast repository of R provides robust solutions tailored for sophisticated statistical graphics, particularly those developed within the framework of generalized additive models. The function we will examine here is specifically engineered to transform three related vectors--representing the X coordinates, the Y coordinates, and their joint probabilities--into an informative, fully customizable 3D perspective plot that clearly communicates the structure of the data.

## Introducing the plotMP() Function and the gamlss.mx Package

The most effective and powerful method for generating a plot of a two-dimensional non-parametric distribution in R involves utilizing the **plotMP()** function. This function is a core component of the **gamlss.mx** package. The `gamlss.mx` package serves as an extension of the broader GAMLSS (Generalized Additive Models for Location, Scale and Shape) framework, focusing specifically on the robust fitting and detailed analysis of finite mixture distributions. While the primary purpose of the parent GAMLSS framework is advanced statistical modeling, its auxiliary packages frequently contain highly powerful visualization tools that are necessary for effective model diagnostics and

professional presentation.

The **plotMP()** function is carefully engineered to process the output derived from such mixture models or any discrete joint probability data set. It accepts discrete (x, y) coordinate pairs and uses the probability mass associated with each pair to determine the height of the plot, thus constructing an effective 3D bar chart rendered in perspective. This particular visualization technique is instrumental in clearly demonstrating where the highest concentrations of probability mass are situated within the two-dimensional space defined by the X and Y axes, making even complex joint distributions immediately understandable to the viewer.

By leveraging the built-in capabilities within the **gamlss.mx** package, users can entirely bypass the complex and time-consuming manual coding that would typically be required to construct such a plot from scratch using base R graphics functions. The function intelligently encapsulates all the necessary steps for data transformation and 3D rendering, offering a simplified syntax that enables statisticians and data scientists to dedicate their attention to interpreting the underlying non-parametric distribution rather than struggling with the intricate mechanics of plot generation. This significant ease of use offers a considerable advantage, especially when rapidly prototyping visualizations for ongoing research or detailed reporting.

## Deciphering the plotMP() Syntax and Essential Arguments

To effectively harness the capabilities of this powerful visualization tool, it is paramount to understand the structure and parameters required by the **plotMP()** function. The function relies on three core input data vectors and offers a comprehensive array of optional arguments for controlling aesthetic appearance and visual perspective. A thorough understanding of these arguments enables complete customization of the resulting three-dimensional graphic, ensuring it meets specific analytical or publication needs.

The **plotMP()** function utilizes the following basic syntax structure, which includes several powerful optional parameters inherited directly from the underlying R's base graphics `persp` function:

**plotMP(x, y, prob, theta = 20, phi = 20, expand = 0.5, col = "lightblue", xlab = "intercept", ylab = "slope", ...)**

The initial three arguments are mandatory and define the data structure being plotted, while the subsequent optional arguments govern the graphical output and viewing angle. The judicious selection of these parameters, particularly the perspective controls (`theta` and `phi`), is crucial for ensuring that the resulting plot clearly and accurately conveys the spatial relationship between the variables and the associated probability mass. This high level of control is necessary because the accurate interpretation of any 3D plot is heavily dependent on the chosen viewing angle.

The arguments are defined as follows, explicitly specifying their critical role in defining both the data structure and the resulting visual appearance:

**x**: A vector containing the points that define the coordinates along the horizontal X-axis. These values typically represent the first of the two variables in the joint distribution.
**y**: A vector containing the points that define the coordinates along the depth (Y) axis. These values represent the second variable in the joint distribution.
**prob**: The critical third vector containing the probabilities or masses associated with each corresponding (x, y) pair. It is absolutely crucial that the sum of all elements in this vector is equal to 1 (or extremely close to 1) for it to represent a valid probability mass function. This vector determines the height (Z-axis) of the plotted bars.
**theta**, **phi**, **expand**, **col**: These are arguments primarily passed directly to the underlying `persp` function, which manages the 3D rendering engine. **theta** and **phi** control the viewing angle (azimuth and colatitude), **expand** scales the plot height, and **col** sets the color of the base plane.
**xlab**: Defines the descriptive label displayed on the X-axis of the plot.
**ylab**: Defines the descriptive label displayed on the Y-axis (the depth axis) of the plot.

## Prerequisites and Installation of the gamlss.mx Package

Before proceeding to use the **plotMP()** function, it is mandatory to ensure that the required package, **gamlss.mx**, is both installed on your system and successfully loaded into your current R session. As this package is not included in the standard R distribution, a simple one-time installation step is necessary. Attempting to call functions from a package that has not been installed or loaded will invariably result in an error, immediately halting the execution of your script or analysis.

The installation process is straightforward and relies on the standard package management function available in R. You can install the package using the following syntax typed directly into the R console. It is important to remember that this installation step only needs to be performed once per R environment setup, unless an update to the package is specifically required:

**install.package('gamlss.mx')**

Once the initial installation is complete, the package must be explicitly loaded into the current session every time you restart R and intend to utilize the functions it contains. The standard `library()` command handles this loading process, effectively importing all functions, including **plotMP()**, into the global workspace. Successfully completing these two steps--installation and subsequent loading--guarantees that you can proceed directly to the visualization examples without encountering dependency errors.

## Practical Example 1: Visualizing Basic Probability Data in R

To clearly illustrate the core functionality of **plotMP()**, we will now walk through a complete example, beginning with the necessary input data creation. We must define three vectors: one for the X coordinates, one for the Y coordinates, and a third for the joint probabilities (P). It is absolutely vital that these three vectors are of equal length, as each index corresponds precisely to a specific point (x, y) and its associated probability mass (p).

We begin the example by defining the data. For this illustration, we create ten distinct points scattered across the X-Y plane and assign a probability mass to each point. Crucially, we ensure that the probability vector sums exactly to 1.0, representing a complete joint probability mass function:

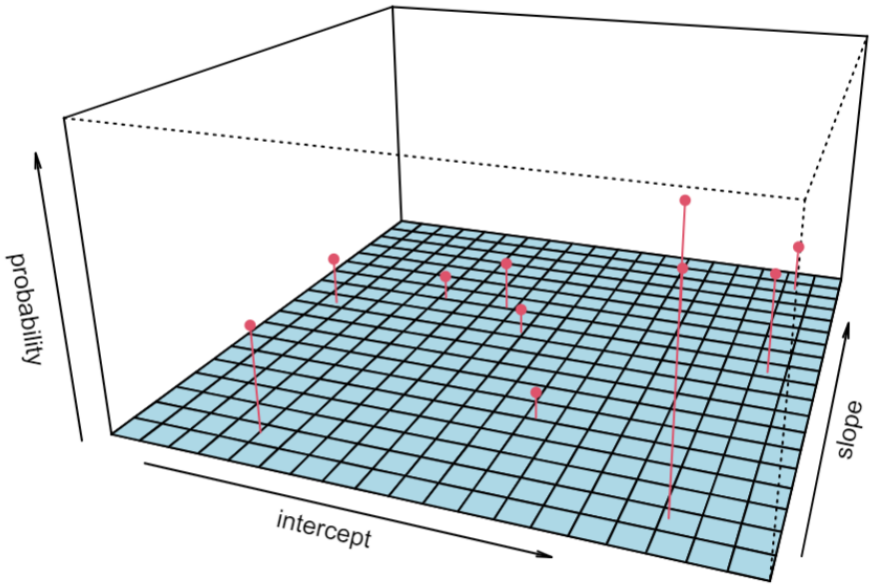
```
#create three vectors
x <- c(-5, -3, -2, 0, 1, 3, 5, 7, 8, 8)
y <- c(3, -2, 4, 4, 3, 0, 5, -2, 7, 3)
p <- c(.1, .2, .05, .1, .05, .05, .1, .5, .1, .2)
```

With the data successfully prepared, we now load the **gamlss.mx** package and execute the **plotMP()** function, passing our newly created `x`, `y`, and `p` vectors as the primary arguments. This straightforward function call generates the default 3D perspective plot:

```
library(gamlss.mx)

#plot values from vectors
plotMP(x, y, p)
```

This execution produces the following plot, which visually represents the joint probability mass function in three dimensions:




In the resulting visualization, the spatial relationships are mapped clearly: the horizontal plane is defined by the values of the `x` and `y` vectors (representing the X-axis and Y-axis, respectively), while the vertical axis (the Z-axis) displays the probability mass defined by the `p` vector. A critical detail to observe is the direct relationship between the probability value and the bar height. The higher the value recorded in the probability vector, the taller the corresponding vertical red line appears in the plot. For instance, the point where the probability is 0.5 stands significantly taller than points with a probability of 0.05, immediately highlighting the regions of highest probability concentration within this bivariate non-parametric distribution.

## Advanced Customization: Controlling Aesthetics and Perspective

Although the default plot provides a statistically accurate representation of the data, professional visualization frequently requires customization to conform to publication standards or simply to enhance clarity for the intended audience. The **plotMP()** function allows users to modify several key aesthetic features, most notably the color of the base plane and the vertical scaling of the plot, through the **col** and **expand** arguments, respectively. These arguments provide fine-grained control over the visual presentation without making any alterations to the underlying statistical data being visualized.

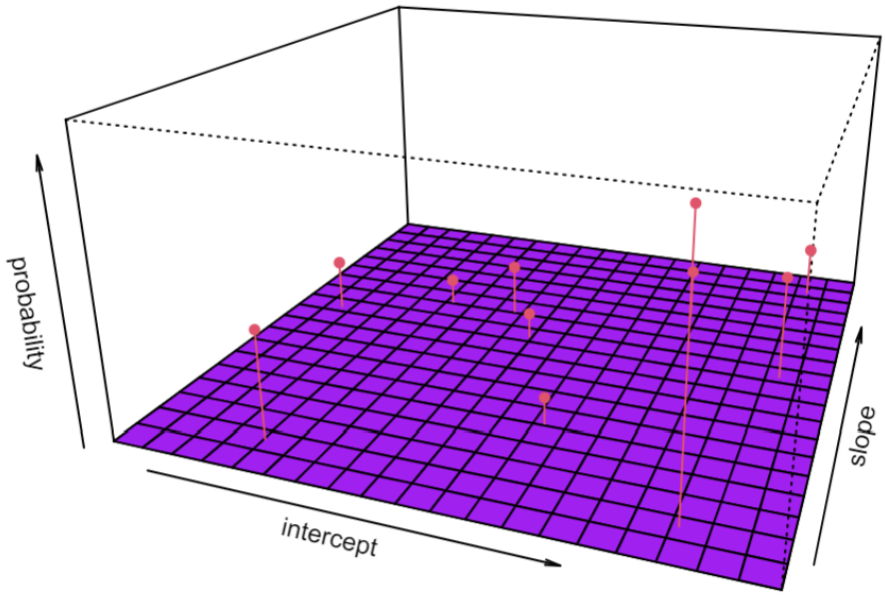
By default, the "floor" or base plane of the plot is rendered in light blue (`lightblue`). If this color conflicts with the background of a document or if a different aesthetic is desired, the **col** argument can be used to specify any valid R color name or hexadecimal code. For example, we might choose a purple color to provide a better visual contrast or to align with a specific thematic design.

This customization is achieved simply by including the `col` argument within the function call:

**library(gamlss.mx)**

#plot values from vectors
plotMP(x, y, p, col='purple')

This simple modification yields a plot where the visual emphasis on the probability bars is successfully maintained, but the base aesthetics are updated to the specified color:



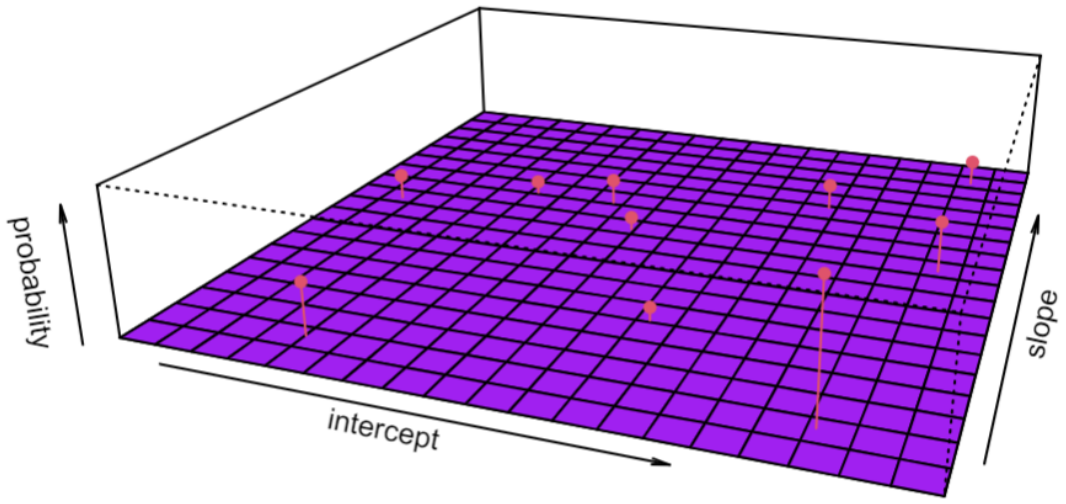

Another critical parameter for controlling the visual impact of the 3D plot is the **expand** argument. This argument directly dictates the scaling factor applied to the Z-axis (height). By manipulating **expand**, users gain control over the overall perceived height of the probability mass peaks relative to the X-Y plane. The default value for **expand** is **0.5**. Utilizing a smaller value compresses the vertical axis, causing the plot to appear shorter and flatter, which can be beneficial when probability differences are subtle but still need visualization without excessive vertical distortion. Conversely, using a larger value significantly exaggerates the vertical scale, which can emphasize differences in probability mass.

To demonstrate the effect of shortening the plot height, we can set the **expand** argument to a value smaller than the default, such as **0.2**, while retaining our custom purple floor color:

**library(gamlss.mx)**

```
#plot values from vectors
plotMP(x, y, p, expand=0.2, col='purple')
```

This adjustment produces the following plot, where the reduced vertical scale is immediately apparent, offering a different visual perspective:




It is important to reiterate that **plotMP()** functions essentially as a convenient wrapper for the powerful base R `persp` function. This integration means that many other arguments accepted by `persp`, such as **theta** and **phi** (which control the azimuth and colatitude viewing angles, respectively), can also be passed directly to **plotMP()**. Experimenting with these perspective parameters--especially the viewing angles--is highly recommended to find the absolute optimal perspective that best highlights the nuances of the two-dimensional non-parametric distribution being analyzed. Users are strongly encouraged to adjust the values for the different arguments in the **plotMP()** function to create the exact plot that perfectly meets their analytical and presentation needs.

## Additional Resources for R Visualization

Mastering the visualization of complex distributions is recognized as a key skill in modern data science and statistical reporting. The following tutorials explain how to perform other common operations in R, providing essential context for further exploration beyond the **gamlss.mx** package and the **plotMP()** function:

<!--

## Featured Posts

-->