# Plot Groups Using PROC SGPLOT in SAS

Authored by

**stats writer**

November 19, 2025

**SAS** is a powerful software suite utilized globally for advanced **data analysis**, reporting, and visualization. Central to its graphical capabilities is the **PROC SGPLOT** procedure. This procedure is specifically designed to generate high-quality statistical graphics with minimal coding effort, enabling analysts to quickly transform raw data into insightful visual displays. Unlike older SAS procedures, **PROC SGPLOT** employs ODS Graphics, which ensures that output plots are scalable, customizable, and presentation-ready.

The versatility of **PROC SGPLOT** spans numerous plot types crucial for exploratory data analysis and reporting. These include standard representations like scatter plots, line plots, box plots, and **histograms**. Crucially, **PROC SGPLOT** excels at handling grouped data, offering sophisticated options for displaying subgroups, which is often essential for comparing distributions or trends across different categories within a single dataset.

Beyond simple plotting, **PROC SGPLOT** provides extensive controls over the aesthetic elements of the visualization. Users can precisely manage the visual appearance by setting custom colors, adjusting marker shapes and sizes for data points, and applying specific styles to lines and curves. Furthermore, enhancing interpretability is straightforward; the procedure supports the seamless addition of informative elements such as dynamic labels, descriptive titles, and essential legends that clearly delineate the different groups or variables being displayed.

## The Importance of Grouped Visualization in Data Analysis

When conducting complex **data analysis**, it is rarely sufficient to visualize the overall distribution of a variable. Real-world datasets typically contain categorical variables that define meaningful subgroups, such as product lines, geographic regions, or competitive teams. Visualizing these groups separately or simultaneously is vital for identifying inter-group differences, verifying assumptions, and detecting anomalies that might be obscured when observing the aggregate data alone.

In the context of the examples provided here, plotting the score distributions for different teams allows an analyst to immediately grasp whether Team A's performance differs significantly from Team B's, both in terms of central tendency (average score) and variability (spread of scores). **PROC SGPLOT** offers two primary, powerful techniques to achieve this grouped visualization, each serving a distinct analytical purpose: generating multiple individual charts or creating a single composite chart with overlaid elements.

## Method 1: Creating Separate Charts Using the BY Statement

One effective way to manage the visualization of categorized data is to generate individual plots for every unique group. This approach is best suited for situations where the analyst needs to examine the internal characteristics of each group without the distraction of comparison, or when the

number of groups is small enough that multiple distinct plots are easy to review. In **SAS** programming, this functionality is primarily achieved through the use of the **BY** statement immediately following the **PROC SGPLOT** call.

The critical functionality of the **BY** statement is its ability to segment the input data stream based on the values of a specified variable. When **PROC SGPLOT** encounters a **BY** statement, it iterates through the data, generating a complete, separate graphical output for every distinct value found in the grouping variable. This ensures that each output chart focuses solely on the observations belonging to one specific group, providing maximum clarity regarding that group's distribution or trend.

For instance, if we aim to visualize the distribution of 'points' scores segmented by 'team', the inclusion of `by team;` within the procedure step dictates that **SAS** will produce one graphical output for Team A, a second graphical output for Team B, and so on. The code snippet below illustrates this mechanism for generating multiple **histograms**, each dedicated to a single team's score distribution:

```
/*create multiple plots that show histogram of points for each team*/
proc sgplot data=my_data;
by team;
histogram points;
density points / type=kernel;
run;
```

## In-Depth Look at the BY Statement Syntax and Logic

The **BY** statement is a fundamental component in many **SAS** procedures, not just **PROC SGPLOT**. When used, it requires the input dataset to be sorted by the **BY** variable. Although modern graphical procedures often handle sorting implicitly, it is best practice to ensure the dataset is pre-sorted using **PROC SORT** to guarantee optimal performance and accurate output, especially when dealing with large datasets. Failure to sort the data might lead to unexpected or incomplete graphical outputs if the data is not ordered correctly.

Within the visualization context, the **BY** statement acts as a powerful segmentation tool. Every output graph generated by this method will automatically include the **BY** variable value (e.g., Team A, Team B) in the title or subtitle, providing immediate context for the viewer. This is crucial for maintaining clarity when reviewing dozens of separate charts generated from a dataset containing many different subgroups.

Furthermore, notice the inclusion of the **DENSITY** statement in the example code: `density points /

type=kernel;`. This statement is used to overlay a continuous probability density curve, specifically using **kernel density estimation**. This curve is non-parametric and serves as an excellent visual summary of the shape of the distribution, smoothing out the noise present in the raw **histogram** bars. When the **BY** statement is active, **SAS** calculates and plots a separate density curve for each individual chart, ensuring the smoothing is relevant only to the data points in that specific group.

## Method 2: Overlaying Groups on a Single Chart Using the GROUP Option

The alternative, and often more powerful, method for comparing distributions is plotting all groups onto a single visualization. This technique is indispensable when the primary analytical goal is immediate comparison--understanding how the distributions overlap, where their centers differ, and comparing their respective spreads directly on the same scale. In **PROC SGPLOT**, this is achieved by utilizing the **GROUP=** option within the plot statement itself (e.g., **HISTOGRAM** or **SCATTER**).

When the **GROUP=** option is applied, **SAS** does not segment the output into separate charts. Instead, it maintains a single coordinate system and generates distinct graphical elements for each unique value of the specified grouping variable (e.g., `group=team`). **SAS** automatically assigns different visual attributes--such as color, line pattern, or marker symbol--to differentiate the groups, and it creates an appropriate legend to map these attributes back to the group labels.

The resulting single chart provides an efficient platform for comparative analysis. For instance, overlaying two **histograms** allows the analyst to instantly see which team has a higher median score or whether the variability of scores is greater in one team versus the other. The following code demonstrates how to use the **GROUP=** option to display separate **histograms** and their corresponding density curves for each team within a unified plot:

```
/*plot histogram of points for each team on one chart*/
proc sgplot data=my_data;
histogram points / group=team;
density points / type=kernel group=team;
run;
```

## Comparing the BY Statement and the GROUP Option

Although both the **BY** statement and the **GROUP=** option achieve the goal of visualizing data by subgroups, their mechanical implementation and resulting analytical benefits are fundamentally different. Understanding these differences is key to choosing the correct approach for a given visualization task. The **BY** statement operates at the procedure level, dictating that **SAS** should execute the entire procedure--including all plot statements--separately for each group, resulting in multiple output graphs.

Conversely, the **GROUP=** option operates at the plot statement level (e.g., within the **HISTOGRAM** statement). It instructs the procedure to draw distinct graphical elements corresponding to each group within a single graphical canvas. This is advantageous for direct comparison but can quickly become visually complex if the number of groups is large (e.g., more than 5 or 6 groups) or if the individual group distributions overlap significantly.

When choosing between the two, analysts should consider the volume of groups and the comparison required. If you have many groups (e.g., 20 states), using the **BY** statement might yield 20 separate, readable charts. If you use **GROUP=** with 20 groups, the resulting single chart would likely be cluttered and illegible due to color overuse and excessive overlap. For small-scale, direct, side-by-side comparison, the **GROUP=** option is superior, as it places all distributions on the exact same axis scale, facilitating immediate visual assessment of differences in means and ranges.

## Practical Demonstration: Setting Up the Sample Dataset

To illustrate these grouping methods effectively, we will utilize a small, fictional dataset detailing the performance scores ('points') for two distinct competitive entities ('team'). This dataset is constructed using a standard **SAS** data step, which is the foundational method for creating and initializing datasets within the **SAS** environment.

The following **SAS** code block defines the input data, assigning points scores to either Team A or Team B. After the data is created, the **PROC PRINT** step is used to display the raw data in the output window, allowing verification of the dataset structure before proceeding with graphical analysis.

```
/*create dataset*/
data my_data;
input team $ points;
datalines;
A 29
A 23
A 20
A 21
A 33
A 35
A 31
B 21
B 14
B 15
```

```
B 11
B 12
B 10
B 15
;
run;

/*view dataset*/
proc print data=my_data;
```

The resulting dataset, which is automatically generated and viewed, confirms that the structure contains two variables: **team** (a character variable denoted by the '$' in the input statement) and **points** (a numeric variable), ready for visualization.

| Obs | team | points |
|---|---|---|
| 1 | A | 29 |
| 2 | A | 23 |
| 3 | A | 20 |
| 4 | A | 21 |
| 5 | A | 33 |
| 6 | A | 35 |
| 7 | A | 31 |
| 8 | B | 21 |
| 9 | B | 14 |
| 10 | B | 15 |
| 11 | B | 11 |
| 12 | B | 12 |
| 13 | B | 10 |
| 14 | B | 15 |



## Example 1: Implementing the BY Statement for Separate Charts

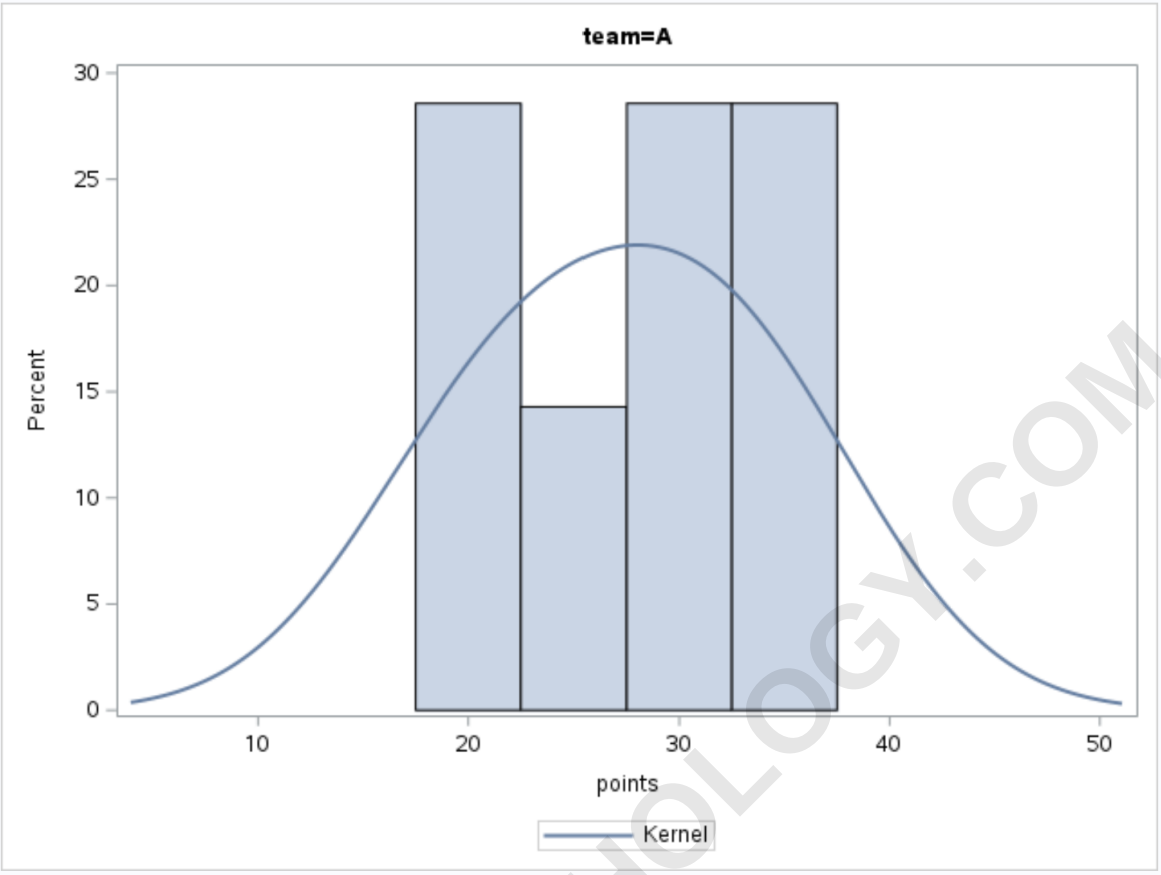
In this first example, we utilize the powerful segmentation capability of the **BY** statement within **PROC SGPLOT**. Our objective is to generate two distinct **histograms**--one detailing the distribution of **points** for Team A, and the other for Team B. This clear separation ensures that each team's performance profile is analyzed in isolation, free from visual interference from the

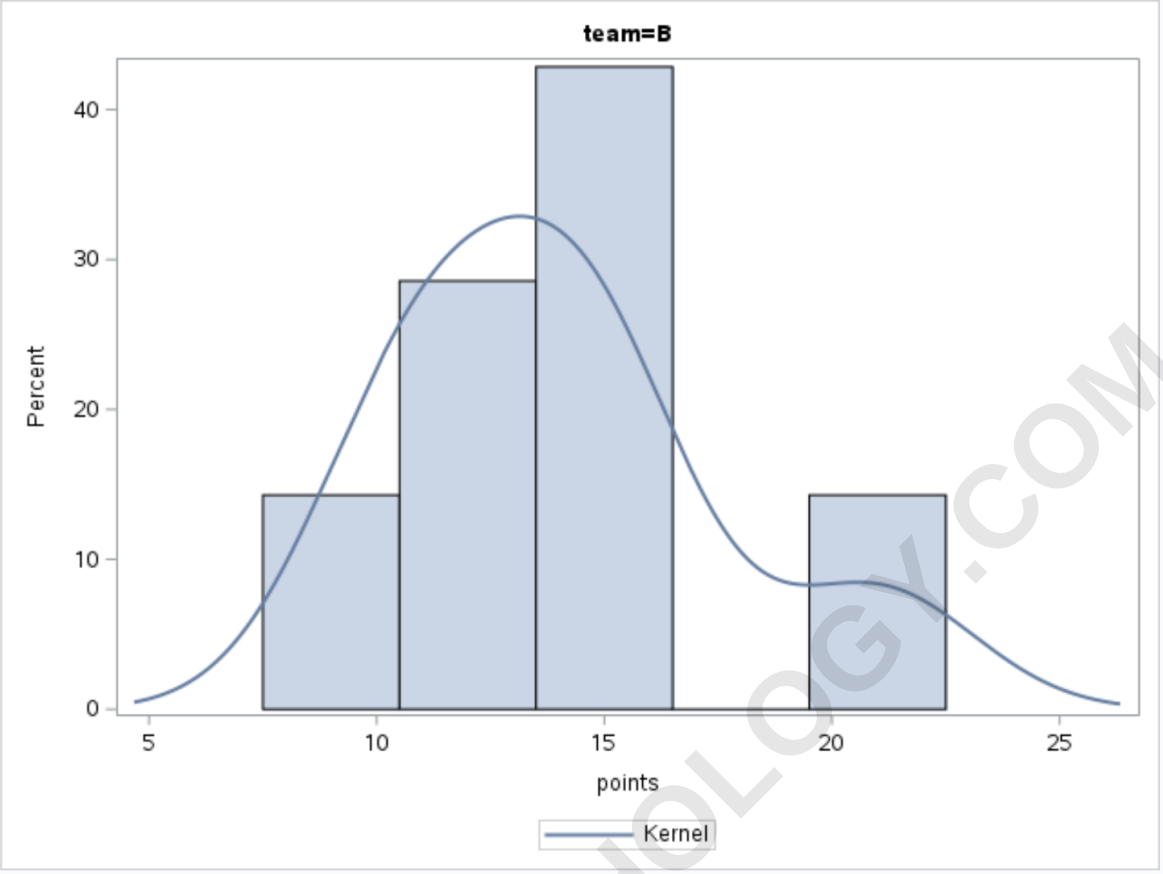
other group.

By simply including the line `by team;` immediately after the procedure statement, we instruct **SAS** to iterate over the dataset, producing a complete histogram plot for every value of the **team** variable. We also include the **DENSITY** statement to overlay a smoothed kernel curve on the **histogram** bars, providing an additional layer of insight into the distributional shape.

```
/*create multiple plots that show histogram of points for each team*/
proc sgplot data=my_data;
by team;
histogram points;
density points / type=kernel;
run;
```

The resulting output confirms that the procedure successfully generated two discrete graphs. The first graph displays the distribution of scores for Team A, which appears to be centered around a higher point value with a moderate spread. The second graph focuses exclusively on Team B, revealing a distribution centered at a lower score, indicating a notable performance gap between the two teams.








The first **histogram** clearly illustrates the score distribution for team A, while the second separate **histogram** is dedicated to displaying the distribution of scores for team B.

**Note**: The optional **density** statement is highly recommended in distributional plotting, as it overlays a continuous probability curve. This curve provides a smooth, non-binned summary of the data's distribution shape, which is often more helpful than the stepped bars of the **histogram** alone, especially when sample sizes are small.

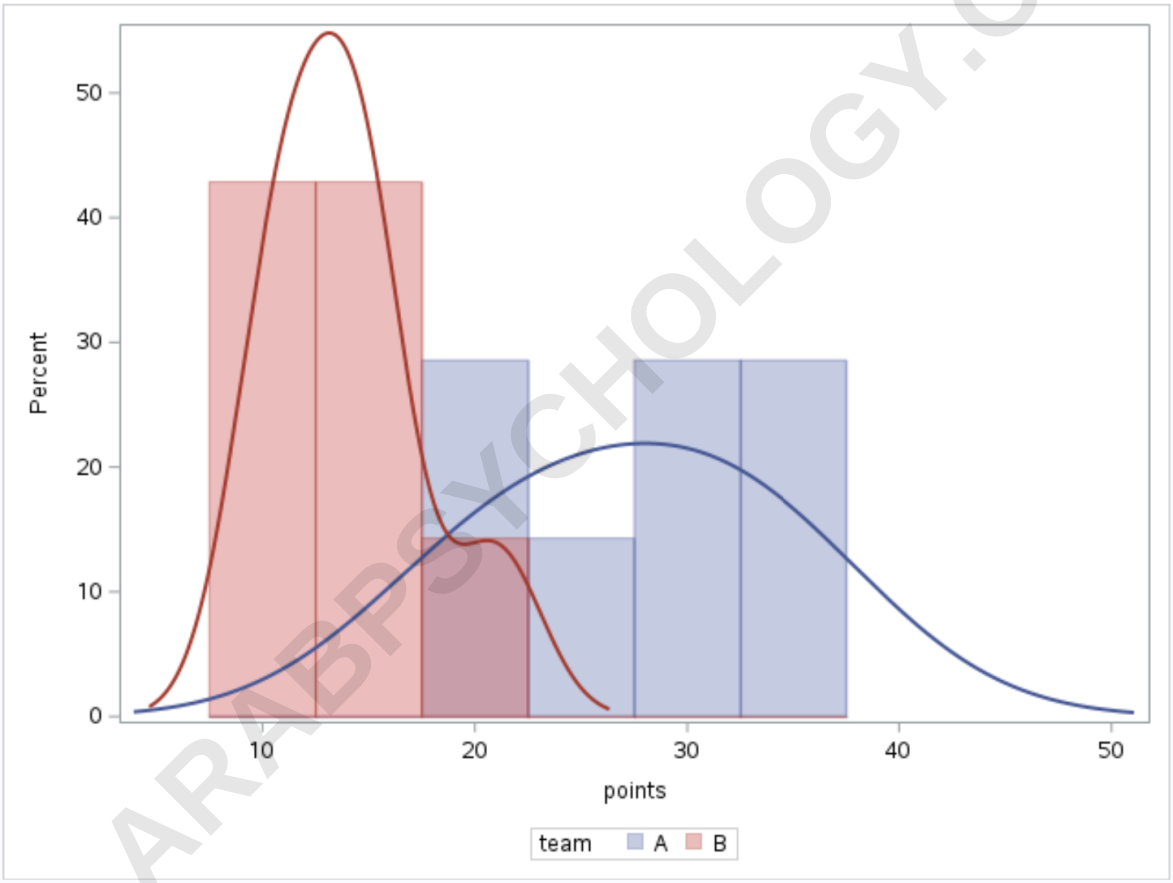
## Example 2: Implementing the GROUP Option for Overlaid Charts

For direct visual comparison, we transition to using the **GROUP=** option within **PROC SGPLOT**. This method is utilized to create a single, unified chart containing overlaid **histograms**, allowing the analyst to contrast the distributions of **points** for each **team** instantly.

By applying the `group=team` option to both the **HISTOGRAM** statement and the **DENSITY** statement, we instruct **SAS** to draw the elements for each team onto the same plot, automatically assigning distinct colors and generating a comprehensive legend. To manage the inevitable overlap when distributions are plotted together, we introduce a critical aesthetic option: **TRANSPARENCY**.

```
/*plot histogram of points for each team on one chart*/
proc sgplot data=my_data;
histogram points / group=team transparency=0.5;
density points / type=kernel group=team;
run;
```

The resulting visualization provides an immediate, powerful comparison. The blue **histogram** displays the distribution of points values for team A and the red **histogram** displays the distribution of points values for team B. The overlap region and the distinct placement of the density curves make the performance disparity visually striking.






The major benefit of using this overlaid approach is the ability to quickly and efficiently compare the distribution of **points** values for each **team**, as both are anchored to the same horizontal axis and scale. This technique is invaluable for rapid analytical insights, especially when assessing whether distributions are sufficiently separated to warrant formal statistical testing.

## Enhancing Visualization: Customizing Density and Transparency

When using the **GROUP=** option, particularly with **histograms**, the issue of overlap must be addressed to ensure readability. If the bars of different groups completely hide each other, the chart loses its effectiveness. This is where the **TRANSPARENCY** option becomes essential.

The **transparency** argument, applied within the plot statement (e.g., `histogram points / group=team transparency=0.5;`), controls the opacity of the graphical elements. The value ranges from 0 (fully opaque) to 1 (fully transparent). By setting this value to **0.5**, as demonstrated in Example 2, the histograms become semi-transparent, allowing the viewer to discern the bars of the underlying group where they overlap with the bars of the foreground group.

Effective use of **TRANSPARENCY**, coupled with the inclusion of the smooth **DENSITY** plot, transforms a potentially confusing chart into a highly informative comparative visualization. The closer you set this value to **1**, the more transparent the histograms become, helping to avoid visual clutter when analyzing highly overlapping distributions.

## Further Learning Resources

For those interested in expanding their knowledge of statistical graphics in **SAS**, the following list includes tutorials on creating other chart types using **PROC SGPLOT** and related procedures:

**Creating Box Plots:** Essential for comparing descriptive statistics (median, quartiles) across multiple groups.
**Generating Scatter Plots:** Ideal for visualizing relationships between two continuous variables, often utilizing the **GROUP=** option to differentiate categorical segments.
**Utilizing PROC GCHART:** While **PROC SGPLOT** is generally preferred for statistical quality, **PROC GCHART** remains relevant for certain business graphics, such as simple bar charts and pie charts.