How to Customize SAS Histograms by Specifying the Number of Bins

Authored by

stats writer

November 22, 2025

Understanding Histograms and Bins in SAS

A histogram is an essential tool in statistical data analysis, offering a visual representation of the distribution of numerical data. When generating a histogram in the **SAS System**, one of the most crucial parameters to control is the definition of the data grouping intervals, commonly referred to as **bins**. These bins determine how observations are aggregated and subsequently displayed on the chart.

While some statistical software uses a simple `BINS=` option to directly specify the count of bins, **SAS** offers a more precise level of control through the explicit definition of **midpoints**. By explicitly setting the midpoints, analysts can customize the visualization, ensuring that the presentation accurately reflects the underlying data structure or meets specific reporting requirements. This level of customization is paramount for preventing misinterpretation caused by overly broad or excessively narrow data grouping.

To effectively control the bin size and placement in SAS, we utilize options available within the **HISTOGRAM** statement of procedures like PROC UNIVARIATE. The ability to specify bin definition, either through simple numerical values or calculated expressions, grants the user powerful flexibility in exploratory data analysis and presentation graphics.

To directly manage the size and count of bins in a SAS histogram, you must leverage the **MIDPOINTS=** option. This is the primary method for overriding the default bin calculation algorithms that SAS uses when visualizing continuous variables.

The MIDPOINTS Option: Customizing Bins in SAS

The most robust way to specify the structure of your histogram, including the number and width of the bins, is by utilizing the MIDPOINTS option within the **HISTOGRAM** statement. Instead of calculating the number of bins directly, SAS expects you to define the sequence of values that represent the center point (midpoint) of each bin. The interval between these midpoints inherently defines the width of each bin and, consequently, the total count of bins required to cover the range of the data.

The syntax for defining these midpoints follows a standard SAS iterative notation: `(start TO end BY interval)`. The `start` value specifies the midpoint of the lowest bin, `end` specifies the midpoint of the highest bin, and the `interval` determines the consistent step size between successive midpoints. This step size is directly equivalent to the width of the bins used in the visualization.

Using this structure provides complete control over the data binning process, ensuring that the resulting visual representation aligns exactly with the analytical narrative you wish to convey. Here

is the foundational syntax demonstrating its usage:

```
proc univariate data=my_data;
histogram my_variable / midpoints=(9 to 36 by 3);
run;
```

In this specific example, we are instructing **SAS** to generate a histogram where the center points of the bins begin at **9** and continue up to **36**. Crucially, the bins are defined such that the distance between consecutive midpoints is **3**. This interval of **3** dictates the width of every bin in the resulting visualization, offering a precise visual breakdown of the `my_variable` distribution.

Practical Example: Setting Up the Sample Data

To illustrate the practical application of the **MIDPOINTS=** option, we will utilize a small, representative dataset containing statistics for basketball players. This example dataset, named `my_data`, includes variables for `team` (character), `points` (numerical), and `reboundss` (numerical). The primary variable we aim to visualize and control the binning for is `points`, which represents the total score accumulated by each player.

Creating this dataset is a prerequisite for running any subsequent procedures in SAS. We use the **DATA step** along with the **INPUT** and **DATALINES** statements to define and populate the variables. Once the data is loaded, we use **PROC PRINT** to verify its structure and content before proceeding to the visualization stage.

The following SAS code block demonstrates the creation and immediate inspection of our working dataset:

```
/*create dataset*/
data my_data;
input team $ points rebounds;
datalines;
A 29 8
A 23 6
A 20 6
A 21 9
A 33 14
A 35 11
A 31 10
B 21 9
B 14 5
```

```
B 15 7
B 11 10
B 12 6
B 10 8
B 15 10
;
run;

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

The resulting table confirms that the data has been loaded correctly, allowing us to proceed with generating the initial histogram.

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

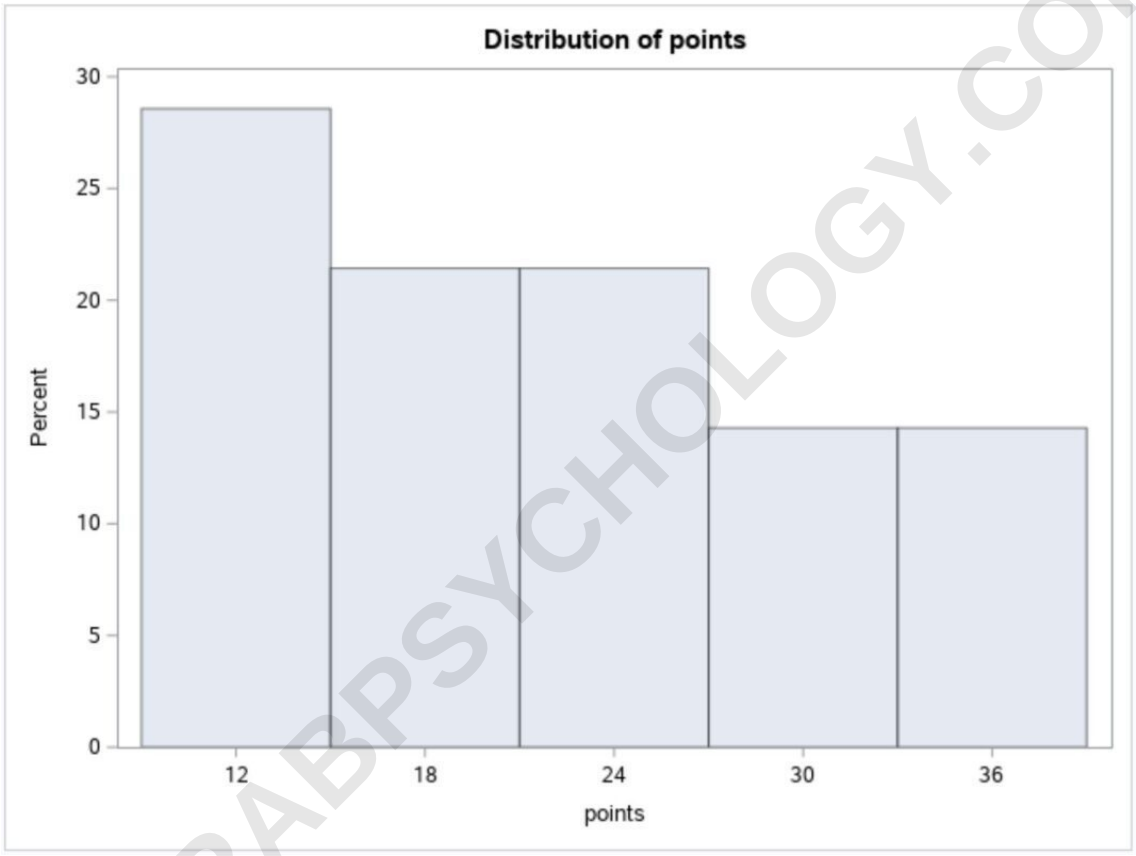
Analyzing the Default Histogram Generation

Before implementing custom bin specifications, it is instructive to observe how SAS handles the visualization when no options are explicitly defined. By simply using the **HISTOGRAM** statement without the MIDPOINTS option, SAS automatically calculates the optimal number of bins based on established statistical rules (such as Sturges' formula or the Freedman-Diaconis rule).

We execute the PROC UNIVARIATE step below to generate the default histogram for the `points` variable:

```
/*create histogram for points variable*/
proc univariate data=my_data;
histogram points;
run;
```

The resulting visualization shows the distribution of scores. The **x-axis** represents the range of the **points** variable, while the **y-axis** typically displays the percentage of observations falling into each designated bin.



Upon examining the default output, we can deduce the bin width chosen by SAS. In this particular visualization, the automatically generated midpoints appear to be spaced at intervals of **6** units. While this default setting provides a reasonable overview, it might obscure finer details of the data distribution, especially if the data is multimodal or contains significant clustering within these 6-unit intervals. This realization prompts the need for manual adjustment to achieve a more granular view.

Increasing Precision: Specifying Smaller Bin Intervals

When the default bin width is too large, it results in a histogram that is overly generalized,

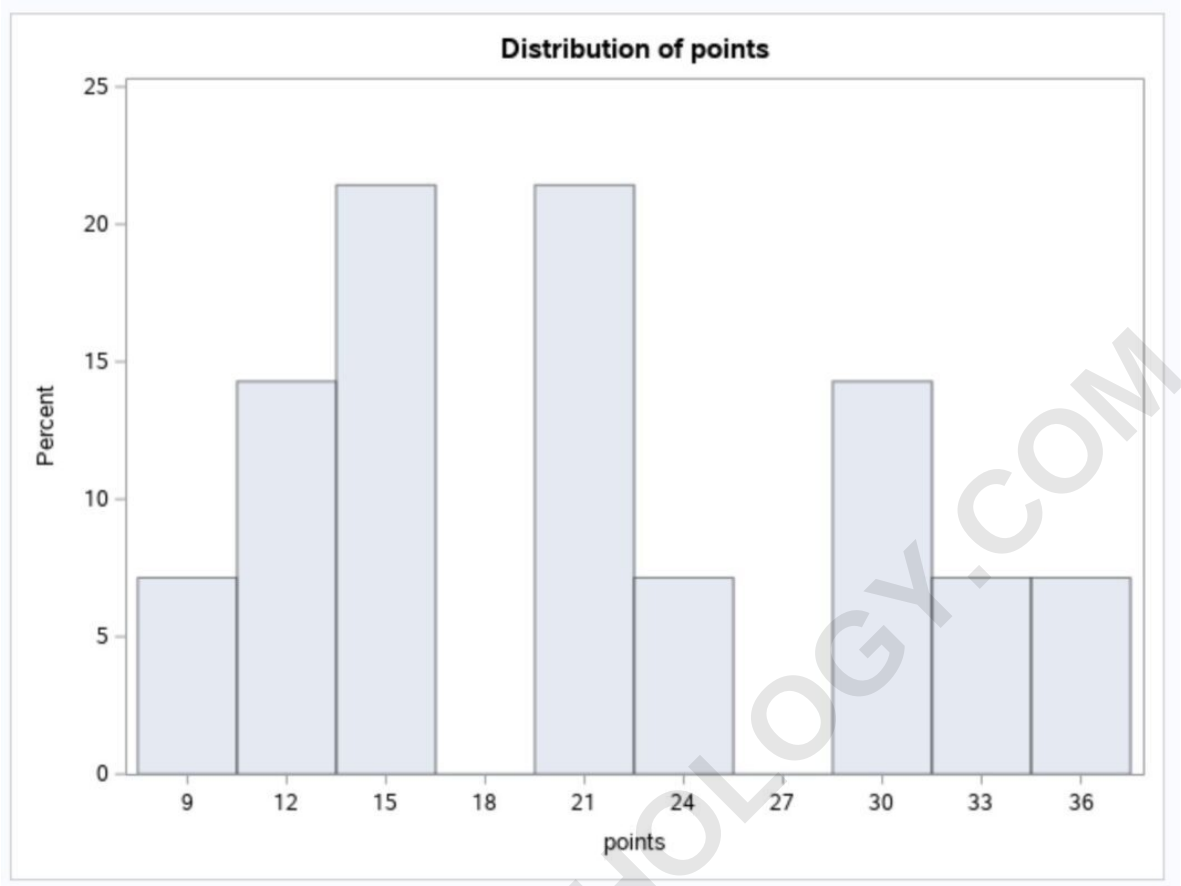
potentially smoothing over crucial peaks or gaps in the distribution. To gain a higher resolution view of the data, we must decrease the bin width, which automatically increases the total number of bins. In SAS, this is achieved by specifying a smaller `interval` value in the **MIDPOINTS=** option.

For our basketball dataset, we observed a default interval of 6. We can now choose to halve this width to **3**. By setting `BY 3`, we are generating twice as many bins within the same range (9 to 36) compared to the default setting, thereby increasing the visual precision and revealing more detail about the frequency density.

This adjustment helps to better characterize the true shape of the distribution. The syntax below reflects the implementation of this finer bin structure:

```
/*create histogram for points variable with custom bins*/
proc univariate data=my_data;
histogram points / midpoints=(9 to 36 by 3);
run;
```

The resultant image clearly demonstrates a change in the visualization compared to the default output. The narrower bins provide a more detailed profile of player scores, highlighting exactly where the clusters and gaps are located across the variable's range.



Decreasing Resolution: Specifying Larger Bin Intervals

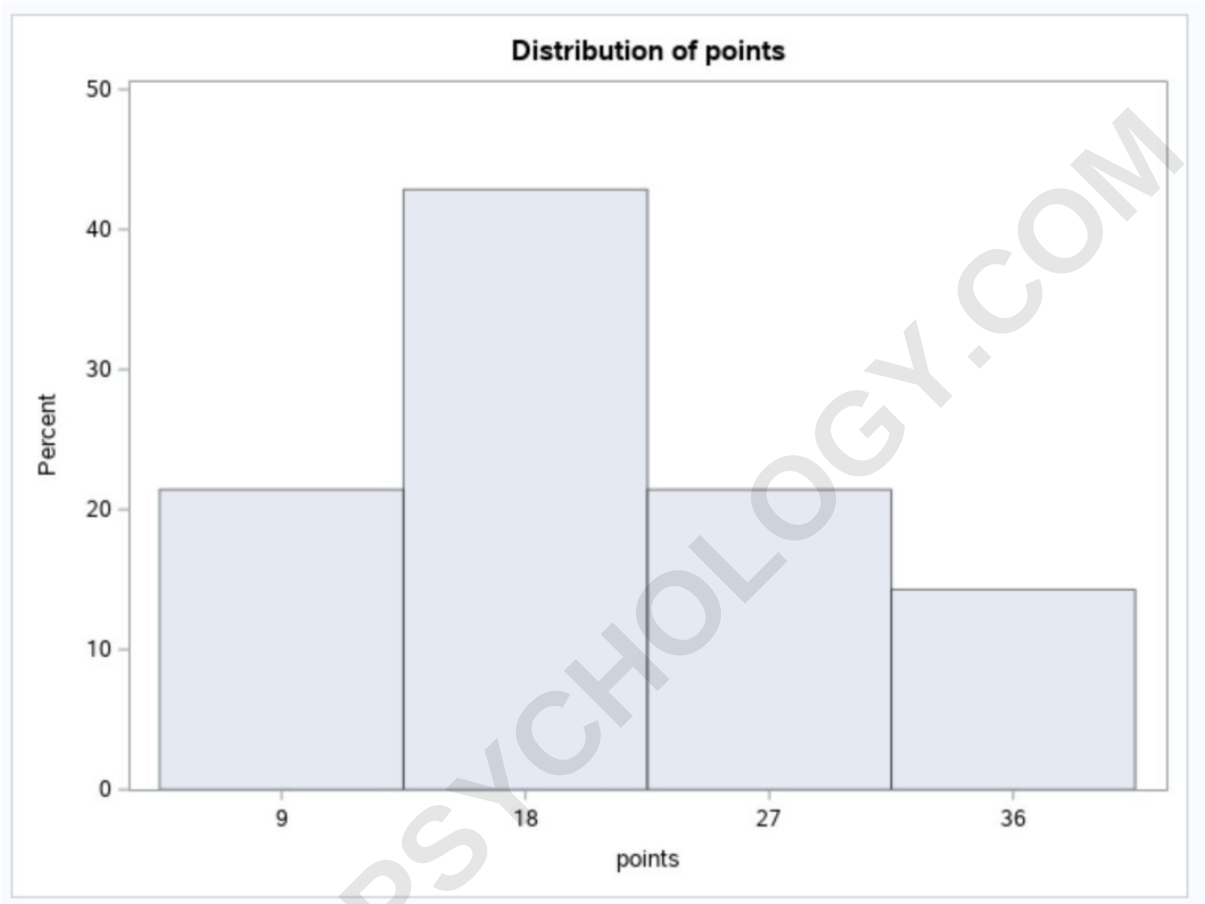
Conversely, there are situations where the goal is to generalize the distribution and smooth out noise or minor fluctuations. This requires decreasing the total number of bins by increasing the interval width. A larger bin width aggregates more observations into fewer groups, which can emphasize the overall trend or shape of the distribution, especially in very large datasets.

For this demonstration, we will significantly increase the interval between midpoints from 6 units (the default view) to **9** units. Setting `BY 9` results in much wider bins that cover a broader range of the `points` variable, thus reducing the total bin count substantially.

This technique is particularly useful in introductory reports or when comparing distributions across different subsets where a high level of detail might be distracting. The resulting histogram offers a high-level summary of the data density:

/*create histogram for points variable with custom bins*/
proc univariate data=my_data;
histogram points / midpoints=(9 to 36 by 9);
run;

Observing the new histogram, it is immediately apparent that there are fewer total bars compared to both the default (interval 6) and the high-precision (interval 3) visualizations. This occurs because the larger interval size means each bin now captures a larger range of values, providing a smoother, more aggregated view of the underlying distribution.



Best Practices for Selecting Histogram Bins

The decision of how many bins to use--or, more accurately in SAS, the interval size for the MIDPOINTS option--is not arbitrary; it significantly impacts the visual interpretation of the data. Too few bins can hide crucial characteristics, potentially making a multimodal distribution look unimodal. Conversely, too many bins can introduce visual noise and make the distribution appear jagged or sparse, especially if the sample size is small.

As a statistical analyst using **SAS**, it is highly recommended to experiment with various bin widths. Start with the default output provided by PROC UNIVARIATE and then incrementally refine the bin width using the **MIDPOINTS=** option. Effective bin selection ensures that the chosen visualization effectively communicates the central tendency, spread, and shape (skewness, modality) of your variable.

Remember that the relationship between the number of bins and the interval width is inverse: increasing the interval width decreases the number of bins, and decreasing the interval width increases the number of bins. Fine-tuning these values is a critical step in producing accurate and informative statistical graphics. We strongly encourage you to iterate through different `start`, `end`, and `interval` parameters in the **MIDPOINTS** statement until the resulting histogram provides the clearest insight into your data's true distribution.

Further SAS Visualization Resources

Mastering histograms is just one aspect of data visualization in the SAS environment. To continue developing your graphical skills, explore the official documentation and related tutorials covering other essential chart types in SAS, such as bar charts, box plots, scatter plots, and more complex graphical outputs generated via ODS Graphics.

The following tutorials explain how to create other charts in SAS: