How to Create a Violin Plot in ggplot2 (With Examples)

Authored by

stats writer

November 22, 2025

Introduction to Violin Plots and ggplot2

The Violin plot is an essential tool in exploratory data analysis, offering a richer understanding of data distribution compared to standard box plots. It combines the traditional summary statistics provided by a box plot with a kernel density trace, illustrating the shape, peaks, and spread of the data. Creating these sophisticated visualizations in ggplot2, the leading data visualization package in R, is a highly structured and efficient process.

Generating a violin plot follows the core principles of the grammar of graphics: mapping data variables to visual **aesthetics** (like position and color), selecting the appropriate geometric layer (the **geom**), and refining the plot with scales and coordinate systems. This approach ensures that complex statistical comparisons are rendered clearly and accurately. The ability to customize these plots--by adding labels, adjusting colors, or layering supplementary statistical markers--allows analysts to produce publication-quality figures tailored to specific analytical needs.

This guide details the steps required to implement violin plots, focusing particularly on how to group data for comparison, how to adjust the plot orientation, and how to integrate vital measures of central tendency, such as the **median**, directly within the visualization for enhanced clarity. Understanding these methods is crucial for effectively communicating distributional properties across multiple categories.

Core Methods for Generating Violin Plots

The foundation of any violin plot in ggplot2 is the `geom_violin()` function. This geometric layer handles the complex calculation of kernel density estimation and renders the resulting shape. Success lies in correctly defining the aesthetic mappings (`aes()`) that link your categorical grouping variable to the X-axis and your continuous value variable to the Y-axis. The methods below demonstrate the versatility of this function for comparative analysis.

You can use the following methods to create a violin plot in ggplot2:

Method 1: Create Violin Plots by Group (Standard Vertical Orientation)

ggplot(df, aes(x=group_var, y=values_var, fill=group_var)) +
geom_violin() +

This standard syntax produces a vertical plot where distributions are displayed along the Y-axis. The `fill=group_var` argument ensures that each category receives a unique color, significantly improving visual discrimination between groups. This is the simplest and most conventional way to compare distributions across distinct categories.

Method 2: Create Horizontal Violin Plots by Group

```
ggplot(df, aes(x=group_var, y=values_var, fill=group_var)) +
geom_violin() +
coord_flip()
```

For scenarios where category labels are long or numerous, flipping the coordinate system can dramatically improve readability. By applying the `coord_flip()` function, the plot retains the same statistical mappings (X maps to group, Y maps to values) but displays the categorical variable vertically and the continuous distribution horizontally. This transformation is particularly useful for visual presentation space optimization.

Method 3: Create Violin Plots by Group and Display Median Value

```
ggplot(df, aes(x=group_var, y=values_var, fill=group_var)) +
geom_violin() +
stat_summary(fun=median, geom='point', size=2)
```

Combining the density visualization with precise summary statistics enhances the analytical depth of the plot. This method utilizes `stat_summary()` to calculate the **median** (the 50th percentile) for each group and renders it using a **point** marker. The explicit inclusion of the median provides a clear visual anchor for comparing the central tendency of the distributions shown in the Violin plot.

Prerequisites and Data Preparation in R

All ggplot2 visualizations require input data to be structured as a well-formed data frame or tibble, containing variables suitable for plotting. Specifically, the violin plot requires at least one categorical variable (for grouping) and one continuous variable (for calculating the density distribution). To demonstrate the examples effectively, we first generate a synthetic dataset in R.

The dataset we create simulates a scenario where performance scores (`points`) are tracked across three competing categories (`team` A, B, and C). Crucially, we use the `rnorm()` function to generate 100 observations for each team, ensuring that Team A is centered around a mean of 10, Team B around 15, and Team C around 20. This deliberate separation in means will ensure the resulting violin plots clearly demonstrate the differences between groups.

Furthermore, we employ `set.seed(1)` to guarantee that every time this code chunk is run, the pseudo-random data generation is identical. This practice of ensuring **reproducibility** is fundamental in statistical programming and is highly recommended for all analytical scripts.

The following examples show how to use each method in practice with the following data frame in

R:

```
#make this example reproducible
set.seed(1)

#create data frame
df <- data.frame(team=rep(c('A', 'B', 'C'), each=100),
points=c(rnorm(100, mean=10),
rnorm(100, mean=15),
rnorm(100, mean=20)))

#view head of data frame
head(df)

team points
1 A 9.373546
2 A 10.183643
3 A 9.164371
4 A 11.595281
5 A 10.329508
6 A 9.179532
```

The resulting data frame, `df`, is perfectly structured for ggplot2 analysis. It includes the categorical variable `team` and the continuous variable `points`, allowing us to proceed directly to visualization using the methods outlined previously.

Note: We used the `set.seed()` function to ensure that this example is reproducible.

Example 1: Visualizing Distribution by Group (Standard Vertical Plot)

The objective of the standard vertical Violin plot is to visually assess the location, spread, and shape of the `points` distribution for each respective `team`. This method provides an intuitive, high-density visualization that reveals characteristics such as skewness or multimodality which might be obscured in simpler box plots.

We initialize the plot by calling `ggplot()`, supplying the data frame `df`, and defining the aesthetic mappings (`aes`). The `team` variable is assigned to the x-axis, the `points` variable to the y-axis, and `team` is also used for the `fill` color. The subsequent addition of the `geom_violin()` layer instructs ggplot2 to calculate and render the kernel density estimate for each group.

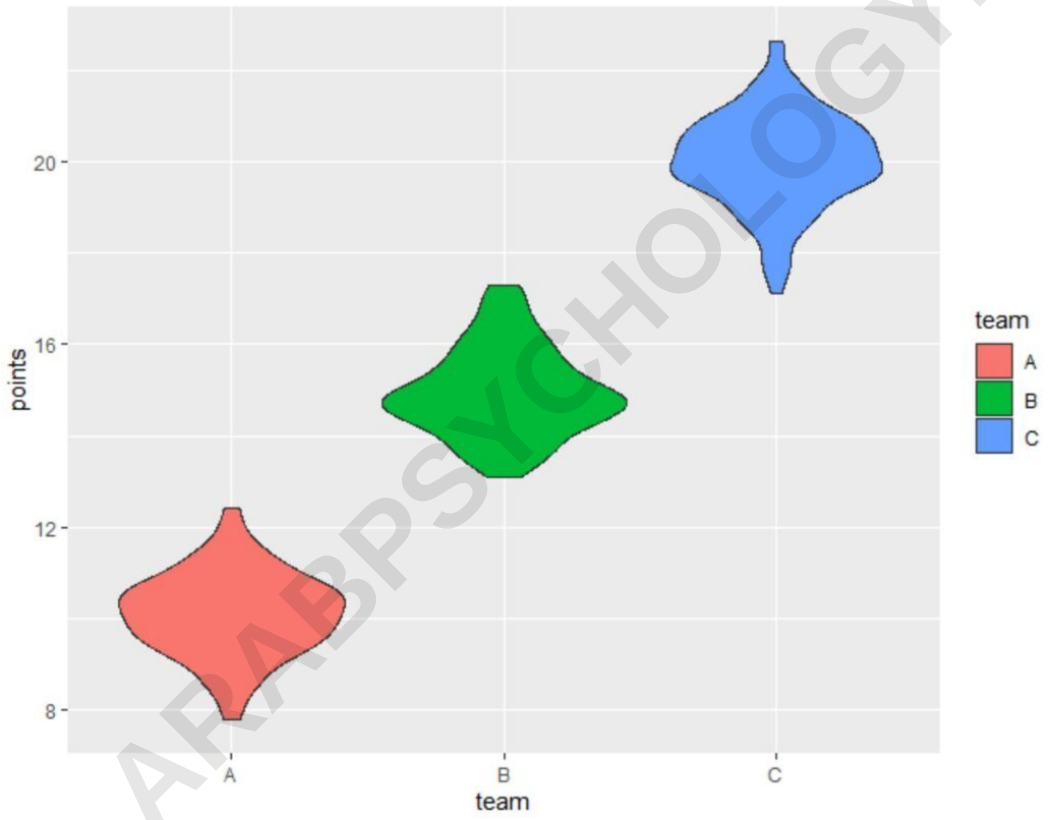
We can use the following syntax to create violin plots that show the distribution of the **points**

variable, grouped by the **team** variable:

```
library(ggplot2)

#create violin plot to visualize distribution of points by team
ggplot(df, aes(x=team, y=points, fill=team)) +
geom_violin()
```

The generated output confirms our data structure: the violin for Team A is centered lower on the y-axis (around 10), Team B is centered higher (around 15), and Team C is highest (around 20). The symmetry and bell shape of the violins indicate that the underlying distributions are approximately normal, consistent with their generation using the `rnorm()` function.



The x-axis displays each team category, and the y-axis displays the distribution of **points** scored by each team. This visualization effectively highlights the shift in group means and verifies the consistency of the variance across all three teams.

Example 2: Creating Horizontal Violin Plots

While the standard vertical plot is widely used, analysts often require horizontal representation for

improved legibility or alignment with reporting standards. The elegance of ggplot2 is that this transformation requires adding only a single command after defining the geometric layers.

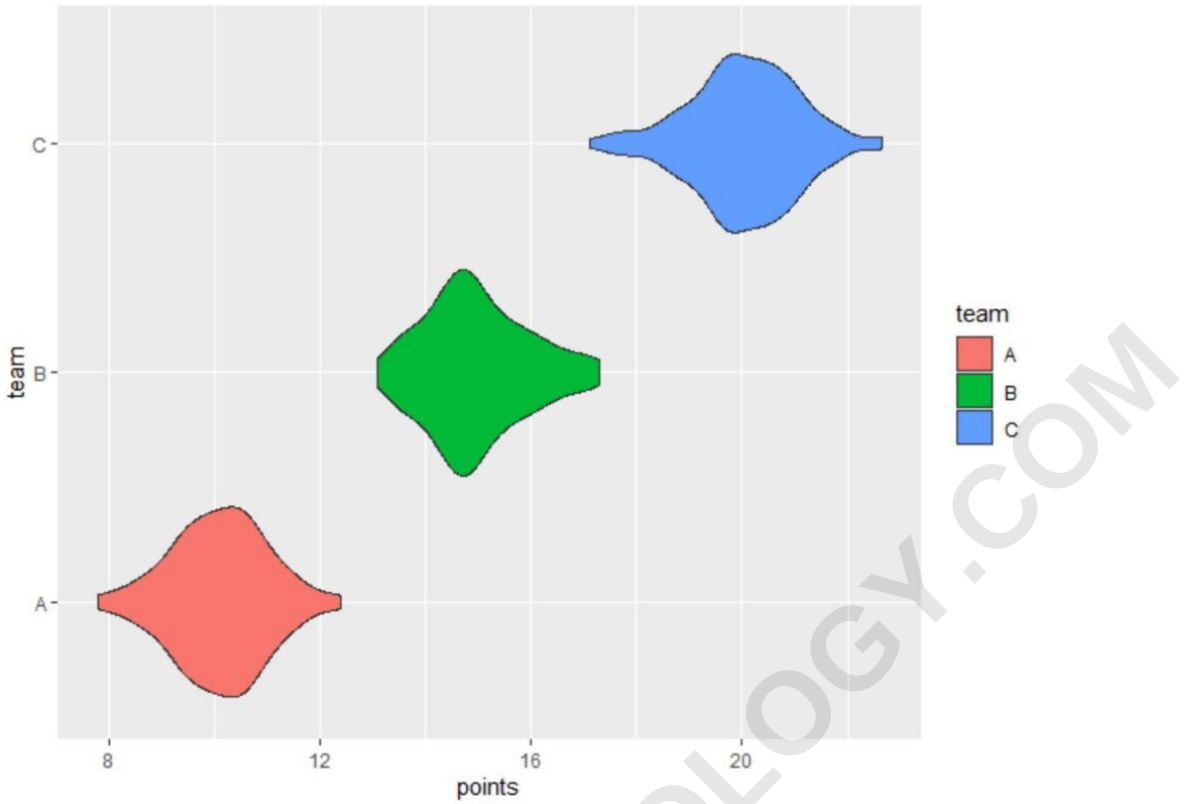
The function responsible for this coordinate system reversal is `coord_flip()`. When appended to the standard plot code, it takes the output and rotates it 90 degrees clockwise. It is important to remember that `coord_flip()` is applied at the end of the plotting chain; the initial mapping (`aes(x=team, y=points)`) remains unchanged, preserving the internal logic of the plot.

To create horizontal violin plots that show the distribution of the **points** variable, grouped by the **team** variable, simply add the **coord_flip()** function:

library(ggplot2)

#create horizontal violin plots to visualize distribution of points by team
ggplot(df, aes(x=team, y=points, fill=team)) +
geom_violin() +
coord_flip()

This modification is purely visual and does not alter the statistical calculations. The resulting plot features the team names vertically stacked, allowing for cleaner display when category labels are lengthy. The distributions themselves extend along the horizontal axis, facilitating comparisons of spread and density along a horizontal plane.



The y-axis now displays each **team**, and the x-axis displays the distribution of **points** scored by each team, providing an alternative, visually impactful way to present the data.

Example 3: Incorporating Measures of Central Tendency (Displaying the Median)

While the smooth density estimate of a violin plot is excellent for showing the overall data shape, viewers often need explicit markers for key summary statistics, such as the median or mean. The `stat_summary()` function in ggplot2 is the ideal tool for layering these calculations onto an existing plot.

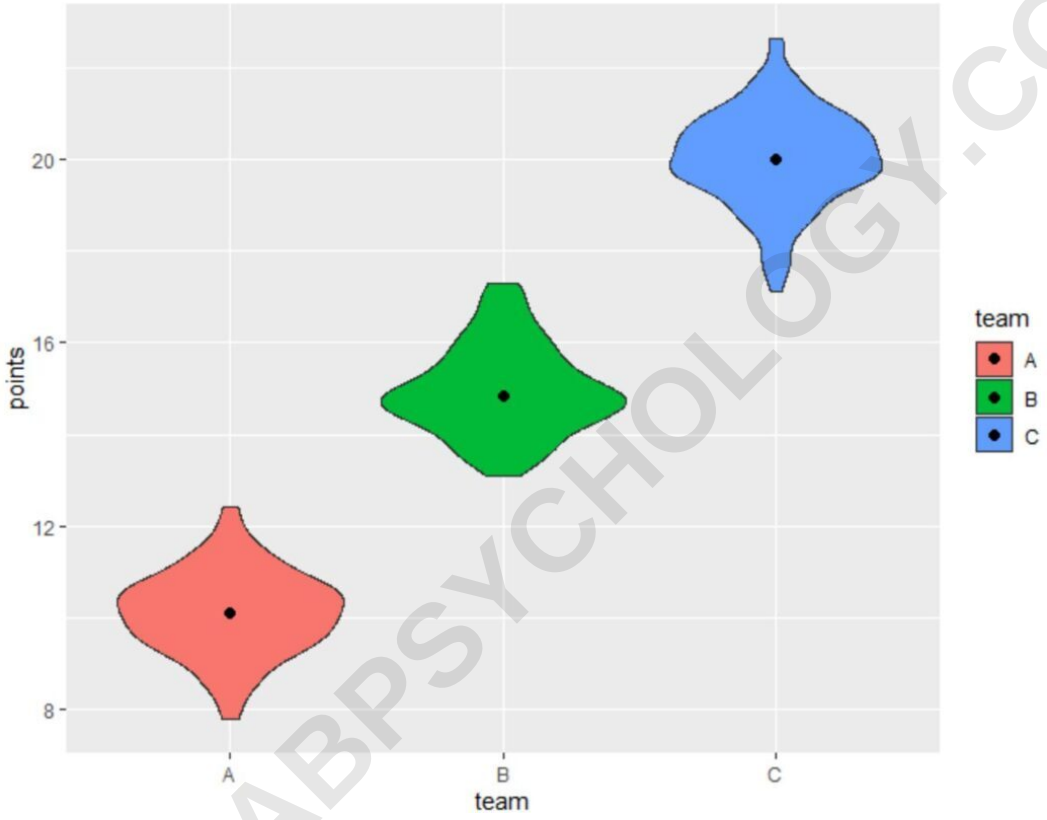
To mark the median point within each violin, we use `stat_summary()`, specifying `fun=median` to define the calculation required. We then specify `geom='point'` to ensure the resulting value is rendered as a simple dot. The `size` parameter is crucial for visibility; setting `size=2` makes the median marker distinct and easy to locate within the wider violin shape.

The following code shows how to create violin plots that show the distribution of the **points** variable, grouped by the **team** variable, with the median points value represented as a circle:

library(ggplot2)

```
#create violin plots and display median points value as circle
ggplot(df, aes(x=team, y=points, fill=team)) +
geom_violin() +
stat_summary(fun=median, geom='point', size=2)
```

This composite visualization provides the best of both worlds: the robust density estimate provided by the geom_violin() layer and the precision of the median marker. For symmetrical distributions like these, the median point will align closely with the widest part of the violin (the mode).



The median points value for each team is clearly represented as a tiny circle inside each violin plot, allowing for rapid comparison of central tendency across the three teams.

Note: To increase the size of the circle, simply increase the value for the **size** argument within the **stat_summary()** function. Customizing marker size is an important step in ensuring accessibility and prominence for key statistical indicators.

Customization and Further Enhancements

The flexibility of ggplot2 allows for deep customization beyond simple color adjustments. For instance, you might want to overlay a narrow `geom_boxplot()` on top of the violin plot to explicitly

show the interquartile range (IQR) and outliers, providing a more comprehensive view of the quartiles alongside the density estimate. Another common refinement is adjusting the **bandwidth** parameter within `geom_violin()`, which controls the smoothness of the density curve and can be necessary when dealing with sparse or highly multimodal data.

Furthermore, controlling the color scheme through `scale_fill_manual()`, adding meaningful titles and axis labels using `labs()`, and defining plot themes using `theme_*()` are essential steps for generating professional, descriptive statistical graphics ready for formal publication or presentation. Mastering these techniques ensures that your violin plots not only display the distribution accurately but also communicate your findings effectively to any audience.

The following tutorials explain how to perform other common tasks in ggplot2: