Learning to Visualize Data: Creating Boxplots with Mean Values in R

Authored by

Mohammed looti

October 29, 2025

Visualizing Data Distribution: Boxplots, Median, and Mean

Effective **statistical analysis** fundamentally relies on powerful visual tools to summarize complex datasets. Among the most popular and informative methods is the **boxplot**, also known as a box-and-whisker plot, which offers a concise graphical representation of numerical data distribution through its **quartiles**. While the primary emphasis of a traditional **boxplot** is the **median**--the 50th percentile--there are many analytical situations where incorporating the **mean** value is essential for a complete understanding of the data's central tendency and symmetry.

The inherent difference between the **median** and the **mean** makes their combined visualization highly valuable. The **median**, represented by the horizontal line inside the box, is a robust statistic, resistant to the distorting effects of extreme values or **outliers** and distribution skewness. Conversely, the **mean** provides the arithmetic average, offering a measure of the expected value, but it is highly sensitive to the presence of these extreme observations.

By plotting both measures of central tendency on the same **boxplot**, data analysts can immediately identify whether the distribution is symmetric or skewed. If the **mean** and **median** are close, the distribution is likely symmetric; a significant divergence suggests skewness, often indicating a non-normal distribution or the presence of influential **outliers**. This article provides a comprehensive guide to implementing this enhanced visualization technique in the **R programming language**, exploring both Base R functionality and the advanced capabilities of the **ggplot2** package.

Setting up the Environment and Sample Data in R

Before diving into the visualization code, proper data preparation is paramount. All plotting methods in **R** require data to be organized in a suitable format, typically a **data frame**. For our demonstration, we will construct a simple, simulated dataset representing performance scores across different groups or 'teams'. This grouped structure is ideal for illustrating how **boxplots** can compare distributions side-by-side while highlighting the specific **mean** for each category.

The following script initializes an R **data frame** named `df`. This structure contains two key variables: the categorical variable `team`, which defines the groups (A, B, C), and the numerical variable `points`, which represents the scores whose distribution we aim to analyze. This setup mirrors typical experimental or observational data where metrics are grouped by distinct factors.

```
# Create a sample data frame for demonstration
df <- data.frame(team=rep(c('A', 'B', 'C'), each=5),
points=c(4, 4, 5, 6, 8, 7, 6, 8, 9, 12,
11, 12, 13, 16, 18))
```

# Display the first few rows of the data frame to verify its structure
head(df)

team points
1 A 4
2 A 4
3 A 5
4 A 6
5 A 8
6 B 7

With the `df` **data frame** successfully created, we have the necessary input for generating our visual comparisons. The subsequent methods will focus on calculating the average `points` per `team` and integrating this statistic visually with the standard **boxplot** representation, ensuring both the **median** and R-calculated averages are presented clearly.

Approach 1: Overlaying Mean Values Using Base R Graphics

The foundational plotting system in Base **R** offers a direct and efficient way to produce visualizations without relying on external libraries. To add custom elements, such as group means, to a `boxplot()`, we utilize a layering technique common in Base R graphics. This approach involves two distinct steps: first, drawing the primary graphical element (the boxplots), and second, using auxiliary functions to overlay the derived summary statistics.

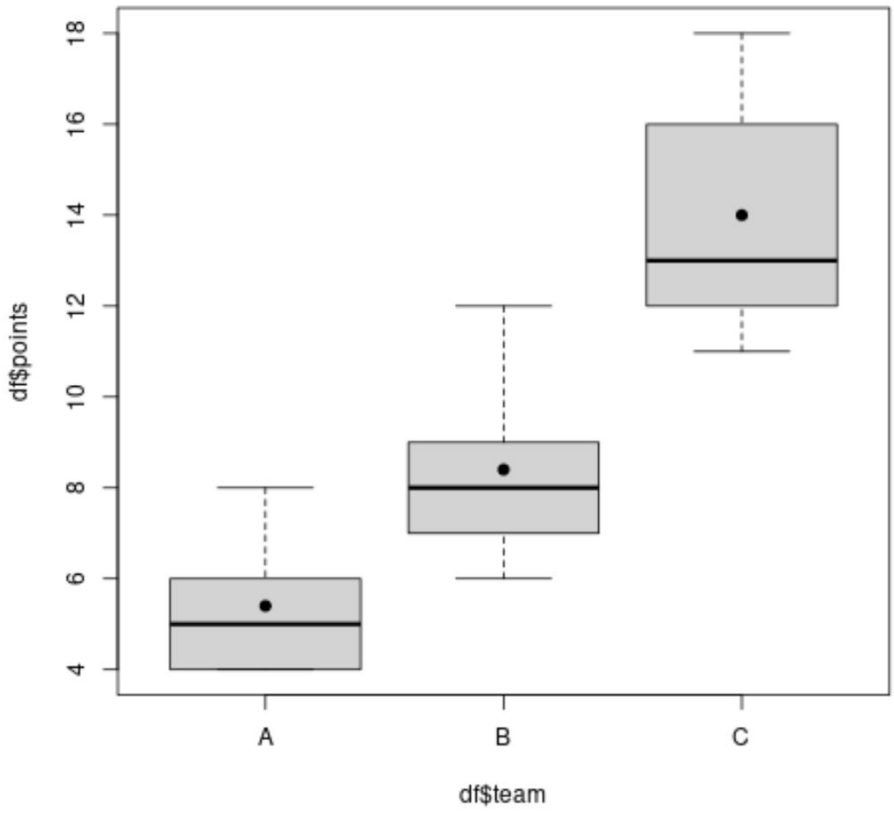
The initial command, `boxplot(df$points~df$team)`, quickly generates the standard boxplots for the `points` variable, grouped by `team`. The crucial step that follows is calculating the group means. We achieve this efficiently using the `tapply()` function, which applies a specified function (in this case, `mean`) to a ragged array, returning the mean score for each distinct team category. This separation of calculation and visualization provides maximum control over the data presentation.

Finally, the calculated means are mapped onto the existing plot using the **`points()`** function, which accepts the vector of calculated means (`means`) and plots them as coordinates corresponding to the boxplot positions. We specify markers using arguments like `pch` (plotting character) and `cex` (character expansion) to ensure the mean markers are distinct and easily visible. This process results in a clear graphical comparison between the median (the line) and the mean (the overlaid marker).

 # Create boxplots for 'points' grouped by 'team'
boxplot(df$points~df$team)

```
# Calculate mean value for 'points' for each 'team' group
means <- tapply(df$points, df$team, mean)

# Add these mean values as solid circles (pch=20) to each boxplot, with increased size (cex=1.5)
points(means, pch=20, cex=1.5)
```



In the resultant Base **R** visualization, the internal horizontal line denotes the **median** score for each team. The prominent black circles, controlled by the `pch=20` and `cex=1.5` settings, distinctly represent the calculated group means. This method provides an effective visual audit, allowing analysts to gauge the consistency between the median and mean at a glance.

Approach 2: Leveraging stat_summary() in ggplot2

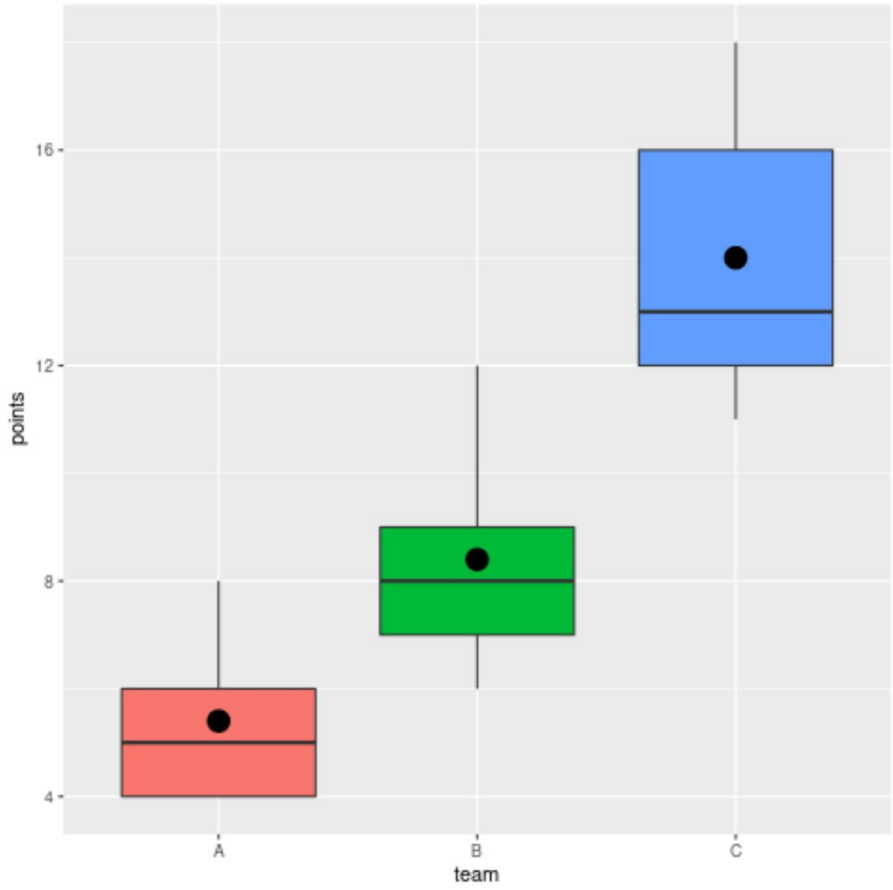
For those prioritizing highly polished, publication-ready graphics and deeper customization capabilities, the **ggplot2** package--a core component of the **tidyverse**--is the preferred visualization framework in R. **ggplot2** operates on the principles of the Grammar of Graphics, allowing visualizations to be built incrementally by adding layers (geoms) and statistical transformations (stats). Integrating the **mean** value into a **boxplot** is handled elegantly and directly within this layered framework.

We initiate the plot using `ggplot()`, defining the data source and the core aesthetic mappings (x, y, and optionally, fill for color differentiation). The initial layer is `geom_boxplot()`, which draws the standard box-and-whisker plot elements. To introduce the mean, we utilize the highly flexible **`stat_summary()`** function. Instead of calculating the means separately, as required in Base R, **ggplot2** handles the computation internally within the plotting pipeline.

By setting the arguments within **`stat_summary()`** to `fun=mean` and `geom='point'`, we explicitly instruct the package to calculate the mean for each group and display that result as a point on the graph. Customizing the appearance of these points, such as setting the `shape` and `size` parameters, ensures the mean marker is visually distinct from the rest of the boxplot features, leading to a professional and clear visualization.

library(ggplot2)

```
# Create boxplots for 'points' grouped by 'team', with 'team' mapping to fill color
ggplot(df, aes(x=team, y=points, fill=team)) +
# Add the boxplot layer
geom_boxplot() +
# Add a summary statistic layer to display the mean as a point (shape 20, size 8)
stat_summary(fun=mean, geom='point', shape=20, size=8) +
# Remove the legend for cleaner visualization
theme(legend.position='none')
```



The resulting **ggplot2** output offers superior aesthetic quality. The **mean** for each team is clearly marked by the large, solid black circle, contrasted against the central **median** line and the colored **boxplot**. The ability to control fill color by group and easily manage thematic elements (like removing the legend) makes this method highly efficient for generating graphical results suitable for formal reports and publications.

Choosing the Right Tool: Base R vs. ggplot2

Both the Base R and the **ggplot2** methods effectively achieve the goal of visualizing boxplots with overlaid mean values, yet they cater to different needs and user preferences within the R environment. Understanding these differences is key to selecting the most appropriate tool for a given analytical task or project workflow.

Base **R** excels in simplicity and speed. It is ideal for quick exploratory data analysis (EDA) where the primary focus is rapid visualization without extensive aesthetic customization. The syntax for Base R plotting is often more compact for fundamental graphs, and crucially, it avoids the dependency overhead of loading external packages. However, achieving complex, multi-layered plots or fine-tuning elements like axis labels, colors, and themes often requires more verbose and

less intuitive code manipulation compared to the structured environment of ggplot2.

In contrast, **ggplot2**, guided by the robust Grammar of Graphics, provides unparalleled control over every element of the plot. While it has a slightly higher initial learning curve and requires loading the library, its consistent layering system ensures that complex visualizations--which might involve adding confidence intervals, multiple statistical summaries, or custom themes--are built logically and predictably. For creating graphics destined for publication, presentations, or detailed data storytelling, ggplot2's aesthetic capabilities and flexibility generally make it the superior choice.

Ultimately, the decision between Base R and **ggplot2** rests on context: if rapid visualization and minimal dependencies are critical, Base R is sufficient. If high-quality aesthetics, precise control, and the potential for future graphical complexity are necessary, ggplot2 provides the most powerful and scalable solution for incorporating statistical summaries like the **mean** onto boxplots.

Summary and Next Steps in Data Visualization

This guide demonstrated two powerful and distinct methodologies within the **R programming language** ecosystem for enhancing **boxplots** by incorporating the **mean** value alongside the **median**. This combined visualization technique is fundamental for robust statistical interpretation, particularly when analyzing the characteristics of grouped distributions, identifying potential skewness, or assessing the influence of extreme values.

By mastering both the Base R approach (using `boxplot()` and `points()`) and the **ggplot2** method (using `geom_boxplot()` and `stat_summary()`), you gain valuable flexibility in your analytical toolkit. Remember that while the **median** offers a location estimate resistant to **outliers**, the **mean** provides the arithmetic average, and their visual relationship offers immediate diagnostic information about the underlying data shape.

We strongly encourage users to apply these methods to their own datasets. Experimenting with different plotting character types, colors, and sizes (using `cex` in Base R or `size` in ggplot2) will help tailor the visualizations to best communicate your specific analytical findings. Continued practice in R visualization ensures that complex data stories are conveyed clearly and compellingly.

Additional Resources

To further refine your skills in creating and customizing **boxplots** and other essential visualizations in R, consult the following authoritative resources:

Boxplots in R (Quick-R)

R Boxplot Tutorial (Data Mentor)
Adding Mean to Boxplots (R Graphics Cookbook)

We hope this detailed guidance proves invaluable in expanding your data visualization capabilities.