Extending ggplot2 statistical geometries

Evangeline Reynolds [*]
Dean Data Cell, West Point
and
Author 2
Department of ZZZ, University of WWW

June 25, 2021

Abstract

Ggplot2, the implementation of the grammar of graphics in the R statistical programming language, is a popular open source project. The package's interface lets creators separate data visualization concerns — including declaration of data, variable representations by visual channels, determination of coordinate systems, selection of geometric shapes taking on the aesthetic representation — which means users have great freedom in the creation and customization of charts. Ggplot2 comes with a large number of geometric shapes that can represent variables in data sets. Some of these shapes are drawn after statistical transformation, such as boxplots, linear regressions, or histograms. But many statistical concepts do not have easy-to-use geometries for representing statistical summaries. This paper presents a new extension package `ggxmean`, that include new `geoms` useful for easily visualizing an important set of additional statistical concepts.

Keywords: 3 to 6 keywords, that do not appear in the title

[*]The authors gratefully acknowledge . . .

1 Introduction

The grammar of graphics framework, proposed by Leland Wilkinson in 1999, that identified 'seven orthogonal components' in the creation of data visualizations.
Wilkinson asserted that if data visualization packages were created using a separation of concerns approach – dividing decision making surrounding these components — the packages would be able to "draw every statistical graphic". The grammar of graphic principles were incredibly powerful and gave rise to a number of visualization platforms including Tableau, vega-lite, and ggplot2.

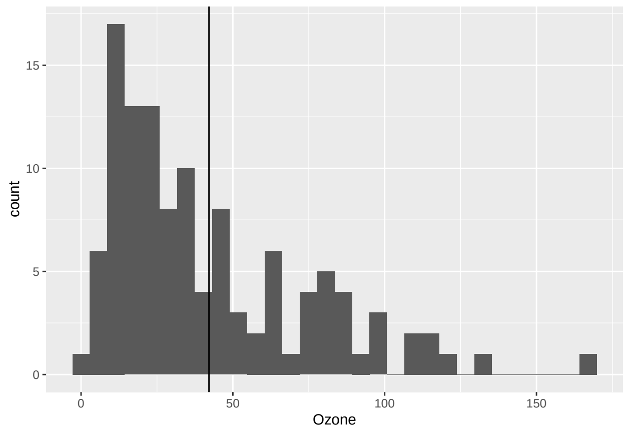
Statistical educators that introduce students to one of these tools arguably are doing more than constructing one-off plots to discuss statistical principles with students: they are introducing students to 'a powerful way of thinking about data visualization'.

Statistical educators often use ggplot2 as their grammar-of-graphics-based data visualization tool as students can learn it along side the rich statistical ecosystem of the R programming language. The R programming language thus may serve as a one-stop-shop for statistical tooling; with recent developments in packages and IDEs writing code is becoming more accessible and welcoming to newcomers.

Still, using ggplot2 for statistical education can be a challenge at times. When it is used to discuss statistical concepts, sometimes it feels as though getting something done with the plotting library derails the focus on discussion of statistical concepts and material.

1.1 The status quo: adding the mean

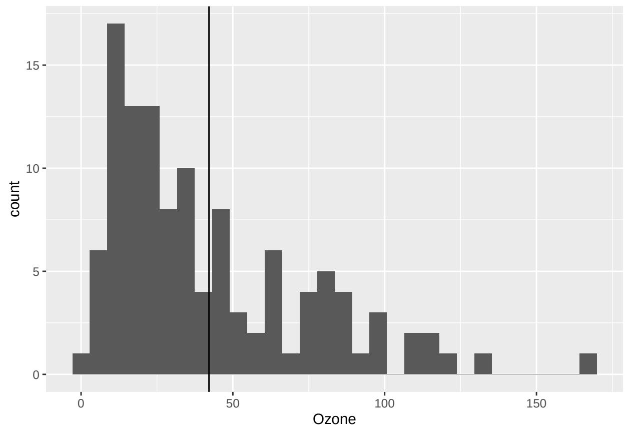
Consider for example, a the seemingly simple enterprise of adding a vertical line at the mean of x, perhaps atop a histogram or density plot.



Creating this plot requires greater focus on ggplot2 *syntax*, likely detracting from discussion of *the mean* that statistical instructors desire.

A syntax that works is as follows:

```
library(tidyverse)
library(magrittr)

ggplot(airquality) +
  aes(x = Ozone) +
  geom_histogram() +
  geom_vline(xintercept =
               mean(airquality$Ozone,
                    na.rm = T))
```



It may require a discussion about dollar sign syntax and how geom_vline is actually a special geom – an annotation – rather than being mapped to the data. None of this is relevant to the point you as an instructor aim to make: maybe that the the mean is the balancing point of the data or maybe a comment about skewness.

Further, for the case of adding a vertical line at the mean for different subsets of the data, a different approach is required. This enterprise may take instructor/analyst/student on an even larger detour – possibly googling, and maybe landing on the following stack overflow page where 11,000 analytics souls (some repeats to be sure) have landed:

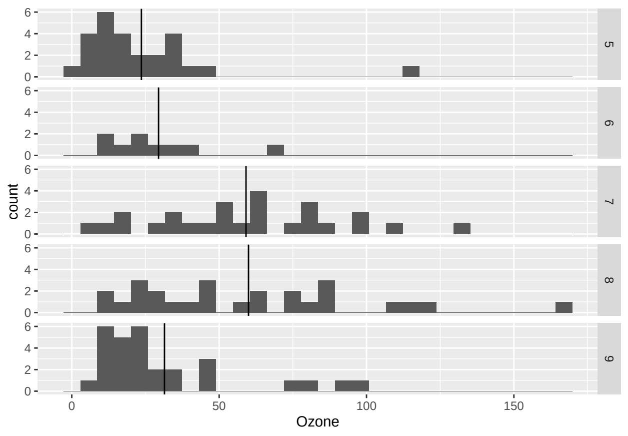
`https://stackoverflow.com/questions/1644661/add-a-vertical-line-with-different-intercept-for-each-panel-in-ggplot2`

The solutions to this problem involve data manipulation *prior* to plotting the data. The solution disrupts the forward flow ggplot build. One must take a pause, which may involve toggling back and forth between stack overflow solutions, disrupting momentum you are working on to talk about the pooled mean and the conditional mean.

"Learning becomes less efficient as the mental load students must carry increases." (Lovett & Greenhouse 2000)

Wanda vision "oh no" `https://media.giphy.com/media/ycBW5N5XMfpXTvamtF/giphy.gif`

```
airquality %>%
  group_by(Month) %>%
  summarise(Ozone_mean = mean(Ozone, na.rm = T)) ->
airquality_by_month


ggplot(airquality) +
  aes(x = Ozone) +
  geom_histogram() +
  facet_grid(rows = vars(Month)) +
  geom_vline(data = airquality_by_month,
             aes(xintercept =
               Ozone_mean))
```



1.2 The promise

But, using statistical geometries *can* feel simple. geom_smooth() adds loess fit of data to visualizations in an effortless way; and an ordinary least squares fit is added almost as easily. geom_boxplot reveals a 5-statistic summary of data medians, the inner quartile range and min/max values in a single call. The popularity of these easy-to-use functions is evident. geom_smooth is used more than 82,000 times in .R and .Rmd files on GitHub and geom_boxplot 93,000 times in .R and .Rmd files.

With these geoms one says "let there be boxplots" and there are boxplots, and "let there be loess smoothing" and there is loess smoothing. The same powerful, declarative experience can be arranged for more statistical concepts.

1.3 Why some statistical geoms exist and others don't

ggplot2 developers and maintainers, with the aim of keeping the code base robust and reliable, have intentionally limited the out-of-the-box geoms (as well as other variants) delivered in base ggplot2.

"ggplot2 is already pretty big … and it's simply not feasible for us to include all visualization possibilities into ggplot2 itself." - Thomas Pederson `https://www.rstudio.com/resources/rstudioconf-2020/extending-your-ability-to-extend-ggplot2/`

But the extension system is designed to allow for more possibilities. "It's much better to … spread it out on multiple maintainers, multiple specific packages and so on."

`https://media.giphy.com/media/relSRdIMtKVcPrSfyT/giphy.gif`

Extending ggplot statistical geometries will allow us as teachers, analysts and students the ease of visual representation for a large number of other desirable statistical summaries as is experienced with geom_boxplot and geom_smooth.

1.4 introducing {ggxmean}

The development package ggxmean (`https://github.com/EvaMaeRey/ggxmean`) is aimed at providing more statistical geometries that are practical for statistical educators and analysts alike.

First, I present the geom_x_mean() function, which can be used in place of the above solutions to drawing a vertical line at the mean at x or the conditional mean of x. Note: functions from the development package package are indicated with a comment.

The function geom_x_mean is used in a declarative way. After building a plot showing the distribution of parts per billion of ozone in NYC, you can add a vertical line at the mean with geom_x_mean(). The mean is computed for the user automatically in the background.

Additionally, the conditional means are computed based on grouping variables with geom_x_mean. Below, a second version of the plot is also produced, where the mean of x computed for subsets of the data – faceting by month. The group-wise computational of each mean is managed in the background by the geom_x_mean function.

```r
library(ggplot2)
library(ggxmean)
ggplot(airquality) +
  aes(x = Ozone) +
  geom_histogram() +
  geom_x_mean() # ggxmean
```

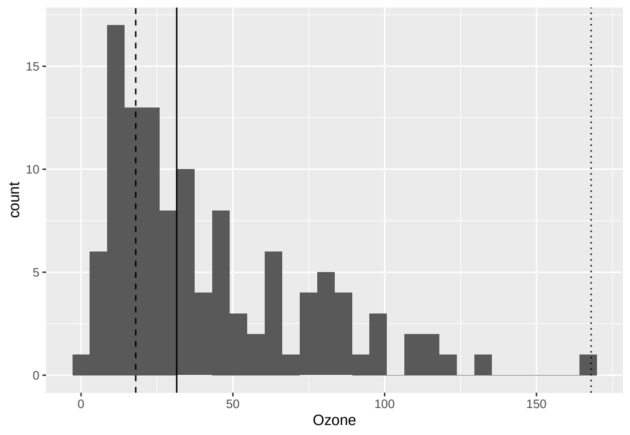
```r
last_plot() +
  facet_grid(rows = vars(Month))
```

1.5 Other statistical geometries in the {ggxmean} package

Beyond providing an easy way to add vertical line at the mean – and conditional means, {ggxmean} provides a number of other handy geometries

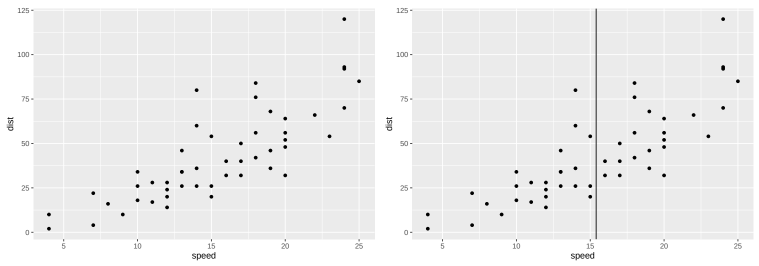
1.5.1 Other univariate markers

```
ggplot(airquality) +
  aes(x = Ozone) +
  geom_histogram() +
  geom_x_median() +
  geom_x_quantile(quantile = .25,
                  linetype = "dashed") +
  geom_x_percentile(percentile = 100,
                    linetype = "dotted")
```



1.5.2 Related to the linear model

```
ggplot(data = cars) +
  aes(speed, dist) +
  geom_point() + #BREAK
  ggxmean::geom_lm() + #BREAK
  ggxmean::geom_lm_fitted(color = "blue",
                                      size = 3) + #BREAK
  ggxmean::geom_lm_residuals() + #BREAK
  ggxmean::geom_lm_conf_int() + #BREAK
  ggxmean::geom_lm_intercept(color = "red",
                                         size = 5) + #BREAK
  ggxmean::geom_lm_formula(size = 10) #BREAK
```



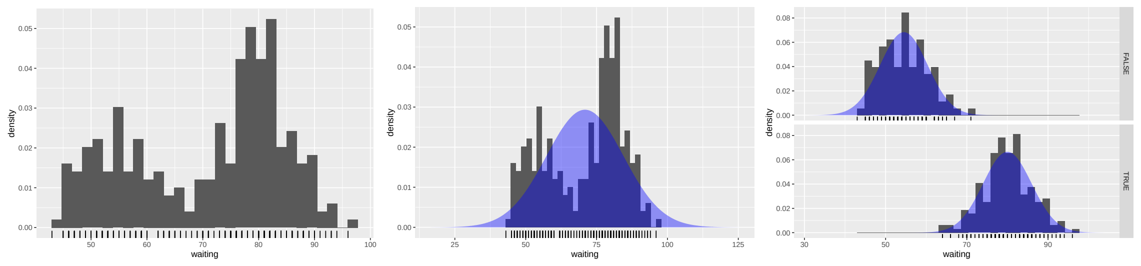
1.5.3 Related to covariance

return to this section

```
ggplot(data = cars) +
  aes(speed, dist) +
  geom_point() + #BREAK
  geom_x_mean()
```

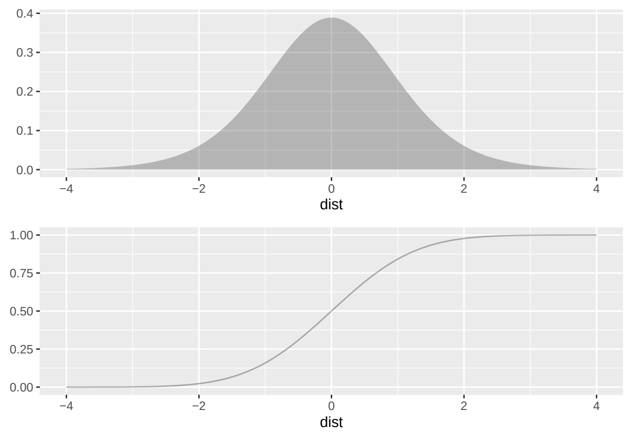
1.5.4 fitting distributions

```
ggplot(data = faithful) +
  aes(waiting) +
  geom_rug() +
  geom_histogram(aes(y = ..density..)) + #BREAK
  ggxmean::geom_normal_dist(fill = "blue") + #BREAK
  facet_grid(rows = vars(eruptions > 3)) #BREAK
```

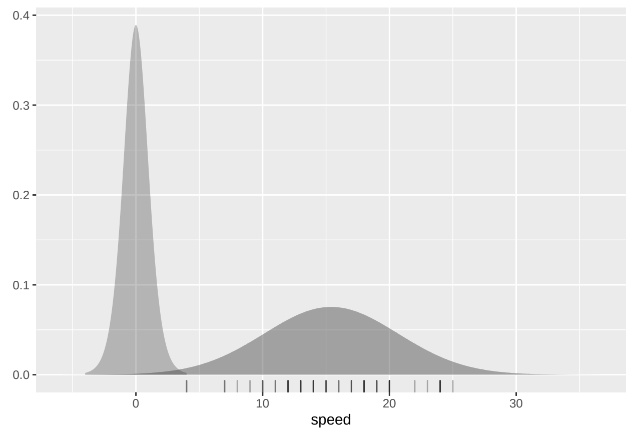


1.6 Annotation: Stamps

```
library(patchwork)
(ggplot(data = cars) +
  aes(dist) +
  ggxmean::stamp_normal_dist() ) /
(ggplot(data = cars) +
  aes(dist) +
  ggxmean::stamp_normal_prob())
```

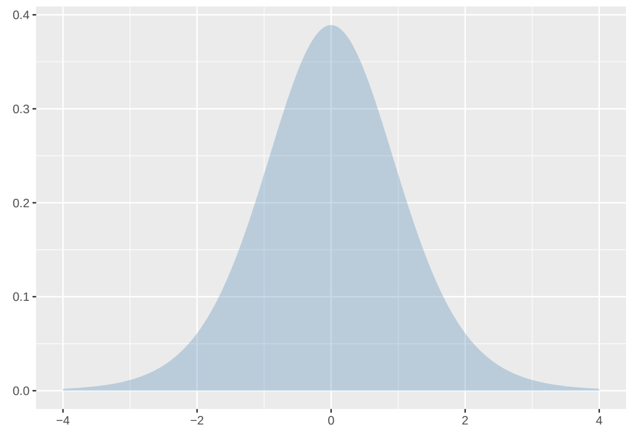


```
ggplot(cars, aes(x = speed)) +
  geom_rug(alpha = .3) +
  geom_normal_dist() +
  stamp_normal_dist()
```

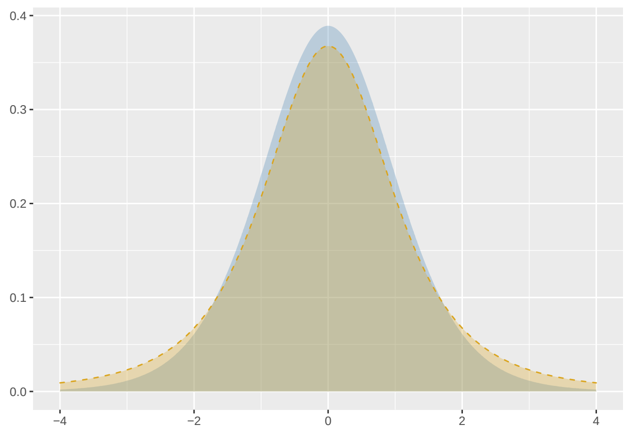


1.6.1 stamp space?

```
ggxmean:::stamp_space() +
  stamp_normal_dist(fill = "steelblue")
```

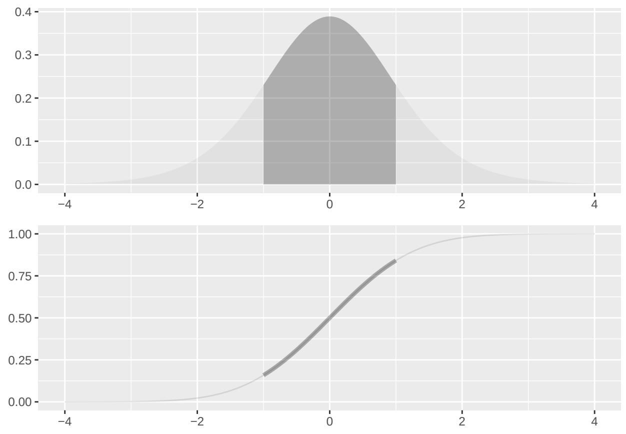


```
last_plot() +
  stamp_t_dist(df = 3,
               fill = "goldenrod",
               linetype = "dashed",
               color = "goldenrod",
               outline.type = "upper")
```

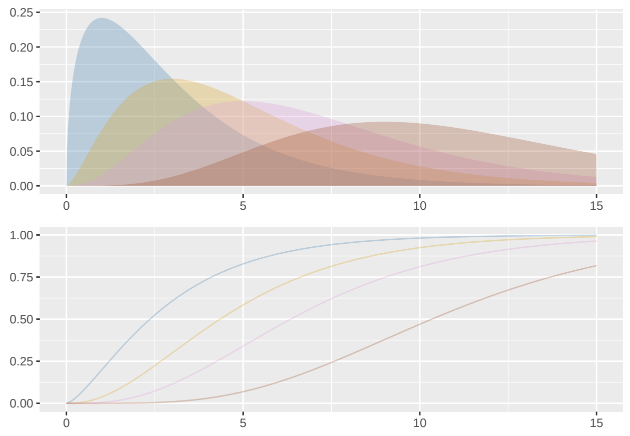


```
library(patchwork)
library(ggplot2)
library(ggxmean)
ggxmean:::stamp_space() +
  stamp_normal_dist()


ggxmean:::stamp_space() +
  stamp_chi_dist() +
  stamp_t_dist()


ggxmean:::stamp_space() +
  ggxmean::stamp_normal_dist(x_min = -1, x_max = 1, color = "black", outline.type = "full
  ggxmean::stamp_normal_dist(x_min = -2, x_max = 2, color = "black", outline.type = "full
  ggxmean::stamp_normal_dist(x_min = -3, x_max = 3, color = "black", outline.type = "full
  ggxmean::stamp_normal_dist(x_min = -4, x_max = 4, color = "black", outline.type = "full
  ggxmean::stamp_normal_dist(x_min = -5, x_max = 5, color = "black", outline.type = "full
  theme_minimal()
```

```
library(patchwork)


ggxmean:::stamp_space() +
  stamp_normal_dist(alpha = .05) +
  stamp_normal_dist(x_min = -1, x_max = 1) ->
p1

  ggxmean:::stamp_space() +
  stamp_normal_prob(alpha = .1) +
  stamp_normal_prob(sd_min = -1, sd_max = 1,
                                     size = 1.5) ->
p2

p1 / p2
```



```
library(patchwork)

ggxmean:::stamp_space() +
  stamp_chisq_dist(df = 3,  fill = "steelblue") +
  stamp_chisq_dist(df = 5,  fill = "goldenrod") +
  stamp_chisq_dist(df = 7,  fill = "plum") +
  stamp_chisq_dist(df = 11, fill = "sienna") ->
g1

ggxmean:::stamp_space() +
  stamp_chisq_prob(df = 3,  color = "steelblue") +
  stamp_chisq_prob(df = 5,  color = "goldenrod") +
  stamp_chisq_prob(df = 7,  color = "plum") +
  stamp_chisq_prob(df = 11, color = "sienna") ->
g2

g1 / g2
```



2 Verifications

This section will be just long enough to illustrate what a full page of text looks like, for margins and spacing.

(Tishkovskaya & Lancaster 2012)

References

Lovett, M. C. & Greenhouse, J. B. (2000), ‘Applying cognitive theory to statistics instruction’, *The American Statistician* **54**(3), 196–206.

Tishkovskaya, S. & Lancaster, G. A. (2012), ‘Statistical education in the 21st century: A review of challenges, teaching innovations and strategies for reform’, *Journal of Statistics Education* **20**(2).