Telling Stories With Data: Comparing Program Outcomes with ggplot2

Andy Grogan-Kaylor
2019-07-19

Contents

1 Background 2
2 Load the Simulated Social Service Agency Data 2
3 Load the Libraries 2
4 First Approach (x is program; y is mental health) 3
5 Add Geometries That Show The Average 3
5.1 Bar Chart 3
5.2 Horizontal Bar Chart 3
5.3 Point Chart 3
5.4 "Lollipop" Chart 4
5.5 Line Chart 4
6 Add Geometries That Show the Distribution 4
6.1 Boxplot 4
6.2 Violin Plot 4
6.3 Points 4
6.4 Jittered Points 5
6.5 Beeswarm Plot 5
7 Second Approach (x is mental health; facet wrap on program) 5
8 Add Geometries 5
8.1 Histogram 5
8.2 Density 5
9 Third Approach (x is mental health; transparent geometries) 5
10 Add Geometries 6
10.1 Histogram 6
10.2 Density 6
1 Background

ggplot2 is a powerful graphing library that can make beautiful graphs. 
ggplot2 can also help us to understand ideas of an underlying "grammar 
of graphics".

However, ggplot can be difficult to learn. I am thinking that one way to 
better understand ggplot might be to see how this graphing library could be 
applied to a concrete example of comparing program outcomes.

In this example, program is a factor and outcome is numeric.

2 Load the Simulated Social Service Agency Data

load("social_service_agency.RData")  # simulated data

Table 1: Table continues below

<table>
<thead>
<tr>
<th>ID</th>
<th>age</th>
<th>gender</th>
<th>program</th>
<th>mental_health_T</th>
</tr>
</thead>
<tbody>
<tr>
<td>4746</td>
<td>26.79</td>
<td>Male</td>
<td>Program B</td>
<td>97.53</td>
</tr>
<tr>
<td>3471</td>
<td>24.86</td>
<td>Male</td>
<td>Program B</td>
<td>82.72</td>
</tr>
<tr>
<td>4343</td>
<td>24.47</td>
<td>Male</td>
<td>Program C</td>
<td>101.2</td>
</tr>
<tr>
<td>3566</td>
<td>23.53</td>
<td>Female</td>
<td>Program C</td>
<td>92.74</td>
</tr>
<tr>
<td>2082</td>
<td>18.71</td>
<td>Male</td>
<td>Program C</td>
<td>87.08</td>
</tr>
<tr>
<td>3963</td>
<td>29.95</td>
<td>Other Identity</td>
<td>Program C</td>
<td>97.98</td>
</tr>
</tbody>
</table>

mental_health_T2  latitude  longitude

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>107.2</td>
<td>42.13</td>
<td>-83.67</td>
</tr>
<tr>
<td>103.9</td>
<td>42.05</td>
<td>-83.8</td>
</tr>
<tr>
<td>94.14</td>
<td>42.25</td>
<td>-83.63</td>
</tr>
<tr>
<td>103.4</td>
<td>42.11</td>
<td>-83.75</td>
</tr>
<tr>
<td>96.56</td>
<td>42.1</td>
<td>-83.62</td>
</tr>
<tr>
<td>92.21</td>
<td>42.34</td>
<td>-83.82</td>
</tr>
</tbody>
</table>

3 Load the Libraries

library(ggplot2)  # beautiful graphs

library(ggthemes)  # beautiful themes
4 First Approach (x is program; y is mental health)

There is a lot of code below. This is where we are setting up the grammatical logic of the graphing approach.

Devoting some time to setting up the initial logic of the plot will pay dividends in terms of exploring multiple geometries later on.

Note that I am adding optional scale... and theme... arguments just to make the graphs look a little nicer, but these are not an essential part of the code.

myplot1 <- ggplot(clients, # the data I am using
  aes(x = program, # x is program
      y = mental_health_T2, # y is mental health
      color = program, # color is also program
      fill = program)) + # fill is also program
  labs(y = "mental health at time 2") + # labels
  scale_color_viridis_d() + # beautiful colors
  scale_fill_viridis_d() + # beautiful fills
  theme_minimal() + # minimal theme
  theme(axis.text.x = element_text(size = rel(.5))) # smaller labels

5 Add Geometries That Show The Average

Now that we have devoted a lot of code to setting up the grammar of the graph, it is a relatively simple matter to try out different geometries. The geometries show the average value.

5.1 Bar Chart

myplot1 + stat_summary(fun.y = "mean", geom = "bar")

5.2 Horizontal Bar Chart

myplot1 + stat_summary(fun.y = "mean", geom = "bar") +
  coord_flip()

5.3 Point Chart

myplot1 + stat_summary(fun.y = "mean", geom = "point", size = 5)
5.4 “Lollipop” Chart

The segments connecting the x axis with the points, require their own geometry that has its own aesthetic.

\[
\text{myplot1 + stat_summary(fun.y = "mean", geom = "point", size = 5) + geom_segment(aes(x = program, xend = program, y = 0, yend = mean(mental_health_T2)))}
\]

5.5 Line Chart

An extra element of the aesthetic is required for lines.

\[
\text{myplot1 + stat_summary(aes(group = 1), color = "black", # consistent color fun.y = "mean", geom = "line")}
\]

6 Add Geometries That Show the Distribution

Now that we have devoted a lot of code to setting up the grammar of the graph, it is a relatively simple matter to try out different geometries. The geometries show the distribution of all values.

6.1 Boxplot

\[
\text{myplot1 + geom_boxplot(fill = "white")}
\]

6.2 Violin Plot

\[
\text{myplot1 + geom_violin()}
\]

6.3 Points

\[
\text{myplot1 + geom_point()}
\]
6.4 Jittered Points

\[
\text{myplot1 + geom\_jitter()}
\]

6.5 Beeswarm Plot

\[
\text{library(ggbeeswarm) \# beeswarm geometry}
\]

\[
\text{myplot1 + geom\_beeswarm()}
\]

7 Second Approach (x is mental health; facet wrap on program)

Again, there is a lot of code below. This is where we are setting up the grammatical logic of the graphing approach.

\[
\text{myplot2 <- ggplot(clients, \# the data I am using}\n\text{  aes(x = mental\_health\_T2, \# x is mental health}\n\text{  fill = program)) + \# fill is program}\n\text{  facet\_wrap(~program) + \# facet on this variable}\n\text{  labs(x = "mental health at time 2") + \# labels}\n\text{  scale\_color\_viridis\_d() + \# beautiful colors}\n\text{  scale\_fill\_viridis\_d() + \# beautiful fills}\n\text{  theme\_bw() \# bw theme makes facets more clear}
\]

8 Add Geometries

However, now that we have devoted a lot of code to setting up the grammar of the graph, it is again a relatively simple matter to try out different geometries.

8.1 Histogram

\[
\text{myplot2 + geom\_histogram()}
\]

8.2 Density

\[
\text{myplot2 + geom\_density()}
\]

9 Third Approach (x is mental health; transparent geometries)

One last time, there is a lot of code below. This is where we are setting up the grammatical logic of the graphing approach.
myplot3 <- ggplot(clients, # the data I am using
    aes(x = mental_health_T2, # x is mental health
        fill = program)) + # fill is program
    labs(x = "mental health at time 2") + # labels
    scale_color_viridis_d() + # beautiful colors
    scale_fill_viridis_d() + # beautiful fills
    theme_minimal() # minimal theme

10 Add Geometries

And again, now that we have devoted a lot of code to setting up the grammar of the graph, it is again a relatively simple matter to try out different geometries.¹

10.1 Histogram

myplot3 + geom_histogram(alpha = 0.5)

10.2 Density

myplot3 + geom_density(alpha = 0.5)

¹ It is important to use (alpha = ...) to create transparency with these geoms.