week2

Hakan Mehmetcik

Data visualization

This week we will engage with the question of why well-designed data graphics are important and describe a taxonomy for understanding their composition.

well-designed data graphics are important

the analogy that creating data graphics is like cooking: Anyone can learn to type graphical commands and generate plots on the computer. Similarly, anyone can heat up food in a microwave. What separates a high-quality visualization from a plain one are the same elements that separate great chefs from novices: mastery of their tools, knowledge of their ingredients, insight, and creativity.

An Example for Thinking of Data Graphics

> **Note**
>
> The U.S. presidential election, held every four years, attracts immense interest. Candidates begin fundraising nearly two years prior, amassing hundreds of millions for national campaigns, showcasing leadership and organizational skills crucial for the presidency. Concerns about money's influence arise, notably due to unlimited political spending by corporations permitted by Citizens United v. FEC. Data science is pivotal in unraveling campaign spending patterns. The FEC logs contributions over $200 and committee spending, alongside election results. Integrating these datasets demands skill. For the timebeing, our current emphasis lies on using graphical displays for clear and accurate messaging.

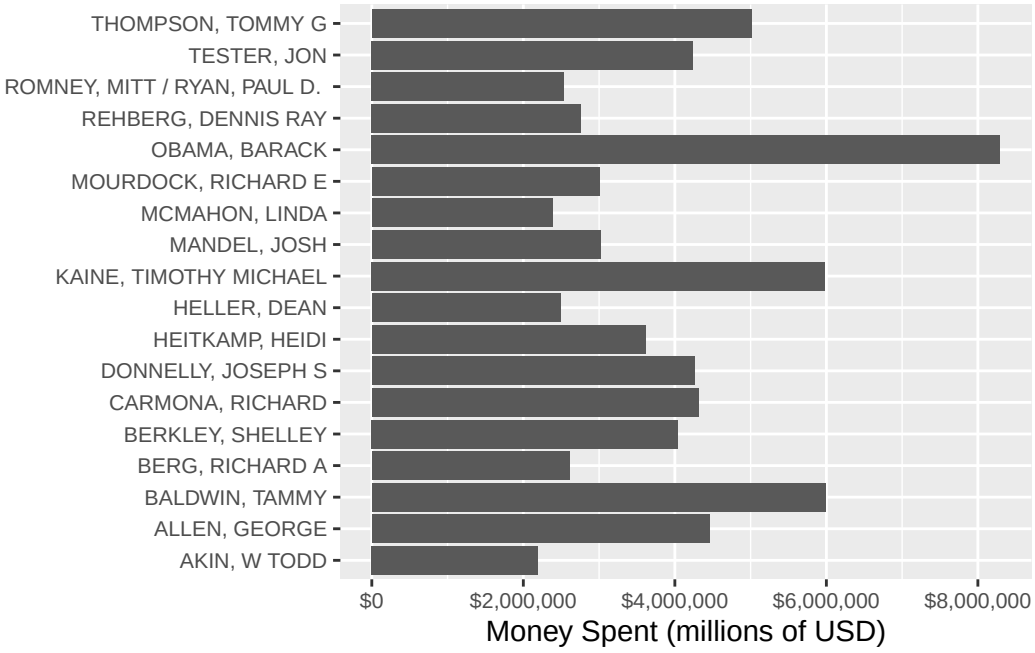
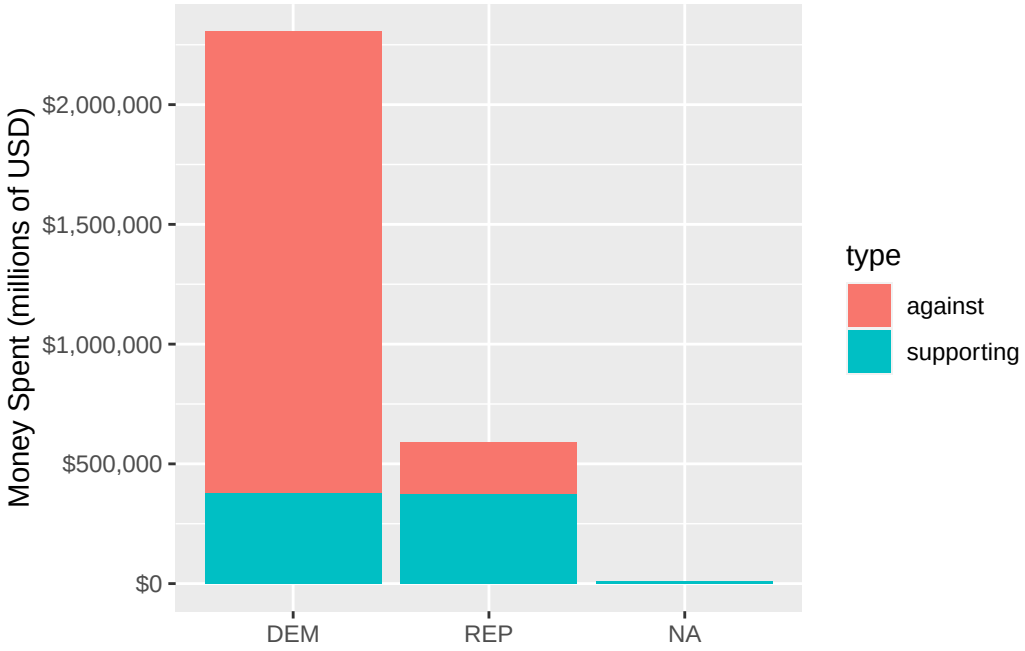


Figure 1: Amount of money spent on individual candidates in the general election phase of the 2012 federal election cycle, in millions of dollars. Candidacies with at least $4 million in spending are depicted.

> **Note**
>
> From the figure above we see that President Barack Obama's re-election campaign outspent all other candidates, notably more than doubling the expenditure of his Republican opponent, Mitt Romney. However, committees aren't restricted to supporting candidates; they can also allocate funds against specific candidates, notably through attack ads. Figure below dissects whether the spending was directed for or against the candidate.

```
spent_tidy <- spent2 %>%
  group_by(party) %>%
  summarize(supporting = sum(supporting), against = sum(against)) %>%
  pivot_longer(-party, names_to = "type", values_to = "spent") %>%
  filter(spent > 1000)

ggplot(data = spent_tidy, aes(x = party, y = spent , fill = type)) +
  scale_x_discrete(name = NULL) +
  scale_y_continuous(name = "Money Spent (millions of USD)", labels = scales::dollar) +
  geom_col()
```

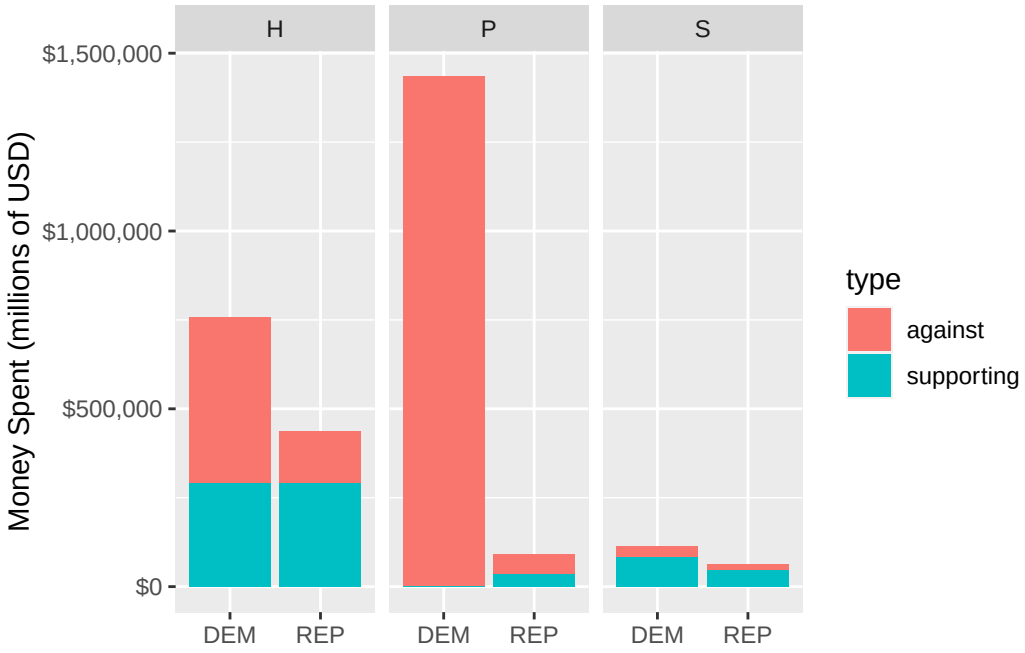


From the figure above, we see that more money was spent supporting Republican candidates than Democrats, and more money was spent attacking Democratic candidates than Republican.

By further subdividing the contributions in Figure above, by the office being sought, we can see in Figure below that while more money was spent supporting Republican candidates for all elective branches of government, it was only in the presidential election that more money was spent attacking Democratic candidates. In fact, slightly more money was spent attacking Republican House and Senate candidates.

```
spent2 %>%
  filter(!is.na(office)) %>%
  group_by(party, office) %>%
  summarize(supporting = sum(supporting), against = sum(against)) %>%
  pivot_longer(-c(party, office), names_to = "type", values_to = "spent") %>%
  filter(spent > 1000) %>%
ggplot(aes(x = party, y = spent , fill = type)) +
  scale_x_discrete(name = NULL) +
  scale_y_continuous(name = "Money Spent (millions of USD)", labels = scales::dollar) +
  geom_col() +
  facet_wrap(~office)
```

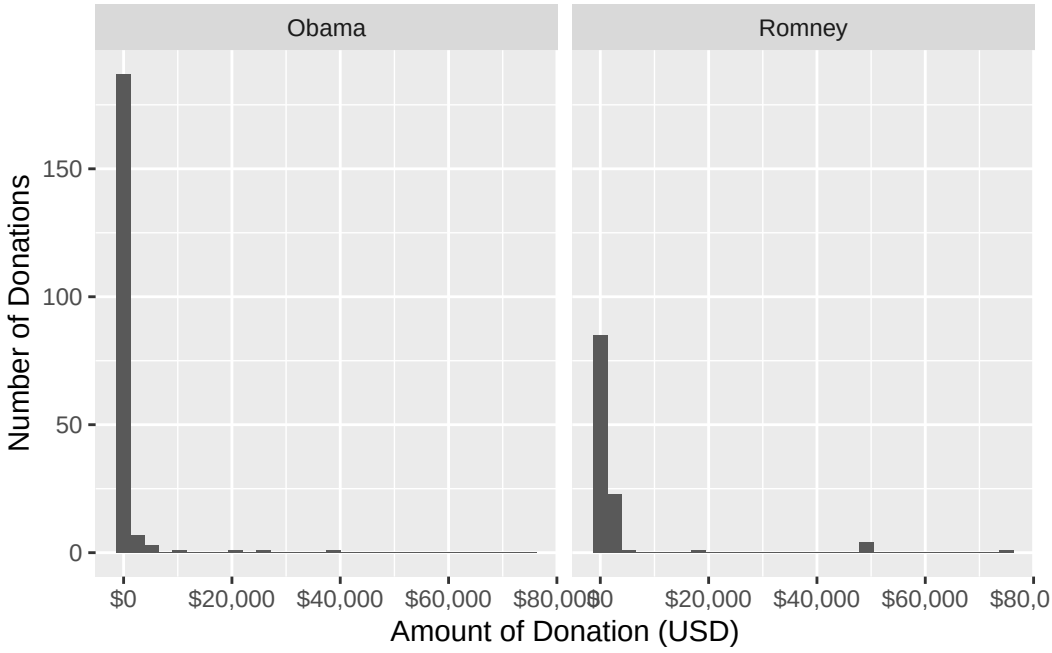
```
`summarise()` has grouped output by 'party'. You can override using the
`.groups` argument.
```



```
load(here::here("data", "fec12_donations.rda"))

ggplot(data = donations, aes(x = transaction_amt)) +
  geom_histogram() +
  scale_x_continuous(name = "Amount of Donation (USD)", labels = scales::dollar) +
  scale_color_discrete(name = NULL) +
  ylab("Number of Donations") +
  facet_wrap(~candidate)
```

```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



During the presidential election, a theme emerged suggesting that Romney's campaign was backed by a select group of wealthy donors, while Obama's support came from a more diverse economic spectrum. To investigate this claim, Figure above presents histograms summarizing over one million donations made by individuals to the major committees supporting each candidate (Obama for America and the Obama Victory Fund 2012 for Obama, and Romney for President and Romney Victory 2012 for Romney). While the histograms indicate that Obama received more smaller donations, suggesting some support for the claim, the evidence is inconclusive. Challenges include both candidates receiving numerous small donations alongside a few larger ones, making it difficult to discern the distribution on the horizontal axis.

Additionally, comparing the histograms side by side is challenging, and donations from both phases of the presidential election are aggregated, further complicating analysis.

```
ggplot(data = donations, aes(x = transaction_amt)) +
  geom_density(aes(color = candidate), adjust = 4) +
  scale_x_log10(name = "Amount of Donation (USD)", labels = scales::dollar) +
  scale_color_discrete(name = NULL) +
#  coord_trans(x = "log10") +
  facet_wrap(~phase)
```

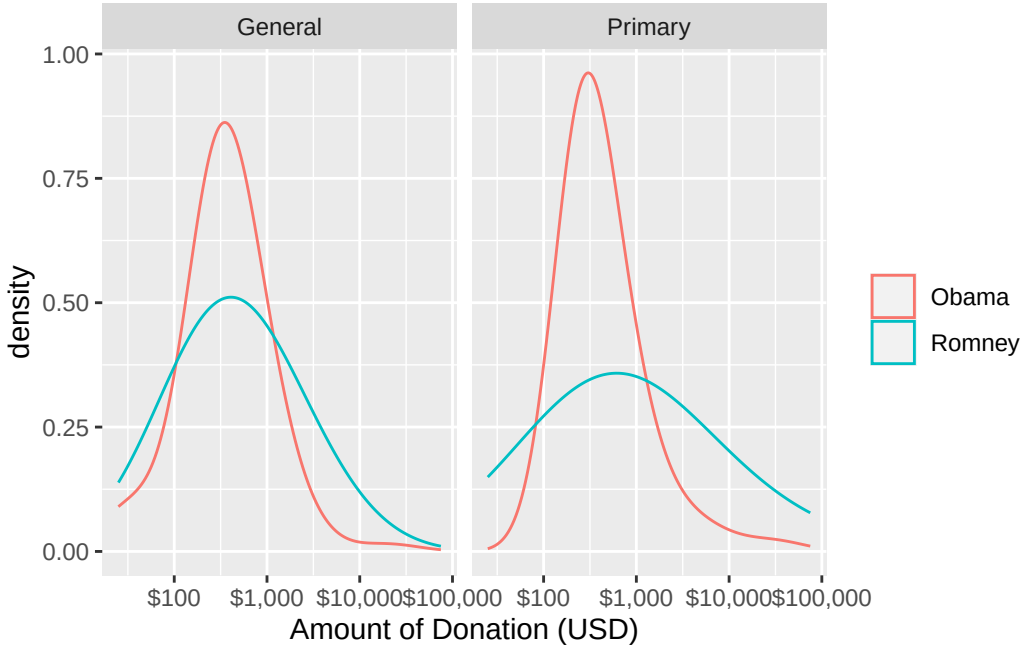


Figure above addresses previous issues by (1) utilizing density curves instead of histograms for direct distribution comparison, (2) employing the logarithm of the donation amount on the horizontal scale to emphasize relevant data, and (3) segregating donations by the election phase. This refinement enables more nuanced conclusions. The right panel corroborates the assertion that Obama's donations originated from a broader base during the primary election phase, with a noticeable influx of smaller donations. However, during the general phase, the distribution of donations to either campaign shows virtually no distinction.

Composing data graphics

Former *New York Times* intern and FlowingData.com creator Nathan Yau makes the analogy that creating data graphics is like cooking: Anyone can learn to type graphical commands and generate plots on the computer. Similarly, anyone can heat up food in a microwave. What separates a high-quality visualization from a plain one are the same elements that separate great chefs from novices: mastery of their tools, knowledge of their ingredients, insight, and creativity (Yau 2013). In this section, we present a framework—rooted in scientific research—for understanding data graphics. Our hope is that by internalizing these ideas you will refine your data graphics palette.

A taxonomy for data graphics

In ggplot framework, data graphics can be understood in terms of four basic elements: visual cues, coordinate systems, scale, and context.

- **Visual Cues:** Visual cues are graphical elements that draw the eye to what you want your audience to focus upon

  Table 2.1: Visual cues and what they signify.

Visual Cue	Variable Type	Question
Position	numerical	where in relation to other things?
Length	numerical	how big (in one dimension)?
Angle	numerical	how wide? parallel to something else?
Direction	numerical	at what slope? in a time series, going up or down?
Shape	categorical	belonging to which group?
Area	numerical	how big (in two dimensions)?
Volume	numerical	how big (in three dimensions)?
Shade	either	to what extent? how severely?
Color	either	to what extent? how severely?

Research into graphical perception (dating back to the mid-1980s) has shown that human beings' ability to perceive differences in magnitude accurately descends in this order (Cleveland and McGill 1984). That is, **humans are quite good at accurately perceiving differences in position (e.g., how much taller one bar is than another), but not as good at perceiving differences in angles.** This is one reason why many people prefer bar charts to *pie charts*. Our relatively poor ability to perceive differences in color is a major factor in the relatively low opinion of *heat maps* that many data scientists have.

- **Coordinate systems:** How are the data points organized? While any number of coordinate systems are possible, three are most common:

  - **Cartesian**: The familiar (x,y)-rectangular coordinate system with two perpendicular axes.
  - **Polar**: The radial analog of the Cartesian system with points identified by their radius and angle .
  - **Geographic**: The increasingly important system in which we have locations on the curved surface of the Earth, but we are trying to represent these locations in a flat two-dimensional plane.

- **Scale** translate values into visual cues. The choice of scale is often crucial. The central question is *how*does distance in the data graphic translate into meaningful differences in quantity? Each coordinate axis can have its own scale, for which we have three different choices:
  - **Numeric**: A numeric quantity is most commonly set on a *linear*, *logarithmic*, or *percentage* scale. Note that a logarithmic scale does not have the property that, say, a one-centimeter difference in position corresponds to an equal difference in quantity anywhere on the scale.
  - **Categorical**: A categorical variable may have no ordering (e.g., Democrat, Republican, or Independent), or it may be *ordinal* (e.g., never, former, or current smoker).
  - **Time**: A numeric quantity that has some special properties. First, because of the calendar, it can be demarcated by a series of different units (e.g., year, month, day, etc.). Second, it can be considered periodically (or cyclically) as a "wrap-around" scale. Time is also so commonly used and misused that it warrants careful consideration.

- **Context:** Context can be added to data graphics in the form of titles or subtitles that explain what is being shown, axis labels that make it clear how units and scale are depicted, or reference points or lines that contribute relevant external information. While one should avoid cluttering up a data graphic with excessive annotations, it is necessary to provide proper context.
- **Small multiples and layers:** One of the fundamental challenges of creating data graphics is condensing multivariate information into a two-dimensional image. While three-dimensional images are occasionally useful, they are often more confusing than anything else. Instead, here are three common ways of incorporating more variables into a two-dimensional data graphic:
  - **Small multiples**: Also known as *facets*, a single data graphic can be composed of several small multiples of the same basic plot, with one (discrete) variable changing in each of the small sub-images.

- **Layers**: It is sometimes appropriate to draw a new layer on top of an existing data graphic. This new layer can provide context or comparison, but there is a limit to how many layers humans can reliably parse.
- **Animation**: If time is the additional variable, then an animation can sometimes effectively convey changes in that variable. Of course, this doesn't work on the printed page and makes it impossible for the user to see all the data at once.

Color

Color is one of the flashiest, but most misperceived and misused visual cues. In making color choices, there are a few key ideas that are important for any data scientist to understand. First, while color can be visually appealing to humans, it often isn't as informative as we might hope. Second, approximately 8% of the population—most of whom are men—have some form of color blindness. To prevent issues with color blindness, avoid contrasting red with green in data graphics. As a bonus, your plots won't seem Christmas-y!

> **i** Note
>
> Sidenote: The **RColorBrewer** package provides functionality to use these palettes directly in **R**

With a little practice, one can learn to dissect data graphics in terms of the taxonomy outlined above. For example, your basic scatterplot uses *position* in the *Cartesian* plane with *linear* scales to show the relationship between two variables.

A grammar for graphics

We have presented a taxonomy for understanding data graphics. Now, we can illustrate how the **ggplot2** package can be used to create data graphics.

In the grammar of **ggplot2**, an *aesthetic* is an explicit mapping between a variable and the visual cues that represent its values. A *glyph* is the basic graphical element that represents one case (other terms used include "mark" and "symbol"). In a scatterplot, the *positions* of a glyph on the plot—in both the horizontal and vertical senses—are the *visual cues* that help the viewer understand how big the corresponding quantities are. The *aesthetic* is the mapping that defines these correspondences. When more than two variables are present, additional aesthetics can marshal additional visual cues. Note also that some visual cues (like *direction* in a time series) are implicit and do not have a corresponding aesthetic.

Table 2: A selection of variables from the first six rows of the CIA countries data table.

country	oil_prod	gdp	educ	roadways	net_users
Afghanistan	0	1900	NA	0.0646244	>5%
Albania	20510	11900	3.3	0.6261305	>35%
Algeria	1420000	14500	4.3	0.0477193	>15%
American Samoa	0	13000	NA	1.2110553	NA
Andorra	NA	37200	NA	0.6837607	>60%
Angola	1742000	7300	3.5	0.0412521	>15%

Example:

We begin with a data set that includes measures that are relevant to answer questions about economic productivity. The `CIACountries` data table contains seven variables collected for each of 236 countries: population (`pop`), area (`area`), gross domestic product (`gdp`), percentage of GDP spent on education (`educ`), length of roadways per unit area (`roadways`), Internet use as a fraction of the population (`net_users`), and the number of barrels of oil produced per day (`oil_prod`).

Aesthetics

In the simple scatterplot shown in the Figure below, we employ the grammar of graphics to build a multivariate data graphic. In **ggplot2**, the `ggplot()` command creates a plot object `g`, and any arguments to that function are applied across any subsequent plotting directives. In this case, this means that any variables mentioned anywhere in the plot are understood to be within the `CIACountries` data frame, since we have specified that in the `data` argument. Graphics in **ggplot2** are built incrementally by elements. In this case, the only glyphs added are points, which are plotted using the `geom_point()` function. The arguments to `geom_point()` specify *where* and *how* the points are drawn. Here, the two *aesthetics* (`aes()`) map the vertical (`y`) coordinate to the `gdp` variable, and the horizontal (`x`) coordinate to the `educ` variable. The `size` argument to `geom_point()` changes the size of all of the glyphs.

```
g <- ggplot(data = CIACountries, aes(y = gdp, x = educ))
g + geom_point(size = 3)
```

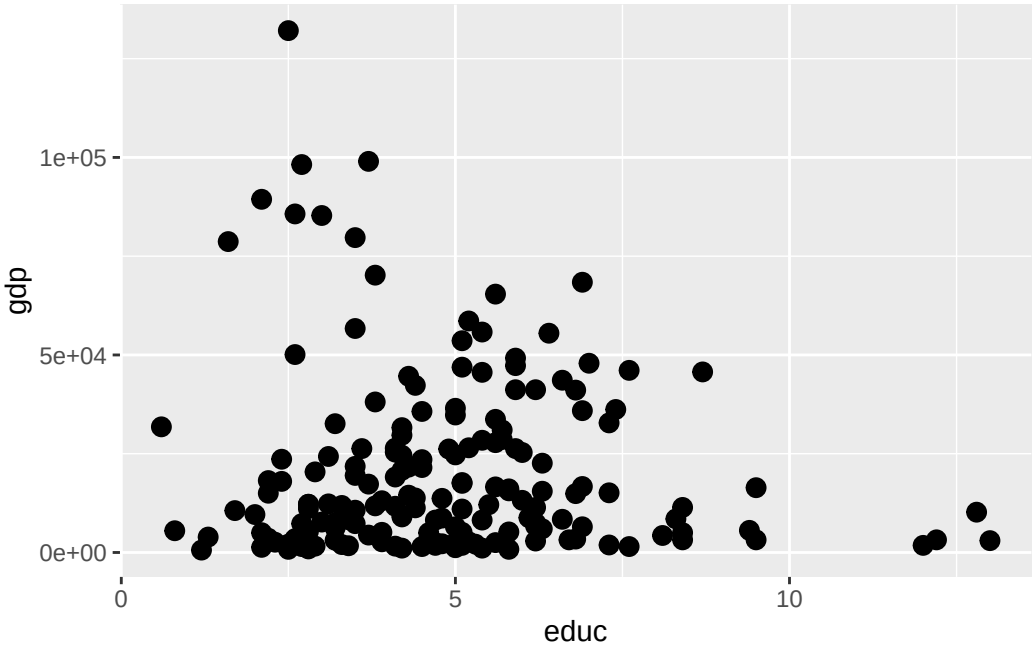


Figure 2: Scatterplot using only the position aesthetic for glyphs.

In Figure above, the glyphs are simple. Only position in the frame distinguishes one glyph from another. The shape, size, etc. of all of the glyphs are identical—there is nothing about the glyph itself that identifies the country.

However, it is possible to use a glyph with several attributes. We can define additional aesthetics to create new visual cues. In Figure below, we have extended the previous example by mapping the color of each dot to the categorical `net_users` variable.

```
g + geom_point(aes(color = net_users), size = 3)
```

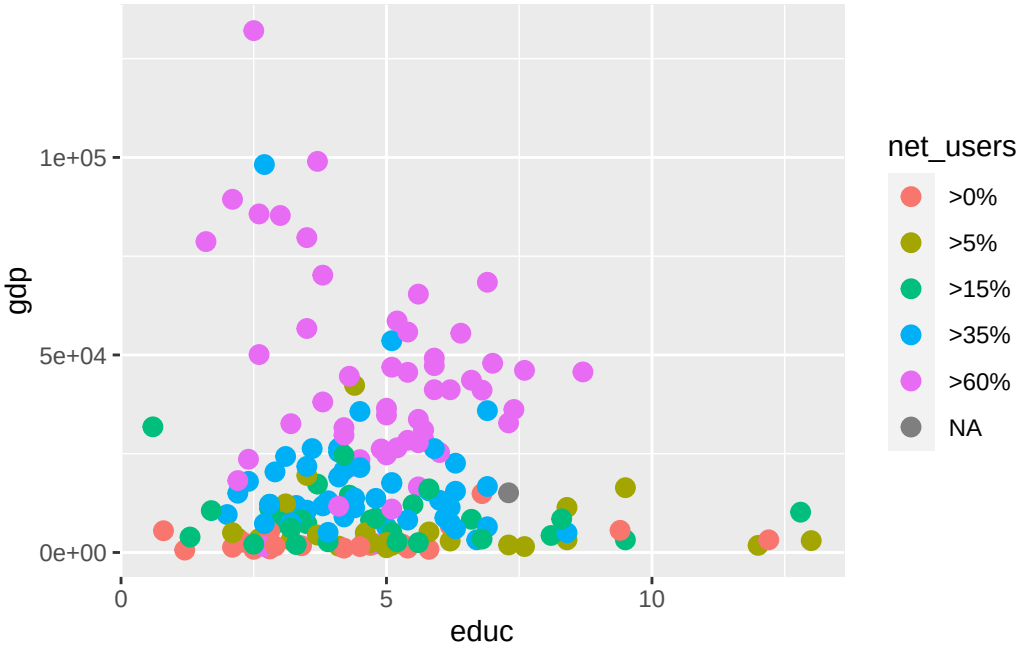


Figure 3: (ref:net-use-color-cap)

Changing the glyph is as simple as changing the function that draws that glyph—the aesthetic can often be kept exactly the same. In the figure below, we plot text instead of a dot.

```
g + geom_text(aes(label = country, color = net_users), size = 3)
```

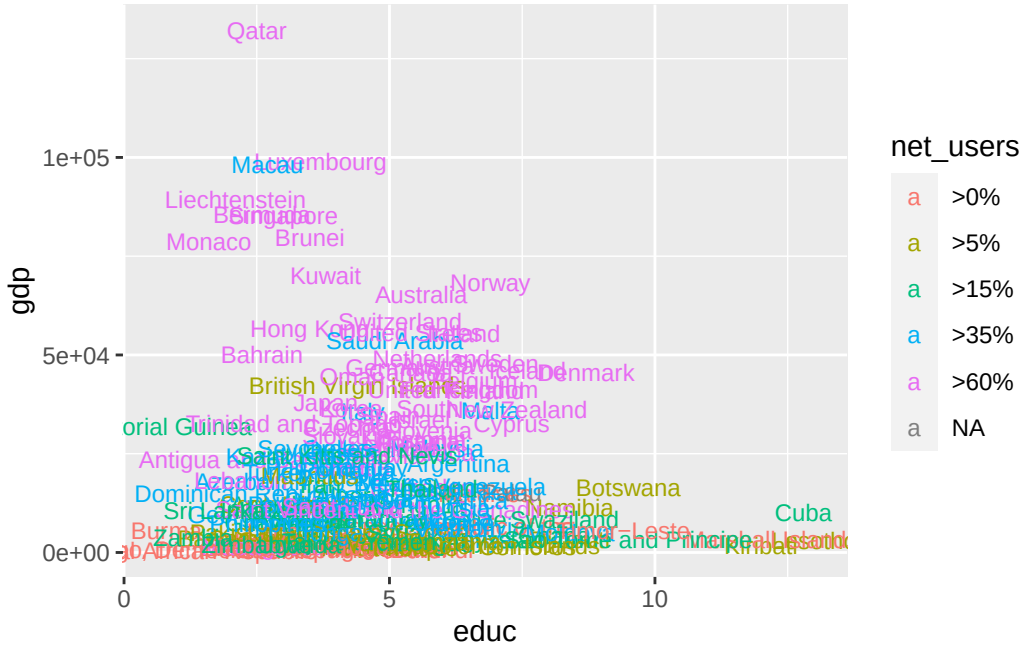


Figure 4: Scatterplot using both location and label as aesthetics.

Of course, we can employ multiple aesthetics. There are four aesthetics in the figure below. Each of the four aesthetics is set in correspondence with a variable—we say the variable is *mapped* to the aesthetic. Educational attainment is being mapped to horizontal position, GDP to vertical position, Internet connectivity to color, and length of roadways to size. Thus, we encode four variables (`gdp`, `educ`, `net_users`, and `roadways`) using the visual cues of position, position, color, and area, respectively.

```
g + geom_point(aes(color = net_users, size = roadways))
```

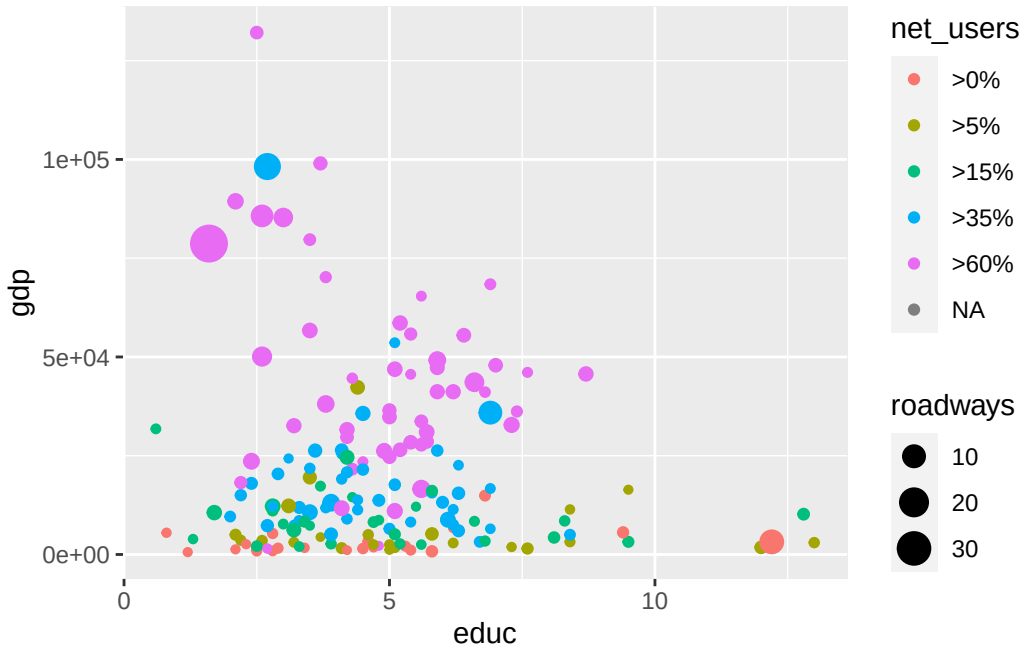


Figure 5: (ref:four-variables-cap)

A data table provides the basis for drawing a data graphic. The relationship between a data table and a graphic is simple: Each case (row) in the data table becomes a mark in the graph. As the designer of the graphic, you choose which variables the graphic will display and how each variable is to be represented graphically: position, size, color, and so on.

Scales

Compare the last figure above to the figure below. In the former, it is hard to discern differences in GDP due to its right-skewed distribution and the choice of a *linear* scale. In the latter, the *logarithmic* scale on the vertical axis makes the scatterplot more readable. Of course, this makes interpreting the plot more complex, so we must be very careful when doing so. Note that the only difference in the code is the addition of the `coord_trans()` directive.

```
g +
  geom_point(aes(color = net_users, size = roadways)) +
  coord_trans(y = "log10")
```

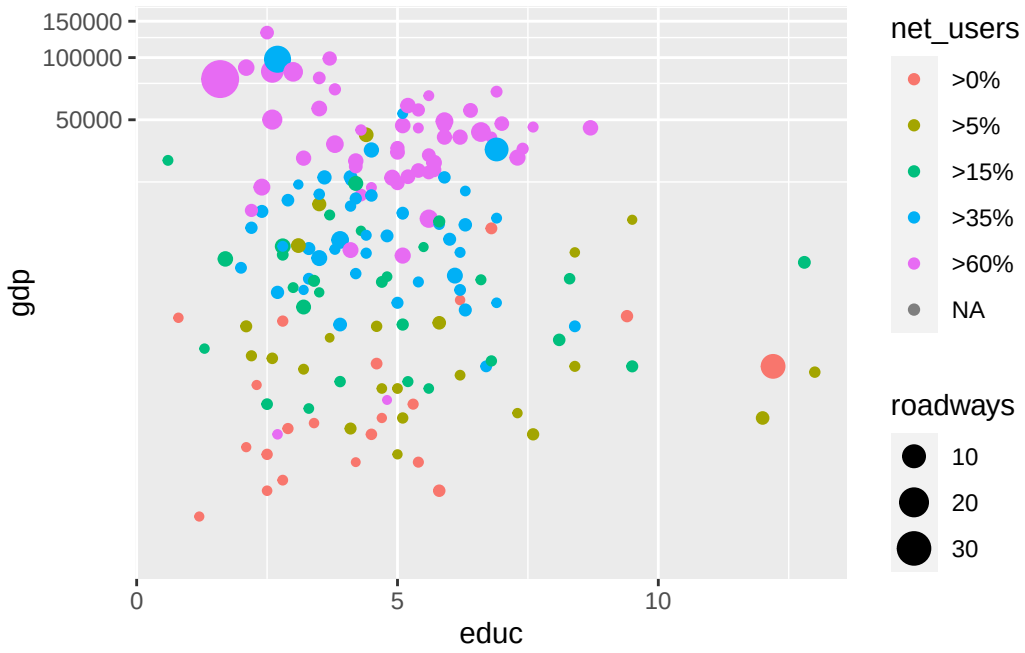


Figure 6: Scatterplot using a logarithmic transformation of GDP that helps to mitigate visual clustering caused by the right-skewed distribution of GDP among countries.

Scales can also be manipulated in **ggplot2** using any of the scale functions. For example, instead of using the `coord_trans()` function as we did above, we could have achieved a similar plot through the use of the `scale_y_continuous()` function. Similarly-named functions (e.g., `scale_x_continuous()`, `scale_x_discrete()`, `scale_color()`, etc.) perform analogous operations on different aesthetics.

```
g +
  geom_point(aes(color = net_users, size = roadways)) +
  scale_y_continuous(
    name = "Gross Domestic Product",
    trans = "log10",
    labels = scales::comma
  )
```

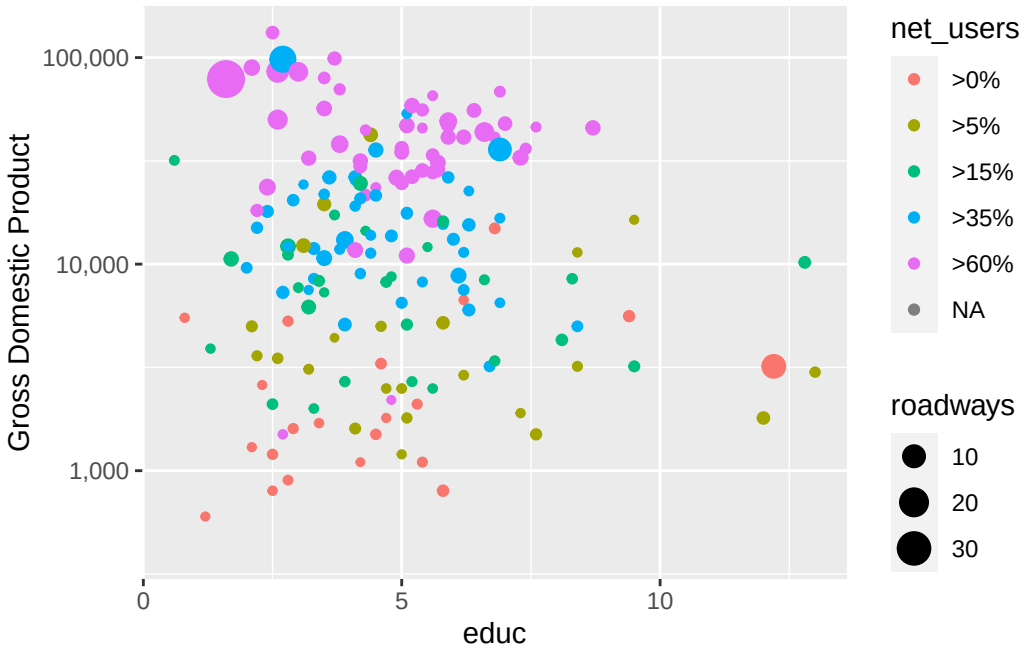


Figure 7: (ref:log-scale2-cap)

> **Note**
>
> Not all scales are about position. For instance, in the figure, `net_users` is translated to color. Similarly, `roadways` is translated to size: the largest dot corresponds to a value of 30 roadways per unit area.

Guides

Context is provided by *guides* (more commonly called legends). A guide helps a human reader understand the meaning of the visual cues by providing context.

For position visual cues, the most common sort of guide is the familiar axis with its tick marks and labels. But other guides exist. In the last two figure we draw, legends relate how dot color corresponds to internet connectivity, and how dot size corresponds to length of roadways (note the use of a log scale). The `geom_text()` and `geom_label()` functions can also be used to provide specific textual annotations on the plot.

Facets

Using multiple aesthetics such as shape, color, and size to display multiple variables can produce a confusing, hard-to-read graph. *Facets*—multiple side-by-side graphs used to display levels of a categorical variable—provide a simple and effective alternative. The figure below, uses facets to show different levels of Internet connectivity, providing a better view than the fourth figure we draw. There are two functions that create facets: `facet_wrap()` and `facet_grid()`. The former creates a facet for each level of a single categorical variable, whereas the latter creates a facet for each combination of two categorical variables, arranging them in a grid.

```
g +
  geom_point(alpha = 0.9, aes(size = roadways)) +
  coord_trans(y = "log10") +
  facet_wrap(~net_users, nrow = 1) +
  theme(legend.position = "top")
```

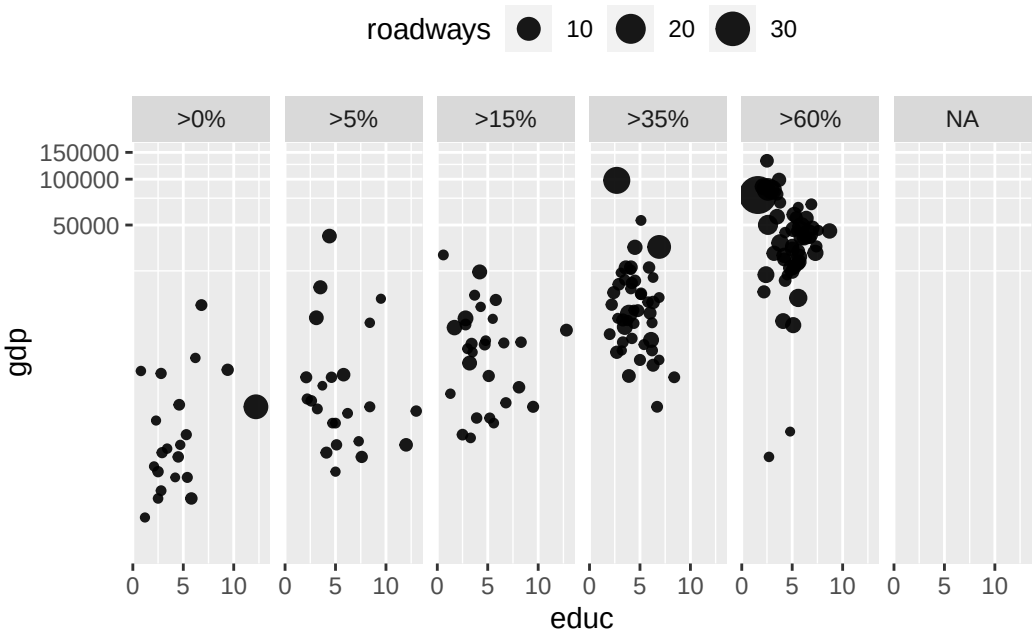


Figure 8: Scatterplot using facets for different ranges of Internet connectivity.

Table 3: Glyph-ready data for the barplot layer.

drg	stateProvider	num_charges	mean_charge
039	NJ	31	35103.81
057	NJ	55	45692.07
064	NJ	55	87041.64
065	NJ	59	59575.74
066	NJ	56	45819.13
069	NJ	61	41916.70
074	NJ	41	42992.81
101	NJ	58	42314.18
149	NJ	50	34915.54
176	NJ	36	58940.98

Layers

On occasion, data from more than one data table are graphed together. For example, the `MedicareCharges` and `MedicareProviders` data tables provide information about the average cost of each medical procedure in each state. If you live in *New Jersey*, you might wonder how providers in your state charge for different medical procedures. However, you will certainly want to understand those averages in the context of the averages across all states. In the `MedicareCharges` table, each row represents a different medical procedure (`drg`) with its associated average cost in each state. We also create a second data table called `ChargesNJ`, which contains only those rows corresponding to providers in the state of New Jersey.

```
ChargesNJ <- MedicareCharges |>
  filter(stateProvider == "NJ")
set.seed(101)
ChargesNJ %>%
  head(10) %>%
  mdsr_table(caption = "Glyph-ready data for the barplot layer.")
```

The first few rows from the data table for New Jersey are shown in the table above. This glyph-ready table can be translated to a chart (the figure below) using bars to represent the average charges for different medical procedures in New Jersey. The `geom_col()` function creates a separate bar for each of the 100 different medical procedures.

```
p <- ggplot(
  data = ChargesNJ,
  aes(x = reorder(drg, mean_charge), y = mean_charge)
) +
```

```
  geom_col(fill = "gray") +
  ylab("Statewide Average Charges ($)") +
  xlab("Medical Procedure (DRG)") +
  theme(axis.text.x = element_text(angle = 90, hjust = 1, size = rel(0.5)))
p
```

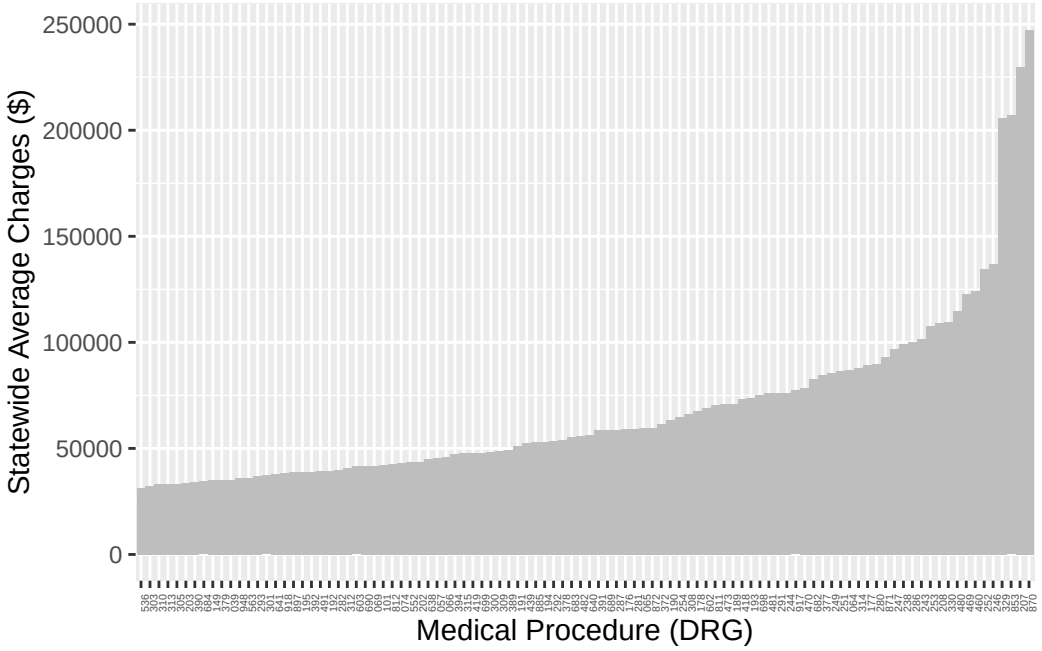


Figure 9: Bar graph of average charges for medical procedures in New Jersey.

How do the charges in New Jersey compare to those in other states? The two data tables, one for New Jersey and one for the whole country, can be plotted with different glyph types: bars for New Jersey and dots for the states across the whole country as in the figure below.

```
p + geom_point(data = MedicareCharges, size = 1, alpha = 0.3)
```

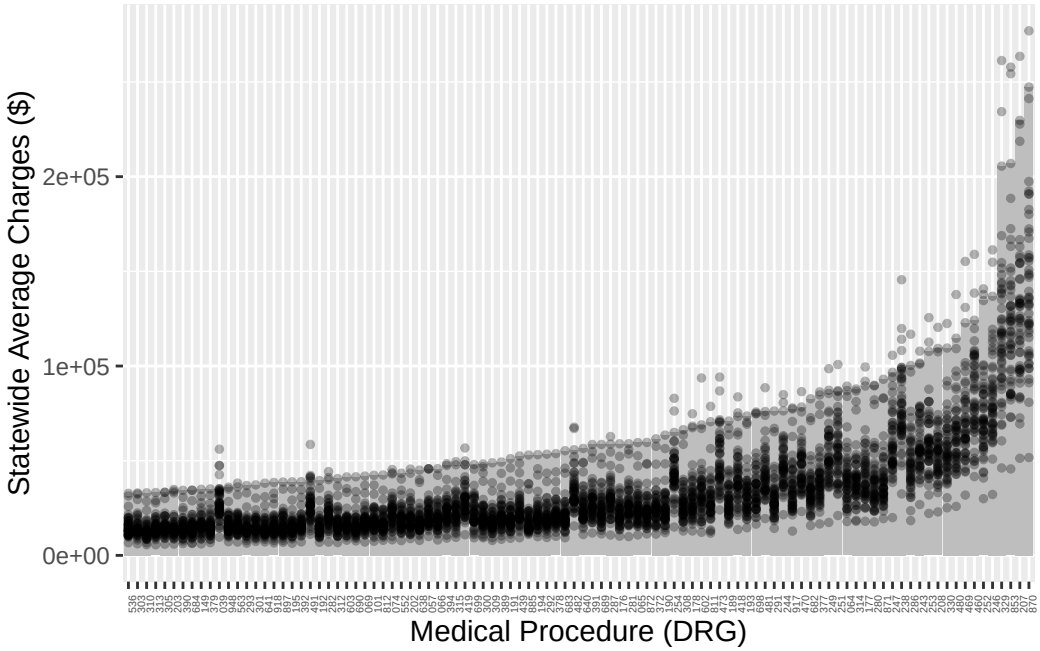


Figure 10: Bar graph adding a second layer to provide a comparison of New Jersey to other states. Each dot represents one state, while the bars represent New Jersey.

With the context provided by the individual states, it is easy to see that the charges in New Jersey are among the highest in the country for each medical procedure.

Canonical data graphics in R

There are several data graphics that every data scientist should know how to make and interpret.

Univariate Displays

Numeric Variables

It is generally useful to understand how a single variable is distributed. If that variable is numeric, then its distribution is commonly summarized graphically using a *histogram* or *density plot*. Both convey the same information, but whereas the histogram uses pre-defined bins to create a discrete distribution, a density plot uses a *kernel smoother* to make a continuous curve.

```r
g <- ggplot(data = SAT_2010, aes(x = math))
```

```r
g + geom_histogram(binwidth = 10) + labs(x = "Average Math SAT score")
```

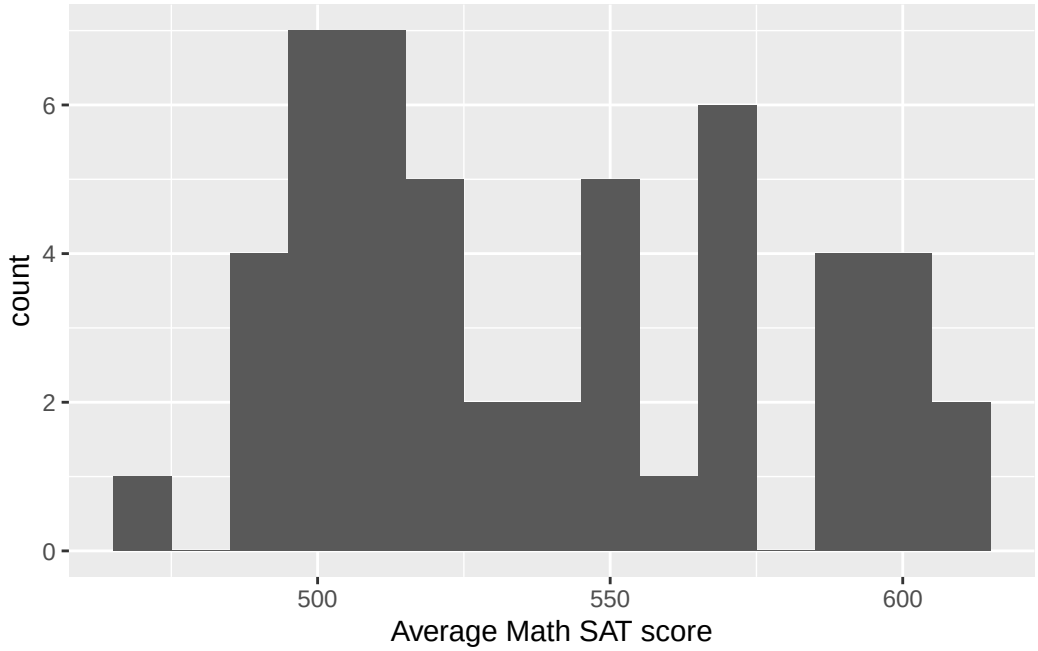


Figure 11: Histogram showing the distribution of math SAT scores by state.

The binwidth argument is utilized to determine the width of bins in a histogram. In the context provided, each bin encompasses a 10-point range of SAT scores. It's important to note that the appearance of a histogram can significantly differ depending on the chosen bin widths, and there isn't a universally optimal choice. Determining the most suitable bin width for your data requires individual assessment and consideration of the dataset's characteristics.

```r
g + geom_density(adjust = 0.3)
```

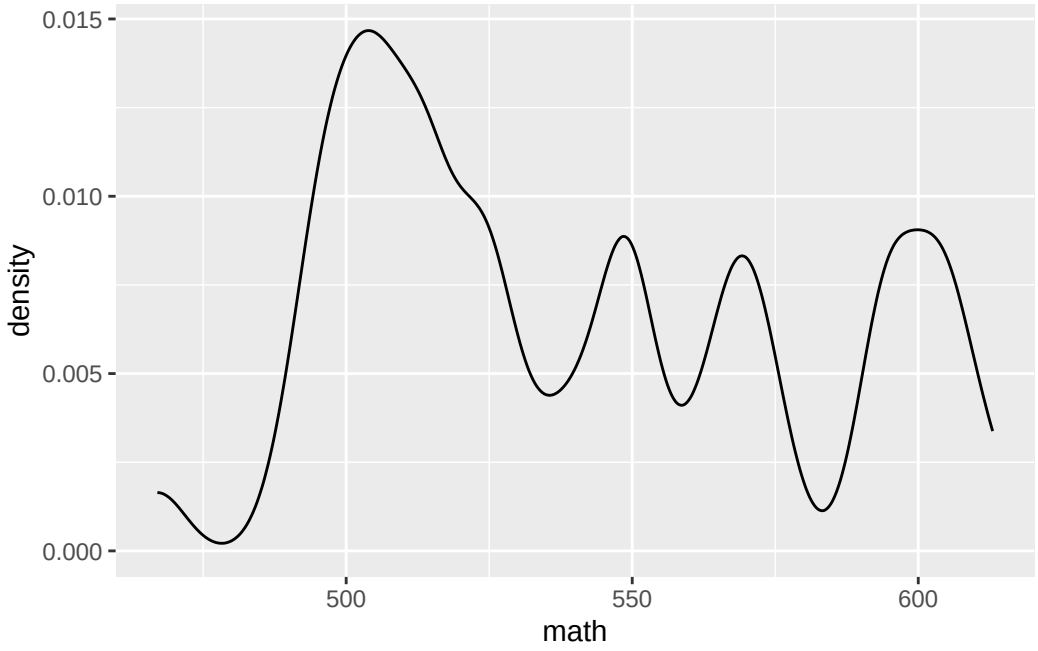


Figure 12: Density plot showing the distribution of average math SAT scores by state.

In the density plot depicted in the figure above, the adjust argument is employed to adjust the bandwidth utilized by the kernel smoother. This adjustment allows for fine-tuning the smoothing process and influencing the level of detail and smoothness in the density estimation.

Categorical Variables

If your variable is categorical, it doesn't make sense to think about the values as having a continuous density. Instead, we can use a *bar graph* to display the distribution of a categorical variable.

```
ggplot(
  data = head(SAT_2010, 10),
  aes(x = reorder(state, math), y = math)
) +
  geom_col() +
  labs(x = "State", y = "Average Math SAT score")
```

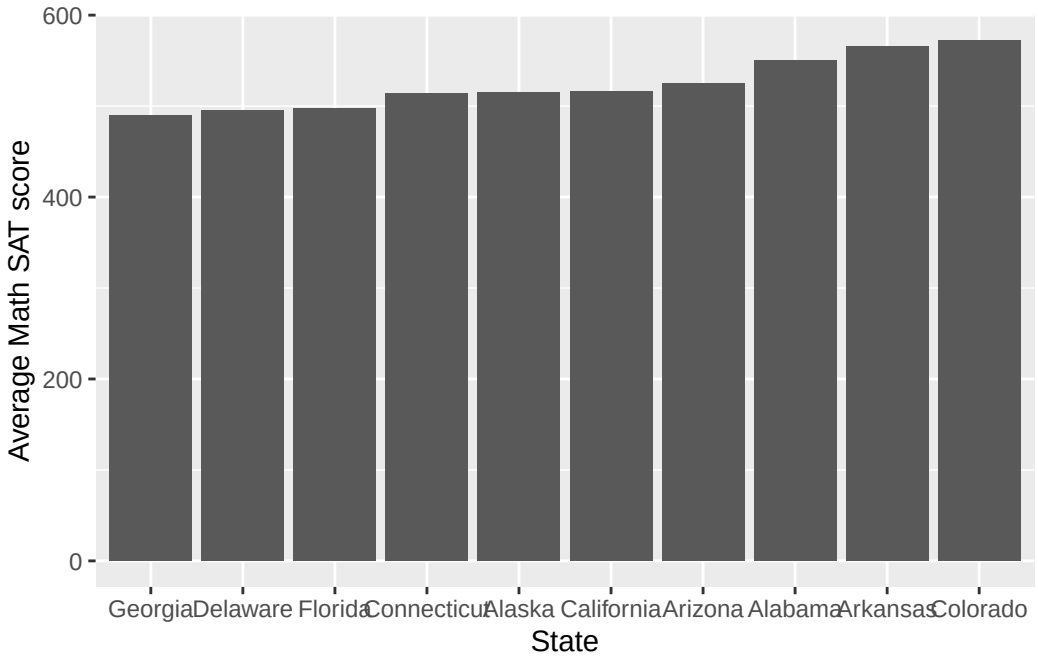


Figure 13: A bar plot showing the distribution of average math SAT scores for a selection of states.

```
#  First, we use the head() function to display only the first 10 states (in alphabetical
```

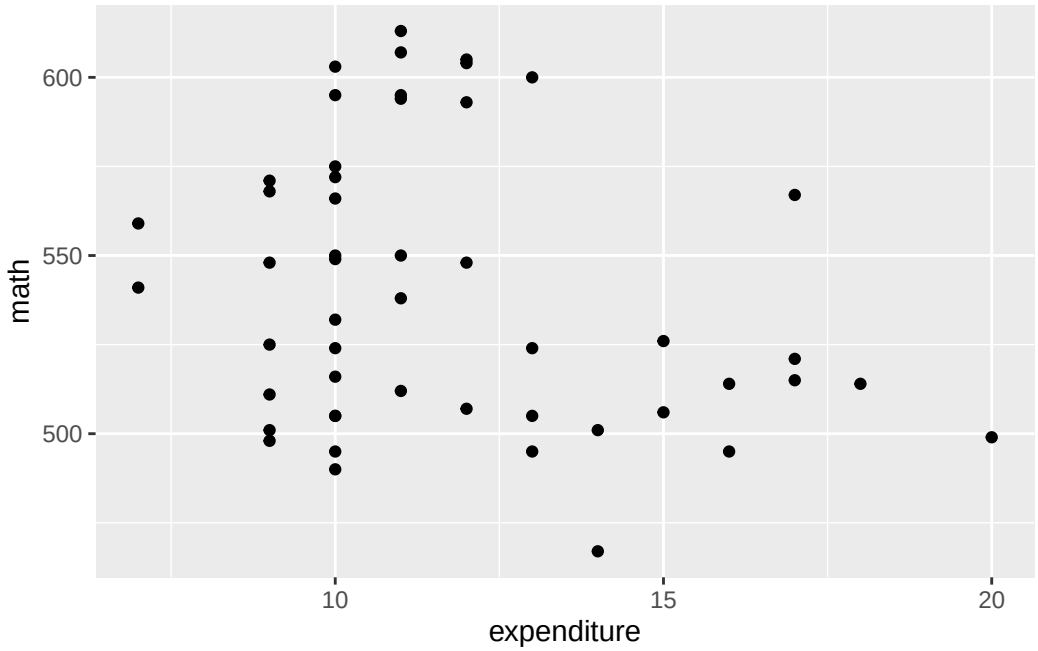
> **Note**
>
> As noted earlier, we recommend against the use of pie charts to display the distribution of a categorical variable since, in most cases, a table of frequencies is more informative.

Multivariate Displays

Two Numeric Variables

Multivariate displays are indeed highly effective for illustrating the relationships between multiple variables simultaneously. Among these, the scatterplot stands out as an excellent method for visualizing observations of two quantitative (or numerical) variables. In the ggplot2 package, the scatterplot is generated using the geom_point() command. The primary objective of a scatterplot is to illustrate the relationship between two variables across numerous cases. Typically, it employs a Cartesian coordinate system where the x-axis represents one variable, and the y-axis represents the value of a second variable.

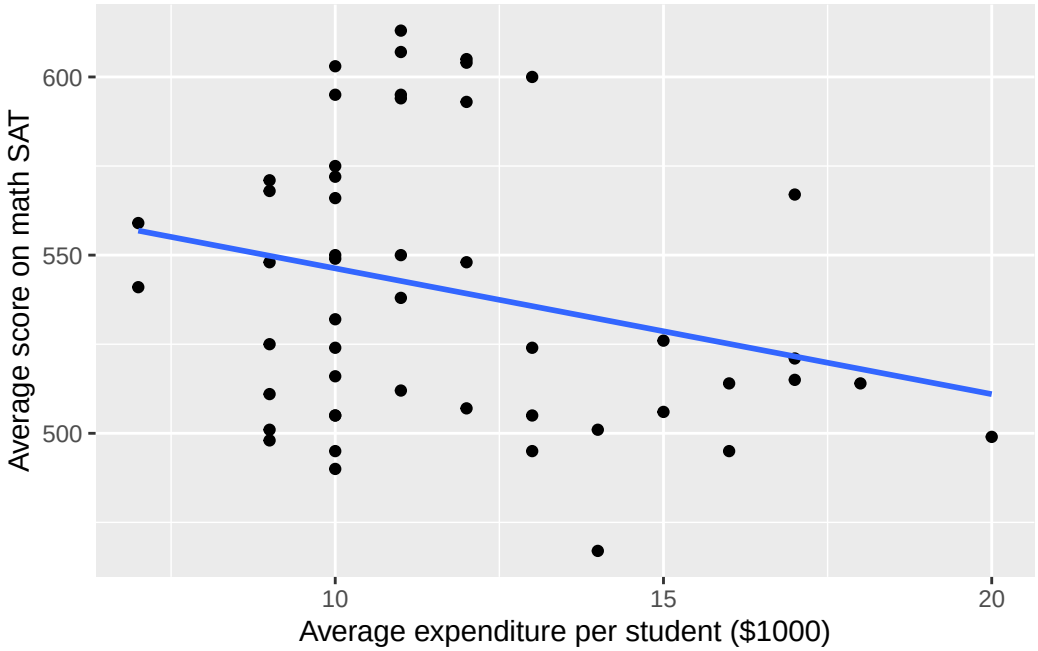
```
g <- ggplot(
  data = SAT_2010,
  aes(x = expenditure, y = math)
) +
  geom_point()
g
```



We will also add a smooth trend line and some more specific axis labels. We use the `geom_smooth()` function in order to plot the simple linear regression line (`method = "lm"`) through the points

```
g <- g +
  geom_smooth(method = "lm", se = FALSE) +
  xlab("Average expenditure per student ($1000)") +
  ylab("Average score on math SAT")
g
```

```
`geom_smooth()` using formula = 'y ~ x'
```



In the figure above, we plot the relationship between the average SAT math score and the expenditure per pupil (in thousands of United States dollars) among states in 2010. A third (categorical) variable can be added through *faceting* and/or *layering*. In this case, we use the `mutate()` function to create a new variable called `SAT_rate` that places states into bins (e.g., high, medium, low) based on the percentage of students taking the SAT. Additionally, in order to include that new variable in our plots, we use the `%+%` operator to update the data frame that is bound to our plot.

```
SAT_2010 <- SAT_2010 %>%
  mutate(
    SAT_rate = cut(
      sat_pct,
      breaks = c(0, 30, 60, 100),
      labels = c("low", "medium", "high")
    )
  )
g <- g %+% SAT_2010

# %+% is used for adding elements to a ggplot2 plot.
```

We can also use the color aesthetic to separate the data by SAT_rate on a single plot (i.e., layering).

```
g + aes(color = SAT_rate)
```

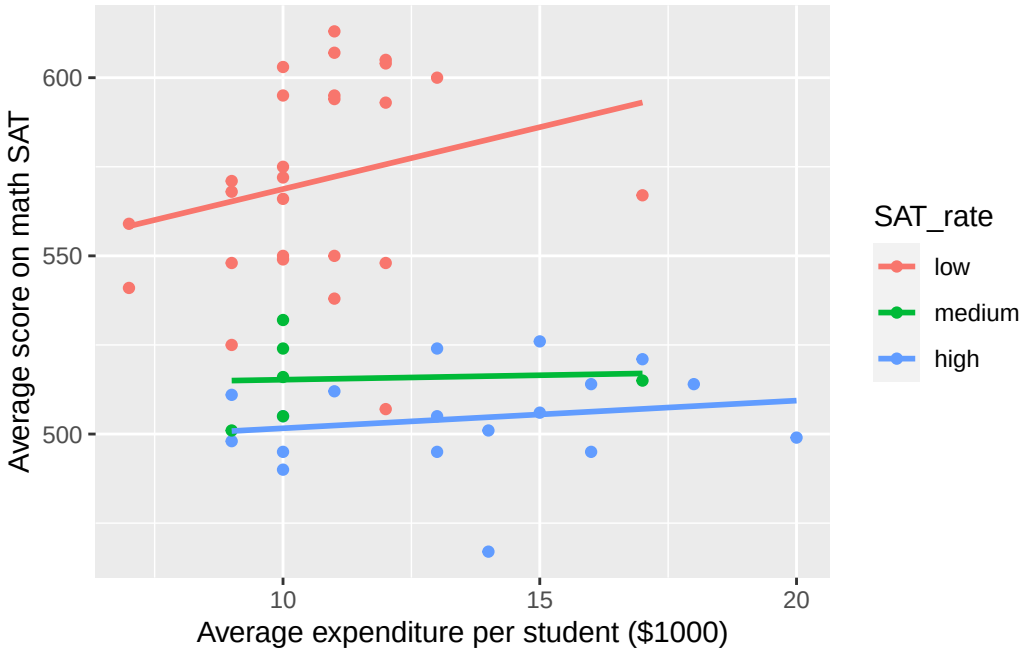


Figure 14: (ref:groups-color-cap)

Same can be done with faceting options via facet_warp() function

```
g + facet_wrap(~ SAT_rate)
```

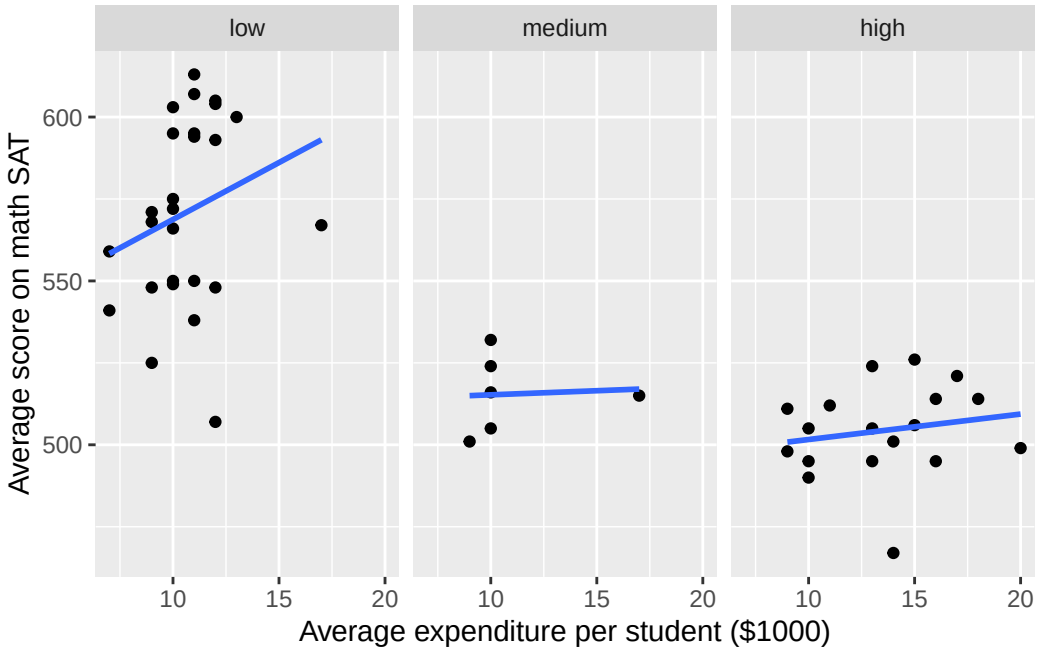


Figure 15: (ref:bar-facet-cap)

The `NHANES` data table provides medical, behavioral, and morphometric measurements of individuals. The scatterplot in the figure below shows the relationship between two of the variables, height and age. Each dot represents one person and the position of that dot signifies the value of the two variables for that person. Scatterplots are useful for visualizing a simple relationship between two variables. For instance, you can see in the figure the familiar pattern of growth in height from birth to the late teens.

It's helpful to do a bit more wrangling (more on this later) to ensure that the spatial relationship of the lines (adult men tend to be taller than adult women) matches the ordering of the legend labels. Here we use the `fct_relevel()` function (from the **forcats** package) to reset the factor levels.

```
library(NHANES)
ggplot(
  data = slice_sample(NHANES, n = 1000),
  aes(x = Age, y = Height, color = fct_relevel(Gender, "male"))
) +
  geom_point() +
  geom_smooth() +
  xlab("Age (years)") +
```

```
  ylab("Height (cm)") +
  labs(color = "Gender")
```

Figure 16: (ref:nhanes-cap)

Some scatterplots have special meanings. A *time series*—such as the one shown in the figure below—is just a scatterplot with time on the horizontal axis and points connected by lines to indicate temporal continuity. In the figure, the temperature at a weather station in western *Massachusetts* is plotted over the course of the year. The familiar fluctuations based on the seasons are evident. Be especially aware of dubious causality in these plots: Is time really a good explanatory variable?

```
library(macleish)
ggplot(data = whately_2015, aes(x = when, y = temperature)) +
  geom_line(color = "darkgray") +
  geom_smooth() +
  xlab(NULL) +
  ylab("Temperature (degrees Celsius)")
```

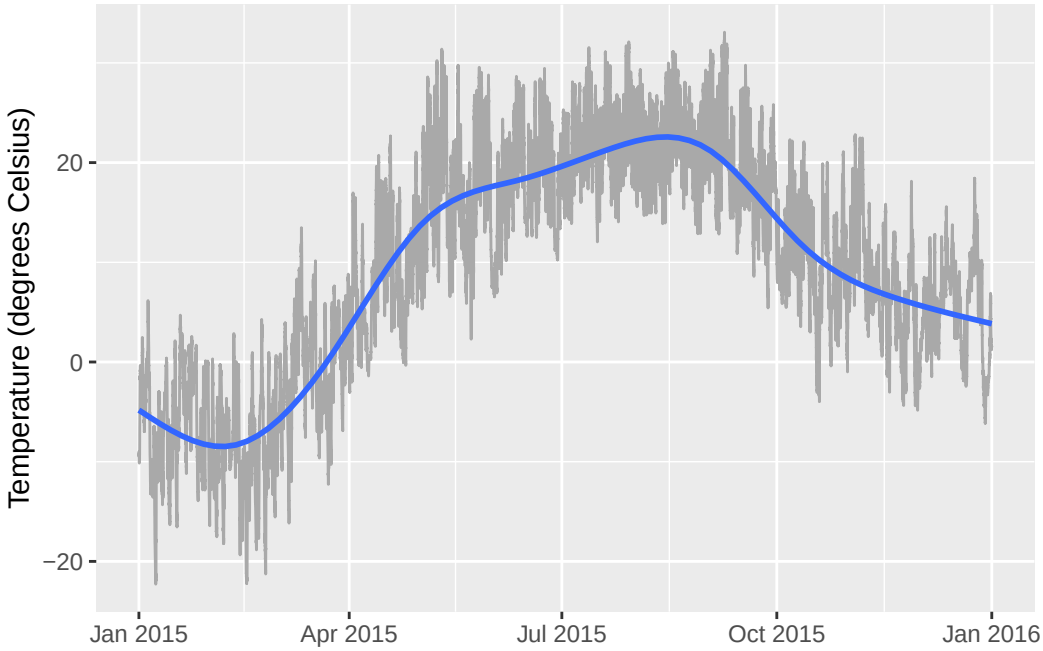


Figure 17: A time series showing the change in temperature at the MacLeish field station in 2015.

Two Categorical Variables

An informative graphical display can be achieved using a *stacked bar plot*, for more than one categorical variables.

```
ggplot(data = mosaicData::HELPrct, aes(x = homeless)) +
  geom_bar(aes(fill = substance), position = "fill") +
  scale_fill_brewer(palette = "Spectral") +
  coord_flip()
```

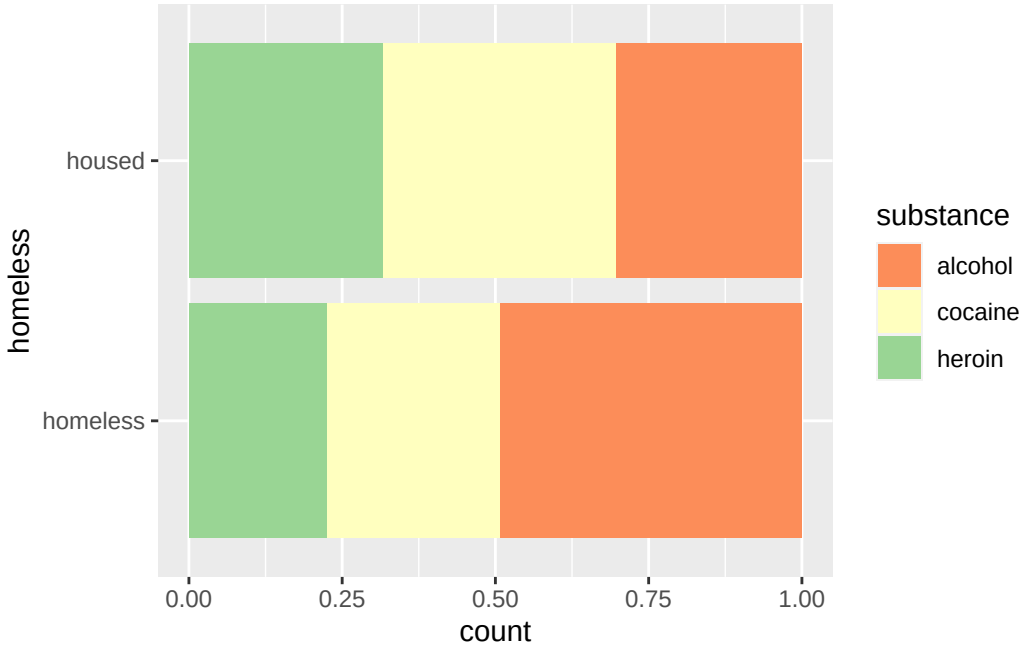


Figure 18: (ref:stacked-bar-cap)

```
# Note that we have used the coord_flip() function to display the bars horizontally instea
```

When both the explanatory and response variables are categorical (or binned), points and lines don't work as well. How likely is a person to have *diabetes*, based on their age and *BMI* (body mass index)? In the *mosaic plot* (or eikosogram) shown in the figure below, the number of observations in each cell is proportional to the area of the box. Thus, you can see that diabetes tends to be more common for older people as well as for those who are obese, since the blue-shaded regions are larger than expected under an independence model while the pink are less than expected. These provide a more accurate depiction of the intuitive notions of probability familiar from *Venn diagrams*.

```
Warning: `unite_()` was deprecated in tidyr 1.2.0.
i Please use `unite()` instead.
i The deprecated feature was likely used in the ggmosaic package.
  Please report the issue at <https://github.com/haleyjeppson/ggmosaic>.
```

i Note

Note that the `geom_mosaic()` function is not part of **ggplot2** but rather is available through the **ggmosaic** package.

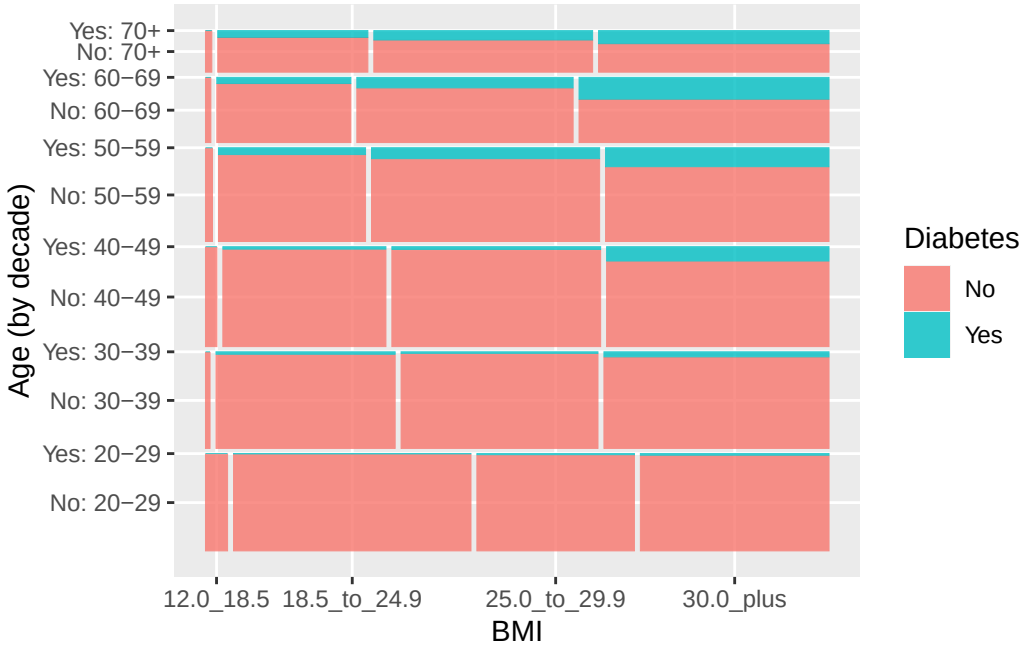


Figure 19: Mosaic plot (eikosogram) of diabetes by age and weight status (BMI).

One Numeric and One Categorical Variables

For displaying a numerical response variable against a categorical explanatory variable, a common choice is a *box-and-whisker* (or box) plot. This also is the easiest to think about this as simply a graphical depiction of the *five-number summary* (minimum [0th percentile], Q1 [25th percentile], median [50th percentile], Q3 [75th percentile], and maximum [100th percentile]).

```
whately_2015 %>%
  mutate(month = as.factor(lubridate::month(when, label = TRUE))) %>%
  group_by(month) %>%
  skim(temperature) %>%
  select(-na)
```

– Variable type: numeric ——————————————————— var month n mean sd p0 p25 p50 p75 p100 1 temperature Jan 4464 -6.37 5.14 -22.3 -10.3 -6.25 -2.35 6.16 2 temperature Feb 4032 -9.26 5.11 -22.2 -12.3 -9.43 -5.50 4.27 3 temperature Mar 4464 -0.873 5.06 -16.2 -4.61 -0.550 2.99 13.5 4 temperature Apr 4320 8.04 5.51 -3.04 3.77 7.61 11.8 22.7 5 temperature May 4464 17.4 5.94 2.29 12.8 17.5 21.4 31.4 6 temperature Jun 4320 17.7 5.11 6.53 14.2 18.0 21.2 29.4 7 temperature Jul 4464 21.6 3.90 12.0 18.6 21.2 24.3 32.1 8 temperature Aug 4464 21.4 3.79 12.9 18.4 21.1 24.3 31.2 9 temperature Sep 4320 19.3 5.07 5.43 15.8 19 22.5 33.1 10

temperature Oct 4464 9.79 5.00 -3.97 6.58 9.49 13.3 22.3 11 temperature Nov 4320 7.28 5.65 -4.84 3.14 7.11 10.8 22.8 12 temperature Dec 4464 4.95 4.59 -6.16 1.61 5.15 8.38 18.4

```
ggplot(
  data = whately_2015,
  aes(
    x = lubridate::month(when, label = TRUE),
    y = temperature
  )
) +
  geom_boxplot() +
  xlab("Month") +
  ylab("Temperature (degrees Celsius)")
```

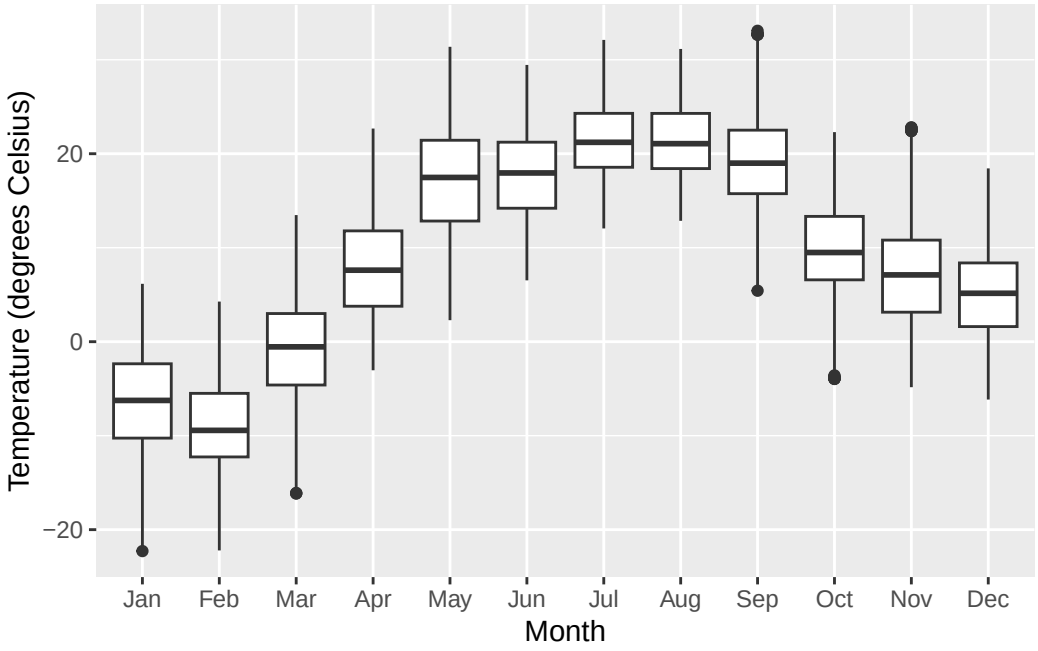


Figure 20: A box-and-whisker of temperatures by month at the MacLeish field station.

Table 4: Table of canonical data graphics and their corresponding **ggplot2** commands. Note that the mosaic plot function is not part of the **ggplot2** package.

response (y)	explanatory (x)	plot type	geom_*()
	numeric	histogram, density	\index{R}{geom_histogram()}geom_h \index{R}{geom_density()}geom_densit
	categorical	stacked bar	\index{R}{geom_bar()}geom_bar()
numeric	numeric	scatter	\index{R}{geom_point()}geom_point
numeric	categorical	box	\index{R}{geom_boxplot()}geom_box
categorical	categorical	mosaic	\index{R}{geom_mosaic()}geom_mos