Data Science for Public Policy

Aaron R. Williams and Alena Stern - Georgetown University

Data viz with ggplot2

Reading

- R for Data Science – Chapter 3 – Data Visualization
- Fundamentals of Data Visualization – Chapter 2 – Mapping data onto aesthetics
- R for Data Science – Chapter 27 – R Markdown

Motivation

1. Data visualization is exploratory data analysis (EDA)
2. Data visualization is diagnosis and validation
3. Data visualization is communication

Motivation (going beyond Excel)

- Flexibility
- Reproducibility
- Scalability
- Relational data vs. positional data

Background

- The toughest part of data visualization is data munging.
- Data frames are the only appropriate input for `library(ggplot2)`.

ggplot2 is an R package for data visualization that was developed during Hadley Wickham's graduate studies at Iowa State University. ggplot2 is formalized in "A Layered Grammar of Graphics" by Hadley Wickham, which was published in the Journal of Statistical Software in 2010.

The grammar of graphics, originally by Leland Wilkinson, is a theoretical framework that breaks all data visualizations into their component pieces. With the layered grammar of graphics, Wickham extends Wilkinson's grammar of graphics and implements it in R. The cohesion is impressive and the theory flows to the code which informs the data visualization process in a way not reflected in any other data viz tool.

There are **eight** main ingredients to the grammar of graphics. We will work our way through the ingredients with many hands-on examples.

Exercise 0

Step 1: Open your `.Rproj`.

Step 2: Create a new `.R` script in your directory called `03_data-visualization.R`.

Exercise 1

Step 1: **Type** (don't copy & paste) the following code below `library(tidyverse)` in `03_data-visualization.R`.

```
ggplot(data = storms) +
  geom_point(mapping = aes(x = pressure, y = wind))
```

Step 2: Add a comment above the ggplot2 code that describes the plot we created.

1 Data are the values represented in the visualization.

`ggplot(data = )` or `data %>% ggplot()`

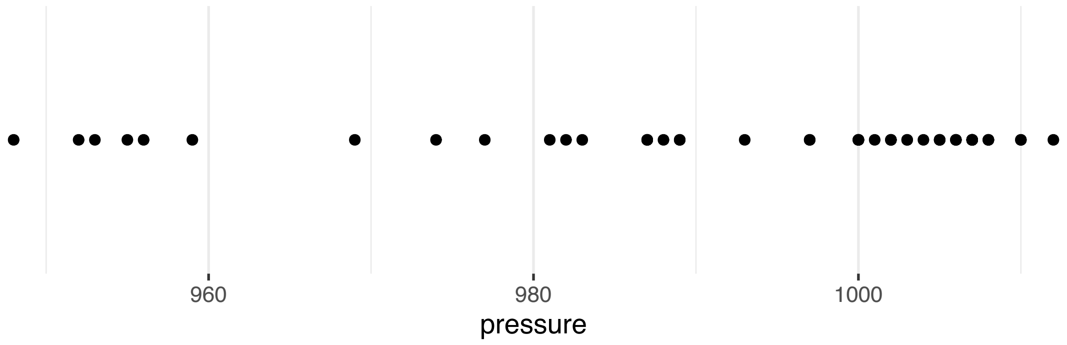
```
storms %>%
  select(name, year, category, lat, long, wind, pressure) %>%
  sample_n(10) %>%
  kable()
```

name	year	category	lat	long	wind	pressure
Gloria	1979	1	40.2	-37.8	75	985
Wilma	2005	1	16.2	-80.3	65	979
Jerry	2013	0	27.3	-45.0	40	1006
Ernesto	2006	0	30.6	-79.9	55	995
Isabel	2003	4	21.9	-60.1	130	935
Karl	2004	1	36.8	-41.9	80	957
Erin	2013	0	14.1	-24.6	35	1007
Jeanne	2004	2	26.5	-75.6	90	960
Erin	2007	0	25.8	-94.0	35	1004
Hortense	1996	-1	14.9	-55.7	30	1006

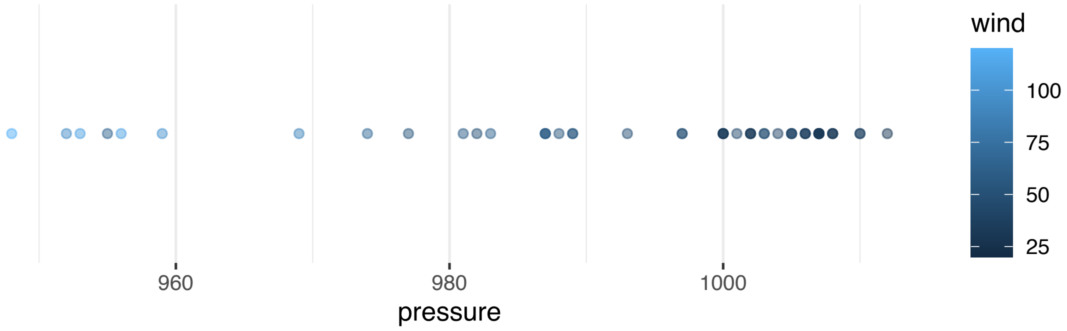
2 Aesthetic mappings are directions for how data are mapped in a plot in a way that we can perceive. Aesthetic mappings include linking variables to the x-position, y-position, color, fill, shape, transparency, and size.

```
aes(x = , y = , color = )
```

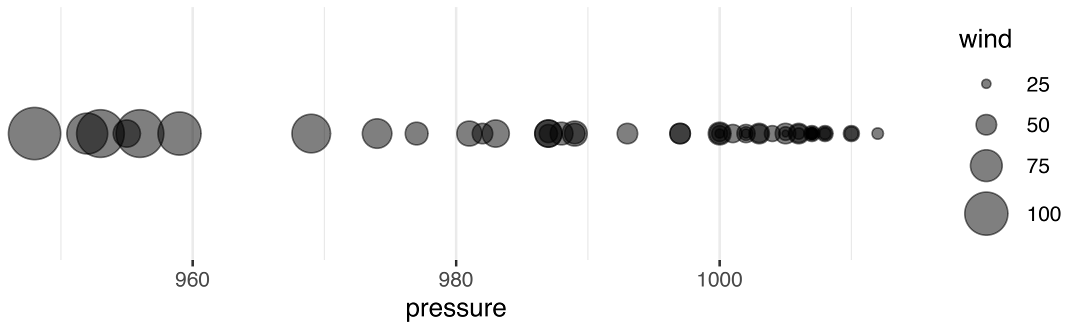
X or Y



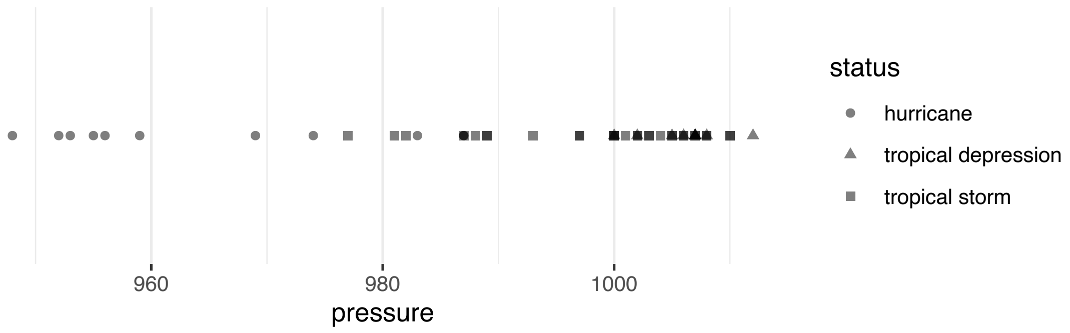
Color or Fill



Size



Shape

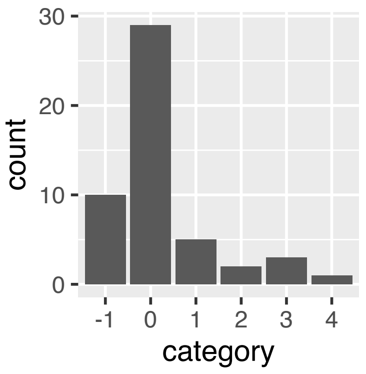


Others: transparency, line type

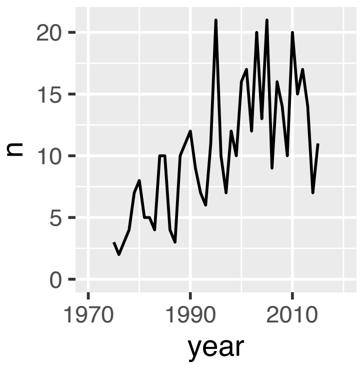
3 Geometric objects are representations of the data, including points, lines, and polygons.

Plots are often called their geometric object(s).

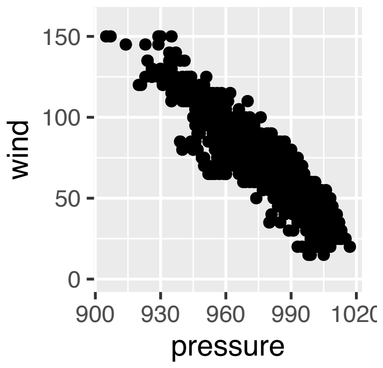
`geom_bar()` or `geom_col()`



`geom_line()`



`geom_point()`



Exercise 2

Step 1: Duplicate the code from exercise 1. Add comments below the data visualization code that describes the argument or function that corresponds to each of the first three components of the grammar of graphics.

Step 2: Inside `aes()`, add `color = category`. Run the code.

Step 3: Replace `color = category` with `color = "green"`. Run the code. What changed? Is this unexpected?

Step 4: Remove `color = "green"` from `aes()` and add it inside inside of `geom_point()` but outside of `aes()`. Run the code.

Step 5: This is a little cluttered. Add `alpha = 0.2` inside `geom_point()` but outside of `aes()`.

Aesthetic mappings like x and y almost always vary with the data. Aesthetic mappings like color, fill, shape, transparency, and size can vary with the data. But those arguments can also be added as styles that don't vary with the data. If you include those arguments in `aes()`, they will show up in the legend (which can be annoying! and is also a sign that something should be changed!).

Exercise 3

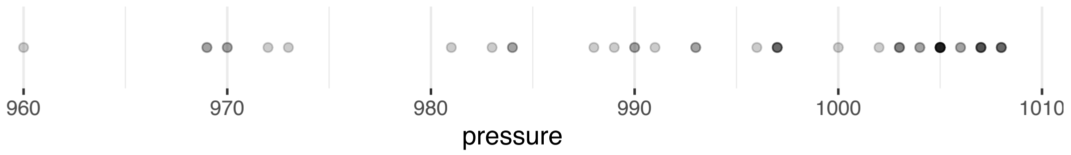
Step 1: Create a new scatter plot using the `msleep` data set. Use `bodywt` on the x-axis and `sleep_total` on the y-axis.

Step 2: The y-axis doesn't contain zero. Below `geom_point()`, add `scale_y_continuous(limits = c(0, NA))`. Hint: add `+` after `geom_point()`.

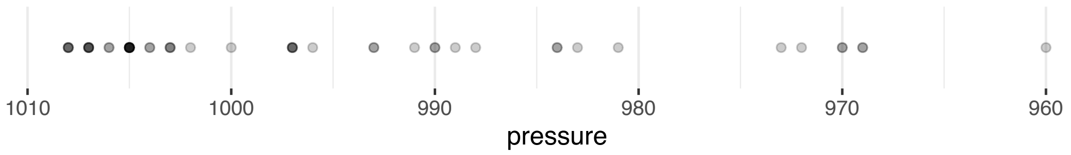
Step 3: The x-axis is clustered near zero. Add `scale_x_log10()` above `scale_y_continuous(limits = c(0, NA))`.

4 Scales turn data values, which are quantitative or categorical, into aesthetic values. This includes not only the x-axis and y-axis, but the ranges of sizes, shapes, and colors of aesthetics.

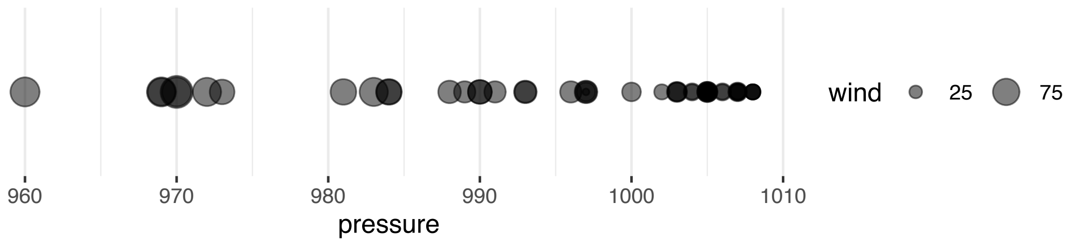
Before `scale_x_continuous()`



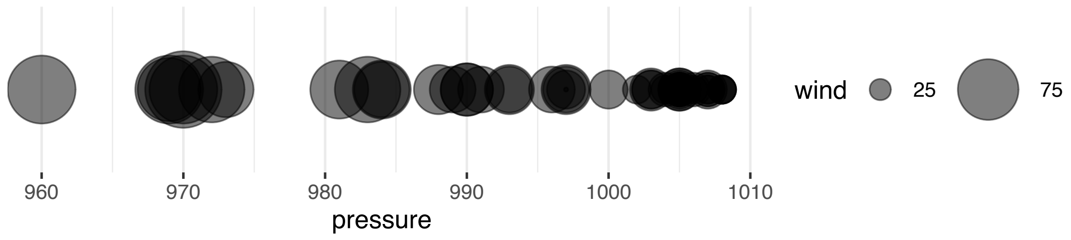
After `scale_x_reverse()`



Before `scale_size_continuous(breaks = c(25, 75, 125))`



After `scale_size_continuous(range = c(0.5, 20), breaks = c(25, 75, 125))`



Exercise 4

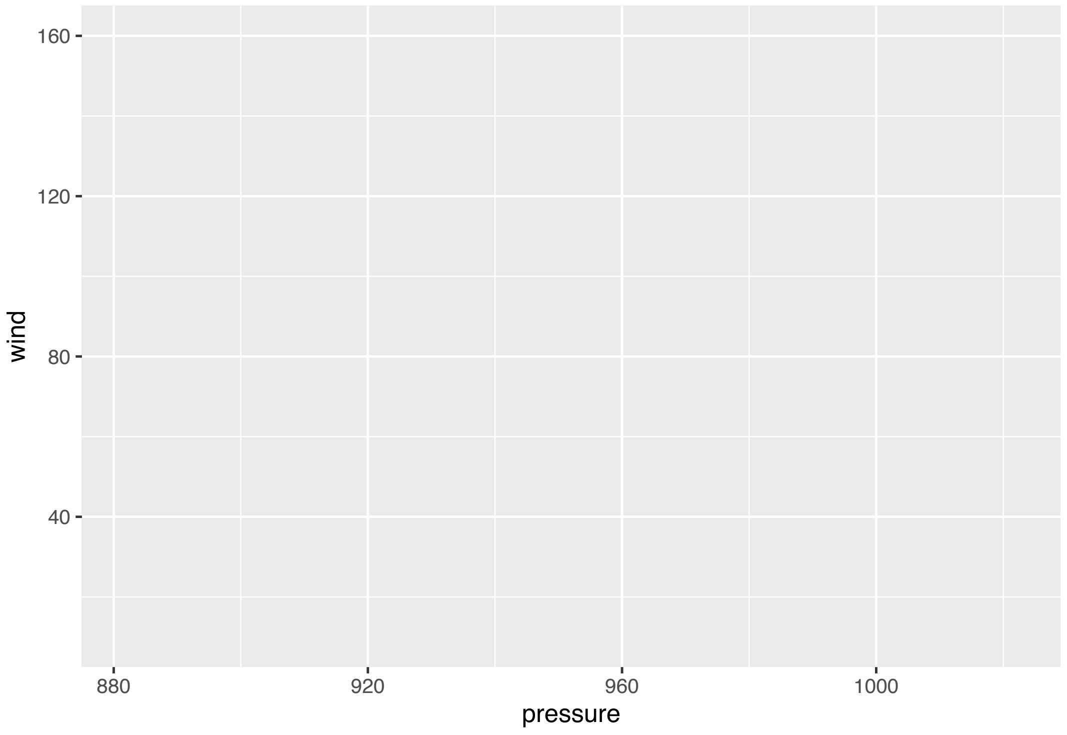
Step 1: Type the following code in your script.

```
data <- tibble(x = 1:10, y = 1:10)
ggplot(data = data) +
  geom_blank(mapping = aes(x = x, y = y))
```

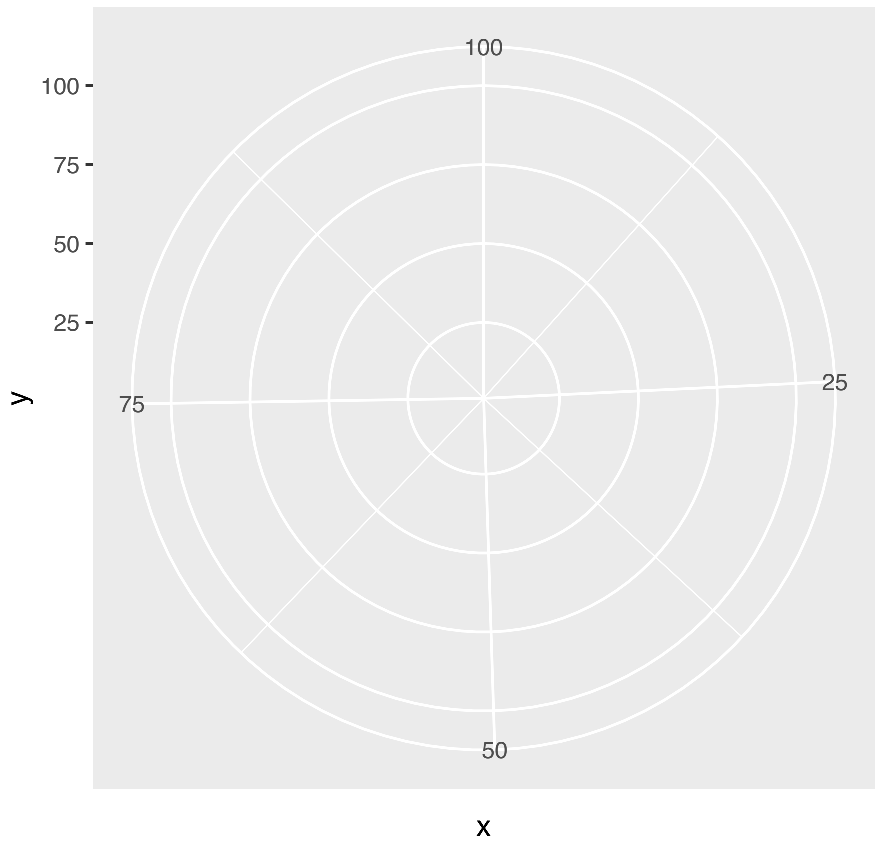
Step 2: Add `coord_polar()` to your plot.

Step 3: Add `labs(title = "Polar coordinate system")` to your plot.

5 Coordinate systems map scaled geometric objects to the position of objects on the plane of a plot. The two most popular coordinate systems are the Cartesian coordinate system and the polar coordinate system.



`coord_polar()`



Exercise 5

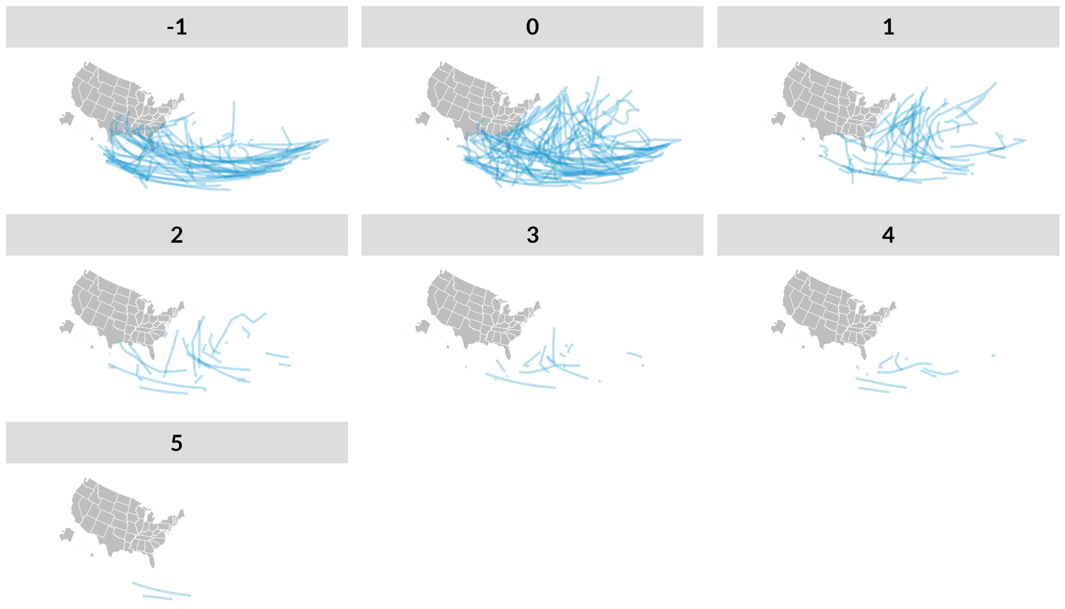
Step 1: Create a scatter plot of the `storms` data set with `pressure` on the x-axis and `wind` on the y-axis.

Step 2: Add `facet_wrap(~ category)`

6 Facets (optional) break data into meaningful subsets. `facet_wrap()`, `facet_grid()`, and `facet_geo()`.

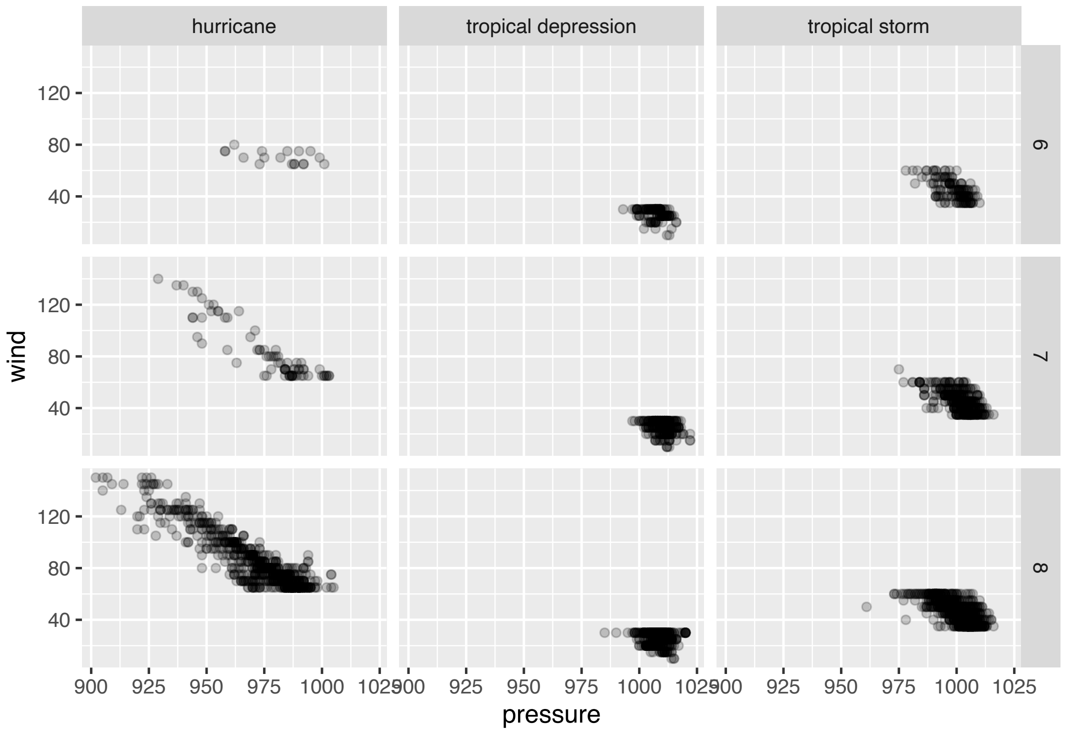
Facet wrap

```
facet_wrap(~ category)
```



Facet grid

```
facet_grid(month ~ status)
```



Exercise 6

Step 1: Add the following code to your script. Submit it!

```
ggplot(storms) +
  geom_bar(mapping = aes(x = category))
```

7 Statistical transformations (optional) transform the data, typically through summary statistics and functions, before aesthetic mapping.

Note: `geom_bar()` performs statistical transformation. Use `geom_col()` to create a column chart with bars that encode individual observations in the data set.

Exercise 7

Step 1: Duplicate Exercise 6.

Step 2: Add `theme_minimal()` to the plot.

Exercise 8

Step 1: Duplicate Exercise 6.

Step 2: Run `install.packages("remotes")` and `remotes::install_github("UrbanInstitute/urbnthemes")` in the console.

Step 3: In the lines preceding the chart add and run the following code:

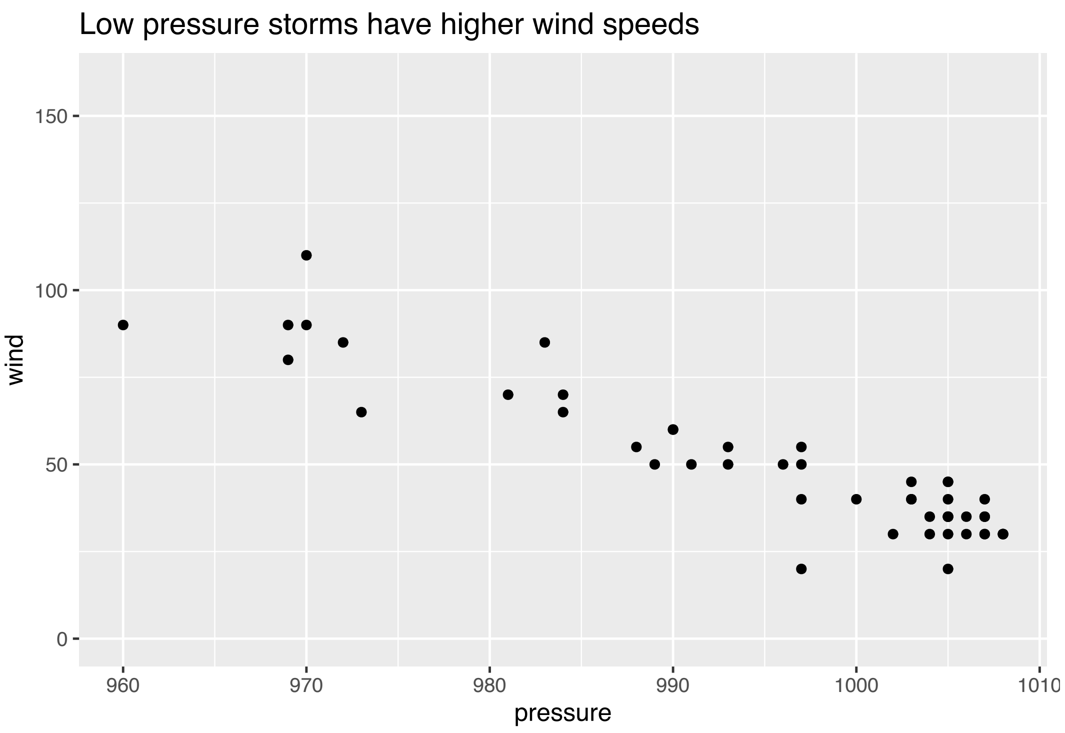
```
library(urbnthemes)
set_urbn_defaults(style = "print")
```

Step 4: Run the code to make the chart.

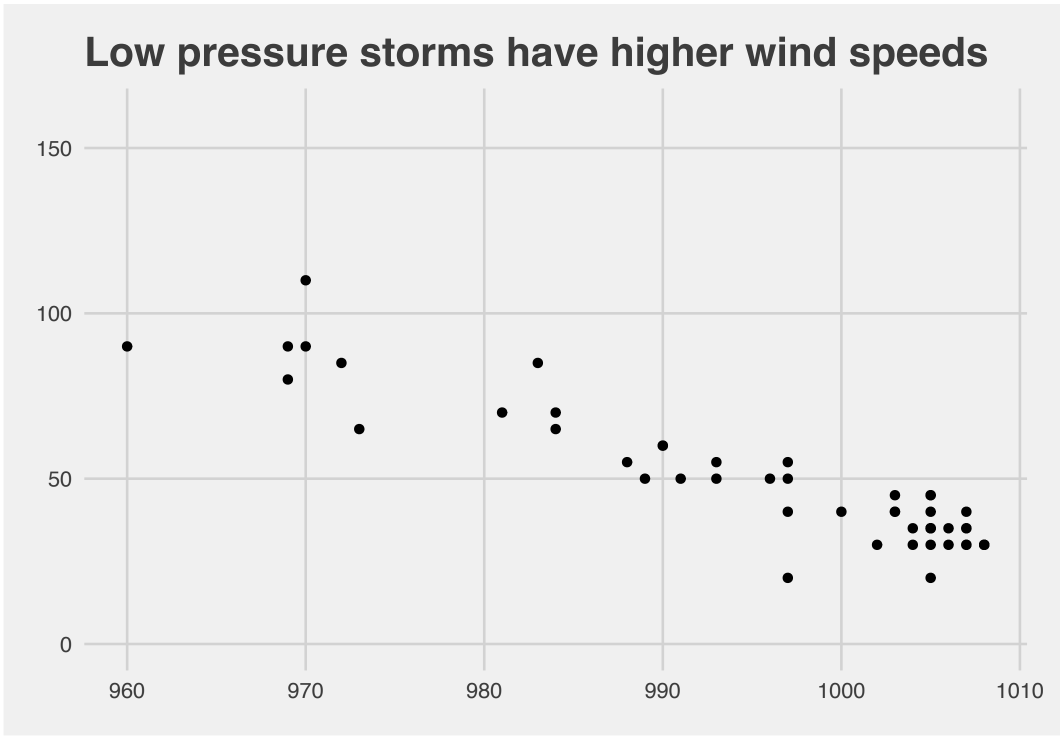
Step 5: Add `scale_y_continuous(expand = expansion(mult = c(0, 0.1)))` and rerun the code.

8 Theme controls the visual style of plot with font types, font sizes, background colors, margins, and positioning.

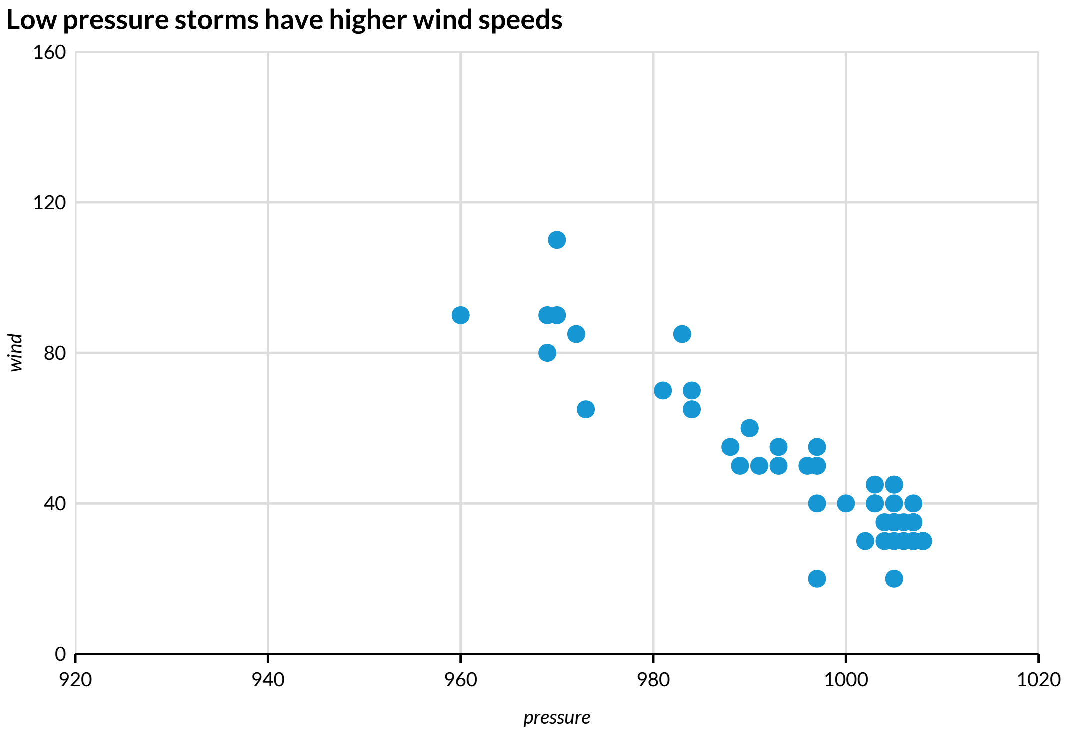
Default theme



fivethirtyeight theme



urbnthemes



Exercise 9 (layers!)

Step 1: Add the following exercise to you script. Run it!

```
storms %>%
  filter(category > 0) %>%
  distinct(name, year) %>%
  count(year) %>%
  ggplot() +
  geom_line(mapping = aes(x = year, y = n))
```

Step 2: Add `geom_point()` after `geom_line()` with the same aesthetic mappings.

Layers allow for multiple geometric objects to be plotted in the same data visualization.

Exercise 10

Step 1: Add the following exercise to you script. Run it!

```
ggplot(data = storms, mapping = aes(x = pressure, y = wind)) +
  geom_point() +
  geom_smooth()
```

Inheritances pass aesthetic mappings from `ggplot()` to later `geom_*()` functions.

Notice how the aesthetic mappings are passed to `ggplot()` *in example 10. This is useful when using layers!*

Exercise 11

Step 1: Pick your favorite plot from exercises 1 through 10 and duplicate the code.

Step 2: Add `ggsave(filename = "favorite-plot.png")` and then save the file. Look at the saved file.

Step 3: Add `width = 6` and `height = 4` to `ggsave()`. Run the code and then look at the saved file.

Functions

- `ggplot()`
- `aes()`
- `geom_*()`
  - `geom_point()`
  - `geom_line()`
  - `geom_col()`
- `scale_*()`
  - `scale_y_continuous()`
- `coord_*()`
- `facet_*()`
- `labs()`

Theory

1. *Data*
2. *Aesthetic mappings*
3. *Geometric objects*
4. *Scales*
5. *Coordinate systems*
6. *Facets*
7. *Statistical transformations*
8. *Theme*

Mapping

- How to Create State and County Maps Easily in R
- library(urbnmapr)
- Urban Institute R Users Group website: mapping

Resources

- Urban Institute R Users Group website
- Why the Urban Institute visualizes data with ggplot2
- R for Data Science: data visualization
- awunderground themes
- R Graph Gallery