

# Lecture 5: Univariate visualizations

## Stats 32: Introduction to R for Undergraduates

Harrison Li

April 16, 2024

# Agenda

- 1 The grammar of graphics
- 2 `ggplot()` call structure
- 3 Bar plots
- 4 `aes()`
- 5 Histograms
- 6 Boxplots

Reading: Sections 2.1, 2.2, 2.5, 2.7, and 2.8

# The grammar of graphics

# The grammar of graphics

The ggplot2 package is based around a *grammar of graphics* pioneered by Leland Wilkinson:

A statistical graphic is a mapping of data variables to aesthetic attributes of geometric objects.

# The grammar of graphics

A statistical graphic is a mapping of data variables to aesthetic attributes of geometric objects.

Three components of a graphic:

- 1 **Data** – Self-explanatory
- 2 **Geometries** – The geometric attributes of the graphic, e.g. points, lines, bars
- 3 **Aesthetics** – The aesthetic attributes of the geoms, e.g. color, shape, size, line type

# Geom list

5 common geoms (“5NG” in the book):

- 1 `geom_bar()` or `geom_col()`: Draws bar plots
- 2 `geom_boxplot()`: Draws boxplots
- 3 `geom_histogram()`: Draws histograms
- 4 `geom_point()`: Draws points, e.g. for scatterplots
- 5 `geom_line()`: Draws line plots

Very useful reference: [ggplot2 cheatsheet](#)

ggplot() call structure

# ggplot() call structure

```
my_tibble %>%  
  ggplot(mapping=aes(x=x_varname, y=y_varname)) +  
    geom_1(mapping=aes(aes1=var1_name)) +  
    geom_2(mapping=aes(aes2=var2_name))
```

- 1 Pipe the tibble containing your **data** into `ggplot()`
- 2 Set up your axes within the call to `ggplot()` by specifying the column(s) of the tibble to be plotted as the x and/or y **aesthetics**
- 3 End the line with a `+`, then on the next line call first desired **geom**, specifying desired non-default **aesthetics** (e.g. line width, fill color)
- 4 Repeat step 3 for all desired **geoms**.

Easiest way to learn: examples!



Bar plots

# Bar plots

**Bar plots** are used to visualize frequencies of a *single categorical variable*.

Distribution of origin airport among NYC flights:

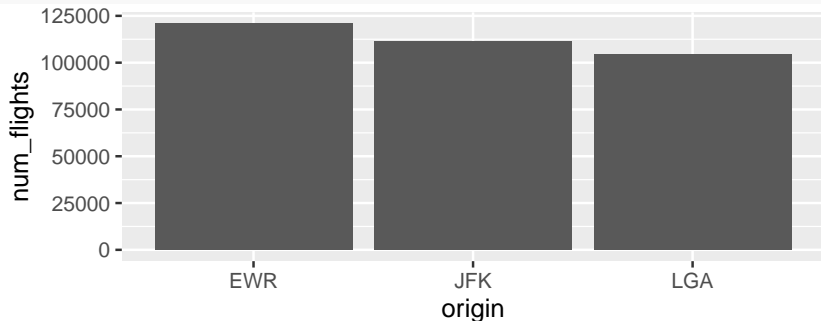
```
library(tidyverse)
library(nycflights13)
origins <- flights %>%
  group_by(origin) %>%
  summarise(num_flights=n())
origins
```

```
## # A tibble: 3 x 2
##   origin num_flights
##   <chr>      <int>
## 1 EWR      120835
## 2 JFK      111279
## 3 LGA      104662
```

## geom\_col()

If our tibble has a column corresponding to the heights of the bars we want to display, we can generate a bar plot using `geom_col()`:

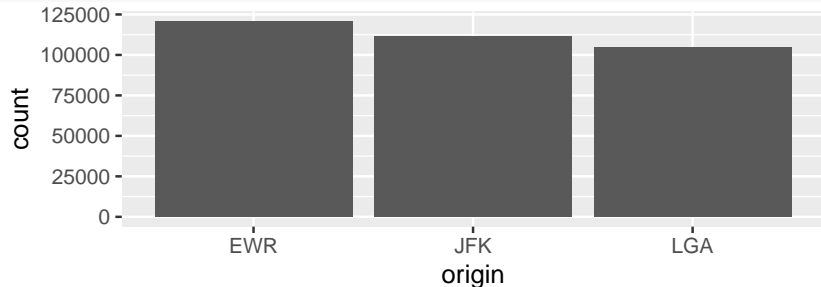
```
origins %>%  
  ggplot(aes(x=origin, y=num_flights)) +  
  geom_col()
```



## geom\_bar()

`geom_bar()` can produce a bar plot when you don't have a column with the bar heights. In particular, `geom_bar()` assumes you want to plot frequencies, and does the counting for you automatically. In this case this allows us to use the original `flights` tibble directly for plotting:

```
flights %>%  
  ggplot(aes(x=origin)) +  
  geom_bar()
```



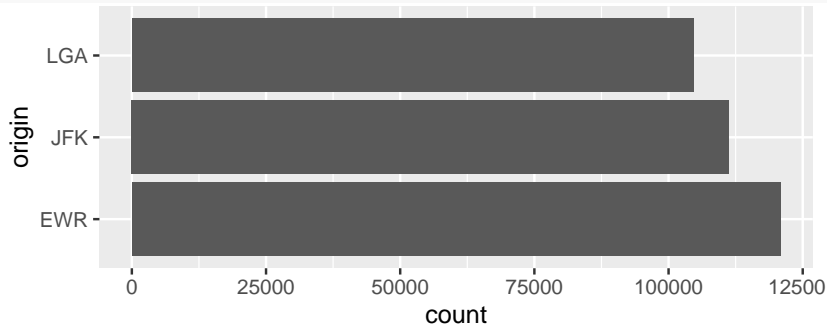
## geom\_bar()

Note we did not specify a y aesthetic for `geom_bar()`'. Why do you think that is okay/expected?

What happens if you specify only the y aesthetic, and no x aesthetic?

# geom\_bar()

```
flights %>%  
  ggplot(aes(y=origin)) +  
  geom_bar()
```



aes()

# aes()

The default aesthetics for `geom_col()` appear to be grey bars.  
Let's try to make the plot more colorful.

`geom_col()` understands the following aesthetics (required aesthetics are in bold):

- **x**
- **y**
- alpha
- colour
- fill
- group
- linetype
- size

Learn more about setting these aesthetics in `vignette("ggplot2-specs")`.

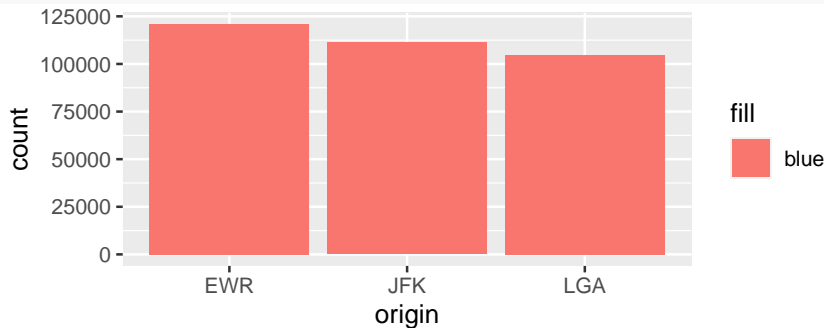
---



# aes()

Colour refers to the color of the *outline* of the bars. Use `fill` to specify the color of the *interior* of the bars.

```
flights %>%  
  ggplot(aes(x=origin)) +  
  geom_bar(aes(fill="blue"))
```



# aes()

Why are the bars red, even though I specified `fill="blue"`?  
Help page for `aes()` (emphasis mine):

Aesthetic mappings describe how **variables in the data** are mapped to visual properties (**aesthetics**) of **geoms**

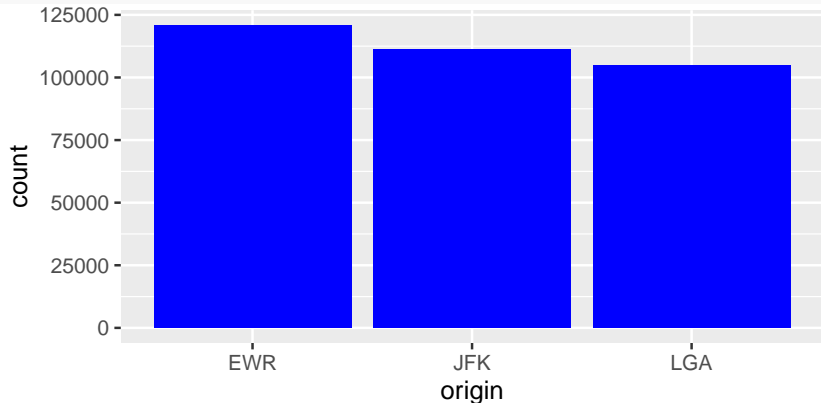
Upshot: only use `aes()` to set aesthetics whose values are *based on data variables*.

*Do not specify static (non data-dependent) aesthetic values inside `aes()`.*

# aes()

Static blue-filled bars from outside the tibble:

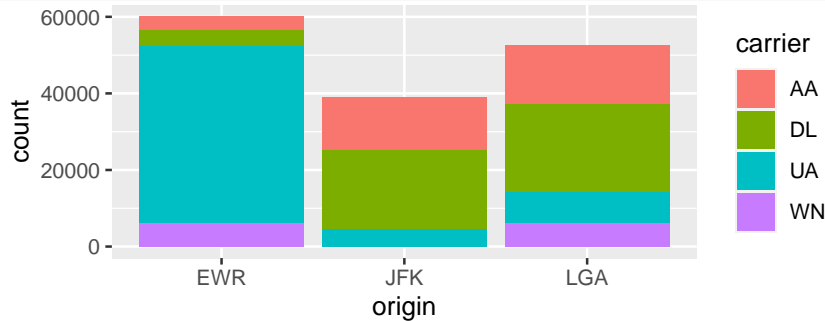
```
flights %>%  
  ggplot(aes(x=origin)) +  
  geom_bar(fill="blue")
```



# aes()

Let's specify a different fill color by carrier (I filtered out to only the four major carriers, for readability). To change the colors, you will need to deal with color palettes (see Lab 5):

```
flights %>%  
  filter(carrier %in% c("UA", "DL", "AA", "WN")) %>%  
  ggplot(aes(x=origin)) +  
  geom_bar(aes(fill=carrier))
```



# Histograms

# Histograms

A **histogram** is used to visualize the distribution of a *single quantitative variable*.

It does so by “binning” the data, which means transforming the quantitative variable into a categorical one by dividing the range of the data into a few distinct “buckets” (or bins) of consecutive values.

# Histograms

A natural example of binning occurs in every letter graded course. Professors take students' numerical grades and bin them into the "buckets" A-, B+, C, etc.

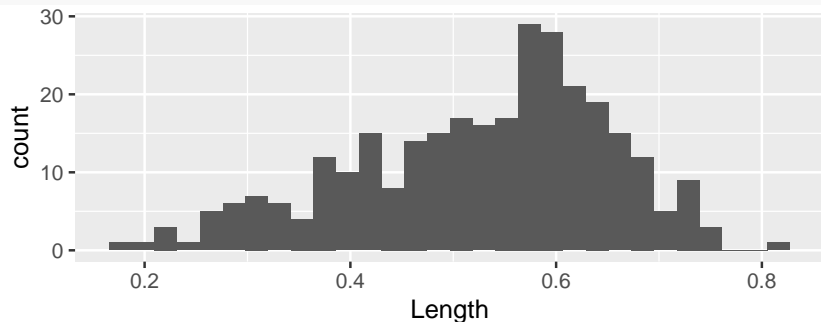
A histogram will then show bars whose heights correspond to the number of observations in each bucket.

It is not a bar plot; in a bar plot, the horizontal axis is not quantitative.

# Histograms

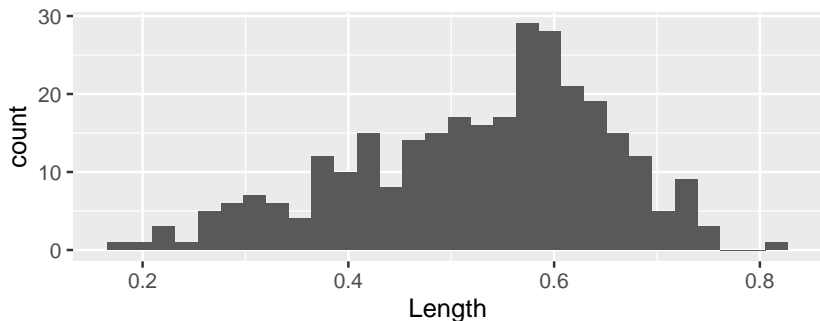
Histogram of abalone lengths:

```
abalone <- read_csv("abalone.csv")  
abalone %>%  
  ggplot(aes(x=Length)) +  
  geom_histogram()
```





# Histograms



The most common abalone length is around 0.6, but typically ranges from about 0.2 to 0.8.

# Boxplots

# Boxplots

A **boxplot** is an alternative way to visualize the distribution of a *single quantitative variable*.

Boxplots indicate the first and third quartiles (25th and 75th percentiles) of your data, in addition to the median (50th percentile) and *outliers* - data points far from the typical value. This is explored in more depth in the lab.

# Boxplots

Here is a boxplot of abalone lengths using `geom_boxplot()`:

```
abalone %>%  
  ggplot(aes(y=Length)) +  
  geom_boxplot()
```

