



# Creating Data Visualizations with Python

---

November 2024

# Housekeeping

- In case of technical problems:

- Something wrong on my end (e.g. power outage), I will send you an email.
- Something wrong on your end, please send me a text message. 508-769-6446
- jcodygroup@gmail.com

- For the session

- I will try to give you an opportunity to stand and stretch every hour.
- We will take at least one 15-minute break near the halfway point of the morning and afternoon.

# About me

## ■ Experience:

- 25+ years consulting and training experience
- Extensive work with “big data” and analytics
- 15 years working with various data visualization tools

## ■ Education

- Ed. M., Technology, Innovation & Education, Harvard University
- PhD Candidate, Education Policy, University of Massachusetts, Amherst

# Learning Objective

## Knowledge, skillsets, mindsets

### ■ Knowledge of:

- A variety of python data visualization packages
- The structure of visualizations
- Package documentation

### ■ Skillsets:

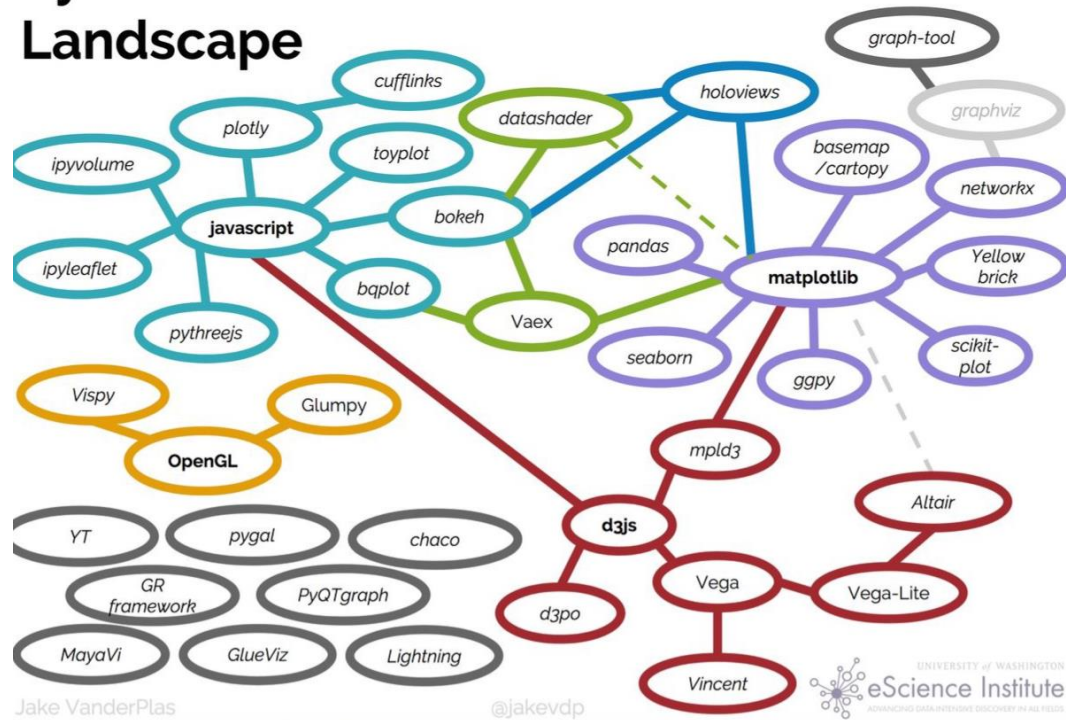
- The ability to produce basic plots with some formatting using matplotlib & seaborn

### ■ Mindsets:

- Think, sketch before coding

Note: We will be focusing on plotting. We will not be doing any data manipulation to prepare for plotting.

## Python's Visualization Landscape



# matplotlib

Version 3.4.3

seaborn: statistical data visualization

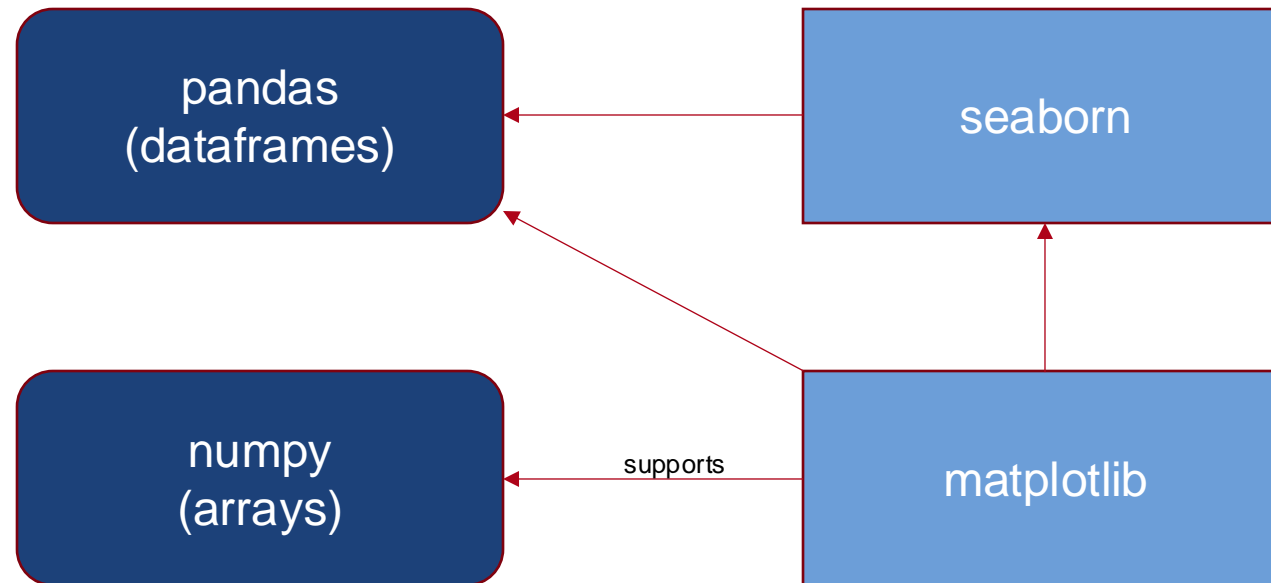
## The Bokeh Visualization Library

plotnine 0.8.0 API Gallery Tutorials Site Page

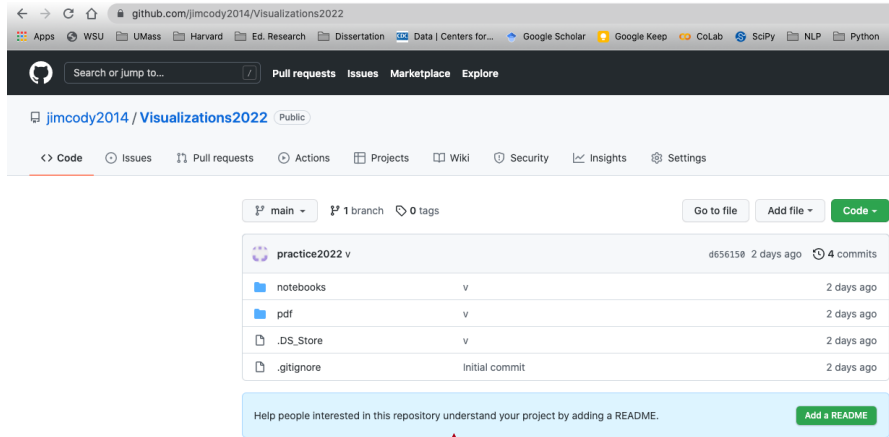
### A Grammar of Graphics for Python

### Altair: Declarative Visualization in Python

# 'Workhorse' data & visualization packages

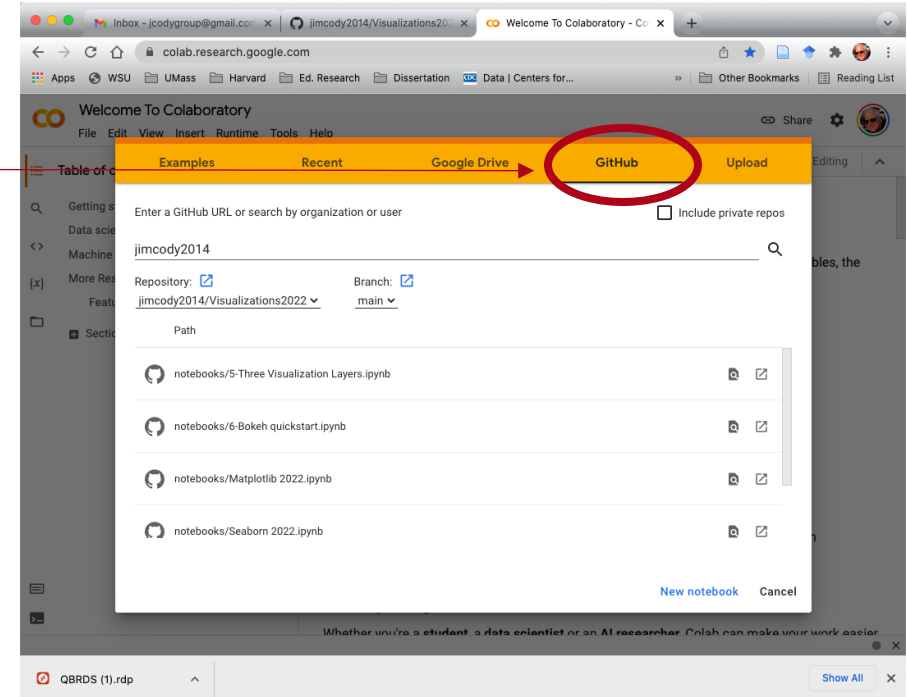


# Accessing jupyter notebooks for class



git push

Jim's Desktop



# You can also download from github (this is optional!)

The screenshot shows the GitHub interface for the repository 'jimcody2014 / Visualizations2022'. The repository is public and has 1 branch and 0 tags. The 'Code' button is highlighted, and a dropdown menu is open showing options to clone the repository using HTTPS, SSH, or GitHub CLI, to open it with GitHub Desktop, or to download it as a ZIP file. The repository's file list includes 'notebooks', 'pdf', '.DS\_Store', and '.gitignore'. The 'About' section indicates no description, website, or topics are provided. The 'Releases' section shows no releases published, and the 'Packages' section shows no packages published.

Search or jump to... Pull requests Issues Marketplace Explore

jimcody2014 / Visualizations2022 Public

Pin Unwatch 1 Fork 0 Star 0

<> Code Issues Pull requests Actions Projects Wiki Security Insights

main 1 branch 0 tags

Go to file Add file Code

practice2022 v

- notebooks v
- pdf v
- .DS\_Store v
- .gitignore Initial comm

Help people interested in this repository understand your project by adding a README. Add a README

Clone ?

HTTPS SSH GitHub CLI

<https://github.com/jimcody2014/Visuali>

Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP

About

No description, website, or topics provided.

0 stars

1 watching

0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

<https://github.com/jimcody2014/Visualizations2022/archive/refs/heads/main.zip>





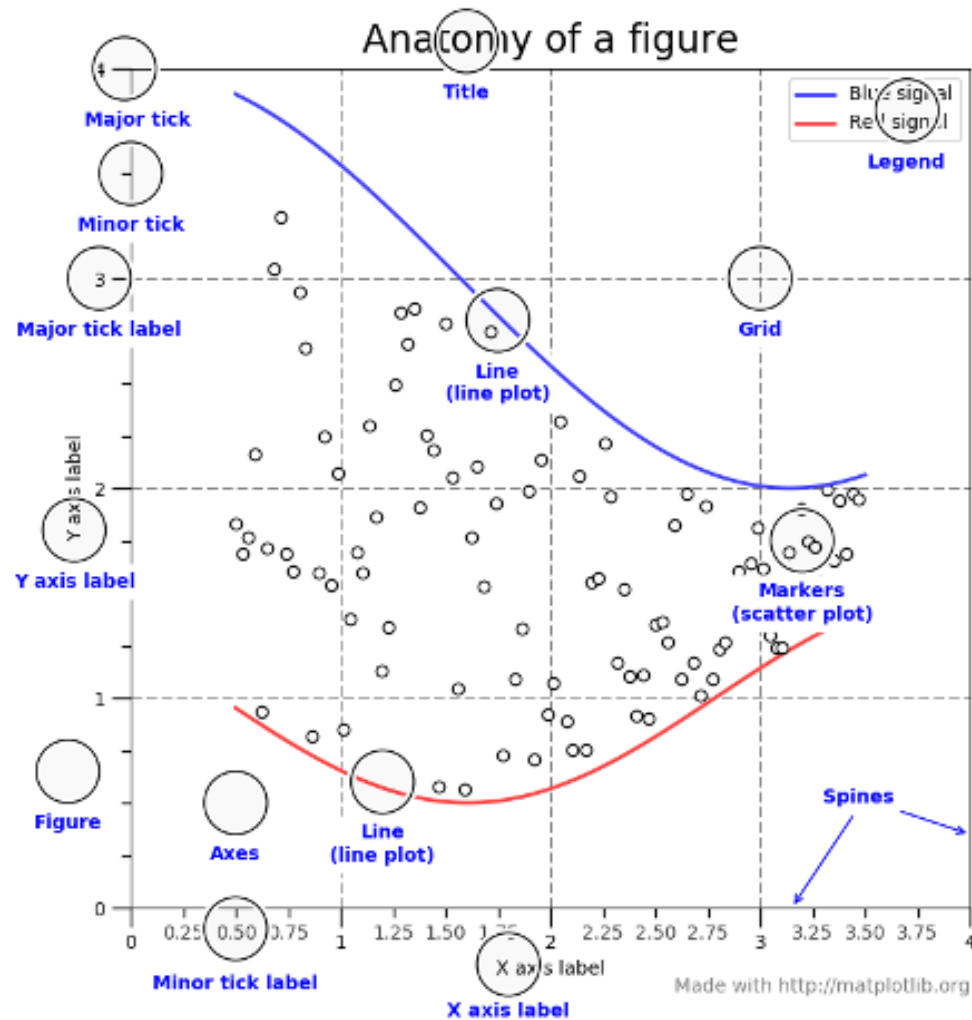
# Matplotlib

---

# Matplotlib

- Matplotlib is a popular Python package used to build plots.
- It was started as a project in the early 2000's to replicate MATLAB's plotting capability with Python.
- Matplotlib has three interfaces (ways to write code)
  - pyplot – hides the complexity of object-oriented coding.
  - object-oriented – provides access to more functions and control over the visualizations
  - pylab (not recommended for use)
- This lesson covers the pyplot and object-oriented interfaces

# Two big concepts to keep in mind



Figure

Axes

Plot

Figure

Axes

Plot 1

Plot 2

Figure

Axes

Plot

Axes

Plot 1

Plot 2

# Objects – modules, methods and parameters

matplotlib.figure

matplotlib.figure.Figure

matplotlib.figure.Figure.add\_axes

matplotlib.figure.Figure.add\_subplot

matplotlib.figure.Figure.subplots

matplotlib.figure.Figure.subplot\_mosaic

matplotlib.figure.Figure.add\_gridspec

matplotlib.figure.Figure.get\_axes

matplotlib.figure.Figure.axes

matplotlib.figure.Figure.delaxes

matplotlib.figure.Figure.subfigures

matplotlib.figure.Figure.add\_subfigure

```
class matplotlib.figure.Figure(figsize=None, dpi=None, *, facecolor=None,
edgecolor=None, linewidth=0.0, frameon=None, subplotpars=None,
tight_layout=None, constrained_layout=None, layout=None, **kwargs) \[source\]
```

The top level container for all the plot elements.

## matplotlib.figure.Figure.subplots

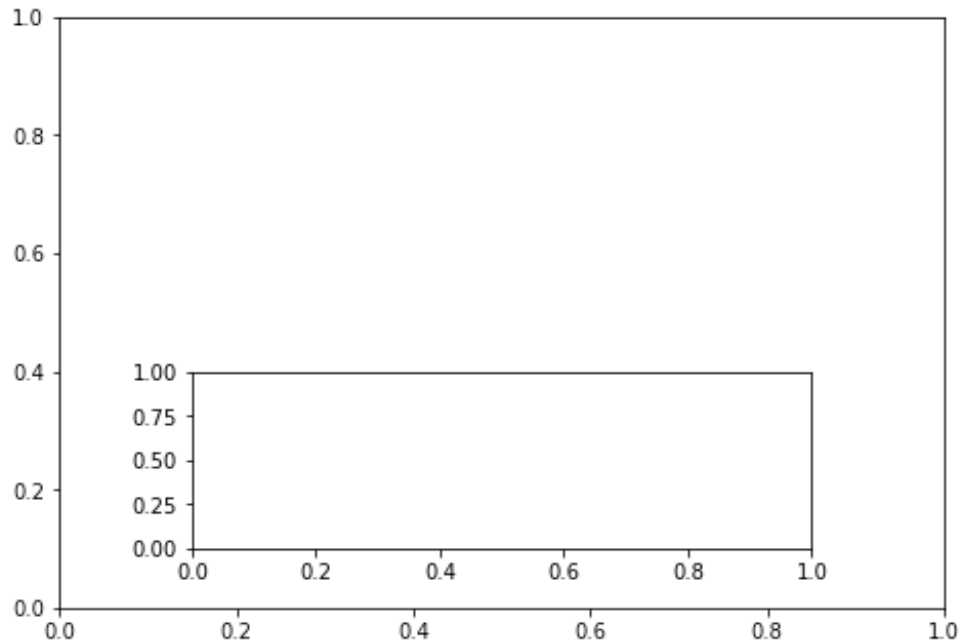
```
Figure.subplots(nrows=1, ncols=1, *, sharex=False, sharey=False,
squeeze=True, width_ratios=None, height_ratios=None, subplot_kw=None,
gridspec_kw=None) \[source\]
```

Add a set of subplots to this figure.

# Adding axes

`fig.add_axes`: for absolute positioning

```
fig = plt.figure()
ax1 = fig.add_axes([0,0,1,1]) # left, bottom, width, height
ax2 = fig.add_axes([0.15, 0.1, 0.7, 0.3])
```

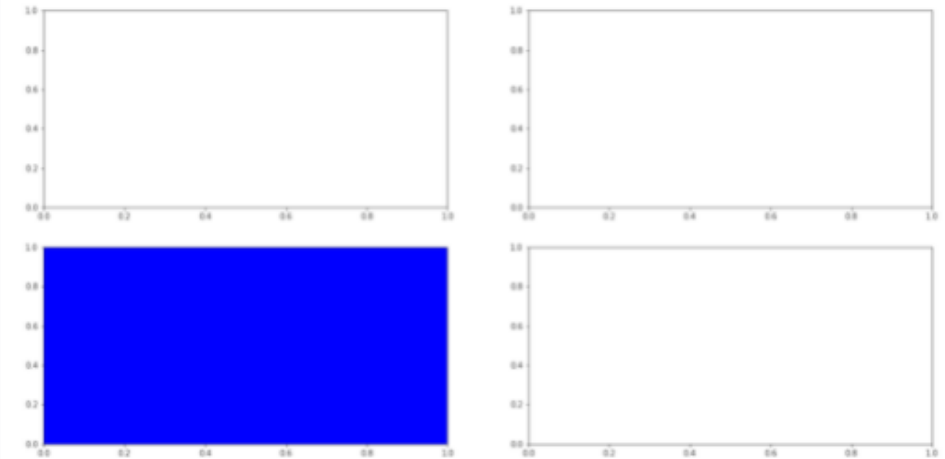


`fig.add_subplot`: for grid positioning

```
# Initialize the plot
fig = plt.figure(figsize=(20,10))
ax1 = fig.add_subplot(2,2,1)
ax2 = fig.add_subplot(222)
ax3 = fig.add_subplot(223)
ax4 = fig.add_subplot(224)

ax3.set_facecolor('blue')

# Show the plot
# plt.show()
```



# Subplots using pyplot

<code>subplot</code>	Add an Axes to the current figure or retrieve an existing Axes.
<code>subplot2grid</code>	Create a subplot at a specific location inside a regular grid.
<code>subplot_mosaic</code>	Build a layout of Axes based on ASCII art or nested lists.
<code>subplot_tool</code>	Launch a subplot tool window for a figure.
<code>subplots</code>	Create a figure and a set of subplots.
<code>subplots_adjust</code>	Adjust the subplot layout parameters.

```
plt.subplot(221)

# equivalent but more general
ax1 = plt.subplot(2, 2, 1)

# add a subplot with no frame
ax2 = plt.subplot(222, frameon=False)

# add a polar subplot
plt.subplot(223, projection='polar')

# add a red subplot that shares the x-axis with ax1
plt.subplot(224, sharex=ax1, facecolor='red')

# delete ax2 from the figure
plt.delaxes(ax2)

# add ax2 to the figure again
plt.subplot(ax2)

# make the first axes "current" again
plt.subplot(221)
```

from the documentation

```
# First create some toy data:
x = np.linspace(0, 2*np.pi, 400)
y = np.sin(x**2)

# Create just a figure and only one subplot
fig, ax = plt.subplots()
ax.plot(x, y)
ax.set_title('Simple plot')

# Create two subplots and unpack the output array immediately
f, (ax1, ax2) = plt.subplots(1, 2, sharey=True)
ax1.plot(x, y)
ax1.set_title('Sharing Y axis')
ax2.scatter(x, y)

# Create four polar axes and access them through the returned array
fig, axs = plt.subplots(2, 2, subplot_kw=dict(projection="polar"))
axs[0, 0].plot(x, y)
axs[1, 1].scatter(x, y)

# Share a X axis with each column of subplots
plt.subplots(2, 2, sharex='col')

# Share a Y axis with each row of subplots
plt.subplots(2, 2, sharey='row')

# Share both X and Y axes with all subplots
plt.subplots(2, 2, sharex='all', sharey='all')

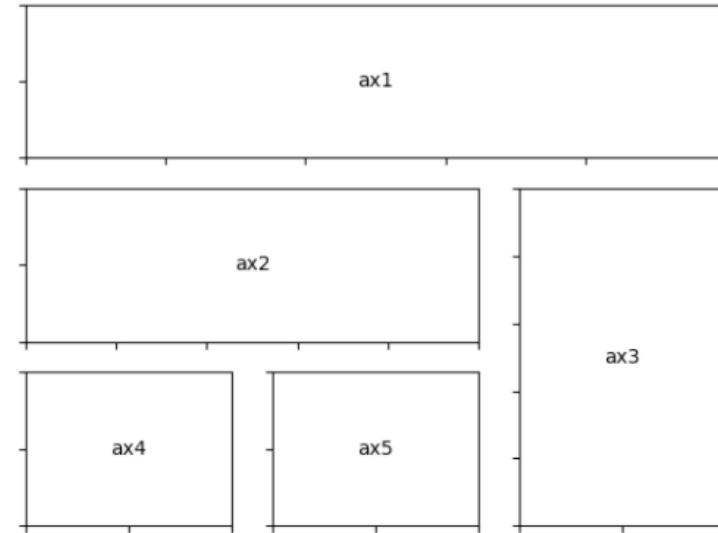
# Note that this is the same as
plt.subplots(2, 2, sharex=True, sharey=True)

# Create figure number 10 with a single subplot
# and clears it if it already exists.
fig, ax = plt.subplots(num=10, clear=True)
```

from the documentation

# Subplots using pyplot

<code>subplot</code>	Add an Axes to the current figure or retrieve an existing Axes.
<code>subplot2grid</code>	Create a subplot at a specific location inside a regular grid.
<code>subplot_mosaic</code>	Build a layout of Axes based on ASCII art or nested lists.
<code>subplot_tool</code>	Launch a subplot tool window for a figure.
<code>subplots</code>	Create a figure and a set of subplots.
<code>subplots_adjust</code>	Adjust the subplot layout parameters.



```
import matplotlib.pyplot as plt

def annotate_axes(fig):
    for i, ax in enumerate(fig.axes):
        ax.text(0.5, 0.5, "ax%d" % (i+1), va="center", ha="center")
        ax.tick_params(labelbottom=False, labelleft=False)

fig = plt.figure()
ax1 = plt.subplot2grid((3, 3), (0, 0), colspan=3)
ax2 = plt.subplot2grid((3, 3), (1, 0), colspan=2)
ax3 = plt.subplot2grid((3, 3), (1, 2), rowspan=2)
ax4 = plt.subplot2grid((3, 3), (2, 0))
ax5 = plt.subplot2grid((3, 3), (2, 1))

annotate_axes(fig)

plt.show()
```

from the documentation



# Seaborn

---



# Seaborn

- Seaborn is a library for making statistical graphics in Python.
- It builds on top of matplotlib and integrates closely with pandas data structures.
- Seaborn plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.
- Its dataset-oriented, declarative API lets you focus on what the different elements of your plots mean, rather than on the details of how to draw them.

# Grammar of Graphics

