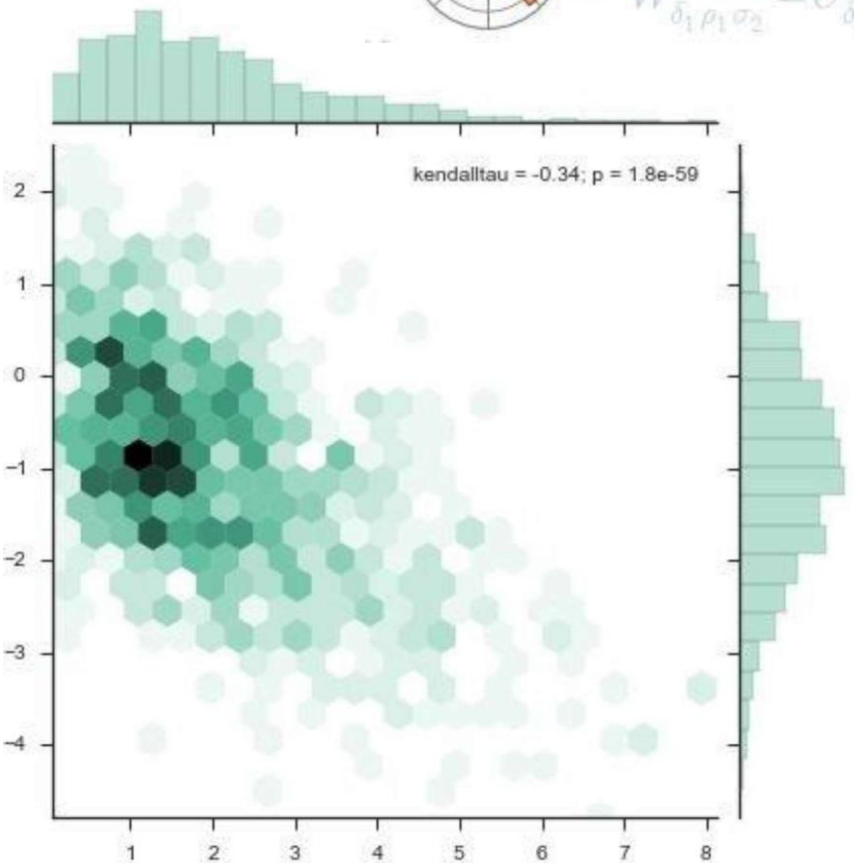
Matplotlib

These slides are based on:
https://www.slideshare.net/bzamecnik/introduction-to-plotting-in-python
https://gist.github.com/bzamecnik/b58579e319287abcb3ca
by Bohumír Zámečník

and
https://www.scikit-yb.org/en/latest/matplotlib.html

What is matplotlib ?



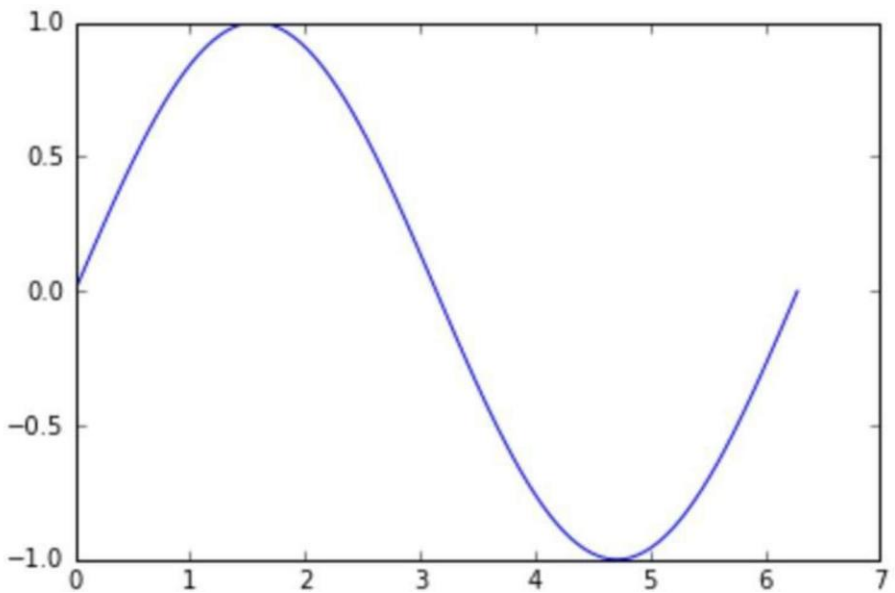
http://stanford.edu/~mwaskom/software/seaborn/examples/hexbin_marginals.html

- Python package
- for 2D plotting
- publication quality
- &interactive plots
- very powerful
- very popular
- many extensions

Hello, world!

in matplotlib

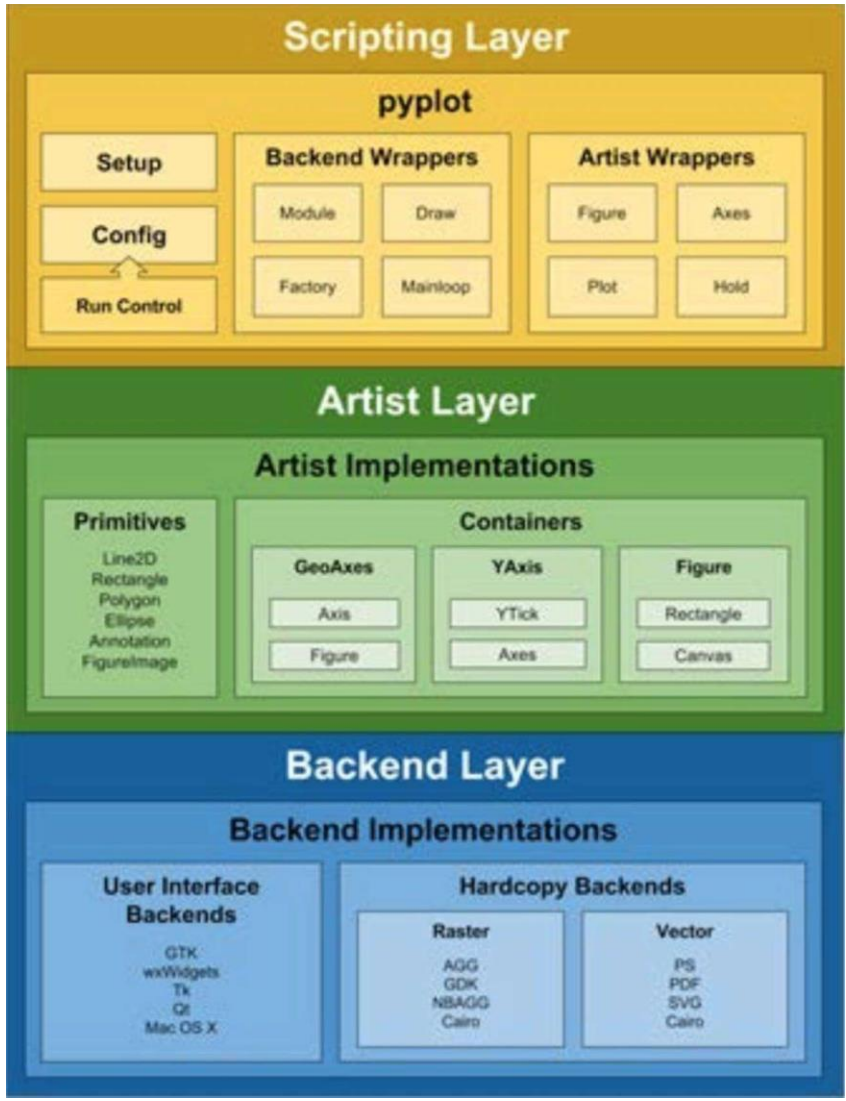
In [3]:
```
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 2 * np.pi, 100)
y = np.sin(x)
plt.plot(x, y);
```



Motto:
"Make easy things easy and hard things possible."

Architecture

- scripting layer
- artist layer
- backend layer



from book:
Mastering matplotlib
by Duncan M. McGreggor
Packt Publishing, 2015

Scripting layer

- pyplot
  - syntax sugar
  - stateful API
  - high-level object API
  - you typically use this

Artist layer

- what should be rendered?
- parts of the plot
- object-oriented API
- primitives - Line2D, Rectangle, Text, Image
- **containers**
  - **Figure** - full plot
  - **Axes** - single subplot
  - **Axis** - one axis

Backend layer

- how it should be rendered?
- interactive
  - Tk, GTK, Qt, OS X, WX
- hardcopy
  - raster – AGG, GDK, NBAGG, Cairo
  - vector – PS, PDF, SVG, Cairo
- FigureCanvas - area where Figure is drawn
- Renderer - knows how to draw

Matplotlib - basics

https://gist.github.com/bzamecnik/b58579e319287abcb3ca

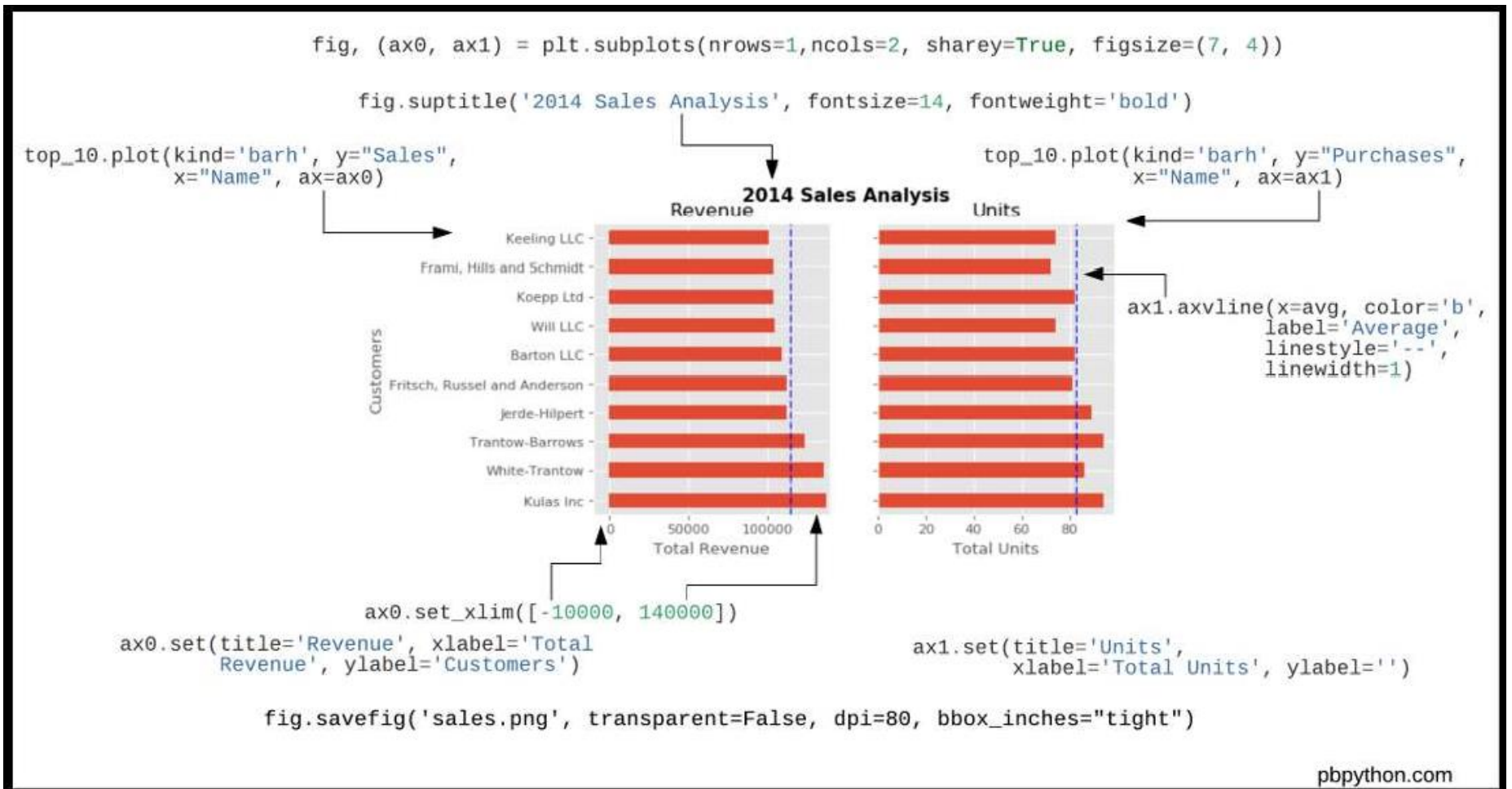
```
# show the plots in Jupyter output
%matplotlib inline
# import the object-oriented API, 'plt' is a conventional alias
import matplotlib.pyplot as plt
import numpy as np
fig, axes = plt.subplots(2, 2, figsize=(10, 5))
# subplots(nrows, ncols), axes[0,0] = top-left subplot
```

Plt.subplots(1,1) is a convenient way to get figure object

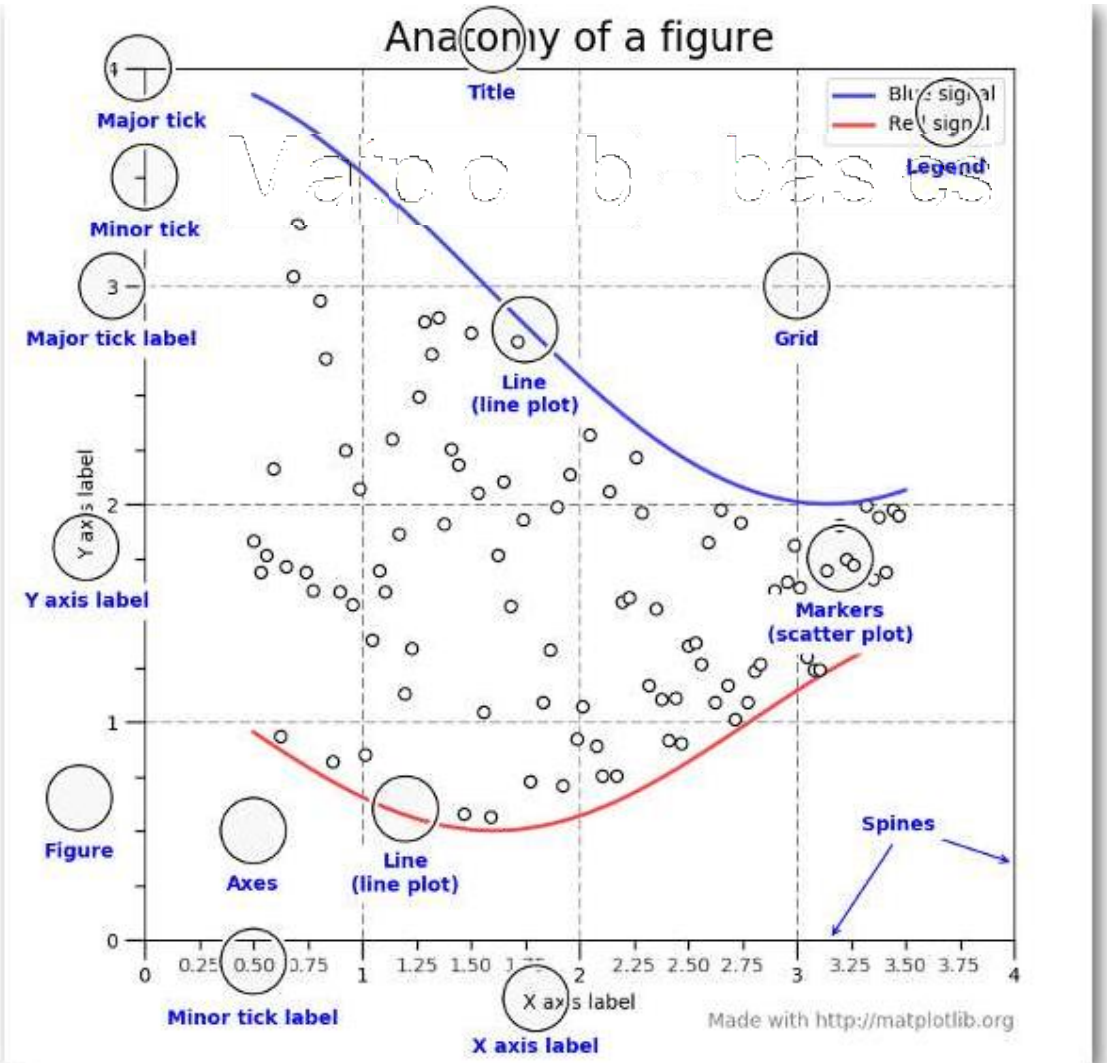
Matplotlib – plot types

Bar, barh, errorbar, hist
Boxplot, violinplot
Coherence
Csd (cross spectral density), psd, Specgram
Hist2d, matshow, xcorr
Pie,
plot, stem, fill, Scatter, plot_date
quiver (arrows)
Specgram
streamplot

https://matplotlib.org/3.1.1/tutorials/introductory/sample_plots.html



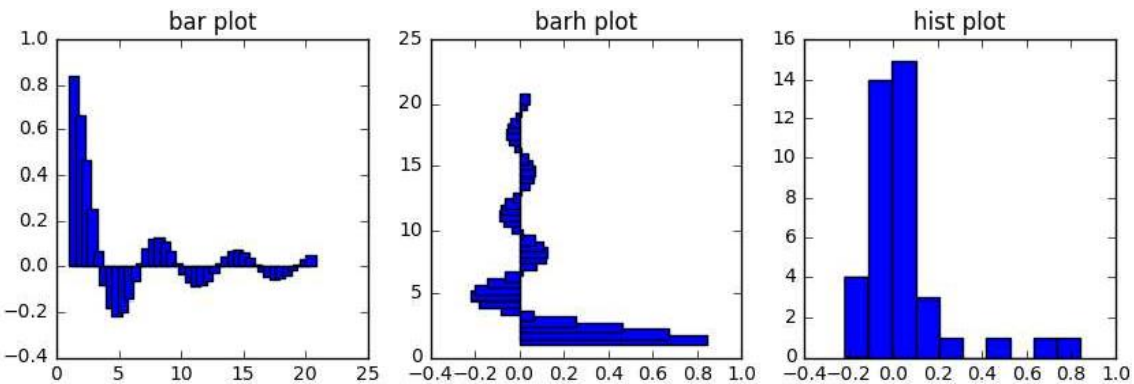
https://www.scikit-yb.org/en/latest/matplotlib.html



https://www.scikit-yb.org/en/latest/matplotlib.html

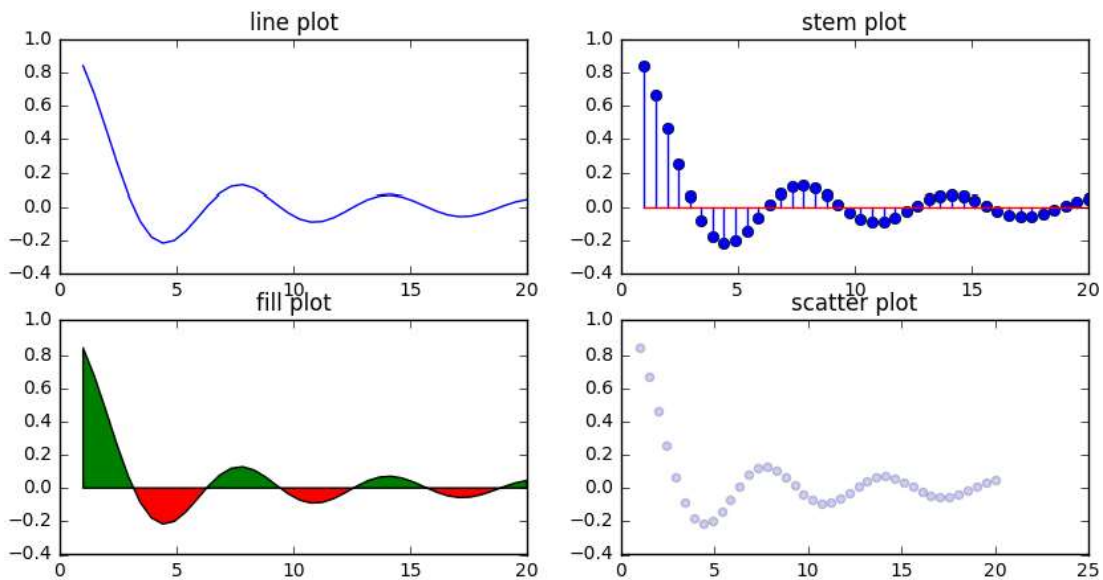
Matplotlib – plot types

```
fig, axes = plt.subplots(1, 3, figsize=(10, 3))

x = np.linspace(1,20, 40)
y = np.sin(x) / x
axes[0].bar(x, y)
axes[0].set_title("bar plot")
axes[1].barh(x, y)
axes[1].set_title("barh plot")
axes[2].hist(y)
axes[2].set_title("hist plot")
plt.savefig("f1.png")
```



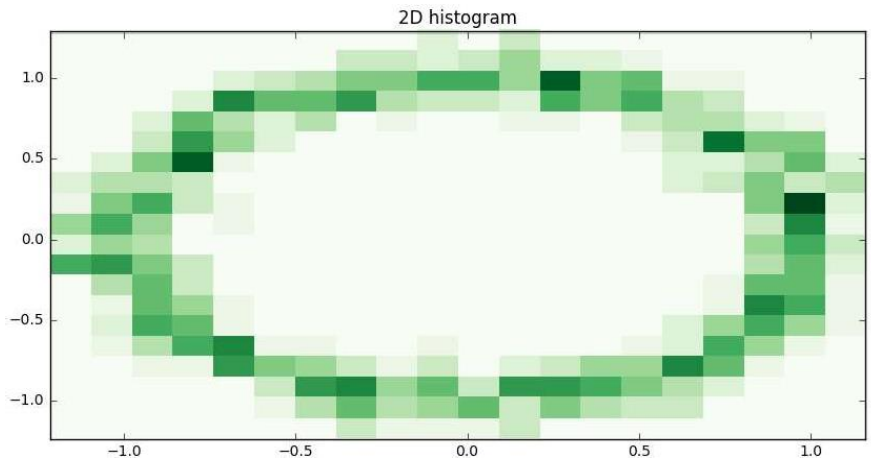
Matplotlib – plot types

```
fig, axes = plt.subplots(2, 2)

axes[0,0].plot(x, y)
axes[0,0].set_title("line plot")
axes[0,1].stem(x, y)
axes[0,1].set_title("stem plot");

axes[1,0].fill_between(x, 0, y, where=y >= 0,
   facecolor='green', interpolate=True)
axes[1,0].fill_between(x, 0, y, where=y < 0,
   facecolor='red', interpolate=True)
axes[1,0].set_title("fill plot");

axes[1,1].scatter(x, y, alpha=0.2)
axes[1,1].set_title("scatter plot")
```



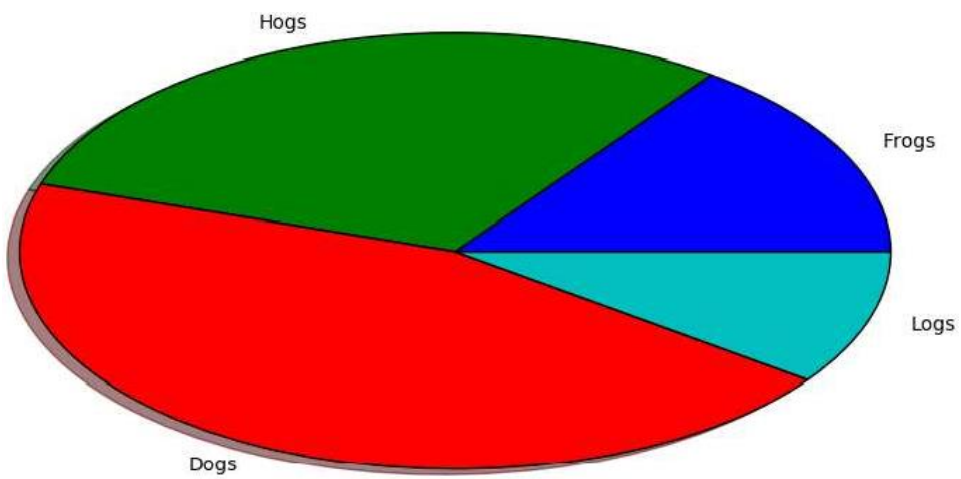
Matplotlib – plot types

```
x = np.linspace(-4 * np.pi, 4 * np.pi, 800)
y1 = np.sin(x)
   + np.random.normal(scale=0.1, size=len(x))
y2 = np.cos(x)
   + np.random.normal(scale=0.1, size=len(x))

plt.hist2d(y1, y2, bins=20, cmap=plt.cm.Greens)
#cmap = color map
plt.gca().set_title("2D histogram")
#get current axes
```



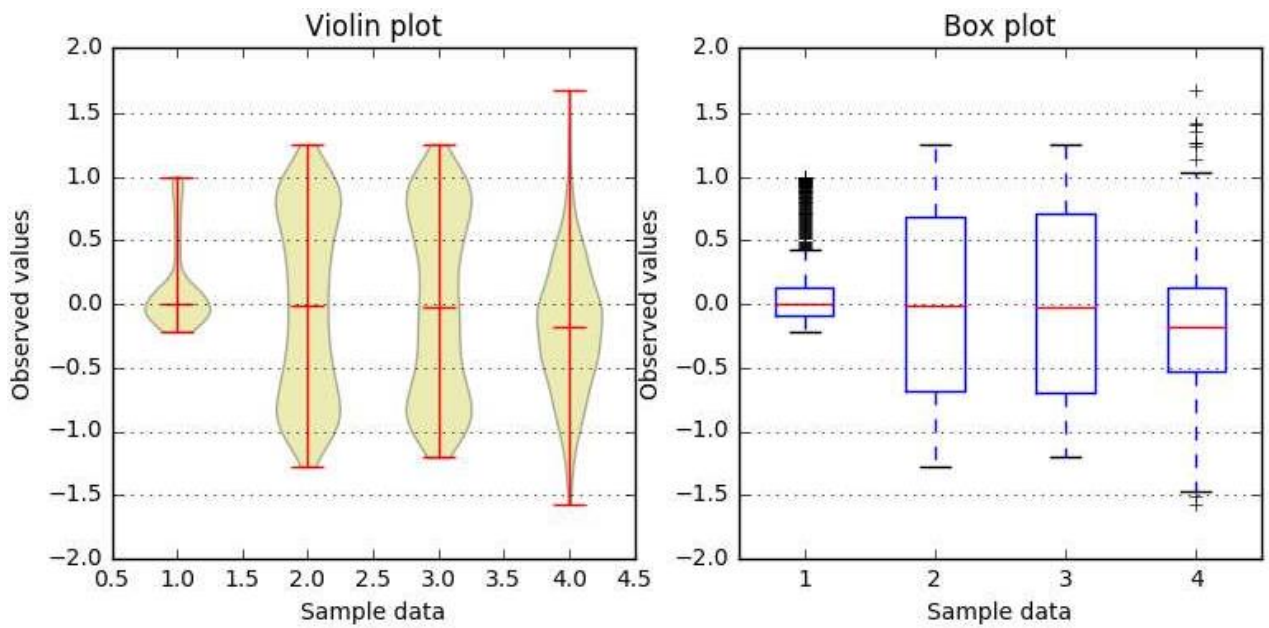
Matplotlib – plot types

```
labels = 'Frogs', 'Hogs', 'Dogs', 'Logs'
fracs = [15, 30, 45, 10]
plt.pie(fracs, labels=labels, shadow=True)
```



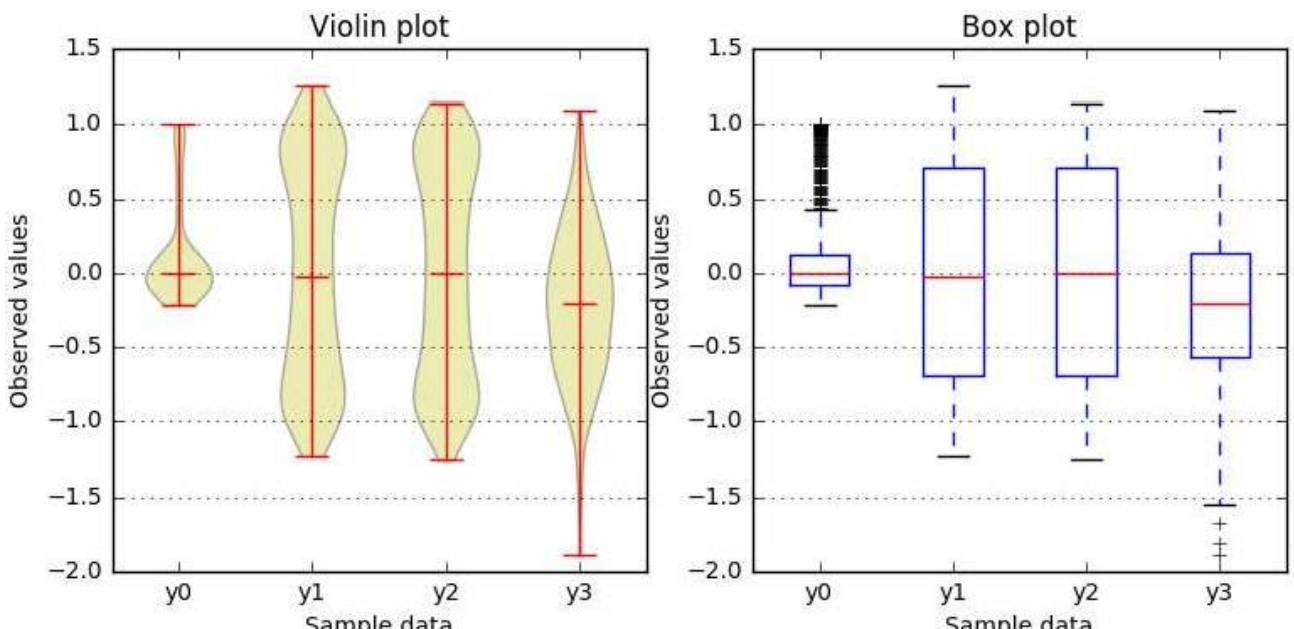
Matplotlib – plot types

```
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(9, 4))
y0 = np.sin(x) / x
y1 = np.sin(x) + np.random.normal(scale=0.1, size=len(x))
y2 = np.cos(x) + np.random.normal(scale=0.1, size=len(x))
y3 =  np.random.normal(loc=-0.2, scale=0.5, size=len(x))
# plot violin plot
axes[0].violinplot([y0,y1,y2,y3],
                   showmeans=False,
                   showmedians=True)
axes[0].set_title('Violin plot')
# plot box plot
axes[1].boxplot([y0,y1,y2,y3])
axes[1].set_title('Box plot')
# adding horizontal grid lines
for ax in axes:
    ax.yaxis.grid(True)
    ax.set_xlabel('Sample data')
    ax.set_ylabel('Observed values')
```



Matplotlib – plot types

```
for ax in axes:
    ax.yaxis.grid(True)
    ax.set_xlabel('Sample data')
    ax.set_ylabel('Observed values')
    ax.set_xticks(range(1,5))#num of features
    ax.set_xticklabels(['y0', 'y1', 'y2', 'y3', 'y4'])
```



Extensions &other packages

seaborn

ggplot

bokeh

plot.ly

https://dsaber.com/2016/10/02/a-dramatic-tour-through-pythons-data-visualization-landscape-including-ggplot-and-altair/

http://pbpython.com/effective-matplotlib.html

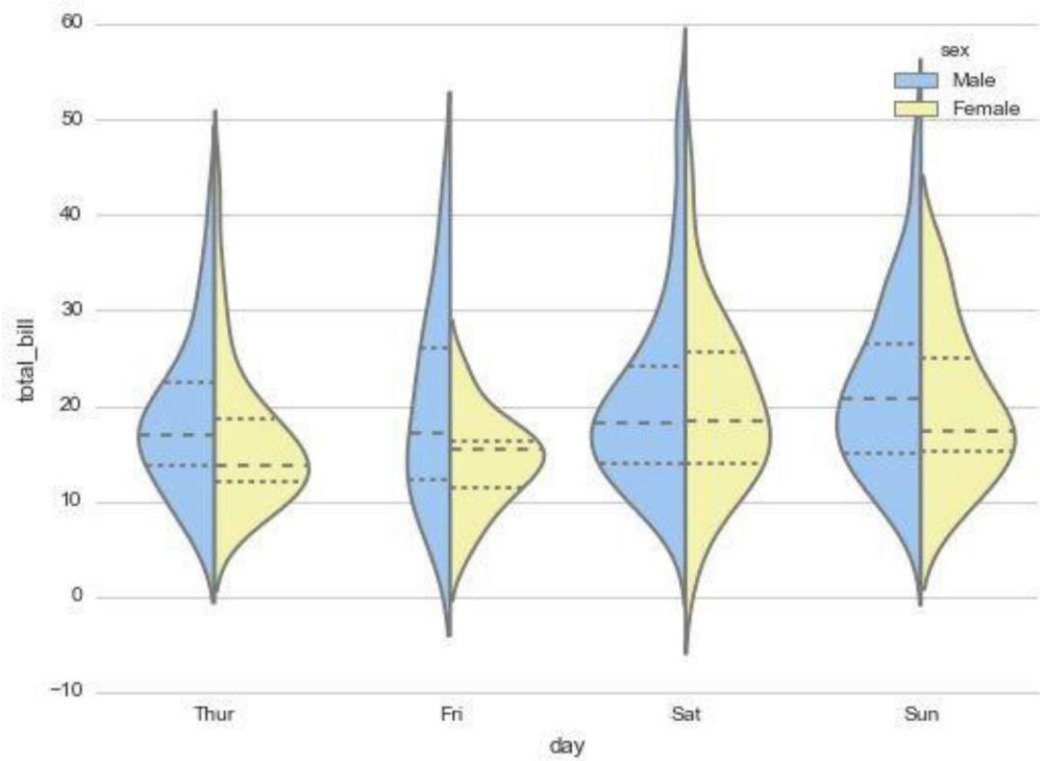
matplotlib drawbacks

- quite old
- too focused on server-side too
- cross-platform
- **not awesomely beautiful by default**

  some ideas may be difficult to write

Seaborn: statistical data visualization

- matplotlib extension
- beautiful theme
- statistical plots
  - distribution
    - hexbin, violin plot, etc.
  - scatterplot
  - pairwise correlations

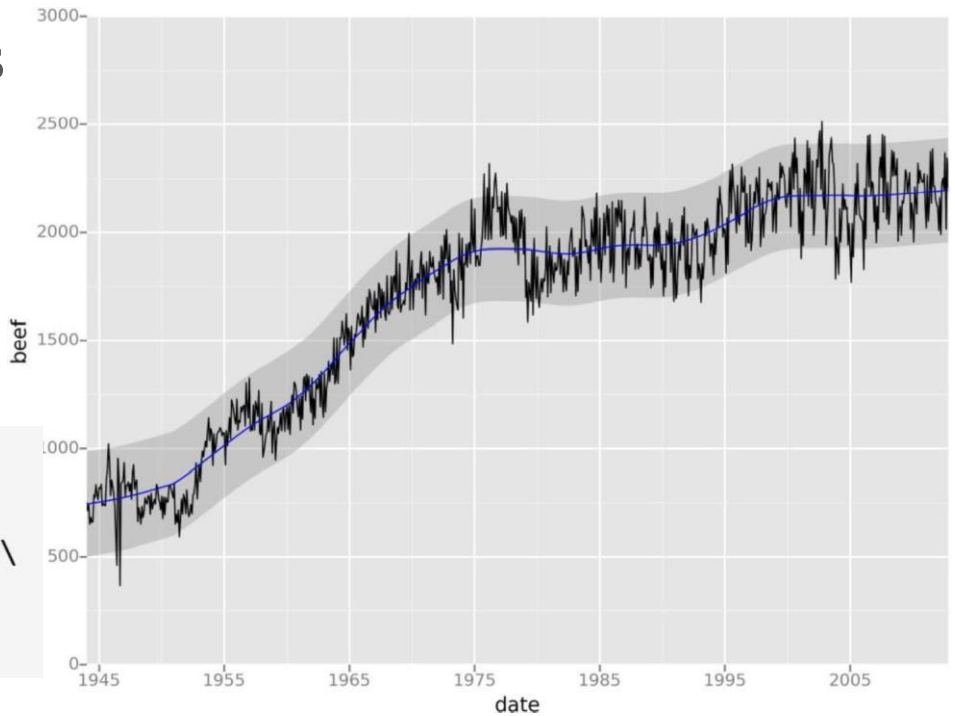


http://stanford.edu/~mwaskom/software/seaborn/examples/grouped_violinplots.html

https://www.slideshare.net/SourabhSahu/python-seaborn-data-visualization
https://www.slideshare.net/nishantupadhyaysbi/python-seaborn-cheatsheet

ggplot from yhat

- based on
  - Grammar of Graphics
  - ggplot2 from R



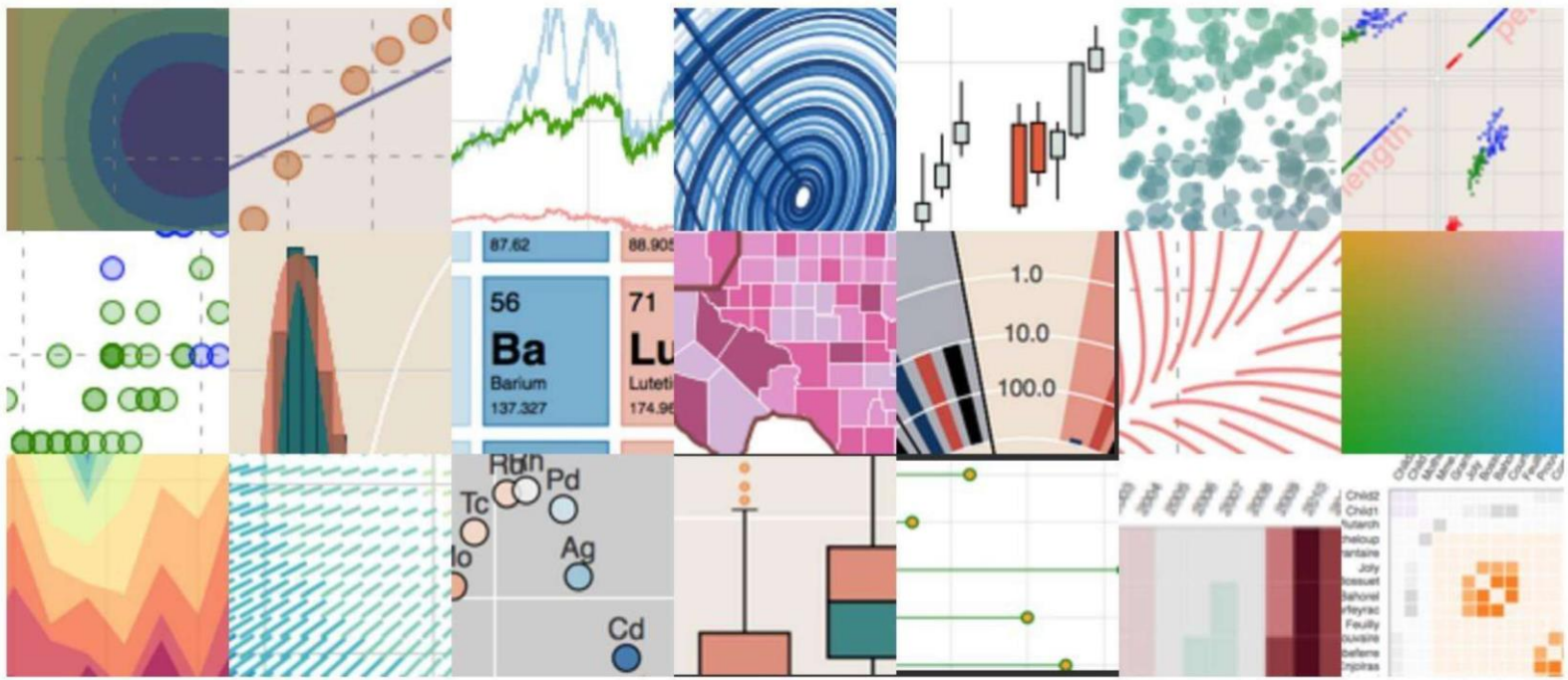

```
from ggplot import *

ggplot(aes(x='date', y='beef'), data=meat) +\
    geom_line() +\
    stat_smooth(colour='blue', span=0.2)
```

http://ggplot.yhathq.com/

Bokeh

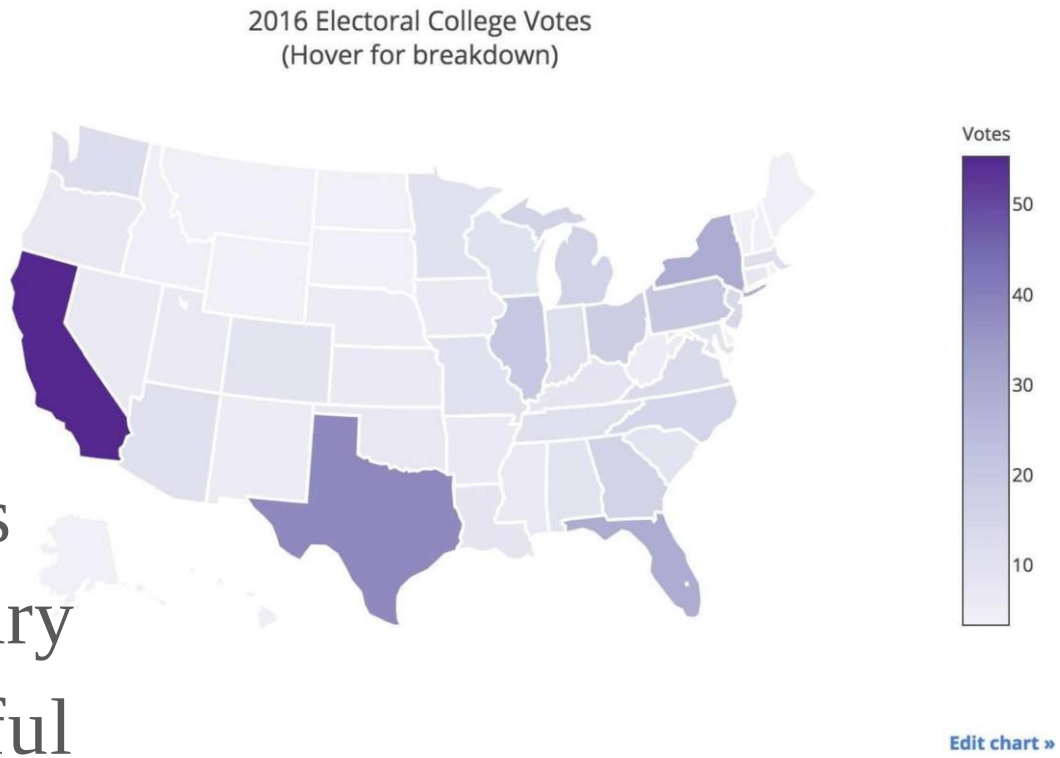
- based on D3.js
- client-side
- interactive
- beautiful
- high-performance

http://bokeh.pydata.org/



Plot.ly

- cloud service
- can publish plots
- free for public plots
- API & Python library
- interactive, beautiful
- good for dashboards



https://plot.ly/python/choropleth-maps/