

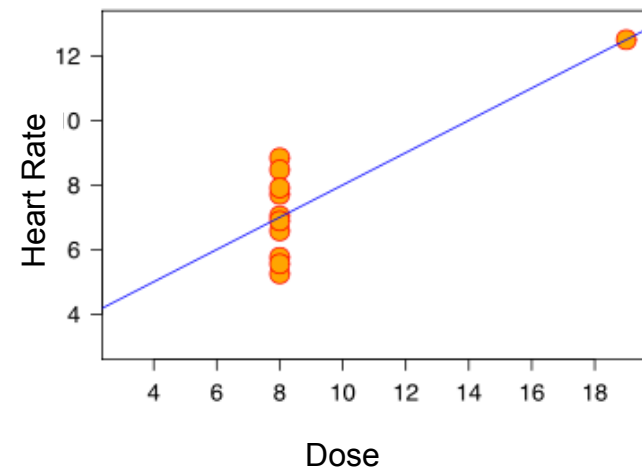
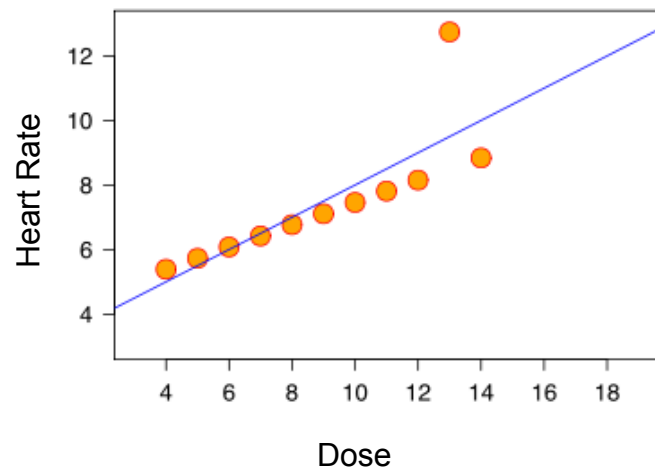
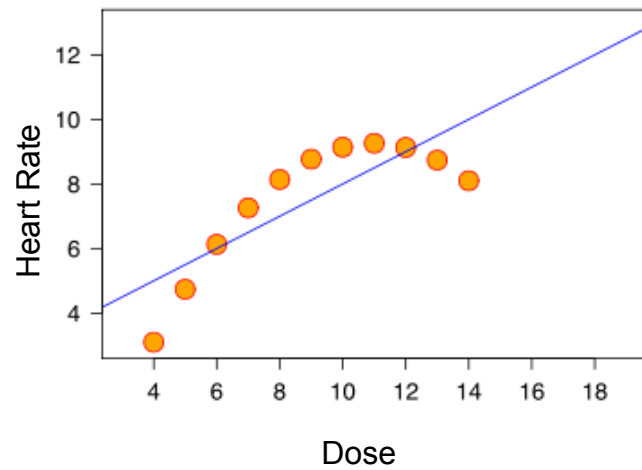
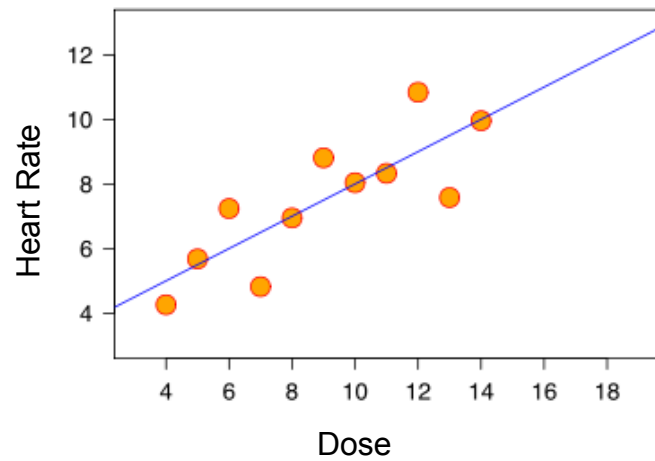
Data Visualization

Lecture Outline

- Why Visualize
- Guidelines
- Color
- Matplotlib
- Pandas

Why Visualize?

Anscombe's Quartet



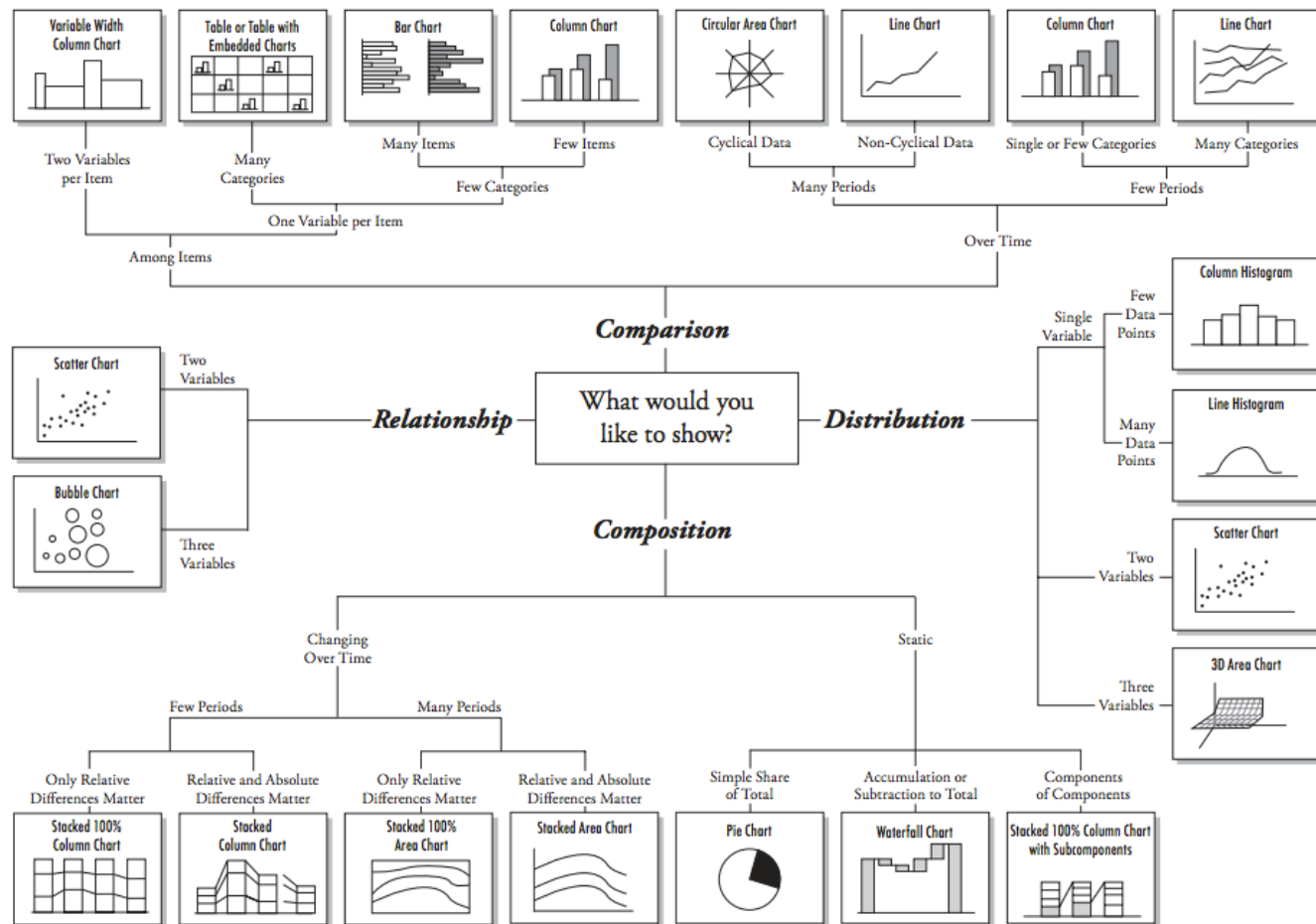
Property	Value
Mean of x in each case	9 (exact)
Variance of x in each case	11 (exact)
Mean of y in each case	7.50 (to 2 decimal places)
Variance of y in each case	4.122 or 4.127 (to 3 decimal places)
Correlation between x and y in each case	0.816 (to 3 decimal places)
Linear regression line in each case	$y = 3.00 + 0.500x$ (to 2 and 3 decimal places, respectively)

Anscombe's quartet

I		II		III		IV	
x	y	x	y	x	y	x	y
10.0	8.04	10.0	9.14	10.0	7.46	8.0	6.58
8.0	6.95	8.0	8.14	8.0	6.77	8.0	5.76
13.0	7.58	13.0	8.74	13.0	12.74	8.0	7.71
9.0	8.81	9.0	8.77	9.0	7.11	8.0	8.84
11.0	8.33	11.0	9.26	11.0	7.81	8.0	8.47
14.0	9.96	14.0	8.10	14.0	8.84	8.0	7.04
6.0	7.24	6.0	6.13	6.0	6.08	8.0	5.25
4.0	4.26	4.0	3.10	4.0	5.39	19.0	12.50
12.0	10.84	12.0	9.13	12.0	8.15	8.0	5.56
7.0	4.82	7.0	7.26	7.0	6.42	8.0	7.91
5.0	5.68	5.0	4.74	5.0	5.73	8.0	6.89

Guidelines

Chart Suggestions—A Thought-Starter



Most
Efficient



Least
Efficient

Position



Length



Slope



Angle



Area



Intensity



Color



Shape



Quantitative

Ordinal

Nominal

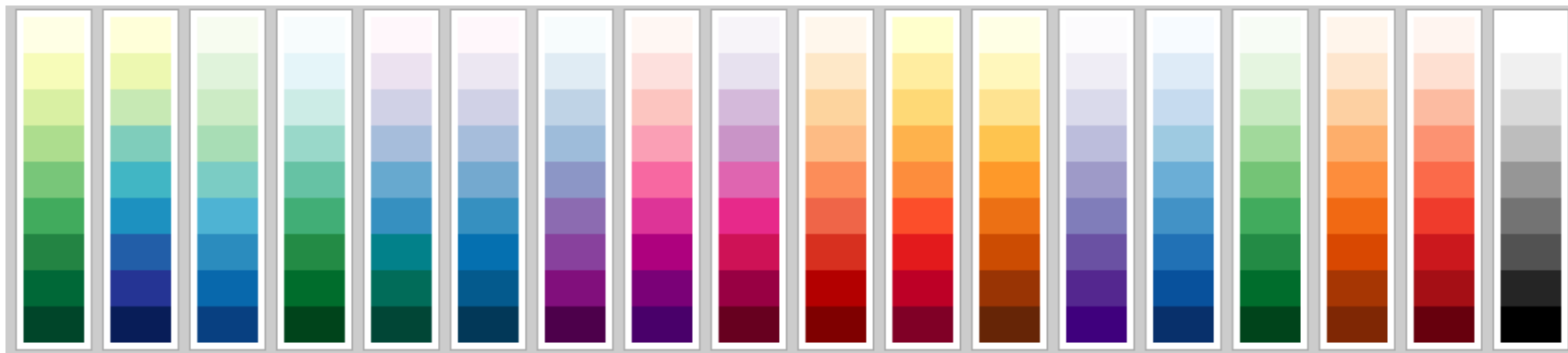
Color

As a data scientist I want:

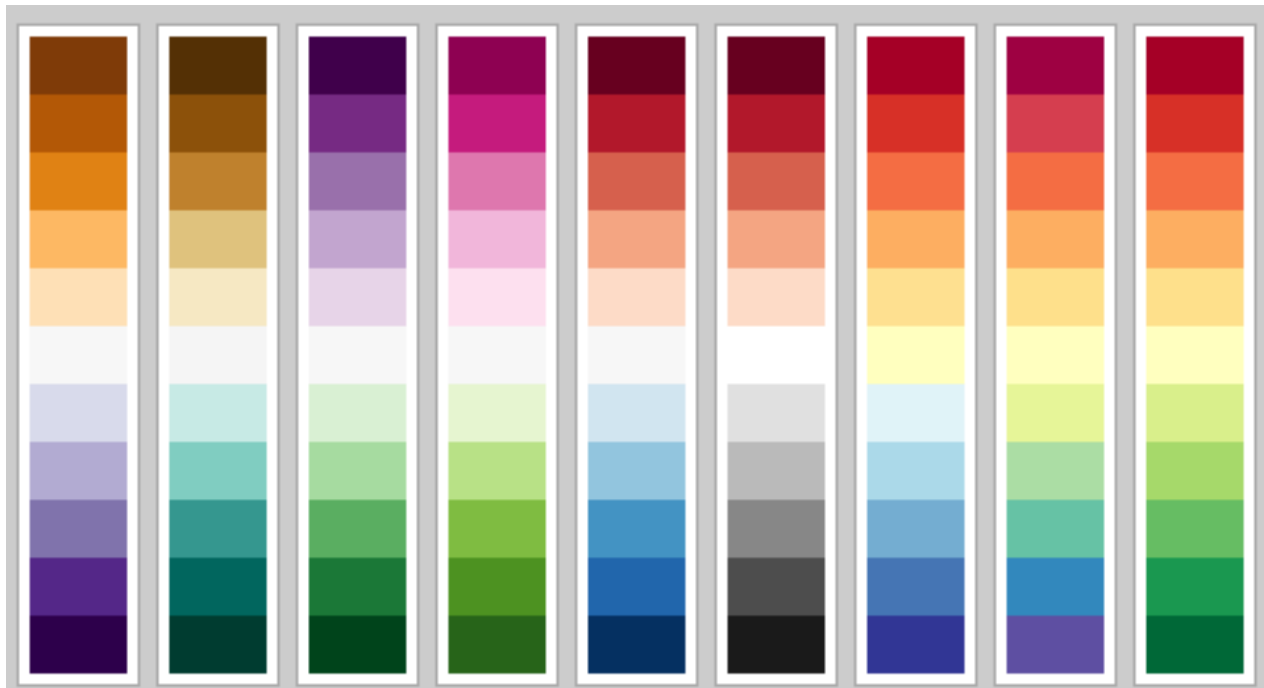
- To **avoid distracting colors**
- To keep **homogeneity**
- To control attention (ability to **mute** certain elements, or to **put focus** somewhere)

...and of course to have **distinct colors**.

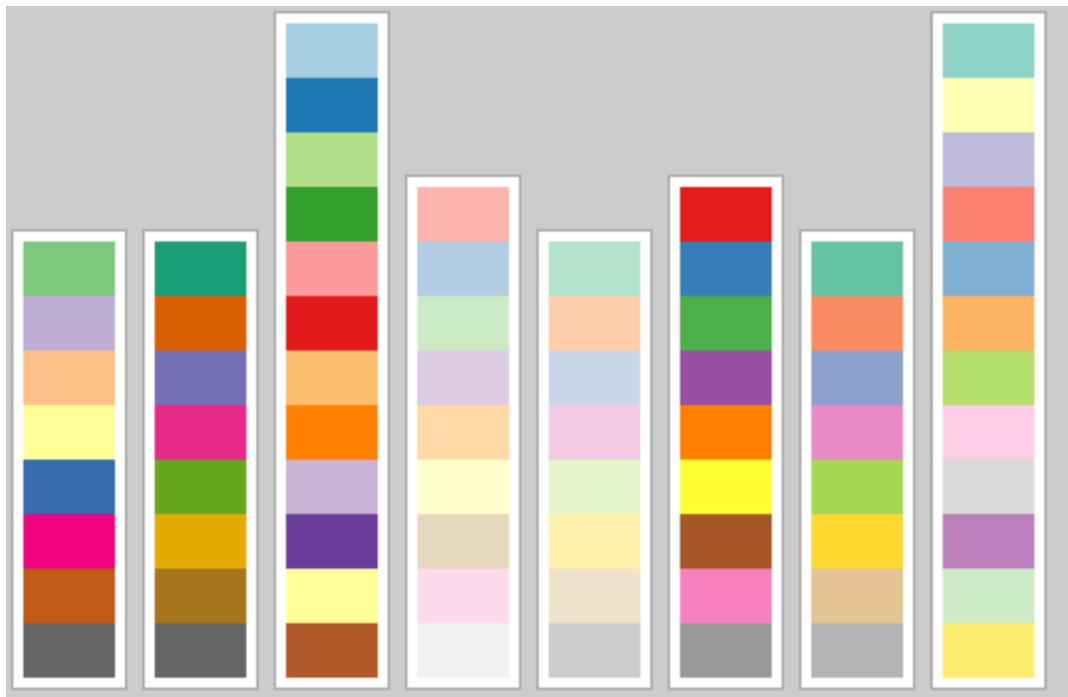
Sequential



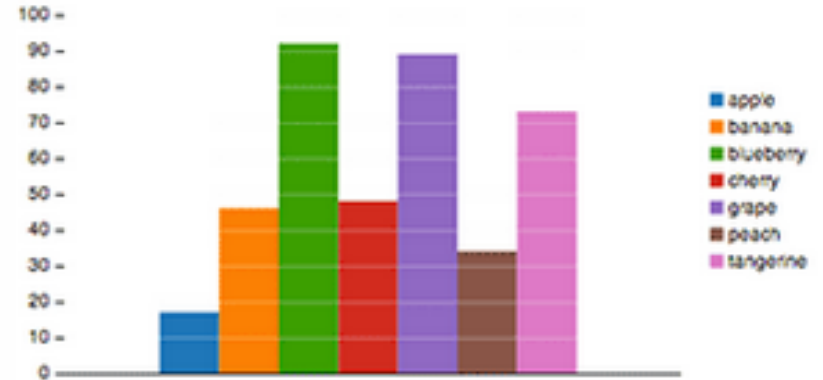
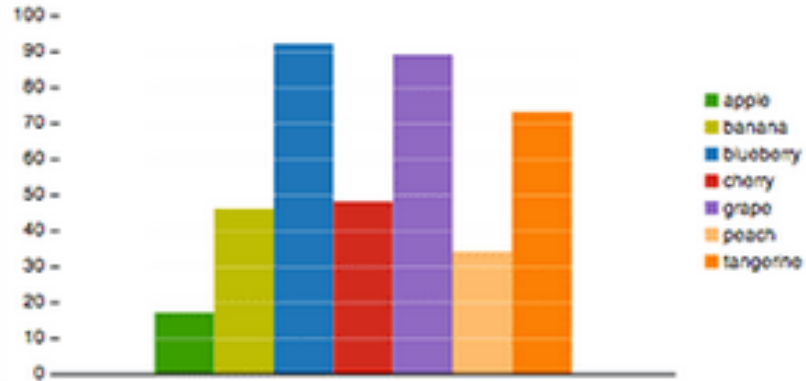
Diverging



Qualitative



Semantic Colors



I Want Hue

[I want hue](#)[Tutorials](#)[Examples](#)[Theory](#)[Experiment](#)[Old version ▾](#)[GitHub](#)[Issues](#)[+ Médialab Tools](#)

Color space

Presets... ▴ ▾

H

168



268

C

0



3

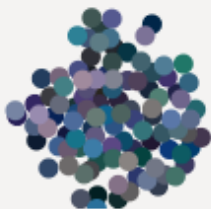
L

0



1.5

☐ Dark background



Palette

3

colors

soft (k-Means) ▴ ▾

↻ Reroll palette



Number of data classes: 3



[how to use](#) | [updates](#) | [downloads](#) | [credits](#)

COLORBREWE
color advice for cartographers

Nature of your data:

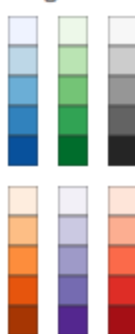
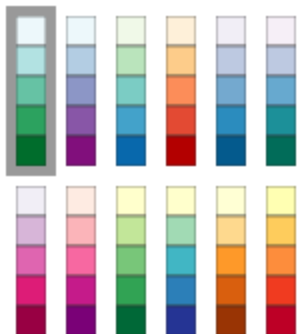


☒ sequential ☐ diverging ☐ qualitative

Pick a color scheme:

Multi-hue:

Single hue:



Only show:



- ☐ colorblind safe
- ☐ print friendly
- ☐ photocopy safe

Context:



- ☐ roads
- ☐ cities
- ☒ borders

Background:

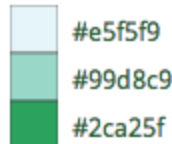
- ☒ solid color



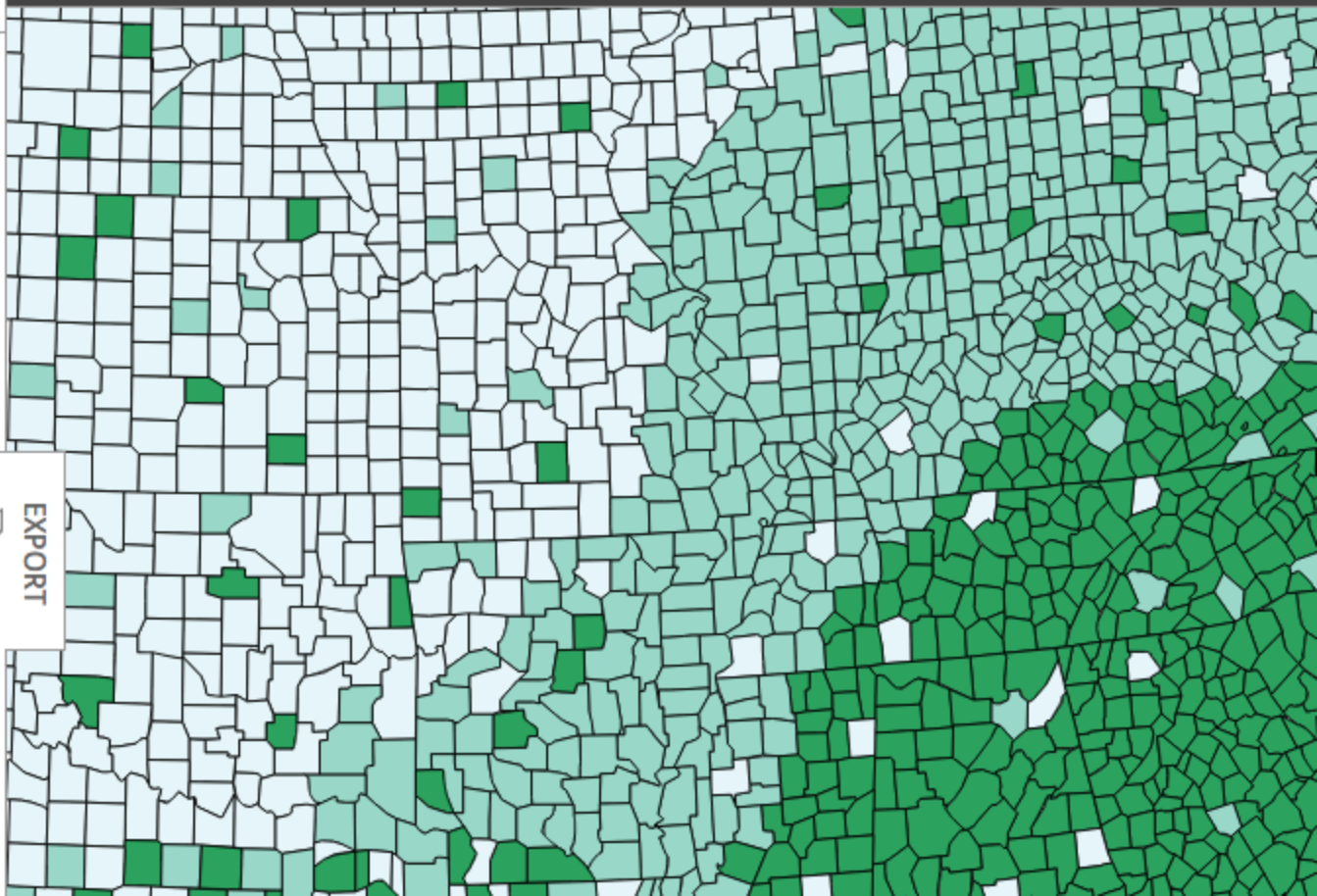
3-class BuGn



HEX



EXPORT



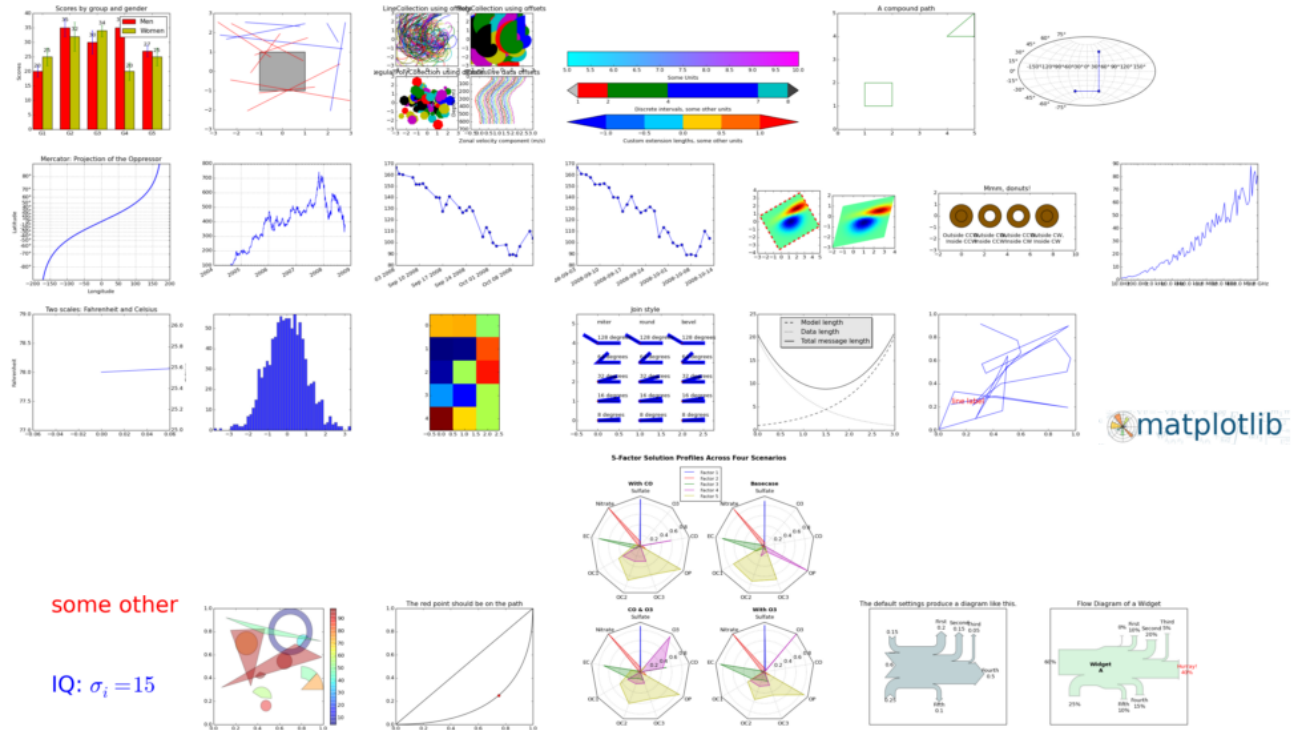
Matplotlib

Matplotlib

matplotlib tries to make easy things easy and hard things possible.

You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc, with just a few lines of code.

Use the Gallery



matplotlib

Search the Examples



[home](#) | [examples](#) | [gallery](#) | [pyplot](#) | [docs](#) »

Matplotlib Examples

Release: 1.3.1

Date: October 10, 2013

- [animation Examples](#)

[animate_decay](#)

[bayes_update](#)

[dynamic_image2](#)

[random_data](#)

[strip_chart_demo](#)

[basic example](#)

[double_pendulum_animated](#)

[histogram](#)

[simple_3danim](#)

[subplots](#)

[basic example writer](#)

[dynamic_image](#)

[moviewriter](#)

[simple_anim](#)

- [api Examples](#)

[agg_oo](#)

[collections_demo](#)

[custom_projection_example](#)

[date_index_formatter](#)

[engineering_formatter](#)

[font_file](#)

[joinstyle](#)

[logo2](#)

[quad_bezier](#)

[sankey_demo_links](#)

[scatter_piecharts](#)

[unicode_minus](#)

[barchart_demo](#)

[colorbar_only](#)

[custom_scale_example](#)

[demo_affine_image](#)

[fahrenheit_celsius_scales](#)

[histogram_path_demo](#)

[legend_demo](#)

[mathtext_asarray](#)

[radar_chart](#)

[sankey_demo_old](#)

[span_regions](#)

[watermark_image](#)

[bbox_intersect](#)

[compound_path](#)

[date_demo](#)

[donut_demo](#)

[font_family_rc](#)

[image_zcoord](#)

[line_with_text](#)

[patch_collection](#)

[sankey_demo_basics](#)

[sankey_demo_rankine](#)

[two_scales](#)

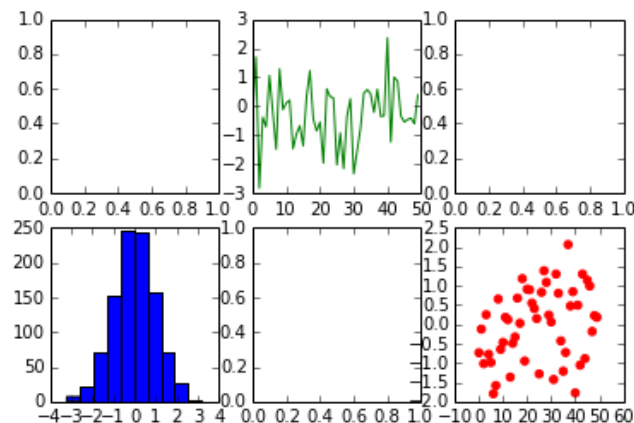
[watermark_text](#)

Figures and Subplots

```
fig, ax = plt.subplots(2,3)

ax[0,1].plot(randn(50), color='green', linestyle='-')
ax[1,2].scatter(np.arange(50), randn(50), color='red')
ax[1,0].hist(randn(1000))
plt.show() #Similar to print()

# Exercise: Swap the top row and bottom row plots
# Exercise: Change the line style to dotted and add circles as markers
# Exercise: change the bins to 100 for the histogram
```

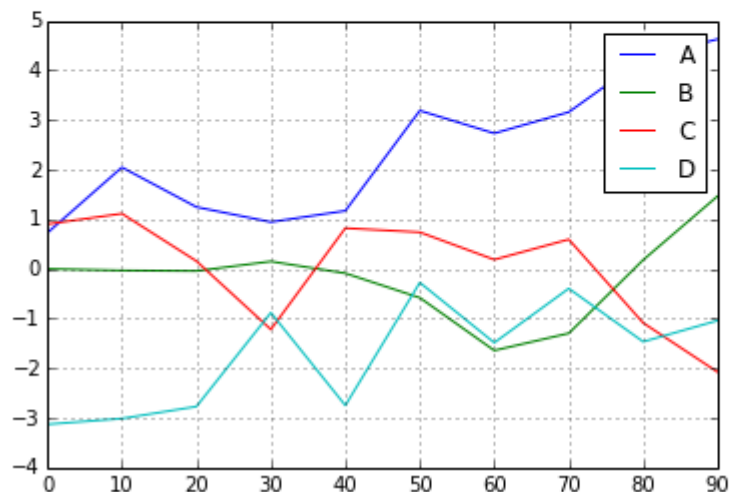


Plotting with Pandas

Line Plots

```
# Pandas data frames have a built in plot function with defaults
df = DataFrame(np.random.randn(10, 4).cumsum(0),
               columns=['A', 'B', 'C', 'D'],
               index=np.arange(0, 100, 10))
df.plot()
```

<matplotlib.axes.AxesSubplot at 0x10681b090>



Bar Plots

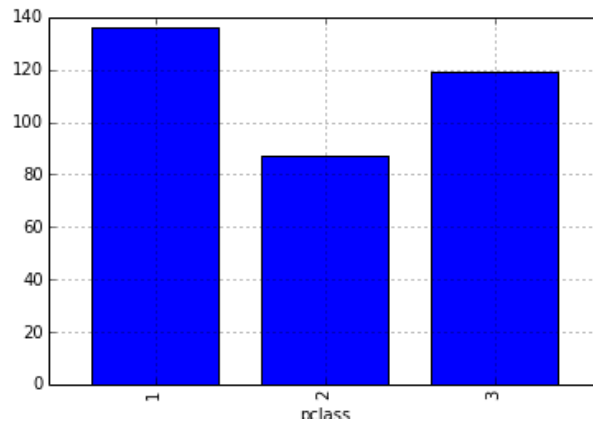
```
Data columns (total 11 columns):  
survived      891  non-null values  
pclass        891  non-null values  
name          891  non-null values  
sex           891  non-null values  
age           714  non-null values  
sibsp         891  non-null values  
parch         891  non-null values  
ticket        891  non-null values  
fare          891  non-null values  
cabin         204  non-null values  
embarked      889  non-null values  
dtypes: float64(2), int64(4), object(5)
```

```
j):
```

Slide

```
# Append .plot to dataframe  
titanic.groupby('pclass').survived.sum().plot(kind='bar')
```

```
j): <matplotlib.axes.AxesSubplot at 0x106cc7f90>
```



Bar Plots

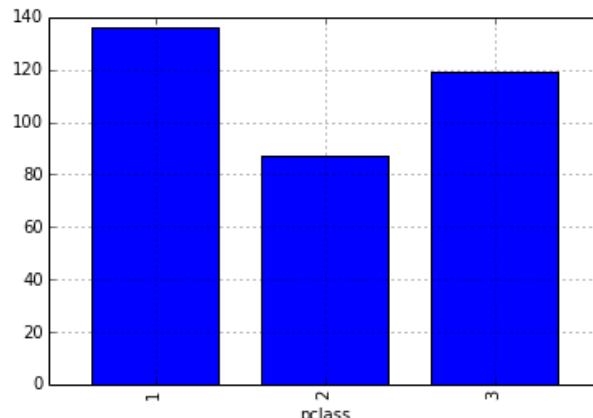
```
Data columns (total 11 columns):  
survived      891  non-null values  
pclass        891  non-null values  
name          891  non-null values  
sex           891  non-null values  
age           714  non-null values  
sibsp         891  non-null values  
parch         891  non-null values  
ticket        891  non-null values  
fare          891  non-null values  
cabin         204  non-null values  
embarked      889  non-null values  
dtypes: float64(2), int64(4), object(5)
```

```
j):
```

Slide

```
# Append .plot to dataframe  
titanic.groupby('pclass').survived.sum().plot(kind='bar')
```

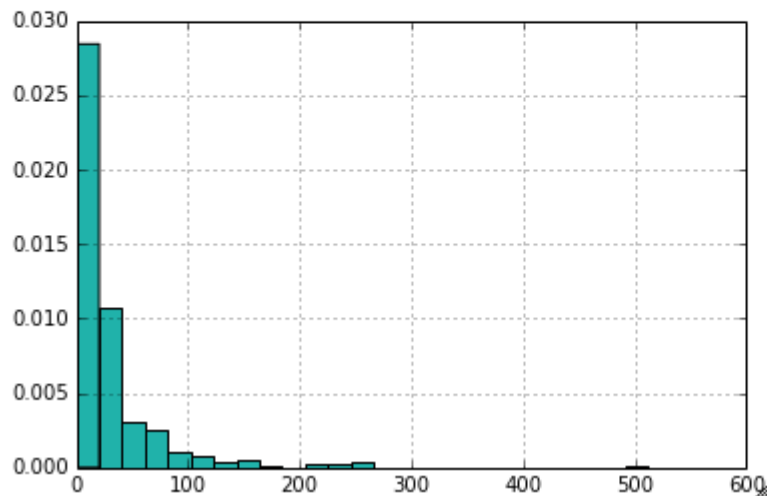
```
j): <matplotlib.axes.AxesSubplot at 0x106cc7f90>
```



Histograms

```
In [22]: titanic.fare.hist(bins=25, normed=True, color='lightseagreen')
```

```
Out[22]: <matplotlib.axes.AxesSubplot at 0x109915310>
```



Read Docs

In [31]:

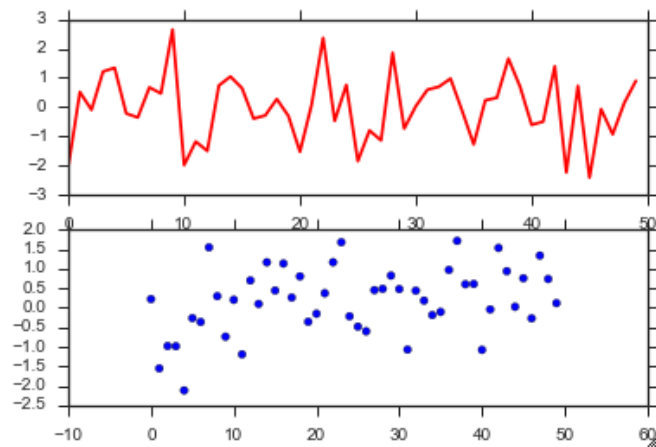
```
fig = figure()

# Two rows, one column, first plot
ax1 = fig.add_subplot(2,1,1)
ax1.plot(randn(50), color='red')

#Two rows, one column, second plot
ax2 = fig.add_subplot(2,1,2)
ax2.scatter(np.arange(50), randn(50))

# Exercise: Try 1 row, two columns
# Exercise: Try 1 row, one column
```

Out[31]: <matplotlib.collections.PathCollection at 0x10bea22d0>



Read Docs

```
help(plt.subplots)
```

Help on function.subplots in module matplotlib.pyplot:

```
subplots(nrows=1, ncols=1, sharex=False, sharey=False, squeeze=True, su  
kw=None, **fig_kw)
```

Create a figure with a set of subplots already made.

This utility wrapper makes it convenient to create common layouts of subplots, including the enclosing figure object, in a single call.

Keyword arguments:

***nrows* : int**

Number of rows of the subplot grid. Defaults to 1.

***ncols* : int**

Number of columns of the subplot grid. Defaults to 1.

***sharex* : string or bool**

If ***True***, the X axis will be shared amongst all subplots. If ***True*** and you have multiple rows, the x tick labels on all but the last row of plots will have visible set to ***False***

If a string must be one of "row", "col", "all", or "none".

"all" has the same effect as ***True***, "none" has the same effect as ***False***.

If "row", each subplot row will share a X axis.

If "col", each subplot column will share a X axis and the x tick labels on all but the last row will have visible set to ***False***

Instructional Set