Data visualization with Matplotlib and Seaborn

Big Data
Kristiania University College

Objectives

1. Motivate the importance of data visualization
2. Avoid some common mistakes in data visualization
3. Choose the proper visualization technique
4. Overview Matplotlib
5. Introduce Seaborn

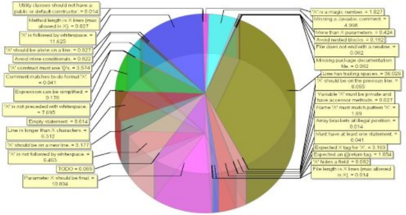
Bibliography

Jake VanderPlas. *Python Data Science Handbook*. Chapters 4. O'Reilly. (Link).

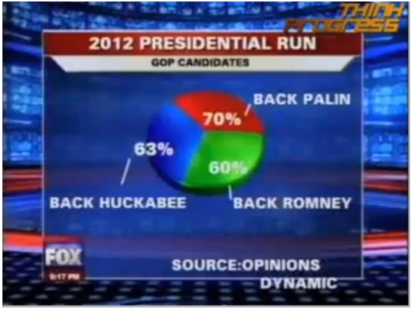
Table of Contents

1. Visualization examples
2. Motivation
3. Matplotlib
    - Seaborn
    - Seaborn datasets
    - Distributions

Visualization examples (I)

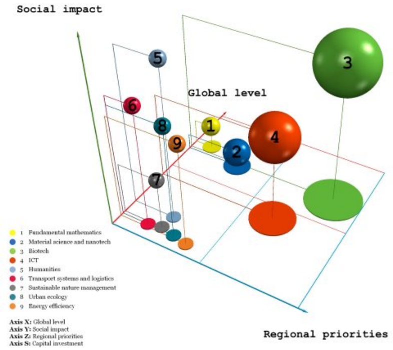


(Source)

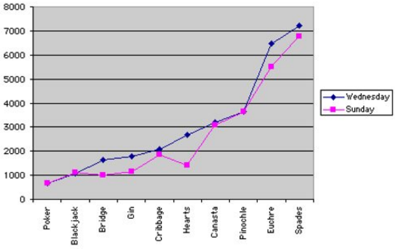


(Source)

Visualization examples (II)

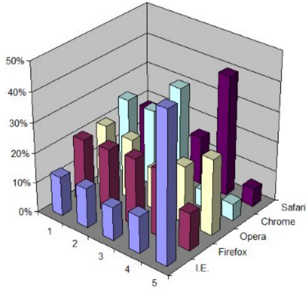


(Source)

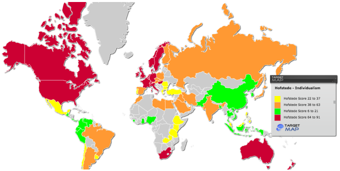


(Source)

Visualization examples (III)

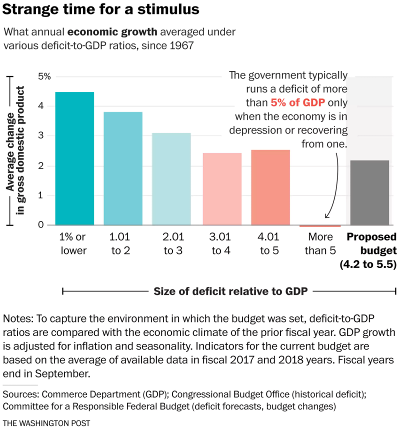


(Source)


TARGET MAP
Hofstede - Individualism
🟨 Hofstede Score 22 to 37
🟧 Hofstede Score 38 to 63
🟩 Hofstede Score 6 to 21
🟥 Hofstede Score 64 to 91
TARGET MAP

(Source)

Visualization examples (IV)



Strange time for a stimulus

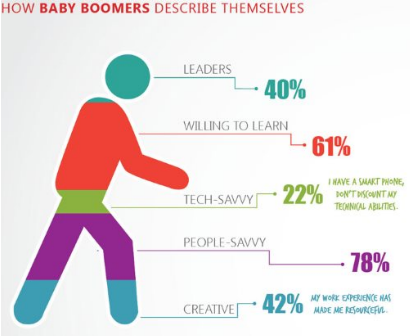
What annual **economic growth** averaged under various deficit-to-GDP ratios, since 1967

Notes: To capture the environment in which the budget was set, deficit-to-GDP ratios are compared with the economic climate of the prior fiscal year. GDP growth is adjusted for inflation and seasonality. Indicators for the current budget are based on the average of available data in fiscal 2017 and 2018 years. Fiscal years end in September.

Sources: Commerce Department (GDP); Congressional Budget Office (historical deficit); Committee for a Responsible Federal Budget (deficit forecasts, budget changes)

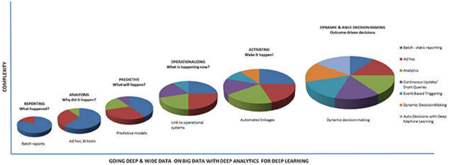
THE WASHINGTON POST

(Source)

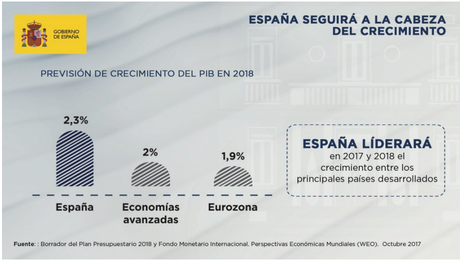


(Source)

Visualization examples (V)



(Source)



(Source)

Motivation (I)

Efficient data visualization tips

- **Define your story**
- The chart must tell the story
- Don't distract from your story (with irrelevant data or visual elements)
- One story, one chart
- Put the story comprension in first term
- Better several simple charts than one complex chart
- Choose colors wisely (color scale or high contrast)
- Elements order must support the story (leyend, bars, etc)
- There is life beyond pies and bars
- Keep it simple, stupid!

Motivation (II)

Know your data

- Categorical or numerical
- Number of dimensions to represent (1D, 2D, 3D, more dimensions)

Can you use other representation?

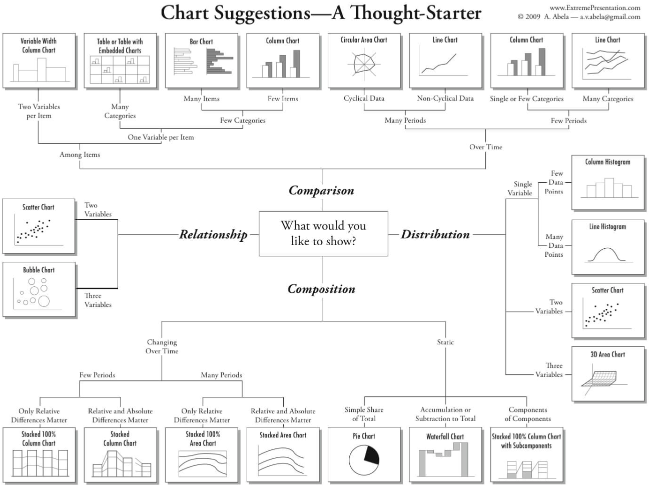
- Chart better than table? ...
- ... that depends

What do you want to represent?

- Distribution, relationship, comparison or composition

Look for templates: (`https://python-graph-gallery.com/`)

Motivation (III)



(Source)
(Alternative resource)

Matplotlib (I)

Matplotlib is a Python package

- Based on NumPy
- Imitates Matlab

Three operation modes

- Scripts.
  Must use `plt.show()` to enter event loop. Use it once!
- IPython shell.
  Must use `%matplotlib`
- IPython notebook. Two modes
  - `%matplotlib inline`
  - `%matplotlib notebook`

Convention

```
import matplotlib as mpl
import matplotlib.pyplot as plt
```

myplot.py

```
import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 10, 100)

plt.plot(x, np.sin(x))
plt.plot(x, np.cos(x))

plt.show()
```

Matplotlib (II)

Matplotlib comes with two interfaces

- Matlab-like. Old-fashioned function-oriented API.
- Object-oriented. Object-oriented and more powerfull API.

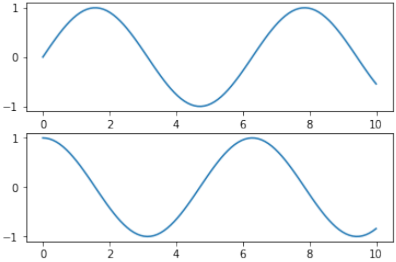
Matlab API

```
plt.figure()  # create a plot

# create the first of two panels and set current axis
plt.subplot(2, 1, 1) # (rows, columns, panel number)
plt.plot(x, np.sin(x))

# create the second panel and set current axis
plt.subplot(2, 1, 2)
plt.plot(x, np.cos(x));
```

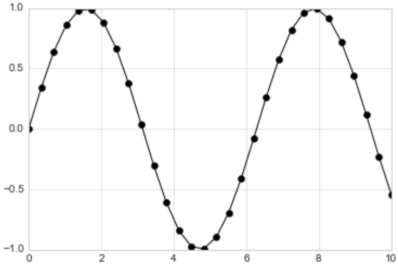
OO API

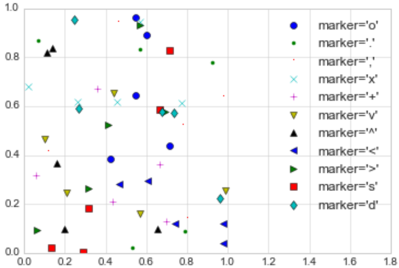
```
plt.figure()  # create a plot

# create the first of two panels and set current axis
plt.subplot(2, 1, 1) # (rows, columns, panel number)
plt.plot(x, np.sin(x))

# create the second panel and set current axis
plt.subplot(2, 1, 2)
plt.plot(x, np.cos(x));
```

Matplotlib (III)



Matplotlib (IV)

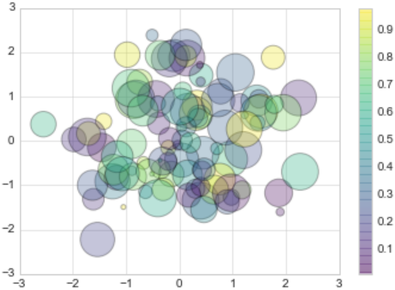


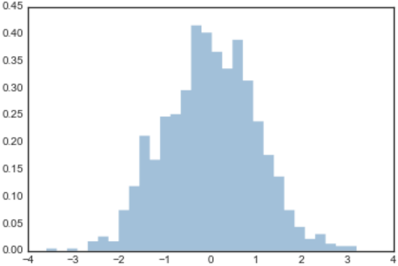


```
^^Iplt.plot(x, np.sin(x), '—ok',
     color='black')
^^I
```

```
for marker in ['o', '.', ',', 'x', '+', 'v', '^', '
    <', '>', 's', 'd']:
^^Iplt.plot(rng.rand(5), rng.rand(5), marker,
^^I^^I^^Ilabel="marker='{0}'".format(marker))
^^Iplt.legend(numpoints=1)
^^Iplt.xlim(0, 1.8);
```

Matplotlib (V)



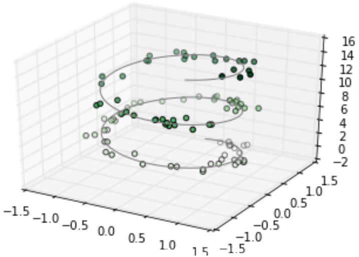


```
rng = np.random.RandomState(0)
x = rng.randn(100)
y = rng.randn(100)
colors = rng.rand(100)
sizes = 1000 * rng.rand(100)

plt.scatter(x, y, c=colors, s=sizes, alpha=0.3,
    cmap='viridis')
    plt.colorbar();  # show color scale
```

```
data = np.random.randn(1000)

plt.hist(data, bins=30, normed=True, alpha=0.5,
    histtype='stepfilled', color='steelblue',
    edgecolor='none');
```

Matplotlib (VI)

```
^^Iax = plt.axes(projection='3d')

^^I# Data for a three—dimensional line
^^Izline = np.linspace(0, 15, 1000)
^^Ixline = np.sin(zline)
^^Iyline = np.cos(zline)
^^Iax.plot3D(xline, yline, zline, 'gray')

^^I# Data for three—dimensional scattered points
^^Izdata = 15 * np.random.random(100)
^^Ixdata = np.sin(zdata) + 0.1 * np.random.randn
      (100)
^^Iydata = np.cos(zdata) + 0.1 * np.random.randn
      (100)
^^Iax.scatter3D(xdata, ydata, zdata, c=zdata, cmap=
      'Greens');
^^I
```



Seaborn (I)

Seaborn is a modern data-visualization Python package

- Based on matplotlib
- ... it uses matplotlib indeed
- Pandas-aware
- High level
- Advanced visualizations
- Easy to use

Still under development! (v. 0.9)

Convention

```
^^Iimport seaborn as sns
^^I
```

This documentation is for Seaborn 0.9 or newer

Seaborn (II)

Display initialization

- `plt.show()`
- `%matplotlib`

Style initialization

- Default Seaborn style `sns.set()`
- By default, same style than matplotlib

Several functions ...

- ... similar parameters

Parameters

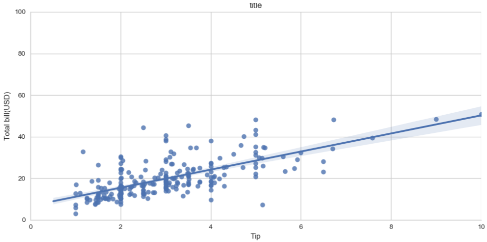
- x: Data axis x
- y: Data axis Y
- data: Dataframe name
- hue: Color
- style: Style
- sizes: Size
- kind: Alternate representation

Seaborn (III)

Typical Seaborn usage

1. Prepare data
2. Set up aesthetics
3. Plot
4. Customize the plot

```
^^I^^I^^Iimport matplotlib.pyplot as plt
^^I^^I^^Iimport seaborn as sns
^^I^^I^^I# Prepare data
^^I^^I^^Itips = sns.load_dataset("tips")
^^I^^I^^I# Set up aesthetics
^^I^^I^^Isns.set_style("whitegrid")
^^I^^I^^I# Plot
^^I^^I^^Ig = sns.lmplot(x="tip",y="total_bill", data=tips,
        aspect=2)
^^I^^I^^I# Plot customization
^^I^^I^^Ig = (g.set_axis_labels("Tip","Total bill(USD)").set(
        xlim=(0,10),ylim=(0,100)))
^^I^^I^^Iplt.title("title")
^^I^^I^^Iplt.show(g)
^^I^^I^^I
```



Seaborn

Datasets (I)

Seaborn comes with several dummy datasets

- `sns.load_dataset('name')`

We will use two datasets

- ’iris’: The classical iris dataset, numerical
- ’tips’: Numeric and categorical variables

Tips dataset

```
»> tips = sns.load_dataset(’tips’)
»> print(tips.head())
```

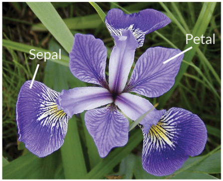
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

Seaborn

Datasets (II)

Iris dataset

```
»> iris = sns.load_dataset('iris')
»> print(iris.head())
   sepal_length  sepal_width  petal_length  petal_width  species
0           5.1          3.5           1.4          0.2  setosa
1           4.9          3.0           1.4          0.2  setosa
2           4.7          3.2           1.3          0.2  setosa
3           4.6          3.1           1.5          0.2  setosa
4           5.0          3.6           1.4          0.2  setosa
```



Iris Versicolor

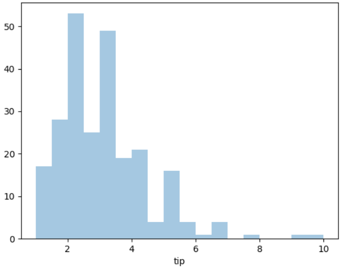
Iris Setosa

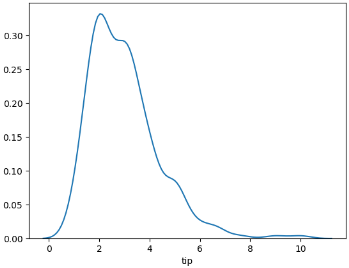
Iris Virginica

(Source)

Seaborn

Distributions (I)





Histogram

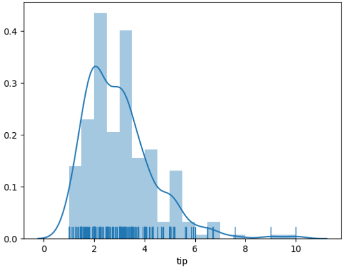
```
^^Isns.distplot(tips['tip'],
        kde=False)
^^I
```

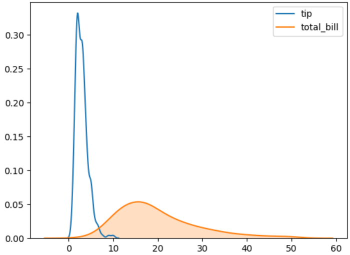
Density plot

```
^^Isns.distplot(tips['tip'],
        hist=False)
^^I
```

Seaborn

Distributions (II)





Histogram + density plot

```
^^Isns.distplot(tips['tip'],
    rug=True)
^^I
```

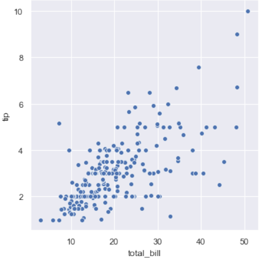
Density plot

```
^^Isns.kdeplot(tips['tip'])
^^Isns.kdeplot(tips['
    total_bill'], shade=True
    )
^^I
```

Seaborn

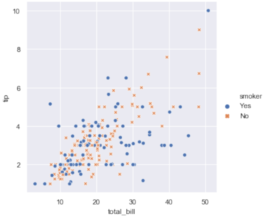
Relationships (I)

Scatterplots

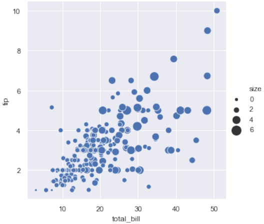


```
^^Isns.relplot(x="total_bill", y=
        "tip", data=tips)
^^I
```

Seaborn >= 0.9



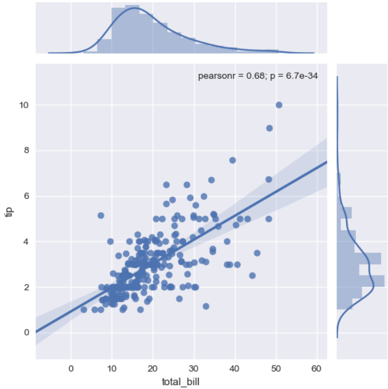
```
^^Isns.relplot(x="total_bill", y=
        "tip", hue="smoker", style=
        "smoker", data=tips)
^^I
```

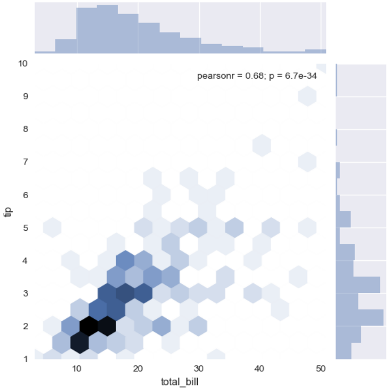


```
^^Isns.relplot(x="total_bill", y=
        "tip", size="size", sizes
        =(15, 200), data=tips);
^^I
```

Seaborn

Relationships (II)



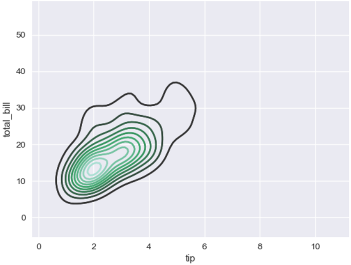


```
^^Isns.jointplot("total_bill", "tip", tips, kind="
        reg")
^^I
```

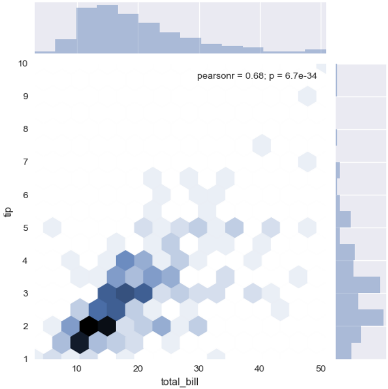
```
^^Isns.jointplot("total_bill", "tip", tips , kind="
        hex")
^^I
```

Seaborn

Relationships (III)



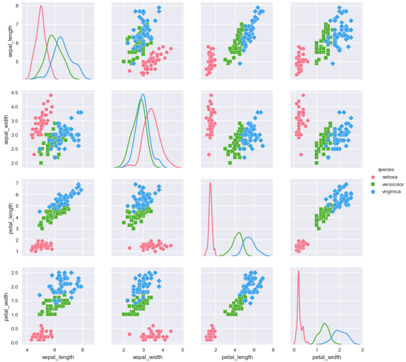
```
^^Isns.kdeplot(tips['tip'], tips['total_bill'])
^^I
```



```
^^Isns.jointplot("total_bill", "tip", tips , kind="
        hex")
^^I
```

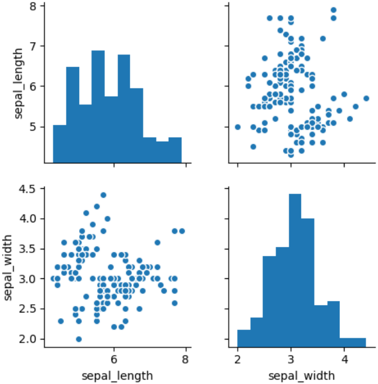
Seaborn

Relationships (IV)



Scatterplot matrix

```
^^Isns.pairplot(iris, hue="species", palette="husl"
    , markers=["o", "s", "D"], diag_kind='kde')
^^I
```

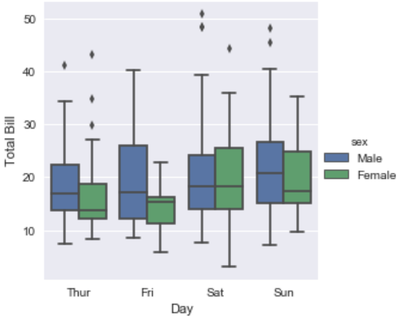


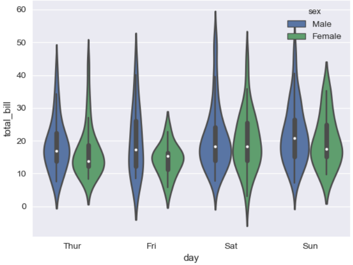
Scatterplot matrix

```
^^Isns.pairplot(iris, vars=["sepal_length", "
    sepal_width"])
^^I
```

Seaborn

Comparisons (I)





Boxplot

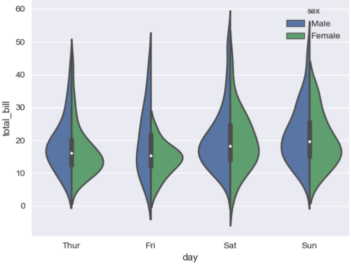
```
^^Iwith sns.axes_style(style='ticks'):
^^I    g = sns.factorplot("day", "total_bill", "sex
    ", data=tips, kind="box")
^^I^^Ig.set_axis_labels("Day", "Total Bill")
^^I
```

Violin plot

```
^^Isns.violinplot("day", "total_bill", "sex", data=
    tips)
^^I
```

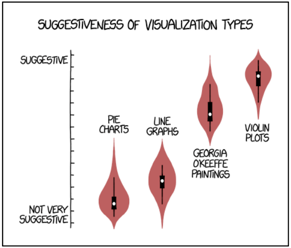
Seaborn

Comparisons (II)



Violin plot

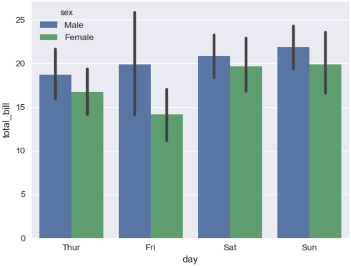
```
^^Isns.violinplot(x="day", y="total_bill", hue="sex
    ", data=tips, split=True)
^^I
```

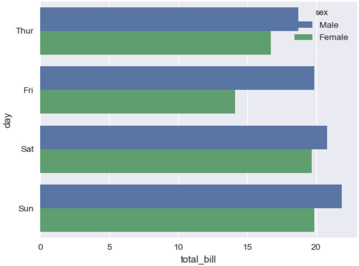

SUGGESTIVENESS OF VISUALIZATION TYPES
SUGGESTIVE
NOT VERY SUGGESTIVE
PIE CHARTS
LINE GRAPHS
GEORGIA O'KEEFFE PAINTINGS
VIOLIN PLOTS

(Source)

Seaborn

Barplots





Barplot

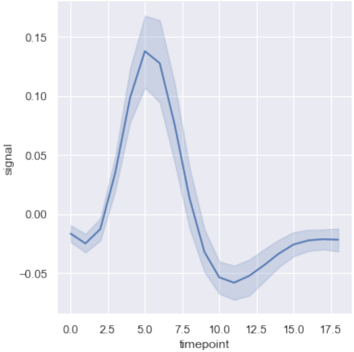
```
^^Isns.barplot(x=”day”, y=”total_bill”, hue=”sex”,
        data=tips)
^^I
```

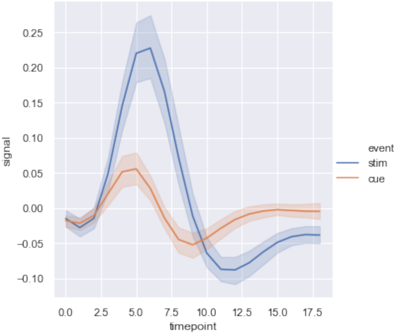
Barplot

```
sns.barplot(x=”total_bill”, y=”day”, hue=”sex”,
        data=tips, ci=None)
^^I
```

Seaborn

Continuity





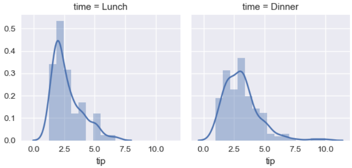
```
^^I^^Ifmri = sns.load_dataset(”fmri”)
^^I^^Isns.relplot(x=”timepoint”, y=”signal”, kind=”
        line”, data=fmri)
^^I^^I
```

```
^^I^^Isns.relplot(x=”timepoint”, y=”signal”, hue=”
        event”, kind=”line”, data=fmri)
^^I^^I
```

Seaborn >= 0.9

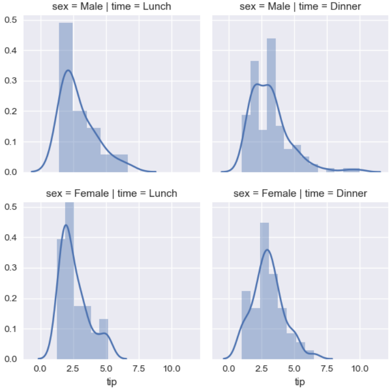
Seaborn

FacetGrid

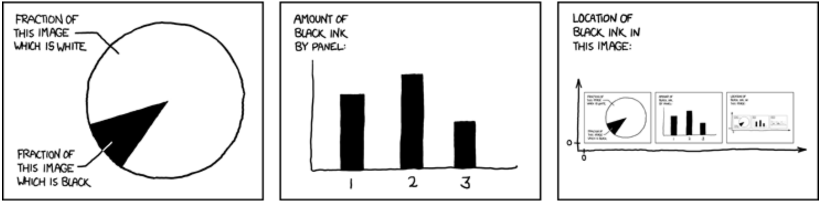


```
^^I^^Ifmri = sns.load_dataset("fmri")
^^I^^Isns.relplot(x="timepoint", y="signal", kind="
      line", data=fmri)
^^I^^I
```

Seaborn >= 0.9



```
^^I^^Ig = sns.FacetGrid(tips, col="time", row="sex"
    )
^^I^^Ig.map(sns.distplot, "tip")
^^I^^I
```


FRACTION OF THIS IMAGE WHICH IS WHITE
FRACTION OF THIS IMAGE WHICH IS BLACK
AMOUNT OF BLACK INK BY PANEL:
1
2
3
LOCATION OF BLACK INK IN THIS IMAGE:
0
0

(Source)