

# Python plotting

A modern approach with Pandas and Seaborn

*Andreas Bjerre-Nielsen*

# Recap

*What have we learned about basic Python?*

-

-

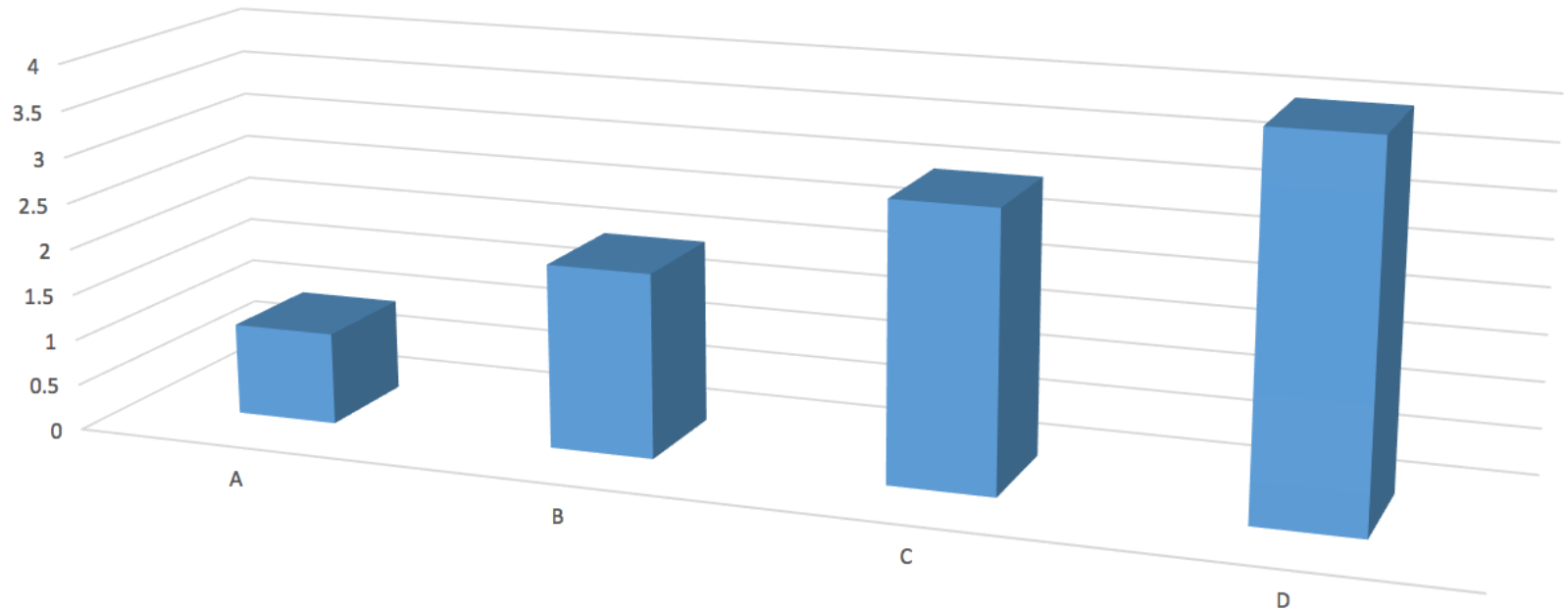
# Agenda

1. Basic exploratory plots with Pandas and Seaborn.
  - plots for single variables (histograms etc.)
  - plots for relationship between two or more variables (box, scatter, etc.)
2. Making explanatory plots useful and beautiful

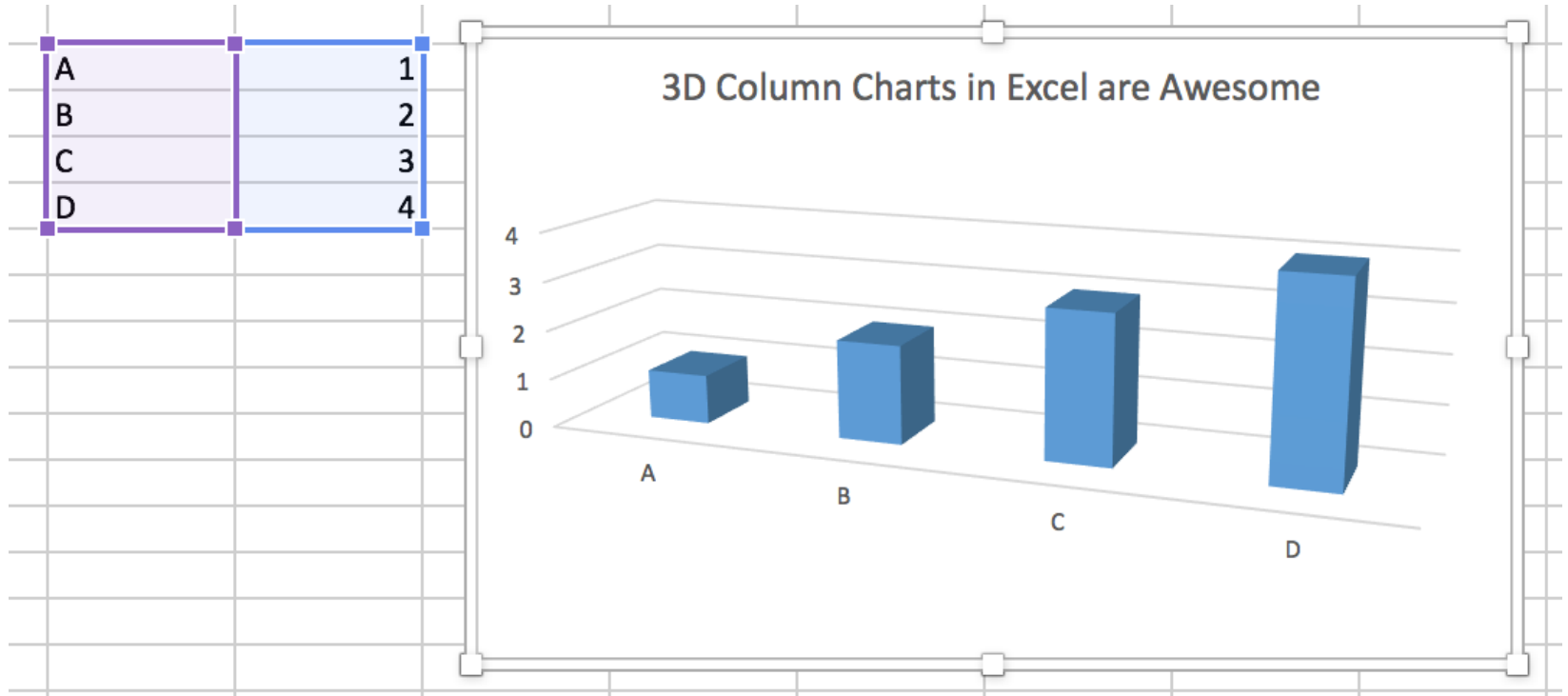
# Understanding plotting

# What values do A,B,C,D have?

3D Column Charts in Excel are Awesome



## The shocking answer



# What are you trying to accomplish?

1. Who's the audience?

- Exploratory (use defaults) vs. explanatory (customize)
- Raw data vs. model results
- Data type: numerical vs. non-numeric (categorical)

2. Graphs should be self explanatory

3. A graph is a narrative - should convey key point(s)

**Analysis preparation**



# Getting prepared (1)

*How do we start our analysis?*

We first load the relevant modules

```
In [2]: import matplotlib.pyplot as plt # fundamental plotting
import numpy as np # matrix framework like matlab
import pandas as pd
import seaborn as sns # high level plotting library

# allow printing in notebook
%matplotlib inline
```

## Getting prepared (2)

*How do we load some data?*

We load a standard dataset: tips.

```
In [3]: tips = sns.load_dataset('tips')
```

## Getting prepared (3)

*How do we see what is in the DataFrame?*

We get preview as follows:

```
In [5]: tips.head()
```

Out[5]:

	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

Quiz: which variables/columns are available in the tips DataFrame?

# DataFrame structures

# Table format

*How do we define a tidy/long table?*

One row for each observation:

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20095360
Brazil	1999	37737	17206362
Brazil	2000	80488	174504898
China	1999	212258	1272015272
China	2000	216766	128042583

variables

country	year	cases	population
Afghanistan	1999	745	19987071
Afghanistan	2000	2666	20095360
Brazil	1999	37737	17206362
Brazil	2000	80488	174504898
China	1999	212258	1272015272
China	2000	216766	128042583

observations

country	year	cases	population
Afghanistan	99	75	19987071
Afghanistan	00	2666	20095360
Brazil	99	37737	17206362
Brazil	00	80488	174504898
China	99	212258	1272015272
China	00	216766	128042583

values

Quiz: *Is our DataFrame, tips, in wide format? Why is tidy smart?*

## Plotting format

When plotting data there are two canonical formats: numeric and categorical.

- Have different plotting techniques.
- Note: numeric data can be binned and be regarded as categorical.

**Case: Plotting one numerical variable**

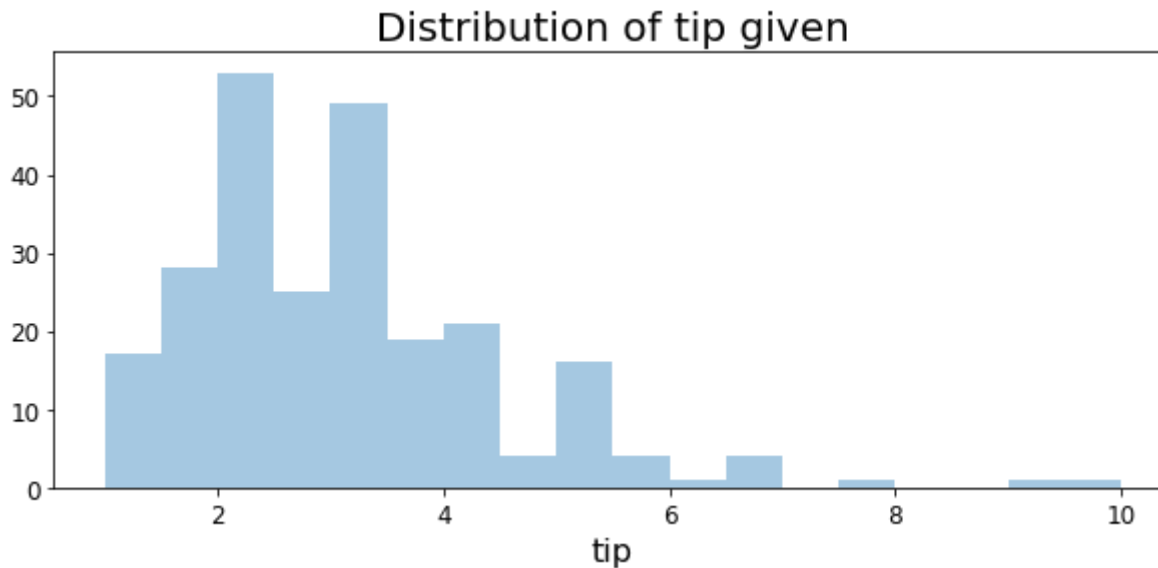
# From exploratory to final output

*How do we plot the distribution of numerical variables?*

We often use the histogram. Let's see what it is:

In [4]: `histplot`

Out[4]:





# Choosing your tool

In this course you will be exposed to several ways of plotting. All tools have their advantages.

Our options:

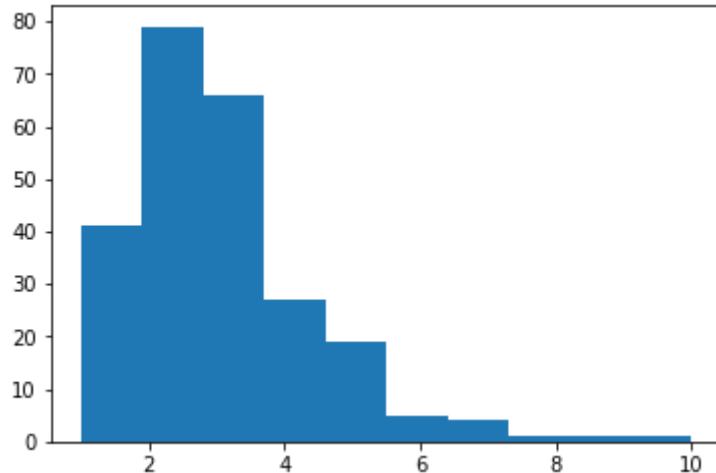
- the fundamental and flexible ~ matplotlib
- quick and dirty for wide format ~ pandas
- a smart choice for long (i.e. tidy) format ~ seaborn

# Histogram with matplotlib

We will begin with the fundamental and flexible way. An old-school way of doing things.

```
In [18]: f,ax = plt.subplots() # create placeholder for plot  
ax.hist(tips.tip) # make plot
```

```
Out[18]: (array([ 41.,  79.,  66.,  27.,  19.,   5.,   4.,   1.,   1.,   1.]),  
          array([ 1. ,  1.9,  2.8,  3.7,  4.6,  5.5,  6.4,  7.3,  8.2,  
                 9.1, 10. ]),  
          <a list of 10 Patch objects>)
```



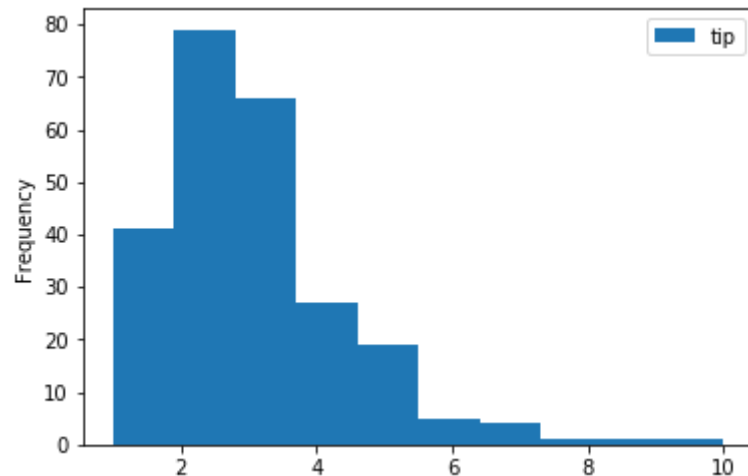
What might we change about this?

# Histogram - pandas

Pandas has a quick and dirty implementation. Let's try the code below.

```
In [8]: tips.plot(y=['tip'], kind= 'hist')
```

```
Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x1aa51a85710>
```

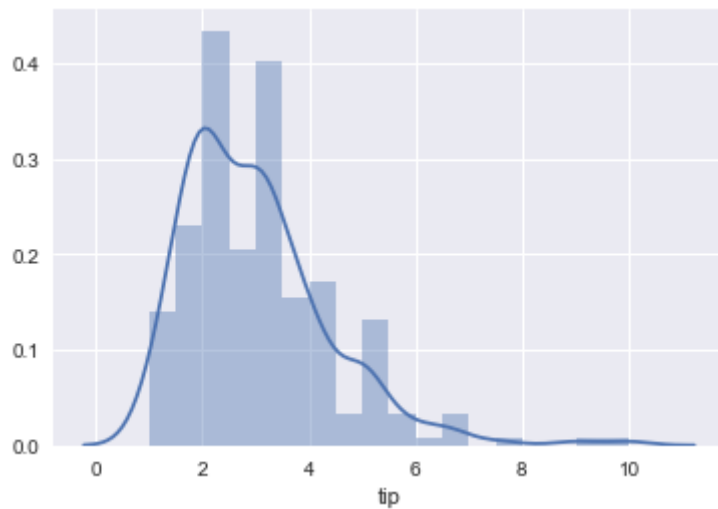


# Histogram - seaborn

```
In [9]: sns.set() # seaborn default
```

```
In [10]: sns.distplot(tips.tip) # histogram for seaborn
```

```
Out[10]: <matplotlib.axes._subplots.AxesSubplot at 0x1aa51b58ef0>
```



What is the line?

# Summing up

Group discussion (2 minutes):

- How did our tools perform?
  - Seaborn best immediate plot.
- Which one seems most adequate for exploratory analysis? Which one for explanatory?
  - Seaborn seems best for exploratory.
  - Matplotlib but requires much work with customizations.
- Which steps could be taken towards improving the Seaborn histogram?
  - Size, add title, bins of histogram, font of labels/title/axis ticks

## Explanatory plotting: the histogram

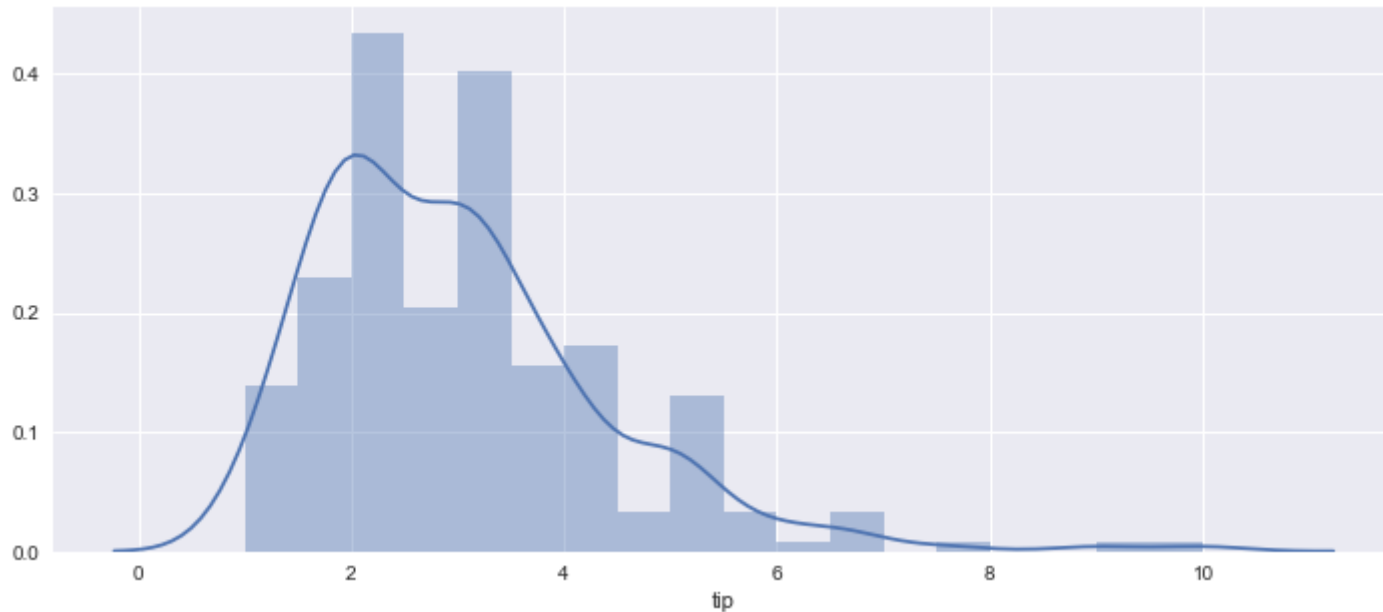
*What can be done change this histogram?*

- How can we achieve the improvements?

## Changing the figure size

```
In [12]: f,ax = plt.subplots(figsize=(12,4)) # set the plot size
sns.distplot(a=tips.tip,
              ax=ax) # use matplotlib defined plot for size)
```

```
Out[12]: <matplotlib.axes._subplots.AxesSubplot at 0x1aa52d35b70>
```



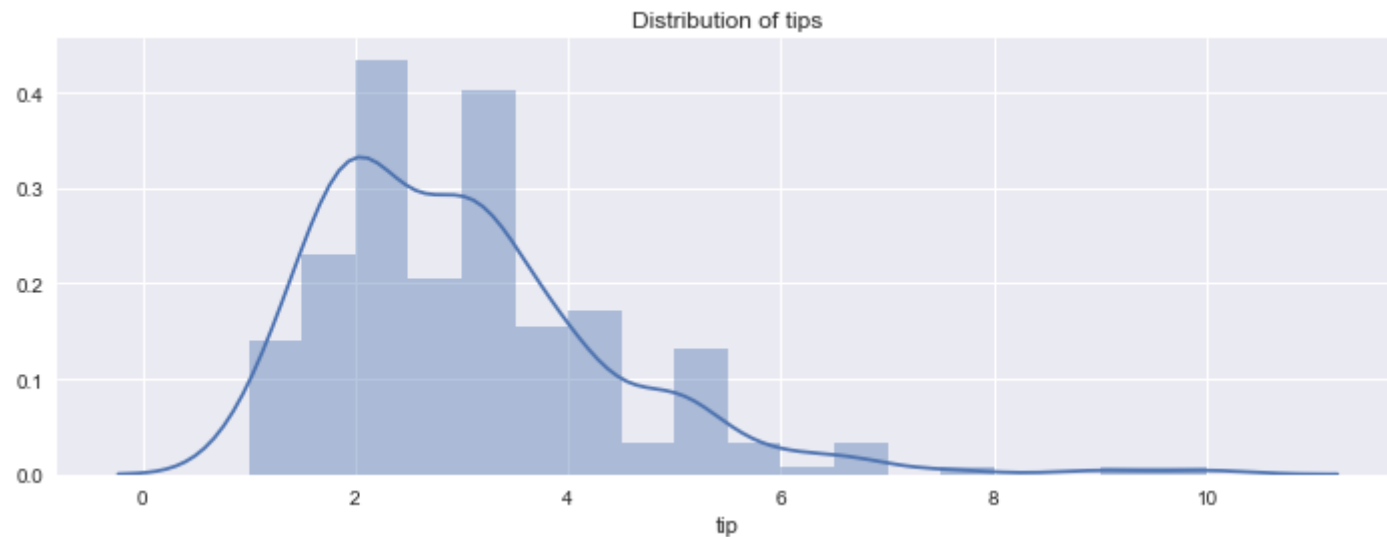


## Set title

```
In [13]: f,ax = plt.subplots(figsize=(12,4))
sns.distplot(a=tips.tip,
             ax=ax)

ax.set_title('Distribution of tips') # setting the title
```

```
Out[13]: <matplotlib.text.Text at 0x1aa52eafc88>
```

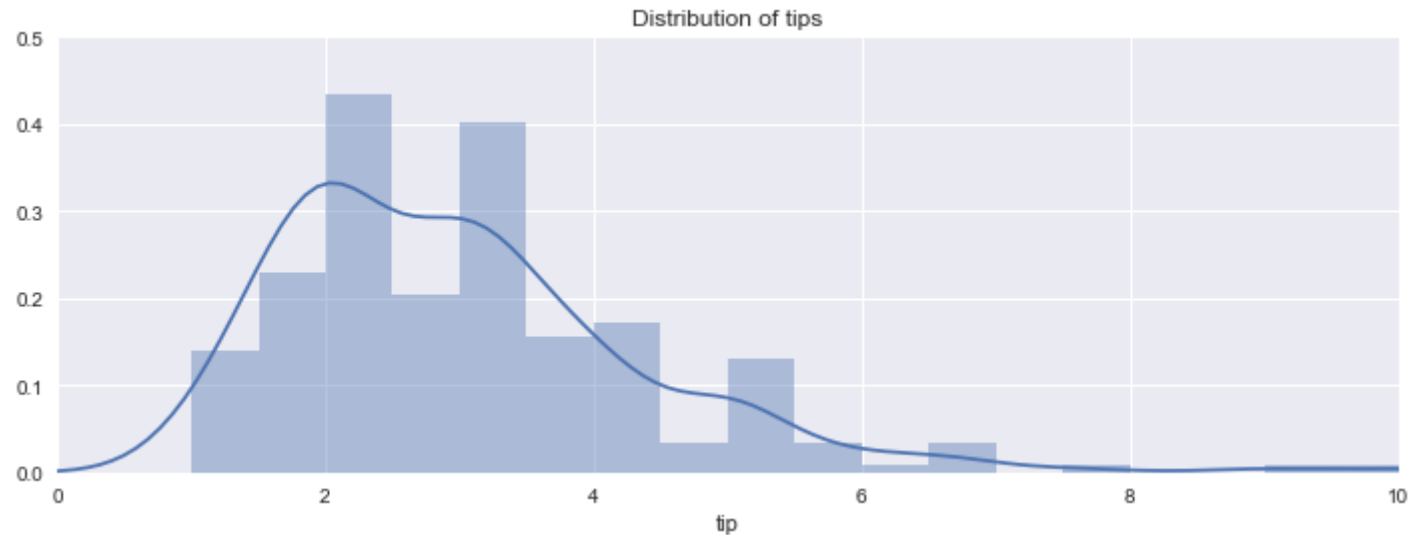


## Change bounds for x- and y-axis

```
In [14]: f,ax = plt.subplots(figsize=(12,4))
sns.distplot(a=tips.tip,
             ax=ax)
ax.set_title('Distribution of tips')

ax.set_xlim(0,10) # set limits for x-axis
ax.set_ylim(0,.5) # set limits for y-axis
```

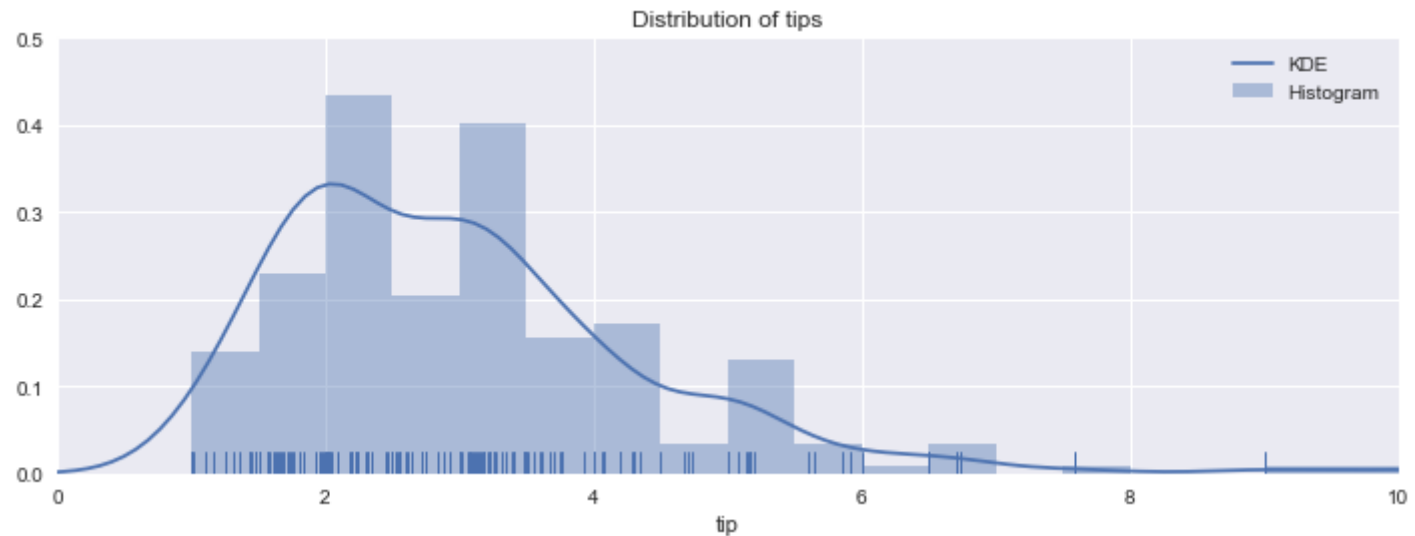
Out[14]: (0, 0.5)



## Add observation rug and legend

```
In [15]: f,ax = plt.subplots(figsize=(12,4))
sns.distplot(a=tips.tip,
             ax=ax,
             rug=True,
             kde_kws={'label': 'KDE'}, # label for KDE plot
             hist_kws={'label': 'Histogram'}) # label for histogram
ax.set_title('Distribution of tips')
ax.set_xlim(0,10)
ax.set_ylim(0,.5)
```

Out[15]: (0, 0.5)



Set font sizes

```
In [18]: f,ax = plt.subplots(figsize=(12,4))
sns.distplot(a=tips.tip, ax=ax, rug=True,
             kde_kws={'label': 'KDE'},
             hist_kws={'label': 'Histogram'})
ax.set_title('Distribution of tips')
ax.set_xlim(0,10)
ax.set_ylim(0,.5)

ax.title.set_fontsize(20) # title
ax.xaxis.label.set_fontsize(16) # xaxis label
tick_labels = ax.get_yticklabels()+ax.get_xticklabels()

# set font sizes
ax.title.set_fontsize(20) # title
ax.xaxis.label.set_fontsize(16) #xaxis label
tick_labels = ax.get_yticklabels()+ax.get_xticklabels()
for item in tick_labels: # axis tickers
    item.set_fontsize(14)

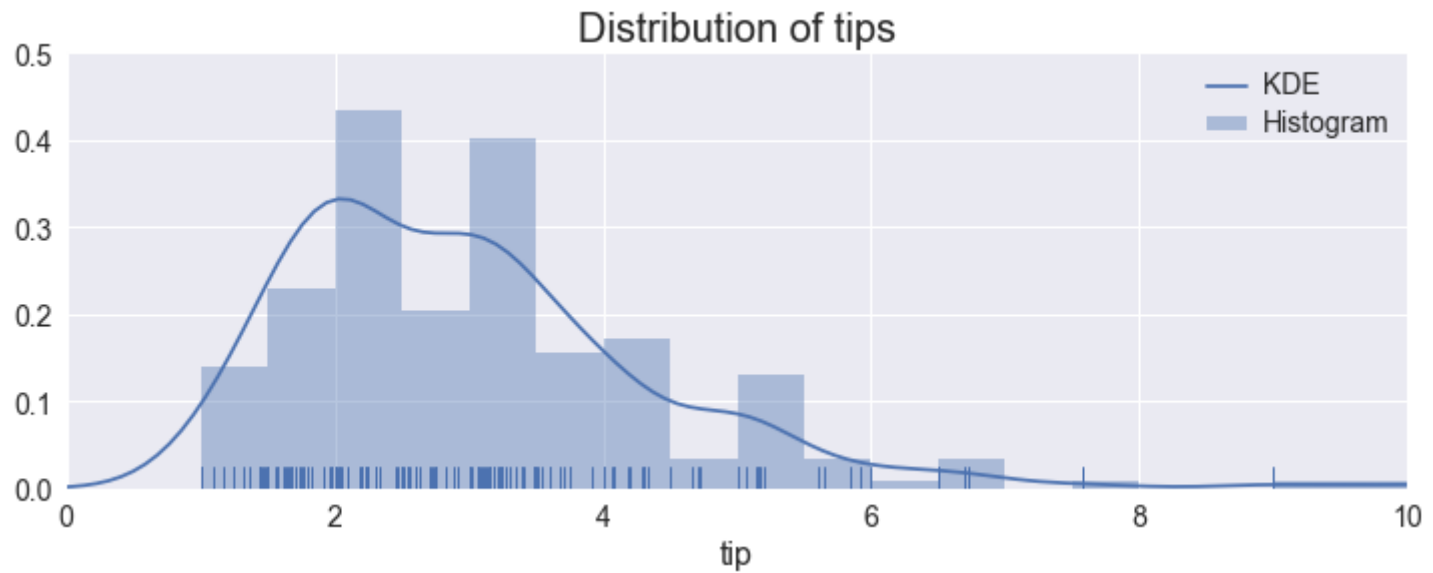
legends = plt.gca().get_legend().get_texts()# Legend labels
plt.setp(legends, fontsize='14') # set size of legend labels
```

```
Out[18]: [None, None, None, None]
```

The final plot

In [48]: f

Out[48]:



## Explanation for the final plot

```
In [ ]: f,ax = plt.subplots(figsize=(12,4)) # set the plot size
sns.distplot(a=tips.tip,
              ax=ax, # use matplotlib defined plot for size
              rug=True, # include raw count
              kde_kws={'label': 'KDE'}, # label for KDE plot
              hist_kws={'label': 'Histogram'}) # label for histogram
ax.set_title('Distribution of tips') # set title
ax.set_xlim(0,10) # set x limits
ax.set_ylim(0,.5) # set y limits

# set font sizes
ax.title.set_fontsize(20) # title
ax.xaxis.label.set_fontsize(16) # xaxis label
for item in ax.get_yticklabels()+ax.get_xticklabels(): # xaxis tickers
    item.set_fontsize(14)
plt.setp(plt.gca().get_legend().get_texts(), fontsize='14') # legend labels
```

Exporting our final plot

```
In [69]: f.figure.savefig('my_histogram.pdf')
```

```
Out[69]: <bound method Figure.savefig of <matplotlib.figure.Figure object at 0x000001AA58115F60>>
```



Setting - standard plot size

```
In [26]: plt.rcParams['figure.figsize'] = 12,5 # set default size of plots
```

# Univariate categorical data

*What if we have categorical data?*

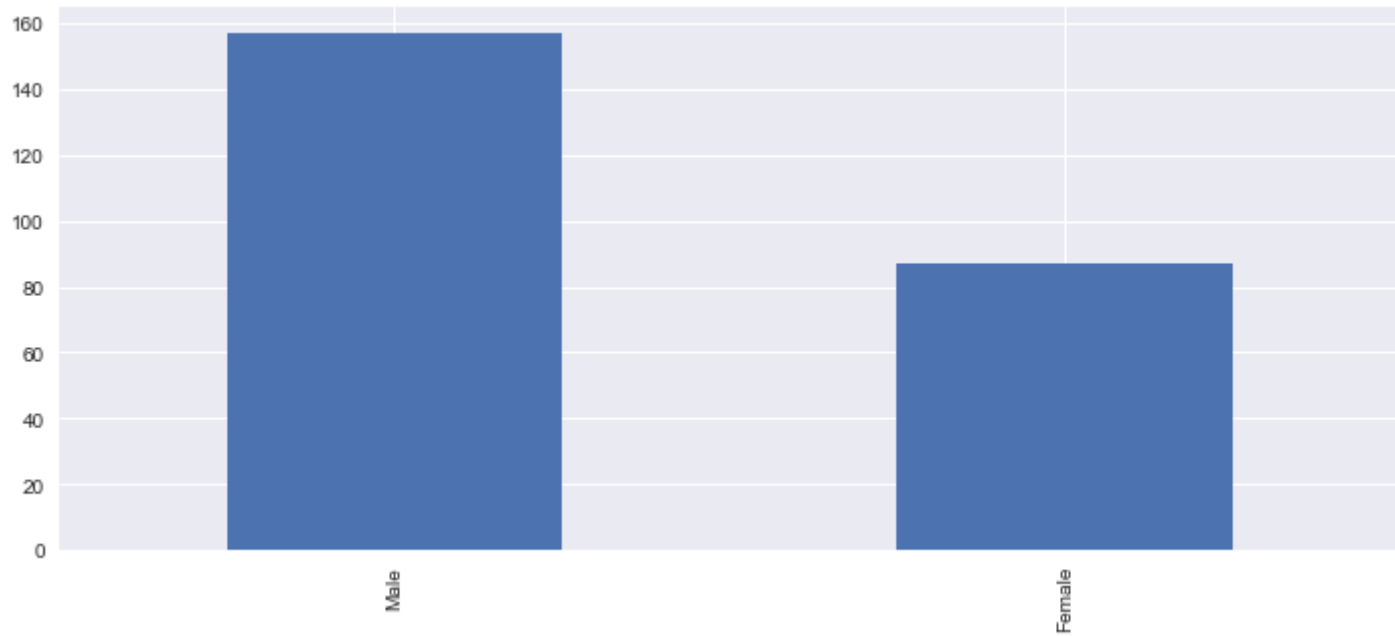
What is categorical data? Example gender count:

```
In [ ]: count_sex = tips.sex.value_counts()
```

Let's plot this with bars:

```
In [28]: count_sex.plot.bar()
```

```
Out[28]: <matplotlib.axes._subplots.AxesSubplot at 0x1aa53b5c9e8>
```



Let's plot this as a pie:

```
In [ ]: count_sex.plot.pie()
```

# Univariate series plots

# Simulating data

Let's create some data

```
In [29]: np.random.seed(123) # set seed - then we get same random data  
  
         ts = np.random.normal(0,1,[1000,3]) # time series with no slope  
  
         dates = pd.date_range(start='20170801', periods=1000, freq='D') # 1000 daily observations beginning Aug 1, 17
```

## Simulating data (2)

We use our data to create a DataFrame with a time series index.

```
In [30]: df_norm = pd.DataFrame(data=ts, # our data
                                index=dates, # our date indices
                                columns=['A', 'B', "C"]) # column names
```

```
In [34]: df = df_norm.cumsum() # use cumulative sum
```

```
In [36]: df['A'] += np.arange(0,60,.06) # add-to 'A' a linear trend with .06 increments
df['B'] += np.arange(0,30,.03) # add-to 'B' a linear trend with .03 increments
```

Quiz: is our data in long or wide format?

# Power of Pandas

*Why is pandas used in fin-tech so much?*

Example: Plotting time series for one variable (e.g. GDP, inflation)

```
In [38]: df['A'].plot()
```

```
Out[38]: <matplotlib.axes._subplots.AxesSubplot at 0x1aa53e672b0>
```





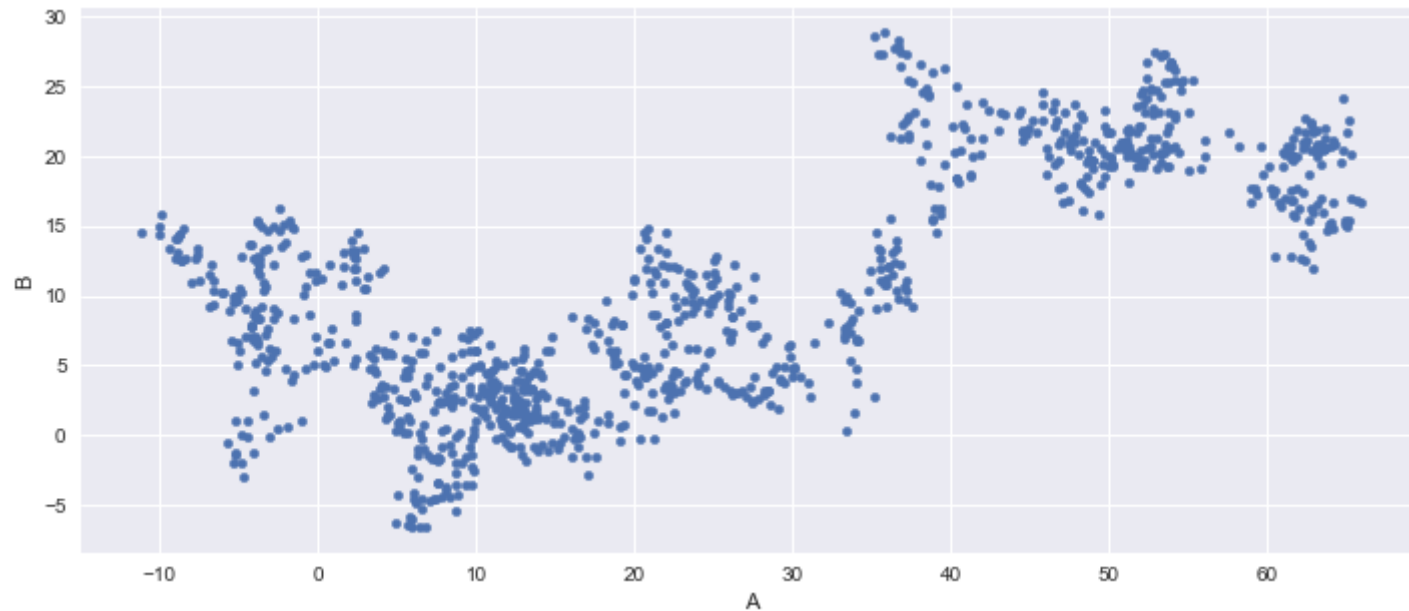
# Scatter and related plots

Raw distribution of two numeric variables

# Pandas scatter plot

```
In [39]: df.plot.scatter('A', 'B')
```

```
Out[39]: <matplotlib.axes._subplots.AxesSubplot at 0x1aa53efbeb8>
```

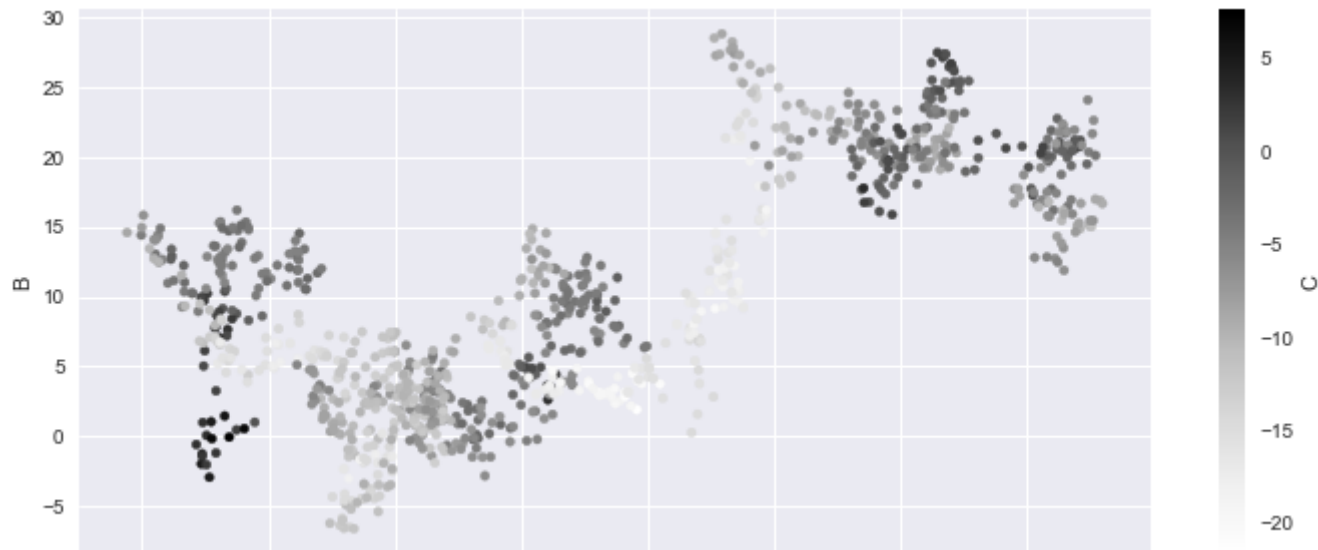


Quiz: How might we alter the scatter plot?

- Let's try to change the colors of the dots:

```
In [40]: df.plot.scatter(x='A', y='B', c='C')
```

```
Out[40]: <matplotlib.axes._subplots.AxesSubplot at 0x1aa53cb9160>
```

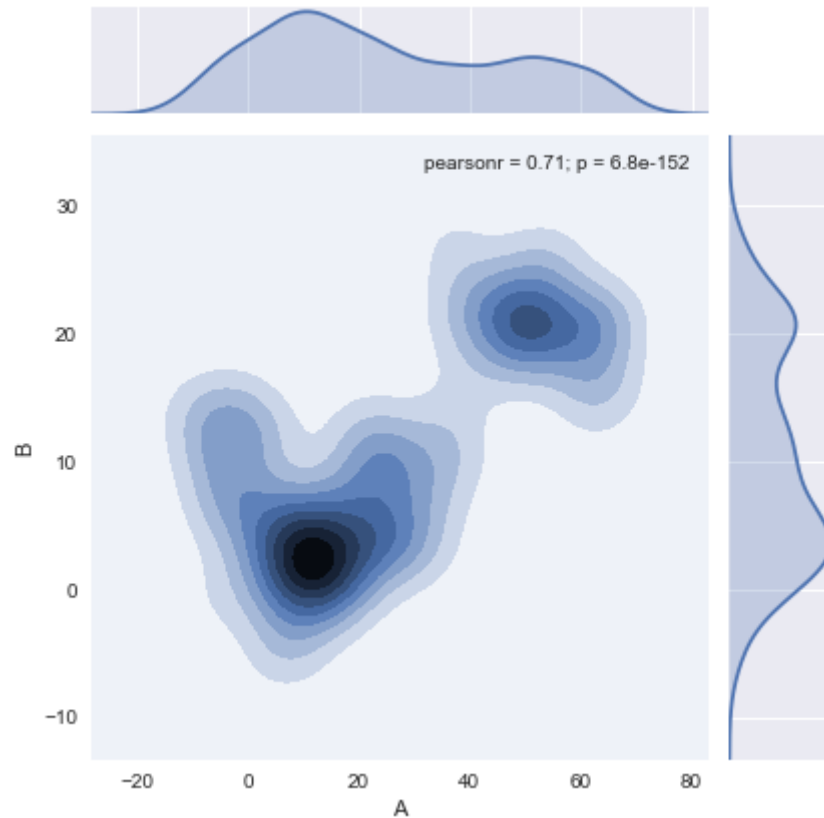


# Seaborn for scatter and related

The jointplot for scatter

```
In [43]: sns.jointplot(x='A',y='B', data=df, kind='kde')
```

```
Out[43]: <seaborn.axisgrid.JointGrid at 0x1aa543a40f0>
```

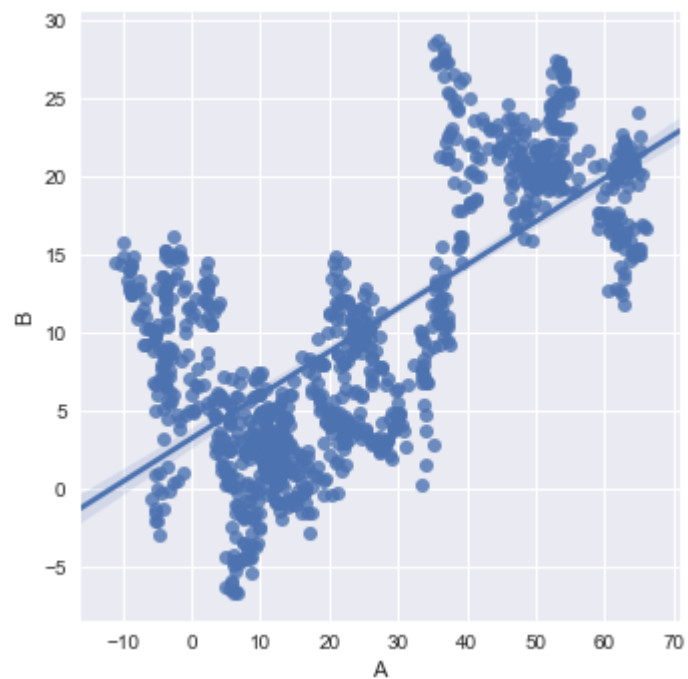


How can we modify this? KDE, hexbin?

- The regression plot

```
In [44]: sns.lmplot('A', 'B', data=df)
```

```
Out[44]: <seaborn.axisgrid.FacetGrid at 0x1aa55711278>
```



- Multiple scatterplots (correlation matrix style)

# Plotting multiple variables

## *Wide formatting*

Which tool should we pick for wide data?

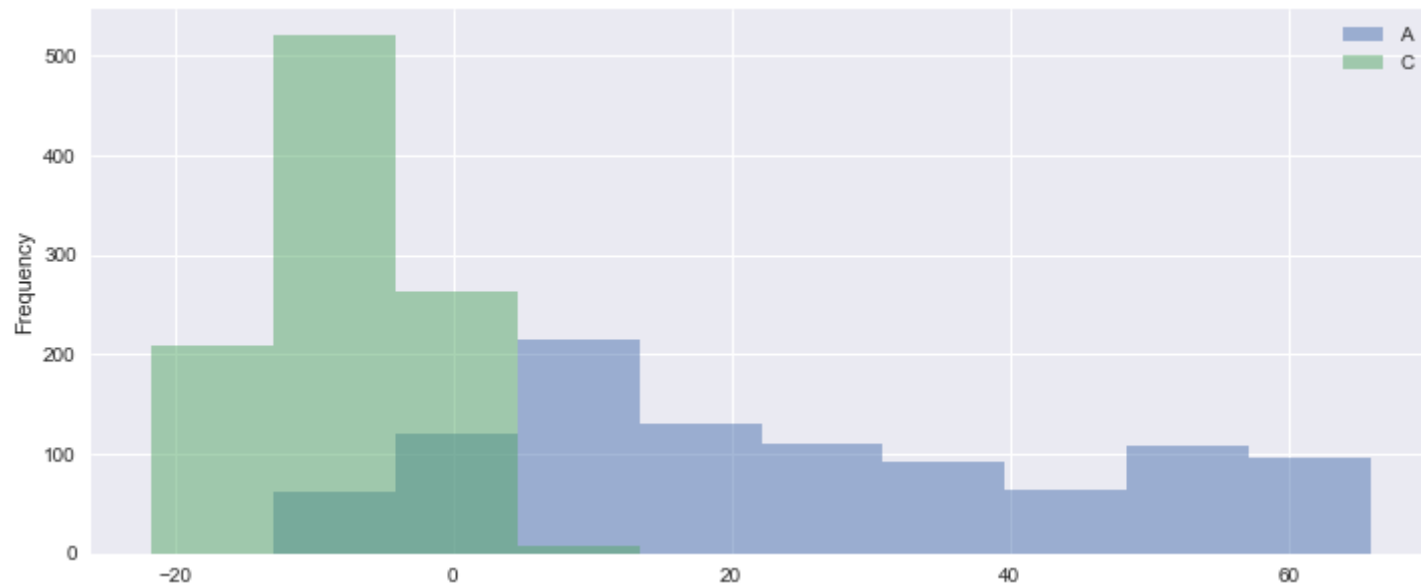
Pandas!



# Histogram

In [48]: `df[['A','C']].plot.hist(alpha=.5)`

Out[48]: `<matplotlib.axes._subplots.AxesSubplot at 0x1aa558667b8>`



# Time series plot

In [49]: `df.plot()`

Out[49]: `<matplotlib.axes._subplots.AxesSubplot at 0x1aa563b8940>`

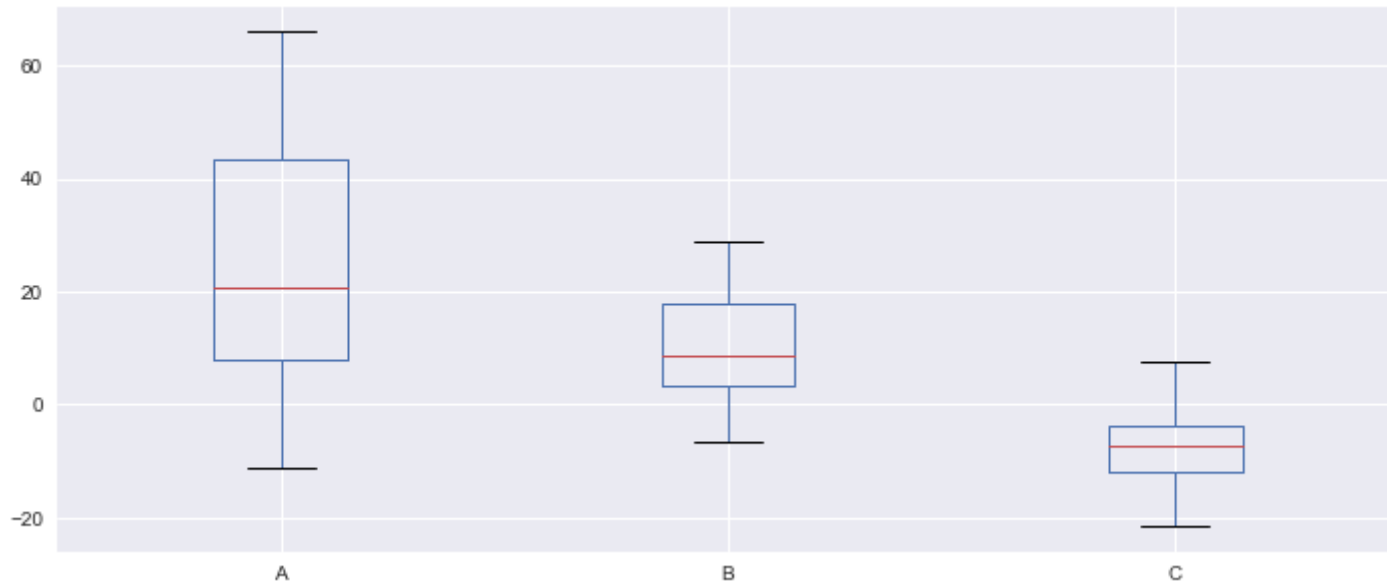


# The boxplot

Measure: median + top,bottom for quartiles and deciles.

```
In [51]: df.plot.box()
```

```
Out[51]: <matplotlib.axes._subplots.AxesSubplot at 0x1aa575d8630>
```



# Plotting multiple variables

## *Using long format*

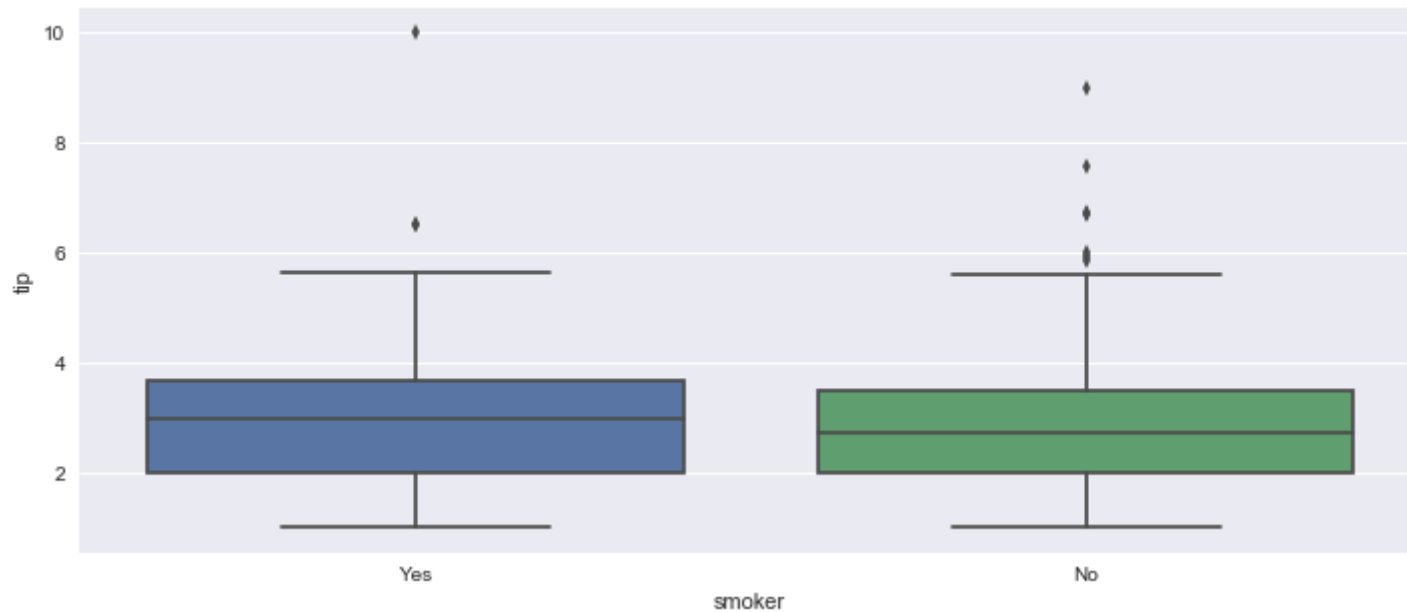
What was long format? (one row per observation)

What columns can we use as extra info? Categorical variables?

Let's make a boxplot of tips - distinguish by smoker:

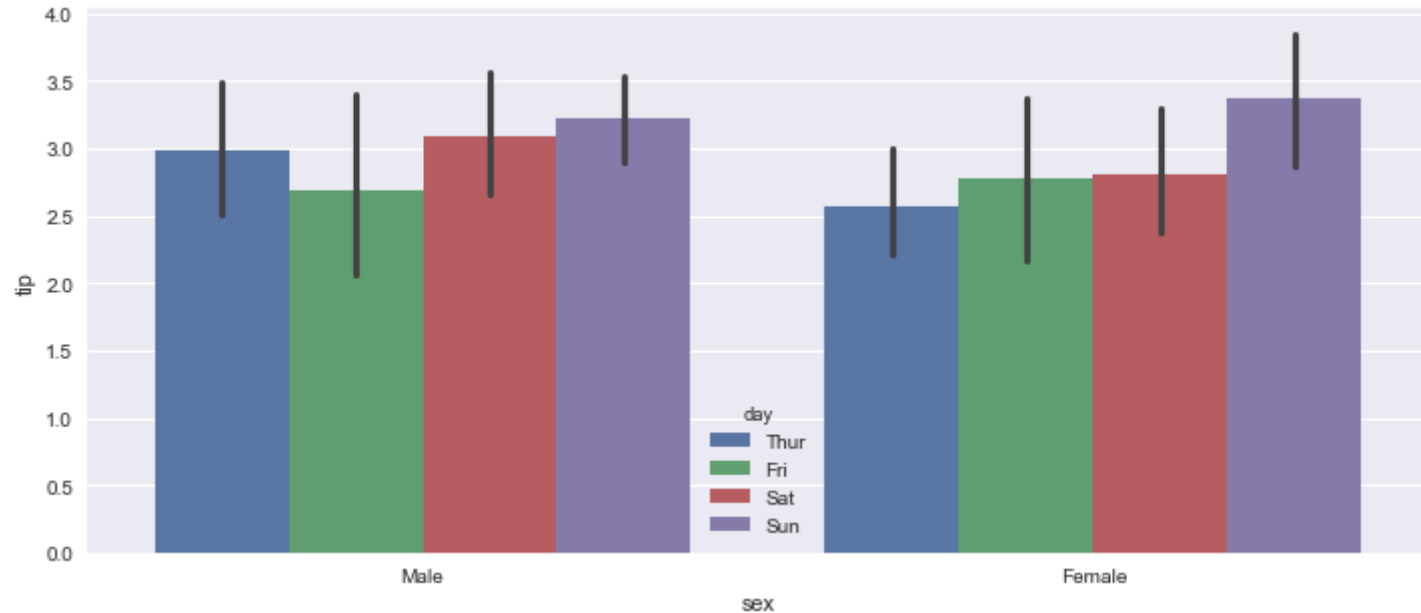
```
In [52]: sns.boxplot(y='tip', x='smoker', data=tips)
```

```
Out[52]: <matplotlib.axes._subplots.AxesSubplot at 0x1aa577885c0>
```



Let's try a barplot of tips. Distinguish in addition by gender:

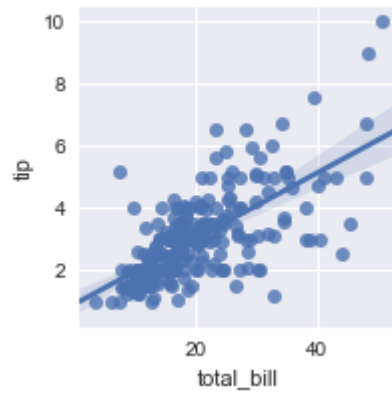
```
In [66]: f = sns.barplot(x='sex', y='tip', hue='day', data=tips)
f.figure.savefig('tip_gender_day.png')
```



# Data exploration

## The FacetGrid

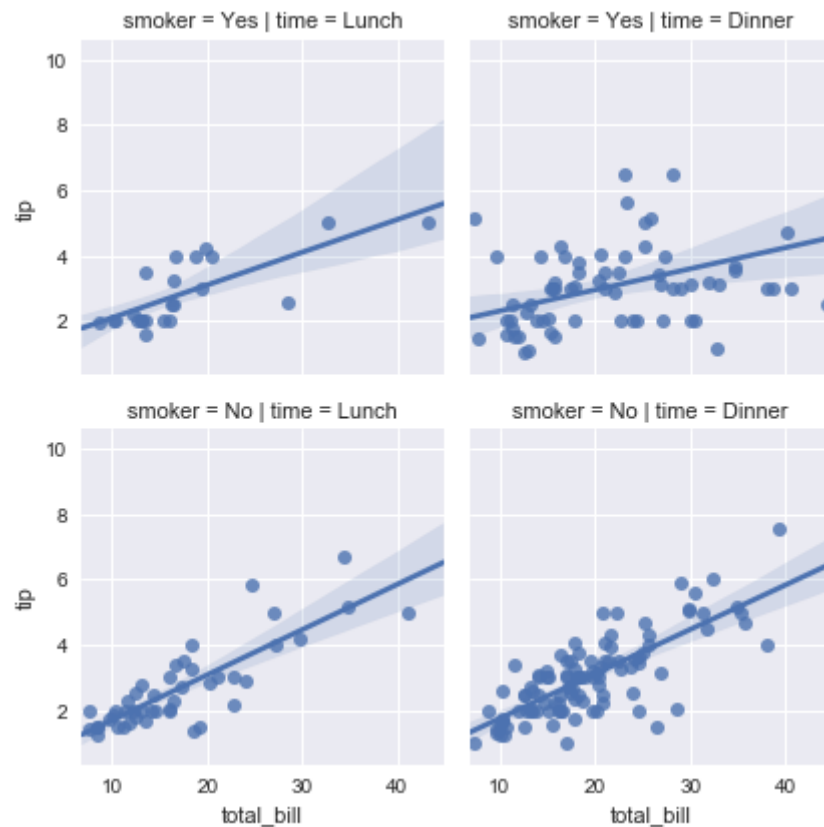
```
In [57]: g = sns.FacetGrid(tips)
g = g.map(sns.regplot, 'total_bill', 'tip')
```





Let's try to add gender distinctive slopes

```
In [63]: g = sns.FacetGrid(tips, row = 'smoker', col='time')  
g = g.map(sns.regplot, 'total_bill', 'tip')
```



Let's try to further add separate estimates for smoking status

```
In [ ]: g = sns.FacetGrid(tips, col = 'smoker')  
g = g.map(sns.regplot, 'total_bill', 'tip')
```

Can we say anything about smokers tipping behavior?

**Summing up**

## Exploratory analysis

- We learned how we could leverage Pandas:
  - data in wide format
  - time series data
- We learned that Seaborn:
  - makes great, first visualization
  - powerful for exploring data patterns

## Explanatory plots

- Customization is time consuming.
- Matplotlib must be configured.

## If you want to learn more

Other useful plots can be found in the tutorials of Seaborn (<https://seaborn.pydata.org/>).

To master plot making in python the tweaking with matplotlib (<https://matplotlib.org/>) is essential. Worth looking into are:

- a general tutorial can be found here ([https://matplotlib.org/users/pyplot\\_tutorial.html](https://matplotlib.org/users/pyplot_tutorial.html));
- subplots ([https://matplotlib.org/examples/pylab\\_examples/subplots\\_demo.html](https://matplotlib.org/examples/pylab_examples/subplots_demo.html)) for multiple figures (with for loops);
- color palettes (<https://matplotlib.org/users/colormaps.html>) for styling figures;

Plotting network and geographic data has other types of plots - see readings (<https://abjier.github.io/sds/readings/>) for references to NetworkX and GeoPandas.