Creating Advanced Visualizations with Matplotlib

Douglas Starnes

SOFTWARE ENGINEER / CONFERENCE SPEAKER / TECH AUTHOR

@poweredbyaltnet http://douglasstarnes.com

Subplots

By default, visualizations are drawn in the same space

Subplots draw in separate spaces

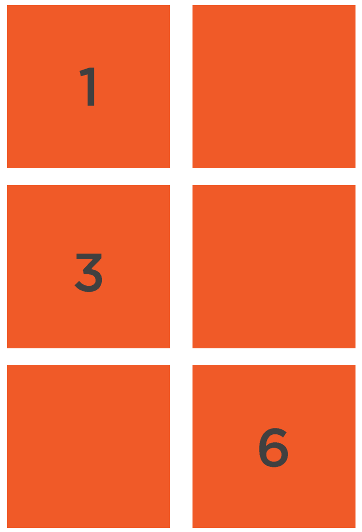
The subplot() function

- Accepts the number of rows
- And number of columns
- Three rows and two columns holds 6 different charts

Indexing Subplots

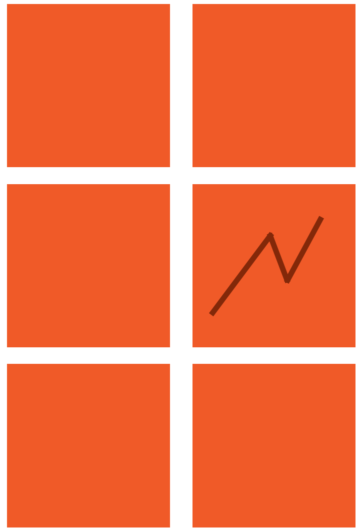
The index is one based!

```
plt.subplot(3, 2, ?)
```



Rows and columns included in *every* call to subplot().

```
plt.subplot(3, 2, 4)
plt.plot( ... )
```



The number of rows, number of columns and index can be specified as a single number

This is not a concatenation of strings, but a number

- ‘323’ (string ‘three two three’)
- 323 (integer ‘three hundred twenty-three’)

The number of rows, columns or index cannot be greater than 9

Subplots: Alternative Method

The subplots() (plural, with an ‘s’) function

Much easier to use

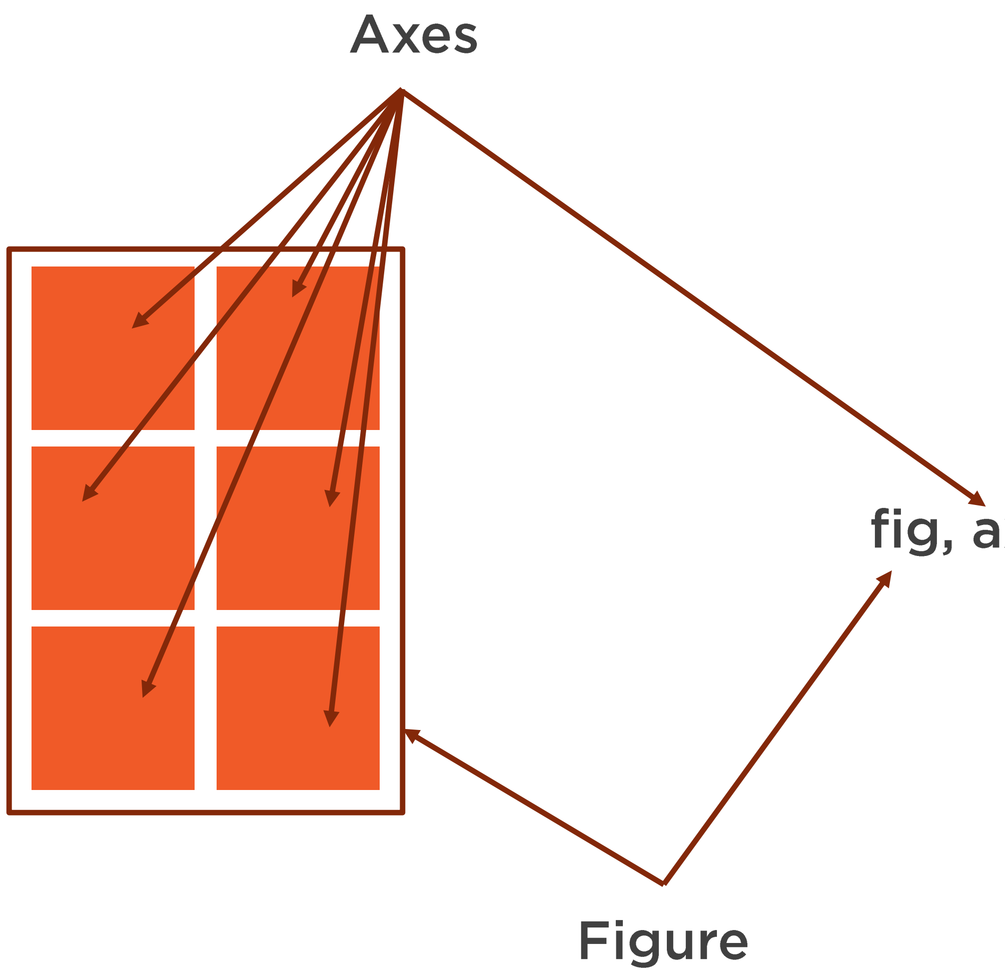
First two values are the number of rows and columns

Return value is a tuple

- Figure
- An ndarray of Axes instances

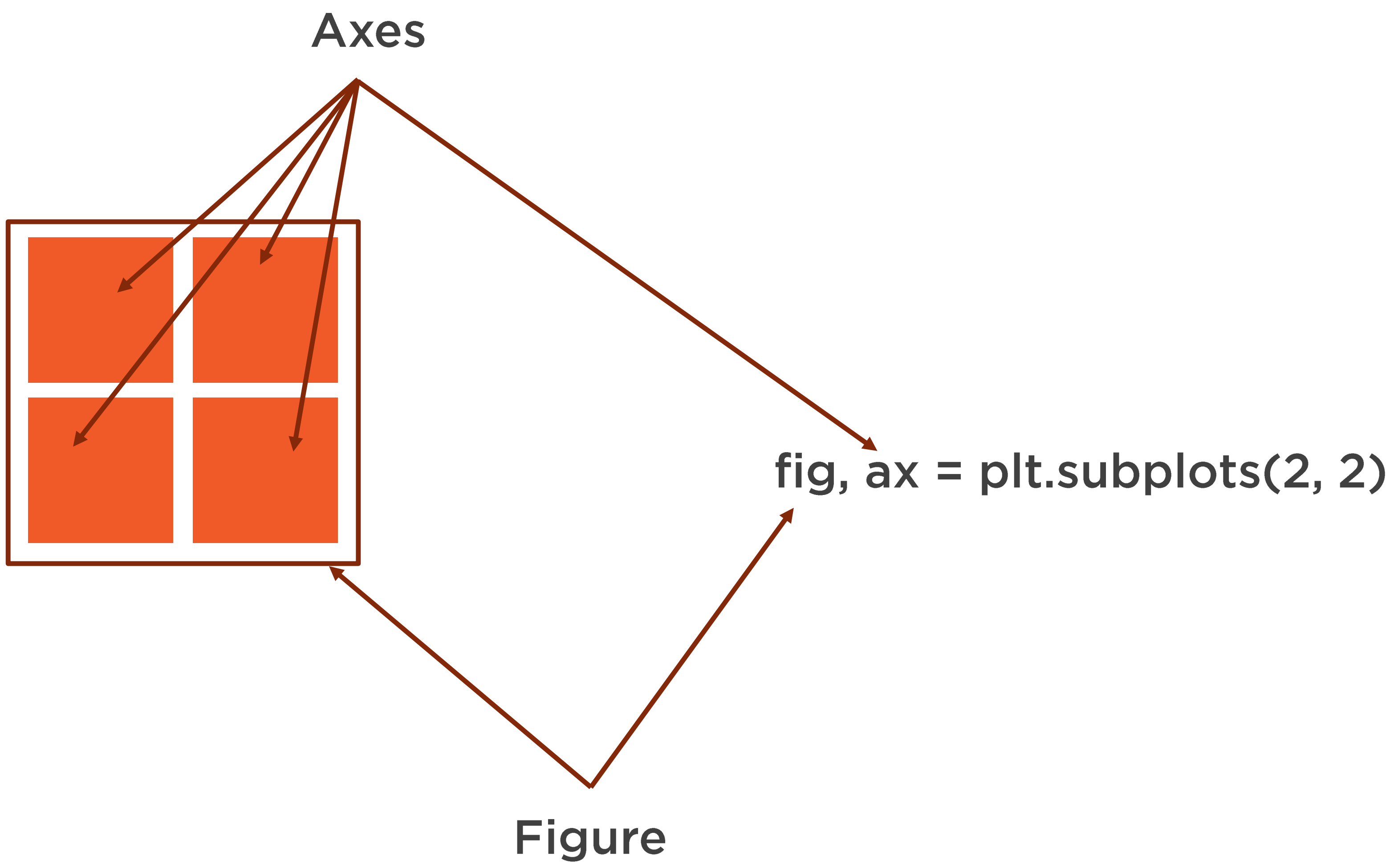
Axes with an ‘e’ and axis with an ‘i’ are different

Figures and Axes



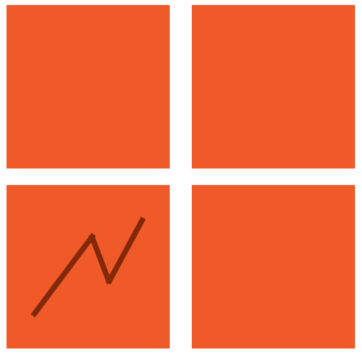
```
fig, ax = plt.subplots(3, 2)
```

Figures and Axes



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

Figures and Axes



```
fig, ax = plt.subplots(2, 2)
ax[1, 0].plot( ... )
```

Text

Several ways to use text have already been covered

- Labels along the x and y axis
- Labels for wedges of pie charts
- Legends

Axis as a whole, not the individual ticks

- The xlabel() and ylabel() functions

Title for the visualization, not a specific part

- The title() function

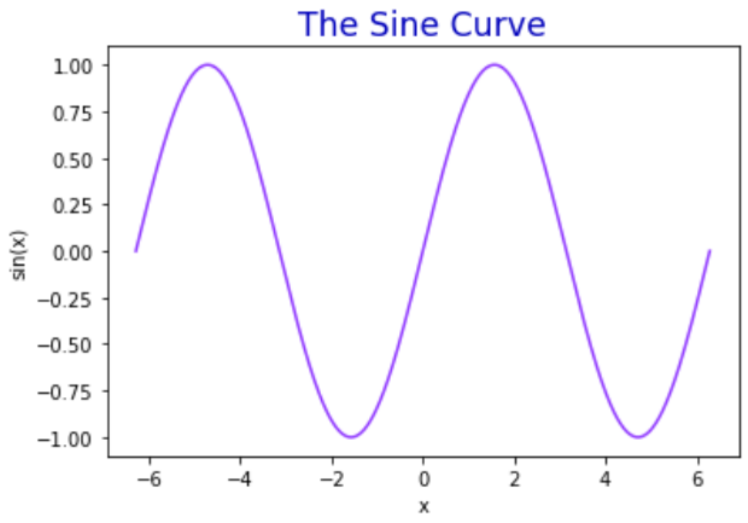
Text Properties

The xlabel(), ylabel() and title() functions return Text instances, like the labels of the pie chart

Methods like set_fontweight() and set_fontstyle() are still valid

The fontdict= keyword argument

- Dictionary-like
- Keys are valid Text properties



```
plt.title('The Sine Curve', fontdict={
    'size': 'xx-large',
    'color': '#0000BB'
})
```

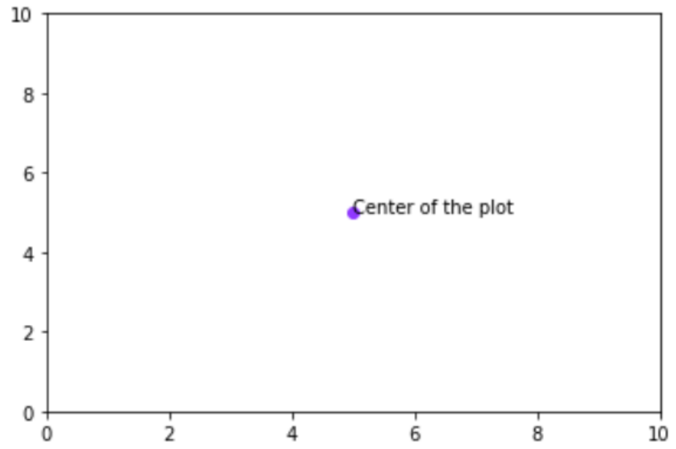
Free Form Text

The text() function accepts

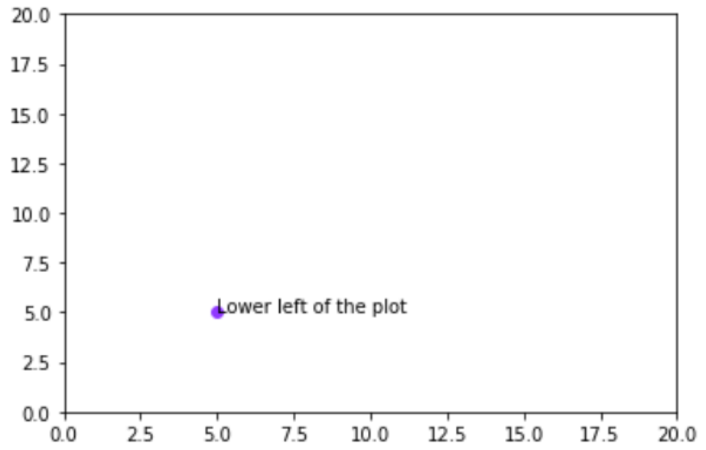
- x-coordinates
- y-coordinates
- String to draw

The x-y coordinates are the lower-left corner of the bounding box of the text

Coordinates are also relative to the ranges of the visualization



```
plt.text(5, 5, "Center
    of the plot")
```



```
plt.text(5, 5, "Lower
    left of the plot")
```

Dashes

The withdash= keyword argument draws a dash to the right of the text

- The dashlength= keyword argument is the length of the dash
- The right endpoint of the dash is anchored at the x-y coordinates passed to text()
- The dashdirection= keyword argument places the dash on the left or right
- The withdash= and dashlength= keyword arguments are a pair
- They are also deprecated

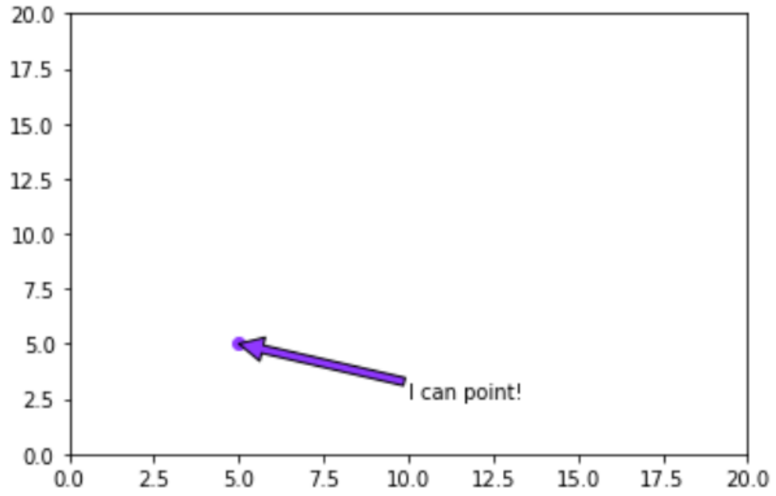
Annotations

The annotate() function

- Accepts the text to annotate
- A tuple with the x-y coordinate of the *point to annotate*
- A tuple with the x-y coordinate of the text (optional)

The arrowprops= keyword argument

- A dictionary used to format an arrow from the text to the point



```
plt.annotate("I can point!", (5, 5), (10, 2.5), arrowprops={})
```