- Shin’s Lab -

Python for Data Visualization

Python for Data Visualization

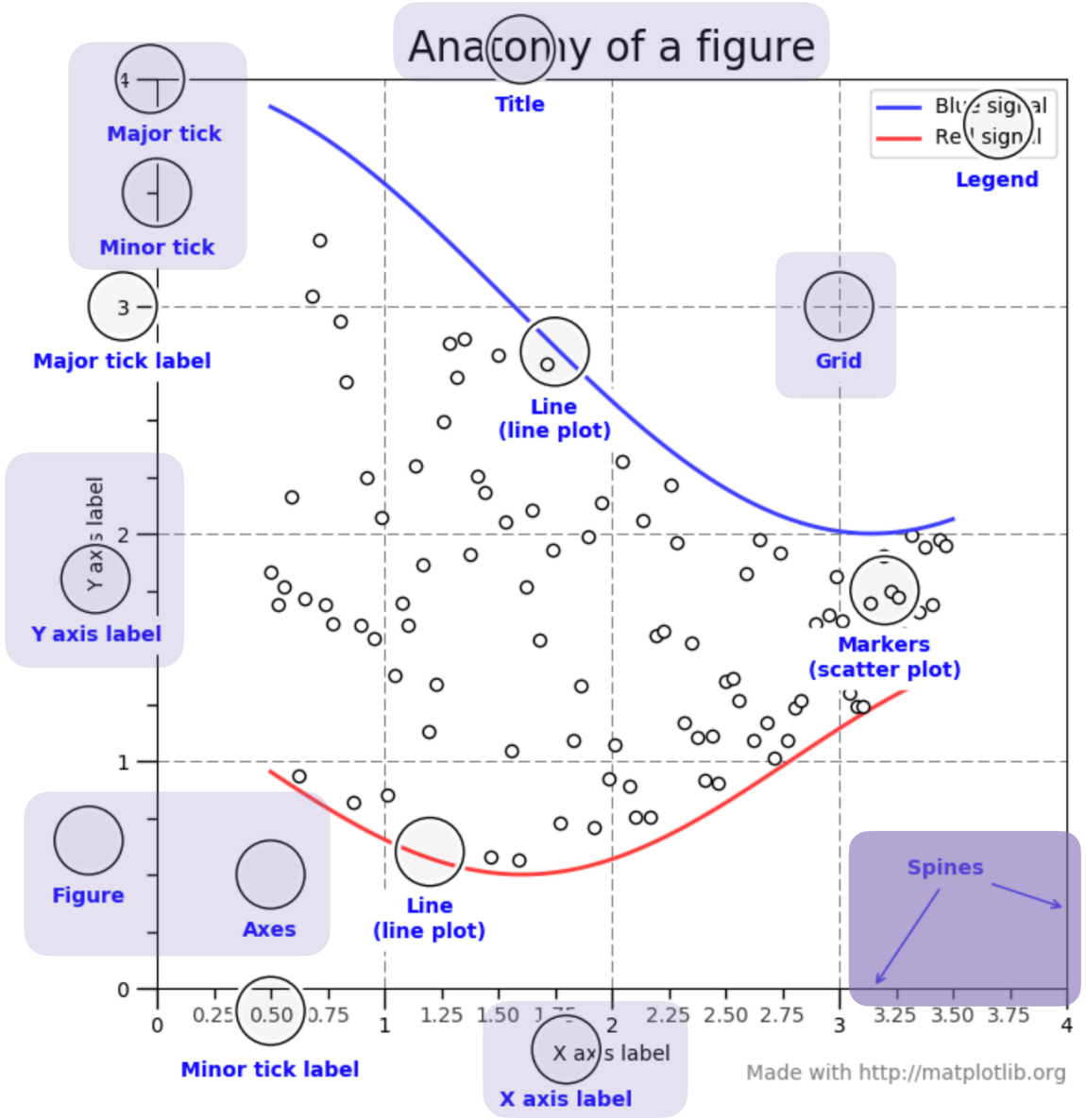
-Chapter.1 Matplotlib Anatomy -

Python for Data Visualization

-Chapter.1 Matplotlib Anatomy -

1-06. Spines

1. Spines
2. spine.set_visible
3. spine.set_position(Basic Usage)
4. spine.set_position(Tuple Argument)


Anatomy of a figure
Title
Major tick
Minor tick
Major tick label
Legend
Grid
Line
(line plot)
Y axis label
Markers
(scatter plot)
Figure
Axes
Line
(line plot)
Spines
Minor tick label
X axis label
Made with http://matplotlib.org

1. Spines

matplotlib.spines

class matplotlib.spines.Spine(*axes*, *spine_type*, *path*, ***kwargs*) [source]

Bases: matplotlib.patches.Patch

An axis spine -- the line noting the data area boundaries.

Spines are the lines connecting the axis tick marks and noting the boundaries of the data area. They can be placed at arbitrary positions. See set_position for more information.

The default position is ('outward', 0).

Spines are subclasses of Patch, and inherit much of their behavior.

Spines draw a line, a circle, or an arc depending if set_patch_line, set_patch_circle, or set_patch_arc has been called. Line-like is the default.

```
'set_aa',
'set_agg_filter',
'set_alpha',
'set_animated',
'set_antialiased',
'set_bounds',
'set_capstyle',
'set_clip_box',
'set_clip_on',
'set_clip_path',
'set_color',
'set_contains',
'set_ec',
'set_edgecolor',
'set_facecolor',
'set_fc',
'set_figure',
'set_fill',
'set_gid',
'set_hatch',
'set_in_layout',
```

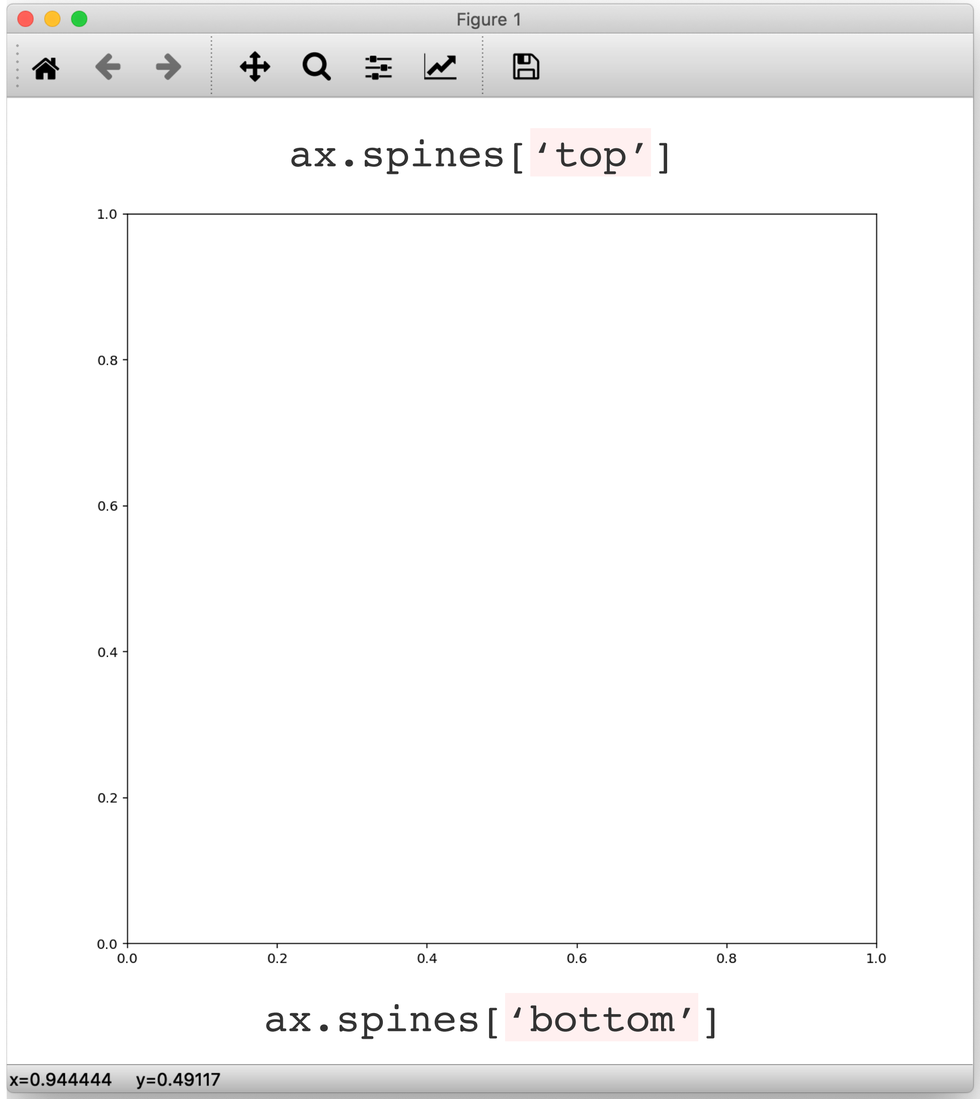
```
'set_joinstyle',
'set_label',
'set_linestyle',
'set_linewidth',
'set_ls',
'set_lw',
'set_patch_arc',
'set_patch_circle',
'set_patch_line',
'set_path_effects',
'set_picker',
'set_position',
'set_rasterized',
'set_sketch_params',
'set_smart_bounds',
'set_snap',
'set_transform',
'set_url',
'set_visible',
'set_zorder',
```

1. Spines(Spine Dictionary)

```
import matplotlib.pyplot as plt

fig, ax = plt.subplots(figsize = (10,10))

print(type(ax.spines)) <class 'collections.OrderedDict'>

print(ax.spines) OrderedDict([('left', <matplotlib.spines.Spine object at 0x7fb4067d3550>),
                              ('right', <matplotlib.spines.Spine object at 0x7fb43a030390>),
                              ('bottom', <matplotlib.spines.Spine object at 0x7fb43a030610>),
                              (‘top', <matplotlib.spines.Spine object at 0x7fb43a0307d0>)])

print(ax.spines.keys()) odict_keys(['left', 'right', 'bottom', 'top'])

print(ax.spines.values())
  odict_values([<matplotlib.spines.Spine object at 0x7fb4067d3550>,
                <matplotlib.spines.Spine object at 0x7fb43a030390>,
                <matplotlib.spines.Spine object at 0x7fb43a030610>,
                <matplotlib.spines.Spine object at 0x7fb43a0307d0>])
```

1. Spines(Spine Location)

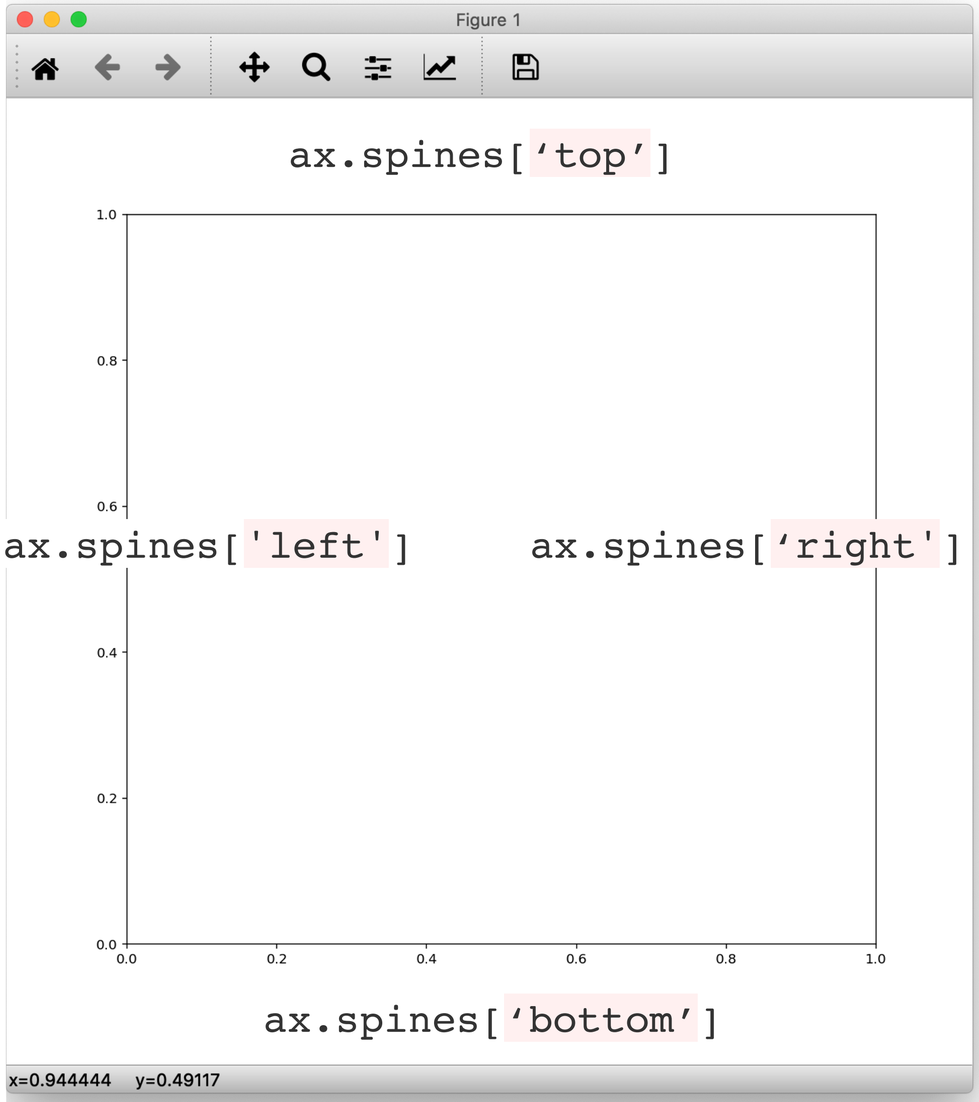
ax.spines['left']



ax.spines['right']

1. Spines(Get Spine Object)

```
import matplotlib.pyplot as plt

fig, ax = plt.subplots()

print(ax.spines['left'])      Spine

print(ax.spines['bottom'])    Spine

print(ax.spines['right'])     Spine

print(ax.spines['top'])       Spine
```

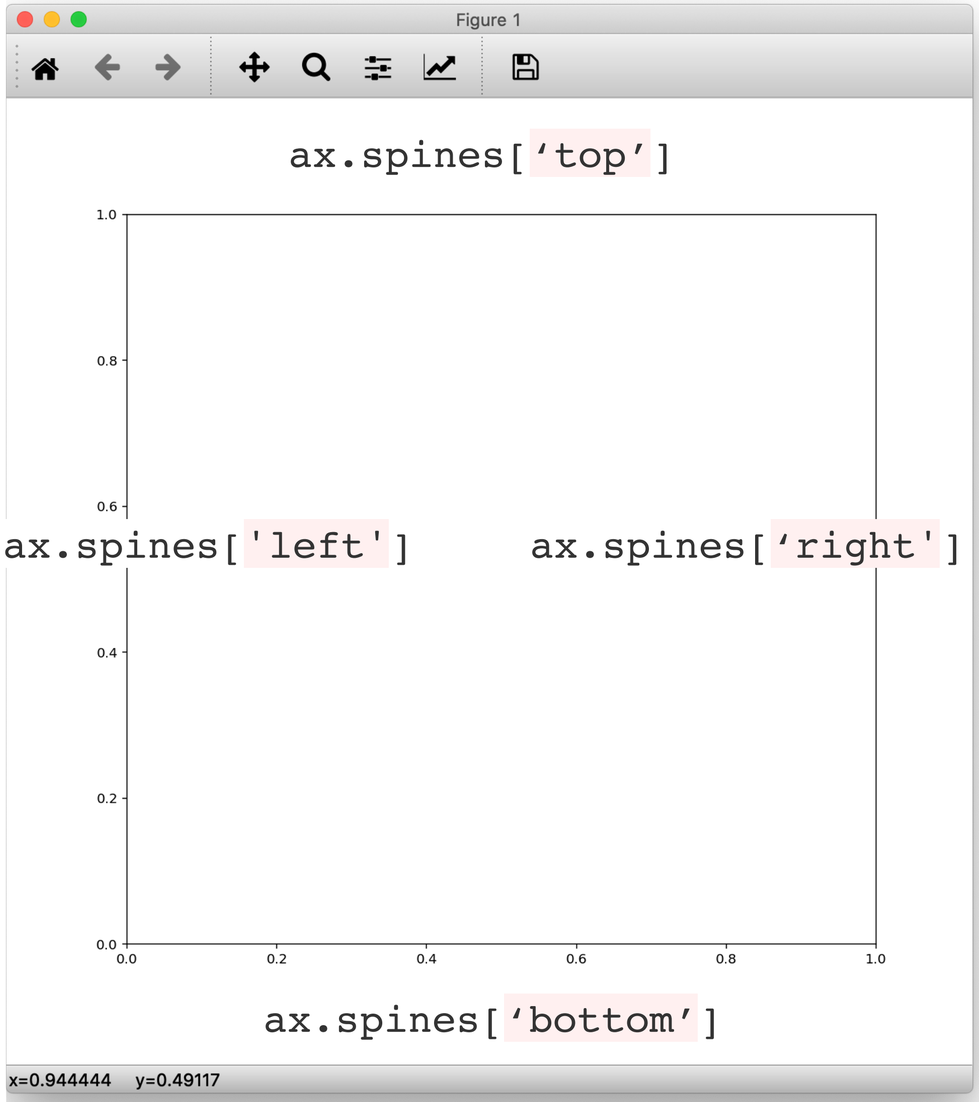


1. Spines(Get Spine Object)

```
import matplotlib.pyplot as plt

fig, ax = plt.subplots()

for spine_loc, spine in ax.spines.items():
    print(spine_loc, spine, sep=' - ')
```

```
left - Spine
right - Spine
bottom - Spine
top - Spine
```

```
for spine_loc, spine in ax.spines.items():
    if spine_loc in ['left', 'bottom']:
        pass

    if spine_loc in ['right', 'top']:
        pass
```



1. Spines(Spine APIs)

```
'set_aa',
'set_agg_filter',
'set_alpha',
'set_animated',
'set_antialiased',
'set_bounds',
'set_capstyle',
'set_clip_box',
'set_clip_on',
'set_clip_path',
'set_color',
'set_contains',
'set_ec',
'set_edgecolor',
'set_facecolor',
'set_fc',
'set_figure',
'set_fill',
'set_gid',
'set_hatch',
'set_in_layout',
```

```
'set_joinstyle',
'set_label',
'set_linestyle',
'set_linewidth',
'set_ls',
'set_lw',
'set_patch_arc',
'set_patch_circle',
'set_patch_line',
'set_path_effects',
'set_picker',
'set_position',
'set_rasterized',
'set_sketch_params',
'set_smart_bounds',
'set_snap',
'set_transform',
'set_url',
'set_visible',
'set_zorder',
```

```
ax.spines[spine_loc].set_visible(False)
```

```
ax.spines[spine_loc].set_position('center')
```

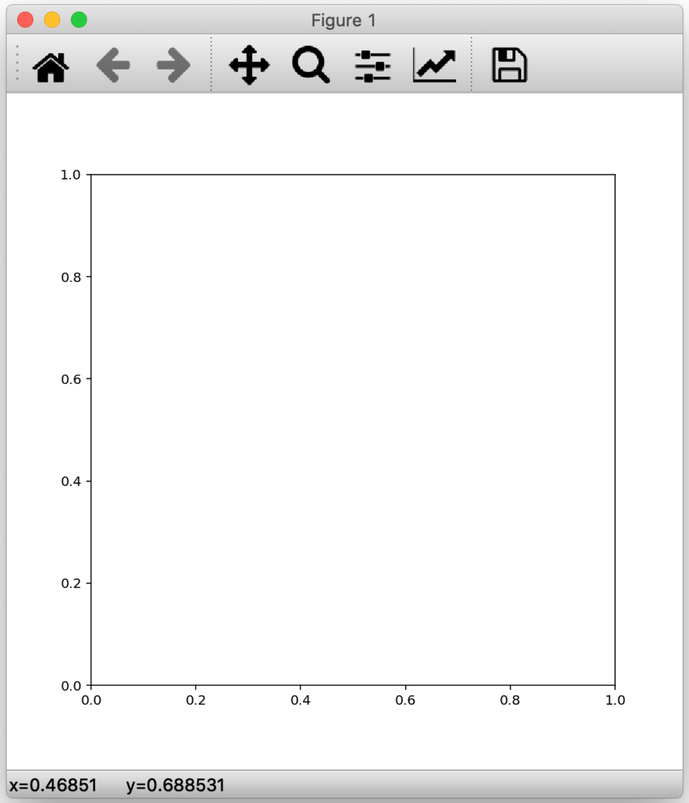
```
ax.spines[spine_loc].set_linewidth(2)
```

```
ax.spines[spine_loc].set_alpha(0.5)
```

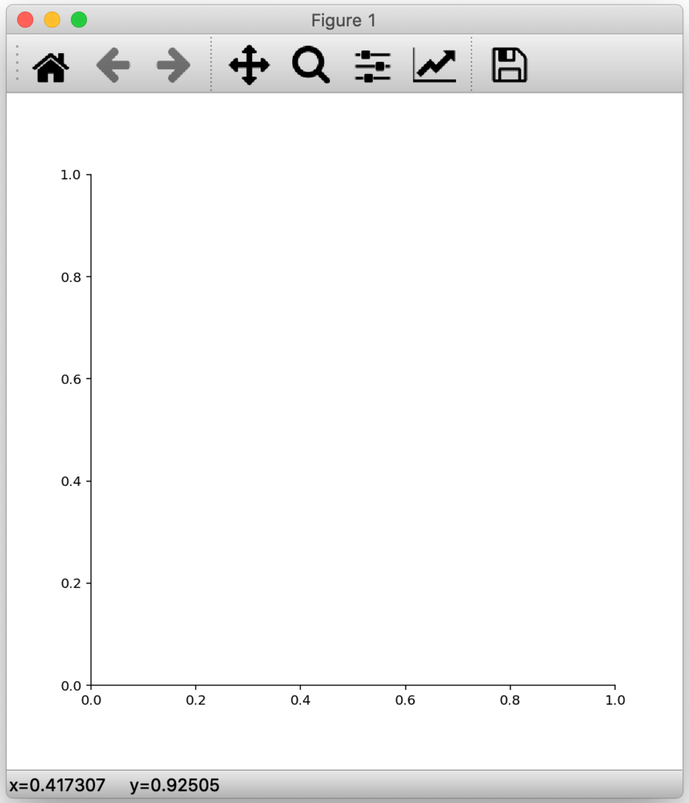
```
ax.spines[spine_loc].set_color('navy')
```

2. spine.set_visible

```
fig, ax = plt.subplots(figsize=(7, 7))
```



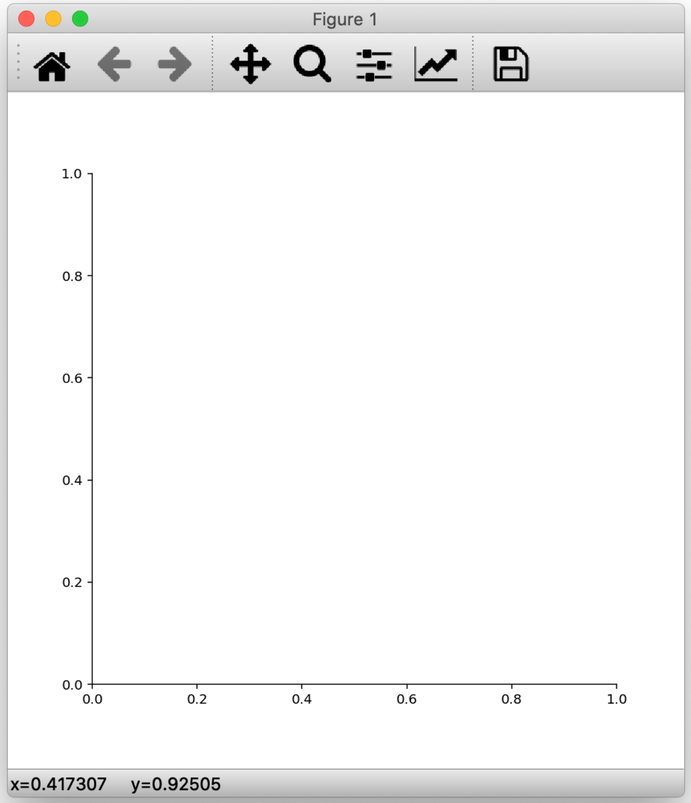

```
fig, ax = plt.subplots(figsize=(7, 7))

ax.spines['right'].set_visible(False)
ax.spines['top'].set_visible(False)
```



2. spine.set_visible(Basic Usage)

```
fig, ax = plt.subplots(figsize=(7, 7))

ax.spines['right'].set_visible(False)
ax.spines['top'].set_visible(False)
```

```
fig, ax = plt.subplots(figsize=(7, 7))

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)
```



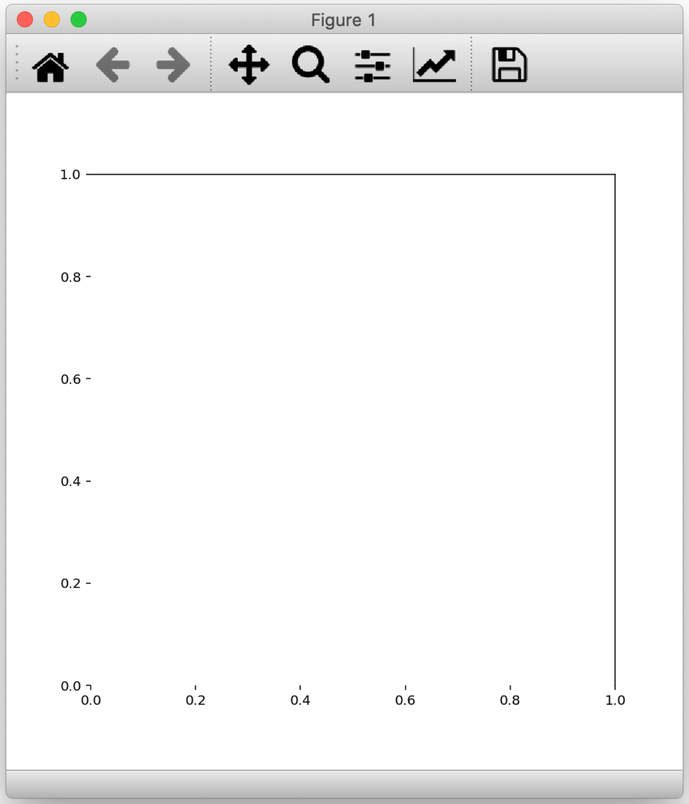
2. spine.set_visible(with Tick and Ticklabels)

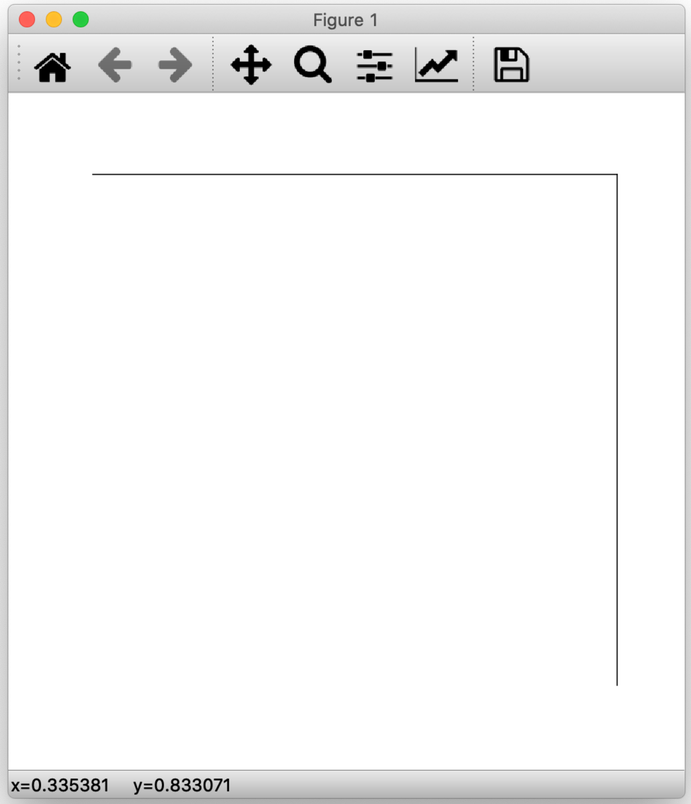
```
fig, ax = plt.subplots(figsize=(7, 7))

ax.spines['left'].set_visible(False)
ax.spines['bottom'].set_visible(False)
```

```
ax.tick_params(left=False, labelleft=False,
               bottom=False, labelbottom=False)
```

Using Tricks(Not Recommended)

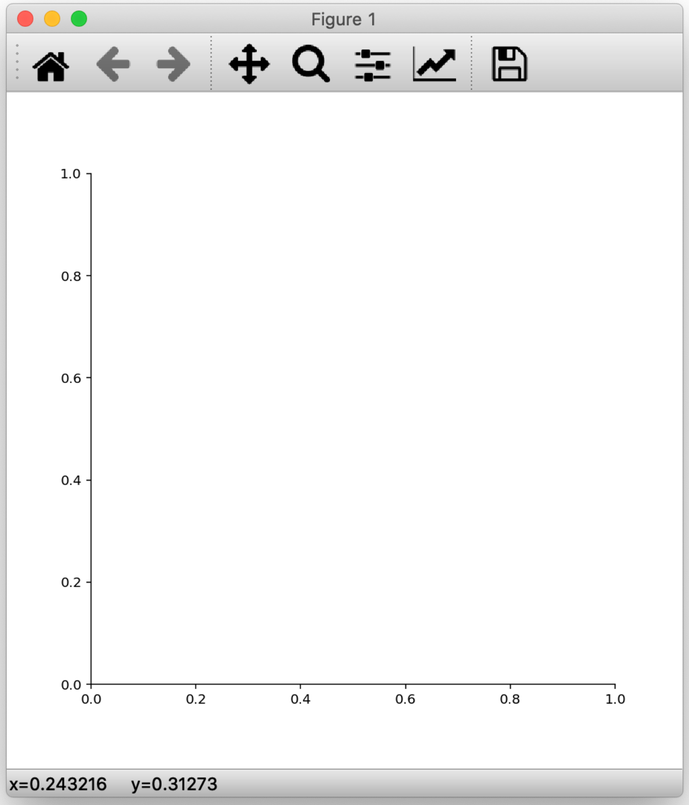
```
ax.tick_params(length=0,
               labelsize=0)
```



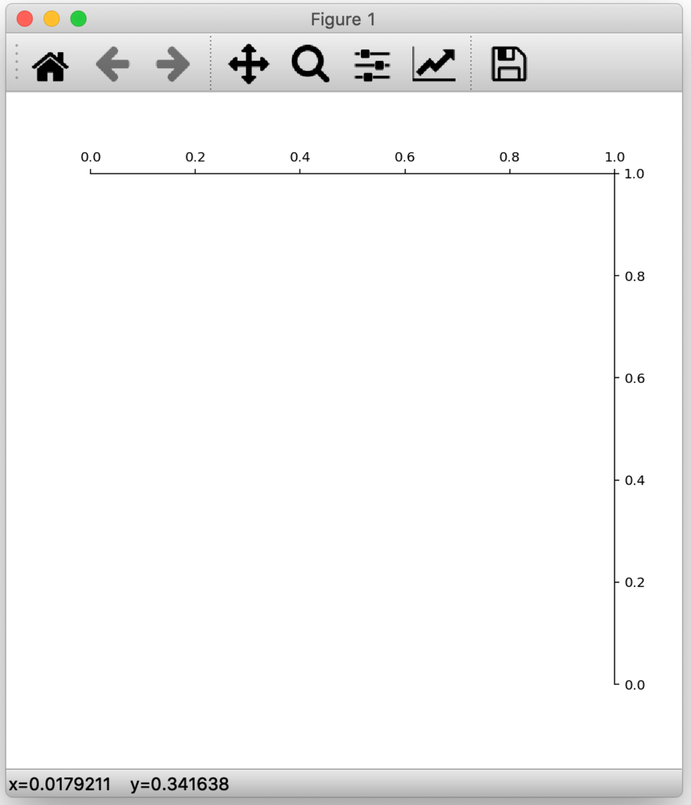


2. spine.set_visible(Upper Right Axis)

```
fig, ax = plt.subplots(figsize=(7, 7))

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)
```



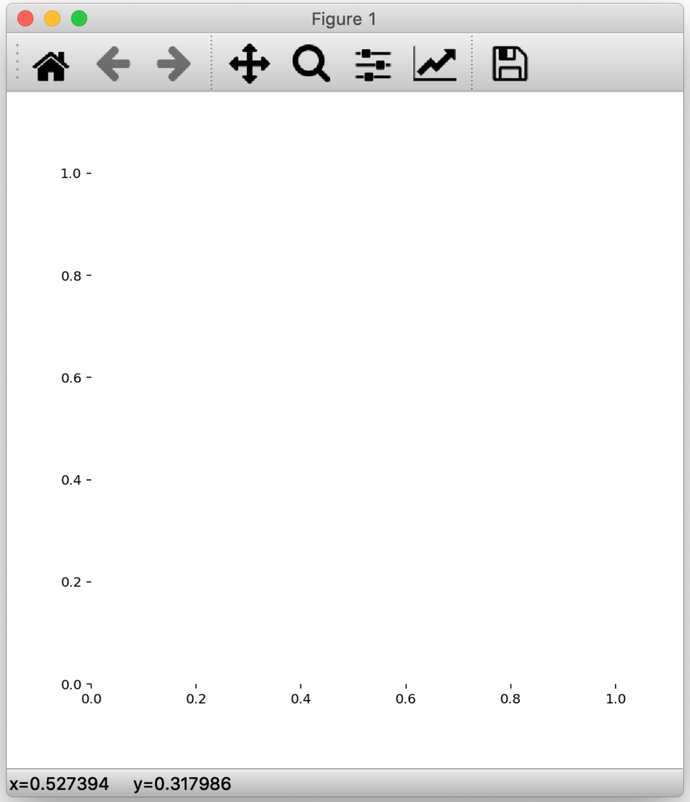

```
for spine_loc, spine in ax.spines.items():
    if spine_loc in ['left', 'bottom']:
        spine.set_visible(False)

ax.tick_params(left=False, labelleft=False,
               right=True, labelright=True,
               bottom=False, labelbottom=False,
               top=True, labeltop=True)
```

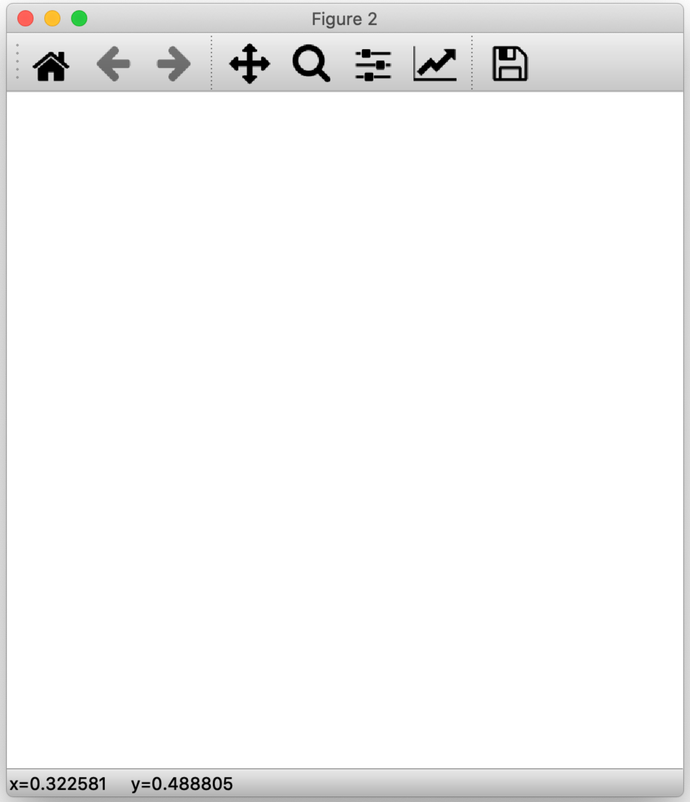


2. spine.set_visible(Entire Spines)

```
fig, ax = plt.subplots(figsize=(7, 7))

for spine_loc, spine in ax.spines.items():
    spine.set_visible(False)
```

```
ax.tick_params(left=False, labelleft=False,
               bottom=False, labelbottom=False)
```

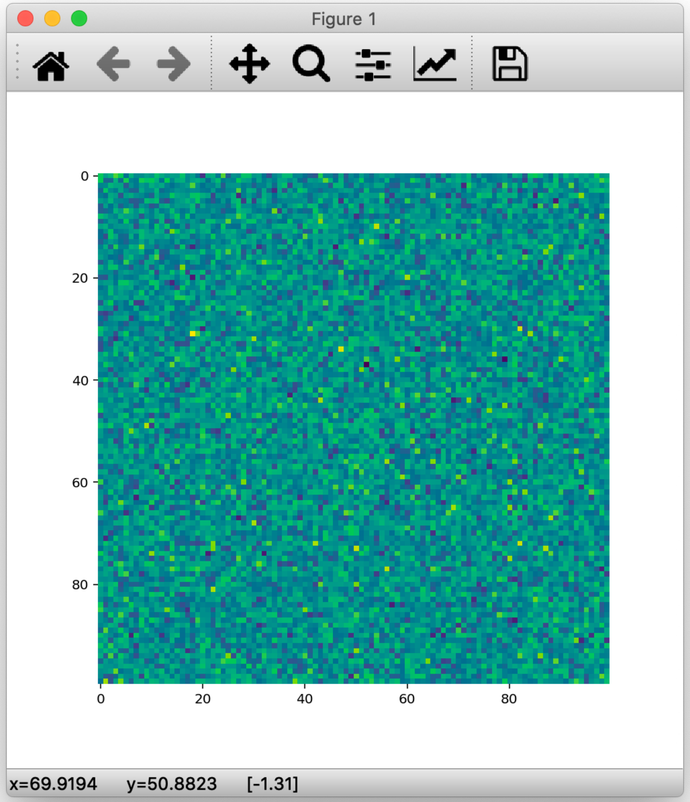


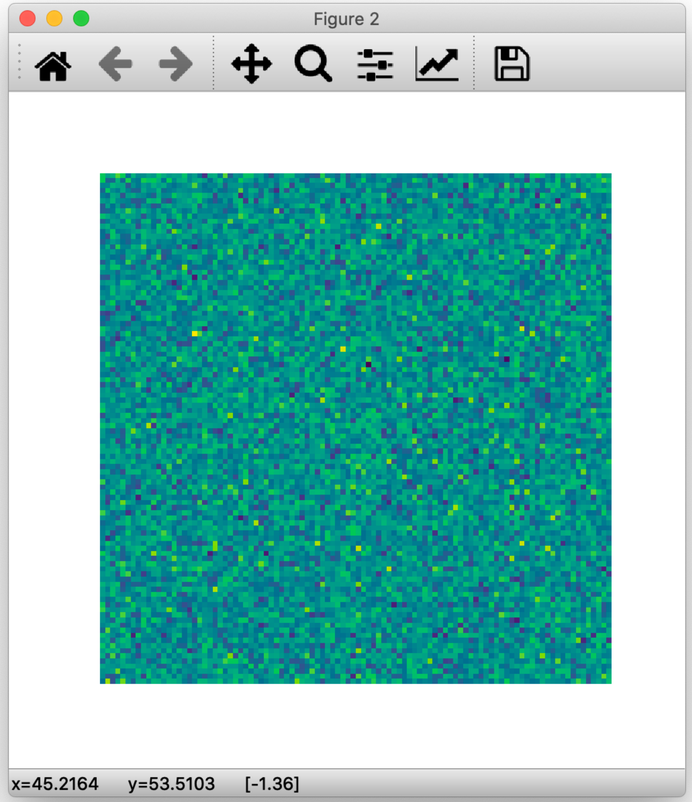


2. spine.set_visible(Entire Spines)

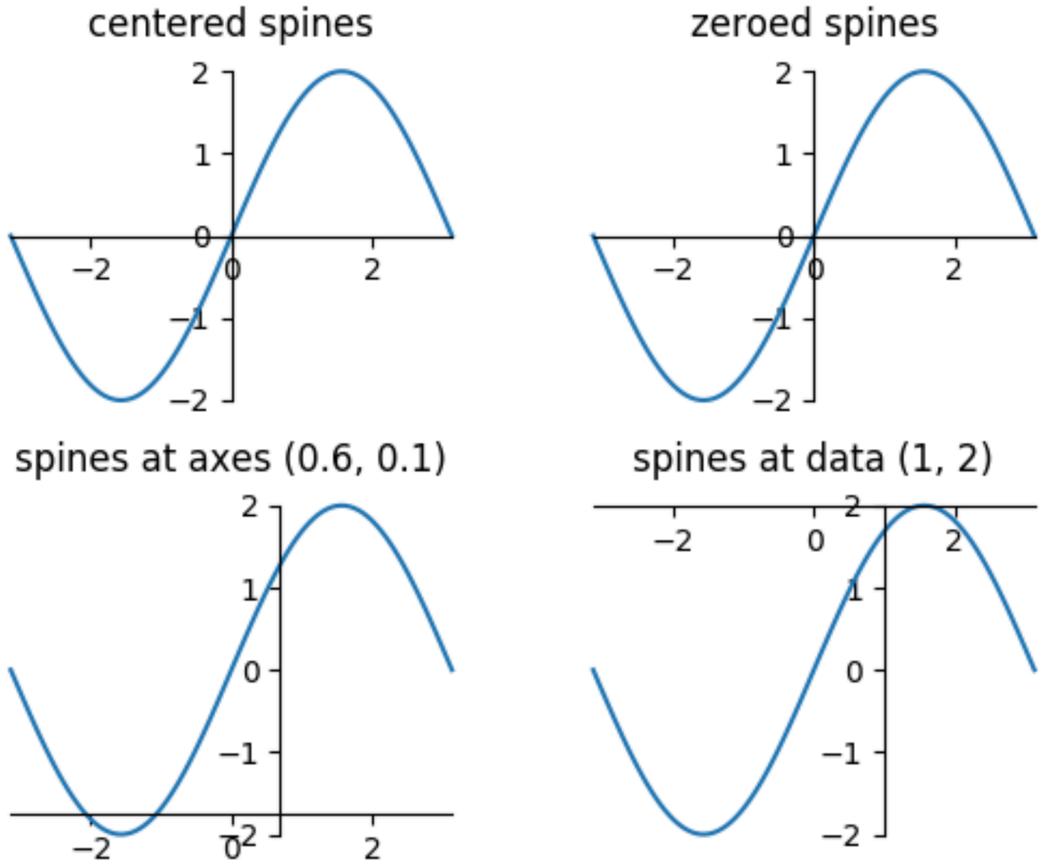
```
import numpy as np
np.random.seed(0)

noise_img = np.random.normal(0, 1, (100, 100))

fig, ax = plt.subplots(figsize=(7, 7))
ax.imshow(noise_img)

for spine_loc, spine in ax.spines.items():
    spine.set_visible(False)
```

```
ax.tick_params(left=False, labelleft=False,
               bottom=False, labelbottom=False)
```





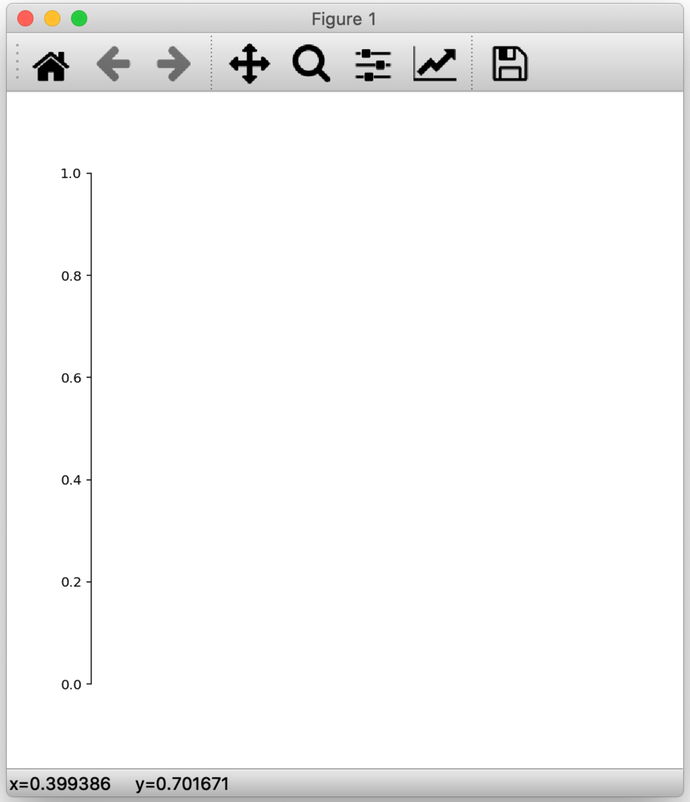
3. spine.set_position

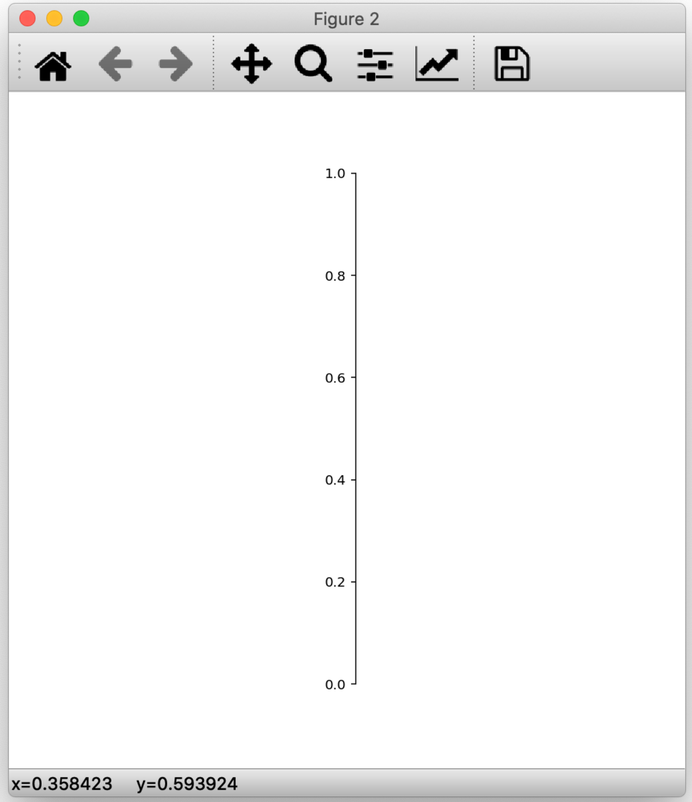


https://matplotlib.org/3.1.1/gallery/ticks_and_spines/spine_placement_demo.html

3. spine.set_position(Central Location)

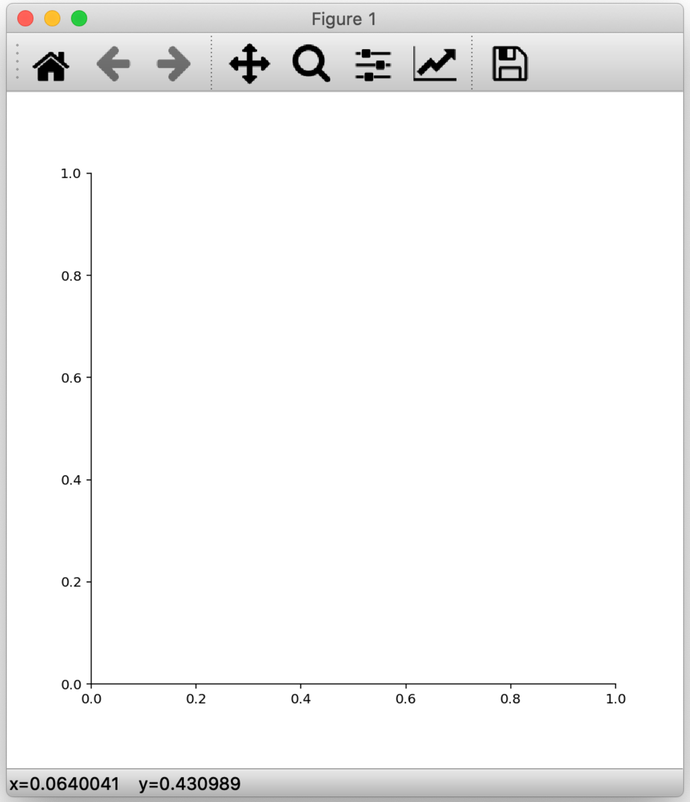
```
import matplotlib.pyplot as plt
import numpy as np

fig, ax = plt.subplots(figsize=(7, 7))

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['bottom', 'right', 'top']:
        spine.set_visible(False)

ax.tick_params(bottom=False, labelbottom=False)

-------------------------------------------------------------
ax.spines['left'].set_position('center')
```

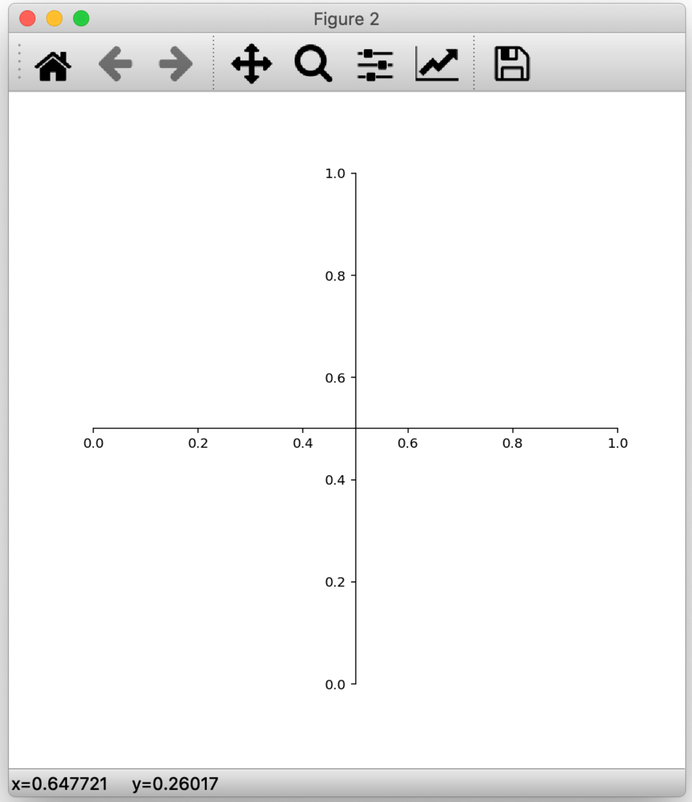




3. spine.set_position(Central Location)

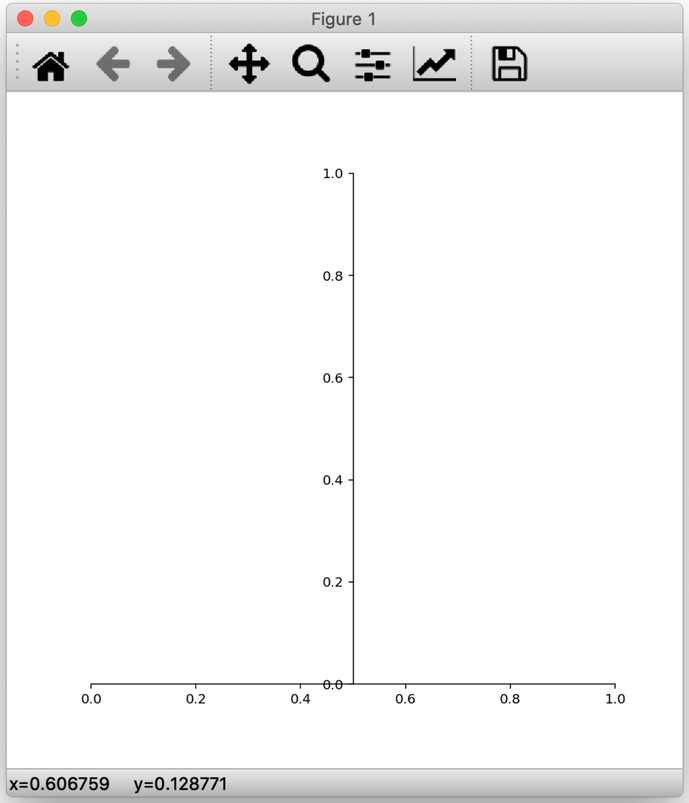
```
fig, ax = plt.subplots(figsize=(7, 7))

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)

    ----------------------------------------
    if spine_loc in ['left', 'bottom']:
        spine.set_position('center')
```



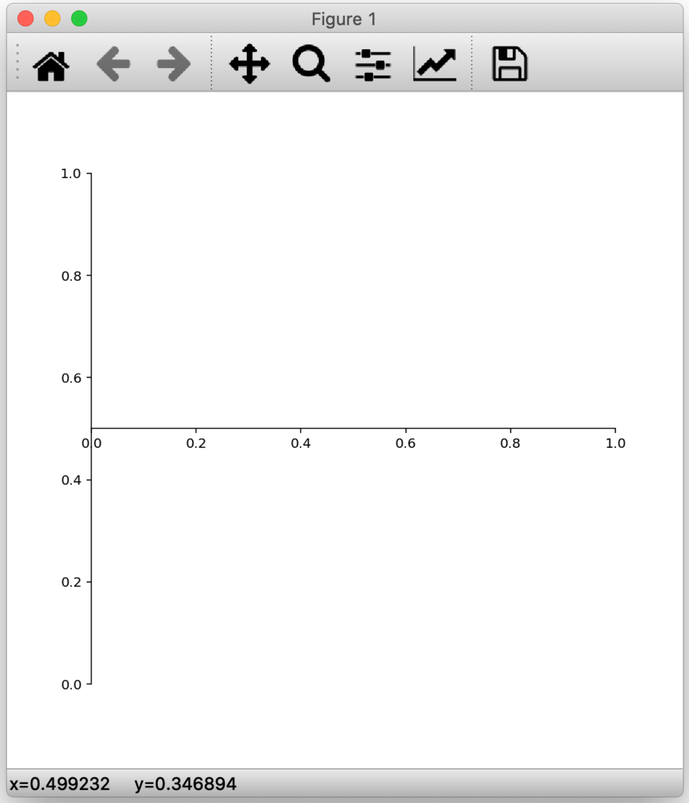


3. spine.set_position(Variations)

```
for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)

    if spine_loc in ['left']:
        spine.set_position('center')
```

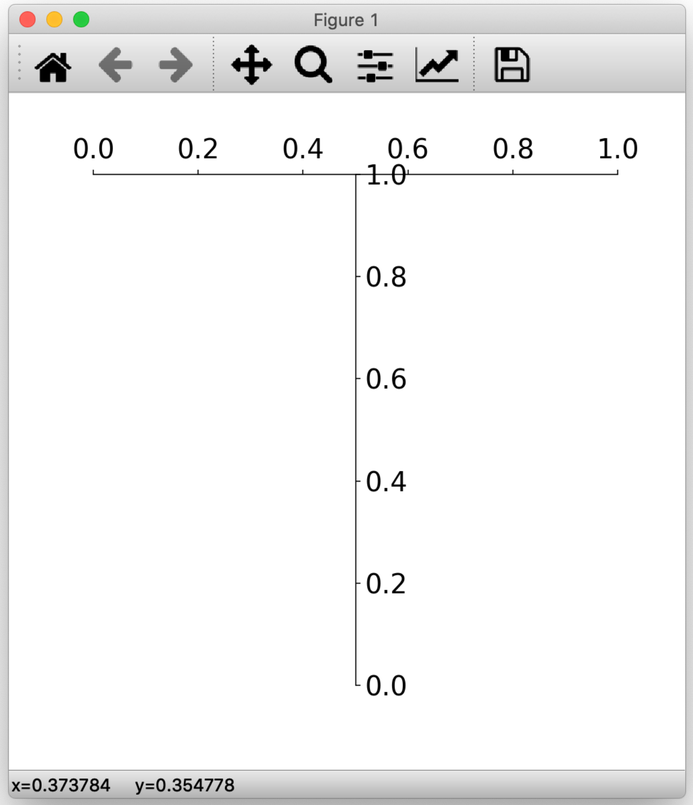



```
for spine_loc, spine in ax.spines.items():
    if spine_k in ['right', 'top']:
        spine.set_visible(False)

    if spine_loc in ['bottom']:
        spine.set_position('center')
```



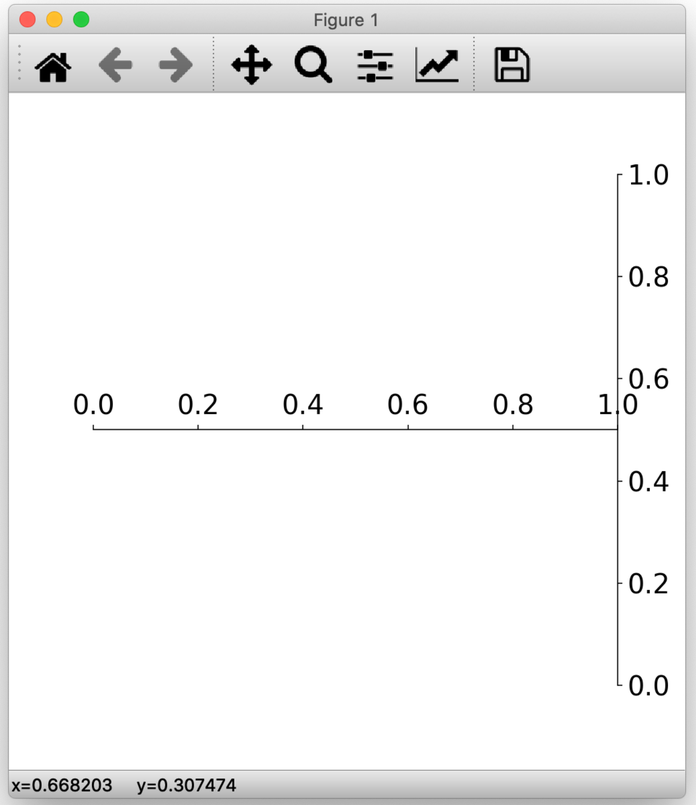
3. spine.set_position(Variations)

```
for spine_loc, spine in ax.spines.items():
    if spine_loc in ['left', 'bottom']:
        spine.set_visible(False)

    if spine_loc in ['right']:
        spine.set_position('center')

ax.tick_params(labelsize=20,
               bottom=False, labelbottom=False,
               left=False, labelleft=False,
               right=True, labelright=True,
               top=True, labeltop=True)
```



3. spine.set_position(Variations)

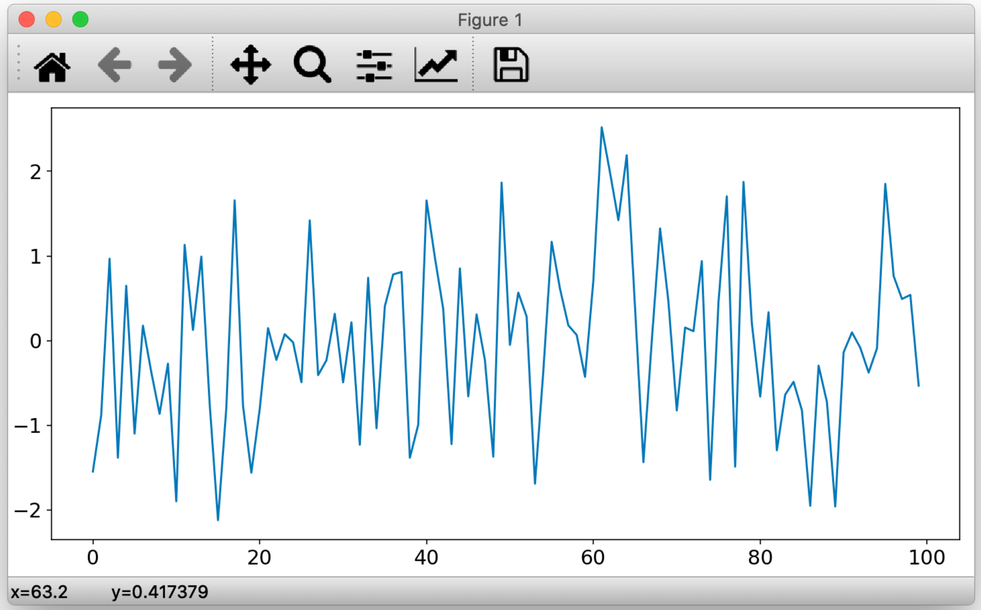
```
for spine_loc, spine in ax.spines.items():
    if spine_loc in ['left', 'bottom']:
        spine.set_visible(False)

    if spine_loc in [‘top']:
        spine.set_position('center')

ax.tick_params(labelsize=20,
               bottom=False, labelbottom=False,
               left=False, labelleft=False,
               right=True, labelright=True,
               top=True, labeltop=True)
```

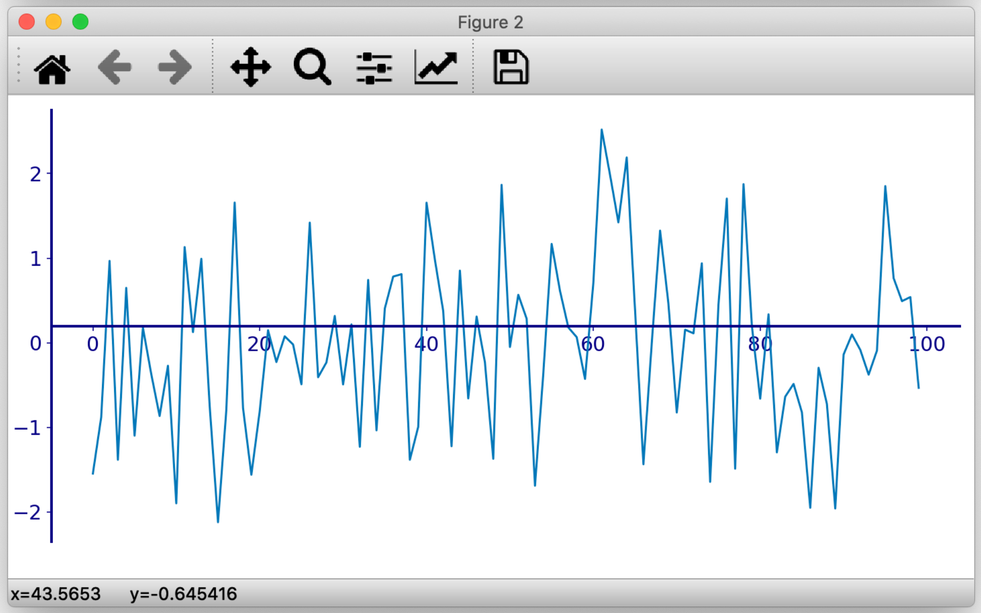


3. spine.set_position(Example1)

```
noise_data = np.random.normal(0, 1, (100,))

fig, ax = plt.subplots(figsize=(10,5))
ax.plot(noise_data)

fig.tight_layout()
ax.tick_params(labelsize=15)
```

```
for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)

    if spine_loc in ['bottom']:
        spine.set_position('center')

    if spine_loc in ['left', 'bottom']:
        spine.set_color('navy')
        spine.set_linewidth(2)
```

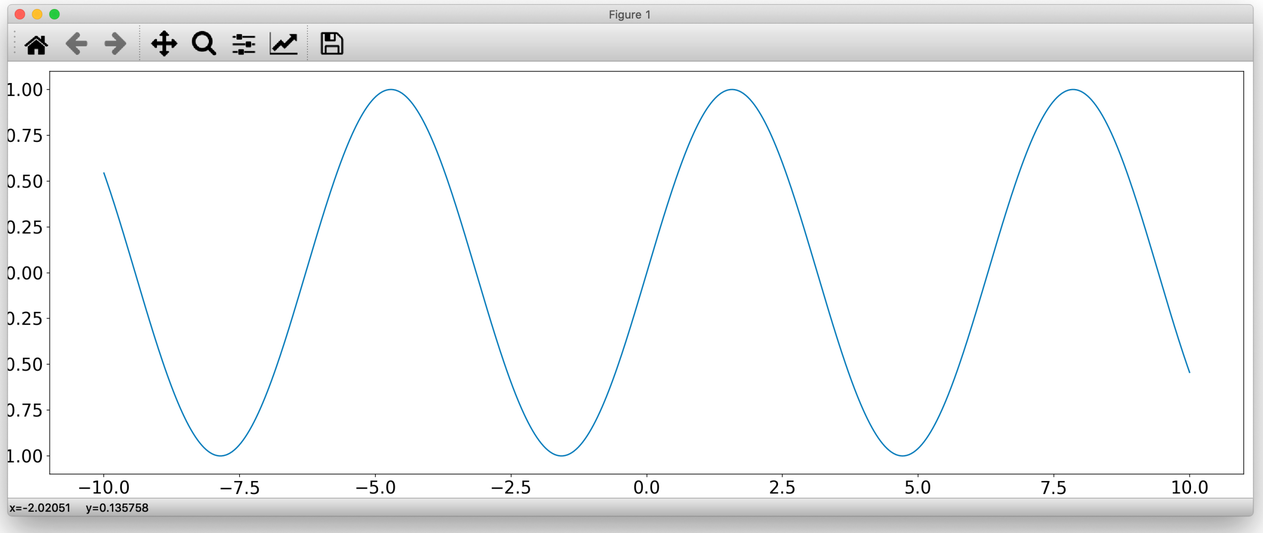


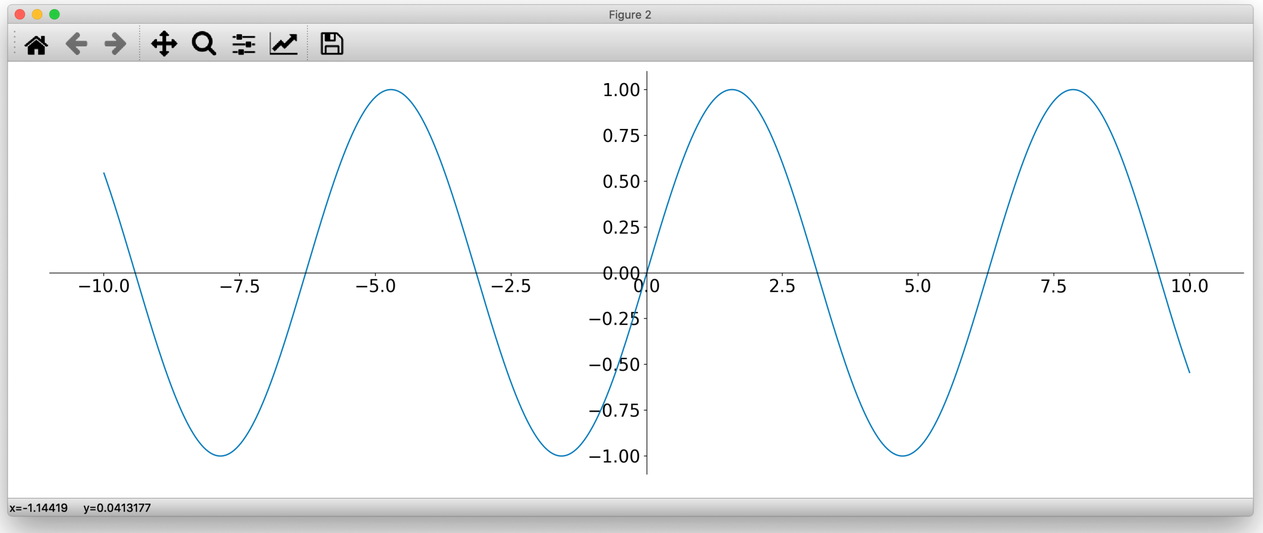


3. spine.set_position(Example2)

```
t = np.linspace(-10, 10, 500)
sin = np.sin(t)

fig, ax = plt.subplots(figsize=(20, 7))
ax.plot(t, sin)

fig.tight_layout()
ax.tick_params(labelsize=20)
```

```
for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)

    if spine_loc in ['bottom', 'left']:
        spine.set_position('center')
```





4. spine.set_position(Tuple Argument)

`set_position(self, position)` [source]

Set the position of the spine.

Spine position is specified by a 2 tuple of (position type, amount). The position types are:

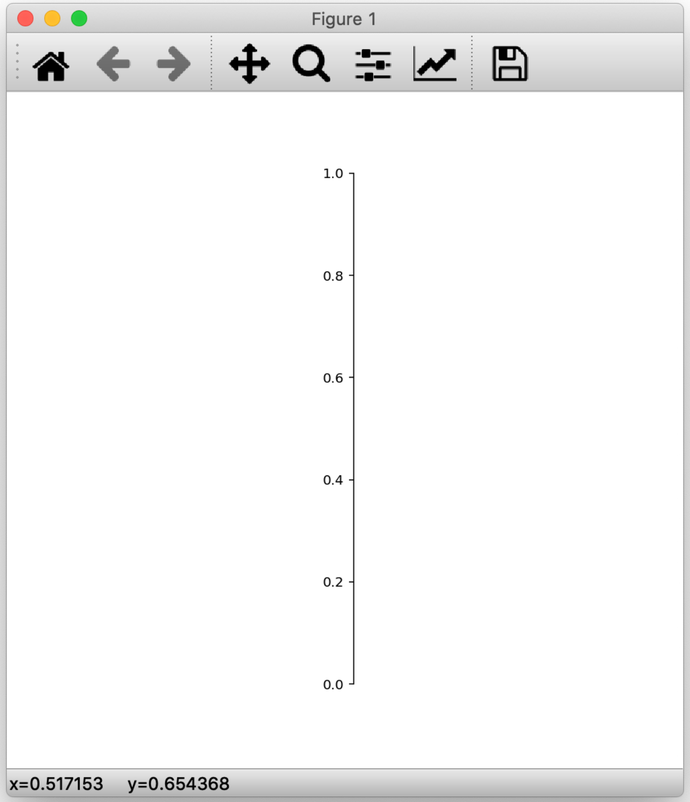
- 'outward': place the spine out from the data area by the specified number of points. (Negative values place the spine inwards.)
- 'axes': place the spine at the specified Axes coordinate (0 to 1).
- 'data': place the spine at the specified data coordinate.

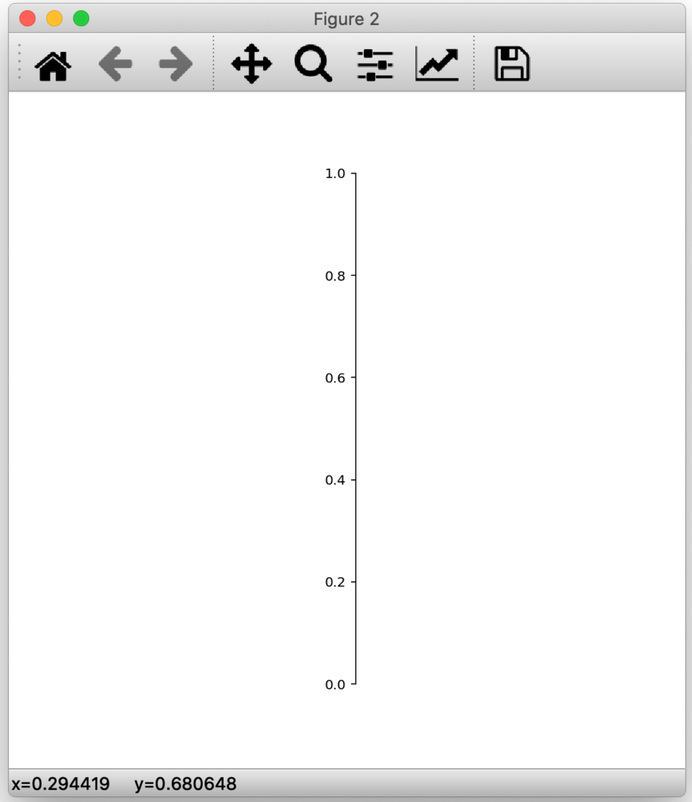
Additionally, shorthand notations define a special positions:

- 'center' -> ('axes', 0.5)
- 'zero' -> ('data', 0.0)

4. spine.set_position(Central Location with Tuple Argument)

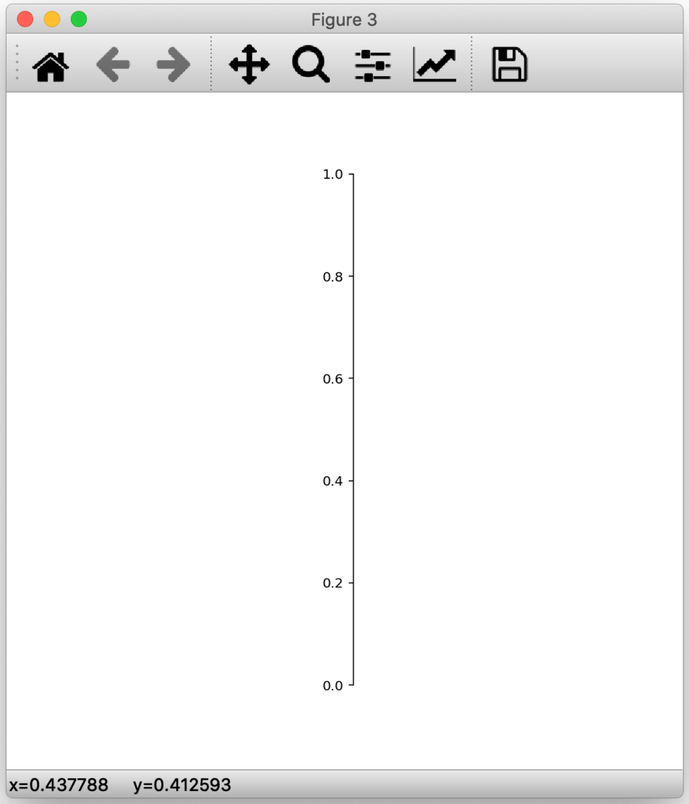
```
fig, ax = plt.subplots(figsize=(7, 7))

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['bottom', 'right', 'top']:
        spine.set_visible(False)

ax.tick_params(bottom=False, labelbottom=False)

ax.spines['left'].set_position('center')

----------------------------------------------------------
ax.spines['left'].set_position(('axes', 0.5))
```



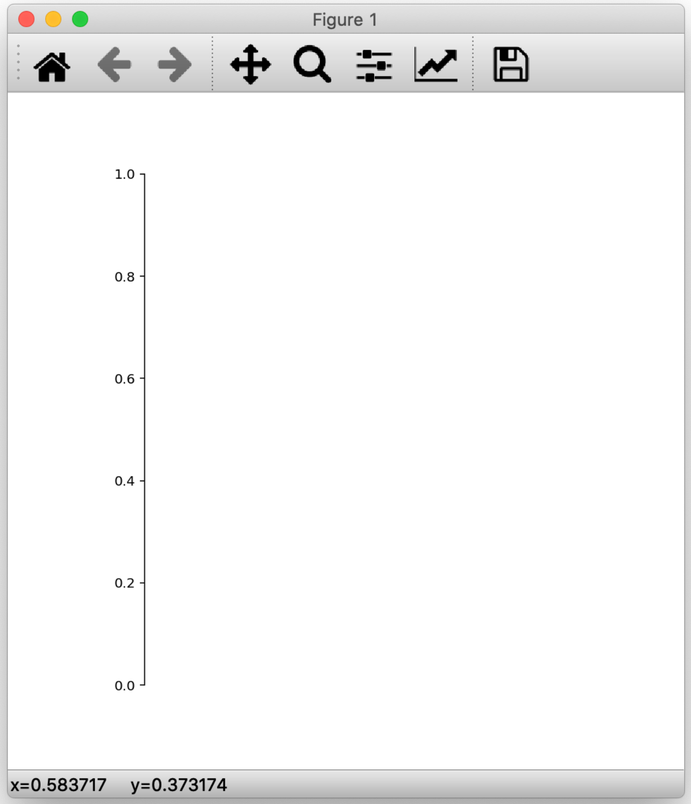


4. spine.set_position(Arbitrary Locations)

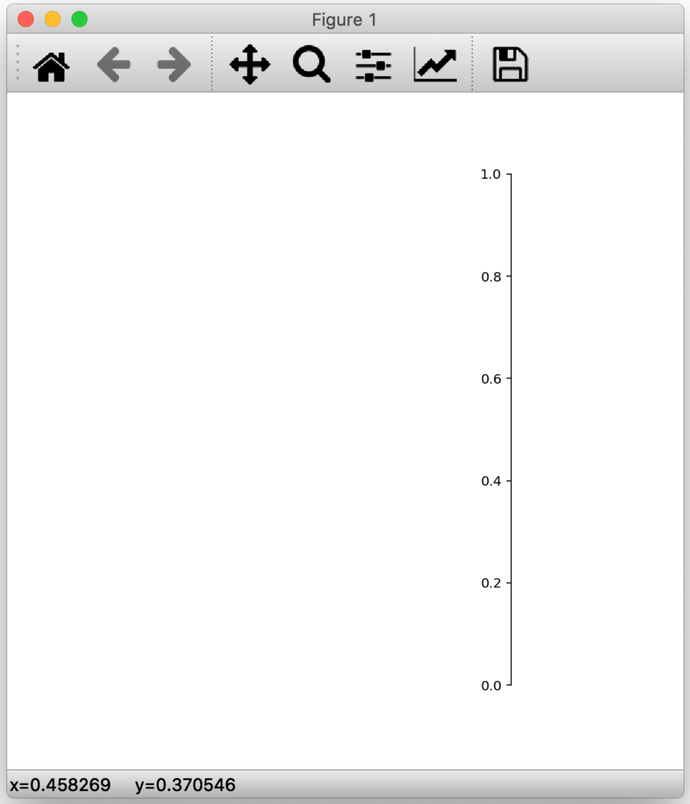
```
ax.spines['left'].set_position(('axes', 0.5))
```




```
ax.spines['left'].set_position(('axes', 0.1))
```

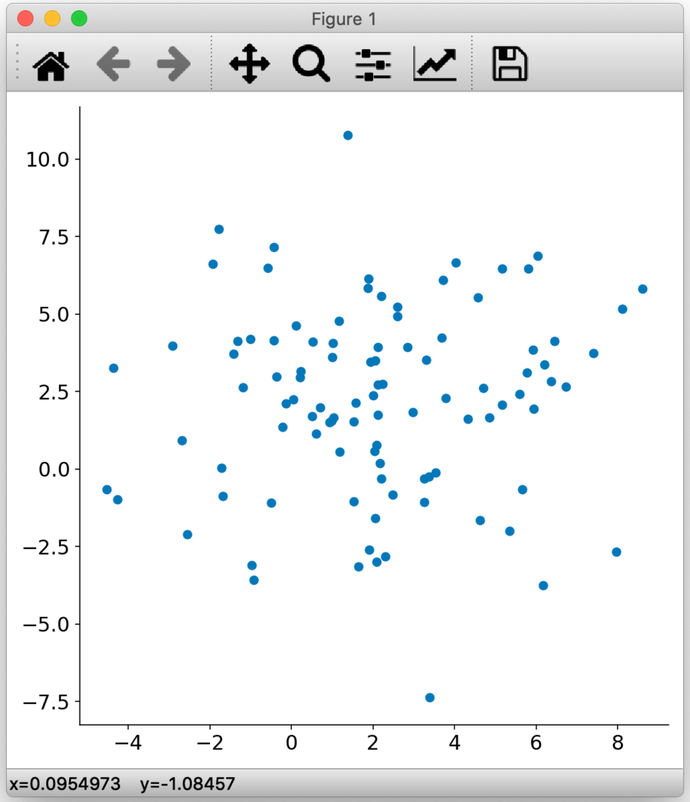


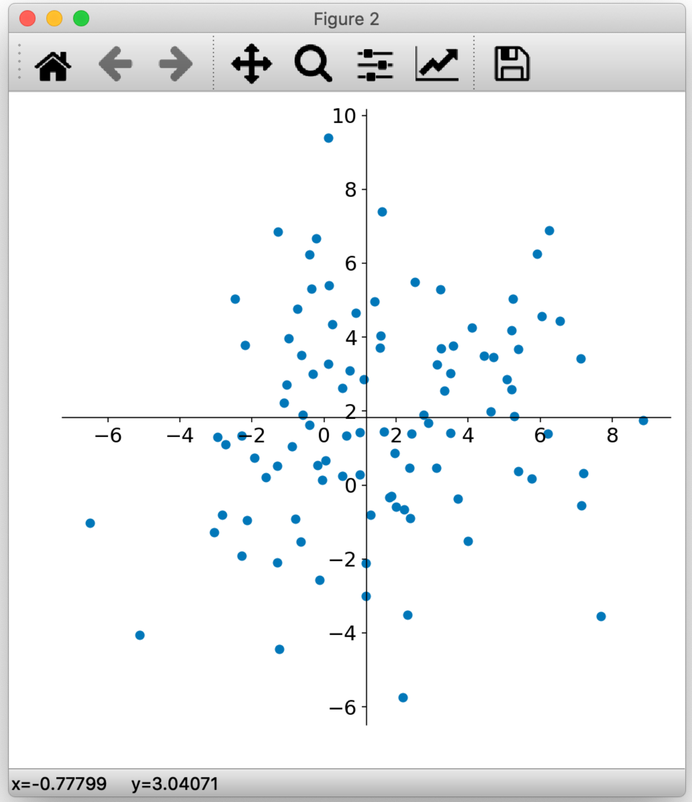

```
ax.spines['left'].set_position(('axes', 0.8))
```



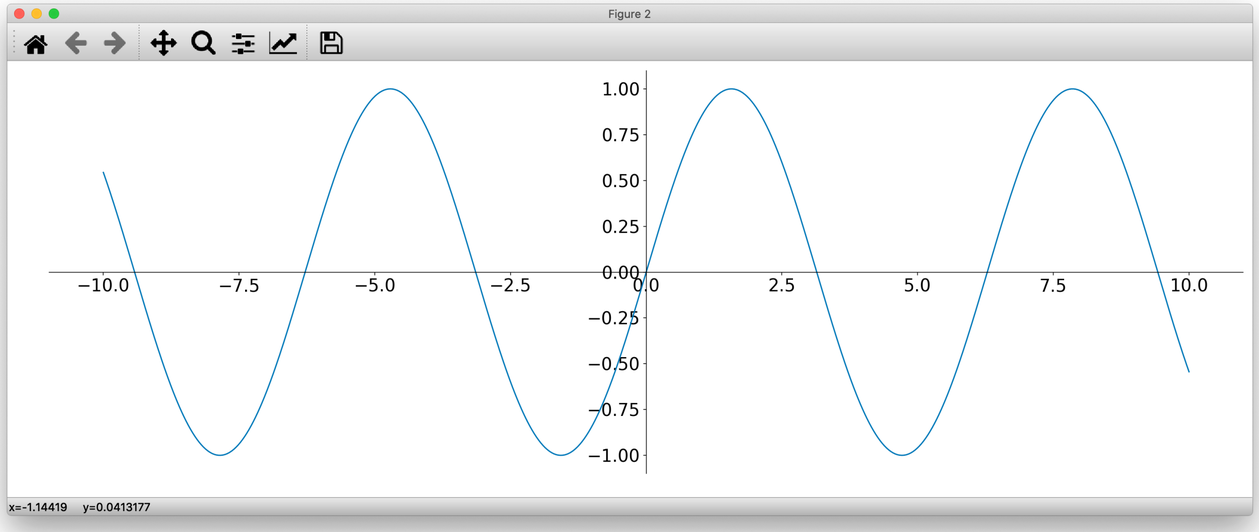
4. spine.set_position(Limitation)

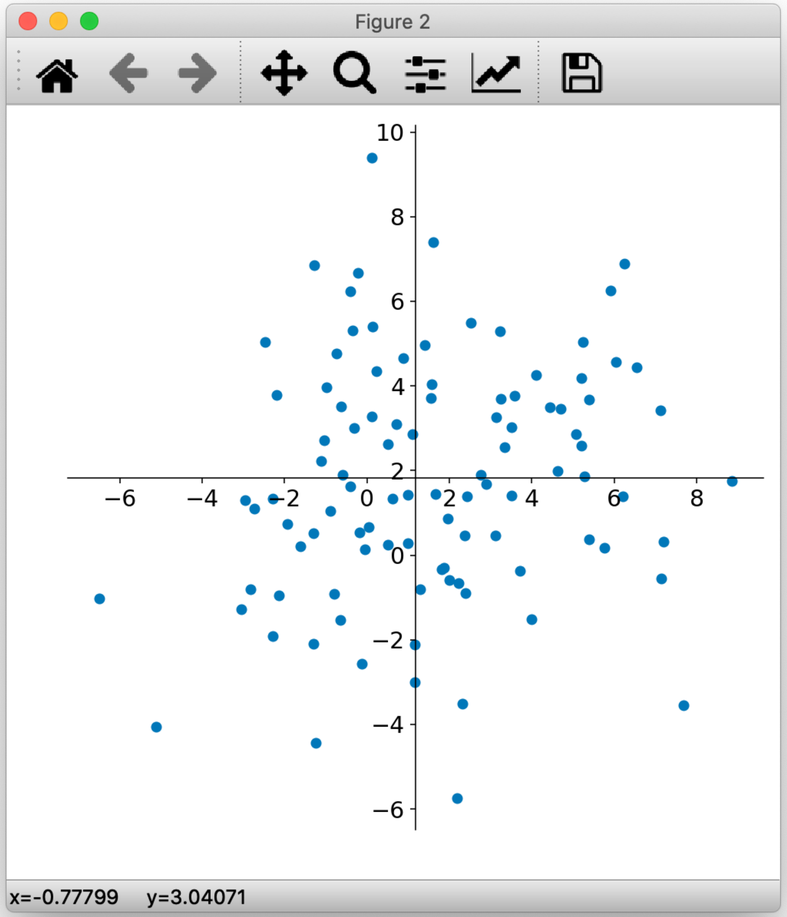
```
x_data = np.random.normal(2, 3, (100,))
y_data = np.random.normal(2, 3, (100,))

fig, ax = plt.subplots(figsize=(7, 7))
ax.scatter(x_data, y_data)

ax.tick_params(labelsize=15)
fig.tight_layout()

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)

    ----------------------------------------
    if spine_loc in ['left', 'bottom']:
        spine.set_position('center')
```





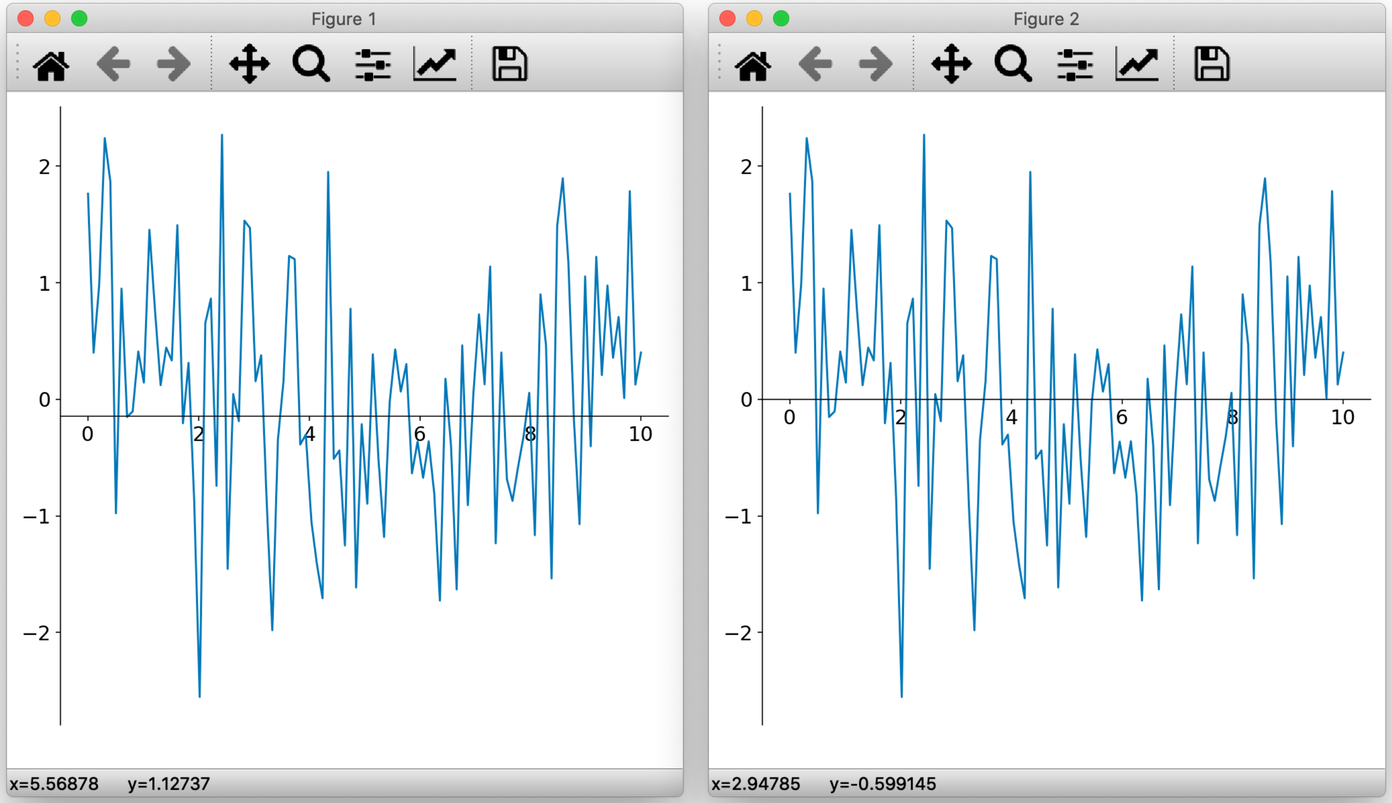
4. spine.set_position(Limitation)





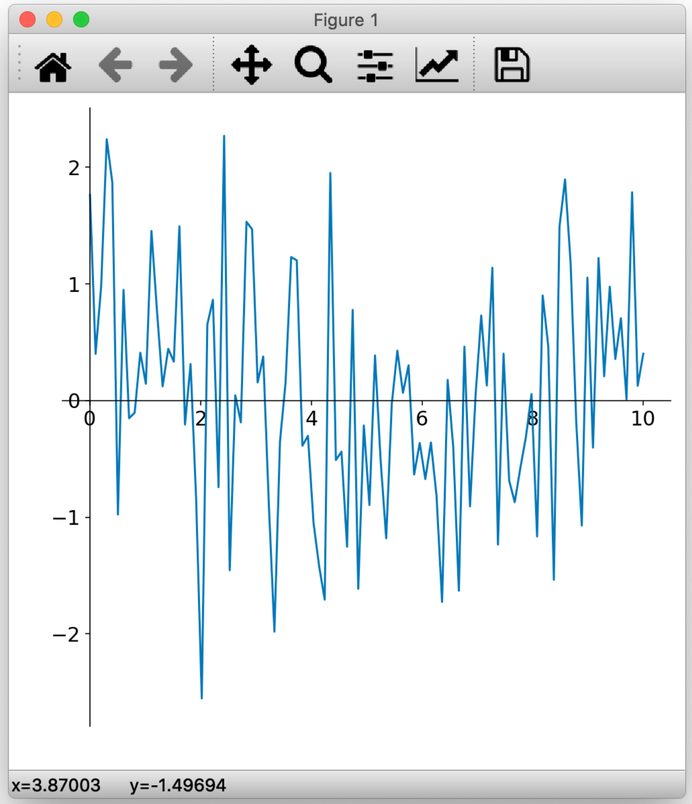
4. spine.set_position

```
np.random.seed(0)

t_min, t_max = 0, 10
n_data = 100

t = np.linspace(t_min, t_max, n_data)
noise_data = np.random.normal(0, 1, size=(n_data,))

fig, ax = plt.subplots(figsize=(7, 7))
ax.plot(t, noise_data)

ax.tick_params(labelsize=15)
fig.tight_layout()

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)

    if spine_loc in ['bottom']:
        spine.set_position('center')
        spine.set_position(('axes', 0.5))

        ------------------------------------
        spine.set_position(('data', 0))
```



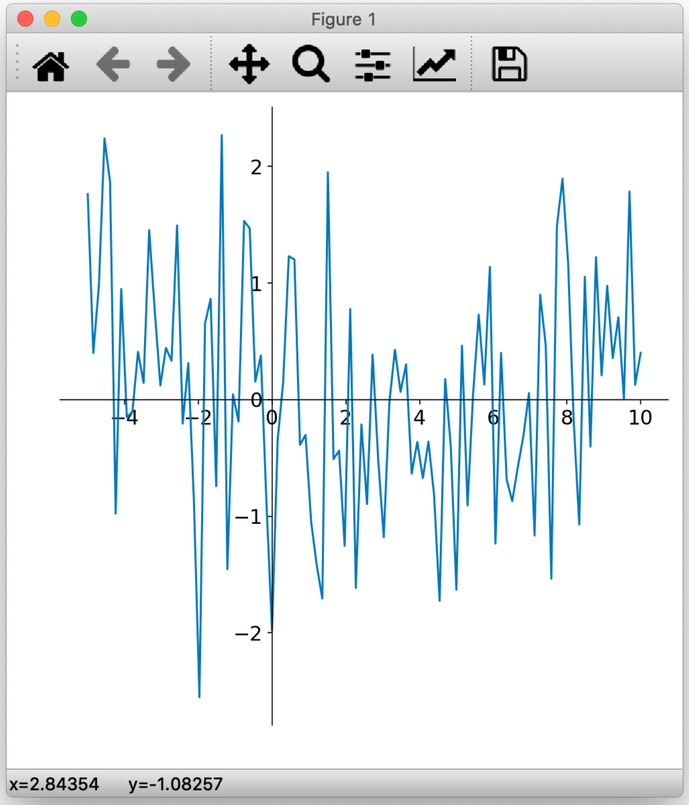
4. spine.set_position

```
np.random.seed(0)

t_min, t_max = 0, 10
n_data = 100

t = np.linspace(t_min, t_max, n_data)
noise_data = np.random.normal(0, 1, size=(n_data,))

fig, ax = plt.subplots(figsize=(7, 7))
ax.plot(t, noise_data)

ax.tick_params(labelsize=15)
fig.tight_layout()

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)

    if spine_loc in ['bottom', 'left']:
        spine.set_position(('data', 0))
```

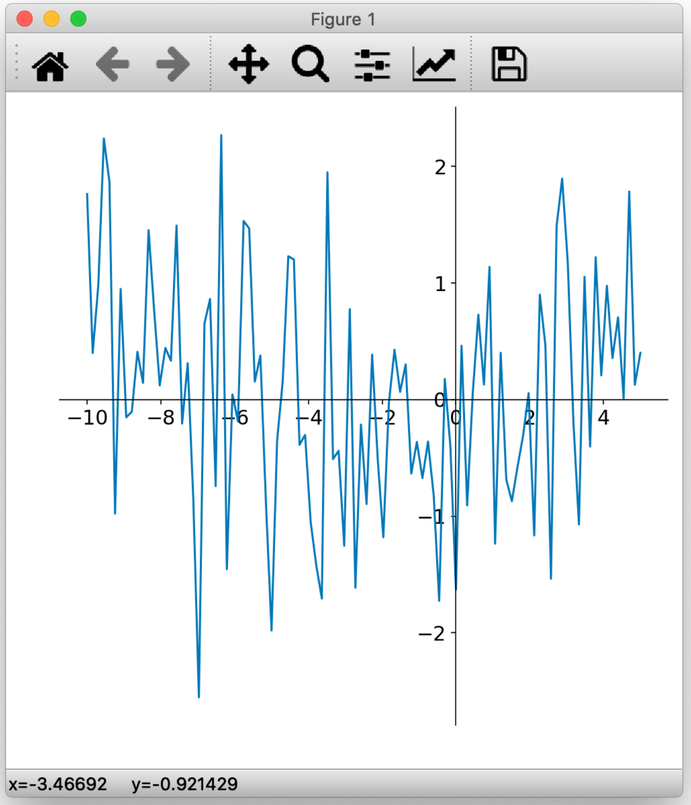


4. spine.set_position

```
t_min, t_max = -5, 10
n_data = 100

t = np.linspace(t_min, t_max, n_data)
noise_data = np.random.normal(0, 1, size=(n_data,))
```

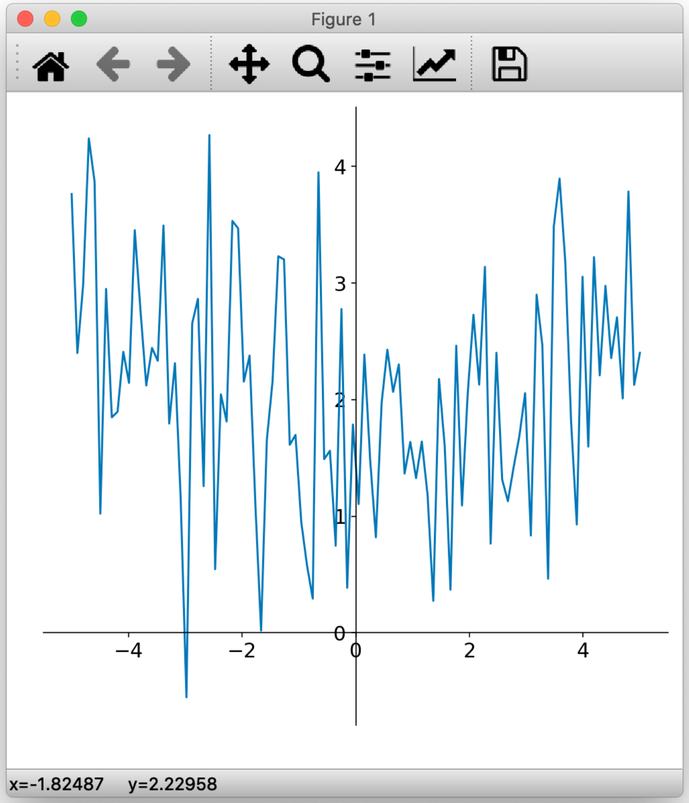



```
t_min, t_max = -10, 5
n_data = 100

t = np.linspace(t_min, t_max, n_data)
noise_data = np.random.normal(0, 1, size=(n_data,))
```

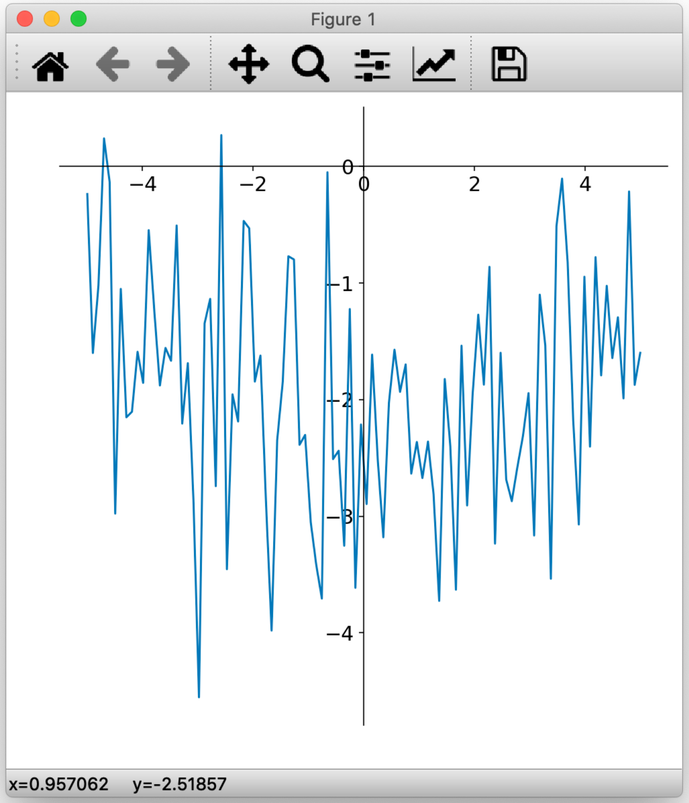


4. spine.set_position

```
t_min, t_max = -5, 5
n_data = 100

t = np.linspace(t_min, t_max, n_data)
noise_data = np.random.normal(0, 1, size=(n_data,)) + 2
```

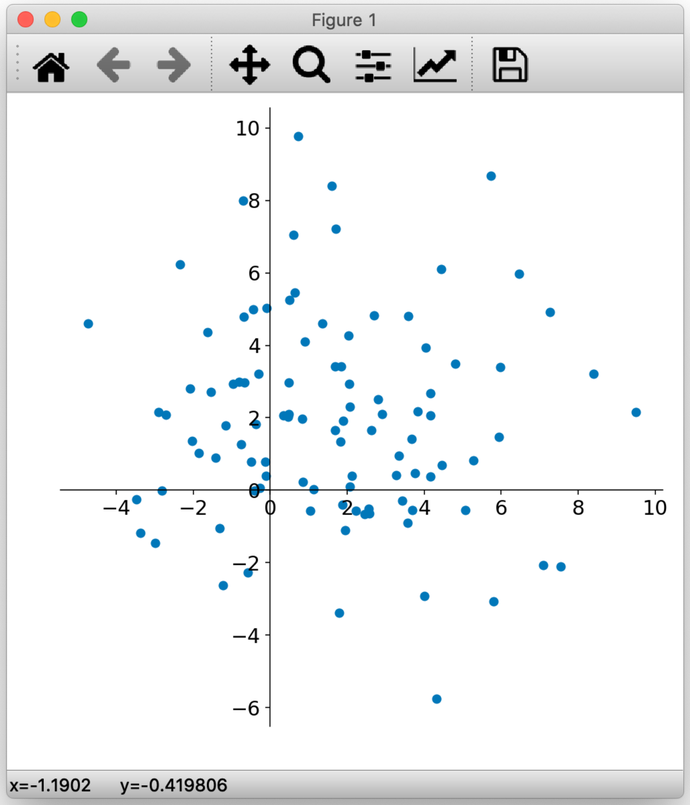



```
t_min, t_max = -5, 5
n_data = 100

t = np.linspace(t_min, t_max, n_data)
noise_data = np.random.normal(0, 1, size=(n_data,)) - 2
```



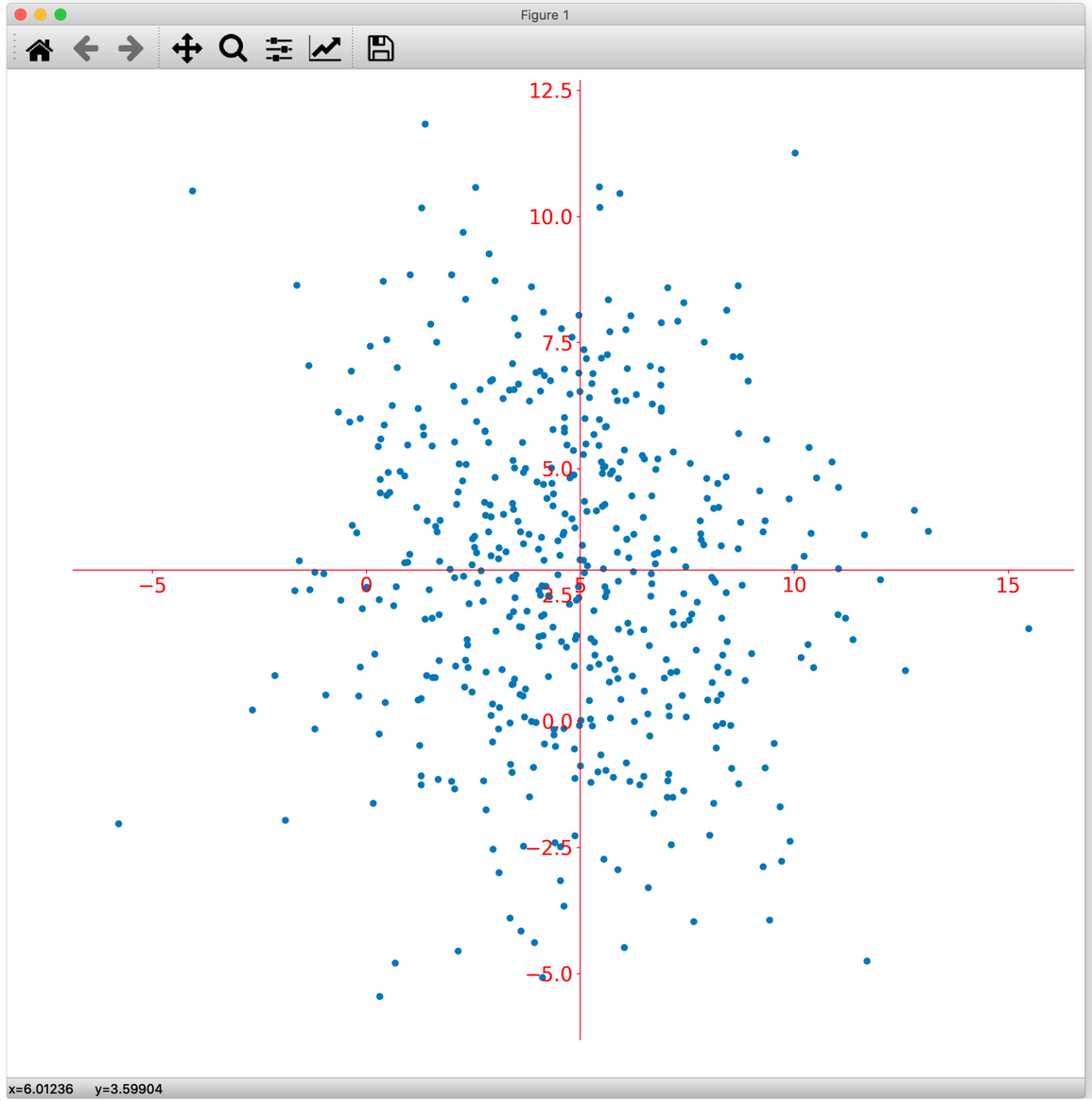
4. spine.set_position

```
x_data = np.random.normal(2, 3, (100,))
y_data = np.random.normal(2, 3, (100,))

fig, ax = plt.subplots(figsize=(7, 7))
ax.scatter(x_data, y_data)

ax.tick_params(labelsize=15)
fig.tight_layout()

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)

    if spine_loc in ['left', 'bottom']:
        spine.set_position(('data', 0))
```



4. spine.set_position

```
x_data = np.random.normal(5, 3, (500,))
y_data = np.random.normal(3, 3, (500,))

fig, ax = plt.subplots(figsize=(15, 15))
ax.scatter(x_data, y_data)

ax.tick_params(labelsize=20,
               colors='red')
fig.tight_layout()

for spine_loc, spine in ax.spines.items():
    if spine_loc in ['right', 'top']:
        spine.set_visible(False)

    if spine_loc in ['left']:
        spine.set_position(('data', 5))
        spine.set_color('red')

    if spine_loc in ['bottom']:
        spine.set_position(('data', 3))
        spine.set_color('red')
```



Python for Data Visualization

-Chapter.1 Matplotlib Anatomy -

1-06. Spines

1. Spines
2. spine.set_visible
3. spine.set_position(Basic Usage)
4. spine.set_position(Tuple Argument)