How can I set the aspect ratio in Matplotlib?

Authored by

stats writer

April 22, 2024

The process of setting the aspect ratio in Matplotlib involves adjusting the dimensions of the plot to maintain a specific ratio between the width and height. This can be achieved by using the "set_aspect" function in the Matplotlib library, which allows for the customization of the aspect ratio based on either a fixed value or a specific data coordinate. This feature is useful for creating visually appealing and accurate plots in Matplotlib.

Set the Aspect Ratio in Matplotlib

The aspect ratio of a matplotlib plot refers to the aspect of the axis scaling, i.e. the ratio of y-unit to x-unit.

This ratio can be modified by using the matplotlib.axes.Axes.set_aspect() function.

Under the hood, the set_aspect() function actually modifies something known as the data coordinate system but in practice we typically want to modify the display coordinate system.

To make this conversion easy, we can use this bit of code:

```
#define y-unit to x-unit ratio
ratio = 1.0

#get x and y limits
x_left, x_right = ax.get_xlim()
```

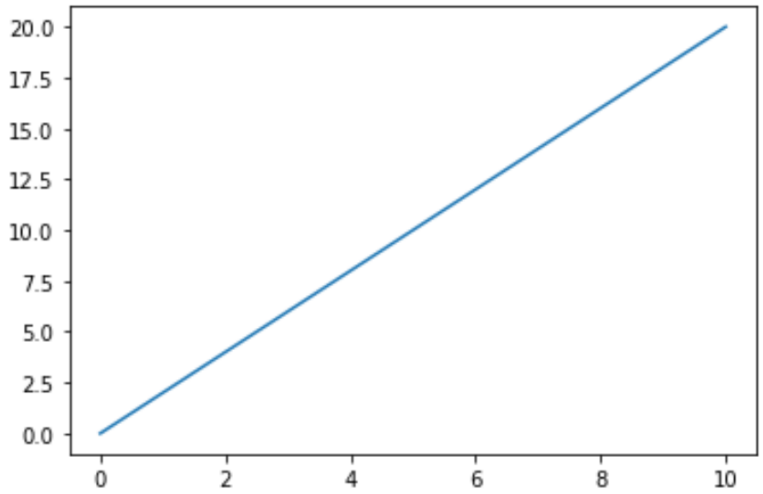
```
y_low, y_high = ax.get_ylim()

#set aspect ratio
ax.set_aspect(abs((x_right-x_left)/(y_low-y_high))*ratio)
```

Let's walk through an example of using this function in practice.

Step 1: Create a Basic Matplotlib Plot

First, let's create a simple line chart using Matplotlib:

```
import matplotlib.pyplot as plt

#define matplotlib figure and axis
fig, ax = plt.subplots()

#create simple line plot
ax.plot(,)

#display plot
plt.show()
```

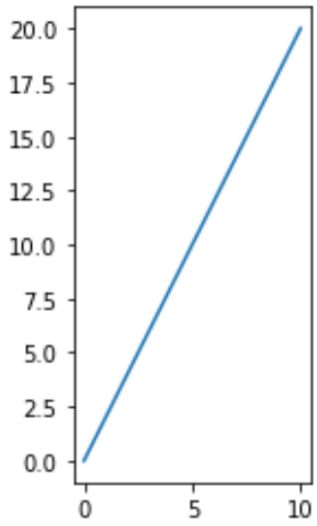


Step 2: Set the Aspect Ratio (The Wrong Way)

Notice that the x-axis is longer than the y-axis. Let's attempt to set the aspect ratio to 1, i.e. the x-axis and y-axis should be equal:

```
import matplotlib.pyplot as plt

#define matplotlib figure and axis
fig, ax = plt.subplots()

#create simple line plot
ax.plot(,)

#attempt to set aspect ratio to 1
ax.set_aspect(1)

#display plot
```

```
plt.show()
```



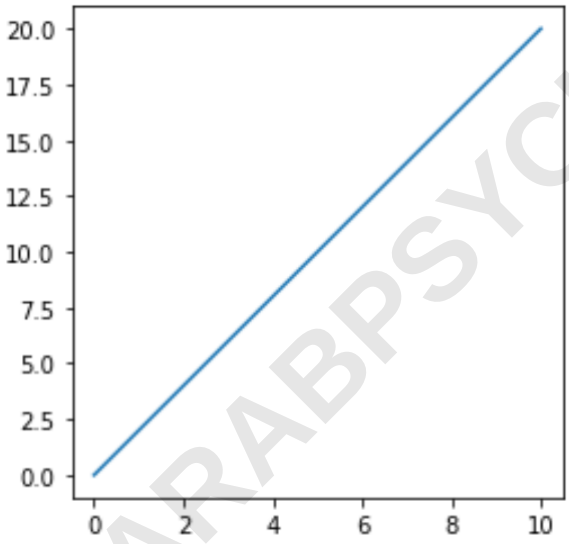
Notice that this did not work as we expected. The y-axis is much longer than the x-axis.

Step 3: Set the Aspect Ratio (The Right Way)

The following code shows how to use a simple calculation to set the correct aspect ratio:

```
import matplotlib.pyplot as plt

#define matplotlib figure and axis
fig, ax = plt.subplots()

#create simple line plot
```

```
ax.plot(,)

#set aspect ratio to 1
ratio = 1.0
x_left, x_right = ax.get_xlim()
y_low, y_high = ax.get_ylim()
ax.set_aspect(abs((x_right-x_left)/(y_low-y_high))*ratio)

#display plot
plt.show()
```



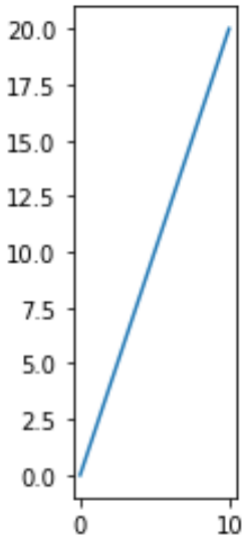
Notice that this plot has the aspect ratio we expected. The x-axis and y-axis are equal lengths.

Step 4: Adjust the Aspect Ratio to Whatever You'd Like

If we'd like the y-axis to be longer than the x-axis, we

can simply specify the aspect ratio to be some number greater than 1:

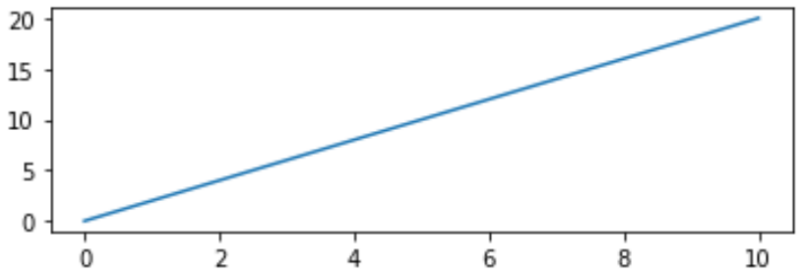
```
import matplotlib.pyplot as plt

#define matplotlib figure and axis
fig, ax = plt.subplots()

#create simple line plot
ax.plot(,)

#set aspect ratio to 3
ratio = 3
x_left, x_right = ax.get_xlim()
y_low, y_high = ax.get_ylim()
ax.set_aspect(abs((x_right-x_left)/(y_low-y_high))*ratio)

#display plot
plt.show()
```



And if we'd like the y-axis to be shorter than the x-axis, we can simply specify the aspect ratio to be some number less than 1:

```
import matplotlib.pyplot as plt

#define matplotlib figure and axis
fig, ax = plt.subplots()

#create simple line plot
ax.plot(,)

#set aspect ratio to .3
ratio = .3
x_left, x_right = ax.get_xlim()
y_low, y_high = ax.get_ylim()
```

```
ax.set_aspect(abs((x_right-x_left)/(y_low-y_high))*ratio)

#display plot
plt.show()
```



You can find more Matplotlib tutorials here.