Team 17

User Guide

Matplotlib: Interactive Legend

April 2020

Table of Contents

Introduction

Welcome to the Matplotlib Interactive Legend User Guide.

Before Using the Interactive Legend, please read the Get Started section for instructions to get started using Matplotlib.

The interactive legend is a new feature which allows you to control what subplots of your graph are visible. For instance, if you have a graph consisting of lines, when you have your graph shown, you will have the ability to hide each line separately. This can be used for comparison of different subplots on the same graph or saving multiple images of the same graph but with different plots visible in each image.

Getting Started

To begin using the features of Matplotlib, you will need to install it to get things running.

For instructions on how to install and set up Matplotlib, please visit the following link for the different installation methods available:
https://matplotlib.org/3.2.1/users/installing.html#installing

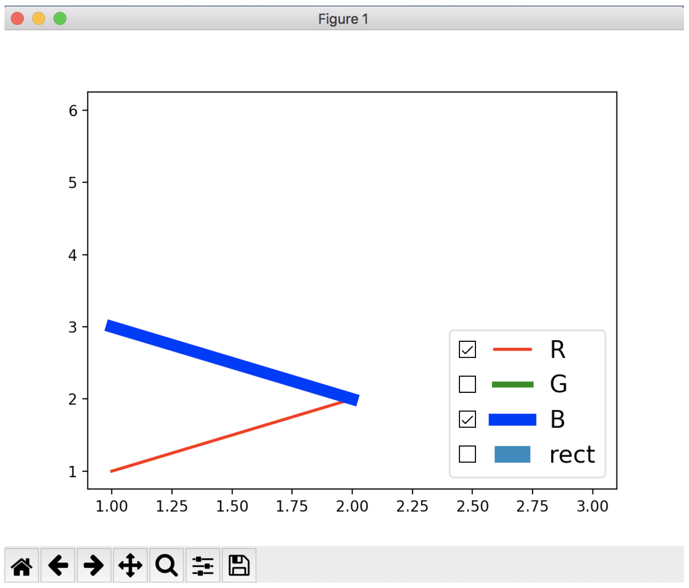
Interactive Legend

After installing Matplotlib onto your machine, you are ready to utilize the interactive legend feature!

Here are the steps to the basic usage of the interactive legend:

1. Download checkboxLegend.py and the checkbox images from the link below: https://github.com/CSCD01/team_17-project/tree/master/Deliverable_4
2. Create a new Python file in the same folder that you have downloaded.
3. Edit the Python file.
4. First, in the file, you will need to import the dependencies. This ensures that your file has access to the methods it needs to access. To do so, paste the follow code:
   ```
   import matplotlib.pyplot as plt
   import matplotlib.patches as mpatches
   from checkboxLegend import checkboxLegend
   ```
5. After that, create your graphs as needed.
6. Next, we will call the checkboxlegend function. The usage of the checkboxLegend function will be the same as the legend method. The parameters and its order will be the same as the legend method.
   a. The differences in the structure of the method calls are listed below:
      ```
      parent.legend(arg1, arg2, ...) -> checkboxLegend(parent, arg1, arg2, ...)
      ```
   b. Here, the parent will be the first parameter in checkboxLegend but after, all the parameters will be in the same order.
7. After that, type the following below to show the graph:
   ```
   plt.show()
   ```
8. Run your code and you should see a graph with the interactive legend. Clicking the checkboxes will toggle the visibility of the respective subplots.

Interactive Legend - Example



1. Create a new python file in the same directory as the checkboxLegend.py file.
2. Paste the following code (this code is also available in the demo file):

```
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
from checkboxLegend import checkboxLegend

line1,  = plt.plot([1, 2], [1, 2], color='r', lw=2)
line2,  = plt.plot([1, 2], [2, 1], color='g', lw=4)
line3,  = plt.plot([1, 2], [3, 2], color='b', lw=8)
line2.set_visible(0)
rect = mpatches.Rectangle((1, 4), 2, 2)
rect.set_visible(0)
ax = plt.gca()
ax.add_patch(rect)
checkboxLegend(plt, [line1, line2, line3, rect], [
               "R", "G", "B", "rect"], fontsize=16)
plt.show()
```

3. Run the code.
4. Click the checkboxes to toggle the visibility of the respective elements.

Interactive Legend - Other Usage

Along with the example shown previously, there are also other ways to call the checkboxLegend() method, similar to the legend's options found here: https://matplotlib.org/3.2.1/api/_as_gen/matplotlib.pyplot.legend.html

1. **Automatic detection of elements to be shown in the legend**
    a. Here, you would not need to pass in all of the arguments to checkboxLegend() that were shown in the previous example. You would be able to specify the label through the Artist creation or using the set_label method.

```
line1,  = plt.plot([1, 2], [1, 2], color='r', lw=2, label="Red")
rect = mpatches.Rectangle((1, 4), 2, 2, label="Rect")
checkboxLegend(plt)
```

2. **Add label for already existing plot elements**
    a. Here, you can pass in the information to checkboxLegend() regarding what the labels should be for the plot elements but exclude the plot elements themselves.

```
line1,  = plt.plot([1, 2], [1, 2], color='r', lw=2)
line2,  = plt.plot([1, 2], [1, 3], color='b', lw=2)
checkboxLegend(plt, ["R","B"])
```

3. **Explicitly declaring the items to be placed in the legend**
    a. This option gives full control over what plot elements are placed in the legend and what the text of the legend should say

```
line1,  = plt.plot([1, 2], [1, 2], color='r', lw=2)
line2,  = plt.plot([1, 2], [1, 3], color='b', lw=2)
checkboxLegend(plt, [line1, line2], [ "R", "B"])
```