MATPLOTLIB ADVANCED PLOTS

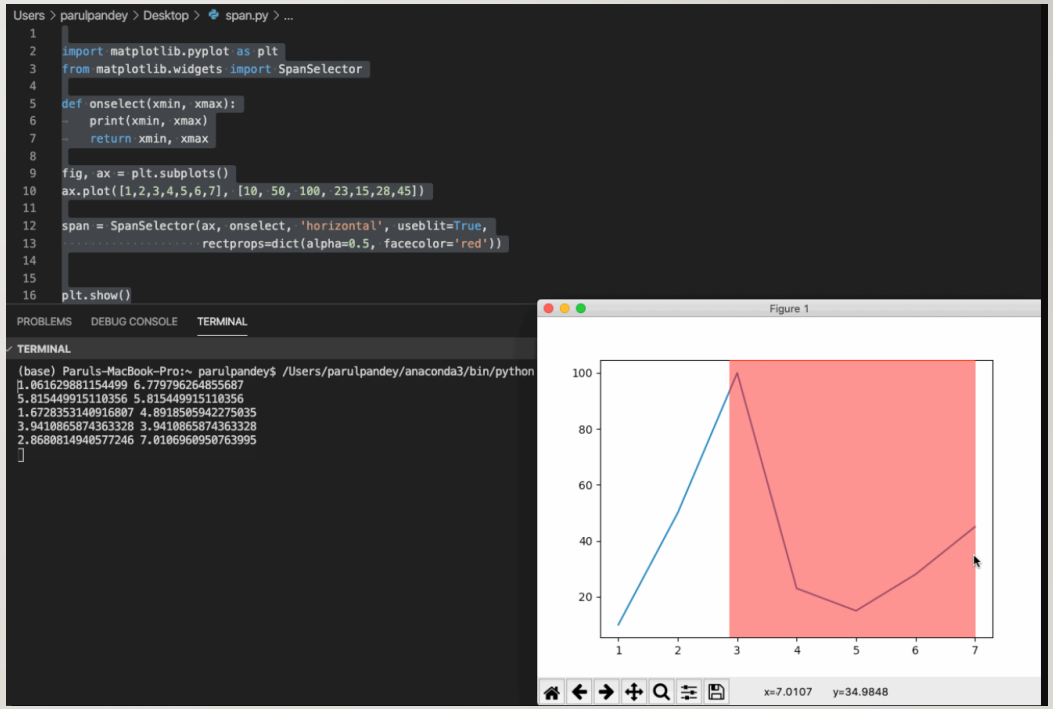
SULTAN DANIYAR

AGENDA

- Span Selector
- Broken Barh—Broken Horizontal Bar plot
- Table Demo
- Watermark Images
- XKCD Plots
- 3D Scatter Plot
- Surface Plot
- Conclusion

SPAN SELECTOR

Span Selector is a mouse widget in matplotlib. Widgets are python objects which are used to include some interactive functionality. Span Selector returns the maximum and minimum values of a selected region in a graph, through the mouse selection.

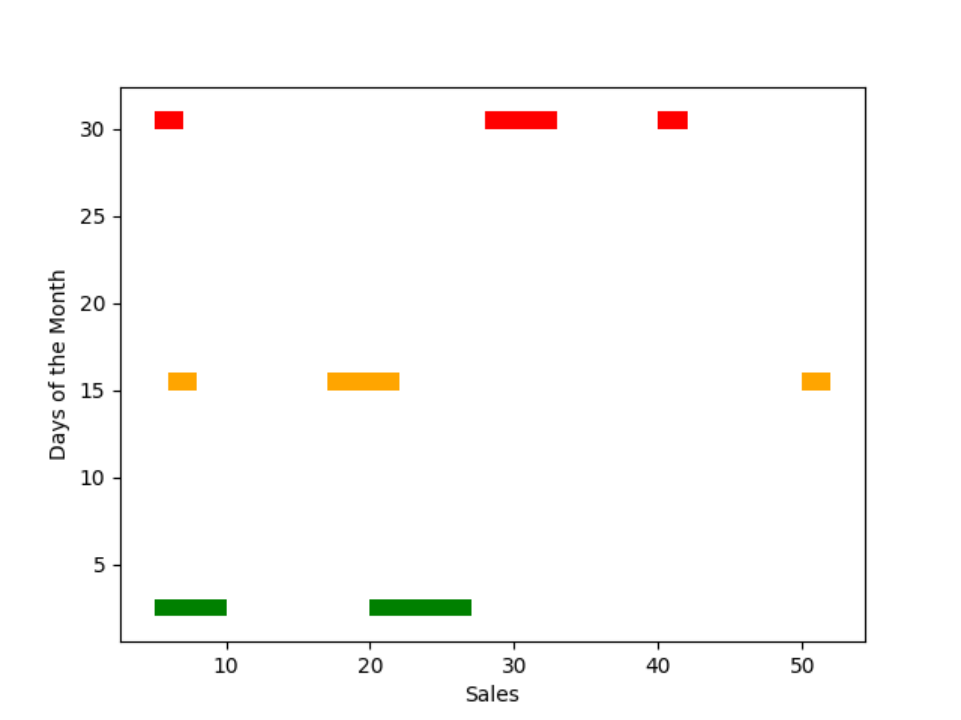


CODE SNIPPET

```
import matplotlib.pyplot as plt

from matplotlib.widgets import SpanSelector

def onselect(xmin, xmax):
        print(xmin, xmax)

        return xmin, xmax

fig, ax = plt.subplots()
ax.plot([1,2,3,4,5,6,7], [10, 50, 100, 23,15,28,45])

span = SpanSelector(ax, onselect, 'horizontal',
useblit=True,  rectprops=dict(alpha=0.5, facecolor='red'))
plt.show()
```

BROKEN BARH—BROKEN HORIZONTAL BAR PLOT

A “broken” horizontal bar plot is a plot that has gaps. It is used in situations when the data has values that vary considerably—for instance, a dataset consisting of extreme temperature ranges. Broken bar charts are ideal in this case since they can plot both the maximum and minimum ranges perfectly.

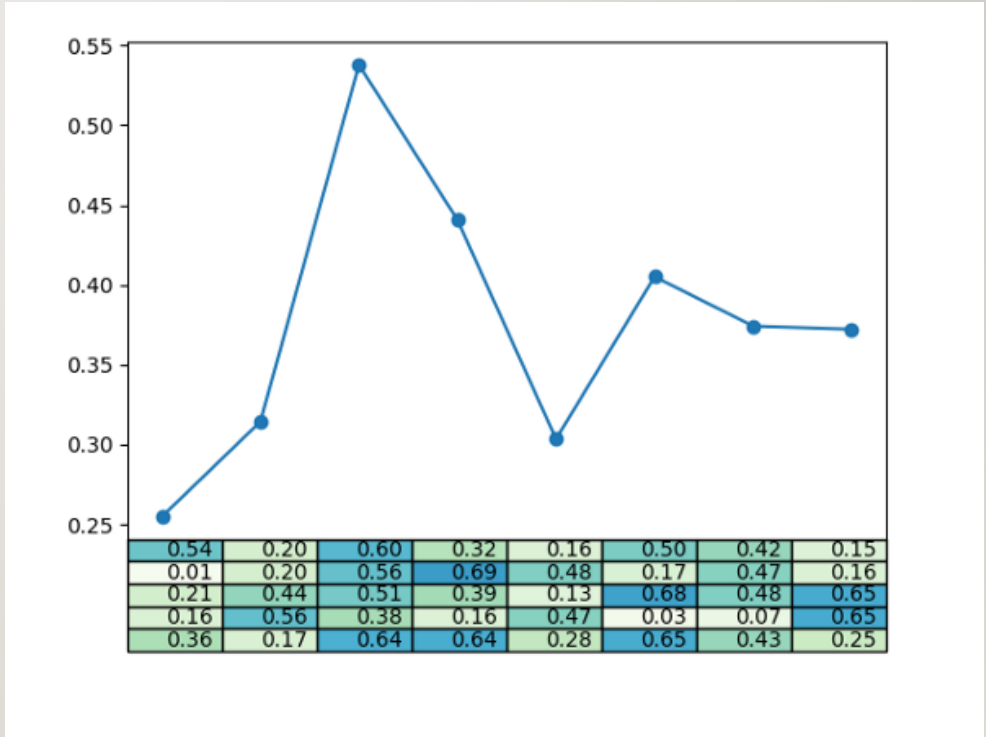


CODE SNIPPET

```
import matplotlib.pyplot as plt
#Defining the x and y ranges
xranges = [(5,5), (20,5),(20,7)]
yrange = (2,1)
#Plotting the broken bar chart
plt.broken_barh(xranges, yrange, facecolors='green')
xranges = [(6,2), (17,5),(50,2)]
yrange = (15,1)
plt.broken_barh(xranges, yrange, facecolors='orange')
xranges = [(5,2), (28,5),(40,2)]
yrange = (30,1)
plt.broken_barh(xranges, yrange, facecolors='red')
plt.xlabel('Sales')
plt.ylabel('Days of the Month')
plt.show()
```

TABLE DEMO

Matplotlib's table function can display a table within a plot. This is especially handy when one wants to see the quickly visualize values in a table in the form of a bar graph keeping the table alongside. The table can be positioned at top, bottom or on sides of the plot. Here is how you can create one easily.

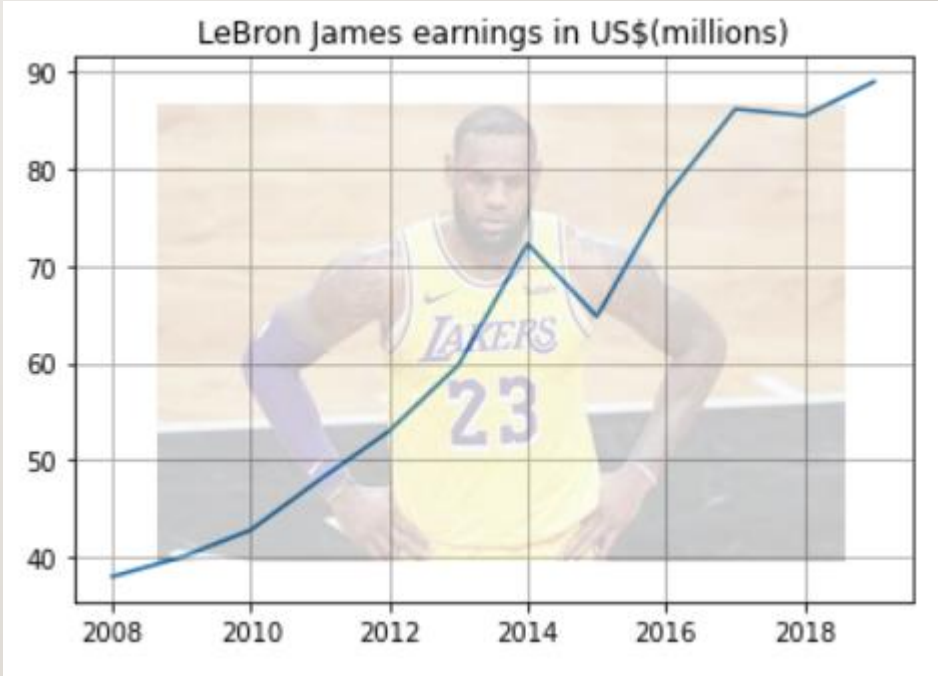


CODE SNIPPET

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

x = np.random.rand(5, 8)*.7
plt.plot(x.mean(axis=0), '-o', label='average per column')
plt.xticks([])

plt.table(cellText=[['%1.2f' % xxx for xxx in xx] for xx in
x],cellColours=plt.cm.GnBu(x),loc='bottom')

plt.show()
```

WATERMARK IMAGES

Sometimes having an image as a watermark helps to add a unique flavour to a plot. For instance, if we were to analyze the earnings of top athletes over the years, having their photographs in the background would help us to differentiate between plots of different players, easily. Let's analyze a dataset consisting of income of a number of athletes. We shall plot a graph of LeBron James earnings in US$(millions) over the years.

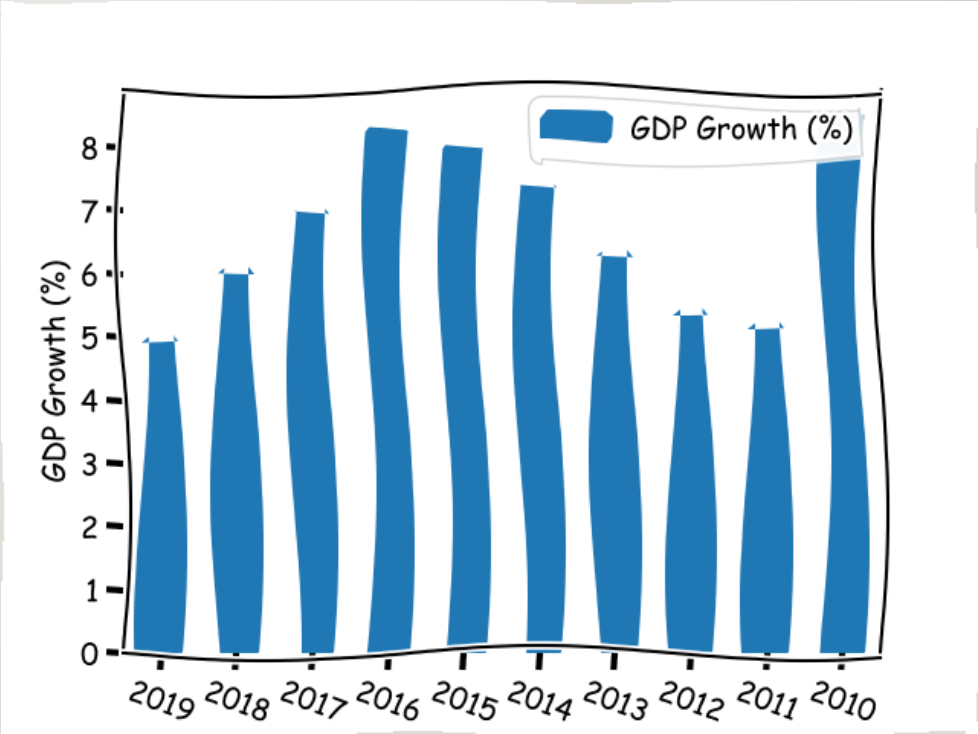


CODE SNIPPET

```
import numpy as np
import matplotlib.image as image
import matplotlib.pyplot as plt
import pandas as pd

df = pd.read_csv('income.csv')
im = image.imread('Lebron_James.jpeg') # Image
lebron_james = df[df['Name']=='LeBron James']
fig, ax = plt.subplots()
ax.grid()
ax.plot('Year','earnings ($ million)',data=lebron_james)
ax.set_title("LeBron James earnings in US$(millions)")
fig.figimage(im, 60, 40,cmap='ocean', alpha=.2)
plt.show()
```

XKCD PLOTS

Now let's add some element of fun in our plots. Xkcd is a webcomic by Randall Munroe and showcases a lot of humorous plots. These plots regularly make an appearance in a lot of data science presentations, for instance, the one below :

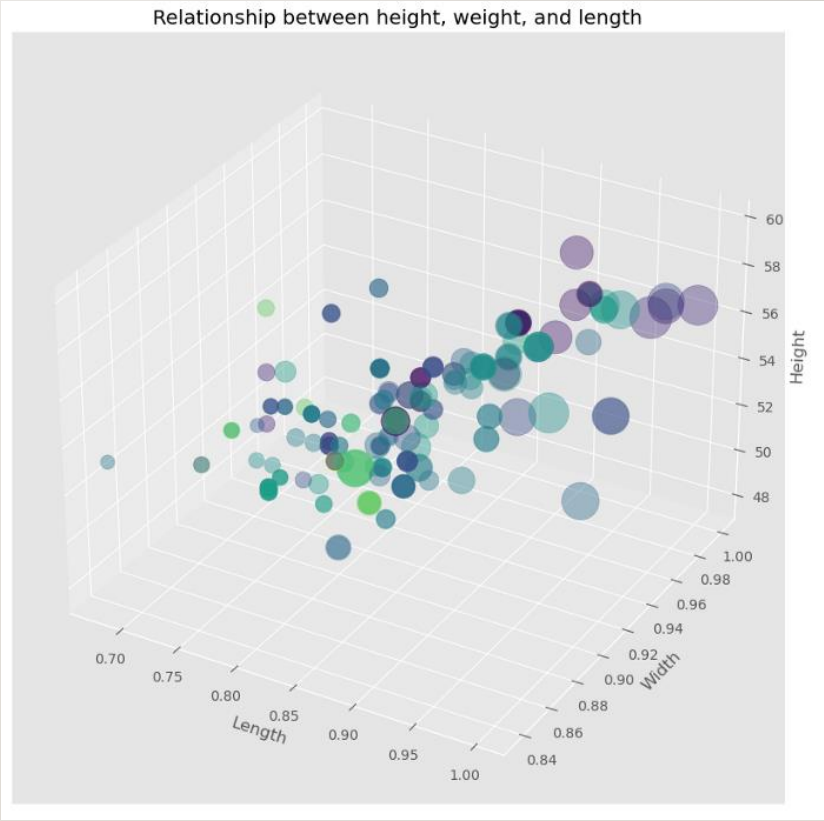


CODE SNIPPET

```
import pandas as pd
import matplotlib.pyplot as plt

df =pd.read_csv('https://raw.githubusercontent.com/parulnith/Website-articles-datasets/master/India%20GDP%20Growth%20Rate%20.csv', parse_dates=['Year'])
df['Year'] = df['Year'].apply(lambda x: pd.Timestamp(x).strftime('%Y'))

#calling xkcd() method
plt.xkcd(scale=5, length=400)
df.plot(x='Year',y='GDP Growth (%)',kind='bar')
plt.ylabel('GDP Growth (%)')
plt.xticks(rotation=-20)
plt.figure(figsize=(10,8))
plt.show()
```

3D SCATTER PLOT

The scatter plot is pretty self-explanatory. I am assuming that you know the 2d scatter plot. To make a 3d scatter plot, we just need to use the ‘scatter3D’ function and pass x, y, and z values. I choose to use the height, width, and length for x, y, and z values.

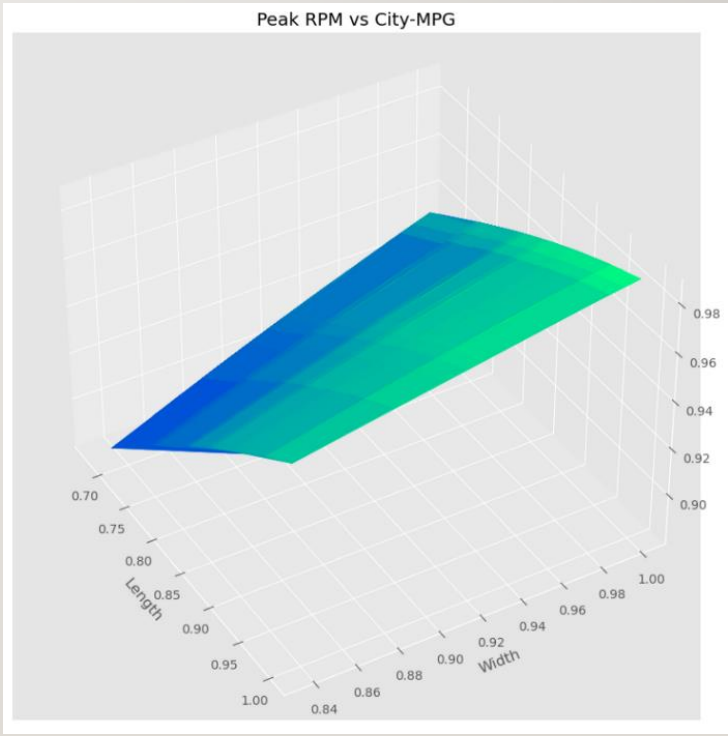


CODE SNIPPET

```
%matplotlib notebook
fig = plt.figure(figsize=(10, 10))
ax = plt.axes(projection="3d")
ax.scatter3D(df['length'], df['width'], df['height'],
c = df['peak-rpm'], s = df['price']/50, alpha = 0.4)
ax.set_xlabel("Length")
ax.set_ylabel("Width")
ax.set_zlabel("Height")
ax.set_title("Relationship between height, weight, and length")
plt.show()
```

SURFACE PLOT

For this type of plot one-dimensional x and y values do not work. So, we need to use the ‘meshgrid’ function to generate a rectangular grid out of two one-dimensional arrays.
This plot shows the relationship between two variables in a 3d setting.



CONCLUSION

These were some of the interesting and advanced functionalities available in matplotib. There are some other cool graphs and plots too, which we will cover in next lectures. In the meantime, grab an interesting dataset and put your newly learnt skills to use to get a good grasp of the topic.