Objective: This worksheet introduces the basics of the three-dimensional visualization using ***Matplotlib*** package of Python.

- First all the important packages (including NumPy and Pandas if required) are imported which would be required. Use *notebook* option if interactivity is required.

```
%matplotlib inline
import matplotlib as mlp
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

- Style is set so that the grid lines would be visible.

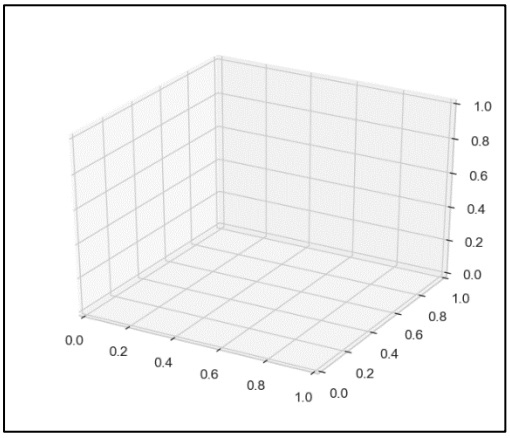
```
plt.style.use('seaborn-whitegrid')
```

- 3D Plotting is enabled importing the mplot3d toolkit.

```
from mpl_toolkits import mplot3d
```

- The function ***axes ()*** is called to create a 3D axes. With this 3D axes enabled, a variety of three-dimensional plot types can be drawn on it. It is shown separately but it is done in the same cell of notebook where plotting is done.

```
fig = plt.figure()
ax = plt.axes(projection='3d')
```



- NumPy provides a function ***linspace ()*** which returns evenly spaced numbers over a specified interval. The function is very useful to get the points for the axes.

- NumPy also provides a function ***meshgrid()*** which is useful to create the grid combinations. For example is x and y are two one-dimensional arrays having certain values, this functions returns the combination for each value of x with each value of y.

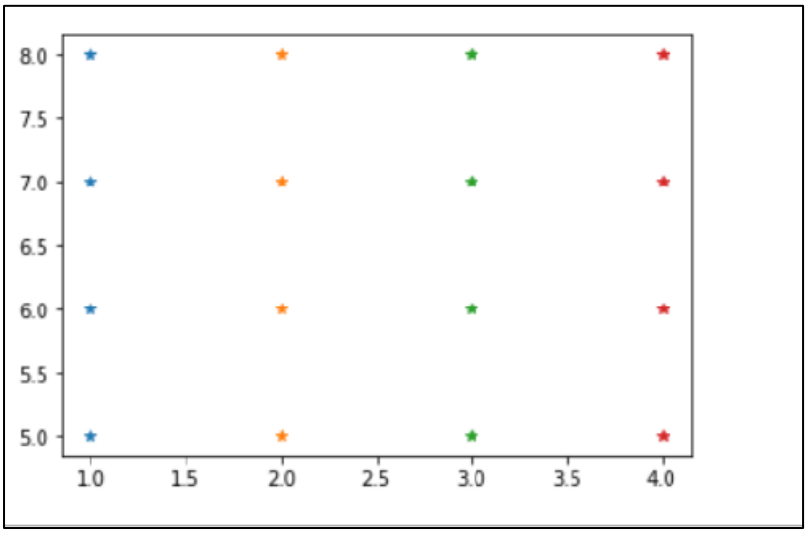
- Below example shows that x has 4 evenly spaced values between 1 and 4 (ends inclusive) and y has 4 evenly spaced values between 5 to 8 (ends inclusive). These values are created using the ***linspace ()*** function. A grid is also created using the ***meshgrid ()*** function.

```
x = np.linspace(1, 4, 4)
y = np.linspace(5, 8, 4)
xComb, yComb = np.meshgrid(x, y)
print (xComb)
print (yComb)
```

```
[[1. 2. 3. 4.]
 [1. 2. 3. 4.]
 [1. 2. 3. 4.]
 [1. 2. 3. 4.]]
[[5. 5. 5. 5.]
 [6. 6. 6. 6.]
 [7. 7. 7. 7.]
 [8. 8. 8. 8.]]
```

- Observe that *xComb* is organized row wise and *yComb* is organized column wise. Now when the following is executed, observe the plot. Each element of *xComb* is associated with each element of *yComb* to create a mesh.

```
plt.plot(xComb, yComb, '*')
```

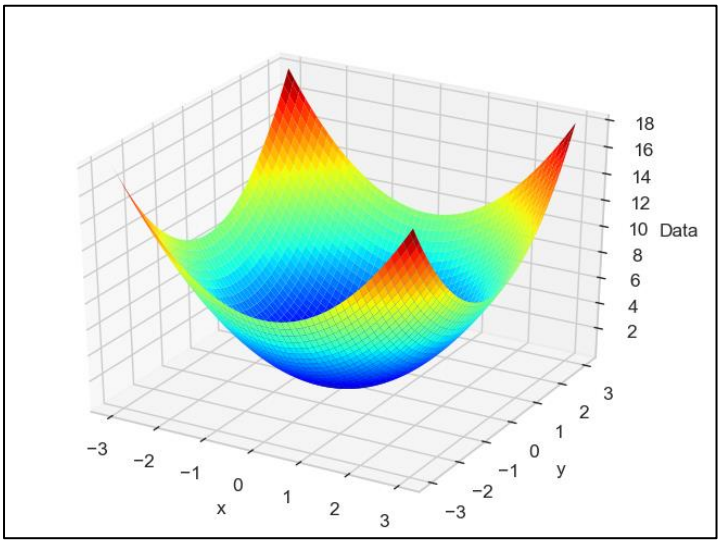


- Above functions are very useful to create 3D plots using Matplotlib. Let us say we have to draw a 3D plot for $z = x^2 + y^2$. Where z is the 3rd dimension and its value is to be calculated for 100 evenly spaced values of x and y each between -3 to 3. Here we need all the combinations of x and y in the given range. This can be achieved as using the following code using the function ***plot_surface().***

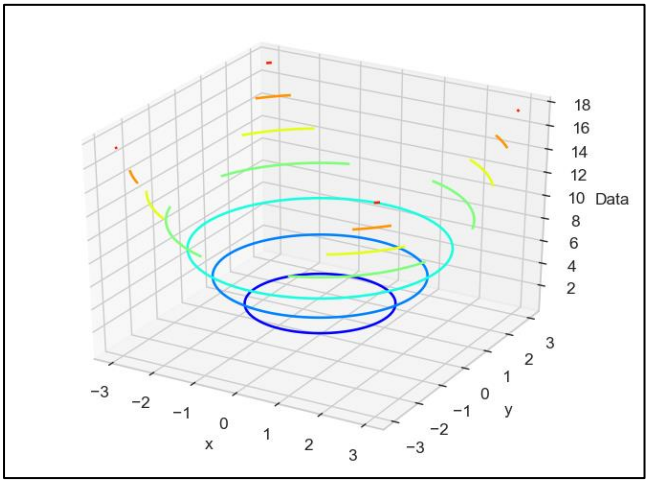
- Note that 3D axes projection is done along with the code which plots the surface for the desired function. Otherwise the plotting will happen along with the notebook cell where axes were projected.

```
points = 100
xLine = np.linspace(-3, 3, points)
yLine = np.linspace(-3, 3, points)
xComb,  yComb = np.meshgrid (xLine, yLine)

z = xComb**2 + yComb**2
fig = plt.figure()
ax = plt.axes(projection='3d')
ax.plot_surface(xComb, yComb, z, cmap='jet')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('Data')
```

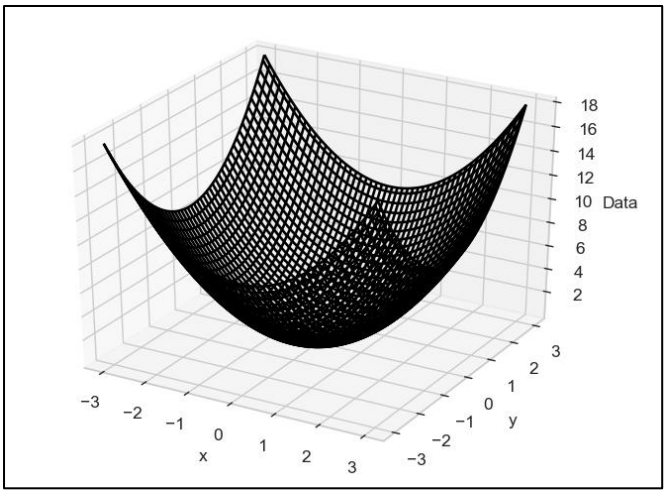


- The function ***contour3D()*** in place of ***plot_surface ()*** can be used to plot the contour skeleton:

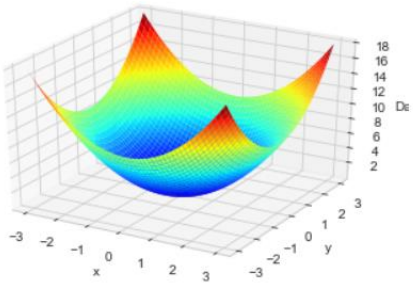


- A wireframe plot can also be drawn using the function ***plot_wireframe().***

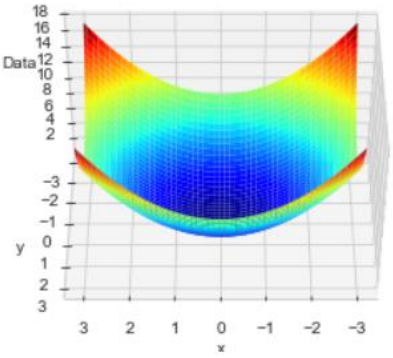
```
ax.plot_wireframe(xComb, yComb, z, color = 'black')
```



- While drawing the plot in the inline mode (static plots), viewing angles need to be adjusted. This can be done using ***view_init ()*** function. It takes two arguments. The first argument specifies the elevation angle (above the x-y axes). The second argument specified the azimuthal angle around the z-axis. For example to view a 3D plot from 60 degrees of elevation and 35 degrees counter clock wise of the z-axis, the function will be called as: `ax.view_init(60, 35)`. Few other examples are shown below:



`ax.view_init(None, None)`



`ax.view_init(45, 90)`

Exercise:

Taking suitable values of x and y, plot the 3D surface, contour and wireframe plots for the following function: $z = \frac{1}{1+e^{-x-y}}$