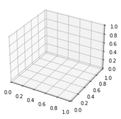
01. 3D Visualizations in Matplotlib

To get ready, you need to install one additional library as follows:

In [1]:
```
!pip3 install PyQt5
```

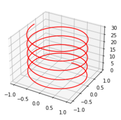
```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: PyQt5 in /home/linus/.local/lib/python3.8/site-packages (5.15.6)
Requirement already satisfied: PyQt5-Qt5>=5.15.2 in /home/linus/.local/lib/python3.8/site-packages (from PyQt5) (5.15.2)
Requirement already satisfied: PyQt5-sip<13,>=12.8 in /home/linus/.local/lib/python3.8/site-packages (from PyQt5) (12.9.1)
```

In [2]:
```
# Qt is a cross-platform library for GUI. PyQt5 is the Python binding for
# Qt. Once the library is installed, you can use the following magical
# command to force Jupyter Notebook to show the visualizations in a
# separate QT window:

%matplotlib qt
%matplotlib inline

# So, when you create visualizations, you are also able to interact
# with them. Let's learn the basics. First, we import all the required
# libraries, as shown here:

import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d

# Then we create a figure object, as shown here:
fig = plt.figure()

# Then we create a 3D axis as follows:
ax = plt.axes(projection='3d')

# You have to add the code for the visualization after this.
# However, for this example, you will create the visualization
# for an empty figure and axes with the following line:
plt.show()
```



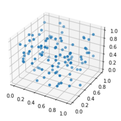
02. Plotting 3D Lines

In [3]:
```
# Let's plot a 3D line. Let's create a figure and axes, as shown here:
%matplotlib qt
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d
fig = plt.figure()
ax = plt.axes(projection='3d')
# Let's create 3D data as follows:
z = np.linspace(0, 30, 1000)
x = np.sin(z)
y = np.cos(z)
# You can create a 3D plot as follows:
ax.plot3D(x, y, z, 'red')
plt.show()
```



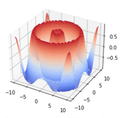
03. Plotting Scatter Plots

In [4]:
```
# You can create random points and show them with a 3D scatter as follows.
# Let's create a figure and axes first, as shown here:
%matplotlib qt
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d
fig = plt.figure()
ax = plt.axes(projection='3d')
# You can create the random data points as follows:
y = np.random.random(100)
x = np.random.random(100)
z = np.random.random(100)
# The points can be visualized with a scatter plot as follows:
ax.scatter3D(x, y, z, cmap='cool');
plt.show()
```



04. 3D Contours

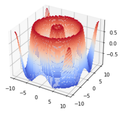
In [9]:
```
# You can create 3D contours with the functions contour() and contour3D().

# Let's create some data to be visualized.
%matplotlib qt
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d

fig = plt.figure()
ax = plt.axes(projection='3d')
x = np.linspace(-10, 10, 30)
y = np.linspace(-10, 10, 30)
X, Y = np.meshgrid(x, y)
Z = np.sin(np.sqrt(X ** 2 + Y ** 2))

# You can create a contour as follows:
fig = plt.figure()
ax.contour(X, Y, Z, 50, cmap='coolwarm')
plt.show()
```



```
<Figure size 432x288 with 0 Axes>
```

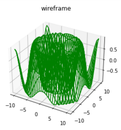
05. Filled contour

In [15]:
```
# You can also create a filled contour with the function contourf()
# as follows:

%matplotlib qt
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d
# ax = plt.axes(projection='3d')
x = np.linspace(-10, 10, 30)
y = np.linspace(-10, 10, 30)
X, Y = np.meshgrid(x, y)
Z = np.sin(np.sqrt(X ** 2 + Y ** 2))
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
ax.contourf(X, Y, Z, 50, cmap='coolwarm')
plt.show()
```

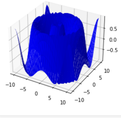


06. Wireframes, Surfaces, and Sample Data

In [111]:
```
# You can plot a wireframe of the same dataset as follows:

%matplotlib qt
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d
# ax = plt.axes(projection='3d')
x = np.linspace(-10, 10, 30)
y = np.linspace(-10, 10, 30)
X, Y = np.meshgrid(x, y)
Z = np.sin(np.sqrt(X ** 2 + Y ** 2))
fig = plt.figure()
ax = plt.axes(projection='3d')
ax.plot_wireframe(X, Y, Z, color='Green')
ax.set_title('wireframe')
plt.show()
```



In [12]:
```
# The same data can be visualized as a 3D surface as follows:

%matplotlib qt
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits import mplot3d
# ax = plt.axes(projection='3d')
x = np.linspace(-10, 10, 30)
y = np.linspace(-10, 10, 30)
X, Y = np.meshgrid(x, y)
Z = np.sin(np.sqrt(X ** 2 + Y ** 2))
fig = plt.figure()
ax = plt.axes(projection='3d');
ax.plot_surface(X, Y, Z, color='Blue')
ax.set_title('Surface Plot')
plt.show()
```



In [13]:
```
# You can also use the sample data that comes with the Matplotlib library
# for demonstrating visualizations. The function get_test_data() can
# fetch that sample data as follows:

%matplotlib qt
%matplotlib inline
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import mplot3d
from mpl_toolkits.mplot3d import axes3d
# ax = plt.axes(projection='3d')
x = np.linspace(-10, 10, 30)
y = np.linspace(-10, 10, 30)
X, Y = np.meshgrid(x, y)
Z = np.sin(np.sqrt(X ** 2 + Y ** 2))
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
X, Y, Z = axes3d.get_test_data(0.02)
ax.plot_wireframe(X, Y, Z,rstride=10,cstride=10)
plt.show()
```