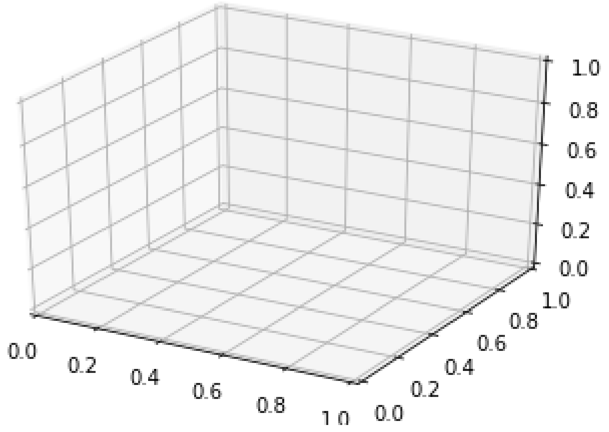
In [1]:

```
# 3D Visualizations in Matplotlib

# Wireframes, surfaces, and 3D contours are used to show volumetric data.
# Bar graphs are used to show categorical data. Quiver plots are used for
# visualizing vectors.

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

!pip3 install PyQt5
```

```
Collecting PyQt5
  Downloading PyQt5-5.15.6-cp36-abi3-win_amd64.whl (6.7 MB)
     ---------------------------------------- 6.7/6.7 MB 79.4 kB/s eta 0:00:00
Collecting PyQt5-Qt5>=5.15.2
  Downloading PyQt5_Qt5-5.15.2-py3-none-win_amd64.whl (50.1 MB)
     -------------------------------------- 50.1/50.1 MB 179.5 kB/s eta 0:00:00
Collecting PyQt5-sip<13,>=12.8
  Downloading PyQt5_sip-12.9.1-cp38-cp38-win_amd64.whl (77 kB)
     -------------------------------------- 77.5/77.5 KB 431.2 kB/s eta 0:00:00
Installing collected packages: PyQt5-Qt5, PyQt5-sip, PyQt5
Successfully installed PyQt5-5.15.6 PyQt5-Qt5-5.15.2 PyQt5-sip-12.9.1
```

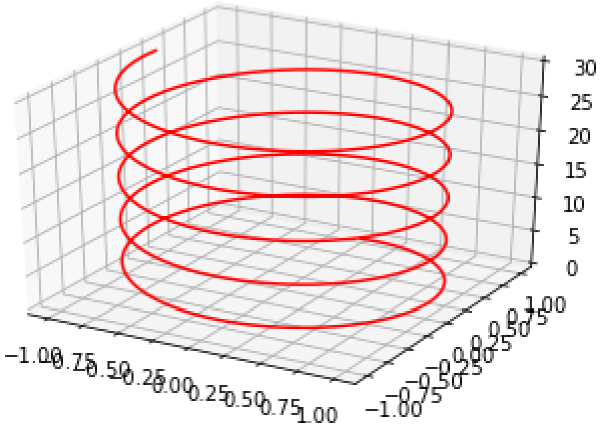
In [3]:

```
# 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()
```



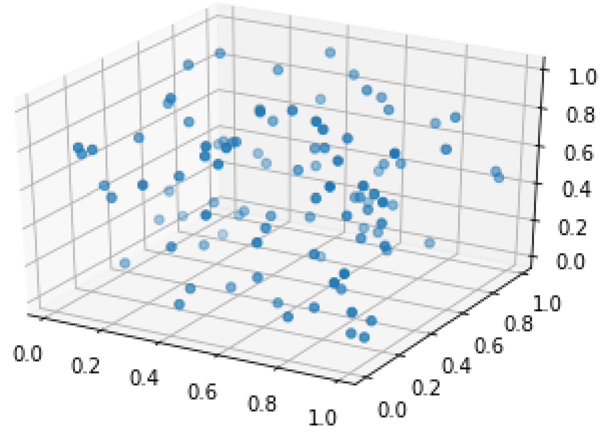
In [5]:

```
# Plotting 3D Lines
# 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()
```



In [6]:

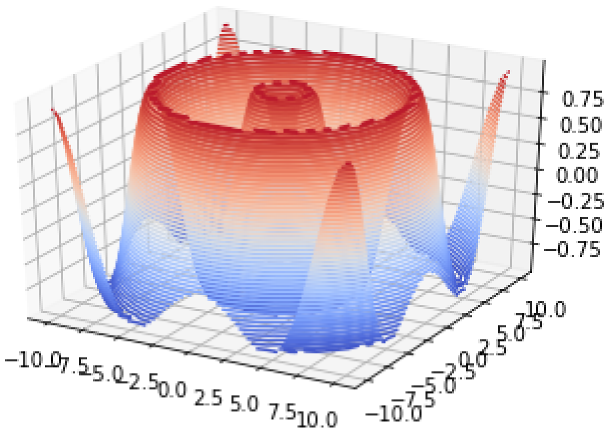
```
# 3D Scatter Plots
# You can create random points and show them with a 3D scatter as follows.
# Let’s create a figure and axes first, as shown here:

%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()
```



In [7]:

```
# 3D Contours
# 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 = fig.add_subplot(projection='3d')
ax.contour(X, Y, Z, 50, cmap='coolwarm')
plt.show()
```

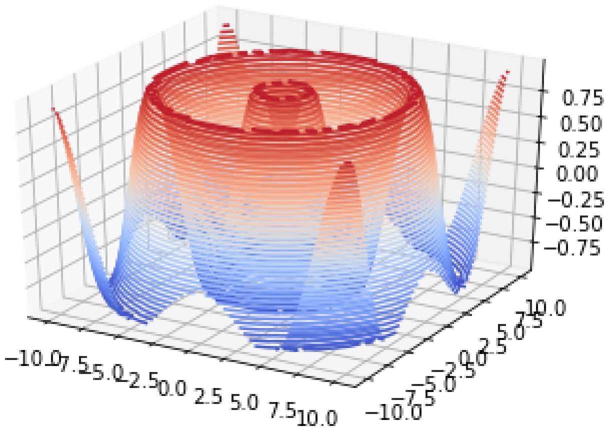
```
c:\python\lib\site-packages\numpy\core\_asarray.py:136: VisibleDeprecationWarning: Creating an ndarray from rag
ged nested sequences (which is a list-or-tuple of lists-or-tuples-or ndarrays with different lengths or shapes)
is deprecated. If you meant to do this, you must specify 'dtype=object' when creating the ndarray
  return array(a, dtype, copy=False, order=order, subok=True)
```



In [13]:

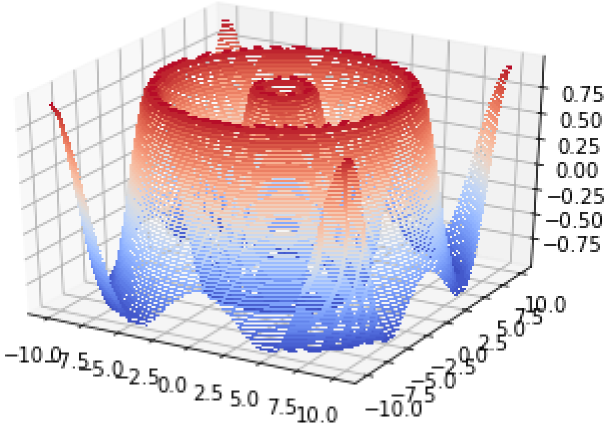
```
# You can obtain similar output as visualized:

%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))
fig = plt.figure()
ax = plt.axes(projection='3d')
ax.contour3D(X, Y, Z, 40, cmap='coolwarm')
plt.show()
```

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



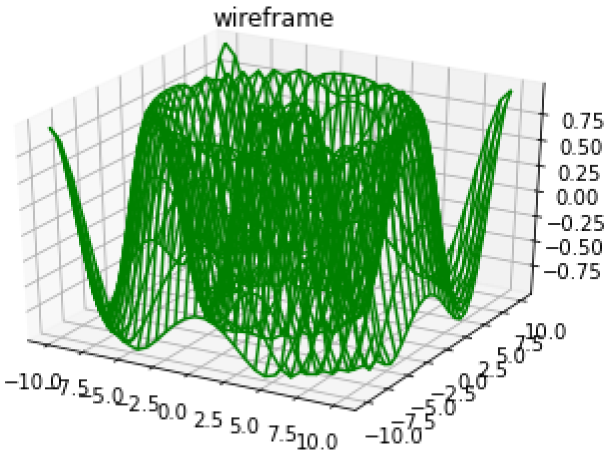
In [12]:

```
# 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()
```



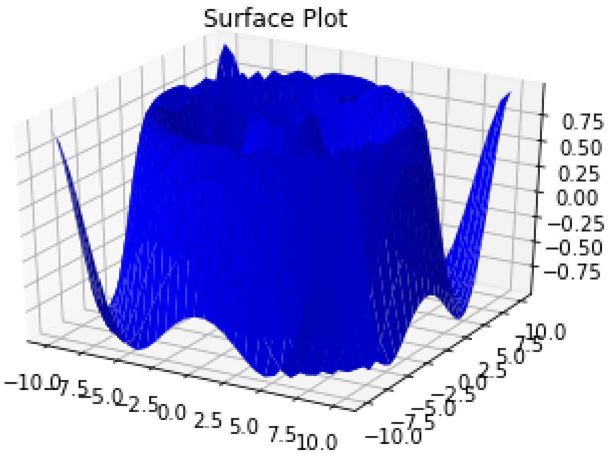
In [11]:

```
# Wireframes, Surfaces, and Sample Data
# 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()
```



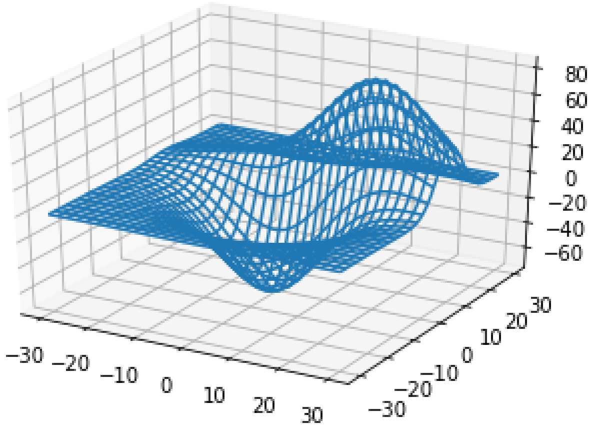
In [14]:

```
# 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 [15]:

```
# 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 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()
```


80
60
40
20
0
−20
−40
−60
−30
−20
−10
0
10
20
30
−30
−20
−10
0
10
20
30