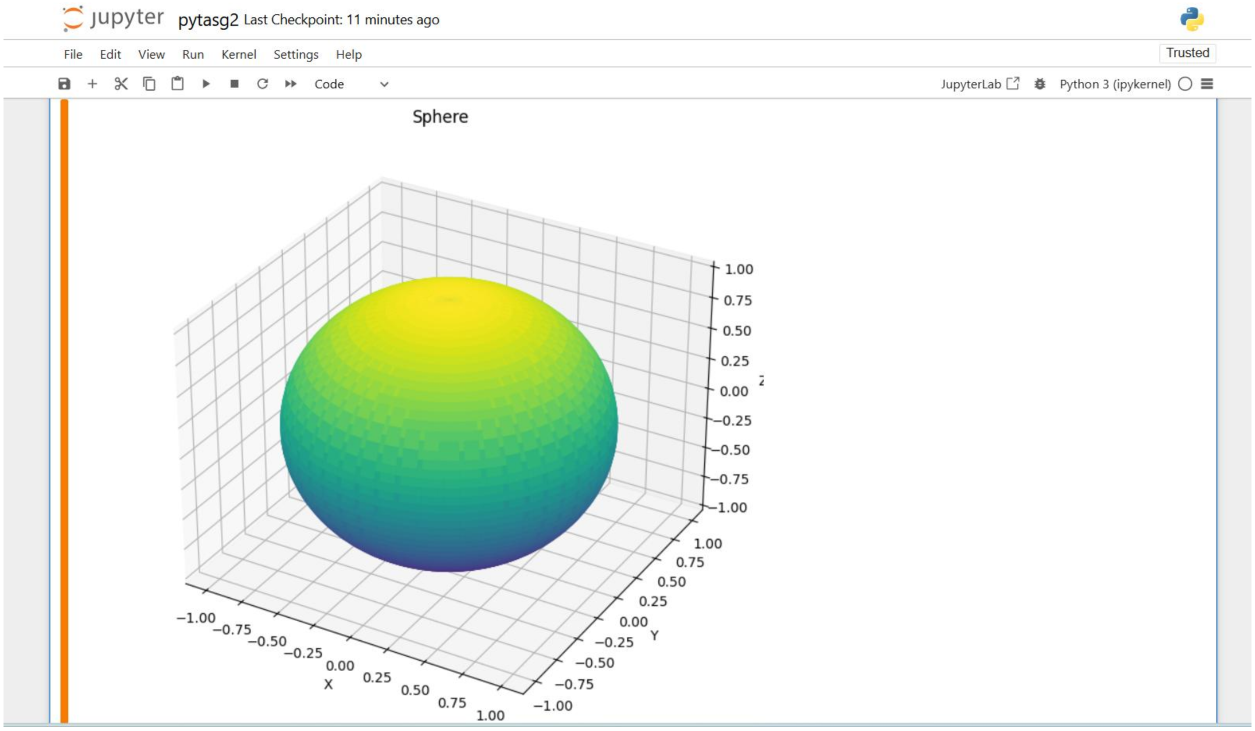
3D visualization of mathematical surfaces using Numpy

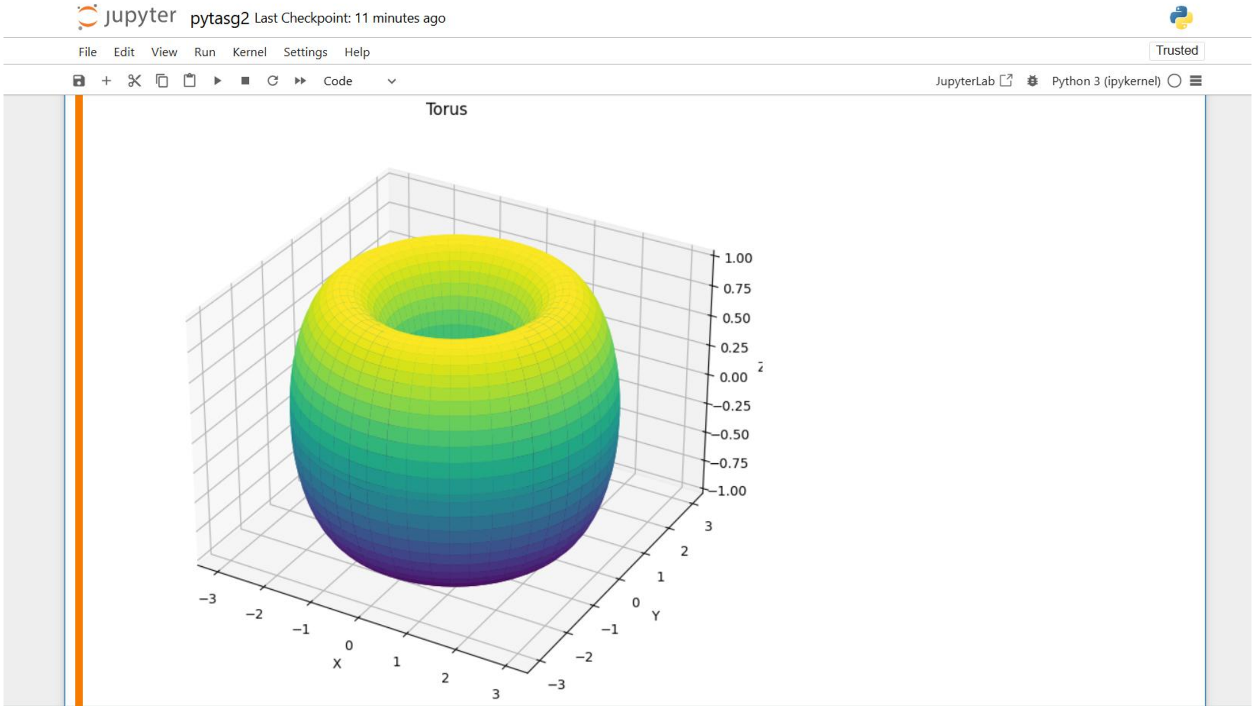
Code :

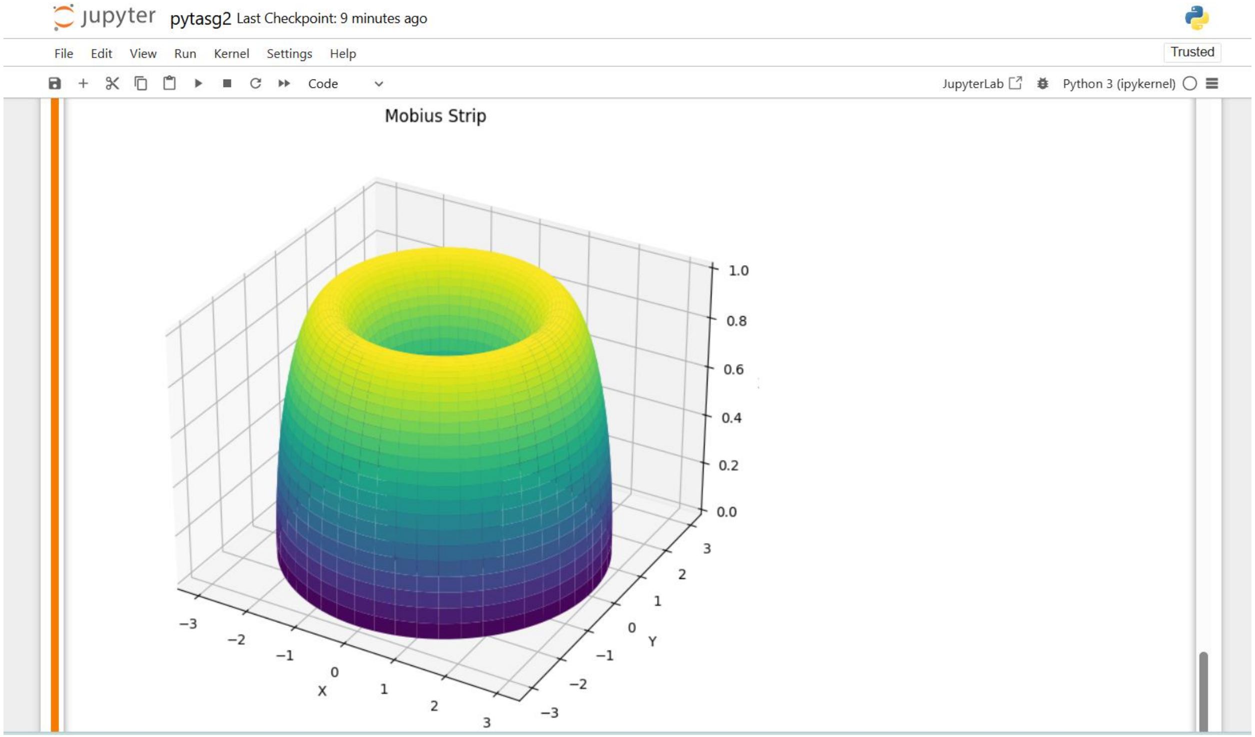
```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D


# Choose mathematical functions to visualize
def sphere(u, v):
    """Sphere surface"""
    x = np.cos(u) * np.sin(v)
    y = np.sin(u) * np.sin(v)
    z = np.cos(v)
    return x, y, z


def torus(u, v):
    """Torus surface"""
    R = 2
    r = 1
    x = (R + r * np.cos(v)) * np.cos(u)
    y = (R + r * np.cos(v)) * np.sin(u)
    z = r * np.sin(v)
    return x, y, z


def mobius(u, v):
    """Mobius strip surface"""
    R = 2
    r = 1
    x = (R + r * np.cos(v/2)) * np.cos(u)
    y = (R + r * np.cos(v/2)) * np.sin(u)
    z = r * np.sin(v/2)
    return x, y, z

# Generate data points for the chosen mathematical function
def generate_data(func, u_range, v_range, num_points):
    u = np.linspace(u_range[0], u_range[1], num_points)
    v = np.linspace(v_range[0], v_range[1], num_points)
    u, v = np.meshgrid(u, v)
    x, y, z = func(u, v)
    return x, y, z

# Create 3D visualization
def visualize_surface(x, y, z, title, cmap='viridis'):
    fig = plt.figure(figsize=(8, 8))
    ax = fig.add_subplot(111, projection='3d')
    ax.plot_surface(x, y, z, cmap=cmap, edgecolor='none')
    ax.set_title(title)
    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')
    plt.show()

# Example usage
u_range = (0, 2 * np.pi)
v_range = (0, 2 * np.pi)
num_points = 100
```

```
x, y, z = generate_data(sphere, u_range, v_range, num_points)

visualize_surface(x, y, z, 'Sphere')


x, y, z = generate_data(torus, u_range, v_range, num_points)

visualize_surface(x, y, z, 'Torus')


x, y, z = generate_data(mobius, u_range, v_range, num_points)

visualize_surface(x, y, z, 'Mobius Strip')
```

output :







This script defines three mathematical functions: **sphere**, **torus**, and **mobius**, which generate the corresponding 3D surfaces. The **generate_data** function generates a grid of points in the parameter space and evaluates the chosen mathematical function at each point to obtain the corresponding coordinates in 3D space. The **visualize_surface** function creates a 3D plot using Matplotlib's **mplot3d** toolkit and customizes the plot appearance with labels, titles, and color schemes.

To use this tool, simply run the script and explore the 3D visualizations of the chosen mathematical surfaces. You can adjust the **u_range** and **v_range** variables to change the parameter ranges and experiment with different mathematical functions and parameter settings.