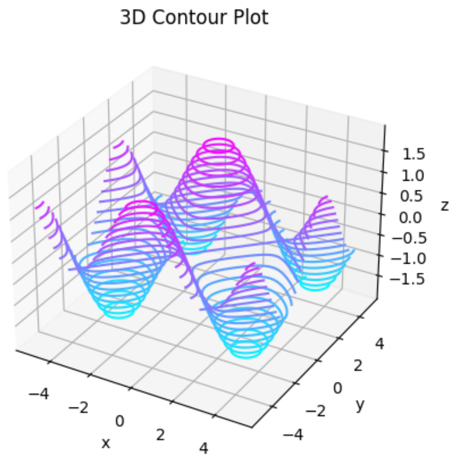
PYTHON GRAPHING: 3D CONTOURS

Author: Javier Huang
Editor: Jim Chen

Date Created: 06/03/23
Date Edited: 06/03/23
Version 1.0

Abstract

In this SOP, the principle of graphing 3D contours in the Python programming language is documented.



Contents

1 Uses of 3D Contour Graphing 3
2 How to install the Python packages 3
3 How to graph a 3D Contour Plot 4

List of Figures

Figure 1 VSCode Terminal 4
Figure 2 Command Prompt 4
Figure 3 Final 3D Contour Plot 6

1 Uses of 3D Contour Graphing

A contour plot is a useful visualization technique that allows us to explore the potential relationship between three variables by representing the 3-dimensional relationship in two dimensions. It involves plotting the x-axis and y-axis variables (often referred to as predictors) on the corresponding scales, while the response values (z) are represented by contours.[1]

Here are some common uses of 3D contour plots:

1. Electrostatic Potential

   3D contour plots serve as a valuable tool for visualizing the electrostatic potential in a given region of space. These plots depict equipotential surfaces, allowing to discern the distribution of electric potential and identify regions characterized by different potentials.

2. Electric and Magnetic Fields

   By graphically depicting field strengths at different points in space, 3D contour plots facilitate the visualization of field patterns and the identification of areas featuring stronger or weaker fields.

3. Temperature Distributions

   3D contour plots are constructed by mapping temperature values onto a three-dimensional grid, which allow for the visualization of temperature variations and gradients, thereby aiding in the analysis of heat transfer processes.

4. Fluid Dynamics

   3D contour plots help to visualize physical quantities associated with fluid flow, such as velocity, pressure, temperature, and vorticity. The plots can demonstrate flow behavior, including regions characterized by high or low velocities, pressure gradients, or areas of significant heat transfer.

5. Quantum Mechanical Wavefunctions

   3D contour plots are extensively utilized to visualize the wavefunctions of particles existing in three-dimensional space. By delineating contours that denote regions of varying probabilities for locating the particles, these plots provide insights into the spatial distribution of wavefunctions.

2 How to install the Python packages

1. Open your Integrated Development Environment (IDE). This SOP will focus on using Visual Studio Code (VSCode).
2. Go to the Terminal section of VSCode (Fig. 1) or open the Windows Command Prompt (Fig. 2).

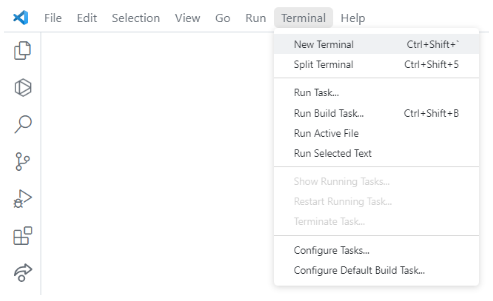


Figure 1: VSCode Terminal

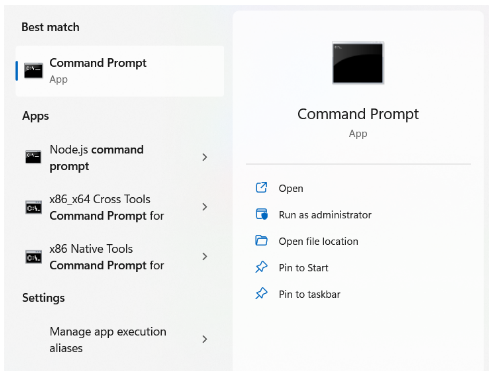


Figure 2: Command Prompt

3. The graphing requires two packages, NumPy and MatPlotLib. To install them. type in the following commands separately

```
pip install numpy

pip install matplotlib
```

3 How to graph a 3D Contour Plot

1. Create a new Python file, and import the libraries

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

2. Generate the data for the contour plot. You can use NumPy to create a grid of x, y, and z values. In this example, we'll create a grid ranging from -5 to 5 for both x and y, and use the function $\sin(\sqrt{x^2+y^2})$ for the z

```
def f(x, y):
    return np.sin(np.sqrt(x ** 2 + y ** 2))

x = np.linspace(-5, 5, 100)
y = np.linspace(-5, 5, 100)
X, Y = np.meshgrid(x, y)
Z = f(X, Y)
```

3. Plot the contour plot by passing the x, y, and z values, along with the number of contour levels (20) and the desired colormap

```
fig = plt.figure()
ax = plt.axes(projection='3d')
ax.contour3D(X, Y, Z, 20, cmap='jet')
```

4. Add labels and customize the plot if desired

```
ax.set_xlabel('X')
ax.set_ylabel('Y')
ax.set_zlabel('Z')
ax.set_title('3D Contour Plot')
```

5. Finally, display the plot

```
plt.show()
```

Here is the complete code for graphing a 3D contour plot in Python

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

def f(x, y):
    return np.sin(np.sqrt(x ** 2 + y ** 2))

x = np.linspace(-5, 5, 100)
y = np.linspace(-5, 5, 100)
X, Y = np.meshgrid(x, y)
Z = f(X, Y)

fig = plt.figure()
ax = plt.axes(projection='3d')
ax.contour3D(X, Y, Z, 20, cmap='jet')
ax.set_xlabel('x')
ax.set_ylabel('y')
ax.set_zlabel('z')
ax.set_title('3D Contour Plot')

plt.show()
```

The output of the program is in Fig. 3

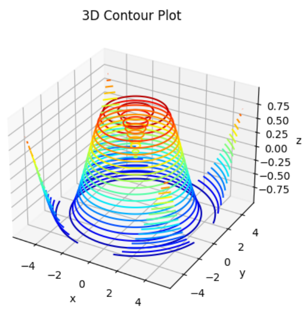


Figure 3: Final 3D Contour Plot

REFERENCES

[1] Contour plots and 3d surface plots. https://support.minitab.com/en-us/minitab/21/help-and-how-to/statistical-modeling/using-fitted-models/supporting-topics/graphs/contour-plots-and-3d-surface-plots/.