Python (/python) > 3D Charts (/python/3d-charts) > 3D Mesh Plots

Suggest an edit to this page (https://github.com/plotly/plotly.py/edit/doc-prod/doc/python/3d-mesh.md)

3D Mesh Plots in Python

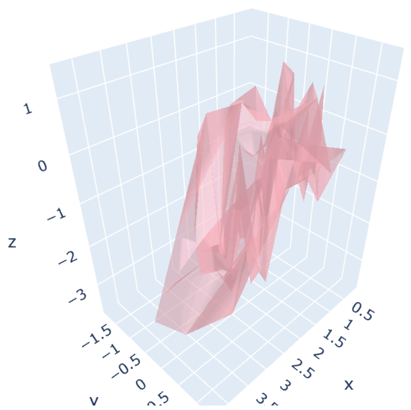
How to make 3D Mesh Plots

or cells

Simple 3D Mesh example

go.Mesh3d draws a 3D set of triangles with vertices given by x, y and z. If only coordinates are given, an algorithm such as Delaunay triangulation (https://en.wikipedia.org/wiki/Delaunay_triangulation) is used to draw the triangles. Otherwise the triangles can be given using the i, j and k parameters (see examples below).

```
import plotly.graph_objects as go
import numpy as np

# Download data set from plotly repo
pts = np.loadtxt(np.DataSource().open('https://raw.githubusercontent.com/plotly/datasets/master/mesh_dataset.txt'))
x, y, z = pts.T

fig = go.Figure(data=[go.Mesh3d(x=x, y=y, z=z, color='lightpink', opacity=0.50)])
fig.show()
```

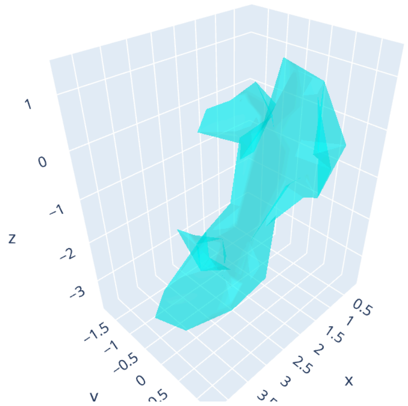


3D Mesh example with Alphahull

The alphahull parameter sets the shape of the mesh. If the value is -1 (default value) then Delaunay triangulation (https://en.wikipedia.org/wiki/Delaunay_triangulation) is used. If >0 then the alpha-shape algorithm (https://en.wikipedia.org/wiki/Alpha_shape) is used. If 0, the convex-hull (https://en.wikipedia.org/wiki/Convex_hull) is represented (resulting in a convex body).

```
import plotly.graph_objects as go
import numpy as np

pts = np.loadtxt(np.DataSource().open('https://raw.githubusercontent.com/plotly/datasets/master/mesh_dataset.txt'))
x, y, z = pts.T

fig = go.Figure(data=[go.Mesh3d(x=x, y=y, z=z,
                   alphahull=5,
                   opacity=0.4,
                   color='cyan')])
fig.show()
```

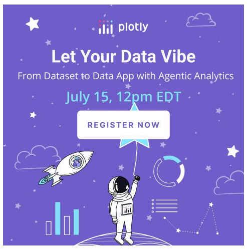
or cells



3D Mesh in Dash

Dash (https://plotly.com/dash/) is the best way to build analytical apps in Python using Plotly figures. To run the app below, run pip install dash, click "Download" to get the code and run python app.py.

Get started with the official Dash docs (https://dash.plotly.com/installation) and **learn how to effortlessly** style (https://plotly.com/dash/design-kit/) **&** deploy (https://plotly.com/dash/app-manager/) **apps like this with** Dash Enterprise (https://plotly.com/dash/)**.**



or cells

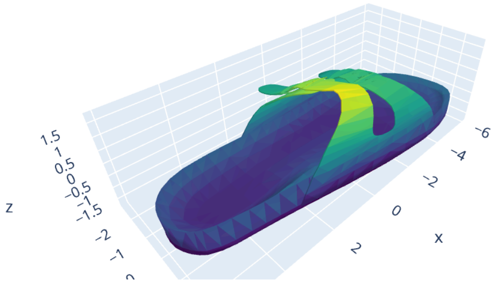
DOWNLOAD

```
from dash import Dash, dcc, html, Input, Output
import plotly.graph_objects as go
import pandas as pd


base_url = "https://raw.githubusercontent.com/plotly/datasets/master/ply/"
mesh_names = ['sandal', 'scissors', 'shark', 'walkman']
dataframes = {
    name: pd.read_csv(base_url + name + '-ply.csv')
    for name in mesh_names
}

app = Dash(__name__)

app.layout = html.Div([
    html.H4('PLY Object Explorer'),
    html.P("Choose an object:"),
    dcc.Dropdown(
        id='dropdown',
        options=mesh_names,
        value="sandal",
        clearable=False
    ),
    dcc.Graph(id="graph"),
])
```

PLY Object Explorer

Choose an object:

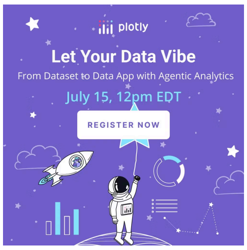
sandal ▾



Sign up for Dash Club → Free cheat sheets plus updates from Chris Parmer and Adam Schroeder delivered to your inbox every two months. Includes tips and tricks, community apps, and deep dives into the Dash architecture. Join now (https://go.plotly.com/dash-club?utm_source=Dash+Club+2022&utm_medium=graphing_libraries&utm_content=inline).

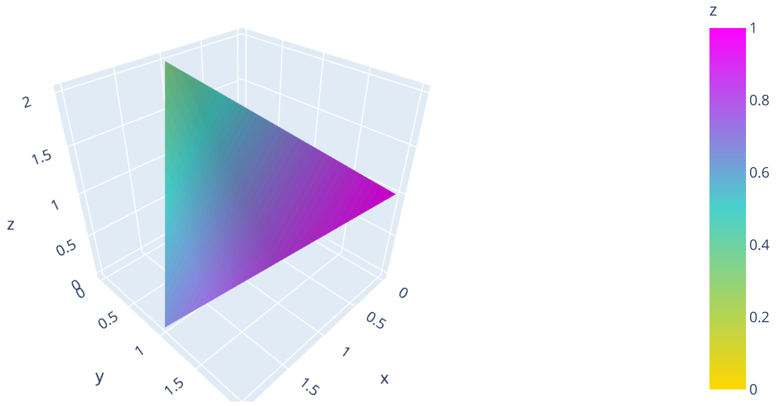
Mesh Tetrahedron

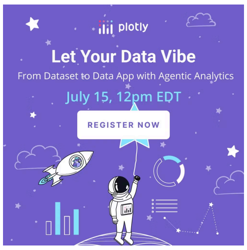
In this example we use the i, j and k parameters to specify manually the geometry of the triangles of the mesh.



or cells

```
import plotly.graph_objects as go

fig = go.Figure(data=[
    go.Mesh3d(
        x=[0, 1, 2, 0],
        y=[0, 0, 1, 2],
        z=[0, 2, 0, 1],
        colorbar=dict(title=dict(text='z')),
        colorscale=[[0, 'gold'],
                    [0.5, 'mediumturquoise'],
                    [1, 'magenta']],
        # Intensity of each vertex, which will be interpolated and color-coded
        intensity=[0, 0.33, 0.66, 1],
        # i, j and k give the vertices of triangles
        # here we represent the 4 triangles of the tetrahedron surface
        i=[0, 0, 0, 1],
        j=[1, 2, 3, 2],
        k=[2, 3, 1, 3],
        name='y',
        showscale=True
    )
])

fig.show()
```

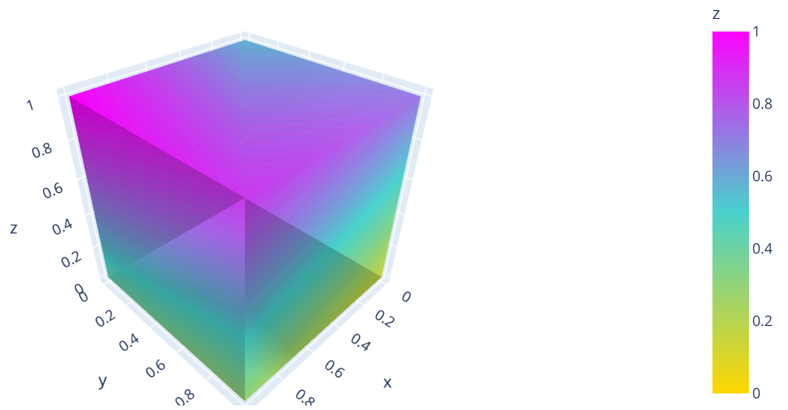




Mesh Cube

or cells

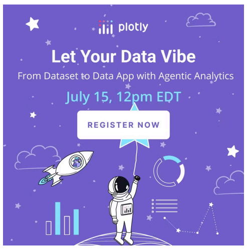
```
import plotly.graph_objects as go
import numpy as np

fig = go.Figure(data=[
    go.Mesh3d(
        # 8 vertices of a cube
        x=[0, 0, 1, 1, 0, 0, 1, 1],
        y=[0, 1, 1, 0, 0, 1, 1, 0],
        z=[0, 0, 0, 0, 1, 1, 1, 1],
        colorbar=dict(title=dict(text='z')),
        colorscale=[[0, 'gold'],
                    [0.5, 'mediumturquoise'],
                    [1, 'magenta']],
        # Intensity of each vertex, which will be interpolated and color-coded
        intensity = np.linspace(0, 1, 8, endpoint=True),
        # i, j and k give the vertices of triangles
        i = [7, 0, 0, 0, 4, 4, 6, 6, 4, 0, 3, 2],
        j = [3, 4, 1, 2, 5, 6, 5, 2, 0, 1, 6, 3],
        k = [0, 7, 2, 3, 6, 7, 1, 1, 5, 5, 7, 6],
        name='y',
        showscale=True
    )
])

fig.show()
```



Intensity values defined on vertices or cells

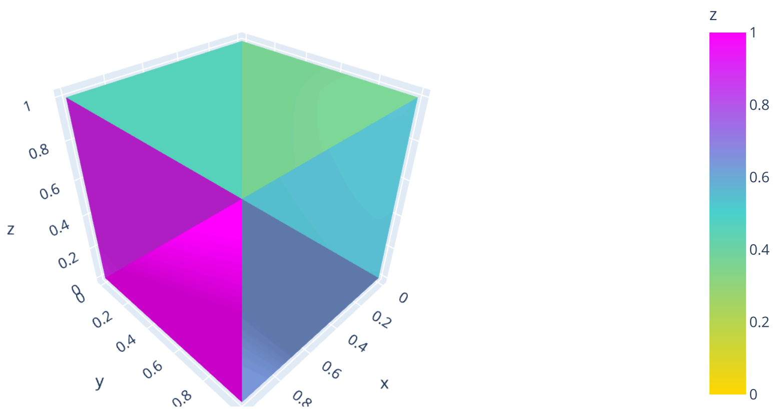
The intensitymode attribute of go.Mesh3d can be set to vertex (default mode, in which case intensity values are interpolated between values defined on vertices), or to cell (value of the whole cell, no interpolation). Note that the intensity parameter should have the same length as the number of vertices or cells, depending on the intensitymode.

Whereas the previous example used the default intensitymode='vertex', we plot here the same mesh with intensitymode='cell'.



or cells

```
import plotly.graph_objects as go
fig = go.Figure(data=[
    go.Mesh3d(
        # 8 vertices of a cube
        x=[0, 0, 1, 1, 0, 0, 1, 1],
        y=[0, 1, 1, 0, 0, 1, 1, 0],
        z=[0, 0, 0, 0, 1, 1, 1, 1],
        colorbar=dict(title=dict(text='z')),
        colorscale=[[0, 'gold'],
                    [0.5, 'mediumturquoise'],
                    [1, 'magenta']],
        # Intensity of each vertex, which will be interpolated and color-coded
        intensity = np.linspace(0, 1, 12, endpoint=True),
        intensitymode='cell',
        # i, j and k give the vertices of triangles
        i = [7, 0, 0, 0, 4, 4, 6, 6, 4, 0, 3, 2],
        j = [3, 4, 1, 2, 5, 6, 5, 2, 0, 1, 6, 3],
        k = [0, 7, 2, 3, 6, 7, 1, 1, 5, 5, 7, 6],
        name='y',
        showscale=True
    )
])

fig.show()
```



Reference

See https://plotly.com/python/reference/mesh3d/ (https://plotly.com/python/reference/mesh3d/) for more information and chart attribute options!

What About Dash?

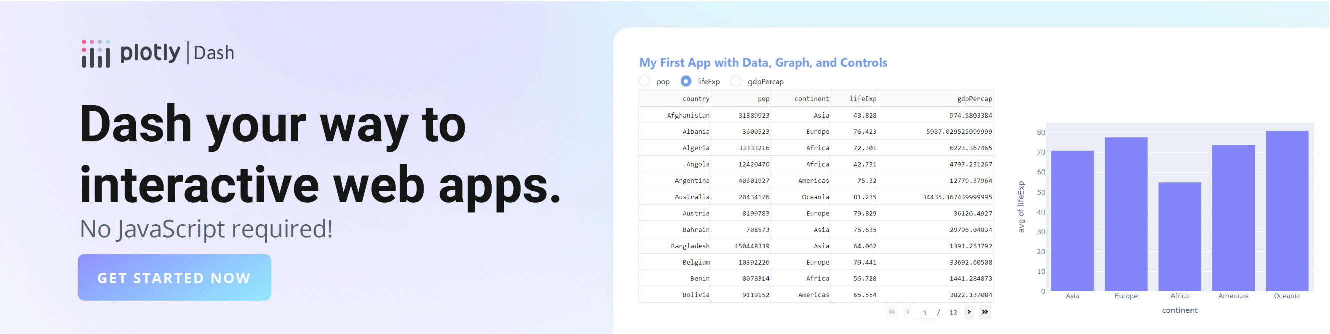
Dash (https://dash.plot.ly/) is an open-source framework for building analytical applications, with no Javascript required, and it is tightly integrated with the Plotly graphing library.

Learn about how to install Dash at https://dash.plot.ly/installation (https://dash.plot.ly/installation).

Everywhere in this page that you see fig.show(), you can display the same figure in a Dash application by passing it to the figure argument of the Graph component (https://dash.plot.ly/dash-core-components/graph) from the built-in dash_core_components package like this:

or cells

```
import plotly.graph_objects as go # or plotly.express as px
fig = go.Figure() # or any Plotly Express function e.g. px.bar(...)
# fig.add_trace( ... )
# fig.update_layout( ... )

from dash import Dash, dcc, html

app = Dash()
app.layout = html.Div([
    dcc.Graph(figure=fig)
])

app.run(debug=True, use_reloader=False)  # Turn off reloader if inside Jupyter
```



My First App with Data, Graph, and Controls

○ pop ● lifeExp ○ gdpPercap

country	pop	continent	lifeExp	gdpPercap
Afghanistan	31889923	Asia	43.828	974.5803384
Albania	3600523	Europe	76.423	5937.029525999999
Algeria	33333216	Africa	72.301	6223.367465
Angola	12420476	Africa	42.731	4797.231267
Argentina	40301927	Americas	75.32	12779.37964
Australia	20434176	Oceania	81.235	34435.367439999995
Austria	8199783	Europe	79.829	36126.4927
Bahrain	708573	Asia	75.635	29796.04834
Bangladesh	150448339	Asia	64.062	1391.253792
Belgium	10392226	Europe	79.441	33692.60508
Benin	8078314	Africa	56.728	1441.284873
Bolivia	9119152	Americas	65.554	3822.137084

« ‹ 1 / 12 › »

(https://dash.plotly.com/tutorial?utm_medium=graphing_libraries&utm_content=python_footer)

JOIN OUR MAILING LIST

Sign up to stay in the loop with all things Plotly — from Dash Club to product updates, webinars, and more!

SUBSCRIBE (HTTPS://GO.PLOT.LY/SUBSCRIPTION)

Products

Dash (https://plotly.com/dash/)
Consulting and Training (https://plotly.com/consulting-and-oem/)

Pricing

Enterprise Pricing (https://plotly.com/get-pricing/)

About Us

Careers (https://plotly.com/careers)
Resources (https://plotly.com/resources/)
Blog (https://medium.com/@plotlygraphs)

Support

Community Support (https://community.plot.ly/)
Documentation (https://plotly.com/graphing-libraries)

Terms of Service (https://community.plotly.com/tos) Privacy Policy (https://plotly.com/privacy/)