plotly | Graphing Libraries (https://plotly.com/)(/graphing-libraries/)

utm_campaign=studio_cloud_launch&utm_content=sidebar)

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

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

3D Cone Plots in Python

How to make 3D Cone plots in Python with Plotly.

> Plotly Studio: Transform any dataset into an interactive data application in minutes with AI. Sign up for early access now. (https://plotly.com/studio/?utm_medium=graphing_libraries&utm_campaign=studio_early_access&utm_content=sidebar)

A cone plot is the 3D equivalent of a 2D quiver plot (/python/quiver-plots/), i.e., it represents a 3D vector field using cones to represent the direction and norm of the vectors. 3-D coordinates are given by x, y and z, and the coordinates of the vector field by u, v and w.
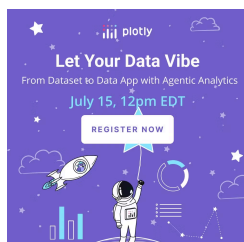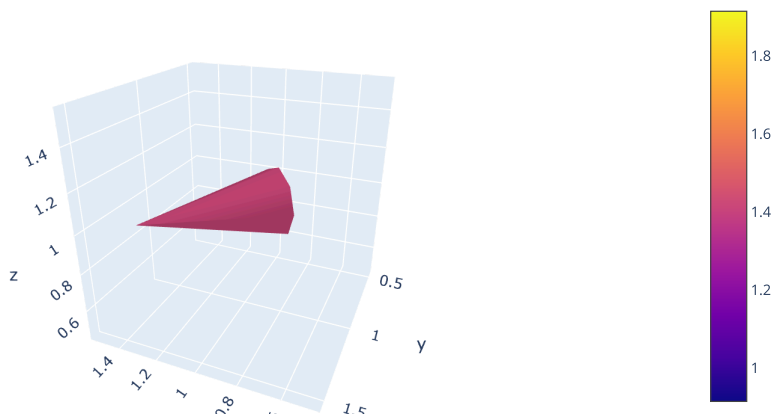
Basic 3D Cone

```python
import plotly.graph_objects as go

fig = go.Figure(data=go.Cone(x=[1], y=[1], z=[1], u=[1], v=[1], w=[0]))

fig.update_layout(scene_camera_eye=dict(x=-0.76, y=1.8, z=0.92))

fig.show()
```
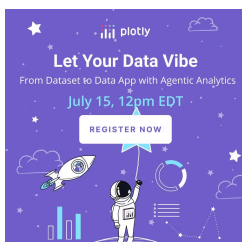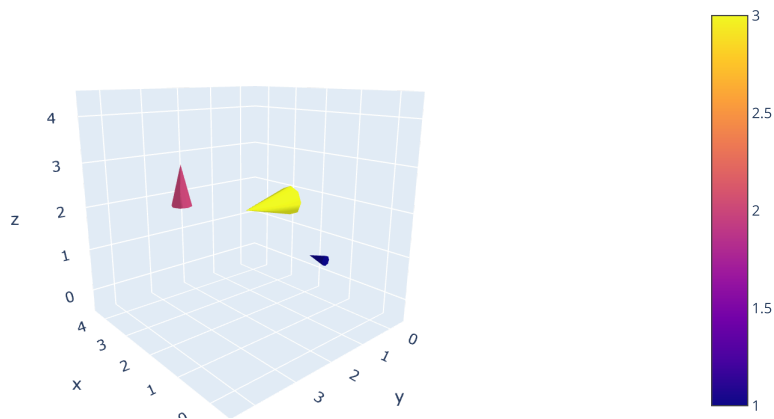
Multiple 3D Cones

```python
import plotly.graph_objects as go

fig = go.Figure(data=go.Cone(
    x=[1, 2, 3],
    y=[1, 2, 3],
    z=[1, 2, 3],
    u=[1, 0, 0],
    v=[0, 3, 0],
    w=[0, 0, 2],
    sizemode="absolute",
    sizeref=2,
    anchor="tip"))

fig.update_layout(
      scene=dict(domain_x=[0, 1],
                  camera_eye=dict(x=-1.57, y=1.36, z=0.58)))

fig.show()
```

3D Cone Lighting

```python
import plotly.graph_objects as go

fig = go.Figure()
fig.add_trace(go.Cone(x=[1,] * 3, name="base"))
fig.add_trace(go.Cone(x=[2,] * 3, opacity=0.3, name="opacity:0.3"))
fig.add_trace(go.Cone(x=[3,] * 3, lighting_ambient=0.3, name="lighting.ambient:0.3"))
fig.add_trace(go.Cone(x=[4,] * 3, lighting_diffuse=0.3, name="lighting.diffuse:0.3"))
fig.add_trace(go.Cone(x=[5,] * 3, lighting_specular=2, name="lighting.specular:2"))
fig.add_trace(go.Cone(x=[6,] * 3, lighting_roughness=1, name="lighting.roughness:1"))
fig.add_trace(go.Cone(x=[7,] * 3, lighting_fresnel=2, name="lighting.fresnel:2"))
fig.add_trace(go.Cone(x=[8,] * 3, lightposition=dict(x=0, y=0, z=1e5),
                      name="lighting.position x:0,y:0,z:1e5"))

fig.update_traces(y=[1, 2, 3], z=[1, 1, 1],
                  u=[1, 2, 3], v=[1, 1, 2], w=[4, 4, 1],
                  hoverinfo="u+v+w+name",
                  showscale=False)

fig.update_layout(scene=dict(aspectmode="data",
                             camera_eye=dict(x=0.05, y=-2.6, z=2)),
                  margin=dict(t=0, b=0, l=0, r=0))


fig.show()
```
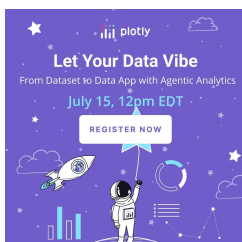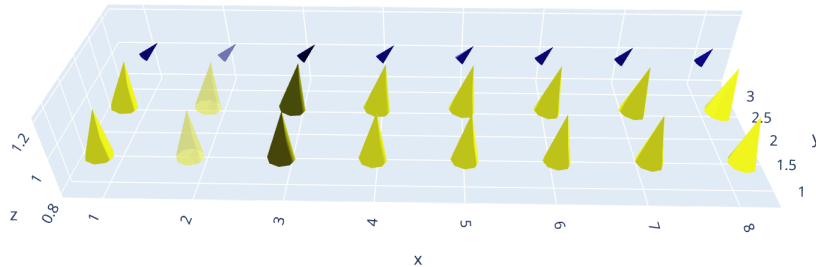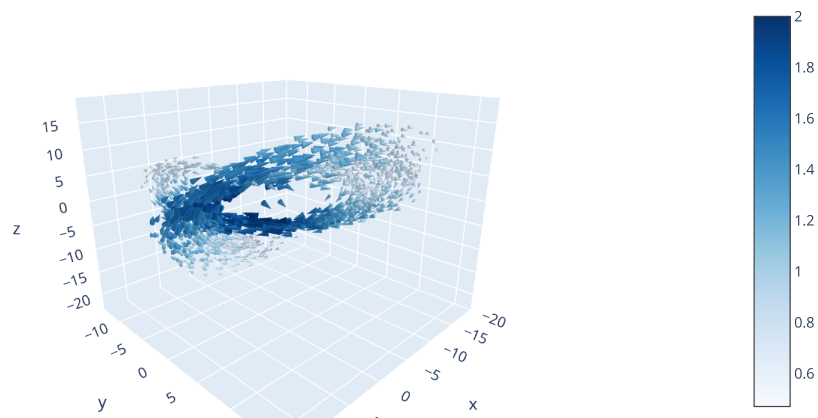
3D Cone Vortex

```python
import plotly.graph_objects as go
import pandas as pd

df = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/vortex.csv")

fig = go.Figure(data = go.Cone(
    x=df['x'],
    y=df['y'],
    z=df['z'],
    u=df['u'],
    v=df['v'],
    w=df['w'],
    colorscale='Blues',
    sizemode="absolute",
    sizeref=40))

fig.update_layout(scene=dict(aspectratio=dict(x=1, y=1, z=0.8),
                             camera_eye=dict(x=1.2, y=1.2, z=0.6)))

fig.show()
```
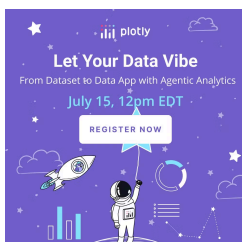

Sizemode

Earlier examples use sizemode="absolute" when adjusting the cone size scaling with sizeref. sizemode also supports raw(new in 5.21) and scaled.
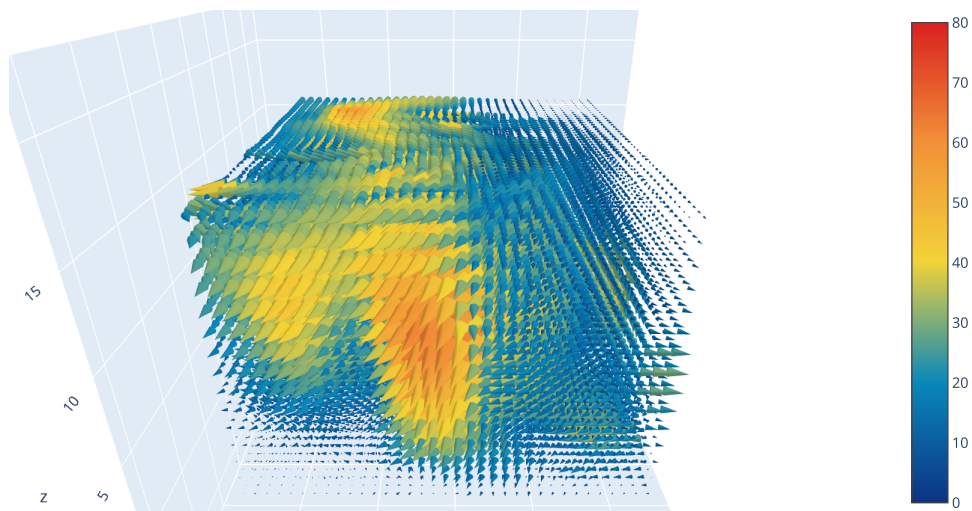
```python
import plotly.graph_objects as go
import pandas as pd

df = pd.read_csv(
    "https://raw.githubusercontent.com/plotly/datasets/master/cone_plot_data.csv"
)

fig = go.Figure(
    data=go.Cone(
        x=df["x"],
        y=df["y"],
        z=df["z"],
        u=df["u"],
        v=df["v"],
        w=df["w"],
        sizemode="raw",
        sizeref=0.1,
        colorscale="Portland",
        cmin=0,
        cmax=80,
        hoverinfo="u+v+w+text",
        text="-> wind <-",
    ),
    layout=dict(
        width=900, height=600, scene=dict(camera=dict(eye=dict(x=1.2, y=0, z=0.6)))
    ),
)


fig.show()
```


Reference

See [https://plotly.com/python/reference/ (https://plotly.com/python/reference/)](https://plotly.com/python/reference/) for more information and chart attribute options!
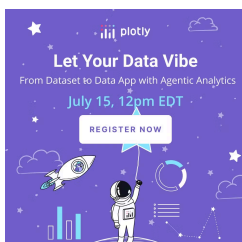
What About Dash?

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:

```python
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
```


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