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

:a-

utm_campaign=studio_cloud_launch&utm_content=sidebar)

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

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

3D Axes in Python

How to format axes of 3d plots in Python with Plotly.

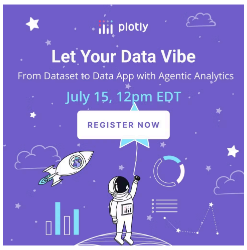
> Plotly Studio: Transform any dataset into an interactive data application in minutes with AI. Sign up for early access now. (https://plotly.com/studio/?utm_medium=graphing_libraries&utm_campaign=studio_early_access&utm_content=sidebar)

Range of axes

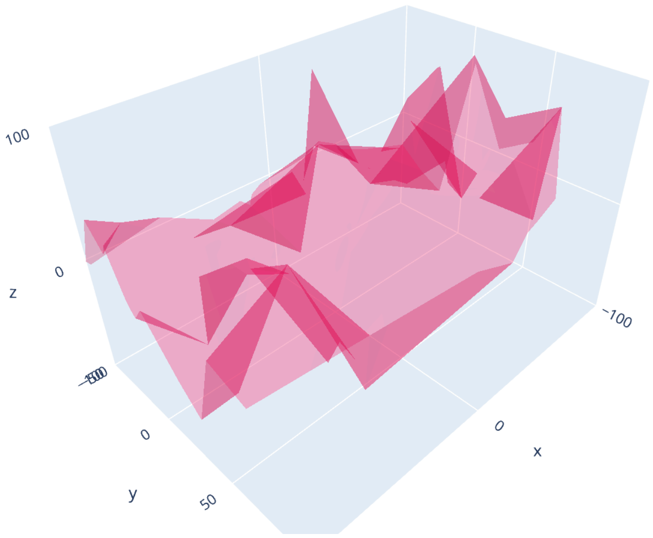
3D figures have an attribute in `layout` called `scene`, which contains attributes such as `xaxis`, `yaxis` and `zaxis` parameters, in order to set the range, title, ticks, color etc. of the axes.

For creating 3D charts, see this page (https://plotly.com/python/3d-charts/).

Set `range` on an axis to manually configure a range for that axis. If you don't set `range`, it's automatically calculated. In this example, we set a `range` on `xaxis`, `yaxis`, and `zaxis`.



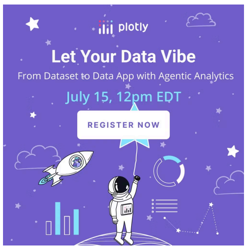
```
import plotly.graph_objects as go
import numpy as np
np.random.seed(1)

N = 70

fig = go.Figure(data=[go.Mesh3d(x=(70*np.random.randn(N)),
                   y=(55*np.random.randn(N)),
                   z=(40*np.random.randn(N)),
                   opacity=0.5,
                   color='rgba(244,22,100,0.6)'
                  )])

fig.update_layout(
    scene = dict(
        xaxis = dict(nticks=4, range=[-100,100],),
                     yaxis = dict(nticks=4, range=[-50,100],),
                     zaxis = dict(nticks=4, range=[-100,100],),),
    width=700,
    margin=dict(r=20, l=10, b=10, t=10))

fig.show()
```



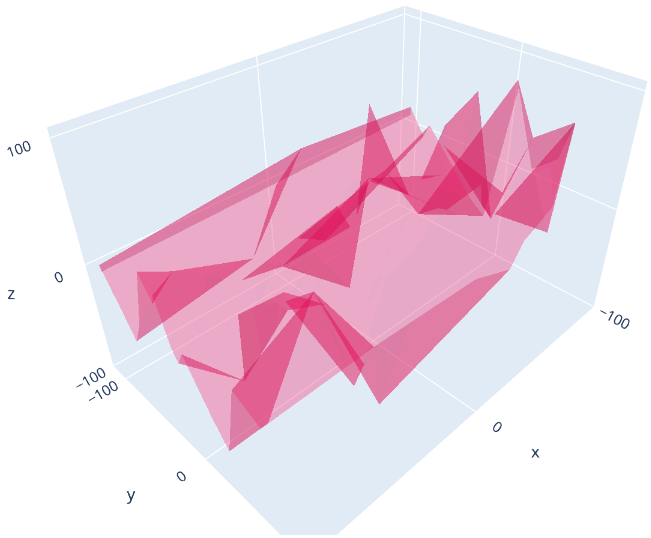
Setting only a Lower or Upper Bound for Range

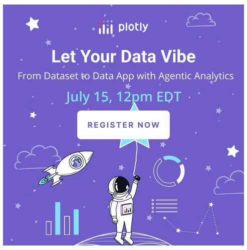
New in 5.17

You can also set just a lower or upper bound for range. In this case, autorange is used for the other bound. In this example, we apply autorange to the lower bound of the yaxis and the upper bound of zaxis by setting them to None.



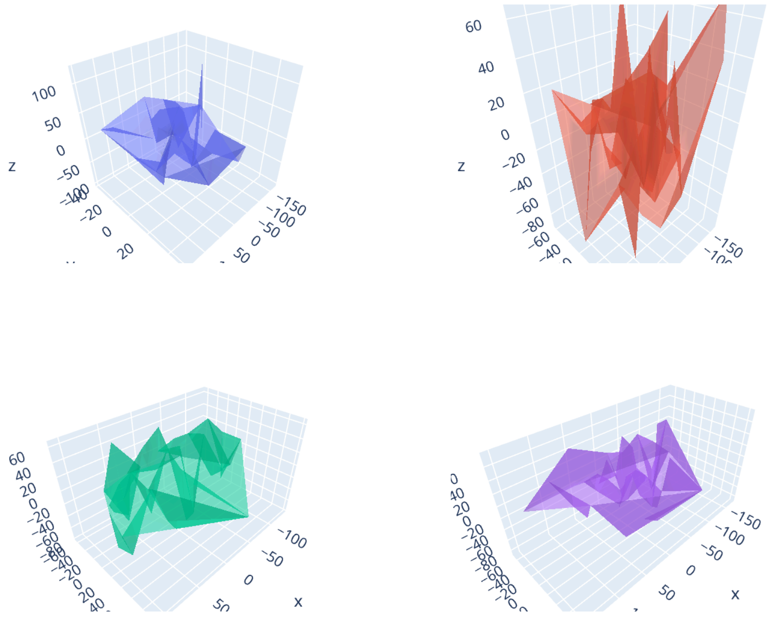
```
import plotly.graph_objects as go
import numpy as np
np.random.seed(1)

N = 70

fig = go.Figure(data=[go.Mesh3d(x=(70*np.random.randn(N)),
                   y=(55*np.random.randn(N)),
                   z=(40*np.random.randn(N)),
                   opacity=0.5,
                   color='rgba(244,22,100,0.6)'
                  )])

fig.update_layout(
    scene = dict(
        xaxis = dict(nticks=4, range=[-100,100],),
                     yaxis = dict(nticks=4, range=[None, 100],),
                     zaxis = dict(nticks=4, range=[-100, None],),),
    width=700,
    margin=dict(r=20, l=10, b=10, t=10))

fig.show()
```

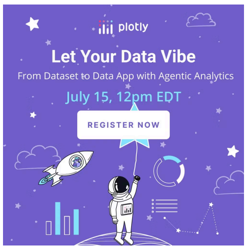




Fixed Ratio Axes

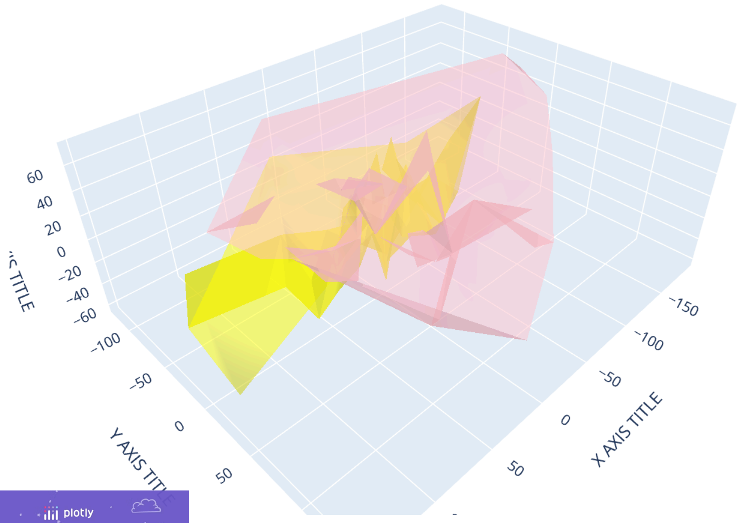
```
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import numpy as np

N = 50

fig = make_subplots(rows=2, cols=2,
                    specs=[[{'is_3d': True}, {'is_3d': True}],
                           [{'is_3d': True}, {'is_3d': True}]],
                    print_grid=False)
for i in [1,2]:
    for j in [1,2]:
        fig.add_trace(
            go.Mesh3d(
                x=(60*np.random.randn(N)),
                y=(25*np.random.randn(N)),
                z=(40*np.random.randn(N)),
                opacity=0.5,
              ),
            row=i, col=j)

fig.update_layout(width=700, margin=dict(r=10, l=10, b=10, t=10))
# fix the ratio in the top left subplot to be a cube
fig.update_layout(scene_aspectmode='cube')
# manually force the z-axis to appear twice as big as the other two
fig.update_layout(scene2_aspectmode='manual',
                  scene2_aspectratio=dict(x=1, y=1, z=2))
# draw axes in proportion to the proportion of their ranges
fig.update_layout(scene3_aspectmode='data')
# automatically produce something that is well proportioned using 'data' as the default
fig.update_layout(scene4_aspectmode='auto')
fig.show()
```





Set Axes Title

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

# Define random surface
N = 50
fig = go.Figure()
fig.add_trace(go.Mesh3d(x=(60*np.random.randn(N)),
                   y=(25*np.random.randn(N)),
                   z=(40*np.random.randn(N)),
                   opacity=0.5,
                   color='yellow'
                  ))
fig.add_trace(go.Mesh3d(x=(70*np.random.randn(N)),
                   y=(55*np.random.randn(N)),
                   z=(30*np.random.randn(N)),
                   opacity=0.5,
                   color='pink'
                  ))

fig.update_layout(scene = dict(
                      xaxis=dict(
                          title=dict(
                              text='X AXIS TITLE'
                          )
                      ),
                      yaxis=dict(
                          title=dict(
                              text='Y AXIS TITLE'
                          )
                      ),
                      zaxis=dict(
                          title=dict(
                              text='Z AXIS TITLE'
                          )
                      ),
                    ),
                    width=700,
                    margin=dict(r=20, b=10, l=10, t=10))

fig.show()
```


'S TITLE
Y AXIS TIT
X AXIS TITLE