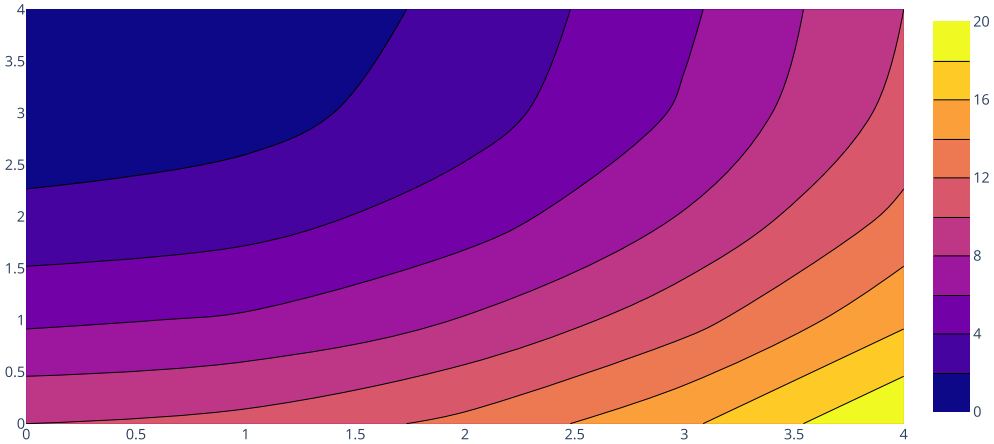
Contour Plots in Python

How to make Contour plots in Python with Plotly.

Basic Contour Plot

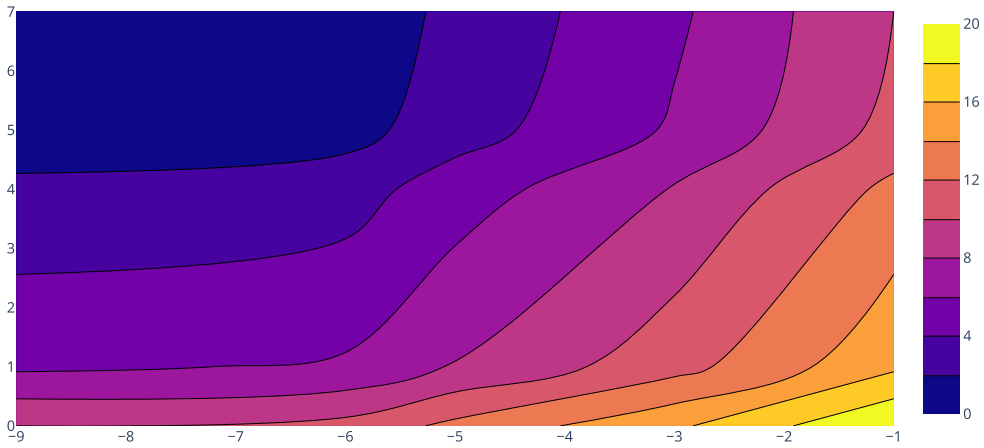
A 2D contour plot shows the contour lines (https://en.wikipedia.org/wiki/Contour_line) of a 2D numerical array z, i.e. interpolated lines of isovalues of z.

```
import plotly.graph_objects as go

fig = go.Figure(data =
    go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]]
    ))
fig.show()
```



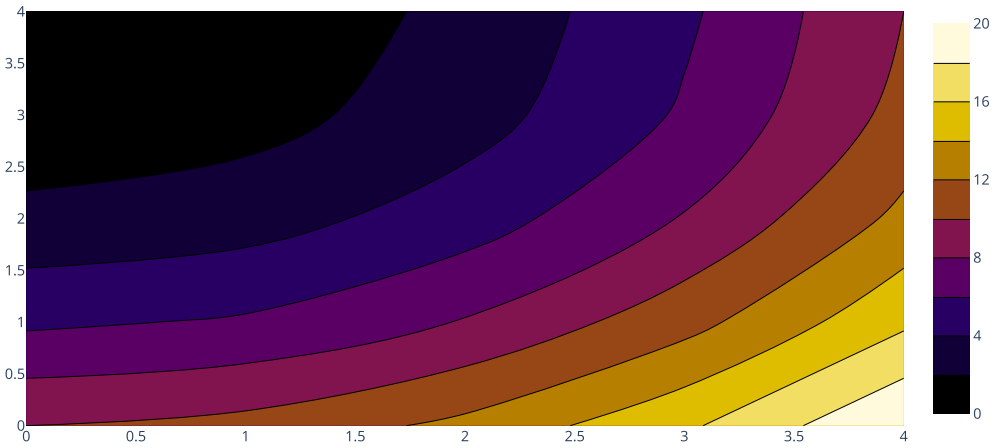
Setting X and Y Coordinates in a Contour Plot

```
import plotly.graph_objects as go

fig = go.Figure(data =
    go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]],
        x=[-9, -6, -5 , -3, -1], # horizontal axis
        y=[0, 1, 4, 5, 7] # vertical axis
    ))
fig.show()
```



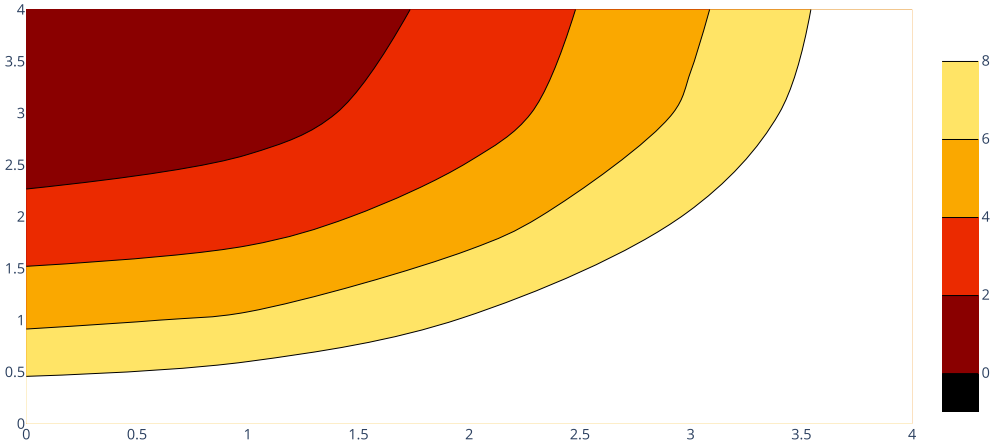
Colorscale for Contour Plot

```
import plotly.graph_objects as go

fig = go.Figure(data =
     go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]],
        colorscale='Electric',
    ))
fig.show()
```



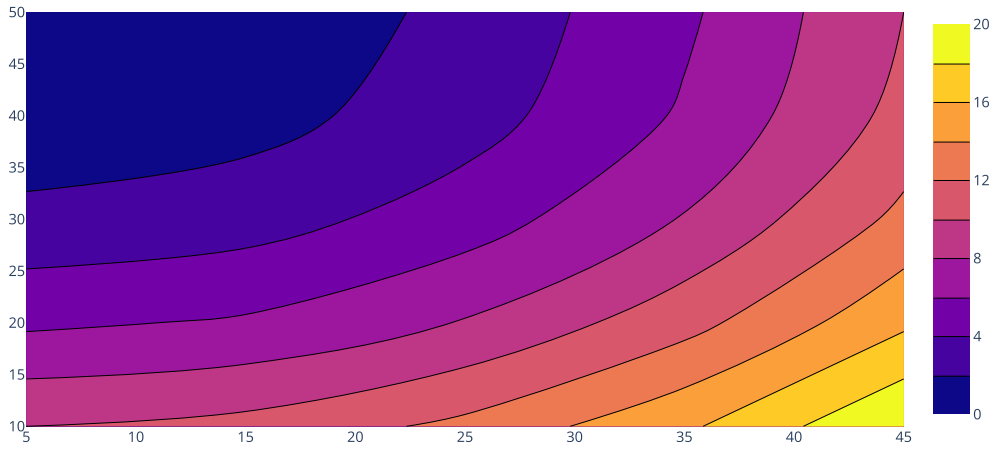
Customizing Size and Range of a Contour Plot's Contours

```
import plotly.graph_objects as go

fig = go.Figure(data =
    go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]],
        colorscale='Hot',
        contours=dict(
            start=0,
            end=8,
            size=2,
        ),
    ))

fig.show()
```



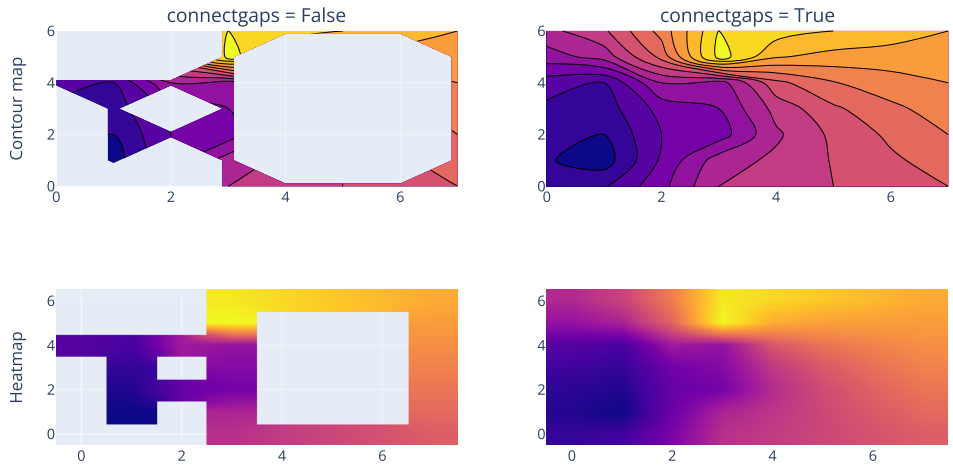
Customizing Spacing Between X and Y Axis Ticks

```
import plotly.graph_objects as go

fig = go.Figure(data =
    go.Contour(
        z= [[10, 10.625, 12.5, 15.625, 20],
            [5.625, 6.25, 8.125, 11.25, 15.625],
            [2.5, 3.125, 5., 8.125, 12.5],
            [0.625, 1.25, 3.125, 6.25, 10.625],
            [0, 0.625, 2.5, 5.625, 10]],
        dx=10,
        x0=5,
        dy=10,
        y0=10,
    )
)

fig.show()
```



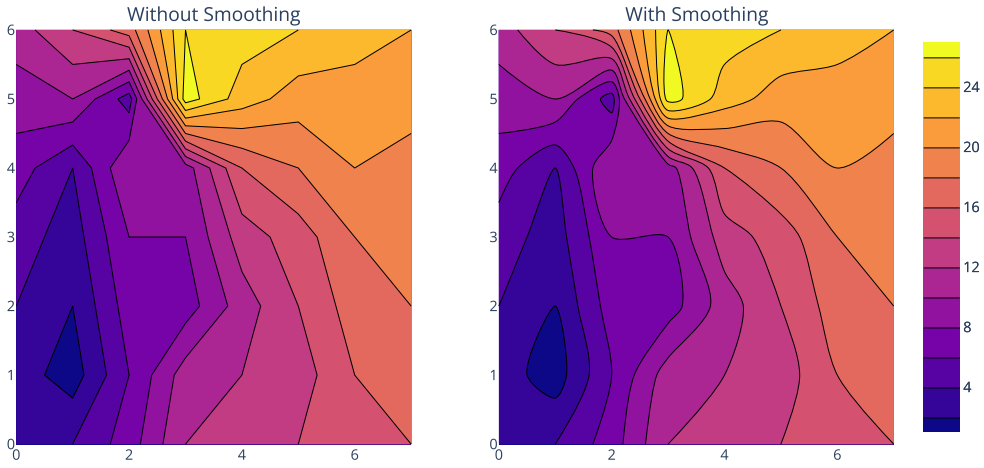
Connect the Gaps Between None Values in the Z Matrix

```
import plotly.graph_objs as go
from plotly.subplots import make_subplots

fig = make_subplots(rows=2, cols=2, subplot_titles=('connectgaps = False',
                                                    'connectgaps = True'))
z = [[None, None, None, 12, 13, 14, 15, 16],
     [None, 1, None, 11, None, None, None, 17],
     [None, 2, 6, 7, None, None, None, 18],
     [None, 3, None, 8, None, None, None, 19],
     [5, 4, 10, 9, None, None, None, 20],
     [None, None, None, 27, None, None, None, 21],
     [None, None, None, 26, 25, 24, 23, 22]]

fig.add_trace(go.Contour(z=z, showscale=False), 1, 1)
fig.add_trace(go.Contour(z=z, showscale=False, connectgaps=True), 1, 2)
fig.add_trace(go.Heatmap(z=z, showscale=False, zsmooth='best'), 2, 1)
fig.add_trace(go.Heatmap(z=z, showscale=False, connectgaps=True, zsmooth='best'), 2, 2)

fig['layout']['yaxis1'].update(title=dict(text='Contour map'))
fig['layout']['yaxis3'].update(title=dict(text='Heatmap'))

fig.show()
```



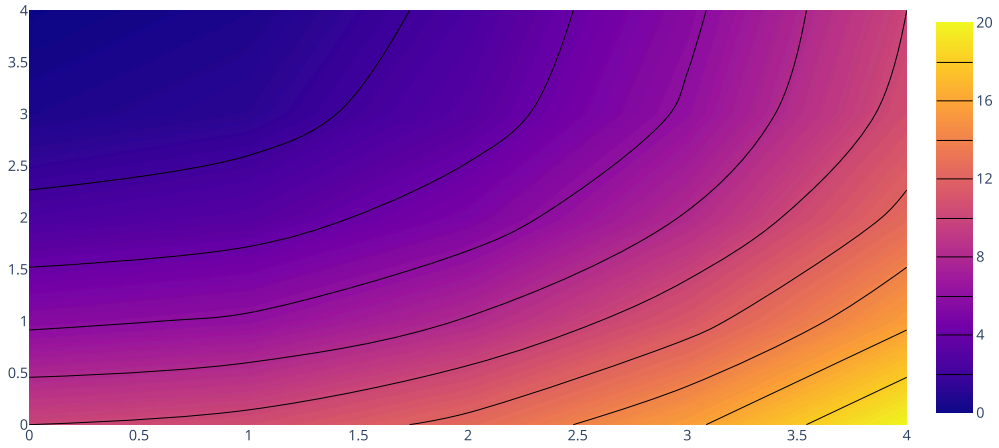
Smoothing the Contour lines

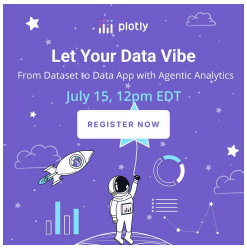
```
import plotly.graph_objects as go
from plotly.subplots import make_subplots
import numpy as np

z =   [[2, 4, 7, 12, 13, 14, 15, 16],
       [3, 1, 6, 11, 12, 13, 16, 17],
       [4, 2, 7, 7, 11, 14, 17, 18],
       [5, 3, 8, 8, 13, 15, 18, 19],
       [7, 4, 10, 9, 16, 18, 20, 19],
       [9, 10, 5, 27, 23, 21, 21, 21],
       [11, 14, 17, 26, 25, 24, 23, 22]]

fig = make_subplots(rows=1, cols=2,
                    subplot_titles=('Without Smoothing', 'With Smoothing'))

fig.add_trace(go.Contour(z=z, line_smoothing=0), 1, 1)
fig.add_trace(go.Contour(z=z, line_smoothing=0.85), 1, 2)

fig.show()
```



Smooth Contour Coloring

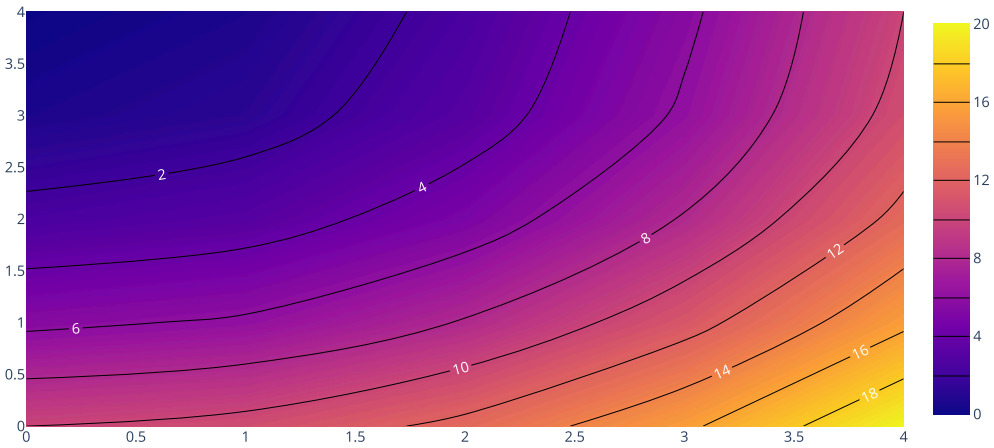
```
import plotly.graph_objects as go

fig = go.Figure(data=
    go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]],
        # heatmap gradient coloring is applied between each contour level
        contours_coloring='heatmap' # can also be 'lines', or 'none'
    )
)

fig.show()
```





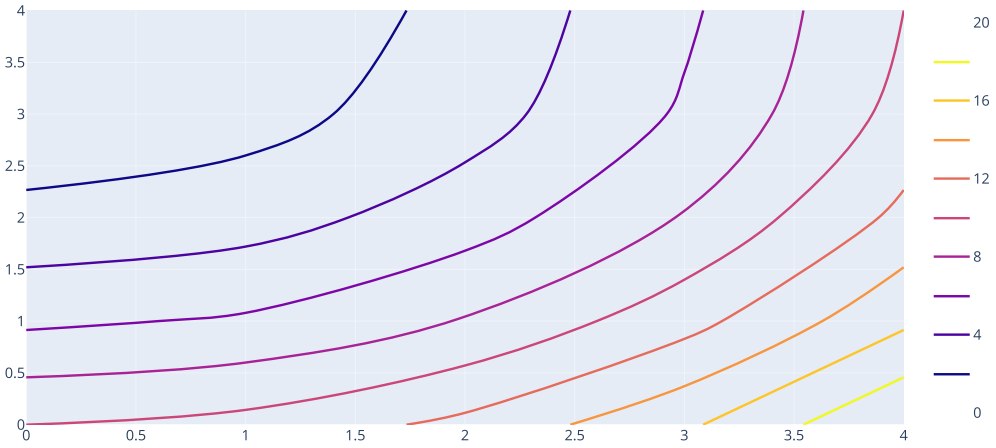
Contour Line Labels

```
import plotly.graph_objects as go

fig = go.Figure(data=
    go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]],
        contours=dict(
            coloring ='heatmap',
            showlabels = True, # show labels on contours
            labelfont = dict( # label font properties
                size = 12,
                color = 'white',
            )
        )))

fig.show()
```



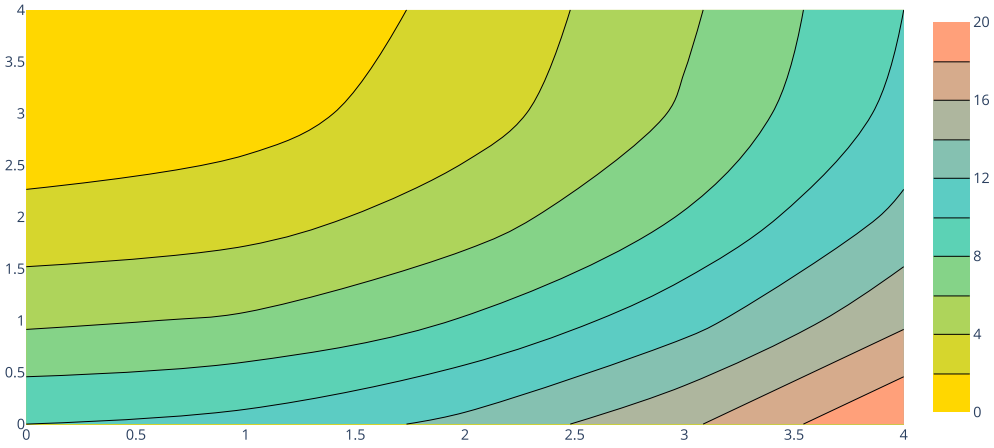
Contour Lines

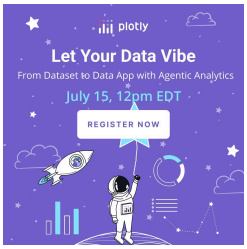
```
import plotly.graph_objects as go

fig = go.Figure(data=
    go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]],
        contours_coloring='lines',
        line_width=2,
    )
)

fig.show()
```



Custom Contour Plot Colorscale

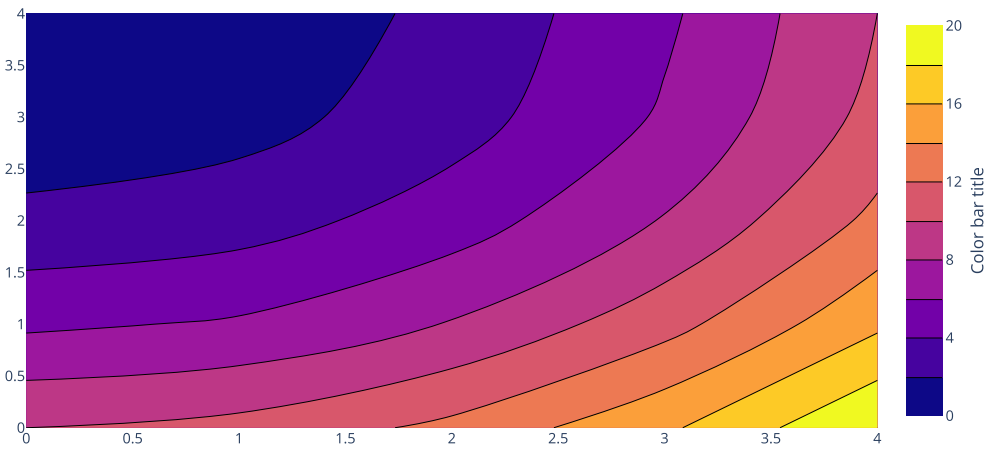
```
import plotly.graph_objects as go

# Valid color strings are CSS colors, rgb or hex strings
colorscale = [[0, 'gold'], [0.5, 'mediumturquoise'], [1, 'lightsalmon']]

fig = go.Figure(data =
    go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]],
        colorscale=colorscale)
)

fig.show()
```





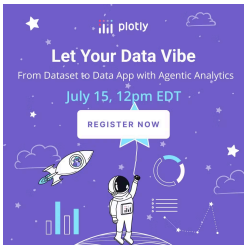
Color Bar Title

```
import plotly.graph_objects as go

fig = go.Figure(data=
    go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]],
        colorbar=dict(
            title=dict(
                text='Color bar title', # title here
                side='right',
                font=dict(
                    size=14,
                    family='Arial, sans-serif')
                )
            ),
        ))

fig.show()
```



Color Bar Size for Contour Plots

In the example below, both the thickness (given here in pixels) and the length (given here as a fraction of the plot height) are set.



ntour

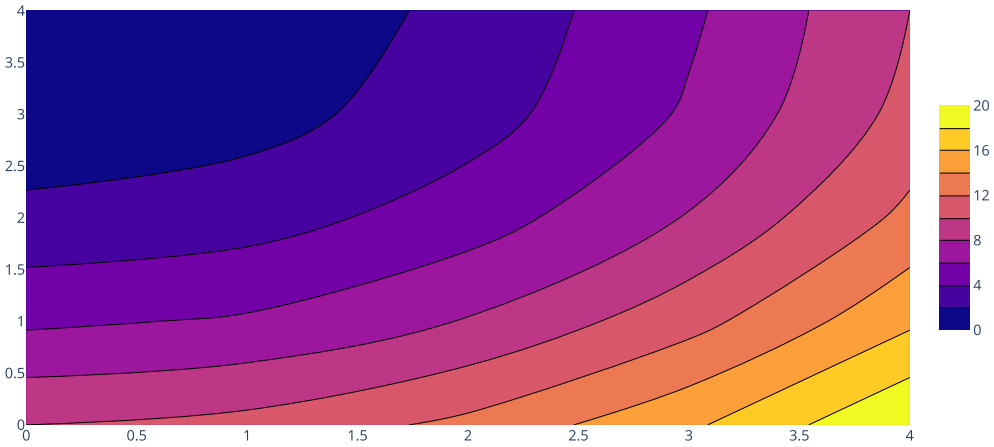
ntour

l Y Axis

alues in

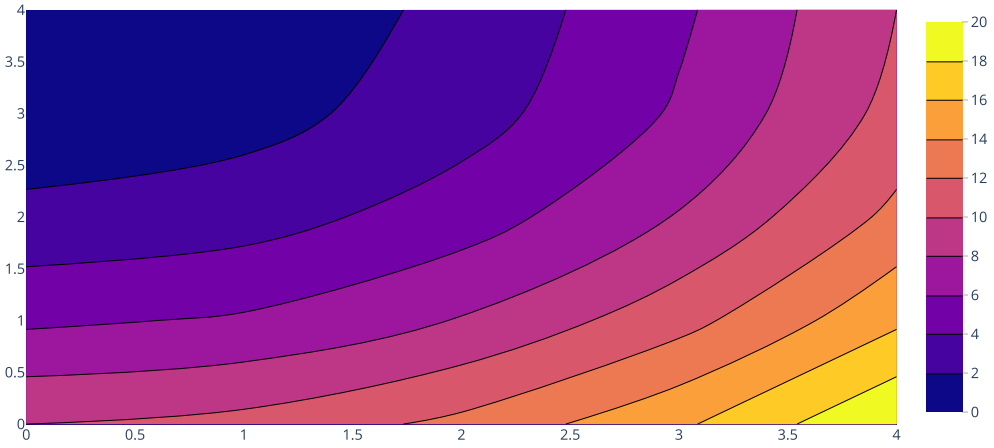
lots

```
import plotly.graph_objects as go

fig = go.Figure(data=
    go.Contour(
        z=[[10, 10.625, 12.5, 15.625, 20],
           [5.625, 6.25, 8.125, 11.25, 15.625],
           [2.5, 3.125, 5., 8.125, 12.5],
           [0.625, 1.25, 3.125, 6.25, 10.625],
           [0, 0.625, 2.5, 5.625, 10]],
        colorbar=dict(
            thickness=25,
            thicknessmode='pixels',
            len=0.6,
            lenmode='fraction',
            outlinewidth=0
        )
    ))

fig.show()
```



Styling Color Bar Ticks for Contour Plots

```
import plotly.graph_objects as go

fig = go.Figure(data =
         go.Contour(
           z=[[10, 10.625, 12.5, 15.625, 20],
              [5.625, 6.25, 8.125, 11.25, 15.625],
              [2.5, 3.125, 5., 8.125, 12.5],
              [0.625, 1.25, 3.125, 6.25, 10.625],
              [0, 0.625, 2.5, 5.625, 10]],
           colorbar=dict(nticks=10, ticks='outside',
                         ticklen=5, tickwidth=1,
                         showticklabels=True,
                         tickangle=0, tickfont_size=12)
            ))

fig.show()
```



Reference

See https://plotly.com/python/reference/contour/ (https://plotly.com/python/reference/contour/) 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:

```
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
```