Ternary contours in Python

How to make Ternary Contour Plots in Python with plotly

Ternary contour plots

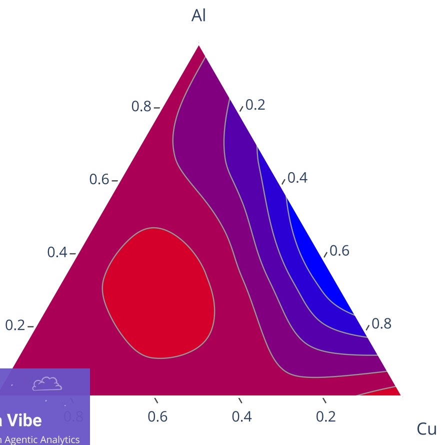
A ternary contour plots represents isovalue lines of a quantity defined inside a ternary diagram (https://en.wikipedia.org/wiki/Ternary_plot), i.e. as a function of three variables which sum is constant. Coordinates of the ternary plot often correspond to concentrations of three species, and the quantity represented as contours is some property (e.g., physical, chemical, thermodynamical) varying with the composition.

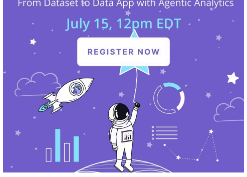
For ternary contour plots, use the figure factory (/python/figure-factories/) called create_ternary_contour. The figure factory interpolates between given data points in order to compute the contours.

Below we represent an example from metallurgy, where the mixing enthalpy is represented as a contour plot for aluminum-copper-yttrium (Al-Cu-Y) alloys.

Simple ternary contour plot with plotly

```
import plotly.figure_factory as ff
import numpy as np
Al = np.array([0. , 0. , 0., 0., 1./3, 1./3, 1./3, 2./3, 2./3, 1.])
Cu = np.array([0., 1./3, 2./3, 1., 0., 1./3, 2./3, 0., 1./3, 0.])
Y = 1 - Al - Cu
# synthetic data for mixing enthalpy
# See https://pycalphad.org/docs/latest/examples/TernaryExamples.html
enthalpy = (Al - 0.01) * Cu * (Al - 0.52) * (Cu - 0.48) * (Y - 1)**2
fig = ff.create_ternary_contour(np.array([Al, Y, Cu]), enthalpy,
                                pole_labels=['Al', 'Y', 'Cu'],
                                interp_mode='cartesian')
fig.show()
```





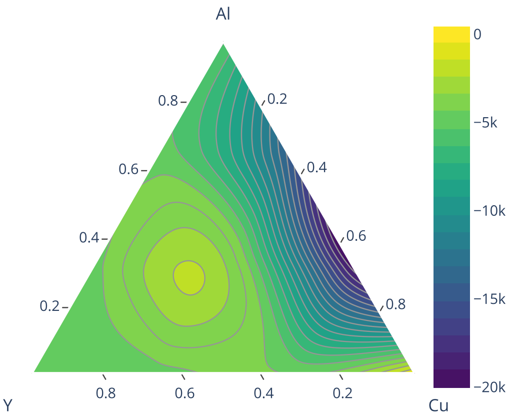
Customized ternary contour plot

ɔtly

ts

```
import plotly.figure_factory as ff
import numpy as np
Al = np.array([0. , 0. , 0., 0., 1./3, 1./3, 1./3, 2./3, 2./3, 1.])
Cu = np.array([0., 1./3, 2./3, 1., 0., 1./3, 2./3, 0., 1./3, 0.])
Y = 1 - Al - Cu
# synthetic data for mixing enthalpy
# See https://pycalphad.org/docs/latest/examples/TernaryExamples.html
enthalpy = 2.e6 * (Al - 0.01) * Cu * (Al - 0.52) * (Cu - 0.48) * (Y - 1)**2 - 5000
fig = ff.create_ternary_contour(np.array([Al, Y, Cu]), enthalpy,
                                pole_labels=['Al', 'Y', 'Cu'],
                                interp_mode='cartesian',
                                ncontours=20,
                                colorscale='Viridis',
                                showscale=True,
                                title=dict(
                                  text='Mixing enthalpy of ternary alloy'
                                ))
fig.show()
```

Mixing enthalpy of ternary alloy

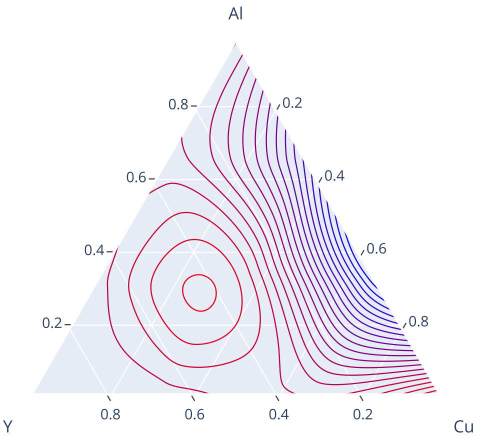


Ternary contour plot with lines only

```
import plotly.figure_factory as ff
import numpy as np
Al = np.array([0. , 0. , 0., 0., 1./3, 1./3, 1./3, 2./3, 2./3, 1.])
Cu = np.array([0., 1./3, 2./3, 1., 0., 1./3, 2./3, 0., 1./3, 0.])
Y = 1 - Al - Cu
# synthetic data for mixing enthalpy
# See https://pycalphad.org/docs/latest/examples/TernaryExamples.html
enthalpy = 2.e6 * (Al - 0.01) * Cu * (Al - 0.52) * (Cu - 0.48) * (Y - 1)**2 - 5000
fig = ff.create_ternary_contour(np.array([Al, Y, Cu]), enthalpy,
                                pole_labels=['Al', 'Y', 'Cu'],
                                interp_mode='cartesian',
                                ncontours=20,
                                coloring='lines')
fig.show()
```

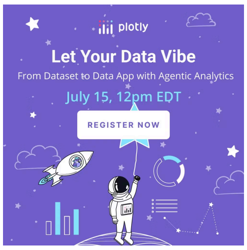
ɔtly

ts



Ternary contour plot with data points

With showmarkers=True, data points used to compute the contours are also displayed. They are best visualized for contour lines (no solid coloring). At the moment data points lying on the edges of the diagram are not displayed, this will be improved in future versions.

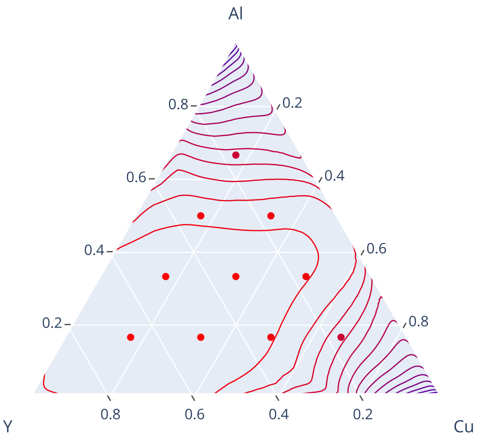


ɔtly

ts

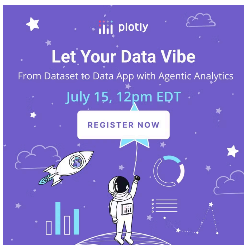
```
import plotly.figure_factory as ff
import numpy as np
Al, Cu = np.mgrid[0:1:7j, 0:1:7j]
Al, Cu = Al.ravel(), Cu.ravel()
mask = Al + Cu <= 1
Al, Cu = Al[mask], Cu[mask]
Y = 1 - Al - Cu

enthalpy = (Al - 0.5) * (Cu - 0.5) * (Y - 1)**2
fig = ff.create_ternary_contour(np.array([Al, Y, Cu]), enthalpy,
                                pole_labels=['Al', 'Y', 'Cu'],
                                ncontours=20,
                                coloring='lines',
                                showmarkers=True)
fig.show()
```



Interpolation mode

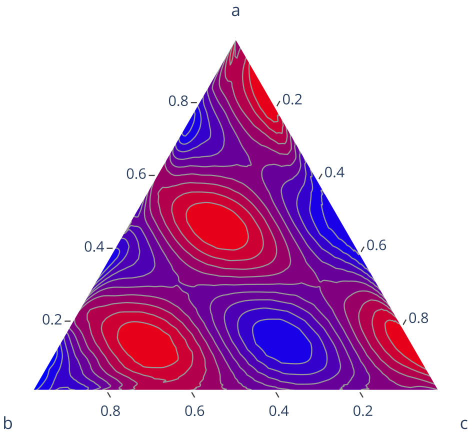
Two modes are available in order to interpolate between data points: interpolation in Cartesian space (interp_mode='cartesian') or interpolation using the isometric log-ratio transformation (https://link.springer.com/article/10.1023/A:1023818214614) (see also preprint (https://www.researchgate.net/profile/Leon_Parent2/post/What_is_the_best_approach_for_diagnosing_nutrient_disorders_and_formulating_fertilizer_recommendations/at interp_mode='ilr'. The ilr transformation preserves metrics in the simplex (https://en.wikipedia.org/wiki/Simplex) but is not defined on its edges.



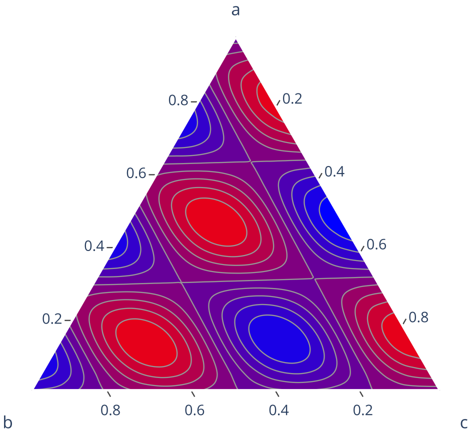
```
a, b = np.mgrid[0:1:20j, 0:1:20j]
mask = a + b <= 1
a, b = a[mask], b[mask]
coords = np.stack((a, b, 1 - a - b))
value = np.sin(3.2 * np.pi * (a + b)) + np.sin(3 * np.pi * (a - b))
fig = ff.create_ternary_contour(coords, value, ncontours=9)
fig.show()
```

otly

ts

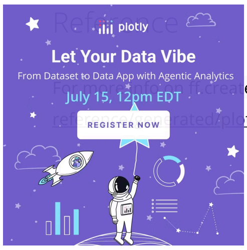


```
a, b = np.mgrid[0:1:20j, 0:1:20j]
mask = a + b <= 1
a, b = a[mask], b[mask]
coords = np.stack((a, b, 1 - a - b))
value = np.sin(3.2 * np.pi * (a + b)) + np.sin(3 * np.pi * (a - b))
fig = ff.create_ternary_contour(coords, value, interp_mode='cartesian',
                                ncontours=9)
fig.show()
```



Reference

For more info on ff.create_ternary_contour(), see the full function reference (https://plotly.com/python-api-reference/generated/plotly.figure_factory.create_ternary_contour.html)



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:

otly

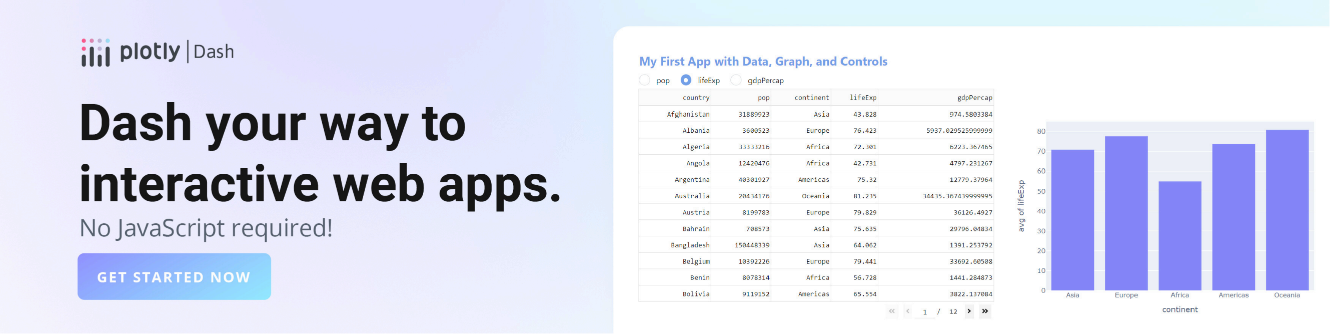
ts

```
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/)