Dendrograms in Python

How to make a dendrogram in Python with Plotly.

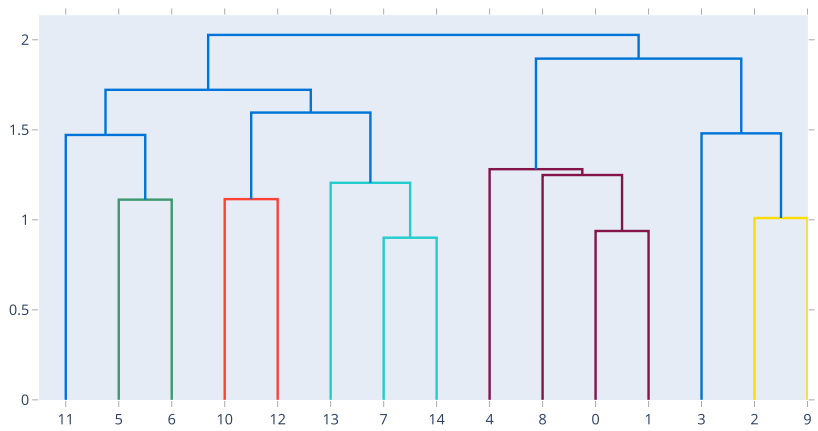
> Plotly Studio: Transform any dataset into an interactive data application in minutes with AI. Sign up for early access now. (https://plotly.com/studio/?utm_medium=graphing_libraries&utm_campaign=studio_early_access&utm_content=sidebar)

Basic Dendrogram

A dendrogram (https://en.wikipedia.org/wiki/Dendrogram) is a diagram representing a tree. The figure factory (/python/figure-factories/) called `create_dendrogram` performs hierarchical clustering (https://en.wikipedia.org/wiki/Hierarchical_clustering) on data and represents the resulting tree. Values on the tree depth axis correspond to distances between clusters.

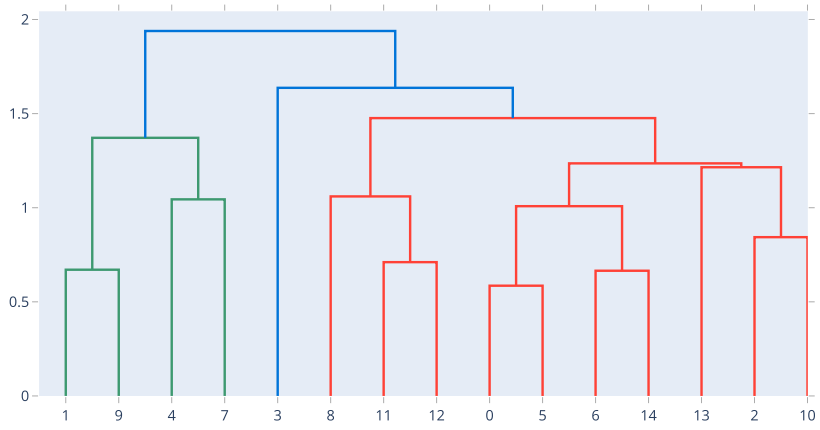
Dendrogram plots are commonly used in computational biology to show the clustering of genes or samples, sometimes in the margin of heatmaps.

```
import plotly.figure_factory as ff
import numpy as np
np.random.seed(1)

X = np.random.rand(15, 12) # 15 samples, with 12 dimensions each
fig = ff.create_dendrogram(X)
fig.update_layout(width=800, height=500)
fig.show()
```



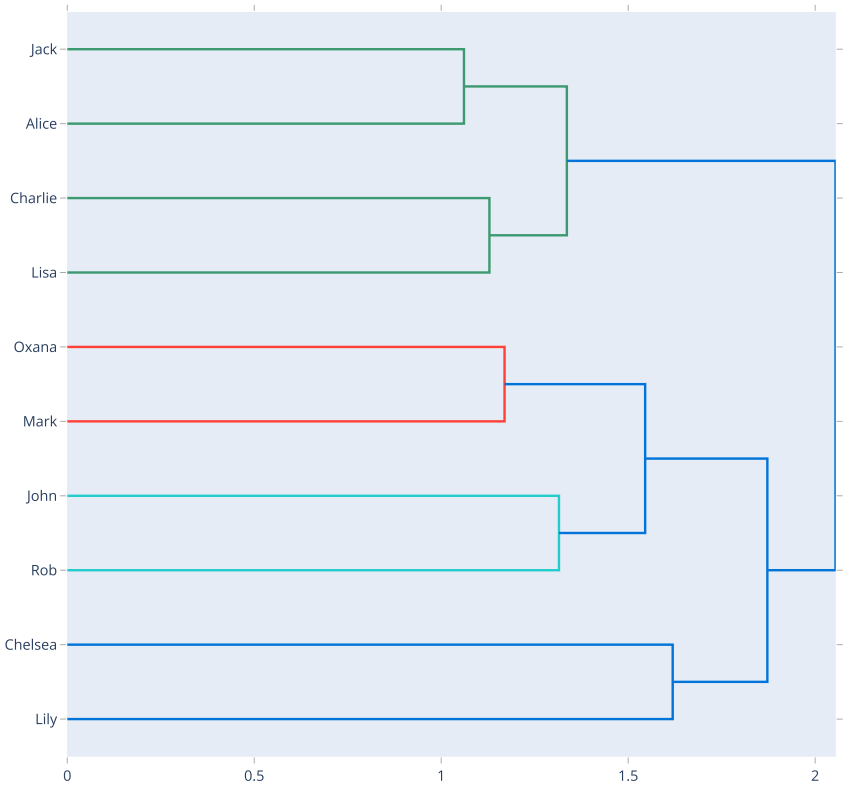
Set Color Threshold

```
import plotly.figure_factory as ff

import numpy as np

X = np.random.rand(15, 10) # 15 samples, with 10 dimensions each
fig = ff.create_dendrogram(X, color_threshold=1.5)
fig.update_layout(width=800, height=500)
fig.show()
```



Set Orientation and Add Labels

```
import plotly.figure_factory as ff

import numpy as np

X = np.random.rand(10, 12)
names = ['Jack', 'Oxana', 'John', 'Chelsea', 'Mark', 'Alice', 'Charlie', 'Rob', 'Lisa', 'Lily']
fig = ff.create_dendrogram(X, orientation='left', labels=names)
fig.update_layout(width=800, height=800)
fig.show()
```

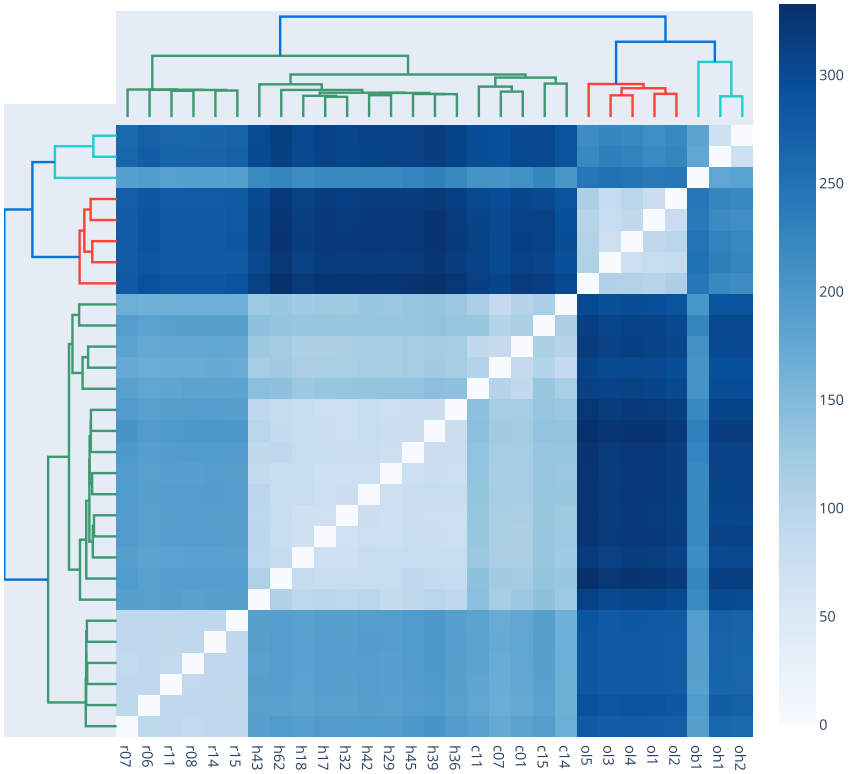


Plot a Dendrogram with a Heatmap

See also the Dash Bio demo (https://dash-bio.plotly.host/dash-clustergram/).

```
import plotly.graph_objects as go
import plotly.figure_factory as ff

import numpy as np
from scipy.spatial.distance import pdist, squareform


# get data
data = np.genfromtxt("http://files.figshare.com/2133304/ExpRawData_E_TABM_84_A_AFFY_44.tab",
                     names=True,usecols=tuple(range(1,30)),dtype=float, delimiter="\t")
data_array = data.view((float, len(data.dtype.names)))
data_array = data_array.transpose()
labels = data.dtype.names

# Initialize figure by creating upper dendrogram
fig = ff.create_dendrogram(data_array, orientation='bottom', labels=labels)
for i in range(len(fig['data'])):
    fig['data'][i]['yaxis'] = 'y2'

# Create Side Dendrogram
dendro_side = ff.create_dendrogram(data_array, orientation='right')
for i in range(len(dendro_side['data'])):
    dendro_side['data'][i]['xaxis'] = 'x2'

# Add Side Dendrogram Data to Figure
for data in dendro_side['data']:
    fig.add_trace(data)

# Create Heatmap
dendro_leaves = dendro_side['layout']['yaxis']['ticktext']
dendro_leaves = list(map(int, dendro_leaves))
data_dist = pdist(data_array)
heat_data = squareform(data_dist)
heat_data = heat_data[dendro_leaves,:]
heat_data = heat_data[:,dendro_leaves]

heatmap = [
    go.Heatmap(
        x = dendro_leaves,
        y = dendro_leaves,
        z = heat_data,
        colorscale = 'Blues'
    )
]

heatmap[0]['x'] = fig['layout']['xaxis']['tickvals']
heatmap[0]['y'] = dendro_side['layout']['yaxis']['tickvals']

# Add Heatmap Data to Figure
for data in heatmap:
    fig.add_trace(data)

# Edit Layout
fig.update_layout({'width':800, 'height':800,
                        'showlegend':False, 'hovermode': 'closest',
                        })
# Edit xaxis
fig.update_layout(xaxis={'domain': [.15, 1],
                                 'mirror': False,
                                 'showgrid': False,
                                 'showline': False,
                                 'zeroline': False,
                                 'ticks':""})
# Edit xaxis2
fig.update_layout(xaxis2={'domain': [0, .15],
                                  'mirror': False,
                                  'showgrid': False,
                                  'showline': False,
                                  'zeroline': False,
                                  'showticklabels': False,
                                  'ticks':""})


                                  axis={'domain': [0, .85],
                                 'mirror': False,
                                 'showgrid': False,
                                 'showline': False,
                                 'zeroline': False,
```

```
                                'showticklabels': False,
                                'ticks': ""
                        })
# Edit yaxis2
fig.update_layout(yaxis2={'domain':[.825, .975],
                                 'mirror': False,
                                 'showgrid': False,
                                 'showline': False,
                                 'zeroline': False,
                                 'showticklabels': False,
                                 'ticks':""})

# Plot!
fig.show()
```



Reference

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

```
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
```


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/)