Lines on Maps in Python

How to draw lines, great circles, and contours on maps in Python.

Below we show how to create geographical line plots using either Plotly Express with px.line_geo function or the lower-level go.Scattergeo object.

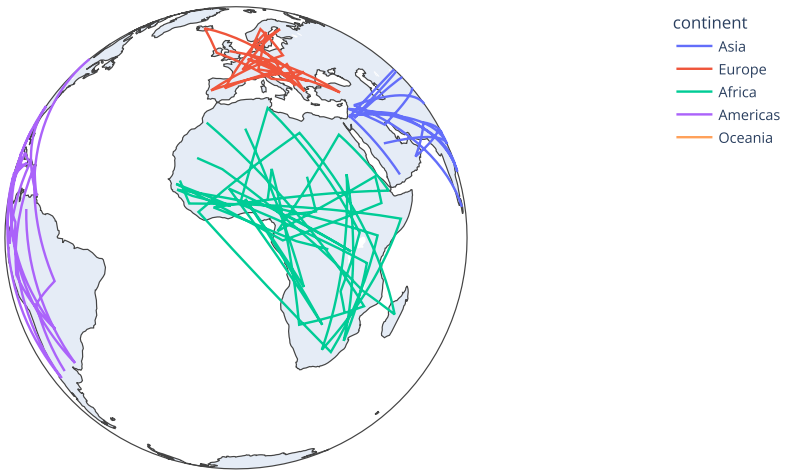
Base Map Configuration

Plotly figures made with Plotly Express (/python/plotly-express/) px.scatter_geo, px.line_geo or px.choropleth functions or containing go.Choropleth or go.Scattergeo graph objects (/python/graph-objects/) have a go.layout.Geo object which can be used to control the appearance of the base map (/python/map-configuration/) onto which data is plotted.

Lines on Maps with Plotly Express

Plotly Express (/python/plotly-express/) is the easy-to-use, high-level interface to Plotly, which operates on a variety of types of data (/python/px-arguments/) and produces easy-to-style figures (/python/styling-plotly-express/).

```
import plotly.express as px
df = px.data.gapminder().query("year == 2007")
fig = px.line_geo(df, locations="iso_alpha",
                  color="continent", # "continent" is one of the columns of gapminder
                  projection="orthographic")
fig.show()
```



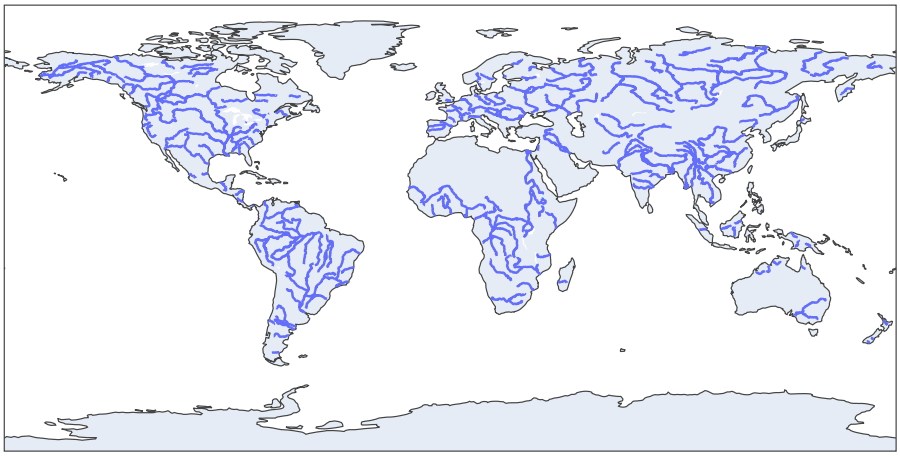
Lines on Maps from GeoPandas

data frame with linestring or multilinestring features, one can extra point data and use px.line_geo().

ects

ny lines

```
import plotly.express as px
import geopandas as gpd
import shapely.geometry
import numpy as np
import wget

# download a zipped shapefile
wget.download("https://plotly.github.io/datasets/ne_50m_rivers_lake_centerlines.zip")

# open a zipped shapefile with the zip:// pseudo-protocol
geo_df = gpd.read_file("zip://ne_50m_rivers_lake_centerlines.zip")

lats = []
lons = []
names = []

for feature, name in zip(geo_df.geometry, geo_df.name):
    if isinstance(feature, shapely.geometry.linestring.LineString):
        linestrings = [feature]
    elif isinstance(feature, shapely.geometry.multilinestring.MultiLineString):
        linestrings = feature.geoms
    else:
        continue
    for linestring in linestrings:
        x, y = linestring.xy
        lats = np.append(lats, y)
        lons = np.append(lons, x)
        names = np.append(names, [name]*len(y))
        lats = np.append(lats, None)
        lons = np.append(lons, None)
        names = np.append(names, None)

fig = px.line_geo(lat=lats, lon=lons, hover_name=names)
fig.show()
```

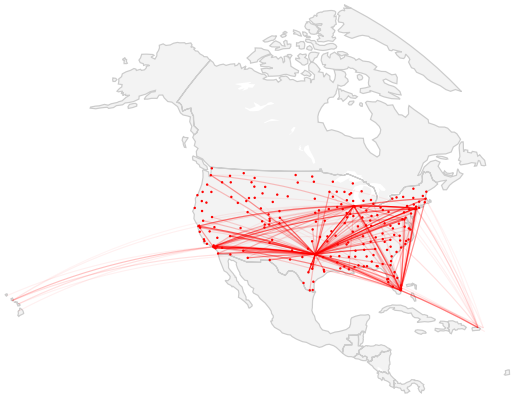


Lines on Maps with plotly.graph_objects

US Flight Paths Map

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

df_airports = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2011_february_us_airport_traffic.csv')
df_airports.head()

df_flight_paths = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2011_february_aa_flight_paths.csv')
df_flight_paths.head()

fig = go.Figure()

fig.add_trace(go.Scattergeo(
    locationmode = 'USA-states',
    lon = df_airports['long'],
    lat = df_airports['lat'],
    hoverinfo = 'text',
    text = df_airports['airport'],
    mode = 'markers',
    marker = dict(
        size = 2,
        color = 'rgb(255, 0, 0)',
        line = dict(
            width = 3,
            color = 'rgba(68, 68, 68, 0)'
        )
    )))

flight_paths = []
for i in range(len(df_flight_paths)):
    fig.add_trace(
        go.Scattergeo(
            locationmode = 'USA-states',
            lon = [df_flight_paths['start_lon'][i], df_flight_paths['end_lon'][i]],
            lat = [df_flight_paths['start_lat'][i], df_flight_paths['end_lat'][i]],
            mode = 'lines',
            line = dict(width = 1,color = 'red'),
            opacity = float(df_flight_paths['cnt'][i]) / float(df_flight_paths['cnt'].max()),
        )
    )

fig.update_layout(
    title_text = 'Feb. 2011 American Airline flight paths<br>(Hover for airport names)',
    showlegend = False,
    geo = dict(
        scope = 'north america',
        projection_type = 'azimuthal equal area',
        showland = True,
        landcolor = 'rgb(243, 243, 243)',
        countrycolor = 'rgb(204, 204, 204)',
    ),
)

fig.show()
```

Feb. 2011 American Airline flight paths
(Hover for airport names)



ects

ny lines

Performance improvement: put many lines in the same trace

For very large amounts (>1000) of lines, performance may become critical. If you can relinquish setting individual line styles (e.g. opacity), you can put multiple paths into one trace. This makes the map render faster and reduces the script execution time and memory consumption.

Use `None` between path coordinates to create a break in the otherwise connected paths.

ects

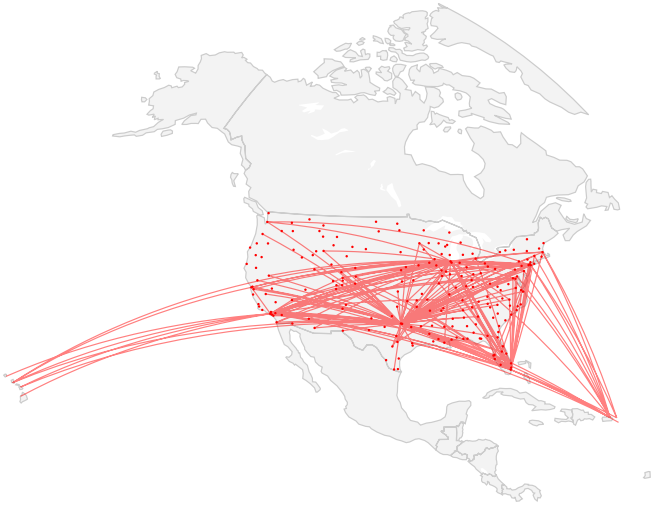
ny lines

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

df_airports = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2011_february_us_airport_traffic.csv')
df_airports.head()

df_flight_paths = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/2011_february_aa_flight_paths.csv')
df_flight_paths.head()

fig = go.Figure()

fig.add_trace(go.Scattergeo(
    locationmode = 'USA-states',
    lon = df_airports['long'],
    lat = df_airports['lat'],
    hoverinfo = 'text',
    text = df_airports['airport'],
    mode = 'markers',
    marker = dict(
        size = 2,
        color = 'rgb(255, 0, 0)',
        line = dict(
            width = 3,
            color = 'rgba(68, 68, 68, 0)'
        )
    )))

lons = []
lats = []
import numpy as np
lons = np.empty(3 * len(df_flight_paths))
lons[::3] = df_flight_paths['start_lon']
lons[1::3] = df_flight_paths['end_lon']
lons[2::3] = None
lats = np.empty(3 * len(df_flight_paths))
lats[::3] = df_flight_paths['start_lat']
lats[1::3] = df_flight_paths['end_lat']
lats[2::3] = None

fig.add_trace(
    go.Scattergeo(
        locationmode = 'USA-states',
        lon = lons,
        lat = lats,
        mode = 'lines',
        line = dict(width = 1,color = 'red'),
        opacity = 0.5
    )
)

fig.update_layout(
    title_text = 'Feb. 2011 American Airline flight paths<br>(Hover for airport names)',
    showlegend = False,
    geo = go.layout.Geo(
        scope = 'north america',
        projection_type = 'azimuthal equal area',
        showland = True,
        landcolor = 'rgb(243, 243, 243)',
        countrycolor = 'rgb(204, 204, 204)',
    ),
    height=700,
)

fig.show()
```

Feb. 2011 American Airline flight paths
(Hover for airport names)

ects

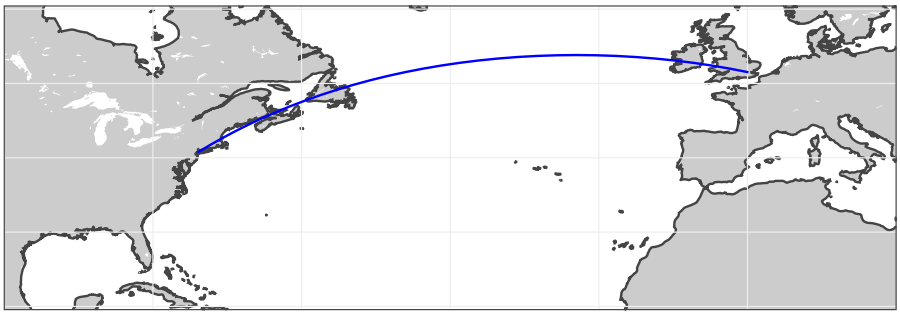
ny lines



London to NYC Great Circle

```
import plotly.graph_objects as go

fig = go.Figure(data=go.Scattergeo(
    lat = [40.7127, 51.5072],
    lon = [-74.0059, 0.1275],
    mode = 'lines',
    line = dict(width = 2, color = 'blue'),
))

fig.update_layout(
    title_text = 'London to NYC Great Circle',
    showlegend = False,
    geo = dict(
        resolution = 50,
        showland = True,
        showlakes = True,
        landcolor = 'rgb(204, 204, 204)',
        countrycolor = 'rgb(204, 204, 204)',
        lakecolor = 'rgb(255, 255, 255)',
        projection_type = "equirectangular",
        coastlinewidth = 2,
        lataxis = dict(
            range = [20, 60],
            showgrid = True,
            dtick = 10
        ),
        lonaxis = dict(
            range = [-100, 20],
            showgrid = True,
            dtick = 20
        ),
    )
)

fig.show()
```

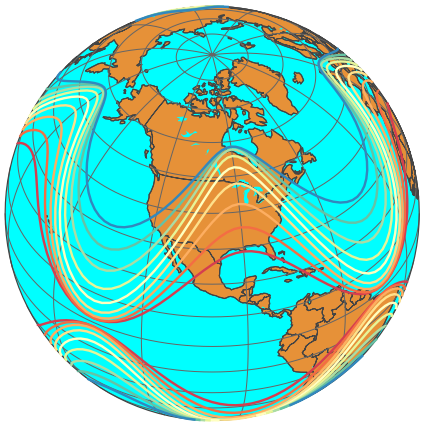
London to NYC Great Circle



Contour lines on globe

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

df = pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/globe_contours.csv')
df.head()


scl = ['rgb(213,62,79)', 'rgb(244,109,67)', 'rgb(253,174,97)', \
    'rgb(254,224,139)', 'rgb(255,255,191)', 'rgb(230,245,152)', \
    'rgb(171,221,164)', 'rgb(102,194,165)', 'rgb(50,136,189)'
]
n_colors = len(scl)

fig = go.Figure()

for i, (lat, lon) in enumerate(zip(df.columns[::2], df.columns[1::2])):
    fig.add_trace(go.Scattergeo(
        lon = df[lon],
        lat = df[lat],
        mode = 'lines',
        line = dict(width = 2, color = scl[i % n_colors]
        )))

fig.update_layout(
    title_text = 'Contour lines over globe<br>(Click and drag to rotate)',
    showlegend = False,
    geo = dict(
        showland = True,
        showcountries = True,
        showocean = True,
        countrywidth = 0.5,
        landcolor = 'rgb(230, 145, 56)',
        lakecolor = 'rgb(0, 255, 255)',
        oceancolor = 'rgb(0, 255, 255)',
        projection = dict(
            type = 'orthographic',
            rotation = dict(
                lon = -100,
                lat = 40,
                roll = 0
            )
        ),
        lonaxis = dict(
            showgrid = True,
            gridcolor = 'rgb(102, 102, 102)',
            gridwidth = 0.5
        ),
        lataxis = dict(
            showgrid = True,
            gridcolor = 'rgb(102, 102, 102)',
            gridwidth = 0.5
        )
    )
)

fig.show()
```

Contour lines over globe
(Click and drag to rotate)



ects

ny lines

Reference

See function reference for px.(line_geo) (https://plotly.com/python-api-reference/generated/plotly.express.line_geo) or https://plotly.com/python/reference/scattergeo/ (https://plotly.com/python/reference/scattergeo/) 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
```