Plotly Tutorial

January 26, 2021

1 Plotly Tutorial

1.1 Prepare

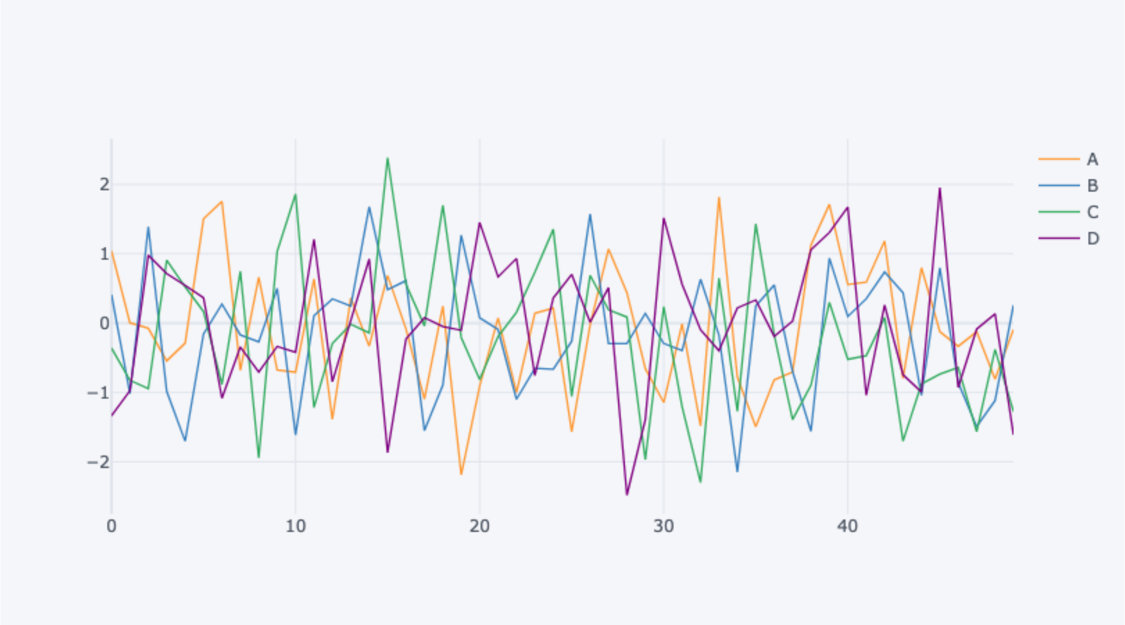
[12]:

```
import matplotlib
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D  # noqa: F401 unused import
import pandas as pd
import plotly.graph_objects as go
import cufflinks as cf
import chart_studio.plotly as py
import seaborn as sns
import plotly.express as px
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
%matplotlib inline
init_notebook_mode(connected=True)
cf.go_offline()
```

1.2 Basics

[19]:

```
arr_1 = np.random.randn(50,4)
df_1 = pd.DataFrame(arr_1, columns=['A', 'B', 'C', 'D'])
df_1.head()
# df_1.plot()
df_1.iplot()
```

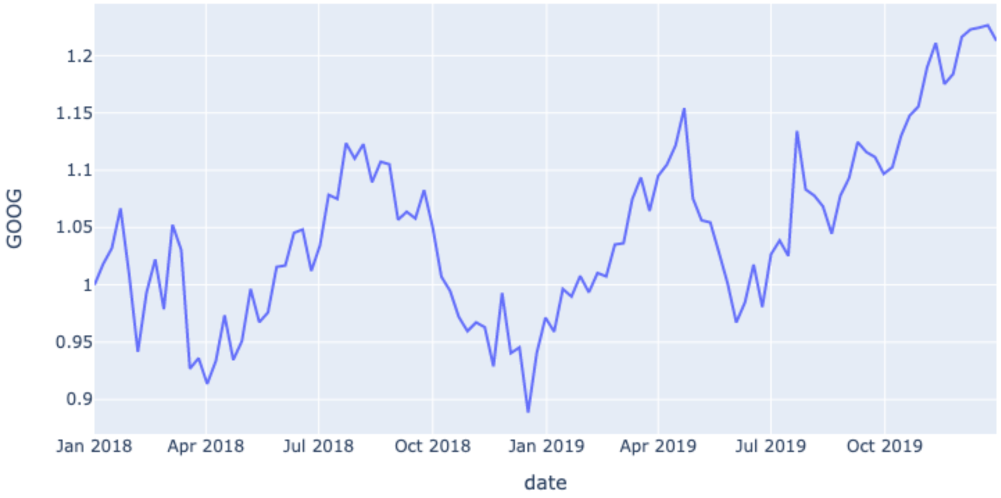


1.3 Line Plots

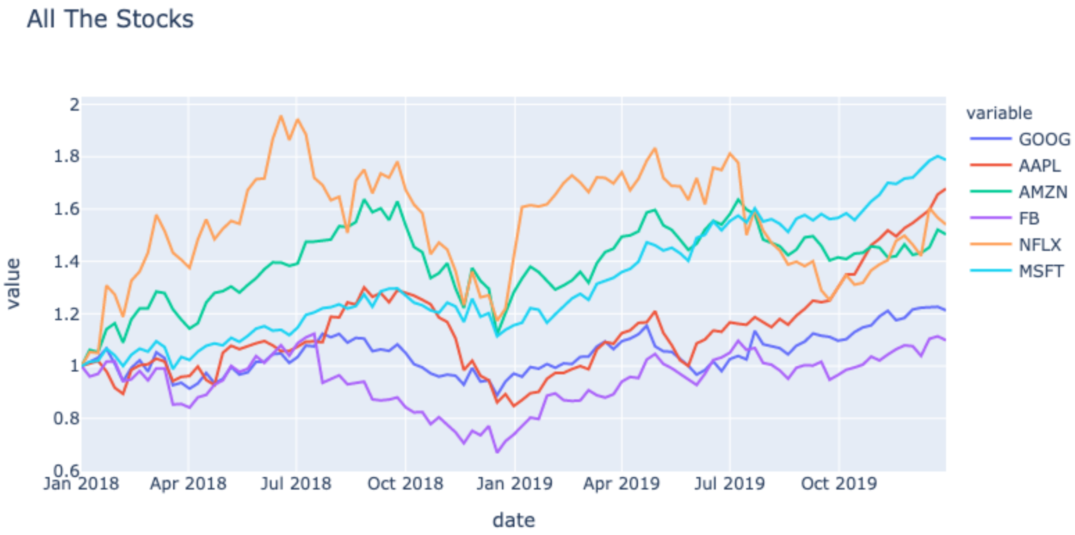
```
[60]: df_stocks = px.data.stocks()
      df_stocks.head()
```

```
[60]:          date      GOOG      AAPL      AMZN        FB      NFLX      MSFT
      0  2018-01-01  1.000000  1.000000  1.000000  1.000000  1.000000  1.000000
      1  2018-01-08  1.018172  1.011943  1.061881  0.959968  1.053526  1.015988
      2  2018-01-15  1.032008  1.019771  1.053240  0.970243  1.049860  1.020524
      3  2018-01-22  1.066783  0.980057  1.140676  1.016858  1.307681  1.066561
      4  2018-01-29  1.008773  0.917143  1.163374  1.018357  1.273537  1.040708
```

```
[56]: px.line(df_stocks, x='date', y='GOOG',
              labels={'x': 'Date', 'y': 'Price' })
```

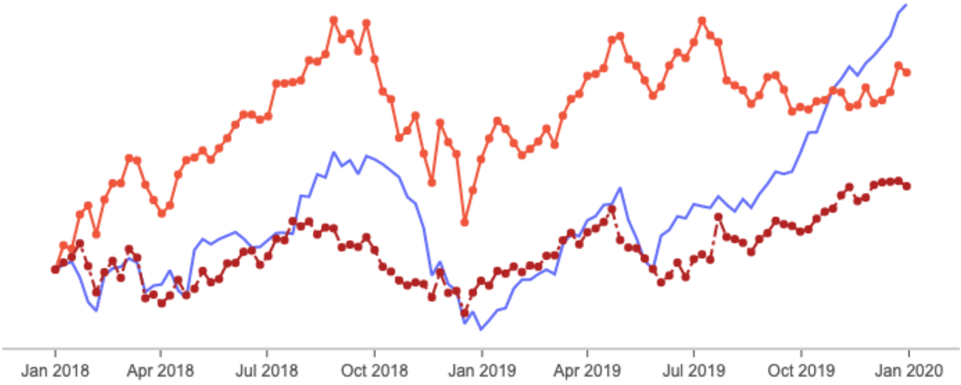


```
[58]: px.line(df_stocks, x='date', y=['GOOG', 'AAPL', 'AMZN', 'FB', 'NFLX', 'MSFT'],
             labels={'x': 'Date', "y": 'Price' }, title='All The Stocks')
```



```
[69]: fig = go.Figure()
      fig.add_trace(go.Scatter(x= df_stocks.date, y = df_stocks.AAPL,
                               mode='lines', name='Apple'))
```

```
fig.add_trace(go.Scatter(x= df_stocks.date, y = df_stocks.AMZN,
                         mode='lines+markers', name='Amazon'))
fig.add_trace(go.Scatter(x= df_stocks.date, y = df_stocks.GOOG,
                         mode='lines+markers', name='Google',
                         line=dict(color='firebrick', width=2,
                                  dash='dashdot')))
# -------------------------------------------------------------------
# fig.update_layout(title='Stock Price Data 2018-2020',
#                   xaxis_title='Date',
#                   yaxis_title='Price',)
# -------------------------------------------------------------------
fig.update_layout(
        xaxis=dict(
            showline=True,
            showgrid=False,
            showticklabels=True,
            linecolor='rgb(204,204,204)',
            linewidth=2,
            ticks='outside',
            tickfont=dict(
                family='Arial',
                size=12,
                color='rgb(82,82,82)'
                ),
        ),
        yaxis=dict(
            showgrid=False,
            zeroline=False,
            showline=False,
            showticklabels=False
            ),
        autosize=False,
        margin=dict(
            autoexpand=False,
            l=100,
            r=20,
            t=110,
            ),
        showlegend=False,
        plot_bgcolor='white',
    )
```

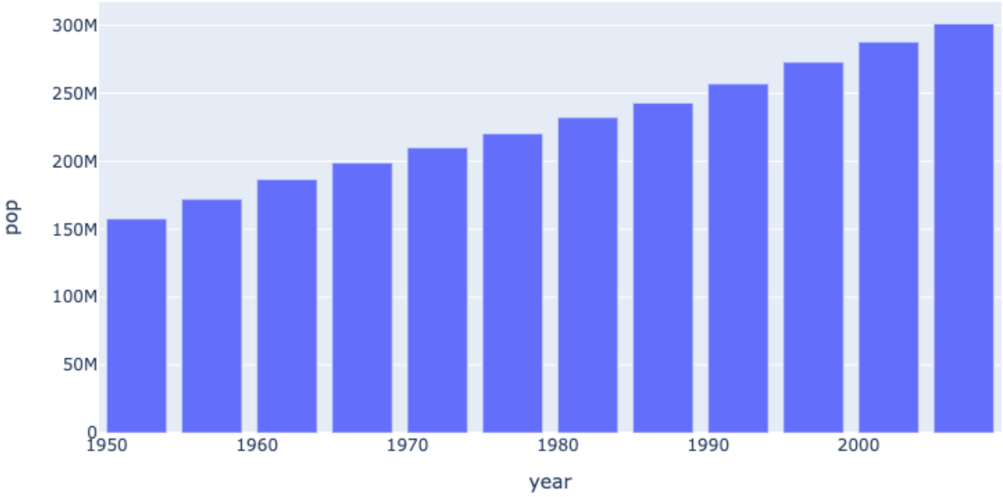


1.4 Bar Charts

```
[74]: df_us = px.data.gapminder().query("country == 'United States'")
      df_us.head()
```

```
[74]:             country continent  year  lifeExp        pop    gdpPercap  \
      1608  United States  Americas  1952    68.44  157553000  13990.48208
      1609  United States  Americas  1957    69.49  171984000  14847.12712
      1610  United States  Americas  1962    70.21  186538000  16173.14586
      1611  United States  Americas  1967    70.76  198712000  19530.36557
      1612  United States  Americas  1972    71.34  209896000  21806.03594

           iso_alpha  iso_num
      1608       USA      840
      1609       USA      840
      1610       USA      840
      1611       USA      840
      1612       USA      840
```

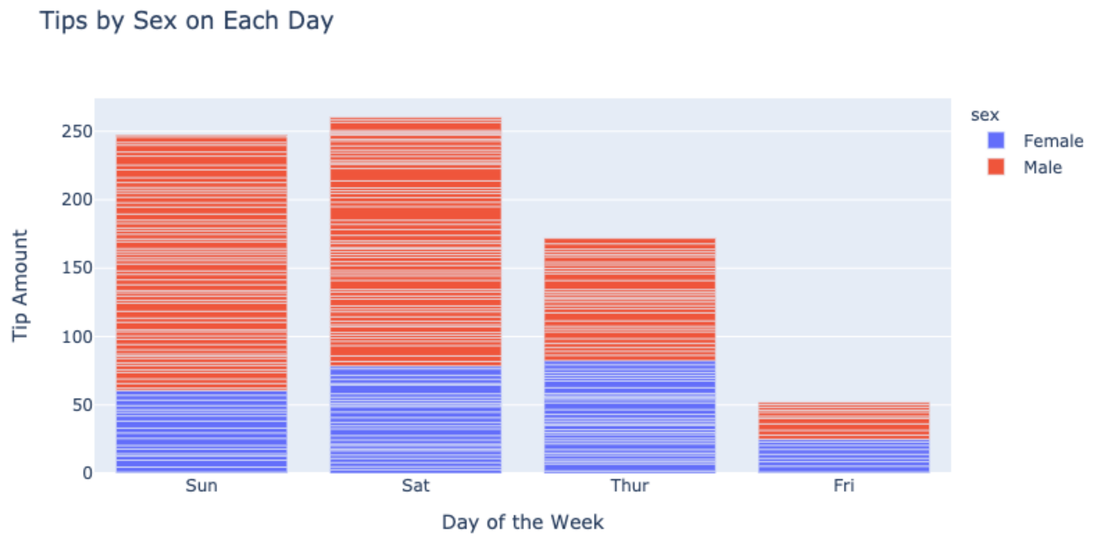
```
[77]: px.bar(df_us, x='year', y='pop')
```



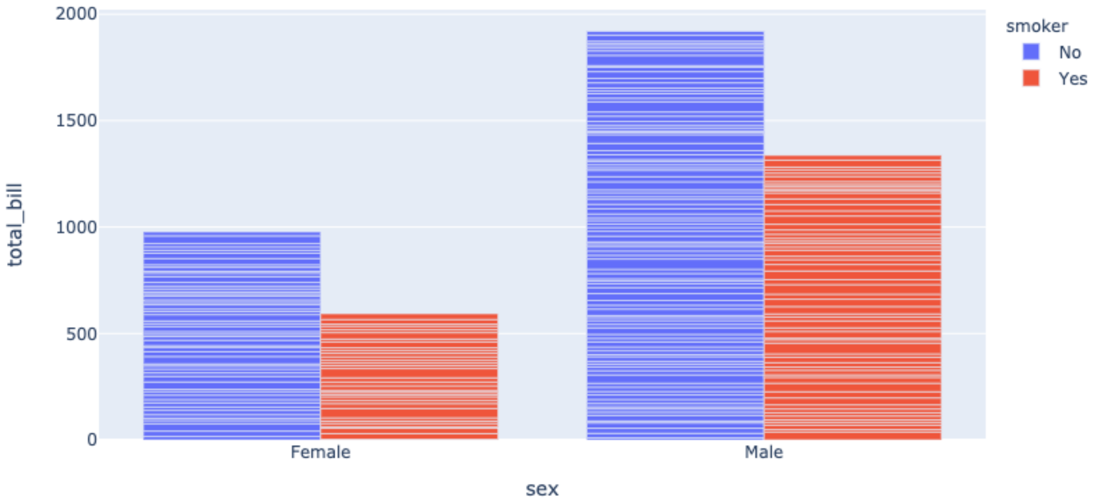
[81]:
```
df_tips = px.data.tips()
df_tips.head()
```

[81]:
```
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
2       21.01  3.50    Male     No  Sun  Dinner     3
3       23.68  3.31    Male     No  Sun  Dinner     2
4       24.59  3.61  Female     No  Sun  Dinner     4
```

[82]:
```
px.bar(df_tips, x='day', y='tip', color='sex',
       title='Tips by Sex on Each Day',
      labels={'tip': 'Tip Amount', 'day': 'Day of the Week'})
```



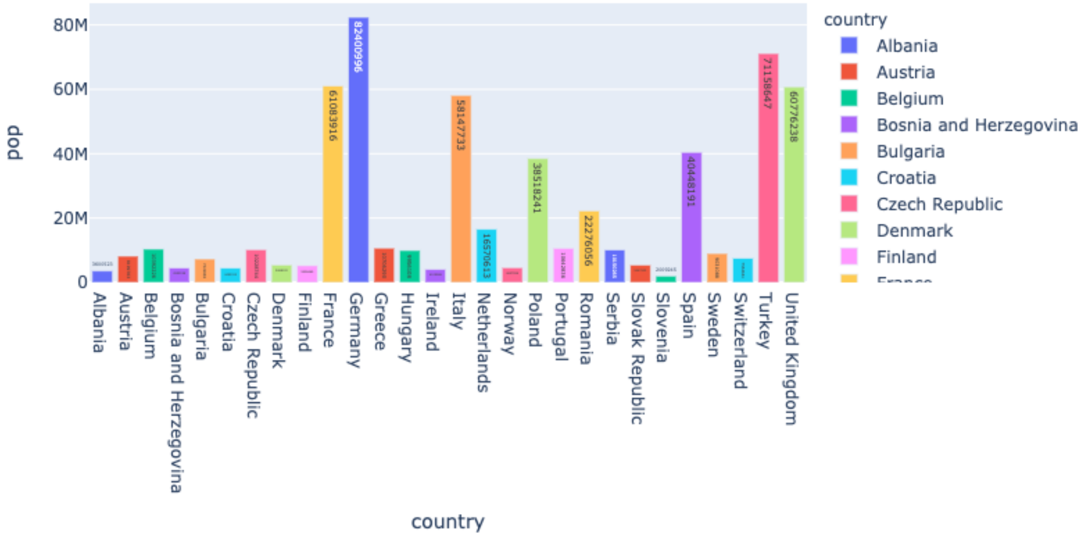
[83]:
```
px.bar(df_tips, x='sex', y='total_bill', color='smoker', barmode='group')
```



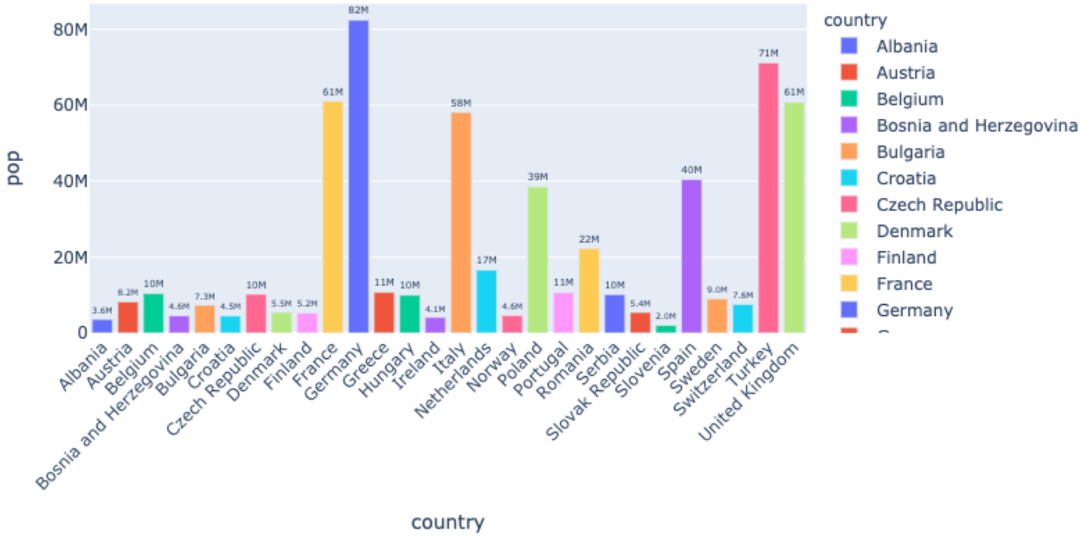
[84]:
```
df_europe = px.data.gapminder().query("continent == 'Europe' and year == 2007␣
↪and pop > 2.e6")
df_europe.head()
```

```
[84]:                    country continent  year  lifeExp       pop     gdpPercap  \
      23                 Albania    Europe  2007   76.423   3600523   5937.029526
      83                 Austria    Europe  2007   79.829   8199783  36126.492700
      119                Belgium    Europe  2007   79.441  10392226  33692.605080
      155 Bosnia and Herzegovina    Europe  2007   74.852   4552198   7446.298803
      191               Bulgaria    Europe  2007   73.005   7322858  10680.792820

          iso_alpha  iso_num
      23        ALB        8
      83        AUT       40
      119       BEL       56
      155       BIH       70
      191       BGR      100
```

```
[88]: fig = px.bar(df_europe, y='pop', x='country', text='pop', color='country')
      fig
```



```
[91]: fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
      fig.update_layout(uniformtext_minsize=8)
      fig.update_layout(xaxis_tickangle=-45)
      fig
```

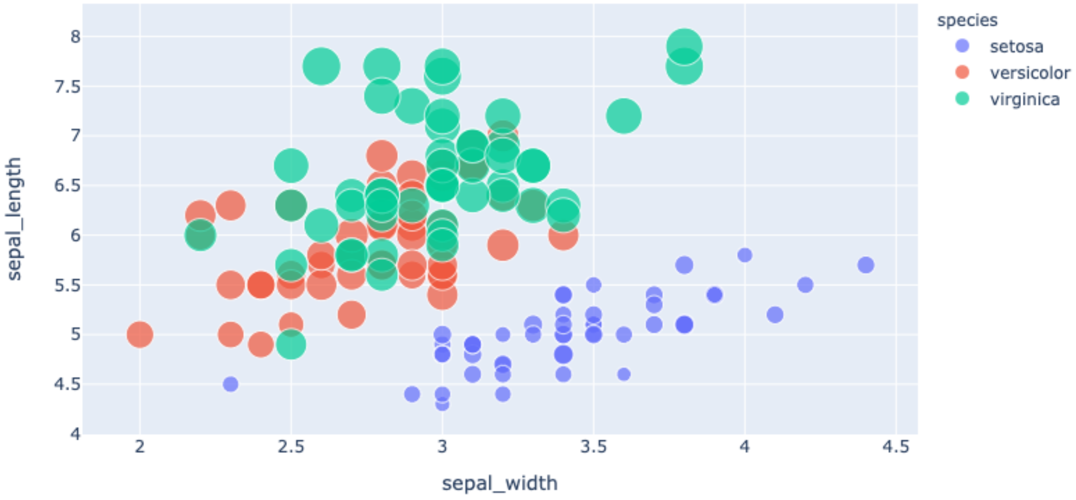


1.5 Scatter Plots

```
[93]: df_iris = px.data.iris()
      df_iris.head()
```

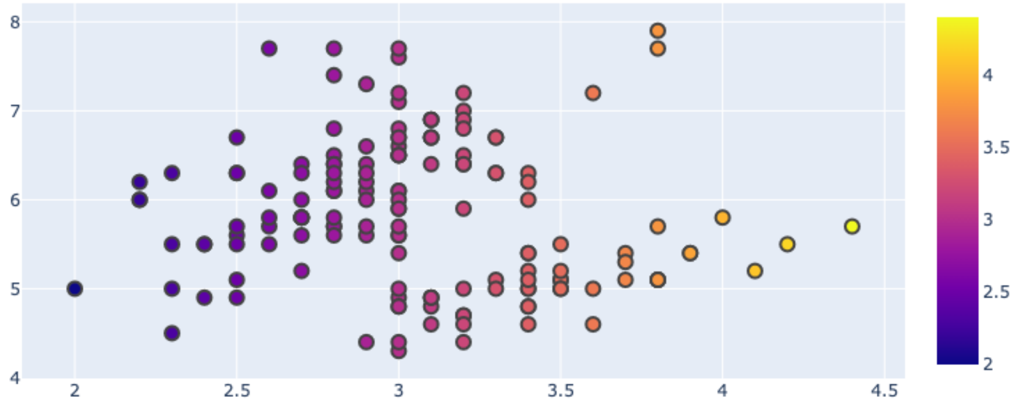
```
[93]:    sepal_length  sepal_width  petal_length  petal_width species  species_id
      0           5.1          3.5           1.4          0.2  setosa           1
      1           4.9          3.0           1.4          0.2  setosa           1
      2           4.7          3.2           1.3          0.2  setosa           1
      3           4.6          3.1           1.5          0.2  setosa           1
      4           5.0          3.6           1.4          0.2  setosa           1
```

```
[94]: px.scatter(df_iris, x='sepal_width', y='sepal_length',
                 color = 'species', size = 'petal_length',
                 hover_data=['petal_width'])
```

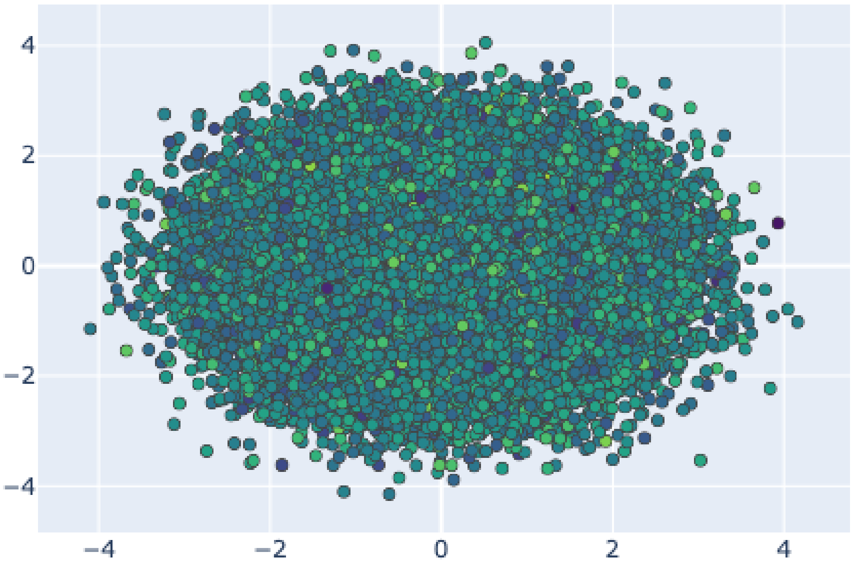


[96]:

```
fig = go.Figure()
fig.add_trace(go.Scatter(
    x=df_iris.sepal_width,
    y=df_iris.sepal_length,
    mode='markers',
    marker_color=df_iris.sepal_width,
    text=df_iris.species,
    marker=dict(
        showscale=True
    )
))

fig.update_traces(marker_line_width=2, marker_size=10)
fig
```



[100]:
```
fig = go.Figure(data=go.Scattergl(
    x=np.random.randn(100000),
    y=np.random.randn(100000),
    mode='markers',
    marker=dict(
        color=np.random.randn(100000),
        colorscale='Viridis',
        line_width=1
    )
))
fig
```



1.6 Pie Charts

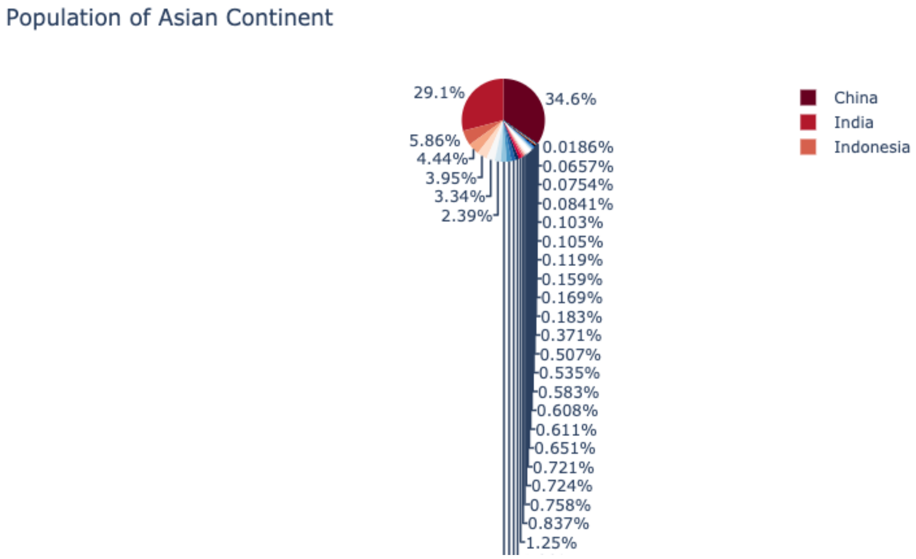
[115]:
```
df_asia = px.data.gapminder().query("year == 2007").query("continent ==␣
↪'Asia'") #.query("pop > 100e6")
df_asia.head()
```

[115]:
```
         country continent  year  lifeExp         pop     gdpPercap iso_alpha  \
11   Afghanistan      Asia  2007   43.828    31889923    974.580338       AFG
95       Bahrain      Asia  2007   75.635      708573  29796.048340       BHR
107   Bangladesh      Asia  2007   64.062   150448339   1391.253792       BGD
227     Cambodia      Asia  2007   59.723    14131858   1713.778686       KHM
299        China      Asia  2007   72.961  1318683096   4959.114854       CHN

     iso_num
11         4
95        48
107       50
227      116
```

```
299       156
```

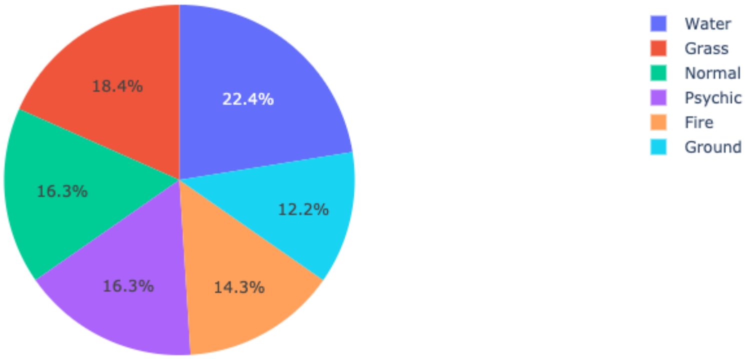
[116]:

```
px.pie(df_asia, values='pop', names='country',
       title='Population of Asian Continent',
       color_discrete_sequence=px.colors.sequential.RdBu
      )
```



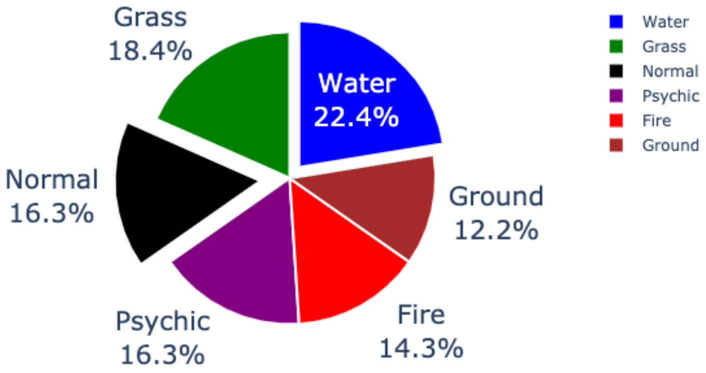
[121]:

```
fig = go.Figure(
    data=go.Pie(
        labels=['Water', 'Grass', 'Normal', 'Psychic', 'Fire', 'Ground'],
        values=[110,90,80,80,70,60]
    )
)
fig
```



[124]:

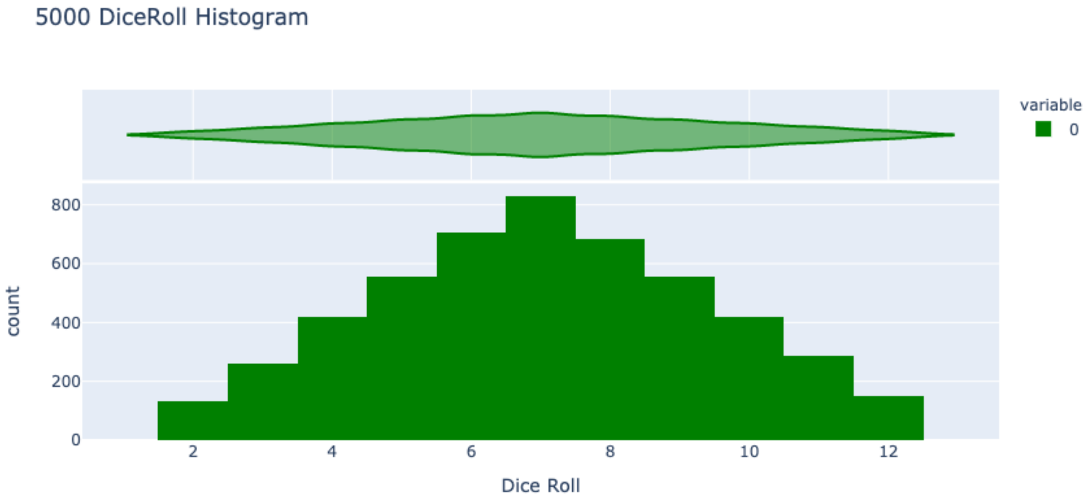
```
colors = ['blue', 'green', 'black', 'purple', 'red', 'brown']
fig = go.Figure(
    data=go.Pie(
        labels=['Water', 'Grass', 'Normal', 'Psychic', 'Fire', 'Ground'],
        values=[110,90,80,80,70,60]
    )
)
fig.update_traces(
    hoverinfo='label+percent',
    textfont_size=20,
    textinfo='label+percent',
    pull=[0.1, 0, 0.2, 0, 0, 0],
    marker=dict(
        colors=colors,
        line=dict(
            color='#FFFFFF',
            width=2
        )
    )
)
fig
```



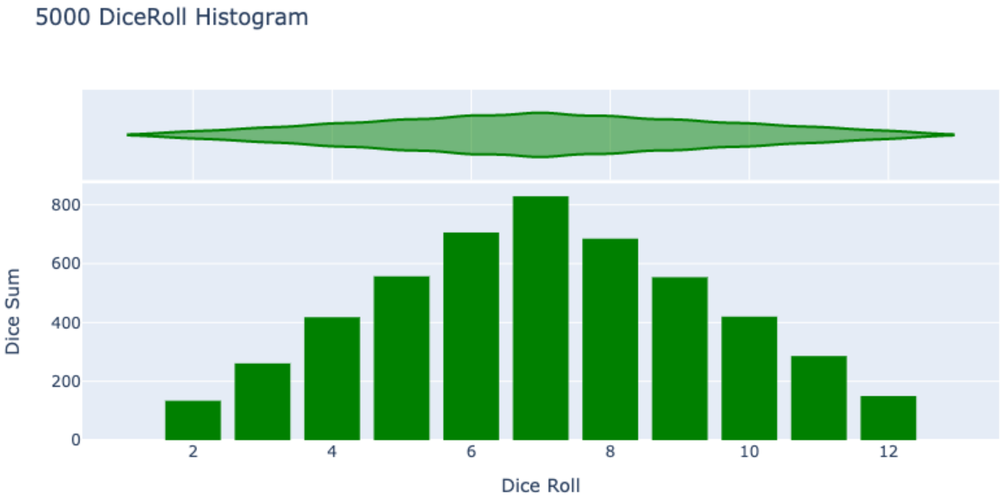
1.7 Histograms

[125]:

```
dice_1 = np.random.randint(1,7,5000)
dice_2 = np.random.randint(1,7,5000)
dice_sum = dice_1 + dice_2
fig = px.histogram(dice_sum,
                   nbins=11,
                   labels = {'value': 'Dice Roll'},
                   title = '5000 DiceRoll Histogram',
                   marginal = 'violin',
                   color_discrete_sequence=['green']
                  )
fig
```

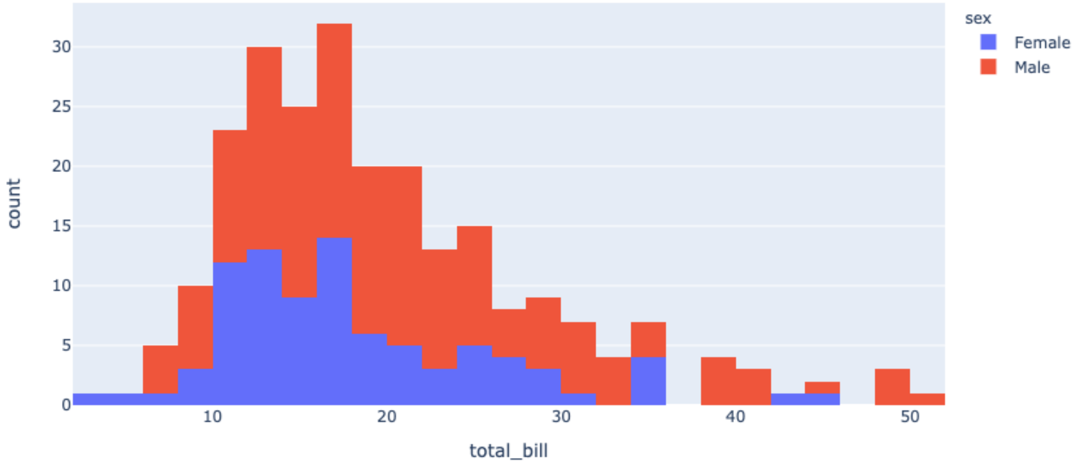


```
[126]: fig.update_layout(
           xaxis_title_text='Dice Roll',
           yaxis_title_text='Dice Sum',
           bargap=0.2,
           showlegend=False
       )
       fig
```



[127]:

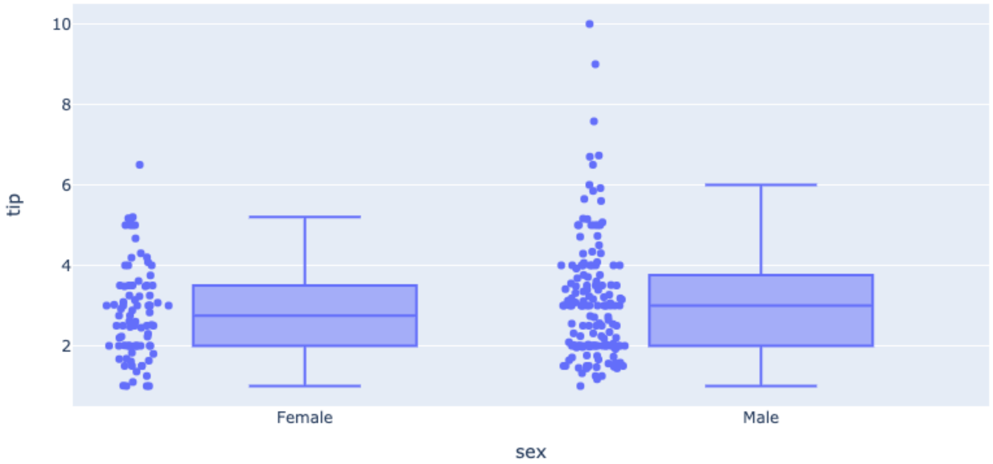
```
df_tips = px.data.tips()
px.histogram(
    df_tips,
    x='total_bill',
    color='sex'
)
```



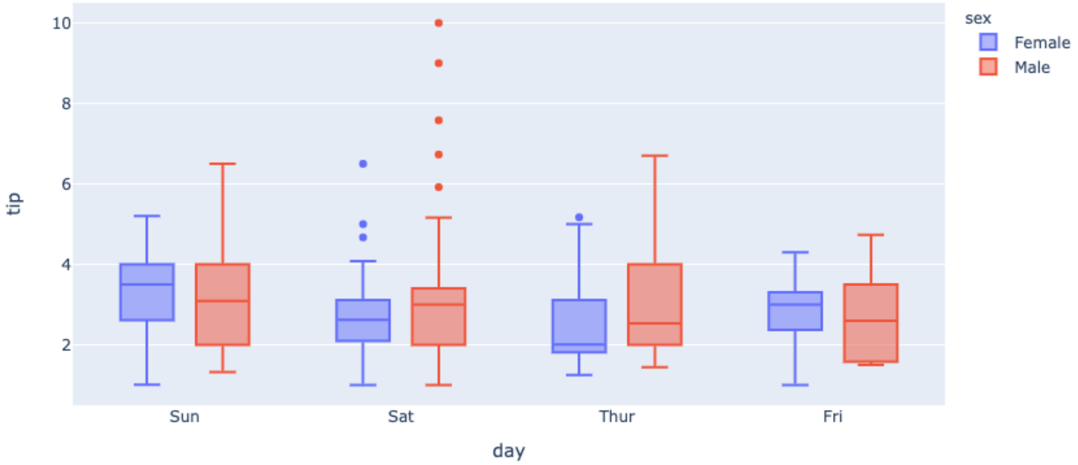
1.8 Box Plots

[128]:

```
df_tips = px.data.tips()
px.box(df_tips, x='sex', y='tip', points='all')
```

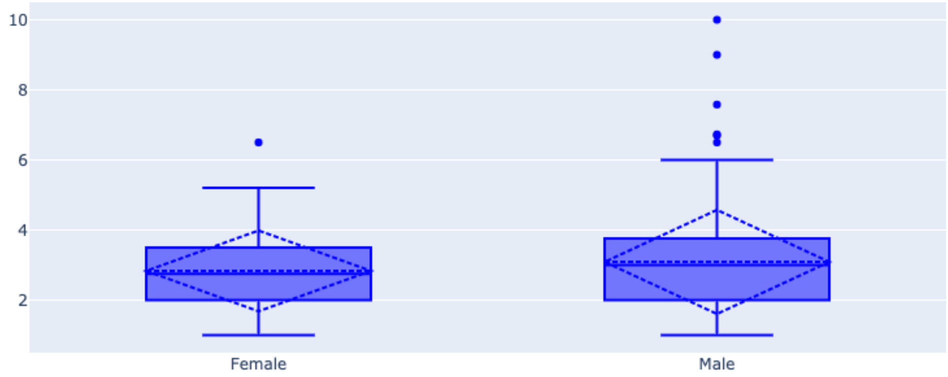


[129]:
```
px.box(df_tips, x='day', y='tip', color='sex')
```



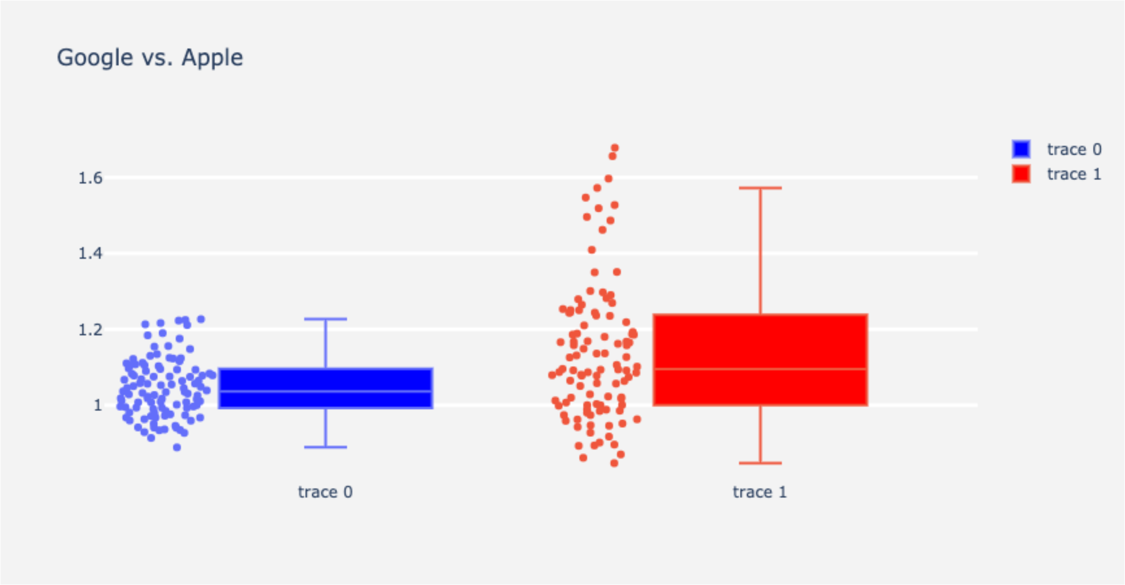
[133]:
```
fig = go.Figure()
fig.add_trace(go.Box(x=df_tips.sex,
                     y=df_tips.tip,
                     marker_color='blue',
                     boxmean='sd'
                    )
```

```
             )
fig
```



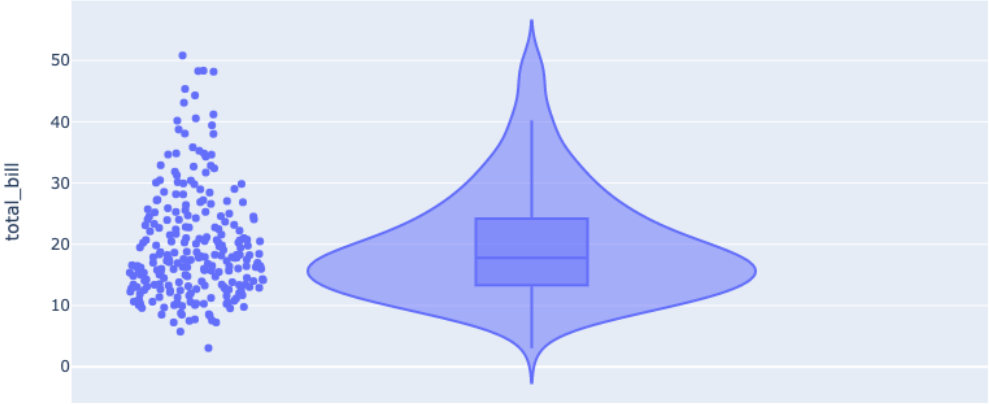
[139]:

```
df_stocks = px.data.stocks()
fig = go.Figure()
fig.add_trace(
    go.Box(
        y=df_stocks.GOOG,
        boxpoints='all',
        fillcolor = 'blue',
        jitter=0.5,
        whiskerwidth=0.2
    )
)
fig.add_trace(
    go.Box(
        y=df_stocks.AAPL,
        boxpoints='all',
        fillcolor = 'red',
        jitter=0.5,
        whiskerwidth=0.2
    )
)
fig.update_layout(
    title='Google vs. Apple',
    yaxis=dict(
        gridcolor='rgb(255,255,255)',
        gridwidth=3
    ),
    paper_bgcolor='rgb(243,243,243)',
    plot_bgcolor='rgb(243,243,243)'
)
fig
```



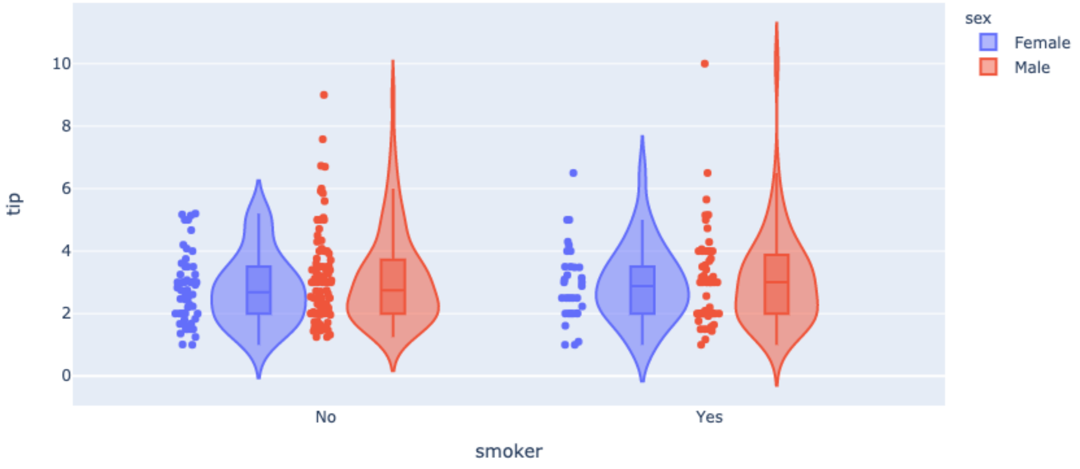
1.9 Violin Plots

[145]:
```
df_tips = px.data.tips()
px.violin(df_tips, y ="total_bill", box=True, points='all')
```



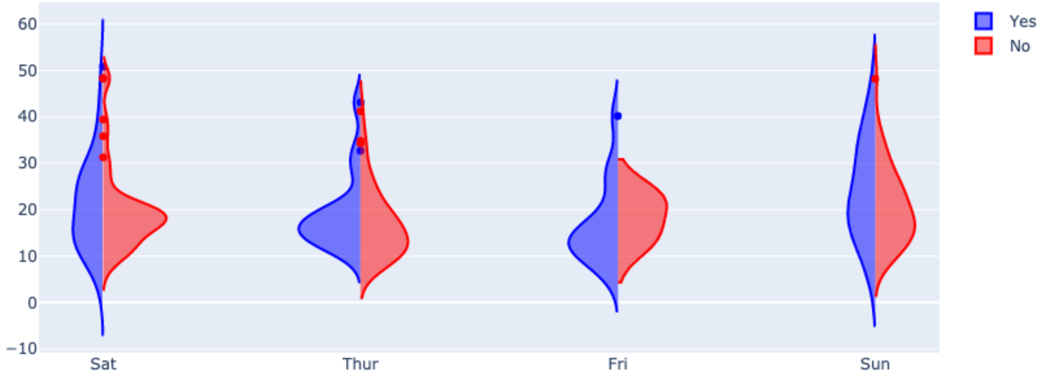
[146]:

```
px.violin(df_tips, y ="tip", x='smoker', color='sex', box=True, points='all',
          hover_data=df_tips.columns)
```



[153]:

```
fig = go.Figure()
fig.add_trace(
    go.Violin(
        x=df_tips['day'][df_tips['smoker'] == 'Yes'],
        y=df_tips['total_bill'][df_tips['smoker'] == 'Yes'],
        legendgroup='Yes',
        scalegroup='Yes',
        name='Yes',
        side='negative',
        line_color='blue'
    )
)
fig.add_trace(
    go.Violin(
        x=df_tips['day'][df_tips['smoker'] == 'No'],
        y=df_tips['total_bill'][df_tips['smoker'] == 'No'],
        legendgroup='Yes',
        scalegroup='Yes',
        name='No',
        side='positive',
        line_color='red'
    )
)
fig
```

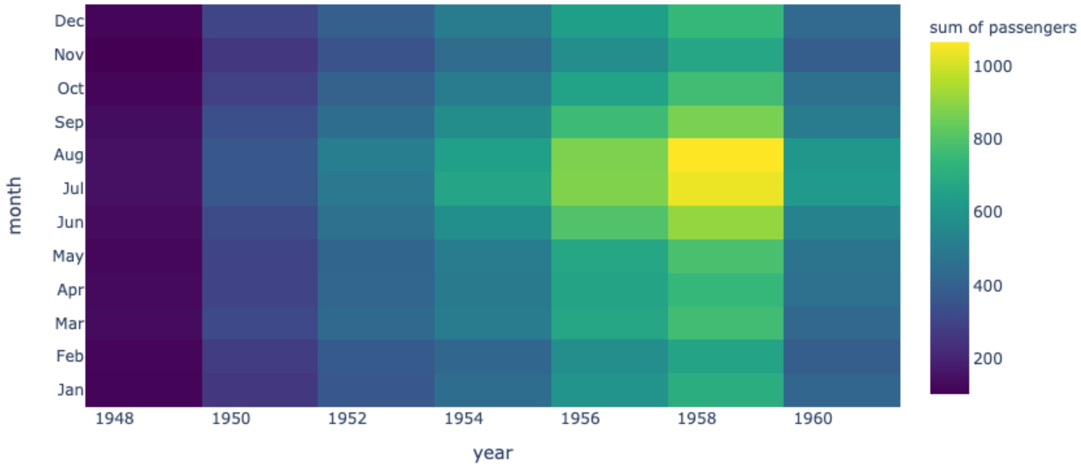


1.10 Density Heatmap

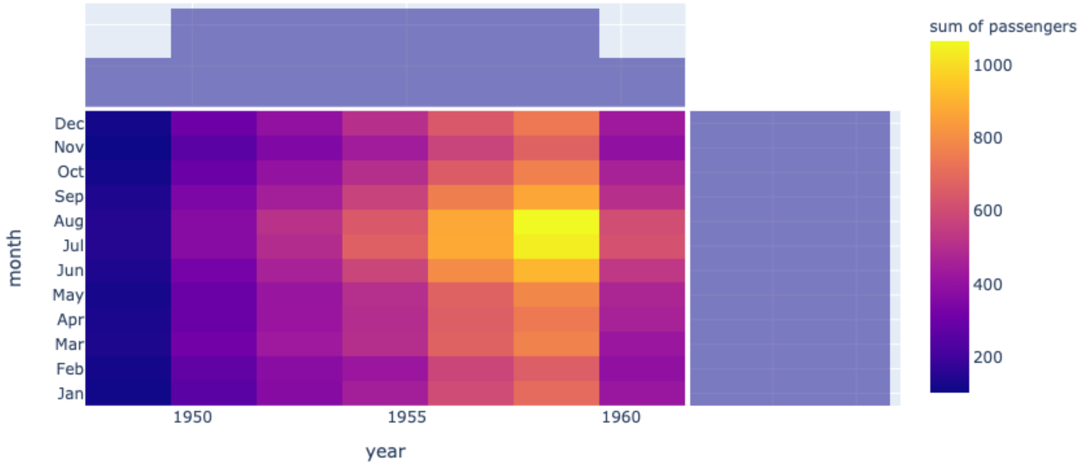
[165]:
```
import ssl
ssl._create_default_https_context = ssl._create_unverified_context
flights = sns.load_dataset("flights")
flights.head()
```

[165]:
```
   year month  passengers
0  1949   Jan         112
1  1949   Feb         118
2  1949   Mar         132
3  1949   Apr         129
4  1949   May         121
```

[166]:
```
fig = px.density_heatmap(
    flights,
    x='year',
    y='month',
    z='passengers',
    color_continuous_scale='Viridis'
)
fig
```

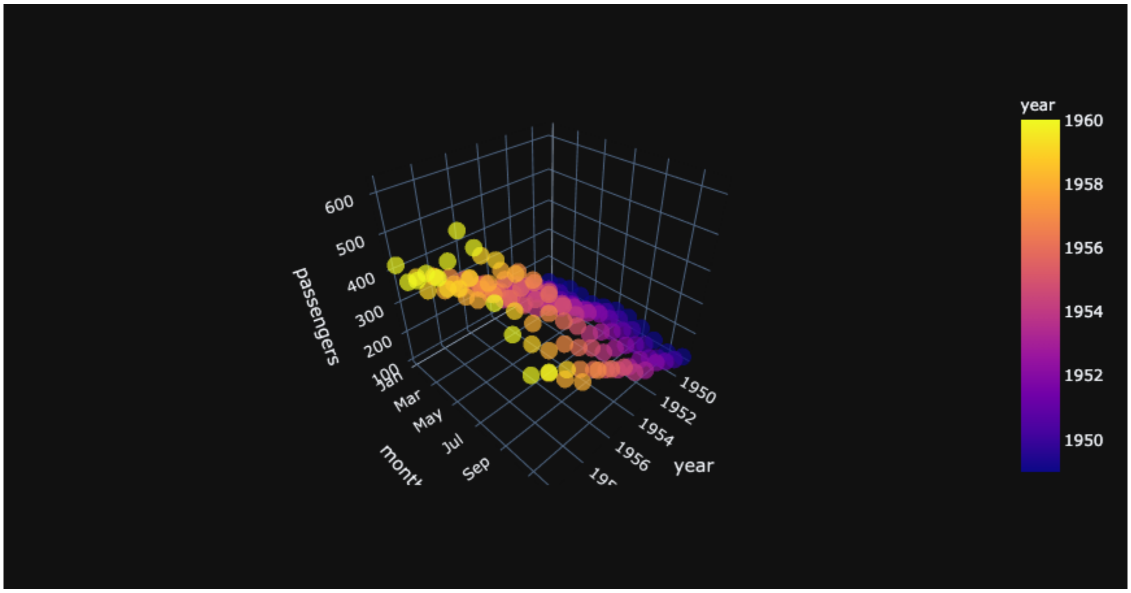


```
[167]: fig = px.density_heatmap(
           fligths,
           x='year',
           y='month',
           z='passengers',
           marginal_x='histogram',
           marginal_y='histogram'
       )
       fig
```



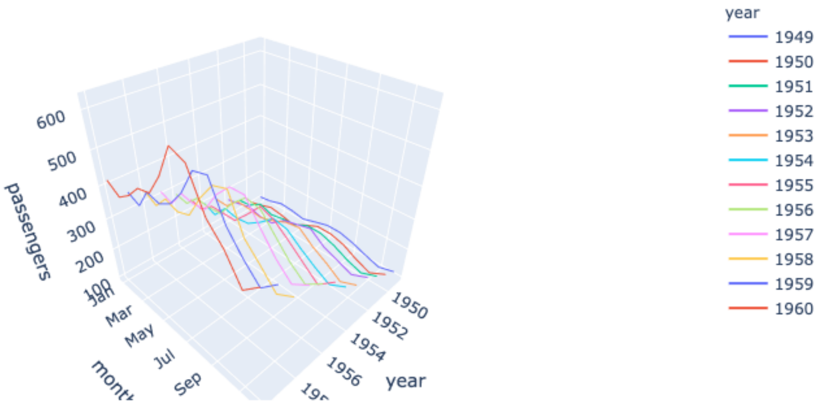
1.11 3D Scatter Plots

[212]:
```
fig = px.scatter_3d(
    flights,
    x='year',
    y='month',
    z='passengers',
    color='year',
    opacity=0.7
)
fig.update_layout(template="plotly_dark")
fig
```



1.12 3D Line Plots

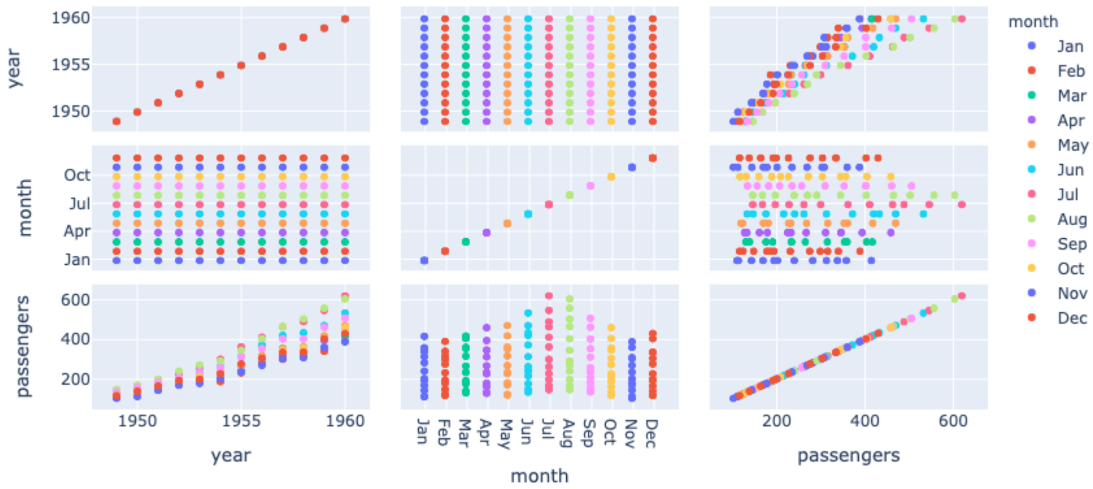
[171]:
```
fig = px.line_3d(
    flights,
    x='year',
    y='month',
    z='passengers',
    color='year',
)
fig
```



1.13 Scater Matrix

[174]:

```python
px.scatter_matrix(
    flights,
    color='month'
)
```



1.14 Map Scatter Plots

[179]:

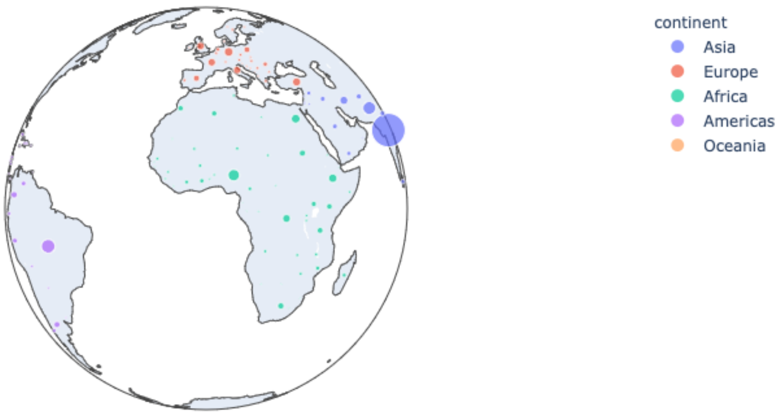
```
df = px.data.gapminder().query("year == 2007")
df.head()
```

[179]:

```
        country continent  year  lifeExp       pop     gdpPercap iso_alpha  \
11  Afghanistan      Asia  2007   43.828  31889923    974.580338       AFG
23      Albania    Europe  2007   76.423   3600523   5937.029526       ALB
35      Algeria    Africa  2007   72.301  33333216   6223.367465       DZA
47       Angola    Africa  2007   42.731  12420476   4797.231267       AGO
59    Argentina  Americas  2007   75.320  40301927  12779.379640       ARG

    iso_num
11        4
23        8
35       12
47       24
59       32
```

[186]:

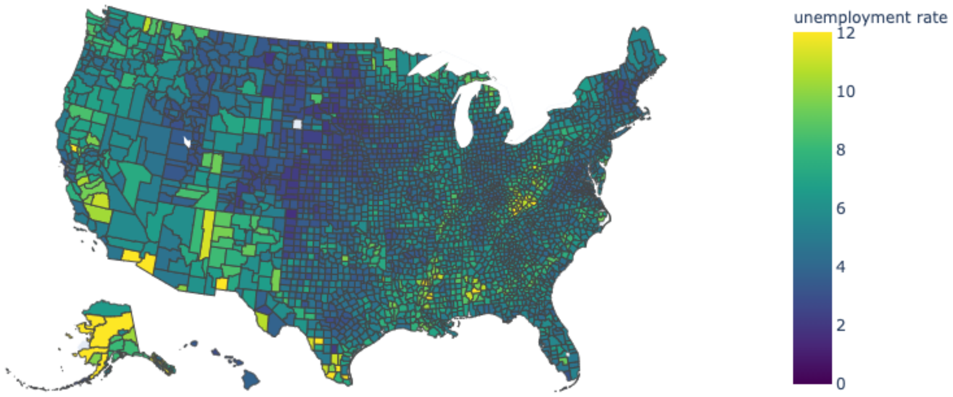
```
fig = px.scatter_geo(
    df,
    locations = 'iso_alpha',
    color="continent",
    hover_name="country",
    size="pop",
    projection="orthographic"
)
fig
```



1.15 Choropleth Maps

[202]:

```
# COPIED:

# You can color complex maps like we do here representing unemployment data

# Allows us to grab data from a supplied URL
from urllib.request import urlopen
# Used to decode JSON data
import json
# Grab US county geometry data
with urlopen('https://raw.githubusercontent.com/plotly/datasets/master/
 ↪geojson-counties-fips.json') as response:
    counties = json.load(response)

# Grab unemployment data based on each counties Federal Information Processing␣
 ↪number
df = pd.read_csv("https://raw.githubusercontent.com/plotly/datasets/master/
 ↪fips-unemp-16.csv",
                   dtype={"fips": str})


# Draw map using the county JSON data, color using unemployment values on a␣
 ↪range of 12
fig = px.choropleth(df, geojson=counties, locations='fips', color='unemp',
                           color_continuous_scale="Viridis",
                           range_color=(0, 12),
                           scope="usa",
                           labels={'unemp':'unemployment rate'}
                          )
fig
```

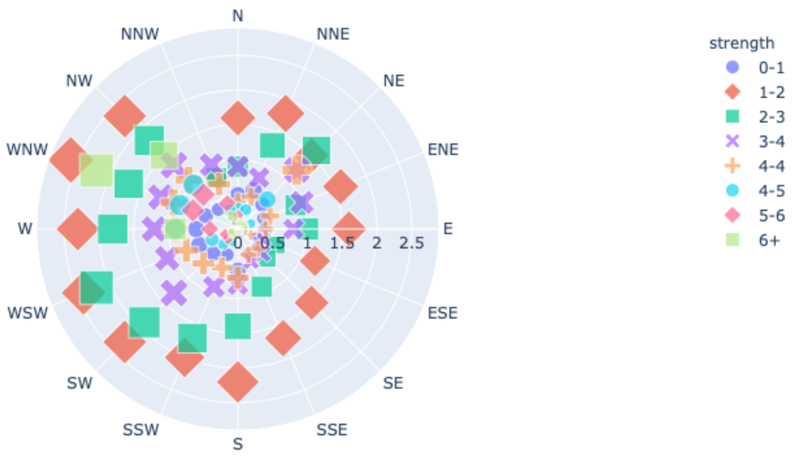


1.16 Polar Charts

[203]:
```
df_wind = px.data.wind()
df_wind.head()
```

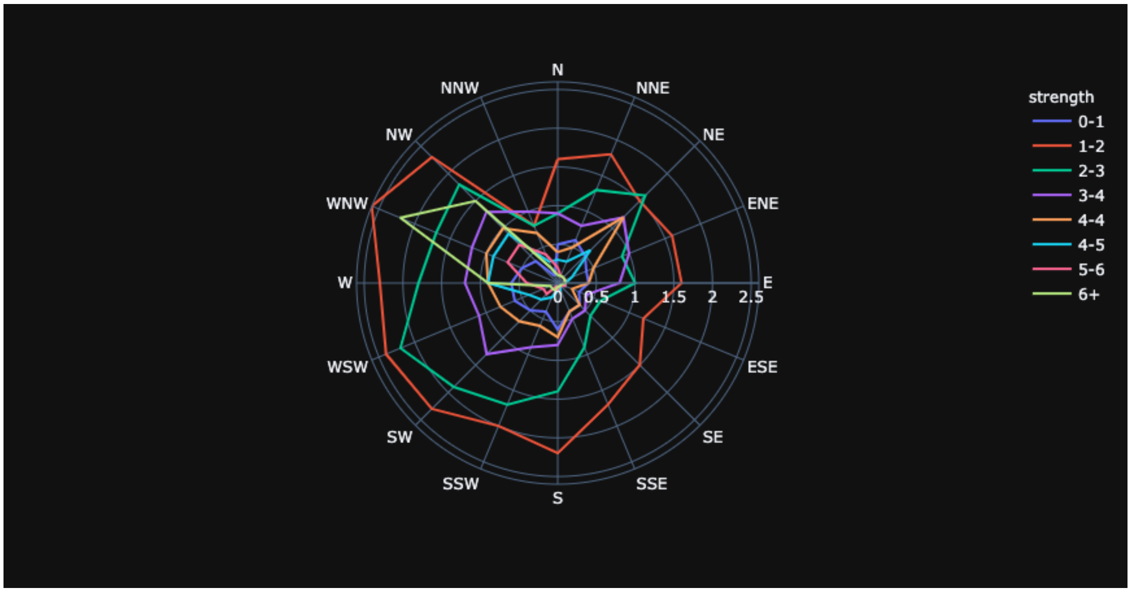
[203]:
```
  direction strength  frequency
0         N      0-1        0.5
1       NNE      0-1        0.6
2        NE      0-1        0.5
3       ENE      0-1        0.4
4         E      0-1        0.4
```

[208]:
```
px.scatter_polar(
    df_wind,
    r='frequency',
    theta="direction",
    color="strength",
    size='frequency',
    symbol="strength"
)
```



[211]:

```
px.line_polar(
    df_wind,
    r='frequency',
    theta="direction",
    color="strength",
    line_close=True,
    template="plotly_dark"
)
# fig.update_layout(template="plotly_dark")
```

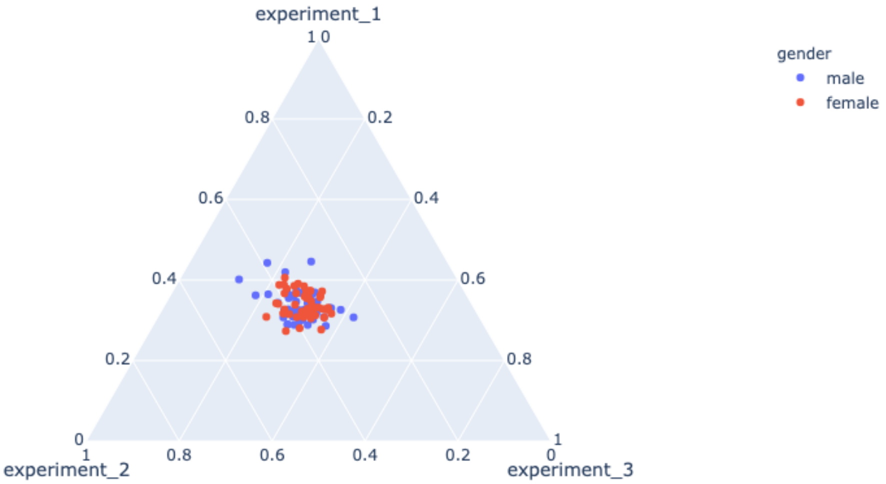


1.17 Ternary Plots

[213]:
```
df_exp = px.data.experiment()
df_exp.head()
```

[213]:
```
   experiment_1  experiment_2  experiment_3  gender      group
0     96.876065     93.417942     73.033193    male    control
1     87.301336    129.603395     66.056554  female    control
2     97.691312    106.187916    103.422709    male  treatment
3    102.978152     93.814682     56.995870  female  treatment
4     87.106993    107.019985     72.140292    male    control
```

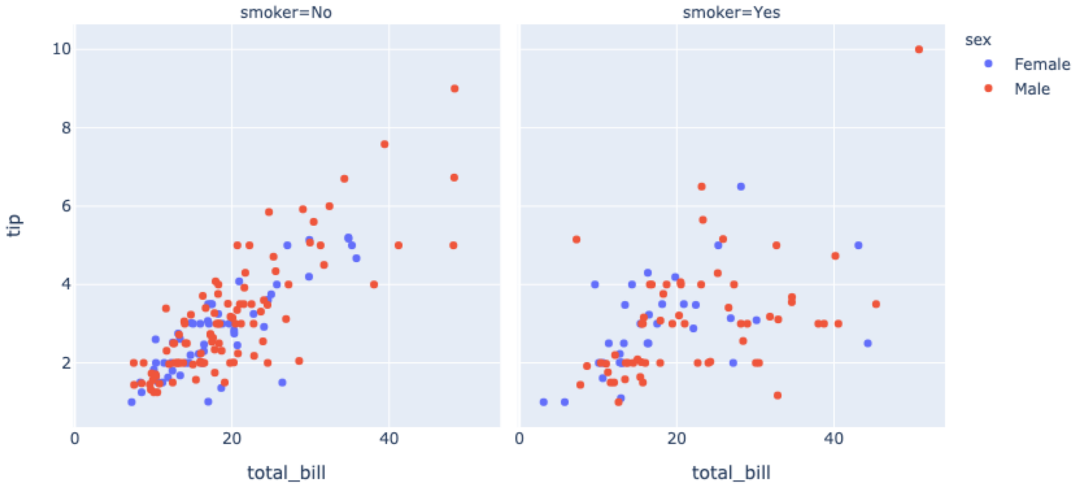
[214]:
```
px.scatter_ternary(
    df_exp,
    a="experiment_1",
    b="experiment_2",
    c="experiment_3",
    hover_name="group",
    color="gender"
)
```



1.18 Facets

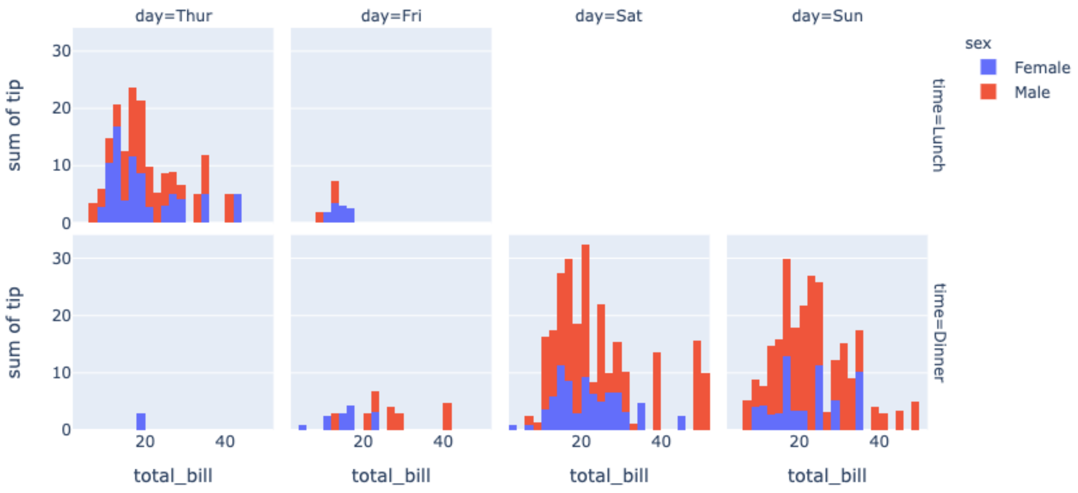
[218]:

```
df_tips = px.data.tips()
px.scatter(
    df_tips,
    x="total_bill",
    y="tip",
    color="sex",
    facet_col="smoker"
)
```



[222]:

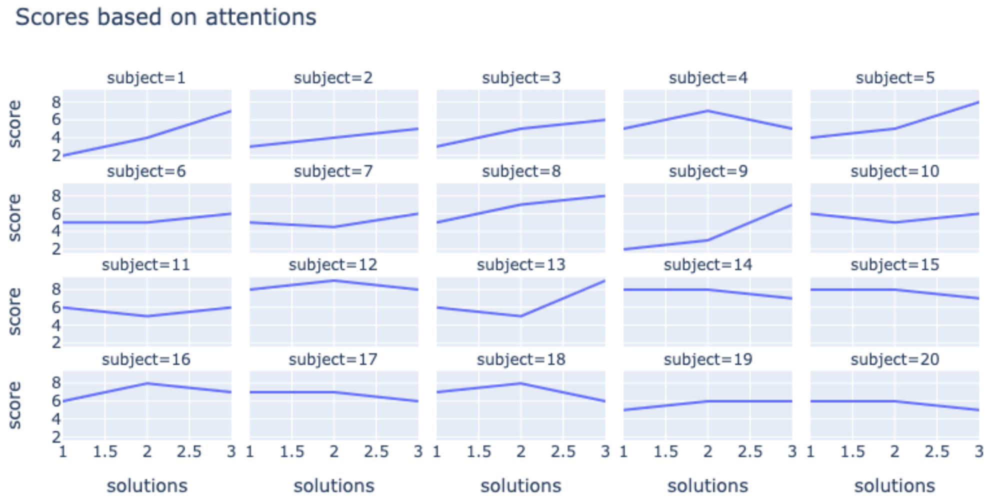
```
px.histogram(
    df_tips,
    x="total_bill",
    y="tip",
    color="sex",
    facet_row="time",
    facet_col="day",
    category_orders={
        "day":['Thur', 'Fri', 'Sat', 'Sun'],
        "time": ["Lunch", "Dinner"],
    }
)
```



```
[223]: att_df = sns.load_dataset("attention")
       att_df.head()
```

```
[223]:    Unnamed: 0  subject attention  solutions  score
       0           0        1   divided          1    2.0
       1           1        2   divided          1    3.0
       2           2        3   divided          1    3.0
       3           3        4   divided          1    5.0
       4           4        5   divided          1    4.0
```

```
[229]: fig = px.line(
           att_df,
           x="solutions",
           y='score',
           facet_col='subject',
           facet_col_wrap=5,
           title='Scores based on attentions'
       )
       fig
```



1.19 Animated Plots

[230]:

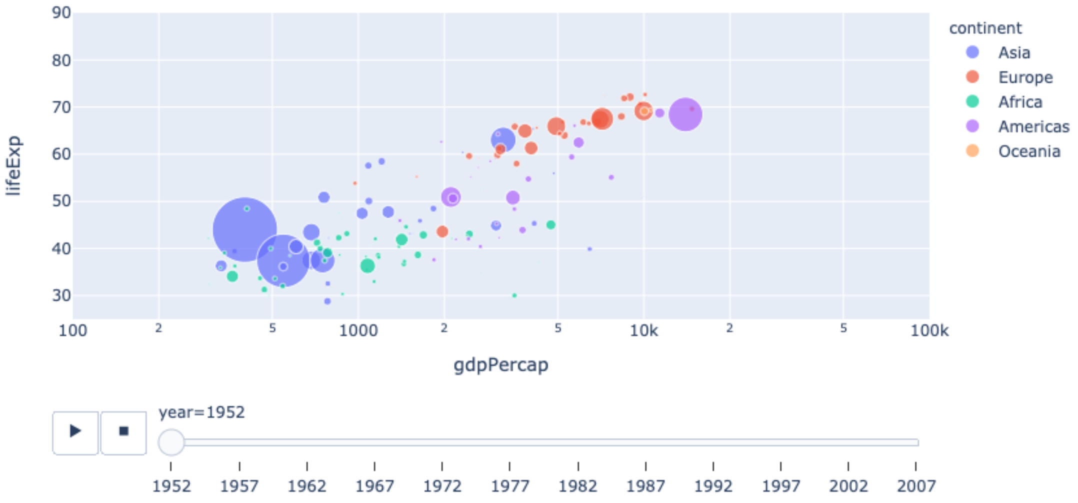
```
df_cnt = px.data.gapminder()
df_cnt.head()
```

[230]:

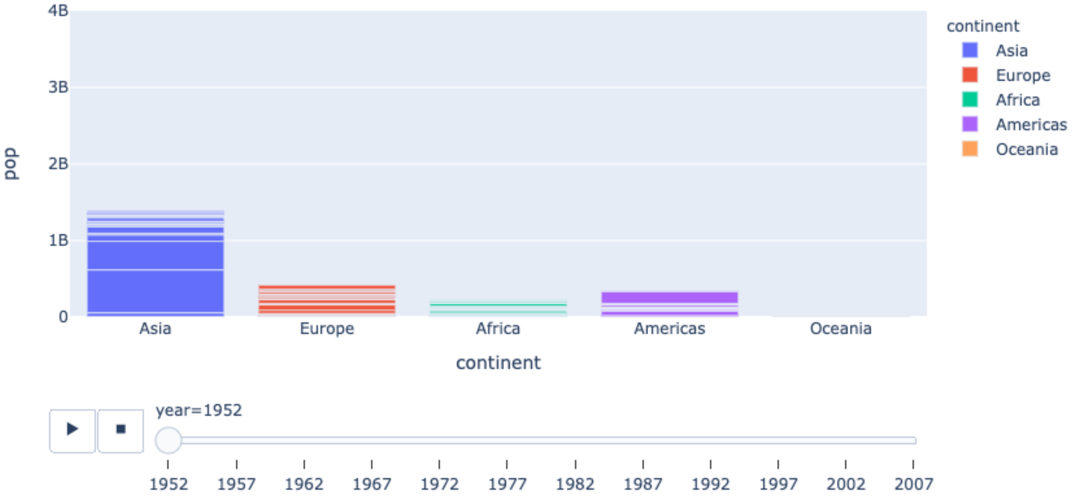
```
       country continent  year  lifeExp       pop   gdpPercap iso_alpha  \
0  Afghanistan      Asia  1952   28.801   8425333  779.445314       AFG
1  Afghanistan      Asia  1957   30.332   9240934  820.853030       AFG
2  Afghanistan      Asia  1962   31.997  10267083  853.100710       AFG
3  Afghanistan      Asia  1967   34.020  11537966  836.197138       AFG
4  Afghanistan      Asia  1972   36.088  13079460  739.981106       AFG

   iso_num
0        4
1        4
2        4
3        4
4        4
```

[235]:

```
px.scatter(
    df_cnt,
    x='gdpPercap',
    y='lifeExp',
    animation_frame='year',
    animation_group="country",
    size='pop',
    color='continent',
    hover_name='country',
    log_x=True,
    size_max=55,
    range_x=[100,100000],
    range_y=[25, 90]
)
```



[241]:
```
px.bar(
    df_cnt,
    x='continent',
    y='pop',
    color='continent',
    animation_frame='year',
    animation_group='country',
    range_y=[0, 4000000000]
)
```


4B
3B
2B
1B
0
pop
Asia
Europe
Africa
Americas
Oceania
continent
continent
Asia
Europe
Africa
Americas
Oceania
year=1952
1952
1957
1962
1967
1972
1977
1982
1987
1992
1997
2002
2007