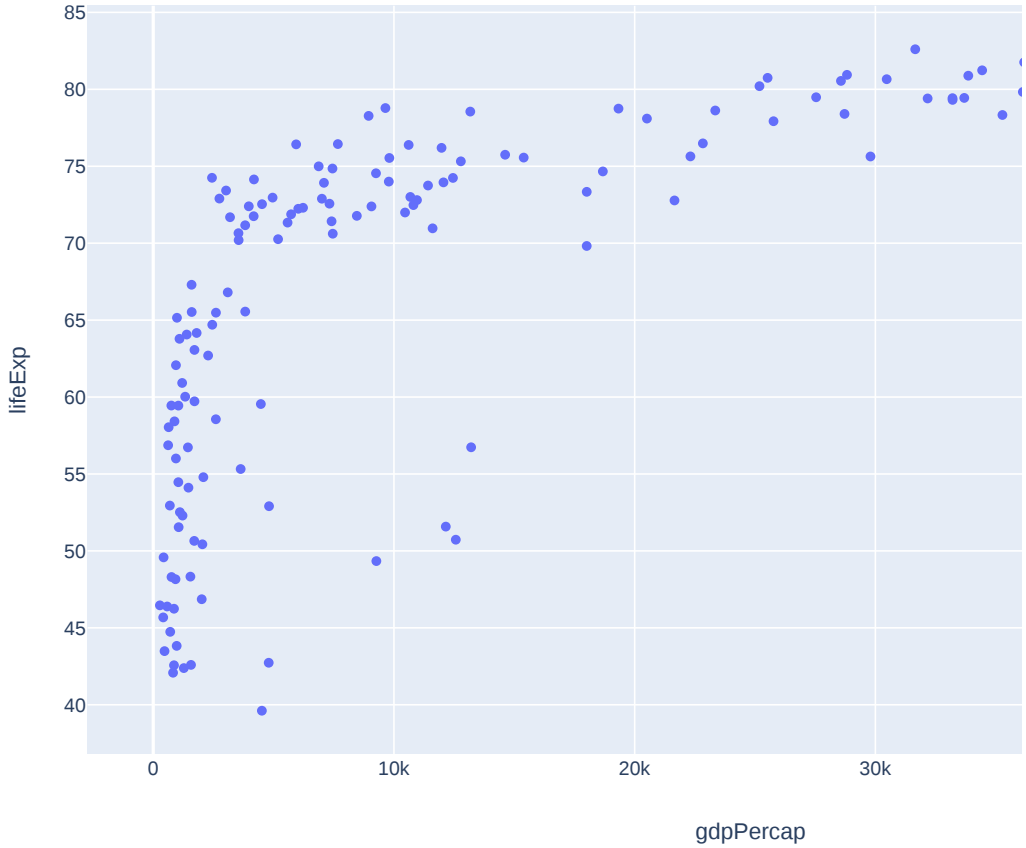
In [1]:
```
import plotly_express as px

gapminder = px.data.gapminder()
gapminder2007 = gapminder.query('year == 2007')
px.scatter(gapminder2007, x='gdpPercap', y='lifeExp')
```

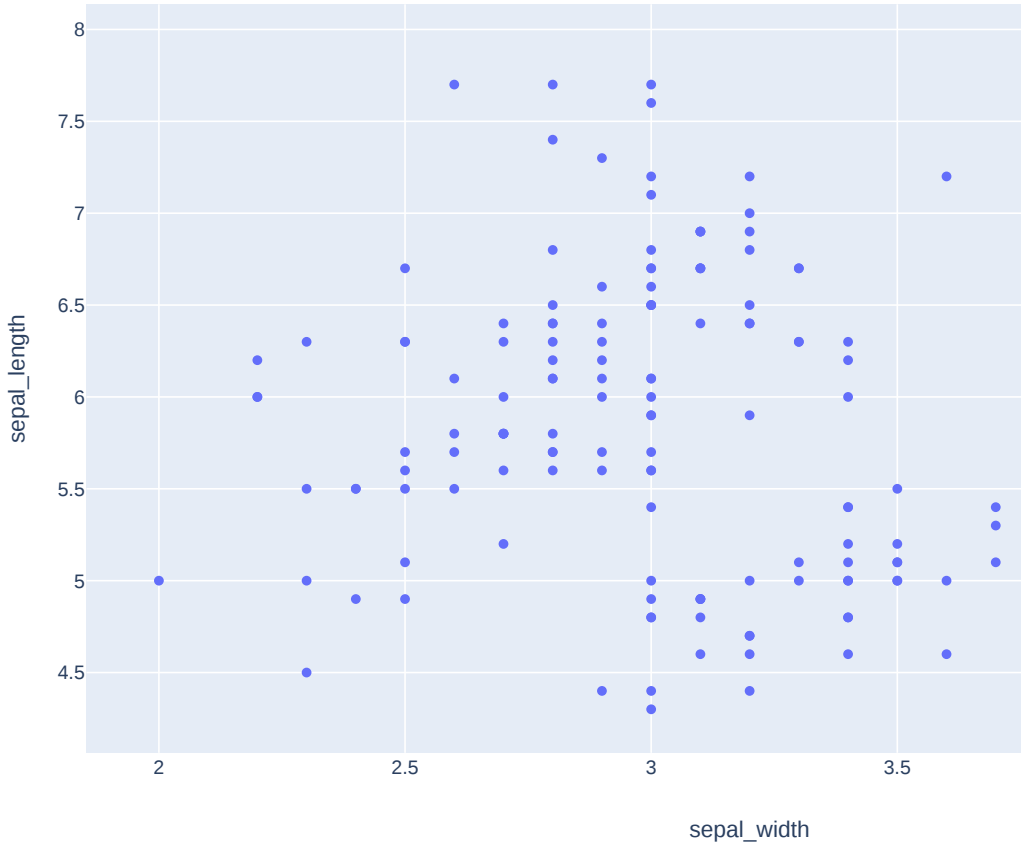


In [2]:
```
print(px.data.iris.__doc__)
iris = px.data.iris()
```

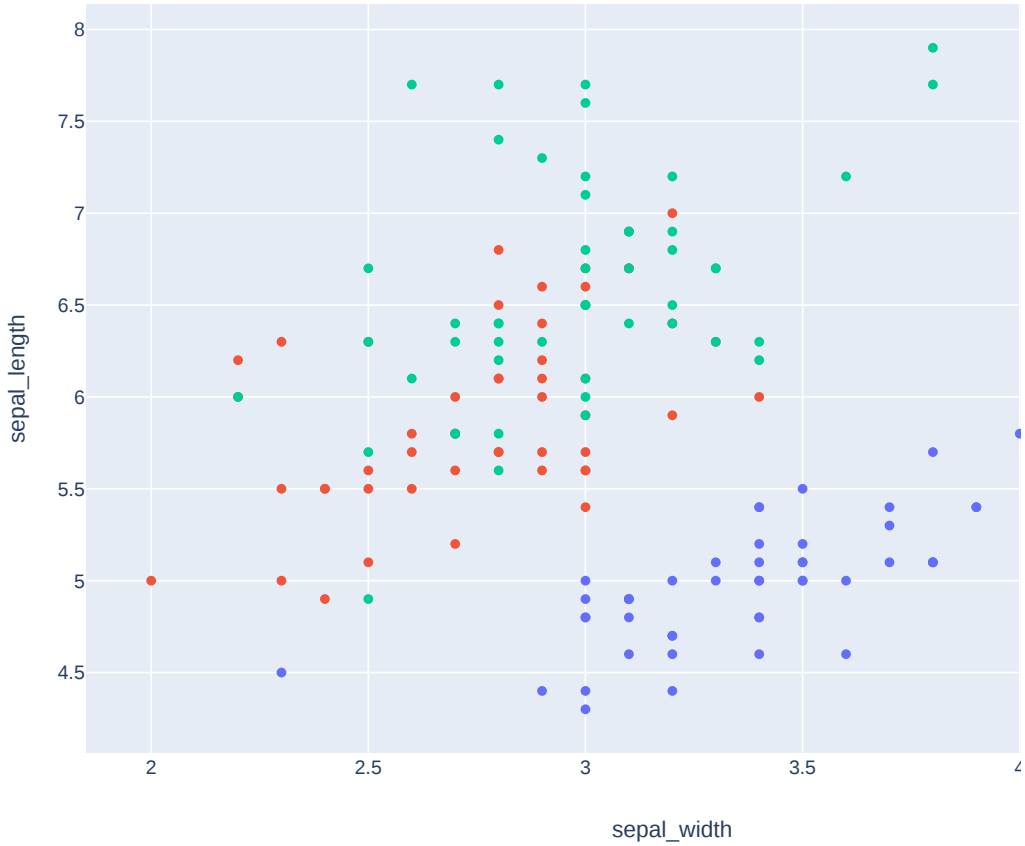
```
    Each row represents a flower.

    https://en.wikipedia.org/wiki/Iris_flower_data_set

    Returns:
        A `pandas.DataFrame` with 150 rows and the following columns: `['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species',
       'species_id']`.
```

In [3]:
```
tips = px.data.tips()
gapminder = px.data.gapminder()
election = px.data.election()
wind = px.data.wind()
carshare = px.data.carshare()
```

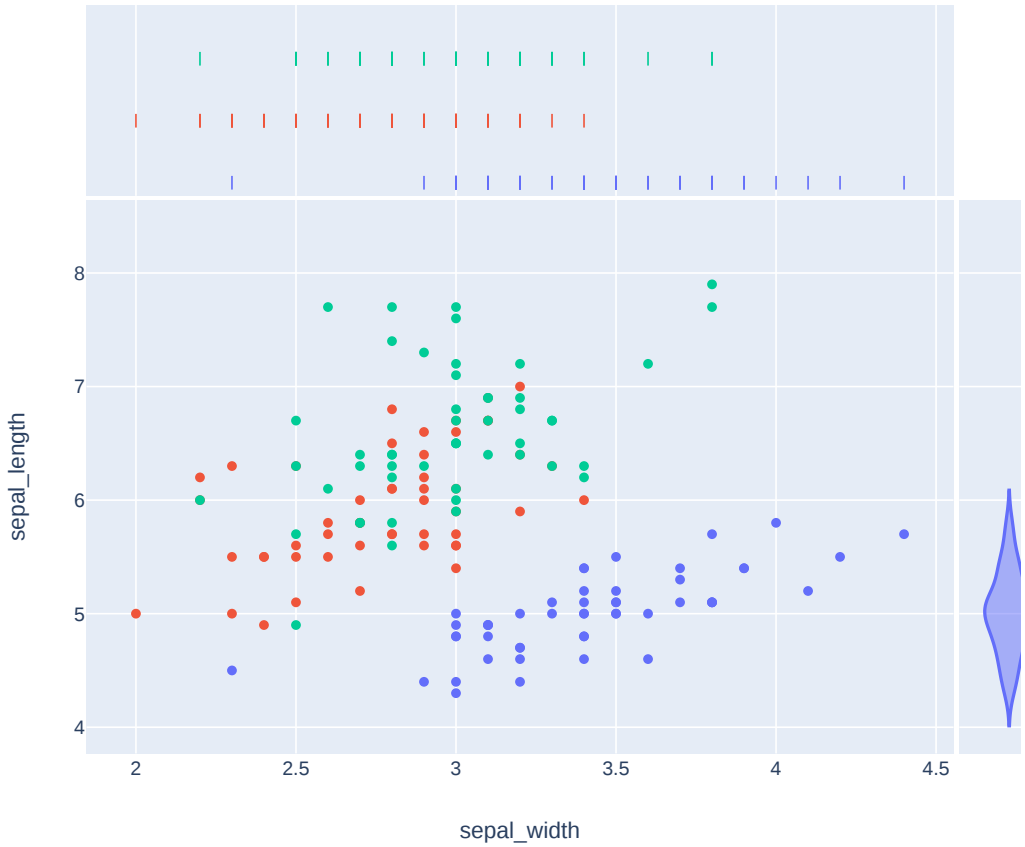
In [4]:
```
px.scatter(iris, x="sepal_width", y="sepal_length")
```



In [5]:
```
px.scatter(iris, x="sepal_width", y="sepal_length", color="species")
```

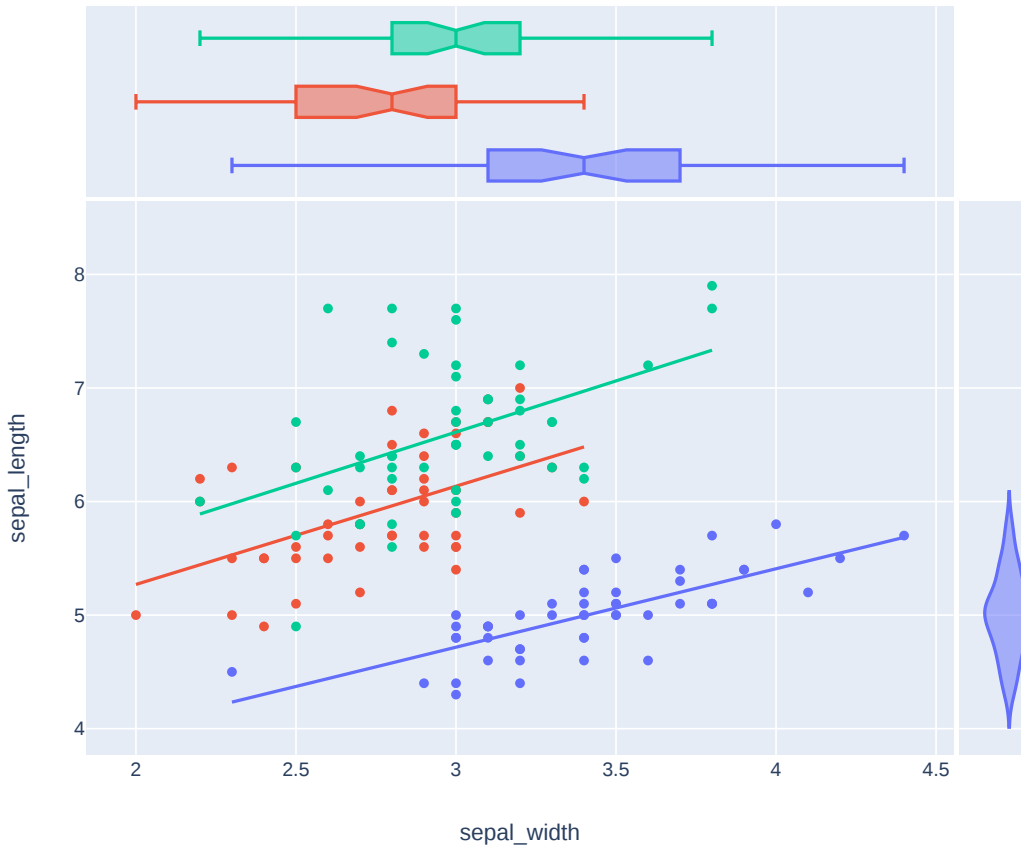


In [6]:
```
px.scatter(iris, x="sepal_width", y="sepal_length", color="species", marginal_y="violin", marginal_x="rug")
```

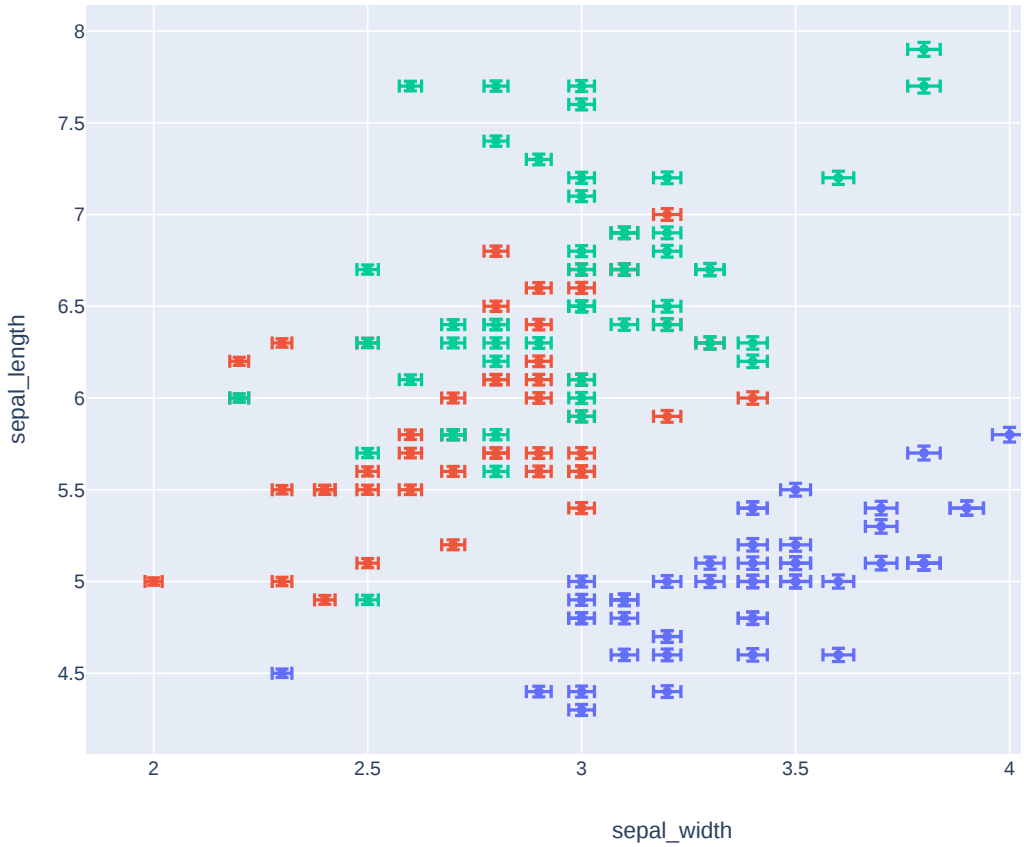


In [7]:
```
px.scatter(iris, x="sepal_width", y="sepal_length", color="species", marginal_y="violin",
           marginal_x="box", trendline="ols")
```

```
/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389:
FutureWarning:

Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389:
FutureWarning:

Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389:
FutureWarning:

Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.
```



In [8]:
```
iris["e"] = iris["sepal_width"]/100
px.scatter(iris, x="sepal_width", y="sepal_length", color="species", error_x="e", error_y="e")
```



In [9]:
```
del iris["e"]
```

In [10]:
```
px.scatter(tips, x="total_bill", y="tip", facet_row="time", facet_col="day", color="smoker", trendline="ols",
        category_orders={"day": ["Thur", "Fri", "Sat", "Sun"], "time": ["Lunch", "Dinner"]})
```

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389: FutureWarning:

Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389: FutureWarning:

Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389: FutureWarning:

Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389: FutureWarning:

Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389: FutureWarning:

Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389: FutureWarning:

Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389: FutureWarning:

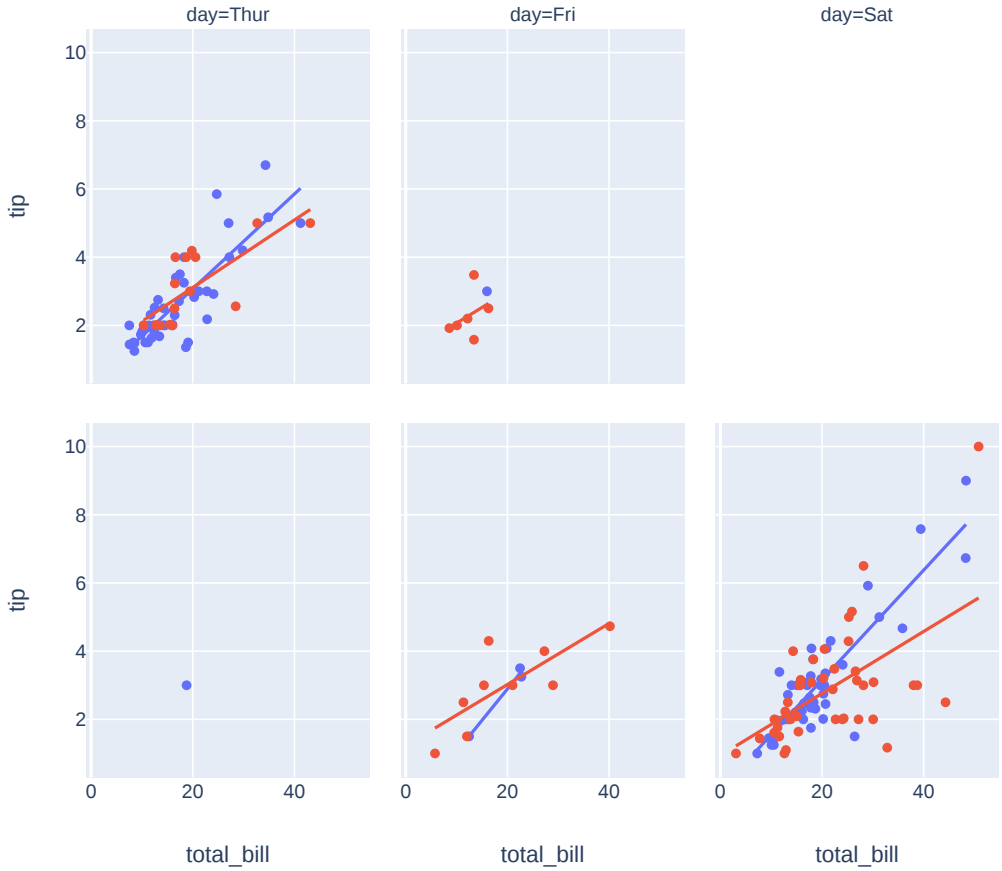
Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389: FutureWarning:

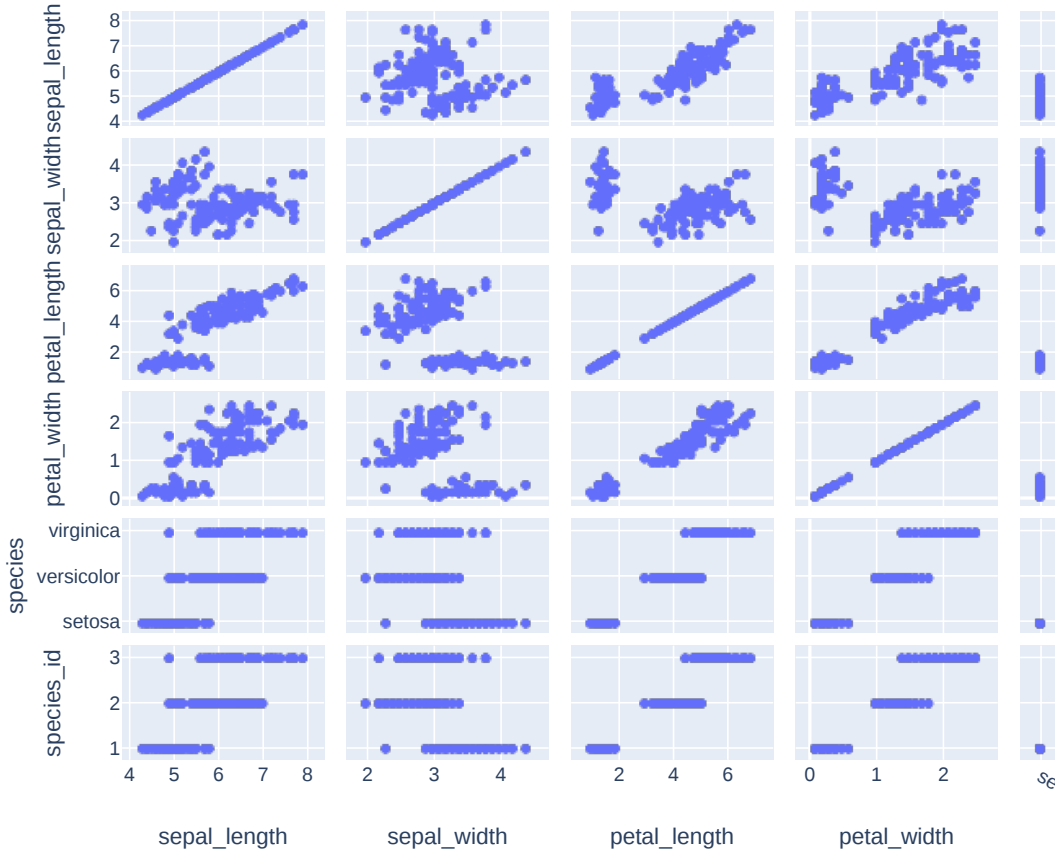
Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.

/home/young/miniconda3/envs/data/lib/python3.6/site-packages/numpy/core/fromnumeric.py:2389: FutureWarning:

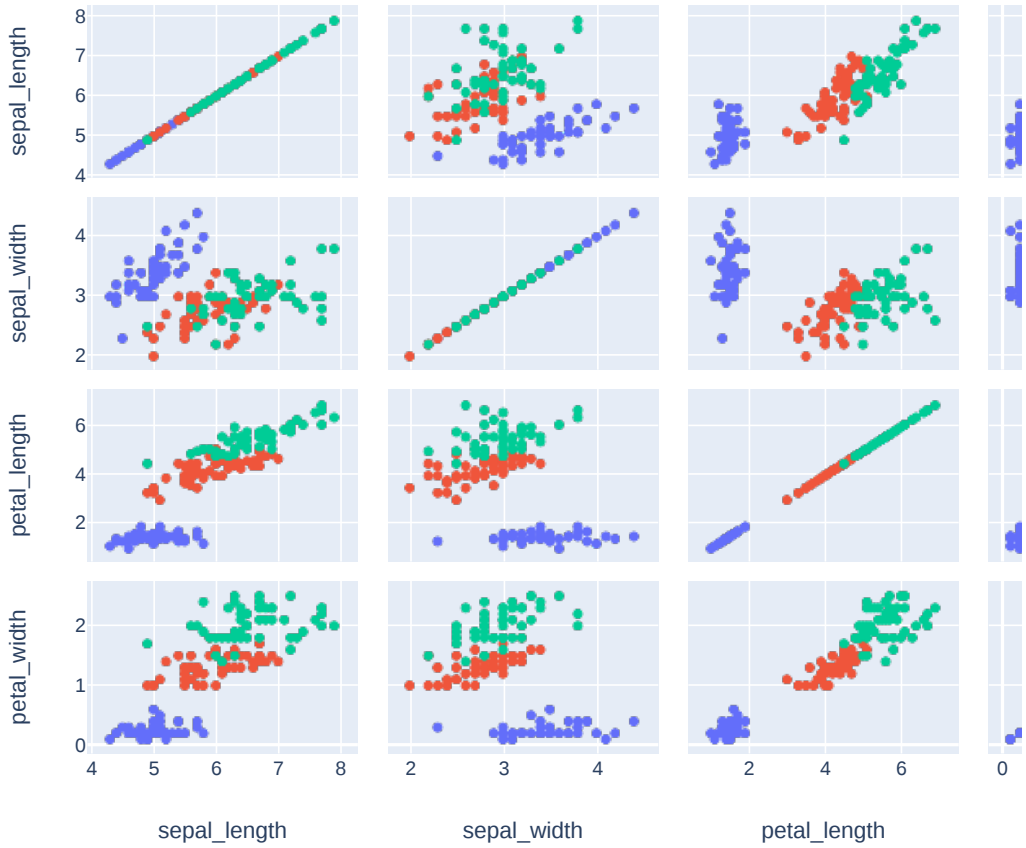
Method .ptp is deprecated and will be removed in a future version. Use numpy.ptp instead.


day=Thur
day=Fri
day=Sat
tip
total_bill
10
8
6
4
2
0
20
40

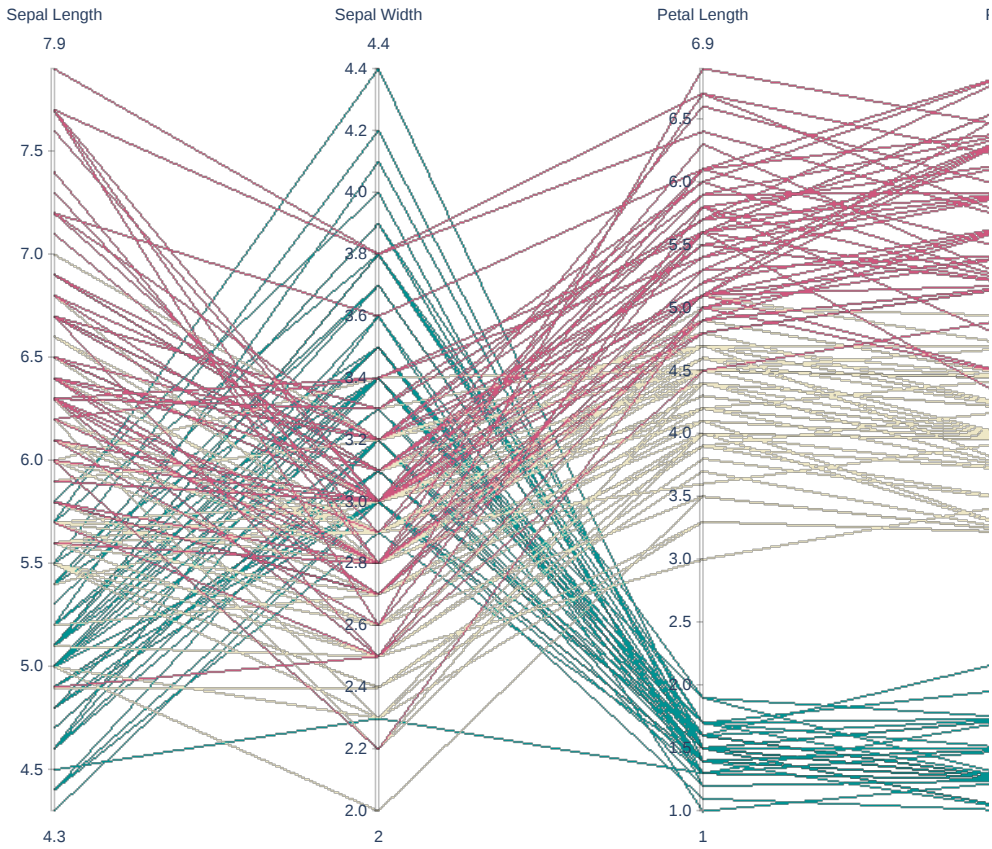
In [11]:
```
px.scatter_matrix(iris)
```



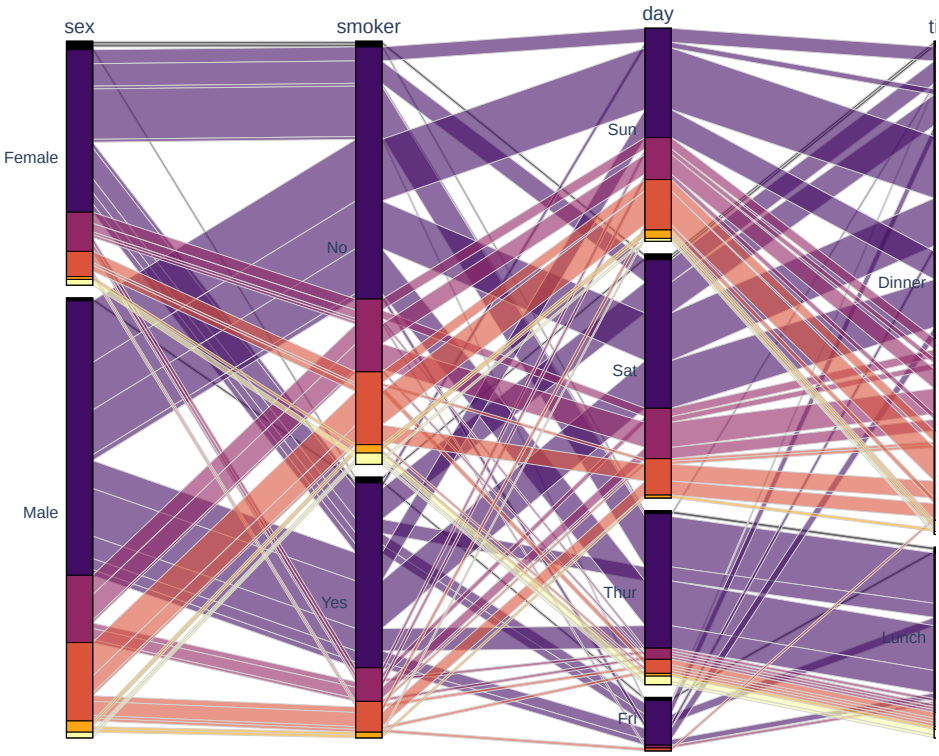
In [12]:
```
px.scatter_matrix(iris, dimensions=["sepal_width", "sepal_length", "petal_width", "petal_length"], color="species")
```



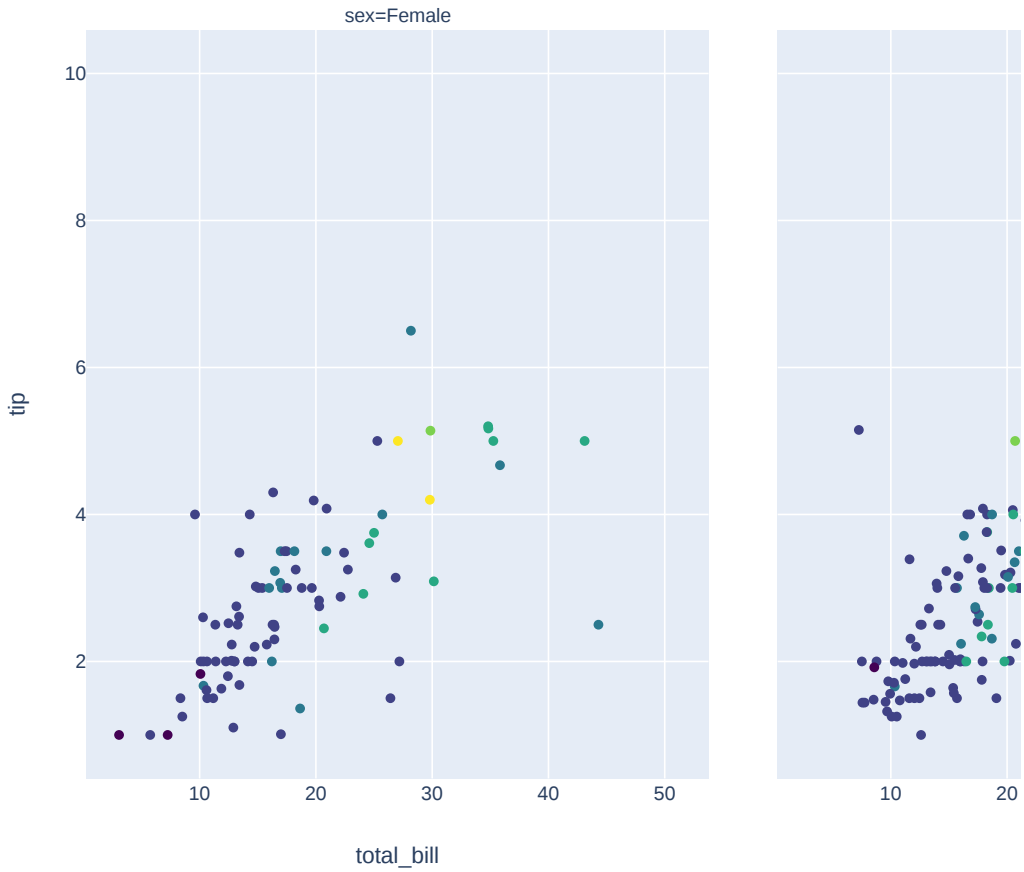
In [13]:
```
px.parallel_coordinates(iris, color="species_id", labels={"species_id": "Species",
                "sepal_width": "Sepal Width", "sepal_length": "Sepal Length",
                "petal_width": "Petal Width", "petal_length": "Petal Length", },
                 color_continuous_scale=px.colors.diverging.Tealrose, color_continuous_midpoint=2)
```



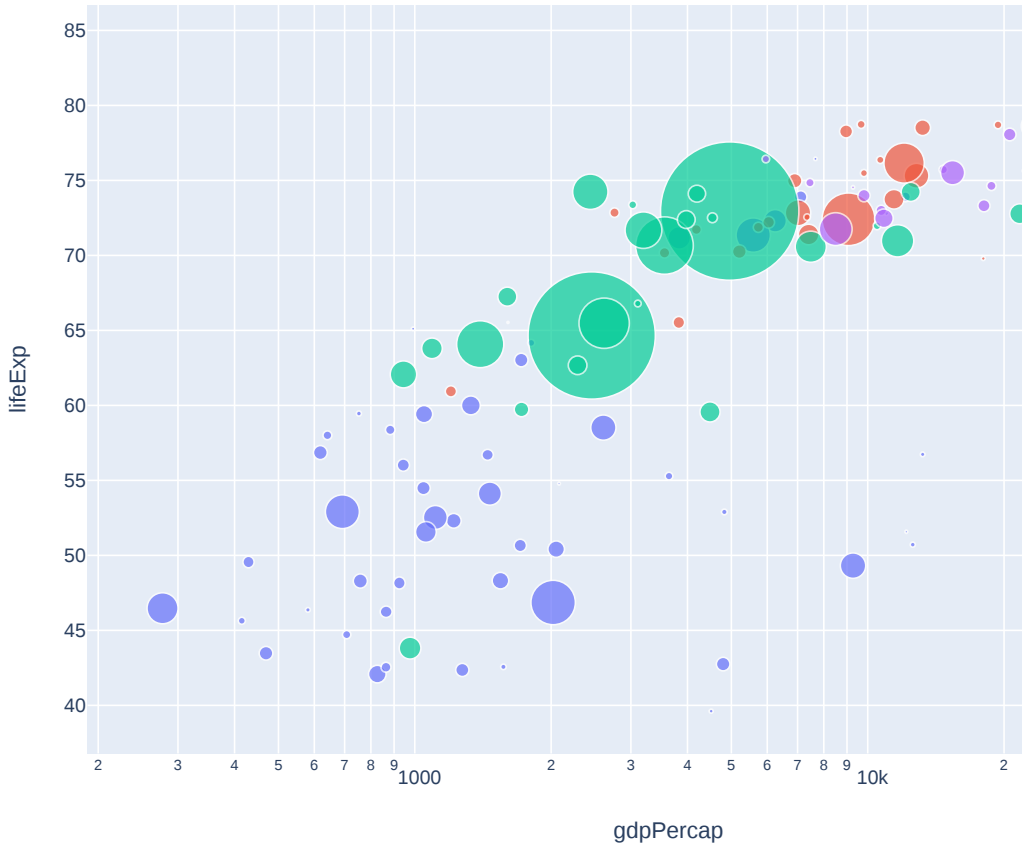
In [14]:
```
px.parallel_categories(tips, color="size", color_continuous_scale=px.colors.sequential.Inferno)
```



In [15]:
```
px.scatter(tips, x="total_bill", y="tip", color="size", facet_col="sex",
           color_continuous_scale=px.colors.sequential.Viridis, render_mode="webgl")
```

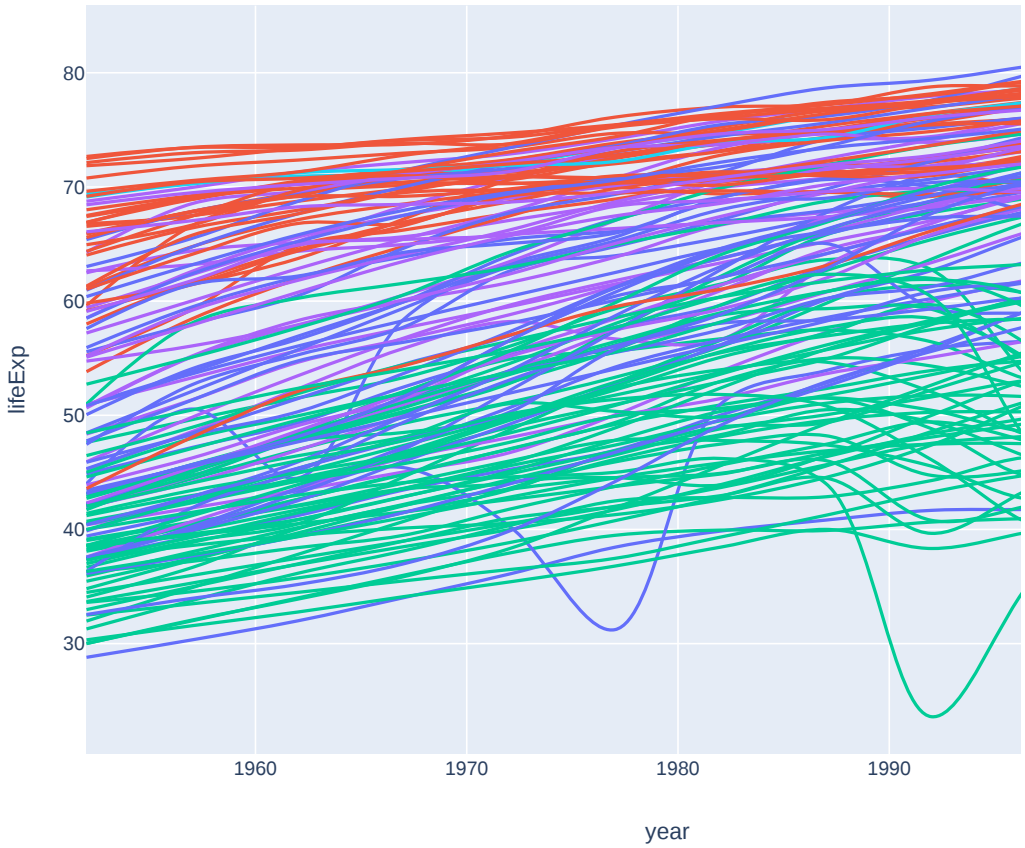


In [16]:
```
px.scatter(gapminder.query("year==2007"), x="gdpPercap", y="lifeExp", size="pop", color="continent",
           hover_name="country", log_x=True, size_max=60)
```

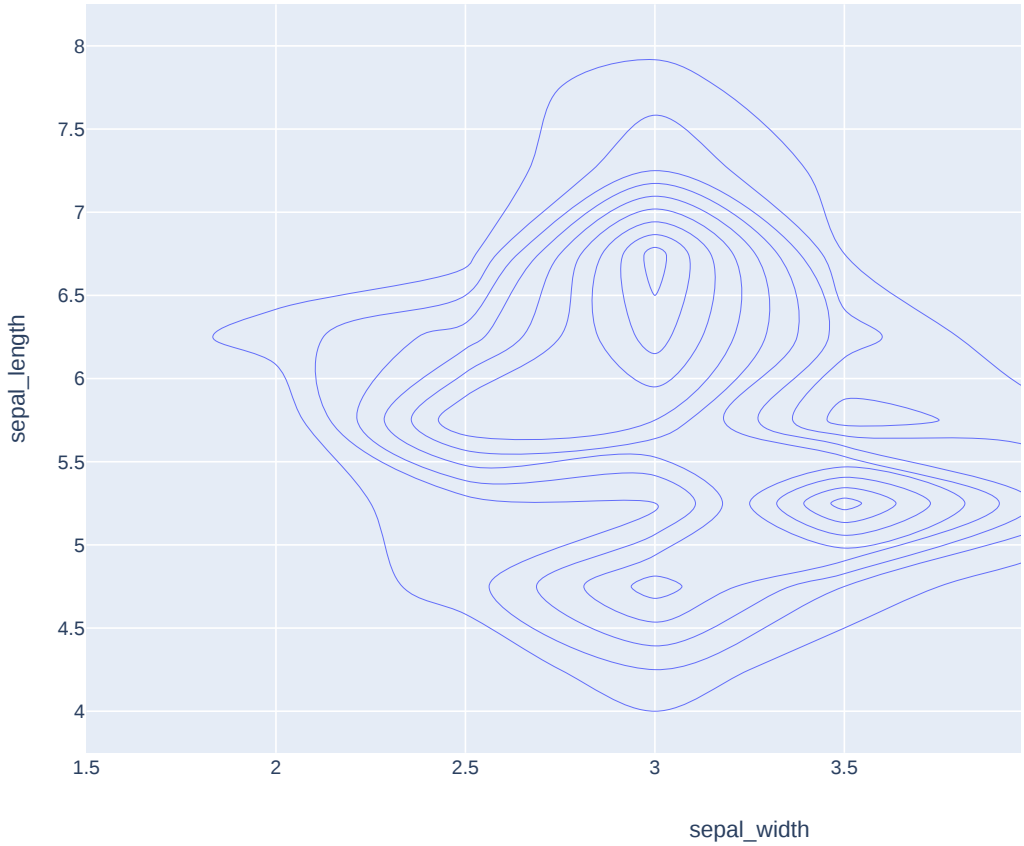


In [17]:
```
px.scatter(gapminder, x="gdpPercap", y="lifeExp", animation_frame="year", animation_group="country",
           size="pop", color="continent", hover_name="country", facet_col="continent",
           log_x=True, size_max=45, range_x=[100,100000], range_y=[25,90])
```

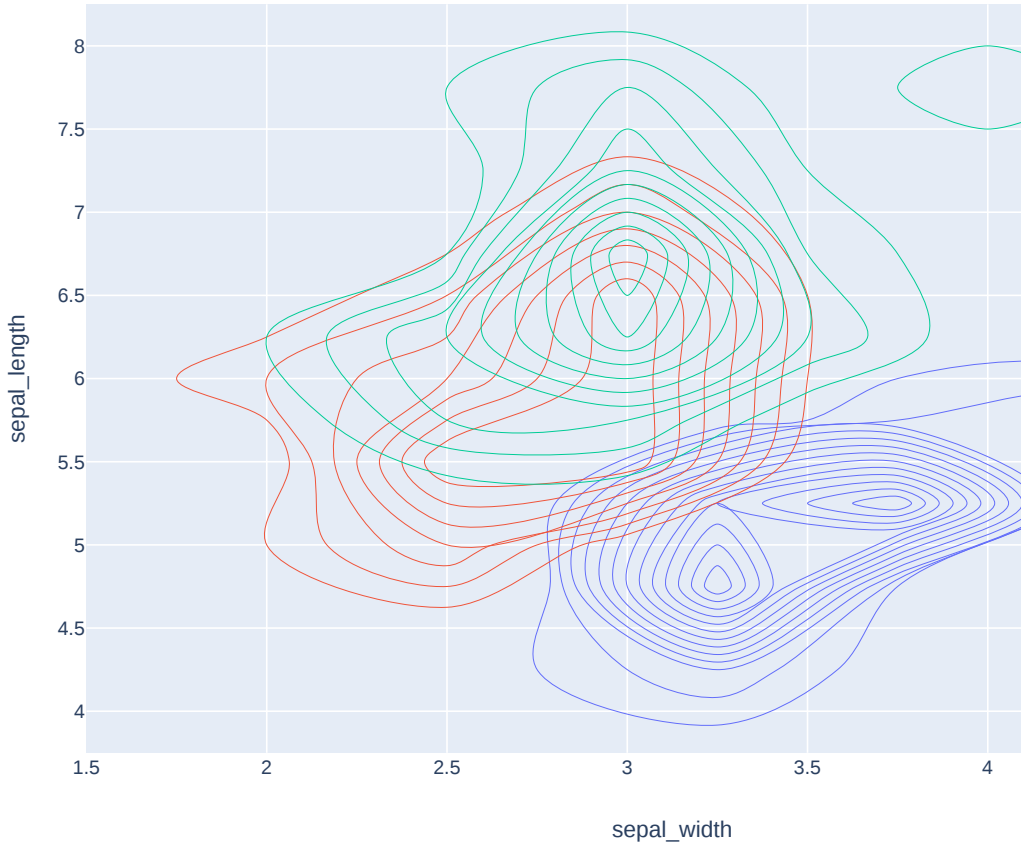
In [18]:
```
px.line(gapminder, x="year", y="lifeExp", color="continent", line_group="country", hover_name="country",
        line_shape="spline")
```



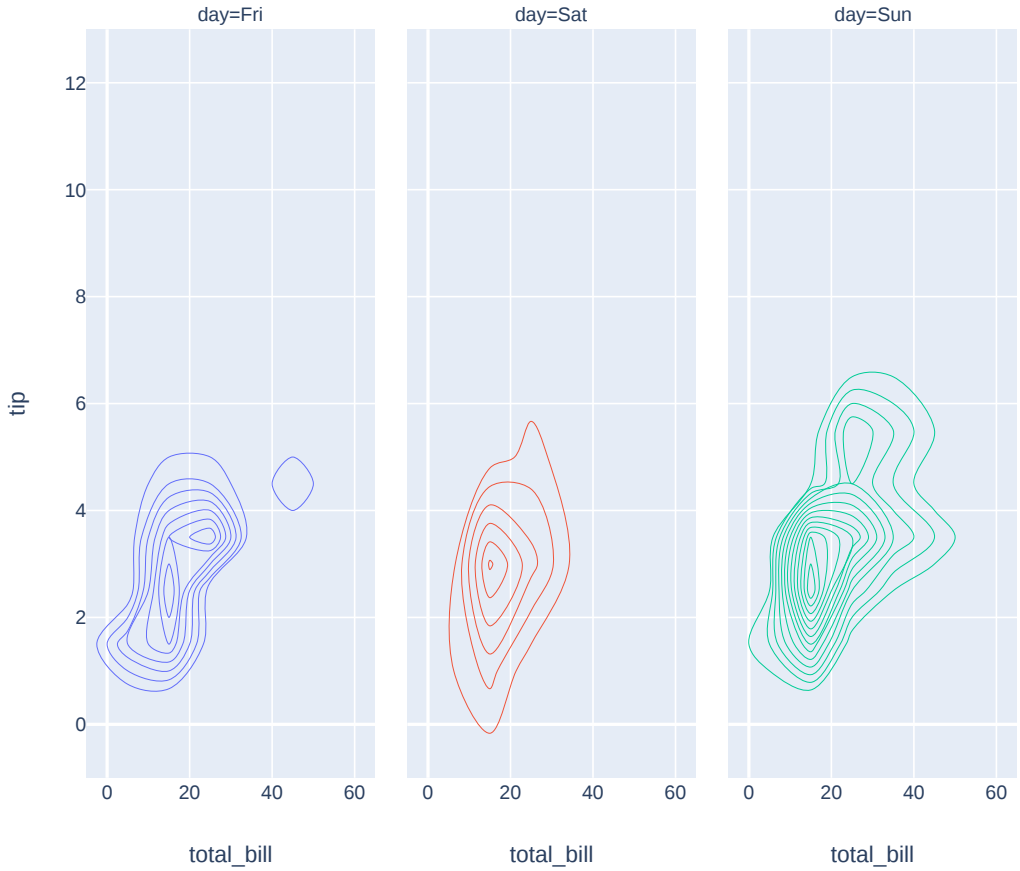
In [19]:
```
px.density_contour(iris, x="sepal_width", y="sepal_length")
```



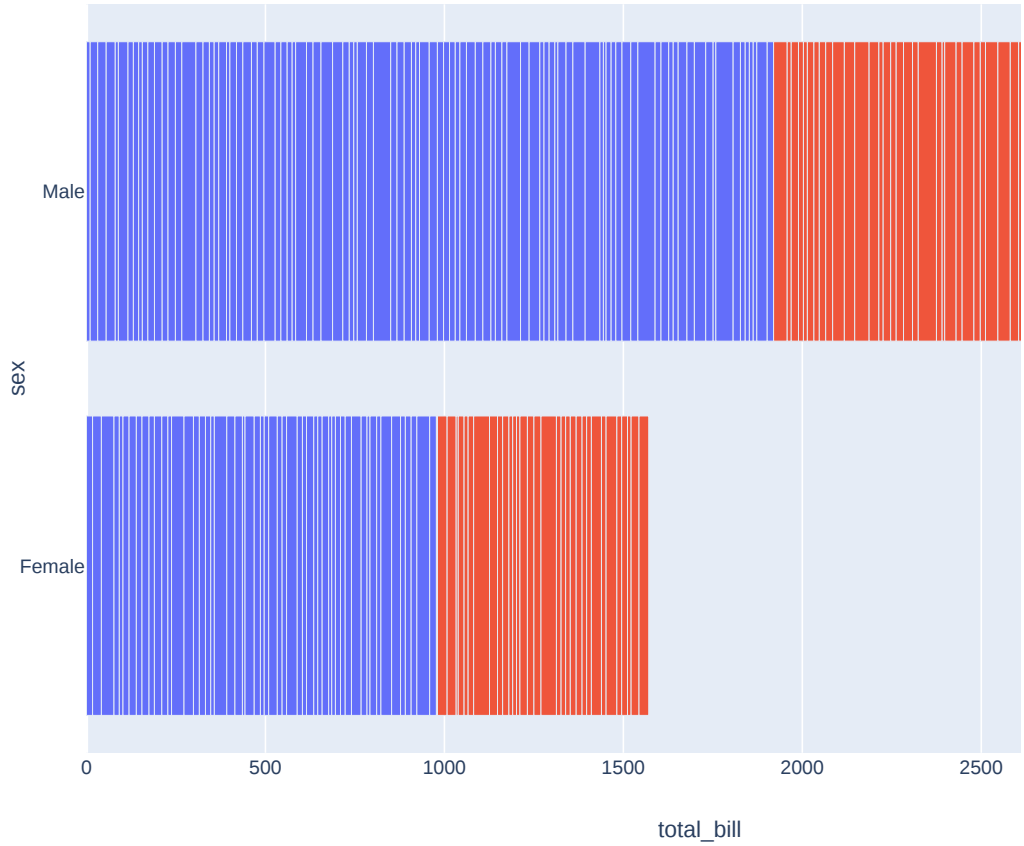
In [20]:
```
px.density_contour(iris, x="sepal_width", y="sepal_length", color="species")
```



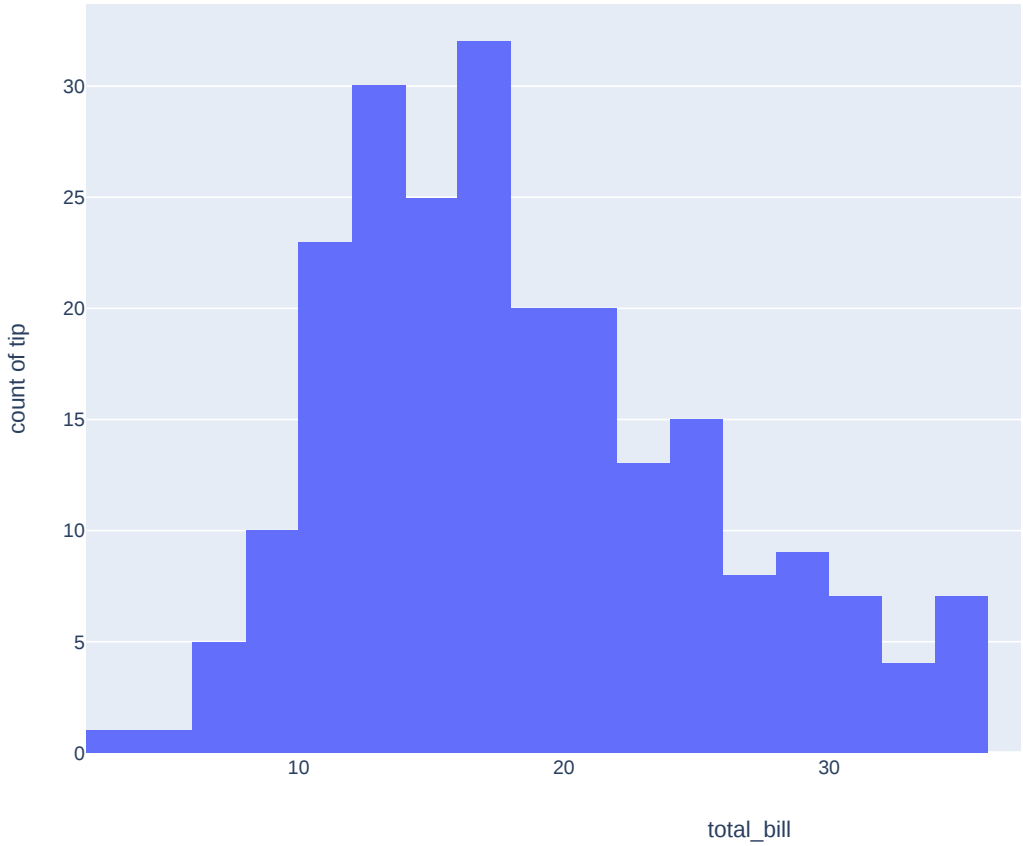
In [21]:
```
px.density_contour(tips, x="total_bill", y="tip", color="day", facet_col="day")
```



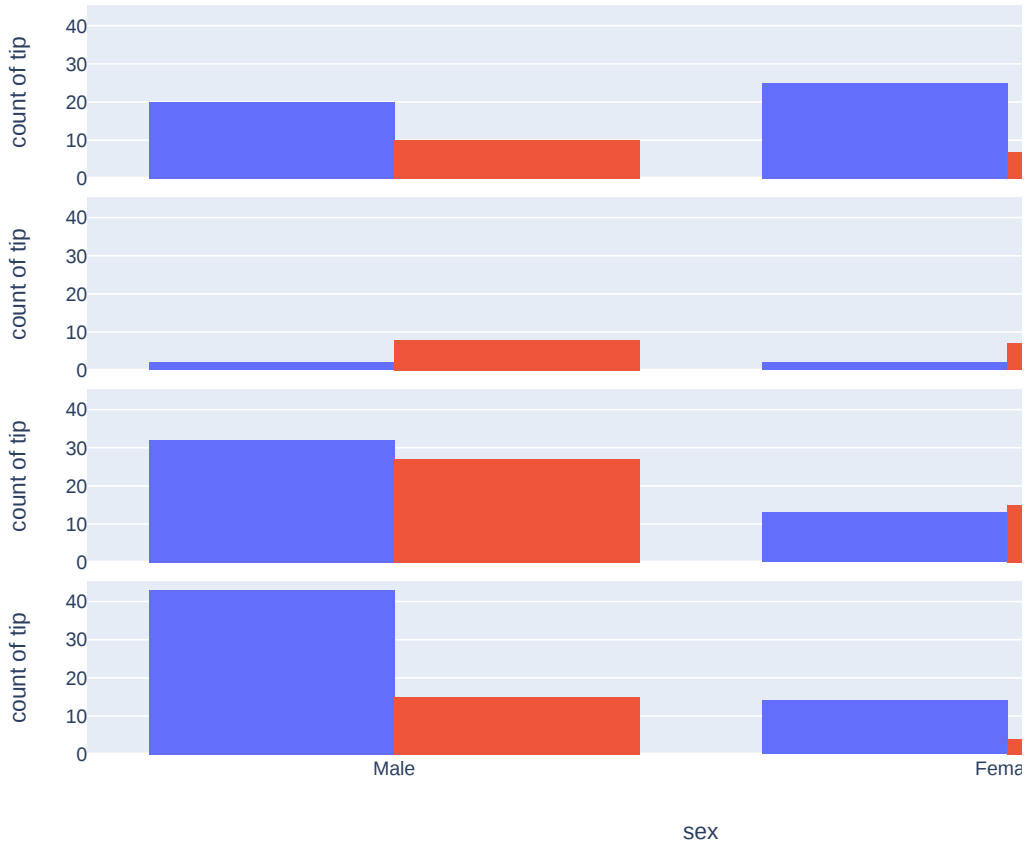
In [22]:
```
px.bar(tips, x="total_bill", y="sex", orientation="h", color="smoker")
```



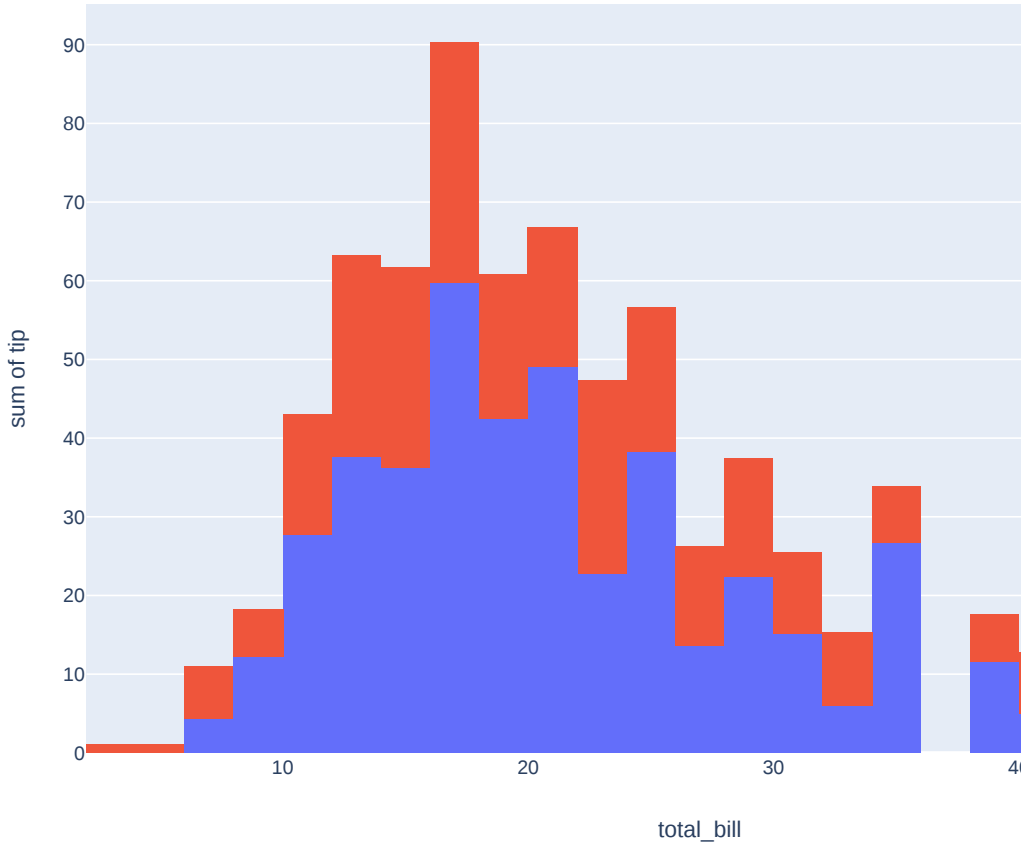
In [23]:
```
px.histogram(tips, x="total_bill", y="tip")
```



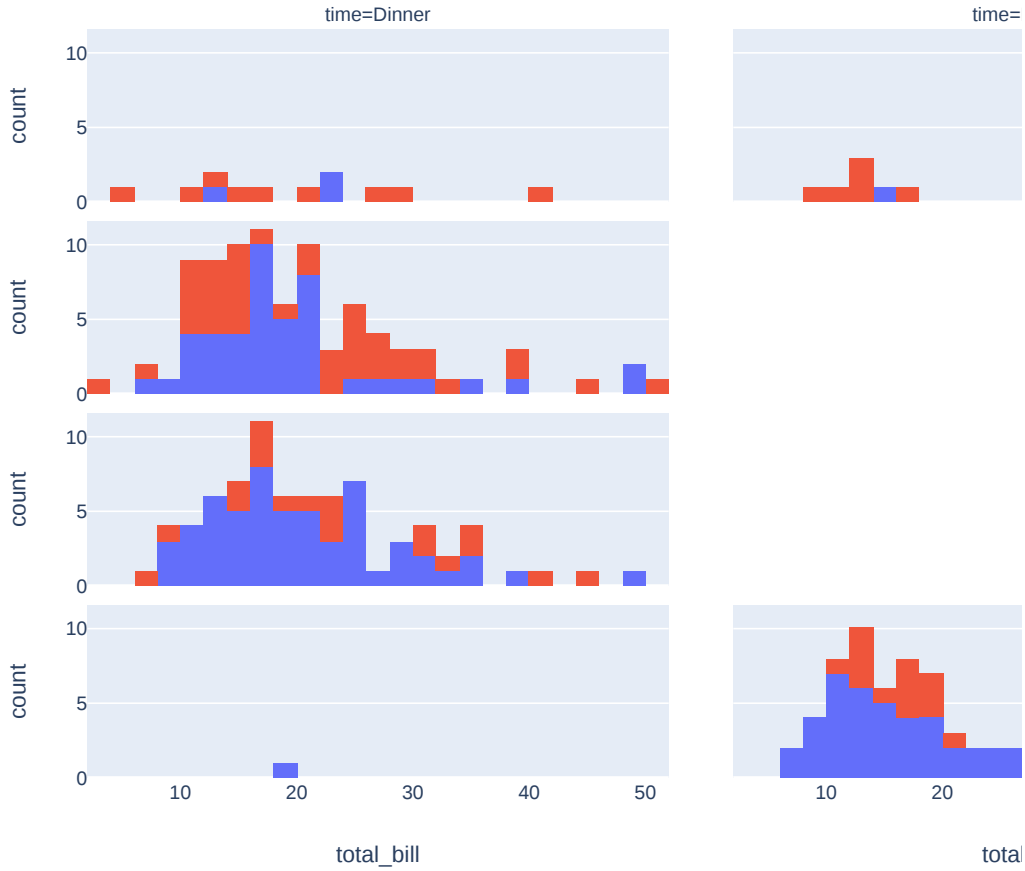
In [24]:
```
px.histogram(tips, x="sex", y="tip", color="smoker", facet_row="day", orientation="v", barmode="grou
p",
    category_orders={"day": ["Thur", "Fri", "Sat", "Sun"]})
```



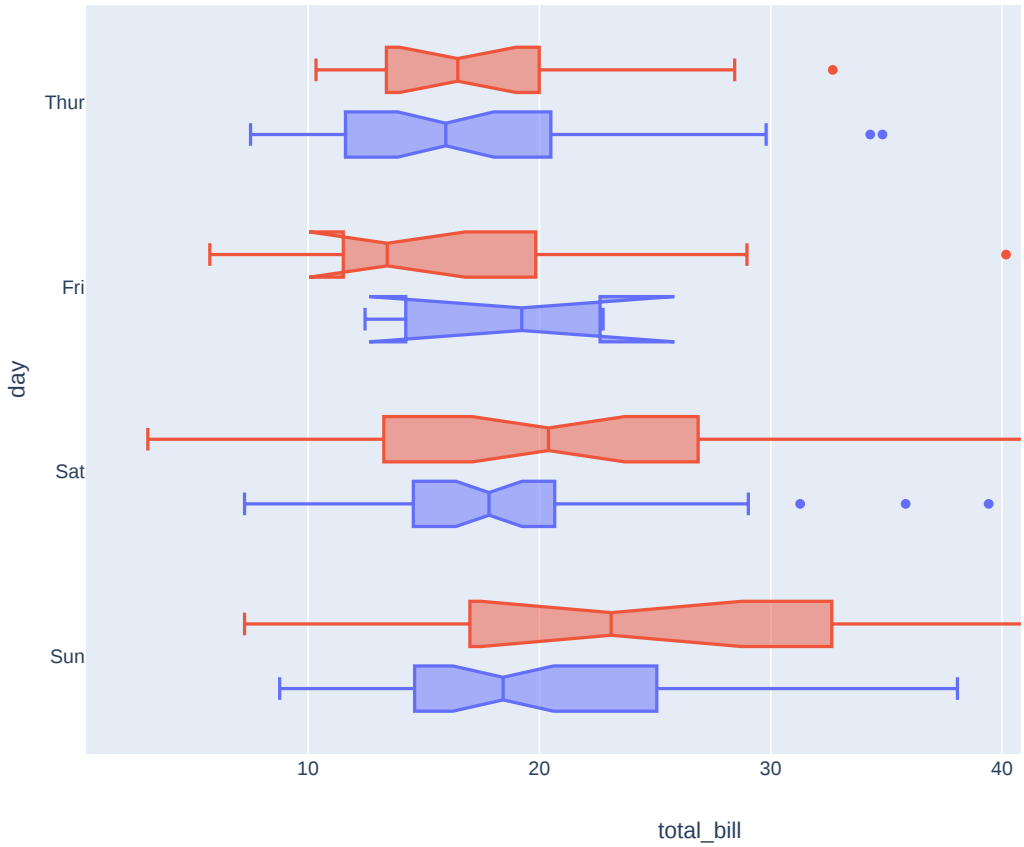
In [25]:
```
px.histogram(tips, x="total_bill", y="tip", histfunc="sum", color="smoker")
```



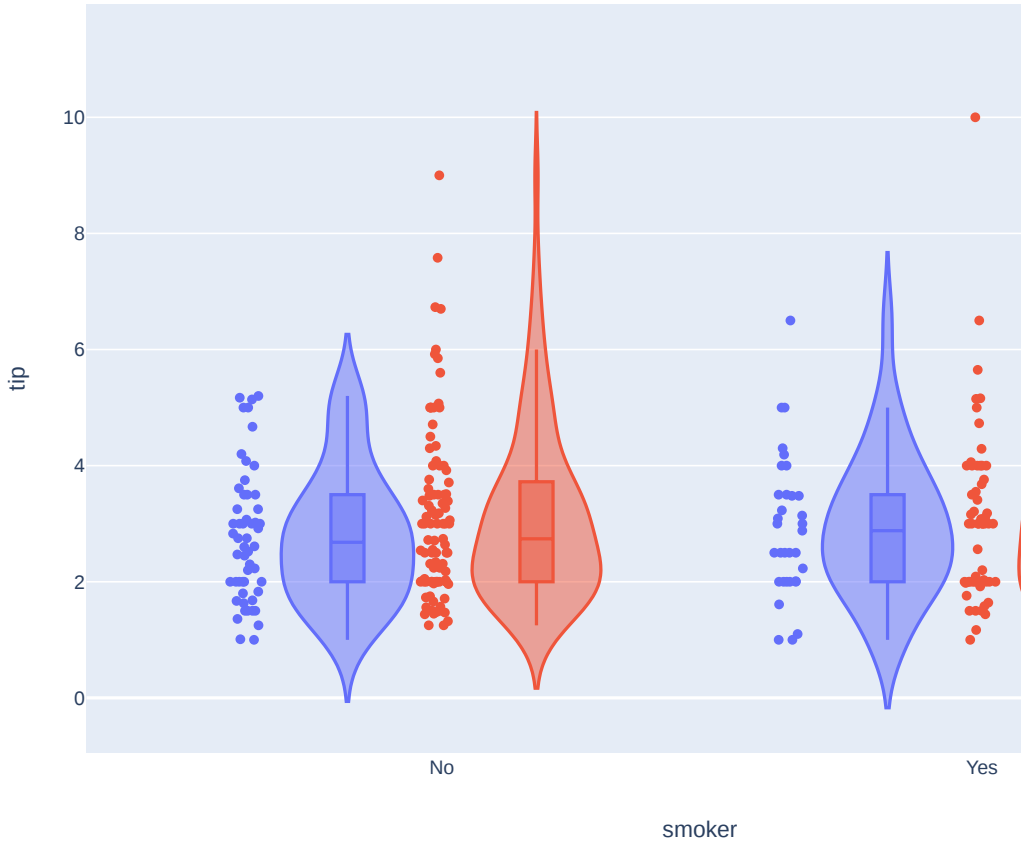
In [26]:
```
px.histogram(tips, x="total_bill", color="smoker", facet_row="day", facet_col="time")
```



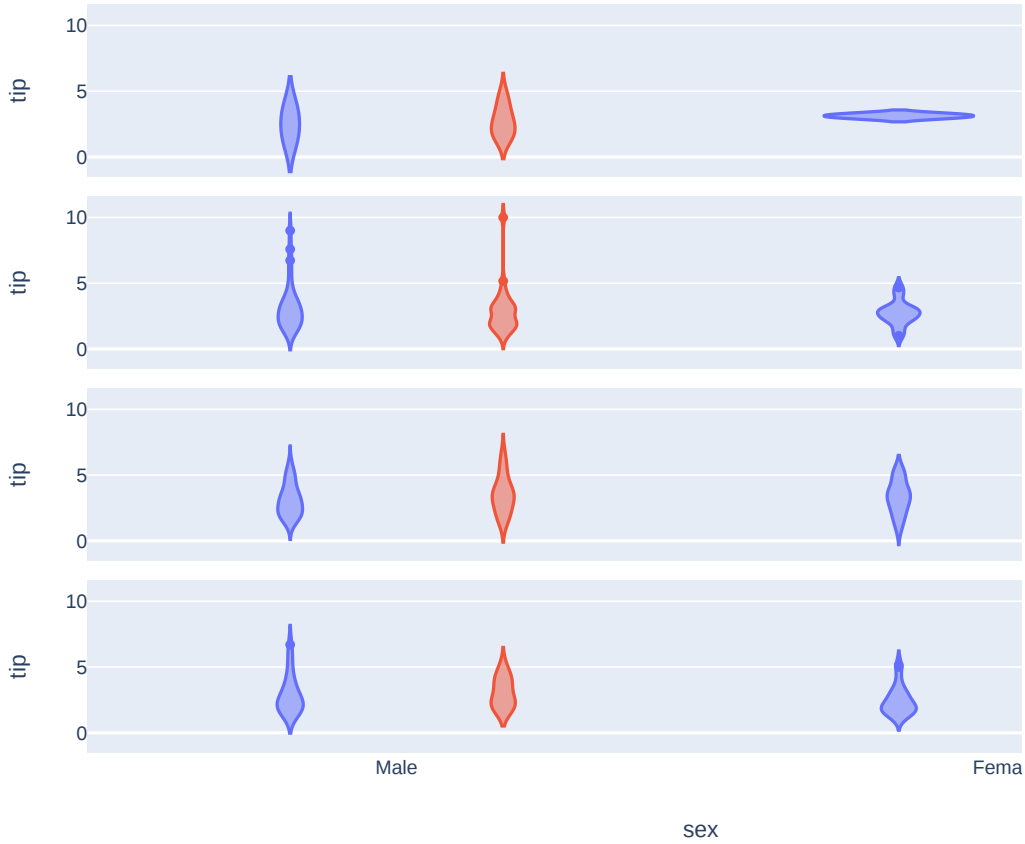
In [27]:
```
px.box(tips, x="total_bill", y="day", orientation="h", color="smoker", notched=True,
       category_orders={"day": ["Thur", "Fri", "Sat", "Sun"]})
```



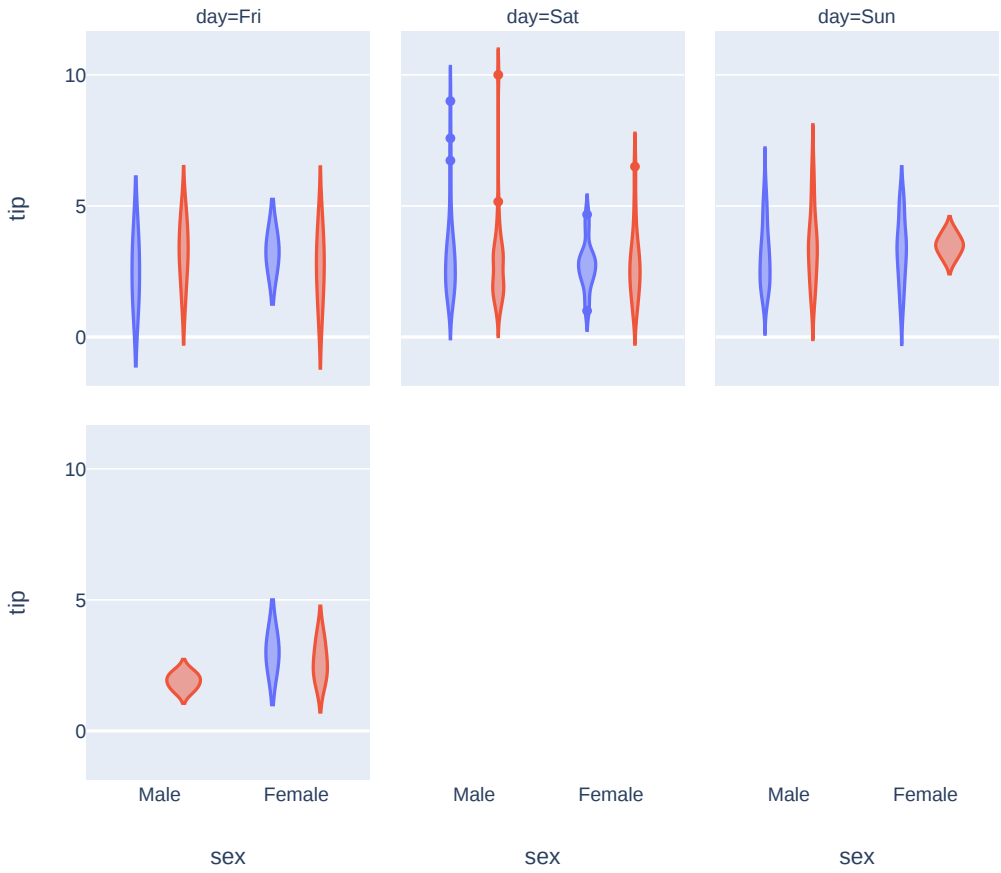
In [28]:
```
px.violin(tips, y="tip", x="smoker", color="sex", box=True, points="all")
```



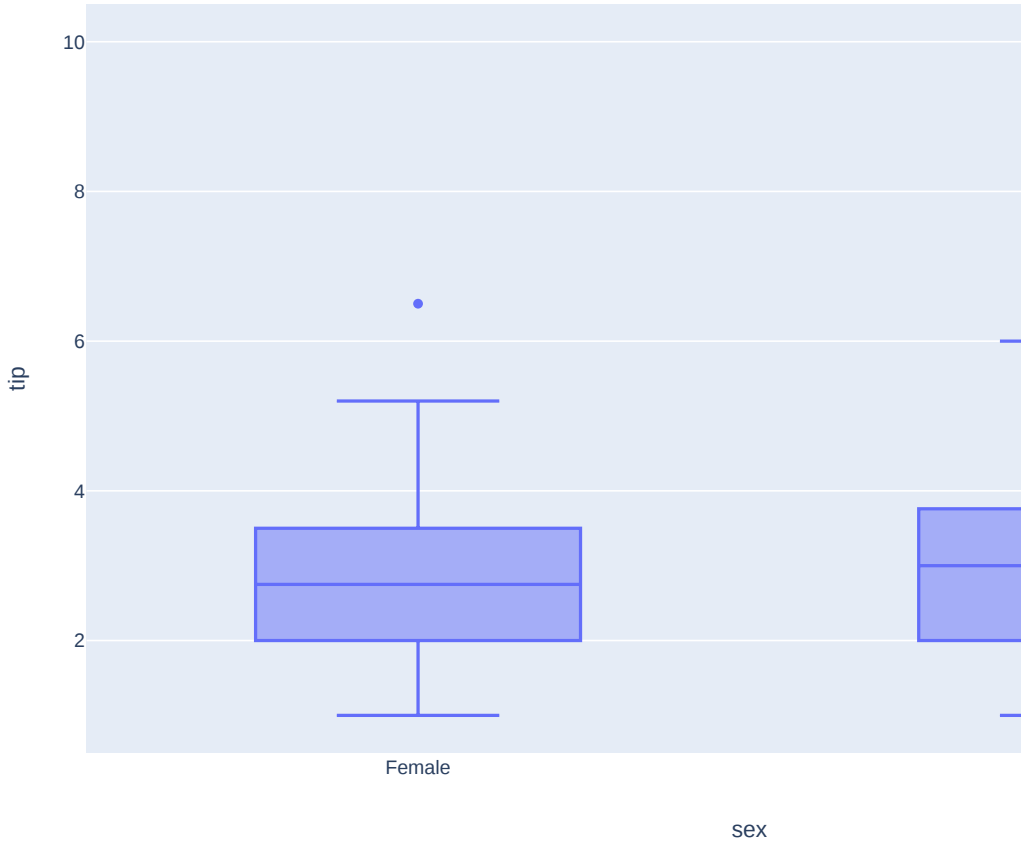
In [29]:
```
px.violin(tips, y="tip", x="sex", color="smoker", facet_row="day")
```



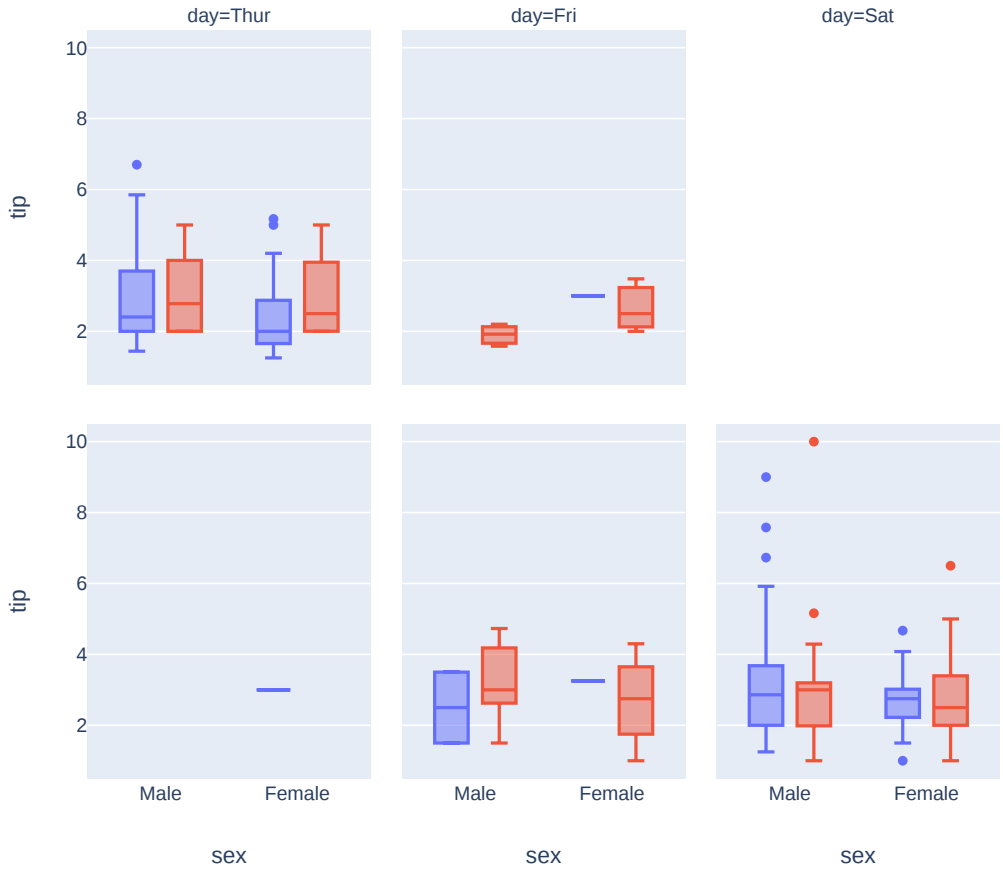
In [30]:
```
px.violin(tips, y="tip", x="sex", color="smoker", facet_col="day", facet_row="time")
```



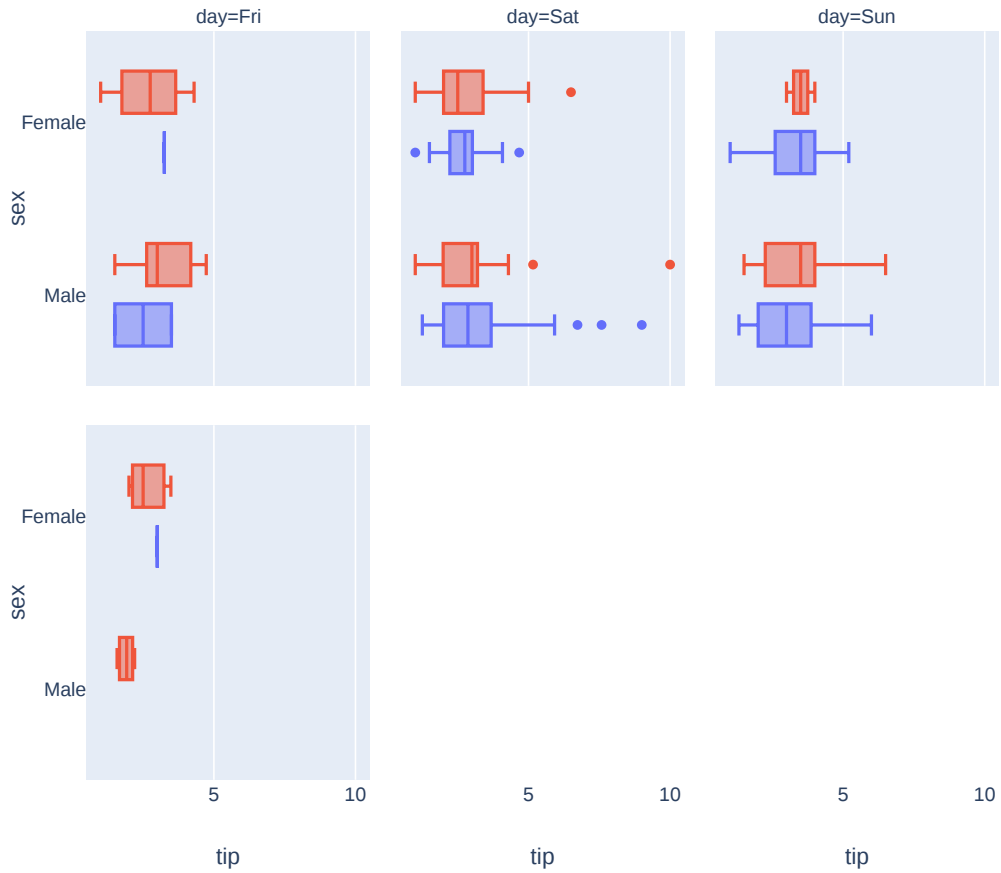
In [31]:
```
px.box(tips, y="tip", x="sex")
```



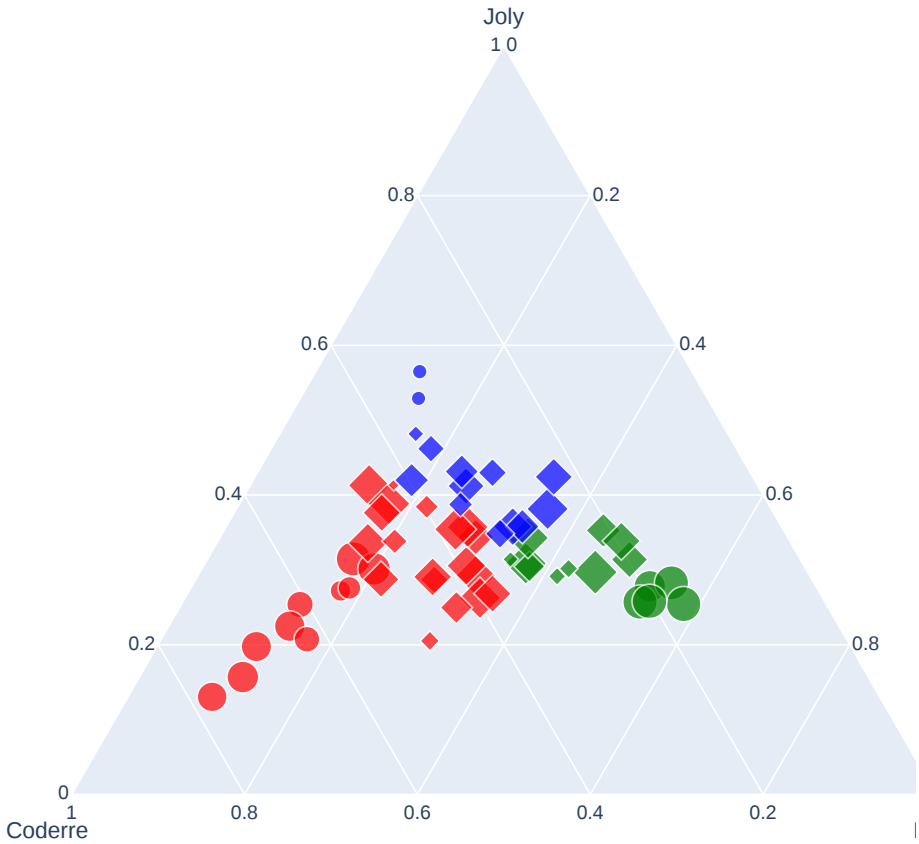
In [32]:
```
px.box(tips, y="tip", x="sex", color="smoker", facet_col="day", facet_row="time",
       category_orders={"day": ["Thur", "Fri", "Sat", "Sun"], "time": ["Lunch", "Dinner"]})
```



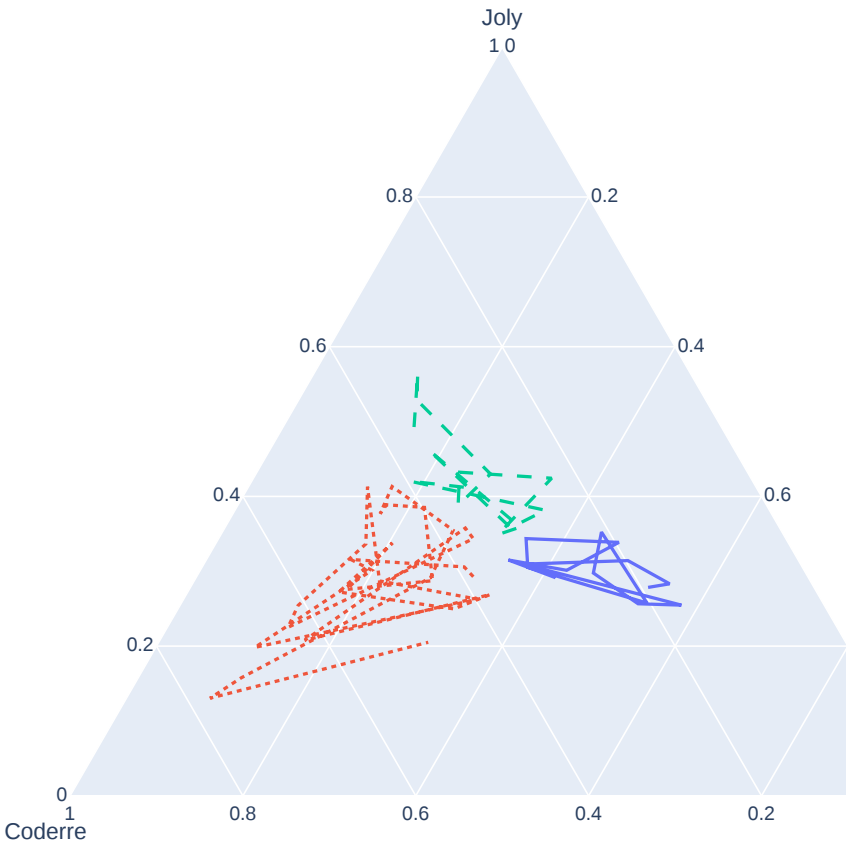
In [33]:
```
px.box(tips, x="tip", y="sex", color="smoker", facet_col="day", facet_row="time", orientation="h")
```



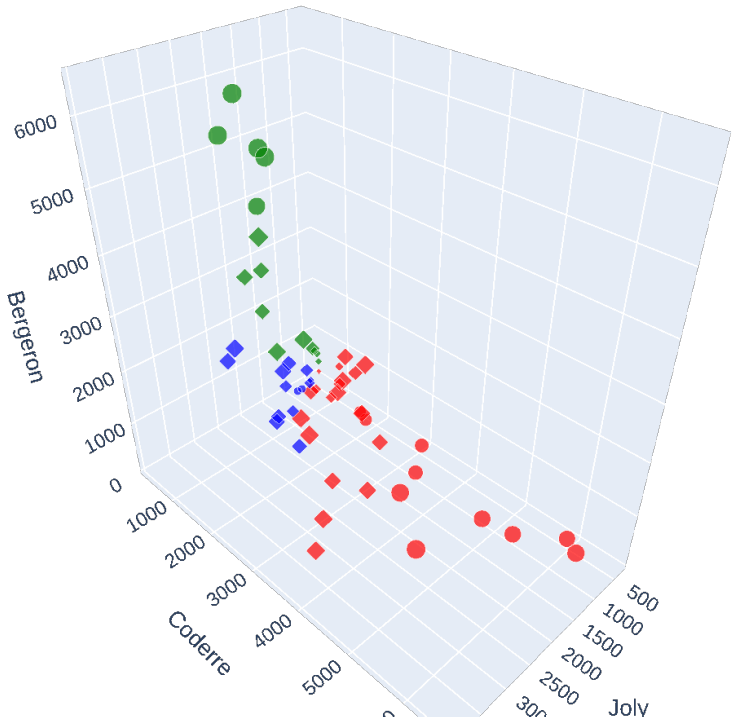
In [34]:
```
px.scatter_ternary(election, a="Joly", b="Coderre", c="Bergeron", color="winner", size="total", hover_name="district",
                   symbol="result",
                   size_max=15, color_discrete_map = {"Joly": "blue", "Bergeron": "green", "Coderre":"red"}
                  )
```



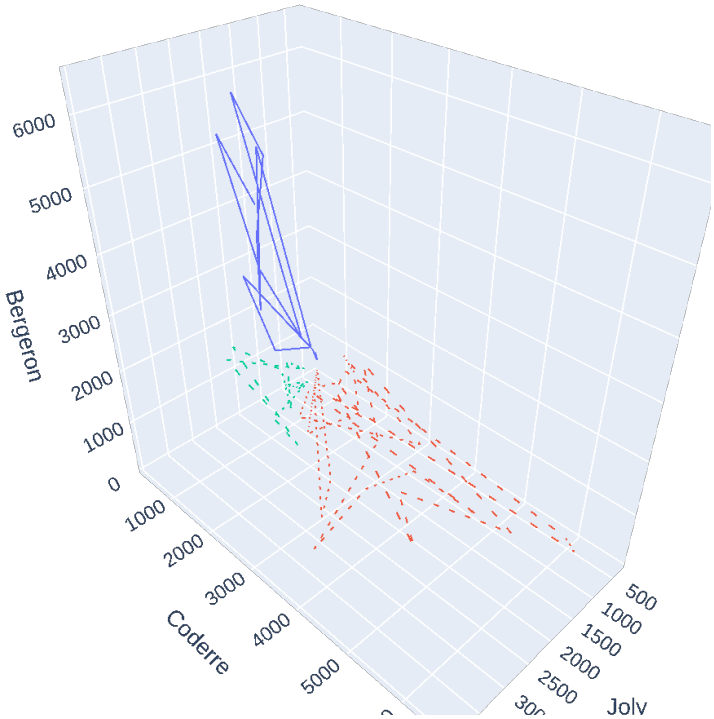
In [35]:
```
px.line_ternary(election, a="Joly", b="Coderre", c="Bergeron", color="winner", line_dash="winner")
```



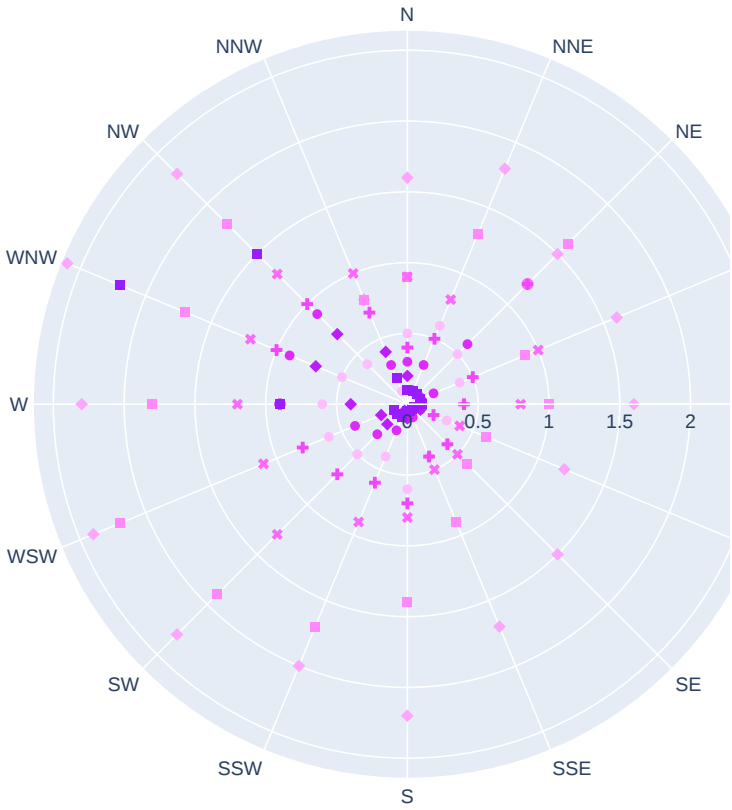
In [36]:
```
px.scatter_3d(election, x="Joly", y="Coderre", z="Bergeron", color="winner", size="total", hover_name="district",
                symbol="result", color_discrete_map = {"Joly": "blue", "Bergeron": "green", "Coderre":"red"})
```



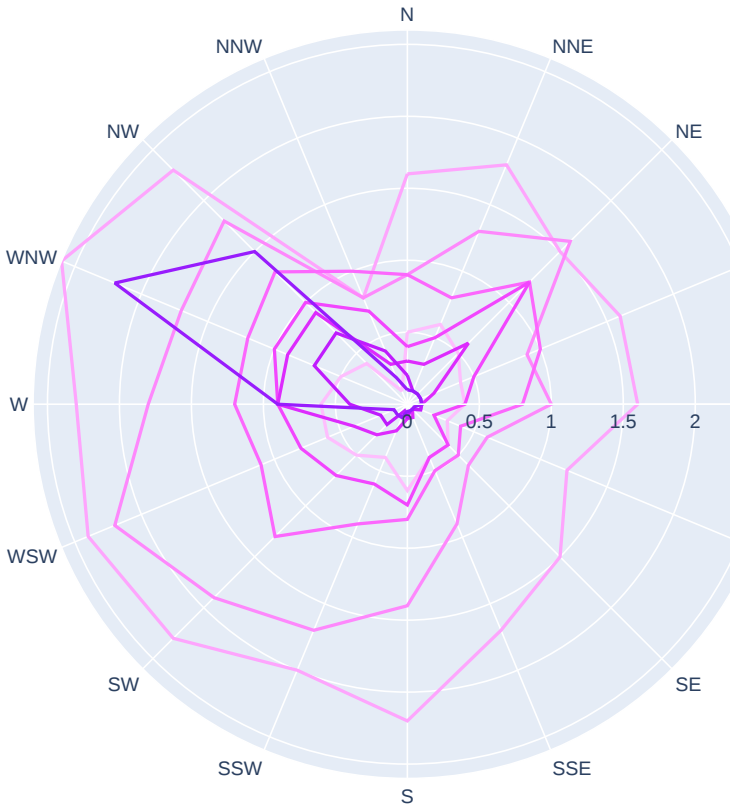
In [37]:
```
px.line_3d(election, x="Joly", y="Coderre", z="Bergeron", color="winner", line_dash="winner")
```



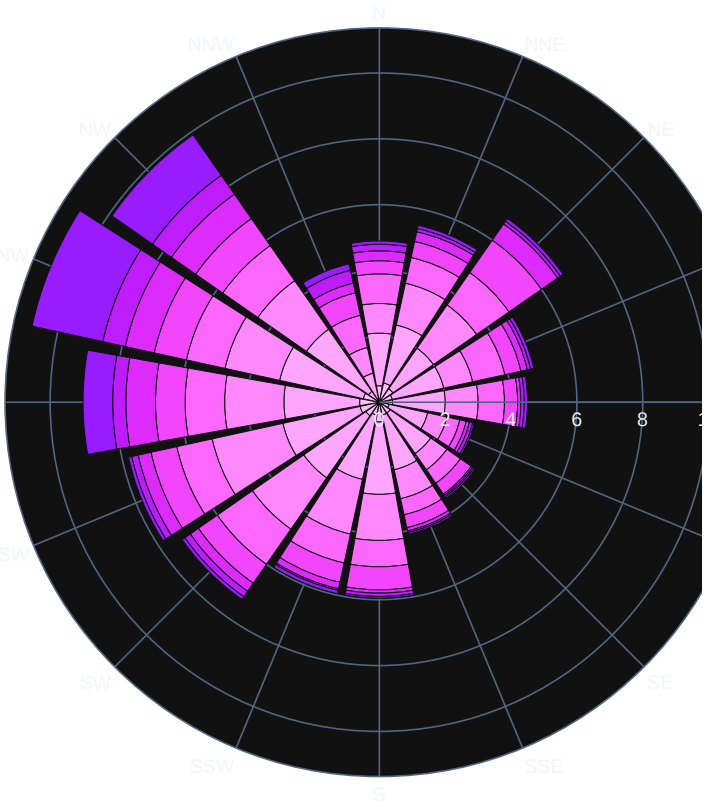
In [38]:
```
px.scatter_polar(wind, r="value", theta="direction", color="strength", symbol="strength",
                 color_discrete_sequence=px.colors.sequential.Plotly[-2::-1])
```



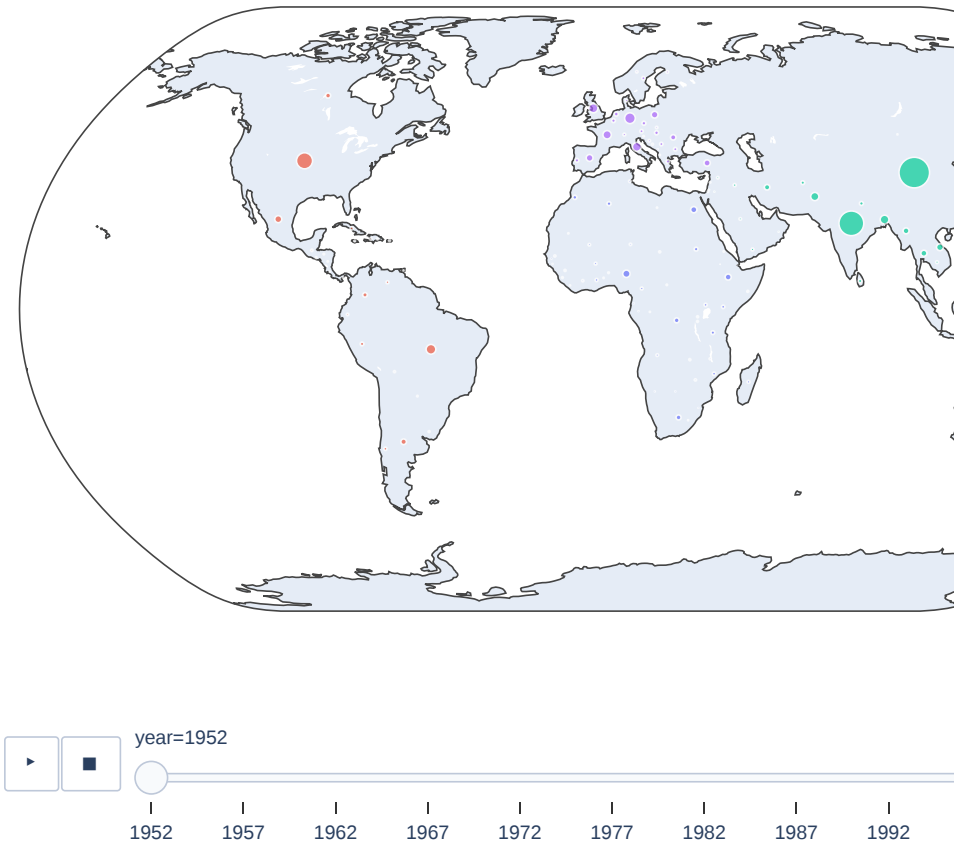
In [39]:
```
px.line_polar(wind, r="value", theta="direction", color="strength", line_close=True,
              color_discrete_sequence=px.colors.sequential.Plotly[-2::-1])
```



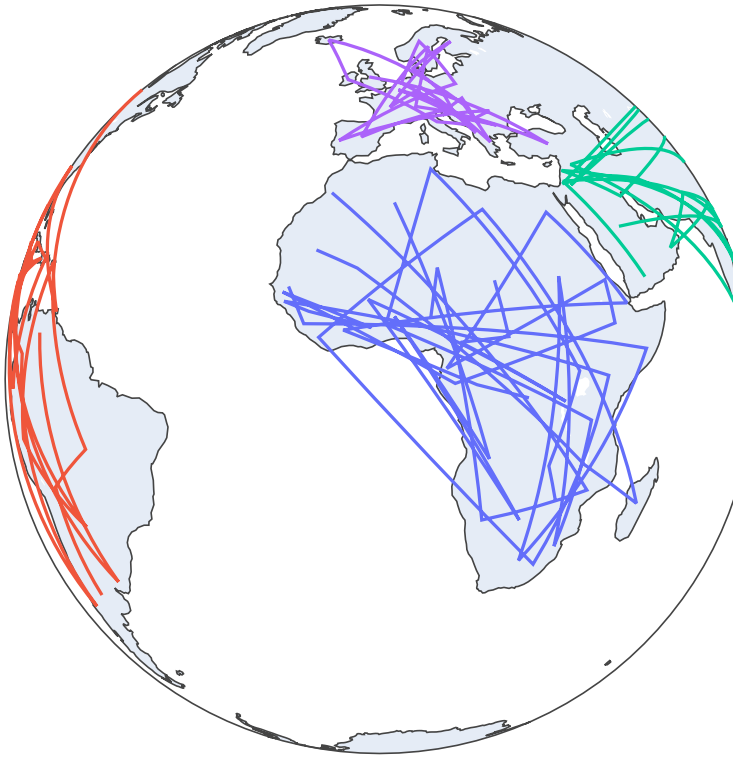
In [40]:
```
px.bar_polar(wind, r="value", theta="direction", color="strength", template="plotly_dark",
             color_discrete_sequence= px.colors.sequential.Plotly[-2::-1])
```



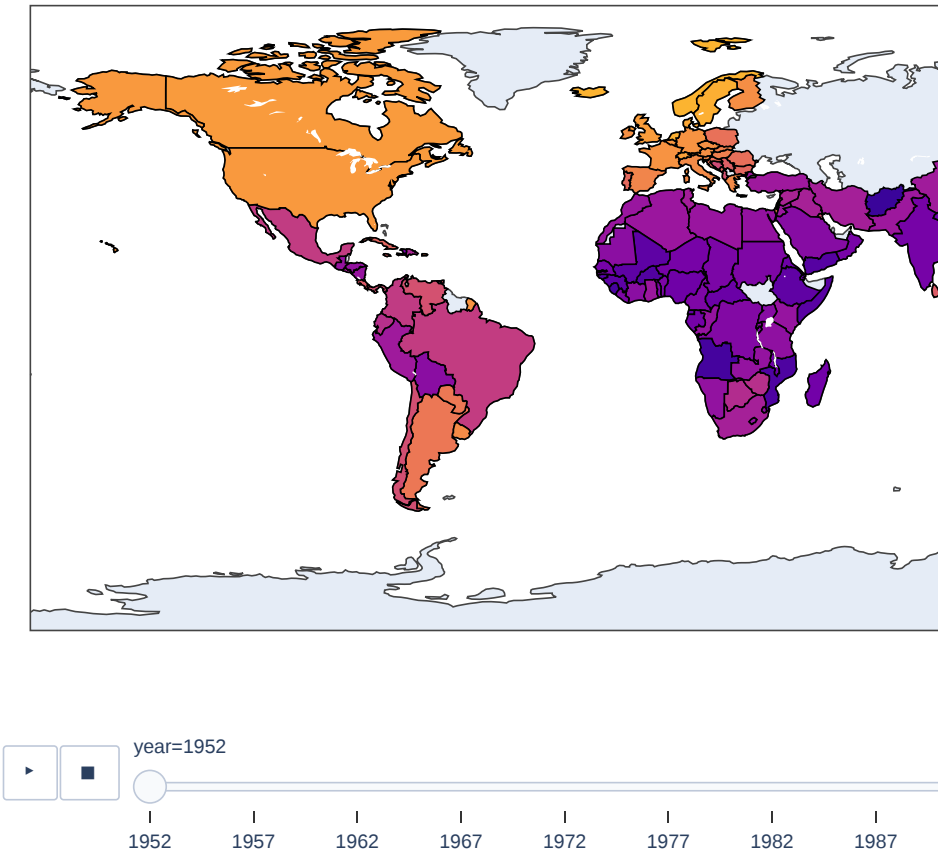
In [42]:
```
px.scatter_geo(gapminder, locations="iso_alpha", color="continent", hover_name="country", size="pop
",
               animation_frame="year", projection="natural earth")
```



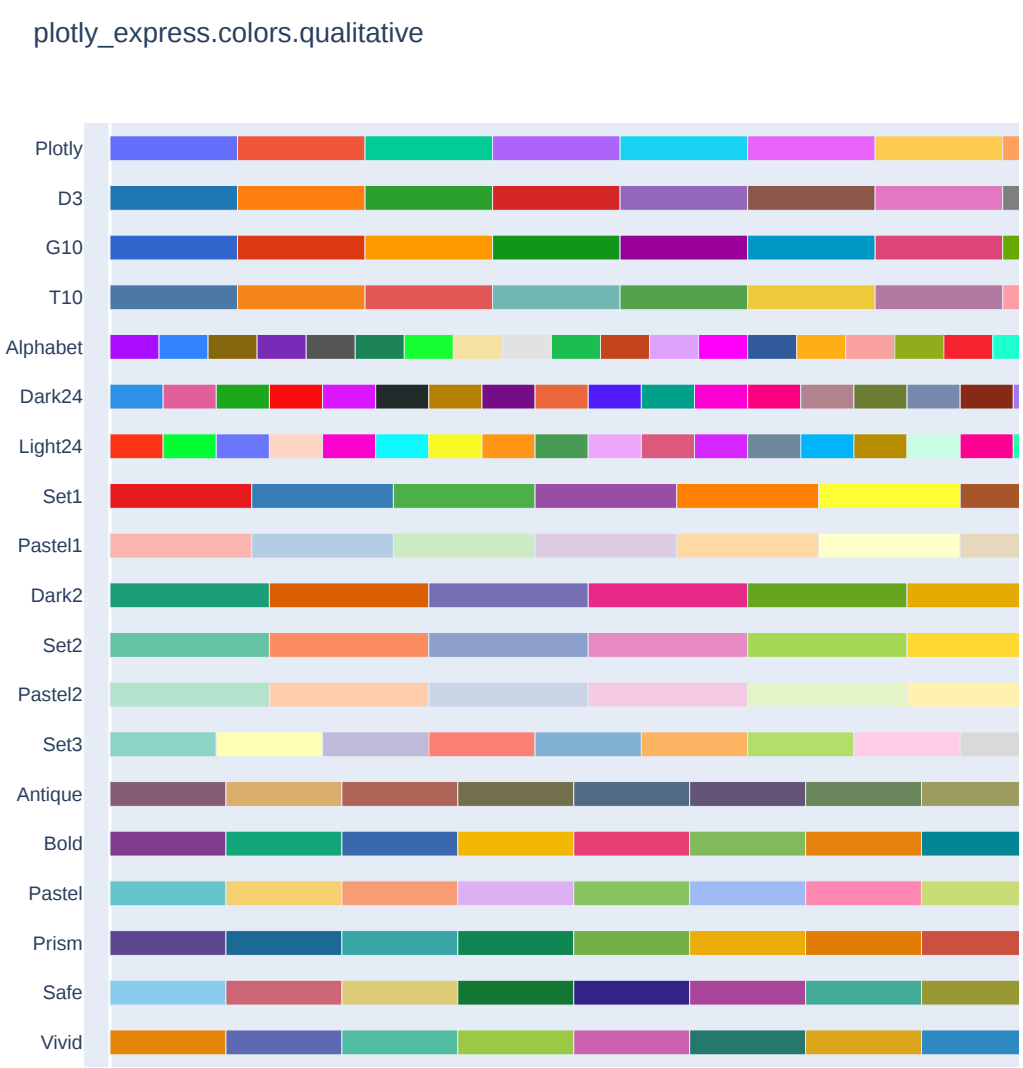
In [43]:
```
px.line_geo(gapminder.query("year==2007"), locations="iso_alpha", color="continent", projection="orthographic")
```



In [44]:
```
px.choropleth(gapminder, locations="iso_alpha", color="lifeExp", hover_name="country", animation_frame="year",
                    color_continuous_scale=px.colors.sequential.Plasma)
```

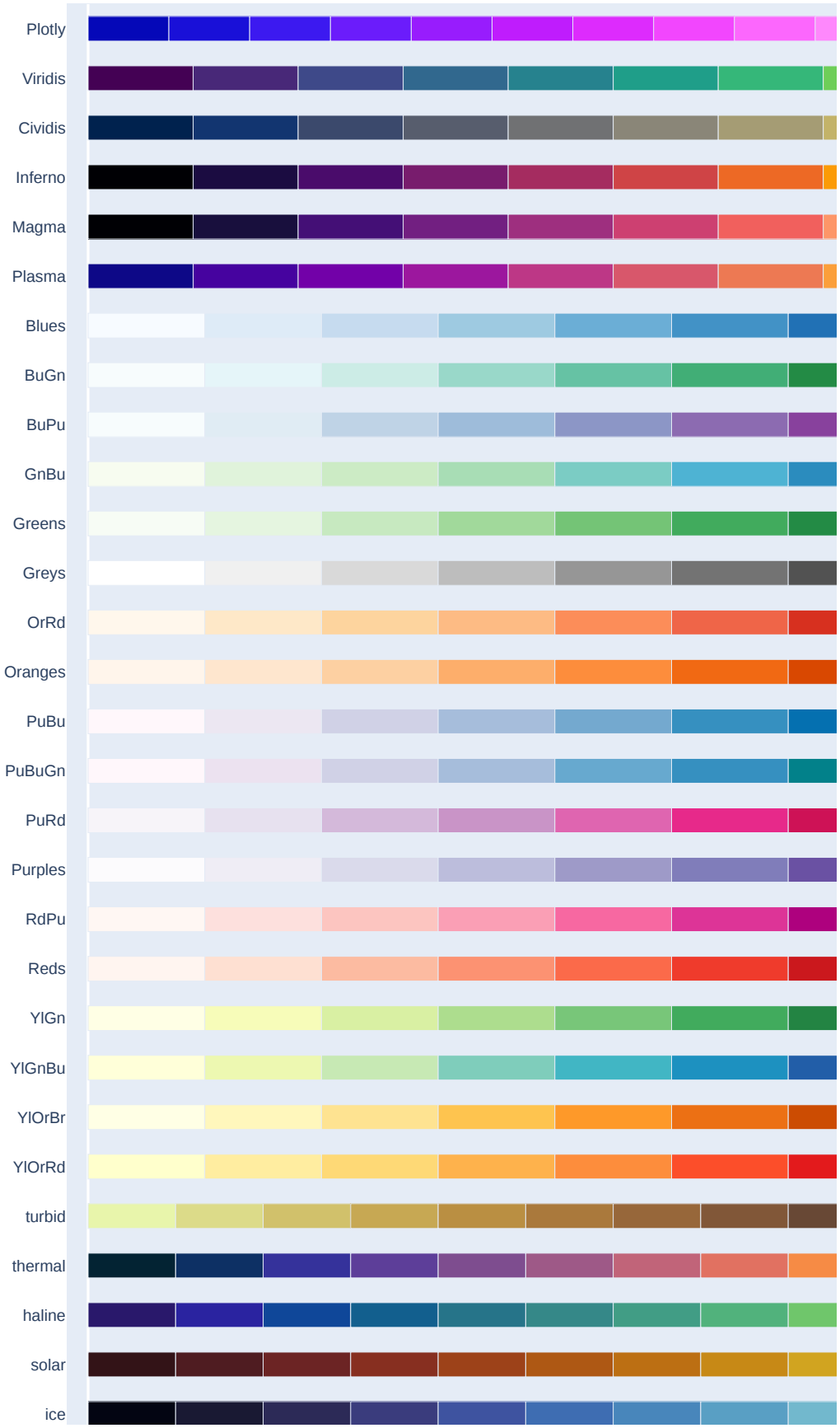


In [45]:
```
px.colors.qualitative.swatches()
```

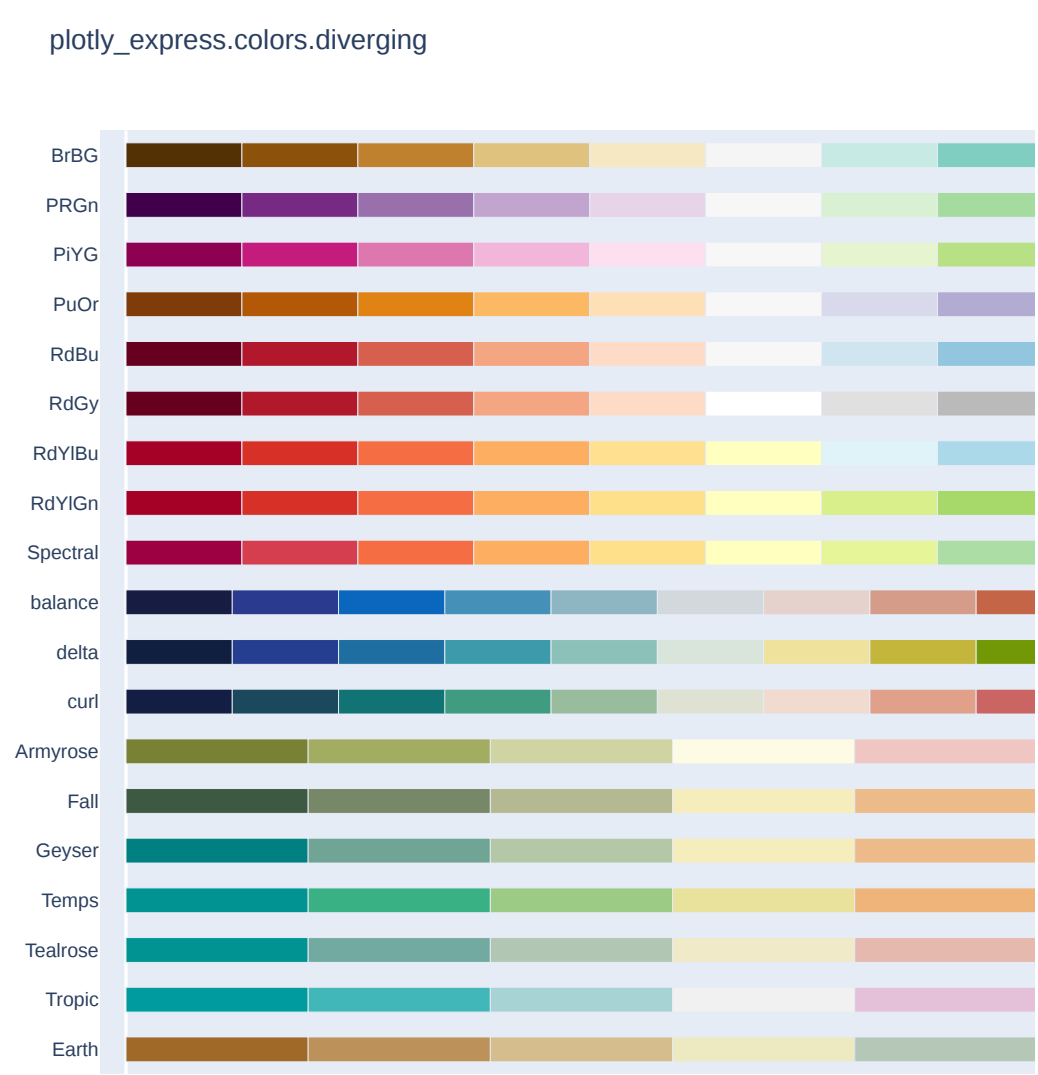


In [46]:
```
px.colors.sequential.swatches()
```

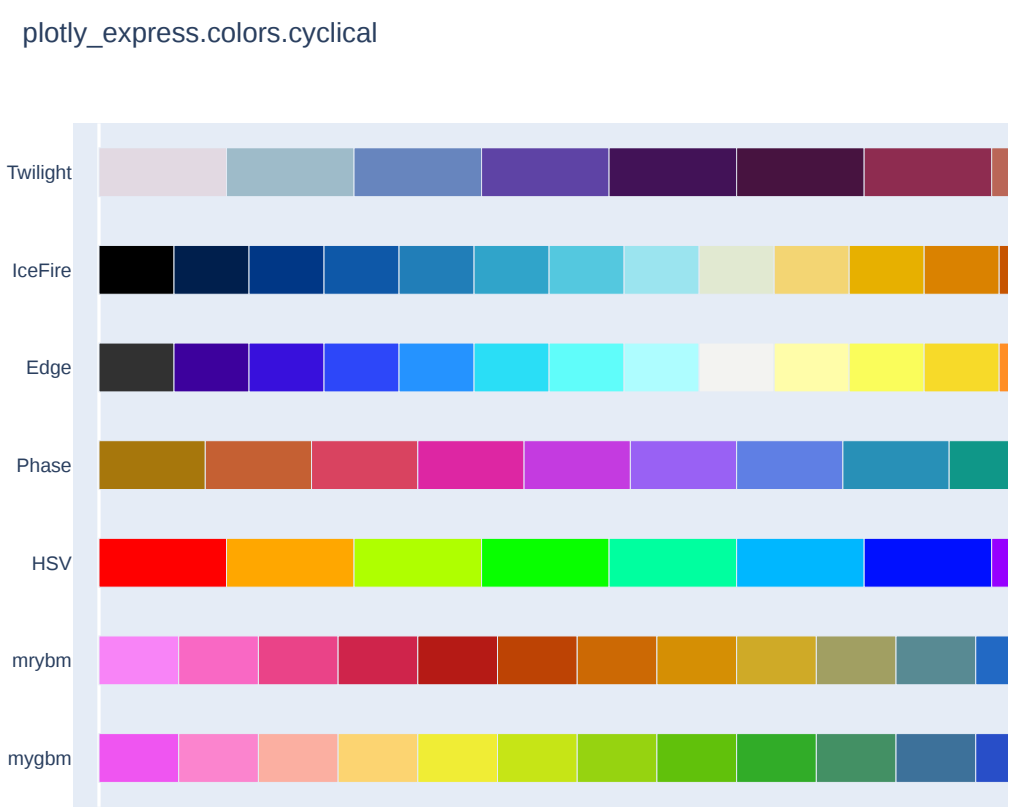
plotly_express.colors.sequential



In [47]:
```
px.colors.diverging.swatches()
```

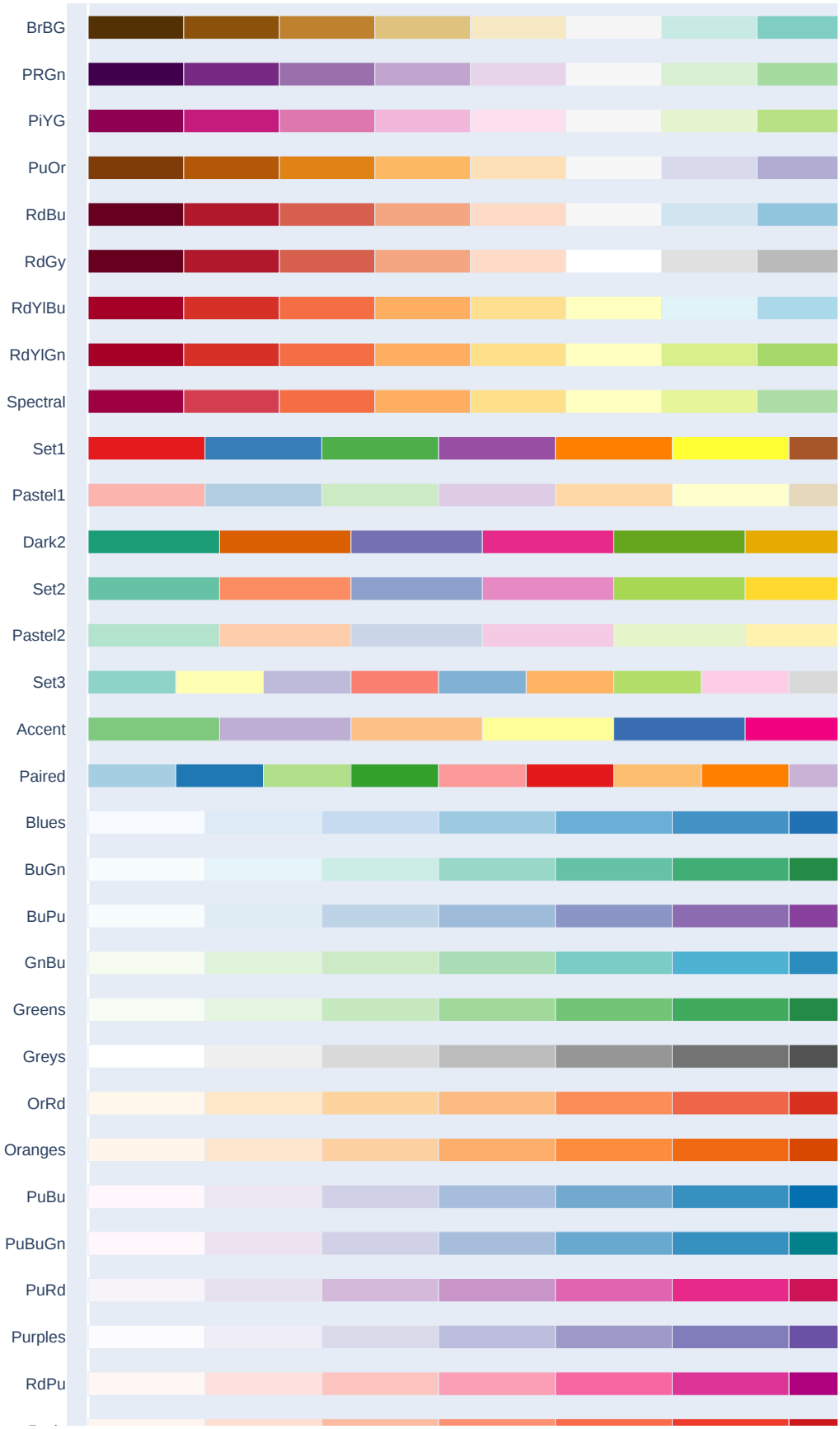


In [48]:
```
px.colors.cyclical.swatches()
```

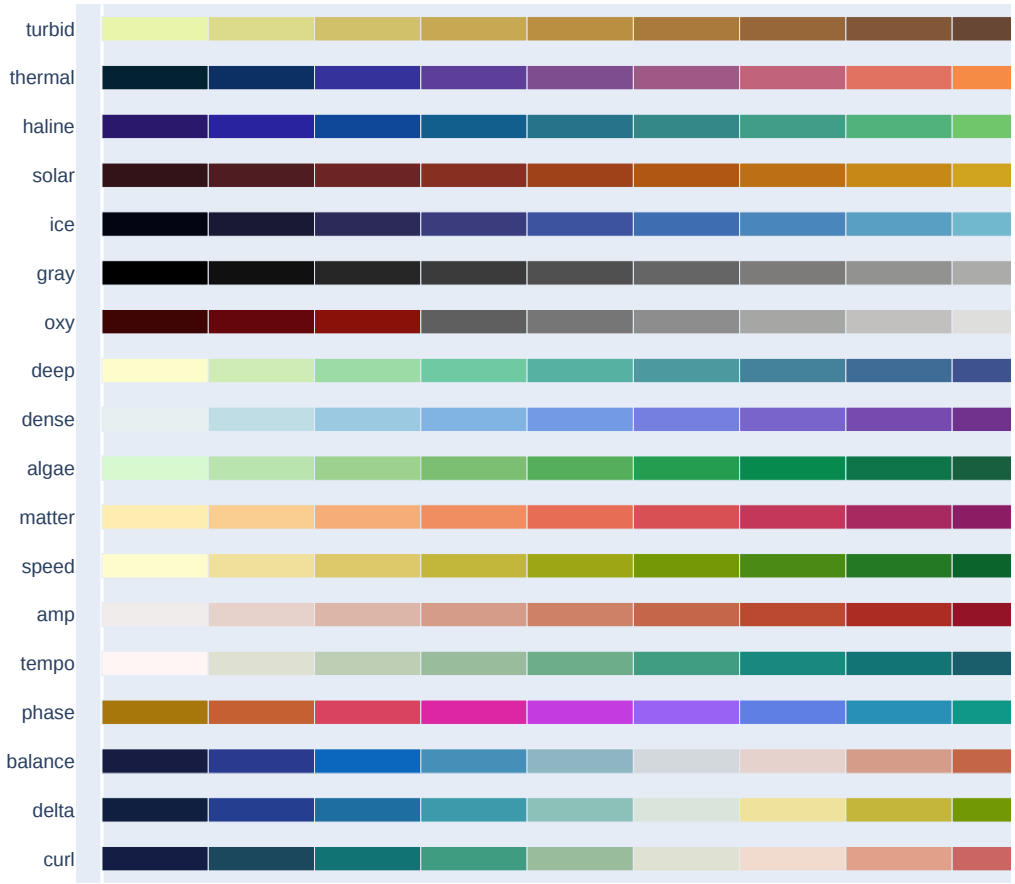


In [49]:
```
px.colors.colorbrewer.swatches()
```

plotly_express.colors.colorbrewer

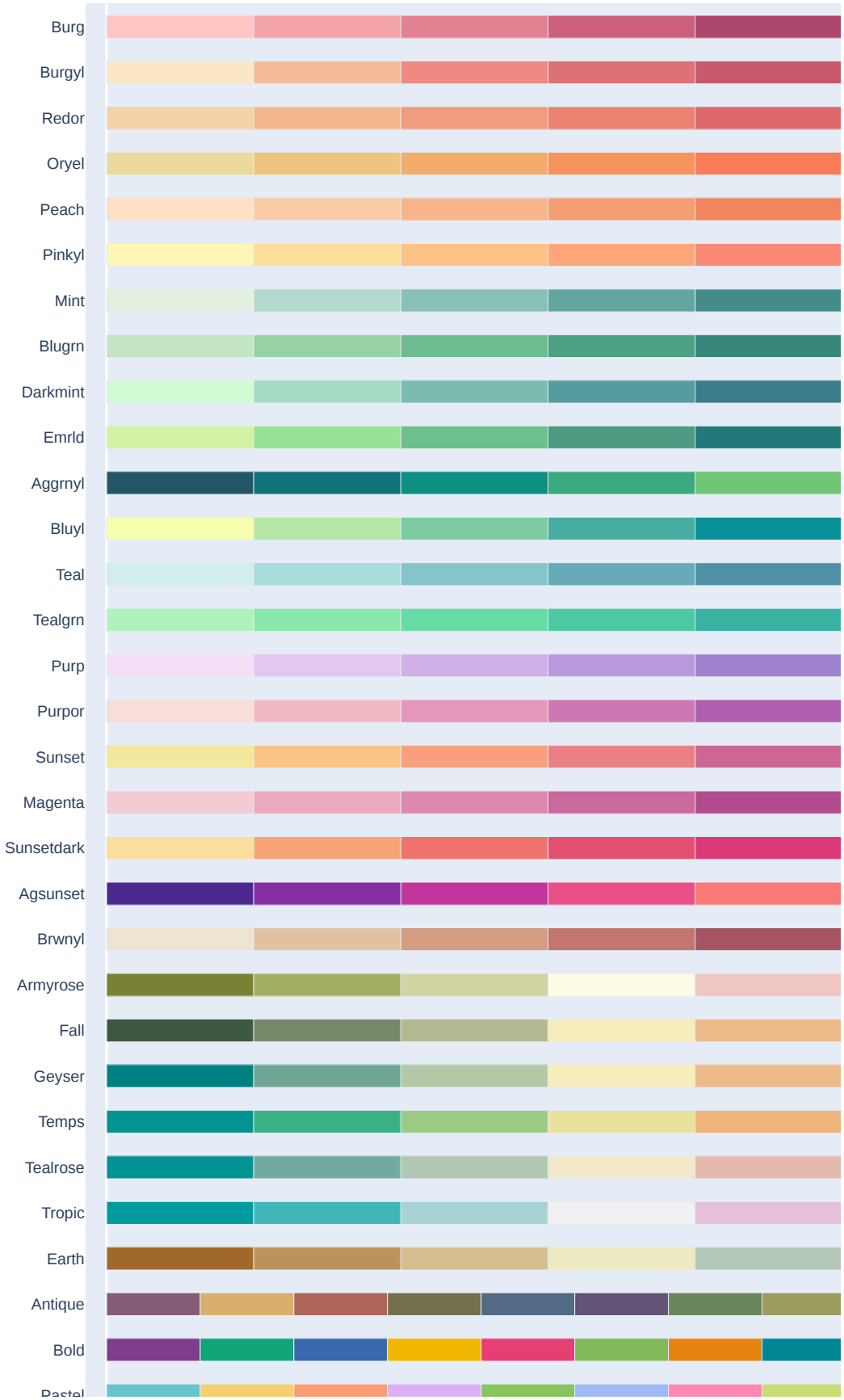


In [50]:
```
px.colors.cmocean.swatches()
```



In [51]:
```
px.colors.carto.swatches()
```

plotly_express.colors.carto



In [ ]: