# SI649 W23 Altair Homework #4

## Overview

We'll focus on maps and cartrographic visualization. In this lab, you will practice:

- Point Maps
- Symbol Maps
- Choropleth maps
- Interactions with maps

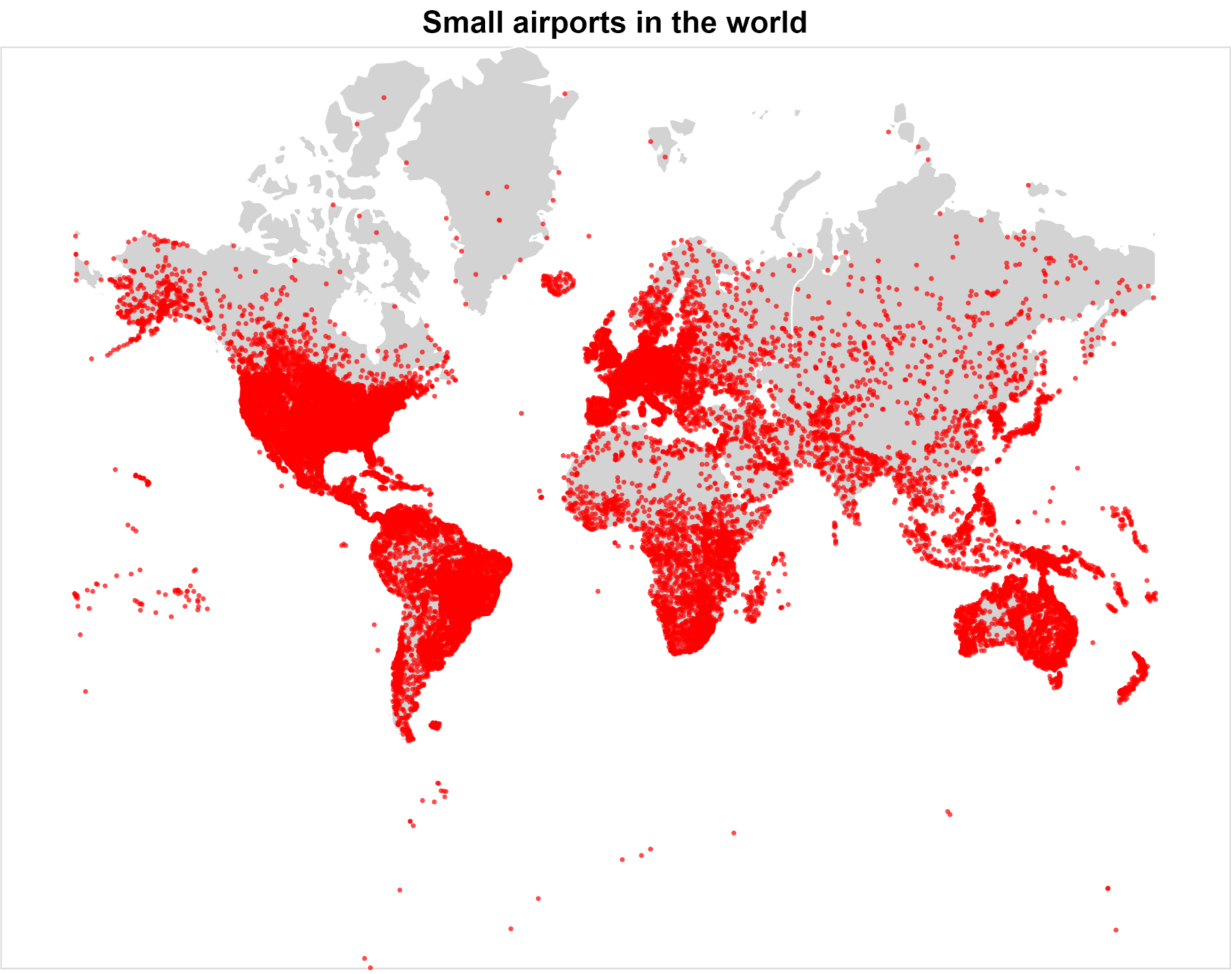
After building these charts, you will make a website with these charts using streamlit.

### Lab Instructions

- Save, rename, and submit the ipynb file (use your username in the name).
- Complete all the checkpoints, to create the required visualization at each cell.
- Run every cell (do Runtime -> Restart and run all to make sure you have a clean working version), print to pdf, submit the pdf file.
- If you end up stuck, show us your work by including links (URLs) that you have searched for. You'll get partial credit for showing your work in progress.

In [1]:
```
import pandas as pd
import altair as alt
from vega_datasets import data

alt.data_transformers.disable_max_rows()

df = pd.read_csv('https://raw.githubusercontent.com/pratik-mangtani/si649-hw/main/airports.csv')
url = "https://raw.githubusercontent.com/pratik-mangtani/si649-hw/main/small-airports.json"

```

## Visualization 1: Dot Density Map



**Description of the visualization:**

We want to visualize the density of small airports in the world. Each small airport is represented by a dot. The visualization has two layers:

- The point map shows different small airports.
- The tooltip shows the **name** of the airport.

**Hint:**

- How can we show continents on the map? Which object can be used from the json dataset ?
- How can we show only small airports on the map?

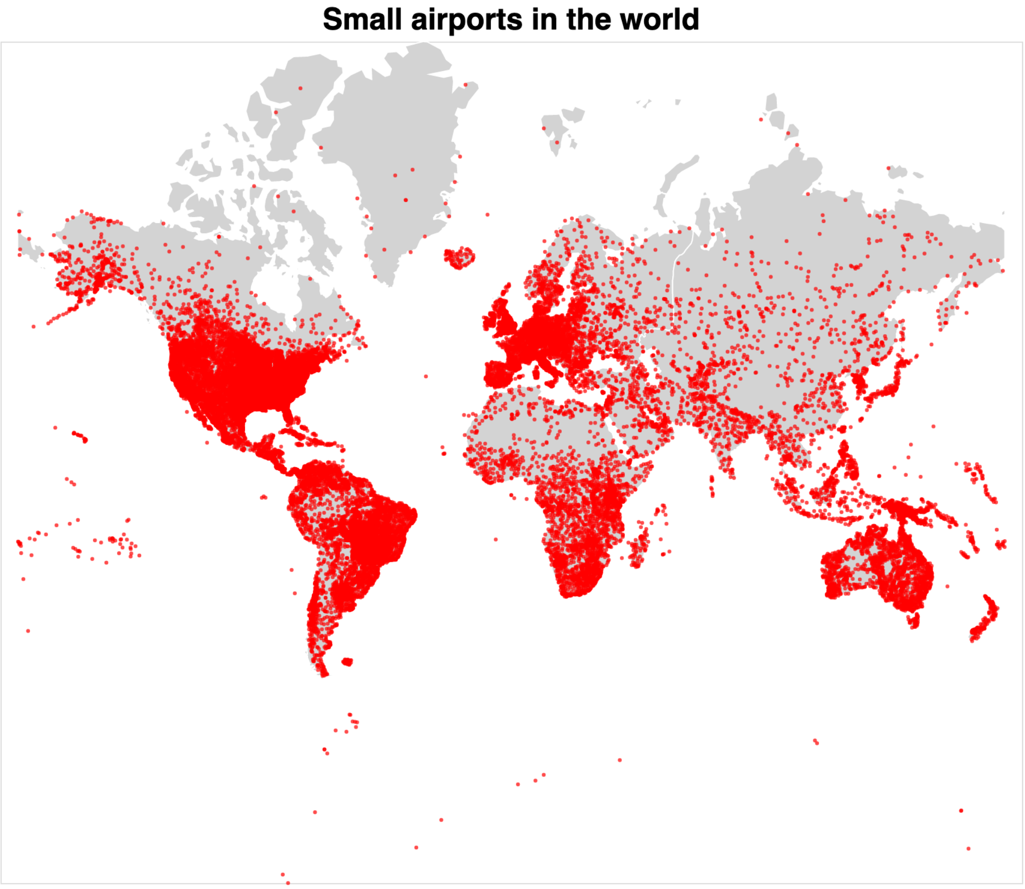
In [2]:

```
world = alt.topo_feature(url, feature = 'continent')
```

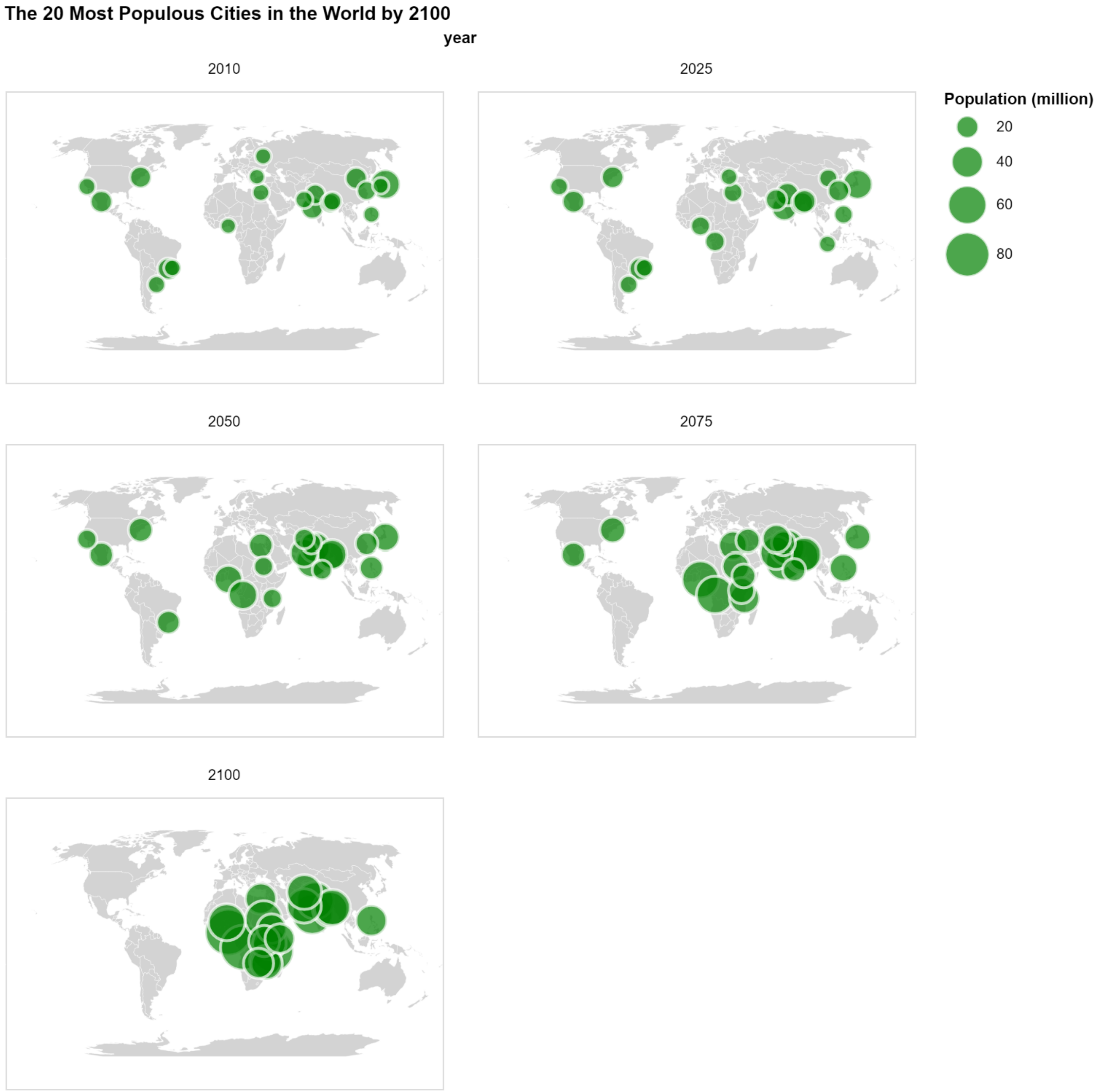
In [3]:

```
base = alt.Chart(world).mark_geoshape(
    fill = 'lightgray',
    stroke = 'white'
).project('mercator')

points = alt.Chart(df).mark_circle().transform_filter(
    alt.datum.type == 'small_airport'
).encode(
    latitude = alt.X('latitude_deg:Q'),
    longitude = alt.Y('longitude_deg:Q'),
    size = alt.value(10),
    color = alt.value('red'),
    tooltip = alt.Tooltip('name:N')
)

alt.layer(
    base, points
).properties(
    width = 850,
    height = 700,
    title = 'Small airports in the world'
).configure_title(
    fontSize = 25
)
```

Out[3]:



## Visualization 2: Propotional Symbol



**Description of the visualization:**

The visualization shows faceted maps pointing the 20 most populous cities in the world by 2100. There are two layers in faceted charts:

- The base layer shows the map of countries.
- The second layer shows size encoded points indicating the population of those countries.
- Tooltip shows **city** name and **population**.

**Hint:**

- Which projection has been used in individual charts?
- How to create a faceted chart with different years and 2 columns?

In [4]:

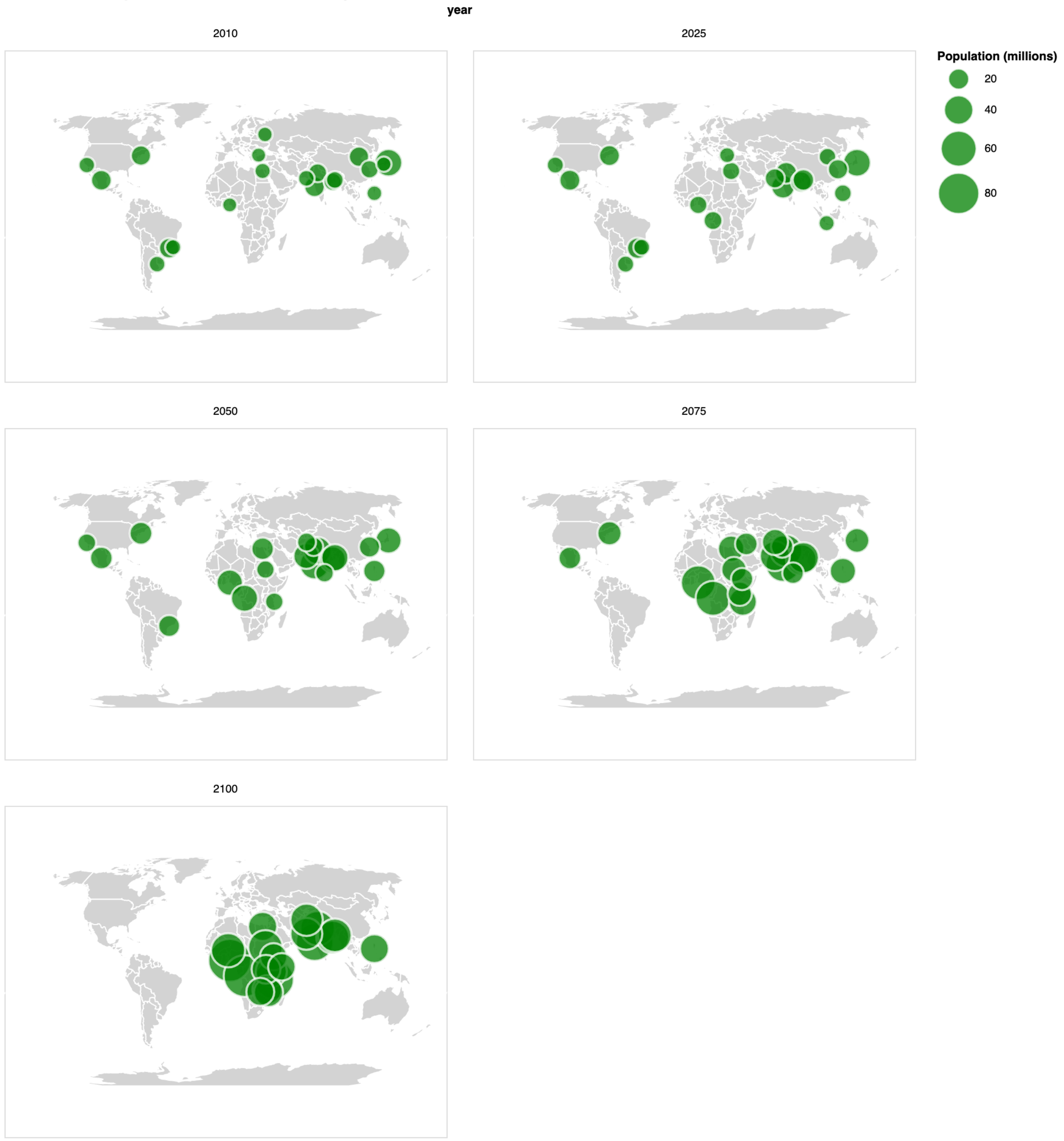
```
countries_url = data.world_110m.url
source = 'https://raw.githubusercontent.com/pratik-mangtani/si649-hw/main/population_prediction.csv'
```

In [5]:

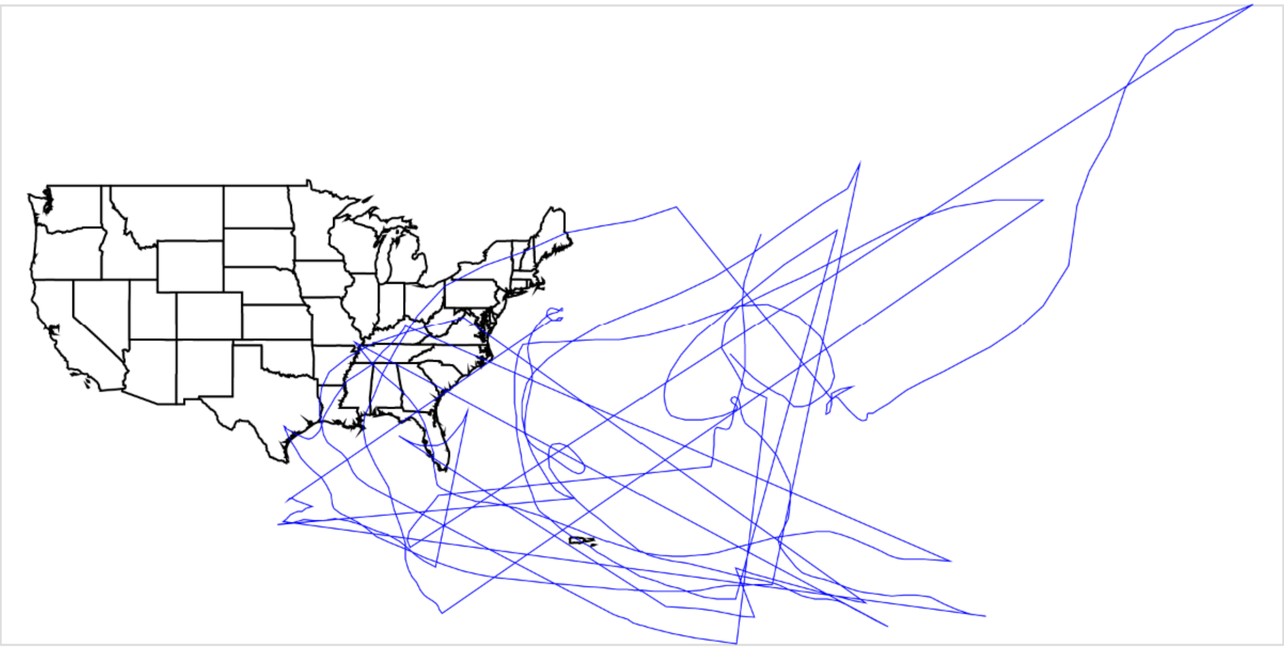
```
world = alt.topo_feature(countries_url, feature = 'countries')

world_map = alt.Chart(world).mark_geoshape(
    fill = 'lightgray',
    stroke = 'white'
).project(
    "naturalEarth1"
)

base = alt.Chart().mark_circle(
    fill = 'green',
    stroke = 'white'
).encode(
    latitude = 'lat:Q',
    longitude = 'lon:Q',
    size = alt.Size('population:Q', scale = alt.Scale(range = [0, 1500]),
                    legend = alt.Legend(title = 'Population (millions)')),
    tooltip = [alt.Tooltip('city:N', title = 'City'), alt.Tooltip('population:Q', title = 'Population')],
    opacity = alt.value(0.75)
)

alt.layer(world_map, base, data = source
).facet('year:N',
        columns = 2
).properties(
    title = 'The 20 Most Populous Cities in the World by 2100'
).configure_title(
    fontSize = 15
)
```

Out[5]:

**The 20 Most Populous Cities in the World by 2100**



### Visualization 3: Hurricane Trajectories



**Description of the visualization:**

Create a map that shows the paths (trajectories) of the 2017 hurricanes. Filter the data so that only 2017 hurricanes are shown. Remove Alaska and Hawaii from the map (Filter out ids 2 and 15).

**Hint:**

- How will you filter out 2017 hurricanes?
- Which object can be used to show state boundaries?

In [6]:

```
states_url = data.us_10m.url
hurricane_data = pd.read_csv('https://raw.githubusercontent.com/pratik-mangtani/si649-hw/main/hurdat2.csv')
hurricane_data.sample(3)
```

Out[6]:

| | identifier | name | num_pts | record_id | status | latitude | longitude | max_wind | min_pressure | datetime |
|---|---|---|---|---|---|---|---|---|---|---|
| **14452** | AL061923 | UNNAMED | 22 | NaN | TS | 24.2 | -92.4 | 60 | -999 | 1923-10-15T12:00:00 |
| **46031** | AL032009 | BILL | 46 | NaN | TS | 13.1 | -41.3 | 60 | 990 | 2009-08-17T00:00:00 |
| **45539** | AL042008 | DOLLY | 31 | NaN | TS | 24.3 | -94.9 | 60 | 990 | 2008-07-22T18:00:00 |

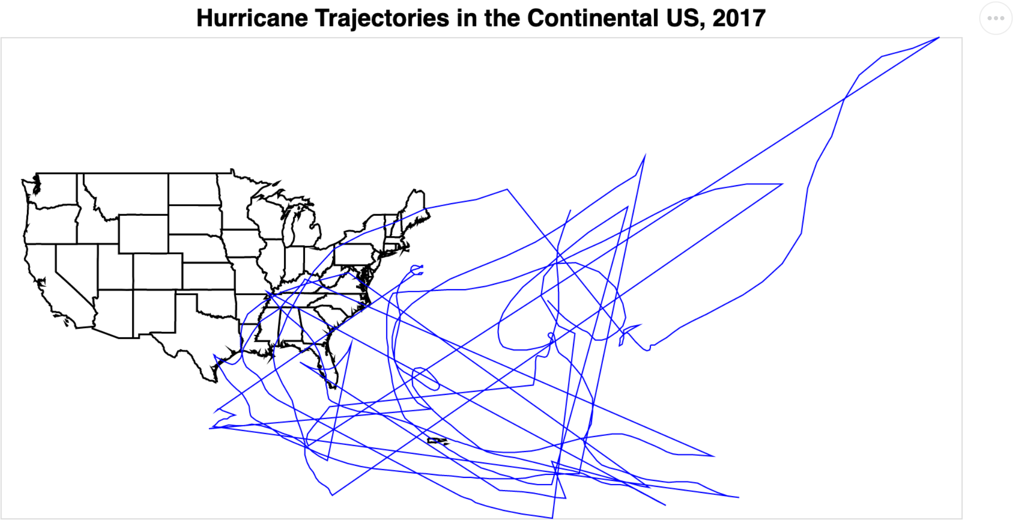
In [7]:

```
hurricane_data['year'] = pd.DatetimeIndex(hurricane_data['datetime']).year
continent_hurricanes = hurricane_data[~hurricane_data.index.isin([2, 15])]
```

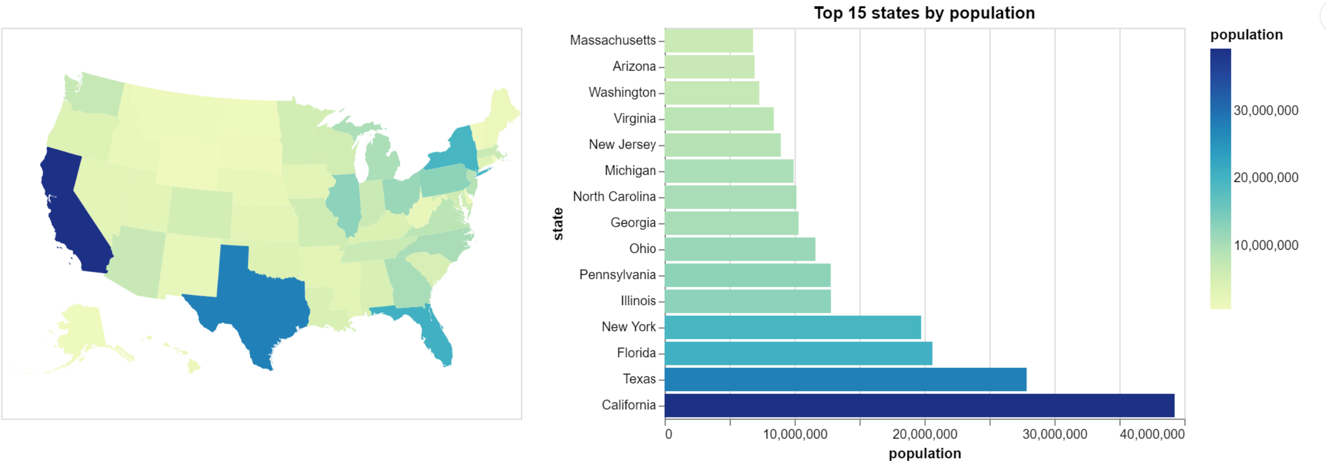
In [8]:

```python
#TODO: Vis 3
states = alt.topo_feature(states_url, feature = 'states')

us = alt.Chart(states).transform_filter(
    (alt.datum.id != 2) & (alt.datum.id != 15)
).mark_geoshape(
    fill = 'white',
    stroke = 'black',
    strokeWidth = 1.5
).project('mercator')

lines = alt.Chart(continent_hurricanes).mark_line(
    stroke = 'blue',
    strokeWidth = 1
).transform_filter(
    (alt.datum.year == 2017)
).encode(
    latitude = 'latitude:Q',
    longitude = 'longitude:Q'
)

alt.layer(
    us, lines
).properties(
    width = 800,
    height = 400,
    title = 'Hurricane Trajectories in the Continental US, 2017'
).configure_title(
    fontSize = 20
)
```

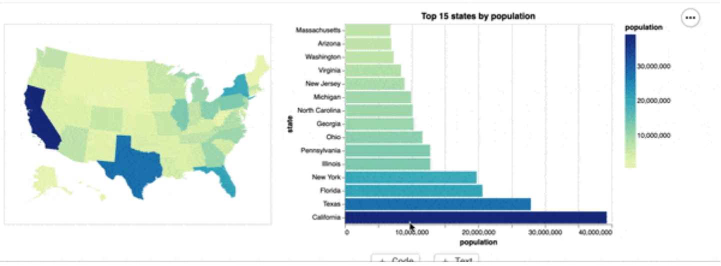
Out[8]:



## Visualization 4: Choropleth Map



**Interaction**



**Description of the visualization:**

The visualization has a choropleth map showing the population of different states and a sorted bar chart showing the top 15 states by population. These charts are connected

In [9]:

```python
state_map = data.us_10m.url
state_pop = data.population_engineers_hurricanes()[['state', 'id', 'population']]
state_pop.sample(5)
```

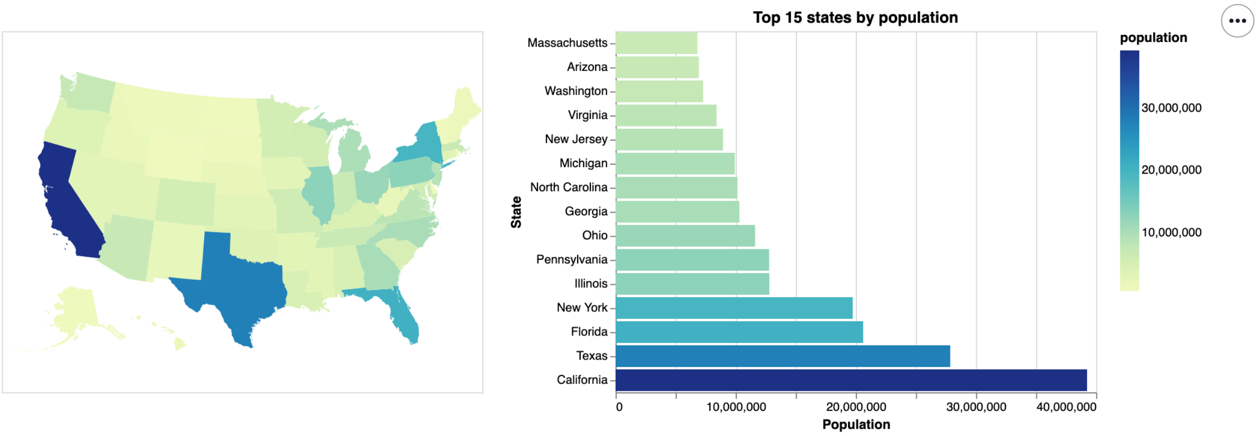
Out[9]:

|  | state | id | population |
|---|---|---|---|
| **48** | West Virginia | 54 | 1831102 |
| **43** | Texas | 48 | 27862596 |
| **27** | Nebraska | 31 | 1907116 |
| **0** | Alabama | 1 | 4863300 |
| **16** | Kansas | 20 | 2907289 |

In [10]:

```python
states = alt.topo_feature(state_map, 'states')

hover_select = alt.selection_single(on = 'click', fields = ['state'])
opacity_condition = alt.condition(hover_select, alt.value(1.0), alt.value(0.25))


states_view = alt.Chart(states).add_selection(hover_select
).mark_geoshape().project(
    'albersUsa'
).transform_lookup(
    lookup = 'id', from_ = alt.LookupData(data = state_pop, key='id', fields=['state', 'population'])
).encode(
    color = alt.Color('population:Q'),
    opacity = opacity_condition
)

states_bar = alt.Chart(state_pop).add_selection(hover_select).mark_bar(
).transform_window(
    rank = 'row_number()',
    sort = [alt.SortField('population', order = 'descending')]
).transform_filter(
    (alt.datum.rank <= 15)
).encode(
    x = alt.X('population:Q', axis = alt.Axis(title = 'Population')),
    y = alt.Y('state:N', sort = alt.EncodingSortField(field = 'population'), axis = alt.Axis(title = 'State')),
    color = alt.Color('population:Q'),
    opacity = opacity_condition
).properties(
    title = 'Top 15 states by population'
)


alt.hconcat(states_view, states_bar)
```

Out[10]:



In [ ]:

```python

```