Lecture 22: Animated and Interactive Visuals
Dynamic and Interactive Visualization

- Last week we learned about static graphs
- Dynamic/Interactive graphs change based on time or user input
- Often used to demonstrate a relationship in the data over time
- Examples
  - Business Insider, Spread of 5 Major Religions Map
    https://www.youtube.com/watch?v=AvFl6UBZLw4
  - NYTimes, Renting vs buying calculator
Python and Dynamic Visualization

- Plotly can do this for us!
- Upgrade plotly:
  
  ```
  pip install plotly --upgrade
  ```

- Dynamic visualizations add a new field to the figure: **Frames**!
Frames

- Frames represent the data at each step in the animation
- Frames are a list of data points, so we need a list of lists for x-values and y-values.

\[
x = [[1, -1], [1, -1], [-1, 1], [-1, 1]]
\]

\[
y = [[1, -1], [-1, 1], [-1, 1], [1, -1]]
\]
Creating data points

We will use plotly’s Scatter class to plot the data points.

go.Scatter(
  x=x_value,
  y=y_value,
  mode = 'markers',
  marker = {'color': 'Blue', 'size':10})

X_value and y_value are two lists. Each value in x_value has a corresponding value in y_value. These two points are the x and y coordinates for each point.

If we pass in:

x_value=[2,-1]
y_value=[3,-2]

The two points we plot will be: (2,3) and (-1,-2).
Creating data points

We will use plotly's Scatter class to plot the data points.

go.Scatter(
    x=x_value,
    y=y_value,
    mode = 'markers',
    marker = {'color': 'Blue', 'size':10})

Here we can pass in "lines", "markers", "text", or a combination of these.

This determines how each point is represented on the graph.

“Marker” will be a point.
Creating data points

We will use plotly’s Scatter class to plot the data points.

go.Scatter(
    x=x_value,
    y=y_value,
    mode = ‘markers’,
    marker = {‘color’ : ‘Blue’, ‘size’:10})

Here we can edit the marker’s color and size.
Creating data points

For each frame we want to create a new scatterplot.

def create_data(x_values, y_values):
    data = [go.Scatter(x = x_values, y = y_values, mode = 'markers', marker = {'color' : 'Blue', 'size':10})]
    return data
Creating frames

- We want to create a list of dictionaries called frames to pass into the plotly Figure object.

```python
frames = [
    {'data': create_data(x[0], y[0])},
    {'data': create_data(x[1], y[1])},
    {'data': create_data(x[2], y[2])},
    {'data': create_data(x[3], y[3])}
]
```

Where `create_data` is a function that returns a `go.Scatter` object.
Creating layout

- Our layout for dynamic visualizations are more complicated because we want users to be able to interact with them.
- In this example, we are creating a button to dynamically show all the frames.

```python
go.Layout(updatemenus=
    [ {'buttons': [{'label': 'Animate',
                   'method': 'animate',
                   'args': [None]},
                  'showactive': False,
                  'type': 'buttons'}
    ])
```
Putting it all together

- Finally, we want to create the figure by passing in all the data, frames, and layout

```python
figure = go.Figure(data=data,
                   layout=layout,
                   frames=frames)
```
Lab Time