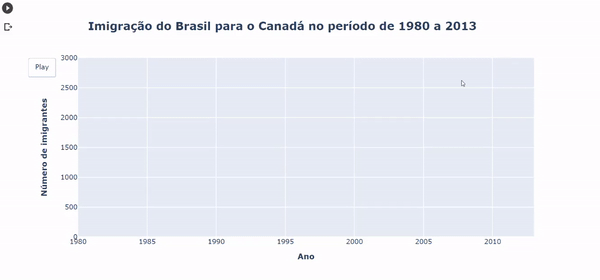
Nós aprendemos como criar gráficos interativos com a biblioteca Plotly, porém você sabia que é possível ir além e criar animações com essa mesma biblioteca? Além de tornar nossos gráficos mais interativos, as animações podem trazer ainda mais dinamismo e criatividade para as nossas visualizações.

Para ilustrar essa ideia, dê uma olhada na captura de tela de um notebook onde criei uma animação com os dados de imigração do Brasil para o Canadá. O código que gera essa animação começa exibindo a figura sem nenhuma linha. Ao clicar no botão "Play", no canto superior esquerdo da figura, a linha começa a surgir no primeiro tick do eixo X, que corresponde ao ano 1980, e vai se movendo até chegar ao final do eixo X, mostrando a evolução dos dados ao longo do tempo:



Com isso, podemos notar que a animação é uma ótima maneira de visualizar dados em evolução ao longo do tempo.

Você pode estar se perguntando: como podemos criar uma animação como essa? Fique tranquilo(a), estou aqui para te mostrar o caminho. Vou te explicar passo a passo!

1 - A primeira etapa após obter o DataFrame apenas com os dados do Brasil, como fizemos no começo do curso, é mudar o tipo de dados da coluna que contém os anos para int ao invés de manter como strings:

dados\_brasil['ano'] = dados\_brasil['ano'].astype(int)

2 - Depois disso, vamos criar um bloco de código onde vamos construir essa animação, importando plotly.graph\_objs, um módulo da biblioteca Plotly que contém classes para criar visualizações de dados interativas e personalizadas.

import plotly.graph\_objs as go

3 - Em seguida, uma figura vazia é criada usando a função go.Figure() e atribuída à variável fig.

fig = go.Figure()

4 - A seguir, uma linha é adicionada ao gráfico usando a função fig.add\_trace(). Nesta função, é passado um objeto *Scatter*, que recebe como argumentos os dados para os eixos X e Y do gráfico. Para que o gráfico seja exibido sem linha antes de clicar no botão play usamos o iloc[0] nas duas variáveis. Isso ocorre porque o iloc[0] seleciona o primeiro valor das colunas ano e imigrantes dos dados do Brasil, respectivamente. Ao adicionar esse ponto de dados à visualização do gráfico, ele será exibido inicialmente como um único ponto, sem linhas que o conectem a outros pontos.Além disso, passamos o modo de exibição lines, que quer dizer linhas e o nome da linha. Também é definida a espessura da linha usando o dicionário line=dict(width=4).

fig.add\_trace(

go.Scatter(x=[dados\_brasil['ano'].iloc[0]], y=[dados\_brasil['imigrantes'].iloc[0]], mode='lines', name='Imigrantes', line=dict(width=4))

)

5 - Depois disso, o título do gráfico e as configurações do eixo X e Y são definidas usando a função fig.update\_layout(). Os argumentos do título são:

* text='<b>Imigração do Brasil para o Canadá no período de 1980 a 2013</b>': define o texto do título como uma string formatada em negrito (usando as tags HTML "<b>" e "</b>")
* x=0.12: define a posição horizontal do título no layout, em relação à largura da figura. O valor 0.12 especifica que o título começará a 12% da largura da figura.
* xanchor='left': define o alinhamento horizontal do título. O valor 'left' significa que o título será alinhado à esquerda do layout.
* font=dict(size=20): define o tamanho do texto do título.

Já Os argumentos para xaxis e yaxis são dicionários, com as seguintes propriedades:

* range=[1980, 2013]: define o intervalo do eixo, ou seja, o menor e o maior valor que serão exibidos. Neste caso, o eixo x terá como menor valor 1980 e como maior valor 2013, enquanto o eixo y terá como menor valor 0 e como maior valor 3000.
* autorange=False: define se os limites do eixo serão ajustados automaticamente (True) ou não (False). Neste caso, os limites não serão ajustados automaticamente.
* title='<b>Ano</b>': define o título do eixo. Neste caso, o eixo x terá o título "Ano", que é formatado em negrito (usando as tags HTML "" e "").
* title='<b>Número de imigrantes</b>': define o título do eixo. Neste caso, o eixo y terá o título "Número de imigrantes", que é formatado em negrito (usando as tags HTML "<b>" e "</b>").

fig.update\_layout(

title=dict(

text='<b>Imigração do Brasil para o Canadá no período de 1980 a 2013</b>',

x=0.12,

xanchor='left',

font=dict(size=20)

),

xaxis=dict(range=[1980, 2013], autorange=False, title='<b>Ano</b>'),

yaxis=dict(range=[0, 3000], autorange=False, title='<b>Número de imigrantes</b>'),

6 - É adicionado um botão "Play" para a animação usando o argumento updatemenus. Esse argumento é uma lista que define as opções de menu para a figura. O valor atribuído a essa lista é um dicionário, com as seguintes propriedades:

* type='buttons': define que o menu será composto por botões.
* showactive=False: define que nenhum botão estará ativo inicialmente.
* buttons=[dict(label='Play', method='animate', args=[None, {'frame': {'duration': 100, 'redraw': True}, 'fromcurrent': True}])]: define o botão que será exibido no menu. Este botão tem a etiqueta "Play" (ou seja, "tocar"), que é exibida no próprio botão. O método animate é usado para ativar a animação dos dados. O argumento args é uma lista que contém dois elementos: o primeiro elemento é None, indicando que nenhum trace (ou camada) do gráfico será afetado pela animação, e o segundo elemento é um dicionário que especifica os parâmetros da animação. O parâmetro frame define a duração de cada quadro da animação e a atualização de cada quadro. O parâmetro fromcurrent define se o quadro atual deve ser mantido ou se a animação deve ser iniciada do primeiro quadro.

updatemenus=[dict(

type='buttons',

showactive=False,

buttons=[dict(

label='Play',

method='animate',

args=[None, {'frame': {'duration': 100, 'redraw': True}, 'fromcurrent': True}]

)]

)],

7- A largura e a altura do gráfico são definidas com os parâmetros width e height, respectivamente:

width=1000,

height=500

8 - Em seguida, as configurações de animação são definidas. A variável frames é uma lista de objetos Frame do Plotly, que contém as informações dos dados para cada quadro da animação. Cada quadro é representado por um objeto Frame que contém um único *trace*, que é um objeto *Scatter*. No caso deste código, cada *Scatter* representa um ponto no gráfico, onde X é o ano e Y é o número de imigrantes.

O loop for é usado para criar um objeto Frame para cada ano no conjunto de dados, até o último ano. O método iloc é usado para selecionar os valores do DataFrame dados\_brasil a partir do índice 0 até o índice atual i+1, o que significa que cada quadro da animação adiciona um ponto adicional ao gráfico. A cada iteração, a lista frames é preenchida com um novo objeto Frame.

Por último, a lista frames é atribuída à propriedade frames do objeto Figure (fig) criado anteriormente. Isso permite que a animação seja exibida no gráfico quando o botão de controle de animação é pressionado. Cada frame contém os dados do gráfico para um determinado ano e é exibido em sequência quando o botão "Play" é clicado.

frames = [go.Frame(data=[go.Scatter(x=dados\_brasil['ano'].iloc[:i+1], y=dados\_brasil['imigrantes'].iloc[:i+1])]) for i in range(len(dados\_brasil))]

fig.frames = frames

9 - Por fim, a função fig.show() é chamada para exibir o gráfico animado no notebook. Quando o botão "Play" é pressionado, o gráfico será animado, mostrando a imigração do Brasil para o Canadá no período de 1980 a 2013.

Temos o código completo abaixo:

import plotly.graph\_objs as go

# Criando uma figura

fig = go.Figure()

# Adicionando a linha do gráfico e definindo a espessura da linha

fig.add\_trace(

go.Scatter(x=[dados\_brasil['ano'].iloc[0]], y=[dados\_brasil['imigrantes'].iloc[0]], mode='lines', name='Imigrantes', line=dict(width=4))

)

# Definindo as configurações de layout

fig.update\_layout(

title=dict(

text='<b>Imigração do Brasil para o Canadá no período de 1980 a 2013</b>',

x=0.12,

xanchor='left',

font=dict(size=20)

),

xaxis=dict(range=[1980, 2013], autorange=False, title='<b>Ano</b>'),

yaxis=dict(range=[0, 3000], autorange=False, title='<b>Número de imigrantes</b>'),

updatemenus=[dict(

type='buttons',

showactive=False,

buttons=[dict(

label='Play',

method='animate',

args=[None, {'frame': {'duration': 100, 'redraw': True}, 'fromcurrent': True}]

)]

)],

width=1000,

height=500

)

# Definir as configurações de animação

frames = [go.Frame(data=[go.Scatter(x=dados\_brasil['ano'].iloc[:i+1], y=dados\_brasil['imigrantes'].iloc[:i+1])]) for i in range(len(dados\_brasil))]

fig.frames = frames

# Mostrando a figura

fig.show()

Além disso, é importante destacar que as possibilidades de criação de animações são muito grandes. Podemos animar não só linhas, mas também pontos, barras, mapas e muito mais. A biblioteca Plotly oferece uma série de ferramentas e recursos para nos ajudar a criar visualizações incríveis e com alto grau de interatividade.