

Project Deliverables

[#A] Data Imports

```
In [2]: !pip install wbdata
!pip install cufflinks
!pip install iso3166

import iso3166 #iso3166.countries.get('country details')
import wbdata
import cufflinks as cf
import pandas as pd
import numpy as np
import plotly
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.offline as py
import plotly.graph_objs as go
cf.go_offline()
```

Collecting wldata

Using cached wldata-0.3.0-py3-none-any.whl (14 kB)
Requirement already satisfied: decorator>=4.0 in /opt/conda/lib/python3.9/site-packages (from wldata) (5.0.9)
Requirement already satisfied: requests>=2.0 in /opt/conda/lib/python3.9/site-packages (from wldata) (2.26.0)
Requirement already satisfied: appdirs<2.0,>=1.4 in /opt/conda/lib/python3.9/site-packages (from wldata) (1.4.4)
Requirement already satisfied: tabulate>=0.8.5 in /opt/conda/lib/python3.9/site-packages (from wldata) (0.9.0)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/lib/python3.9/site-packages (from requests>=2.0->wldata) (3.1)
Requirement already satisfied: certifi>=2017.4.17 in /opt/conda/lib/python3.9/site-packages (from requests>=2.0->wldata) (2021.10.8)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /opt/conda/lib/python3.9/site-packages (from requests>=2.0->wldata) (1.26.7)
Requirement already satisfied: charset-normalizer~2.0.0 in /opt/conda/lib/python3.9/site-packages (from requests>=2.0->wldata) (2.0.0)
Installing collected packages: wldata
Successfully installed wldata-0.3.0
Requirement already satisfied: cufflinks in /opt/conda/lib/python3.9/site-packages (0.17.3)
Requirement already satisfied: ipywidgets>=7.0.0 in /opt/conda/lib/python3.9/site-packages (from cufflinks) (7.7.2)
Requirement already satisfied: colorlover>=0.2.1 in /opt/conda/lib/python3.9/site-packages (from cufflinks) (0.3.0)
Requirement already satisfied: setuptools>=34.4.1 in /opt/conda/lib/python3.9/site-packages (from cufflinks) (58.2.0)
Requirement already satisfied: six>=1.9.0 in /opt/conda/lib/python3.9/site-packages (from cufflinks) (1.16.0)
Requirement already satisfied: ipython>=5.3.0 in /opt/conda/lib/python3.9/site-packages (from cufflinks) (8.9.0)
Requirement already satisfied: pandas>=0.19.2 in /opt/conda/lib/python3.9/site-packages (from cufflinks) (1.3.5)
Requirement already satisfied: plotly>=4.1.1 in /opt/conda/lib/python3.9/site-packages (from cufflinks) (5.2.1)
Requirement already satisfied: numpy>=1.9.2 in /opt/conda/lib/python3.9/site-packages (from cufflinks) (1.21.6)
Requirement already satisfied: pexpect>4.3 in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (4.8.0)
Requirement already satisfied: pickleshare in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (0.7.5)
Requirement already satisfied: stack-data in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (0.6.2)
Requirement already satisfied: matplotlib-inline in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (0.1.6)
Requirement already satisfied: prompt-toolkit<3.1.0,>=3.0.30 in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (3.0.36)
Requirement already satisfied: pygments>=2.4.0 in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (2.14.0)
Requirement already satisfied: traitlets>=5 in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (5.9.0)
Requirement already satisfied: decorator in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (5.0.9)
Requirement already satisfied: jedi>=0.16 in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (0.18.2)

Requirement already satisfied: backcall in /opt/conda/lib/python3.9/site-packages (from ipython>=5.3.0->cufflinks) (0.2.0)

Requirement already satisfied: ipykernel>=4.5.1 in /opt/conda/lib/python3.9/site-packages (from ipywidgets>=7.0.0->cufflinks) (6.19.4)

Requirement already satisfied: widgetsnbextension~=3.6.0 in /opt/conda/lib/python3.9/site-packages (from ipywidgets>=7.0.0->cufflinks) (3.6.1)

Requirement already satisfied: jupyterlab-widgets<3,>=1.0.0 in /opt/conda/lib/python3.9/site-packages (from ipywidgets>=7.0.0->cufflinks) (1.1.1)

Requirement already satisfied: ipython-genutils~=0.2.0 in /opt/conda/lib/python3.9/site-packages (from ipywidgets>=7.0.0->cufflinks) (0.2.0)

Requirement already satisfied: python-dateutil>=2.7.3 in /opt/conda/lib/python3.9/site-packages (from pandas>=0.19.2->cufflinks) (2.8.0)

Requirement already satisfied: pytz>=2017.3 in /opt/conda/lib/python3.9/site-packages (from pandas>=0.19.2->cufflinks) (2021.1)

Requirement already satisfied: tenacity>=6.2.0 in /opt/conda/lib/python3.9/site-packages (from plotly>=4.1.1->cufflinks) (8.1.0)

Requirement already satisfied: comm>=0.1.1 in /opt/conda/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (0.1.2)

Requirement already satisfied: jupyter-client>=6.1.12 in /opt/conda/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (7.2.0)

Requirement already satisfied: nest-asyncio in /opt/conda/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (1.5.6)

Requirement already satisfied: packaging in /opt/conda/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (21.3)

Requirement already satisfied: debugpy>=1.0 in /opt/conda/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (1.6.6)

Requirement already satisfied: pyzmq>=17 in /opt/conda/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (25.0.0)

Requirement already satisfied: tornado>=6.1 in /opt/conda/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (6.2)

Requirement already satisfied: psutil in /opt/conda/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (5.9.4)

Requirement already satisfied: parso<0.9.0,>=0.8.0 in /opt/conda/lib/python3.9/site-packages (from jedi>=0.16->ipython>=5.3.0->cufflinks) (0.8.3)

Requirement already satisfied: ptyprocess>=0.5 in /opt/conda/lib/python3.9/site-packages (from pexpect>4.3->ipython>=5.3.0->cufflinks) (0.7.0)

Requirement already satisfied: wcwidth in /opt/conda/lib/python3.9/site-packages (from prompt-toolkit<3.1.0,>=3.0.30->ipython>=5.3.0->cufflinks) (0.2.6)

Requirement already satisfied: notebook>=4.4.1 in /opt/conda/lib/python3.9/site-packages (from widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (6.4.12)

Requirement already satisfied: executing>=1.2.0 in /opt/conda/lib/python3.9/site-packages (from stack-data->ipython>=5.3.0->cufflinks) (1.2.0)

Requirement already satisfied: asttokens>=2.1.0 in /opt/conda/lib/python3.9/site-packages (from stack-data->ipython>=5.3.0->cufflinks) (2.2.1)

Requirement already satisfied: pure-eval in /opt/conda/lib/python3.9/site-packages (from stack-data->ipython>=5.3.0->cufflinks) (0.2.2)

Requirement already satisfied: jupyter-core>=4.9.2 in /opt/conda/lib/python3.9/site-packages (from jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (5.2.0)

Requirement already satisfied: entrypoints in /opt/conda/lib/python3.9/site-packages (from jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (0.4)

Requirement already satisfied: prometheus-client in /opt/conda/lib/python3.

9/site-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (0.16.0)
Requirement already satisfied: jinja2 in /opt/conda/lib/python3.9/site-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (3.1.2)
Requirement already satisfied: nbformat in /opt/conda/lib/python3.9/site-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (5.7.3)
Requirement already satisfied: Send2Trash>=1.8.0 in /opt/conda/lib/python3.9/site-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (1.8.0)
Requirement already satisfied: terminado>=0.8.3 in /opt/conda/lib/python3.9/site-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (0.17.1)
Requirement already satisfied: nbconvert>=5 in /opt/conda/lib/python3.9/site-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (7.2.9)
Requirement already satisfied: argon2-cffi in /opt/conda/lib/python3.9/site-packages (from notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (21.3.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/lib/python3.9/site-packages (from packaging->ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (3.0.9)
Requirement already satisfied: platformdirs>=2.5 in /opt/conda/lib/python3.9/site-packages (from jupyter-core>=4.9.2->jupyter-client>=6.1.12->ipykernel>=4.5.1->ipywidgets>=7.0.0->cufflinks) (2.6.2)
Requirement already satisfied: markupsafe>=2.0 in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (2.1.2)
Requirement already satisfied: defusedxml in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (0.7.1)
Requirement already satisfied: bleach in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (6.0.0)
Requirement already satisfied: tinycss2 in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (1.2.1)
Requirement already satisfied: jupyterlab-pygments in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (0.2.2)
Requirement already satisfied: nbclient>=0.5.0 in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (0.7.2)
Requirement already satisfied: beautifulsoup4 in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (4.9.3)
Requirement already satisfied: importlib-metadata>=3.6 in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (6.0.0)
Requirement already satisfied: pandocfilters>=1.4.1 in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (1.5.0)
Requirement already satisfied: mistune<3,>=2.0.3 in /opt/conda/lib/python3.9/site-packages (from nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (2.0.4)

Requirement already satisfied: fastjsonschema in /opt/conda/lib/python3.9/site-packages (from nbformat->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (2.16.2)

Requirement already satisfied: jsonschema>=2.6 in /opt/conda/lib/python3.9/site-packages (from nbformat->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (4.17.3)

Requirement already satisfied: argon2-cffi-bindings in /opt/conda/lib/python3.9/site-packages (from argon2-cffi->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (21.2.0)

Requirement already satisfied: zipp>=0.5 in /opt/conda/lib/python3.9/site-packages (from importlib-metadata>=3.6->nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (3.12.0)

Requirement already satisfied: pyparsing!=0.17.0,!=0.17.1,!=0.17.2,>=0.14.0 in /opt/conda/lib/python3.9/site-packages (from jsonschema>=2.6->nbformat->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (0.19.3)

Requirement already satisfied: attrs>=17.4.0 in /opt/conda/lib/python3.9/site-packages (from jsonschema>=2.6->nbformat->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (19.3.0)

Requirement already satisfied: cffi>=1.0.1 in /opt/conda/lib/python3.9/site-packages (from argon2-cffi-bindings->argon2-cffi->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (1.14.6)

Requirement already satisfied: soupsieve>1.2 in /opt/conda/lib/python3.9/site-packages (from beautifulsoup4->nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (2.3.2.post1)

Requirement already satisfied: webencodings in /opt/conda/lib/python3.9/site-packages (from bleach->nbconvert>=5->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (0.5.1)

Requirement already satisfied: pycparser in /opt/conda/lib/python3.9/site-packages (from cffi>=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook>=4.4.1->widgetsnbextension~=3.6.0->ipywidgets>=7.0.0->cufflinks) (2.20)

Collecting iso3166

Using cached iso3166-2.1.1-py3-none-any.whl (9.8 kB)

Installing collected packages: iso3166

Successfully installed iso3166-2.1.1

/opt/conda/lib/python3.9/site-packages/geopandas/_compat.py:111: UserWarning:

The Shapely GEOS version (3.10.3-CAPI-1.16.1) is incompatible with the GEOS version PyGEOS was compiled with (3.10.4-CAPI-1.16.2). Conversions between both will be slow.

```
In [3]: #wbdata.get_source() to get all sources
        #wbdata.get_topic()
        #wbdata.get_indicator()

        source_id = 40 #Population Estimates and Projections
        indicators = wbdata.get_indicator(source=source_id)
```

[#A] Population Pyramids

```
In [4]: # Data from WDI on age-sex comes in the forms of variables
        # which take the form "SP.POP.LLHH.MA" for males
```

```

# and "SP.POP.LLHH.FE" for females, where LL is the *low* end of
# age range, like "05" for 5-yo, and HH is the *high* end.

# We construct a list of age-ranges.

# Start with an empty list of age-rages
age_ranges = []

# Ranges top out at 80, and go in five year increments
for i in range(0,80,5):
    age_ranges.append(f"{i:02d}"+"f"{i+4:02d}")

age_ranges.append("80UP")

male_variables = {"SP.POP."+age_range+".MA": "Males "+age_range for age_range
female_variables = {"SP.POP."+age_range+".FE": "Females "+age_range for age_r

variables = male_variables
variables.update(female_variables)

# WLD is the World; substitute your own code or list of codes.
# Remember you can search for the appropriate codes using
# wbdata.search_countries("")

df = wbdata.get_dataframe(variables, country="KOR")

py.init_notebook_mode(connected=True)

layout = go.Layout(barmode='overlay',
                    yaxis=go.layout.YAxis(range=[0, 90], title='Age'),
                    xaxis=go.layout.XAxis(title='Number'))

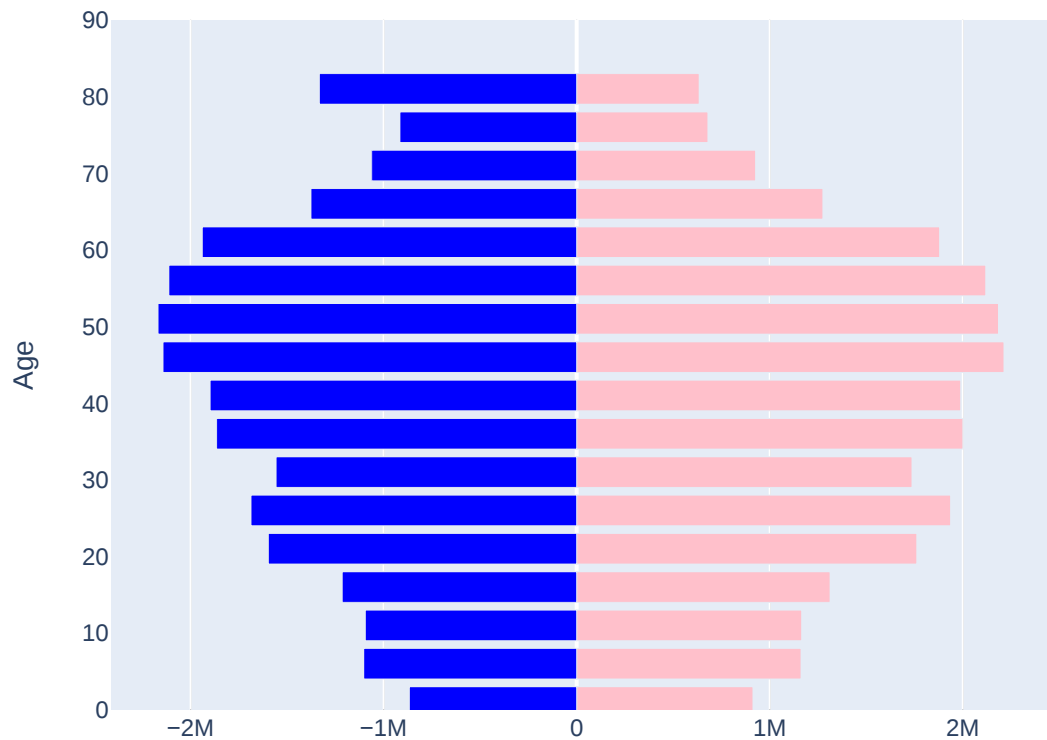
year = 2020

bins = [go.Bar(x = df.loc[str(year),:].filter(regex="Male").values,
                y = [int(s[:2])+1 for s in age_ranges],
                orientation='h',
                name='Men',
                marker=dict(color='pink'),
                hoverinfo='skip'
                ),

        go.Bar(x = -df.loc[str(year),:].filter(regex="Female").values,
                y=[int(s[:2])+1 for s in age_ranges],
                orientation='h',
                name='Women',
                marker=dict(color='blue'),
                hoverinfo='skip',
                )
        ]

py.iplot(dict(data=bins, layout=layout))

```



```
In [5]: df = wbdata.get_dataframe(variables, country="MMR")

py.init_notebook_mode(connected=True)

layout = go.Layout(barmode='overlay',
                    yaxis=go.layout.YAxis(range=[0, 90], title='Age'),
                    xaxis=go.layout.XAxis(title='Number'))

year = 2020

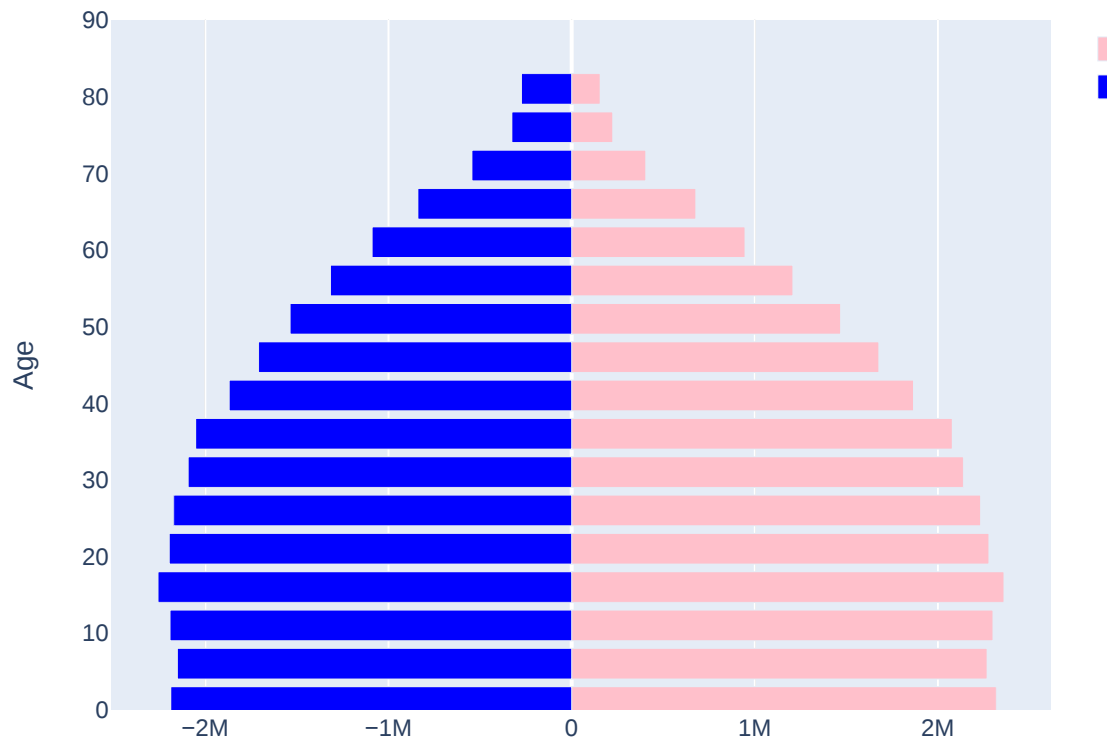
bins = [go.Bar(x = df.loc[str(year),:].filter(regex="Male").values,
               y = [int(s[:2])+1 for s in age_ranges],
               orientation='h',
               name='Men',
               marker=dict(color='pink'),
               hoverinfo='skip'
               ),

        go.Bar(x = -df.loc[str(year),:].filter(regex="Female").values,
               y=[int(s[:2])+1 for s in age_ranges],
               orientation='h',
               name='Women',
               marker=dict(color='blue'),
               hoverinfo='skip',
               )
        ]
```

```

    )
]
py.ipplot(dict(data=bins, layout=layout))

```



[C#] Animated Population Pyramids

```

In [40]: age_ranges = []

# Ranges top out at 80, and go in five year increments
for i in range(0,80,5):
    age_ranges.append(f"{i:02d}"+"f"{i+4:02d}")

age_ranges.append("80UP")

print(age_ranges)

male_variables = {"SP.POP."+age_range+".MA":"Males "+age_range for age_range
female_variables = {"SP.POP."+age_range+".FE":"Females "+age_range for age_r

variables = male_variables
variables.update(female_variables)

print(variables)

```



```

# MMR
df = wbdata.get_dataframe(variables, country="MMR")

py.init_notebook_mode(connected=True)

layout = go.Layout(barmode='overlay',
                    yaxis=go.layout.YAxis(range=[0, 90], title='Age'),
                    xaxis=go.layout.XAxis(title='Number'))

year = 2021

bins = [go.Bar(x = df.loc[str(year),:].filter(regex="Male").values,
               y = [int(s[:2])+1 for s in age_ranges],
               orientation='h',
               name='Men',
               marker=dict(color='purple'),
               hoverinfo='skip'
               ),

        go.Bar(x = -df.loc[str(year),:].filter(regex="Female").values,
               y=[int(s[:2])+1 for s in age_ranges],
               orientation='h',
               name='Women',
               marker=dict(color='pink'),
               hoverinfo='skip'
               )
        ]

py.iplot(dict(data=bins, layout=layout))

# Count down by increments of 20 years
years = range(2021, 1961, -20)

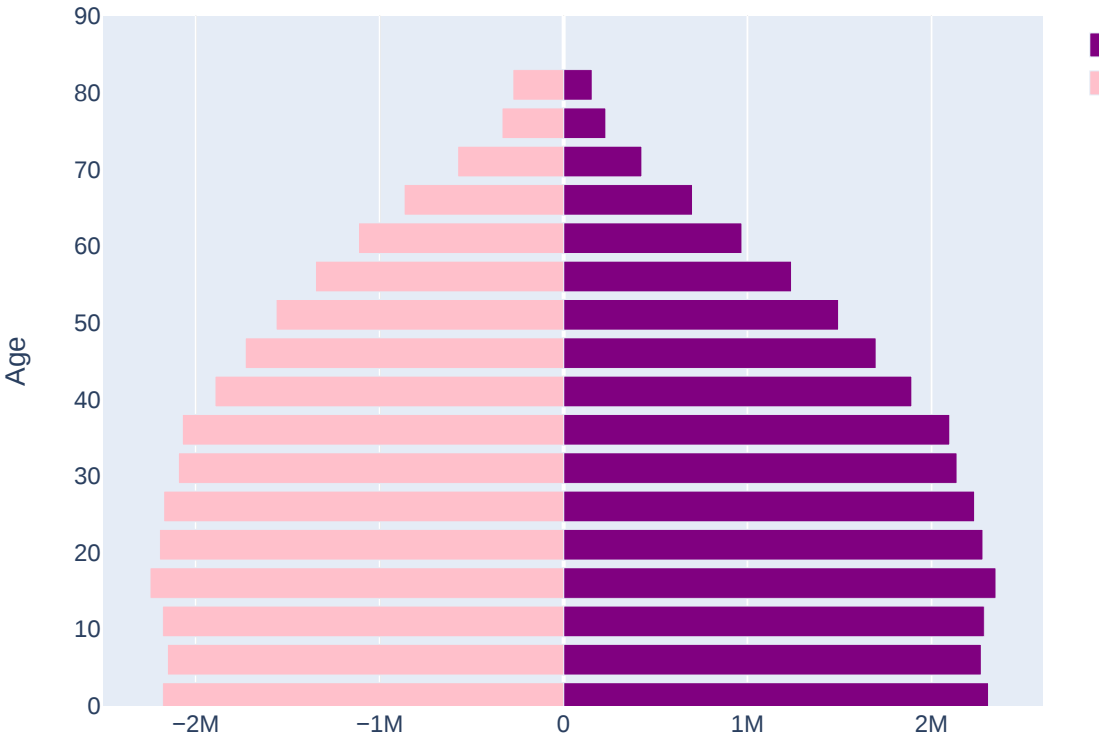
# This makes a list of graphs, year by year
bins = [go.Bar(x = df.loc[str(year),:].filter(regex="Male").values,
               y = [int(s[:2])+1 for s in age_ranges],
               orientation='h',
               name='Men {:d}'.format(year),
               hoverinfo='skip'
               )
        for year in years]

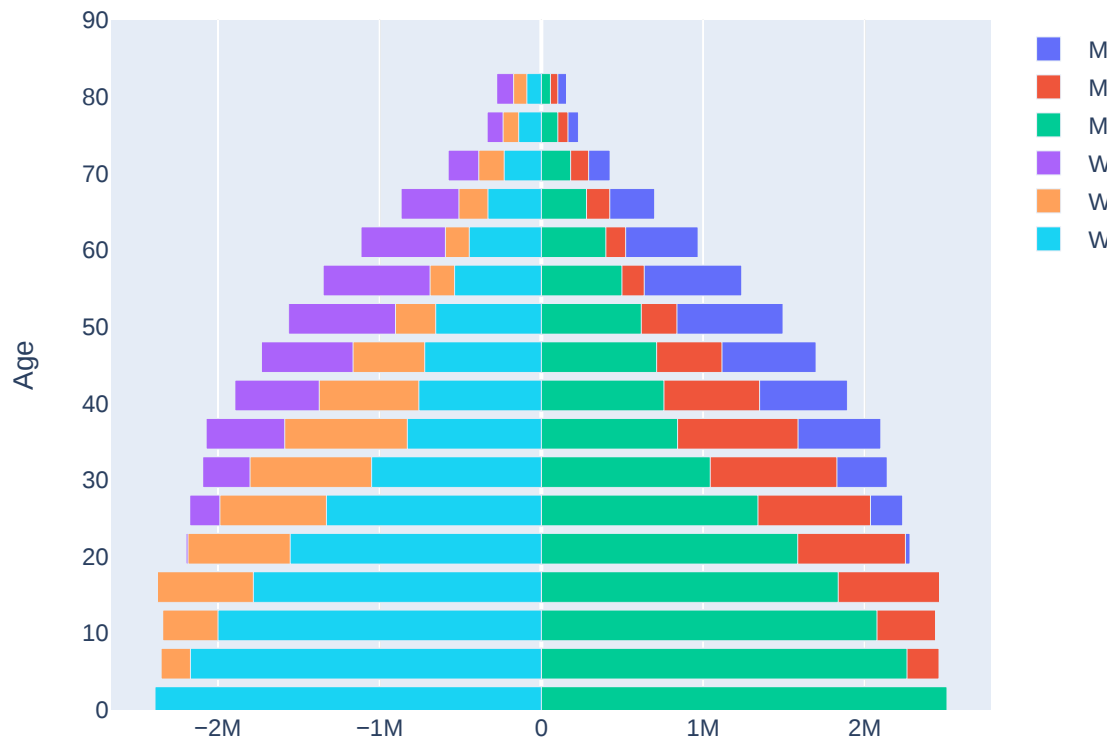
bins += [go.Bar(x = -df.loc[str(year),:].filter(regex="Female").values,
                y=[int(s[:2])+1 for s in age_ranges],
                orientation='h',
                name='Women {:d}'.format(year),
                hoverinfo='skip'
                )
        for year in years]

py.iplot(dict(data=bins, layout=layout))

```

['0004', '0509', '1014', '1519', '2024', '2529', '3034', '3539', '4044', '4549', '5054', '5559', '6064', '6569', '7074', '7579', '80UP']
{'SP.POP.0004.MA': 'Males 0004', 'SP.POP.0509.MA': 'Males 0509', 'SP.POP.1014.MA': 'Males 1014', 'SP.POP.1519.MA': 'Males 1519', 'SP.POP.2024.MA': 'Males 2024', 'SP.POP.2529.MA': 'Males 2529', 'SP.POP.3034.MA': 'Males 3034', 'SP.POP.3539.MA': 'Males 3539', 'SP.POP.4044.MA': 'Males 4044', 'SP.POP.4549.MA': 'Males 4549', 'SP.POP.5054.MA': 'Males 5054', 'SP.POP.5559.MA': 'Males 5559', 'SP.POP.6064.MA': 'Males 6064', 'SP.POP.6569.MA': 'Males 6569', 'SP.POP.7074.MA': 'Males 7074', 'SP.POP.7579.MA': 'Males 7579', 'SP.POP.80UP.MA': 'Males 80UP', 'SP.POP.0004.FE': 'Females 0004', 'SP.POP.0509.FE': 'Females 0509', 'SP.POP.1014.FE': 'Females 1014', 'SP.POP.1519.FE': 'Females 1519', 'SP.POP.2024.FE': 'Females 2024', 'SP.POP.2529.FE': 'Females 2529', 'SP.POP.3034.FE': 'Females 3034', 'SP.POP.3539.FE': 'Females 3539', 'SP.POP.4044.FE': 'Females 4044', 'SP.POP.4549.FE': 'Females 4549', 'SP.POP.5054.FE': 'Females 5054', 'SP.POP.5559.FE': 'Females 5559', 'SP.POP.6064.FE': 'Females 6064', 'SP.POP.6569.FE': 'Females 6569', 'SP.POP.7074.FE': 'Females 7074', 'SP.POP.7579.FE': 'Females 7579', 'SP.POP.80UP.FE': 'Females 80UP'}





In [27]: # KOR

```
df = wbdata.get_dataframe(variables, country="KOR")

py.init_notebook_mode(connected=True)

layout = go.Layout(barmode='overlay',
                    yaxis=go.layout.YAxis(range=[0, 90], title='Age'),
                    xaxis=go.layout.XAxis(title='Number'))

year = 2021

bins = [go.Bar(x = df.loc[str(year),:].filter(regex="Male").values,
                y = [int(s[:2])+1 for s in age_ranges],
                orientation='h',
                name='Men',
                marker=dict(color='purple'),
                hoverinfo='skip'
                ),

        go.Bar(x = -df.loc[str(year),:].filter(regex="Female").values,
                y=[int(s[:2])+1 for s in age_ranges],
                orientation='h',
                name='Women',
```

```

        marker=dict(color='pink'),
        hoverinfo='skip',
    )
]
py.iplot(dict(data=bins, layout=layout))

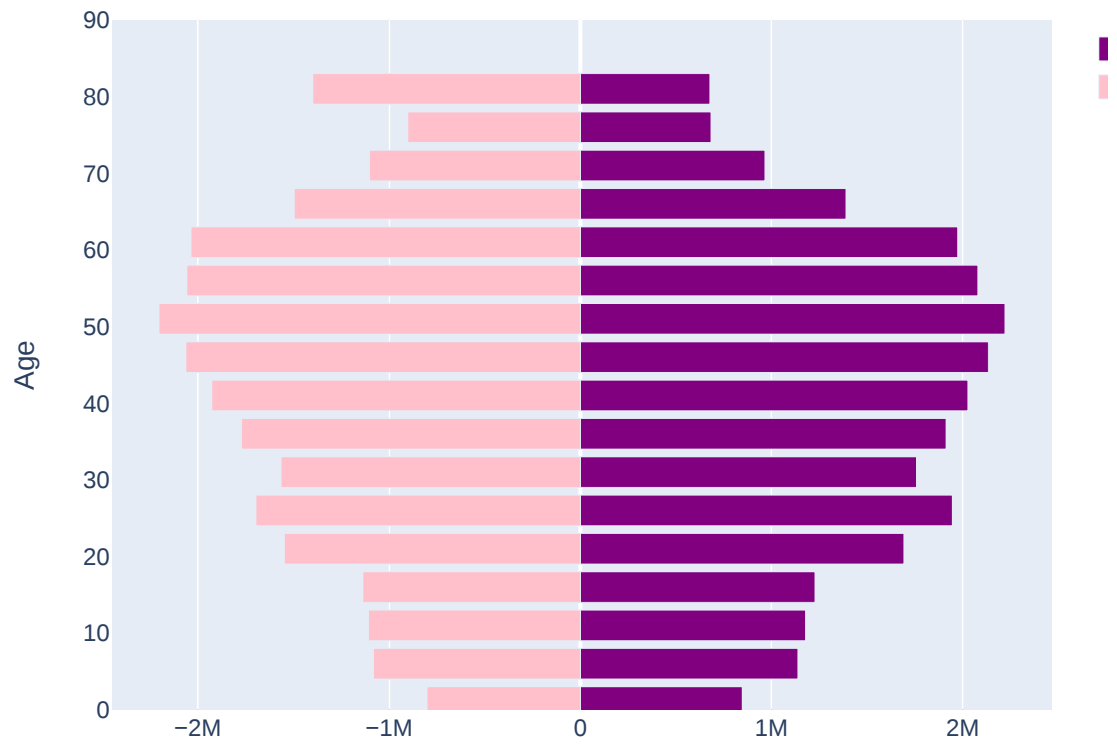
# Count down by increments of 20 years
years = range(2021,1961,-20)

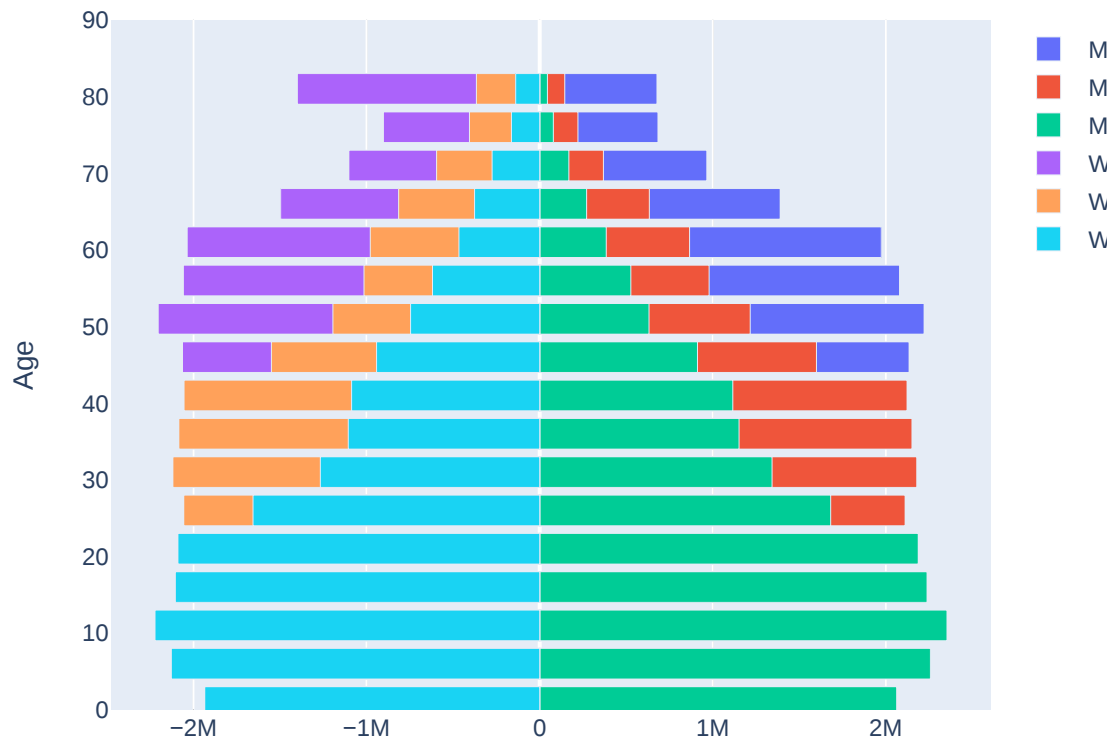
# This makes a list of graphs, year by year
bins = [go.Bar(x = df.loc[str(year),:].filter(regex="Male").values,
               y = [int(s[:2])+1 for s in age_ranges],
               orientation='h',
               name='Men {:d}'.format(year),
               hoverinfo='skip'
            )
        for year in years]

bins += [go.Bar(x = -df.loc[str(year),:].filter(regex="Female").values,
               y=[int(s[:2])+1 for s in age_ranges],
               orientation='h',
               name='Women {:d}'.format(year),
               hoverinfo='skip',
            )
        for year in years]

py.iplot(dict(data=bins, layout=layout))

```





[#A] Population Dataframe

```
In [43]: def dataframefunction(agelowerbound=0, ageupperbound=80, givencountry='world',
    age_ranges = []
    editedcountry = givencountry
    if len(givencountry) != 3:
        editedcountry = wbdata.search_countries(givencountry)[0]['id']
        if editedcountry == 'ARB' and givencountry == 'World':
            editedcountry = 'WLD'
    # Ranges top out at 80, and go in five year increments
    for i in range(agelowerbound, ageupperbound, 5):
        age_ranges.append(f"{i:02d}" + f"{i+4:02d}")

    if ageupperbound == 80:
        age_ranges.append("80UP")

    male_variables = {"SP.POP." + age_range + ".MA": "Males " + age_range for age_range in age_ranges}
    female_variables = {"SP.POP." + age_range + ".FE": "Females " + age_range for age_range in age_ranges}

    variables = male_variables
    variables.update(female_variables)

    country = editedcountry
```

```

df = wbdata.get_dataframe(variables, country)
if len(country) == 3:
    country = wbdata.get_country(country)[0]['name']
df['Country'] = [country for i in range(0, len(df))]
df.insert(0, "Country", df.pop("Country"))
return df
df

```

Out[43]:

	Males 0004	Males 0509	Males 1014	Males 1519	Males 2024	Males 2529	Males 3034	Males 3539
date								
2021	2310513.0	2271404.0	2287917.0	2351085.0	2280229.0	2234744.0	2139374.0	2099590.0
2020	2318768.0	2268374.0	2300425.0	2360501.0	2278275.0	2232782.0	2138928.0	2078186.0
2019	2320731.0	2272722.0	2315383.0	2362041.0	2284447.0	2229585.0	2147964.0	2048967.0
2018	2318211.0	2282127.0	2335591.0	2358455.0	2295470.0	2221996.0	2163090.0	2016469.0
2017	2313547.0	2294378.0	2360918.0	2350164.0	2305666.0	2213232.0	2171965.0	1985639.0
...
1964	1940328.0	1624079.0	1378332.0	1062397.0	915000.0	864320.0	820503.0	720349.0
1963	1905143.0	1584722.0	1319238.0	1024788.0	903413.0	864333.0	809531.0	704501.0
1962	1872075.0	1542314.0	1251804.0	1000488.0	896565.0	864135.0	796036.0	688656.0
1961	1839764.0	1497163.0	1190921.0	976197.0	893178.0	862143.0	781703.0	674505.0
1960	1806858.0	1450332.0	1134966.0	954682.0	891857.0	857596.0	767393.0	662759.0

62 rows × 34 columns

[A#] Population Statistics

```

In [54]: def population(year='', sex='', age_range=(0), place=''):
    newplace = place
    if len(place) != 3:
        newplace = wbdata.search_countries(place)[0]['id']
        if newplace == 'ARB' and place == 'World':
            newplace = 'WLD'
    upperage = age_range[1]
    lowerage = age_range[0]
    if upperage > 80:
        upperage = 80
    theage = (upperage - lowerage)//5 + 1
    theyear = 2021 - int(year)
    if sex == 'Male' or sex == 'male':
        value = dataframefunction(lowerage, upperage, newplace).iloc[theyear]
        return 'In ' + str(year) + ', there are ' + str(value) + ' ' + str(s
    if sex == 'Female' or sex == 'female':
        value = dataframefunction(lowerage, upperage, newplace).iloc[theyear]
        return 'In ' + str(year) + ', there are ' + str(value) + ' ' + str(s
    if sex == 'People' or sex == 'people':
        value = dataframefunction(lowerage, upperage, newplace).iloc[theyear]
        return 'In ' + str(year) + ', there are ' + str(value) + ' ' + str(s

```

```
population(year='2011', sex='female', age_range=(40, 80), place='MMR')
```

Out[54]: 'In 2011, there are 227136.0 females aged 40 to 80 in the Myanmar'

In []:

[#C] Other Visualization Tools

1: GDP Visualizations showing the growth rate of GDP over time in South Korea vs Myanmar and how GDP has changed over time in each country

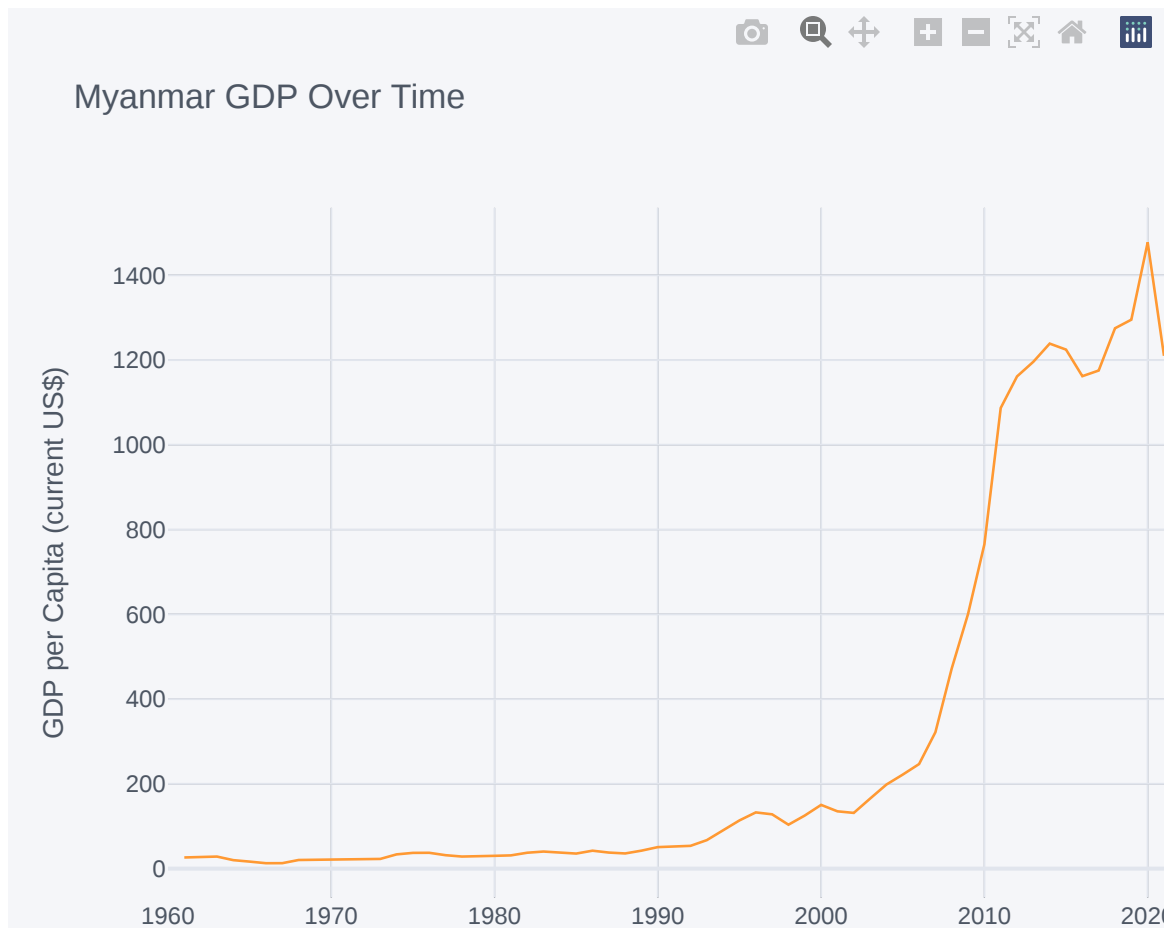
```
In [13]: # Give variable for clarity
variable_labels = {"NY.GDP.PCAP.CD": "GDP per capita"}

myanmar = wbdata.get_dataframe(variable_labels, country="MMR")

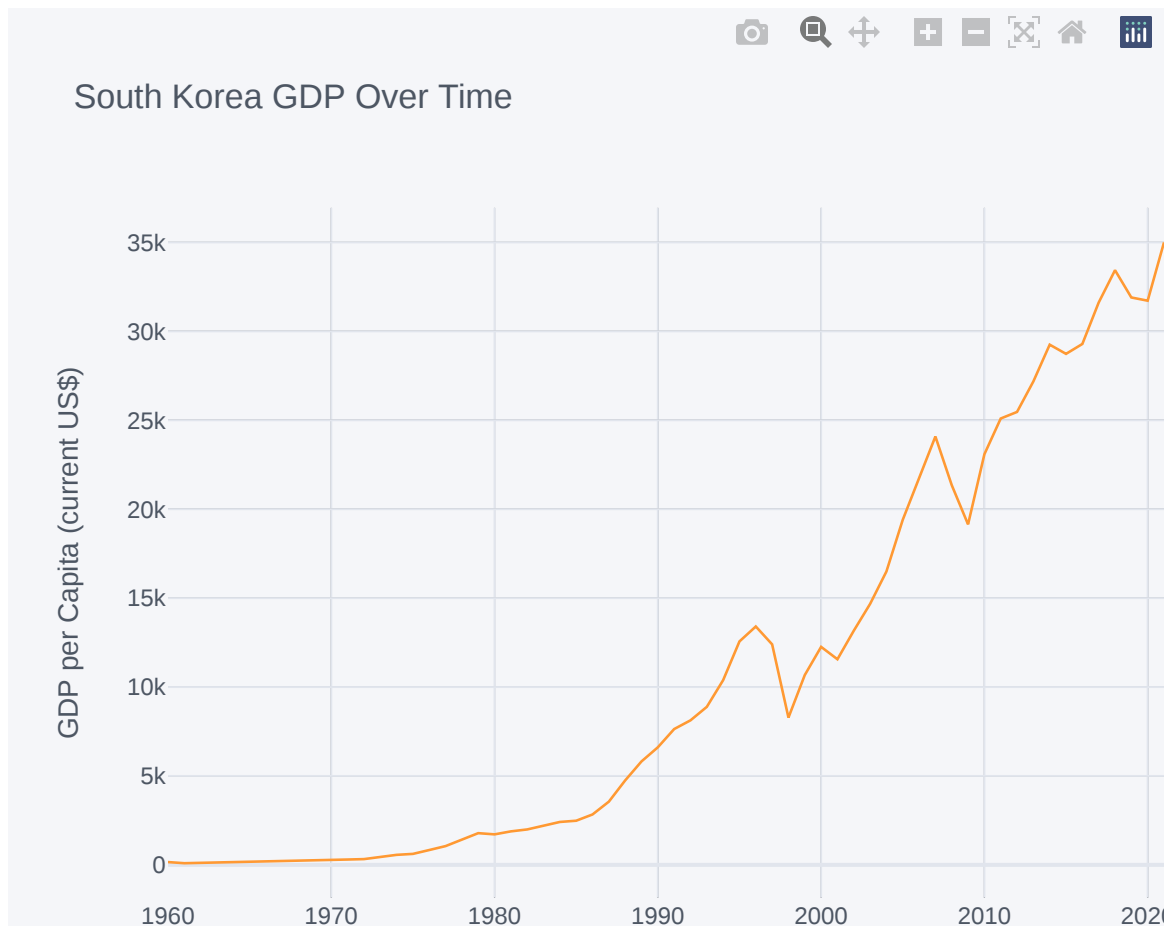
# Date index is of type string; change to integers
myanmar.index = myanmar.index.astype(int)

# Print a few years' data
myanmar.head()

myanmar.iplot(title="Myanmar GDP Over Time", xTitle='Year', yTitle='GDP per Ca
```

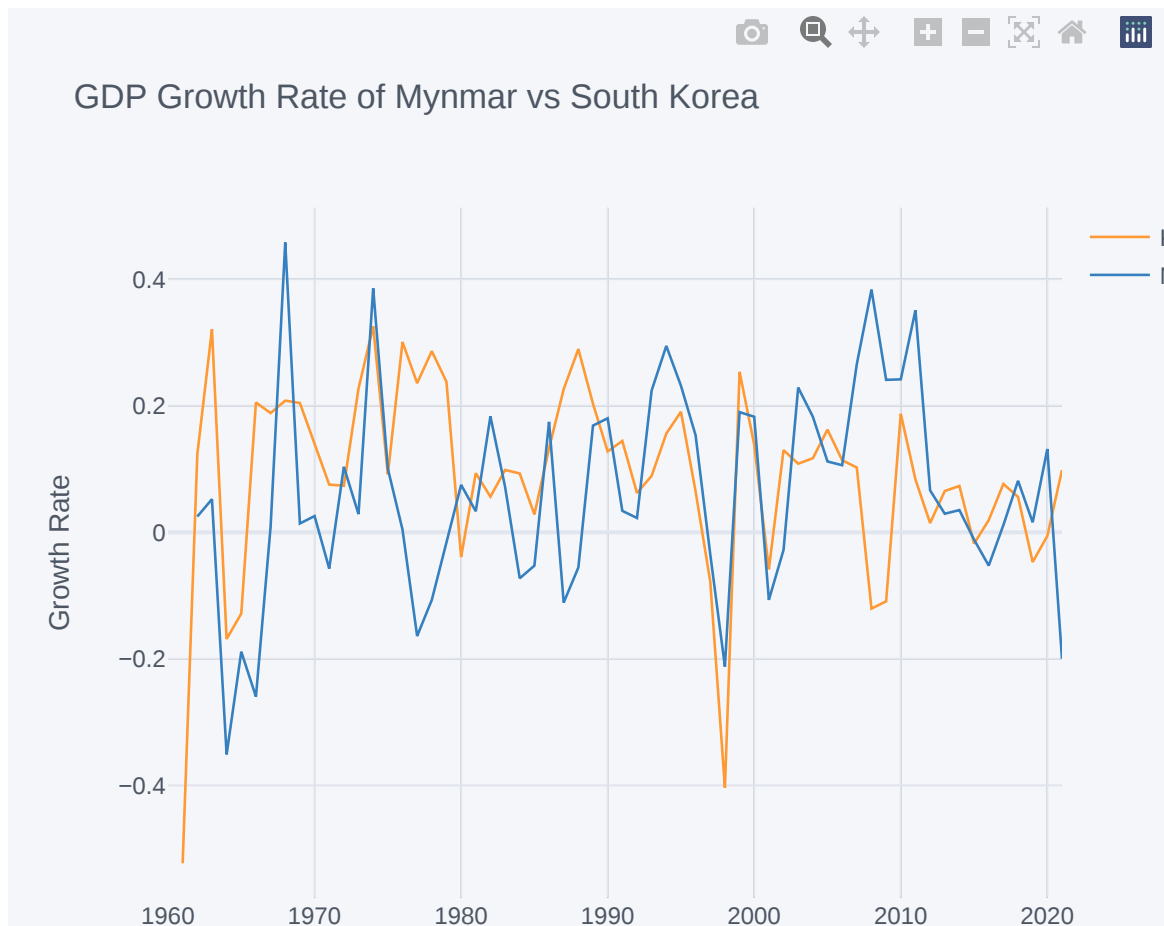



```
In [12]: variable_labels = {"NY.GDP.PCAP.CD": "GDP per capita"}
south_korea = wbdata.get_dataframe(variable_labels, country="KOR")
south_korea.index = south_korea.index.astype(int)
south_korea.head()
south_korea.iplot(title="South Korea GDP Over Time", xTitle='Year', yTitle='GDP per Capita')
```



```
In [14]: gdp_indicator = {"NY.GDP.PCAP.CD": "GDP per capita"}
countries = {"MMR": "Mynmar",
            "KOR": "South Korea"}

gdp = wbdata.get_dataframe(gdp_indicator, country = countries).squeeze().uns
gdp.iplot(title="GDP per Capita of Myanmar and South Korea Over Time", xTitle
```

2. Education Visualizations of South Korea and Myanmar showing the number of children out of primary school, the number of children in secondary education, and the percent of children enrolled in primary school

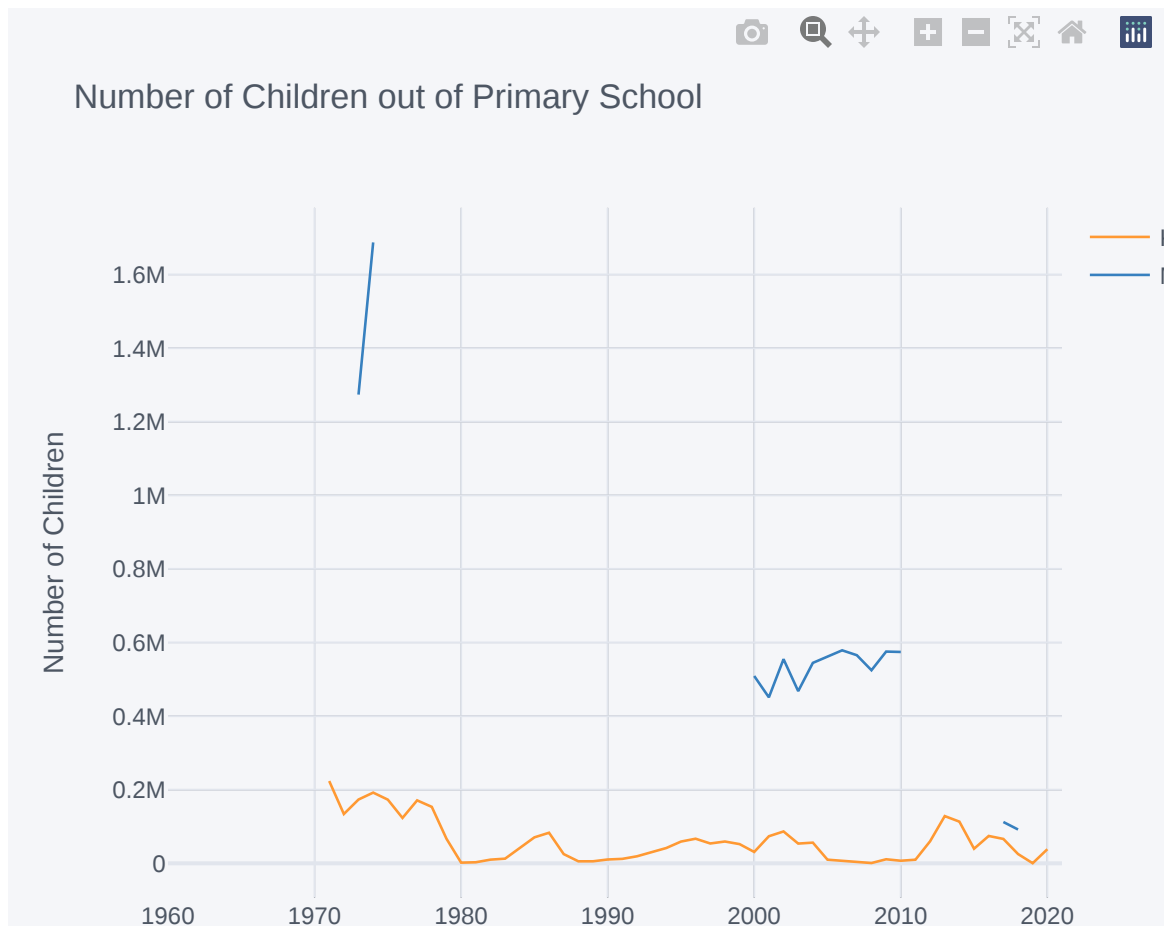
```
In [16]: #variable_labels = {"SE.PRM.UNER": "Children out of primary school"}

# Three letter codes come from wbdata.get_country()
countries = {"MMR": "Myanmar",
             "KOR": "South Korea"}
variable_labels1 = {"SE.PRM.UNER": "Children out of primary school"}

df = wbdata.get_dataframe(variable_labels1, country = countries).squeeze()

df = df.unstack('country')
# Date index is of type string; change to integers
df.index = df.index.astype(int)

# Differences (over time) in logs give us growth rates
df.iplot(title="Number of Children out of Primary School",
          yTitle="Number of Children", xTitle='Year')
```

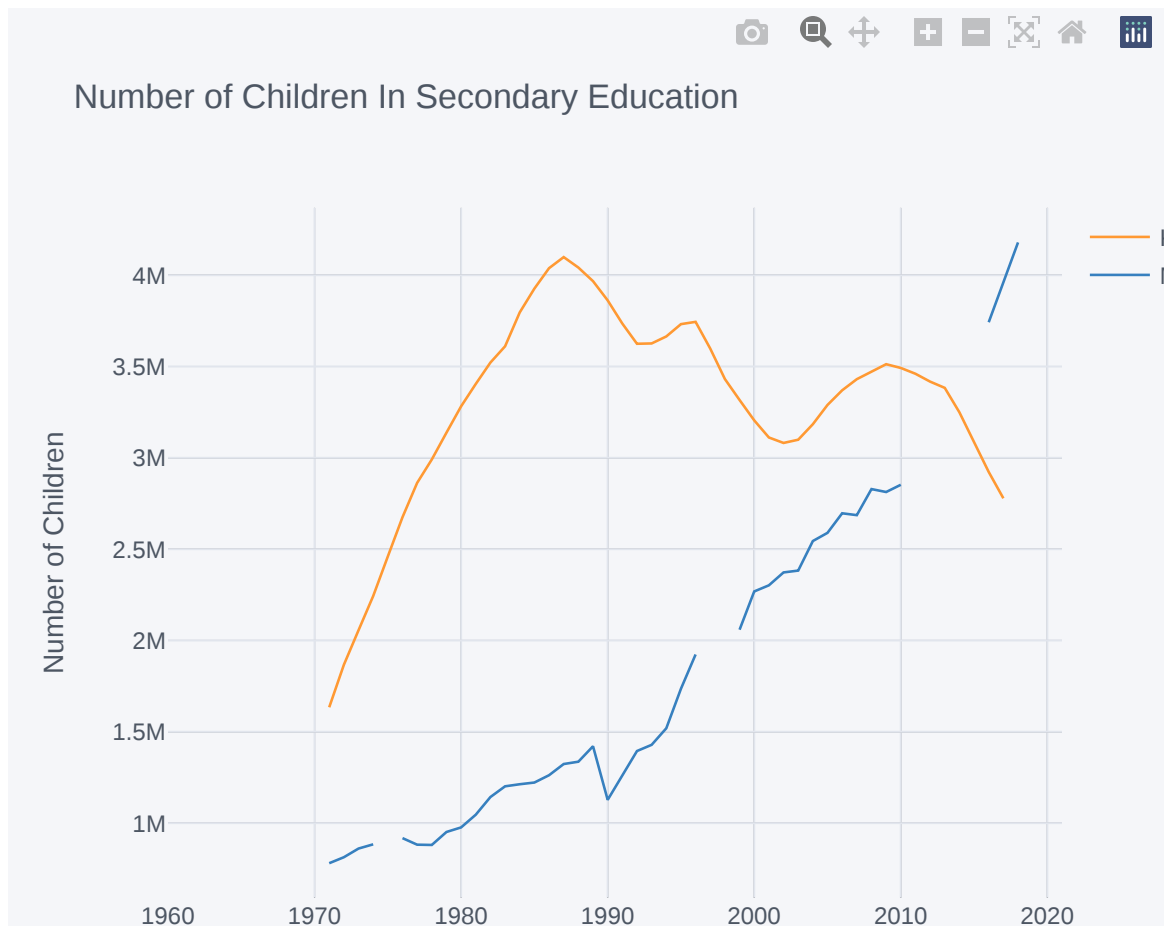


```
In [17]: variable_labels2 = {"SE.SEC.ENRL.GC": "Secondary Education"}
countries = {"MMR": "Myanmar",
            "KOR": "South Korea"}

df = wbdata.get_dataframe(variable_labels2, country = countries).squeeze()

df = df.unstack('country')
# Date index is of type string; change to integers
df.index = df.index.astype(int)

# Differences (over time) in logs give us growth rates
df.iplot(title="Number of Children In Secondary Education",
         yTitle="Number of Children", xTitle='Year')
```

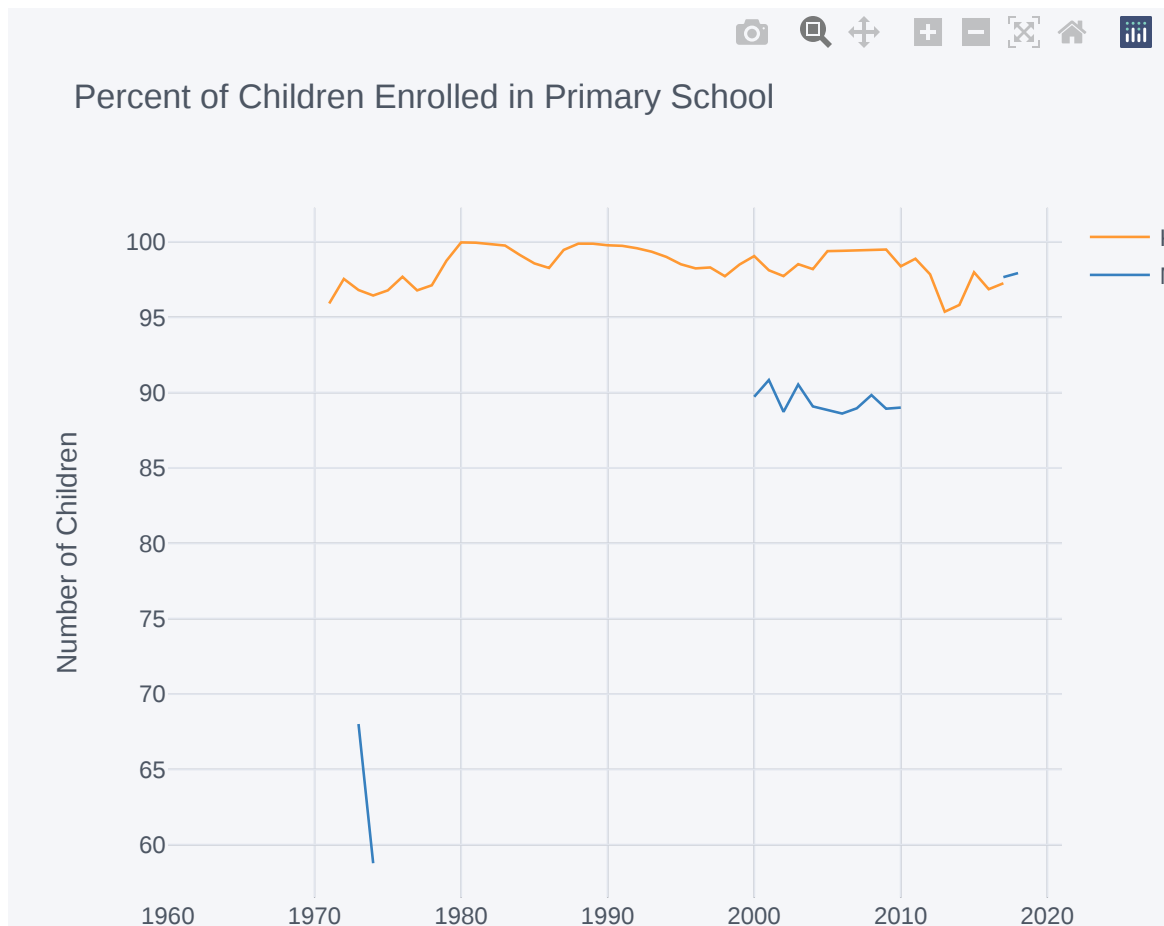


```
In [18]: variable_labels3 = {"SE.PRM.NENR": "Percent Net Primary School Enrollment"}
countries = {"MMR": "Myanmar",
            "KOR": "South Korea"}

df = wbdata.get_dataframe(variable_labels3, country = countries).squeeze()

df = df.unstack('country')
# Date index is of type string; change to integers
df.index = df.index.astype(int)

# Differences (over time) in logs give us growth rates
df.iplot(title="Percent of Children Enrolled in Primary School",
         yTitle="Number of Children", xTitle='Year')
```



3. Tax Visualizations showing the tax revenue over time for Myanmar and South Korea

```
In [19]: variable_labels4 = {"GC.TAX.TOTL.GD.ZS": "Tax Revenue"}

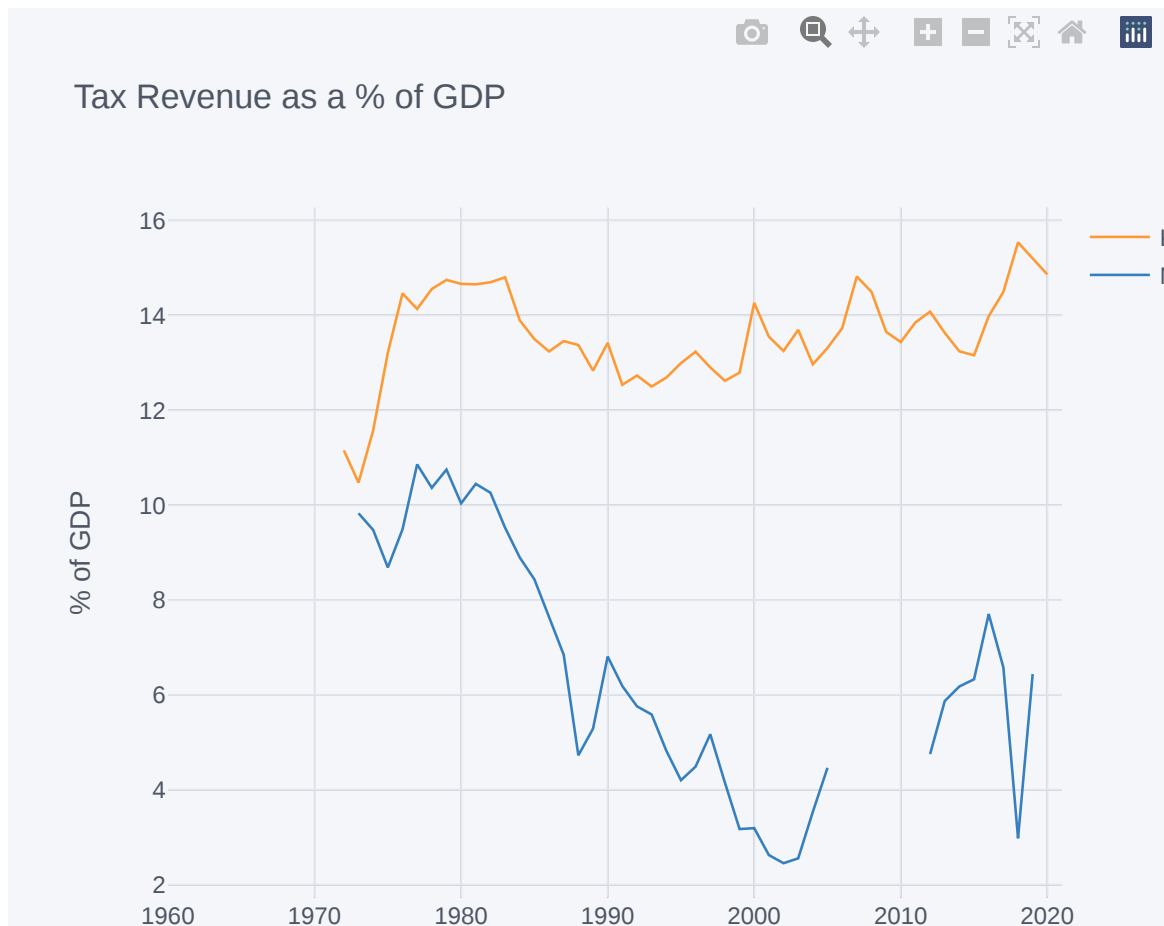
Myanmar4 = wbdata.get_dataframe(indicators=variable_labels2, country="MMR")
Korea4 = wbdata.get_dataframe(variable_labels4, country="KOR")

countries = {"MMR": "Myanmar",
             "KOR": "South Korea"}

df = wbdata.get_dataframe(variable_labels4, country = countries).squeeze()

df = df.unstack('country')
# Date index is of type string; change to integers
df.index = df.index.astype(int)

# Differences (over time) in logs give us growth rates
df.iplot(title="Tax Revenue as a % of GDP",
          yTitle="% of GDP", xTitle='Year')
```



4. Foreign Direct Investment Visualizations showing how FDI changes between Myanmar and South Korea

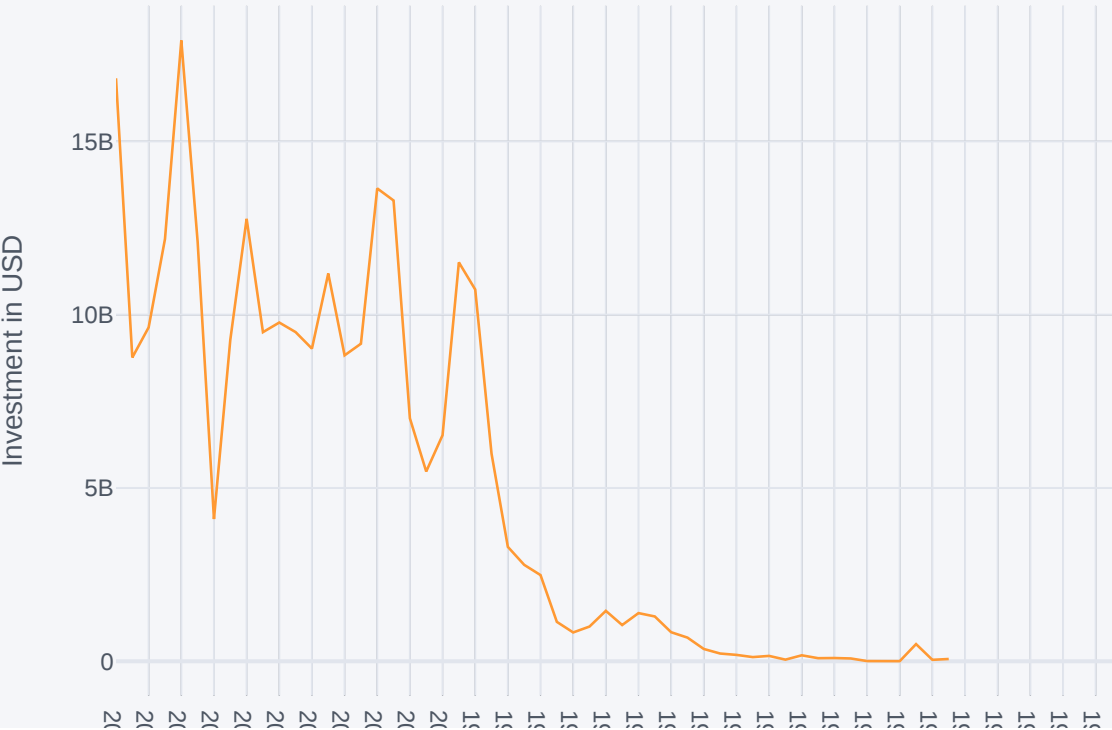
```
In [20]: variable_labels4 = {"BX.KLT.DINV.CD.WD": "Foreign Direct Investment, net inflows (BoP, current)",
                             "BX.KLT.DINV.CD.WD.KR": "Foreign Direct Investment, net inflows (BoP, current) - Korea",
                             "BX.KLT.DINV.CD.WD.MM": "Foreign Direct Investment, net inflows (BoP, current) - Myanmar"}

korea_fdi = wbdata.get_dataframe(variable_labels4, country="KOR")
myanmar_fdi = wbdata.get_dataframe(variable_labels4, country="MMR")

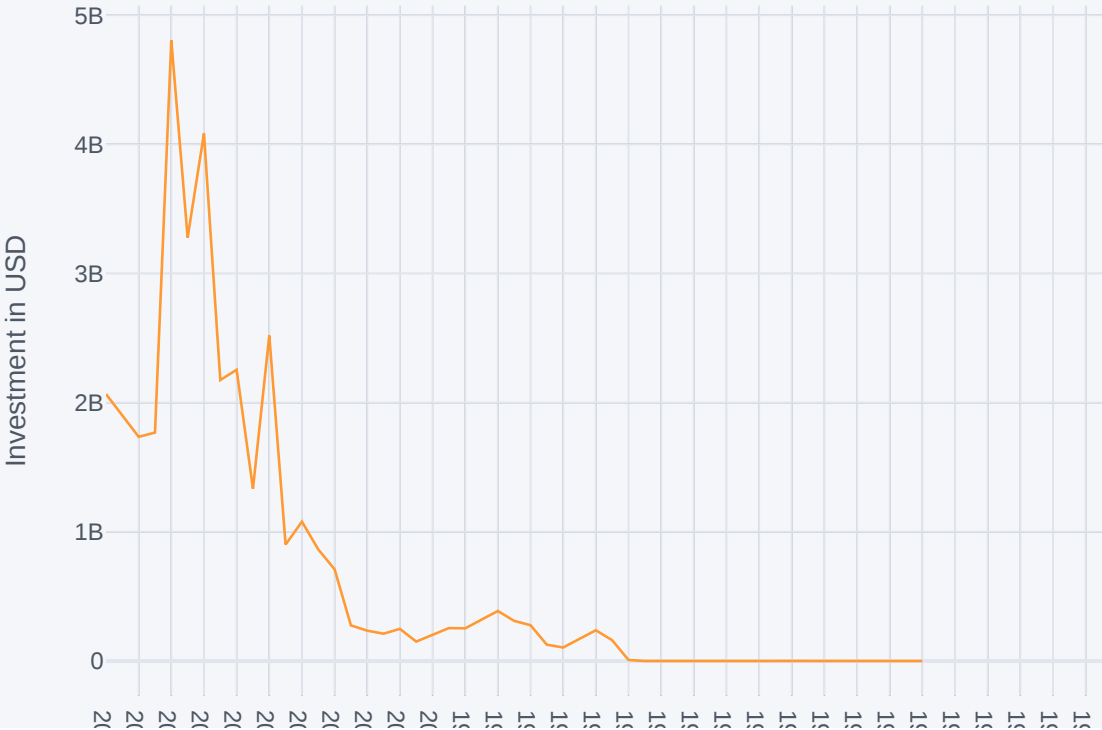
korea_fdi.iplot(title="Foreign Direct Investment Inflows", xTitle='Year', yTitle='FDI Inflows (BoP, current)')
myanmar_fdi.iplot(title="Foreign Direct Investment Inflows", xTitle='Year', yTitle='FDI Inflows (BoP, current)')

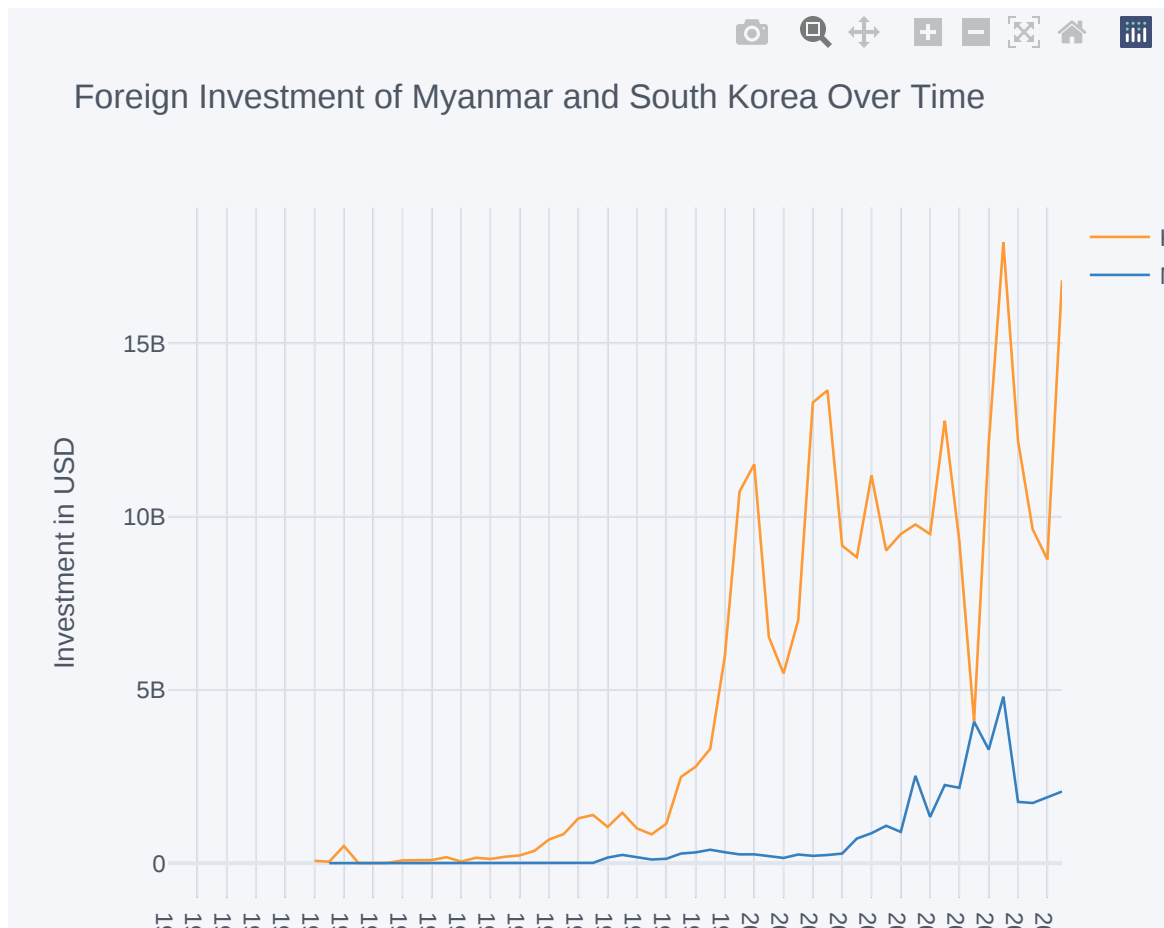
foreign_investment = wbdata.get_dataframe(variable_labels4, country=country)
foreign_investment.iplot(title="Foreign Investment of Myanmar and South Korea", xTitle='Year', yTitle='FDI Inflows (BoP, current)')
```


Foreign Direct Investment Inflows



Foreign Direct Investment Inflows





5. Export of goods and services visualizations showing how FDI changes between Myanmar and South Korea

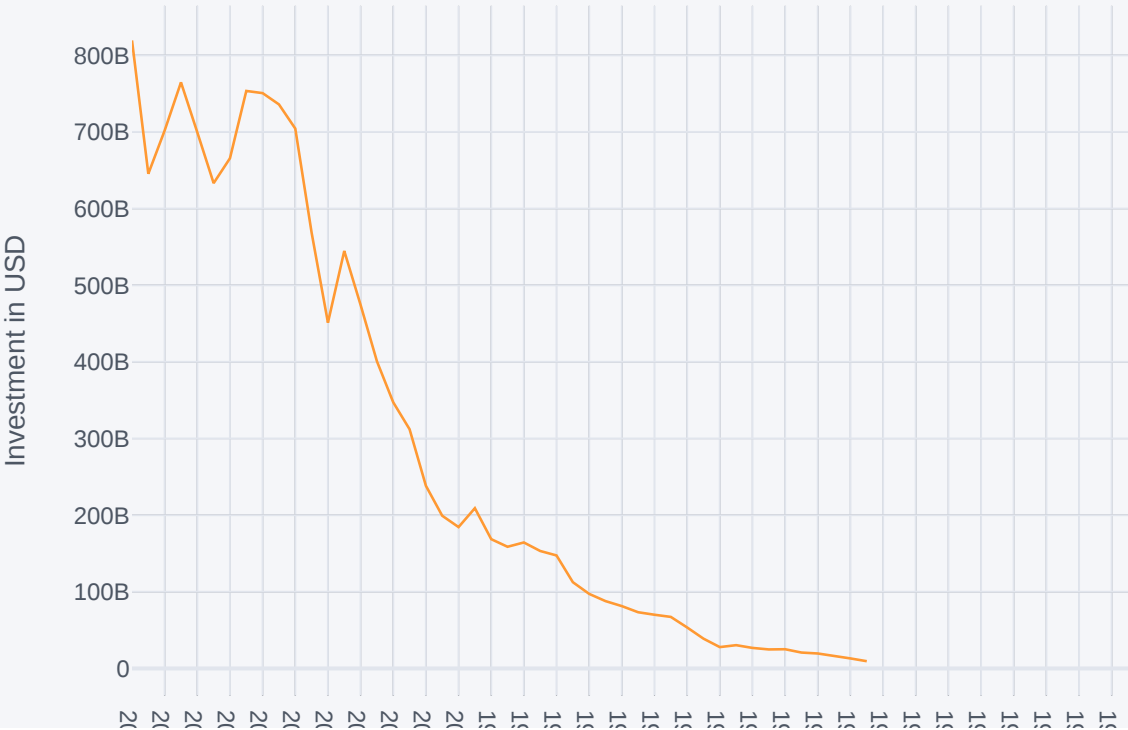
```
In [25]: variable_labels5 = {"BX.GSR.TOTL.CD": "Exports of Goods and Services in USD"}

korea_exports = wbdata.get_dataframe(variable_labels5, country="KOR")
myanmar_exports = wbdata.get_dataframe(variable_labels5, country = "MMR")

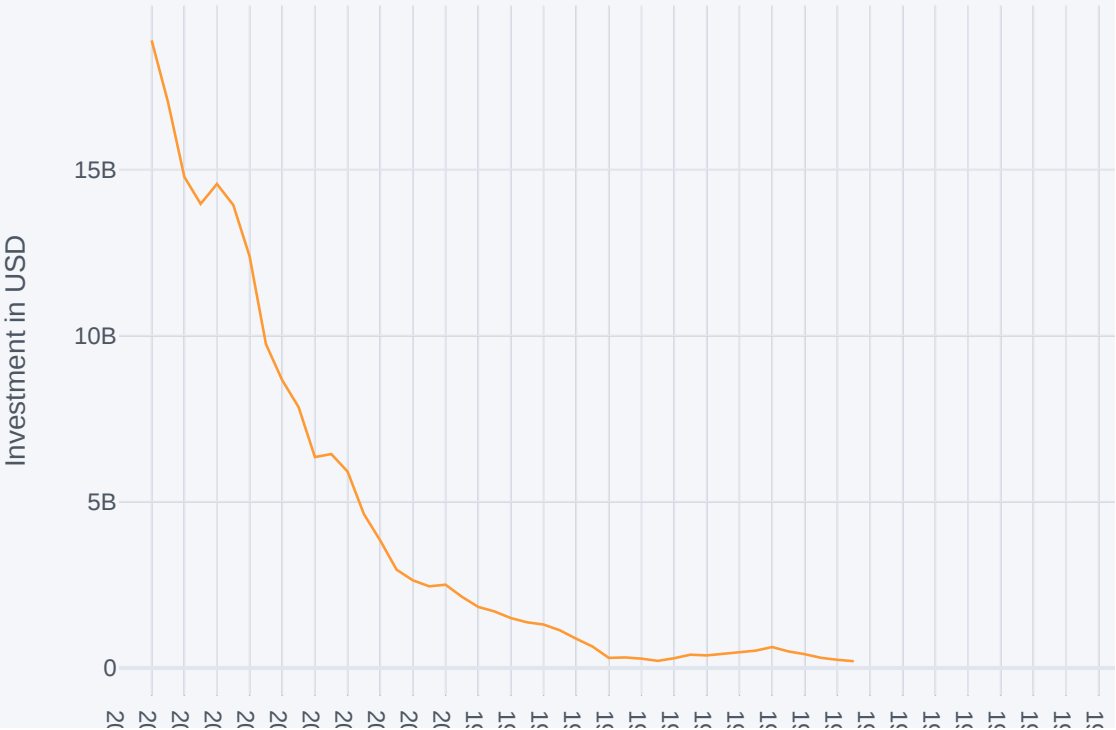
exports= wbdata.get_dataframe(variable_labels5, country= countries).squeeze()

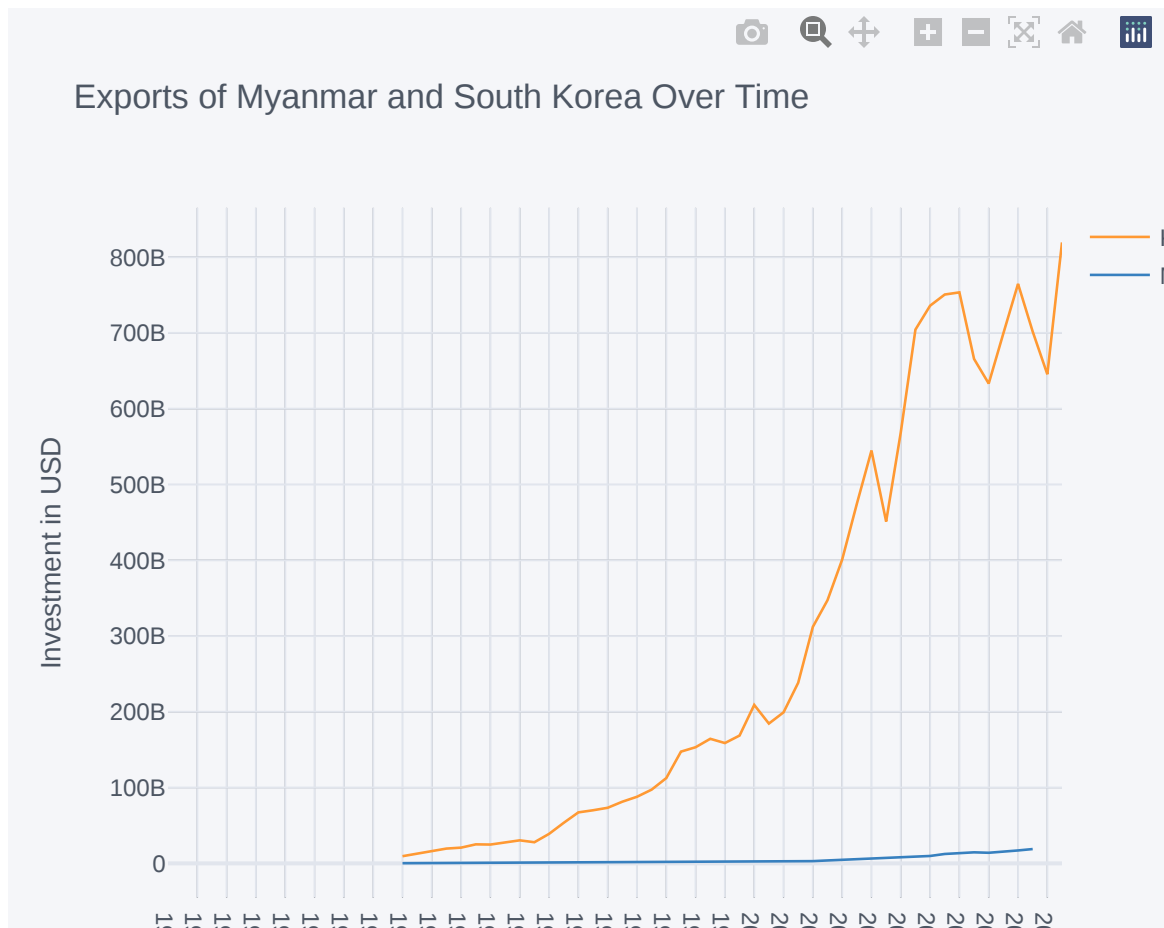
korea_exports.iplot(title= "Exports of South Korea Over Time", xTitle= 'Year')
myanmar_exports.iplot(title= "Exports of Myanmar Over Time", xTitle= 'Year')
exports.iplot(title= "Exports of Myanmar and South Korea Over Time", xTitle=
```

Exports of South Korea Over Time



Exports of Myanmar Over Time





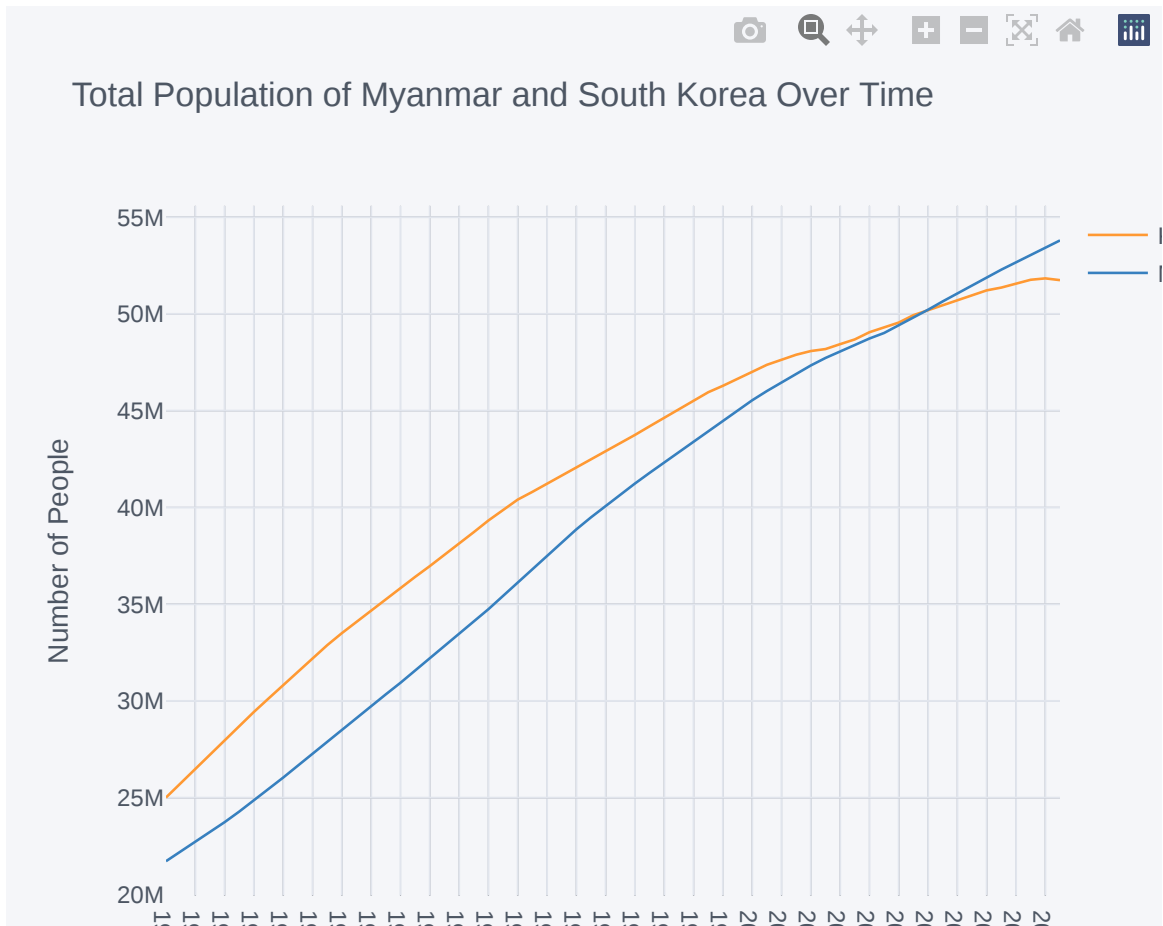
6. Population Visualizations of South Korea and Myanmar over Time

```
In [42]: variable_labels6 = {"SP.POP.TOTL": "Total Population"}

korea_exports = wbdata.get_dataframe(variable_labels6, country="KOR")
myanmar_exports = wbdata.get_dataframe(variable_labels6, country = "MMR")

exports= wbdata.get_dataframe(variable_labels6, country= countries).squeeze()

#korea_exports.iplot(title= "Total Population of South Korea Over Time", xTitle="Year")
#myanmar_exports.iplot(title= "Total Population of Myanmar Over Time", xTitle="Year")
exports.iplot(title= "Total Population of Myanmar and South Korea Over Time", xTitle="Year")
```



In []: