用 Python 来进行多条曲线动态演示全球疫情变化

出品：Python 数据之道

作者：Lemon

01 Apr, 2020

Contents

1 升级版，用 Python 来进行多条曲线动态演示全球疫情变化

各位同学早上好，我是 Lemon 。

最近一段时间，我写了几篇用 Python 的交互式可视化工具 Plotly 来演示全球疫情情况的文章，如下：

- 用 Python 可视化神器 Plotly 动态演示全球疫情变化趋势
- 用 Plotly 动态柱状图来演示全球疫情变化趋势
- 超火动态排序疫情变化图，这次我们用 Plotly 来绘制
- 用 Python 动态曲线图来对全球疫情进行演示

如今（截至 3 月 31 日），全球确诊数量将近 80 万人，美国确诊数量已超过 16 万人，成为目前人数最多的国家，美国疫情的快速发展也给全球增添了更大的不确定性。

今天，Lemon 继续来分享用 Plotly 对疫情情况进行可视化。作为上次文章内容的升级版，本次我们用 Plotly 来实现多条曲线的动态演示。先来看最终的效果：

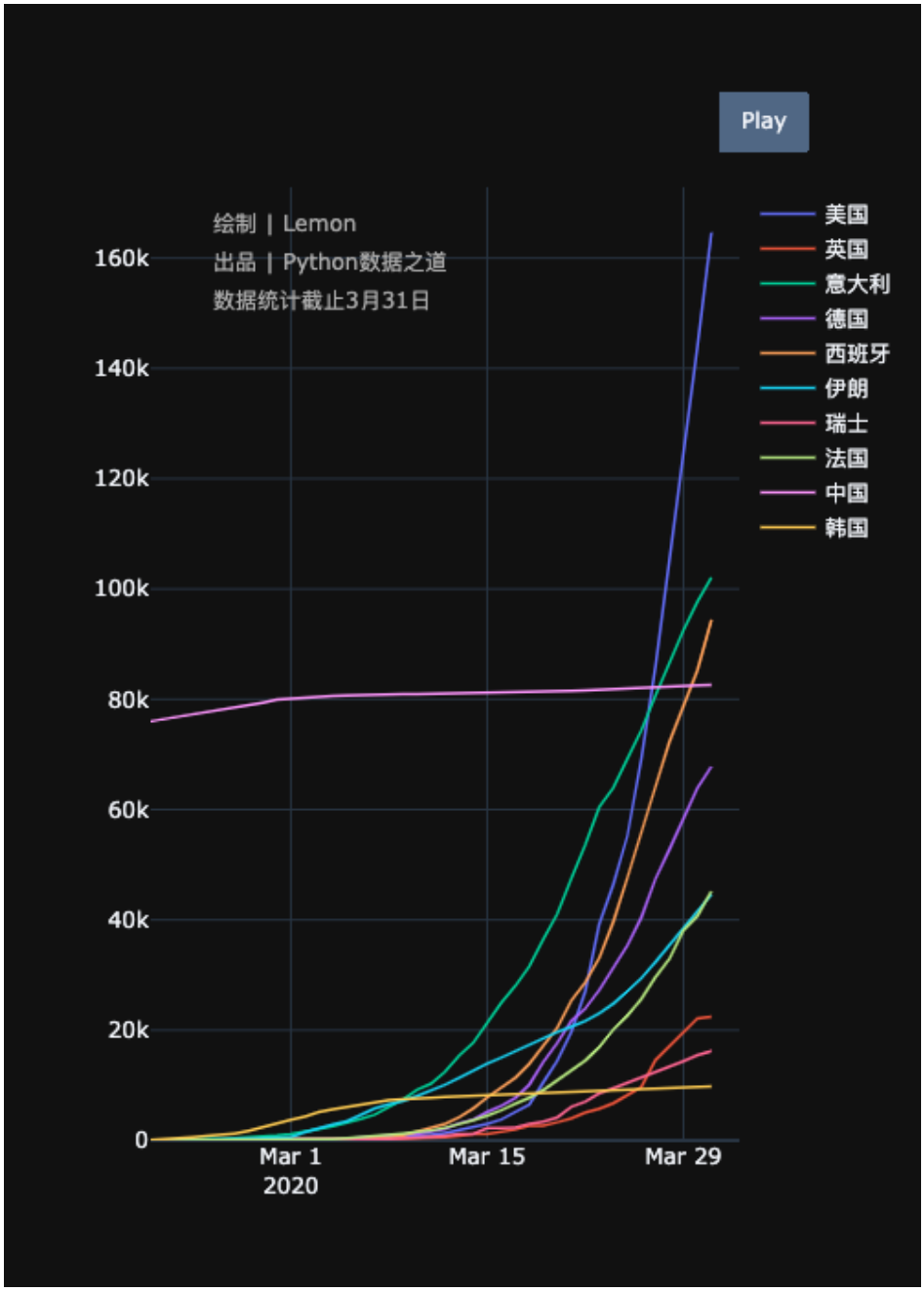

Figure 1.1: covid-19

上图的动态效果，请在公众号「Python 数据之道」2020 年 4 月 1 日发布的文章中查看。

1.1 数据来源

本次我们主要使用动态曲线来可视化分析疫情的发展情况，疫情的数据来源于开源项目 Akshare 。为了代码和演示情况能够复现，这里我提供了保存好的数据供大家练习使用，**本文的完整代码及数据文件在文末提供了获取方式。**

1.2 准备工作

照例，还是先介绍下我运行的环境。

- Mac 系统
- Anaconda（Python 3.7）
- Jupyter Notebook

我是在 Jupyter Notebook 中运行代码的，本次使用到的 Python 库包括 akshare, pandas, plotly 等，首先我们需要将这些工具进行导入。

```
import akshare as ak
import pandas as pd
import plotly
from plotly.offline import iplot, init_notebook_mode,plot
import plotly.express as px
from datetime import datetime
import numpy as np
import plotly.graph_objs as go

init_notebook_mode()
```

本次，Lemon 使用的几个 Python 库的版本如下：

```
print(f'pandas version: {pd.__version__}')
print(f'akshare version: {ak.__version__}')
print(f'plotly version: {plotly.__version__}')

# pandas version: 1.0.1
# akshare version: 0.4.27
# plotly version: 4.5.0
```

接着，我们读取已获得的数据（已保存的数据是截至 3 月 31 日）。

```
# 从 akshare 获取数据
# df_all_history = ak.epidemic_history()

# 从csv文件获取数据，这个数据文件的数据截止到3月31日
df_all_history = pd.read_csv('epidemic_all_20200331.csv',index_col=0)

df_all_history
```

获取数据后，根据本次数据分析和可视化的目的，Lemon 对数据进行了初步整理

```
# 整理数据

# 将数据复制一份
df_all = df_all_history

# 将字符串格式的日期 另保存为一列
# df_all['dates'] = df_all_history['date']

# 将字符串格式的日期转换为 日期格式
df_all['date'] = pd.to_datetime(df_all['date'])

# 将时间格式转为字符串格式的日期，以 YYYY-mm-dd 的形式保存
df_all['dates'] = df_all['date'].apply(lambda x:x.strftime('%Y-%m-%d'))

# 添加现存确诊列
df_all['current'] = df_all['confirmed'] - df_all['cured'] - df_all['dead']

df_all.fillna('', inplace=True)

print(df_all.info())

df_all
```

1.3 获取国外和国内的疫情数据

上面的数据，是全球的数据，其中也包括国内各个省市的数据。我们可以将数据进行整理，分别提取出中国和海外国家的数据。

```
# 国内总计数量
df_china_total = df_all.query("country=='中国' and province==''")
df_china_total = df_china_total.sort_values('date',ascending=False)
# df_china_total

# 国外，按国家统计
df_oversea = df_all.query("country!='中国'")
df_oversea.fillna(value="", inplace=True)
# df_oversea

# # 全球总计
df_global =  df_china_total.append(df_oversea)
df_global = df_global.sort_values(['country','date'])
df_global
```

获取全球最近日期的数据：

```
# 按日期进行排序
df_global_date = df_global.sort_values('date',ascending=False)

```

```
# 获取最新的日期
latest_day = df_global_date['dates'].values[0]

# query 函数中，变量需要加 @ ，
df_global_latest = df_global_date.query('dates==@latest_day')

df_global_latest = df_global_latest.sort_values('confirmed',ascending=False)

df_global_latest.head(10)
```

进一步，我们可以梳理出海外国家的总计概况，同时，将国内的数据也提取出相应的字段。

```
df_oversea_total = df_oversea.groupby(['date','dates'])['confirmed','cured','
    dead','current'].sum()
df_oversea_total.reset_index(level=1,inplace=True)
df_oversea_total['district'] = 'oversea'
# df_oversea_total

df_china_sum = df_china_total[['date','dates','confirmed','cured','dead','
    current']]
df_china_sum.set_index('date',inplace=True)
df_china_sum['district'] = 'China'
# df_china_sum

df_total = df_oversea_total.append(df_china_sum)
df_total.sort_index(ascending=True,inplace=True)
df_total
```

由于国外从 1 月 16 日起，才开始有统计数据，因此我们的可视化从这个日期开始。

```
# 国外从 1月16日起，才开始有统计数据
df_total_analysis = df_total['20200116':]
df_total_analysis
```

date	dates	confirmed	cured	dead	current	district
2020-01-16	2020-01-16	45	15	2	28	China
2020-01-16	2020-01-16	2	0	0	2	oversea
2020-01-17	2020-01-17	62	19	2	41	China
2020-01-18	2020-01-18	198	24	3	171	China
2020-01-19	2020-01-19	275	25	4	246	China
...	...	...	...	...	...	...

Figure 1.2: covid-19

1.4 疫情可视化

我们先来用 plotly express 对海外国家和国内情况的发展来做一个总的概览。

```
fig_total = px.line(df_total_analysis, x='dates', y='confirmed', line_group='
    district',
                    color='district', color_discrete_sequence=px.colors.
                        qualitative.D3,
                    hover_name='district',template='plotly_white',
                   width=500,height=600,
                   title=dict(text='Covid-19-trend',
                              font=dict(size=16,color='#0071c1'),
                             x=0.5)
                    )

fig_total.show()
```

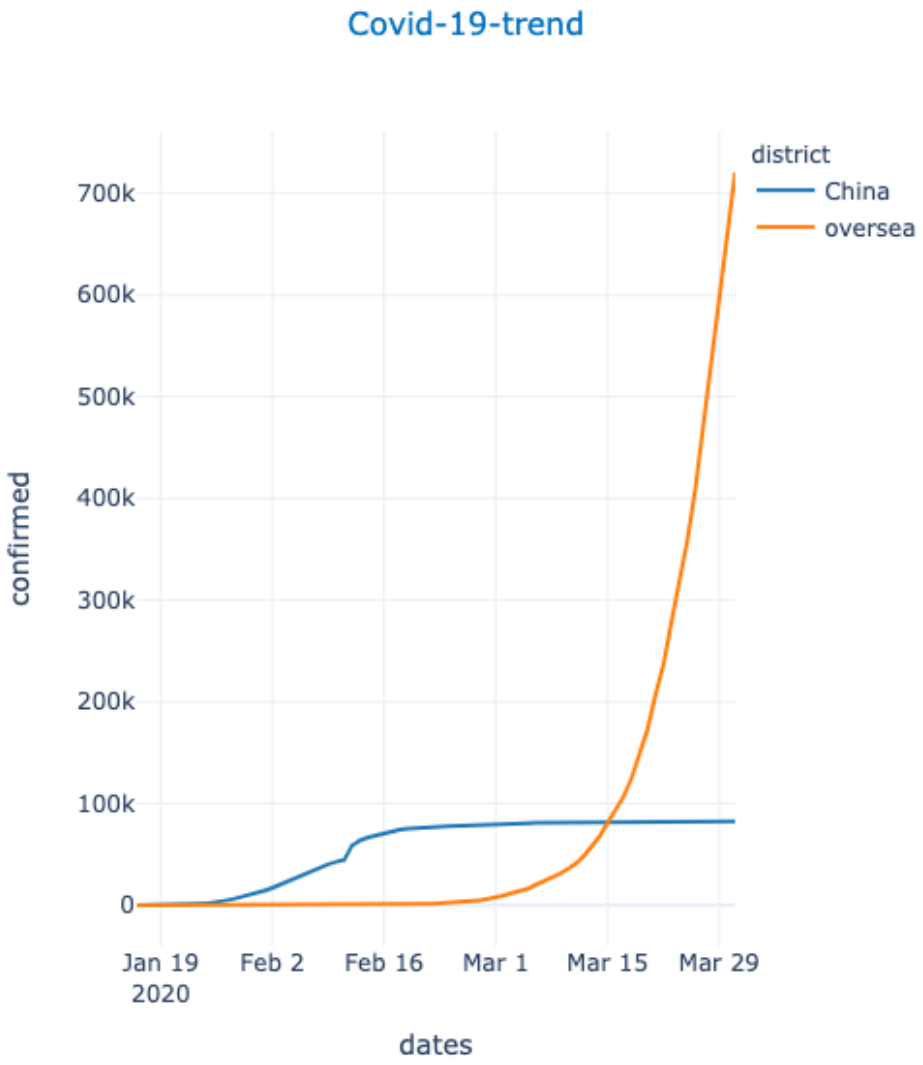

Figure 1.3: covid-19

从上面的趋势来看，国内基本趋稳了，而海外国家作为总体来看，还在快速发展。

> 这里仅仅是做简单的可视化，至于将中国和海外国家总体做对比分析，是否合理，这里只是个示例，不做进一步阐述。

上图中的曲线，并不能动态的演示变化过程，在 `px.line` 中，Lemon 也研究了下，暂时没有实现将曲线进行动态可视化的功能。

下面，Lemon 将用 Plotly 原生的功能来实现多条曲线的动态可视化。

1.5 动态曲线演示疫情情况

在 Plotly 中，将曲线（Line）进行动态演示，需要通过几个步骤来实现

1. 对初始状态进行可视化，每条曲线将起始的两个点绘制成曲线；
2. 通过构造字典的形式，在 `frames` 中实现曲线的动态变化；
3. 添加演示按钮，

多条曲线的动态可视化，关键是如何便捷的添加多条曲线，尤其考虑有时曲线数量较多，有时数量经常发生变化。

针对上述情况，我们需要将程序设计的相对灵活些，这里介绍的方法，主要是将 `dataframe` 以 `value` 的形式装入到字典里。

至于曲线的多少，可以将元素包含在列表 `list` 中，这样就可以灵活的实现曲线数量的变化。

多条曲线动态演示的代码如下：

```
# 日期
m = df_global_latest.iloc[0]['date'].month
d = df_global_latest.iloc[0]['date'].day
text_today = f'数据统计截止{m}月{d}日'

df_compare = df_global.set_index('date')
df_compare = df_compare['20200220':]

list_countries = [”美国”,”英国”,”意大利”,”德国”,”西班牙”,”伊朗”,”瑞士”,”法国”,”中国”,”韩国”]
n_countries = len(list_countries)
n_list = [i for i in range(n_countries)] # 曲线数量编号
# print(n_list)

n_frame={}

for country_i in list_countries:
    df_i = df_compare.query('country==@country_i') # @ 代表变量
    df_i = df_i.sort_index(ascending=True)
    n_frame[country_i] = df_i

# 计算 最大的确诊人数
# 初始值为0
y_max = 0
for country_i in list_countries:
    y_max_i = n_frame[country_i]['confirmed'].max()
    y_max = max(y_max,y_max_i)

df_01 = n_frame[list_countries[0]] # 第一个国家
# print(df_01)

traces = [go.Scatter(
                    x = value.index[:2],   # value 是 国家的数据，dataframe
                    y = value['confirmed'][:2],
```

```
                      mode='lines', # markers+lines
                      name=key,  # key 是国家名称
                      line=dict(width=1.5)
                      ) for key,value in n_frame.items()
          ]

frames = [dict(data= [dict(type='scatter',
                           x=value.index[:k+1],
                           y=value['confirmed'][:k+1]) for key,value in n_frame
                               .items()
                     ],
               traces= n_list,
              )for k  in  range(1, len(n_frame[list_countries[0]]))]

layout = go.Layout(width=500,
                   height=700,
                   showlegend=True,
                   template='plotly_dark',
                   hovermode='closest',
                   updatemenus=[dict(type='buttons', showactive=False,
                               y=1.10,
                               x=1.15,
                               xanchor='right',
                               yanchor='top',
                               pad=dict(t=0, r=10),
                               buttons=[dict(label='Play',
                                             method='animate',
                                             args=[None,
                                                   dict(frame=dict(duration
                                                       =100,
                                                                   redraw=
                                                                       False),
                                                        transition=dict(
                                                            duration=1),
                                                        fromcurrent=True,
                                                        mode='immediate')])])
                                                            ],
                   )

layout.update(xaxis =dict(range=[df_01.index[0],
                                 df_01.index[len(df_01)-1]+pd.Timedelta(days=2)
                                ],
                          autorange=False),
              yaxis =dict(range=[0, y_max*1.05], autorange=False))
fig = go.Figure(data=traces, frames=frames, layout=layout)

# Pycharm，VS Code， Spider 等模式下
# Jupyter Notebook 下也可以用
plot(fig,filename='covid-19-multiple-lines.html')

# Jupyter Notebook 中
```

```
80  # fig.show()
```

上面代码中，`go.Layout` 中的 `duration` 可以来控制按钮点击后变化的速度。

运行上述代码后，得到的效果如下：

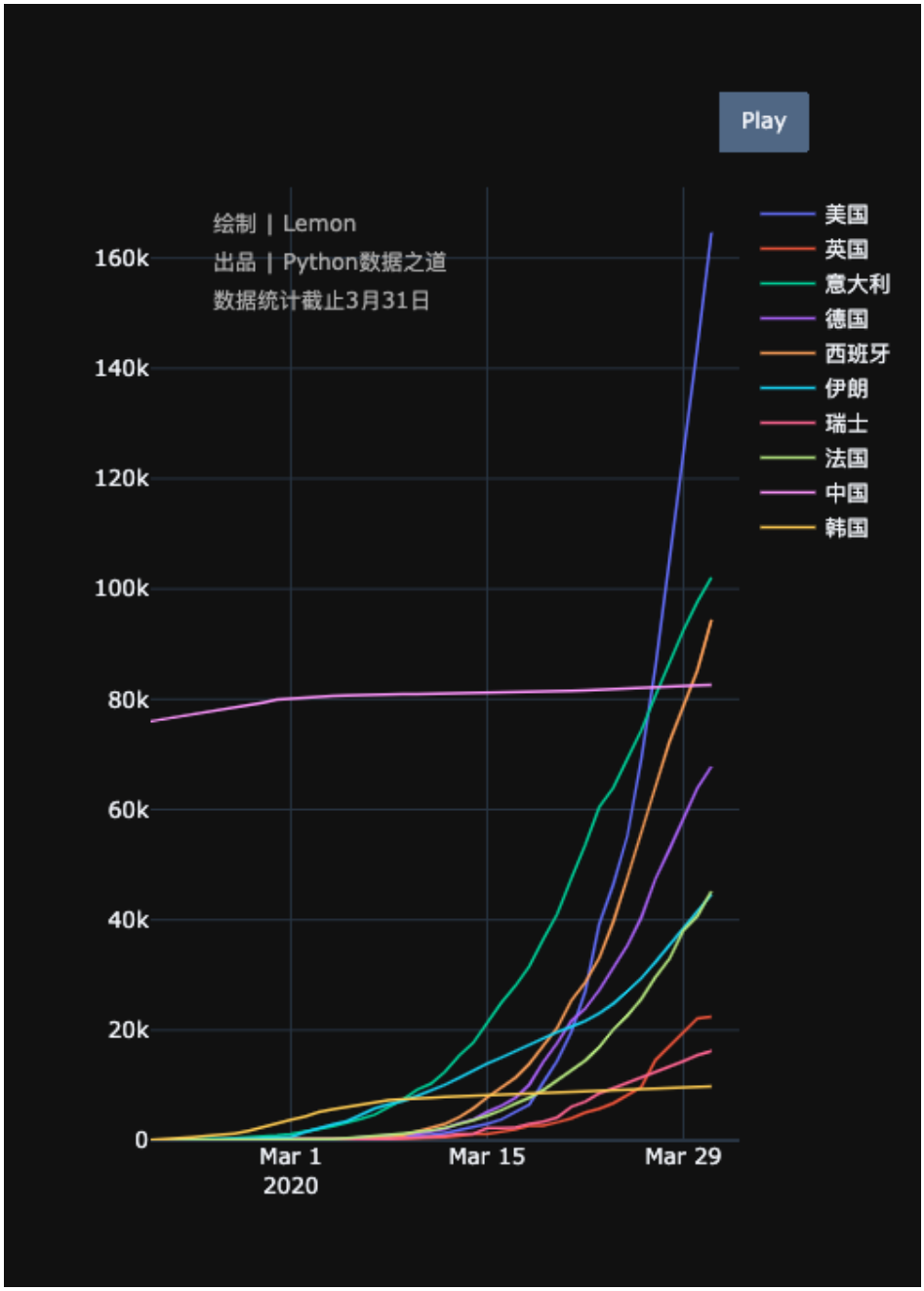

Figure 1.4: covid-19

曲线可以动起来了，是不是很棒啊。

为方便大家进行操作，Lemon 给大家提供了本文的 PDF 版内容（含完整的代码）以及数据文件，可以在公众号**「Python 数据之道」**后台回复数字「661」进行获取。

2 关于作者

2.1 我是谁

各位读者好，我是 Lemon ，公众号「Python 数据之道」号主。

公众号「Python 数据之道」秉承“让数据更有价值”的理念，主要分享数据相关的内容，包括数据分析，挖掘，可视化，机器学习，深度学习等，希望能给大家分享有价值的内容。

若对我写的内容有兴趣，欢迎大家通过以下途径来关注。

2.1.1 微信公众号

Figure 2.1: Python 数据之道

「Python 数据之道」是我分享关于 Python 及数据分析相关内容的主阵地。

此外，为防意外，Lemon 还有一个小号 **「柠檬数据」（ID：LemonDataLab）**，会不定期分享关于数据的故事，也墙裂建议大家一并关注，以防突然某天失联了 ~~

Figure 2.2: 柠檬数据

2.1.2 个人网站

网址：http://liyangbit.com

Lemon 的个人网站中，包含更多的文章，并且在不断的进行更新。目前，网站中涉及了 Python 相关一系列内容，包括 Python 基础、Python 数据科学、项目实战等内容，欢迎访问。

2.2 部分优质内容

2.2.1 《Python 知识手册》

「Python 数据之道」整理并出品了《Python 知识手册》，大家可以在公众号「Python 数据之道」后台回复数字**「600」**来获取高清 PDF 版。

2.2.2 推荐文章

- 用 Python 可视化神器 Plotly 动态演示全球疫情变化趋势
- 用 Plotly 动态柱状图来演示全球疫情变化趋势
- 超火动态排序疫情变化图，这次我们用 Plotly 来绘制
- 用 Python 动态曲线图来对全球疫情进行演示
- 深度好文｜ Matplotlib 可视化最有价值的 50 个图表（附完整 Python 源代码）
- 用 Python 读取巴菲特近期持仓数据

- 推荐一个牛逼的生物信息 Python 库 - Dash Bio
- 轻松用 Seaborn 进行数据可视化
- 用 Python 快速分析和预测股票价格
- 干货推荐: 轻松玩转 Bokeh 可视化（项目实战经验分享）
- 巧用 Matplotlib 动画, 让你的 Python 可视化大放异彩