

# Government response to COVID-19 pandemic:

## KPI and correlation analysis

Daria Siniatulova

# Abstract

In this project I create the tool to conduct automated monitoring of both COVID statistics and government measures to understand situation and government reaction in different countries on the threats of 2020.

I explore the correlation between COVID spread statistics and government response to it.

I used 3 datasets in order to gather pandemic statistics (JHU CSSE COVID-19 Dataset), structured this data by country, region, income level (WDI Dataset) and date since January 1<sup>st</sup> 2020 till today, and then combined resulting dataset with government response index (Oxford Covid-19 Government Response Tracker) to calculate correlations and get insights on different countries reaction to this threat.

I mostly use pandas built-in methods to work with tabular data, along with several visualization techniques to present the research results.

# Datasets

## **Dataset #1:**

Name: JHU CSSE COVID-19 Dataset.

Source: <https://github.com/CSSEGISandData/>

## **Dataset #2:**

Name: Oxford Covid-19 Government Response Tracker

Source: <https://github.com/OxCGRT/>

## **Dataset #3 (supplementary):**

Name: World Development Indicators Dataset (WDI)

Source: <http://api.worldbank.org/v2/country/all/indicator/>

# Motivation

I decided to explore what new indicators could reveal summary on governments respond to COVID-19 pandemic. Being a municipal servant myself I would like to have the tool to monitor this activity to understand reaction in different countries on the threats of 2020. My goal was to write the code that would let me update the information on the fly daily to prepare summary of situation monitoring automatically.

I consider government policies indicators and pandemic statistics to be one of the most interesting datasets to research in mid 2020. I specifically use visualization technics that are different from JHU approach to their famous [dashboard](#) and suggest supplementary visual solutions.

I believe the ideas of this project could be valuable for many people like me who is concerned with pandemic issues or try to compare different governments approaches to this problem.

# Data preparation and cleaning

I described my approach to cleaning and preparation of the data in special paragraphs of each chapter of this project. Particular steps are always related to the structure and completeness degree of the data, as well as my goals in terms of using data: for calculating, for creating animated map, for plotting interactive line graphs, bubble graphs or sunbursts.

The problems I have encountered with the dataset were the following: lack of values for particular countries or dates, excessive values that can not be plotted on map or can not be visible there even if plotted (for example, COVID cases on cruise ships "Diamond Princess and "MS Zaandam", or in Vatican), wrong ISO-3 codes for the countries, outdated choropleth maps (for example, Kosovo could not be plotted, even though the shape is there, and I cleaned the dataset, and I fixed the ISO-3 codes).

# Research Questions

I have formulated 3 research questions to answer with this project:

- is there a significant correlation between cumulative cases of COVID-19 and cumulative deaths related to it?
- is there a significant correlation between maximum government response index and government healthcare expenditure in previous years?
- is there strong positive correlation between confirmed COVID-19 cases and government response index over time?

# Hypothesis to check

- yes, there is significant positive correlation between cases of COVID-19 and deaths;
- no, strict response to COVID-19 in 2020 and better funding healthcare systems in 2019 are not significantly correlated;
- yes, correlation between confirmed cases and government response over time is strong.

Note: I explore indicators using Russia, Canada, USA, China and Brazil to describe and compare trends. I also plotted world's average data to provide additional context to comparisons.

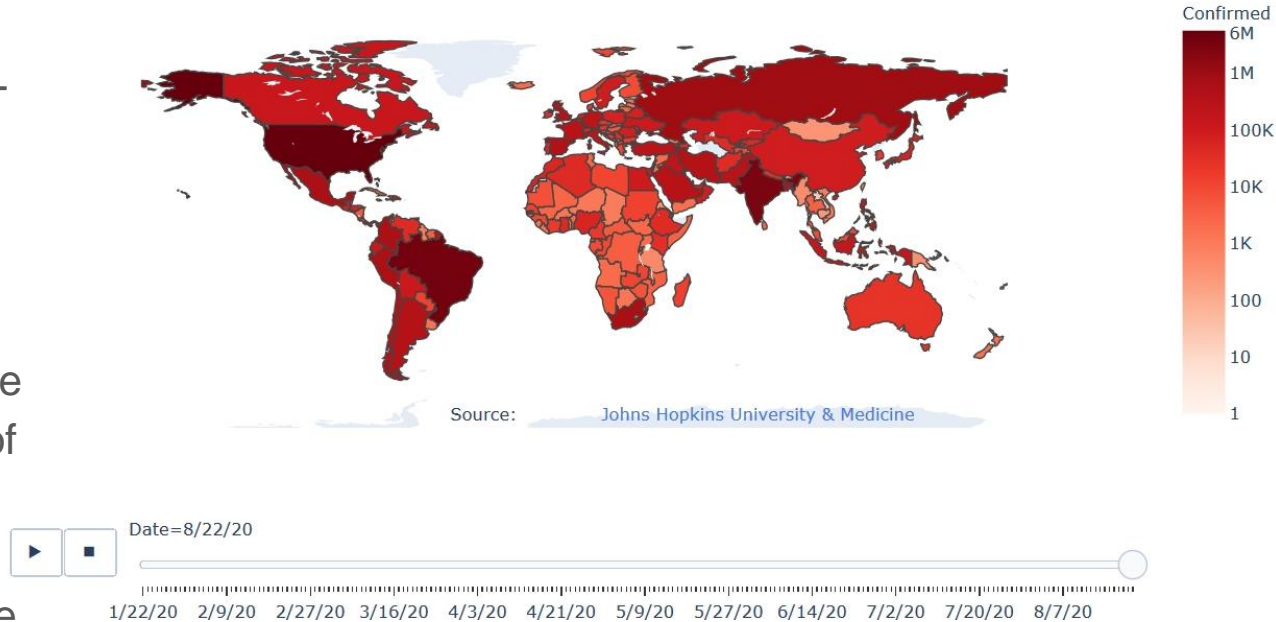
# Methods

- I mostly used pandas to work with tabular data. I used plotly express, plotly graphic objects, choropleth and matplotlib to visualize the results, and then used the visualizations as tools to guide my data analysis.
- I read scientific research by the largest universities on the topic of interest.
- I used pandas built-in methods to access the datasets, connecting my code directly to the repositories (Datasets 1 and 2) or API (Dataset 3).
- I implemented 3 different visualization techniques to present the research results.
- I used several comparison methods: correlation coefficients, scatter plots, merging datasets, and others using examples of Russia, Canada, USA, China and Brazil.
- To describe and compare trends I also plotted world and world's average to provide additional context.
- I use interactive plots and animated world maps where everyone can find their countries and explore the depths of the data.



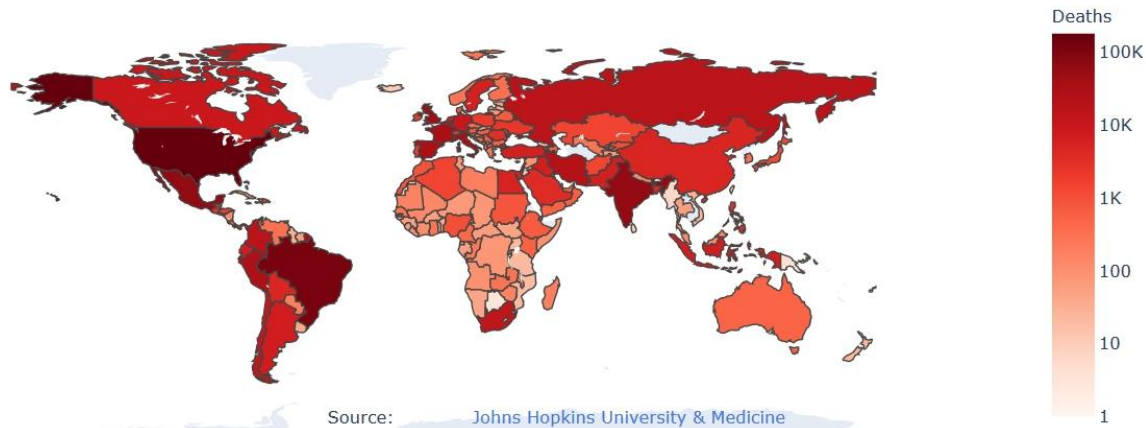
# Confirmed cases

As of August 20, 2020  
maximum number of cases -  
5.6 M - were confirmed in  
USA. There were performed  
about 69.6 M tests there, so  
testing has covered every  
210 out of 1000 people in the  
country. This could be one of  
the reasons of the record  
high number of confirmed  
cases in comparison with the  
rest of the world.



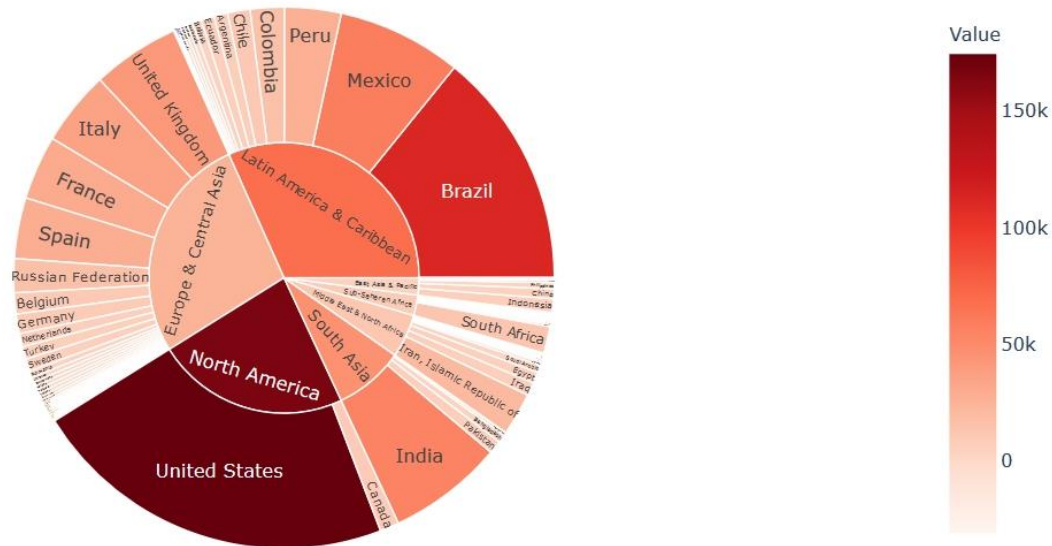
# Deaths

As of August 2020 maximum number of lethal end cases were registered in USA. Mongolia and Turkmenistan have not reported COVID-related deaths.



# Deaths: Structure by regions

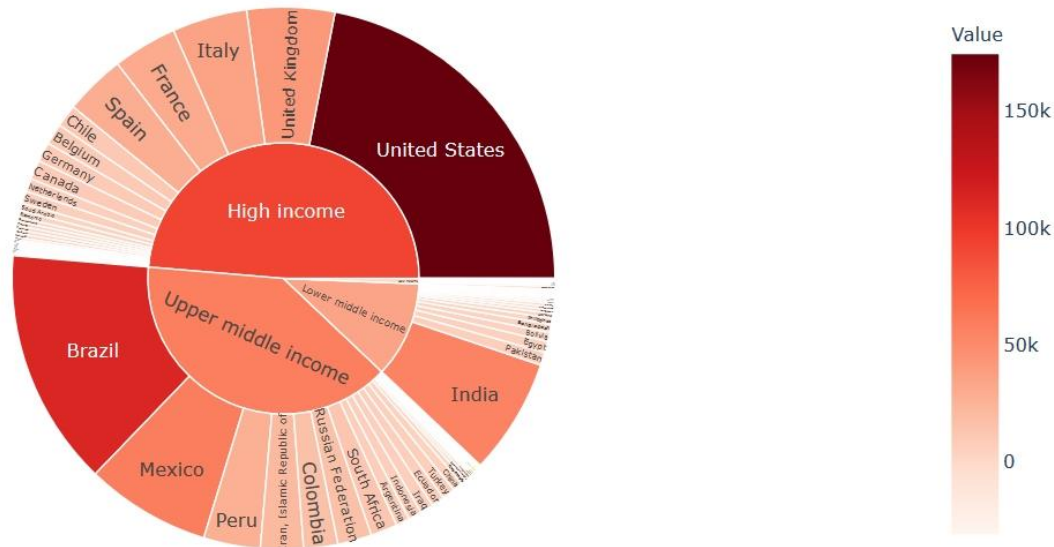
United States in North American Region, Brazil and Mexico in Latin America & Caribbean region, India in South Asia and UK, Italy, France and Spain in Europe region are the countries most affected by COVID-related deaths.



# Deaths: Structure by income level

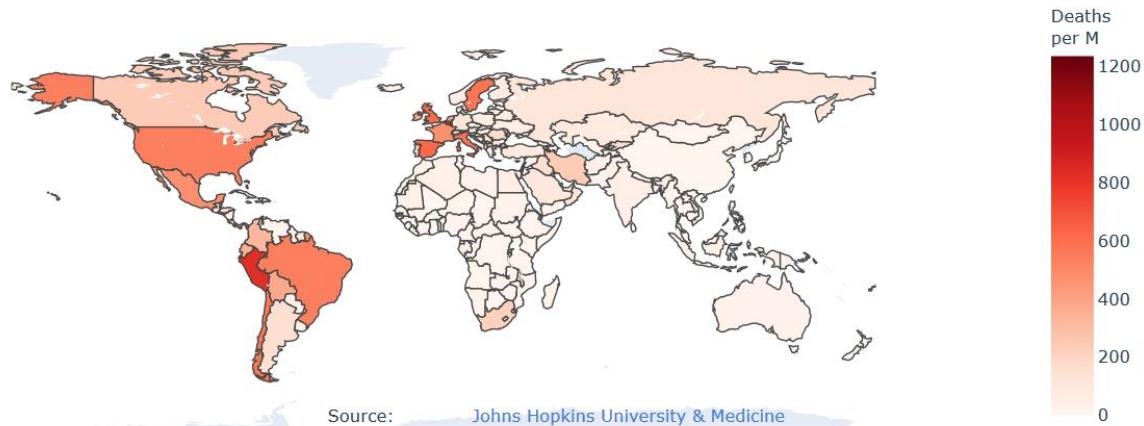
“High income” countries and “Upper medium income” countries cover 41% and 39% of total COVID-19 deaths respectively.

“Low” and “Lower medium income” countries cover other 20% cases.



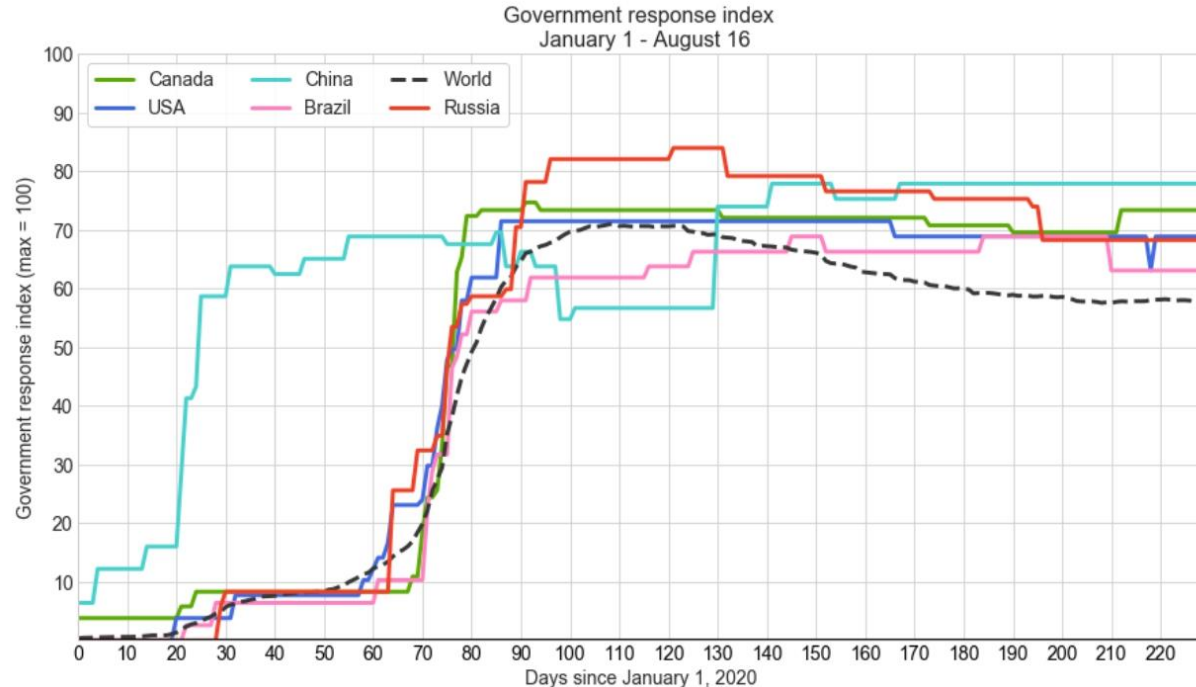
# Deaths per million population

As of August 21, 2020  
maximum number of lethal  
end cases per million  
population was registered in  
Belgium (873.4) and Peru  
(838.8); losses are high in  
UK (622.7), Spain (616.7)  
and Italy (586.1).



# Government response index

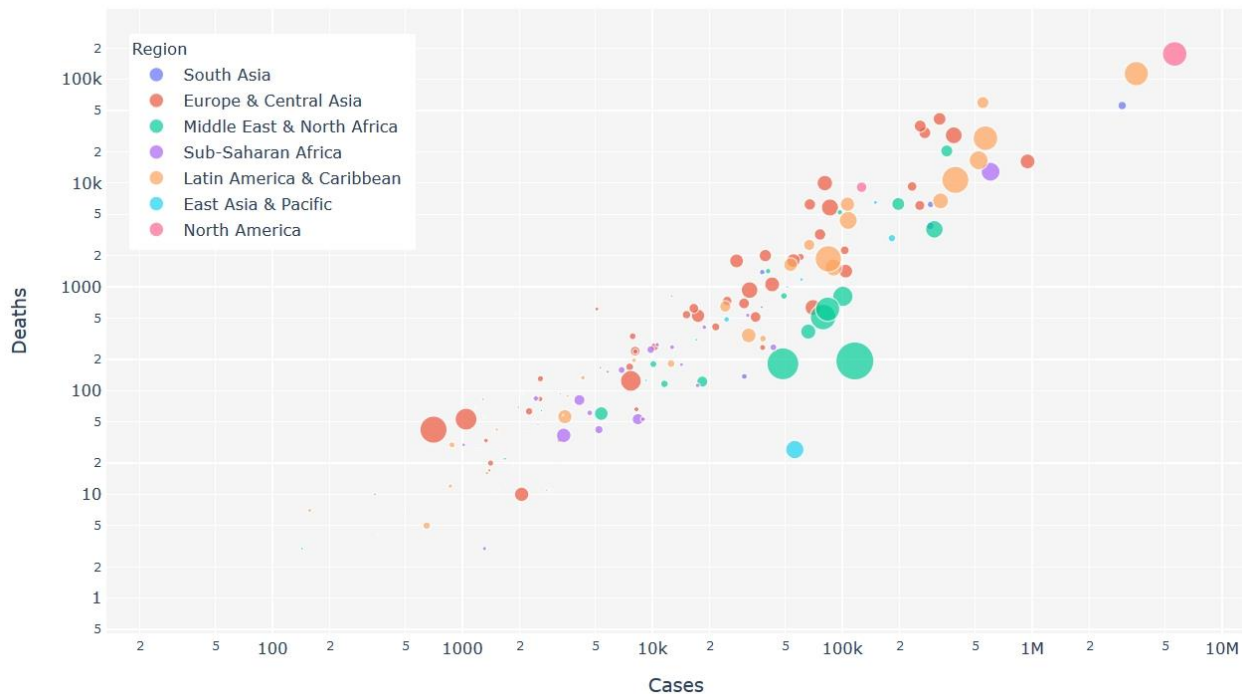
Government of China responded first, dealing with Wuhan outbreak in December 2019. It took actions 50-70 days ahead of the rest of the world. Other countries preferred not to make costly decisions and were not enforcing strict measures till mid-March.



# Confirmed cases and deaths

Number of confirmed cases and COVID-related deaths over time have strong positive correlation.

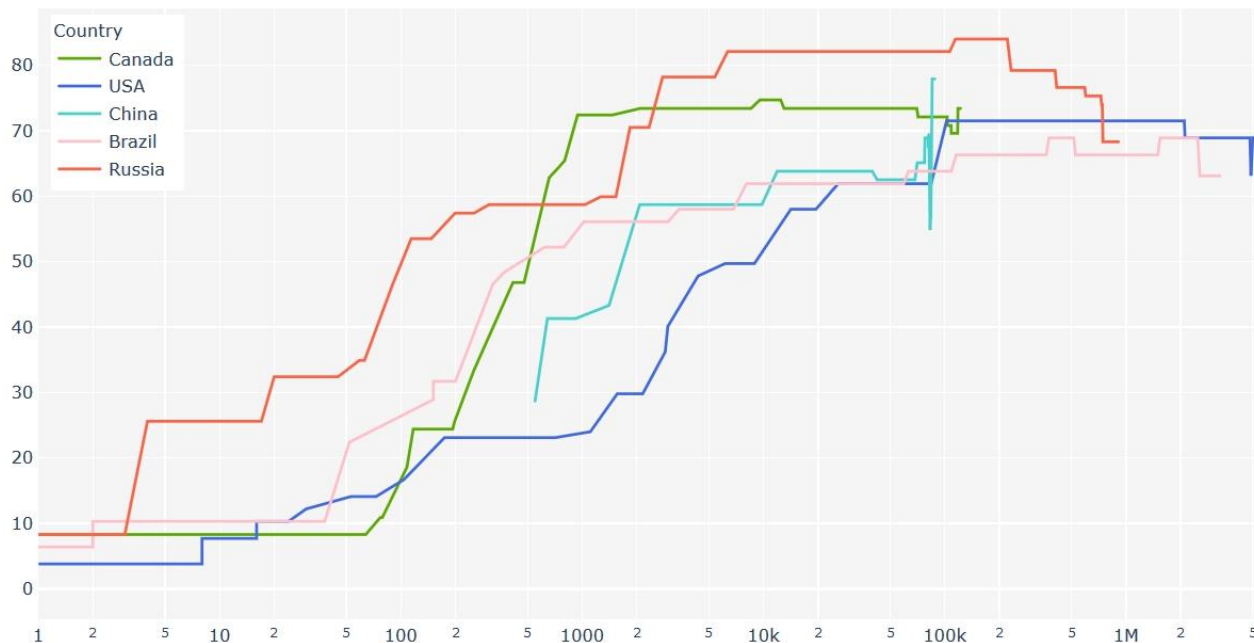
That is why the governments need to react timely and must be proactive to prevent number of new cases from rising. This issue is crucial and time-sensitive.



# Confirmed cases and government response index

Number of confirmed cases (x axis) and government response index (x axis, max = 100) over time have strong positive correlation: correlation coefficient is equal 0.94.

The plot allows to compare different countries approach to response measures and compare the effects they may have on a daily basis.





# Findings 1 – Cases and Deaths

- Yes, there is significant positive correlation between cumulative number of cases and deaths from COVID-19. That is why it was (and still is) so important for the governments to act strict and timely to prevent further virus spread and it's damage to society and economy.

# Findings 2 – Response and Health expenditures

- No, there is no correlation between maximum government response index and government healthcare expenditure in previous years. The governments that demonstrate higher response index to COVID-19 in 2020 were not necessarily better in funding their healthcare systems in previous years.

## Findings 3 – Cases and Response

Yes, correlation between confirmed cases over time and government response index over time is very strong (coefficient is equal to 0.94). At the same time, correlation between maximum cumulative number of cases and maximum government response is very weak (0.26), and correlation between deaths and maximum government response is even weaker (0.19).

Time factor is crucial to government response in order for measures to have desired effect and prevent virus from rapid spreading and it's damage to society and economy.

# Limitations

The Dataset 2 has inherent limitations. As the indicators are work in progress, and research team has to monitor 17 sub-indicators to calculate another 4 aggregated ones for 200+ countries to publish daily, their data has the lag for 1-2 weeks back. This factor should be taken into account at all times when comparing with the daily updated pandemic statistics.

In order to check the results of this research the indicators should be monitored longer, that current available period (January-July or January-August).

Some of the described trends and findings are not fully confirmed for the countries of the lower middle income and low income due to limitations of the dataset and COVID-19 stage in these countries.

# Conclusions

I have confirmed all 3 hypothesis: I found out that positive correlation between cases of COVID-19 and deaths from it is very strong, as well as between confirmed cases and government response. Government response index and government healthcare expenditure in previous years are not correlated. These findings are relevant for different regions, but their relevancy for the lower middle income countries and low income countries is not fully confirmed yet due to limitations of the dataset.

It is too early to make final conclusions about COVID mortality for different regions and income levels as disease statistics database is only growing and information available now is incomplete. The indicators that we could use to describe the situation unfolding now are work in progress themselves. I would do my best to add my share to the knowledge and ideas database for this important work.

# Acknowledgements

1. Johns Hopkins University & Medicine provided the dataset “JHU CSSE COVID-19”
2. Blavatnik School of Government, University of Oxford provided the dataset “Oxford Covid-19 Government Response Tracker”.
3. World Bank provided the dataset “World Development Indicators”

# References

I have used a research paper by Blavatnik School of Government on their project “Oxford Covid-19 Government Response Tracker”:

“Variation in government responses to COVID-19”, May 2020 - Blavatnik School of Government, University of Oxford, by team of authors:

Thomas Hale, Noam Angrist, Beatriz Kira,  
Anna Petherick, Toby Phillips, Samuel Webster

Source: [www.bsg.ox.ac.uk](http://www.bsg.ox.ac.uk)

Annex:

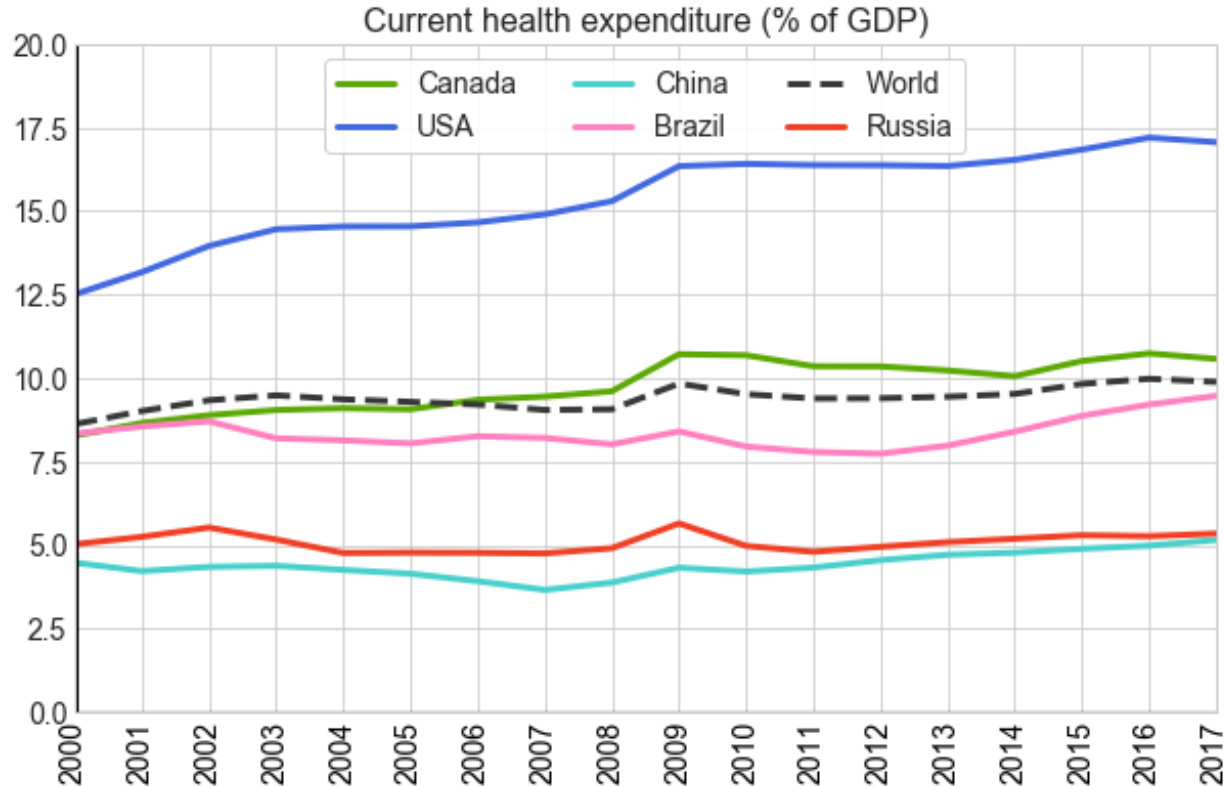
Government health expenditure trends and maps

Additional slides from Week 6: Mini-Project with  
World Development Indicators Dataset

# Health expenditure trends (% GDP)

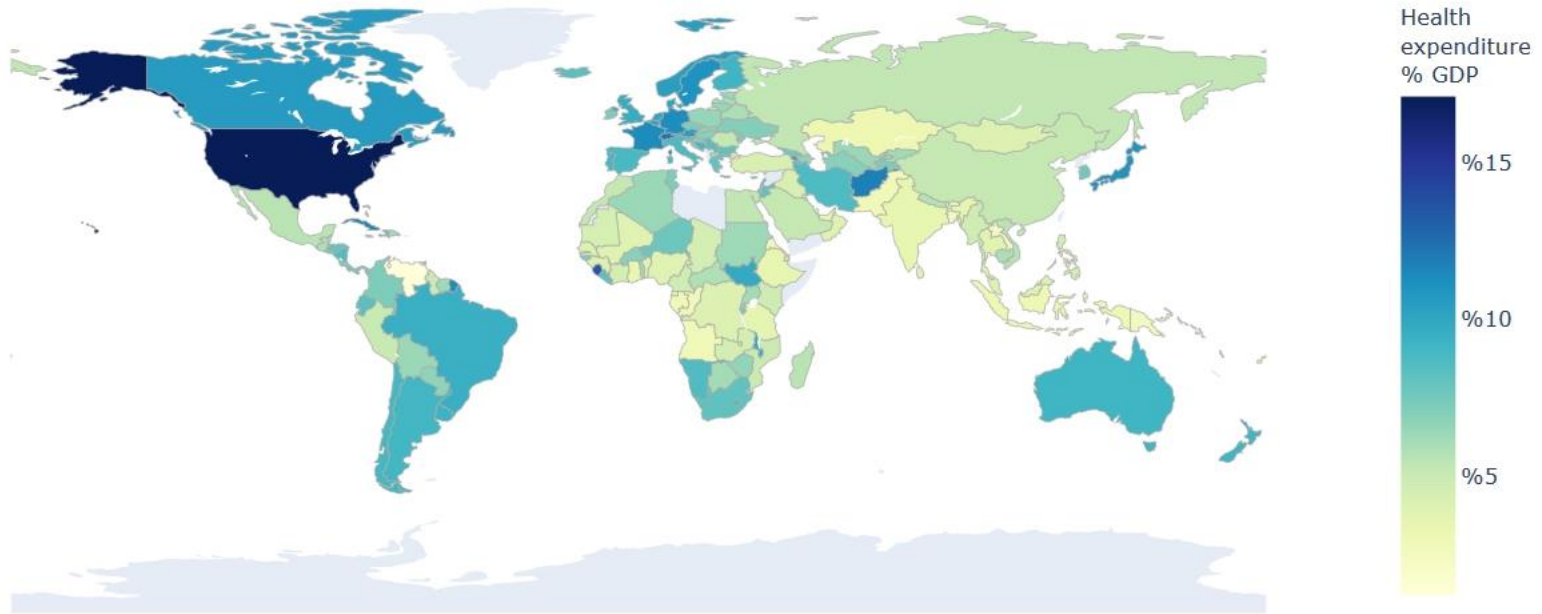
Canada increased health expenditure from 8.3 to 10.7% of GDP, while Russia - from 4.7 to 5.6%.

Russia's expenditure on healthcare system are twice lower than world's average, although its GDP per capita is greater than average since 2003.





# Health expenditure map (% GDP), 2017

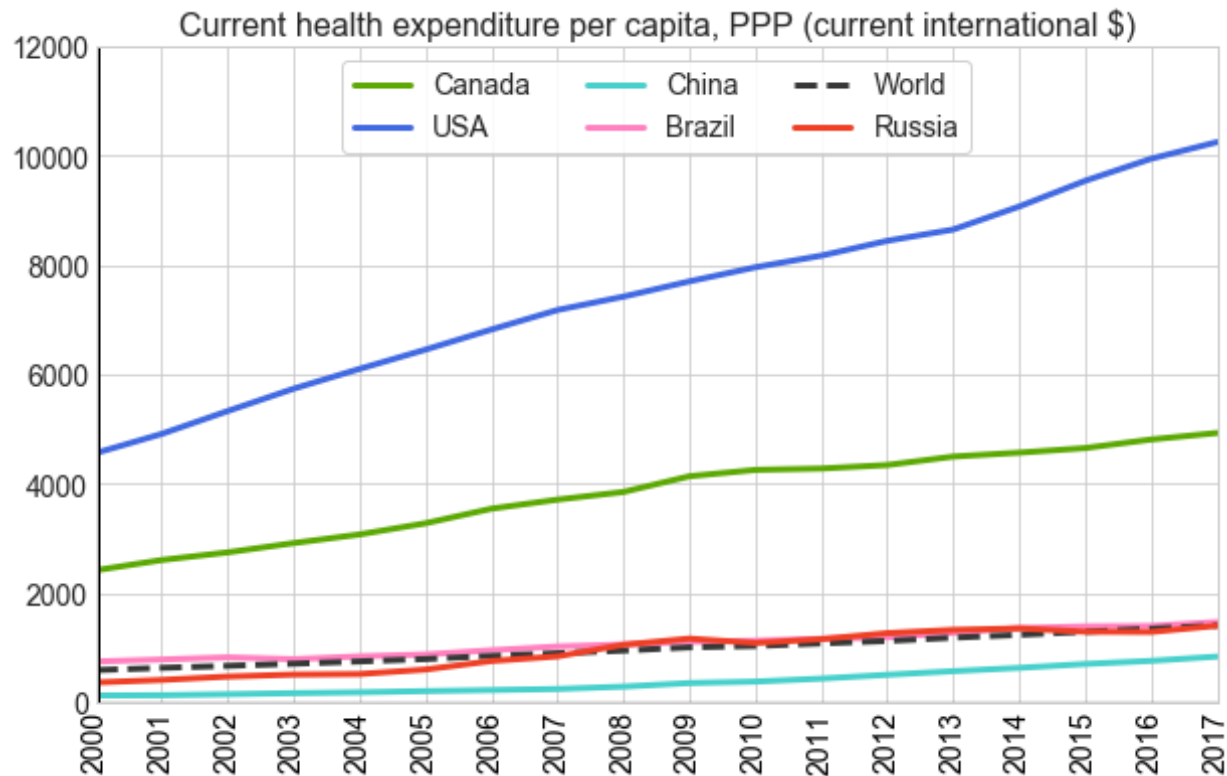


# Health expenditure trends (int. \$ by PPP)

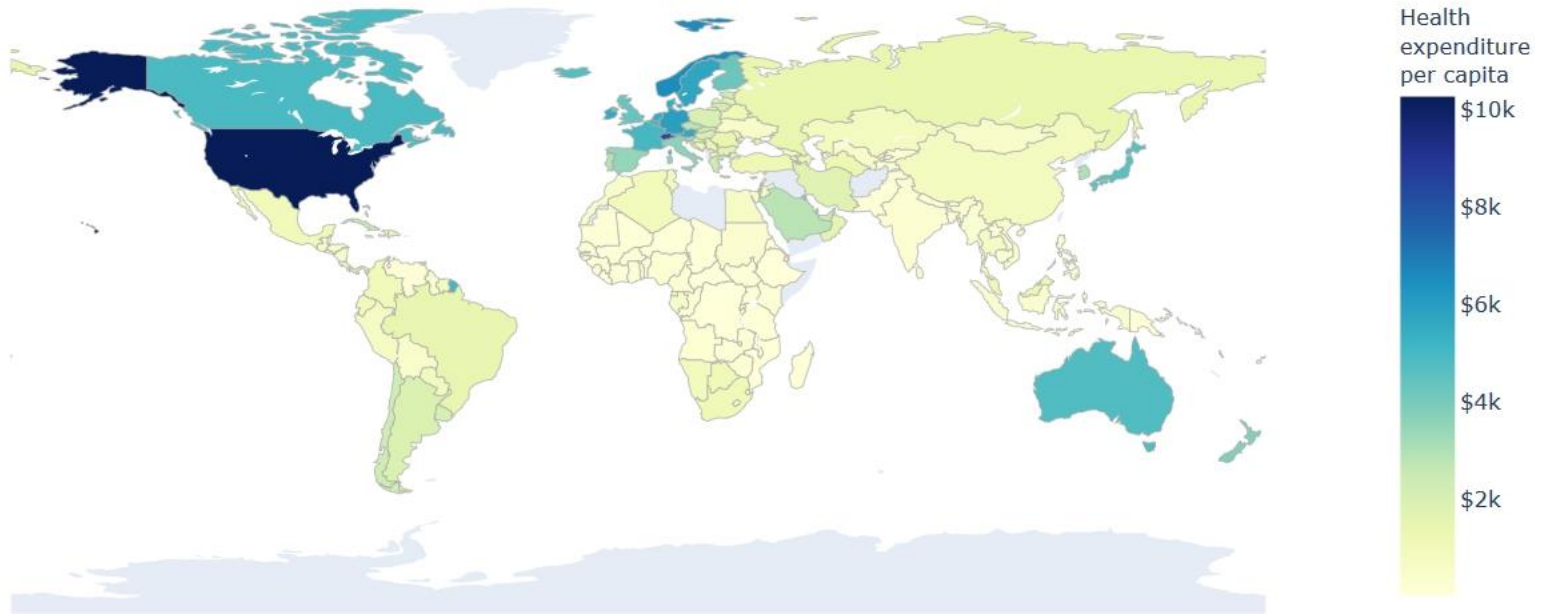
Canada increased health expenditure from 2418 to 4929 int. \$, and Russia - from 368 to 1404 int. \$.

In Canada health expenditure were 3.5 times greater than world's average in 2017.

In Russia they were lower than world's average.



# Health expenditure map (int. \$), 2017

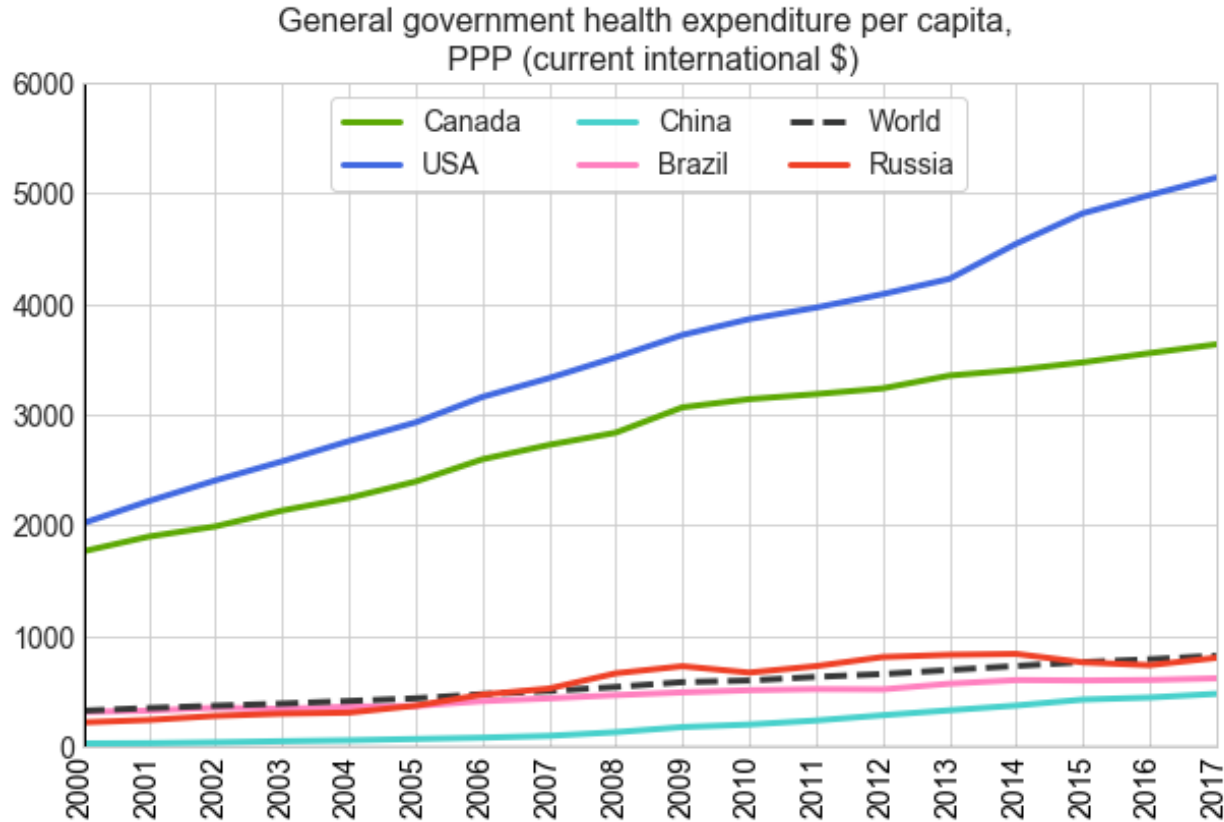


# Government health expenditure trends

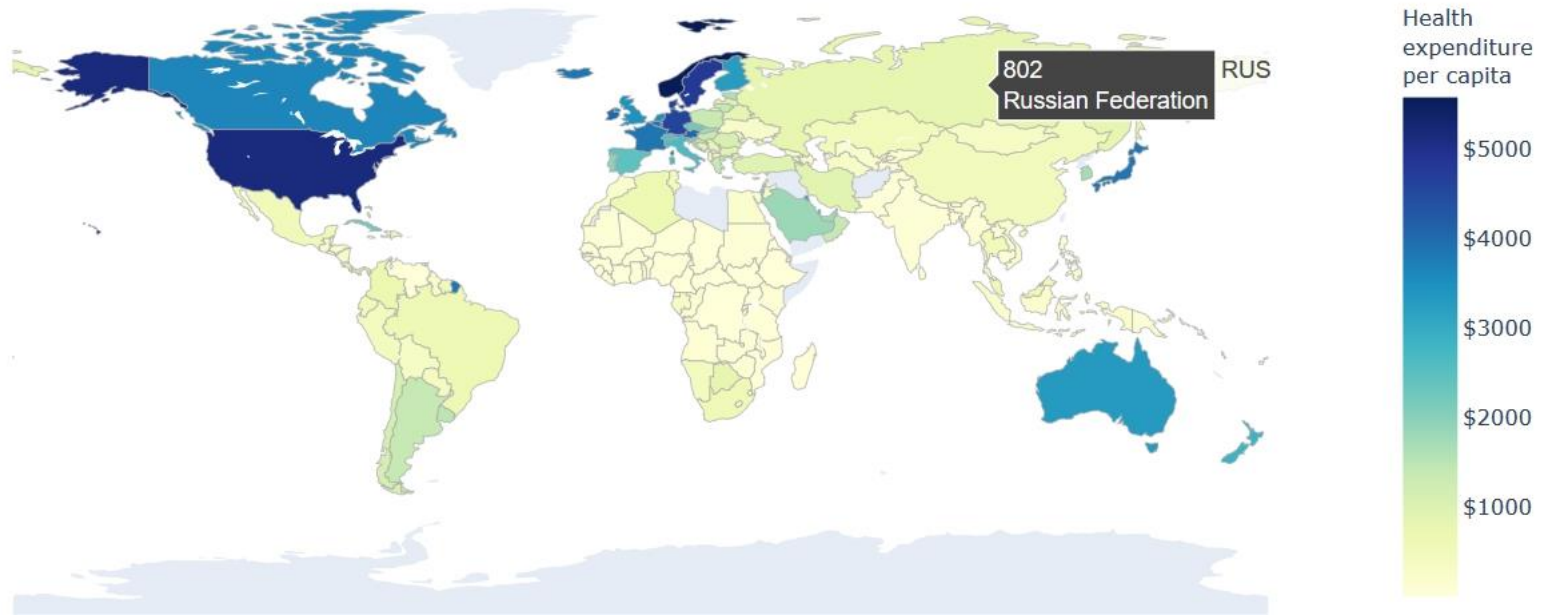
In Canada government health expenditure were 3633 int. \$, and Russia - from 802 int. \$ in 2017.

In Canada government health expenditure were 4.4 times greater than average in 2017.

In Russia they were lower than world's average.



# Government health expenditure map (int. \$), 2017



## Week 9. Final project: COVID-19 and government response

I decided to conduct my Week 9 research project on exploring COVID statistics and its correlation with new indicators of government response or measures taken against COVID virus spreading.

### Research questions

- is there a significant correlation between cumulative cases and deaths from COVID-19?
- is there a significant correlation between government response index and government health expenditure in previous years?
- is there strong positive correlation between confirmed COVID-19 cases and government response index over time?

### Hypothesis to check

- yes, there is significant positive correlation between deaths from COVID-19 and maximum government response;
- no, strict response to COVID-19 in 2020 and better funding healthcare systems in previous years are not significantly correlated.
- yes, correlation between confirmed cases and government response over time is strong.

### Methods

To explore COVID statistics I use COVID-19 Data Repository by the Center for Systems Science and Engineering at Johns Hopkins University, or JHU CSSE COVID-19 Dataset (here - Dataset 1) accessing it via API.

To explore new indicators of government response (measures taken against COVID virus spreading) I use Oxford Covid-19 Government Response Tracker (here - Dataset 2).

Part I of this project presented as chapter "Dataset 1: Johns Hopkins University & Medicine (JHU)" with paragraphs 1 to 4, where I add some improvements to it using Dataset 3 in order to have new structures in the dataset and new insights in research.

Part II is presented as "Dataset 2: Oxford Covid-19 Government Response Tracker" and paragraph 5, and it shows the aggregated dataset and correlation analysis.

## Dataset 1: Johns Hopkins University & Medicine (JHU)

Timeseries from January 22, 2020 to August 20, 2020 are available for downloading: [https://github.com/CSSEGISandData/COVID-19/tree/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series](https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_time_series) ([https://github.com/CSSEGISandData/COVID-19/tree/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series](https://github.com/CSSEGISandData/COVID-19/tree/master/csse_covid_19_data/csse_covid_19_time_series))

Raw data links to cumulative data:

Confirmed cases: [https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series/time\\_series\\_covid19\\_confirmed\\_global.csv](https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv) ([https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series/time\\_series\\_covid19\\_confirmed\\_global.csv](https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_confirmed_global.csv))

Deaths: [https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series/time\\_series\\_covid19\\_deaths\\_global.csv](https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_deaths_global.csv) ([https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series/time\\_series\\_covid19\\_deaths\\_global.csv](https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_deaths_global.csv))

Recovered: [https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series/time\\_series\\_covid19\\_recovered\\_global.csv](https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_recovered_global.csv) ([https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse\\_covid\\_19\\_data/csse\\_covid\\_19\\_time\\_series/time\\_series\\_covid19\\_recovered\\_global.csv](https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_time_series/time_series_covid19_recovered_global.csv))

## 1. Confirmed cases statistics

### §1.1. Cleaning and preparing for animation

The GIS technologies have played an important role in many aspects, including the data integration, and geospatial visualization of epidemic information, spatial tracking of confirmed cases, prediction of regional transmission, and many more. These provide support information for government sectors to fight against the COVID-19 spreading.

The Center for Systems Science and Engineering (CSSE) at Johns Hopkins University & Medicine (JHU) had provided the dashboard created with ESRI ArcGIS operation dashboard (<https://www.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6> (<https://www.arcgis.com/apps/opsdashboard/index.html#/bda7594740fd40299423467b48e9ecf6>)). But feature for visualizing the change of data overtime on the map is missing. Later JHU created animated map on confirmed cases here <https://coronavirus.jhu.edu/data/animated-world-map> (<https://coronavirus.jhu.edu/data/animated-world-map>), separately from the dashboard, but the users can only observe the map changing colors, they have no access to view the actual numbers or zoom in the map, as it is not interactive and does not show the actual data.

So I decided to create animated maps to explore data changes over time. In order to do that my current dataset structure should be changed. Now the data structure is that every day's statistics is a separate column, so the values are "scattered" in unique cells for each day and country; I will move all the values to a single "value" column, and move all days labels from columns names to single "Date" column. It will transform the dataset to its long variation with repeating country rows and date rows.

```
In [1]: 1 import pandas as pd #to work with tabular data
        2 import pycountry #to get the three-letter country codes ISO 3166-1 for each country
        3
```

```
4 df_cases=pd.read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_1
5
6 # Aggregate the dataset
7 df_cases = df_cases.drop(columns=['Province/State', 'Lat', 'Long'])
8 df_cases = df_cases.groupby('Country/Region').agg('sum')
9 date_list = list(df_cases.columns)
10
11 # Get the country codes for each country
12 #list(pycountry.countries) #uncomment to load the list of available data
13
14 def get_country_code(name):
15     """
16     Return ISO-3 letter code for country by its name;
17     Return None if name is not found in the pycountry.countries
18     """
19     try:
20         return pycountry.countries.lookup(name).alpha_3
21     except:
22         return None
23
24 df_cases['Country'] = df_cases.index
```

Several countries' names are written differently than pycountry expects, so I change their names to match and get the code

```
In [2]: 1 df_cases.loc[df_cases.Country=="Burma", 'Country']='Myanmar'
2 df_cases.loc[df_cases.Country=="Brunei", 'Country']='Brunei Darussalam'
3 df_cases.loc[df_cases.Country=="Iran", 'Country']='Iran, Islamic Republic of'
4 df_cases.loc[df_cases.Country=="Congo (Brazzaville)", 'Country']='Congo, The Democratic Republic of the'
5 df_cases.loc[df_cases.Country=="Congo (Kinshasa)", 'Country']='Republic of the Congo'
6 df_cases.loc[df_cases.Country=="Cote d'Ivoire", 'Country']="Côte d'Ivoire"
7 df_cases.loc[df_cases.Country=="Korea, South", 'Country']="Korea, Republic of"
8 df_cases.loc[df_cases.Country=="Syria", 'Country']="Syrian Arab Republic"
9 df_cases.loc[df_cases.Country=="Taiwan*", 'Country']="Taiwan, Province of China"
10 df_cases.loc[df_cases.Country=="Russia", 'Country']='Russian Federation'
11 df_cases.loc[df_cases.Country=="West Bank and Gaza", 'Country']='Palestine, State of'
12 df_cases.loc[df_cases.Country=="Venezuela", 'Country']='Venezuela, Bolivarian Republic of'
13 df_cases.loc[df_cases.Country=="US", 'Country']='United States'
```

As soon as names are unified, I can add their ISO-3 codes.

```
In [3]:
188
```

```
In [4]:
```

```
In [5]: 1 # Transform the dataset in a long format
```

```
In [6]: 1 # Rename columns
```

```
In [7]:
```

There is one "None" value left in the df\_confirmed\_long[0:60] slice ("Diamond Princess").

```
In [8]:
```

```
In [9]: 1 #df_cases[120:160] #checking the slice 3 in the dataset on confirmed cases, uncomment to load
```

There are "None" values for "MS Zaandam", "Holy See" (Vatican) and "Kosovo" left in the df\_confirmed\_long[60:120] slice. First is not a country, second is too small and excessive to dataset (population is 809 people), but Kosovo is important to show on the map as this European country has population more than 1.8 mln people and 11 thousands of confirmed cases.

The problem is, that "Kosovo" is not listed in pycountry dictionary (although the World Bank added XKX code to Kosovo in June 2017 according to archives <https://libraries.acm.org/binaries/content/assets/libraries/archive/world-bank-list-of-economies.pdf> (<https://libraries.acm.org/binaries/content/assets/libraries/archive/world-bank-list-of-economies.pdf>)), that is why I need to "fix Kosovo" after adding all other codes with apply(get\_country\_code).

```
In [10]: 1 #add ISO-3 code manually as it is not listed in pycountry dictionary
```

```
In [11]: 1 #check ISO-3 for "Kosovo" to make sure the code is applied
```

Out[11]:

	Country	ISO-3	Date	Value
92	Kosovo	XKX	1/22/20	0
280	Kosovo	XKX	1/23/20	0
468	Kosovo	XKX	1/24/20	0
656	Kosovo	XKX	1/25/20	0
844	Kosovo	XKX	1/26/20	0

Now it is safe to drop "None" values.

```
In [12]: 1 df_cases = df_cases.dropna()
```

Out[12]:

	Country	ISO-3	Date	Value
47	Denmark	DNK	1/22/20	0



```

Country ISO-3 Date Value
103 Luxembourg LUX 1/22/20 0
105 Madagascar MDG 1/22/20 0
106 Malawi MWI 1/22/20 0

```

Dataset is cleaned and has data on 185 countries over January - August 2020.

```

1 print(len(df_cases['ISO-3'].unique().tolist()))
2 print(len(df_cases['ISO-3']))

```

```

185
39775
True

```

```

Country False
ISO-3 False
Date False
Value False
dtype: bool

```

## §1.2. Animation of the map over time: cases

Now I can use Plotly Express to create animated map. The cumulative cases animation shows the total number of cases reported in each country at each point in time, regardless of how many people have recovered. Visualizing cumulative cases demonstrates the overall toll of coronavirus on a country over time.

```

1 import plotly.express as px
2 import numpy as np
3
4 df = df_cases
5 fig = px.choropleth(df,                                # input dataframe
6                     locationmode='ISO-3',              # set of locations used to map 'locations'
7                     locations="ISO-3",                 # identify country by code
8                     color=np.log10(df['Value']),        # identify values and replace linear scale with logarithmic scale
9                     hover_name="Country",              # identify column to add as name to hover information
10                    animation_frame="Date",             # identify date column
11                    projection="equiarectangular",       # select projection
12                    hover_data=[df['Value']],           # hover text
13                    center = {"lat": 14.883333, "lon": 5.266667}, # set map center
14                    color_continuous_scale=px.colors.sequential.Reds, # set color scale, "_r" to reverse color
15                    range_color=[0,round(np.log10(df['Value']).max(),2)], # set the range of dataset
16                    )
17
18 #customize layout
19 fig.update_layout(
20     title_text='Confirmed cases by country over time<br>January 22, 2020 - August 20, 2020',
21     geo=dict(showframe=False, showcoastlines=False, projection_type='equiarectangular'),
22
23     annotations = [dict(x=0.8,y=0.0,xref='paper',yref='paper',
24                         text='Source: <a href="https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_data.json">https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_data.json</a>',
25                         Johns Hopkins University & Medicine</a>'],

```

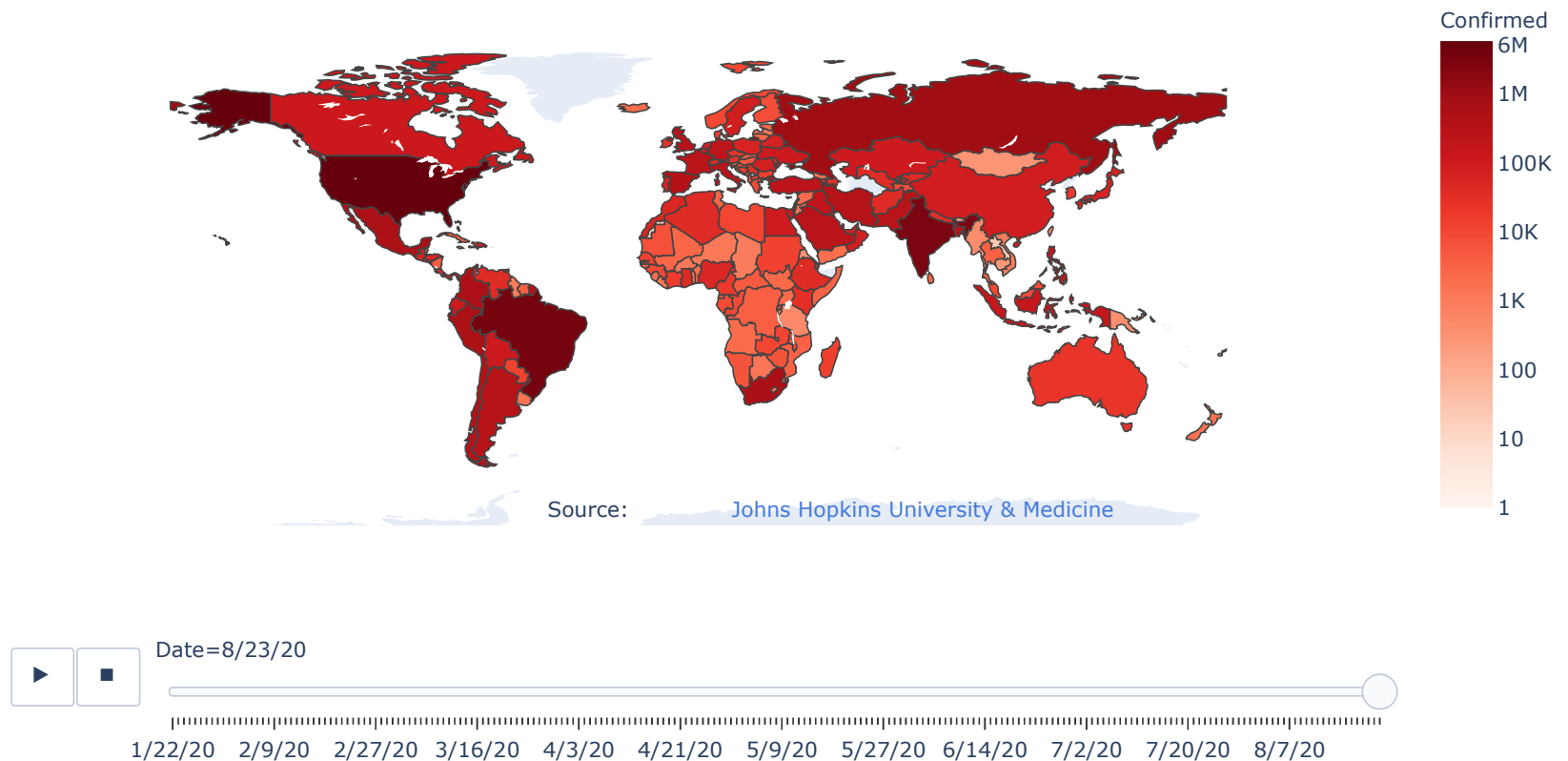


```

26     showarrow = False)],
27
28     #customize colorbar
29     coloraxis_colorbar=dict(
30         title='Confirmed',
31         tickvals=[0, 1, 2, 3, 4, 5, 6, 6.7], #customize colorbar title and ticks values
32         ticktext = ['1', '10', '100', '1K', '10K', '100K', '1M', '6M'] #replace log10 colorbar ticks
33     )
34
35 fig.show()
36 fig.write_html("Confirmed_map.html")

```

### Confirmed cases by country over time January 22, 2020 - August 20, 2020



The map has "Play" and "Stop" buttons near the Date of observation mark, and allows zooming and observing the number of cases, ISO-3 codes and dates of observation in hover info for each country.

The map reveals later some interesting details. JHU CSSE COVID-19 Dataset does not contain information on confirmed cases in Somaliland (part of Somalia, ISO-3 code of Somalia is "SOM"), North Korea (ISO-3 code is "PRK") and Turkmenistan ("TKM").

Somaliland has declared independence, but is not recognized internationally (hence not in the ISO list), so choropleth module has to particular code to use to map the data. Plotting by country name is also not possible because Somaliland borders are not interationally set and recognized.

North Korea escalates coronavirus response, but extent of outbreak is unclear; there are no confirmed cases of COVID-19 in North Korea, the government has taken extensive measures, including quarantines and travel restrictions. North Korea didn't admit to its 1st case until July, although city of Kaesong has been focus of quarantines. Since the end of December till August, according to unofficial data North Korea has quarantined and released 25,905 people, 382 of them foreigners.

Lack of information is not surprising in the first and the second case, but Turkmenistan is missing for different reasons. There is no official statistics on COVID-19 spread in Turkmenistan at all. The state-controlled media are not allowed to use the word "coronavirus" and it has even been removed from health information brochures distributed in schools, hospitals and workplaces (according to Turkmenistan Chronicle, one of the few sources of independent news, whose site is blocked within the country). Turkmenistan 2020 population is estimated at 6.0 mln people at mid year according to UN data.

In [17]:

```
Max confirmed cases: 5701679
```

As of August 20, 2020 maximum number of cases - 5.6 mln - were confirmed in USA, and there were performed about 69.6 mln tests there. Testing has covered every 208 out of 1000 people in the country.

### §1.3. Structure by region and income level: cases

I would like to see bigger picture for data, not only by country, but also by region and by income level. To make this happen I add region and income level columns to all countries. I use the World Bank data to create dataframe-converter and merge additional columns to my dataset.

```
In [18]: 1 df_convert=pd.read_csv("iso3_region_income_country.csv")
```

Out[18]:

	ISO-3	Region	IncomeLevel	Country_WB
0	AFG	South Asia	Low income	Afghanistan
1	AGO	Sub-Saharan Africa	Lower middle income	Angola
2	ALB	Europe & Central Asia	Upper middle income	Albania

```
In [19]: 1 df_cases=df_cases.merge(df_convert,on='ISO-3')
```

Out[19]:

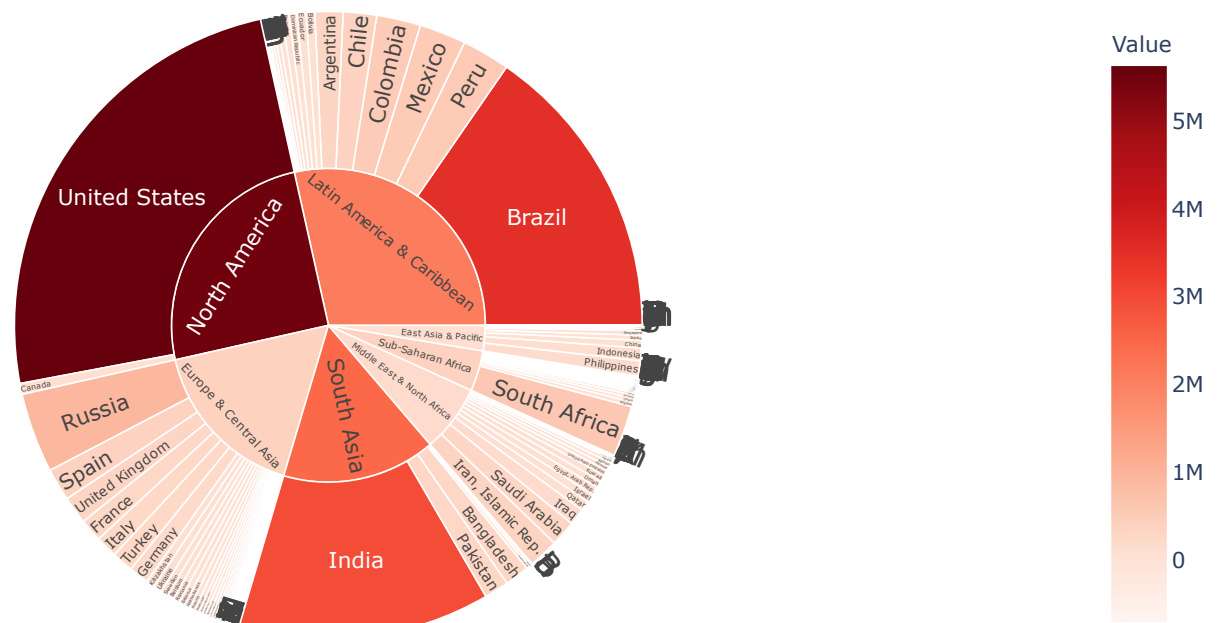
	Country	ISO-3	Date	Value	Region	IncomeLevel	Country_WB
0	Afghanistan	AFG	1/22/20	0	South Asia	Low income	Afghanistan

```
Country      False
ISO-3        False
Date         False
Value        False
Region       False
IncomeLevel  False
Country_WB   False
dtype: bool
```

Interactive sunburst plot represents hierarchial data as sectors laid out over several levels of concentric rings. Next sunburst graph shows countries within world's regions where the most cases of virus were confirmed. It is United States and Brazil in Americas, India - in South Asia, Russia - in Europe and Central Asia, and South Africa in African continent.

```
1 import numpy as np
2 import plotly.express as px
3 df = df_cases[df_cases['Date']=='8/21/20'] # take the last day of observation so cumulative values are maximum
4 fig = px.sunburst(df, path=['Region', 'Country_WB'], values=df.Value,
5                  color=df.Value, color_continuous_scale='Reds',
6                  title = 'Confirmed cases by regions and countries<br>by August 21, 2020',
7                  color_continuous_midpoint=np.average(df.Value,weights=df.Value))
8 fig.show()
```

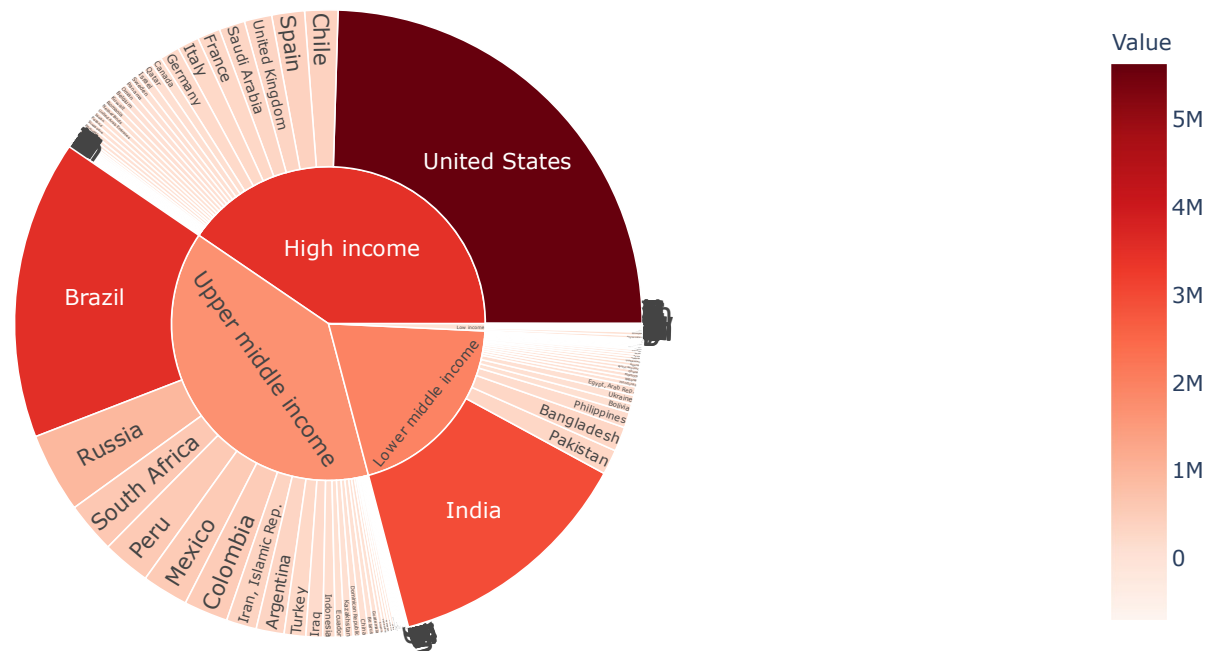
Confirmed cases by regions and countries  
by August 21, 2020



Is there a pattern in terms of virus spread between different regions of income? The next sunburst graph shows that 'High income' countries and 'Upper medium income' countries cover 41% and 39% of total COVID-19 cases respectively; "Lower middle income" countries cover less than 19.5% of total COVID-19 cases, and share of cases confirmed in low income countries is about 0.5%.

```
In [22]: 1 df = df_cases[df_cases['Date']=='8/21/20'] # take the last day of observation so cumulative values are maximum
2 fig = px.sunburst(df, path=['IncomeLevel', 'Country_WB'], values=df['Value'],
3                 color=df['Value'], color_continuous_scale='Reds',
4                 color_continuous_midpoint=np.average(df['Value'], weights=df['Value']),
5                 title = 'Confirmed cases by income level and countries<br>by August 21, 2020')
6 fig.show()
7
```

Confirmed cases by income level and countries  
by August 21, 2020



At first it could look like there is a correlation, as 80% of cases are confirmed in countries where income level is higher than medium. But it is important to note, that the number of confirmed cases is lower than the number of actual cases at all times, the main reason for that is limited testing. On one hand, this especially could make effect on COVID-19 statistics in lower income countries where the virus is harder to diagnosed due to various limitations. On the other hand, low income countries population is less globally mobile and this factor is probably slowing down the spreading of virus there in comparison with high income countries.

In any case there are lots of controversial effects from different groups of factors and it is too early to make conclusions at this stage given the available statistics.

## 2. Deaths statistics analysis

### §2.1. Cleaning and preparing for animation: deaths

```
In [23]: 1 import pandas as pd
2 import pycountry
3
4 df_deaths=pd.read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_data/deaths.csv")
5
6 # Aggregate the dataset
7 df_deaths = df_deaths.drop(columns=['Province/State', 'Lat', 'Long'])
8 df_deaths = df_deaths.groupby('Country/Region').agg('sum')
9 date_list = list(df_deaths.columns)
10
11 df_deaths['Country'] = df_deaths.index
```

Several countries' names are written differently than pycountry expects, so I change their names to match and get the code

```
In [24]: 1 df_deaths.loc[df_deaths.Country=="Burma", 'Country']='Myanmar'
2 df_deaths.loc[df_deaths.Country=="Brunei", 'Country']='Brunei Darussalam'
3 df_deaths.loc[df_deaths.Country=="Iran", 'Country']='Iran, Islamic Republic of'
4 df_deaths.loc[df_deaths.Country=="Congo (Brazzaville)", 'Country']='Congo, The Democratic Republic of the'
5 df_deaths.loc[df_deaths.Country=="Congo (Kinshasa)", 'Country']='Republic of the Congo'
6 df_deaths.loc[df_deaths.Country=="Cote d'Ivoire", 'Country']='Côte d'Ivoire'
7 df_deaths.loc[df_deaths.Country=="Korea, South", 'Country']='Korea, Republic of'
```

```

8 df_deaths.loc[df_deaths.Country=="Syria", 'Country']="Syrian Arab Republic"
9 df_deaths.loc[df_deaths.Country=="Taiwan*", 'Country']="Taiwan, Province of China"
10 df_deaths.loc[df_deaths.Country=="Russia", 'Country']="Russian Federation"
11 df_deaths.loc[df_deaths.Country=="West Bank and Gaza", 'Country']="Palestine, State of"
12 df_deaths.loc[df_deaths.Country=="Venezuela", 'Country']="Venezuela, Bolivarian Republic of"
13 df_deaths.loc[df_deaths.Country=="US", 'Country']="United States"

```

As soon as names are unified, I can add their ISO-3 codes.

In [25]:

In [26]:

```

1 # Transform the dataset in a long format
2 df_deaths = pd.melt(df_deaths, id_vars=['Country', 'ISO-3'], value_vars=date_list)

```

Out[26]:

Country	ISO-3	variable	value
---------	-------	----------	-------

In [27]:

```

1 df_deaths = df_deaths.rename(columns={"variable": "Date", "value": "Value"})

```

Out[27]:

Country	ISO-3	Date	Value
---------	-------	------	-------

In [28]:

```

1 #add Kosovo ISO-3 code as it is not listed in pycountry dictionary
2 df_deaths.loc[df_deaths.Country=="Kosovo", 'ISO-3']="XKX"

```

## §2.2. Animation of the map over time: deaths

In [29]:

```

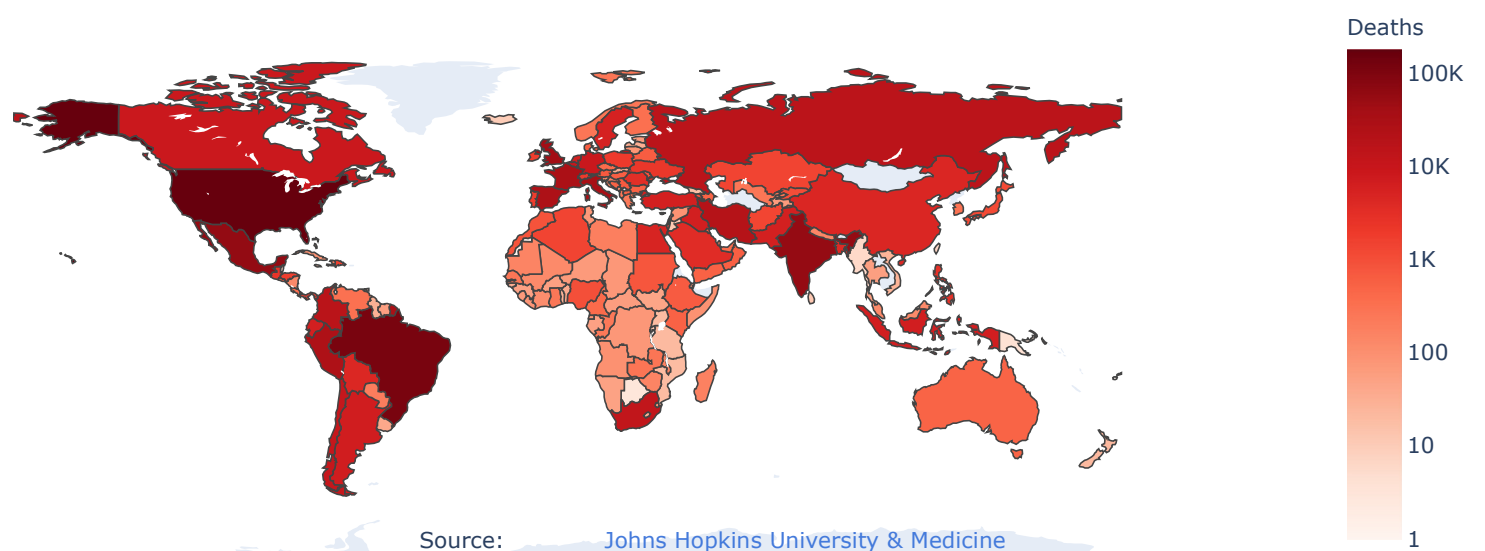
1 import plotly.express as px
2 df = df_deaths
3 fig = px.choropleth(df,                                # input dataframe
4                     locationmode='ISO-3',              # set of locations used to map 'locations'
5                     locations="ISO-3",                 # identify country by code
6                     color=np.log10(df['Value']),        # identify values and replace linear scale with logarithmic scale
7                     hover_name="Country",              # identify column to add as name to hover information
8                     animation_frame="Date",            # identify date column
9                     projection="equirectangular",       # select projection
10                    hover_data=[df['Value']],           # hover text
11                    center = {"lat": 14.883333, "lon": 5.266667}, # set map center
12                    color_continuous_scale=px.colors.sequential.Reds, # set color scale, "_r" to reverse color
13                    range_color=[0, round(np.log10(df['Value']).max(), 2)], # set the range of dataset
14                    )
15
16 #customize layout
17 fig.update_layout(
18     title_text='Deaths from COVID-19 by country over time<br>January 22, 2020 - August 21, 2020',
19     geo=dict(showframe=False, showcoastlines=False, projection_type='equirectangular'),
20
21     annotations = [dict(
22         x=0.8,
23         y=0.0,
24         xref='paper',
25         yref='paper',
26         text='Source: <a href="https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_data_v20200821.json">https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_data_v20200821.json</a>',
27         showarrow = False
28     )],
29 )
30
31 #customize colorbar
32 coloraxis_colorbar=dict(title='Deaths',
33                          tickvals=[0, 1, 2, 3, 4, 5, 5.3010299957], #customize colorbar title and ticks values
34                          ticktext = ['1', '10', '100', '1K', '10K', '100K', '200K'] #replace log10 colorbar ticks to
35                          )
36
37 fig.show()
38 fig.write_html("Deaths_map.html")

```

C:\Users\depth\anaconda3\lib\site-packages\pandas\core\series.py:679: RuntimeWarning:

divide by zero encountered in log10

Deaths from COVID-19 by country over time  
January 22, 2020 - August 21, 2020



As of August 2020 maximum number of lethal end cases were registered in USA. Mongolia has not reported COVID-19 deaths.

```
In [30]:  
  
Max number of deaths: 176802
```

```
In [31]: 1 df_deaths = df_deaths.merge(df_convert,on='ISO-3')
```

Out[31]:

	Country	ISO-3	Date	Value	Region	IncomeLevel	Country_WB
0	Afghanistan	AFG	1/22/20	0	South Asia	Low income	Afghanistan

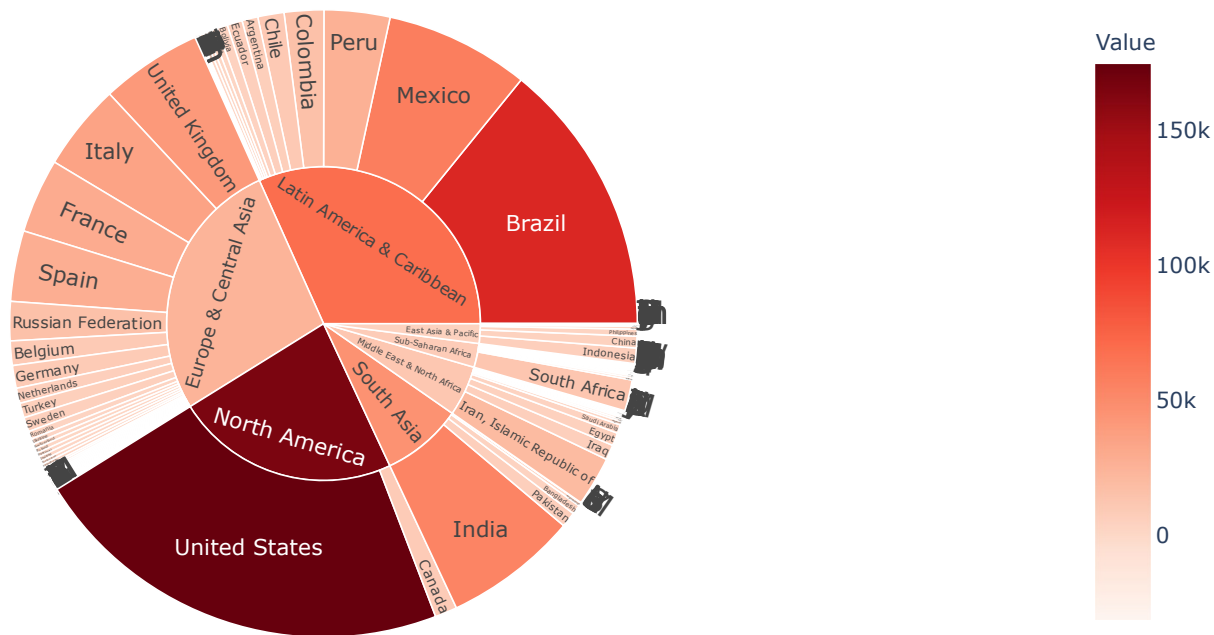
```
In [32]:  
  
Out[32]: Country      False  
ISO-3        False  
Date         False  
Value        False  
Region       False  
IncomeLevel  False  
Country_WB   False  
dtype: bool
```

§2.3. Structure by region and income level: deaths

Interactive sunburst graph shows countries within world's regions where the lethal end cases of COVID-19 were reported by August 20, 2020.

```
In [33]: 1 import numpy as np  
2 import plotly.express as px  
3  
4 # filter the rows for the last day of observation so cumulative values would be maximum and latest  
5 df = df_deaths[df_deaths['Date'] == '8/20/20']  
6  
7 #exclude rows with zero values of 12 countries with no deaths reported  
8 df = df[df['Value'] != 0]  
9  
10 fig = px.sunburst(df, path = ['Region', 'Country'], values = df.Value,  
11                  color = df.Value, color_continuous_scale='Reds',  
12                  title = 'Deaths from COVID-19 by regions and countries<br>by August 20, 2020',  
13                  color_continuous_midpoint=round(np.average(df.Value, weights = df.Value),1))  
14 fig.show()
```

Deaths from COVID-19 by regions and countries by August 20, 2020



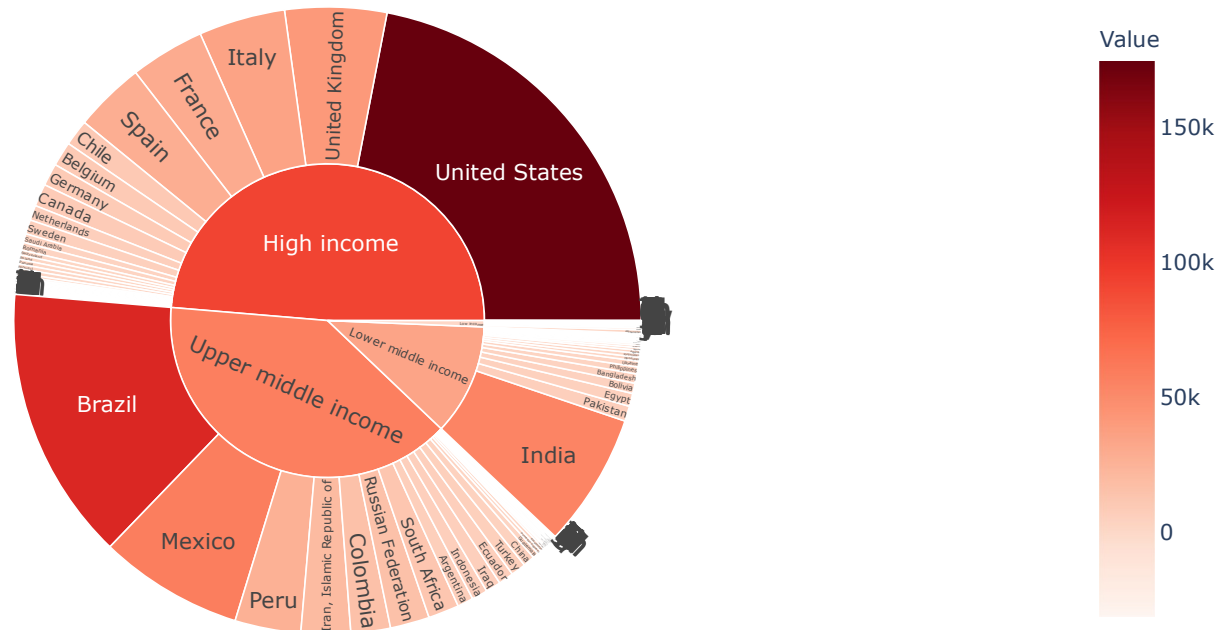
Is there a pattern in terms of virus spread between different regions of income? The sunburst graph shows that 'High income' countries and 'Upper medium income' countries cover 41% and 39% of total COVID-19 cases respectively; Lower middle income countries cover less than 19.5% of total COVID-19 cases; virus spread in low income countries are not that significant yet, or maybe it just was not diagnosed properly.

```
In [34]: 1 # filter the rows for the last day of observation so cumulative values would be maximum and latest  
2 df = df_deaths[df_deaths['Date'] == '8/20/20']  
3  
4 #exclude rows with zero values of 12 countries with no reported deaths
```



```
5 df = df[df['Value'] != 0]
6
7 fig = px.sunburst(df, path=['IncomeLevel', 'Country'], values=df.Value,
8                  color = df.Value, color_continuous_scale='Reds',
9                  title = 'Deaths from COVID-19 by income level and country<br>by August 20, 2020',
10                  color_continuous_midpoint=round(np.average(df.Value, weights = df.Value),1))
11 fig.show()
```

Deaths from COVID-19 by income level and country  
by August 20, 2020



3. Recovered patients

§3.1. Cleaning and preparing for animation: recovery

```
In [35]: 1 import pandas as pd
2 import pycountry
3
4 df_Recovered=pd.read_csv("https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_cov
5 # Aggregate the dataset
6 df_Recovered = df_Recovered.drop(columns=['Province/State', 'Lat', 'Long'])
7 df_Recovered = df_Recovered.groupby('Country/Region').agg('sum')
8 date_list = list(df_Recovered.columns)
9 df_Recovered['Country'] = df_Recovered.index
```

Several countries' names are written differently than pycountry expects, so I change their names to match and get the code

```
In [36]: 1 df_Recovered.loc[df_Recovered.Country=="Burma", 'Country']='Myanmar'
2 df_Recovered.loc[df_Recovered.Country=="Brunei", 'Country']='Brunei Darussalam'
3 df_Recovered.loc[df_Recovered.Country=="Iran", 'Country']='Iran, Islamic Republic of'
4 df_Recovered.loc[df_Recovered.Country=="Congo (Brazzaville)", 'Country']='Congo, The Democratic Republic of the'
5 df_Recovered.loc[df_Recovered.Country=="Congo (Kinshasa)", 'Country']='Republic of the Congo'
6 df_Recovered.loc[df_Recovered.Country=="Cote d'Ivoire", 'Country']='Côte d'Ivoire'
7 df_Recovered.loc[df_Recovered.Country=="Korea, South", 'Country']='Korea, Republic of'
8 df_Recovered.loc[df_Recovered.Country=="Syria", 'Country']='Syrian Arab Republic'
9 df_Recovered.loc[df_Recovered.Country=="Taiwan*", 'Country']='Taiwan, Province of China'
10 df_Recovered.loc[df_Recovered.Country=="Russia", 'Country']='Russian Federation'
11 df_Recovered.loc[df_Recovered.Country=="West Bank and Gaza", 'Country']='Palestine, State of'
12 df_Recovered.loc[df_Recovered.Country=="Venezuela", 'Country']='Venezuela, Bolivarian Republic of'
13 df_Recovered.loc[df_Recovered.Country=="US", 'Country']='United States'
```

As soon as names are unified, I can add their ISO-3 codes.

```
In [37]:
```

```
In [38]: 1 # View data structure
```

Out[38]:

	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	1/29/20	1/30/20	1/31/20	...	8/16/20	8/17/20	8/18/20	8/19/20	8/20/20	8/21/20	8/22/20
Country/Region																		
Afghanistan	0	0	0	0	0	0	0	0	0	0	...	27166	27166	27166	27166	27681	28016	28016
Albania	0	0	0	0	0	0	0	0	0	0	...	3794	3816	3871	3928	3986	4096	4184

2 rows × 217 columns

```
In [39]: 1 # Transform the dataset in a long format
2 df_Recovered = pd.melt(df_Recovered, id_vars=['Country', 'ISO-3'], value_vars=date_list)
```

Out[39]:

Country	ISO-3	variable	value
---------	-------	----------	-------

In [40]:

```
In [41]: 1 #add Kosovo ISO-3 code as it is not listed in pycountry dictionary
2 df_Recovered.loc[df_Recovered.Country=="Kosovo", 'ISO-3']="XKX"
```

```
In [42]: 1 # View data shape
```

```
Out[42]: (39775, 4)
```

As of August 20, 2020 maximum number of recovered patients in one country were registered in Brazil.

In [43]:

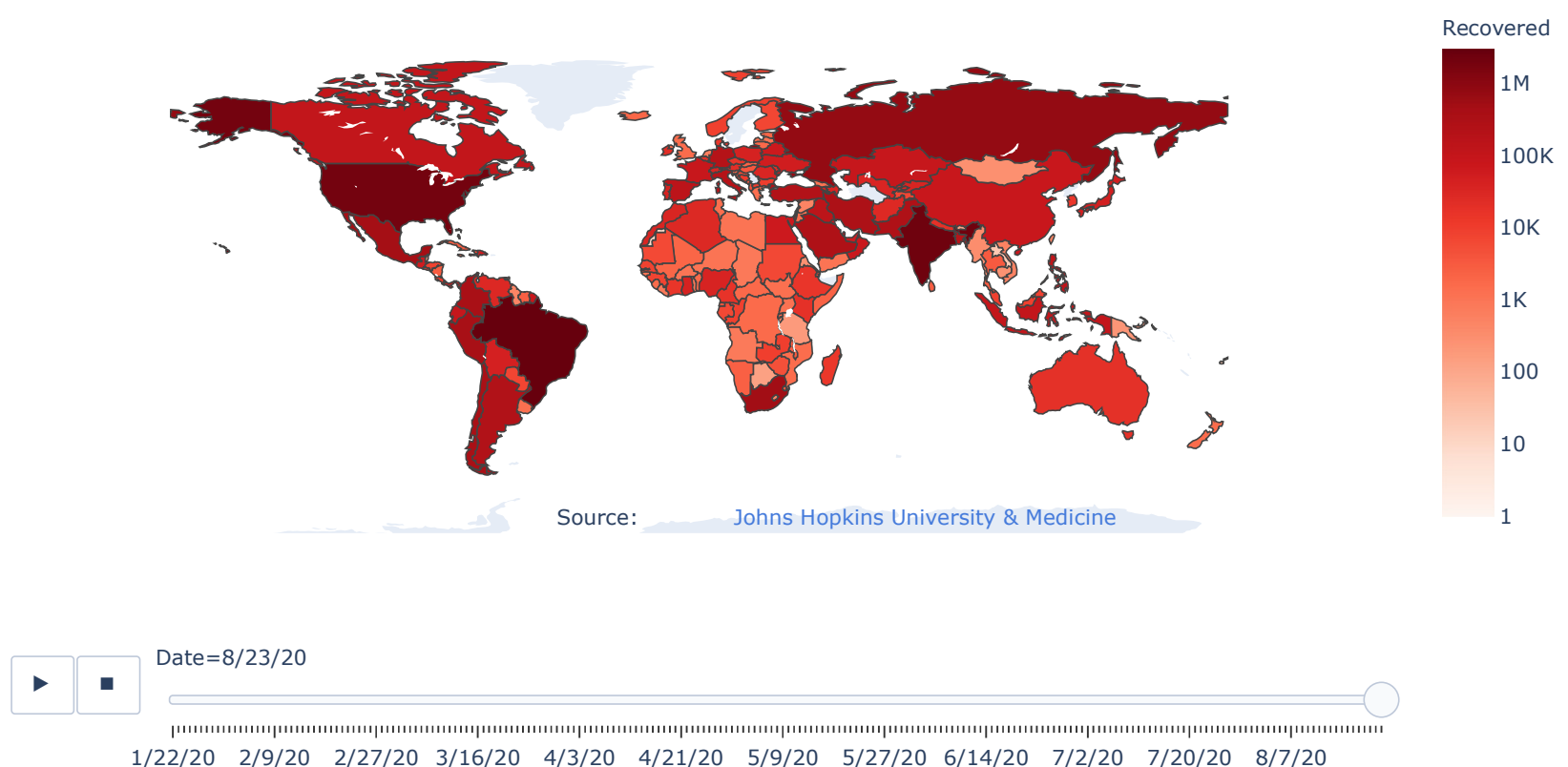
```
Max number of recovered: 2947784
```

### §3.2. Animation of the map over time: recovery

In [119]:

```
1 import plotly.express as px
2 df = df_Recovered
3 fig = px.choropleth(df,                                # input dataframe
4                     locationmode='ISO-3',              # set of locations used to map 'locations'
5                     locations="ISO-3",                 # identify country by code
6                     color=np.log10(df['Value']),        # identify values and replace linear scale with logarithmic scale
7                     hover_name="Country",              # identify column to add as name to hover information
8                     animation_frame="Date",            # identify date column
9                     projection="equiarectangular",      # select projection
10                    hover_data=[df['Value']],           # hover text
11                    center = {"lat": 14.883333, "lon": 5.266667}, # set map center
12                    color_continuous_scale=px.colors.sequential.Reds, # set color scale, "_r" to reverse color
13                    range_color=[0,round(np.log10(df['Value']).max(),2)], # set the range of dataset
14                    )
15
16 #customize layout
17 fig.update_layout(
18     title_text='Recovered from COVID-19 by country over time<br>January 22, 2020 - August 21, 2020',
19     geo=dict(showframe=False, showcoastlines=False, projection_type='equiarectangular'),
20
21     annotations = [dict(
22         x=0.8,
23         y=0.0,
24         xref='paper',
25         yref='paper',
26         text='Source: <a href="https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_data" >https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/csse_covid_19_data</a>',
27         showarrow = False
28     )],
29 )
30
31 #customize colorbar
32 coloraxis_colorbar=dict(title='Recovered',
33                          tickvals=[0,1,2, 3, 4, 5, 6, 6.5], #customize colorbar title and ticks values
34                          ticktext = ['1','10','100', '1K', '10K', '100K', '1M', '3M'] #replace log10 colorbar ticks
35                          )
36
37 fig.show()
38 fig.write_html("Recovered_map.html")
```

Recovered from COVID-19 by country over time  
January 22, 2020 - August 21, 2020



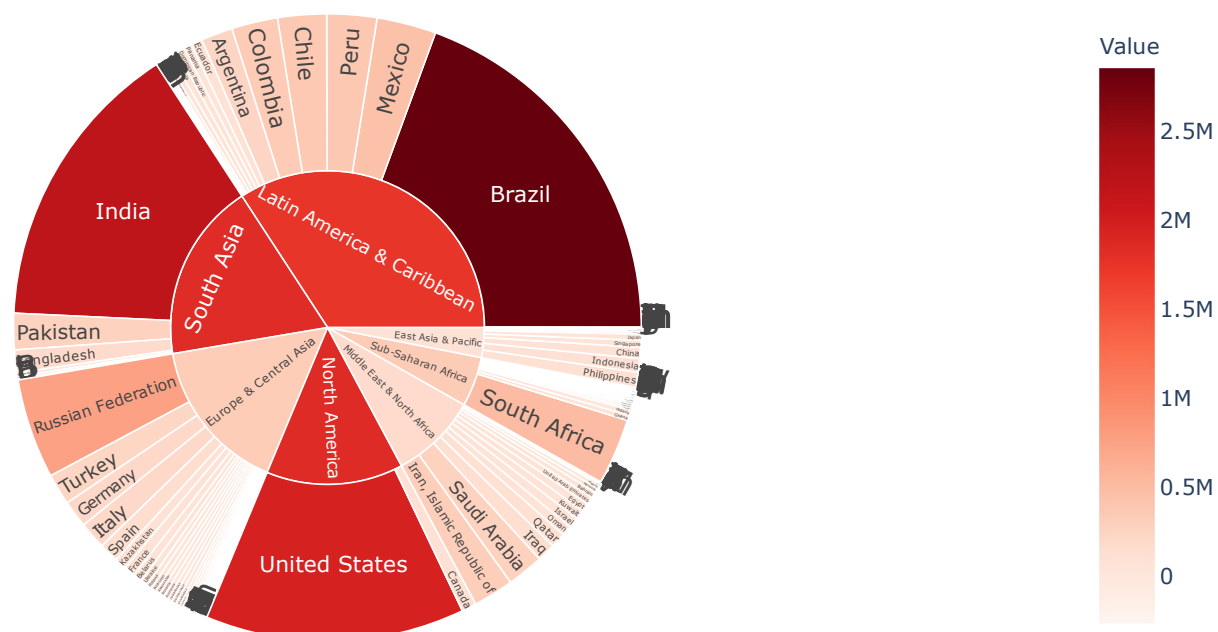
Interactive map shows where are located the most number of recovery cases. It is Brazil (2.6 mln), India (1.9 mln), United States (1.8 mln), Russia (0.7 mln) and South Africa (0.5 mln).

Out[45]:

```
Out[46]: Country      False
          ISO-3        False
          Date          False
          Value         False
          Region        False
          IncomeLevel   False
          Country_WB    False
          dtype: bool
```

Interactive sunburst graph shows countries within world's regions where the recovery cases of COVID-19 were reported by August 21, 2020.

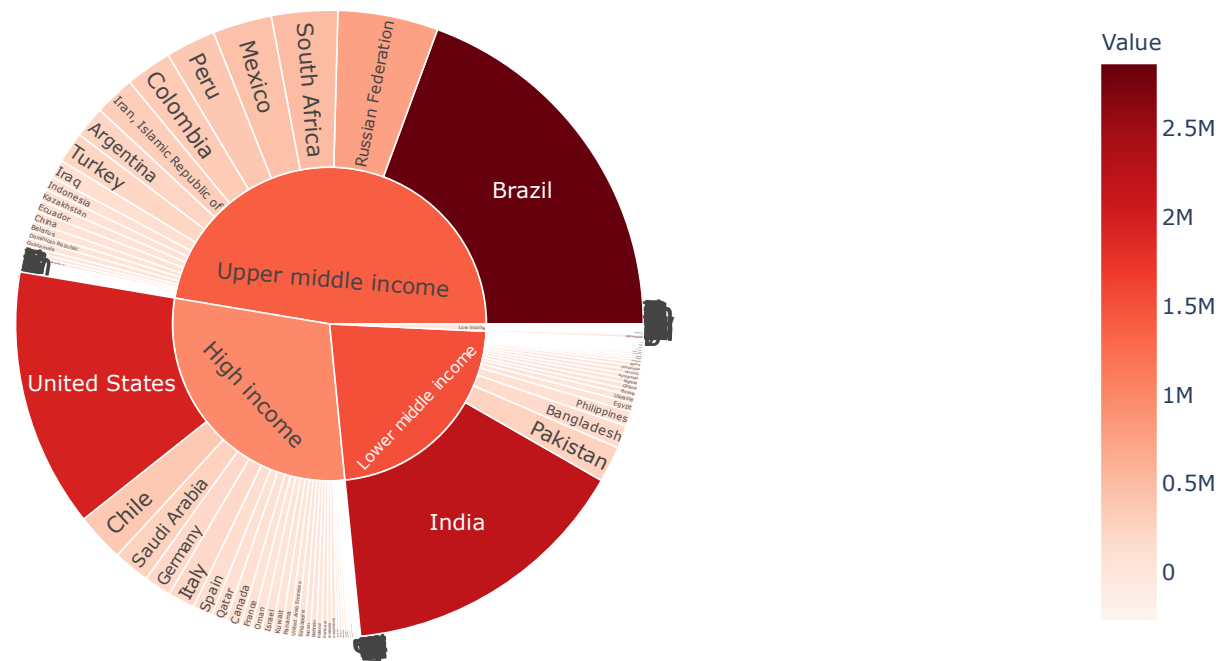
Recovered from COVID-19 by regions and countries  
by August 21, 2020





```
In [48]: 1 # filter the rows for the last day of observation so cumulative values would be maximum and latest
2 df = df_Recovered[df_Recovered['Date'] == '8/21/20']
3
4 #exclude rows with zero values of 12 countries with no reported deaths
5 df = df[df['Value'] != 0]
6
7 fig = px.sunburst(df, path=['IncomeLevel', 'Country'], values=df.Value,
8                  color = df.Value, color_continuous_scale='Reds',
9                  title = 'Recovered from COVID-19 by income level and country<br>by August 21, 2020',
10                  color_continuous_midpoint=round(np.average(df.Value, weights = df.Value),1))
11 fig.show()
```

Recovered from COVID-19 by income level and country  
by August 21, 2020



## 4. Calculating additional data

### §4.1. Importing, cleaning and calculating

In order to make meaningful comparison of deaths from virus between the countries I need to import population data first, and calculate deaths per million ratio.

```
In [49]: 1 import pandas as pd #to work with tabular data
2 import requests #to access the csv from the url string
3 import io #to read the csv directly from the url string
4
5 url_pop = "https://raw.githubusercontent.com/CSSEGISandData/COVID-19/master/csse_covid_19_data/UID_ISO_FIPS_LookUp_Table/population/v2018_01_01/iso3/population_v2018_01_01.csv"
6 raw=requests.get(url_pop).content
7 df_pop=pd.read_csv(io.StringIO(raw.decode('utf-8'))))
8
9 # View data shape
10 df_pop.shape
```

Out[49]: (4153, 12)

I expected to see about 200 rows in the dataset (1 row per country), but its shape contains 4153 rows. The reason for that is, as we can see below, that it contains not only population of the countries, but mostly population of provinces, states and cities for some countries.

```
In [50]: 1 # View a slice of loaded data
```

Out[50]:

	UID	iso2	iso3	code3	FIPS	Admin2	Province_State	Country_Region	Lat	Long_	Combined_Key	Population
78	15217	CL	CHL	152.0	NaN	NaN	Unknown	Chile	NaN	NaN	Unknown, Chile	NaN
79	170	CO	COL	170.0	NaN	NaN	NaN	Colombia	4.5709	-74.2973	Colombia	50882884.0
80	17001	CO	COL	170.0	NaN	NaN	Amazonas	Colombia	-1.4429	-71.5724	Amazonas, Colombia	76589.0
81	17002	CO	COL	170.0	NaN	NaN	Antioquia	Colombia	7.1986	-75.3412	Antioquia, Colombia	6407102.0
82	17003	CO	COL	170.0	NaN	NaN	Arauca	Colombia	7.0762	-70.7105	Arauca, Colombia	262174.0
83	17004	CO	COL	170.0	NaN	NaN	Atlantico	Colombia	10.6966	-74.8741	Atlantico, Colombia	2535517.0
84	17005	CO	COL	170.0	NaN	NaN	Bolivar	Colombia	8.6704	-74.0300	Bolivar, Colombia	2070110.0
85	17006	CO	COL	170.0	NaN	NaN	Boyaca	Colombia	5.4545	-73.3620	Boyaca, Colombia	1217376.0
86	17007	CO	COL	170.0	NaN	NaN	Caldas	Colombia	5.2983	-75.2479	Caldas, Colombia	998255.0
87	17008	CO	COL	170.0	NaN	NaN	Capital District	Colombia	4.7110	-74.0721	Capital District, Colombia	7412566.0
88	17009	CO	COL	170.0	NaN	NaN	Caqueta	Colombia	0.8699	-73.8419	Caqueta, Colombia	401489.0
89	17010	CO	COL	170.0	NaN	NaN	Casanare	Colombia	5.7589	-71.5724	Casanare, Colombia	420504.0
90	17011	CO	COL	170.0	NaN	NaN	Cauca	Colombia	2.7050	-76.8260	Cauca, Colombia	1464488.0

	UID	iso2	iso3	code3	FIPS	Admin2	Province_State	Country_Region	Lat	Long_	Combined_Key	Population
91	17012	CO	COL	170.0	NaN	NaN	Cesar	Colombia	9.3373	-73.6536	Cesar, Colombia	1200574.0
92	17013	CO	COL	170.0	NaN	NaN	Choco	Colombia	5.2528	-76.8260	Choco, Colombia	534826.0
93	17014	CO	COL	170.0	NaN	NaN	Cordoba	Colombia	8.0493	-75.5740	Cordoba, Colombia	1784783.0
94	17015	CO	COL	170.0	NaN	NaN	Cundinamarca	Colombia	5.0260	-74.0300	Cundinamarca, Colombia	2919060.0
95	17016	CO	COL	170.0	NaN	NaN	Guainia	Colombia	2.5854	-68.5247	Guainia, Colombia	48114.0
96	17017	CO	COL	170.0	NaN	NaN	Guaviare	Colombia	1.0654	-73.2603	Guaviare, Colombia	82767.0
97	17018	CO	COL	170.0	NaN	NaN	Huila	Colombia	2.5359	-75.5277	Huila, Colombia	1100386.0

To avoid double counting I need to clean the dataset from excessive information. To do that, I filter only the rows with values NaN in column "Province\_State" using the isnull function. After that I can simply drop "UID", "iso2", "code3", "FIPS", "Admin2", "Province\_State", "Combined\_Key" columns.

```
In [51]: 1 df_pop = df_pop[pd.isnull(df_pop.Province_State)]
2 df_pop = df_pop.drop("UID", axis=1)
3 df_pop = df_pop.drop("iso2", axis=1)
4 df_pop = df_pop.drop("FIPS", axis=1)
5 df_pop = df_pop.drop("Admin2", axis=1)
6 df_pop = df_pop.drop("Province_State", axis=1)
7 df_pop = df_pop.drop("Combined_Key", axis=1)
```

Now the dataset contains only population of 188 countries, and 5 columns instead of 12.

```
In [52]:
```

```
Out[52]: (188, 5)
```

There were 2 empty cells in population column of the dataframe, they contained data on "MS Zaandam" and "Diamond Princess".

```
In [53]: 1 df_pop = df_pop[df_pop.Country_Region != 'MS Zaandam']
2 df_pop = df_pop[df_pop.Country_Region != 'Diamond Princess']
```

```
Out[53]: (186, 5)
```

```
In [54]:
```

If I check maximum population, I see population of China.

```
In [55]:
```

```
Out[55]: 1404676330.0
```

Minimum population is in Vatican ("Holy See").

```
In [56]:
```

```
Out[56]: 809.0
```

Vatican population is too small to be visible on the map or graph, so it is better to get rid of it in dataset.

```
In [57]: 1 df_pop = df_pop[df_pop.Country_Region != 'Holy See']
```

```
Out[57]: (185, 5)
```

Now the dataset has the same number of countries, as the datasets of COVID-19 statistics. In order to use population data for calculation, I unify the "iso3"/"ISO-3" names of columns and merge "Population" column to df\_deaths dataframe.

```
In [58]:
```

```
In [59]:
```

```
Out[59]:
```

	Country	ISO-3	Date	Value	Region	IncomeLevel	Country_WB
0	Afghanistan	AFG	1/22/20	0	South Asia	Low income	Afghanistan

Both dataframes based on the same list of 185 countries.

```
In [60]:
```

```
Out[60]: True
```

```
In [61]:
```

```
In [62]:
```

```
In [63]: 1 df_deaths_pop["Deaths per mln"] = round((df_deaths_pop.Value / df_deaths_pop.Population * 1000000), 2)
```

```
Out[63]:
```

	Country	ISO-3	Date	Value	Region	IncomeLevel	Country_WB	Population	Deaths per mln
39555	Zimbabwe	ZWE	8/19/20	150	Sub-Saharan Africa	Lower middle income	Zimbabwe	14862927.0	10.09
39556	Zimbabwe	ZWE	8/20/20	151	Sub-Saharan Africa	Lower middle income	Zimbabwe	14862927.0	10.16
39557	Zimbabwe	ZWE	8/21/20	152	Sub-Saharan Africa	Lower middle income	Zimbabwe	14862927.0	10.23

	Country	ISO-3	Date	Value	Region	IncomeLevel	Country_WB	Population	Deaths per mln
39558	Zimbabwe	ZWE	8/22/20	153	Sub-Saharan Africa	Lower middle income	Zimbabwe	14862927.0	10.29

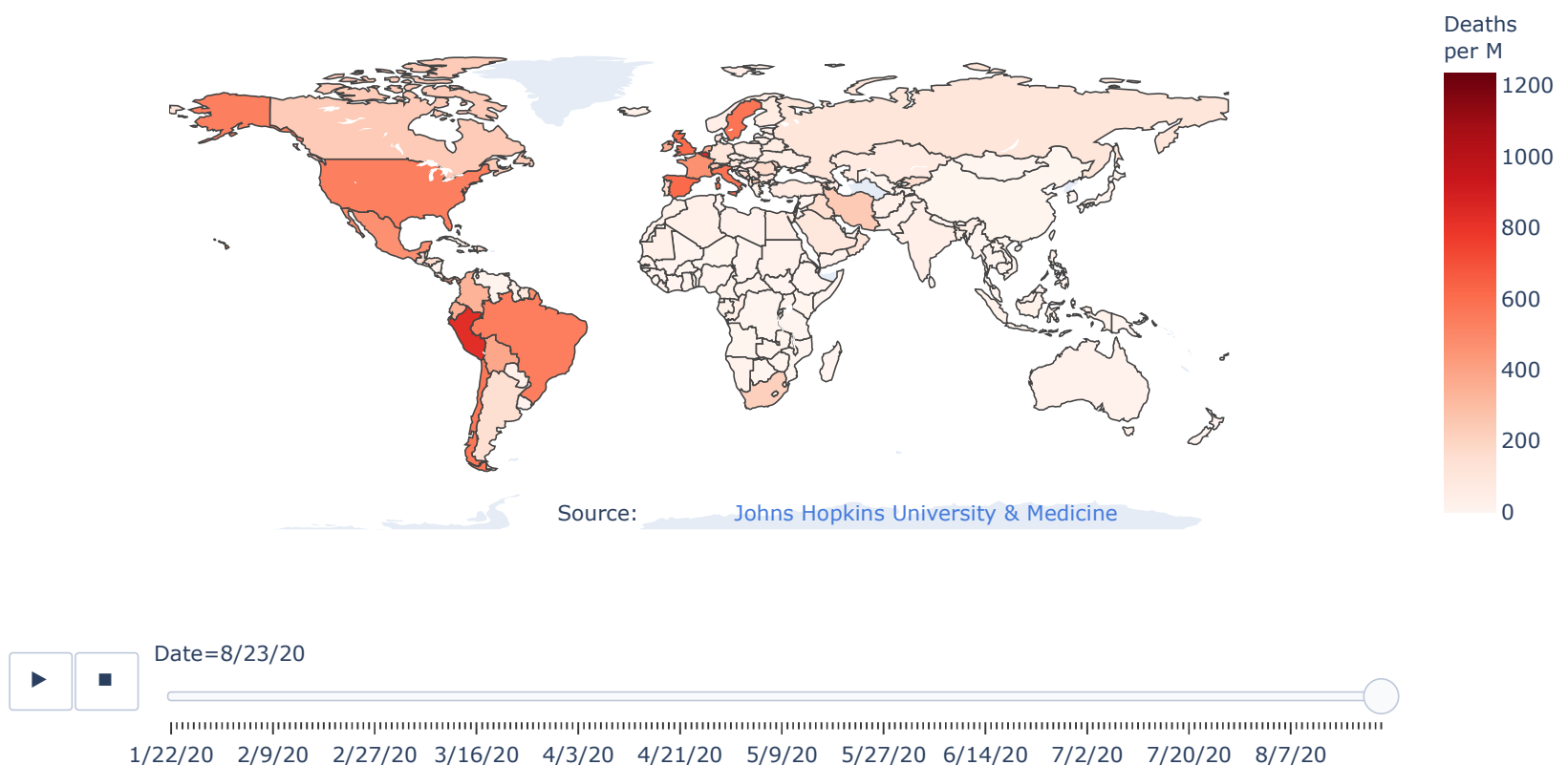
```
In [64]: 1 print(df_deaths_pop["Deaths per mln"].min())
2 print(df_deaths_pop["Deaths per mln"].max())
```

0.0  
1237.55  
40.58

## §2.2. Animation of the map over time: deaths per million

```
In [65]: 1 import plotly.express as px
2 df = df_deaths_pop
3 fig = px.choropleth(df,                                # input dataframe
4                     locationmode='ISO-3',              # set of locations used to map 'locations'
5                     locations="ISO-3",                 # identify country by code
6                     color=df['Deaths per mln'],         # identify values
7                     hover_name="Country",              # identify column to add as name to hover information
8                     animation_frame="Date",            # identify date column
9                     projection="equirectangular",       # select projection
10                    hover_data=[df['Deaths per mln']],   # hover text
11                    center = {"lat": 14.883333, "lon": 5.266667}, # set map center
12                    color_continuous_scale=px.colors.sequential.Reds, # set color scale, "_r" to reverse color
13                    range_color=[0,round(df["Deaths per mln"].max(),0)] # set the range of dataset
14                )
15
16 #customize layout
17 fig.update_layout(
18     title_text='Deaths from COVID-19 per million people by country over time<br>January 22, 2020 - August 21, 2020',
19     geo=dict(showframe=False, showcoastlines=False, projection_type='equirectangular'),
20
21     annotations = [dict(x=0.8,y=0.0,xref='paper',yref='paper', text='Source: <a href="https://raw.githubusercontent.com/Johns Hopkins University & Medicine</a>', showarrow = False)],
22
23
24 #customize colorbar
25 coloraxis_colorbar=dict(title='Deaths<br>per M')
26 )
27 fig.show()
28 fig.write_html("Deaths_per_mln_map.html")
```

Deaths from COVID-19 per million people by country over time  
January 22, 2020 - August 21, 2020



## Dataset 2: Oxford Covid-19 Government Response Tracker

This part of the project is based on ideas "Variation in government responses to COVID-19" paper by Oxford:

"As governments continue to respond to COVID-19, it is imperative to study what measures are effective and which are not. While the data presented here do, of course, not measure effectiveness directly, they can be useful input to studies that analyse factors affecting disease progression. OxCGRT seeks to contribute to this knowledge gap by providing comparable measures of individual policy actions, as well as several comparable aggregate indices. We find significant variation in both the measures that governments adopt and when they adopt them. Going forward, governments will benefit from adopting an evidence-based approach to the measures they deploy."

Paper: <https://www.bsg.ox.ac.uk/sites/default/files/2020-05/BSG-WP-2020-032-v6.0.pdf> (<https://www.bsg.ox.ac.uk/sites/default/files/2020-05/BSG-WP-2020-032-v6.0.pdf>)

Data: <https://github.com/OxCGRT/covid-policy-tracker> (<https://github.com/OxCGRT/covid-policy-tracker>)

## 5. Government response index

## §5.1. Cleaning and preparing for plotting

The Oxford Covid-19 Government Response Tracker (OxCGRT) collects systematic information on which governments have taken which measures, and when. The data is daily published at project's GitHub page <https://github.com/OxCGRT/covid-policy-tracker> (<https://github.com/OxCGRT/covid-policy-tracker>)

```
In [66]: 1 import requests #to access the csv from the url string
2 import io #to read the csv directly from the url string
3 import pandas as pd #to work with tabular data
4 import pycountry #to get the three-letter country codes ISO 3166-1 for each country
5
6 url_OxCGRT = "https://raw.githubusercontent.com/OxCGRT/covid-policy-tracker/master/data/OxCGRT_latest.csv"
7 raw=requests.get(url_OxCGRT).content
8 df_OxCGRT=pd.read_csv(io.StringIO(raw.decode('utf-8')))
```

Out[66]: (43845, 42)

```
In [67]: 1 print("Minimum date in YYYYMMDD format is", df_OxCGRT.Date.min())
```

Minimum date in YYYYMMDD format is 20200101  
Maximum date in YYYYMMDD format is 20200824

In [68]:

Out[68]:

	CountryName	CountryCode	Date	C1_School closing	C1_Flag	C2_Workplace closing	C2_Flag	C3_Cancel public events	C3_Flag	C4_Restrictions on gatherings	...	StringencyIndex	StringencyInd
181	Aruba	ABW	20200630	0.0	NaN	1.0	1.0	0.0	NaN	0.0	...	32.41	
418	Afghanistan	AFG	20200630	3.0	1.0	3.0	0.0	2.0	1.0	4.0	...	78.70	
655	Angola	AGO	20200630	3.0	1.0	2.0	0.0	2.0	1.0	3.0	...	75.93	

3 rows × 42 columns

```
In [69]: 1 # drop excessive columns
2 df_OxCGRT_indexes = df_OxCGRT.drop(axis=1, columns = ['C1_School closing', 'C1_Flag',
3                                                     'C2_Workplace closing', 'C2_Flag',
4                                                     'C3_Cancel public events', 'C3_Flag',
5                                                     'C4_Restrictions on gatherings', 'C4_Flag',
6                                                     'C5_Close public transport', 'C5_Flag',
7                                                     'C6_Stay at home requirements', 'C6_Flag',
8                                                     'C7_Restrictions on internal movement', 'C7_Flag',
9                                                     'C8_International travel controls',
10                                                    'E1_Income support', 'E1_Flag',
11                                                    'E2_Debt/contract relief', 'E3_Fiscal measures',
12                                                    'E4_International support',
13                                                    'H1_Public information campaigns', 'H1_Flag',
14                                                    'H2_Testing policy', 'H3_Contact tracing',
15                                                    'H4_Emergency investment in healthcare',
16                                                    'H5_Investment in vaccines',
17                                                    'M1_Wildcard',
18                                                    'StringencyIndex',
19                                                    'StringencyLegacyIndex',
20                                                    'StringencyLegacyIndexForDisplay',
21                                                    'GovernmentResponseIndex',
22                                                    'ContainmentHealthIndex',
23                                                    'EconomicSupportIndex'
24                                                    ])
25 df_OxCGRT_indexes[df_OxCGRT_indexes.Date == 20200820]
```

Out[69]:

	CountryName	CountryCode	Date	ConfirmedCases	ConfirmedDeaths	StringencyIndexForDisplay	GovernmentResponseIndexForDisplay	ContainmentHea
232	Aruba	ABW	20200820	1296.0	5.0	47.22	51.28	
469	Afghanistan	AFG	20200820	37759.0	1383.0	62.04	49.36	
706	Angola	AGO	20200820	1966.0	90.0	79.17	67.63	
943	Albania	ALB	20200820	7812.0	234.0	53.70	55.13	
1180	Andorra	AND	20200820	1024.0	53.0	44.44	58.97	
...	...	...	...	...	...	...	...	
42892	Kosovo	RKS	20200820	11545.0	390.0	69.44	64.10	
43129	Anguilla	AIA	20200820	3.0	0.0	26.85	36.54	
43366	Falkland Islands	FLK	20200820	13.0	0.0	27.78	39.74	
43603	Montserrat	MSR	20200820	NaN	NaN	64.81	65.38	
43840	Pitcairn Islands	PCN	20200820	NaN	NaN	11.11	NaN	

185 rows × 9 columns

The Oxford Covid-19 Government Response Tracker tracks individual policy measures across 17 indicators and calculate several indices to give an overall impression of government activity.

I am particularly interested in 2 aggregated indexes calculated by Oxford:

- GovernmentResponseIndexForDisplay (all 17 indicators)
- EconomicSupportIndexForDisplay (2 indicators)

Each of these indices report a number between 0 to 100 that reflects the level of the governments response along certain dimensions. This is a measure of how many of the relevant indicators a government has acted upon, and to what degree. The index cannot say whether a government's policy has been implemented effectively.

Each index dataframe could be downloaded as separate .csv file from <https://github.com/OxCGRT/covid-policy-tracker/tree/master/data/timeseries>  
(<https://github.com/OxCGRT/covid-policy-tracker/tree/master/data/timeseries>)

```
In [70]: 1 df_iGovResp=pd.read_csv("index_governmentresponse.csv")
172
```

I will plot the data of United States, Canada, China, Brazil and Russia, and compare those countries' indexes to each other and world's average (calculated as mean of 172 countries available in this dataset). First, identify parts of dataframe I need to plot:

```
In [71]: 1 ca_iGovResp = df_iGovResp[df_iGovResp.CountryName == 'Canada'].drop(axis=1, columns = ['CountryCode']) #Canada
2 ru_iGovResp = df_iGovResp[df_iGovResp.CountryName == 'Russia'].drop(axis=1, columns = ['CountryCode']) #Russia
3 wld_iGovResp = df_iGovResp.mean(axis=0, skipna=True).drop(axis=1, columns = ['CountryCode']) #"World", 172 countries
4 cn_iGovResp = df_iGovResp[df_iGovResp.CountryName == 'China'].drop(axis=1, columns = ['CountryCode']) #China
5 br_iGovResp = df_iGovResp[df_iGovResp.CountryName == 'Brazil'].drop(axis=1, columns = ['CountryCode']) #Brazil
6 us_iGovResp = df_iGovResp[df_iGovResp.CountryName == 'United States'].drop(axis=1, columns = ['CountryCode']) #United
```

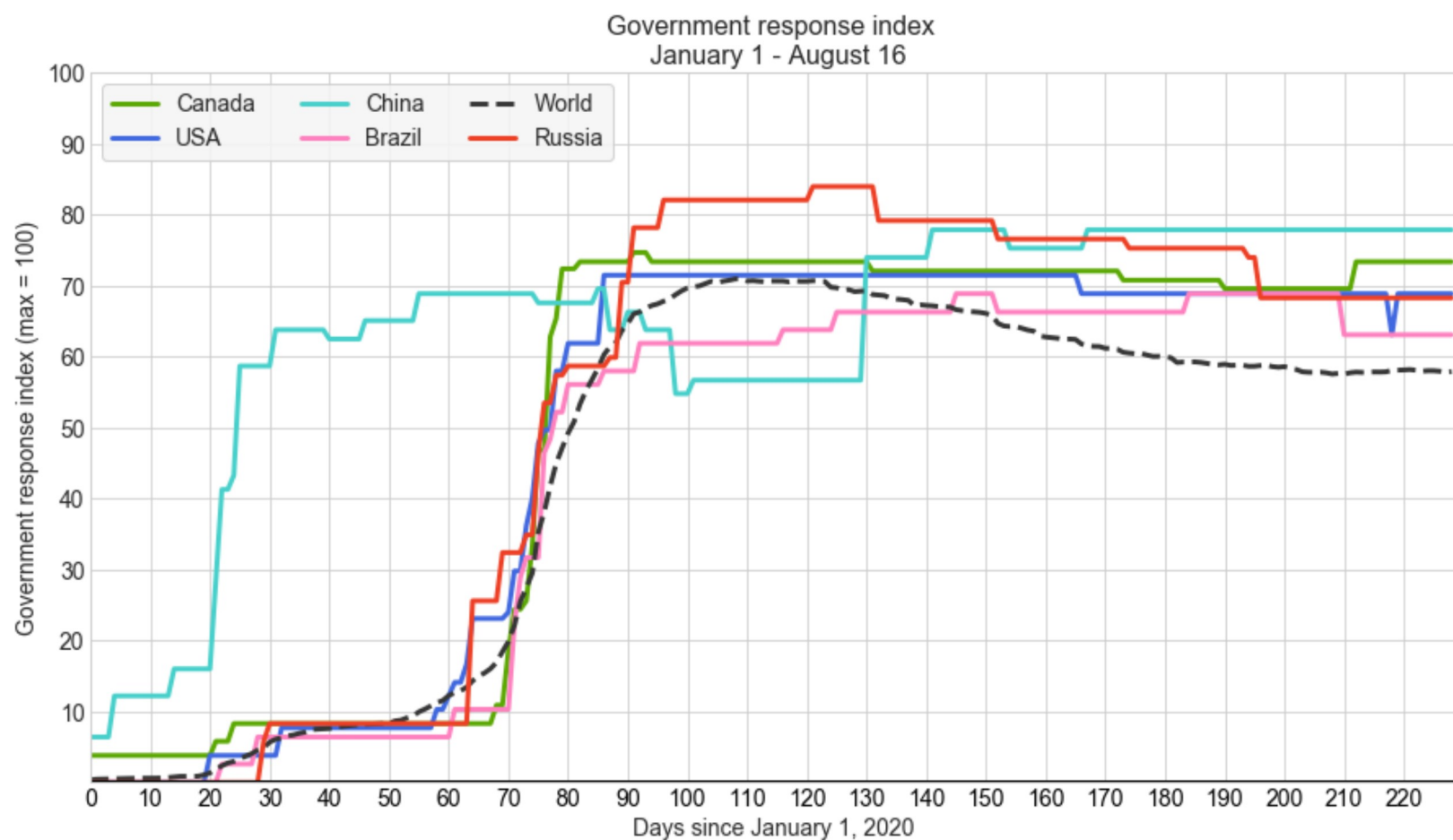
```
In [72]: 1 import matplotlib.pyplot as plt
2 import matplotlib.style as style
3 import numpy as np
4 import seaborn as sns
5 %matplotlib inline
6
7 SMALL_SIZE = 12
8 MEDIUM_SIZE = 14
9 BIGGER_SIZE = 16
10
11 plt.rc('font', size=MEDIUM_SIZE) # controls default text sizes
12 plt.rc('axes', titlesize=BIGGER_SIZE) # fontsize of the axes title
13 plt.rc('axes', labelsiz=MEDIUM_SIZE) # fontsize of the x and y labels
14 plt.rc('xtick', labelsiz=MEDIUM_SIZE) # fontsize of the tick labels
15 plt.rc('ytick', labelsiz=MEDIUM_SIZE) # fontsize of the tick labels
16 plt.rc('legend', fontsize=MEDIUM_SIZE) # legend fontsize
```



```

In [73]: 1 import matplotlib.pyplot as plt
2 from matplotlib.animation import FuncAnimation
3
4 plt.figure(figsize=(15,8))
5 plt.style.use('seaborn-whitegrid')
6
7 plt.title('Government response index \n January 1 - August 16')
8 plt.axhline(y = 0, color = 'black', linewidth = 1.3, alpha = .7)
9 plt.tick_params(axis = 'both', which = 'major')
10
11 x = np.linspace(0, 228, 229) # x axis start point, end point and number of intervals
12 xlim = (0, 229) # y axis start point, end point
13 y_pos = np.arange(len(wld_iGovResp))
14
15
16 # use the plt.xticks function to custom labels
17 plt.xticks(y_pos, color='black', rotation=False)
18 plt.xticks(np.arange(0, 230, 10.0))
19 plt.yticks(np.arange(10, 110, 10))
20 plt.xlabel('Days since January 1, 2020')
21 plt.ylabel('Government response index (max = 100)')
22
23 y1 = np.array(ca_iGovResp.drop(axis=1, columns = 'CountryName').values.tolist()[0])
24 y2 = np.array(us_iGovResp.drop(axis=1, columns = 'CountryName').values.tolist()[0])
25 y3 = np.array(cn_iGovResp.drop(axis=1, columns = 'CountryName').values.tolist()[0])
26 y4 = np.array(br_iGovResp.drop(axis=1, columns = 'CountryName').values.tolist()[0])
27 y5 = np.array(wld_iGovResp)
28 y6 = np.array(ru_iGovResp.drop(axis=1, columns = 'CountryName').values.tolist()[0])
29
30 ca = plt.plot(x, y1, label='Canada',c="xkcd:leaf green", lw=3, animated =True)
31 us = plt.plot(x, y2, label='USA', c="royalblue", lw=3, animated =True)
32 cn = plt.plot(x, y3, label='China', c="mediumturquoise", lw=3, animated =True)
33 br = plt.plot(x, y4, label='Brazil', c="xkcd:pink", lw=3, animated =True)
34 wld = plt.plot(x, y5, label='World', ls='--',c='xkcd:dark grey', lw=3, animated =True)
35 ru = plt.plot(x, y6, label='Russia', c="xkcd:tomato", lw=3, animated =True)
36
37 plt.axis([0, 229, 0, 100])
38 plt.axhline(0, c='black', ls='--', lw=2)
39 plt.axvline(0, c='black', ls='--', lw=2)
40 plt.legend(loc='upper left', frameon=True, fancybox=True, framealpha=0.9, facecolor='whitesmoke', ncol=3)

```



The graph shows government response index to COVID-19 for period since January 1st till August 16th.

Government of China was the first one that responded to COVID-19, dealing with the outbreak first identified in Wuhan in December 2019. It took actions 50-70 days ahead of the rest of the world. Measures taken were more strict in terms of methods, to prevent more damage from virus. Both developing countries (like Brazil and Russia) and developed countries (United States and Canada) preferred not to make unpopular and costly decisions and not enforce strict measures to prevent virus spread till mid-March.

Brazil government reaction was relatively mild in comparison with both all of the rest plotted countries and world's average. Russian government took more strict measures and keeps that level above world's average since March.

## §5.2. Aggregating data for correlation analysis

```

In [74]: 1 data_all = df_deaths_pop #start aggregating data to single dataframe
2 data_all = data_all[data_all.Date == '8/21/20'] #filter the latest day to research
3 print(data_all.shape)
4 data_all = data_all.rename(columns = {'Value':'Deaths',
5                                     'IncomeLevel':'Income',
6                                     'Death_per_mln':'Deaths per mln'})

```

(184, 9)

```
In [75]: 1 #merge Cases to data_all
2 Cases = df_cases[df_cases['Date']=='8/21/20']
3 Cases = Cases[Cases['Value']>0]
4 Cases = Cases.drop(axis=1, columns = ['Country','Date','Region','IncomeLevel','Country_WB'])
5 Cases = Cases.rename(columns = {'Value':'Cases'})
6 data_all = data_all.merge(Cases, on = "ISO-3")
7
8 #Calculate Cases per mln of population
9 data_all['Cases per mln'] = data_all.Cases / data_all.Population *1000000
```

(184, 11)

## §5.3. Correlation analysis

### Correlation analysis: Cases and Deaths

In [101]:

Out[101]:

	Country	ISO-3	Date	Deaths	Region	Income	Country_WB	Population	Deaths per mln	Cases	Cases per mln
0	Afghanistan	AFG	8/21/20	1385	South Asia	Low income	Afghanistan	38928341.0	35.58	37894	973.429615

I explore the correlation between columns.

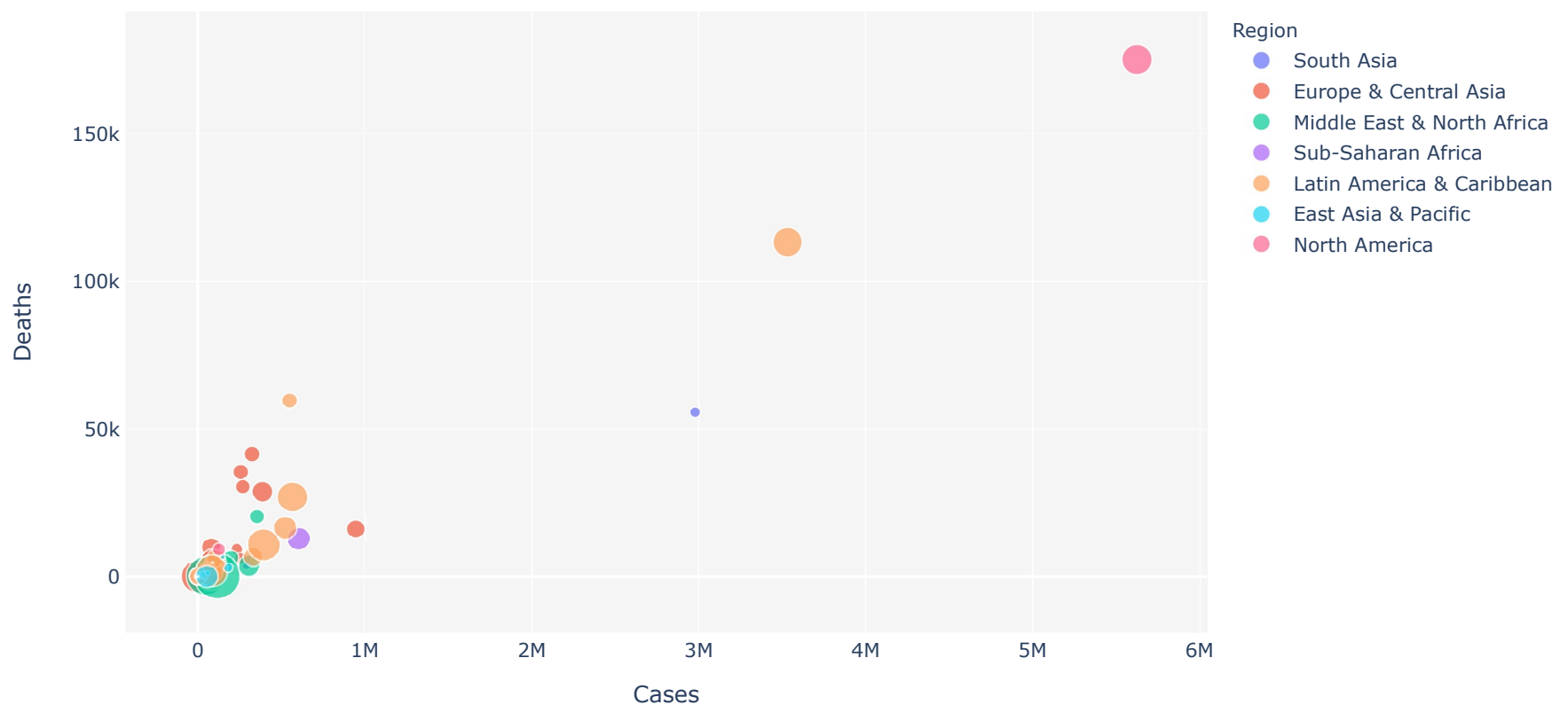
```
In [102]: 1 print("Q: Do cumulative cases correlate with cumulative deaths over time?")
2 print("A: Yes, correlation is positive and strong, standard correlation coefficient is 0.94")
3 data_all_nonulls = data_all[data_all.Deaths != 0] #filter out 12 countries that have not reported deaths
4 r = data_all_nonulls.Deaths.corr(data_all_nonulls.Cases)
```

Q: Do cumulative cases correlate with cumulative deaths over time?

A: Yes, correlation is positive and strong, standard correlation coefficient is 0.94

Out[102]: 0.94

```
In [103]: 1 # Cases vs Deaths linear plot
2 import plotly.express as px
3 df = data_all_nonulls
4 fig = px.scatter(df, x=df['Cases'], y=df['Deaths'],
5                 color='Region', size=round(df['Cases per mln'],1),
6                 hover_data=['Country'],
7                 labels={'x':'Cases', 'y':'Deaths', 'size':'Cases per mln'},
8                 )
9 fig.update_layout(plot_bgcolor = 'whitesmoke')
```



```
In [104]: 1 print("Q: log(Deaths) correlate with log(Cases)?")
2 print("A: Yes, correlation is positive and very strong, r = 0.93")
3 r = (np.log(data_all_nonulls.Deaths).corr(np.log(data_all_nonulls.Cases), method = "pearson"))
```

Q: log(Deaths) correlate with log(Cases)?

A: Yes, correlation is positive and very strong, r = 0.93

Out[104]: 0.94

In [105]:

Out[105]:

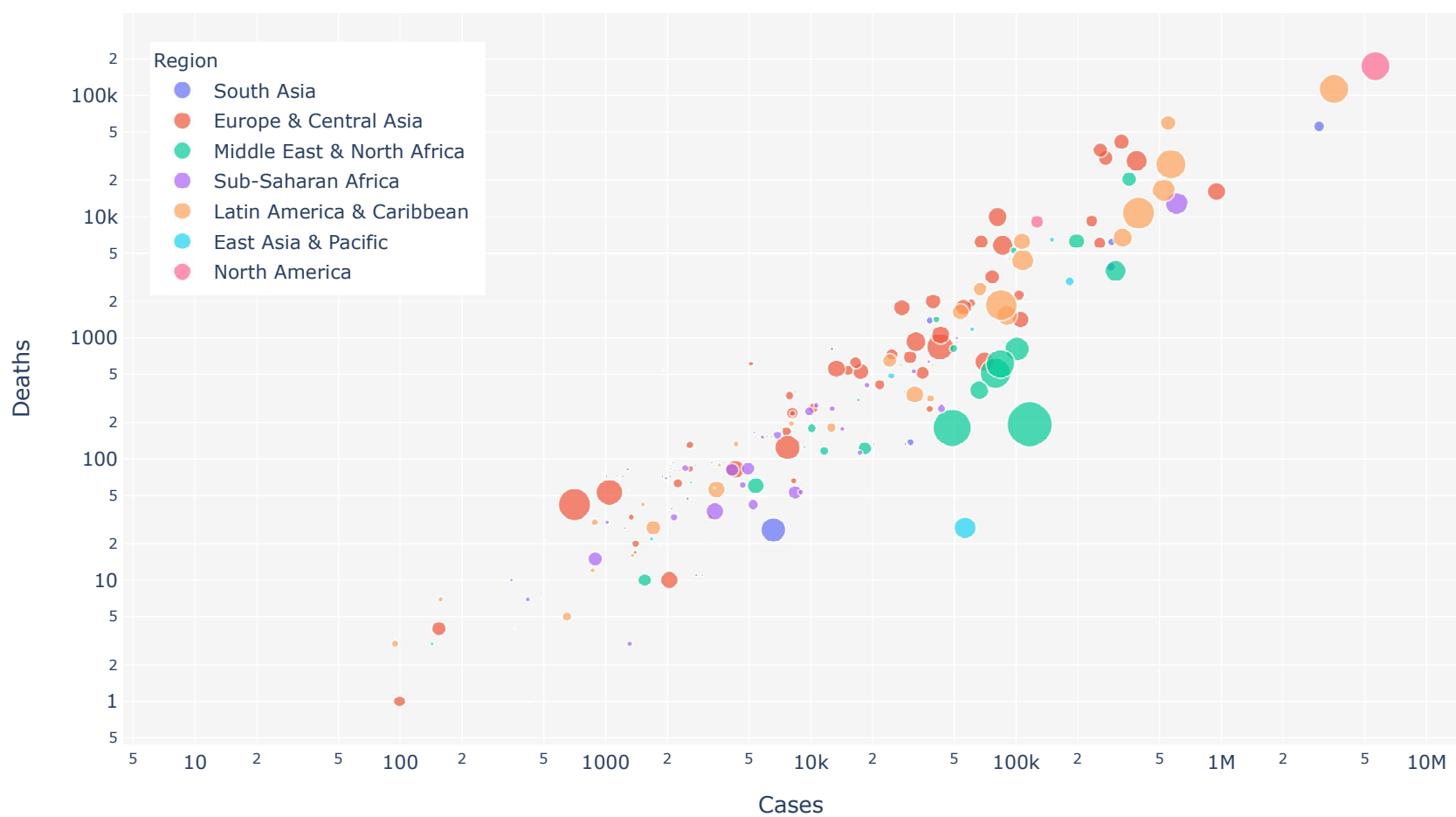
	Country	ISO-3	Date	Deaths	Region	Income	Country_WB	Population	Deaths per mln	Cases	Cases per mln
--	---------	-------	------	--------	--------	--------	------------	------------	----------------	-------	---------------

	Country	ISO-3	Date	Deaths	Region	Income	Country_WB	Population	Deaths per mln	Cases	Cases per mln
0	Afghanistan	AFG	8/21/20	1385	South Asia	Low income	Afghanistan	38928341.0	35.58	37894	973.429615
1	Albania	ALB	8/21/20	240	Europe & Central Asia	Upper middle income	Albania	2877800.0	83.40	8119	2821.252346
2	Algeria	DZA	8/21/20	1418	Middle East & North Africa	Lower middle income	Algeria	43851043.0	32.34	40667	927.389572
3	Andorra	AND	8/21/20	53	Europe & Central Asia	High income	Andorra	77265.0	685.95	1045	13524.881900
4	Angola	AGO	8/21/20	94	Sub-Saharan Africa	Lower middle income	Angola	32866268.0	2.86	2068	62.921656
...	...	...	...	...	...	...	...	...	...	...	...
179	Palestine, State of	PSE	8/21/20	122	Middle East & North Africa	Lower middle income	West Bank and Gaza	5101416.0	23.91	18313	3589.787620
180	Western Sahara	ESH	8/21/20	1	Sub-Saharan Africa	Lower middle income	Western Sahara	597330.0	1.67	10	16.741165
181	Yemen	YEM	8/21/20	542	Middle East & North Africa	Low income	Yemen	29825968.0	18.17	1906	63.904045
182	Zambia	ZMB	8/21/20	277	Sub-Saharan Africa	Lower middle income	Zambia	18383956.0	15.07	10627	578.058390
183	Zimbabwe	ZWE	8/21/20	152	Sub-Saharan Africa	Lower middle income	Zimbabwe	14862927.0	10.23	5815	391.241914

The next scatter plot shows 3 indicators: number of cases (axis x), number of deaths (axis y) and number of deaths per million population (size of the bubble).

```
In [106]: 1 # log(Cases) correlation with log(Deaths), logarithmic plot
2 # Cases vs Deaths
3 import plotly.express as px
4 df = data_all_nonulls
5 fig = px.scatter(df, x=df['Cases'], y=df['Deaths'],
6                 color='Region', size=round(df['Cases per mln'],1),
7                 hover_data=['Country'],
8                 labels={'x':'Cases', 'y':'Deaths', 'size':'Cases per mln'})
9
10
11 fig.layout
12 fig.update_layout(xaxis_type="log", yaxis_type="log",
13                  height=600, width=1000,
14                  title_text="Correlation between cases and deaths",
15                  showlegend=True,
16                  legend=dict(yanchor="top",
17                             y=0.96,
18                             xanchor="left", x=0.02),
19                  plot_bgcolor = 'whitesmoke'
20                  )
```

Correlation between cases and deaths



In [ ]:

```
In [107]: 1 print("Q: Deaths correlate with Deaths per mln?")
2 print("A: Yes, but positive correlation is weak and insignificant, r = 0.44")
3 r = data_all_nonulls.Deaths.corr(data_all_nonulls['Deaths per mln'], method = "pearson")
```

Q: Deaths correlate with Deaths per mln?

A: Yes, but positive correlation is weak and insignificant, r = 0.44

Out[107]: 0.44



```
In [108]: 1 print("Q: log(Deaths) correlate with log(Deaths per mln)?")
2 print("A: Yes, positive correlation is quite strong, r = 0.7")
3 r = np.log(data_all_nonulls.Deaths).corr(np.log(data_all_nonulls['Deaths per mln']),method = "pearson")
```

Q: log(Deaths) correlate with log(Deaths per mln)?  
A: Yes, positive correlation is quite strong, r = 0.7

Out[108]: 0.68

### Correlation analysis: Cases, Deaths and Response

```
In [109]: 1 #Creating dataframe on maximum government response index
2 resp = pd.DataFrame(df_iGovResp.max(axis=1))
3 ind = pd.DataFrame(df_iGovResp.CountryCode)
4 Response = (resp.merge(ind,
5                         left_index=True,
6                         right_index=True)).rename(columns={"CountryCode": "ISO-3",
7                                                         0: "Response"})
```

(185, 2)

```
In [110]: 1 # Merging maximum government response index into data_all, ignoring 0 deaths
2 data_all_nonulls = data_all[data_all.Deaths != 0] #filter out 12 countries not reported deaths
3 data_all_nonulls = data_all_nonulls.merge(Response, on = "ISO-3")
```

(158, 12)

```
In [111]: 1 print("Q: Do total cumulative deaths correlate with maximum government response?")
2 print("A: No, r = 0.2")
3 r = (np.log(data_all_nonulls.Deaths).corr(data_all_nonulls.Response, method = "pearson"))
```

Q: Do total cumulative deaths correlate with maximum government response?  
A: No, r = 0.2

Out[111]: 0.19

Total deaths are not correlated to maximum government response, as many countries started to take measures before epidemic led to deaths number growth in order to prevent the losses, and many of them keep high level of restrictions after the deaths number starts to decrease.

```
In [112]: 1 print("Q: Maximum deaths per mln correlate with maximum government response index?")
2 print("A: No, r = 0.1")
```

Q: Maximum deaths per mln correlate with maximum government response index?  
A: No, r = 0.1

Out[112]: 0.12

```
In [113]: 1 print("Q: Total cumulative cases correlate with maximum government response?")
2 print("A: No, r = 0.3")
3 r = (np.log(data_all_nonulls['Cases']).corr(data_all_nonulls.Response, method = "pearson"))
```

Q: Total cumulative cases correlate with maximum government response?  
A: No, r = 0.3

Out[113]: 0.26

### Correlation analysis: government response and government health expenditure

```
In [114]: 1 # Merging Response and Government health expenditure per capita 2017, international $
2 import world_bank_data as wb
3 df_govHealth = pd.DataFrame(wb.get_series('SH.XPD.GHED.PP.CD',date='2017',id_or_value="id",simplify_index=True).dropna())
4 df_govHealth['ISO-3'] = df_govHealth.index
5 Response_Funding = Response.merge(df_govHealth, on = 'ISO-3')
```

```
In [116]: 1 print("Q: Does government response index correlate with government health expenditure?")
2 print("A: No, there is no correlation, r = 0.06")
```

Q: Does government response index correlate with government health expenditure?  
A: No, there is no correlation, r = 0.06  
0.06

### Correlation analysis: confirmed cases and government response index over time

```
In [92]: 1 # log(Cases) correlation with Response over time
2 #RESPONSE Jan 22-Aug 16
3 y1[21:] #ca_iGovResp Jan 22-Aug 16
4 y2[21:] #us_iGovResp Jan 22-Aug 16
5 y3[21:] #cn_iGovResp Jan 22-Aug 16
6 y4[21:] #br_iGovResp Jan 22-Aug 16
7 #y5[21:] #wld_iGovResp Jan 22-Aug 16
8 y6[21:] #ru_iGovResp Jan 22-Aug 16
9 #print(len(y1[21:]))
10
11 #CASES Jan 22-Aug 16
12 cases_part = pd.read_csv("df_cases.csv")
13 #print(len(cases_part.Date.unique()))
14 x1 = np.array(cases_part.Value[cases_part['ISO-3']=='CAN'])
15 x2 = np.array(cases_part.Value[cases_part['ISO-3']=='USA'])
16 x3 = np.array(cases_part.Value[cases_part['ISO-3']=='CHN'])
17 x4 = np.array(cases_part.Value[cases_part['ISO-3']=='BRA'])
```

```

18 #x5 = np.array(cases_part.groupby('Date')['Value'].mean()) #world's average, !sorted
In [93]: 1 Canada = pd.DataFrame(x1,y1[21:])
2 Canada['Response'] = Canada.index
3 Canada['Country'] = "Canada"
4 Canada = Canada.rename(columns={0: "Cases"}).reset_index(drop=True)
5
6 USA = pd.DataFrame(x2,y2[21:])
7 USA['Response'] = USA.index
8 USA['Country'] = "USA"
9 USA = USA.rename(columns={0: "Cases"}).reset_index(drop=True)
10
11 China = pd.DataFrame(x3,y3[21:])
12 China['Response'] = China.index
13 China['Country'] = "China"
14 China = China.rename(columns={0: "Cases"}).reset_index(drop=True)
15
16 Brazil = pd.DataFrame(x4,y4[21:])
17 Brazil['Response'] = Brazil.index
18 Brazil['Country'] = "Brazil"
19 Brazil = Brazil.rename(columns={0: "Cases"}).reset_index(drop=True)
20
21 Russia = pd.DataFrame(x6,y6[21:])
22 Russia['Response'] = Russia.index
23 Russia['Country'] = 'Russia'

```

To find HEX codes for this colors (Canada 'leaf green', USA 'royalblue', China 'mediumturquoise', Brazil 'pink', Russia 'tomato'), view the list of colors from matplotlib:

```

In [94]: 1 import matplotlib
2 colorname = []
3 colorid = []
4
5 for name, hex in matplotlib.colors.cnames.items():
6     colorname.append(name)
7     colorid.append(hex)
8 zippedcolors = list(zip(colorname, colorid))
9 zippedcolors = sorted(zippedcolors, key=lambda x: x[1])

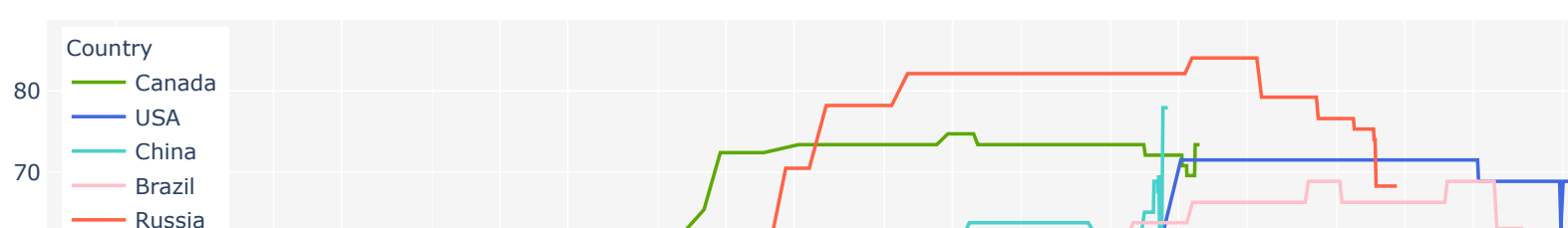
```

```

In [95]: 1 # log(Cases) correlation with (Government response index) over time
2 import plotly.express as px
3 from plotly.subplots import make_subplots
4 import plotly.graph_objects as go
5
6 fig = make_subplots(rows=1, cols=1)
7
8 fig.append_trace(go.Scatter(x = Canada['Cases'],
9                             y = Canada['Response'], name="Canada",
10                             mode="lines", line=dict(color='#5ca904')),
11                  row=1, col=1)
12
13 fig.append_trace(go.Scatter(x = USA['Cases'],
14                             y = USA['Response'], name="USA",
15                             mode="lines", line=dict(color='#4169E1')),
16                  row=1, col=1)
17
18 fig.append_trace(go.Scatter(x = China['Cases'],
19                             y = China['Response'], name="China",
20                             mode="lines", line=dict(color='#48D1CC')),
21                  row=1, col=1)
22
23 fig.append_trace(go.Scatter(x = Brazil['Cases'],
24                             y = Brazil['Response'], name="Brazil",
25                             mode="lines", line=dict(color='#FFC0CB')),
26                  row=1, col=1)
27
28 fig.append_trace(go.Scatter(x = Russia['Cases'],
29                             y = Russia['Response'], name="Russia",
30                             mode="lines", line=dict(color='#FF6347')),
31                  row=1, col=1)
32
33 fig.layout
34 fig.update_layout(xaxis_type="log", height=600, width=1000,
35                   title_text="Correlation of cases and response",
36                   showlegend=True,
37                   legend=dict(yanchor="top",
38                               y=0.99,
39                               xanchor="left", x=0.01),
40                   legend_title_text='Country',
41                   plot_bgcolor = 'whitesmoke'
42
43 )

```

Correlation of cases and response



## Conclusion

### Research questions

- is there a significant correlation between cases and deaths from COVID-19?
- is there a significant correlation between government response index and government funding healthcare in previous years?
- is there strong positive correlation between confirmed COVID-19 cases and government response index over time?

### Research results

- Yes, positive correlation between confirmed cases and deaths over time is very strong (coefficient is equal to 0.93).
- No, there is no correlation between government response index and government funding healthcare in previous years. The governments that show higher response index to COVID-19 in 2020 were not necessarily better in funding their healthcare systems in previous years. For example, China response was the most strict and effective, Russia respond is strict but delayed, but neither China no Russia have no history of government health expenditure at least at world's average level in previous years.
- Correlation between confirmed cases over time and government response over time is very strong (coefficient is equal to 0.94). But correlation between total cumulative number of cases and maximum government response is very weak (0.26), and correlation between deaths and maximum government response is even weaker (0.19). It means, that time factor is crucial to government response in order for measures taken have positive effect on the situation with virus spread.

I have confirmed all 3 hypothesis:

I found out that positive correlation between cases of COVID-19 and deaths from it is very strong, as well as between confirmed cases and government response. Government response index and government healthcare expenditure in previous years are not correlated. These findings are relevant for different regions, but their relevancy for the lower middle income countries and low income countries is not fully confirmed yet due to limitations of the dataset. The code I wrote could be used for daily motinoring of the situation and government response to it.

It is too early to make final conclusions about correlations of COVID mortality for different regions or income levels as disease statistics database is only forming and growing, and information available now is incomplete.

The indicators that we could use to describe the situation unfolding now are work in progress.

In [ ]: