Data Visualisation

Milestone 3 Report

Rayane Laraki, Stanislas Jouven, Olivier Lam

May 2020

1 Genesis of our work

1.1 Problematic

The COVID-19 virus significantly impacts our daily life for months and naturally questions arise:

- Where and how fast the virus spreads among the world?
- What were the government measures to face the virus?
- How government decisions impacted the spread of the virus?

In the COVID-19 era, the world is looking for answers to those questions. Even if many visualizations are already set up, our approach to tackle the problem stand out. Indeed, most of the visualizations found on the web answer to one specific question without giving a general overview. Furthermore, showing insights of the repercussions of politic countermeasures on the general spread of the virus and also compare the efficacity of different politics on the evolution of the pandemic is an original aspect.

1.2 Research

After having determined the problematic we wanted to tackle, we first have made a work of research about existing visualisations for our problematic to be able to identify and assert that our approach was meaningful and original.

We then researched about the datasets available that could be interesting for our idea. Several dataset kept at first our attention such as datasets from the European Union Open Data that shows some basic statistics, as the number of cases and the number of deaths per country and per continent as well as the total population for each country. But as time went by, more complete datasets were made available.

For example, we found an interesting one displaying the stringency index for each country throughout the evolution of the pandemic. The stringency index is a government measure tracker based on policy indicators such as school closures or movement restriction, record economic policy such as income support to citizens or provision of foreign and finally record health system policies such as the COVID-19 testing regime or emergency investments into healthcare.

Those researches finally led us to discover several tools for the visualization that we will talk more about later on (see section **3.1**).

2 Conceptualization

We finally decided to three datasets :

1. General Johns Hopkins Dataset (number of cases/deaths/recovered per day per country)
2. Oxford coronavirus governement response tracker
3. Testing dataset

On we have gathered our researches and selected our data, we created a visual overview of the project, organizing the material we have collected previously into a coherent whole. We brainstormed on a way to communicate effectively the relevant informations from the different datasets.

This resulted in the following wire-frame: the whole screen website is separated into two parts as shown in Figures 1, 2, 3. The map on the left part of the website plays a key role in the visualisation part. As it can be seen from the figures, on the right hand part of the website, plots in agreement with the map will be shown to the user. This plotting section has a scroll bar in order to add as many charts as needed.
The key thing in our visualisation is that if the user wants to know more about a country, the user has to click on it on the map and the information of that country will be added on the plots.

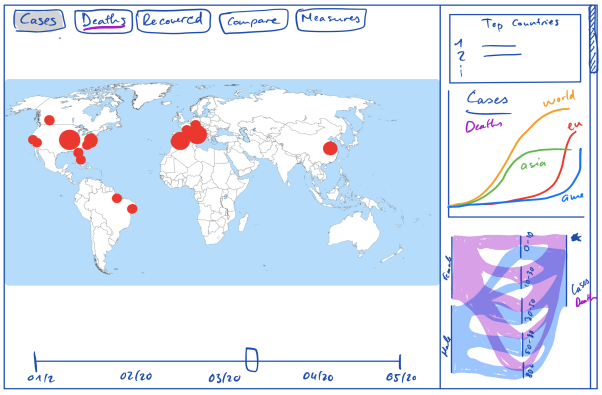

Figure 1: Visualisation for COVID-19 cases

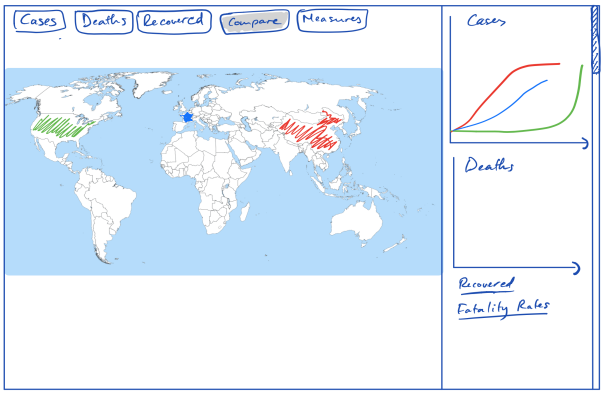

Figure 2: Visualisation for the comparison of data of the countries affected by the COVID-19

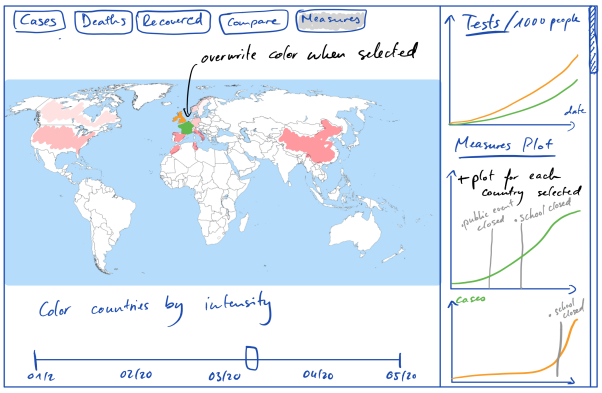

Figure 3: Visualisation for the measures taken by the countries affected by the COVID-19

Furthermore the data and consequently the way the data is shown on the map can be changed by clicking on one of the 5 buttons:

- **Cases** which displays the number of cases per country on the map (represented by red dots, see Figure 1). On the plot section, we'll display visualisation of the most affected countries, a line chart with the number of cases per continent and a Sankey diagram the cases according to the gender and the age.
- **Deaths** which displays the number of death per country in a similar fashion as the Cases map. Same plots as **Cases** are displayed in the corresponding section, but for the number of deaths.
- **Recovered** which the number of recovered people per country (represented by dots as in figure 1). Same plots as **Cases** and **Deaths** are displayed in the corresponding section, but for the number of recovered persons.
- **Compare** in which selected countries by the user are compared in the corresponding plot section. Selected countries are colored on the map (see Figure 2). For example clicking on the United States, France, China adds the green, blue and red lines respectively on the charts. On the plot section we will displays the cases, deaths, recovered for each date (line plots) and finally the fatality rate of the corona virus.
- **Measures** which displays a choropleth map to show the intensity of the government measures for each country (see Figure 3). Clicking on a country will this time overwrite the choropleth country color with a distinct color and will represent the color for that country on the charts. On the plot side, we will show the the number of tests conducted for each selected country(line plot) and then we will use line plots with annotation to show the impact of the mitigation measures on the number of cases.

3 Realization

The realization of the website, from the beginning to the final result was proceeded in multiple steps as we will see below.

3.1 Map

In a first instance, as the map plays a key role in the visualization of the website, our main priority was first to create the choropleth map representing the stringency index by country over the time.
Right after creating the map, the zoom feature has been included, then the slider showing the evolution of the results over time.
Then, we wanted to create the resize functionality which corresponds to resize the map when the user resizes the screen. At this point a first custom implementation design was about to be set up. Indeed, in order to have the dimension of the svg containing the map, we came up with the idea to regenerate the map with the new width and new height each time the user is generating a resize event.
Finally, a legend was added to complete the first created map.
After that, the bubble map followed up in order to express the number of deaths, confirmed cases and recovered cases. The creation of the circles representing the bubbles made things more challenges since each bubble represents a *circle* tag and thus additive reflection was needed when implementing the zoom or the slider. Indeed, once zooming every circles needed to stay in the same geographic position. Furthermore, once zooming the circles should be scaled proportionally with the scaling factor. When the user is zooming, a lot of call to the zoom function is done by the computer creating a lags. In order to correct that unwanted behaviour, we implemented a threshold and only when the zoom scaling factor exceeds this threshold the computer recomputes the radius of every drawn circles.
At this point the implementation of the maps by themselves were nearly done. However, before going further on the development of the project we took a step back and started to brainstorm on the best way to load the data and move from one map to another. With the general concensus of the team, we restructured our code as follow:

- The whole data used to generate the maps is stored in an array when launching the website. This choice was motivated by having the most fluent transitions when the website is running despite a longer launching time and more memory used.
- Concerning the structure of the implemented maps, in order to be as modular as possible, we decided to have a superclass called *MapPlot* which *MapChoropleth* and *MapBubble* extend from it, as shown below figure 4

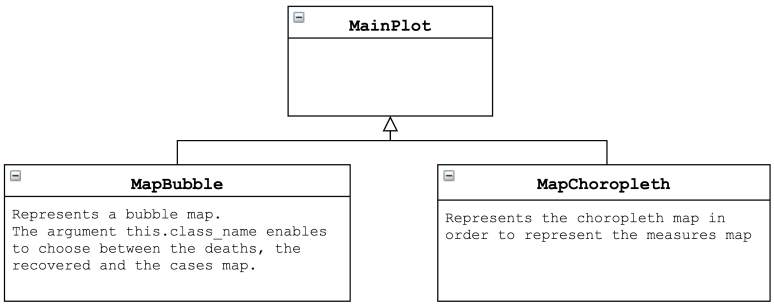

Figure 4: Map classes structure

3.2 Scrolling part

As mentioned earlier, we decided to use a scrolling div in which we display the different generated plots. We made that choice because in the Measures part, a plot is created every time the user selects a country on the map. This allows him to assess or not the effectiveness of measures taken on the decrease of the number of cases.

A plot is created for each country to be able to compare the different timelines of measures and effectiveness of those in different countries.

At first, we planned to use libraries based on D3.js in order to generate these plots. We began with Britecharts for the line plots and D3plus for the sankey diagrams but we struggled a lot in order to make them interactive with the other visualizations. We decided to switch to Plotly for the line charts and got the same issues with Britecharts, despite noticing that it is very convenient to add interactive features with these libraries. Finally, we decided to implement every plot with D3.js from scratch, in order to allow us more freedom with data manipulation.

The class *linePlot* takes care of plotting the different line plots given the section we selected (Cases, Deaths, Recovered, Measures). Similarly to the classes for the Map part, the creation of an object of *linePlot* appends an svg to the scrolling div. It has a *draw*() function which parses the data given the class attribute class_name $\in \{Cases, Deaths, Recovered, Measures\}$ and creates the desired plot. It has a function *update*() which for the Cases, Deaths and Recovered adds a line in an existing plot corresponding to the data of a country, where this country is selected on the map. Deselecting it removes then the line from the plot. If there is no country selected, then the plot is removed from the scrolling div. Similarly, in order to create a plot for a given country, the user has to select that country. The (multi)-line plots are also interactive with the user. By passing the mouse over a circle or the line, the user gets precise information on the data. We also decided to implement an horizontal zoom in the purpose of giving a better user interface. Regarding this functionality, we first implemented the zoom on the whole svg but this approach didn't allow us to slide on the right-scroll plot as we were "stuck" in the plots. We therefore restricted the horizontal zoom area to only the x-axis of each plot.

Concerning the government decision making data, we used the already implemented *linePlot* class. We want to show which decisions from the government have been taken at which stage of the outbreak. We plot the curves of the number of cases for the countries selected and add circles, which represent any measures from the government. All the data for the line plots are pre-processed using python, where we generated .json files.

To give to the user information for each country about the ratio of male/female cases and male/female deaths, we created sankey diagram. Similarly to the linePlot behavior, each time the user clicks on a country, it generates its corresponding sankey diagram.

3.3 Interaction between maps and plots

Finally, the last implementation part is able to interactively select or deselect a country in each map, resulting in adding or deleting data of that clicked country onto the scrolling plots. Clicked countries can be easily noticed by their red stroke.
An example of that event can be noticed by looking at figures 5, 6. We observe from the first plot(figure 5), that only Russia has been selected, therefore only one curve has been drawn in the line plot. Then, clicking on France and USA yields the second plot (figure 6) where we observe three lines in the line plot corresponding to the three selected countries.

This task, representing one of our most interactive and important part of our project was considered as a challenging task for us.
Therefore, altogether, we made the decision to have a global variable *list_countries* initialized in *main.js* that keep track of the selected countries. Then, when a country is clicked, a *mouseclick* event is launched and the updated *list_countries* array enables to fetch the corresponding data. The scrolling plots then use that updated data to plot their usual chart with the updated data.

However it has to noticed that our current implementation differs from the original intent. Indeed, in the original plan, measures should have been interactive whereas deaths, recovered and cases map should have been static. Furthermore, another map named *compare* was intended to be clickable and fully interactive for four countries: France, Belgium, Germany and United States of America as explained in section 2. Instead, in our current implementation we have made the choice to clearly separate each type of data (measures, cases, recovered, deaths), each time by making the map interactive with the scrolling plots on the right hand side of the website.

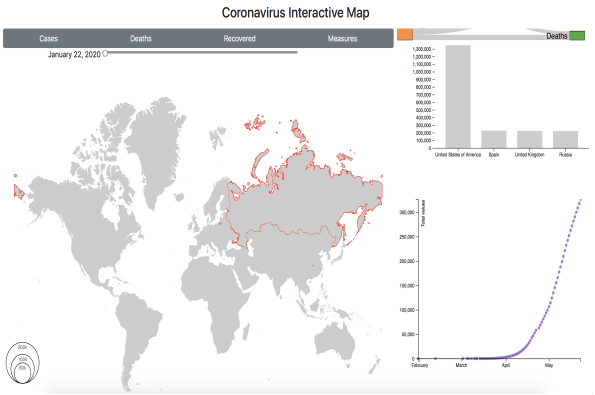

Figure 5: Visualization of the link between the map and the plots for the number of cases: Russia selected

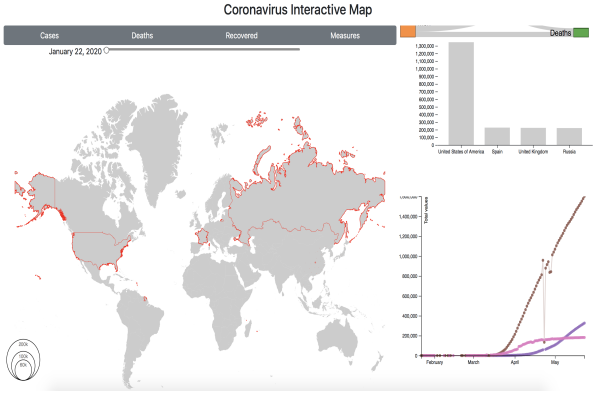

Figure 6: Visualization of the link between the map and the plots for the number of cases: Russia, France, USA selected

3.4 Design

Regarding the design, we first elaborated the zoning and wireframe of the website in function of the different visualisations we wanted to present. The main element is the map which shows 4 different types of informations following the buttons we clicked on. The other main part is the right scroll view that displays different plots as explained above. The resulting wireframe and zoning is shown in the mock-ups in section 2.1 (Figure 1, 2 and 3). Whenever the whole website, especially when the map and the scrolling part successfully worked, we decided to focus on the design.
Our first design choice was to add the same blue background image to the whole website. The color blue was chosen to be in total agreement with our central element which is the map as blue is the color of the ocean.
Then, in order to clearly represent text elements, axis and the map, a contrasting light grey color was chosen.
Thus our general theme is about mixing blue and light grey colors.
Then, using boostrap library, a darker grey was selected to represent the buttons. Since buttons are the only way to know in which map we are on, a darker grey color appears when the button is clicked.
Finally, colors for representing the maps were picked:

- Red associated with meaning of danger to represent the number of cases.
- Black, the color of deaths and fear to represent the number of deaths cases.
- Green, the color of life and renewal to represent the number of recovered cases.
- For the measures map, colors goes from light to darker green.

The impact of the design implementation on the final result(figure 7) depicts the newly and final version of our website which can be compared with the initial design (figure 6).

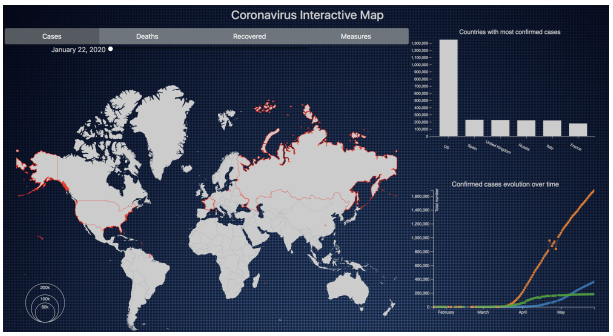

Figure 7: Visualization of the cases with the new design

4 Peer assessment

Our vision of work to create every project as follows: each part have been done by group of minimum two in order that if one get stuck by a bug for a too long time, the other one have the ability to give a second eye on the problem.
Because our knowledge in javascript were at the beginning limited, we worked together for every component of the project. Nevertheless, we can still enunciates some parts that were individually or dualy done:

- The map were mainly done by Stanislas Jouven and Rayane Laraki. For example, the resize function has been done by Rayane while for example the slider and the zooming functionnalities has been done Stanislas.
- The plots appearing in the scrolling part were mainly done by Olivier Lam.
- The design were totally made together with a shared screen (since our worked have been done in zoom)

References

[1] University of Oxford, Blavatnik school of governemt: Corona government response tracking
`https://www.bsg.ox.ac.uk/research/research-projects/coronavirus-government-response-tracker`

[2] Global health 5050: COVID 19 sex-disaggregated data tracker
`https://globalhealth5050.org/covid19/sex-disaggregated-data-tracker/`

[3] : Corona virus ressource center
`https://coronavirus.jhu.edu/map.html`