Data Visualisation and Analysis Report – Lab 3

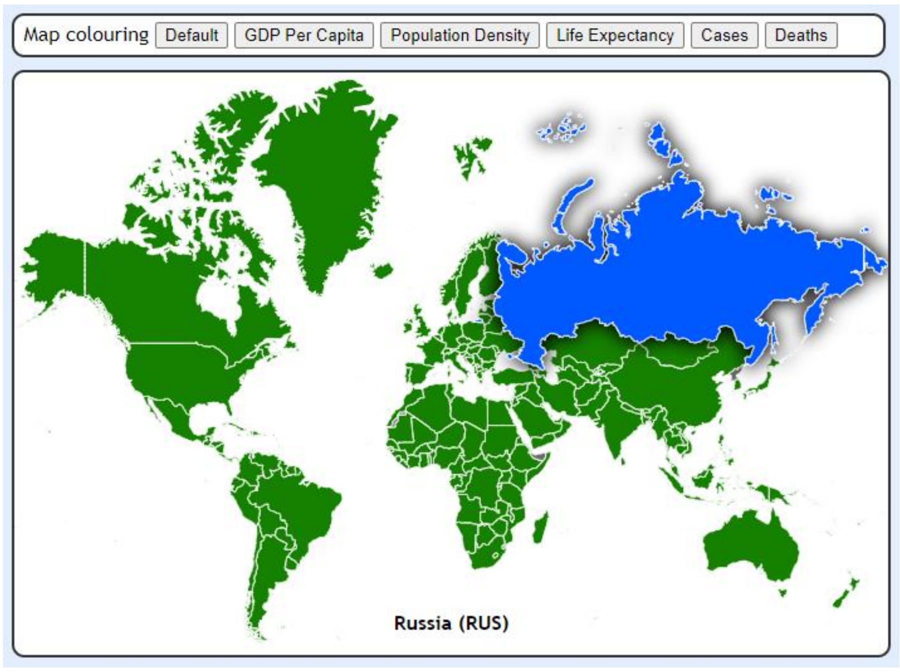
Christopher Murdoch, cm151

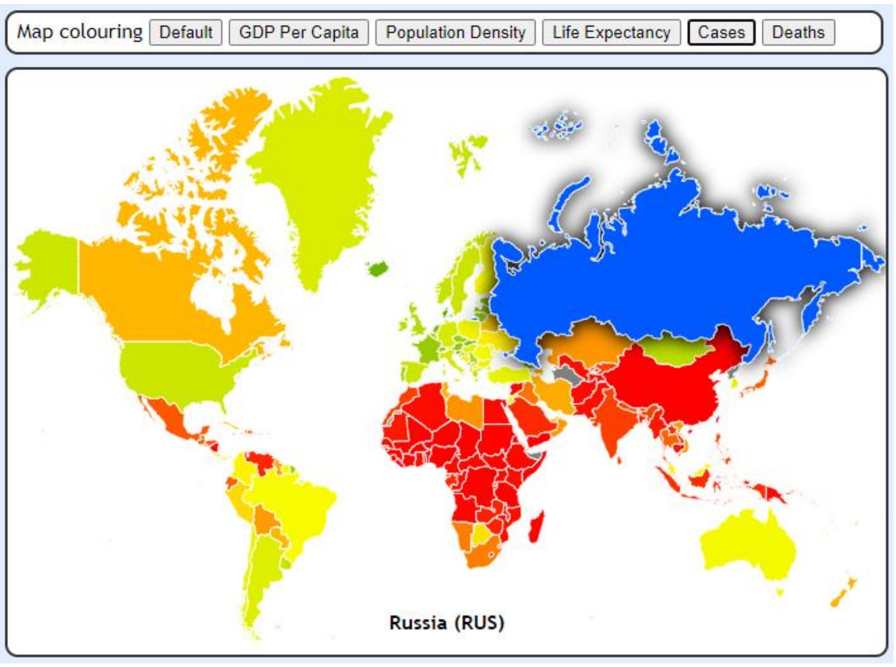
Demonstrated to Amit on 18/03/22

Map Layout

To select the region for which the user wants to view data, I have created an interactive map layout using d3 and a GeoJSON sourced from Natural Earth.

The map reacts to hover events and highlights a region in blue when clicked. Once clicked, the map class invokes a user-provided callback function with the relevant region's ISO code as the only argument. This allows the user to script custom behaviour for click events and help reduce interdependence between modules.



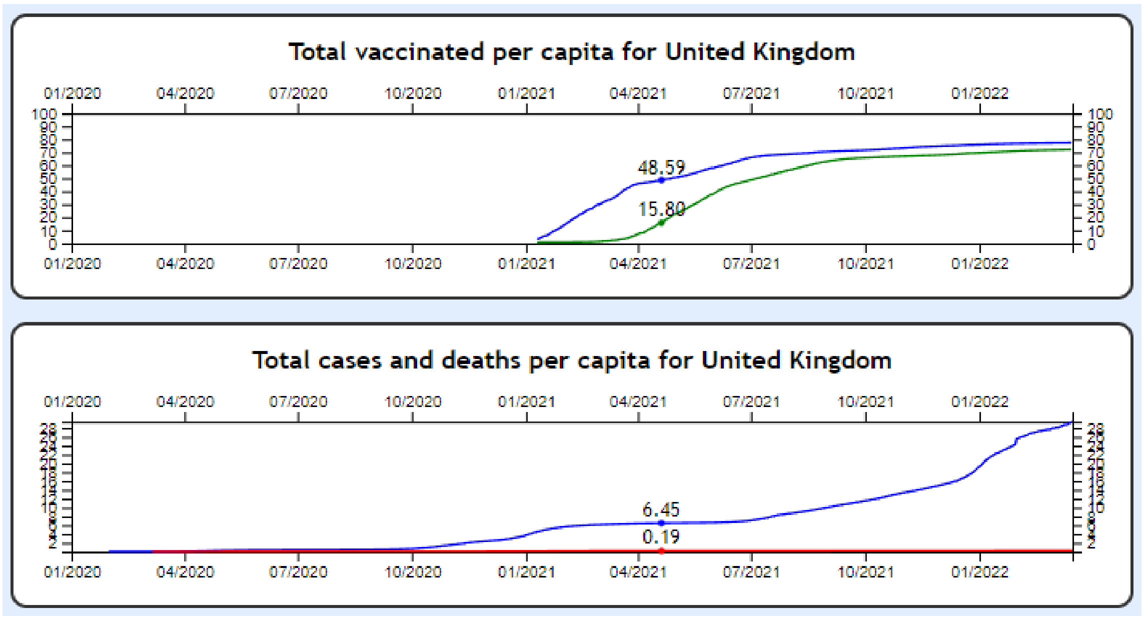


Another feature of the map is that the buttons along the top dynamically change how the map is coloured. Selecting *'Cases'*, for example, uses the number of cases per-capita to colour each region using a three-colour scale.

This dynamic colouring of regions helps to highlight the effect of Covid-19 on different regions of the world. The map also displays other factors such as GDP and population density.

One improvement that could be made is to include a legend for the colour scale used. Currently, regions can be comparatively evaluated by colour, but the true value (cases, deaths, gdp etc.) isn't visible.

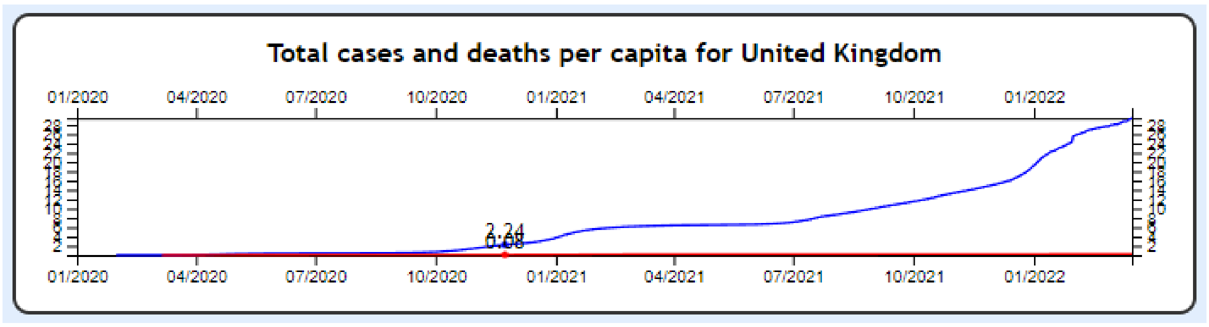
Regional Line Graphs



To illustrate the evolution of Covid-19 over time for a given region, I created a line graph class with mutable line data. The update call for these line graphs can be passed as the callback to the previously-demonstrated map class, allowing encapsulated, cross-layout interaction.

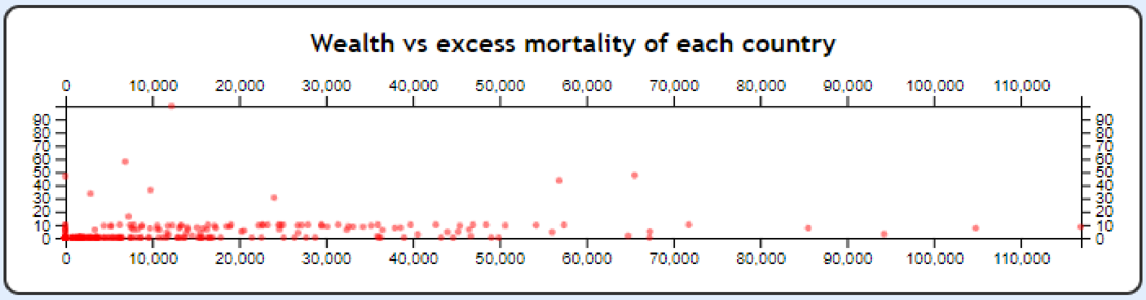
When a region on the map is clicked, the line graphs change to display the relevant vaccination and case data for that region. The new data fades in as the old data fades out and the axes adjust for the new data range.

Additionally, hovering over data points on any of the graphs will display the data for that data point, as well as any other corresponding datapoints on other graphs. The makes it easy to compare the vaccination rates and cases per capita for a given date.



An improvement that could be made to these graphs is the placement of textual data displays when hovering a datapoint. Currently, each text display is scripted to hover a set amount of space above the respective datapoint. However, this means that the text displays can overlap when multiple lines on a graph are close to each other. This spacing is an improvement I considered, but ultimately decided against in favour of other features to be implemented in the limited development time.

Inter-Regional Scatter Plots



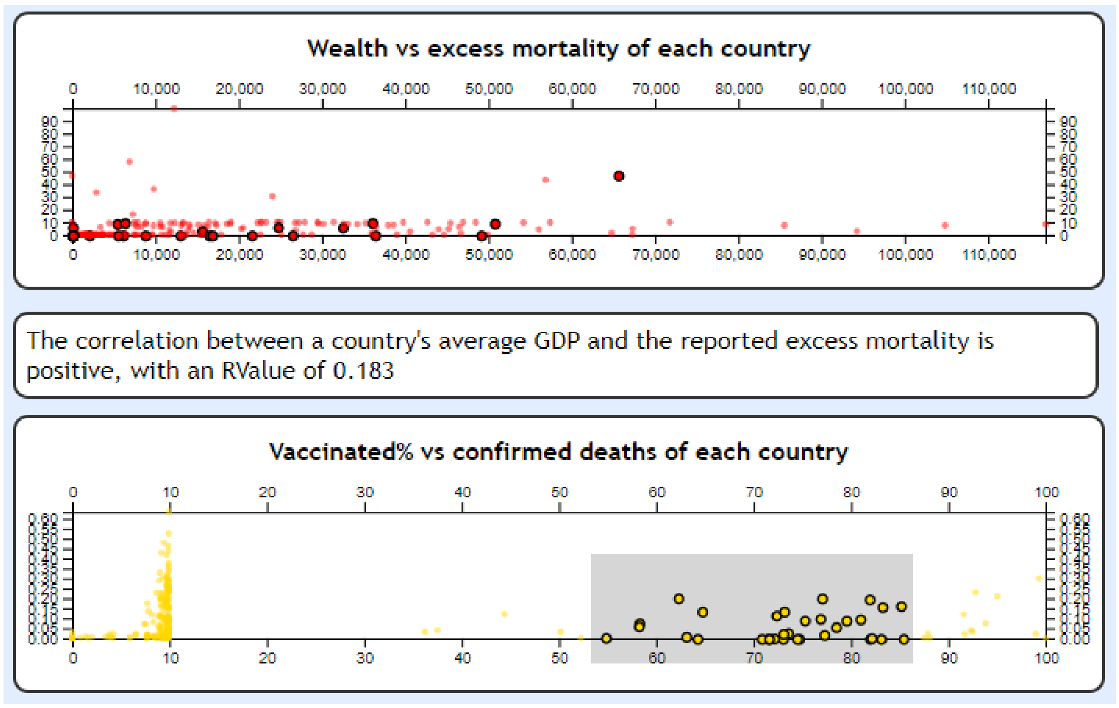
The correlation between a country's average GDP and the reported excess mortality is positive, with an RValue of 0.183

To compare the effects of Covid-19 on regions and their respective wealth, I created interactive scatter plots to display countries along different axes, depending on variables such as average GDP, cases, and excess mortality.

I also created a function to calculate the r value for sets of points in order to quantify the correlation between certain features.

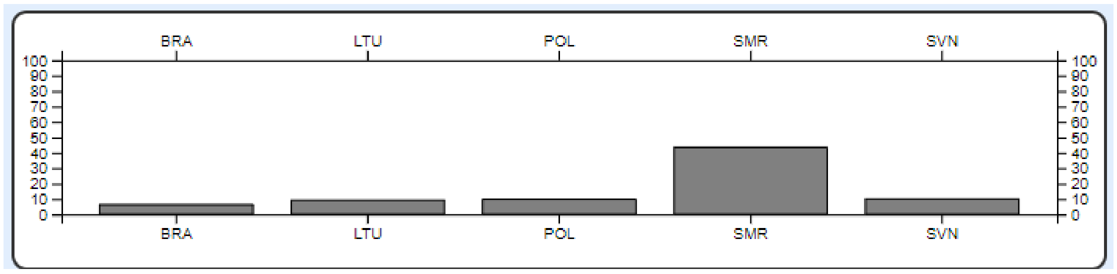
Oddly, r value analysis as well as visual analysis of the data actually suggests a positive correlation between the wealth of a region, and both their reported Covid-19 deaths and reported excess mortality. The key word here, however, is *reported*. When comparing a region's GDP and any reported value, the amount of reporting acts as a confounding variable. Naturally, regions with higher GDP have much more budget to spend on censuses and data infrastructure, resulting in higher amounts of reported deaths/excess mortality.

The calculated correlation between a region's vaccinations per capita and cases is negative, as expected. The R-value for this correlation is -0.13. Although not particularly strong, this correlation does suggest that vaccinations reduce the number of cases in a region.

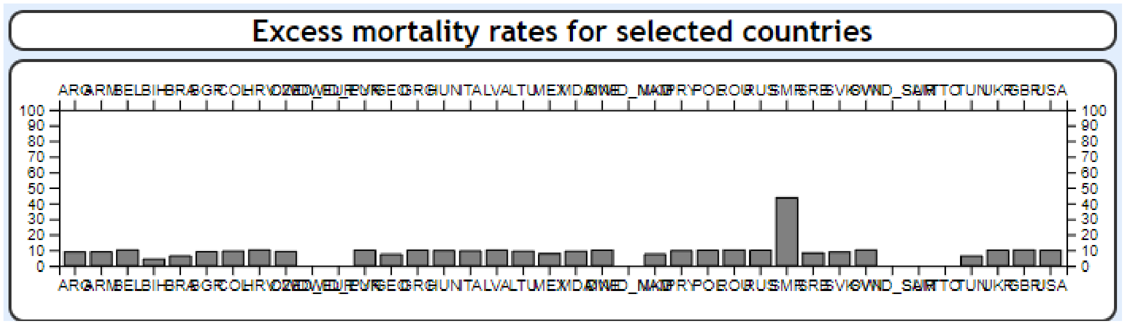

Wealth vs excess mortality of each country
Vaccinated% vs confirmed deaths of each country

The correlation between a country's average GDP and the reported excess mortality is positive, with an RValue of 0.183

Another feature of the scatter plots is that brushing over one highlights the relevant points in other plots. This helps to identify similar clusters between plots based on different features. For example, highlighting high-wealth, low-case regions in one plot may highlight primarily higher vaccination per capita regions in another plot.

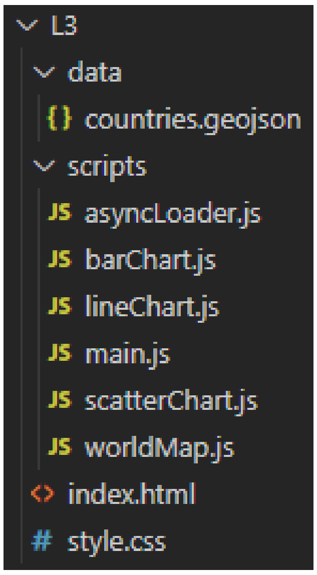


When a cluster of regions are brushed in one of the scatter plots, an additional bar chart displays the excess mortality for those regions. This can be configured by the user to display any data relevant to the regions, and helps to give a closer comparison between the regions' different variables.



An improvement that could be made to this section is the handling of many datapoints. Currently, selecting many datapoints results in overlapping text. This could be solved by widening the bar chart and adding side-scrolling functionality.

Code Design and Layout



```
L3
  data
    countries.geojson
  scripts
    asyncLoader.js
    barChart.js
    lineChart.js
    main.js
    scatterChart.js
    worldMap.js
  index.html
  style.css
```

The code for this project is organised similarly to the previous labs. The JavaScript for each type of chart are stored in classes, each with their own file under *scripts/*.

The geojson data is stored under *data/* and has been pushed to the public GitHub repository where it can be accessed with a raw resource URL.

As the website is quite small, the CSS is all stored in one file. If the CSS for the site became large and unmanageable, however, the code could be split up by function into multiple, smaller files.

This lab has been designed with OOP principles in mind. Each layout type is defined by a JavaScript class, and interaction between types is entirely encapsulated. If the user wants map region clicks to change data in a scatterplot, they only need to provide the scatterplot's data update function as a callback to the map's constructor. This means that different modules can be removed/interchanged without breaking another.

The dataset used in this lab is heavily enriched before use to provide optimal time complexity. Instead of keeping an unmanageably-large list of data records, each record is indexed by ISO code in a HashMap. This means that the data for a region can be accessed in constant time, without having to repeatedly filter the entire list in linear time.

Computed values are also added to the original data during enrichment, including peak excess mortality, latest reported deaths, and country-specific Covid-19 survival rates.

To improve user experience, all of the relevant Covid data is loaded asynchronously. A lot of data is used so loading it all at once helps to reduce the loading duration when entering the page for the first time. D3 also naturally caches the files used which improves loading time on subsequent visits to the page.