PROJECT: COVID-19 INFECTIONS

Introduction

In this project, we study a databas that contains a lot of information on the spread of the COVID-19 virus.

Part 1: Getting acquainted with the data

We first only study the covid cases day by day. Download the file `day_wise.csv` from Canvas and load into Python as a `DataFrame` using `pandas`. This file contains time-series data with information on the spread of the COVID-19 virus from January the 22nd to July the 27th in the year 2020.

Some graphical displays. Start by making line graphs of the number of new cases, the number of deaths and the number of recovered people over time. Make these figures in separate subplots and adequately label the axes.

Improve on the figure by implementing a function that takes a `Series` of dates and outputs the figure limited to these dates.
Hint: Use the functionalities from the library `datetime` *to easily compare dates*

The SIR model with deaths. Epidemics are often modelled using the SIR model. Here, a population is followed where individuals can be Susceptible (S), Infected (I) or Recovered (R). In our case, there is an extra state, namely Deceased (D). The transitions each individual can make are shown in Figure 1

$$\begin{tikzcd}
 & R \arrow[dl, "\alpha"'] & \\
S \arrow[r, "\beta"'] & I \arrow[u, "\gamma"'] \arrow[r, "\mu"] & D
\end{tikzcd}$$

Figure 1. Diagram of the SIR model with deaths

On each day, an individual can either stay in the state they are in or move to one of the adjacent states in the diagram. That is, someone who is infected can, die, recover or stay infected on each day and someone who is dead, stays dead. α, β, γ and μ represent the respective rates at which these transitions take place. On each day, the following changes can be observed:

$$\begin{aligned}
\Delta S(t) &= \alpha R(t) - \beta S(t)\frac{I(t)}{N} \\
\Delta I(t) &= \beta S(t)\frac{I(t)}{N} - \mu I(t) - \gamma I(t) \\
\Delta R(t) &= \gamma I(t) - \alpha R(t) \\
\Delta D(t) &= \mu I(t)
\end{aligned} \tag{1}$$

Observe that here $\Delta S(t)+\Delta I(t)+\Delta R(t)+\Delta D(t) = 0$, i.e. the total size of the population (including deaths) is assumed to stay constant. Choose arbitrary values for $\alpha, \beta, \gamma, \mu$ and numerically solve Equation (1) on each day of the dataset. That is, start with values $S(0), I(0), R(0), D(0)$ on day 0 of the dataset and for each day, obtain the new value through $S(t+1) = S(t) + \Delta S(t)$, etc.

Make figures for each of the components over time. Take for $I(0)$, $R(0)$ and $D(0)$ the values for `Active`, `Recovered` and `Deaths` on the first day of the dataset and take $S(0) = 17,000,000$ and $N = S(0) + I(0) + R(0) + D(0)$.

We have the following information on the parameters:

- An infected individual, on average, affects β susceptible individuals
- On average, individuals stay $1/\gamma$ days infected
- On average, individuals are susceptible $1/\alpha$ days after being recovered
- μ is the mortality rate of the virus

Think of ways to estimate these parameters based on the data. Which variables are useful for this task? Improve by updating the estimate each day. Also try to estimate the basic reproduction number, R_0, which is given by β/γ.

When finished, use your own creativity to discover any other features about this dataset you find interesting or noteworthy! Look for relations between variables of create attractive visualizations of the data. Points will be awarded for creative insights or features of the final product!

Create one `.py`-file which, upon execution produces all required figures

Part 2: Setting up a GitHub repository

The final product needs proper version control. Therefore everyone in the group will need an account on GitHub. Set up a private repository with your group and add a `.py`-file to it containing the code used for Part 1. Make sure to add a `README.md` file to it.

The `README.md`-file is very important as it contains information for users of the software you create. It should therefore serve as a guide for a use to use your library. `.md`-files can me opened and edited in any text editor, including Visual Studio Code.
Initialize `git` Take turns within your group and add small sections to the `README.md`-file each time. Between these turns `commit` and `push` the changes until you are satisfied with the file.Make sure to keep updating the `README` throughout the project, as this file will also be judged at the end.

Part 3: Interacting with the database

The database `covid_database.md` is now available on Canvas. This database contains the following tabular data:

- `country_wise`: contains the confirmed cases, deaths recovered, active cases etc. split per country
- `day_wise`: the file you have been working with for part 1.
- `usa_county_wise`: contains information on the cases per date and per us county
- `worldometer_data`: contains general information on the world population.

Use the library `sqlite3` to connect to the database. For each task of the remaining parts, it is most prudent to write `SQL`-queries to acces the parts of the database needed for that task, rather then loading it all into one or more `pd.DataFrame`s.

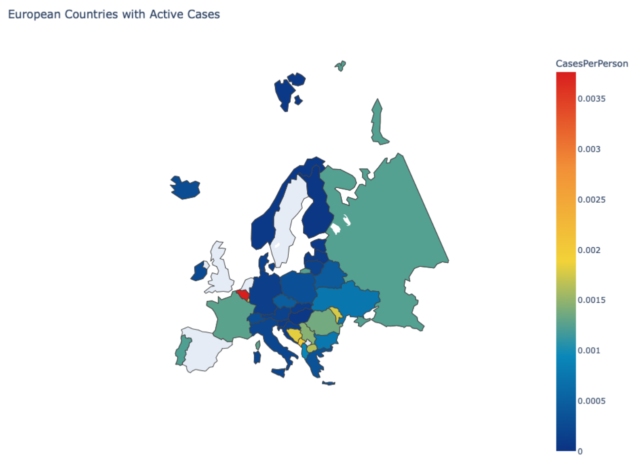
For this part, you have to investigate the entire database and link the different tabular data together. There is a lot to investigate. Below are some interesting tasks to complete with this database, but as with part 1, feel free to investigate further!

- Start with some visualizations. Make graphs similar to the ones you made in part 1, which now depict the total amount of active cases, deaths and recovered cases, all divided by the population of the country, on each date for a given country. For this task you may assume the population stays constant.
- Let us proceed with estimating the parameters. It is not hard to deduct from (1) that the death rate can be estimated on each day through $\hat{\mu}(t) = \Delta D(t)/I(t)$. The remaining system does not have a unique solution, as the equations are linearly dependent. Assume that it takes on average 4.5 days for an individual to recover after being infected. Then we have an estimate $\hat{\gamma}$ for γ. Plug in the estimates for γ and μ and derive formulas for estimates for α and β based on the information on day t. That is, derive formulas

$$\hat{\alpha}(t) = \dots \qquad \text{and} \qquad \hat{\beta}(t) = \dots .$$

  Make estimates for the four parameter for an arbitrarily chosen country using the data from `country_wise` and `worldometer_data`.
- I addition, write a function that, given a country, returns the trajectory of the R_0-value over time.
- Make a map of Europe in color-coded for the amount of `ActiveCases` divided by the `Population` in the table `worldometer_data`. You may want to use

the funcion `chloropleth` with the keyword `scope=‘europe’` from the library `plotly.express` for this purpose. Here is an example of a result



- Compare the estimated death rate (μ) over the different continents. Visualize this in a barplot
- Find the top 5 US counties with the most deaths and the most recorded cases for the duration of the dataset.

Part 4: Data wrangling

On canvas, there is one extra dataset made availlable: `complete.csv`. Download this dataset and use `sqlite3` to include it in the database. This table contains the day-wise counts split per country.

- First look for missing values and think of ways to resolve them.
- Next look for duplicate entries in the table.
- Now produce the figures required in the first task of part 3, now with the actual data per country instead of a simulation.
- The same can be done for individual US counties using the table `usa_county_wise`.
- The analyses made on the worldwide data can now be made split per country using groupby operations with the file `complete.csv`.
- Next week, the specifics of the dashboard will be released. Count on a requirement being that both graphical and numerical statistical summaries are displayed per country, county or continent as wel as summaries of the SIR model that models the disease, so aggregate the data as much as possible to this goal and gather useful summaries. In particular make sure that the functions you write take a country name as input, so that a dashboard user can simply click on a country and obtain results!
- The dashboard will also contain a general results tab that can show anything you find remarkable in the data. Think of things you would like to display here. Are there interesting relations to be seen in the data? Does the virus evolve differently

in different continents? Is the SIR model an accurate fit for the data? Come up with convincing graphical or numerical summaries that can be displayed in this part of the dashboard to display the results of the study.

Part 5: Creating a dashboard

Install `streamlit` and set up a script used to create the dashboard. You are free to choose the layout of the dashboard as well as the different options for navigation through it. Below, the minimal requirements are listed. The users of this dashboard will be the scientists analysts working for working at government agencies to analyse the pandemic.

Minimal dashboard requirements:

- An opening page with general statistics of the research carried out. This page should contain at least one graphical summary and one numerical summary of the data.
- The parameters of the SIR model should be incorporated in the dashboard.
- In the sidebar, there should be an option to select a country and see statistics of the pandemic for that particular country. These can be general summary statistics, a visualisation of the spread over time or both.
- The dashboard should have some time-component. That is, the data on some of the pages can be displayed limited to given dates, or times of day.

See the Assignment page for a detailed grading rubric of the assignment. A point will be deducted if the dashboard does not meet the minimal requirements. However, most points can be earned by including extra features using the research done in earlier parts of the project.

Part 6: Deployment and submission

Publish the repository. Go to your GitHub repository and go to settings. In the "Danger zone", you find the option to `Change visibility`, which can be used to set it to public. To check that it worked, try finding the repository in incognito mode, while not logged into GitHub.

Deploy the dashboard. Sign up in the Steamlit community cloud and connect your GitHub account following the steps in the slides of Lecture 7. Again, to see whether it worked, you, or a group member, can look for the dashboard incognito.

Submit the assignment. To submit the assignment, one of your group members must hand in two links:

- A link to the location of the dashboard.
- A link to the GitHub repository.

After submitting, the repository should not be changed anymore.

The group projects will be graded per group and not individually, unless the GitHub repository suggests the work is not distributed equally enough. In this case, your group will be contacted.