Vaccine Impact Modelling Tool

Joely Nelson (joelyn@cs.washington.edu)[1], Kenny Krivanek (kbk8@cs.washington.edu)[1], Joe Ammatelli (jamma@cs.washington.edu)[1], and Tevin Stanley (tevins@uw.edu)[1]

[1]*Paul G. Allen School of Computer Science & Engineering University of Washington, Seattle Washington*

ABSTRACT

In December 2020, the FDA issued the first emergency use authorization for the use of COVID-19 vaccines. A natural question that arose was the best way to administer vaccines to reduce severe pandemic outcomes. Although tools exist to display models or up-to-date COVID-19 data, few tools incorporate both. To help with this situation, we designed an interactive modeling tool which, given user input, models pandemic outcomes and displays them on a global map. The Vaccine Modeling Impact Tool allows users to select and adjust target parameters and useful data to visualize. They can view an interactive global map of the model's output with more data available on zooming and clicking. We designed the visualization front end, a data scraper to retrieve current COVID-19 data, a model to simulate pandemic outcomes, and a server to handle user requests and data flow. We aimed for our project to be user-friendly enough that someone with only basic public health knowledge could use our tool and could gain useful insights.

Author Keywords

Coronavirus, COVID-19, Mathematical Model, Vaccine Modeling, Vaccine Impact, Visualization Tool, Interactive Visualization, D3, Our World In Data

INTRODUCTION

In December 2020, the FDA issued the first emergency use authorization for the use of Covid-19 vaccines. Some countries began by restricting eligibility to vulnerable groups, whereas others had no access to the vaccine and had to wait. As time went on, eligibility requirements have changed for countries and more vaccines are being approved. Many people, from policy makers to the general public, have questions about vaccine allocation. What is the best allocation strategy to help prevent severe pandemic outcomes? How does vaccine efficacy affect the number of infections? What is the minimum percent of the population that needs to be vaccinated to ensure herd immunity? One way to answer this question is by using a model which can simulate pandemic outcomes. Although such tools currently exist, they might be overly complicated for the general public, be exclusive for a particular country or state, or lack up-to-date data.

DIS2020, July 06–10, 2020, Eindhoven, NL

ISBN 123-4567-24-567/08/06…$15.00

DOI: http://dx.doi.org/10.475/123_4

To contribute to the solution, we designed The Vaccine Modeling Impact Tool. This interactive modeling tool is hosted on a website that allows the user to select and adjust target parameters (such as the reproduction number, mortality rate, vaccination rate, and vaccine efficacy); and useful data to visualize (such as the number of individuals who have died or the number of current infected). When the user clicks simulate, they are then able to view an interactive global map of model output with more data available on zooming and clicking. The tool gathers data daily to display on the map and use as input for the model. We aimed for our tool to help policy makers use modeling tools to inform vaccine policy decisions, public health researchers to compare the efficacy of various different kinds of vaccine distribution methods and models, and the general public understand the likely future course vaccine distribution. We aimed for our project to be user-friendly enough that someone with only basic public health knowledge could use our tool and could gain useful insights.

We created several pieces for the backend and frontend and fit them together to create this project. We designed a simple deterministic SEIRD model to model the pandemic. This model is supplied with data from a data collection and parsing tool that aggregates data from various different sources and parses through it to find relevant data. We created a custom map that allows users to view pandemic outcomes on a global scale along with graphs to view the raw data. We designed a flexible frontend with parameter tuning that is formatted according to a configuration file. In this way, a user can easily tune model parameters, and new models can be added to the system in a straightforward way.

As a group, we learned many things by working on this project. We learned to balance the line between complexity and simplicity in the model; ensuring it was simple enough for a user to understand and complex enough to provide reasonable results. Visualizations can have many approaches, and knowing how broad or narrow you want to go is an acquired skill. Creating a framework through which to interface arbitrary models and parameter tuning schemes requires considerable consideration of trade offs between simplicity and customization.

In this paper, we begin by discussing related work by exploring existing tools, discussing what we took inspiration from and what we aimed to change. We then talk about our design process, first talking about our overall plan and then going into more depth regarding the individual components. We then detail our system architecture, discussing precisely how the individual pieces work in practice and how they come together. We then discuss takeaways for the project and areas for future improvement.

RELATED WORK

A number of tools meant to display pandemic related information and model the pandemic already exist. Most aim to either model different vaccination scenarios or display current data for a particular area. However, we found no existing tool that had the ability to scrape up-to-date data, show multiple regions at once, and the user to modify parameters and the model.

One tool we took inspiration from was the COVID-19 Scenario Analysis Tool, made by the MRC Centre for Global Infectious Disease Analysis, Imperial College London [9]. This interactive web page allows a user to choose a country and edit parameters, such as different reproductive numbers and vaccine allocation, then run a simulation that displays pandemic outcomes such as number of deaths, cases, and the healthcare demand. We took inspiration from the interactivity of this tool. Our work allows the user to change parameters and simulate the result of this by clicking a simulate button. However, in the COVID-19 Scenario Analysis Tool a user must select a specific country. Our work differs from this by allowing the user to model and view multiple countries at once on a map. As the COVID-19 Scenario Analysis Tool was mostly developed before vaccine roll-out got going, it requires most input regarding vaccination roll-out and efficacy to be provided by the user. We wanted to take the burden off of the user to find this data, and so our work scrapes up to date data regarding vaccination. The model provided in the COVID-19 Scenario Analysis Tool is also hard coded and cannot be switched out for another model, which was something we wanted to do to provide more flexibility for modelers.

Another tool we wanted to emulate were visualizations created by news outlets, such as that by the New York Times [2]. These interactive articles display maps containing up-to-date COVID vaccination related data for the United States, such as the current number of people vaccinated per county, the vaccine roll-out over time, and the current number allocation of vaccines by age group. We took inspiration from the clean and easy to parse visualizations in these articles, as well as their usefulness in displaying current data that is retrieved daily. However, these New York Times articles are often made for a specific area, typically the United States. We wanted our tool to not be exclusive to a single region or area. Although these articles may present projections, such as how many people are expected to be vaccinated by the end of the year, these articles have little interactivity from the user or predictive power. A user cannot play with parameters for vaccination and see how it changes. We wanted to combine the clean visualizations and up-to-date nature of these articles with an interactive tool.

Various research on pandemic vaccination models was conducted. We eventually landed on a custom deterministic SEIR (Suceptible, Exposed, Infected, Removed) model after looking into various other types of models. A deterministic model was chosen because it runs quickly when compared to non-deterministic types and is easier to understand. In addition, deterministic models that use systems of ordinary differential equations were familiar to one of the members on our team. Models that were considered included SEIR models that did not distinguish between recovered and dead [8], SEIR models that took virus-induced fatality rate as well as background death rate [5], a SEIRS model that took into account reinfections[1]. We combined the ideas behind these models to create a SEIR model that separated dead and recovered into two categories, but did not take into account background fatality rate. We also took inspiration from models that took into account age-structured models [6] and multiple vaccinations [10]. Modeling age, multiple vaccinations, and varying reproduction numbers for different times in the pandemic would have been more accurate, but given the short time and ambitious goal to model the globe instead of a single country, this was unfortunately was beyond the scope of the project.

More discussion of model research is discussed in the design section under models.

DESIGN PROCESS

Overview and System Goals

The design of the system revolved around an initial assessment of where a quantitative COVID-19 modeling tool would be especially useful. After researching existing products and brainstorming ways that the given problem space could be addressed, it was decided that a new modeling tool designed specifically for public health officials would prove valuable. In particular, such a tool would provide valuable data-driven insights and would improve upon on existing products that require a significant amount of epidemiology expertise, have a user interface that is complicated or verbose, and/or focus analysis on a smaller scale (country specific, state specific, etc.).

Supporting background information and insights were provided by Dr. Abie Flaxman of the Institute for Health Metrics and Evaluation (IHME) at the University of Washington. Dr. Flaxman introduced two pressing issues that would be exciting to incorporate into a modelling tool: (1) vaccine hesitancy and (2) global vaccine distribution.

Coupling the initial use case evaluation with the input from Dr. Flaxman, the following system goals were established:

1. Design a system that facilitates streamlined modeling of COVID-19 vaccination and pandemic outcomes
2. To the best of our ability, implement a placeholder model that models the role of vaccination on pandemic outcomes
3. Implement a simple and intuitive user interface for tuning the model and viewing the model output
4. Aggregate data from multiple reliable sources to streamline up-to-date modeling
5. Create a modular system that promotes ease of use and lets involved parties focus on their respective areas of expertise (modeling, interpretation of data, etc.)
6. Focus modeling on a global scale
7. Incorporate model tuning for at least one of vaccine hesitancy or global vaccine distribution

8. Design the system to be flexible; build a mechanism for easily adding/removing different models

9. Ensure the system is robust and efficient

Each component of the greater system was designed with these core goals in mind. Specific design processes are discussed in greater depth below.

Backend

Data Collection/Parsing

Any type of COVID-19 vaccination modeling tool requires some basic data in order to run; for example, vaccination rates, death rates, reproduction numbers, and others. Therefore, for our COVID-19 modeling tool, we composed a separate tool for gathering data from various sources in order to make it easier for model designers to gather the data necessary to run their models. This tool also provides important data necessary for display in the visualization tools and for the prototype/default model. Additionally, the tool does some data parsing in order to filter out the relevant data from these sources.

Data Collection

The primary data source used for this project is Oxford's Our World in Data (OWID) project. OWID has a collection of COVID-19 pandemic data useful for running any substantive model.[3] The data at the OWID project is in turn drawn from multiple sources, including John Hopkins University, the Center for Disease Control in the U.S. (CDC), and the World Health Organization (WHO). From this hub of sources collected by the OWID project, the following data was obtained: (1) the full data regarding COVID-19 death and case numbers for every country throughout the pandemic, in the file `full-data.csv`; (2) the full COVID-19 vaccination numbers for each country during the pandemic, in the file `vaccinations.csv`; (3) a file containing some of the most recent COVID-19 data for each country during the pandemic, in the file `owid-covid-latest.csv`; and (4) a list of the most common vaccines in many countries by their manufacturer (Pfizer, Moderna, etc.), in the file `vaccinations-by-manufacturer.csv`.

Meanwhile, one of the other relevant fields, population, was drawn from a different data hub called JohnSnowLabs, which contains the population data for each country over the last several decades which is turn obtained from UN sources, in the file `population-figures-by-country.csv` .[4] From both sources the data is obtained in the form of raw `.csv` files where each row represents a data for a given country. This data is retrieved on a daily basis, in order to obtain the most recent data for each country.

Data Parsing

In order to produce some data more useful for a general modeling tool, the parsing tool performs several tasks that make usage of the data easier. For example, in files that contain data reaching back to the beginning of the pandemic, the data parser can filter to keep only most recent data and output it into a separate `.csv` file. This is done for both weekly COVID-19 death and case numbers for each country during the pandemic, with the files `deaths.csv` and `cases.csv` respectively, as well as with population numbers, in the file `population-figures-by-country-parsed.csv`. It can also filter through `.csv` files with multiple irrelevant fields to obtain just the necessary data, which is done with the file `owid-covid-latest.csv` to obtain reproduction numbers stored in `r-values.csv`. Additionally, since the prototype model requires a daily average of vaccination numbers, the data parsing tool also calculates the daily average number of vaccinations over the last 7 days for each country, which are stored in `vaccinations-parsed.csv`.

Moreover, the data parsing tool is able to convert files between `.json` and `.csv` format. This is because in some instances - such as with the visualization tool and our prototype model - `.json` files may be more useful, while for other possible models it could be that `.csv` files are in fact more useful. This is done with all the parsed data files mentioned above. Additionally, the data parsing tool is able to create a full "master `.json`" file that stores all the necessary data for running the prototype model - this includes, on a country-by-country basis, the number of days after the pandemic that vaccinations start, the average efficacy of the vaccine, the daily vaccination rate, the mortality rate, the reproduction number, vaccination uptake percentage, the average number of days a person infected by or exposed to the virus, and the population of the country. This is stored in the file `master-json.json`.

Model

In order to create a presentable prototype, a simple model was created to model pandemic outcomes. A deterministic SEIRD model (a variation of the SEIR model) was chosen to be used to model the pandemic before vaccination. After vaccination a more complex version of the deterministic SEIRD model which beaks the population into 3 categories (unvaccinated and willing/able to be, unvaccinated but unwilling/unable to be, and vaccinated) is used to model both hesitancy and the introduction of vaccines.

Model Before Vaccinations Begin

Before vaccination begins, the pandemic is modeled using a deterministic SEIRD model. In an SEIRD model, an individual is considered being part of 1 of 5 states: Susceptible (S), Exposed (E), Infectious (I), Recovered (R), or Dead (D). The contact rate, β, controls the rate of spread. It represents the probability of transmitting disease between a susceptible and an infectious individual. The incubation rate, ε, is the rate at which exposed individuals become infectious. $\varepsilon = 1/e_d$, where e_d represents the average duration of the incubation period. Recovery rate, $\gamma = 1/i_d$, is determined by the average duration, i_d, of infection. The death rate, α, controls the rate at which infectious individuals die. $\alpha = \gamma \cdot m$ where m represents the average virus induced mortality rate (ie for every 1 infected person, m people die). β is related to the basic reproduction number (R_0) the average number of people that one infected person is likely to infect in a population without any immunity, as follows: $\beta = R_0 \cdot \gamma$. The model also requires N, the total number of people in the population.

The SEIRD equation can be described as a set of ordinary differential equations.

$$\frac{dS}{dt} = -\frac{\beta SI}{N} \tag{1}$$

$$\frac{dE}{dt} = \frac{\beta SI}{N} - \varepsilon E \tag{2}$$

$$\frac{dI}{dt} = \varepsilon E - \gamma I - \alpha I \tag{3}$$

$$\frac{dR}{dt} = \gamma I \tag{4}$$

$$\frac{dD}{dt} = \alpha I \tag{5}$$

This model was designed after considering several other types of SEIR models. The simplest SEIR models do not distinguish between dead and recovered, placing everyone after they became infected into a removed category[8]. Since we wanted to display the number of deaths, we needed some way to model the number of deaths so decided to create the two separate populations for dead and recovered. Other models did take deaths into account, but also included a background death rate in their models in addition to a virus induced fatality rate[5]. The background rate of death added an extra layer of complexity that did not seem necessary for the level of modeling we desired. A third variant briefly considered was the SEIRS, a model where there is a possibility of reinfection after a person recovers [1]. Current research reports that reinfections have been reported but appear to be very rare, but very little data has been collected on this [7] Although it could be playing a part in the current pandemic outcomes, it seems to be currently very small and was omitted from our models.

Initial reasonable values for the SEIRD model values β, ε, γ, α, were first gathered by trial and error. In order to do this, the model was run with different parameters for the USA. The predicted number of deaths from the model was compared to the actual recorded deaths from the Covid Tracking Project (https://covidtracking.com/data/download), which displays daily data from 1/13/2020 to 3/7/2021. After somewhat reasonable values were estimated, minimizing the residual sum of squares via gradient descent was used with these initial values to further fit the parameters.

The model is run for a number of days specified by the user. For each day it returns the number of individuals in each category for that day.

Model After Vaccinations Begin

When vaccination begins, a more complex model is utilized. This SEIRD variant splits the population into 3 groups: able/willing to be vaccinated, unable/unwilling to be vaccinated, and vaccinated. The vaccine is assumed to have some efficacy that prevents infection, and efficacy at preventing critical outcomes from the virus.

In this model, the population is modeled where an individual is considered being part of 1 of 13 states: $S_1, E_1, I_1, R_1, S_2, E_2, I_2, R_2, S_3, E_3, I_3, R_3, D$. S_1, E_1, I_1, R_1 states are for individuals who are Susceptible, Exposed, Infectious, Recovered respectively, and are unwilling/unable to be vaccinated. S_2, E_2, I_2, R_2 states are for individuals who are willing/able to be vaccinated. S_3, E_3, I_3, R_3 are states for individuals who are vaccinated. D is the number of all individuals who have died.

The SEIRD before vaccination model is run for a number of days equal to when the day vaccinations begin. At this point the values for S, E, I, R, D are taken and used to calculate the initial number of people in the categories $S_1, E_1, I_1, R_1, S_2, E_2, I_2, R_2$. Let u be the fraction of the population willing/able to be vaccinated. At this point, $S_1 = (1-u)S$, $E_1 = (1-u)E$, $I_1 = (1-u)I$, $R_1 = (1-u)R$, and $S_2 = (u)S$, $E_2 = (u)E$, $I_2 = (u)I$, $R_2 = (u)R$. Values for S_3, E_3, I_3, R_3 begin as 0 as vaccination has not yet begun.

S_2 individuals are vaccinated at a daily constant rate *vac_rate*, which represents the number of vaccinations per day. When an individual is vaccinated they move from S_2 to S_3. Individuals in the S_3 can be infected but are less likely to become so. The equations for this vaccinated group are as follows:

$$\frac{dS_3}{dt} = vac\_rate - \frac{\beta S_3 I}{N} * (1 - v_i) \tag{6}$$

$$\frac{dE_3}{dt} = \frac{\beta S_3 I}{N} * (1 - v_i) - \varepsilon E_3 \tag{7}$$

$$\frac{dI_3}{dt} = \varepsilon E_3 - \gamma I_3 - \alpha I_3 (1 - v_d) \tag{8}$$

$$\frac{dR_3}{dt} = \gamma I_3 \tag{9}$$

Where $I = I_1 + I_2 + I_3$, vac_rate is the number of individuals vaccinated per day, v_i is vaccine efficacy at preventing infection, and v_d is vaccine efficacy at preventing death. Using these two numbers has been seen in other models and based off evidence that vaccinations can prevent severe instances of the disease [10].

Frontend

Flexible IO and Customization

To build a flexible and interactive modeling framework (to satisfy goals 1, 5, and 7), several levels of tuning and customization were planned and incorporated.

Parameter Tuning

Public health officials require precise tuning of model parameters to test various pandemic scenarios. Accordingly, building frontend means for tuning parameters was deemed a necessary component of the minimum viable product. To simultaneously promote the maximum level of customization and ease of use, we elected to include ways to tune both global parameters (i.e. apply the same value to all countries) and local parameters (i.e. set a parameter for a particular country). We moreover elected to allow the user to choose between custom values and default values generated daily from real data (or model defaults in the event real data could not be found).

Considerable thought was dedicated to choosing the types of parameters the user should be able to tune. We questioned whether it would be more useful (and powerful) for the user to

explicitly adjust numeric model parameters (e.g. vaccine efficacy, rate of transmission, etc.), which would require more epidemiological knowledge yet would provide more fine grained tuning, or for the user to choose various qualitative scenarios for a given category (e.g. vaccine type, high/low/medium transmission rate, etc.), which would be more intuitive but potentially limit the space of possible tests. For the sake of the tool proof of concept, we elected to provide tuning for model parameters (which were designed to be intuitive). In doing so we provided a high level of customization and a more flexible dynamic between the front end and back end (discussed below).

Changing Models

Building a modeling tool around a single model limits the utility of the tool since every model makes different assumptions and generates different results; that is, different models provide different insights. Accordingly, it was important to create a flexible front end and back end that together accommodate different models.

To accomplish this goal, we prioritized a modular design in which the model in the back end conforms to a limited set of constraints. We moreover planned to create a front end that is sensitive to a collection of configuration files that reflect the given model. The modeler can specify which parameters can be tuned on the front end. Adding this functionality greatly expanded upon existing tools, which are generally specific to a single model (i.e. essentially just interactive visualization added to a model). As a consequence of adding this feature, however, we made it more difficult add scenario specific tuning, since the configuration is specifically intended for numerical parameter tuning.

Visualization

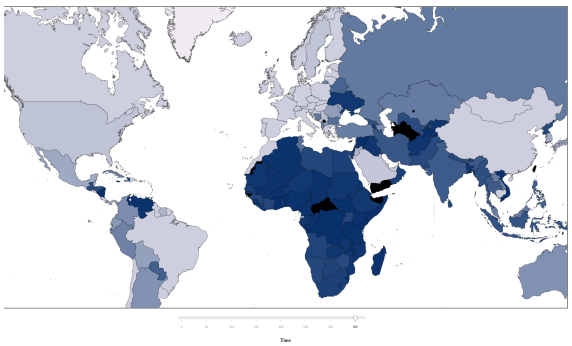

Figure 1. This is an image of map showing "susceptible" population data at Time (336) on tool

The overarching goal for this product was something very user friendly, globally focused, that intuitively shows all the data needed to make decisions about covid vaccines. With this in mind we found tools that are related and wanted to expand on that idea. Most pre-existing tools had lots of graphs and data, but we wanted it to also be accessible without looking at numbers, so we decided to include a world map as the primary source for viewing data. With the map being the main focus, we knew we still had to produce the same data that modelers would need available. We included a drop down menu to choose what outcome you wanted to view. Then we included a time slider to see the status of outcomes at a given time on the map. We also included a color scale to these maps so that although numbers are not shown, relative numbers can be interpreted from the colors shown. Each country having its own color scale (see figure 1.)

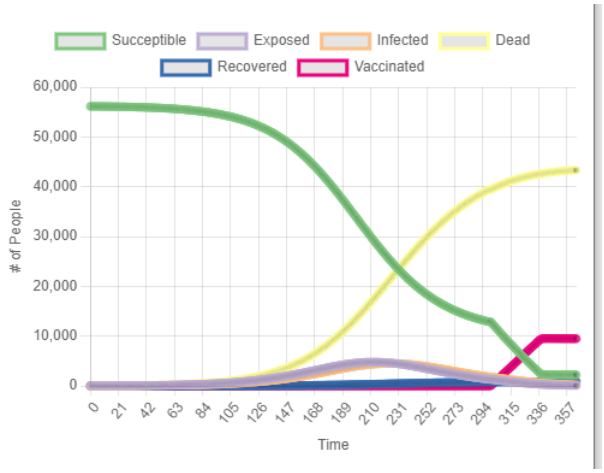

Figure 2. This is an image of the graph being shown for Greenland after it was clicked on in the map

On the data side we included current data available on mouse hover to ensure that the current data that's being shown on the map can be interpreted for the raw numbers, but we also included the functionality to get graphs on mouse click, to dive deeper into the data. This opens up a new window and allows user to view the graphs of the given country they clicked on.

SYSTEM ARCHITECTURE

System Overview

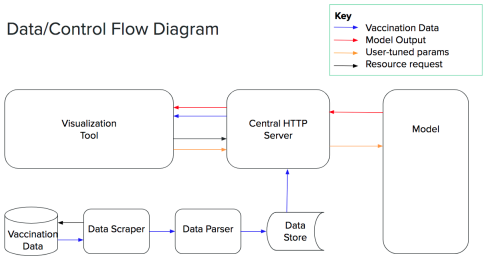

Figure 3. Flow of data across system components

The overall system is composed of 4 key components separated across the frontend and backend. The backend is composed of a central server that serves client requests for web resources and model output, a data aggregation and parsing tool that compiles vaccination data daily, and a modelling engine that generates output given certain parameters. The frontend is composed of a visualization tool that has a interactive map and parameter tuning panes.

The data flow for a given model simulation is shown in figure 3 and has the following sequence:

1. Once daily the data scraper fetches data from Our World In Data.

2. The data parser first formats the data, patches holes, and compiles all of the data into a 'master-json' file. The parser then adds modeller-specified variables and default values for each country based on a configuration file.

3. The master-json file is saved in a data store on the central server.

4. Once daily the default model output is updated with the new master-json file

5. When a user makes a connection to the web page, all of the page contents are returned by the central server. Additionally, a copy of the master-json file and default model output is served to front end.

6. The front end displays the default data and maintains a copy of the default parameters in the master-json.

7. The user tunes parameters as needed and clicks 'Simulate'

8. The customized parameters are sent as a POST request to the central server

9. The central server parses the json into a dictionary and calls the `simulate_world`function, passing the parameters

10. The model simulates the pandemic for the given number of days using the given parameters for each country and returns the results

11. The central server returns the model results to the front end

12. The frontend displays the new model output

Backend

Parsing and Obtaining Data

The retrieval of data from both the OWID project and John-SnowLabs is done by the file `retriever.py`. The file makes simple HTTP requests to each server through which the data for each file is obtained. This program also handles file I/O; essentially, the retriever tool also helps to transfer parsed/unparsed data in between files. After doing all data collection and parsing, the `retriever.py` tool waits for one day before running again.

The parsing itself is handled by the file `vax_data_parser.py`. The parsing of the population data, which obtains the population data for each country, is done by the function `population_parser`. Notably, the country of Eritrea doesn't have as recent of data as other countries, so the population data for Eritrea is slightly older. The daily average of vaccinations over the last 7 days is calculated by the function `vax_parser`. In the case where vaccination data is missing for one of the last 7 days for a given country, the average will be calculated only over the most recent days included in `vaccinations.csv`. Meanwhile, the most recent weekly COVID-19 death and case numbers for each country are obtained by the function `cases_deaths_parser`, and the reproduction numbers for each country are obtained by the function `r_values_parser`.

The file `json_maker.py` contains a number of useful functions related to `.json` files. For converting between `.csv` and `.json` files, the fuction `make_json` can create a `.json` file by mapping any column in a `.csv` file to a separate column in the file. There is also a function, `make_iso_json`, that compiles a file, `countries-to-iso.json`, which maps the country names in these files to their three-letter ISO codes. This is done in order to make sure that in each file, regardless of any different spellings of country names, the data for each country all ends up being stored in the same place in the overall `master-json.json` file, since ISO codes are always uniform. Since the `full-data.csv` file does not contain ISO codes, a few individual ISO codes had to be hard-coded in due to some different spellings in that file, specifically for the country Micronesia. Finally, the `make_master_json` function compiles the `master-json.json` file, which contains all the default data necessary for running the model and visualization tools. The data for vaccination rates, reproduction numbers, population, and mortality rates are obtained from the files `vaccinations.json`, `r-values.json`, `population-figures-by-country-json.json`, `cases.json`,and `deaths.json` (where mortality rates are found by dividing the weekly deaths from each country by the weekly cases). Where data is missing from these files for any country, default values are obtained from the file `global_defaults.json`,which could be modified by users. Since no data is obtained for vaccine efficacy, vaccine start date, average days infected or exposed, and vaccination uptake rate, the defaults in `global_defaults.json` will always be used for these fields. These values are each mapped in a nested dictionary to each country's ISO code (see the section Generating Model Results for the format).

Generating Model Results

Results from the model are retrieved by calling the `simulate_world` function in `models.py`. `simulate_world` takes in two arguments. The first is `param_dict` which is a nested dictionary that matches the format of the master-json file. The keys in the outer dictionary are country codes, and the values are dictionaries which have various keys (such as vaccination start date, vaccination efficacy, etc.) and the associated value for that country. Similar to the following format:

```
{"AGO": {"vac_start": 200, "vac_efficacy": 85,
       "vac_rate": "18902",
       "avg_days_in_exposed": 14,
       "r": "1.1", "vac_uptake": 80.1,
       "avg_days_in_infected": 14,
       "mortality": 2.0259740259740258,
       "population": "28813463"},
"LIE": {"vac_start": 200, ....},
...
}
```

The second parameter is `num_sim_days`, the number of days to simulate.

For each country, the `simulate_region` function is called, which uses the parameters provided for that country to run the model. `simulate_region` returns a vector called `v`.

`v[i][j]`corresponds to the number of individuals in category `j` on the `i`th day after simulation start. $j = 0$ for susceptible, $j = 1$ for exposed, $j = 2$ for infected, $j = 3$ for dead, $j = 4$ for recovered, $j = 5$ for vaccinated.

`simulate_world` returns a dictionary where the keys are the country codes, and the values are the vector returned by running `simulate_region` for that country.

Central Server

A central server facilitates client web requests and data exchanges, linking all of the system parts together. The server was build on top of the simple python `http.server` test server. We added a custom `\POST` path for parsing parameters passed from frontend and invoking the model on the backend. The server was not designed with load or security concerns in mind. Rather, it was designed to be a simple module for linking the parts together for proof of concept.

Frontend

Customizing Parameters

Tuning of parameters in the frontend is accomplished using a combination of html and javascript. The frontend maintains two sets of data structures storing model parameters and model output for each country. The first set of data structures is for all of the default data and remains unchanged throughout the modeling process. The second set of data structures is for all of the customized data and reflects changes the user makes to parameters using the UI. In both cases, the data structures map country name to various parameter/model output values.

On page open, the output corresponding to the default country parameters is displayed on the map and all of the custom parameters are set equal the default parameters. Whenever the user updates a parameter using the global parameter pane or the country specific tuning pane, the parameters in the custom parameter data structure are updated. Whenever the user clicks reset, the customized parameters are set equal to the default parameters. Because the default parameters have already been cached, the reset functionality is very efficient. Whenever the user clicks simulate, the customized parameters are passed to the server backend as a `.json` file using a `POST` request. The model output associated with those parameters is then passed back to the frontend where it is displayed. The `simResults` variable points to the model output that should be displayed on the map based on whether the user has customized any of the parameters.

Flexible Frontend

The parameter tuning radio/numeric input html elements in the frontend reflect a configuration file created by the modeler. In this way, the frontend can easily be modified when a new model is added, and a modeler can choose how a public health official interacts with their model.

On page load, the frontend loads the json configuration file and creates a new html element for each item in the file. Default values, ranges, and steps are assigned according to the configuration file. Each html element has a generic name "input" followed by a number. This number is used to reference the parameter being tuned. In the configuration file, the modeler specifies a mapping from the parameter name to the variable in the custom parameters data structure (same fields as the master-json) that should be updated when the parameter is tuned. The front end maintains a data structure mapping input number to master-json variable name so that changes for a html element changes are reflected in the custom parameter data structure. Event listeners are bound to each element, and on input change the number is parsed from the element name and the appropriate variable in the custom parameter data structure is updated.

Map Visualization using D3

The main code for the map uses javascript. The driving source of the map is a library called D3, which gives access to interactive maps. In D3 to draw the map we load in a `geojson` file that holds data for the entire world. This gives us the backbone of the map that we can build upon. The map has access to a variable called `simResults` which holds the values of the output from our model. We created an `update()` function which updates the map according to which time index and scenario we are currently in. The update evaluates the `simResults` variable and chooses a color based on the value relative to total population size to show for the given country (darker being higher percentage).

Slider is also built on d3 and calls the `update()` function when user changes position.

The map also has a `on_click()` function that will trigger a popup window. The graph that is shown on the popup window is built on a library chart.js that allows user friendly graphs. With this library we evaluate the `simResults` and pass the according data to the chart.js library to graph for the user.

The popup holds input for the user to choose values for given parameters for a particular country which is discussed more in the server module.

DISCUSSION

Take Aways

Whenever discussing models, the quote “All models are wrong, but some are useful” by George E. P. Box comes to mind. There are many different ways to model a pandemic, but not all are useful. If the model is too simple, then the results may be too inaccurate to be even considered as a benchmark. If the model is too complex, even though the results might be more accurate, it may be too complex for a user of our system to understand and use. We had to balance this while working on the model for our project, as we wanted the model to be somewhat accurate while being user friendly for someone who was not an expert in public health.

Creating a "one size fits all" generalized modeling framework is a real challenge. Balancing parameter tuning simplicity, minimal configuration, and support for a wide variety of parameters, models and scenarios is a very complex task that requires careful selection of certain trade offs. In the end, our system privileges highly granular customization at the cost of configuration complexity and the loss of scenario level of abstraction tuning.

Visualization is a hard task, there are often lots of libraries that do similar things and knowing which way to go is a tough decision. For something like d3 there is so much customization that it can also make it a lot harder to do simple things. Having many libraries at play it can also be difficult to get them to interact with each other the way you want them to.

Areas of Improvement

The models used are highly simplified and do not account for many important features of the pandemic or vaccinations. It assumes a uniform R_0 throughout the entire time period, which is not true to real life where the effect of policies such as social distancing causes the R_0 to vary. Older people and essential workers are more severely affected by Covid-19, but our model does not take different population subsets, divided by age or profession, into account. A model that took this into account would likely much more accurate and would allow the user to experiment with different allocations for different groups. The vaccine model assumes that the vaccine is a single dose that is effective right away, which is not true of COVID-19 vaccines. It also assumes that vaccines are distributed at a single daily rate, which does not reflect real world scenarios. Most countries begin their vaccination programs slow and speed up as they get more supply. Future work should either be work to improve the current model, or integrate one made by public health experts.

While we successfully implemented a degree of flexibility in the greater system, the mechanism for changing models is still somewhat cumbersome, and the types of models the system supports are limited. To address this in future iterations, it would be desirable to implement a page on the frontend and that enables model/tuning configuration so the user can bypass editing `.json` configuration files (less readable more mistake prone). Additionally, it would be powerful to implement a more generalized model interface so the tool can use models written in languages other than python. It would be ideal if the system supported a user optionally uploading their own model into framework for testing.

Additionally, certain segments of the frontend and backend could be optimized further to reduce browser burden and improve user experience. Currently the system will repeat simulation for a country even if no parameters have changed. It would improve the response time of the frontend if the system instead only repeated simulation for countries whose parameters had been changed by the user on the frontend. Additionally, country level simulation could be conducted in parallel in the backend to greatly improve the time it takes to run the model.

The map does not currently have a legend. It would be good to have a legend depict what each color on the map stands for, and what values are mapped to those colors, as the values are dynamic depending on the scenario outcome and what scale is applied for color.

Another area of improvement in terms of mapping is that the current mapping has a bit of lag, considerably because of the redrawing that is going on in terms of updating. Maybe there could be a different library or other functions of d3 that can speed up this process so the interactions with the graphs and sliders are seem less.

CONCLUSION

Through SEIRD models, data scraping online data sources, server side post and get requests, and JavaScript/HTML visualizations, we were able to create a tool that effectively allows public health officials to make informed decisions about COVID-19 vaccinations. Nevertheless there is lots of room to grow, as modeling can be better, speed can be improved, and general UI changes could be made. However, the skills we gained through learning new languages, different approaches to modeling, and different ways to represent data across multiple libraries and languages has been a great success across our team. We're hopeful that this tool can be used as a building block for future work in this field and improvements can be made to really make an impact on COVID and future viruses to come.

REFERENCES

[1] Shea K. Krzywinski M. et al Bjørnstad, O.N. 2020. The SEIRS model for infectious disease dynamics. *Nat Methods* (2020). DOI:http://dx.doi.org/https://doi.org/10.1038/s41592-020-0856-2

[2] Jasmine C. Lee Amy Schoenfeld Walker Lazaro Gamio Josh Holder Denise Lu Derek Watkins Adeel Hassan Jordan Allen Alex Lemonides Brillian Bao Elisha Brown Alyssa Burr Sarah Cahalan Matt Craig Yves De Jesus Brandon Dupre Benjamin Guggenheim Barbara Harvey Lauryn Higgins Alex Lim Alex Leeds Matthews Jaylynn Moffat-Mowatt Laney Pope Cierra S. Queen Natasha Rodriguez Jess Ruderman Alison Saldanha Brandon Thorp Kristine White Bonnie G. Wong Danielle Ivory, Mitch Smith, John Yoon. Data acquisition, Eleanor Lutz Bea Malsky Ilana Marcus additional work contributed by Avery Dews, Tiff Fehr, and Josh Williams. 2021. See How Vaccinations Are Going in Your County and State. (2021). https://www.nytimes.com/interactive/2020/us/covid-19-vaccine-doses.html

[3] Diana Beltekian Edouard Mathieu Joe Hasell Bobbie Macdonald Charlie Giattino Cameron Appel Lucas Rodés-Guirao Hannah Ritchie, Esteban Ortiz-Ospina and Max Roser. 2020. Coronavirus Pandemic (COVID-19). *Our World in Data* (2020). https://ourworldindata.org/coronavirus.

[4] JohnSnowLabs. 2018. Population Figures By Country. *JohnSnowLabs* (2018). https://datahub.io/JohnSnowLabs/population-figures-by-countryreadme.

[5] Claudio Bagaini José M. Carcione, Juan E. Santos and Jing Ba. 2020. A Simulation of a COVID-19 Epidemic Based on a Deterministic SEIR Model. *Front Public Health* (2020). DOI: http://dx.doi.org/10.3389/fpubh.2020.00230

[6] Mark Kimathi, Samuel Mwalili, Viona Ojiambo, and Duncan Kioi Gathungu. 2021. Age-structured model for

COVID-19: Effectiveness of social distancing and contact reduction in Kenya. *Infectious Disease Modelling* 6 (2021), 15–23. DOI: http://dx.doi.org/https://doi.org/10.1016/j.idm.2020.10.012

[7] Heidi Ledford. 2021. COVID reinfections are unusual — but could still help the virus to spread. *Nature Methods* (2021). DOI:http://dx.doi.org/https://doi.org/10.1038/d41586-021-00071-6

[8] Aidalina Mahmud and Poh Ying Lim. 2020. Applying the SEIR Model in Forecasting The COVID-19 Trend in Malaysia: A Preliminary Study. *medRxiv* (2020). DOI: http://dx.doi.org/10.1101/2020.04.14.20065607

[9] Imperial College London MRC Centre for Global Infectious Disease Analysis. 2021. COVID-19 Scenario Analysis Tool. (2021). https://www.covidsim.org/v4.20210322/?place=us

[10] Michael J Tildesley Louise Dyson Prof Matt J Keeling Sam Moore, Edward M Hill. 2021. Vaccination and non-pharmaceutical interventions for COVID-19: a mathematical modelling study. *The Lancet* (2021). DOI: http://dx.doi.org/https://doi.org/10.1016/S1473-3099(21)00143-2