Presentation Script by Jiwei Zhang

Here is my outline of today’s presentation. I would like to display all my graphs and give a brief description for each of them first. Then I will focus on the key process of each week and describe how I solve the difficulties during my work. These difficulties include how I clean up the dataset, how I change the ticks of axis, how I draw the doubling lines and how I deal with unexpected data.

Let us move to the first graph, the graph of world-wide confirmed cases. The y-axis is the total number of confirmed cases of Coronavirus and the x-axis represent the date. The key point of this graph is to make y-axis use log scale and to make the x-axis show date every 3 days.

The next graph is a combination of daily new confirmed cases and its average over 3 days. It combines line plot and bar plot. The x-axis is the same as the last one, but the y-axis remains the default one.

Then it is the week 4 graph, showing how the total confirmed number of each country perform since it reached 100. To be noticed, only selected countries’ data will be highlighted with its name, all the rest countries will be displayed in grey. We can choose which country to display whenever we want. This Graph’s y-axis is in log as well and several dashed lines called doubling lines are added to represent different increasing speeds of total number, which is the key feature of this graph. And I will cover it later in this presentation.

The last one shows daily new confirmed cases vs. total confirmed cases, which kind of the combination of previous graphs. The only thing remarkable about this graph is how I dealt with non-positive values.

These are all my graphs, then let us go back the week 2 and talk about my work in each week.

The first key process is how I cleaned up the dataset.

The raw dataset has columns Province/State, Country/Region, Lat, Long, and a sequence of dates. But our study is based on countries, so I dropped Latitude and Longitude first, then use groupby.sum() function to get the data for each country. To make it easier for future use, I transposed the dataset to list countries by columns. After I transfer dates into datetime object I found that it will cause problems when I use datetimeIndex as axis when I draw the graph, so I used strftime() function to transfer the date object to formatted string.

The second key process is how I changed axis ticks.

Because it has been a long time since the first datapoint, there are hundreds of dates in the dataset. If I want to list all of them, it will just look like the first graph. But what I want is the second. To achieve that, I used arange() in NumPy to generate an array of arithmetic sequence from 0 to the length of data with difference 3. If I set this array as x-axis ticks, the date will only be displayed every 3 days like the graph. Of course, only axis ticks are changed, no data is lost. For the y-axis, to make it in log is not difficult, but to make it show more ticks instead of only power of 10 cost me a lot of time. Finally, I decided to use major\_locator and major\_formatter functions so that it can show ‘n times power of 10’ like 200 and 5000. Also, because I asked the formatter to display as %d, the problem of using Scientific notation got solved as well.

The next topic is how did I generate the doubling line in a log graph.

This one is actually very similar to the one before, I started with generating a Natural Number array by Numpy arange() function as x values, then get the corresponding y values, then put them into the graph. Just like a linear function y = ax + b, here it becomes y = pow (2, x/a) \* b. The difficult part of this graph actually is the rotation of annotation. But because my way is very complicated, I do not think it is worth to be talked. Phoebe gave me another way to calculate the rotation on Wednesday, I tried few times but I always failed with bugs for some reasons, so I still used my old version.

For the last graph, it is almost the combination of week 3 and week 4 graphs, so I don’t want to talk about how I generate the graph, instead the way how I made the graph look better. When the first time I finished the graph, I noticed there are a lot of vertical lines, which made the graph messy. Then I figured out there are 2 situations cause the problem. The first one is when the y value is 0, which means there is no new case confirmed in a day. This is normal, usually happened when a country has controlled the situation. We focus on the relationship of increasing number vs. total number, so I do not think 0 matters in our study, because it varies by countries. Thus, I dropped duplicated total numbers. The graph looks better but vertical lines didn’t disappear. That happens because the increasing number sometimes is negative even it is not expected like that. My solution is to sort the data. Here is my example, Canada. Let us look at the graphs, only the error part got changed. Because we want to know how a country perform in a long time period, to sort the data will only change a single point. So, it won’t affect a lot as the increasing number expected as non-negative. I have put more details of this change in my code as comments. After these 2 changes, no vertical lines exist anymore.

That is all about my presentation, any questions?

Thank you.