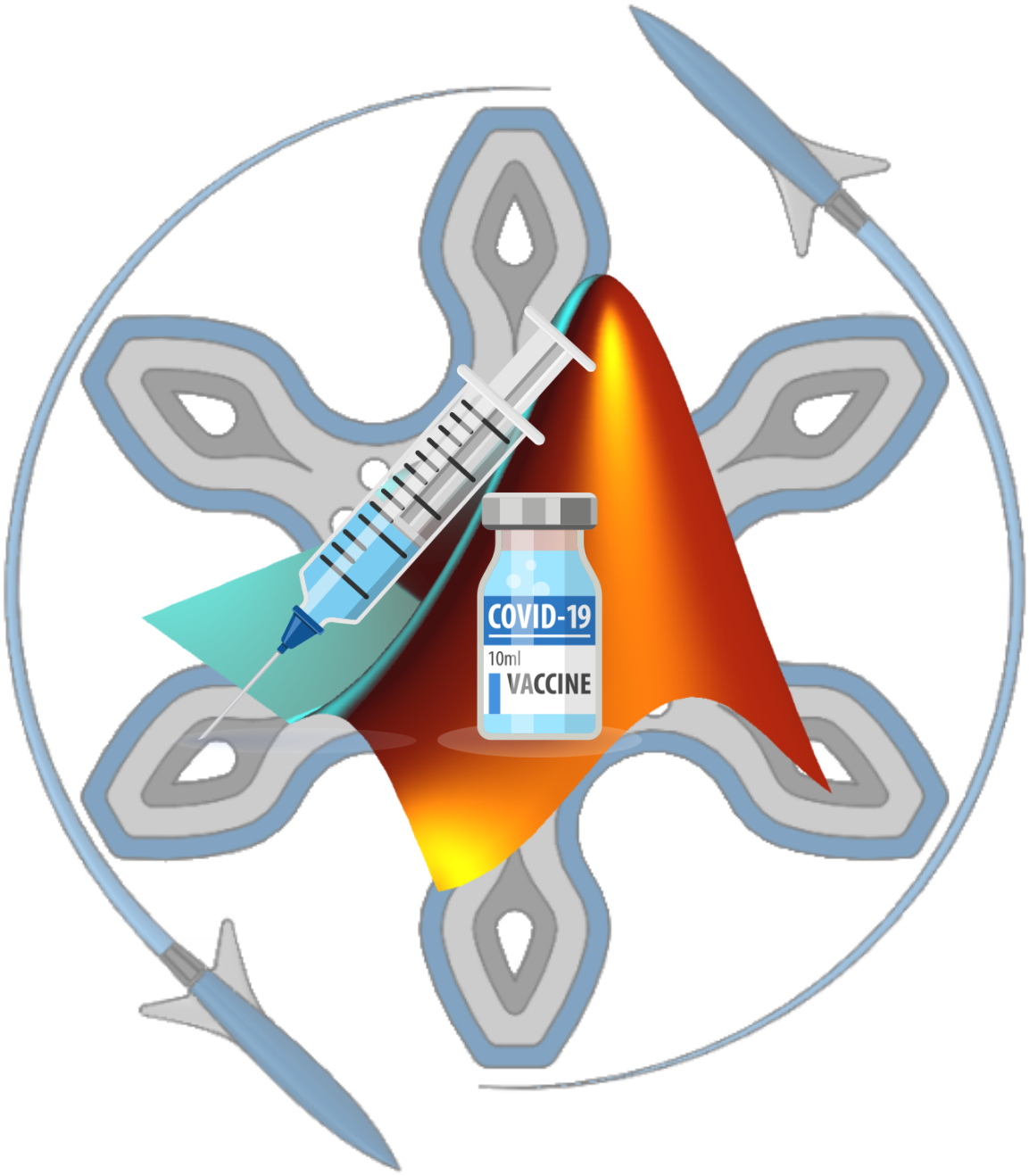


COVID-19 DATA

BY: Nathan Delos Santos 5-23-2021

OVERVIEW

In this project, I will be showing the number of Covid-19 vaccinations within the top five most populous counties in California. I will be showing the total cumulative number of vaccinations, the percentage of population vaccinated, the number of vaccinations by manufacturer, the number of new people with at least one dose, the number of new people fully vaccinated, and the total new vaccinations per day. To do this, I will be using loops, and readtable() to access and work with data given to me on the population and vaccination csv sheets. I will need to log down all the indices for the specific things I am looking for on those sheets, and having the program search for that.

My top level steps were basically calling functions with loops. These functions took in inputs, and using loops, sifted through the tables, and searched for the required indices. With those indices, I looped through the table again, and stored what information I needed from the table through the specified indices. I then plotted this information. The loop then reset all the variables so that it could repeat.

CODE WRITE-UP

```
%%Project 2 Nathan Delos Santos
    population = readtable("cali_county_pop.csv");
    vaccines = readtable("covid19vaccinesbycounty.csv");
    cases = readtable("covid19cases_test");
    vaxStateTotals = readtable("cdc-vaccination-state-totals");
    partsPer = 1000;%Parts per Thousand in this case
    mov = 7;%days
    vaccines.Properties.VariableNames(1) = "County";
    vaccines.Properties.VariableNames(15) = "New_People_With At_Least_One_Dose";
    vaccines.Properties.VariableNames(13) = "New_People_Fully_Vaccinated";
    vaccines.Properties.VariableNames(3) = "Total_Doses";
    vaccines.Properties.VariableNames(4) = "Cumulative_Total_Doses";
    vaccines.Properties.VariableNames(6) = "Cumulative_Pfizer_Doses";
    vaccines.Properties.VariableNames(8) = "Cumulative_Moderna_Doses";
    vaccines.Properties.VariableNames(10) = "Cumulative_J&J_Doses";
    cases.Properties.VariableNames(2) = "County";
    vaxStateTotals.Properties.VariableNames(2) = "County";
    datesList = min(vaccines.administered_date):max(vaccines.administered_date);
    datesList.Format = "MM-dd";
    datesSortedByTime = sort(datesList(end):-1*mov:datesList(1));
    datesSortedByTime = [datesList(1),datesSortedByTime];
```

Defining Variables

I have defined all of my variables here. Changing the values here means that I will not have to go back into my code for every block to change each value. Changing it in an easy to see spot at the top changes it for ALL code. I set the variable “partsPer” to 1000 because I will need to look at the scatter plots per thousand, and I set the variable “mov” to 7, setting the dates to be recorded every 7 days.

Sorting With Functions

```
%%Top Counties
    topNumber = 15;
      %Sets how many counties to look at based on population
    %%Top Vaccinated Percentage Counties
        popPercName = [categorical(population.County)]';
        [vaxPercentageIndex,vaxPercentageTable] = ...
            indexSorting(vaccines,popPercName,vaccines.County,...
            "Cumulative_Total_Doses",height(population),"yes");
        percentages = percentFunc(vaccines,vaxPercentageIndex,...

["cumulative_fully_vaccinated"],population,height(population));
        population.percentages = [percentages]';
        population = sortrows(population);
        vaxPercentageTable = sortrows(vaxPercentageTable);
        vaxPercentageTable.percentages = [percentages]';
        vaxPercentagesNames = ...
            nameSorting(vaxPercentageTable,"percentages",topNumber);
        vaxPercentagesNames = ...

reordercats(vaxPercentagesNames,string(vaxPercentagesNames));
        [vaxPercentageIndexShort,vaxPercentageTableShort] = ...
            indexSorting(vaxPercentageTable,vaxPercentagesNames,...
            vaxPercentageTable.County,"Cumulative_Total_Doses",...
            topNumber,"yes");
        [topVaxPercentageIndex,topVaxPercentageTable] = ...

indexSorting(vaxPercentageTableShort,vaxPercentagesNames,...

vaxPercentageTableShort.County,"Cumulative_Total_Doses",...
            topNumber,"yes");

        topPopPerc = sortrows(population,"percentages","descend");
        topPopPerc(topNumber+1:height(topPopPerc),:) = [];
        topPopPercName = [categorical(topPopPerc.County)]';
        barGraphOutput(topVaxPercentageIndex,vaxPercentagesNames,...
            ["percentages"],topNumber,topVaxPercentageTable,...
            "Total Vaccination Percentage")
```

Here, I called my user-defined functions to sort the counties. I was asked to sort the top 15 counties with the highest percentage of their population vaccinated. For the first function I called, it returned the vaccines table with ONLY those counties, and only the LATEST entry, or latest date for those counties, and the indices in which they appear. I then ran it through the percentages function, which ran through the vaccines table and the population table. It then divided each cumulative vaccinations for each county, divided by its population. It then multiplied it by 100. It added this new variable, or new column to the table. With this, I was able to rank in order, from most percentage vaccinated, to least percentage vaccinated, for the top counties. In the specified order, of most percentage vaccinated to least, for the top counties, I made a bar graph showing each county's vaccination percentage. To get the graphs of the other counties, I created a list of names for ALL counties in alphabetical order

```
%%Doses Bar Charts
    function [barOutput] = ...

barGraphOutput(countiesIndex,countyNames,manufacturers,number,vaxTable,titles)
    %Function that returns a bar graph for a "manufacturer" for each
county. Takes the
    %inputs: index of counties, county names, manufacturers, number of
counties,
    %vaccine table, and a title
        cumulative = [];
        vaxName = [];
        colors = [];
        for i = 1:length(manufacturers)
            vaxName=[vaxName,strrep(manufacturers(i),"_"," ")];
      %Removes the underscores "_" from the manufacturer names for the
legend
            colors=[colors,[rand,rand,rand]'];
      %Randomly generates colors for the many possible manufacturers
        end
        figure;
        for i = 1:number
         %Plots Bars in groups of (how many manufacturers), for each
county
            for ii = 1:length(manufacturers)
                cumulative(ii) =
vaxTable{countiesIndex(i),manufacturers(ii)}; %For this one specific
county, documents the cumulative for each manufacturer
            end
            b=bar(countyNames(i),cumulative);
      %Adds (how many manufacturers) bars at a time
            for c = 1:length(manufacturers)
      %Assigns each bar one of (how many manufacturers) colors.
                b(c).FaceColor = colors(1:3,c);
            end
            legend(vaxName)
      %Adds the legend without the underscores
            hold on
      %Allows for the other counties' bars to be displayed on the same
plot
        end
        title(titles + " By County" + " Up To " +
datestr(max(vaxTable.administered_date))...
            + " (Top" + sprintf(" %.0f ",number)+ "Most Populous)")
        %Adds the title. Lets you know how many counties, and most
recent date.
        ax=gca;
        ax.YGrid = 'on';
        if manufacturers == ["percentages"]
            %ylim([0,100])
            ytickformat('%g\%')
            legend hide
        else
            legend show
            ax.YAxis.Exponent = 0;
            ytickformat("%.0f");
        end
        hold off;
end

%%Sorting
    function [sortedNames] = nameSorting(table,catergory,number)
        top = sortrows(table, catergory, "descend");
        top([number+1:height(top)],:) = [];
        topNames = top.County;
        sortedNames = categorical(topNames);
    end

%%Indexing
    function [sortedIndex,indexOnlyTable] = ...
        indexSorting(table1,table2,counties,catergory,number,mostRecent)
        indecies = [];
        if mostRecent == "yes"
            for i = 1:number
                indecies(i) = max(find(counties == table2(i)));
            end
        end
        newTable = table1;
        if height(table1)>number
            removedIndecies = 1:1:height(table1);
            removedIndecies(indecies) = [];
            newTable(removedIndecies,:) = [];
        end
        newTable = sortrows(newTable, catergory, "descend");
        sortedIndex = indecies;
        indexOnlyTable = newTable;
    end
```

Finding and Sorting Populations

```
%%Cases Per Thousand for Top Vaccinated Percentage Counties
        cases = sortrows(cases,"County");
        allNames = categorical(population.County);
        casesIndecies = [];
        totalCases = [];
        for i = 1:length(allNames)
            casesIndecies = find(cases.County == allNames(i));
            totalCases(i) = sum(cases{casesIndecies(1):...
            casesIndecies(length(casesIndecies)),"cases"},"omitnan");
            casesIndecies = [];
        end
        for i = 1:height(population)
            indecies(i) = max(find(cases.County == allNames(i)));
        end
        removedIndecies = 1:1:height(cases);
        removedIndecies(indecies) = [];
        cases(removedIndecies,:) = [];
        cases.Total_Cases = [totalCases'];
        perCases = [];
        for i = 1:height(cases)
            perCases(i) = ...
            (cases{i,"Total_Cases"}/
population{i,"Population"})*partsPer;
        end
        cases.perCases = [perCases]';
        casesPerentageArray = [];
        for i = 1:height(population)
            index = find(cases.County == allNames(i));
            casesPercentageArray(i) = ...
            vaxPercentageTable{index,"percentages"};
        end
        cases.percentages = [casesPercentageArray]';
```

Here, I find the cases per 1000 population for each county

1. I sort the table alphabetically. This way, all the counties are next to each other, Just so it is in the same alphabetical order as the table with the most recent vaccination idecies.
2. I then loop through the table in alphabetical order, finding each counties' most recent date.
3. I then proceed to delete all other indices that are not the most recent date
4. I then loop through each county, and take their cases, and population. I then multiply it by 1000.
5. I add all of these values, in order of the alphabetical counties and in order of the dates, to a single array.
6. I then turn this array into a column
7. I add a new variable, or column to the the cases table, and add this per 1000 stat
8. Matching with the alphabetical most recent county vaccination table, I take the percentages from the vaccination table, and copy it into an array
9. I turn that array into a column
10. I add that column to the cases table
11. Now, I only need to loop through one table to get my stats for the scatter plot

```
scatterPlotter(cases.percentages,cases.perCases,cases.County,partsPer,...
    [min(cases.date),max(cases.date)],...
    0.5,"COVID-19 Cases vs. Percent Of Population Fully Vaccinated")
    figure;
```

```
%Scatter Plot
function [scatterGraph] = scatterPlotter(x,y,county,parts,dates,offset,partTitle)
    figure;
    scatter(x,y)
    text(x+offset,y+offset,string(county))
    xlabel("Percentage of Population Fully Vaccinated")
    ylabel(sprintf("Number Of Cases Per %.0f of Population",parts))
    title(partTitle + " From "+datestr(dates(1))+" To " + datestr(dates(2)))
    hold off
end
```

12. In the scatter plot, I input the county vaccination percentage on the X axis, and the per 1000 cases on the Y axis

Finding the Percentages Each Day

```
%%Percentage Vaccinated Per Day
        vaccines = dateRemover(vaccines,"administered_date",...
            datesSortedByTime);
        [vaccines,percentagesPerDay] = ...
            percentPerTimeFunc(vaccines,vaccines.County,population,...
            popPercName,"cumulative_fully_vaccinated",topNumber);
        vaccines.percentagesPerDay = [percentagesPerDay];
        [vaccines,percentagesPerDayPfizer] = ...
            percentPerTimeFunc(vaccines,vaccines.County,...
            population,popPercName,"Cumulative_Pfizer_Doses",topNumber);
        vaccines.percentagesPerDayPfizer = [percentagesPerDayPfizer];
        [vaccines,percentagesPerDayModerna] = ...
            percentPerTimeFunc(vaccines,vaccines.County,...
            population,popPercName,"Cumulative_Moderna_Doses",topNumber);
        vaccines.percentagesPerDayModerna = [percentagesPerDayModerna];
        [vaccines,percentagesPerDayJJ] = ...
            percentPerTimeFunc(vaccines,vaccines.County,...
            population,popPercName,"Cumulative_J&J_Doses",topNumber);
        vaccines.percentagesPerDayJJ = [percentagesPerDayJJ];
        vaxStateTotals = sortrows(vaxStateTotals,"County");
        vaxStateTotals.doses_administered_percent = ...
            100*vaxStateTotals.doses_administered_percent;
        vaxStateTotalsNames = categorical(vaxStateTotals.County);
        vaxStateTotalsNames = unique(vaxStateTotalsNames,'stable');
        allMaxVaxTable = vaccines;
        [allMaxVaxIndecies,allMaxVaxTable] = ...
            indexSorting(allMaxVaxTable,allNames,...
            allMaxVaxTable.County,"County",length(allNames),"yes");
        allMaxVaxTable = sortrows(allMaxVaxTable,"County","ascend");
```

```
%%Percentage Vaccinated Per Day Top
    topVax = vaccines;
    indecies = [];
    for i = 1:length(topPopPercName)
        indecies = [indecies,[find(topVax.County == topPopPercName(i))]'];
    end
    removedIndecies = 1:1:height(topVax);
    removedIndecies(indecies) = [];
    topVax(removedIndecies,:) = [];
    topVaxPercentageTable = sortrows(topVaxPercentageTable,"County");
    vaxPercentagesNamesAlpha = categorical(topVaxPercentageTable.County);
    perc = [];
    for i = 1:topNumber
        count = length(find(vaxPercentagesNamesAlpha(i) == topVax.County));
        addPerc = [];
        for ii = 1:count
            addPerc(ii) =  topVaxPercentageTable{i,"percentages"};
        end
        perc = [perc,addPerc];
    end
    topVax.maxPercent = [perc]';
    topVax = sortrows(topVax,"maxPercent","descend");
```

Here, I find how much of the population of each county is fully vaccinated each day.

1. Earlier, I had sorted the days to go by week, meaning that the table should count by every 7 days. I remove every date that is not one of those weeks listed earlier.
2. I pass the tables through the percentPerTime function. The function loops through all the dates, and for each county, divides its population from the population table with the cumulative fully vaccinated from the vaccines table. I repeat this for all manufacturers, and cumulative doses.
3. The only difference for the allMaxVaxTable is that I omit all indices that re not the most recent date

```
%%Percentages Per Day
    function [newTable,newColumn] = ...
        percentPerTimeFunc(vaxTable,countiesIndex,popul,popName,variable,number)
        indecies = [];
        for i = 1:height(popul)
            indecies = [indecies,[find(countiesIndex == popName(i))]'];
        end
        removedIndecies = 1:1:height(vaxTable);
        removedIndecies(indecies) = [];
        length(removedIndecies);
        vaxTable(removedIndecies,:) = [];
        percentageVaxPerTime = [];
        for i = 1:height(popul)
            casesIndecies = find(vaxTable.County == popName(i));
            for ii = casesIndecies
                percentageVaxPerTime = ...
                    [percentageVaxPerTime,[100*vaxTable{ii,variable}/popul{i,"Population"}]'];
            end
            casesIndecies = [];
        end
        newTable = vaxTable;
        newColumn = [percentageVaxPerTime]';
    end
```

```
%%Percentages
    function [percentArray] = percentFunc(vaxTable,countiesIndex,manufacturers,popul,number)
        cumulative = [];
        for i = 1:number
            for ii = 1:length(manufacturers)
                cumulative = [cumulative,100*vaxTable{countiesIndex(i),...
                    manufacturers(ii)}/popul.Population(i)];
                %For this one specific county, documents the cumulative
                %percentages for each manufacturer
            end
        end
        percentArray = cumulative;
    end
```

```
%%Date Remover
    function [newTable] = dateRemover(table,dateVar,dates)
        indecies = [];
        for i = 1:height(table)
            if ismember(table{i,dateVar},dates) == false
                indecies = [indecies,i];
            end
        end
        table(indecies,:) = [];
        newTable = table;
    end
```

Heatmaps

```
%%Calling Heatmap Functions
    figure;
    matrixHeatmapDates(population,datesSortedByTime,vaccines.County,...
        popPercName,vaccines,"administered_date",...
        "percentagesPerDay","Percentage Of Population Fully Vaccinated Per Day")
    figure;
    matrixHeatmapDates(population,datesSortedByTime,vaccines.County,...
        popPercName,vaccines,"administered_date",...
        "percentagesPerDayPfizer","Cumulative Percentage Of Pfizer Doses Given Per Da
    figure;
    matrixHeatmapDates(population,datesSortedByTime,vaccines.County,...
        popPercName,vaccines,"administered_date",...
        "percentagesPerDayModerna","Cumulative Percentage Of Moderna Doses Given Per Day")
    figure;
    matrixHeatmapDates(population,datesSortedByTime,vaccines.County,...
        popPercName,vaccines,"administered_date",...
        "percentagesPerDayJJ","Cumulative Percentage Of J % J Doses Given Per Day")
    figure;
    matrixHeatmapDates(vaxPercentagesNames,datesSortedByTime,topVax.County,...
        vaxPercentagesNames,topVax,"administered_date",...
        "percentagesPerDay",...
        "Percentage Of Population Fully Vaccinated Per Day Sorted By Top "...
        + topNumber + " Vaccinated Counties")
    figure;
```

```
matrixHeatmapTotalManufacturer(["percentagesPerDayPfizer",...
    "percentagesPerDayModerna","percentagesPerDayJJ",...
    "percentagesPerDay"],allMaxVaxTable.County,...
    popPercName,allMaxVaxTable,...
    ["Pfizer","Moderna","J & J","Cumulative"],...
    "Percentage Of Cumulative Doses Given By Manufacturer")
```

```
datesList = min(vaxStateTotals.date):max(vaxStateTotals.date);
datesList.Format = "MM-dd";
datesSortedByTime = sort(datesList(end):-1*mov:datesList(1));
datesSortedByTime = [datesList(1),datesSortedByTime];
vaxStateTotals = dateRemover(vaxStateTotals,"date",datesSortedByTime);
figure;
matrixHeatmapDates(vaxStateTotalsNames,datesSortedByTime,...
    vaxStateTotals.County,vaxStateTotalsNames,vaxStateTotals,"date",...
    "doses_administered_percent","Percentage Of Doses Administered For Doses Distributed"
```

Here, I plot heatmaps for vaccinations.
Per Day (County and State):

1. I preallocate a blank matrix full of NaN
2. I input a county, and find which dates it has data available for. I also log the percentage of cumulative fully vaccinated population
3. Knowing which dates the county has available, I log the indices for the row of the heatmap. This means that for example, if a county is missing the 2nd and 4th dates, it will update every other tile, but leave the ones on the 2nd and 4th dates blank.
4. The matrix is now populated with the correct percentages.
5. I input it into the heatmap
6. Note, for the state vaccinations, I had to rest the dates array because I would end up with a large swath of NaN on my heatmap before Feb 1

Per manufacturer:

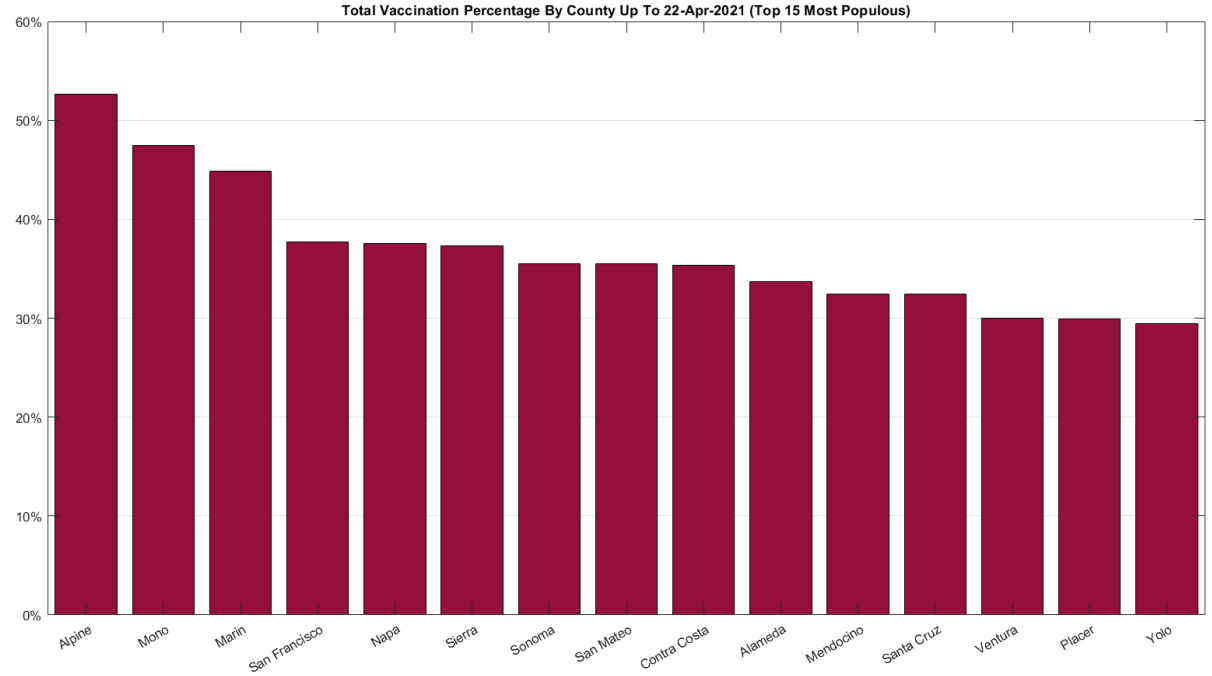
1. I preallocate a blank matrix full of NaN
2. I input a county, and find which manufacturers it has data available for. I also log the percentage of cumulative fully vaccinated population
3. Knowing which manufacturers the county has available, I log the indices for the row of the heatmap. This means that for example, if a county is missing the 1st and 3rd manufacturers, it will update every other tile, but leave the ones on the 1st and 3rd manufacturers blank.
4. The matrix is now populated with the correct percentages.
5. I input it into the heatmap

```
%%Heatmap Along Dates
    function [matrixHeat] =...
        matrixHeatmapDates(popTable,dates,counties,names,vaxTable,...
        tableDates,variable,partTitle)
        fully_vac_day = NaN(height(popTable),length(dates));
        for i = 1:height(popTable)
            countyIndecies = find(counties == names(i));
            factorIndecies = [];
            for ii = 1:length(countyIndecies)
                for iii = 1:length(dates)
                    if vaxTable{countyIndecies(ii),tableDates} == dates(iii)
                        factorIndecies(iii) = iii;
                    end
                end
            end
            factorIndecies = factorIndecies(factorIndecies~=0);
            for ii = 1:length(factorIndecies)
                fully_vac_day(i,factorIndecies(ii)) = vaxTable{countyIndecies(ii),variable};
            end
        end
        heatmap(dates,names,fully_vac_day,"Colormap",turbo);
        caxis([0 100]);
        title(partTitle + " Up To " + datestr(max(dates)))
    end
```

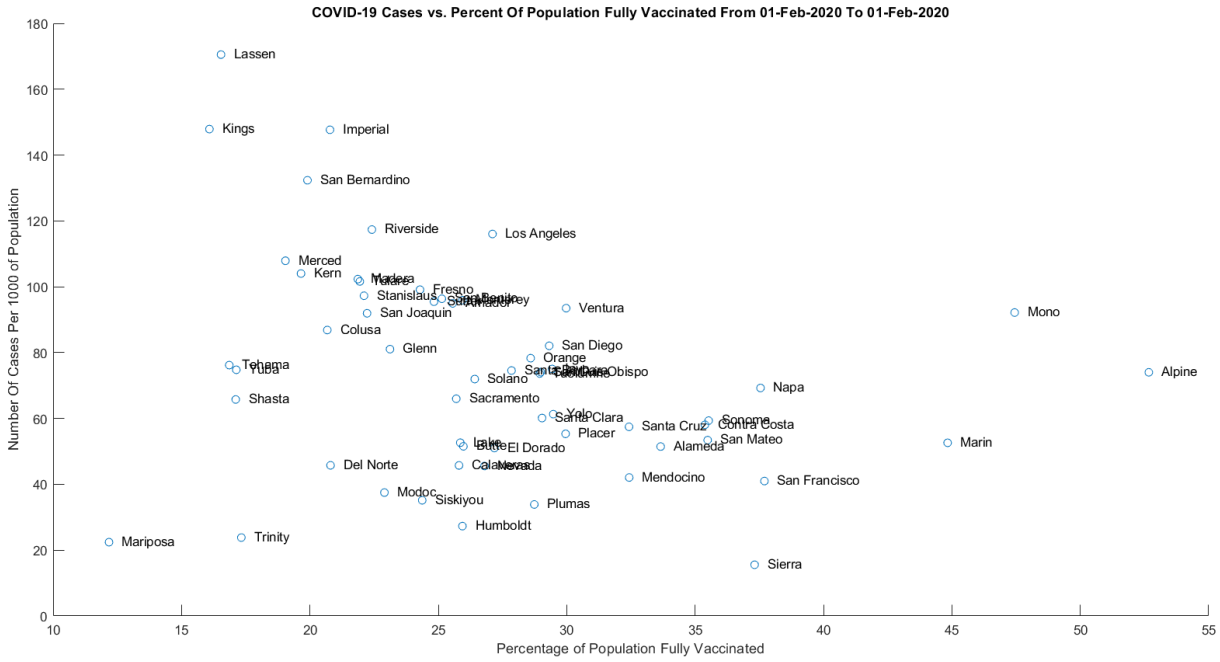
```
%%Heatmap Along Manufacturer
    function [matrixHeat] = ...
        matrixHeatmapTotalManufacturer(manufacturers,counties,names,...
        vaxTable,manufacturersList,titlePart)
        fully_vac_manuf = NaN(length(names),length(manufacturers));
        for i = 1:length(names)
            countyIndex = find(counties == names(i));
            factorIndecies = [];
            for ii = 1:length(countyIndex)
                for iii = 1:length(manufacturers)
                    if isnan(vaxTable{countyIndex(ii),manufacturers(iii)}) == false
                        factorIndecies(iii) = iii;
                    end
                end
            end
            for ii = 1:length(factorIndecies)
                if factorIndecies(ii) ~= 0
                    fully_vac_manuf(i,factorIndecies(ii)) = ...
                        vaxTable{countyIndex,manufacturers(ii)};
                end
            end
        end
        heatmap(manufacturersList,names,fully_vac_manuf,"Colormap",turbo);
        caxis([0 100]);
        title(titlePart + " Up To " + datestr(max(vaxTable.administered_date)))
    end
```

BAR GRAPHS, SCATTER PLOTS & HEATMAPS

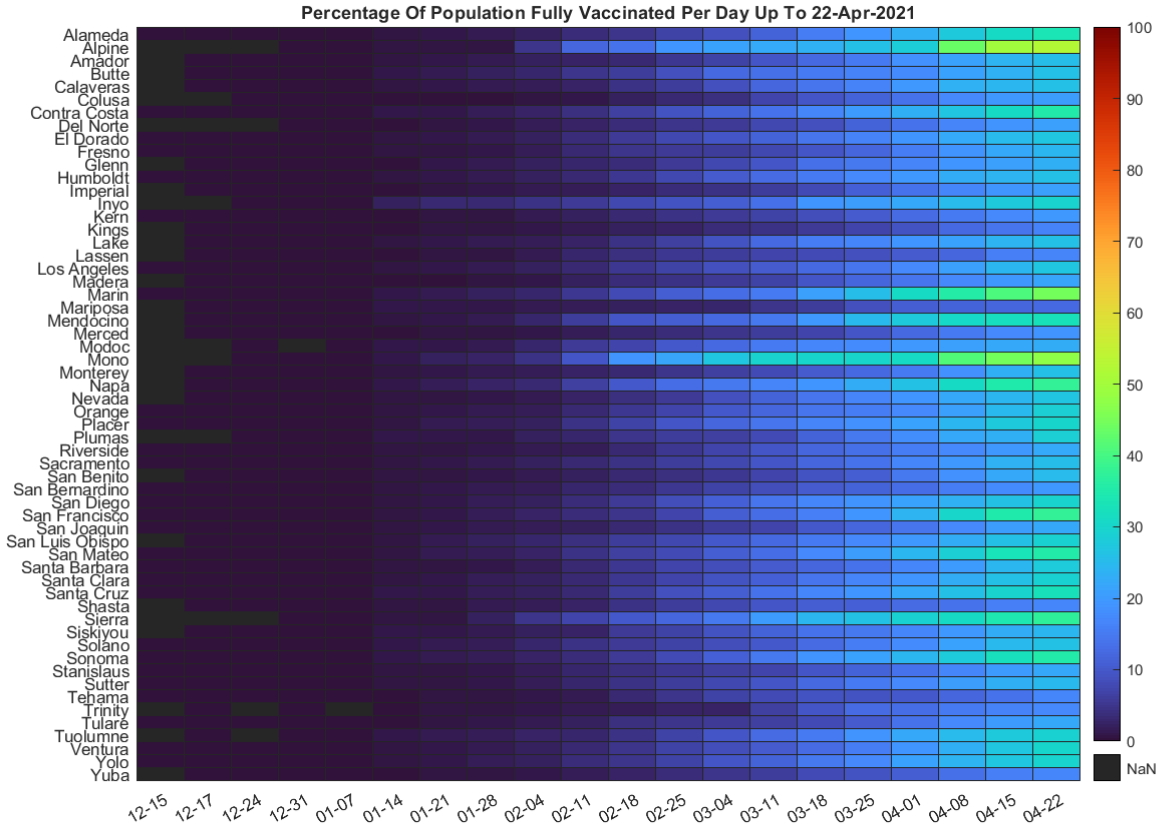
Percentage Of Population Fully Vaccinated For Top 15 Most Vaccinated Counties

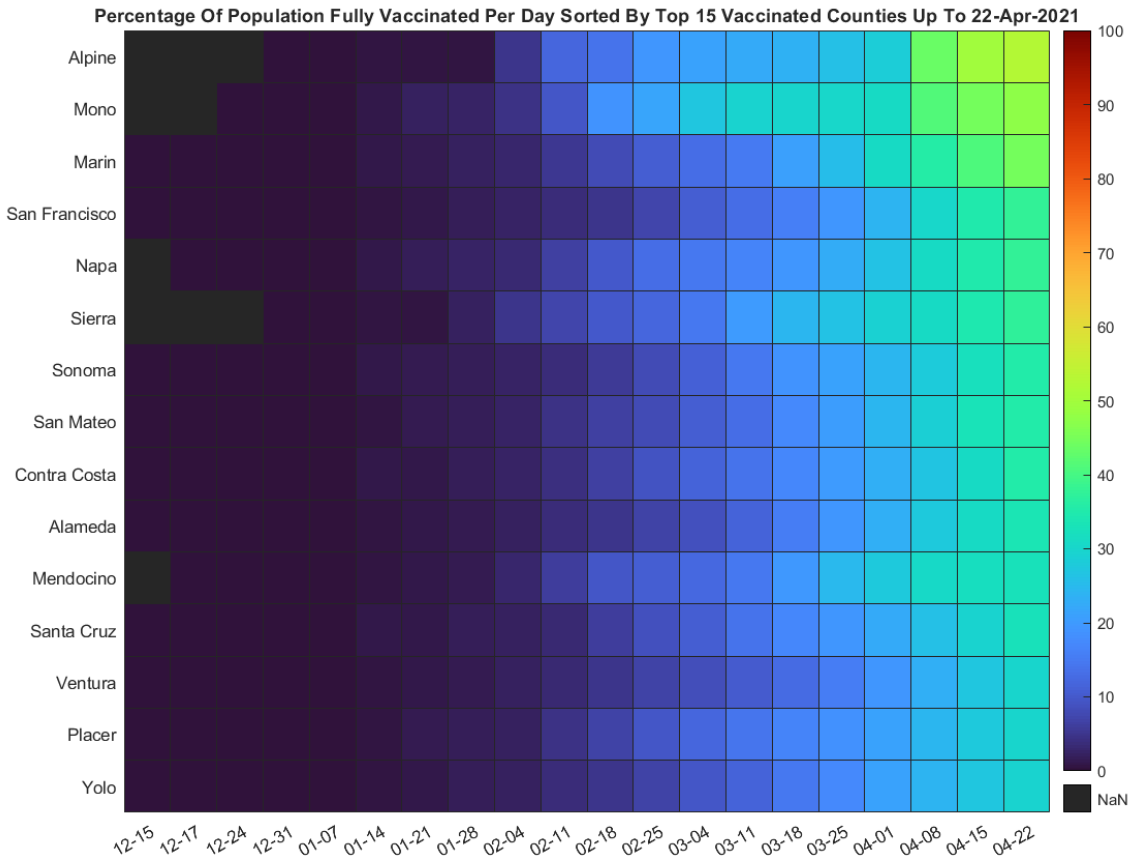


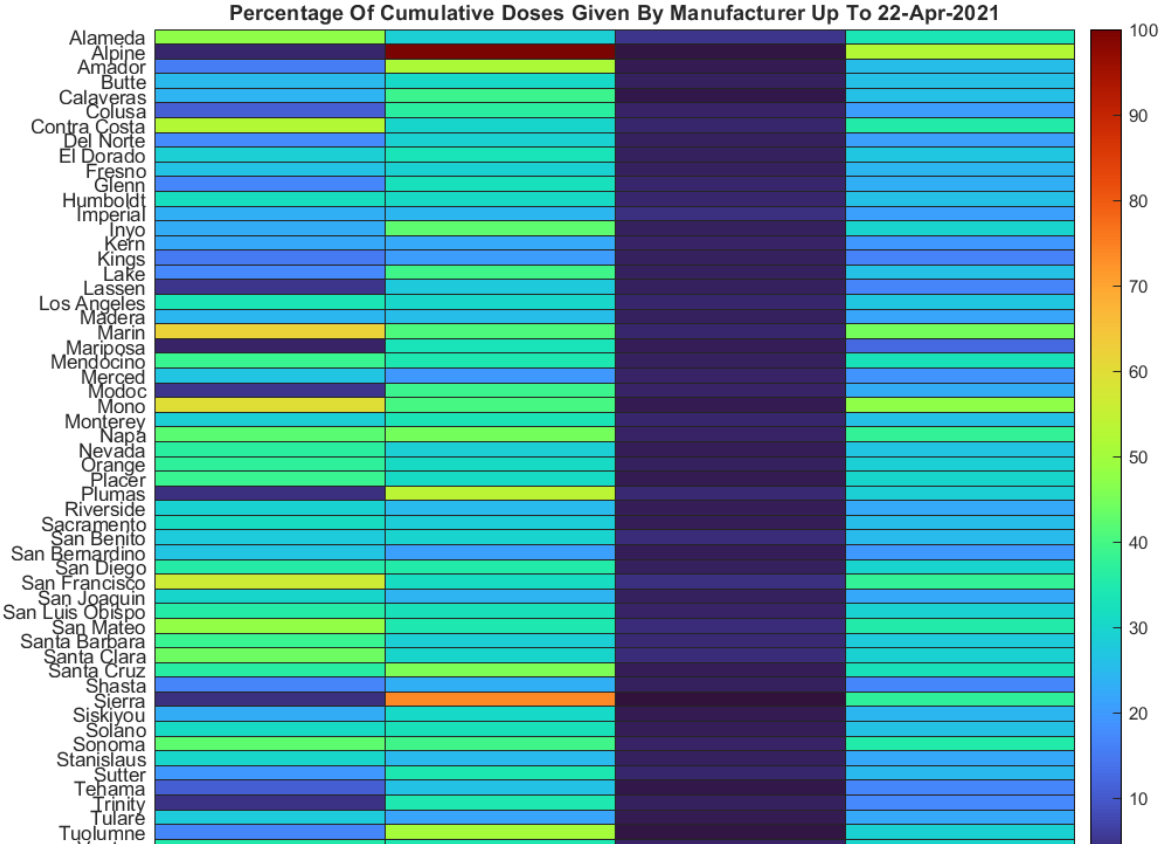
COVID-19 Cases Per 1000 Population Vs. Percent Of Population Fully Vaccinated

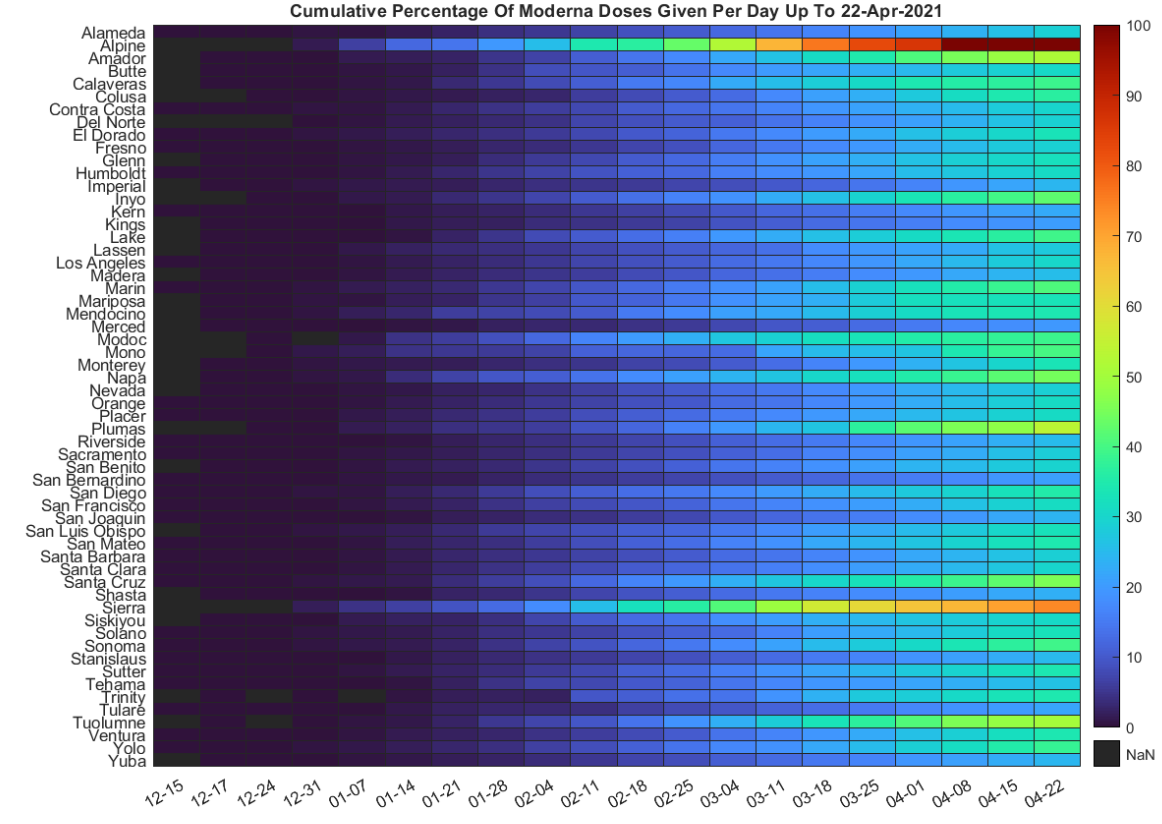


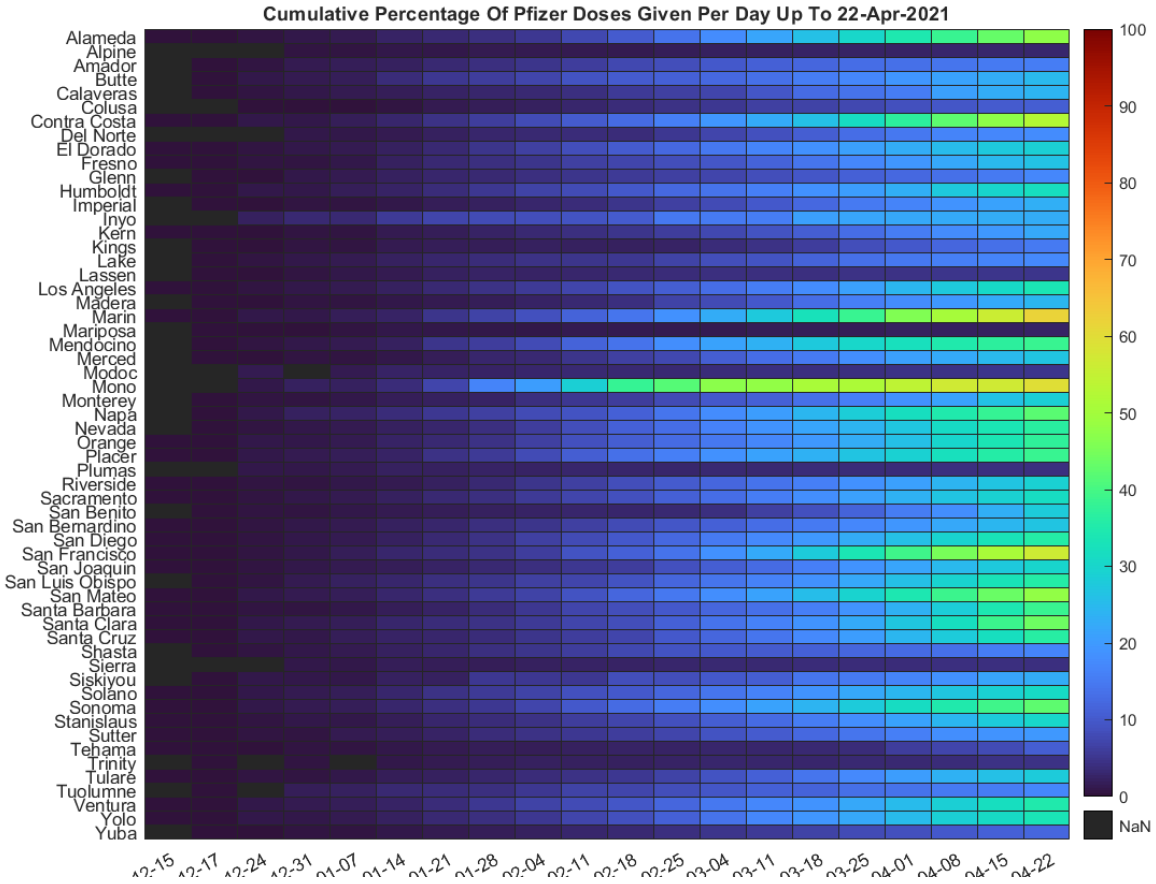
Heatmaps of Vaccinations

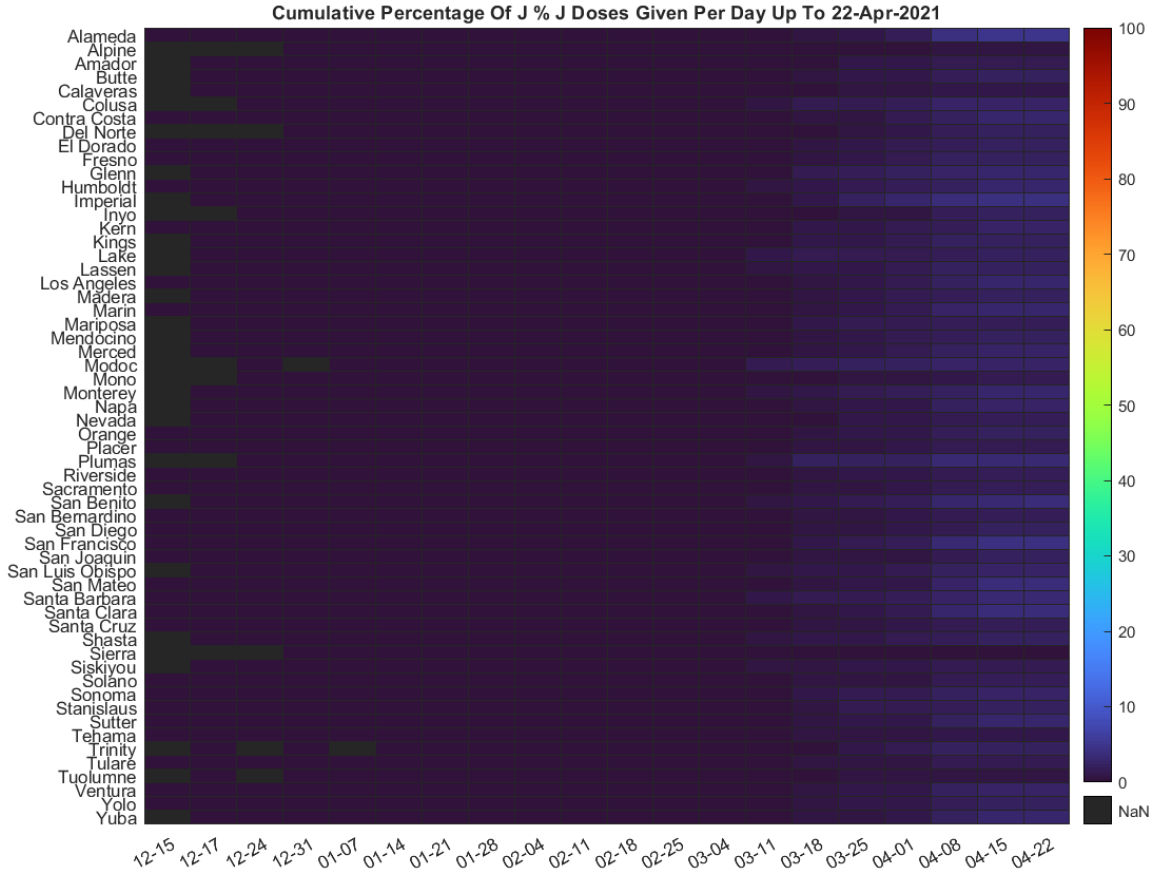


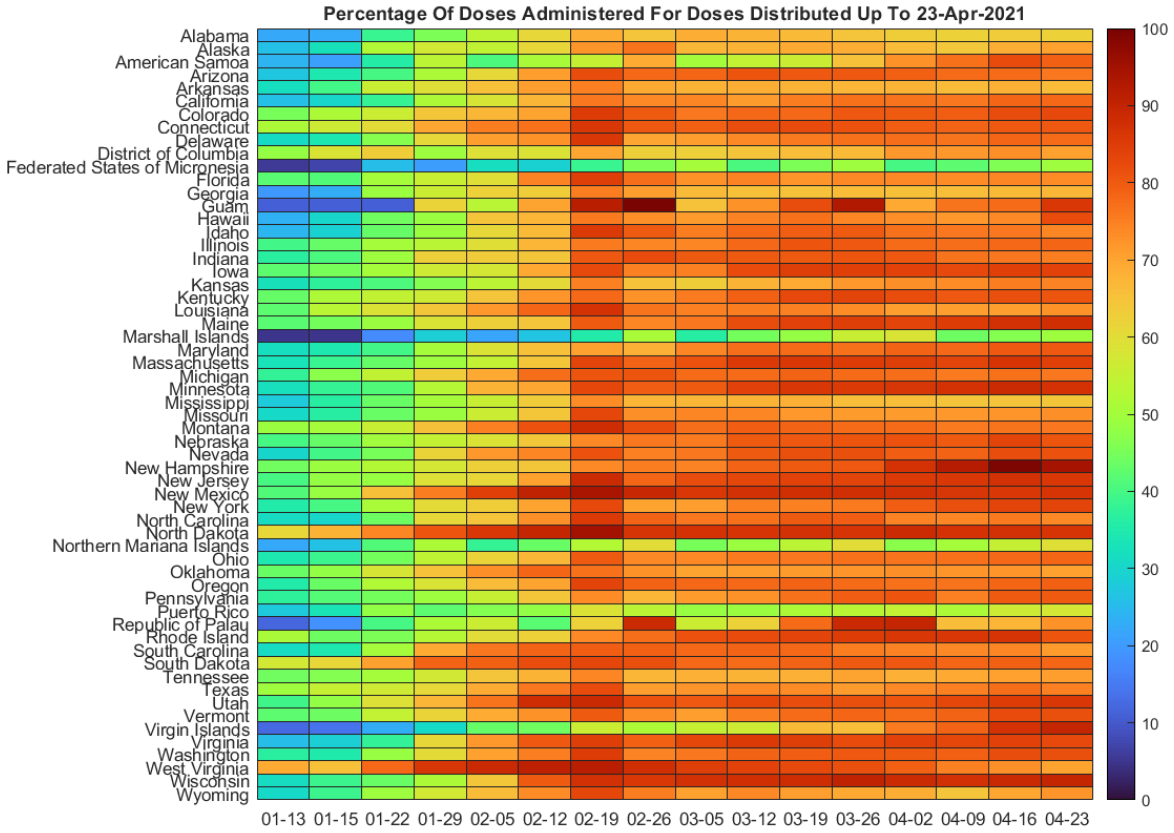



Percentage Of Cumulative Doses Given By Manufacturer Up To 22-Apr-2021
Alameda
Alpine
Amador
Butte
Calaveras
Colusa
Contra Costa
Del Norte
El Dorado
Fresno
Glenn
Humboldt
Imperial
Inyo
Kern
Kings
Lake
Lassen
Los Angeles
Madera
Marin
Mariposa
Mendocino
Merced
Modoc
Mono
Monterey
Napa
Nevada
Orange
Placer
Plumas
Riverside
Sacramento
San Benito
San Bernardino
San Diego
San Francisco
San Joaquin
San Luis Obispo
San Mateo
Santa Barbara
Santa Clara
Santa Cruz
Shasta
Sierra
Siskiyou
Solano
Sonoma
Stanislaus
Sutter
Tehama
Trinity
Tulare
Tuolumne
100
90
80
70
60
50
40
30
20
10


Cumulative Percentage Of Moderna Doses Given Per Day Up To 22-Apr-2021
Alameda
Alpine
Amador
Butte
Calaveras
Colusa
Contra Costa
Del Norte
El Dorado
Fresno
Glenn
Humboldt
Imperial
Inyo
Kern
Kings
Lake
Lassen
Los Angeles
Madera
Marin
Mariposa
Mendocino
Merced
Modoc
Mono
Monterey
Napa
Nevada
Orange
Placer
Plumas
Riverside
Sacramento
San Benito
San Bernardino
San Diego
San Francisco
San Joaquin
San Luis Obispo
San Mateo
Santa Barbara
Santa Clara
Santa Cruz
Shasta
Sierra
Siskiyou
Solano
Sonoma
Stanislaus
Sutter
Tehama
Trinity
Tulare
Tuolumne
Ventura
Yolo
Yuba
12-15
12-17
12-24
12-31
01-07
01-14
01-21
01-28
02-04
02-11
02-18
02-25
03-04
03-11
03-18
03-25
04-01
04-08
04-15
04-22
100
90
80
70
60
50
40
30
20
10
0
NaN


Cumulative Percentage Of Pfizer Doses Given Per Day Up To 22-Apr-2021
Alameda
Alpine
Amador
Butte
Calaveras
Colusa
Contra Costa
Del Norte
El Dorado
Fresno
Glenn
Humboldt
Imperial
Inyo
Kern
Kings
Lake
Lassen
Los Angeles
Madera
Marin
Mariposa
Mendocino
Merced
Modoc
Mono
Monterey
Napa
Nevada
Orange
Placer
Plumas
Riverside
Sacramento
San Benito
San Bernardino
San Diego
San Francisco
San Joaquin
San Luis Obispo
San Mateo
Santa Barbara
Santa Clara
Santa Cruz
Shasta
Sierra
Siskiyou
Solano
Sonoma
Stanislaus
Sutter
Tehama
Trinity
Tulare
Tuolumne
Ventura
Yolo
Yuba
100
90
80
70
60
50
40
30
20
10
0
NaN
12-15
12-17
12-24
12-31
01-07
01-14
01-21
01-28
02-04
02-11
02-18
02-25
03-04
03-11
03-18
03-25
04-01
04-08
04-15
04-22


Cumulative Percentage Of J % J Doses Given Per Day Up To 22-Apr-2021
Alameda
Alpine
Amador
Butte
Calaveras
Colusa
Contra Costa
Del Norte
El Dorado
Fresno
Glenn
Humboldt
Imperial
Inyo
Kern
Kings
Lake
Lassen
Los Angeles
Madera
Marin
Mariposa
Mendocino
Merced
Modoc
Mono
Monterey
Napa
Nevada
Orange
Placer
Plumas
Riverside
Sacramento
San Benito
San Bernardino
San Diego
San Francisco
San Joaquin
San Luis Obispo
San Mateo
Santa Barbara
Santa Clara
Santa Cruz
Shasta
Sierra
Siskiyou
Solano
Sonoma
Stanislaus
Sutter
Tehama
Trinity
Tulare
Tuolumne
Ventura
Yolo
Yuba
100
90
80
70
60
50
40
30
20
10
0
NaN
12-15
12-17
12-24
12-31
01-07
01-14
01-21
01-28
02-04
02-11
02-18
02-25
03-04
03-11
03-18
03-25
04-01
04-08
04-15
04-22


Percentage Of Doses Administered For Doses Distributed Up To 23-Apr-2021
Alabama
Alaska
American Samoa
Arizona
Arkansas
California
Colorado
Connecticut
Delaware
District of Columbia
Federated States of Micronesia
Florida
Georgia
Guam
Hawaii
Idaho
Illinois
Indiana
Iowa
Kansas
Kentucky
Louisiana
Maine
Marshall Islands
Maryland
Massachusetts
Michigan
Minnesota
Mississippi
Missouri
Montana
Nebraska
Nevada
New Hampshire
New Jersey
New Mexico
New York
North Carolina
North Dakota
Northern Mariana Islands
Ohio
Oklahoma
Oregon
Pennsylvania
Puerto Rico
Republic of Palau
Rhode Island
South Carolina
South Dakota
Tennessee
Texas
Utah
Vermont
Virgin Islands
Virginia
Washington
West Virginia
Wisconsin
Wyoming
01-13
01-15
01-22
01-29
02-05
02-12
02-19
02-26
03-05
03-12
03-19
03-26
04-02
04-09
04-16
04-23
100
90
80
70
60
50
40
30
20
10
0

MATLAB CODE

```
clc;
clear;

%%Project 2 Nathan Delos Santos
   population = readtable("cali_county_pop.csv");
   vaccines = readtable("covid19vaccinesbycounty.csv");
   cases = readtable("covid19cases_test");
   vaxStateTotals = readtable("cdc-vaccination-state-totals");
   partsPer = 1000;%Parts per Thousand in this case
   mov = 7;%days
   vaccines.Properties.VariableNames(1) = "County";
   vaccines.Properties.VariableNames(15) = "New_People_With At_Least_One_Dose";
   vaccines.Properties.VariableNames(13) = "New_People_Fully_Vaccinated";
   vaccines.Properties.VariableNames(3) = "Total_Doses";
   vaccines.Properties.VariableNames(4) = "Cumulative_Total_Doses";
   vaccines.Properties.VariableNames(6) = "Cumulative_Pfizer_Doses";
   vaccines.Properties.VariableNames(8) = "Cumulative_Moderna_Doses";
   vaccines.Properties.VariableNames(10) = "Cumulative_J&J_Doses";
   cases.Properties.VariableNames(2) = "County";
   vaxStateTotals.Properties.VariableNames(2) = "County";
   datesList = min(vaccines.administered_date):max(vaccines.administered_date);
   datesList.Format = "MM-dd";
   datesSortedByTime = sort(datesList(end):-1*mov:datesList(1));
   datesSortedByTime = [datesList(1),datesSortedByTime];

%%Top Counties
   topNumber = 15;                                    %Sets how many counties to look at
based on population
   %%Top Vaccinated Percentage Counties
      popPercName = [categorical(population.County)]';
      [vaxPercentageIndex,vaxPercentageTable] =
indexSorting(vaccines,popPercName,vaccines.County,"Cumulative_Total_Doses",height(popula
tion),"yes");
      percentages =
percentFunc(vaccines,vaxPercentageIndex,["cumulative_fully_vaccinated"],population,height(po
pulation));
      population.percentages = [percentages]';
      population = sortrows(population);
      vaxPercentageTable = sortrows(vaxPercentageTable);
      vaxPercentageTable.percentages = [percentages]';
      vaxPercentagesNames = nameSorting(vaxPercentageTable,"percentages",topNumber);
```

```
    vaxPercentagesNames =
reordercats(vaxPercentagesNames,string(vaxPercentagesNames));
    [vaxPercentageIndexShort,vaxPercentageTableShort] =
indexSorting(vaxPercentageTable,vaxPercentagesNames,vaxPercentageTable.County,"Cumula
tive_Total_Doses",topNumber,"yes");
    [topVaxPercentageIndex,topVaxPercentageTable] =
indexSorting(vaxPercentageTableShort,vaxPercentagesNames,vaxPercentageTableShort.Coun
ty,"Cumulative_Total_Doses",topNumber,"yes");

    topPopPerc = sortrows(population,"percentages","descend");
    topPopPerc(topNumber+1:height(topPopPerc),:) = [];
    topPopPercName = [categorical(topPopPerc.County)]';

barGraphOutput(topVaxPercentageIndex,vaxPercentagesNames,["percentages"],topNumber,to
pVaxPercentageTable,"Total Vaccination Percentage")
  %%Cases Per Thousand for Top Vaccinated Percentage Counties
    cases = sortrows(cases,"County");
    allNames = categorical(population.County);
    casesIndecies = [];
    totalCases = [];
    for i = 1:length(allNames)
        casesIndecies = find(cases.County == allNames(i));
        totalCases(i) =
sum(cases{casesIndecies(1):casesIndecies(length(casesIndecies)),"cases"},"omitnan");
        casesIndecies = [];
    end
    for i = 1:height(population)
        indecies(i) = max(find(cases.County == allNames(i)));
    end
    removedIndecies = 1:1:height(cases);
    removedIndecies(indecies) = [];
    cases(removedIndecies,:) = [];
    cases.Total_Cases = [totalCases'];
    perCases = [];
    for i = 1:height(cases)
        perCases(i) = (cases{i,"Total_Cases"}/population{i,"Population"})*partsPer;
    end
    cases.perCases = [perCases]';
    casesPerentageArray = [];
    for i = 1:height(population)
        index = find(cases.County == allNames(i));
        casesPercentageArray(i) = vaxPercentageTable{index,"percentages"};
    end
    cases.percentages = [casesPercentageArray]';
```

```
    %%Percentage Vaccinated Per Day
        vaccines = dateRemover(vaccines,"administered_date",datesSortedByTime);
        [vaccines,percentagesPerDay] =
percentPerTimeFunc(vaccines,vaccines.County,population,popPercName,"cumulative_fully_vac
cinated",topNumber);
        vaccines.percentagesPerDay = [percentagesPerDay];
        [vaccines,percentagesPerDayPfizer] =
percentPerTimeFunc(vaccines,vaccines.County,population,popPercName,"Cumulative_Pfizer_
Doses",topNumber);
        vaccines.percentagesPerDayPfizer = [percentagesPerDayPfizer];
        [vaccines,percentagesPerDayModerna] =
percentPerTimeFunc(vaccines,vaccines.County,population,popPercName,"Cumulative_Modern
a_Doses",topNumber);
        vaccines.percentagesPerDayModerna = [percentagesPerDayModerna];
        [vaccines,percentagesPerDayJJ] =
percentPerTimeFunc(vaccines,vaccines.County,population,popPercName,"Cumulative_J&J_Do
ses",topNumber);
        vaccines.percentagesPerDayJJ = [percentagesPerDayJJ];
        vaxStateTotals = sortrows(vaxStateTotals,"County");
        vaxStateTotals.doses_administered_percent =
100*vaxStateTotals.doses_administered_percent;
        vaxStateTotalsNames = categorical(vaxStateTotals.County);
        vaxStateTotalsNames = unique(vaxStateTotalsNames,'stable');
        allMaxVaxTable = vaccines;
        [allMaxVaxIndecies,allMaxVaxTable] =
indexSorting(allMaxVaxTable,allNames,allMaxVaxTable.County,"County",length(allNames),"yes"
);
        allMaxVaxTable = sortrows(allMaxVaxTable,"County","ascend");

        %%Percentage Vaccinated Per Day Top
            topVax = vaccines;
            indecies = [];
            for i = 1:length(topPopPercName)
                indecies = [indecies,[find(topVax.County == topPopPercName(i))]'];
            end
            removedIndecies = 1:1:height(topVax);
            removedIndecies(indecies) = [];
            topVax(removedIndecies,:) = [];
            topVaxPercentageTable = sortrows(topVaxPercentageTable,"County");
            vaxPercentagesNamesAlpha = categorical(topVaxPercentageTable.County);
            perc = [];
            for i = 1:topNumber
                count = length(find(vaxPercentagesNamesAlpha(i) == topVax.County));
                addPerc = [];
```

```
            for ii = 1:count
                addPerc(ii) =  topVaxPercentageTable{i,"percentages"};
            end
            perc = [perc,addPerc];
        end
        topVax.maxPercent = [perc]';
        topVax = sortrows(topVax,"maxPercent","descend");

%%Calling Heatmap Functions

scatterPlotter(cases.percentages,cases.perCases,cases.County,partsPer,[min(cases.date),max(
cases.date)],0.5,"COVID-19 Cases vs. Percent Of Population Fully Vaccinated")
   figure;

matrixHeatmapDates(population,datesSortedByTime,vaccines.County,popPercName,vaccines,"
administered_date","percentagesPerDay","Percentage Of Population Fully Vaccinated Per
Day")
   figure;

matrixHeatmapDates(population,datesSortedByTime,vaccines.County,popPercName,vaccines,"
administered_date","percentagesPerDayPfizer","Cumulative Percentage Of Pfizer Doses Given
Per Day")
   figure;

matrixHeatmapDates(population,datesSortedByTime,vaccines.County,popPercName,vaccines,"
administered_date","percentagesPerDayModerna","Cumulative Percentage Of Moderna Doses
Given Per Day")
   figure;

matrixHeatmapDates(population,datesSortedByTime,vaccines.County,popPercName,vaccines,"
administered_date","percentagesPerDayJJ","Cumulative Percentage Of J % J Doses Given Per
Day")
   figure;

matrixHeatmapDates(vaxPercentagesNames,datesSortedByTime,topVax.County,vaxPercentag
esNames,topVax,"administered_date","percentagesPerDay","Percentage Of Population Fully
Vaccinated Per Day Sorted By Top " + topNumber + " Vaccinated Counties")
   figure;

matrixHeatmapTotalManufacturer(["percentagesPerDayPfizer","percentagesPerDayModerna","p
ercentagesPerDayJJ","percentagesPerDay"],allMaxVaxTable.County,popPercName,allMaxVaxT
able,["Pfizer","Moderna","J & J","Cumulative"],"Percentage Of Cumulative Doses Given By
Manufacturer")
```

```
    datesList = min(vaxStateTotals.date):max(vaxStateTotals.date);
    datesList.Format = "MM-dd";
    datesSortedByTime = sort(datesList(end):-1*mov:datesList(1));
    datesSortedByTime = [datesList(1),datesSortedByTime];
    vaxStateTotals = dateRemover(vaxStateTotals,"date",datesSortedByTime);
    figure;

matrixHeatmapDates(vaxStateTotalsNames,datesSortedByTime,vaxStateTotals.County,vaxStat
eTotalsNames,vaxStateTotals,"date","doses_administered_percent","Percentage Of Doses
Administered For Doses Distributed")



%%Sorting
    function [sortedNames] = nameSorting(table,catergory,number)
        top = sortrows(table, catergory, "descend");                    %Puts the most [insert variable]
counties first
        top([number+1:height(top)],:) = [];                             %Everything after the top few
counties is erased
        topNames = top.County;                                          %Gathers a list of names of those
counties. The previous line gathered ALL information about the county.
        sortedNames = categorical(topNames);                            %makes the array usable
elsewhere
    end

%%Indexing
    function [sortedIndex,indexOnlyTable] =
indexSorting(table1,table2,counties,catergory,number,mostRecent)
        indecies = [];                                                  %Creating an array of the indexes of the
counties
        if mostRecent == "yes"
            for i = 1:number
                indecies(i) = max(find(counties == table2(i)));         %Finds the most recent index
of the county in the vaccine table
            end
        end
        newTable = table1;
        if height(table1)>number
            removedIndecies = 1:1:height(table1);
            removedIndecies(indecies) = [];
            newTable(removedIndecies,:) = [];
        end
        newTable = sortrows(newTable, catergory, "descend");
        sortedIndex = indecies;
```

```
    indexOnlyTable = newTable;
  end

%%Doses Bar Charts
  function [barOutput] =
barGraphOutput(countiesIndex,countyNames,manufacturers,number,vaxTable,titles)
  %Function that returns a bar graph for a "manufacturer" for each county. Takes the inputs:
index of counties, county names, manufacturers, number of counties, vaccine table, and a title
    cumulative = [];
    vaxName = [];
    colors = [];
    for i = 1:length(manufacturers)
      vaxName=[vaxName,strrep(manufacturers(i),"_"," ")];          %Removes the
underscores "_" from the manufacturer names for the legend
      colors=[colors,[rand,rand,rand]'];                    %Randomly generates colors for the
many possible manufacturers
    end
    figure;
    for i = 1:number                                    %Plots Bars in groups of (how many
manufacturers), for each county
      for ii = 1:length(manufacturers)
        cumulative(ii) = vaxTable{countiesIndex(i),manufacturers(ii)}; %For this one specific
county, documents the cumulative for each manufacturer
      end
      b=bar(countyNames(i),cumulative);                   %Adds (how many
manufacturers) bars at a time
      for c = 1:length(manufacturers)                     %Assigns each bar one of (how
many manufacturers) colors.
        b(c).FaceColor = colors(1:3,c);
      end
      legend(vaxName)                                   %Adds the legend without the
underscores
      hold on                                           %Allows for the other counties' bars to be
displayed on the same plot
    end
    title(titles + " By County" + " Up To " + datestr(max(vaxTable.administered_date))+ " (Top" +
sprintf(" %.0f ",number)+ "Most Populous)") %Adds the title. Lets you know how many counties,
and most recent date.
    ax=gca;
    ax.YGrid = 'on';
    if manufacturers == ["percentages"]
      %ylim([0,100])
      ytickformat('%g\%')
      legend hide
```

```
        else
            legend show
            ax.YAxis.Exponent = 0;
            ytickformat("%.0f");
        end
        hold off;
    end                                                 %Repeats for all counties
%%Percentages
    function [percentArray] = percentFunc(vaxTable,countiesIndex,manufacturers,popul,number)
        cumulative = [];
        for i = 1:number                                %Plots Bars in groups of (how many
manufacturers), for each county
            for ii = 1:length(manufacturers)
                cumulative =
[cumulative,100*vaxTable{countiesIndex(i),manufacturers(ii)}/popul.Population(i)]; %For this one
specific county, documents the cumulative percentages for each manufacturer
            end
        end
        percentArray = cumulative;
    end
%%Percentages Per Day
    function [newTable,newColumn] =
percentPerTimeFunc(vaxTable,countiesIndex,popul,popName,variable,number)
        indecies = [];
        for i = 1:height(popul)
            indecies = [indecies,[find(countiesIndex == popName(i))]'];
        end
        removedIndecies = 1:1:height(vaxTable);
        removedIndecies(indecies) = [];
        length(removedIndecies);
        vaxTable(removedIndecies,:) = [];
        percentageVaxPerTime = [];
        for i = 1:height(popul)
            casesIndecies = find(vaxTable.County == popName(i));
            for ii = casesIndecies
                percentageVaxPerTime =
[percentageVaxPerTime,[100*vaxTable{ii,variable}/popul{i,"Population"}]'];
            end
            casesIndecies = [];
        end
        newTable = vaxTable;
        newColumn = [percentageVaxPerTime]';
    end
```

```
%%Rolling Average Line Charts
   function [roll] =
rollingAvgChart(factors,vaxTable,countyNames,countyPop,number,parts,moving)
   %Function that returns (how many factors) line plots, displaying the factors on a 7 day
moving average for each county
      for i = 1:factors.length                                %Loops through one factor at a time
         vaxNumbers = [];                                     %Resets the factor numbers after each
factor
         for ii = 1:number
            vaxNumbersIndex = [];                             %Resets the array of indecies after
each county
            vaxNumbersIndex = find(vaxTable.county == countyNames(ii)); %Finds all of the
indecies for that county for that specific factor. Each index is a day.
            temp = [];                                        %Resets the temporary variable for each
county
            for iii = 1:length(vaxNumbersIndex)               %Loops through each day for that
county for that factor
               temp(iii) =  [vaxTable{vaxNumbersIndex(iii),factors(i)}]'; %Adds the factor number to
the temporary array AS A COLUMN. This way, it is easier  for humans to see the divisions for
each county.
               temp(iii) = parts * temp(iii)/countyPop{ii,"Population"}; %Divides that newly added
factor by the population, and multiplies it by the "Parts Per" variable
            end
            vaxNumbers(:,ii) = temp;                          %Adds the column of factor numbers
         end
         strFactor = strrep(factors(i),"_"," ");              %Removes the underscores "_" from
the factors for the legend
         figure;
         title(strFactor + " Per Day Per "+ sprintf("%.0f",parts) + " Up To " +
datestr(max(vaxTable.administered_date))+ "(Top" + sprintf(" %.0f ",number)+ "Most Populous)")
%Adds the title. Lets you know how many counties, and most recent date.
         ax=gca;
         ax.YGrid = 'on';
         for i = 1:length(countyNames)                        %For this one factor, plots ALL top
populous counties' stats.
            plotVal = movmean(vaxNumbers(:,i),moving);        %Creates a rolling avereage
of the data. You can choose how many days to make a rolling average. I chose 7.
            hold on                                           %Allows for the other counties' lines to show
up on the same plot.
            plot(vaxTable{find(vaxTable.county ==
countyNames(i)),"administered_date"},plotVal,'DisplayName',string(countyNames(i))) %Plots a
county's factor stats per day.
            legend show
            legend('Location','northwest')
```

```
            title(legend,sprintf("%.0f",moving)+' Day Rolling Average','FontSize',12) %Titles the
legend, letting you know that it is a rolling average
        end
        xlim([min(vaxTable.administered_date) max(vaxTable.administered_date)]) %Sets the x
limits from the minimum date t the latest date
        plotVal = [];                                   %Resets the rolling average array for the
next plot
        hold off                                        %Ends the plot so that no new lines will be
plotted onto the same graph
    end                                                 %Repeats for all factors
  end

%%Scatter Plot
  function [scatterGraph] = scatterPlotter(x,y,county,parts,dates,offset,partTitle)
    figure;
    scatter(x,y)
    text(x+offset,y+offset,string(county))
    xlabel("Percentage of Population Fully Vaccinated")
    ylabel(sprintf("Number Of Cases Per %.0f of Population",parts))
    title(partTitle + " From "+datestr(dates(1))+" To " + datestr(dates(2)))
    hold off
  end

%%Date Remover
  function [newTable] = dateRemover(table,dateVar,dates)
    indecies = [];
    for i = 1:height(table)
      if ismember(table{i,dateVar},dates) == false
        indecies = [indecies,i];
      end
    end
    table(indecies,:) = [];
    newTable = table;
  end

%%Heatmap Along Dates
  function [matrixHeat] =
matrixHeatmapDates(popTable,dates,counties,names,vaxTable,tableDates,variable,partTitle)
    fully_vac_day = NaN(height(popTable),length(dates));
    for i = 1:height(popTable)
      countyIndecies = find(counties == names(i));
      factorIndecies = [];
      for ii = 1:length(countyIndecies)
        for iii = 1:length(dates)
```

```
            if vaxTable{countyIndecies(ii),tableDates} == dates(iii)
                factorIndecies(iii) = iii;
            end
        end
    end
    factorIndecies = factorIndecies(factorIndecies~=0);
    for ii = 1:length(factorIndecies)
        fully_vac_day(i,factorIndecies(ii)) = vaxTable{countyIndecies(ii),variable};
    end
end
heatmap(dates,names,fully_vac_day,"Colormap",turbo);
caxis([0 100]);
title(partTitle + " Up To " + datestr(max(dates)))
end

%%Heatmap Along Manufacturer
function [matrixHeat] =
matrixHeatmapTotalManufacturer(manufacturers,counties,names,vaxTable,manufacturersList,titl
ePart)
    fully_vac_manuf = NaN(length(names),length(manufacturers));
    for i = 1:length(names)
        countyIndex = find(counties == names(i));
        factorIndecies = [];
        for ii = 1:length(countyIndex)
            for iii = 1:length(manufacturers)
                if isnan(vaxTable{countyIndex(ii),manufacturers(iii)}) == false
                    factorIndecies(iii) = iii;
                end
            end
        end
        for ii = 1:length(factorIndecies)
            if factorIndecies(ii) ~= 0
                fully_vac_manuf(i,factorIndecies(ii)) = vaxTable{countyIndex,manufacturers(ii)};
            end
        end
    end
    heatmap(manufacturersList,names,fully_vac_manuf,"Colormap",turbo);
    caxis([0 100]);
    title(titlePart + " Up To " + datestr(max(vaxTable.administered_date)))
end
```