```
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
        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,"Cu
        [topVaxPercentageIndex,topVaxPercentageTable] =
 indexSorting(vaxPercentageTableShort,vaxPercentagesNames,vaxPercentageTableShort.

        topPopPerc = sortrows(population,"percentages","descend");
        topPopPerc(topNumber+1:height(topPopPerc),:) = [];
```

```
        topPopPercName = [categorical(topPopPerc.County)]';
        barGraphOutput(topVaxPercentageIndex,vaxPercentagesNames,
["percentages"],topNumber,topVaxPercentageTable,"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_fu
        vaccines.percentagesPerDay = [percentagesPerDay];
        [vaccines,percentagesPerDayPfizer] =
 percentPerTimeFunc(vaccines,vaccines.County,population,popPercName,"Cumulative_Pf
        vaccines.percentagesPerDayPfizer = [percentagesPerDayPfizer];
        [vaccines,percentagesPerDayModerna] =
 percentPerTimeFunc(vaccines,vaccines.County,population,popPercName,"Cumulative_Mo
        vaccines.percentagesPerDayModerna =
 [percentagesPerDayModerna];
        [vaccines,percentagesPerDayJJ] =
 percentPerTimeFunc(vaccines,vaccines.County,population,popPercName,"Cumulative_J&
        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(allNam
        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,vacci
     Of Population Fully Vaccinated Per Day")
        figure;

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

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

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

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

 matrixHeatmapTotalManufacturer(["percentagesPerDayPfizer","percentagesPerDayModer
["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,va
 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))
 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,par
        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 Dates
    function [matrixHeat] =
 matrixHeatmapTotalManufacturer(manufacturers,counties,names,vaxTable,manufacturer
        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
```

Published with MATLAB® R2020b