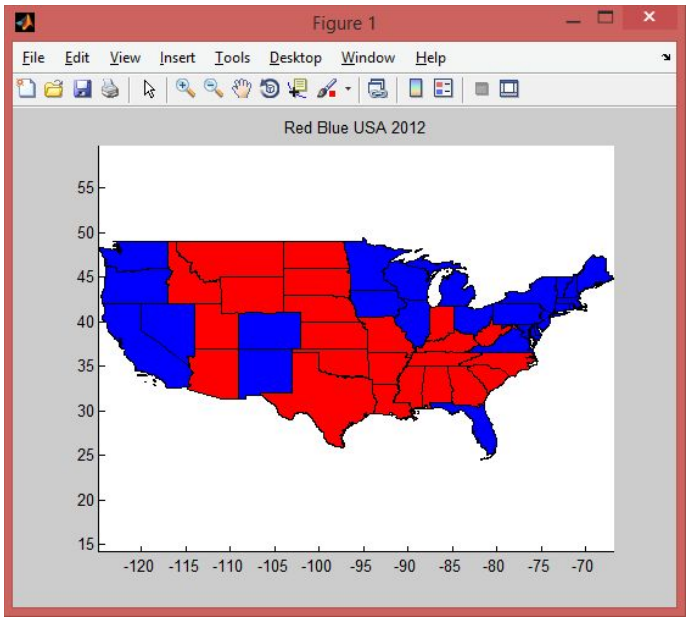
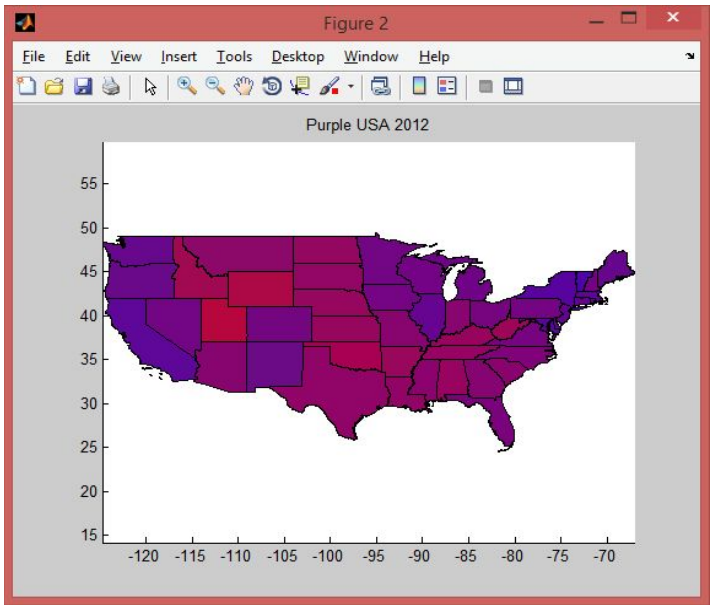
Election Maps

you will write a program that visualizes U.S. presidential election results. During coverage of the 2000 presidential election, Tim Russert (NBC) coined the political terms *red states* and *blue states* to refer to states that predominantly vote for the Republican presidential candidate (red) or the Democratic presidential candidate (blue). Typically the news media use red-state blue-state maps to display election results as shown in Figure 1. For this project, you will also create a more refined (and less polarizing) purple map based on the ideas of Robert Vanderbei as shown in Figure 2.





1. Requirements

Your main script file should be named **electionMapper.m**

1.1. Geographic Data

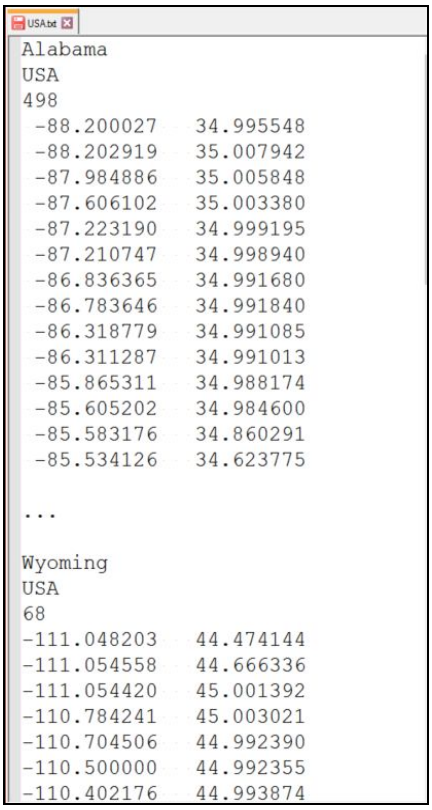
We provide geographic data files (sourced from the U.S. Census) that describes the boundary of each state and county in the United States. In the files, there is one block for each region (with a blank line to separate blocks):

- The first line of a block is a string that is the name of the **subregion (state or county)**.
- The second line of a block is a string that is the name of the **region (USA or state)**.
- The third line of a block is an integer *N* that specifies the number of points in the polygon describing the **subregion**.
- The remaining *N* lines of the block describe the polygonal boundary, given as *N* pairs of real numbers, representing the longitude and latitude coordinates.

Note that the number of subregions in the file **USA.txt** is not 50 for two reasons: first, we do not include either Alaska nor Hawaii; second, we include an entry for each polygonal subregion—some states (such as Michigan, Florida, and California) comprise several polygonal subregions.

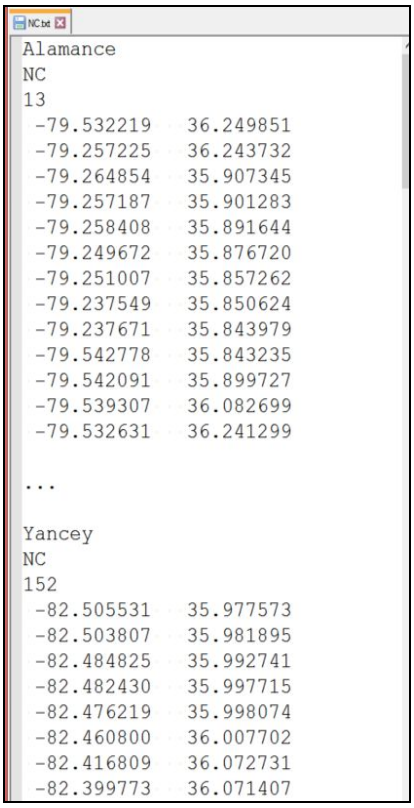
Here are example excerpts from three files with geographic data.

USA.txt



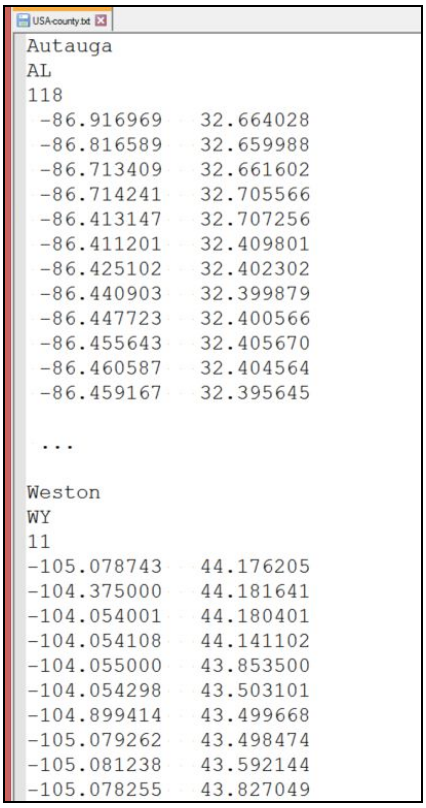
```
Alabama
USA
498
 -88.200027    34.995548
 -88.202919    35.007942
 -87.984886    35.005848
 -87.606102    35.003380
 -87.223190    34.999195
 -87.210747    34.998940
 -86.836365    34.991680
 -86.783646    34.991840
 -86.318779    34.991085
 -86.311287    34.991013
 -85.865311    34.988174
 -85.605202    34.984600
 -85.583176    34.860291
 -85.534126    34.623775

...

Wyoming
USA
68
-111.048203    44.474144
-111.054558    44.666336
-111.054420    45.001392
-110.784241    45.003021
-110.704506    44.992390
-110.500000    44.992355
-110.402176    44.993874
```

NC.txt



```
Alamance
NC
13
 -79.532219    36.249851
 -79.257225    36.243732
 -79.264854    35.907345
 -79.257187    35.901283
 -79.258408    35.891644
 -79.249672    35.876720
 -79.251007    35.857262
 -79.237549    35.850624
 -79.237671    35.843979
 -79.542778    35.843235
 -79.542091    35.899727
 -79.539307    36.082699
 -79.532631    36.241299

...

Yancey
NC
152
 -82.505531    35.977573
 -82.503807    35.981895
 -82.484825    35.992741
 -82.482430    35.997715
 -82.476219    35.998074
 -82.460800    36.007702
 -82.416809    36.072731
 -82.399773    36.071407
```

USA-county.txt

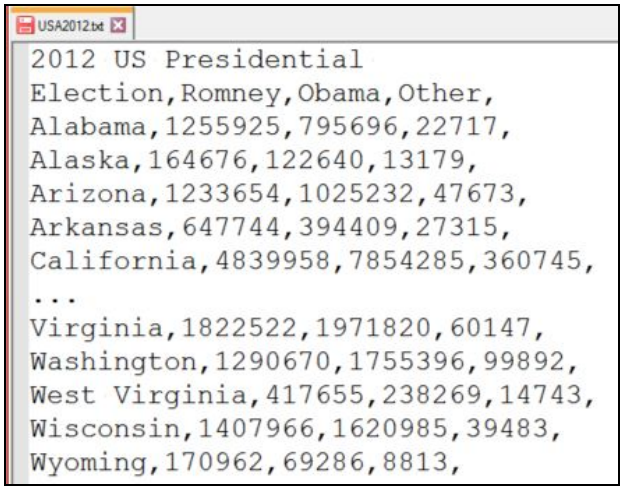


```
Autauga
AL
118
 -86.916969    32.664028
 -86.816589    32.659988
 -86.713409    32.661602
 -86.714241    32.705566
 -86.413147    32.707256
 -86.411201    32.409801
 -86.425102    32.402302
 -86.440903    32.399879
 -86.447723    32.400566
 -86.455643    32.405670
 -86.460587    32.404564
 -86.459167    32.395645

 ...

Weston
WY
11
-105.078743    44.176205
-104.375000    44.181641
-104.054001    44.180401
-104.054108    44.141102
-104.055000    43.853500
-104.054298    43.503101
-104.899414    43.499668
-105.079262    43.498474
-105.081238    43.592144
-105.078255    43.827049
```

1.2. Election Results Data

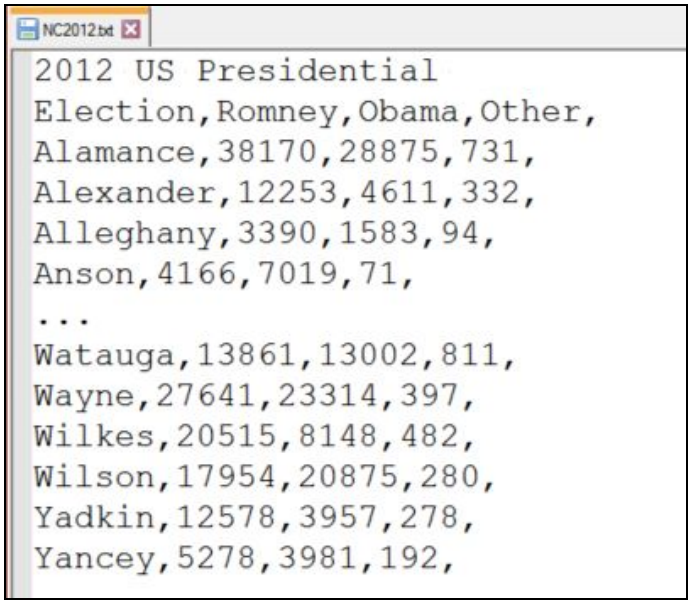
We also provide election results data (sourced from Dave Leip's Atlas of U.S. Presidential Elections) that describes the results for each presidential election, by state and county. In these files, each row consists of four fields, separated by commas: the name of a subregion, the number of votes for the Republican candidate, the number of votes for the Democratic candidate, and the number of votes for the Independent (or third party) candidate. Here are excerpts from two election result files.

USA2012.txt



```
2012 US Presidential
Election,Romney,Obama,Other,
Alabama,1255925,795696,22717,
Alaska,164676,122640,13179,
Arizona,1233654,1025232,47673,
Arkansas,647744,394409,27315,
California,4839958,7854285,360745,
...
Virginia,1822522,1971820,60147,
Washington,1290670,1755396,99892,
West Virginia,417655,238269,14743,
Wisconsin,1407966,1620985,39483,
Wyoming,170962,69286,8813,
```

NC2012.txt



```
2012 US Presidential
Election,Romney,Obama,Other,
Alamance,38170,28875,731,
Alexander,12253,4611,332,
Alleghany,3390,1583,94,
Anson,4166,7019,71,
...
Watauga,13861,13002,811,
Wayne,27641,23314,397,
Wilkes,20515,8148,482,
Wilson,17954,20875,280,
Yadkin,12578,3957,278,
Yancey,5278,3981,192,
```

You can download all of the geographic and election result data files collectively as **purple-america-data.zip** from Moodle.

1.1. Interacting with the User

Upon execution of your main script file **electionMapper.m**

1. the user is prompted to pick what geographical region to analyze using a **menu**. The options in the **menu** are USA, State or USA-county.
   - If the user selects a state then he/she is prompted in the Command Window for the state abbreviation.
   - If the user types an incorrect state abbreviation he/she is re-prompted until a correct state abbreviation is entered. Both uppercase and lowercase are accepted.
2. the user is then prompted to pick an election year to analyze via a **menu** from 1960 to 2012.
3. Finally, a message is printed in the Command Window, showing the user selection.

1.2. Calculating the color for each region/subregion

Before you generate the maps, you need to calculate both the red/blue and purple coloring for each region.

1.2.1. Calculate the Red/Green/Blue Color

For the red/blue map, you need to calculate whether to color the region/subregion blue, red or green based on the maximum number of votes. To do this you are **required to write and use** the following user defined function. Don't change the number of input and output arguments or their data types. For this function, they are scalars of type double.

```
function [R, G, B] = getPrimaryColor( republicanVotes,democratVotes, otherVotes )
%determines the color of a region/subregion based on max number of votes
%Inputs: republicanVotes is a double for the republican votes
%        democratVotes is a double for the democrat votes
%        otherVotes is a double for the independent/third party votes
%Returns: R the red -- 1 if republicanVotes is max, 0 otherwise
%         G the green -- 1 if otherVotes is max, 0 otherwise
%         B the blue -- 1 if democratVotes is max, 0 otherwise
%         if two votes are equal, assign 0.5 to their color, i.e. republicanVotes is equal to
%         democratVotes then R is set to 0.5 and B is set to 0.5
%         if three votes are equal, assign 0.33 to their color
```

Here are examples of calling this function

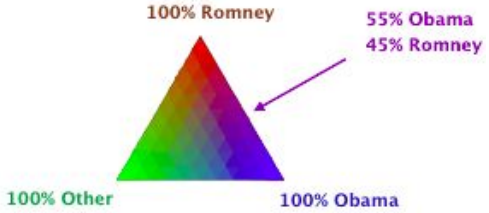
```
>> [R, G, B] = getPrimaryColor( 100, 50, 2 )
R =
     1
G =
     0
B =
     0
>> [R, G, B] = getPrimaryColor( 50, 100, 2 )
R =
     0
G =
     0
B =
     1
>> [R, G, B] = getPrimaryColor( 50, 100, 200 )
R =
     0
G =
     1
B =
     0
```

```
>> [R, G, B] = getPrimaryColor (5, 5, 1)
R =
    0.5000
G =
     0
B =
    0.5000
>> [R, G, B] = getPrimaryColor (5, 3, 5)
R =
    0.5000
G =
    0.5000
B =
     0
>> [R, G, B] = getPrimaryColor (5, 5, 5)
R =
    0.3300
G =
    0.3300
B =
    0.3300
```

1.2.2. Calculate the Purple Color

For the purple map, you need to calculate the shade of color for the region/subregion. Specifically, if the Republican, Independent, and Democratic candidates receive a_1, a_2, and a_3 votes, respectively, then the color is calculated using the following formula:

$$(R,\ G,\ B) = \left(\frac{a_1}{a_1 + a_2 + a_3},\ \frac{a_2}{a_1 + a_2 + a_3},\ \frac{a_3}{a_1 + a_2 + a_3} \right)$$



To do this you are **required to write and use** the following user defined function. Don't change the number of input and output arguments or their data types. For this function, they are scalars of type double.

```
function [R, G, B] = getPurpleColor( republicanVotes,democratVotes, otherVotes )
%determines the color of a region/subregion based on the proportion of votes
%received by republican, democrat or independent candidate
%Inputs: republicanVotes is a double for the republican votes
%        democratVotes is a double for the democrat votes
%        otherVotes is a double for the independent votes
%Returns: R is red between 0 and 1
%         G is green between 0 and 1
%         B is blue between 0 and 1
```

Here is an example of calling this function

```
>> [R, G, B] = getPurpleColor( 100, 100, 200 )
R =
    0.2500
G =
    0.5000
B =
    0.2500
```

1.3. Generating the plots

Finally, the script must generate two figures: a blue/red map and a purple map. The functions to do this are provided, you just need to figure out how to use them.

1.3.1. Get the boundary geolocation points

First, get the boundary points for the region of interest using the provided **getBoundaryDataFromFile.m** function. This function reads from the geographic data files and generates a structure array, which is used for the plotting. Here is how to use this function and what it outputs.

```
>> mapBoundaries = getBoundaryDataFromFile('data/NC.txt')
mapBoundaries =
1x104 struct array with fields:
    regionName
    longitude
    latitude
>> mapBoundaries(1).regionName
ans =
NC_alamance
>> size(mapBoundaries(1).longitude)
ans =
     1    13
>> size(mapBoundaries(1).latitude)
ans =
     1    13
```

For **NC.txt** there are 104 subregions/counties

The first one is Almance county and its boundary
Is described by 13 points

```
>> mapBoundaries = getBoundaryDataFromFile('data/USA.txt')
mapBoundaries =
1x103 struct array with fields:
    regionName
    longitude
    latitude
>> mapBoundaries(1).regionName
ans =
USA_alabama
>> size(mapBoundaries(1).longitude)
ans =
     1   498
>> size(mapBoundaries(1).latitude)
ans =
     1   498
```

For **USA.txt** there are 103 subregions

The first one is for Alabama and its boundary
is described by 498 points

```
>> mapBoundaries = getBoundaryDataFromFile('data/USA-county.txt')
mapBoundaries =
1x3200 struct array with fields:
    regionName
    longitude
    latitude
>> mapBoundaries(1).regionName
ans =
AL_autauga
>> size(mapBoundaries(1).longitude)
ans =
     1   118
>> size(mapBoundaries(1).latitude)
ans =
     1   118
```

For **USA- county.txt** there are 3200 counties

The first one is for Alabama, Autauga its boundary
is described by 118 points

1.4. Create the maps

Next, create the maps using the provided **plotMap.m** function, which has the following specification:

```
function plotMap( regionColor, mapBoundaries, plotTitle)
%generates a map of the given region using the given colors
% Inputs:regionColor is a structure array that contains the color of each
%                    region/subregion.
%        mapBoundaries is the structure array generated by
%                    the provided function getBoundaryDataFromFile.m
%        plotTitle is a string for the map's title
% Returns: creates a plot
```

The first input argument of this function is **regionColor,** a structure array based on whether the program is generating a map for a particular state, USA or for all the USA counties. Note, the structure has two fields: **regionName** and **color**. You will using the functions for Sec 1.2 and Sec 1.3 to create the structure array **regionColor**. Below, are three different example structure arrays that could be passed as the first argument **regionColor** to this function.

```
K>> regionPurpleColor
regionPurpleColor =
1x100 struct array with fields:
    regionName
    color
K>> regionPurpleColor(1).regionName
ans =
NC_alamance
K>> regionPurpleColor(1).color
ans =
    0.5417    0.0090    0.4494
```

For **NC** for 2012
for the purple map

There are 100 subregions (counties) and
They are labeled **NC_** and **lowercase county name**

```
K>> regionPrimaryColor
regionPrimaryColor =
1x51 struct array with fields:
    regionName
    color
K>> regionPrimaryColor(1).regionName
ans =
USA_alabama
K>> regionPrimaryColor(1).color
ans =
     1     0     0
```

For **USA** for 1988
For the red/blue map

There are 51 regions and they are
labeled as **USA_** and **lowercase state name**

```
K>> regionPurpleColor
regionPurpleColor =
1x3129 struct array with fields:
    regionName
    color
K>> regionPurpleColor(1).regionName
ans =
AL_autauga
K>> regionPurpleColor(1).color
ans =
    0.4646         0    0.5354
K>> regionPurpleColor(100).regionName
ans =
AZ_navajo
K>> regionPurpleColor(100).color
ans =
    0.5727         0    0.4273
```

For **USA-county** for 1960
for the purple map

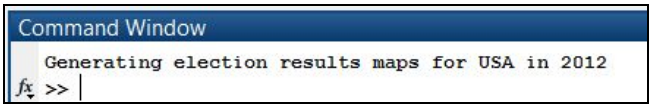
There are 3129 subregions (counties) and
They are labeled **uppercase state name underscore** and **lowercase county name**. It is important to do this because different states have the same county names.

1.5. Example Output

Below are some three example outputs upon the execution of the script **electionMapper.m** In each run, first a **menu** pops up with the different regions to analyze and then the user is asked what election year to analyze, again via a **menu**

Example Run 1:

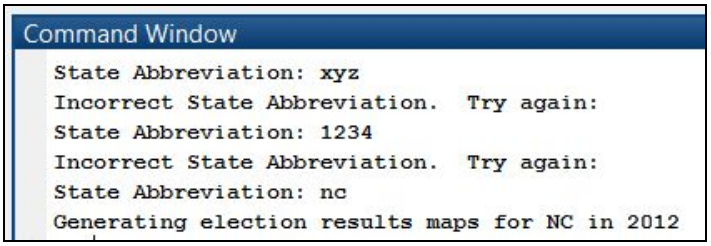
Suppose the user picked USA for 2012. Here is the message printed in the Command Window that indicates what was picked.



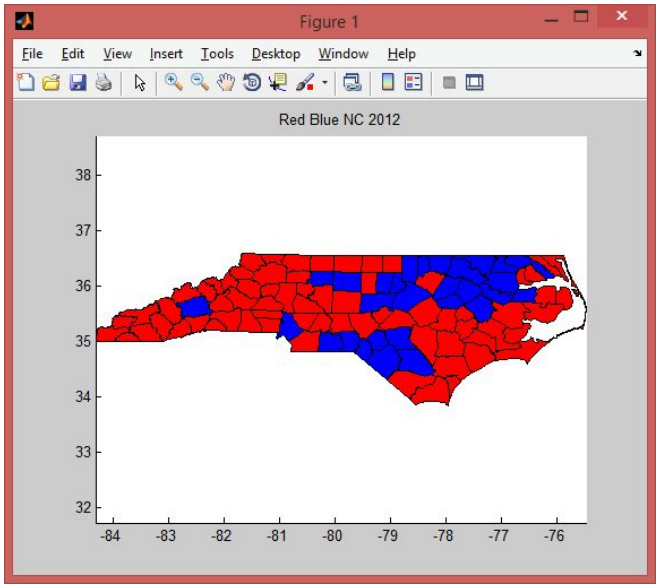
And then Figure 1 and Figure 2 are generated as shown on the first page of this document.

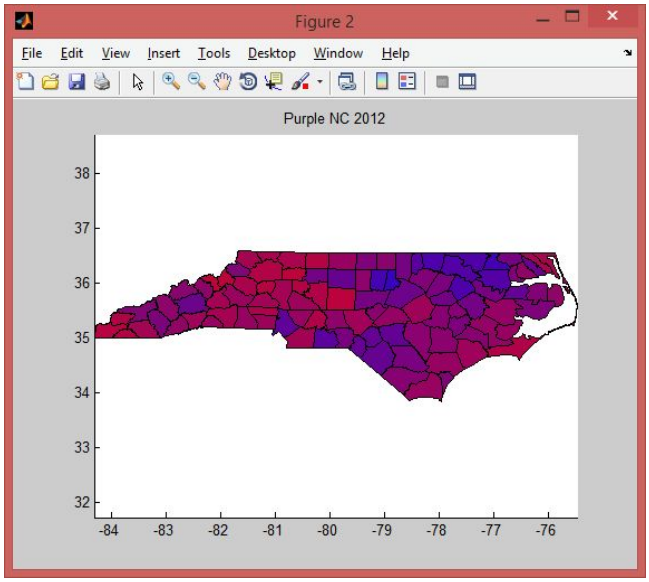
Example Run 2:

Suppose the user picked NC for 2012. In the Command Window, the user is prompted for the State Abbreviation. If the user types in an incorrect abbreviation, the program re-prompts until a correct one is entered.



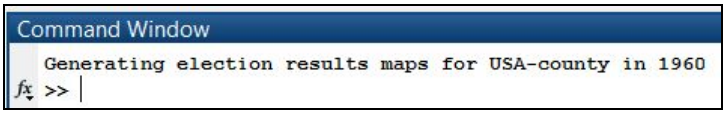
Then these two plots are generated:



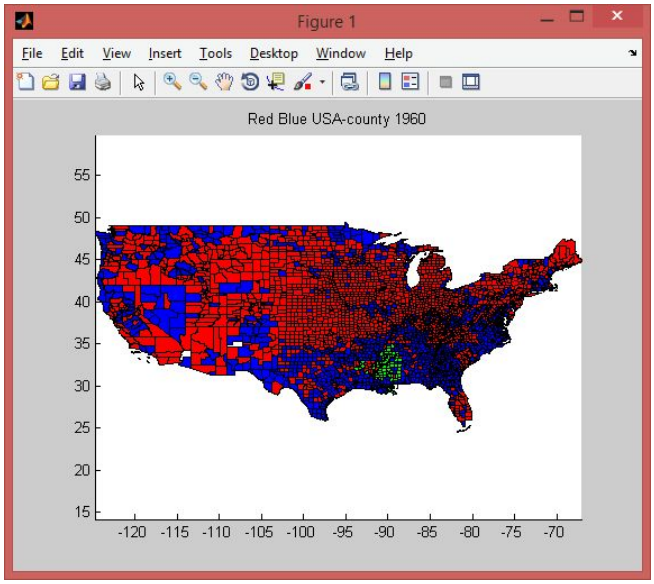


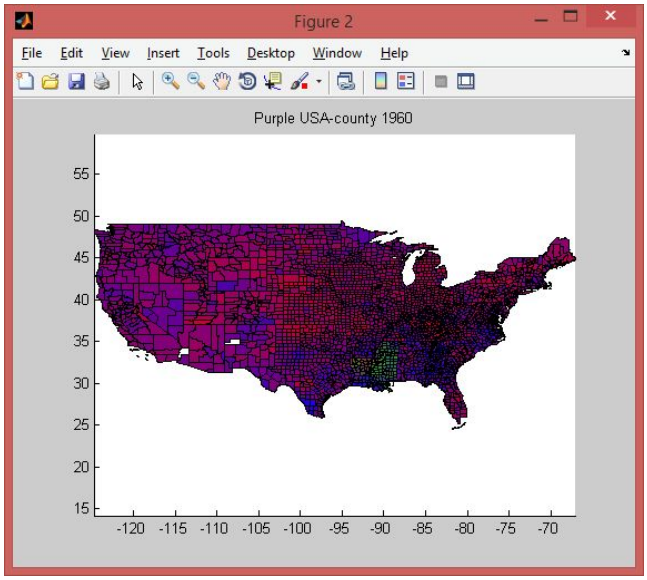
Example Run 3:

Suppose the user picked USA-county for 1960.



Then these two plots are generated:





2. Implementation Details

- The provided functions: **getBoundaryDataFromFile.m** and **plotMap.m**, have to be saved in the same directory as your main script **electionMapper.m**
- All the data files provided must be saved in a sub-directory called **data**. Thus if you need to open the file called **USA.txt**, you will need to use the string **'data/USA.txt'**
- You **must** write and use the two user-defined functions **getPrimaryColor.m** and **getPurpleColor.m** and they have to meet the given specifications.
- If you like, you can write any additional user-defined functions (this is not required).
- Here are a few built-in functions you might find useful: **ismember**, **num2str**, **str2num**, **upper**, **lower**
- Also, note that you can do string concatenation using square brackets [ ]

3. Styling Directions:

At the top of **EACH m-files that you write**, add the following information:

% Name (s)
% Date
% Lab Section (s) #
% Project 3: Election Mapper, 2016 sp

Make sure that you suppress all *unnecessary* output.

th

You need to submit at least **three** files.

- **electionMapper.m**
- **getPrimaryColor.m**
- **getPurpleColor.m**

If you write any additional functions, you need to submit those as well.

Please take care of following point to help me to undersand better

		Description
User Input		
		menu to determine region to analyze
		error checking for State abbreviations
		menu to pick year
		Correctly figuring out the names of the needed data files
		Printing user's selections of region & year to Command Window
Determining the Color of the regions		
		Correctly opening the needed data files
		Correctly reading & tokenizing the input from the data files
		getPrimaryColor.m written and used correctly
		getPurpleColor.m written and used correctly
		Correctly creating the structure array needed for the first argument of plotMap.m
Generating the Maps		
		Correctly using **getBoundaryDatFromFile.m** to generate **mapBoundaries** structure array
		Using **plotMap.m** to generate the correct red/blue plot
		Using **plotMap.m** to generate the correct purple plot

		Description
		Name(s), date, section(s), etc. not included at the top of each .m file
		Incorrectly Named Files
		Input data files NOT placed in a sub-directory **data**
		Code is not well commented
		Code is not cell blocked
		Please dont do Hardcoding
