Marco Radaelli and Matt Moderwell

Assignment 3: Interactive Visualization
Interactive U.S. County Map

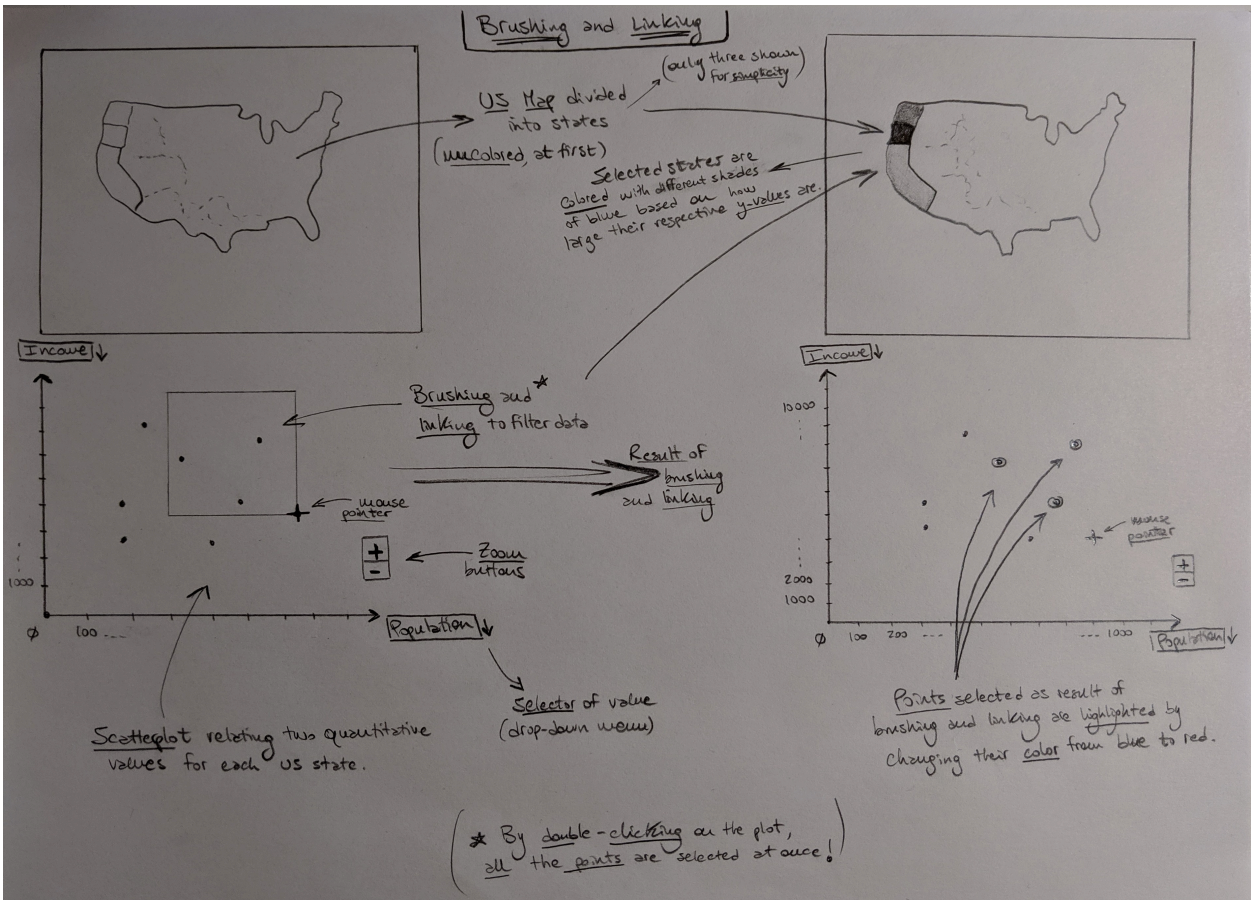
Introduction

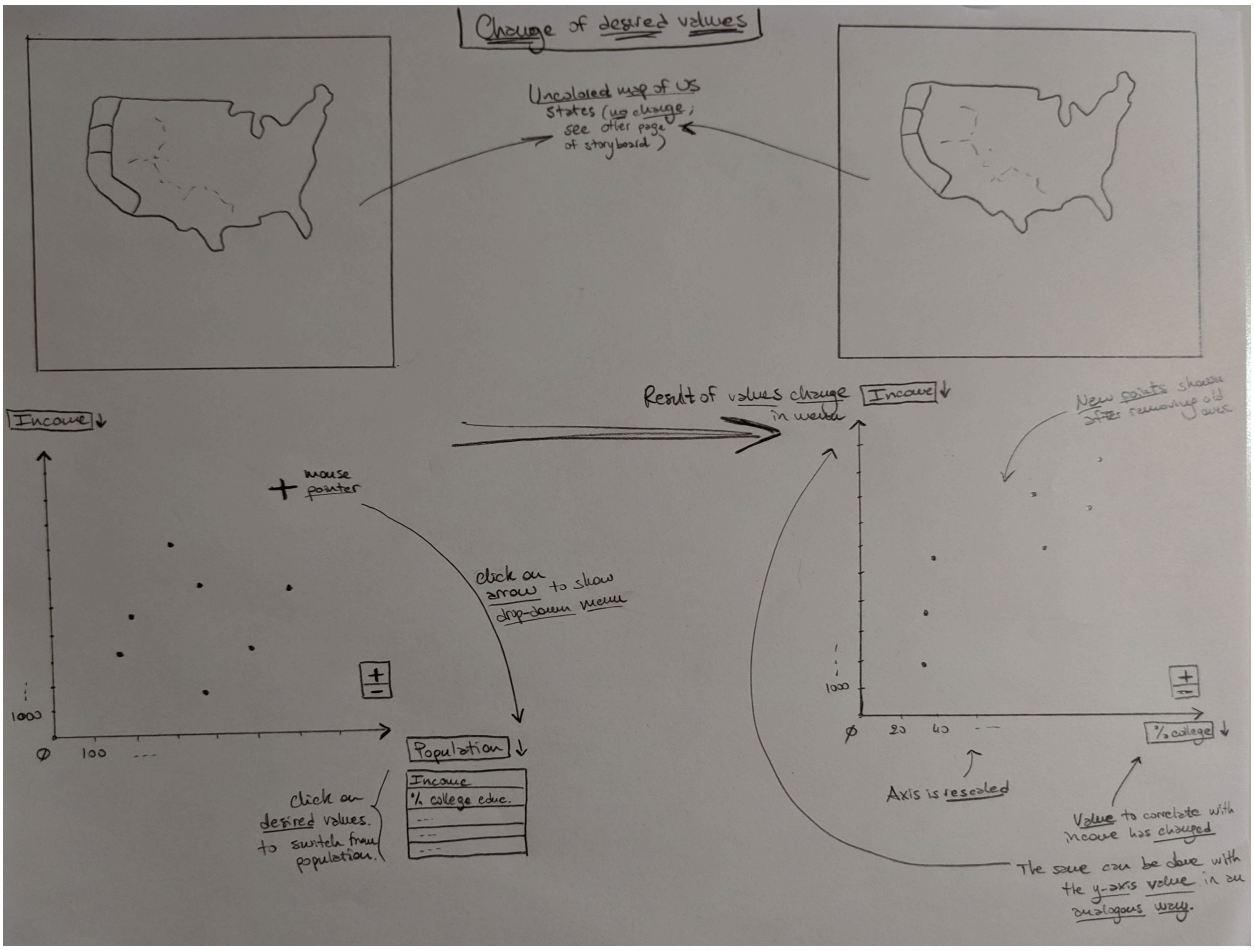
Whether the goal is to uncover specific demographic patterns, look for possible causation relationships, or simply to collect additional information about the area we live in, looking for the correlation between two selected variables on a geographical level can provide incredible insights. Our goal with this project is to allow people to dynamically uncover the relationships between any two demographic variables for the US population, while mapping such data on a representation of the whole Nation in order to locate regions which show specific behaviors. Drawing data from the *US Census Bureau*, our interactive implementation of the above — presented in more detail below — is to be accompanied by the present documentation describing its development process and functionalities.

Storyboards and data domain

As mentioned in the *Introduction* section, the data we used was sourced from the *US Census* website, census.gov. The website is the largest public container of demographic data on the US population, providing freely downloadable datasets covering everything from employment statistics to migration patterns, from sex distributions to population densities, and much more.
Each individual dataset on the website can only be downloaded in JSON format, so we wrote a script that turns any input dataset into a *.csv* format dataset and merges it with all the other datasets previously downloaded, if any. This allows us to construct a final, aggregate dataset relating each and every demographic variable chosen. Furthermore, this is also infinitely expandable as more datasets become available and/or of interest to the potential users — i.e. it suffices to download another dataset from census.gov, run it through the same script, and add it to the code as needed (see below the *Final visualization web application* section).
In order to map this data onto an effective interactive visualization system, we have devised a single-page website, presented below.

The following storyboards present our conceptual design of the interactive web interface we conceived, which was then implemented with the minor changes highlighted below in the *Final visualization web application* section. The storyboards call attention to the two major interactive functions of our visualization: brushing and linking; drop-down menu selector.


Brushing and Linking
US Map divided into states
(only three shown for simplicity)
(uncolored at first)
Selected states are colored with different shades of blue based on how large their respective y-values are
Income
Brushing and linking to filter data
Result of brushing and linking
mouse pointer
Zoom buttons
Population
Selector of value (drop-down menu)
Scatterplot relating two quantitative values for each US state.
Points selected as result of brushing and linking are highlighted by changing their color from blue to red.
(* By double-clicking on the plot, all the points are selected at once!)


Change of desired values
Uncolored map of US states (no change; see other page of storyboard)
Result of values change in menu
Income
mouse pointer
click on arrow to show drop-down menu
Population
Income
% college educ.
click on desired values to switch from population.
New points shown after removing old ones
% college
Axis is rescaled
Value to correlate with income has changed
The same can be done with the y-axis value in an analogous way.

As the above drawings show, our interactive visualization will be made up of two neighboring charts: one map of the US, non-interactive and segmented into the individual states; an interactive scatterplot relating two demographic variables. The former will be used to show the differences between states (colored with different shades of blue) with regards to the selected y-axis variable in the scatterplot below the map itself. The latter, instead, will allow users — through the brushing-and-linking technique — to select a specific subset of states whose point values are located in a desired region of the scatterplot itself. The result of this interaction will automatically be mirrored by the map through a coloring of the selected states in order to show which ones are currently being considered. Other possible interactions with the scatterplot will include a zoom button which allows users to focus on a specific area of the graph, making it easier to discern between points which would otherwise appear too close to one another. Furthermore, a double-click of the graph area will let users select all the points at once in order to look at macro trends across the entire US.
The second major interaction technique implemented on the scatterplot will be the drop-down menus allowing users to select which demographic variables to plot both on the x-axis and the y-axis. Any change in the variables will automatically be mirrored by the map, which will change coloring accordingly. The two axes will also be rescaled as needed.

We believe that the above visualization with its listed functionalities will let users easily and quickly answer most of the geo-demographic questions they might have for the following reasons:

1. The map-scatterplot pairing allows users to simultaneously see the correlation between two demographic variables and the geographic distribution of one, answering questions like: In which US states do high-income people live? Is there any correlation between income and state population?
2. The brushing-and-linking technique, whose results are mirrored by the map, allows users to easily focus on geographic areas characterized by a specific pair of values, answering questions like: Which high-income states have a population of over 10 million people?
3. The drop-down menus allow users to answer a much higher set of questions analogous to the ones listed above, for as many demographic variables as desired.

Final visualization web application

Our final web application implements the above design with some minor changes due to challenges faced throughout the development process and limitations posed by the development environment and language (D3.js).
The website can be accessed at any time at the following link: cs396.mmoderwell.com

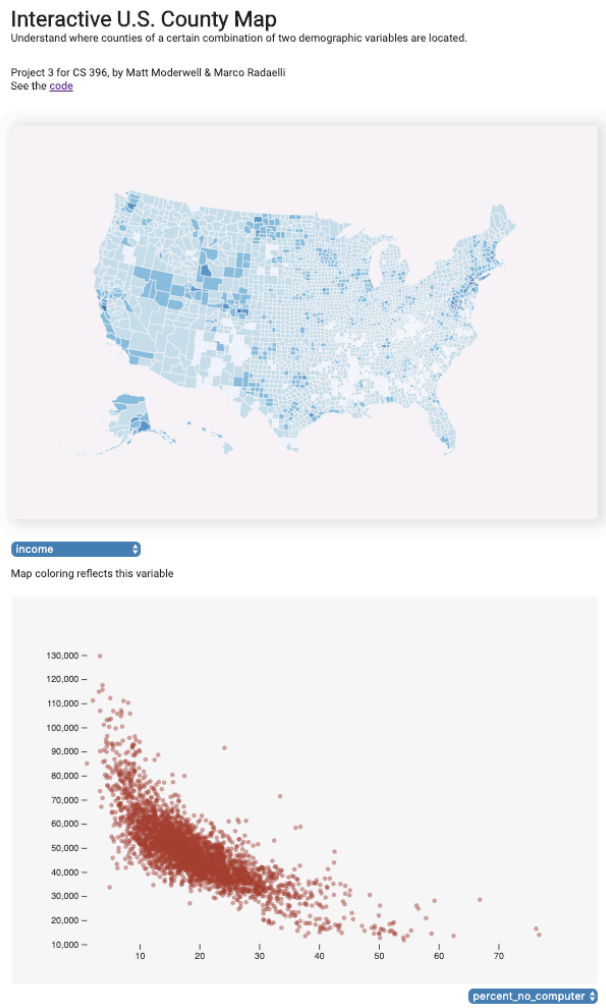
What follows is a description of our implementation, with a focus on the differences from the storyboards included in the *Storyboards and data domain* section.

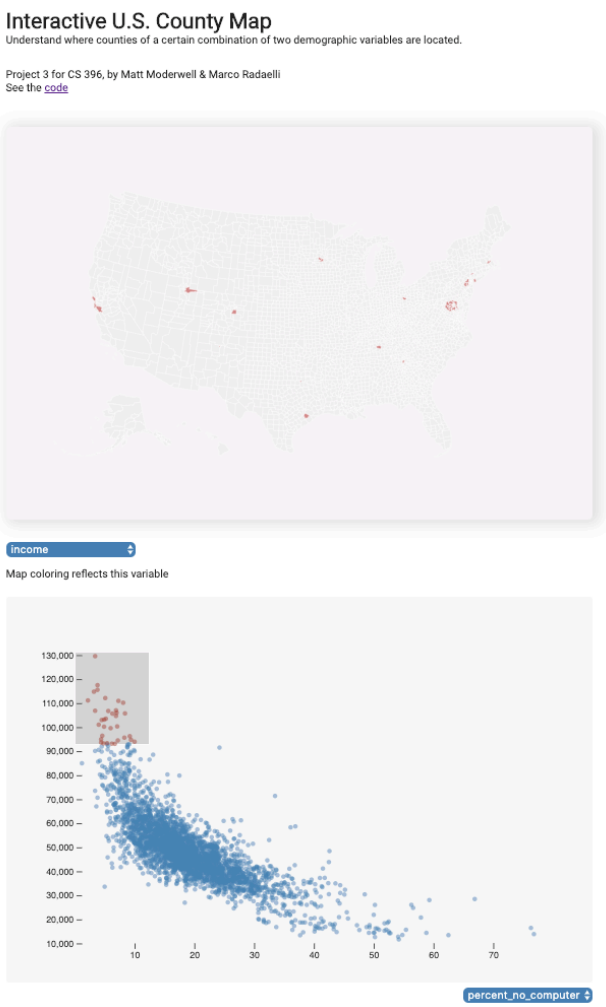
Overall, our final implementation mirrors that conceived in the storyboards: we indeed have a map-scatterplot charts pairing, connected as previously described. However:

1. The level of detail has been increased from states to individual counties. We initially thought that this would increase complexity to excessive levels, but we then discovered that the increase in complexity was compensated for by the additional information obtainable from the map. In fact, for some of the data variables we selected, such as *median_commute*, the differences between large cities and less populated areas of the same state are very noticeable, and leaving such differences out of the analysis would have significantly decreased the usability of our system in answering more in-depth questions.
2. The map now starts out as fully colored (in different shades of blue) based on the selected y-axis variable, but does not keep its colors when brushing-and-linking filtering is performed on the scatterplot. The reason behind this is that we found it more effective to immediately show the whole variable mapping in order for users to instantly decide whether the selected variable is indeed one worth of further investigation. Furthermore, we opted for a uniform color coding (single shade of red) for the counties selected as a result of brushing and linking. This is because we decided to devote the brushing-and-linking interaction to uncovering the subsets of counties which present a specific pairing of values, rather than it being a geographic filter of the initial pair of datasets. Moreover, our scale of different shades of blue used to relate two counties' values of the y-axis variable would have otherwise made the subset of counties with low y-axis values hardly visible, when we instead wanted to make them pop out as a result of the brushing-and-linking.
3. The zoom function was not implemented due to technical difficulties. We were, in fact, able to implement only one of the brushing-and-linking and the zoom function at a time, failing to make them work together. This is, however, the focus of our current development efforts, and the issue should be resolved soon (check cs396.mmoderwell.com regularly to monitor the improvements).

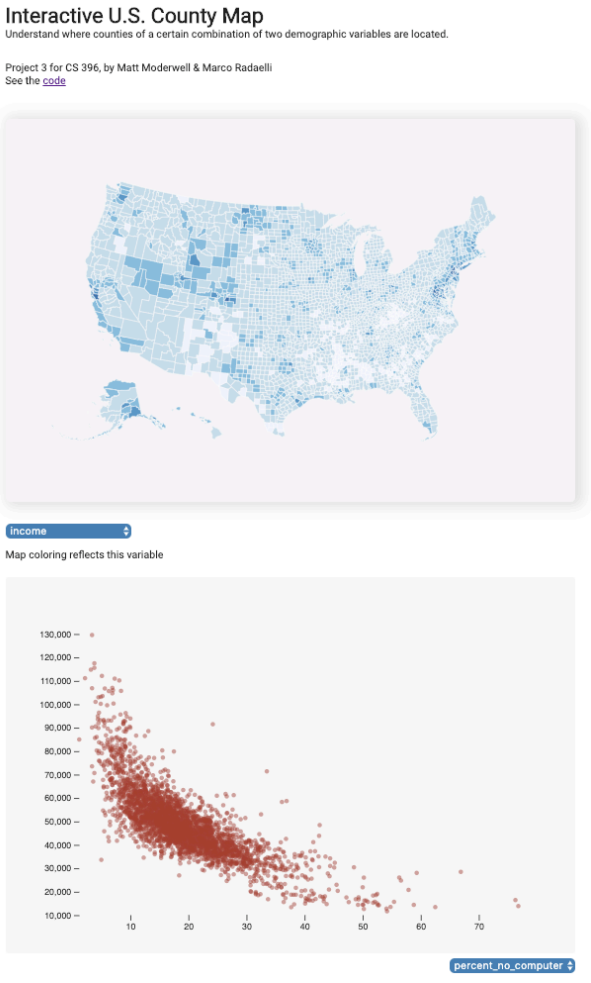
The following screenshots of our interactive visualization, which mirror the logic of our storyboards, show a sample user session:

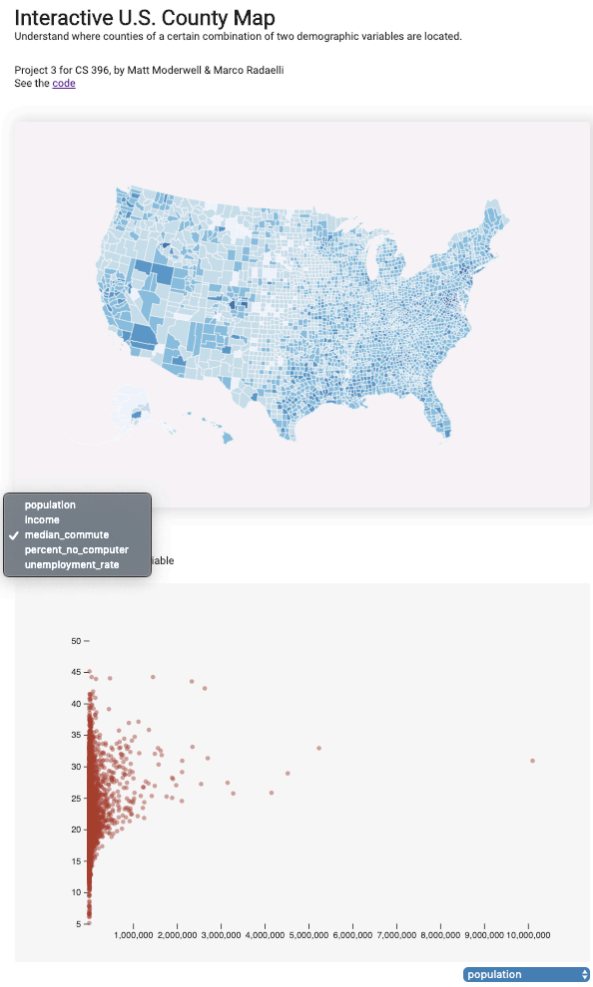
Brushing and linking:





Change of desired variables:





Development process

Since neither of us had any previous hands-on knowledge of D3.js, we found this assignment particularly challenging, as the language requires the implementation of very specific built-in functions. Overall, the entire development process took each of us around 15 hours. Besides the implementation of the zoom function, already discussed in the *Final visualization web application* section above, the most challenging aspects were the following:

1. Updating the charts to reflect a change in the selected variables. In fact, we frequently ran into the issue of old variables not being correctly or completely removed before redrawing new ones, which resulted in wrong visualizations or extremely long processing times due to the amount of data handled at once. After a number of trails, we successfully implemented this functionality by using data variables (*selectedOption*, *selectedOption2*) which are dynamically updated together with the drop-down menu selectors, and correctly taking advantage of the *enter, update, exit* pattern learned in class.
2. Keeping a good average performance when performing brushing-and-linking. In fact, by using county data instead of state data, the space complexity went up significantly, so we needed a way to keep time complexity low. We were able to guarantee this by allowing O(1) data access through the use of a dictionary (js object) for indexing.

In terms of the breakdown of the above, each of us took care of one of the two major interaction techniques, with Matt focusing more on the brushing-and-linking and also guiding the overall process given his additional past experience with web development.