Final Report

CS 4460 - Milestone 7

Georgia Institute of Technology

Team In Sync

Sneha Ganesh

Dakota Lindsey

Eric Morphis

Nikhil Nandish

Goals of project

The primary goal of our project was to create a visualization to show the various trends in music from the 1960's to today. Our specific code implementation goals (which we succeed in) are as follows. We created a Stacked Graph of "Genre" Vs "Time and "Song Length" Vs "Time", where Genre and Song Length were two options given to the user. We also have a bar chart which is linked to the Stacked Graph such that if we click on a particular genre, say Hip Hop, the bar chart will display the 10 most popular artists in that genre along with the number of "Billboard Top 100" songs each of them have had in that genre from 1960 to today. The Bar Chart also has details on demand to display the names, ranks and year of the top 5 songs for each artist. A line chart is linked to the "Song Length" stacked bar chart. This line chart displays the average song length, aggregated across all genres, from 1960 to present. Hovering over the line chart or a year on the stacked bar chart reveals variation in song length over time. Variation can be displayed in three formats: one standard error of the mean (default), standard deviation, and 95% confidence intervals.

Our team goals to complete the project included:

- Team meetings two weeks before each Milestone was due to think through how each one of us was going to contribute for the project.
- Keeping in touch via email about how much each one has progressed in their tasks/responsibilities.
- Setting deadlines to complete our individual parts at least one week before each milestone was due so that we had sufficient time to put everything together.
- Reaching out to the Professor/TA when we needed any guidance and feedback for our project.
- Make sure that the visualization represents our data well so that we are able to answer all the questions accurately.
- Make sure that we have sufficient interaction techniques that would support the visualization and aid the user in better understanding the trends in music.
- Rehearse the presentation in advance so that we as a team, can effectively portray our visualization and project.

Questions

Our questions as of the midterm report were:

- Which artist had the most singles in the Billboard Top 100, by genre?

- What trends in genre can be seen over the years? Which genre was most popular in each year? For example, was rock and roll popular in the 60's? Has metal come back into style since the 90's?
- What variations in song length have we encountered over the years? Which song length is the most popular?

We have revised them to the following:

- How did the popularity of genres evolve relative to one another from 1960 to 2013 (for instance, among hip-hop, rock, and funk, did any become more or less popular relative to the other two)?
- What are the top 10 artists in each genre from 1960 to 2013? What are the top 5 songs by each of these artists?
- Which ranges of song duration were most popular in which time periods from 1960 to 2013?
- How has the statistical distribution of song durations changed from 1960 to 2013?

Data Sources

- http://www.bobborst.com/popculture/top100songsoftheyear/?year=1971

The above link contains the top 100 charting songs for each year from 1960 to 2013.

- http://www.last.fm/api/show/track.getTags

The billboard data was imported into a csv file using python. A list of genre tags for each song was
obtained by using the above API. "last.fm", has hundreds of thousands of genre tags, so we chose the 17 most popular genre tags, and mapped some less popular genres to these. This list of genre tags was then added to the billboard data in our csv file. We only considered the top 3 genre tags for each song. The durations for each song were also obtained from the above API. We further preprocessed this data in python into four csv files so that each view can import the data in a format that requires less processing by d3. The four csv files are as follows:

- ***genre_primary_data***, which contains for each year the number of hits in each of the 17 genres
- ***genre_secondary_data***, which contains a row with information for each song that was made by one of the top 10 artists for each genre, sorted by genre and then by artist and then by rank
- ***length_primary_data***, which contains a row for each year, containing the durations of every hit made during that year
- ***length_secondary_data***, which contains a row for each year, containing mean song duration, standard deviation, standard error, and 95% confidence interval

These four files, as well as our raw data file, are contained in our code zip.

Screen Shots

Screenshot 1

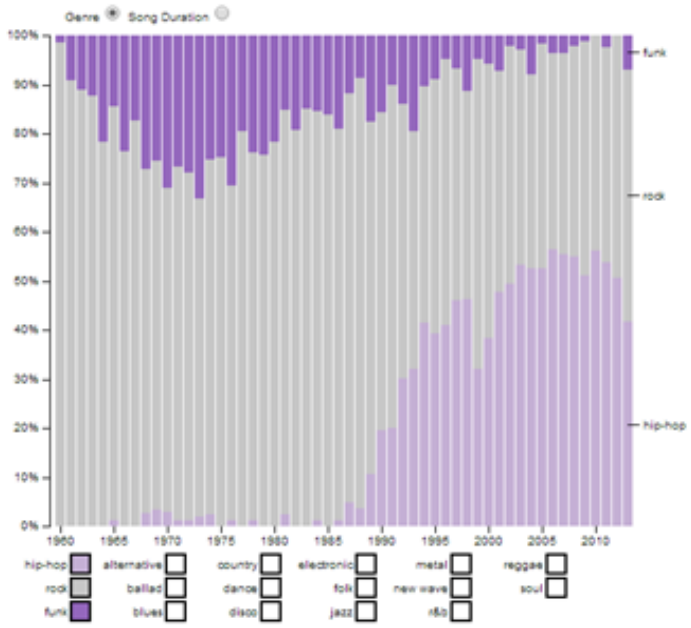

Figure 1: This screen shot displays the default view of the visualization. The “Genre” stacked bar chart is shown with three genres selected: Hip-Hop, Rock, and Funk. Each genre has a unique bar color.

Screenshot 2

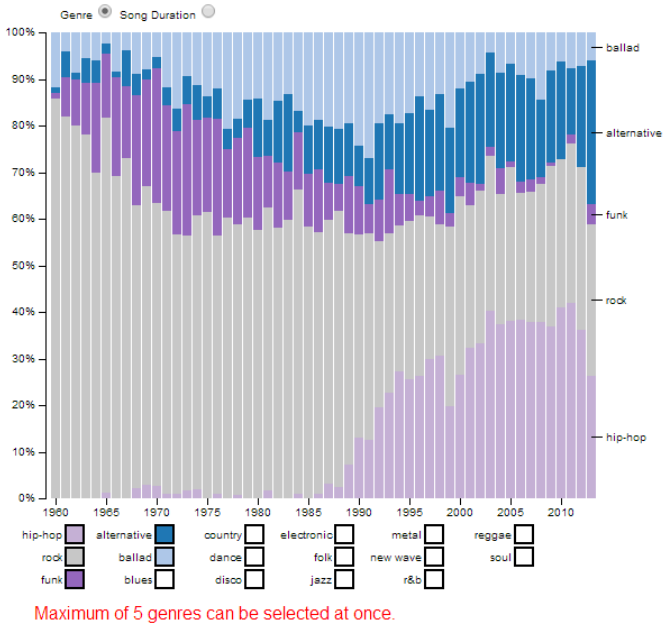

Figure 2: Genres can be added and removed by selecting the checkboxes. The chart will update dynamically. A maximum of 5 genres can be shown simultaneously; the number was limited to prevent clutter.

Screenshot 3

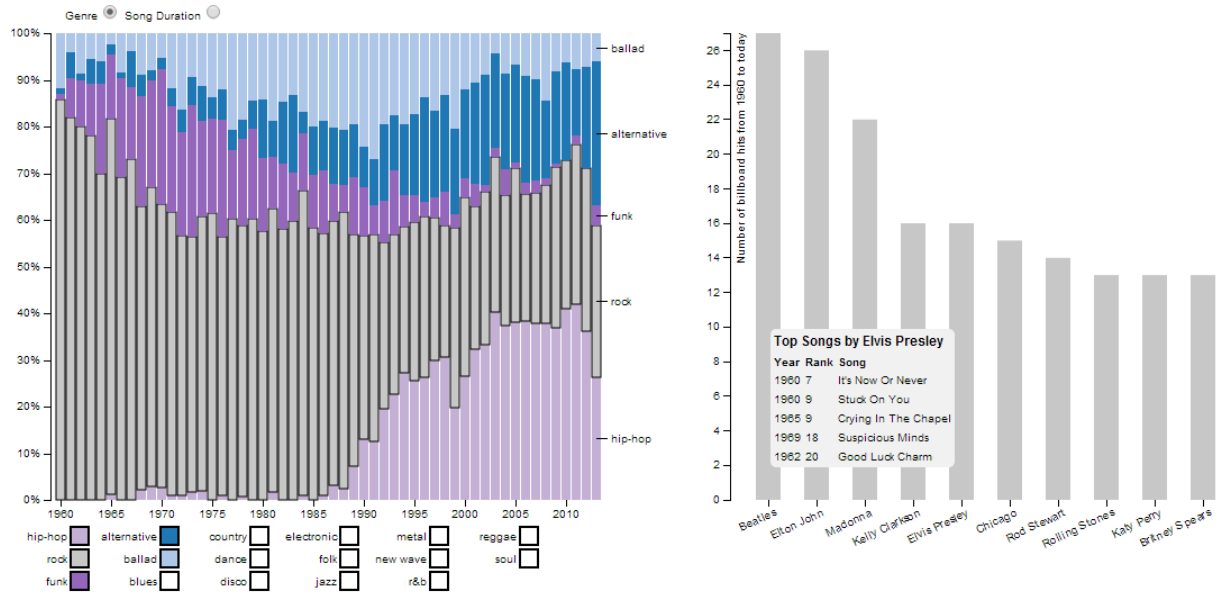

Figure 3: Selecting a genre within the stacked bar chart will display the connected bar chart for that genre. This bar chart shows the ten most popular artists for that genre and the number of billboard songs they have in that genre. Hovering over the bar for an artist will display that artist's five most popular songs.

Screenshot 4:

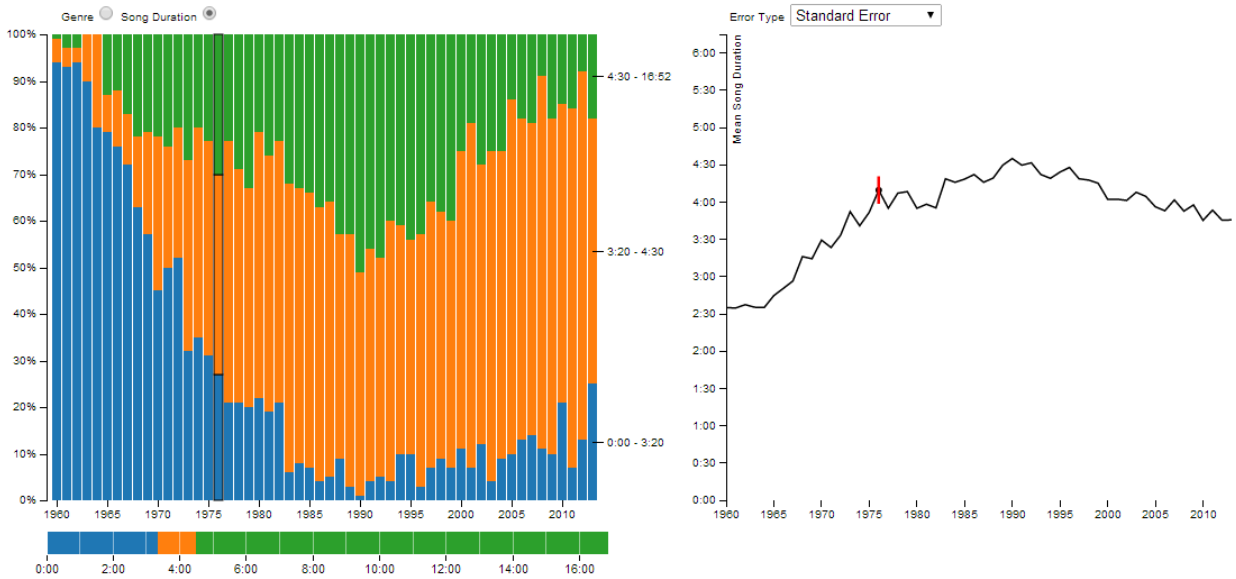

Figure 4: This screen shot displays the default view of the visualization when the “Song Length” radio button is selected at the top. The “Song Length” stacked bar chart is shown on the left, divided into three time intervals. The linked line chart, displaying the mean song lengths, is shown on the right. Hovering over a year in the stacked bar chart will display error bars for that year on the line chart.

Screenshot 5:

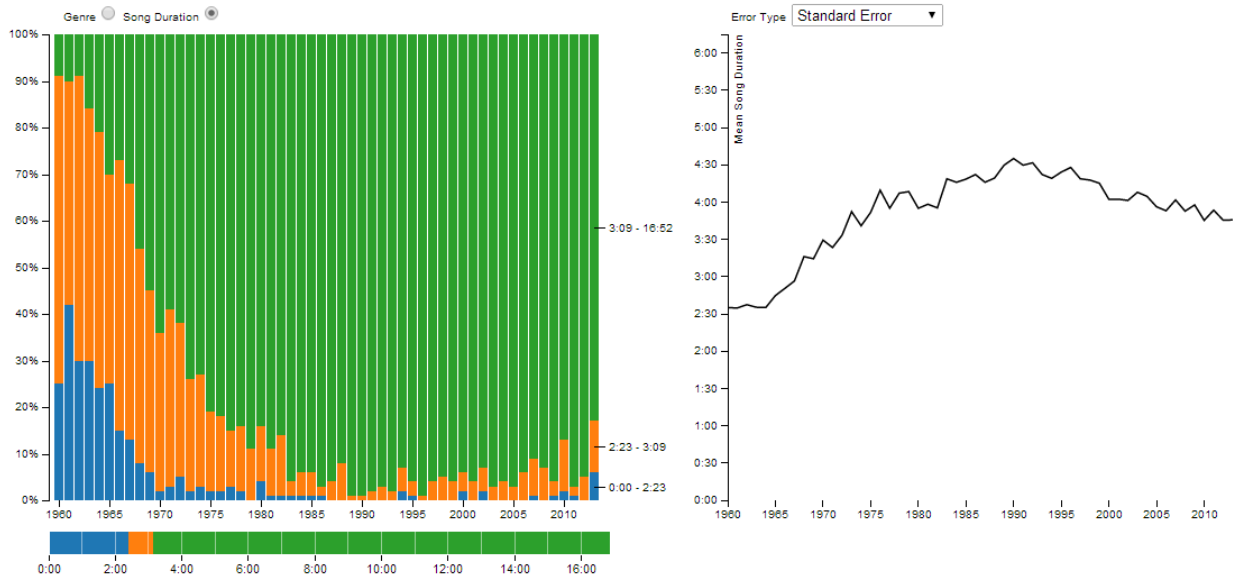

Figure 5: The time intervals for the stacked bar chart can be changed by interacting with the time slider beneath it. This will dynamically update the stacked bar chart.

Screenshot 6:

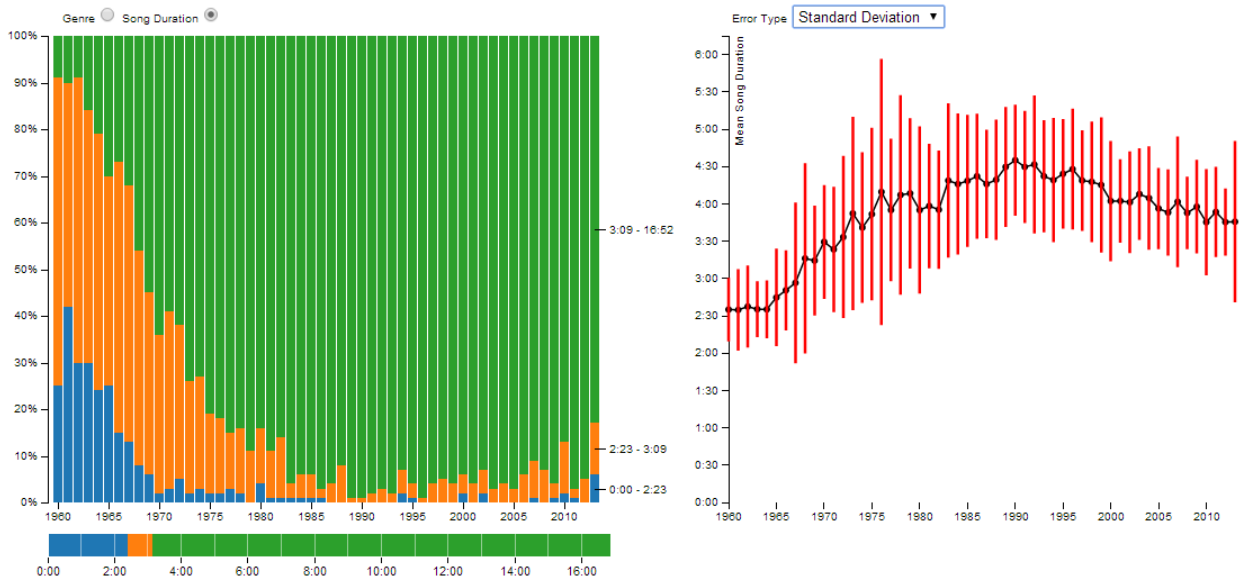

Figure 6: The error type can be changed using the drop-down menu above the line chart. The options are one standard error of the mean, standard deviation, and 95% confidence interval. Hovering over the line chart will display the error bars for all years simultaneously.

Evaluation of D3

Our project primarily used D3.js. **Data-Driven Documents or D3** is a JavaScript library that uses digital data to create interactive data visualizations that run in web browsers. D3 uses Scalable Vector Graphics (SVG), JavaScript, HTML and Cascading Style Sheets (CSS). D3 is extremely useful in that it allows users complete control over the final visualization produced. D3 in particular has some great functionality like smoother transitions and built in colors that the user can use freely instead of coding it themselves (like in the case of C or JavaScript).

One thing that's great about D3 is how widely used it is. We were able to find tons of code skeletons related to the types of things we wanted to create at (http://bl.ocks.org/mbostock) and it was often easy to find relevant tutorials and debugging information as needed to help as when we encountered road blocks in our coding efforts. For instance, when wondering how to add error bars to the line chart, Dakota found a very specific Stack Overflow question that described how to do just that (http://stackoverflow.com/questions/22620064/adding-y-error-bars-to-grouped-bar-chart-with-d3-js). On the other hand, the lack of error bars in the D3 library is concerning. We had to create our own function which drew them. Error bars are widely used in statistical inference, so it is a shame that D3 does not provide a easy way to create them.

Another useful feature of D3 is how it deals with data. Using the enter() and exit() functions, D3 allows some functions to be executed only when the data nodes are created and removed vs. some that are executed each time a data node is updated. This makes it so that the entire visualization doesn't have to be recreated each time the data changes and allows for easy, smooth, visual transitions.

One difficulty that we found in using D3 was creating the dynamic query. D3 has a feature for brushing, which we used to create our slider for the song durations, but it wasn't exactly what we were looking for. We had to use some additional JavaScript and CSS to make it look and work in such a way that sliding the dividers between the different colors changed the song length ranges, which is what we were going for.

Since software like Tableau and Spotfire are commercial in nature, their scope is limited. With D3, the user can customize visualizations to suit his needs. D3 offers a more powerful outlet for programmers looking to develop something better. The use of interaction techniques would also be limited in commercial software. For example, it would not have been possible to connect the bar chart and line graph to the stacked bar chart on a commercial visualization tool such as

Tableau. Also, the details on demand interaction technique achieved by using tool-tips which was possible using D3 would not be possible on Tableau.

Individual Roles and Lessons learned about Teamwork

---

1) Eric Morphis: He was responsible for programming the normalized stacked graph. He also implemented the multiple views interaction technique of the stacked graph and the filter for "genre" mode.
2) Sneha Ganesh: She was responsible for programming the sliders on the time chart which is the implementation of the dynamic query interaction method.
3) Nikhil Nandish: He was responsible for programming the bar chart which will be connected to the normalized bar chart such that the bar chart only focuses on the genre that has been selected.
4) Dakota Lindsey: He was responsible for the line graph in "song length" mode and linking it to the stacked bar chart.

Overall, we believe that our team functioned efficiently and we, the team members, were in sync with each other throughout the project. Over the course of this project, we learned that communication was very important. We were always in contact, usually through email and we made sure to meet at least once a week in class to discuss our progress. In addition, we had scheduled meetings before each milestone was due. These meetings were always decided approximately a week in advance to allow for any last minute changes in schedule. Meetings followed an open discussion structure and all final decisions were based on a consensus. Time management and organizational structure was also central to our group's workings. We always made sure to meet our deadlines and stick to the parts of the project we were assigned. The decentralization of the work load proved to be the right decision. It made each member feel a sense of responsibility toward the project and consequently we each finished our portions of the project at the decided times. Essentially we learned that for a team to succeed we would have to have open discussion, frequent communication and a good organizational structure.