# Exploring the Trends of Top Spotify Songs

CS573 Final Project Process Book

---

Evelyn Tran

# Table of Contents

# Overview and Motivation

This project aims to display music trends and patterns by showing visualizations of Spotify's top songs from 2010 to 2019. Each song can be broken down to different characteristics such as genre, tempo, energy, etc. Based on these traits, we can discover and explore any changes in trends or if society's music taste has been the same.

The question I'm trying to answer with this project is how have people's music tastes changed or stayed the same over the 2010 - 2019?

## Related Work

Here are some publications that I referenced when developing this project:

- Havre developed ThemeRiver, which is a prototype system that displays thematic changes over time [1].
- Byron goes through the design process in creating a complex layered graph, including techniques, design decisions, and how it compares to traditional stacked graphs [2].
- Aigner provides appropriate methods and frameworks when analyzing time-oriented data for visual representation [3].
- Gulik develops a new way to visualize and navigate music based on other contextual information. The artist map UI proposes an interesting way to visualize songs based on their attributes [5].
- Nguyen utilizes dendrograms and maps based on a similarity/distance matrix to find correlations between classical European music composers [4].

# Overall Process

## Meeting with Professor

I had a very rough start with the final project due to other classes and the workload I had for those. Fortunately, after talking with the professor, I was able to submit the final project a bit later.

At this point of time, I knew I would have to get rid of extra elements of my original proposal. This included the ability to rearrange the layers, which I knew would take me some time to learn and figure out. In addition to that, the aesthetics of the website would not be a main priority for me.

## Stream Chart

### Data

For the stream chart, I had to filter out each year and get the number of songs in each genre. To do this, I used the crossfilter library to filter each year and grouped up each genre by creating a temporary crossfilter for the filtered year.

From there, the organized data was held in the array *orgData.* Each object represents a year and contains the number of songs in each genre.

To get to this point, I originally had *orgData* as a hashmap, where the key was the year and the value would be an array of objects with a genre and a number. It wasn't until I was trying to input the data that I realized that I needed to mark genres with 0 songs instead of not adding them into the year for d3.area() to draw the particular path. I continued to play around with arrays and objects, as well as seeing if filtering by genre, then year would make a difference. In the end, I decided on filtering by year and then genre based on the data of a stream chart I was referencing.

With *orgData*, I could finally use it as an input for d3.stack() for the stream chart areas to be drawn.

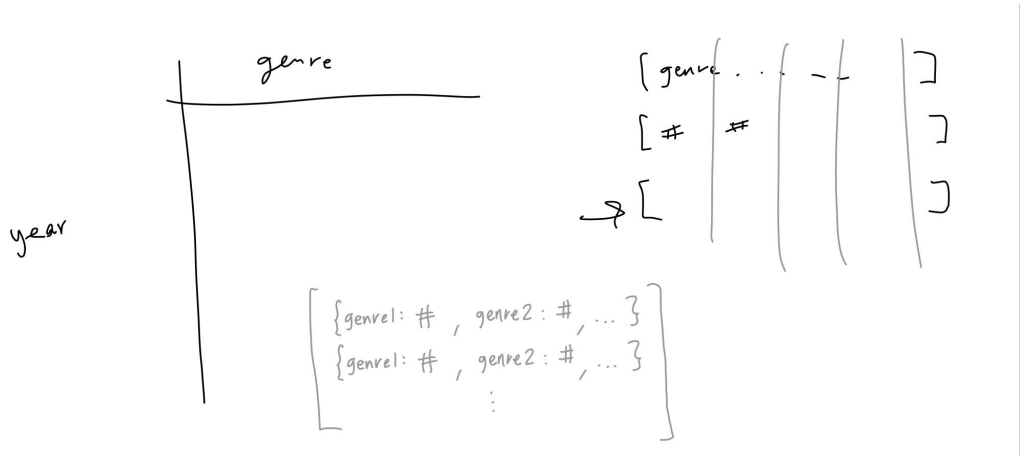


*Figure 1: Trying to determine the best way to organize my data*


```
▼(10) [{…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}, {…}]
  ▶0: {year: 2010, neo mellow: 1, detroit hip hop: 1, dance pop: 31, pop: 3, …}
  ▶1: {year: 2011, neo mellow: 0, detroit hip hop: 0, dance pop: 38, pop: 4, …}
  ▶2: {year: 2012, neo mellow: 0, detroit hip hop: 0, dance pop: 15, pop: 7, …}
  ▶3: {year: 2013, neo mellow: 0, detroit hip hop: 0, dance pop: 42, pop: 5, …}
  ▶4: {year: 2014, neo mellow: 6, detroit hip hop: 0, dance pop: 27, pop: 9, …}
  ▶5: {year: 2015, neo mellow: 0, detroit hip hop: 0, dance pop: 52, pop: 8, …}
  ▶6: {year: 2016, neo mellow: 2, detroit hip hop: 0, dance pop: 46, pop: 1, …}
  ▶7: {year: 2017, neo mellow: 0, detroit hip hop: 0, dance pop: 31, pop: 5, …}
  ▶8: {year: 2018, neo mellow: 0, detroit hip hop: 1, dance pop: 38, pop: 9, …}
  ▶9: {year: 2019, neo mellow: 0, detroit hip hop: 0, dance pop: 7, pop: 9, …}
    length: 10
  ▶__proto__: Array(0)
```


*Figure 2: Final array orgData*

## Aesthetics and Other Features

While I was initially inspired by stream charts such as themeRiver [1], because there were so many genres, I decided against smoothing out the lines since it would make the stream chart even harder to read. I was not expecting as many genres, so adding the hovering for different areas helped a lot with the charts readability. In addition to hovering over the different areas, having a tooltip say what genre was being hovered over was very helpful.

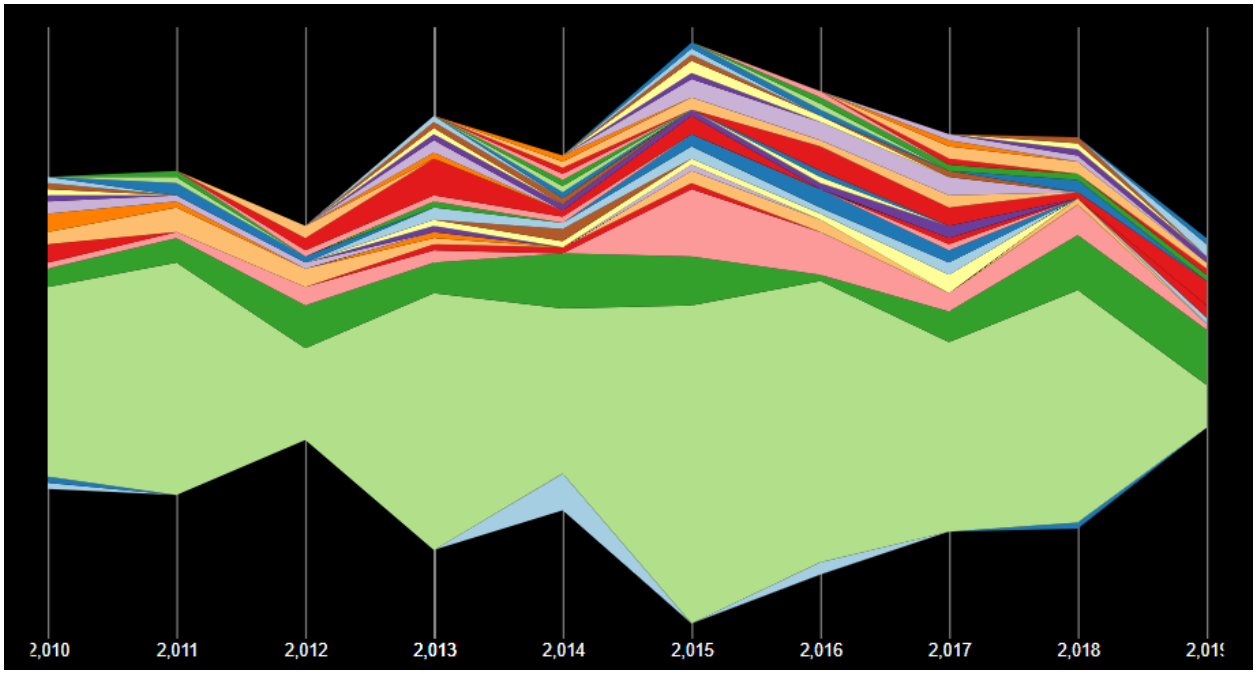


*Figure 3: Final stream chart*

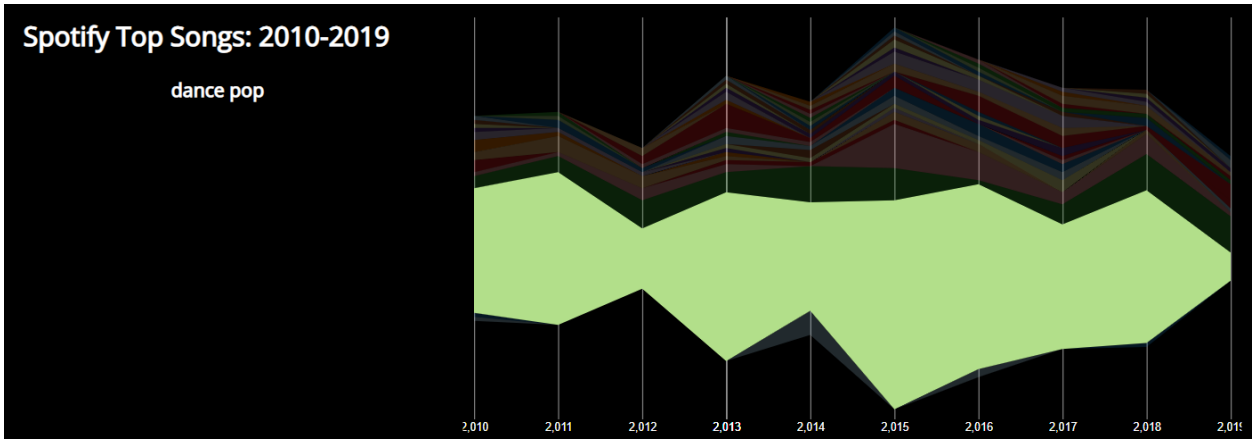


*Figure 4: Hovering over an area in the stream chart*

# Scatter Plots

## Data

For the scatter plots, I chose not to add the popularity data that was included in the dataset. Given what I originally intended for this project, the popularity of the song didn't really speak to the song's traits and the trends in music.

## Implementation

To implement the scatter plots, instead of individually creating each one, I developed a function to create a scatter plot based on the song attribute that is inputted. To link the circles to one another, I assigned a class to each circle that included its index in the data.

## Tooltip

To start, when hovering the circles, I had the tooltip display the song, the artist and the genre. This just used the one line that was being used by the stream chart. I figured it would be more useful to have all the exact numbers for each of the attributes of the song. The problem was that there was a lot of information and I couldn't create a new line between each trait for readability. To get around this, I created a <p> for each line that I needed and an array of strings that contained what each <p> would contain.

After adding the legend button functionality, the hovering for the stream chart and scatterplots were disabled since it would highlight other areas and remove the filtering. I thought it would be better to keep the hovering feature for the scatterplots, but only for the filtered circles. To do this, I needed to check to see if a button was clicked and only update the tooltip. If a button wasn't clicked, then I would update the tooltip and highlight the circle. I decided to keep a global variable to keep track of the button press.

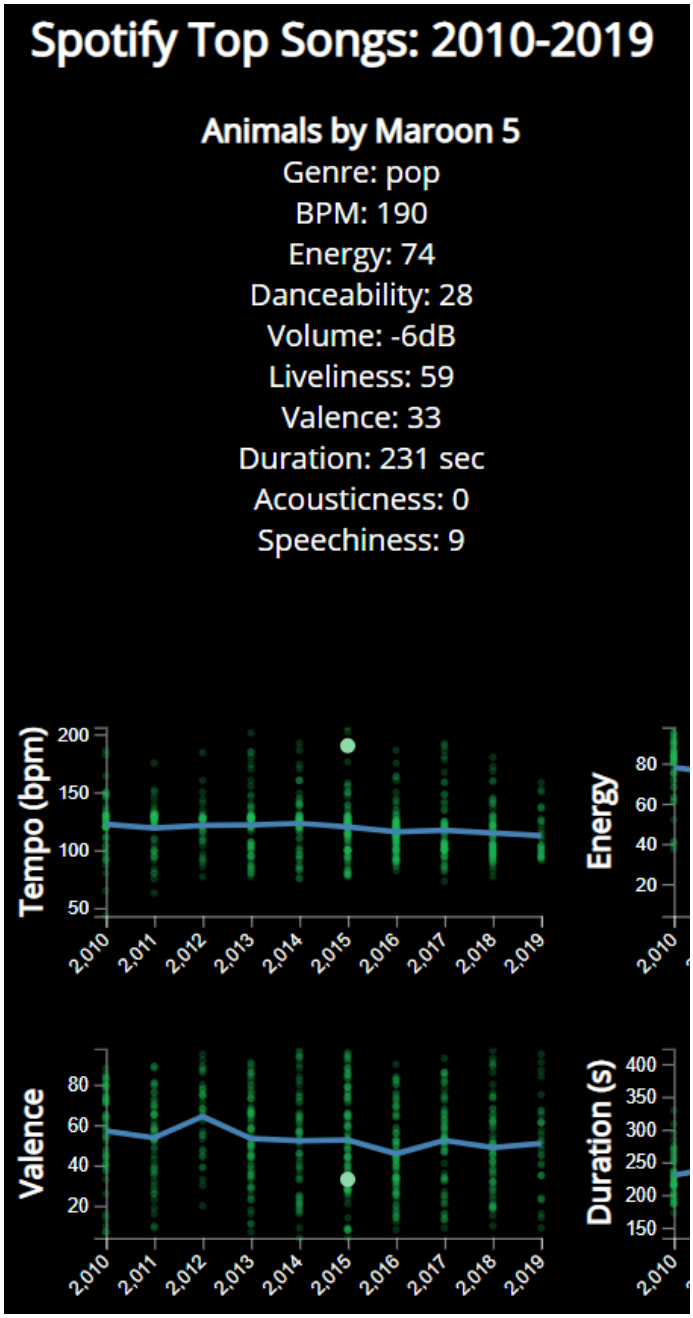


*Figure 5: Tooltip when hovering over a circle in the scatter plot. The song is highlighted in the other scatterplots as well.*

## Trendline and Average Data

To add the trendline, I had to iterate through the data again and determine the average of each characteristic by each year. Each year is stored as an object with the year and the average, and all the years are stored in an array. A path is drawn based on the resulting array.

```
▼ Array(10)
  ▶ 0: {year: 2010, avg: 122.05882352941177}
  ▶ 1: {year: 2011, avg: 119.0754716981132}
  ▶ 2: {year: 2012, avg: 121.08571428571429}
  ▶ 3: {year: 2013, avg: 121.67605633802818}
  ▶ 4: {year: 2014, avg: 123}
  ▶ 5: {year: 2015, avg: 119.76842105263158}
  ▶ 6: {year: 2016, avg: 115.77215189873418}
  ▶ 7: {year: 2017, avg: 116.8}
  ▶ 8: {year: 2018, avg: 114.59375}
  ▶ 9: {year: 2019, avg: 112.45161290322581}
    length: 10
  ▶ __proto__: Array(0)
```

*Figure 6: Array with the average tempos; later updated to round the averages to the nearest hundredth*

Originally I was going to just leave it as a line. However, because everything else had interactive elements, I decided to add in a tooltip for it as well. Instead of having the information where the other song info was located, I opted to have the tooltip to hover by mouse because the scatterplots were already so small.

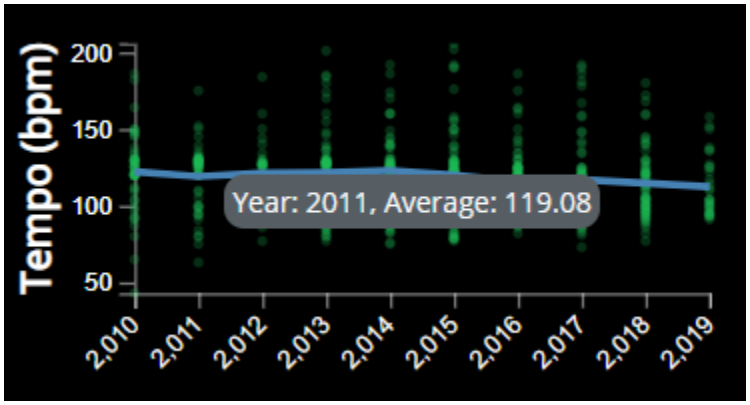


*Figure 7: Tooltip when hovering on the trendline*

## Legend

To create the buttons, I decided to use the array of genres I made for the coloring scheme and iterate through it to create each button. Originally, I only intended to use the buttons to highlight areas on the stream chart. However, because I implemented the hovering aspect of the stream chart, I decided to highlight the songs of that particular genre in the scatter plot as well. To link the circles and the areas, I set a “genre” attribute to each, and only selected DOM elements with the genre that the legend button called for.

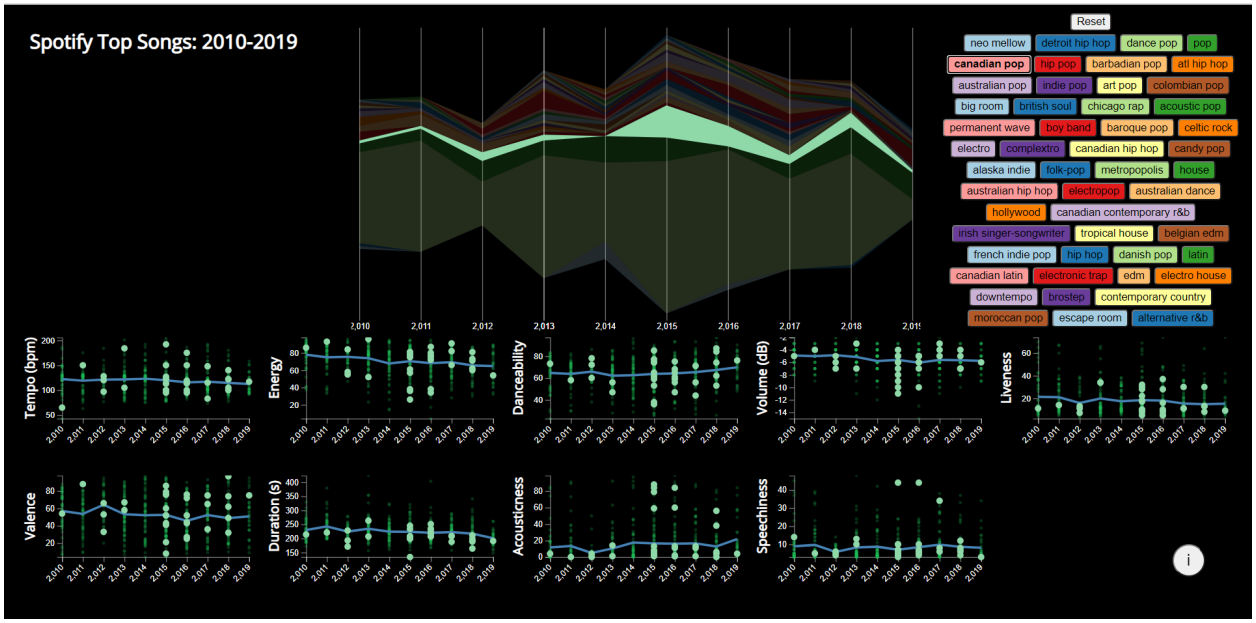


*Figure 8: Filtering “canadian pop” by clicking on legend button*

# Web Page Layout/Other

## Layout

Because the number of genres was greater than I anticipated and I added a tooltip, I needed to rearrange the webpage to look better on the screen. Here were some of the mockups I designed:

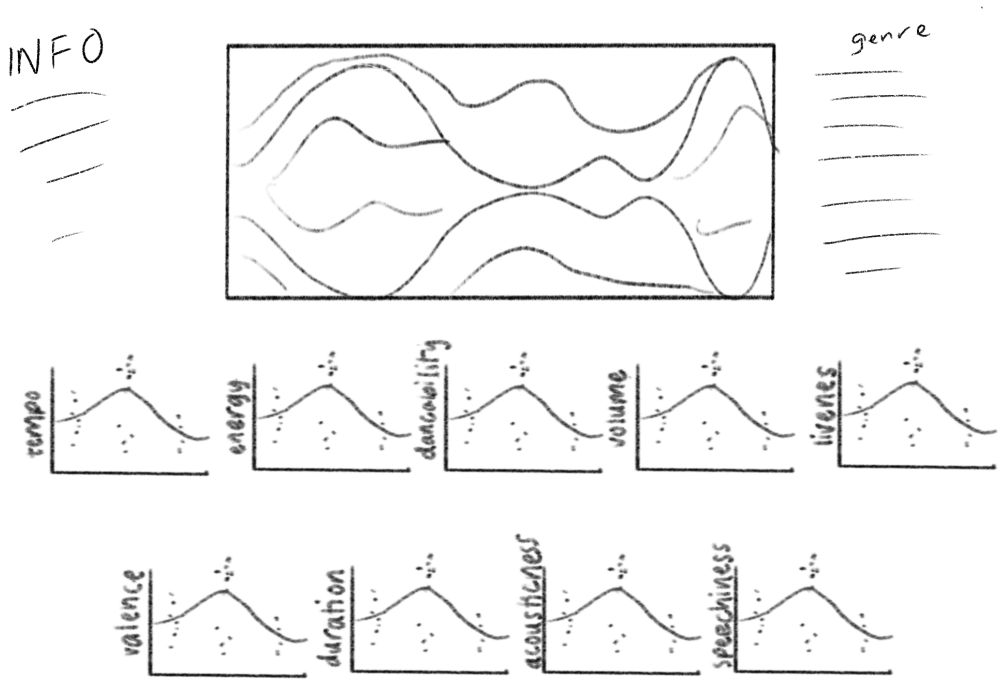


*Figure 9: Mockup 1*

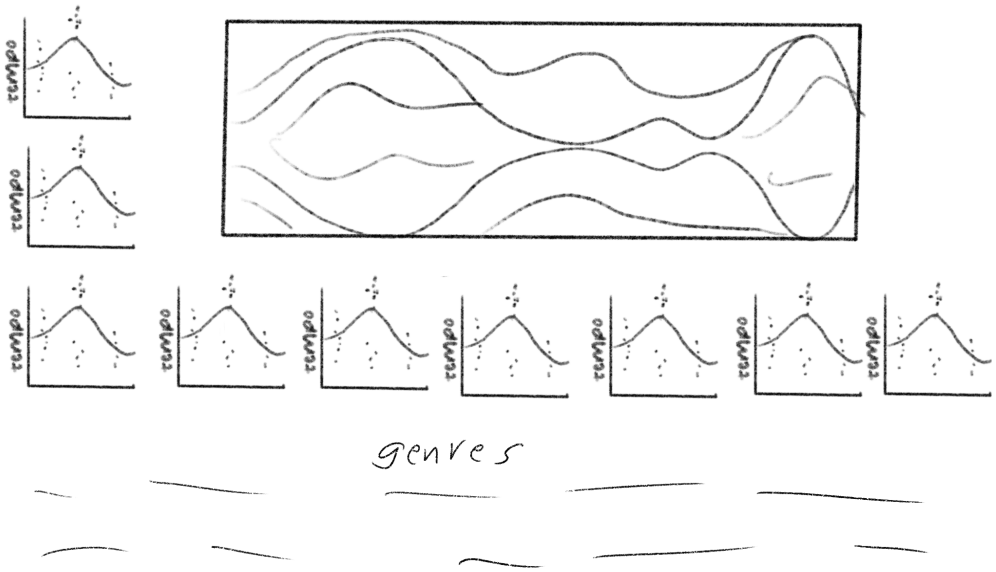

*Figure 10: Mockup 2*

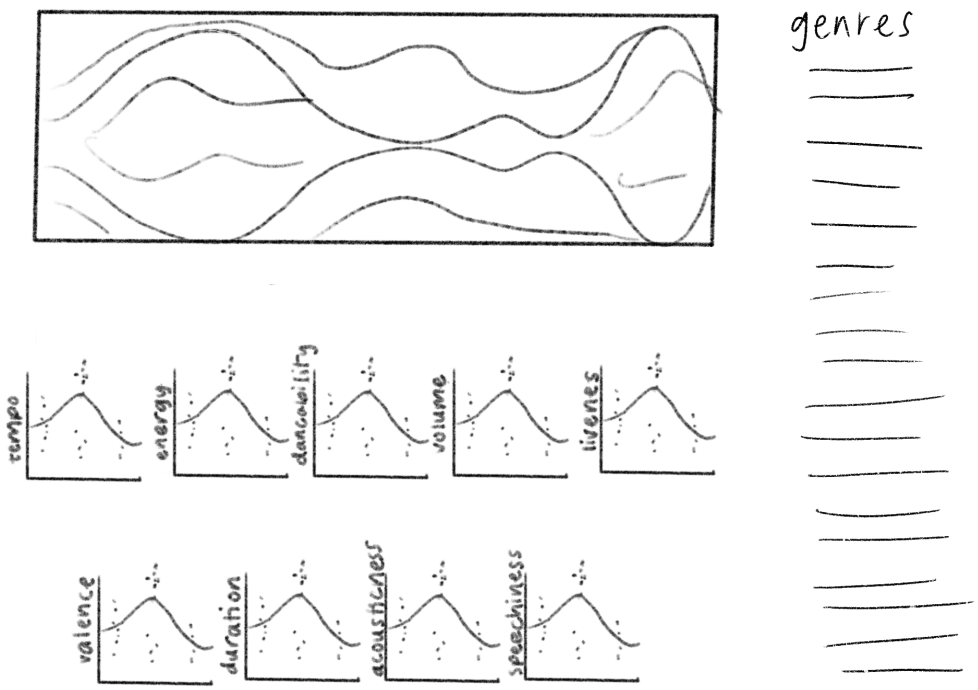


*Figure 11: Mockup 3*

I initially went with Mockup 3, with the tooltip on the left of the stream chart, similar to Mockup 1. However the list of genres ran long and required the user to scroll down for some of the buttons. I then opted for a layout similar to Mockup 1. While Mockup 2 seemed really cool, the layout of the scatterplots would require users to look over a larger area rather than just having all the scatterplots in one area.

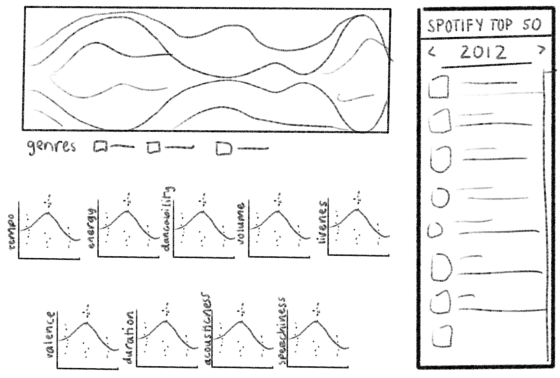


*Figure 12: Original Mockup*

## Color Scheme

There were 50 genres, so having a unique for each area in the stream chart may not be as helpful. In addition to that, in the interest of time, I decided to use a color scheme provided in the d3.js library. I knew that I had to sacrifice some of the aesthetics aspects from the beginning. The colors in the stream chart are mostly used just to differentiate the areas rather than for identifying. The hovering aspect of the stream chart and the filtering through the legend already covered the identification of each area. I tried out the different color schemes in the d3.js library and opted for the “Paired” color scheme.


*Figure 13: d3.js color schemes*

# Evaluation

Here are some of my findings from the data I presented:

- Through the visualization we can see a large majority of Spotify's Top Songs from 2010 - 2019 have been composed of the dance pop genre. Even so, generally most of the top songs of Spotify were some sort of "pop" genre or "hip pop".
- We can also see that the energy of these top songs are slowly declining throughout the years. Of course, while there is a large range, the averages are decreasing. This could also tie into the declining tempos. While it isn't as apparent as the energy trendline, by using the tooltip, we can see that the BPM throughout the years increases slightly at 2015 and then proceeds to decrease.This is interesting because the danceability of songs have increased through the years, as the tempo and energy have decreased.
- Another more immediate trend we can see is that the acousticness of songs have increased, as well as a decrease in song duration.

I do believe that the visualizations I provided do bring some insight into how music has changed or stayed the same throughout the years. To further improve this website, here are some things that could be implemented:

- Open individual scatter plots and see them up close
- Group the genres more broadly
- Find a better color scheme for the stream chart
- Filter out certain timeframe
- Search up songs based on their attributes
- Song list for specific genres or for each year

# References

[1] S. Havre, B. Hetzler and L. Nowell, "ThemeRiver: visualizing theme changes over time," *IEEE Symposium on Information Visualization 2000. INFOVIS 2000. Proceedings*, Salt Lake City, UT, USA, 2000, pp. 115-123, doi: 10.1109/INFVIS.2000.885098.

[2] L. Byron and M. Wattenberg, "Stacked Graphs – Geometry & Aesthetics," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 6, pp. 1245-1252, Nov.-Dec. 2008, doi: 10.1109/TVCG.2008.166.

[3] W. Aigner, S. Miksch, W. Müller, H. Schumann and C. Tominski, "Visual Methods for Analyzing Time-Oriented Data," in *IEEE Transactions on Visualization and Computer Graphics*, vol. 14, no. 1, pp. 47-60, Jan.-Feb. 2008, doi: 10.1109/TVCG.2007.70415.

[4] Georges, P., Nguyen, N. Visualizing music similarity: clustering and mapping 500 classical music composers. *Scientometrics* 120, 975–1003 (2019). https://doi.org/10.1007/s11192-019-03166-0

[5] Gulik, van, R.G.C. ; Vignoli, F. ; Wetering, van de, H.M.M. Mapping music in the palm of your hand, explore and discover your collection. *Proceedings 5th International Conference on Music Information Retrieval* (ISMIR 2004, Barcelona, Spain, October 10-14, 2004). 2004.