Visualizations

Novel Methods for Visualizing Fragment Recruitment Plots

Nadeem Husain[1,*]

[1]Department of Bioinformatics, AAP John Hopkins University

Received on 12/19/2018; revised on 12/19/2018; accepted on 12/19/2018

ABSTRACT

Motivation: The importance of sound data visualizations in the realm of biology requires a scientist to move with the times and adopt newer methods to show plots. Fragment Recruitment Plots (FRP) were introduced at the beginning of this course and for our midterm project, we were tasked to create some from metagenomic samples. After searching for a nicer version to make FRPs, I came across some novel ways to create these plots.

1 INTRODUCTION

Data visualization is the presentation of data in a pictorial or graphical format. It enables decision-makers to see analytics presented visually, so they can grasp difficult concepts or identify new patterns. With interactive visualization, we can take the concept a step further by using technology to drill down into charts and graphs for more detail, interactively changing what data you see and how it's processed.

The concept of using pictures to understand data has been around for centuries, from maps and graphs in the 17th century to the invention of the pie chart in the early 1800s. Several decades later, one of the most cited examples of statistical graphics occurred when Charles Minard mapped Napoleon's invasion of Russia. The map depicted the size of the army as well as the path of Napoleon's retreat from Moscow – and tied that information to temperature and timescales for a more in-depth understanding of the event. Technology, however, allowed rapid advancements in the area of data visualization. Computers made it possible to process large amounts of data at lightning-fast speeds. Today, data visualization has become a rapidly evolving blend of science and art that is certain to change the corporate landscape over the next few years.[1]

As humans, our brains process information, using charts or graphs to visualize large amounts of complex data is easier than poring over spreadsheets or reports. Data visualization is a quick, easy way to convey concepts in a universal manner – and one can experiment with different scenarios by making slight adjustments.

Before implementing new technology, there are some steps one should take. Not only do we need to have a solid grasp of the data, but also need to understand the goals, needs, and audience. To being our visualization journey we first:

- Understand the data one is trying to visualize, including its size and cardinality (the uniqueness of data values in a column).
- Determine what we're trying to visualize and what kind of information you want to communicate.
- Know our audience and understand how it processes visual information.
- Use a visual that conveys the information in the best and simplest form for the audience.

Once answered, questions about the type of data we have and the audience who'll be consuming the information, we need to prepare for the amount of data you'll be working with. The -Omics bring new challenges to visualization because of large volumes, different varieties and different. As a scientist, we need to remember data is often generated faster that it can be managed and analyzed

2 METHODS

2.1 Data Collection

We obtained our data from our midterm project focused on 6 of the 15 samples that were given. We did this due to the large file size, which will be discussed later. Bacterium samples used include:

(1) Belicobacter Pylori
(2) Bacteroides Vulgatus
(3) Acinetobacter Baumannii
(4) Listeria Monocytogenes
(5) Rhodobacter Sphaeroides
(6) Streptococcus Pneumoniae

The aforementioned bacterium's FASTA files were all obtained from NCBI's database to ensure accuracy.

2.2 Conversion of Data

Fragment recruitment, a process of aligning sequencing reads to reference genomes, is a crucial step in metagenomic data analysis. FR-HIT first constructs a k-mer hash table for the reference genome sequences. Then for each query, it performs seeding, filtering, and banded alignment to identify the alignments to reference sequences that meet user-defined cutoffs. [2] As such, we implemented the use of FR-HIT, for quick fragment recruitment algorithm and outputs a SOP file. According to our tests, by using FR-HIT ability for multithreaded processes, runtimes were decreased roughly 4x. The generated .sop file was handled much like a .csv for further testing

2.3 State of Visualization Tools

When picking a programming language to work on our plots, we narrowed it down to R, Python, Java, and Javascript. Unfortunately, we didn't have too much practice with either Java and Javascript. Along those same thoughts, like how R interfaced with the data and how it may not a scale as well as other programs. Thus, we opted for python and its libraries.

The current landscape for data visualization packages in python is very fragmented and have niche areas. New packages are on the rise and some old ones continue to operate as intended. One of the older pythonic visualization libraries is *matplotlib* and is the library we used for the midterm. Unfortunately, with our last endeavor not going as planned, we decided to walk away from it. We turned our attention to three other libraries, which showed the potential to do the visualization we consider:

(1) Bokeh
(2) D3.js
(3) plot.ly

In the end, we decided to use investigate Bokeh and plot.ly, for their documentation and ease implementation with Jupyter Notebook and Google's Collaboratory. An honorable mention needs to be made to the developers of *pygal*. Pygal allows usesr to make beautiful graphical representations in mere minutes, but in the end, we believe that the documentation of the aforementioned outweighed everything.

2.4 Bokeh vs Plot.ly

Like ggplot, Bokeh is based on The Grammar of Graphics, but unlike ggplot, it's native to Python, not ported over from R. Its strength lies in the ability to create interactive, web-ready plots, which can be easily outputted as JSON objects, HTML documents, or interactive web applications. Bokeh also supports streaming and real-time data.

Plot.ly is differentiated by being an online tool for doing analytics and visualization. It has a robust API and includes one for python. Because of this API, we recommend creating an account and inputting your own username/API-Key to make plots work seamlessly. Also much like Bokeh, Plotly integrates seamlessly with pandas, which we used heavily for our analysis. Upon further investigation of plot.ly, we came around the idea of using their dashboard package, Dash, to visualize our plots and took inspiration from *this project*.

2.5 Hexagonal Binning

In our quest to find the best graphical representation, we stumbled across hexagonal bin plots. After much thought and investigation, we believe it may be a more efficient way to represent thousands of points and for our -Omic data it may be perfect. Hexagonal Binning is another way to manage the problem of having too many points that start to overlap. Hexagonal binning plots density, rather than points. Points are binned into gridded hexagons and distribution (the number of points per hexagon) is displayed using either the color or the area of the hexagons. There are many reasons for using hexagons instead of squares for binning a 2D surface as a plane. The most evident is that hexagons are more similar to a circle than square. This translates in more efficient data aggregation around the bin center.

3 Results

After cleaning the data of all points that would not be needed, we started our journey into creating the plots. First, we tackled the aforementioned Hexagonal Plots, using *Bokeh* and python. Figure 1 shows our output of genomic data that was discussed in previous sections. We can see that this is very different than other plots within the -Omics realm, but believe that it does a good job representing more data than a simple FRP. We plotted the read lengths, given in nucleotide, against the percent identity for all six samples. As we can see the Hexagonal Plot gave us a similar output but also gave us the concentration of where the most overlapping was found (yellow). The darker almost purple, hexagons are where the least concentration of overlapping occurred. Furthermore in production, once the user hovers over a hexagon, will give the end-user the count of genomic overlapping in a certain area.

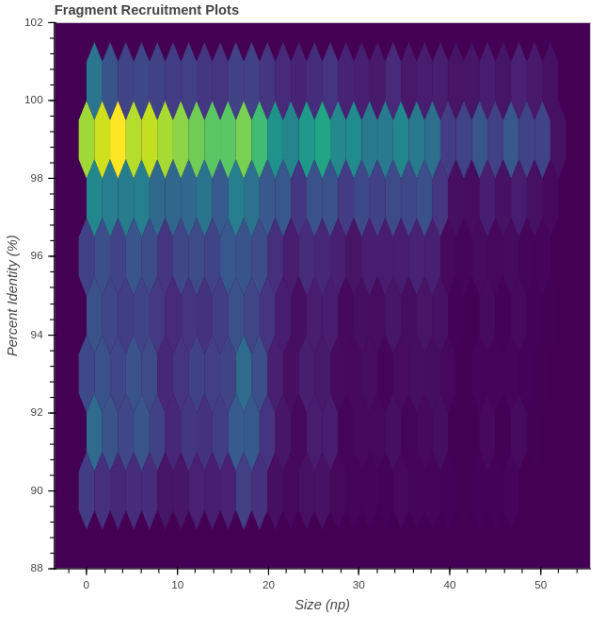

Figure 1: FRP using Hexagonal Binning

We expanded this codebase for plot.ly and wanted to create a FRP using a stacked scatterplot. As we did with the Hexagonal plot, data was cleaned of all erroneous points to keep the input file as small as possible. During the implementation of the scatter plot, we wanted to create a plot which would allow the end user to toggle off genomes that were not needed. Figure 2 below shows our completed FRP.

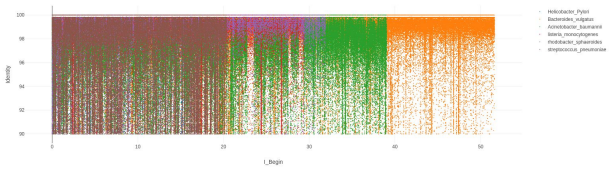

Figure 2 : FRP Scatterplot using Plot.ly API

Comparing the two plots, we can see that both are telling the same story, but in different ways. Our hexagonal plot is able to show the sheer concentration of nucleotides found at a particular point. Conversely, our stacked scatterplot shows the same, but due to it being stacked we cannot see the concentration until we look at each sample by itself.

4 DISCUSSION

Many problems occurred on way and we will try to explain them in the following sections.

4.1 Issues with Plot.ly Dash

As discussed above we wanted to use Plot.ly Dash to create plots, which would be web-based and easy to toggle on/off. Unfortunately, after much tinkering with the codebase, we found that it was rather difficult, due to our lack of understanding of Javascript syntax. While extremely powerful, with a little better understanding of the documentation, we believe it would be possible to create a dashboard for each genome that we covered.

4.2 Issues with Package install

As with starting any project package control, inevitably, became our Achilles heel. This package control was especially difficult with Bokeh, as none of our code-based worked. We believe that this could be lack of dependencies downloaded or the mixture of both pip and pip3 mixups. In future implementations, we believe it maybe wiser to use a Docker container to remedy this issue and also allow for future reproducibility.

4.3 Issues with Colaboratory

Colaboratory is a strong app that mimics Juypter notebook. Colaboratory allows users to rapid prototype projects, while also allowing for the installs of packages. Furthermore, Colaboratory allows for further GPU and multiple integrations. Much like the aforementioned package install problems, we found that plotting inline created some issues. After some searches on the internet, we were able to fully visualize our data inline, but this took precious time away from the creation of the codebase. We would love to stick with using Colaboratory for rapid prototyping, but believe that if we fix the package control above going with live code is preferred.

4.4 Potential issues with scalability of Data

After looking at our data, we quickly understood that dynamic implementation would be difficult. For reference when we combined all of our genomics, the file was well over 65 MB. After cleaning the data, we brought it down to 40+MB. The sheer size of the data in our sample creates a code base that is quite slow to begin but is worth it in the end. In future implementations, we believe it would be better to use a LAMP stack. This would most probably decrease the time to render our plots but also would allow for our data to be in a database.

5 CONCLUSION

With this being our first foray into programming such a large project we came out with many lessons that were learned, including features that we include and tweak. Some of the points that we taught ourselves are listed below

- Genomic data files are large and it's probably a better idea to use a database, which could feed dynamic graphs
- SysAdmining your home machine is fun
- Each program has their positives and negatives, but we chose two that made sense to us. By no means are we stating that these are better, but it was better for us. If we were a Javascript pro, we would scrap everything and use D3.js
- The codebase is finicky and sometimes it's best to just keep trying
- There were many times when we wanted to stop, but through sheer determination and positive affirmations from friends/friends go a long way
- Copious amounts of coffee and tea can make the world much better.

Sanity was brought to you in part by: Branch Street Coffee, Suzie's Dogs and Drafts, National Football Association, NCAA Division I Fooball, Rhinegeist Brewery (Truth IPA), and Poland Local Schools.

REFERENCES

"The History of the History of Data Visualization." IDashboards, www.idashboards.com/blog/2017/11/29/the-history-of-the-history-of-data-visualization/.

Niu, Beifang, et al. "FR-HIT, a Very Fast Program to Recruit Metagenomic Reads to Homologous Reference Genomes." Bioinformatics, vol. 27, no. 12, 2011, pp. 1704–1705., doi:10.1093/bioinformatics/btr252.