1 Discussion

This work contributes a mathematical description of the mapping A from data to visual representation. Combining Butler's proposal of a fiber bundle model of visualization data with Spivak's formalism of schema lets this model support a variety of datasets, including discrete relational tables,, multivariate high resolution spatio temporal datasets, and complex networks. Decomposing the artist into encoding ν, assembly Q, and reindexing ξ provides the specifications for producing visualization where the structure and properties match those of the input data. These specifications are that the graphic must have continuity equivalent to the data, and that the visual characteristics of the graphics are equivariant to their corresponding components under monoid actions. This model defines these constraints on the transformation function such that they are not specific to any one type of encoding or visual characteristic. Encoding the graphic space as a fiber bundle provides a structure rich abstraction of the target graphical design in the target display space.

The toy prototype built using this model validates that is usable for a general purpose visualization tool since it can be iteratively integrated into the existing architecture rather than starting from scratch. Factoring out glyph formation into assembly functions allows for much more clarity in how the glyphs differ. This prototype demonstrates that this framework can generate the fundemental marks: point (scatter plot), line (line chart), and area (bar chart). Furthermore, the grouped and stacked bar examples demonstrate that this model supports composition of glyphs into more complex graphics. These composite examples also rely on the fiber bundles section base book keeping to keep track of which components contribute to the attributes of the glyph. Implementing this example using a Pandas dataframe demonstrates the ease of incorporating existing widely used data containers rather than requiring users to conform to one standard.

1.1 Limitations

So far this model has only been worked out for a single data set tied to a primitive mark, but it should be extensible to compositing datasets and complex glyphs. The examples and prototype have so far only been implemented for the static 2D case, but nothing in the math limits to 2D and expansion to the animated case should be possible because the model is formalized in terms of the sheaf. While this model supports equivariance of figurative glyphs generated from parameters of the data[1, 2], it currently does not have a way to evaluate the semantic accuracy of the figurative representation. Effectiveness is out of scope for this model because it is not part of the structure being preserved, but potentially a developer building a domain specific library with this model could implement effectiveness criteria in the artists. Also, even though the model is designed to be backend and format independent, it has only really been tested against PNGs rendered with the AGG backend. It is especially unknown how this framework interfaces with high performance rendering libraries such as openGL[3]. Because this model has been limited to the graphic design space, it does not address the critical task of laying out the graphics in the image

This model and the associated prototype is deeply tied to Matplotlib's existing architecture. While the model is expected to generalize to other libraries, such as those built on Mackinlay's APT framework, this has not been worked through. In particular, Mackinlay's formulation of graphics as a language with semantic and syntax lends itself a declarative interface[4], which Heer and Bostock use to develop a domain specific visualization language that they argue makes it simpler for designers to construct graphics without sacrificing expressivity [5]. Similarly, the model presented in this work formulates visualization as equivariant maps from data space to visual space, and is designed such that developers can build software libraries with data and graphic topologies tuned to specific domains.

1.2 Future Work

While the model and prototype demonstrate that generation of simple marks from the data, there is a lot of work left to develop a model that underpins a minimally viable library. Foremost is implementing a data object that encodes data with a 2D continous topology and an artist that can consume data with a 2D topology to visualize the image[6–8] and also encoding a separate heatmap[9, 10] artist that consumes 1D discrete data. A second important proof of concept artist is a boxplot[11] because it is a graphic that assumes computation on the data side and the glyph is built from semantically defined components and a list of outliers. The model supports simple composition of glyphs by overlaying glyphs at the same position, but more work is needed to define an operator where the fiber bundles have shared $S_2 \hookrightarrow S_1$ such that fibers could be pulled back over the subset. While the model's simple addition supports axes as standalone artists with overlapping visual position encoding, the complex operator would allow for binding together data that needs to be updated together. Additionally, implementing the complex addition operator and explicit graphic to data maps would allow for developing a mathematical formalism and prototype of how interactivity would work in this model. In summary, the proposed scope of work for the dissertation is

- expansion of the mathematical framework to include complex addition
- formalization of definition of equivalance class A'
- implementation of artist with explicit ξ
- specification of interactive visualization
- mathematical formulation of a graphic with axes labeling
- implementation of new prototype artists that do not inherit from Matplotlib artists
- provisional mathematics and implementation of user level composite artists
- proof of concept domain specific user facing library

Other potential tasks for future work is implementing a data object for a non-trivial fiber bundle and exploiting the models section level formalism to build distributed data source models and concurrent artists. This could be pushed further to integrate with topological[12] and functional [13] data analysis methods. Since this model formalizes notions of structure preservation, it can serve as a good base for tools that assess quality metrics[14] or invariance [15] of visualizations with respect to graphical encoding choices. While this paper formulates visualization in terms of monoidal action homomorphisms between fiberbundles, the model lends itself to a categorical formulation[16, 17] that could be further explored.

2 Conclusion

An unoffical philosophy of Matplotlib is to support making whatever kinds of plots a user may want, even if they seem nonsensical to the development team. The topological framework described in this work provides a way to facilitate this graph creation in a rigorous manner; any artist that meets the equivariance criteria described in this work by definition generates a graphic representation that matches the structure of the data being represented. We leave it to domain specialists to define the structure they need to preserve and the maps they want to make, and hopefully make the process easier by untangling these components into seperate constrained maps and providing a fairly general data and display model.