CI Compass
TECH NOTES

ci-compass.org

Visualizing Major Facility data with web-compatible technologies: Data Visualization for Jupyter Notebooks and Web Dashboards

Valerio Pascucci[1], Giorgio Scorzelli[1], Steve Petruzza[1,2]
[1]Center for Extreme Data Management Analysis and Visualization (CEDMAV), University of Utah, UT, USA
[2]Utah State University, UT, USA
Date Published: September 22, 2022
DOI: 10.5281/zenodo.7105790

Data Visualization for Jupyter Notebooks and Web Dashboards

Data visualization is a key component of most National Science Foundation (NSF) Large Facilities' cyberinfrastructure since it can provide powerful ways to explore, understand, and share key scientific insights generated by the mission of the projects involved, and facilitate building and connecting a community of users. In recent years, Jupyter Notebooks and web dashboards have become increasingly popular and have been adopted by many scientists in diverse communities.

Jupyter Notebooks' first and foremost strength derives from their interactivity. Jupyter Notebooks provide a unique ability to experiment with data in real-time and see the code's results for each typed command without delay. This makes them ideal for scientists and researchers whose focus is on data exploration and understanding, and not on code development. With Jupyter notebooks, displaying data does not require developing a complete application. A user can continuously render intermediate results while writing a script without separating the activity of developing a visualization system and using it.

Online Web Dashboards have also gained popularity since they provide a really powerful mechanism to communicate the results of a scientific project to a large community without facing the challenges of distributing a tool that other people need to install. Until recently, dashboards have been hard and expensive to build since they required coding in specialized web languages such as Javascript (e.g. note.js) or Ruby (e.g. Ruby on Rails).

In this brief technical note we list a few options that may allow accomplishing effective visualization for either or both environments and report possible pitfalls. In particular, Table 1 reports the main set of libraries that may accomplish the work effectively in Python. Note that this is an area of very active development and new solutions emerge continuously. This makes it hard to keep a comprehensive list (feedback for additions or removals is highly appreciated) and creates the risk of adopting either a technology that is robust but obsolete or one that may look promising but has not gained enough popularity to expect long-term support.

This project is supported by the National Science Foundation Office of Advanced Cyberinfrastructure in the Directorate for Computer Information Science under Grant #2127548.

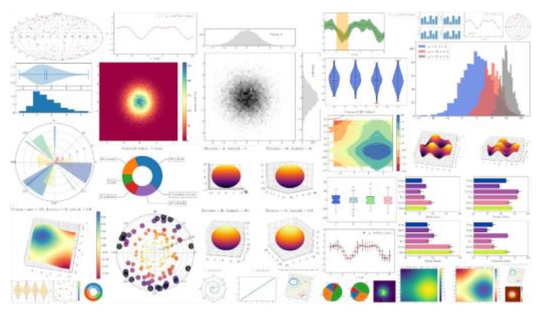
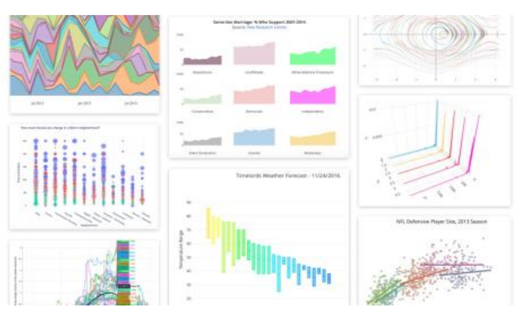
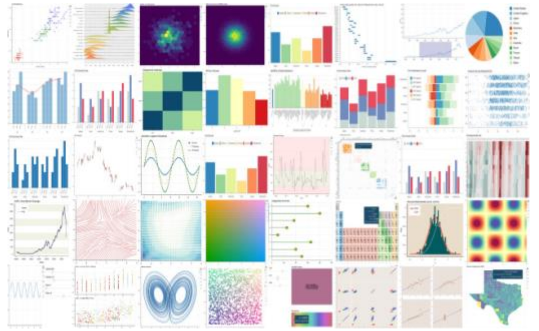
Table 1: List of main Python data visualization libraries

1	matplotlib	Matplotlib	https://matplotlib.org/
2	plotly	Plotly	https://plotly.com/
3	bokeh	Bokeh	https://bokeh.org/
4	HoloViz	HoloViz	https://holoviz.org/
5	seaborn	Seaborn	https://seaborn.pydata.org/
6	Altair	Altair	https://altair-viz.github.io/
7	Streamlit	Streamlit	https://streamlit.io/

While we feel it is important to provide the users with a good list of options, a comprehensive evaluation is beyond the scope. We focus on the three main alternatives provided by Matplotlib, Plotly and Bokeh. Table 2 provides a basic, high-level comparison of the plotting capabilities of these libraries. All three provide a broad set of plotting primitives and are likely to be very effective for most visualization needs, although their difference in design and user interface may make different tasks easier or harder to accomplish and provide output results of varying quality.

Matplotlib is by far the most common open-source Python library for visualizing data. Matplotlib provides a complete set of features; it comes with extensive documentation, and a large set of examples can be found on the web. Matplotlib can provide high-quality output for publishing detailed figures in scientific journals. The downside of Matplotlib is that its older design lacks a number of modern features. Integration with interactive widgets is not always easy, and it definitely lacks good support for web publishing, unless used in combination with other libraries. The more recent library **Seaborn** (number 5 in Table 1) is based on Matplotlib and is designed to provide access to the Matplotlib features through a more modern, higher-level interface. For use only within Python and Jupyter Notebooks, it is certainly a very effective option.

Table 2: Basic comparison of plotting capabilities between three main Python data visualization libraries

Matplotlib

Plotly

Bokeh


Plotly is also an open-source library for data visualization that can be used freely. At its core it is a Javascript library (plotly.js) with bindings in a number of languages including Python, R, Julia, Matlab, etc. It is backed by a private corporation (Plotly, Inc.), which can provide strong support and development of new features upon request. For web development, Plotly can be combined with the Dash tool to provide a very simple deployment of pure python apps. A free version of Dash is available and can be used independently but

This project is supported by the National Science Foundation Office of Advanced Cyberinfrastructure in the Directorate for Computer Information Science under Grant #2127548.

seems to need advanced knowledge of the platform. A subscription model for Dash Enterprise is available and seems to be the best option but it can be expensive, so it is probably recommended only for private corporations or projects with a significant budget allocation for this purpose. Simplicity of use is the first design principle and scaling a dashboard for large data applications may be problematic.

Bokeh is an open-source library for data visualization that is supported by a community of active users. Visualizations generated in Bokeh can be very flexible and are well connected to interactive widgets. A major strength of Bokeh is its server capability that allows taking a Jupyter Notebook and turning it into a web dashboard without any addition of code. Running a Bokeh server from the command line is trivial but some investment in time is needed to set up a server on the web. Still, the advantages are tremendous in simplicity of shareability of identical visualizations and interactive interfaces either as notebooks or as portals. Bokeh is all Python based but also allows light Javascript callbacks if one wants to avoid setting up a full server (not recommended). The success of Bokeh is demonstrated by a variety of third-party libraries that use Bokeh as a foundation to develop high-level user interfaces. The documentation is very good for the most common features, but more advanced use cases may require more work to explore the complete set of options provided by the library. While not as established as Matplotlib, at this stage, Bokeh seems to have gathered a large enough user community that it is not at any risk of disappearing, and, in fact, it seems the primary choice as a truly free option for web development of user interfaces purely in Python.

Acknowledgments

We acknowledge the valuable input from all the staff members and leadership of the National Center for Atmospheric Research (NCAR) and National Ecological Observatory Network (NEON), which created the impetus for this evaluation of visualization libraries. We want to specifically thank Gordon Bonan, David Durden, Dawn Lenz, Danica Lombardozzi, Mike SanClements, Negin Sobhani, and Will Wieder for the valuable input and stimulation discussions.

This material is based in part on work supported by the National Science Foundation under Grant Numbers 2127548, 1842042 and 1851037. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

For more information, please visit ci-compass.org or email us at contact@ci-compass.org.