

Altair: Interactive Statistical Visualizations for Python

Jacob VanderPlas¹, Brian E. Granger², Jeffrey Heer¹, Dominik Moritz¹, Kanit Wongsuphasawat¹, Eitan Lees³, Ilya Timofeev⁴, Ben Welsh⁵, and Scott Sievert⁶

1 University of Washington 2 California Polytechnic State University, San Luis Obispo 3 Florida State University 4 TTS Consulting 5 Los Angeles Times Data Desk 6 University of Wisconsin–Madison

DOI: [10.21105/joss.01057](https://doi.org/10.21105/joss.01057)

Software

- [Review](#) ↗
- [Repository](#) ↗
- [Archive](#) ↗

Submitted: 15 October 2018

Published: 08 December 2018

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC-BY](#)).

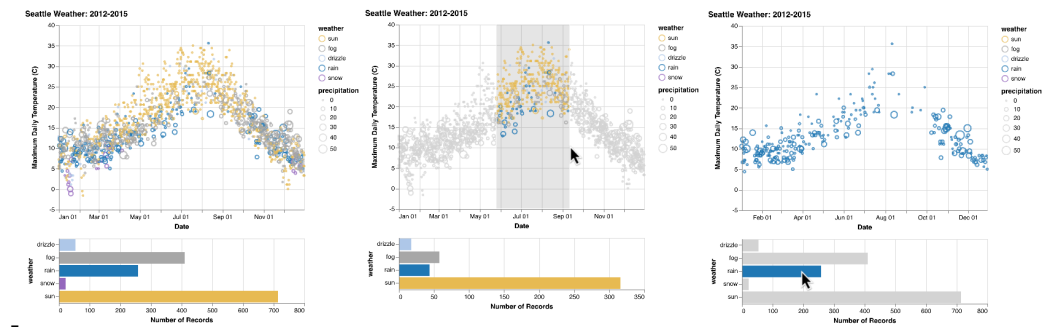
Summary

Altair is a declarative statistical visualization library for Python. Statistical visualization is a constrained subset of data visualization focused on the creation of visualizations that are helpful in statistical modeling. The constrained model of statistical visualization is usually expressed in terms of a visualization grammar (Wilkinson, 2005) that specifies how input data is transformed and mapped to visual properties (position, color, size, etc.).

Altair is based on the Vega-Lite visualization grammar (Satyanarayan, Moritz, Wongsuphasawat, & Heer, 2017), which allows a wide range of statistical visualizations to be expressed using a small number of grammar primitives. Vega-Lite implements a view composition algebra in conjunction with a novel grammar of interactions that allow users to specify interactive charts in a few lines of code. Vega-Lite is declarative; visualizations are specified using JSON data that follows the [Vega-Lite JSON schema](#). As a Python library, Altair provides an API oriented towards scientists and data scientists doing exploratory data analysis (Tukey, 1977). Altair's Python API emits Vega-Lite JSON data, which is then rendered in a user-interface such as the Jupyter Notebook, JupyterLab, or nteract using the [Vega-Lite JavaScript library](#). Vega-Lite JSON is compiled to a full Vega specification (Satyanarayan, Russell, Hoffswell, & Heer, 2016), which is then parsed and executed using a reactive runtime that internally makes use of D3.js (Bostock, Ogievetsky, & Heer, 2011).

The declarative nature of the Vega-Lite visualization grammar (Satyanarayan et al., 2017; Wilkinson, 2005), and its encoding in a formal JSON schema, provide Altair with a number of benefits. First, much of the Altair Python code and tests are generated from the Vega-Lite JSON schema, ensuring strict conformance with the Vega-Lite specification. Second, the JSON data produced by Altair and consumed by Vega-Lite provides a natural serialization and file format for statistical visualizations. This is leveraged by JupyterLab, which provides built-in rendering of these files. Third, the JSON data provides a clean integration point for non-programming based visualization user-interfaces such as Voyager (Wongsuphasawat et al., 2016, 2017).

In addition to [static documentation](#), Altair includes a set of Jupyter Notebooks with examples and an interactive tutorial. These notebooks can be read by anyone with only a web-browser through [binder](#).



The example above is an interactive Altair visualization of the weather in Seattle. The plot on the *left* shows the initial state: a scatterplot showing the temperature and dominant weather type between January and December, and a bar chart showing the counts grouped by weather type. The plot in the *middle* shows a brush that the user has drawn to focus on the summers; which are dominantly sunny. In the last plot on the *right*, the user has clicked on the a bar to filter the scatterplot.

These interactions are achieved through two selections: an interval selection on the scatterplot and a multi selection on the bar chart. The selections drive filters in the other plot. The code for this and other examples is in the [Altair gallery](#).

Acknowledgements

We thank the many contributors that created examples, wrote documentation, and reported bugs. You can find [an up-to-date list of contributors on GitHub](#).

References

- Bostock, M., Ogievetsky, V., & Heer, J. (2011). D3: Data-driven documents. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*. doi:[10.1109/TVCG.2011.185](#)
- Satyanarayan, A., Moritz, D., Wongsuphasawat, K., & Heer, J. (2017). Vega-lite: A grammar of interactive graphics. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*. doi:[10.1109/TVCG.2016.2599030](#)
- Satyanarayan, A., Russell, R., Hoffswell, J., & Heer, J. (2016). Reactive Vega: A streaming dataflow architecture for declarative interactive visualization. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*. doi:[10.1109/TVCG.2015.2467091](#)
- Tukey, J. W. (1977). *Exploratory data analysis* (Vol. 2). Reading, Mass. doi:[10.1002/bimj.4710230408](#)
- Wilkinson, L. (2005). *The grammar of graphics*. Statistics and computing. Secaucus, NJ, USA: Springer-Verlag New York, Inc. doi:[10.1007/0-387-28695-0](#)
- Wongsuphasawat, K., Moritz, D., Anand, A., Mackinlay, J., Howe, B., & Heer, J. (2016). Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*. doi:[10.1109/TVCG.2015.2467191](#)
- Wongsuphasawat, K., Qu, Z., Moritz, D., Chang, R., Ouk, F., Anand, A., Mackinlay, J., et al. (2017). Voyager 2: Augmenting visual analysis with partial view specifications. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17 (pp. 2648–2659). New York, NY, USA: ACM. doi:[10.1145/3025453.3025768](#)