Python Data Visualization -- Huge leaps forward!

James A. Bednar, Senior Manager of Technical Services, Anaconda, Inc.
About me

**Day job:** Managing our Services projects with clients

**Night job:** Managing our open-source visualization tools available at [HoloViz.org](https://holoviz.org), including [Panel](https://panel.pydata.org), [hvPlot](https://hvplot.pydata.org), [HoloViews](https://holoviews.org), [GeoViews](https://geoviews.org), [Datashader](https://datashader.org), and [Colorcet](https://colorcet.readthedocs.io).

**Side job:** Running [PyViz.org](https://pyviz.org), covering all viz and dashboarding tools.
It's been an exciting year since ACON 2019!

Major improvements in Python data visualization and dashboarding:

1. Several major new libraries for Python dashboarding
2. Widget library unification (Bokeh/Panel/ipywidgets)
3. Ultra-fast server-side rendering/aggregation
4. Use the APIs you already know
5. Annotators (Plotly/Bokeh/HoloViews)
6. Other new developments
7. What limitations does Python have for viz?
8. Exploring further

We'll cover each of these in turn.
1. Dashboarding

People used to say Python lagged R/Shiny for easy dashboard building tools, but now we have:

- **Dash**
- **Panel**
- **Voilà**
- **Streamlit**

(see [pyviz.org/dashboarding](http://pyviz.org/dashboarding))
Dash is now the old timer:

- Scalable server for Plotly plots
- Requires relatively heavy HTML/CSS knowledge
- Typically complex apps with callback structure
- Historically focused only on deployed apps
- [jupyter-dash](https://github.com/plotly/jupyter-dash) now allows incremental dashboard building
- [Dash Oil and Gas Demo](https://github.com/plotly/dash-oil-gas) (repo)
1. Dashboarding: Panel (Jun 2019)

- Easy dashboarding for any plotting library
- Deep (but optional!) Jupyter integration for easy iteration/development
- Zero cost to switch from interactive prototype to deployed app and back
- Easy static HTML/CSS output with live widgets
- glaciers.pyviz.demo.anaconda.com
1. Dashboarding: Voilà (Jun 2019)

- Turns a Jupyter notebook into a shareable dashboard
- Deploys with a full Jupyter kernel (not scalable)
- To build complex layouts, uses ipywidgets or templates (or now Panel!)
- [voila-gpx-viewer.pyviz.demo.anaconda.com](http://voila-gpx-viewer.pyviz.demo.anaconda.com)
1. Dashboarding: Streamlit (Oct 2019)

- Turn your Python scripts into dashboards
- Use your favorite editor (no Jupyter support)
- Re-runs entire script on any interaction (simple to reason about at first, but relies on heavy caching)
- bgexploration.herokuapp.com (blog)
2. Widget library unification

- Previously: Jupyter supported ipywidgets, separate from Bokeh and Dash widgets
- Painful to switch between libraries
- Complex widgets (ipyvolume, VTK) usable with only one ecosystem
- Caused unnecessary split between user groups
2. Widget library unification

- New: `ipywidgets_bokeh` and `jupyter_bokeh`  
- Wrap Bokeh models as ipywidgets, and vice versa  
- Deploy Bokeh or Panel apps in Voila  
- Deploy ipyvolume or bqplot in Bokeh Server  
- Now everyone can get along:  
  - Different notebook/deployment technologies have different strengths  
  - Use whichever ones meet your needs!
3. Ultrafast server-side rendering/aggregation

- **Datashader** and **Vaex** (2016):
  - Initially standalone projects
  - Render billions of points to a pixelated representation
  - Only a fixed-size array is sent to the local browser for display

- Datashader now supports GPU/Dask arrays/dataframes
- Vaex renders points to 1D/2D/3D
- Datashader renders to 2D, now including points/lines/areas/trimesh/quadmesh/rasters/polygons
3. Ultrafast server-side rendering/aggregation

Server-side rendering now integrated into many plotting packages:

- vaex supports bqplot, matplotlib
- datashader supported by HoloViews+Bokeh, hvPlot, and now Plotly
- datashader also integrated into Nvidia cuxfilter, umap
- mpl_scatter_density also available (points only) for mpl
4. Use the APIs you already know

- Pandas Matplotlib `.plot()` API becoming even more widely supported as a standard:
  - Pandas plotting backend is now officially configurable
  - Now (partially) supported by Plotly Express
  - See [pyviz.org/tools.html#high-level-shared-api](https://pyviz.org/tools.html#high-level-shared-api)

- `hvPlot` now supports `.plot()` API for Pandas, Xarray, Dask, Streamz, Intake, GeoPandas, and NetworkX objects -- learn API once, use it everywhere!
4. Use the APIs you already know

- **HoloViews** now offers **.df and .xr** accessors
  - Calls native Pandas or Xarray methods directly on data
  - Avoids need to learn HoloViews data API
- **Upcoming**: **interactive** support for building entire interactive apps directly from Pandas or Xarray APIs:
  - `time,q = IntSlider(...), FloatSlider(...)`
  - `ds.air.interactive(loc='left_top').isel(time=time).quantile(q=q, dim='lon').plot()`
5. Annotators/Drawing tools

- Both Bokeh and Plotly now offer shape-drawing tools:
  - user-generated shapes on top of plots
  - useful for capturing user annotations, labeling ML data, setting up analyses or simulations, ...

- Example for ML annotations: examples.pyviz.org/ml_annotators
6. Other new developments

- New spatial-processing libraries: `xarray_spatial`, `spatialpandas`, `pygeos`, `scikit-geometry`, `cuSpatial`
- New editor support for interactive plots in VSCode
- Colorcet: 256-color categorical colormaps
- HoloViews: automatic linked brushing
7. What limitations does Python have for viz?

• Not much, now!
• Of course, some users will always want GUI BI tools (Tableau, Excel, etc.)
• Direct development of JavaScript apps will always be useful for the most scalable and responsive websites
• R users will prefer R solutions, of course!
8. Exploring further

• See full list of all Python viz tools at pyviz.org/tools
• See reproducible larger examples at examples.pyviz.org, awesome-panel.org, and awesome-streamlit.org
• See each package's website for more details
  ◦ holoviz.org, panel.holoviz.org, hvplot.holoviz.org, datashader.org
  ◦ plotly.com/python, plotly.com/dash
  ◦ matplotlib.org, bokeh.org, altair-viz.github.io
Thank you.