Bokeh Basics

Bryan Van de Ven
@bigreddot

Aug 2019

Bokeh ?

Interactive visualization, widgets, and tools

Versatile and high-level graphics

Streaming, dynamic, large data

For the browser, with or without a server

No JavaScript

https://bokeh.org

What

Bokeh ?

Interactive visualization, widgets, and tools

Versatile and high-level graphics

Streaming, dynamic, large data

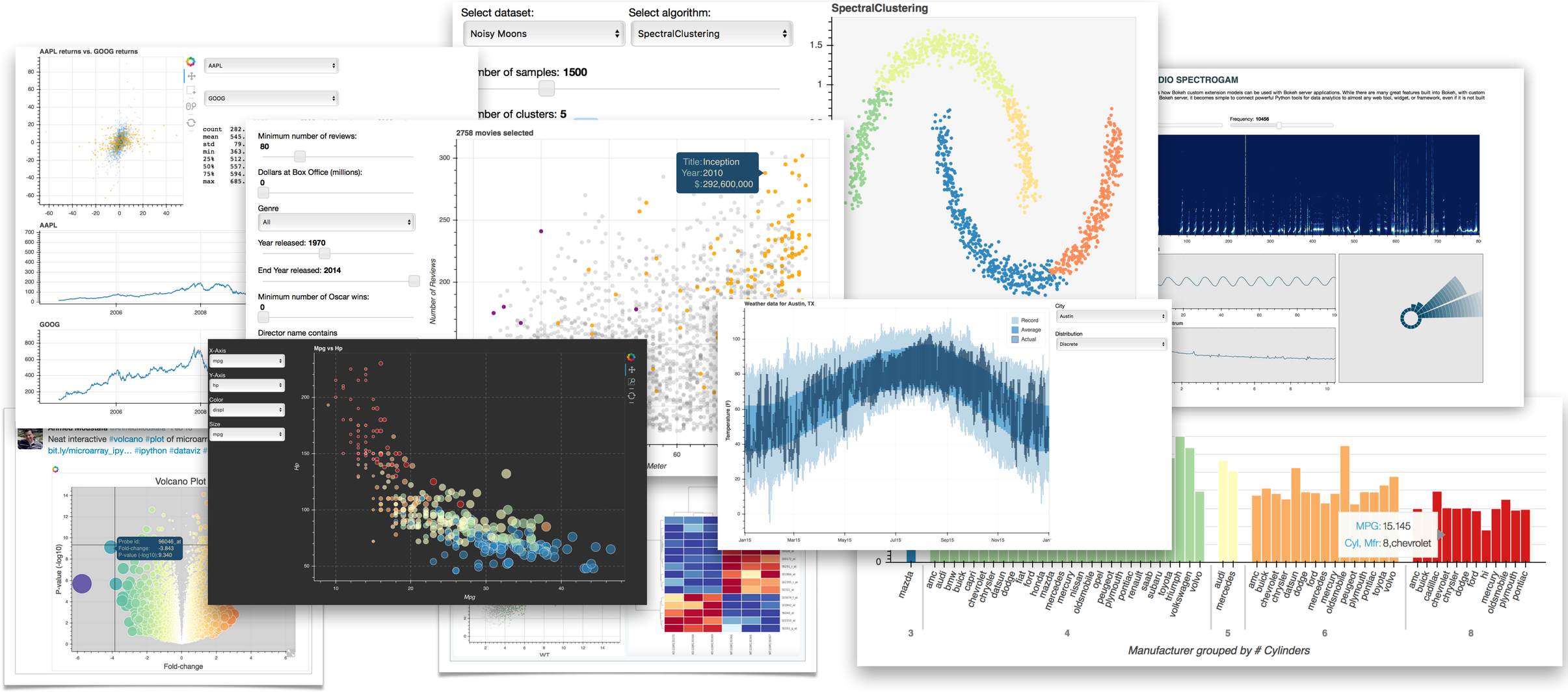
For the browser, with or without a server

No JavaScript

https://bokeh.org

What

Simple (to write) Apps for Data Science



What

Concentrate on your work

Why

Concentrate on your work

Mostly or completely written in Python, little HTML or CSS coding

Concentrate on your work

Mostly or completely written in Python, little HTML or CSS coding

Simple python scripts, no special classes of frameworks

Concentrate on your work

Mostly or completely written in Python, little HTML or CSS coding

Simple python scripts, no special classes of frameworks

Useful for exploratory analysis or sharing and publishing

Concentrate on your work

Mostly or completely written in Python, little HTML or CSS coding

Simple python scripts, no special classes of frameworks

Useful for exploratory analysis or sharing and publishing

Automatically mirrors and synchronizes Python and browser state

Concentrate on your work

Mostly or completely written in Python, little HTML or CSS coding

Simple python scripts, no special classes of frameworks

Useful for exploratory analysis or sharing and publishing

Automatically mirrors and synchronizes Python and browser state

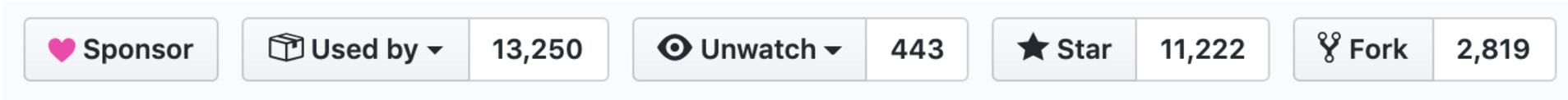
Connect the full PyData stack to interactive web apps

Yesterday and Today

Circa June 2014

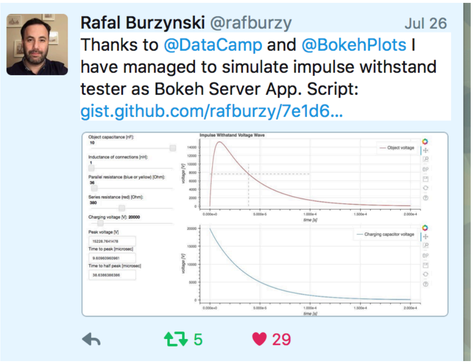
Release	
Status	
Conda	
PyPI	

Circa This Week 2019



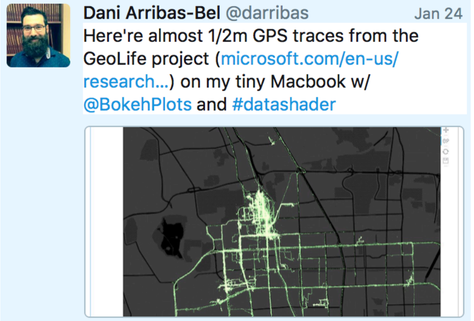
Latest Release	version 1.3.4 npm package 1.3.4	Conda	conda 332k/month
License	license BSD-3-Clause	PyPI	downloads 463k/month
Sponsorship	powered by NumFOCUS	Live Tutorial	launch binder
Build Status	build failing build passing	Support	discourse 14k posts
Static Analysis	Better Code 7 / 10	Twitter	Follow 10k

Users Gonna Use



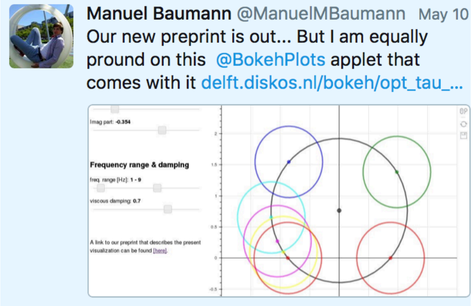
Rafal Burzynski @rafburzy Jul 26
Thanks to @DataCamp and @BokehPlots I have managed to simulate impulse withstand tester as Bokeh Server App. Script: gist.github.com/rafburzy/7e1d6...

5 29



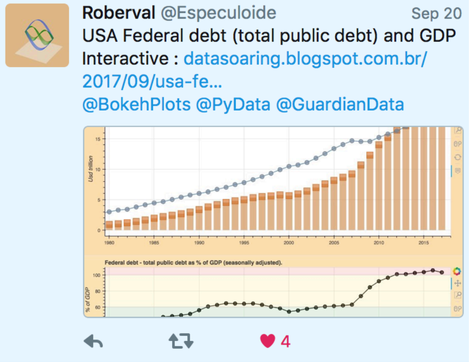
Dani Arribas-Bel @darribas Jan 24
Here're almost 1/2m GPS traces from the GeoLife project (microsoft.com/en-us/research...) on my tiny Macbook w/ @BokehPlots and #datashader

1 14



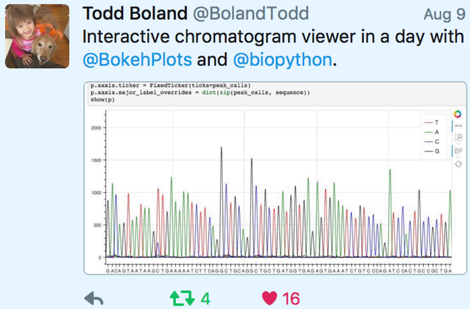
Manuel Baumann @ManuelMBaumann May 10
Our new preprint is out... But I am equally pround on this @BokehPlots applet that comes with it delft.diskos.nl/bokeh/opt_tau_...

9 16



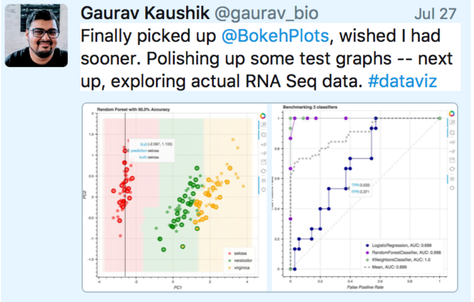
Roberval @Especuloide Sep 20
USA Federal debt (total public debt) and GDP Interactive : datasoaring.blogspot.com.br/2017/09/usa-fe... @BokehPlots @PyData @GuardianData

4



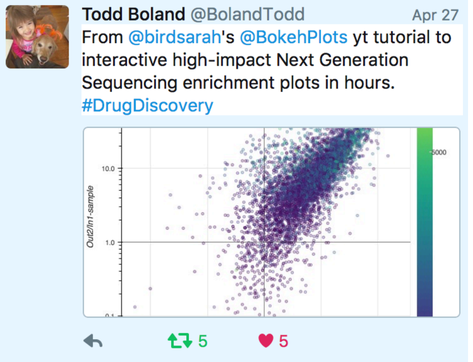
Todd Boland @BolandTodd Aug 9
Interactive chromatogram viewer in a day with @BokehPlots and @biopython.

4 16



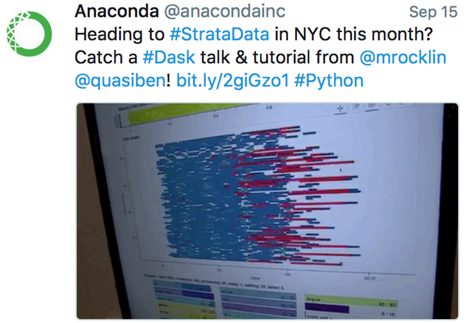
Gaurav Kaushik @gaurav_bio Jul 27
Finally picked up @BokehPlots, wished I had sooner. Polishing up some test graphs -- next up, exploring actual RNA Seq data. #dataviz

21 72



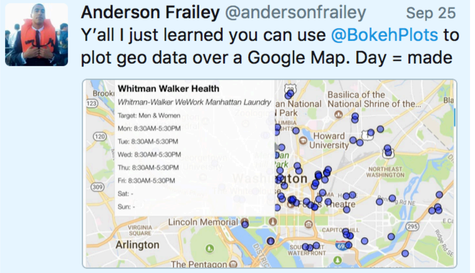
Todd Boland @BolandTodd Apr 27
From @birdsarah's @BokehPlots yt tutorial to interactive high-impact Next Generation Sequencing enrichment plots in hours. #DrugDiscovery

5 5



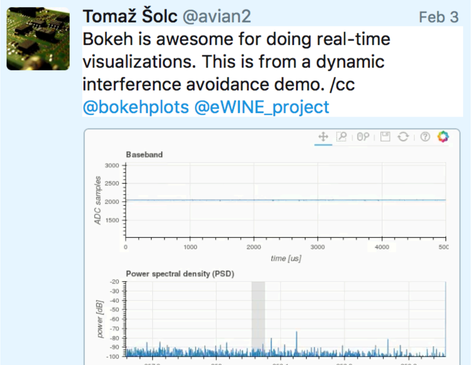
Anaconda @anacondainc Sep 15
Heading to #StrataData in NYC this month? Catch a #Dask talk & tutorial from @mrocklin @quasiben! bit.ly/2giGzo1 #Python

7 18



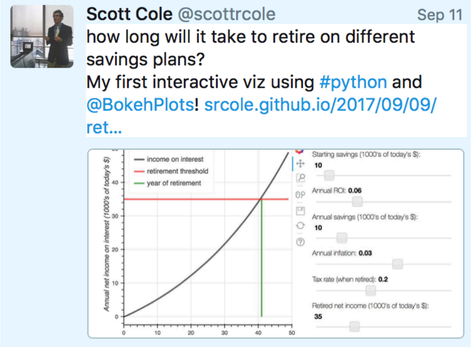
Anderson Frailey @andersonfrailey Sep 25
Y'all I just learned you can use @BokehPlots to plot geo data over a Google Map. Day = made

10 34



Tomaž Šolc @avian2 Feb 3
Bokeh is awesome for doing real-time visualizations. This is from a dynamic interference avoidance demo. /cc @bokehplots @eWINE_project

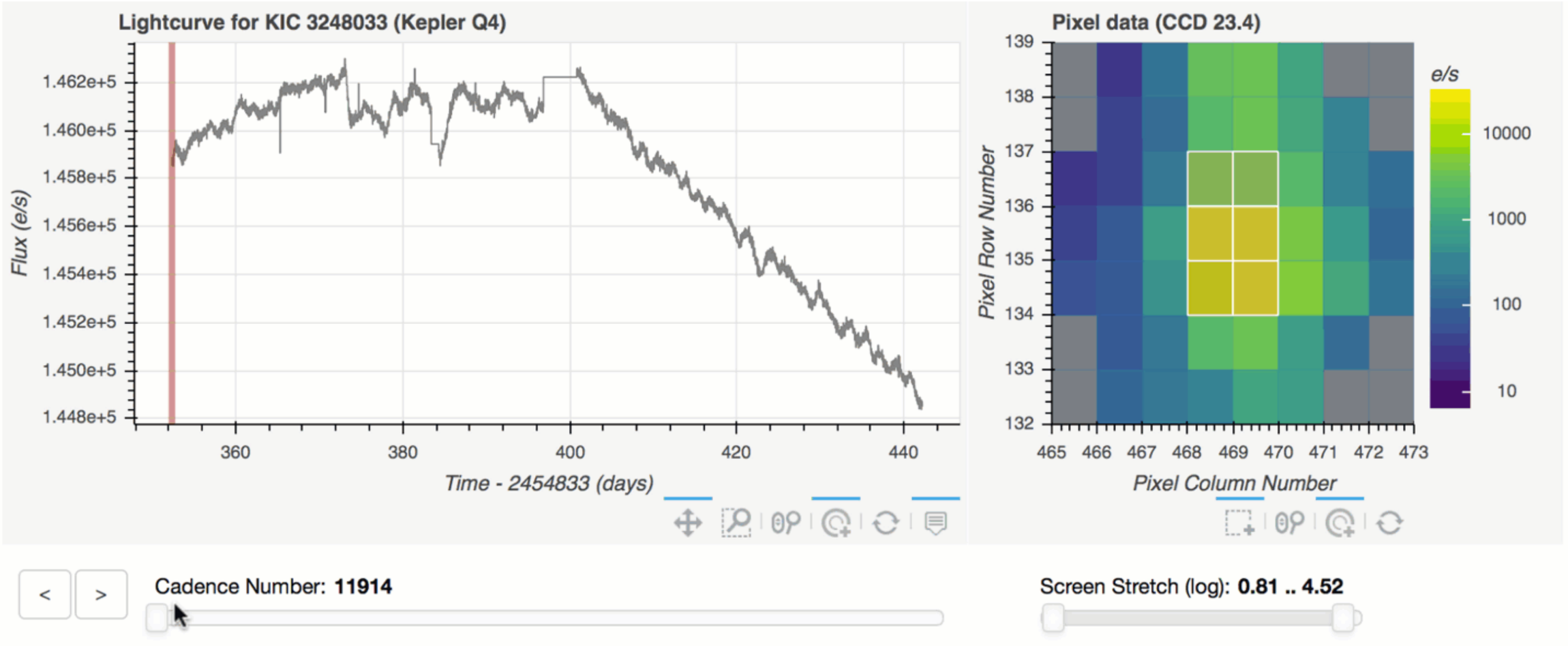
45 70



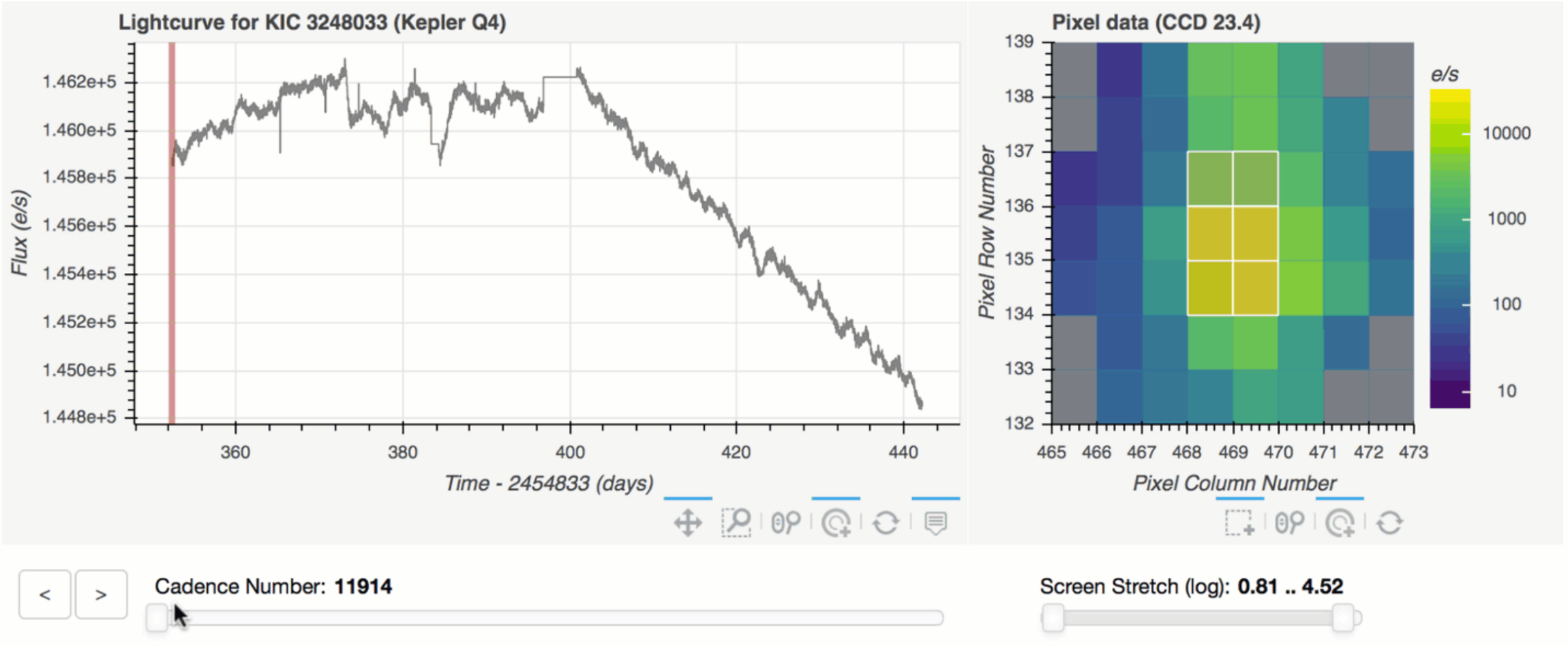
Scott Cole @scottrcole Sep 11
how long will it take to retire on different savings plans?
My first interactive viz using #python and @BokehPlots! srcole.github.io/2017/09/09/ret...

8 30

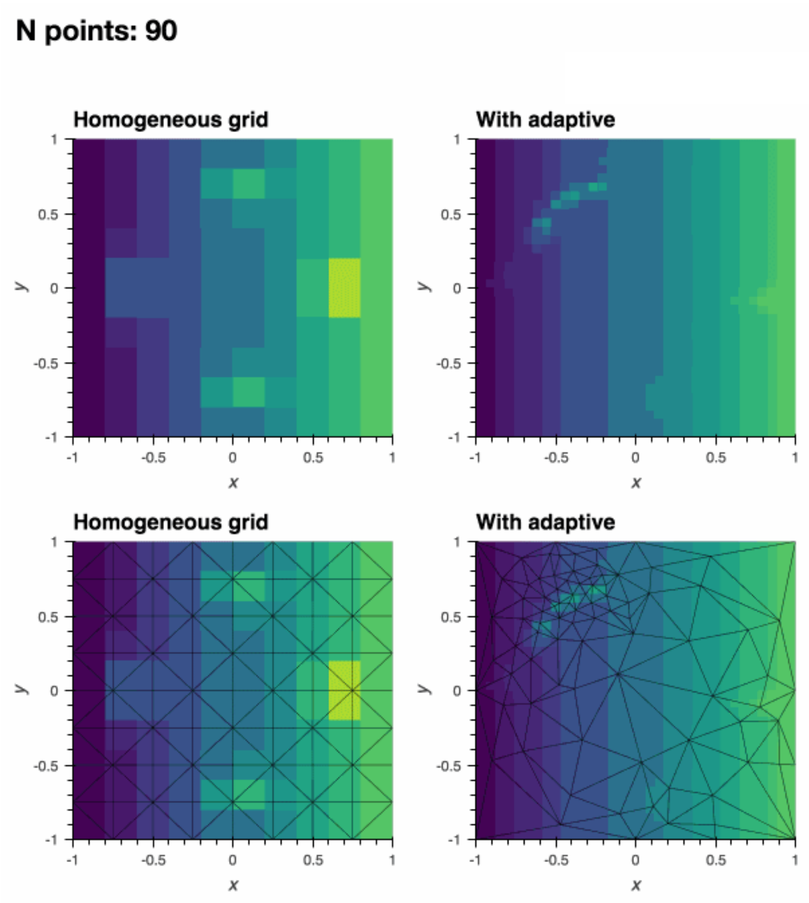
KeplerGO/lightkurve



KeplerGO/lightkurve

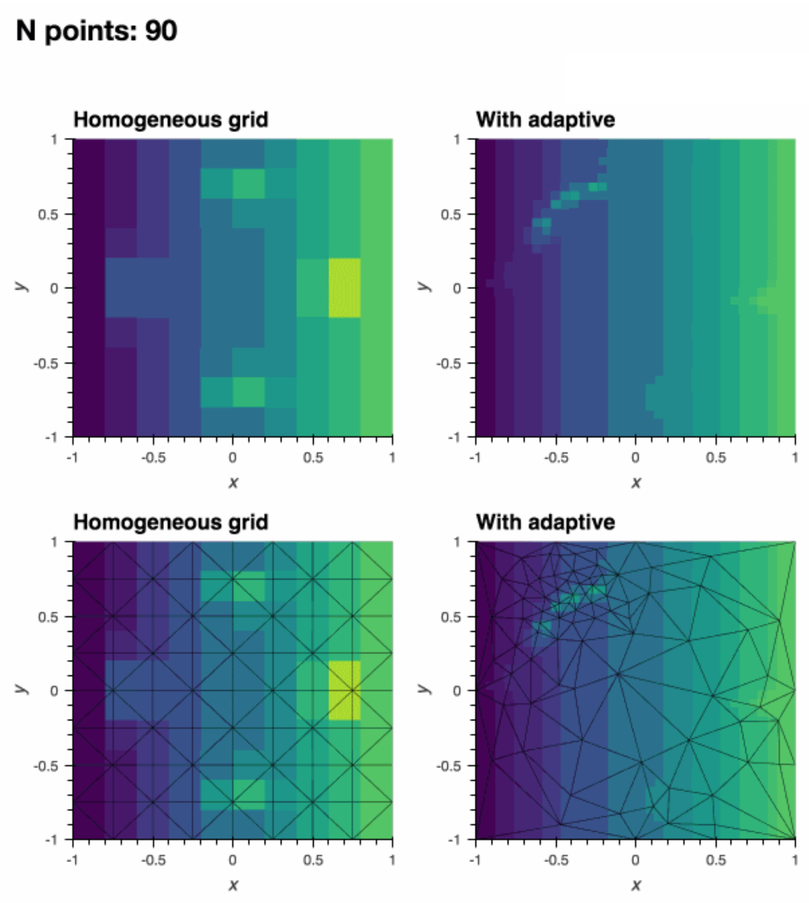


Adaptive



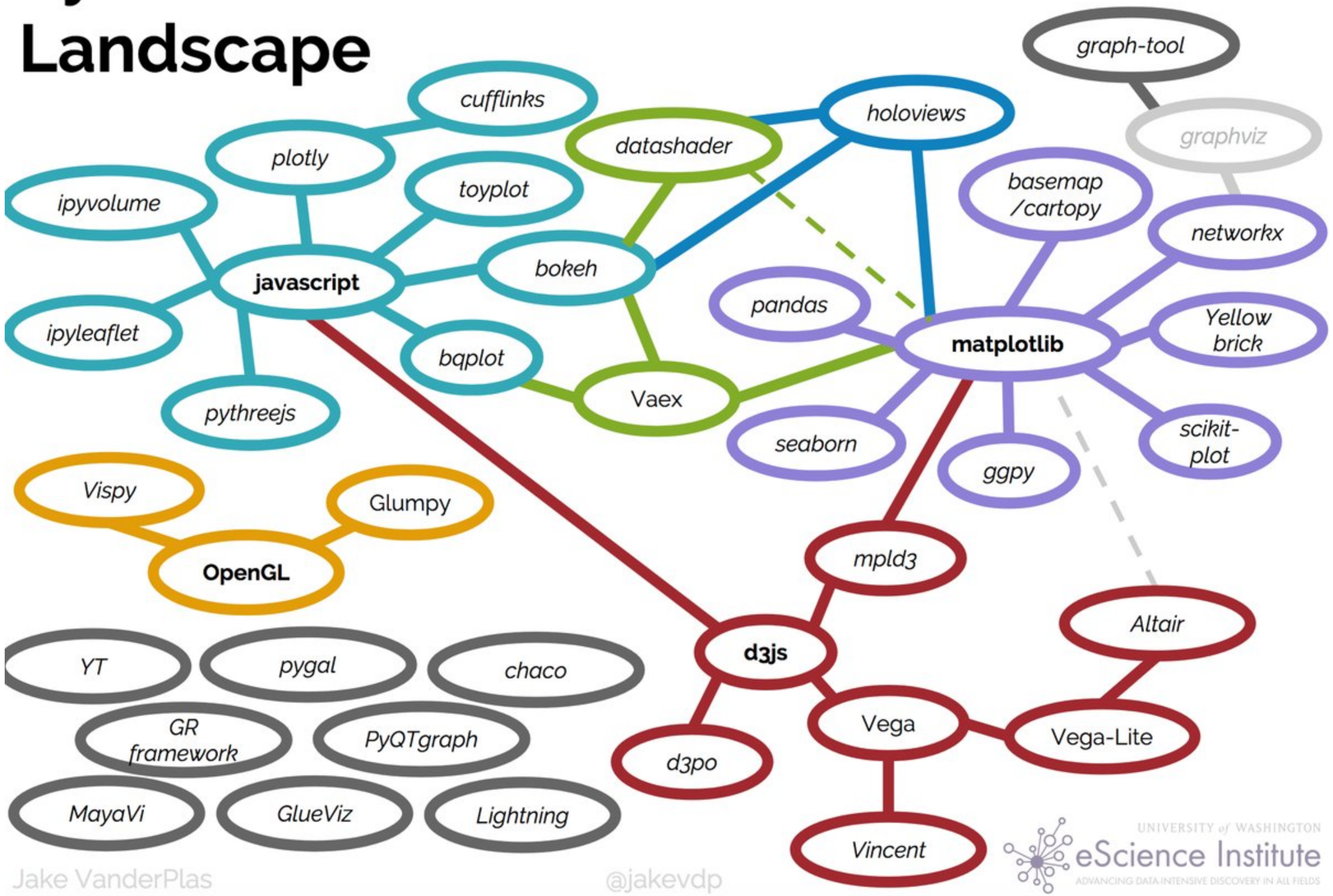
https://github.com/python-adaptive/adaptive/

Adaptive



https://github.com/python-adaptive/adaptive/

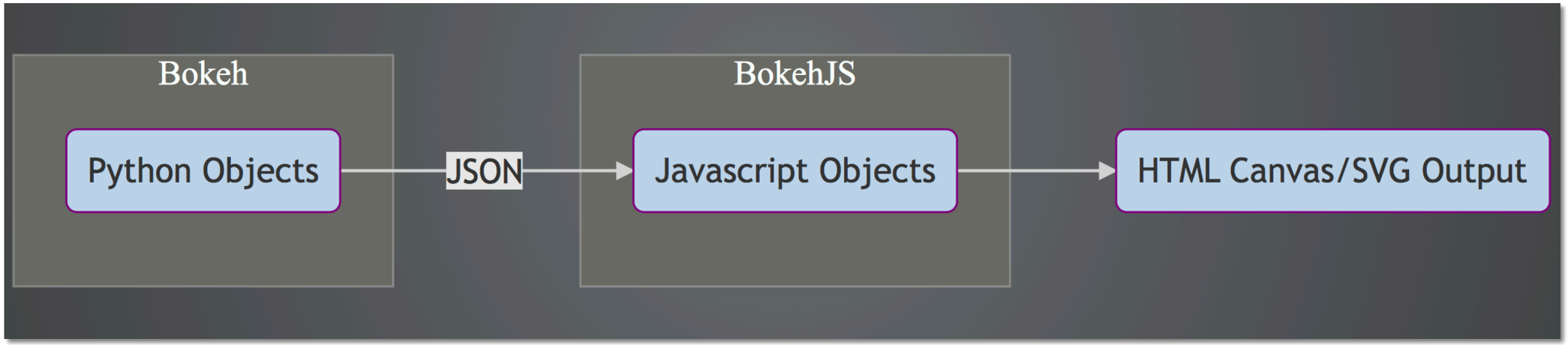
Python's Visualization Landscape



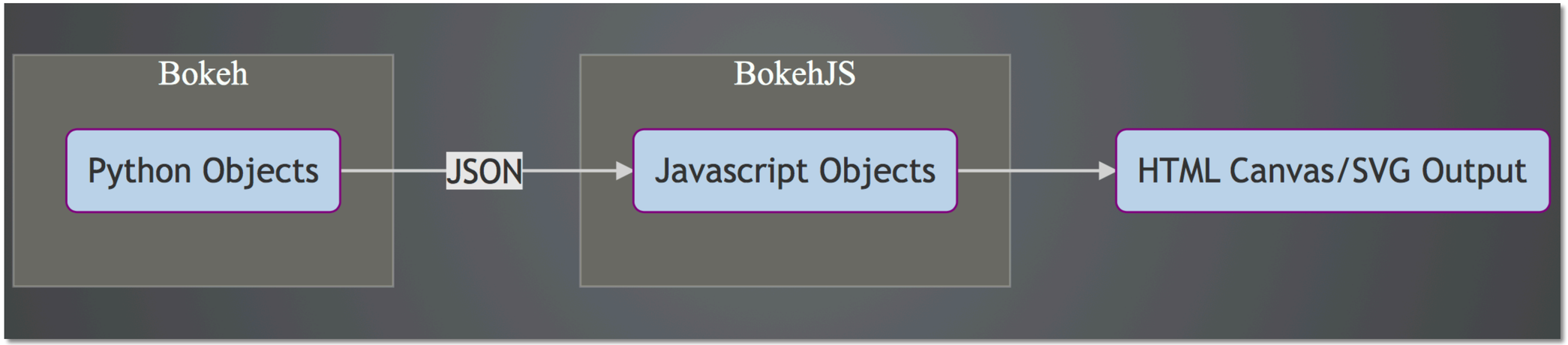
Basic Principles

Building Blocks

Building Blocks



Building Blocks




```
>>> from bokeh.models import Circle
>>> circle = Circle(x=5, y=10, fill_color="red")
>>> circle.to_json(include_defaults=False)
{'fill_color': {'value': 'red'}, 'id': '1002', 'x': {'value': 5}, 'y': {'value': 10}}
>>>
```

Building Blocks

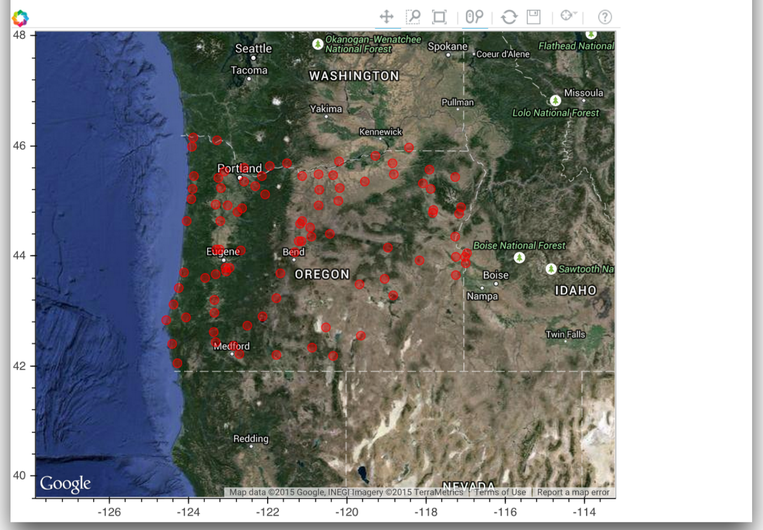
```html
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>iris.py example</title>
        <link rel="stylesheet" href="../../../../../anaconda/envs/bokehdemo/lib/python2.7/site-packages/bokeh/server/static/css/bokeh.min.css" type="text/css" />
        <script type="text/javascript" src="../../../../../anaconda/envs/bokehdemo/lib/python2.7/site-packages/bokeh/server/static/js/bokeh.min.js"></script>
        <script type="text/javascript">
            $(function() {
                var all_models = [{"attributes": {"column_names": ["fill_color", "line_color", "x", "y"], "doc": null, "selected": [], "discrete_ranges": {}, "cont_ranges": {}, "data": {"line_color": ["red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red",
"red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "green", "green", "green", "green", "green", "green", "green", "green", "green",
"green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green",
"green", "green", "green", "green", "green", "green", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue",
"blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue"], "x": [1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5, 1.5, 1.6, 1.4, 1.1, 1.2, 1.5, 1.3, 1.4, 1.7, 1.5, 1.7, 1.5, 1.0, 1.7, 1.9, 1.6, 1.6, 1.5, 1.4, 1.6, 1.6, 1.5, 1.5, 1.4, 1.5,
1.2, 1.3, 1.4, 1.3, 1.5, 1.3, 1.3, 1.3, 1.6, 1.9, 1.4, 1.6, 1.4, 1.5, 1.4, 4.7, 4.5, 4.9, 4.0, 4.6, 4.5, 4.7, 3.3, 4.6, 3.9, 3.5, 4.2, 4.0, 4.7, 3.6, 4.4, 4.5, 4.1, 4.5, 3.9, 4.8, 4.0, 4.9, 4.7, 4.3, 4.4, 4.8, 5.0, 4.5, 3.5, 3.8, 3.7, 3.9, 5.1, 4.5, 4.5, 4.7, 4.4, 4.1, 4.0, 4.4, 4.6, 4.0, 3.3, 4.2, 4.2, 4.2, 4.3,
3.0, 4.1, 6.0, 5.1, 5.9, 5.6, 5.8, 6.6, 4.5, 6.3, 5.8, 6.1, 5.1, 5.3, 5.5, 5.0, 5.1, 5.3, 5.5, 6.7, 6.9, 5.0, 5.7, 4.9, 6.7, 4.9, 5.7, 6.0, 4.8, 4.9, 5.6, 5.8, 6.1, 6.4, 5.6, 5.1, 5.6, 6.1, 5.6, 5.5, 4.8, 5.4, 5.6, 5.1, 5.1, 5.9, 5.7, 5.2, 5.0, 5.2, 5.4, 5.1], "fill_color": ["red", "red", "red", "red", "red", "red",
"red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "red", "green",
"green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green",
"green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "green", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue",
"blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue", "blue"], "y": [0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2, 0.2, 0.1, 0.1, 0.2, 0.4, 0.4, 0.3, 0.3, 0.3,
0.2, 0.4, 0.2, 0.5, 0.2, 0.2, 0.4, 0.2, 0.2, 0.2, 0.2, 0.4, 0.1, 0.2, 0.2, 0.2, 0.2, 0.1, 0.2, 0.2, 0.3, 0.3, 0.2, 0.6, 0.4, 0.3, 0.2, 0.2, 0.2, 0.2, 1.4, 1.5, 1.5, 1.3, 1.5, 1.3, 1.6, 1.0, 1.3, 1.4, 1.0, 1.5, 1.0, 1.4, 1.3, 1.4, 1.5, 1.0, 1.5, 1.1, 1.8, 1.3, 1.5, 1.2, 1.3, 1.4, 1.4, 1.7, 1.5, 1.0, 1.1, 1.0, 1.2,
1.6, 1.5, 1.6, 1.5, 1.3, 1.3, 1.3, 1.2, 1.4, 1.2, 1.0, 1.3, 1.2, 1.3, 1.3, 1.1, 1.3, 2.5, 1.9, 2.1, 1.8, 2.2, 2.1, 1.7, 1.8, 1.8, 2.5, 2.0, 1.9, 2.1, 2.0, 2.4, 2.3, 1.8, 2.2, 2.3, 1.5, 2.3, 2.0, 2.0, 1.8, 2.1, 1.8, 1.8, 1.8, 2.1, 1.6, 1.9, 2.0, 2.2, 1.5, 1.4, 2.3, 2.4, 1.8, 1.8, 2.1, 2.4, 2.3, 1.9, 2.3, 2.5, 2.3,
1.9, 2.0, 2.3, 1.8]}, "id": "5e71b46a-0d81-4a18-8402-188816471c0c"}, "type": "ColumnDataSource", "id": "5e71b46a-0d81-4a18-8402-188816471c0c"}, {"attributes": {"sources": [{"source": {"type": "ColumnDataSource", "id": "5e71b46a-0d81-4a18-8402-188816471c0c"}, "columns": ["x"]}], "id":
"bbaf66fb-48b8-474a-8dae-910a995186f6", "doc": null}, "type": "DataRange1d", "id": "bbaf66fb-48b8-474a-8dae-910a995186f6"}, {"attributes": {"sources": [{"source": {"type": "ColumnDataSource", "id": "5e71b46a-0d81-4a18-8402-188816471c0c"}, "columns": ["y"]}], "id": "8377dd3b-9c4e-41ce-8930-76a92a68e907", "doc": null},
"type": "DataRange1d", "id": "8377dd3b-9c4e-41ce-8930-76a92a68e907"}, {"attributes": {"doc": null, "id": "24c8ae7c-f3c8-4c88-9f5d-dcbe59506791"}, "type": "BasicTickFormatter", "id": "24c8ae7c-f3c8-4c88-9f5d-dcbe59506791"}, {"attributes": {"doc": null, "id": "3720fa34-cea8-4b54-a51b-c738a1ef96fb"}, "type":
"BasicTicker", "id": "3720fa34-cea8-4b54-a51b-c738a1ef96fb"}, {"attributes": {"plot": {"type": "Plot", "id": "iris"}, "doc": null, "bounds": "auto", "id": "0ae2ae05-3abd-414a-9840-c5e804de9661", "location": "min", "formatter": {"type": "BasicTickFormatter", "id": "24c8ae7c-f3c8-4c88-9f5d-dcbe59506791"}, "ticker":
{"type": "BasicTicker", "id": "3720fa34-cea8-4b54-a51b-c738a1ef96fb"}, "dimension": 0}, "type": "LinearAxis", "id": "0ae2ae05-3abd-414a-9840-c5e804de9661"}, {"attributes": {"plot": {"type": "Plot", "id": "iris"}, "doc": null, "axis": {"type": "LinearAxis", "id": "0ae2ae05-3abd-414a-9840-c5e804de9661"}, "id":
"25d48bf1-6583-4aff-9e47-dd57e304fb7a", "dimension": 0}, "type": "Grid", "id": "25d48bf1-6583-4aff-9e47-dd57e304fb7a"}, {"attributes": {"doc": null, "id": "d88bdf6f-b1a7-49c1-b71e-df2c1156f202"}, "type": "BasicTickFormatter", "id": "d88bdf6f-b1a7-49c1-b71e-df2c1156f202"}, {"attributes": {"doc": null, "id":
"434ab651-0a3a-4bab-aa7a-34844b833bce"}, "type": "BasicTicker", "id": "434ab651-0a3a-4bab-aa7a-34844b833bce"}, {"attributes": {"plot": {"type": "Plot", "id": "iris"}, "doc": null, "bounds": "auto", "id": "53cf6b9d-1c82-48d2-8094-5a81fed497d9", "location": "min", "formatter": {"type": "BasicTickFormatter", "id":
"d88bdf6f-b1a7-49c1-b71e-df2c1156f202"}, "ticker": {"type": "BasicTicker", "id": "434ab651-0a3a-4bab-aa7a-34844b833bce"}, "dimension": 1}, "type": "LinearAxis", "id": "53cf6b9d-1c82-48d2-8094-5a81fed497d9"}, {"attributes": {"plot": {"type": "Plot", "id": "iris"}, "doc": null, "axis": {"type": "LinearAxis", "id":
"53cf6b9d-1c82-48d2-8094-5a81fed497d9"}, "id": "21bd25eb-22d3-427c-a6a3-5e3afc96cc2a", "dimension": 1}, "type": "Grid", "id": "21bd25eb-22d3-427c-a6a3-5e3afc96cc2a"}, {"attributes": {"data_source": {"type": "ColumnDataSource", "id": "5e71b46a-0d81-4a18-8402-188816471c0c"}, "server_data_source": null, "doc": null,
"nonselection_glyphspec": {"line_color": {"value": "#1f77b4"}, "angle_units": "deg", "fill_color": {"value": "#1f77b4"}, "visible": null, "line_dash_offset": 0, "line_join": "miter", "size": {"units": "screen", "value": 10}, "line_alpha": {"units": "data", "value": 0.1}, "radius_units": "screen", "end_angle_units":
"deg", "valign": null, "length_units": "screen", "start_angle_units": "deg", "line_cap": "butt", "line_dash": [], "line_width": {"units": "data", "field": "line_width"}, "type": "circle", "fill_alpha": {"units": "data", "value": 0.1}, "halign": null, "y": {"units": "data", "field": "y"}, "x": {"units": "data",
"field": "x"}, "margin": null}, "xdata_range": null, "ydata_range": null, "glyphspec": {"line_color": {"units": "data", "field": "line_color"}, "line_alpha": {"units": "data", "value": 1.0}, "fill_color": {"units": "data", "field": "fill_color"}, "line_width": {"units": "data", "field": "line_width"}, "fill_alpha":
{"units": "data", "value": 0.2}, "y": {"units": "data", "field": "y"}, "x": {"units": "data", "field": "x"}, "type": "circle", "size": {"units": "screen", "value": 10}}, "id": "093300cf-6759-4449-877b-7731476588a0"}, "type": "Glyph", "id": "093300cf-6759-4449-877b-7731476588a0"}, {"attributes": {"plot": null, "doc":
null, "renderers": [{"type": "Glyph", "id": "093300cf-6759-4449-877b-7731476588a0"}], "id": "9a60e0da-efe5-4b08-a4f6-7ed315d67b9b"}, "type": "BoxSelectTool", "id": "9a60e0da-efe5-4b08-a4f6-7ed315d67b9b"}, {"attributes": {"doc": null, "tool": {"type": "BoxSelectTool", "id": "9a60e0da-efe5-4b08-a4f6-7ed315d67b9b"},
"id": "5a0e8c76-4893-452b-b2e8-cefb1a232437"}, "type": "BoxSelection", "id": "5a0e8c76-4893-452b-b2e8-cefb1a232437"}, {"attributes": {"plot": {"type": "Plot", "id": "iris"}, "dimensions": ["width", "height"], "doc": null, "id": "a4b154c7-b674-4f86-93f8-770cf7a0d9b5"}, "type": "PanTool", "id": "a4b154c7-
b674-4f86-93f8-770cf7a0d9b5"}, {"attributes": {"plot": {"type": "Plot", "id": "iris"}, "dimensions": ["width", "height"], "doc": null, "id": "3ba5854b-e047-47c2-989b-15b5b79cb205"}, "type": "WheelZoomTool", "id": "3ba5854b-e047-47c2-989b-15b5b79cb205"}, {"attributes": {"plot": {"type": "Plot", "id": "iris"}, "id":
"0a583af8-4db5-45ea-b09b-16562035ccc4", "doc": null}, "type": "PreviewSaveTool", "id": "0a583af8-4db5-45ea-b09b-16562035ccc4"}, {"attributes": {"plot": {"type": "Plot", "id": "iris"}, "id": "5621f214-17c9-417f-aaed-f841745f489f", "doc": null}, "type": "ResizeTool", "id": "5621f214-17c9-417f-aaed-f841745f489f"},
{"attributes": {"plot": {"type": "Plot", "id": "iris"}, "id": "98be8a66-dfaa-4f2d-95cd-0296a3647da1", "doc": null}, "type": "ResetTool", "id": "98be8a66-dfaa-4f2d-95cd-0296a3647da1"}, {"attributes": {"outer_height": 600, "x_range": {"type": "DataRange1d", "id": "bbaf66fb-48b8-474a-8dae-910a995186f6"}, "y_range":
{"type": "DataRange1d", "id": "8377dd3b-9c4e-41ce-8930-76a92a68e907"}, "outer_width": 600, "renderers": [{"type": "LinearAxis", "id": "0ae2ae05-3abd-414a-9840-c5e804de9661"}, {"type": "Grid", "id": "25d48bf1-6583-4aff-9e47-dd57e304fb7a"}, {"type": "LinearAxis", "id": "53cf6b9d-1c82-48d2-8094-5a81fed497d9"}, {"type":
"Grid", "id": "21bd25eb-22d3-427c-a6a3-5e3afc96cc2a"}, {"type": "BoxSelection", "id": "6451a3a2-d1d7-401e-8ec6-ed92c626f448"}, {"type": "BoxSelection", "id": "5a0e8c76-4893-452b-b2e8-cefb1a232437"}, {"type": "Glyph", "id": "093300cf-6759-4449-877b-7731476588a0"}], "id": "iris", "data_sources": [], "doc": null,
"canvas_height": 600, "title": "Plot", "tools": [{"type": "PanTool", "id": "a4b154c7-b674-4f86-93f8-770cf7a0d9b5"}, {"type": "WheelZoomTool", "id": "3ba5854b-e047-47c2-989b-15b5b79cb205"}, {"type": "BoxZoomTool", "id": "a047dc9b-0dd1-4883-8575-550cd63409fa"}, {"type": "PreviewSaveTool", "id": "0a583af8-4db5-45ea-
b09b-16562035ccc4"}, {"type": "ResizeTool", "id": "5621f214-17c9-417f-aaed-f841745f489f"}, {"type": "BoxSelectTool", "id": "9a60e0da-efe5-4b08-a4f6-7ed315d67b9b"}, {"type": "ResetTool", "id": "98be8a66-dfaa-4f2d-95cd-0296a3647da1"}], "canvas_width": 600}, "type": "Plot", "id": "iris"}, {"attributes": {"plot":
{"type": "Plot", "id": "iris"}, "id": "a047dc9b-0dd1-4883-8575-550cd63409fa", "doc": null}, "type": "BoxZoomTool", "id": "a047dc9b-0dd1-4883-8575-550cd63409fa"}, {"attributes": {"doc": null, "tool": {"type": "BoxZoomTool", "id": "a047dc9b-0dd1-4883-8575-550cd63409fa"}, "id": "6451a3a2-d1d7-401e-8ec6-ed92c626f448"},
"type": "BoxSelection", "id": "6451a3a2-d1d7-401e-8ec6-ed92c626f448"}, {"attributes": {"doc": null, "children": [{"type": "Plot", "id": "iris"}], "id": "475ad0da-baf5-48be-902b-166b060b6978"}, "type": "PlotContext", "id": "475ad0da-baf5-48be-902b-166b060b6978"}];
                var modelid = "475ad0da-baf5-48be-902b-166b060b6978";
                var modeltype = "PlotContext";
                var elementid = "8bb1deb5-74cb-4b28-b44f-c89dc5701d69";
                console.log(modelid, modeltype, elementid);
                Bokeh.load_models(all_models);
                var model = Bokeh.Collections(modeltype).get(modelid);
                var view = new model.default_view({model: model, el: '#8bb1deb5-74cb-4b28-b44f-c89dc5701d69'});
            });
        </script>
    </head>
    <body>
        <div class="plotdiv" id="8bb1deb5-74cb-4b28-b44f-c89dc5701d69">Plots</div>
    </body>
</html>
```

Not just for Python

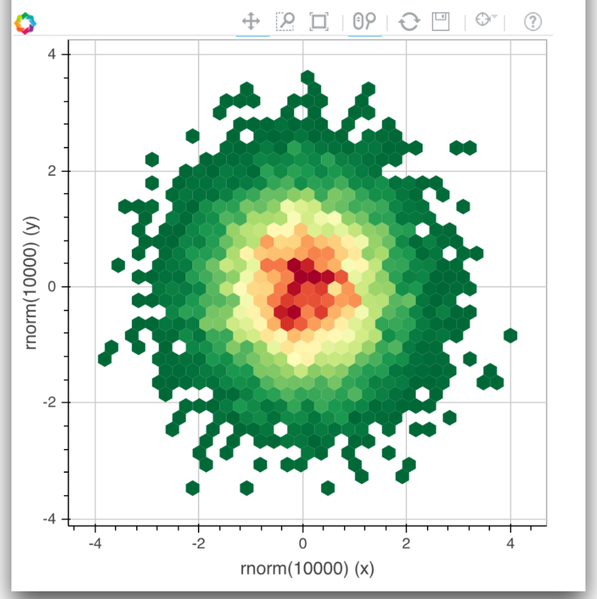
Plays well with R ecosystem: HTMLwidget, RMarkdown…

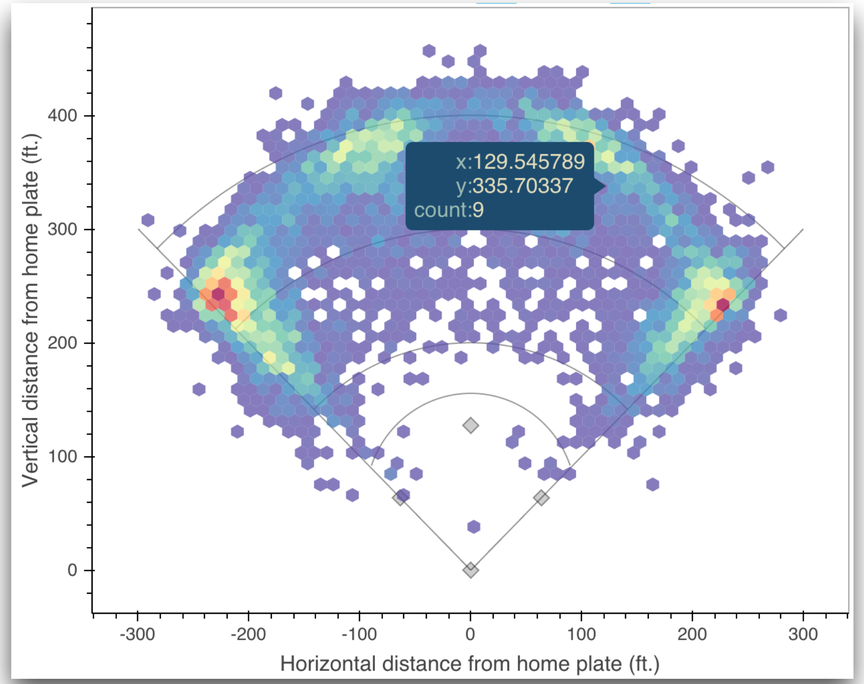
```
orstationc <- read.csv("http://geog.uoregon.edu/bartlein/old_courses/geog414s05/data/orstationc.c

gmap(lat = 44.1, lng = -120.767, zoom = 6, width = 700, height = 600) %>%
  ly_points(lon, lat, data = orstationc, alpha = 0.8, col = "red",
    hover = c(station, Name, elev, tann))
```




```
figure() %>% ly_hexbin(rnorm(10000), rnorm(10000))
```





http://hafen.github.io/rbokeh

Map Visual Properties To Data Columns

```
from bokeh.models import ColumnDataSource

source = ColumnDataSource({
    'time': [...],
    'value': [...],
    'color': [...],
})

plot = figure()

plot.circle(x='time', y='value', fill_color='color')
```

Pick what graphical primitives to use, provide the data, and specify how to map visual properties to data fields. Bokeh will take care of the rest.

Streaming Data

Two data source methods:

.stream to append data incrementally to column ends

.patch for random access updates anywhere

Styling

```yaml
attrs:
  Axis:
    major_tick_line_color: "#E0E0E0"
    minor_tick_line_color: null
    axis_line_color: "#E0E0E0"
    major_label_text_color: "#E0E0E0"
    major_label_text_font_size: "0.7em"
    axis_label_standoff: 10
    axis_label_text_color: "#E0E0E0"
    axis_label_text_font_size: "0.8em"
  Grid:
    grid_line_color: null
  Plot:
    background_fill_color: "#20262B"
    border_fill_color: null
    outline_line_color: null
    sizing_mode: "scale_width"
    min_border_left: 60
    min_border_right: 60
    min_border_top: 0
    min_border_bottom: 4
  Title:
    text_color: "#CCCCCC"
```

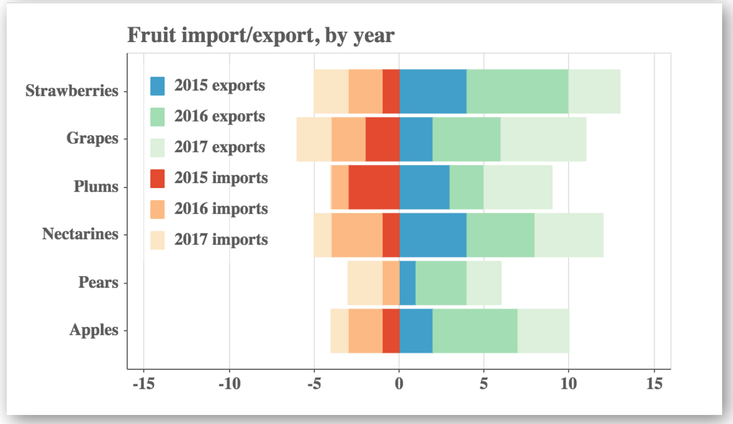
Any Bokeh property can be targeted

Simple YAML or JSON format

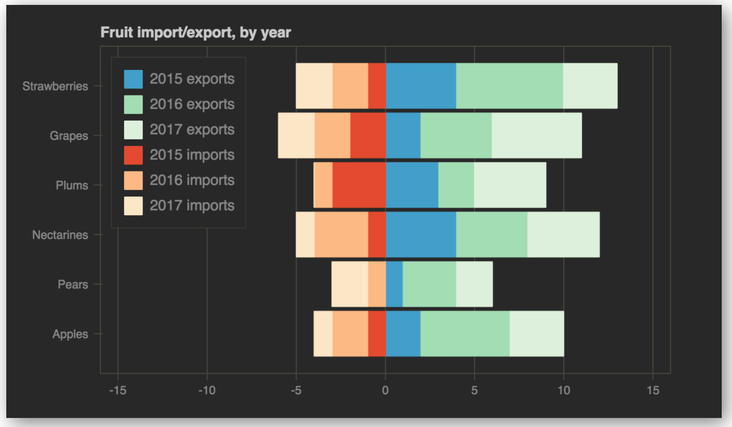
Useful for a consistent look across plots

Some User Contributed Styles

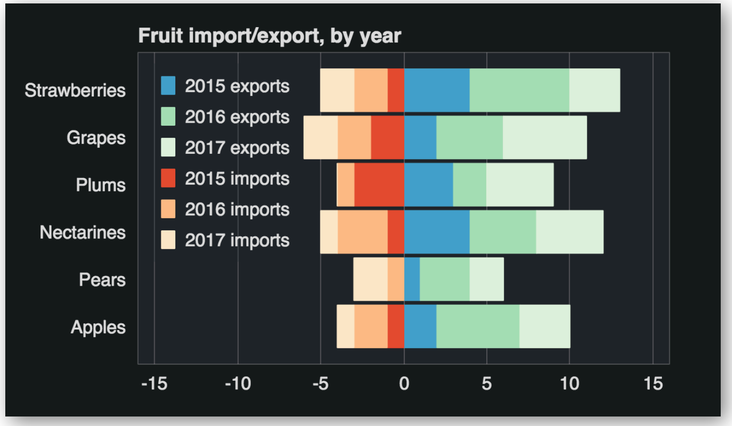
Caliber



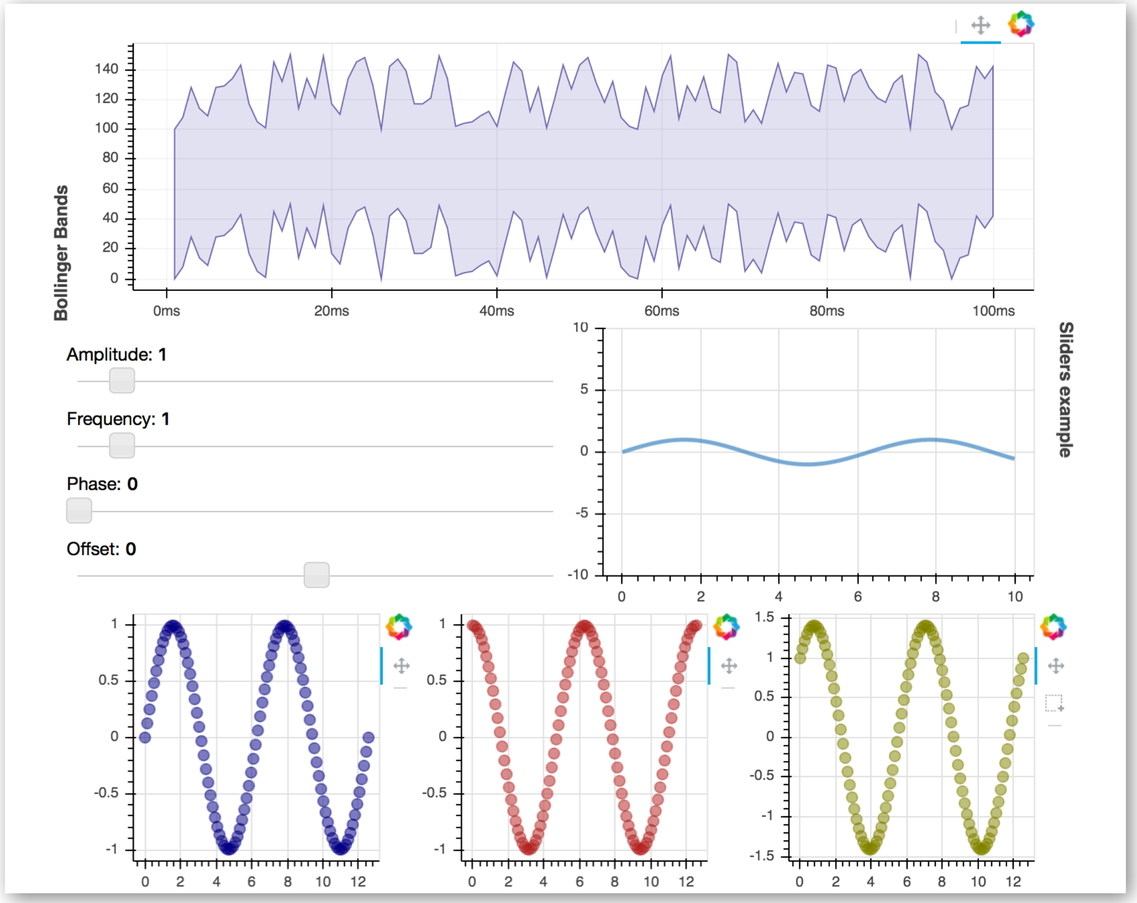
Monokai Dark



Minimal Dark



Built-in Layouts

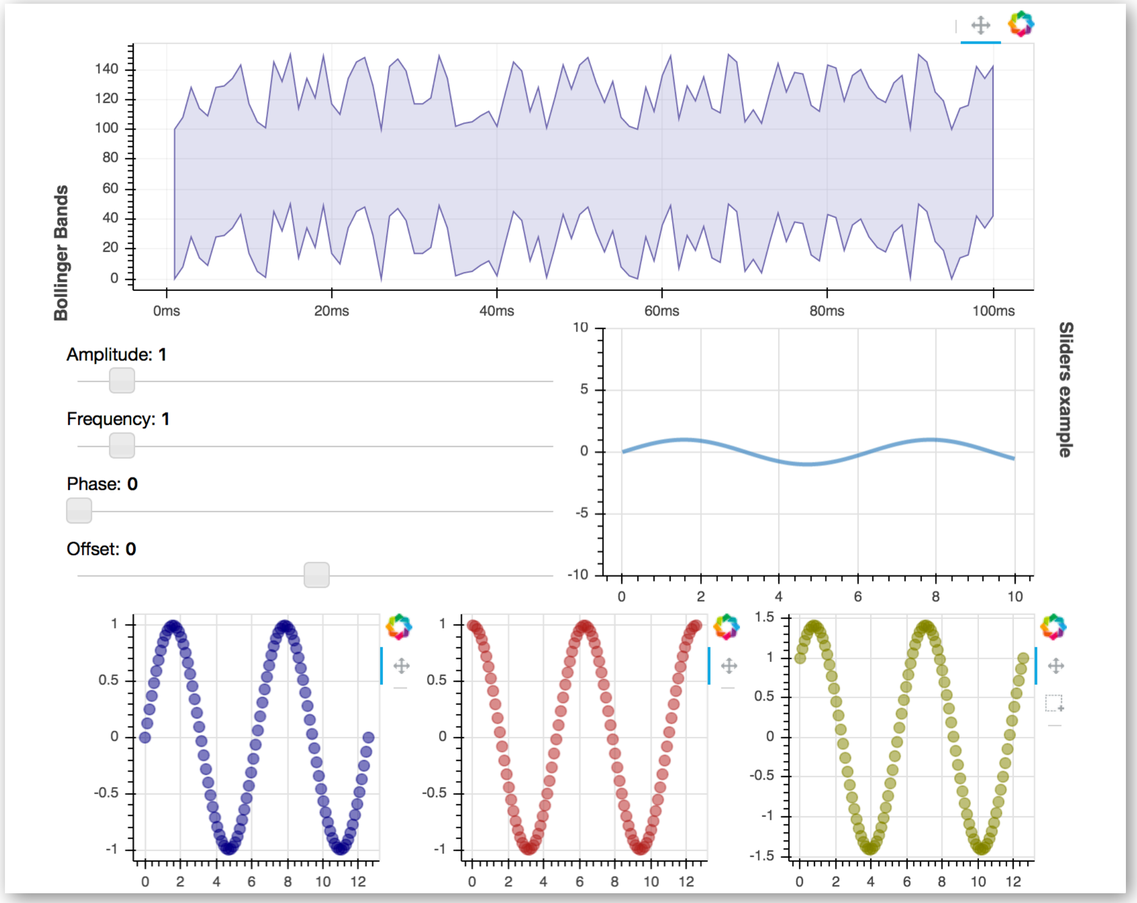


Bokeh offers "rows and columns" layouts

```
page = layout([
    bollinger(),
    slider(),
    linked_panning(),
], sizing_mode='stretch_both')

show(page)
```

Built-in Layouts



Bokeh offers "rows and columns" layouts

```
page = layout([
    bollinger(),
    slider(),
    linked_panning(),
], sizing_mode='stretch_both')

show(page)
```

Bokeh objects can be responsive

Custom Templates

Import embed function

```
{% from macros import embed %}
```

Load BokehJS resources

```
<head>
    <meta charset="utf-8">
    <title>Face Detection</title>
    {{ bokeh_css | indent(8) }}
    {{ bokeh_js | indent(8) }}
</head>
```

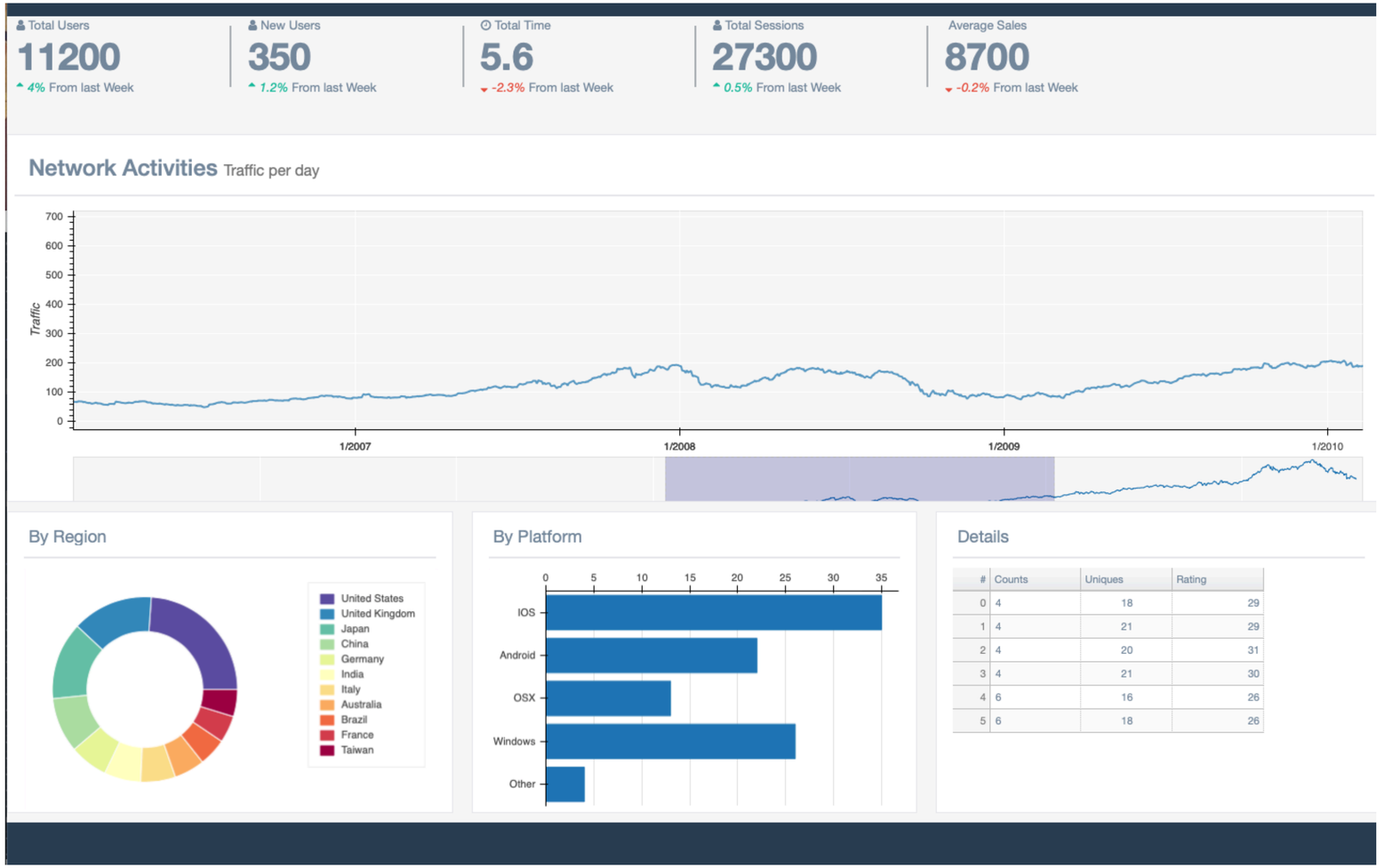
Embed Bokeh objects

```
<div class="col-lg-8">
    <div class="my-2">
        {{ embed(roots.plot1) }}
    </div>
    <div class="my-2">
        {{ embed(roots.plot2) }}
    </div>
</div>
```

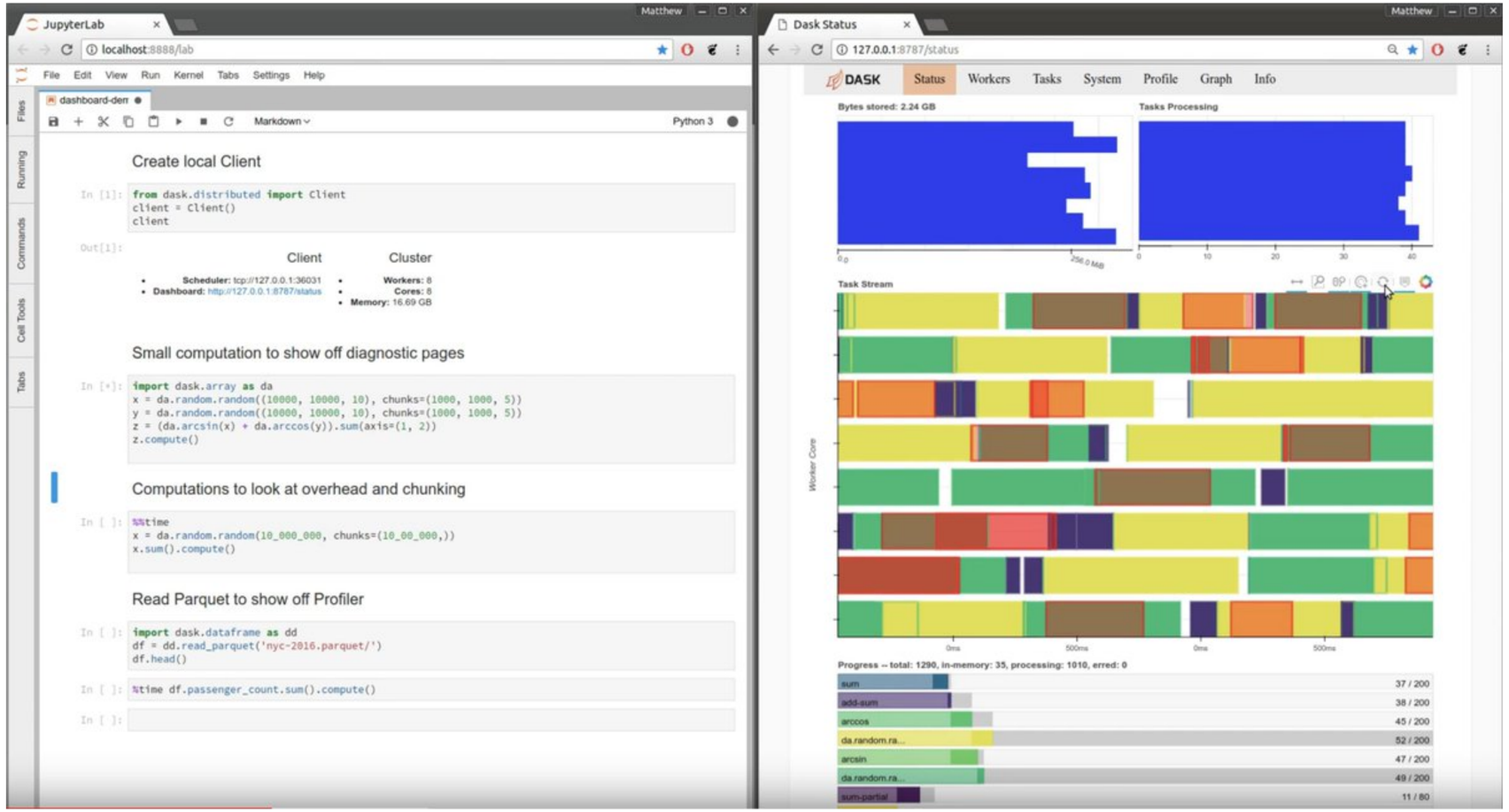
Boilerplate script

```
{{ plot_script }}
```

Custom Templates



Dask UI Uses CSS Grid



http://examples.dask.org

Custom Extensions

JavaScript implementation

```
_ = require "underscore"
$ = require "jquery"
p = require "core/properties"
LayoutDOM = require "models/layouts/layout_dom"

class CustomView extends LayoutDOM.View

  initialize: (options) ->
    super(options)
    @render()
    @listenTo(@model.slider, 'change', () => @render())

  render: () ->
    @$el.html("<h1>#{ @model.text }: #{ @model.slider.value }</h1>")
    @$('h1').css({ 'color': '#686d8e', 'background-color': '#2a3153' })

class Custom extends LayoutDOM.Model
  default_view: CustomView
  type: "Custom"

  @define {
    text:   [ p.String ]
    slider: [ p.Any    ]
  }

module.exports =
  Model: Custom
  View: CustomView
```

Python class interface

```
from bokeh.core.properties import String, Instance
from bokeh.models import LayoutDOM, Slider

class Custom(LayoutDOM):

    text = String(default="Custom text")

    range = Instance(Slider)
```

Normal usage

```
from bokeh.io import show
from bokeh.layouts import column
from bokeh.models import Slider

from custom import Custom

slider = Slider(start=0, end=10, step=0.1, value=0, title="value")

custom = Custom(text="Special Slider Display", slider=slider)

layout = column(slider, custom)

show(layout)
```

http://bokeh.pydata.org/en/latest/docs/user_guide/extensions.html

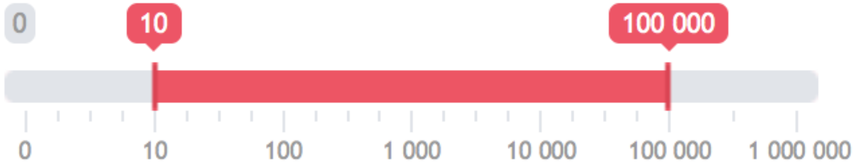
Custom Extensions

Add capability for…

Custom Extensions

Add capability for…

Wrapping different widgets

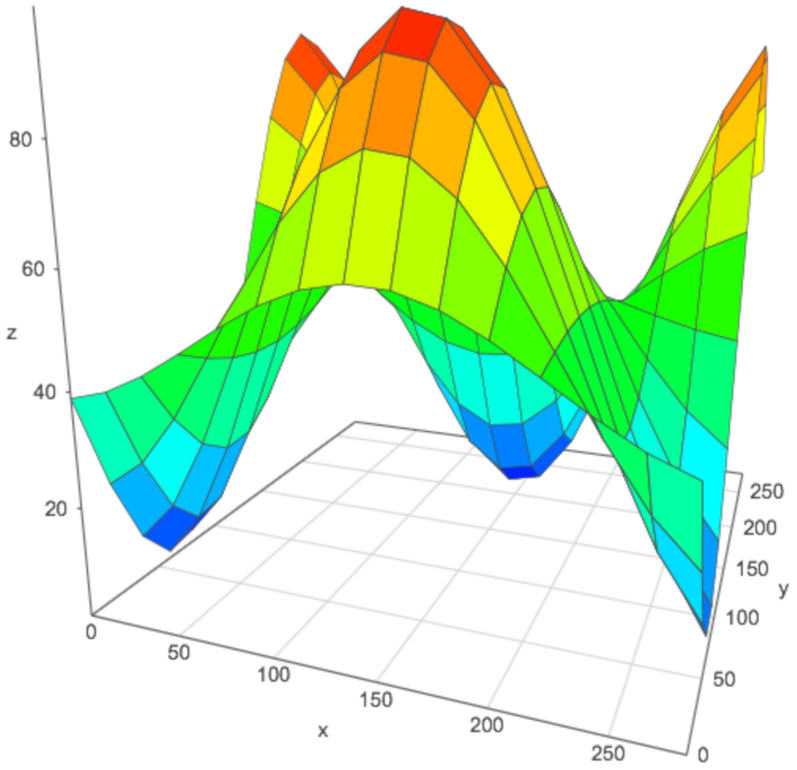


Custom Extensions

Add capability for…

Wrapping different widgets

Adapting 3D JavaScript libraries



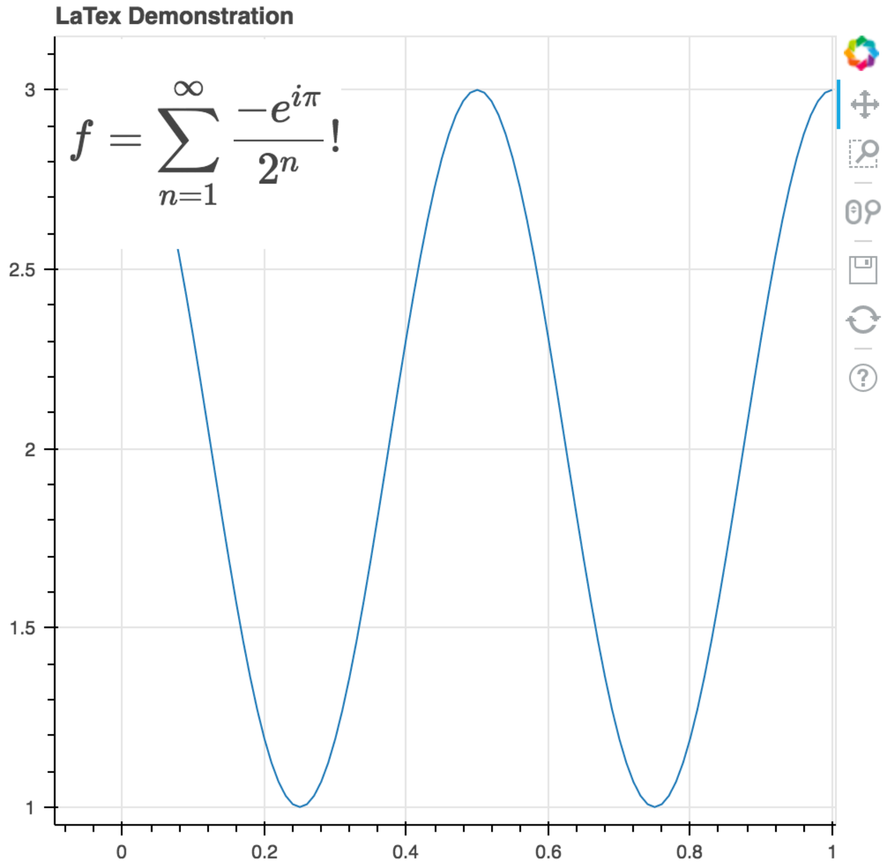
Custom Extensions

Add capability for…

Wrapping different widgets

Adapting 3D JavaScript libraries

Adding LaTeX labels



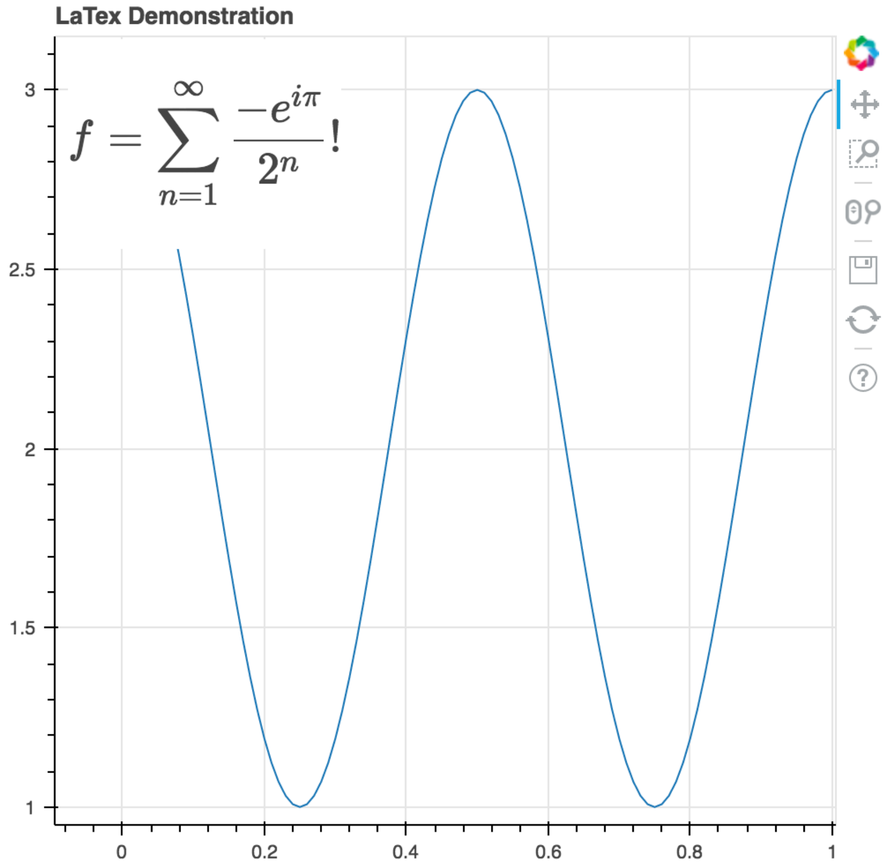
Custom Extensions

Add capability for…

Wrapping different widgets

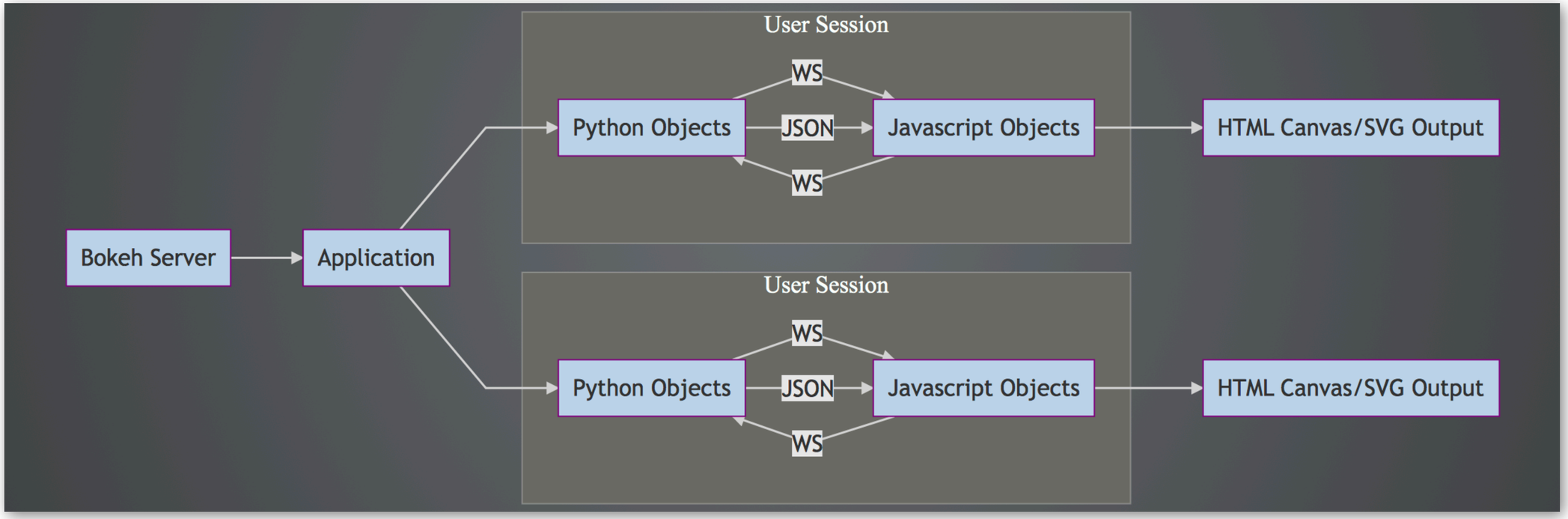
Adapting 3D JavaScript libraries

Adding LaTeX labels



Community doesn't have to wait!

Bokeh Server Applications



Bokeh Server automatically keeps Python and JavaScript in sync

Bokeh Server Applications

```
def update(attr, old, new):
    plot.title.text = new

x = Select(title='X-Axis', value='mpg', options=columns)
x.on_change('value', update)
```

Python callbacks can execute on property changes (and JS will react too)

Bokeh Server Applications

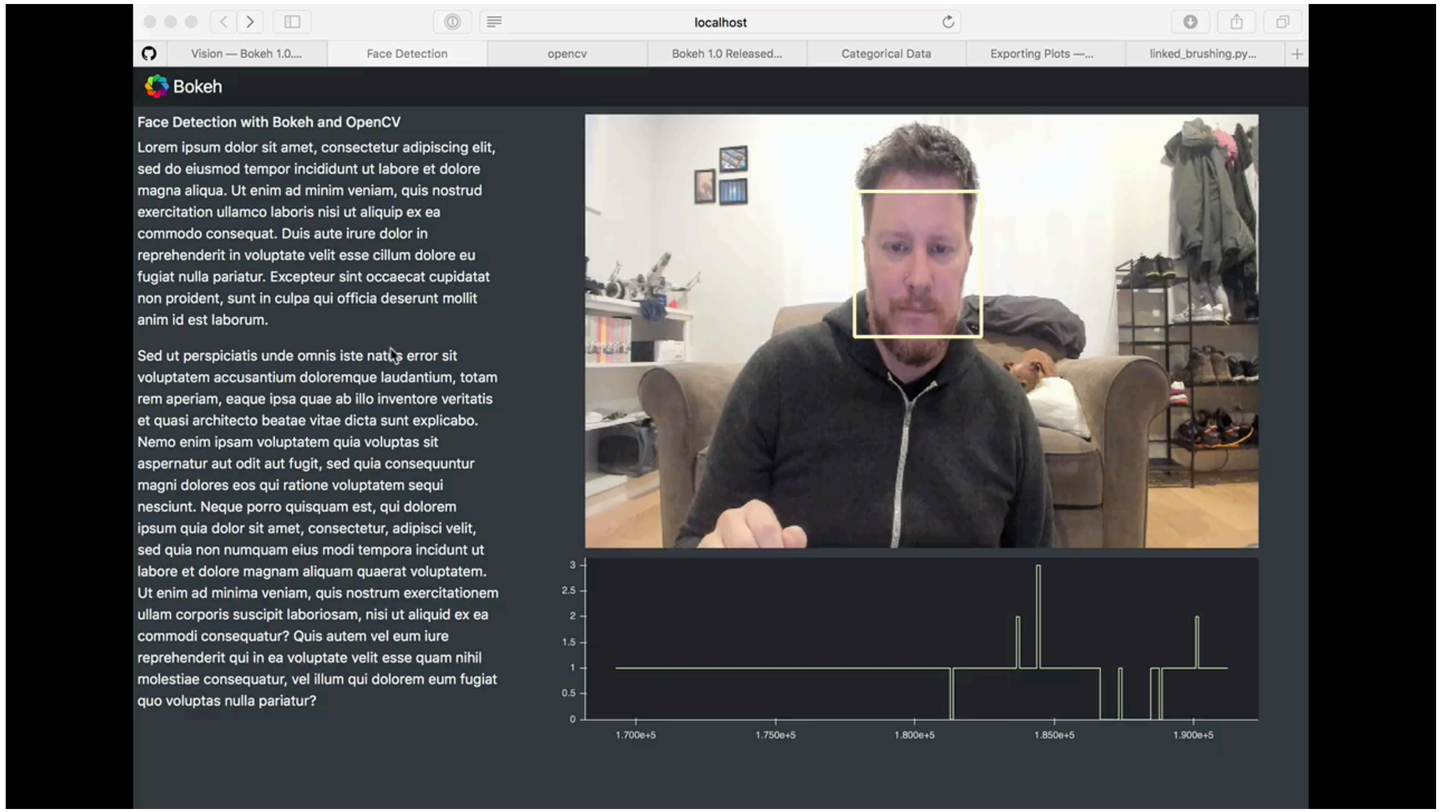
```
def stream_image():
    ...
    source.data["image"] = [image]

doc.add_periodic_callback(acquire_image, 100)
```

Document level callbacks can perform server-driven updates

An Example

Goal



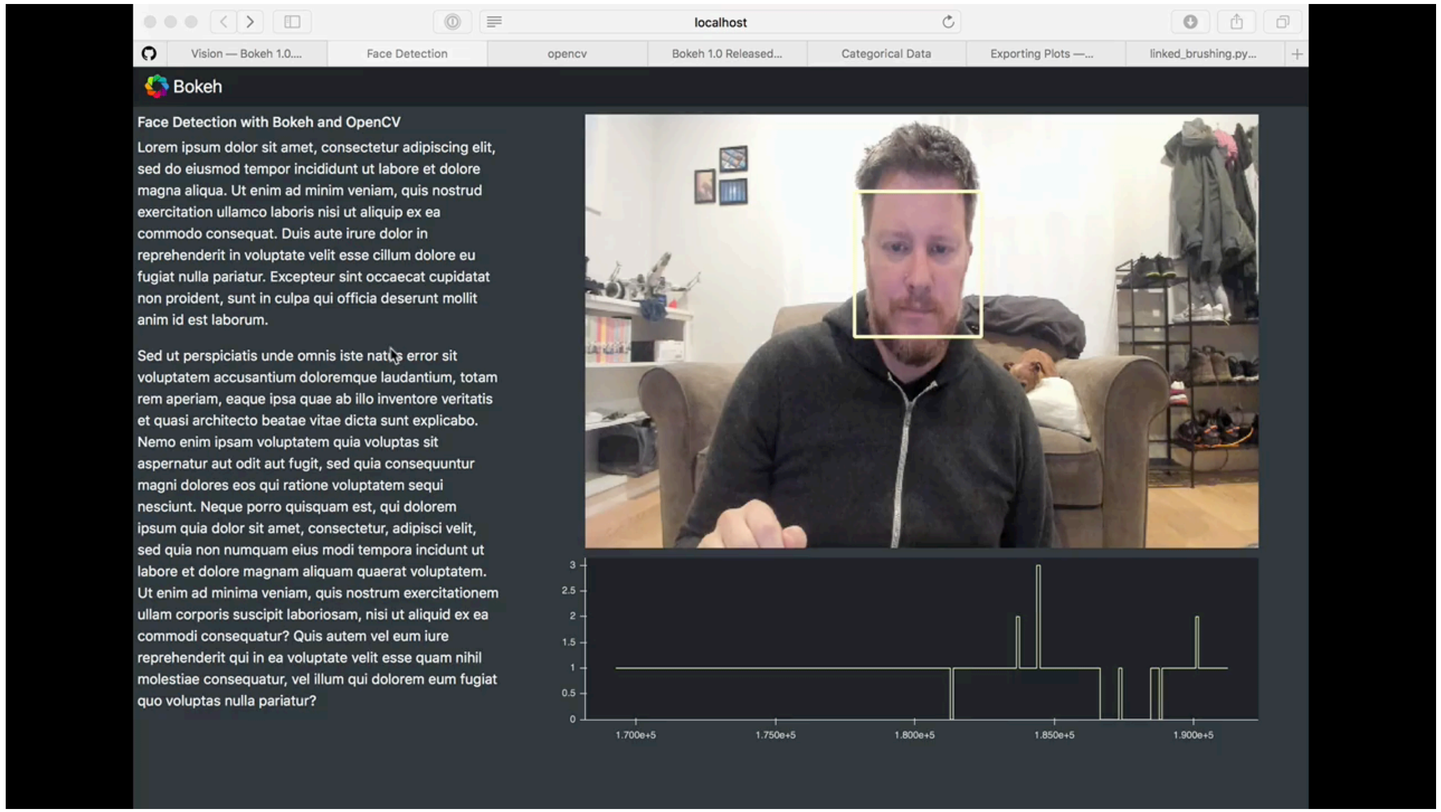
Demonstrates:

Example of streaming API

Connects to PyData tools

Custom layout for nice look

Goal



Demonstrates:

Example of streaming API

Connects to PyData tools

Custom layout for nice look

The Python Bits

```python
from datetime import datetime as dt
from os.path import dirname, join

import cv2

from bokeh.io import curdoc
from bokeh.models import ColumnDataSource
from bokeh.plotting import figure
from bokeh.sampledata.haar_cascade import frontalface_default_path

CAMERA_WIDTH, CAMERA_HEIGHT = (1280, 780)

# try an external camera device first, fall back to default camera
video_capture = cv2.VideoCapture(1)
if not video_capture.isOpened():
    video_capture = cv2.VideoCapture(0)
video_capture.set(cv2.CAP_PROP_FRAME_WIDTH, CAMERA_WIDTH)
video_capture.set(cv2.CAP_PROP_FRAME_HEIGHT, CAMERA_HEIGHT)

# train our cascade classifier
from bokeh.sampledata.haar_cascade import frontalface_default_path
face_cascade = cv2.CascadeClassifier(frontalface_default_path)

img_plot = figure(plot_width=CAMERA_WIDTH//2, plot_height=CAMERA_HEIGHT//2,
                  x_range=(0, CAMERA_WIDTH), y_range=(0, CAMERA_HEIGHT),
                  x_axis_type=None, y_axis_type=None,
                  tools="", toolbar_location=None, name="image")

image_source = ColumnDataSource(dict(image=[]))
img_plot.image_rgba('image', x=0, y=0, dw=CAMERA_WIDTH, dh=CAMERA_HEIGHT,
                    source=image_source)

rect_source = ColumnDataSource(dict(x=[], y=[], w=[], h=[]))
img_plot.rect('x', 'y', width='w', height='h', source=rect_source,
              fill_color=None, line_color="#fffdd0", line_width=4)

ts_plot = figure(plot_width=CAMERA_WIDTH//2, plot_height=150,
                 tools="", toolbar_location=None, name="ts")
ts_plot.y_range.start = 0
ts_plot.y_range.min_interval = 2

step_source = ColumnDataSource(dict(t=[], n=[]))
ts_plot.step('t', 'n', source=step_source, line_color="#fffdd0")
```

```python
t0 = dt.now()

empty_rects = dict(x=[], y=[], w=[], h=[])

def update():
    ret, frame = video_capture.read()

    if not ret: return

    faces_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    faces = face_cascade.detectMultiScale(faces_frame,
                                          scaleFactor=1.1,
                                          minNeighbors=5,
                                          minSize=(30, 30))

    img_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGBA)
    img_frame = img_frame.view(dtype="uint32").reshape(frame.shape[:2])
    img_frame = img_frame[::-2, ::2] # lightly decimate and invert for plotting

    if len(faces) == 0:
        rect_source.data = empty_rects
    else:
        # the faces rects origin is top left so we need to fix up
        faces = [(x+w/2, CAMERA_HEIGHT-y-h/2, w, h) for x, y, w, h in faces]
        rect_source.data = dict(zip(('x', 'y', 'w', 'h'), zip(*faces)))

    image_source.data["image"] = [img_frame]

    step_source.stream({
        't': [(dt.now() - t0).total_seconds() * 1000],
        'n': [len(faces)]
    }, rollover=200)

curdoc().add_root(img_plot)
curdoc().add_root(ts_plot)
curdoc().add_periodic_callback(update, 100)
curdoc().title = "Face Detection"
```

The Template Bits

```
{% from macros import embed %}

<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Face Detection</title>
    <link rel="stylesheet" href="app/static/css/base.css" />
    <link rel="stylesheet" href="/app/static/css/bootstrap.min.css" />
    {{ bokeh_css | indent(8) }}
    {{ bokeh_js | indent(8) }}
  </head>
  <body>
    <nav class="navbar navbar-dark">
      <a class="navbar-brand py-0" href="https://bokeh.pydata.org/en/latest/">
        <img src="app/static/images/bokeh_logo.svg" width="28" height="28" class="d-inline-block align-top" alt="">
        Bokeh
      </a>
    </nav>
    <div class="bg-dark">
      <div id="container-fluid">
        <div class="row">
          <div class="col-lg-4">
            <h6 class="m-2 text-light">
              Face Detection with Bokeh and OpenCV
            </h6>
            <p class="mx-2 text-light">
              Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis
            </p>
            <p class="mx-2 text-light">
              Sed ut perspiciatis unde omnis iste natus error sit voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa quae ab illo inventore ver
            </p>
          </div>
          <div class="col-lg-8">
            <div class="my-2">
              {{ embed(roots.image) | indent(10) }}
            </div>
            <div class="my-2">
              {{ embed(roots.ts) | indent(10) }}
            </div>
          </div>
        </div>
      </div>
      {{ plot_script | indent(8) }}
    </div>
  </body>
</html>
```

Live Demos