OPEN DATA SCIENCE CONFERENCE

#ODSC

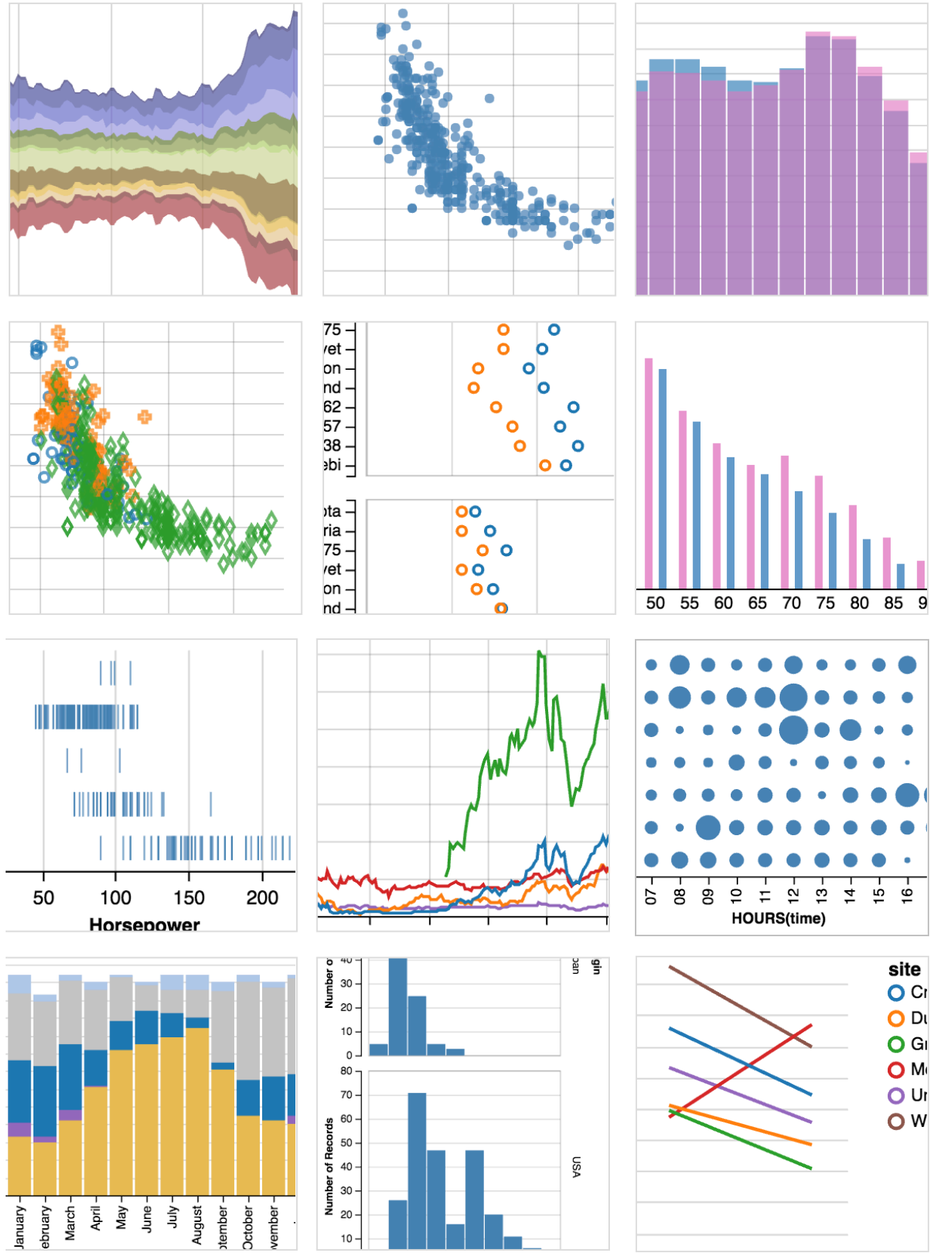
@ODSC

Boston | April 30 - May 4, 2019

Declarative Data Visualization with **Vega-Lite** and **Altair**

Kanit "Ham" Wongsuphasawat
@kanitw Apple

Arvind Satyanarayan
@arvindsatya1 MIT

Imperative

- Specify *how* should something be done
- “Put a red circle here and a blue circle here”
- Couple specification with execution

Declarative

- Specify *what* should be done
- “Map <x> and <y> to position-encodings, <z> to color-encoding”
- Separate specification from execution

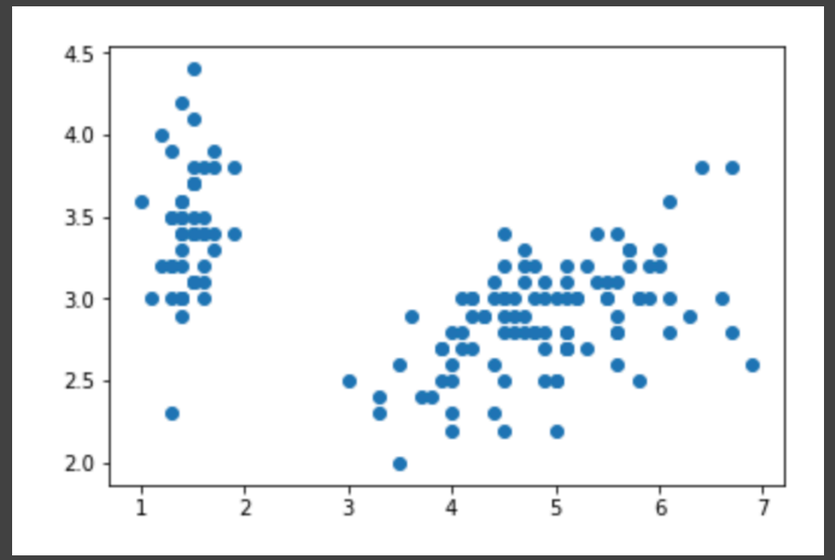
With a declarative language that directly maps code to visualization concepts, we can focus on **data** and **relationships**, rather than incidental details.

Slide adapted from https://speakerdeck.com/jakevdp/altair-tutorial-intro-pycon-2018

Scatterplot in Matplotlib

```
iris = df.read_csv(“iris.csv")

plt.scatter(iris.petalLength, iris.sepalWidth)
```

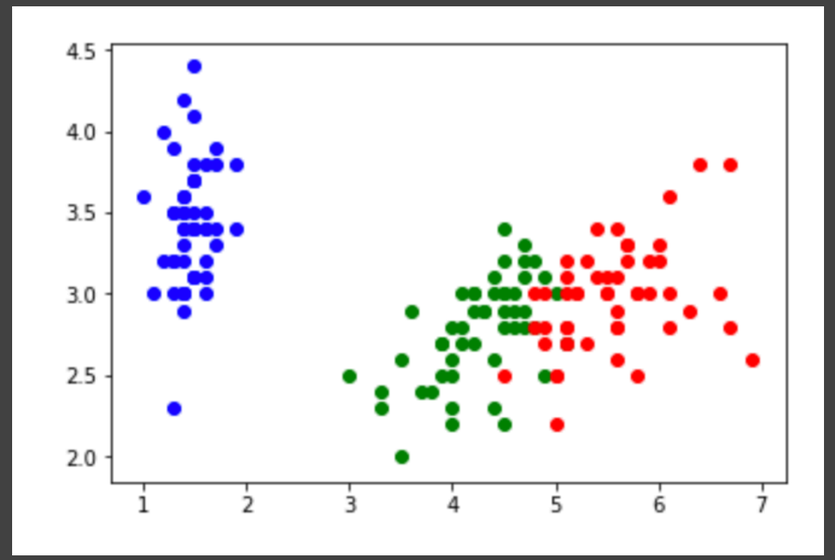
	petalLength	petalWidth	sepalLength	sepalWidth	species
0	1.4	0.2	5.1	3.5	setosa
1	1.4	0.2	4.9	3.0	setosa
2	1.3	0.2	4.7	3.2	setosa
3	1.5	0.2	4.6	3.1	setosa
4	1.4	0.2	5.0	3.6	setosa



Example Adapted from https://speakerdeck.com/jakevdp/altair-tutorial-intro-pycon-2018

Colored Scatterplot in Matplotlib

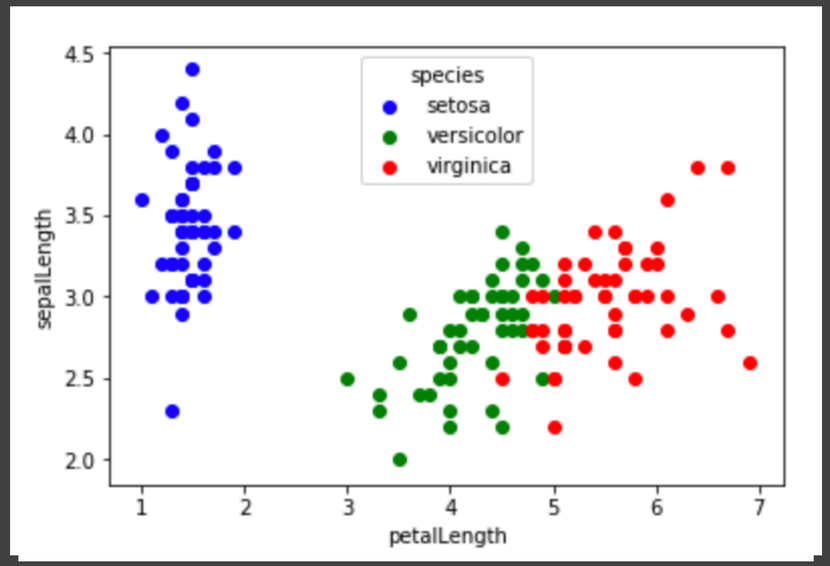
```
iris = df.read_csv(“iris.csv")

# Need to manually specify color map
color_map =  dict(zip(iris.species.unique(), ['blue', 'green', 'red']))

# Adding color involves for loop and groupby
for species, group in iris.groupby('species'):
    plt.scatter(group.petalLength, group.sepalWidth,
color=color_map[species],label=species)
```



Example Adapted from https://speakerdeck.com/jakevdp/altair-tutorial-intro-pycon-2018

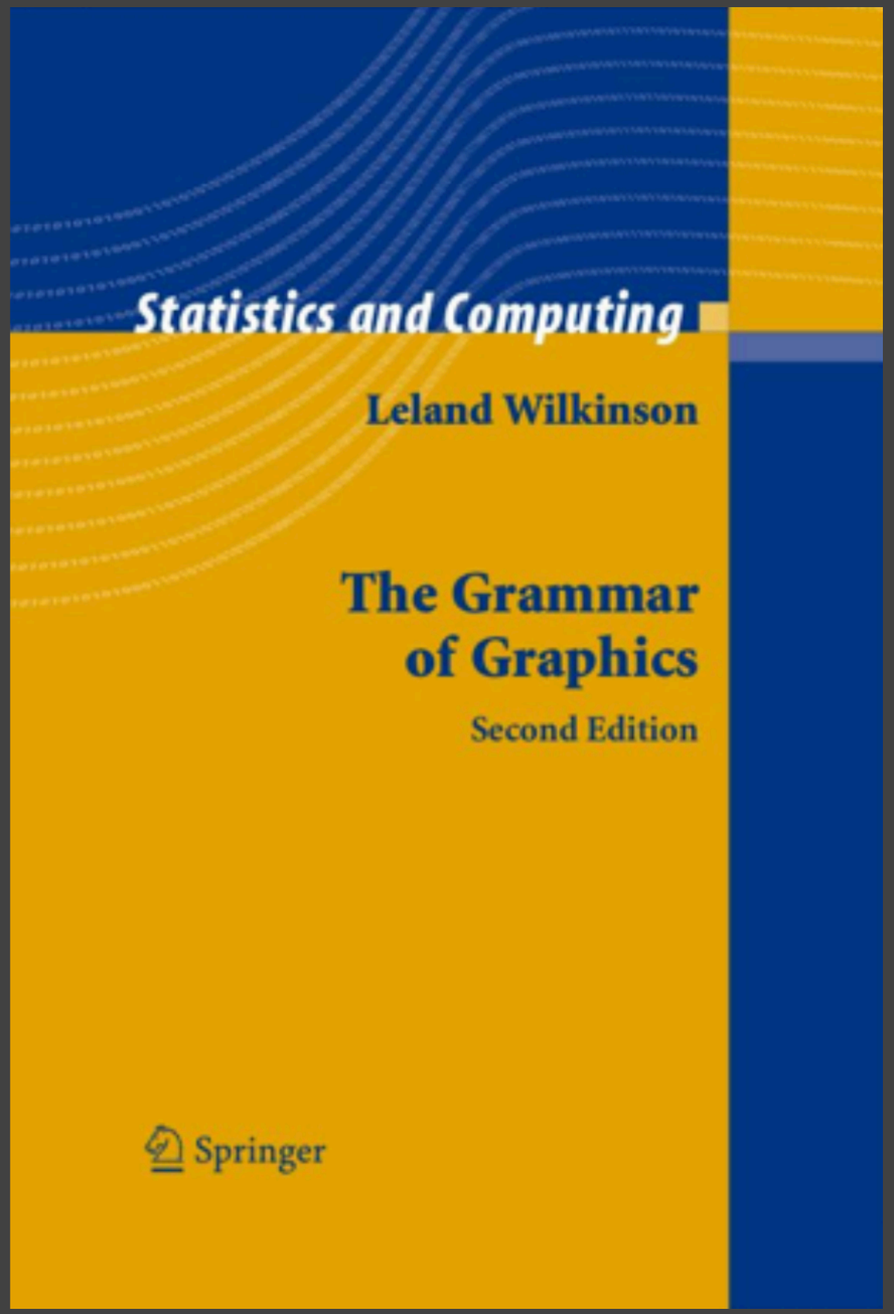
Colored Scatterplot in Matplotlib with Legend & Titles

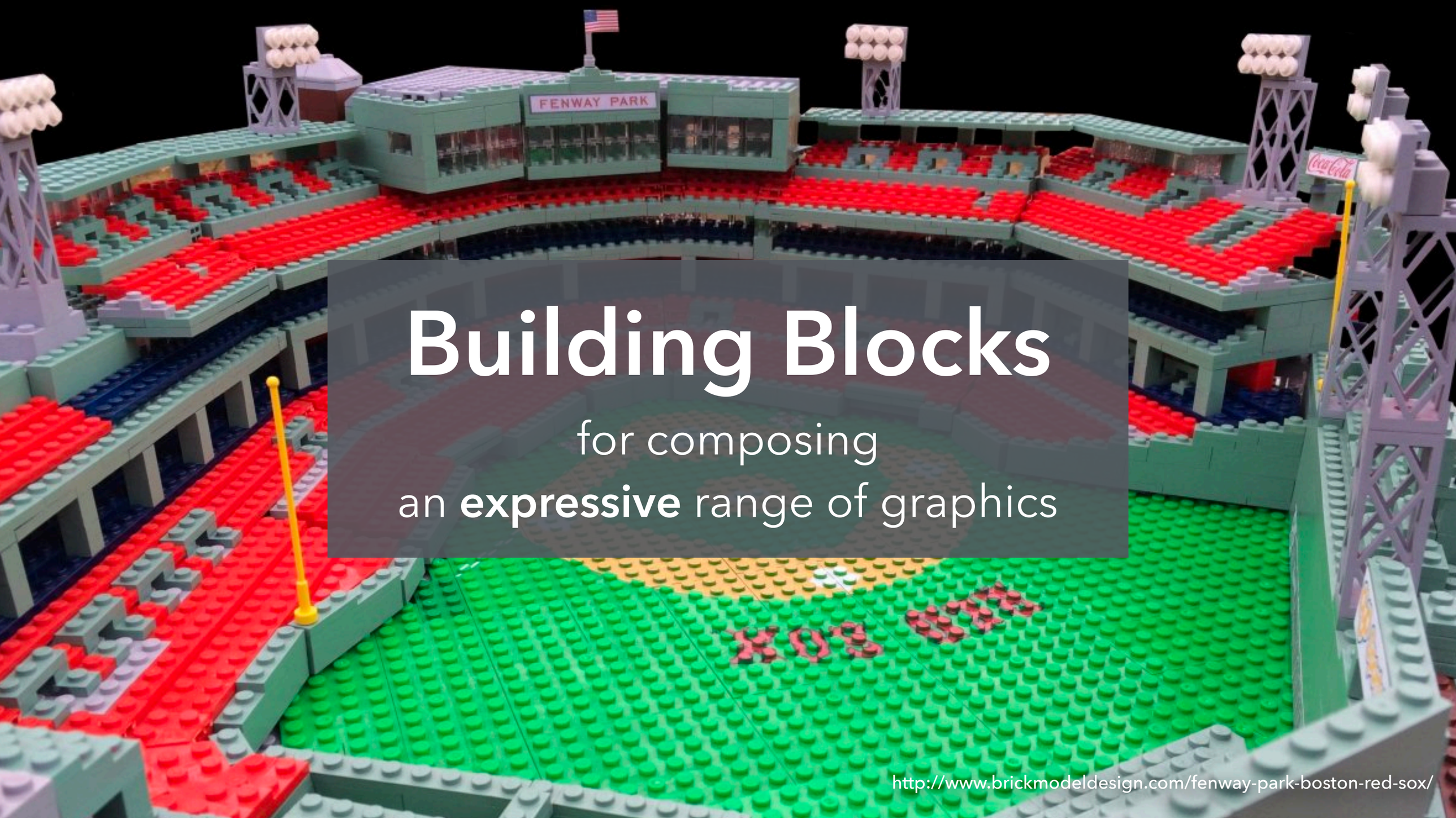
```
iris = df.read_csv(“iris.csv")

# Need to manually specify color map
color_map =  dict(zip(iris.species.unique(), ['blue', 'green', 'red']))

# Adding color involves for loop and groupby
for species, group in iris.groupby('species'):
    plt.scatter(group.petalLength, group.sepalWidth,
color=color_map[species],label=species)

# Need to manually add legend and field titles
plt.legend(frameon=True, title='species')
plt.xlabel('petalLength')
plt.ylabel('sepalLength')
```



Example Adapted from https://speakerdeck.com/jakevdp/altair-tutorial-intro-pycon-2018

Declarative Data Visualization


Statistics and Computing
Leland Wilkinson
The Grammar of Graphics
Second Edition
Springer



Building Blocks

for composing
an **expressive** range of graphics

http://www.brickmodeldesign.com/fenway-park-boston-red-sox/

Data Visualization Concepts

Data Input data source to visualize.

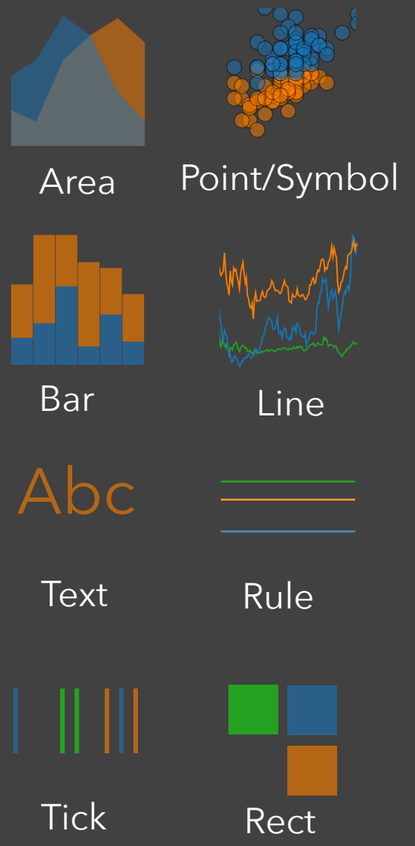
Transform Filter, aggregation, binning, etc.

Mark Data-representative graphics.

Encoding Mapping between data and mark properties.

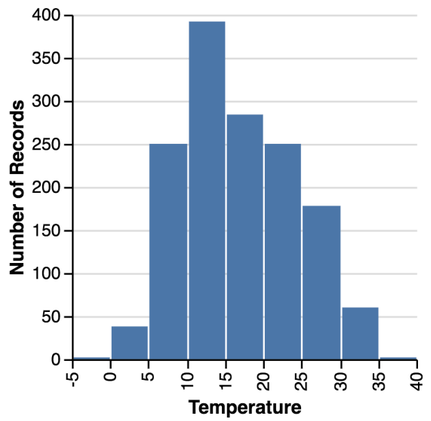
Scale Functions that map data values to visual values.

Guides Axes & legends that visualize scales.

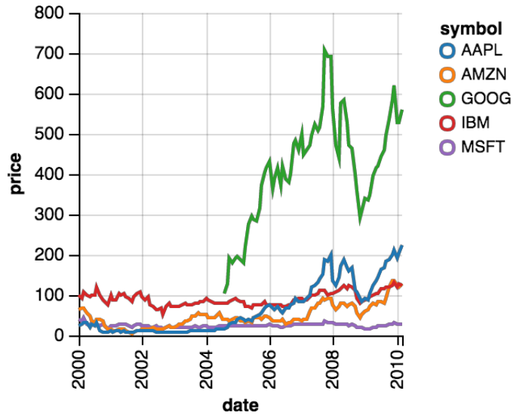


Vega-Lite: a Grammar of Graphics

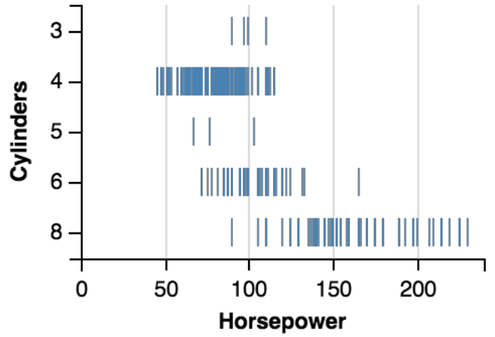
Histogram



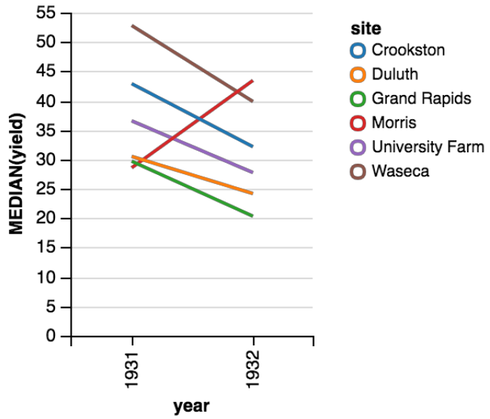
Multi-series Line Chart



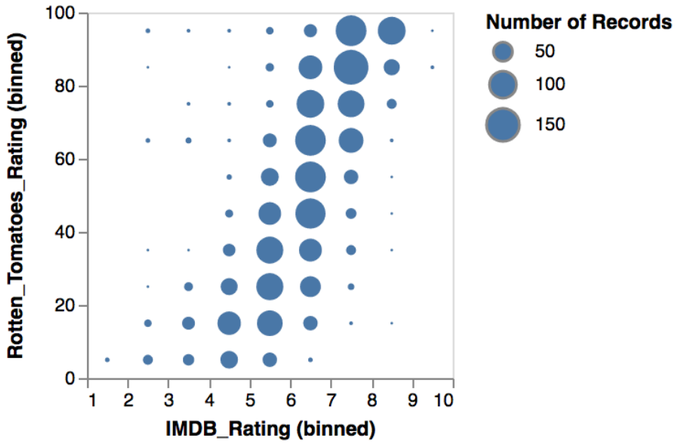
Stripplot



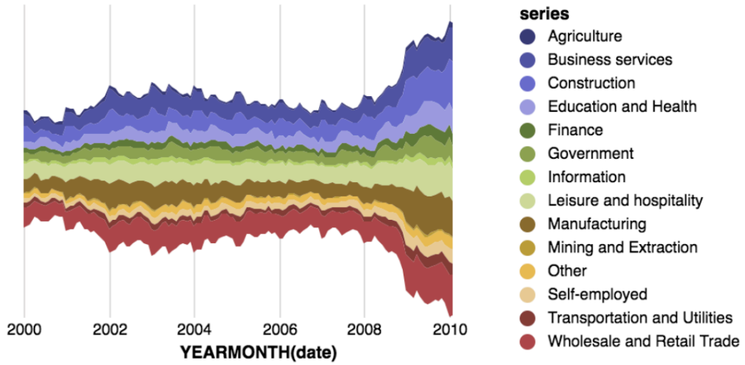
Slope Graph



Binned Scatterplot

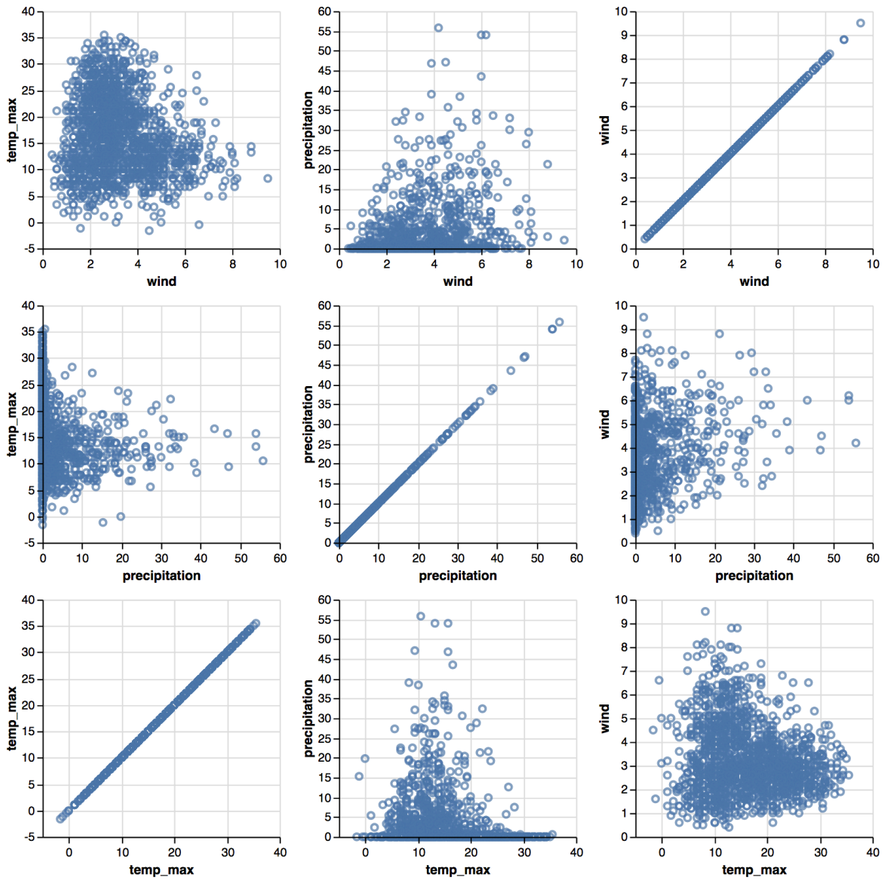


Area Chart

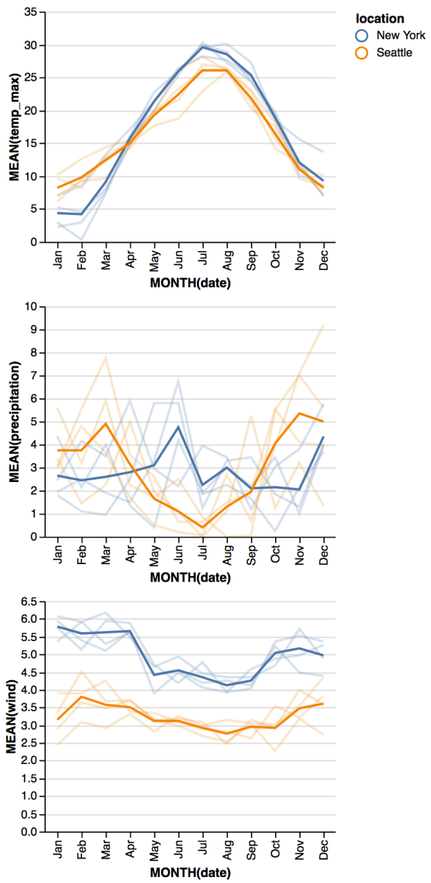


Vega-Lite: a Grammar of Multi-View Graphics

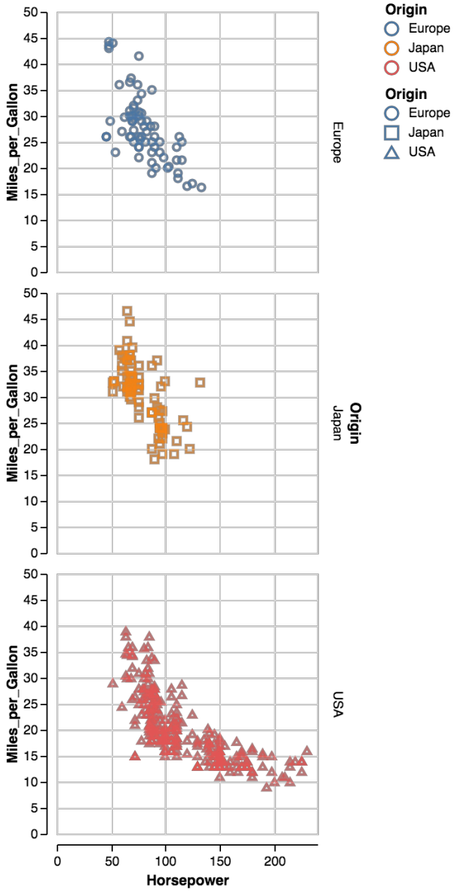
Scatterplot Matrix



Concatenated & Layered View

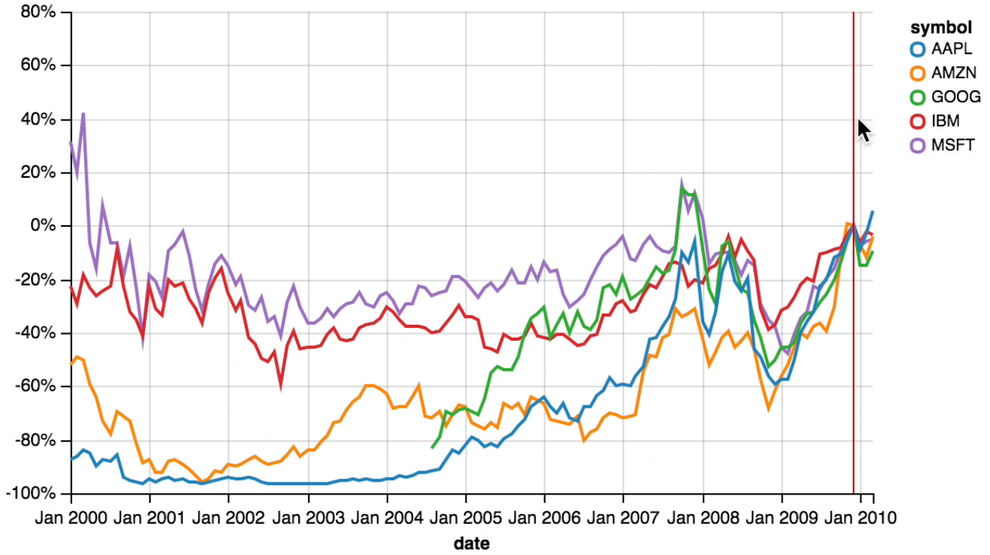


Faceted View

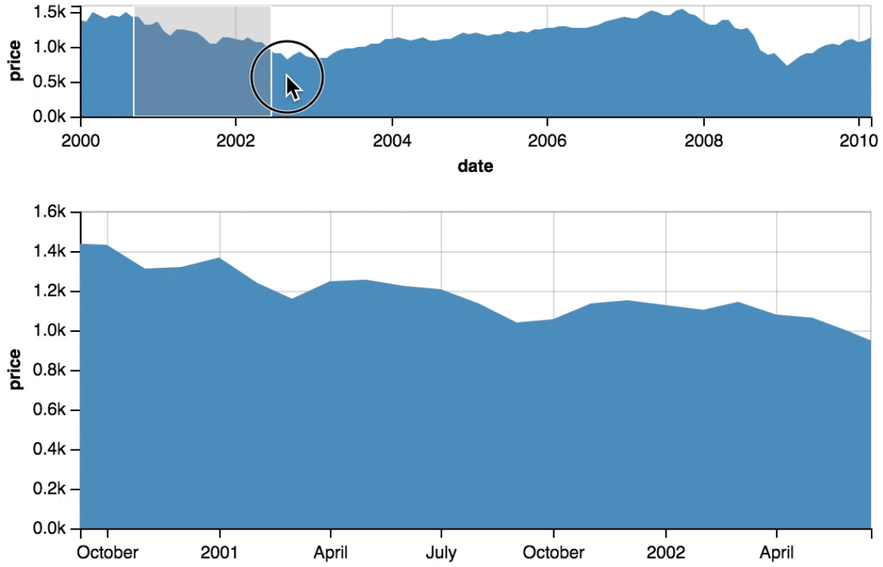


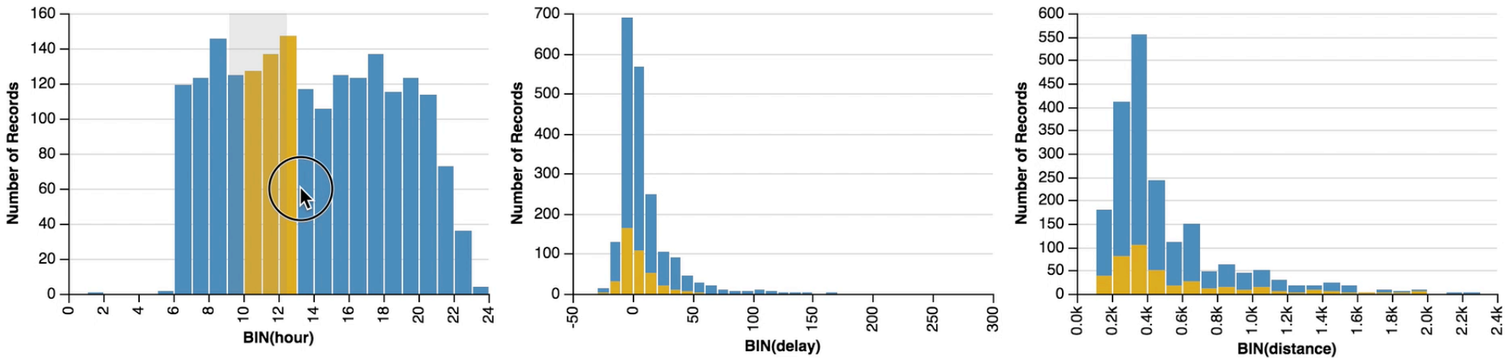
Vega-Lite: a Grammar of Interactive Multi-View Graphics

Indexed Chart



Focus+Context

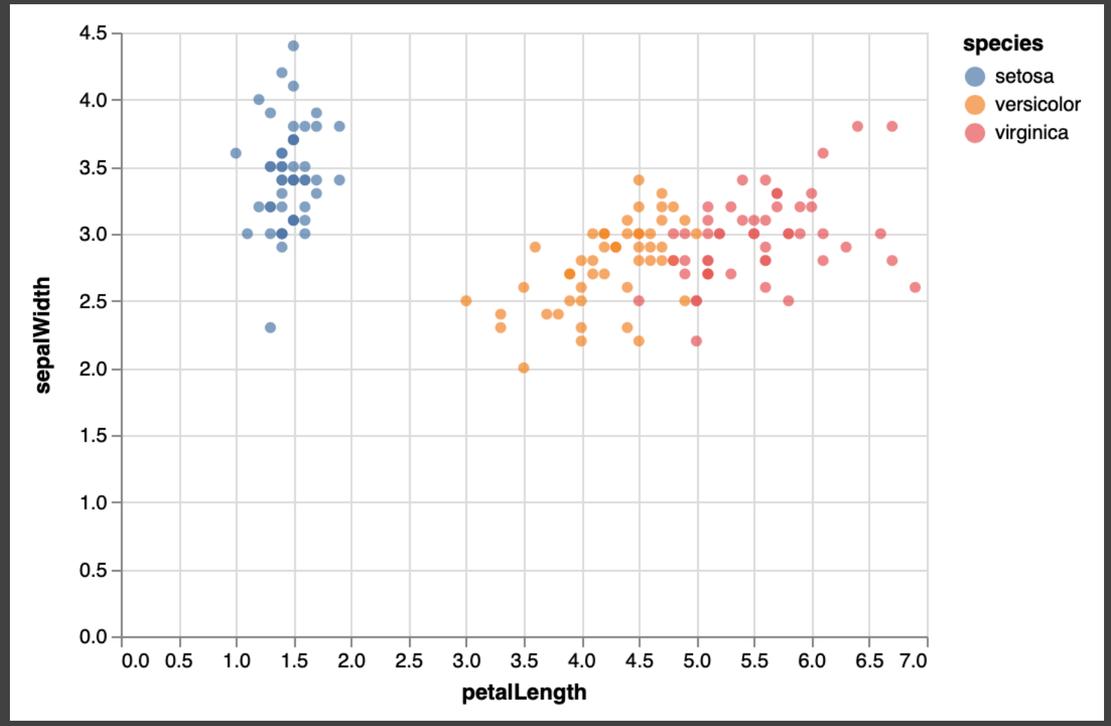




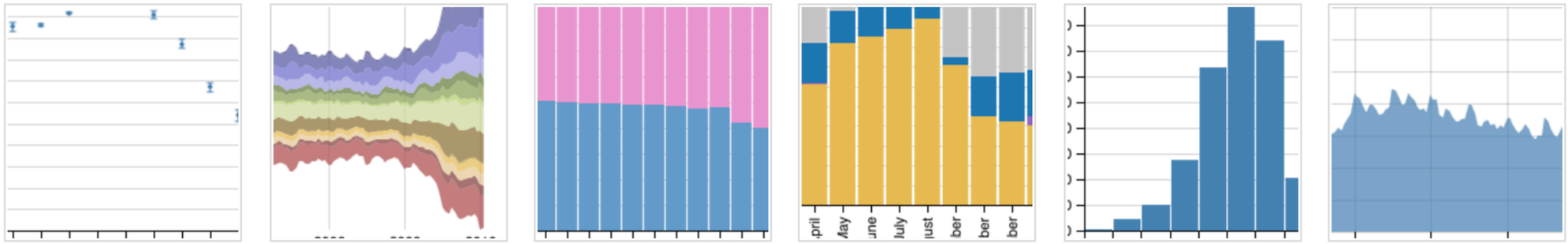
Cross-filtering

Vega-Lite: Declarative Visualization in JSON

```
{
  "data": {"url": "data/iris.json"},
  "mark": "circle",
  "encoding": {
    "x": {"field": "petalLength", "type": "quantitative"},
    "y": {"field": "sepalLength", "type": "quantitative"},
    "color": {"field": "species", "type": "nominal"}
  }
}
```



Declarative Visualization in Python



Altair is a declarative statistical visualization library for Python, based on **Vega-Lite**.

With Altair, you can spend more time understanding your data and its meaning. Altair's API is simple, friendly and consistent and built on top of the powerful **Vega-Lite** visualization grammar. This elegant simplicity produces beautiful and effective visualizations with a minimal amount of code.

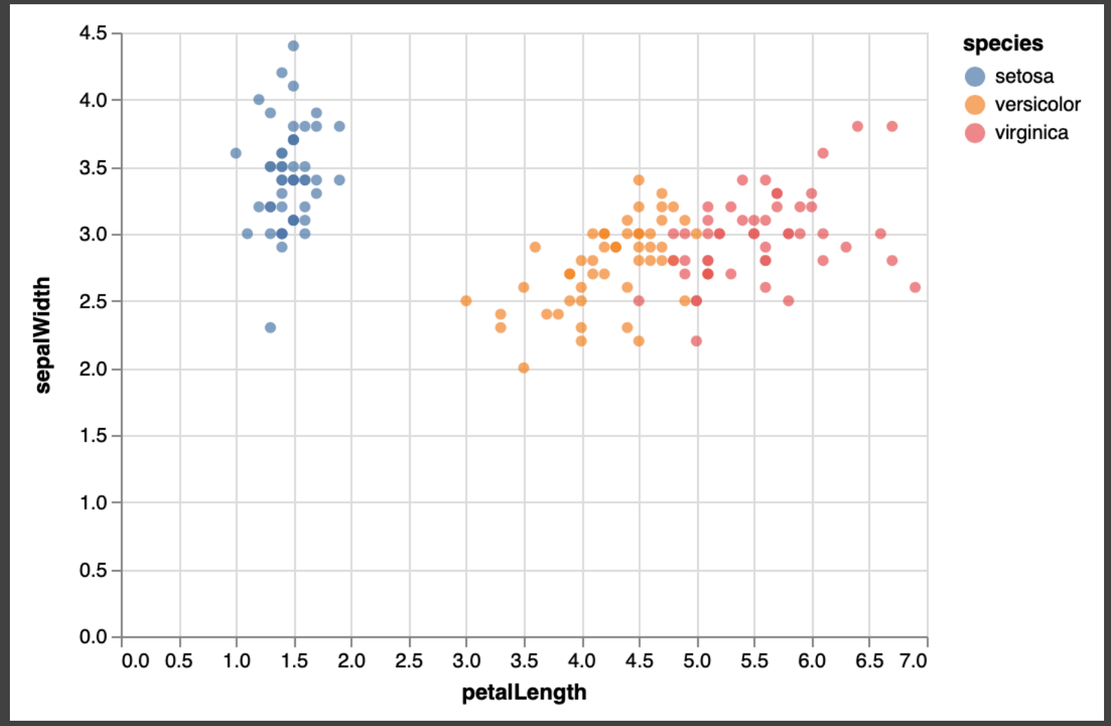
Altair: Vega-Lite in Python

Led by Jake VanderPlas and Brian Granger.

Colored Scatterplot in Vega-Lite & Altair

```
{
  "data": {"url": "data/iris.json"},
  "mark": "circle",
  "encoding": {
    "x": {"field": "petalLength", "type": "quantitative"},
    "y": {"field": "sepalLength", "type": "quantitative"},
    "color": {"field": "species", "type": "nominal"}
  }
}
```

```
alt.Chart(iris).mark_circle().encode(
    x='petalLength',
    y='sepalWidth',
    color='species'
)
```



Outline

Vega-Lite & Altair

Hands-on Altair

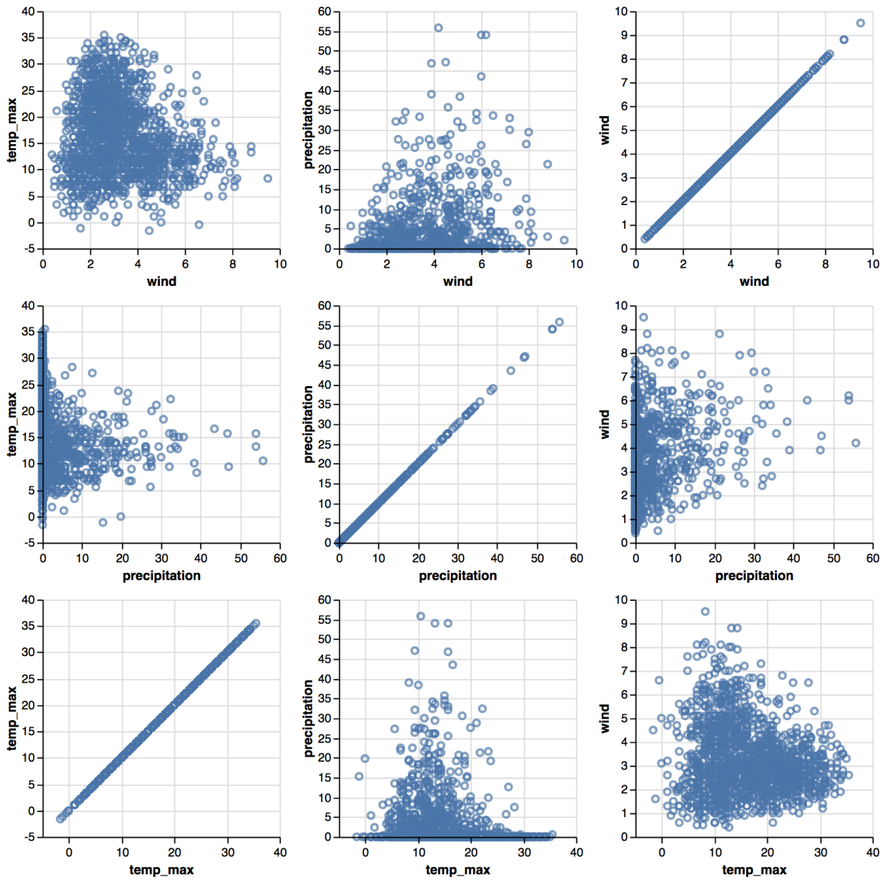
- **Single View** Specification
- **Layered** and **Multi-view** Composition
- **Interactions** with Selections and Tooltip

Additional Resources

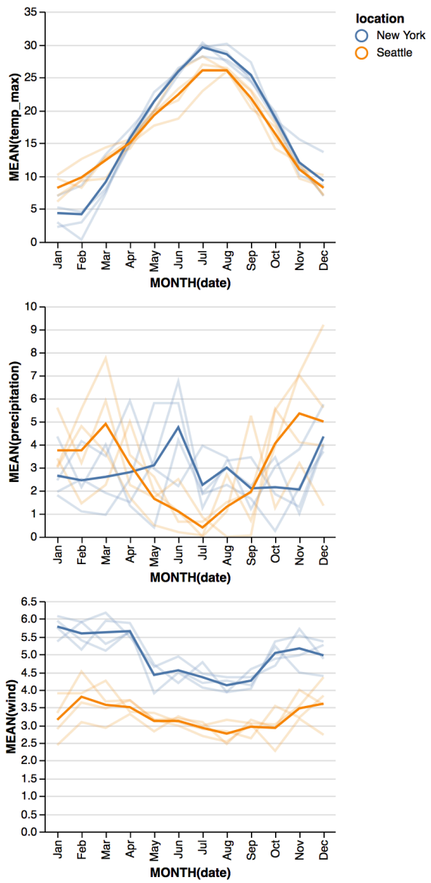
Notebook 1: Introduction

Altair: Multi-View Graphics

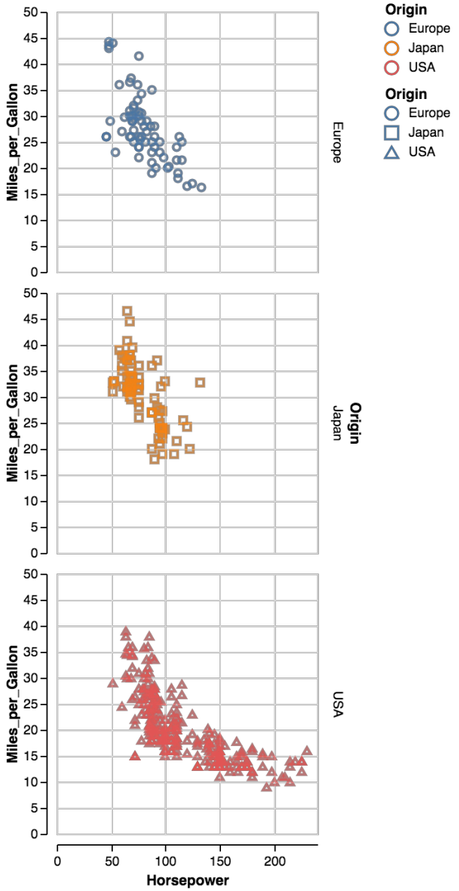
Scatterplot Matrix



Concatenated & Layered View



Faceted View

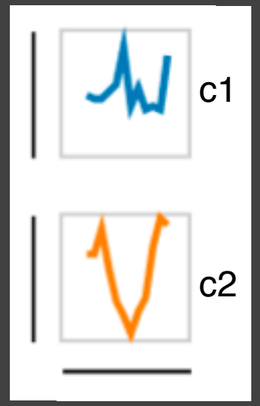


View Composition Operators

View Composition Operators

facet row: C

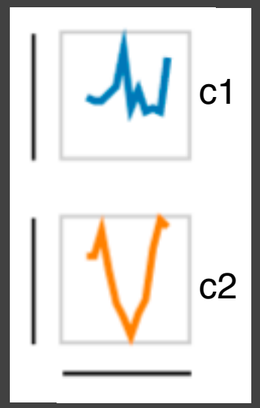
=

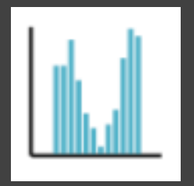


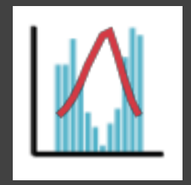
View Composition Operators

facet row: C

=



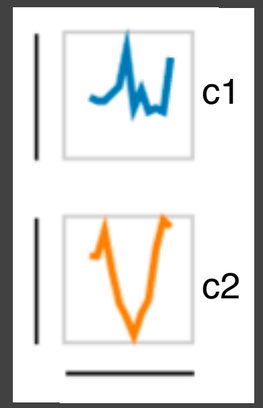
layer: [ , ] =

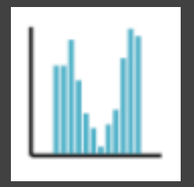
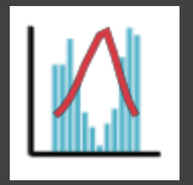


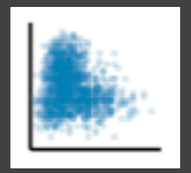
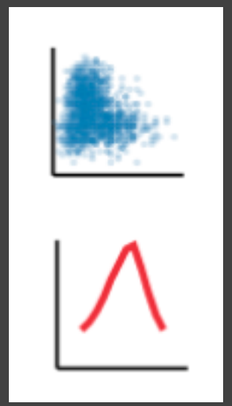
View Composition Operators

facet row: C

=



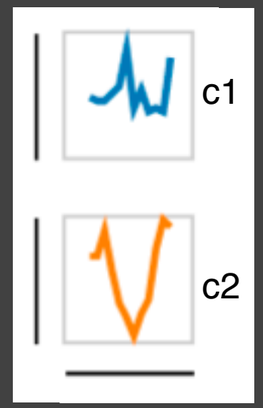
layer: [ , ] = 

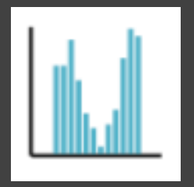
vconcat: [ , ] = 

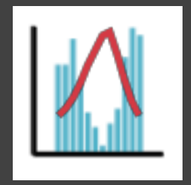
View Composition Operators

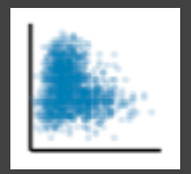
facet row: C

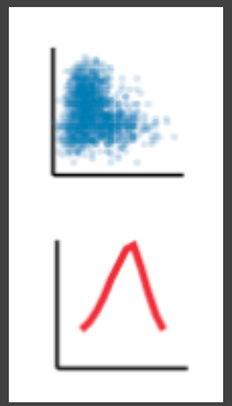
=



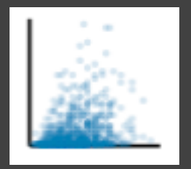
layer: [ , ] =



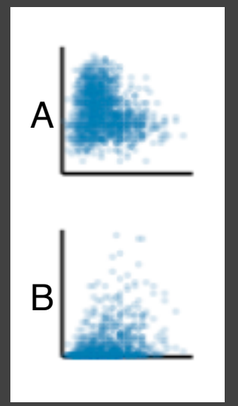
vconcat: [ , ] =



repeat row: [A,B]



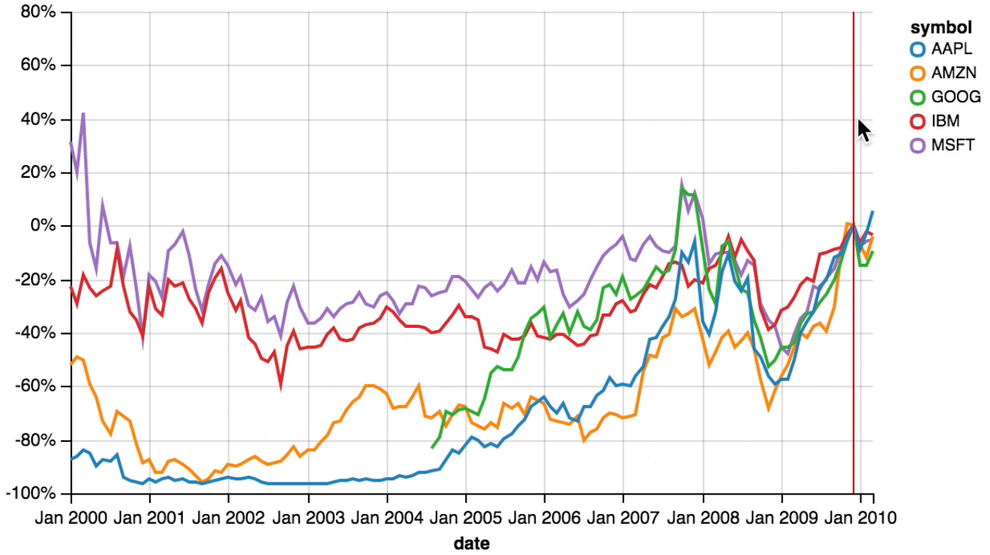
=



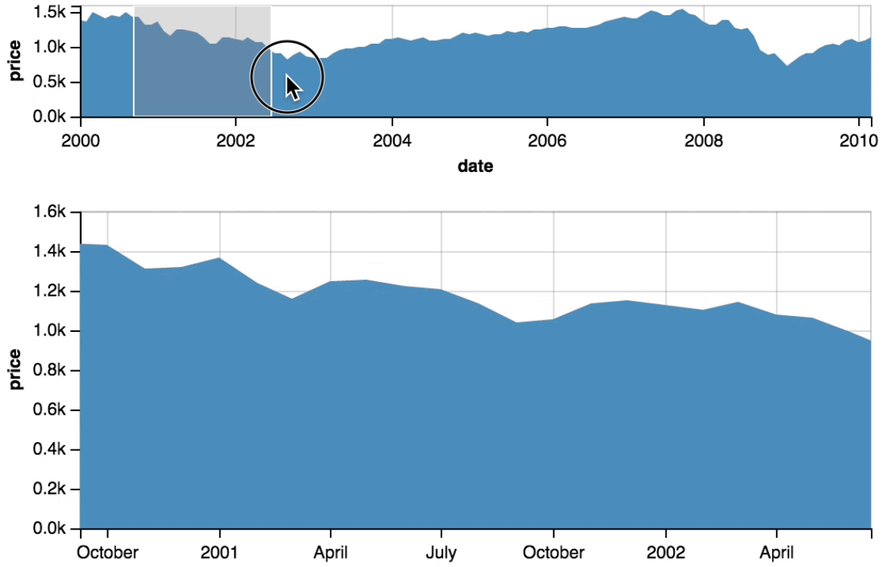
Notebook 2: View Composition

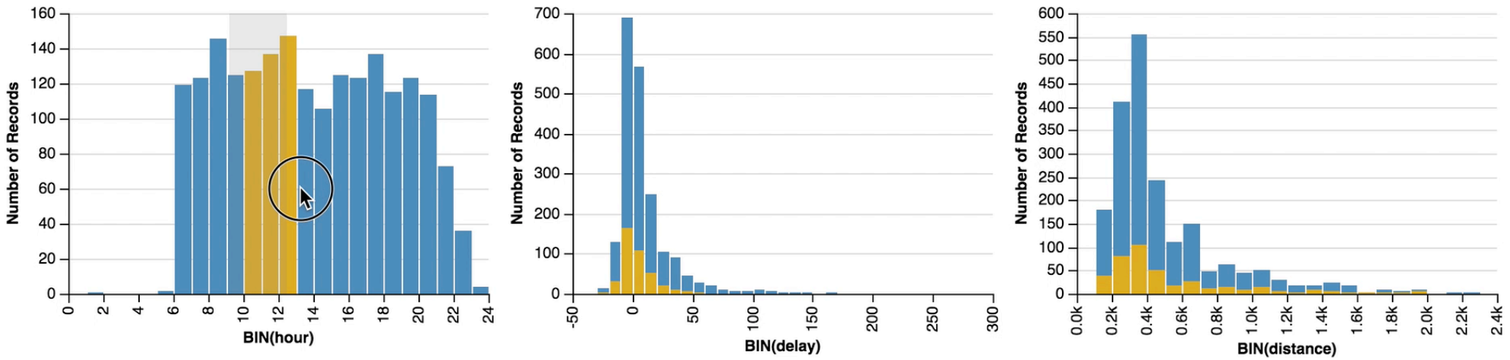
Altair: Interactive Multi-View Graphics

Indexed Chart



Focus+Context





Cross-filtering

Selections

Selections

The core interactive building blocks. Define three components:

1. Event processing – how does the interaction occur?
2. Points of interest – which marks/data points were interacted with?
3. Predicate function – what is the full set of selected marks/data points?

Notebook 3: Selection

Outline

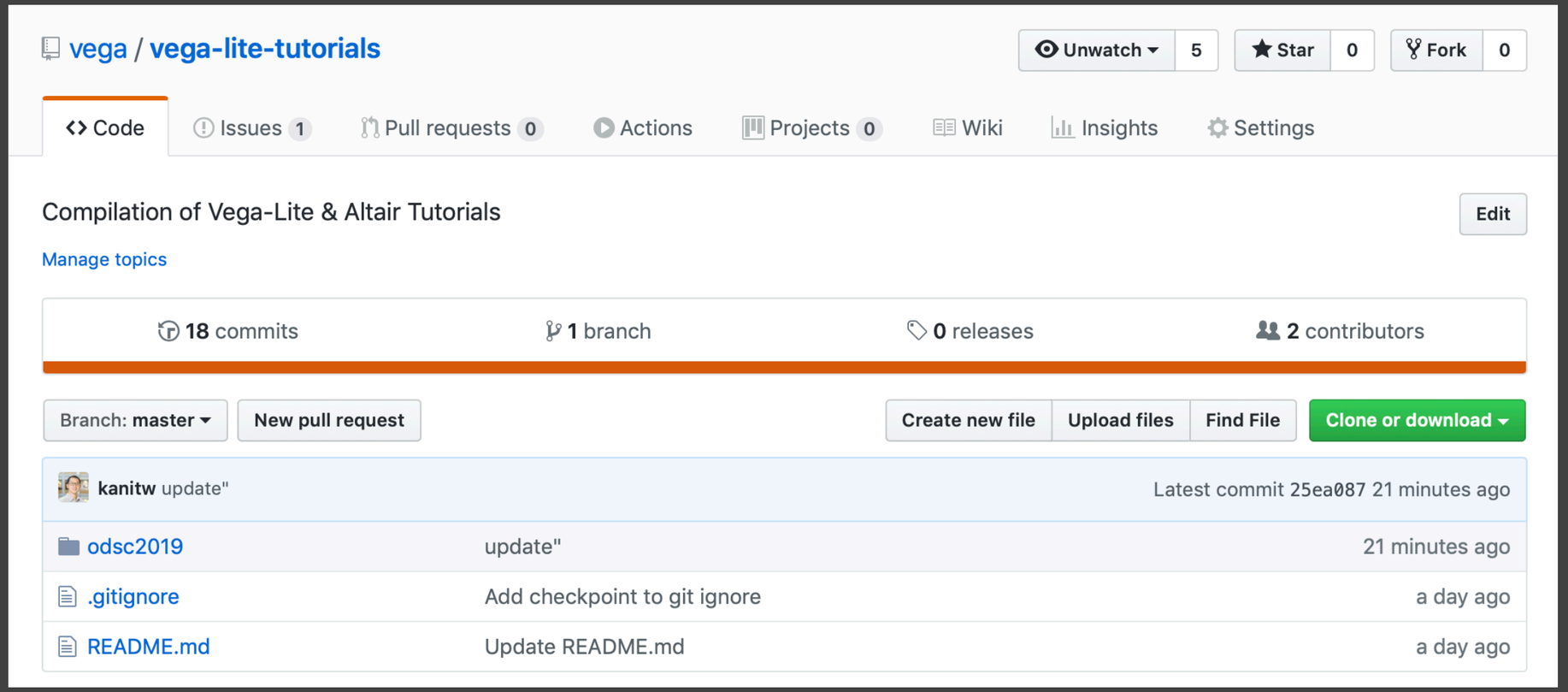
Vega-Lite & Altair

Hands-on Altair

- **Single View** Specification
- **Layered** and **Multi-view** Composition
- **Interactions** with Selections and Tooltip

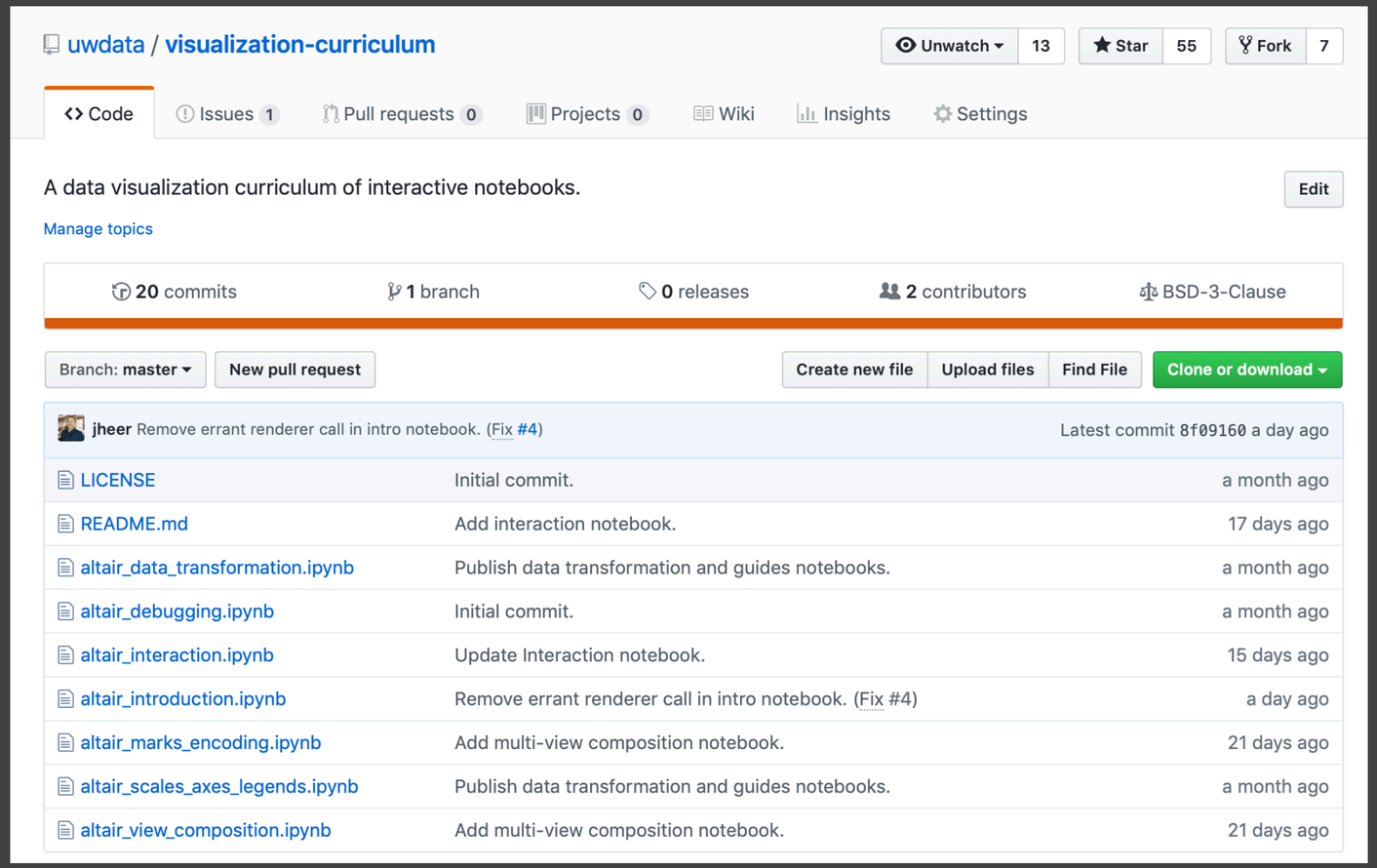
Additional Resources

Materials from the Workshop Today



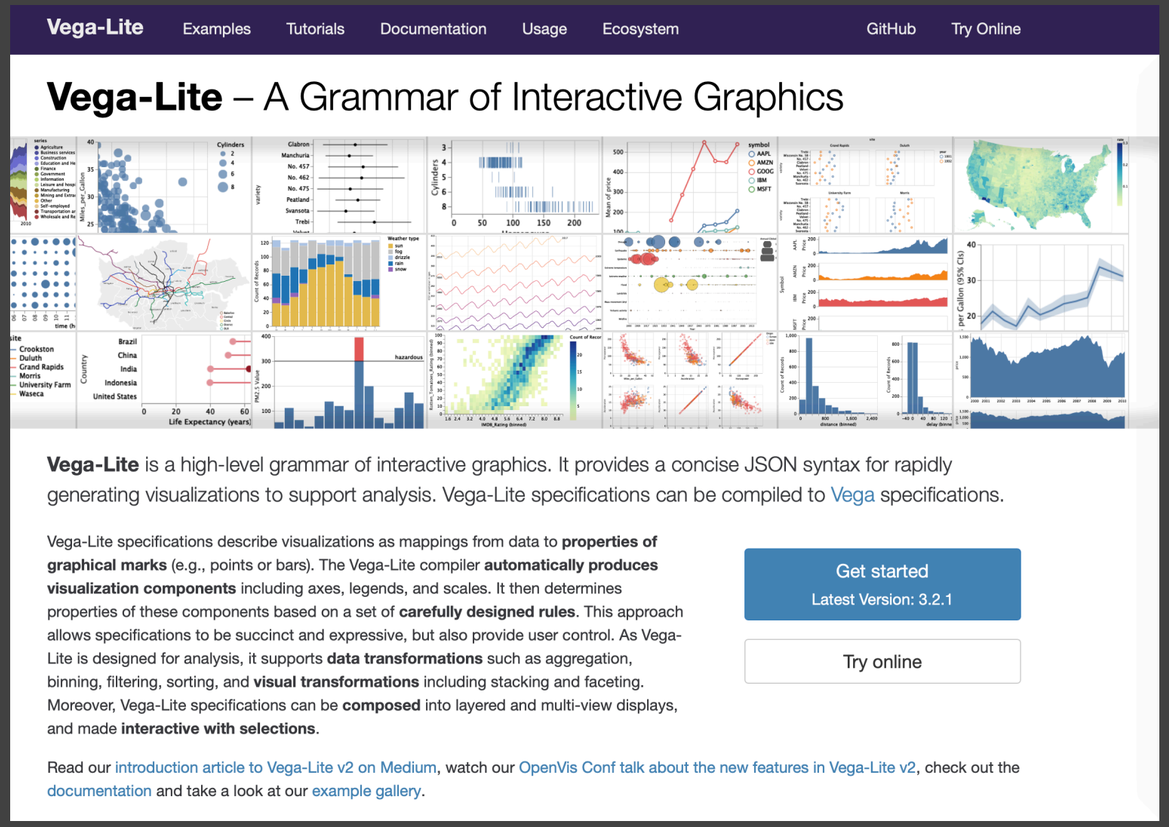
https://github.com/vega/vega-lite-tutorials

UW's Visualization Curriculum

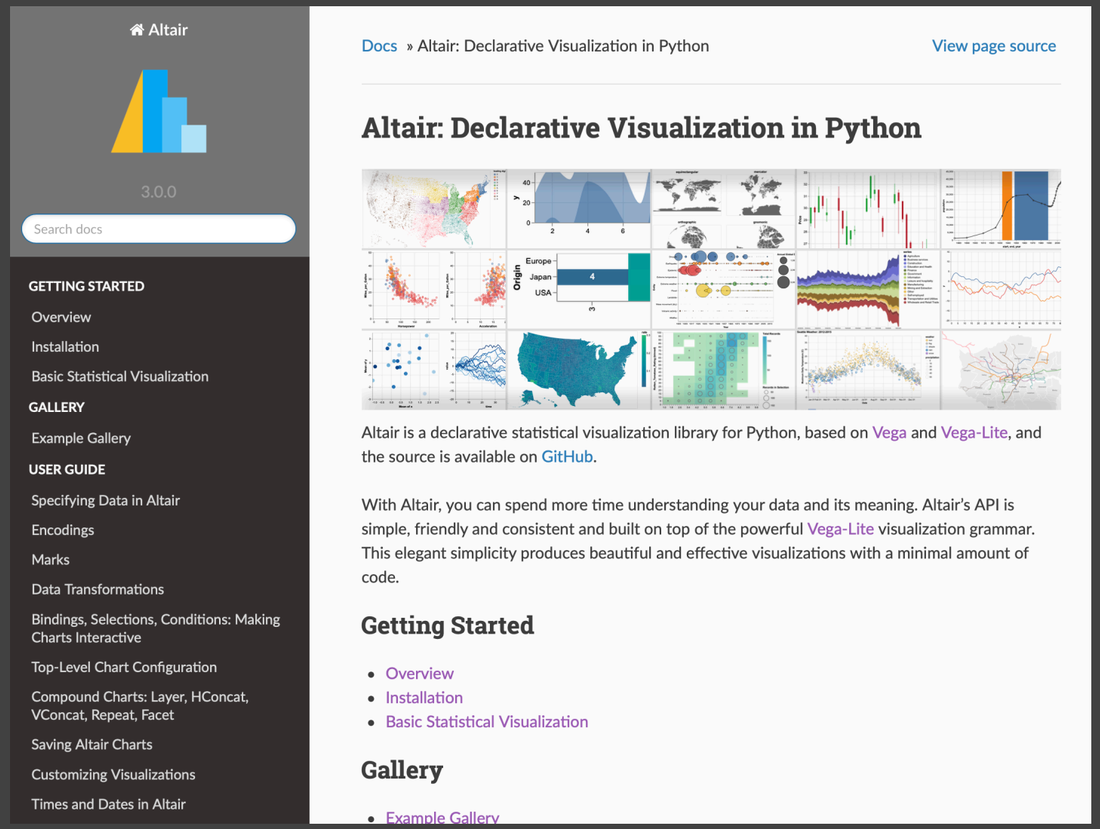


https://github.com/uwdata/visualization-curriculum

Documentations

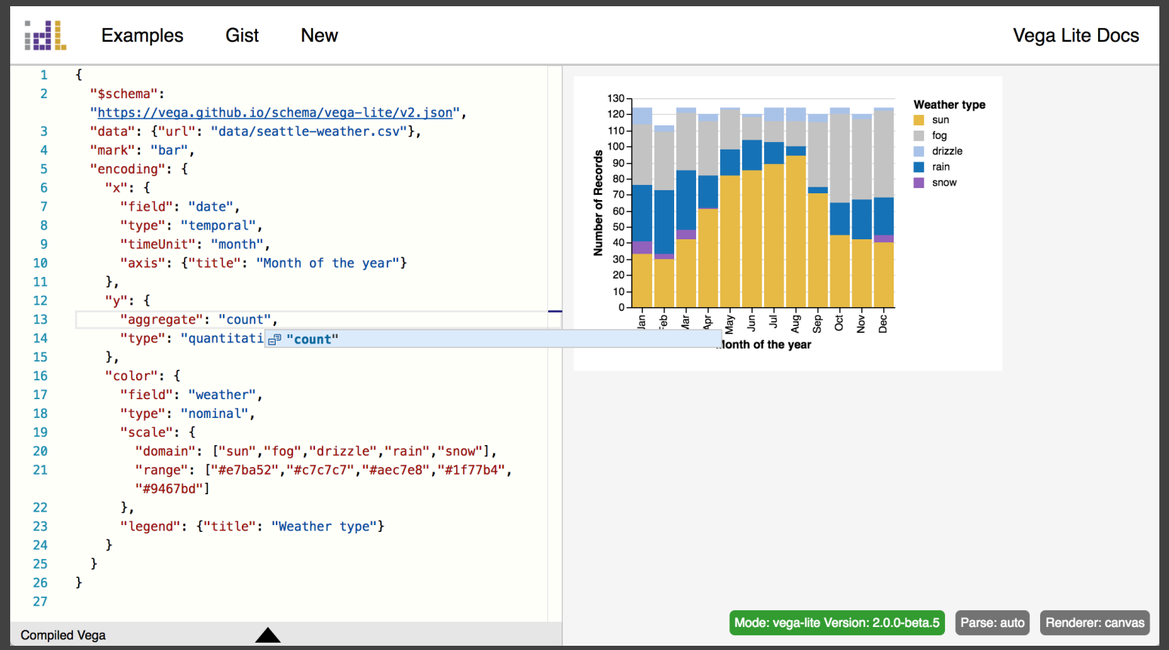


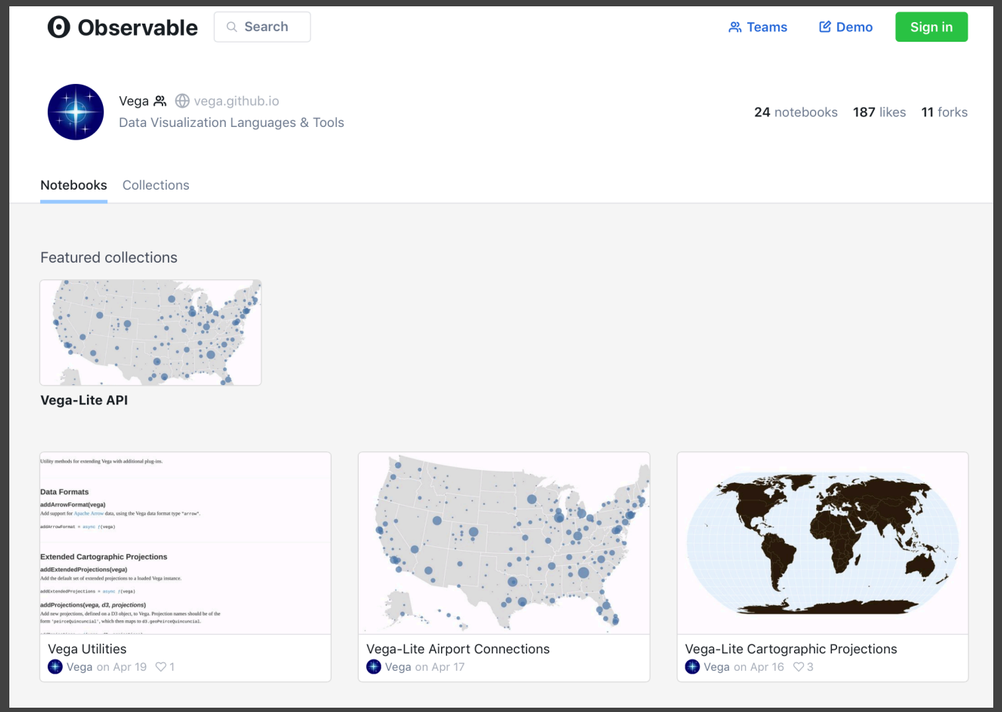
http://vega.github.io/vega-lite



http://altair-viz.github.io

Play with Vega-Lite online





http://vega.github.io/editor

https://observablehq.com/@vega