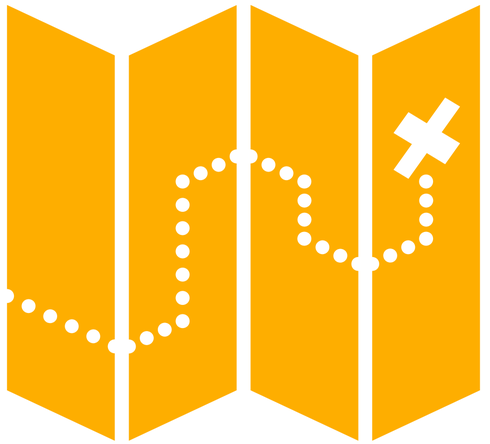
Mapping in Altair



Michael Colaresi

Maps and Computational Social Science

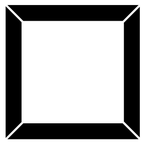
- Computational Social Science has three facets
  - New data
    - Eg digital traces: often unstructured/multi-scale (eg text, images, etc)
  - New tools for collection and inference
    - Eg scraping, machine learning, Bayesian computation
  - New User-interfaces
    - Interactive graphics and documents

Maps and Computational Social Science

- Maps are a particularly important form of visualization for CSS
  - Data with context
  - Geography encodes domain knowledge about places
- Maps do not have to be geographic
  - Network maps
  - Maps of textual documents

Mapping in Altair

Review

- Altair is a Python port for Vega-lite, a declarative syntax for building interactive graphics
- Chart — a canvas we are going to plot on 
- Mark — the visual cues we are going to populate the chart with 
  - eg bars, ticks, circles, points, etc)
- Encode —data is encoded to visual elements
  - data.batteryLife is encoded to x position 
  - data.memory is encoded to y position 

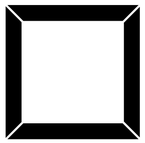
Mapping in Altair

Review

- Altair is a Python port for Vega-lite, a declarative syntax for building interactive graphics
- Chart — a canvas we are going to plot on 
- Mark — the visual cues we are going to populate the chart with 
  - eg bars, ticks, circles, points, etc)
- Encode —data is encoded to visual elements
  - data.batteryLife is encoded to x position 
  - data.memory is encoded to y position 

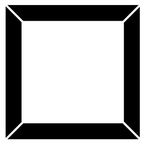
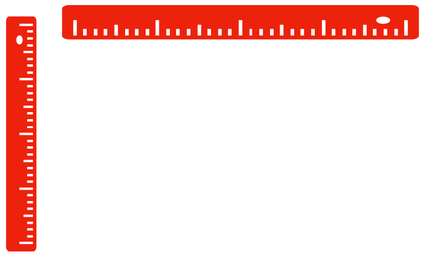
Mapping in Altair

Review

- Altair is a Python port for Vega-lite, a declarative syntax for building interactive graphics
- Chart — a canvas we are going to plot on 
- Mark — the visual cues we are going to populate the chart with 
  - eg bars, ticks, circles, points, etc)
- Encode —data is encoded to visual elements
  - data.batteryLife is encoded to x position 
  - data.memory is encoded to y position 

Mapping in Altair

Review

- Altair is a Python port for Vega-lite, a declarative syntax for building interactive graphics
- Chart — a canvas we are going to plot on 
- Mark — the visual cues we are going to populate the chart with 
  - eg bars, ticks, circles, points, etc)
- Encode —data is encoded to visual elements
  - data.batteryLife is encoded to x position
  - data.memory is encoded to y position 

Scatter plot example

alt.Chart(data=…)

Scatter plot example

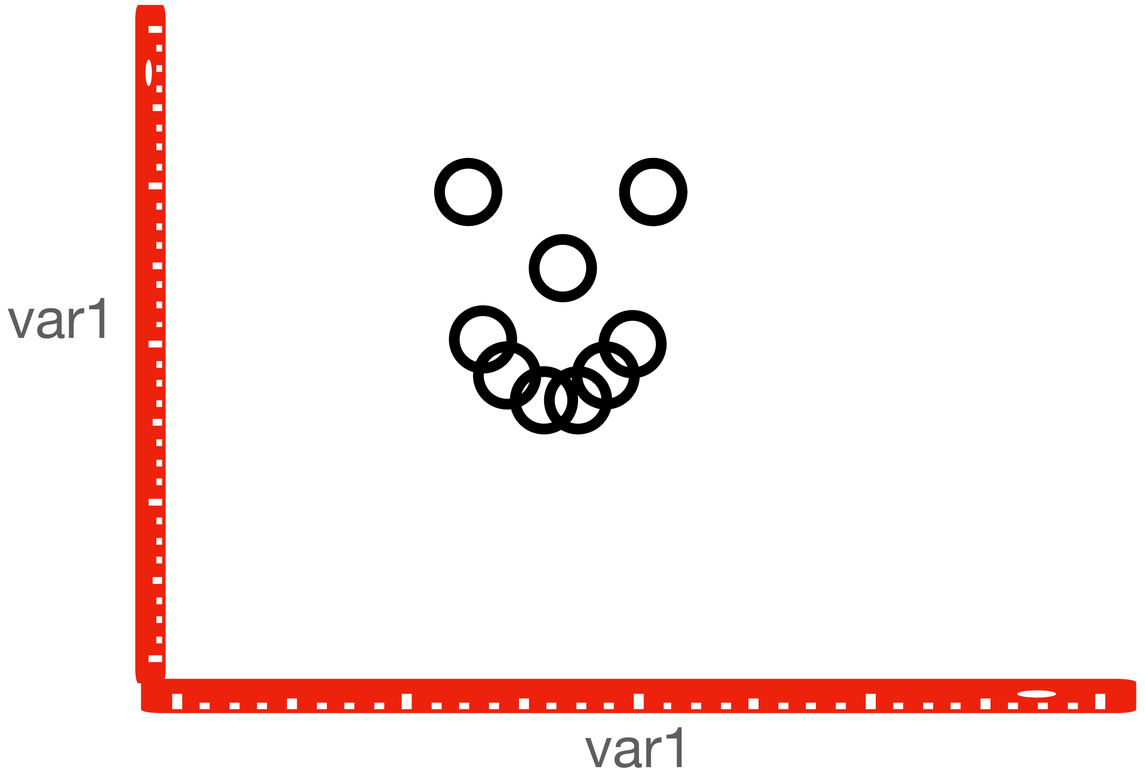
```
alt.Chart(data=…)
.mark_point()
```

Scatter plot example

```
alt.Chart(data=…)
    .mark_point()
    .encode(
        alt.X(“var1”)
        alt.Y
```

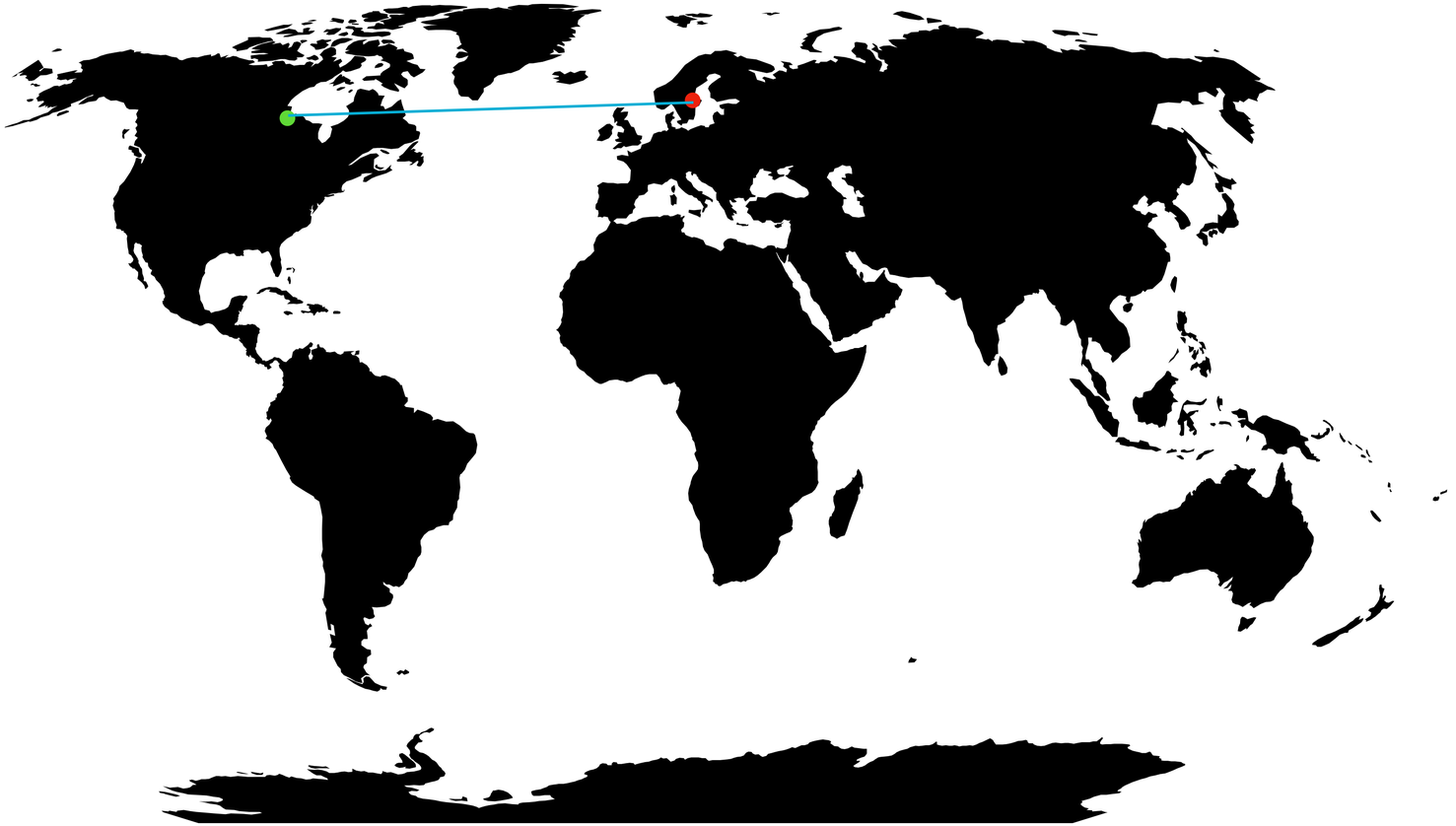
Scatter plot example

```
alt.Chart(data=…)
.mark_point()
.encode(
alt.X(“var1”)
alt.Y(“var2”))
```



Maps

- mark_geoshape
  - polygons 
  - points
  - line



GeoJson

Lots of work done by data object

- Json is like yaml, stores objects
- GeoJSON is a format for geographic information (inefficient)
  - But common
- You will also see shapefiles
- These are map-like objects that hold meta-data and shapes to plot

TopoJson

Lots of work done by data object

- Json is like yaml, stores objects
- GeoJSON is a format for geographic information (inefficient)
- TopoJSON more efficient storage