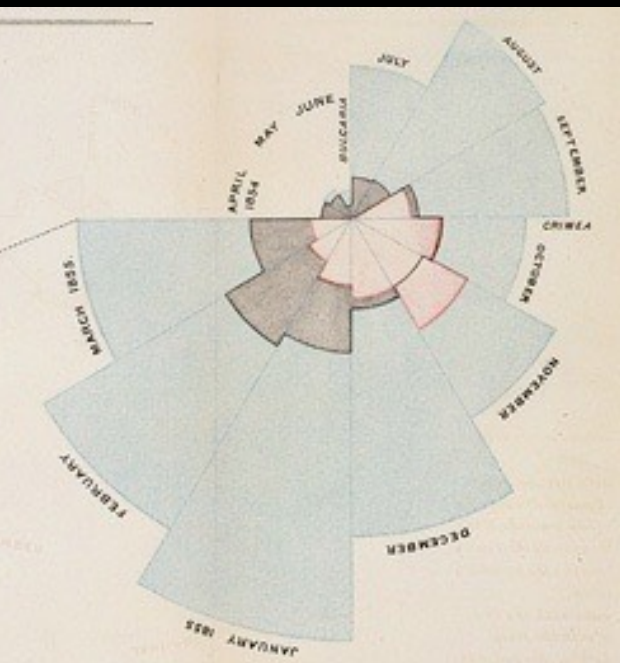
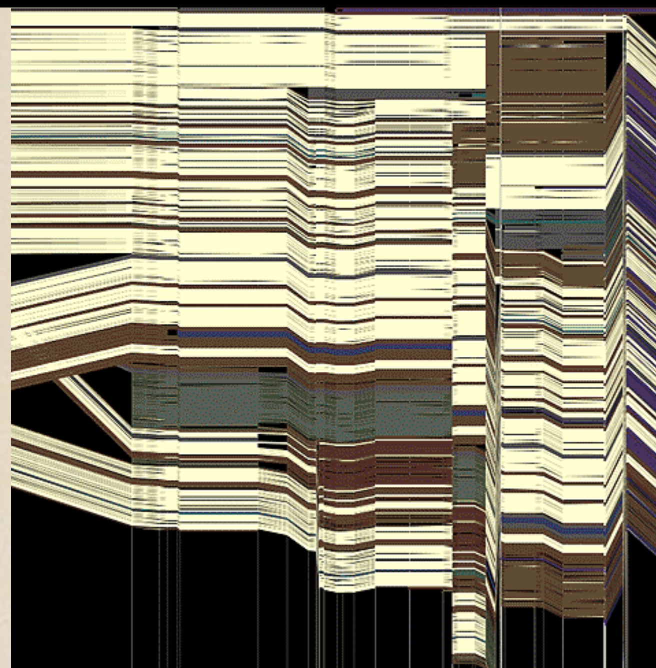
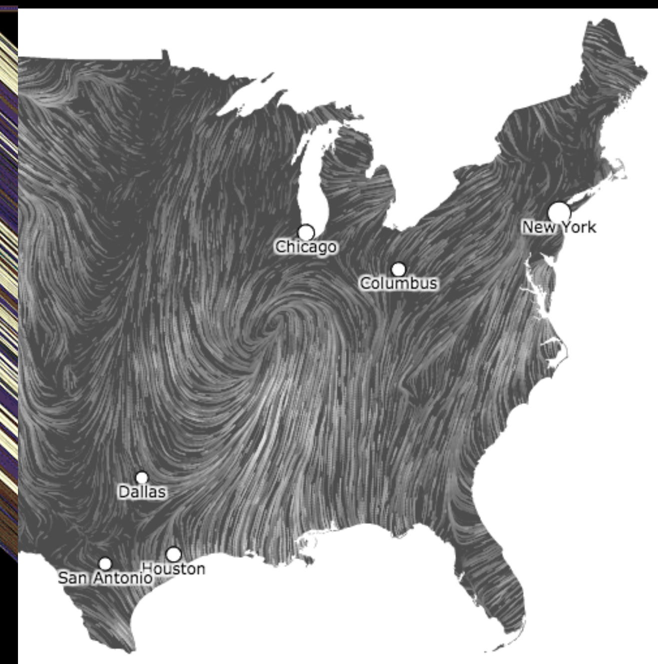
CSE 512 - Data Visualization

Visualization Tools







Jeffrey Heer University of Washington

How do people create visualizations?

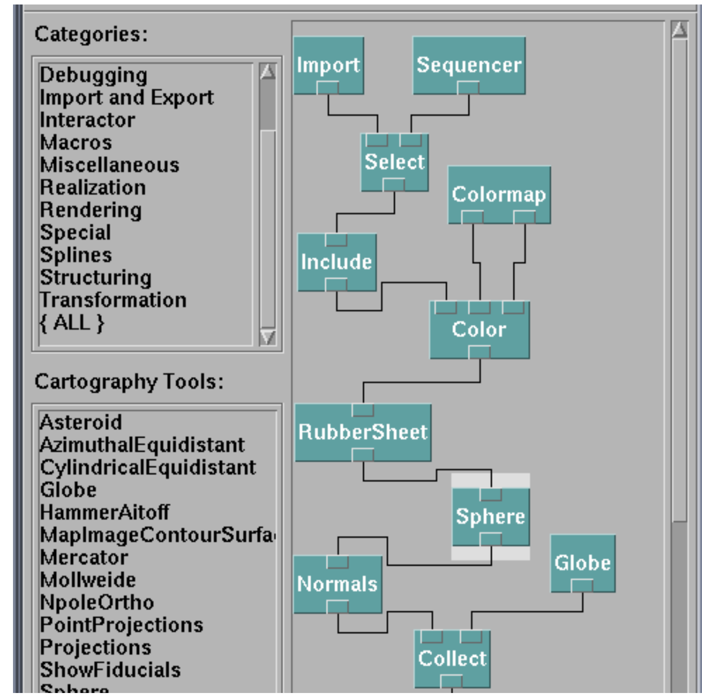


Chart Typology

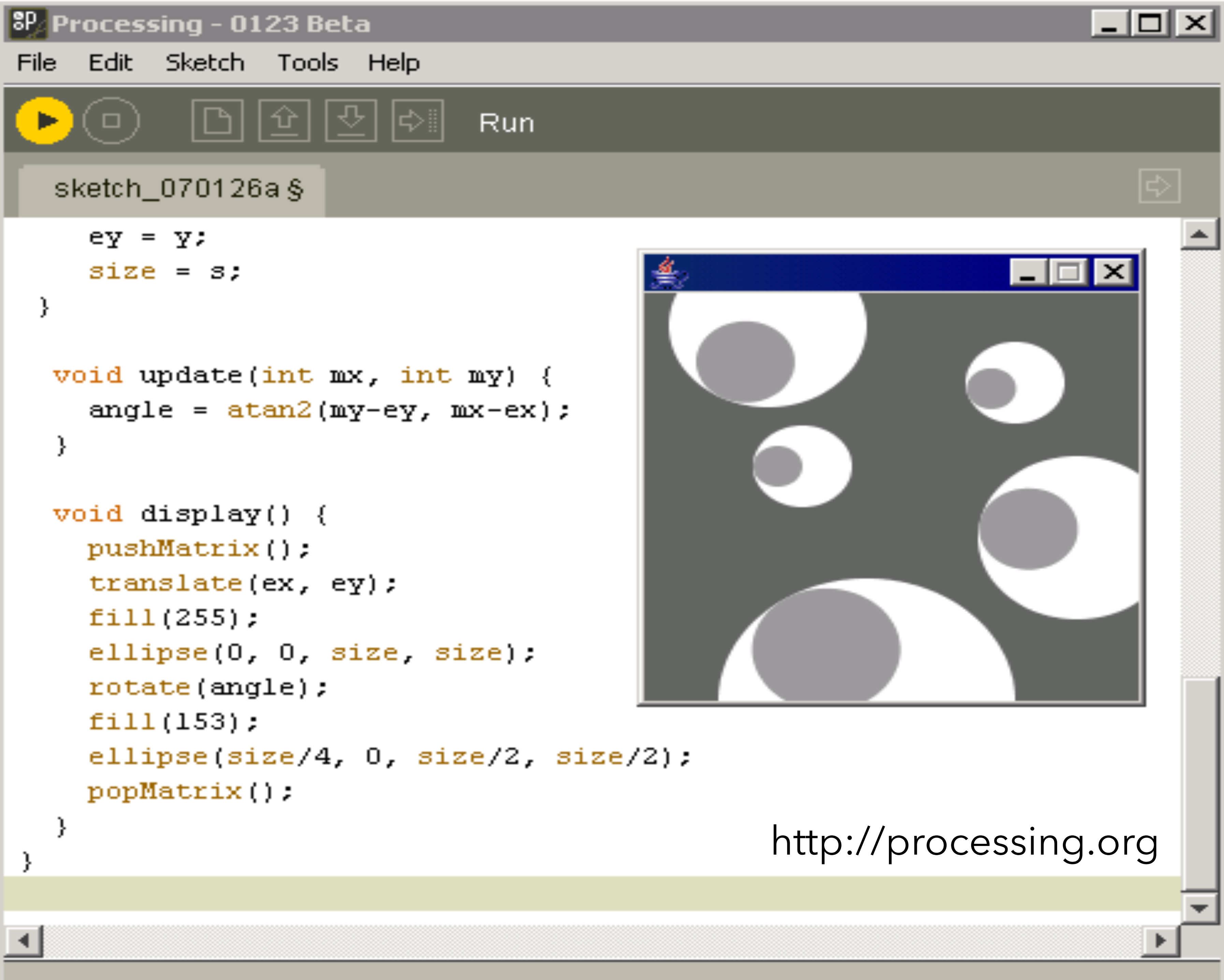
Pick from a stock of templates
Easy-to-use but limited expressiveness
Prohibits novel designs, new data types

Component Architecture

Permits more combinatorial possibilities
Novel views require new operators,
which requires software engineering

Graphics APIs

Canvas, OpenGL, Processing


Processing - 0123 Beta
File
Edit
Sketch
Tools
Help
Run
sketch_070126a §
    ey = y;
    size = s;
  }

  void update(int mx, int my) {
    angle = atan2(my-ey, mx-ex);
  }

  void display() {
    pushMatrix();
    translate(ex, ey);
    fill(255);
    ellipse(0, 0, size, size);
    rotate(angle);
    fill(153);
    ellipse(size/4, 0, size/2, size/2);
    popMatrix();
  }
}
http://processing.org

US Air Traffic, Aaron Koblin

Graphics APIs

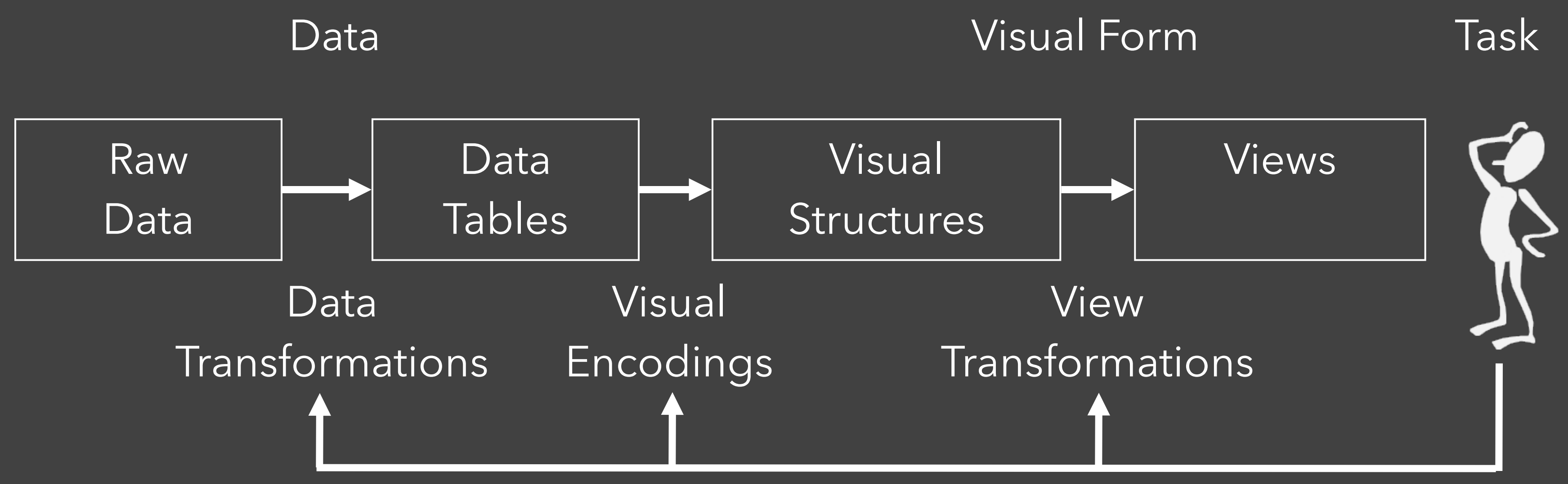
Canvas, OpenGL, Processing

Component Architectures

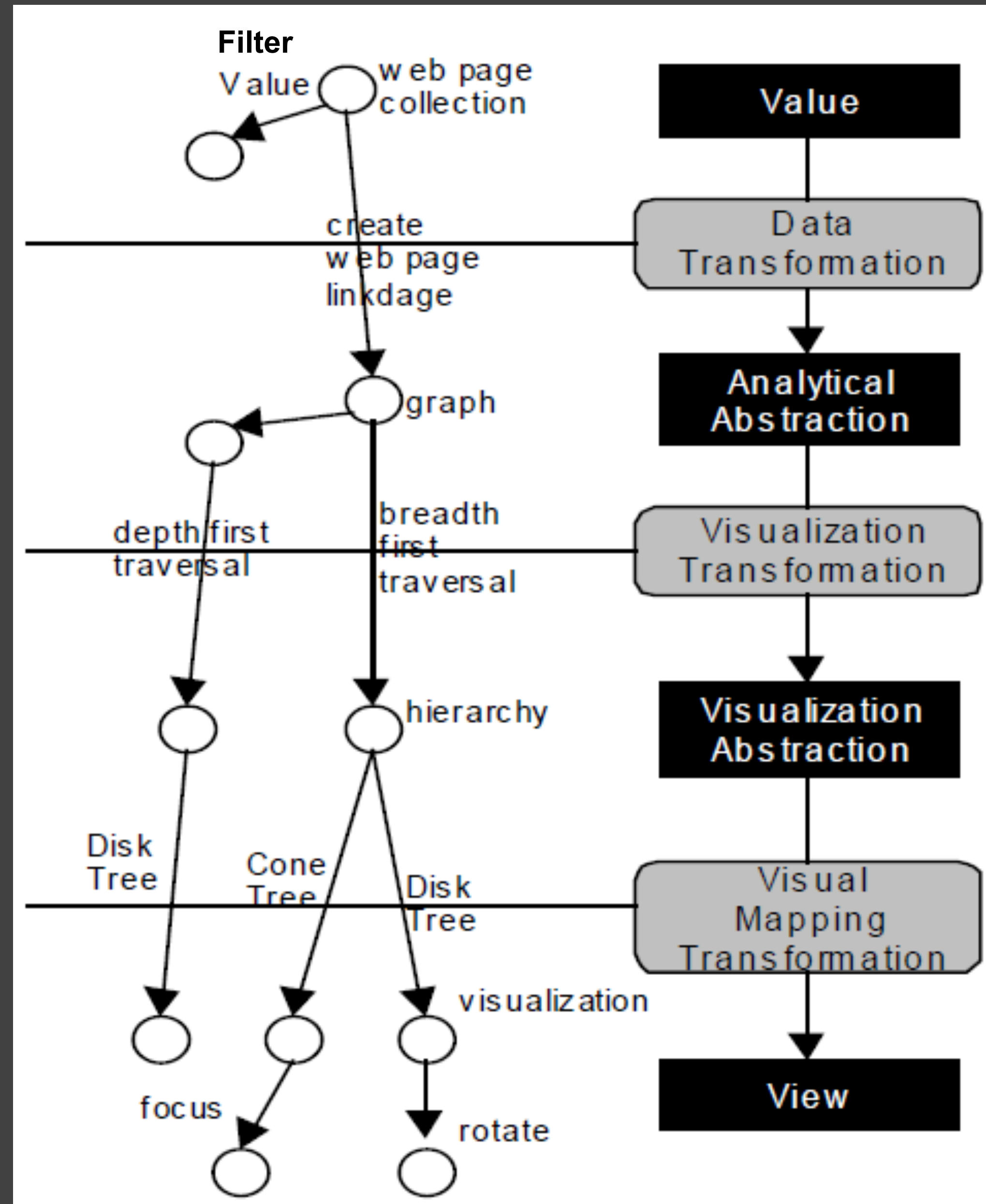
Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing


Data
Visual Form
Task
Raw Data
Data Tables
Visual Structures
Views
Data Transformations
Visual Encodings
View Transformations

Data State Model
[Chi 98]



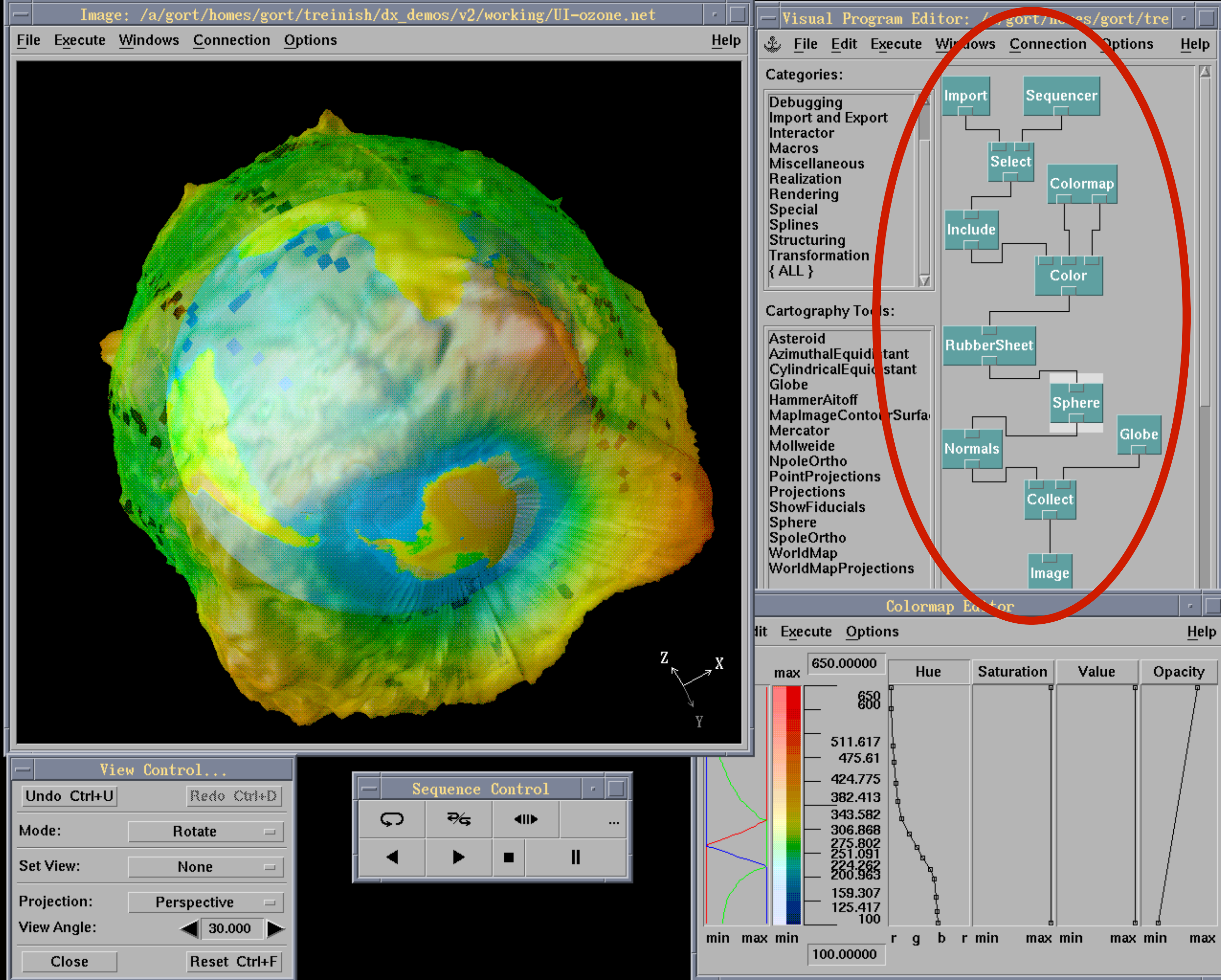
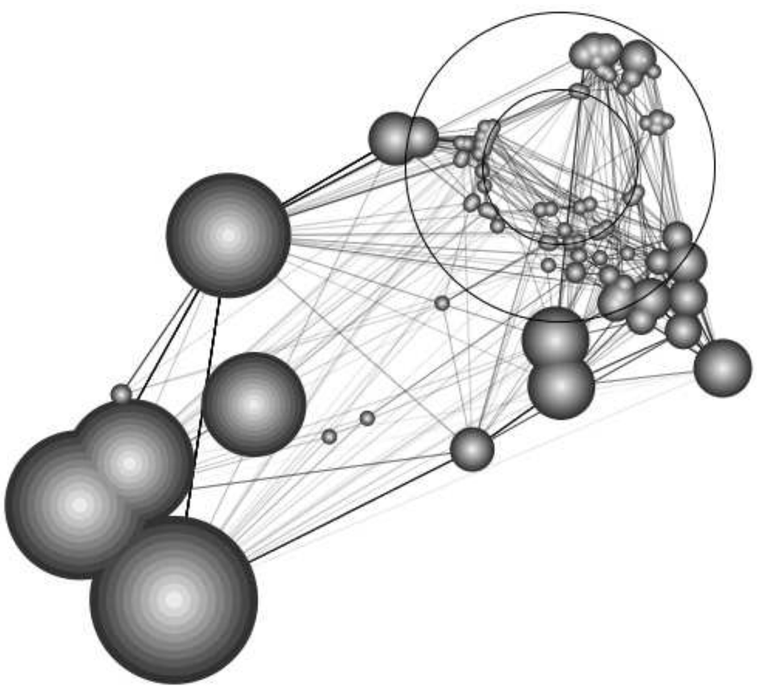


Image: /a/gort/homes/gort/treinish/dx_demos/v2/working/UI-ozone.net
File Execute Windows Connection Options Help
Visual Program Editor
File Edit Execute Windows Connection Options Help
Categories:
Debugging
Import and Export
Interactor
Macros
Miscellaneous
Realization
Rendering
Special
Splines
Structuring
Transformation
{ ALL }
Asteroid
AzimuthalEquidistant
CylindricalEquidistant
Globe
HammerAitoff
Mercator
Mollweide
NpoleOrtho
PointProjections
Projections
ShowFiducials
Sphere
SpoleOrtho
WorldMap
WorldMapProjections
Import
Sequencer
Select
Colormap
Include
Color
RubberSheet
Sphere
Normals
Globe
Collect
Image
Colormap Editor
Execute Options Help
Hue Saturation Value Opacity
650.00000
100.00000
View Control...
Undo Ctrl+U
Redo Ctrl+D
Mode: Rotate
Set View: None
Projection: Perspective
View Angle: 30.000
Close
Reset Ctrl+F
Sequence Control

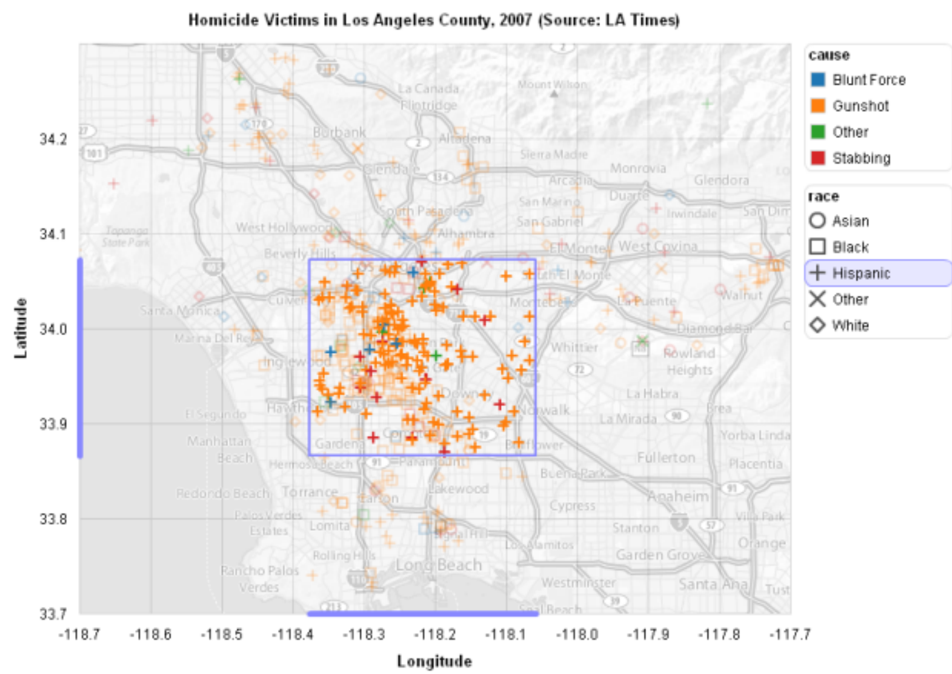
Prefuse & Flare

Operator-based toolkits for visualization design

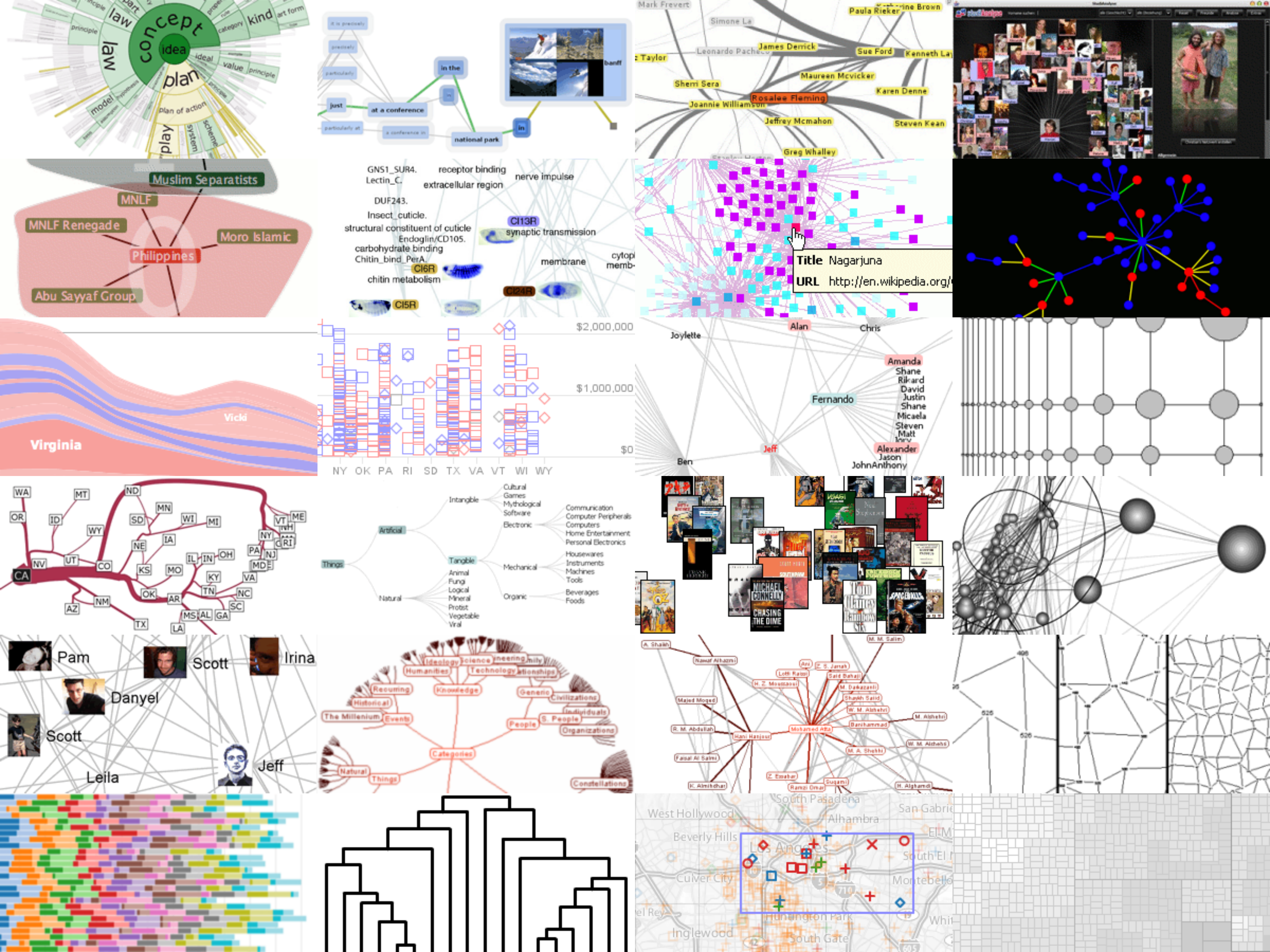
Vis = (Input Data -> Visual Objects) + Operators



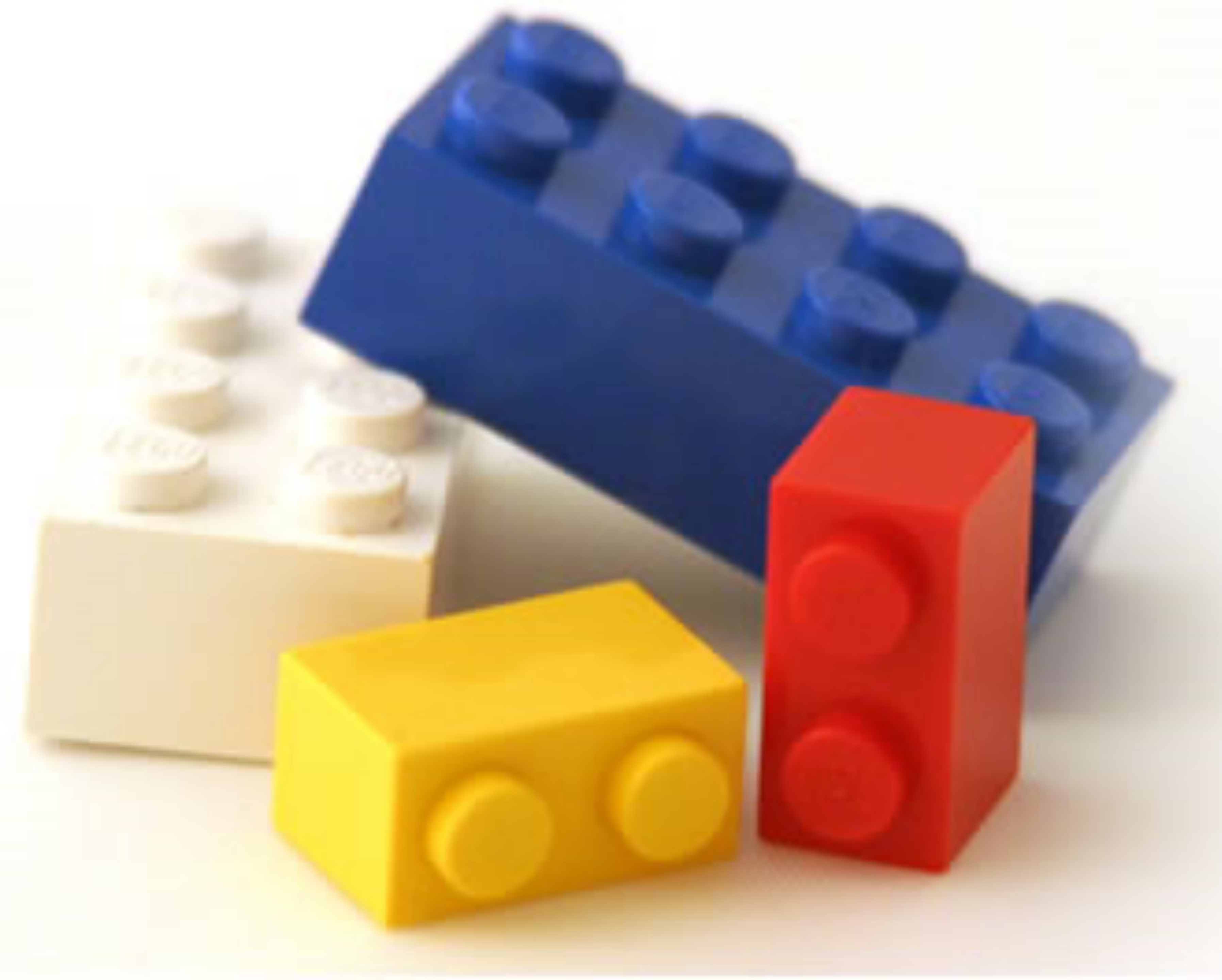
Prefuse (http://prefuse.org)



Flare (http://flare.prefuse.org)



concept
idea
law
kind
plan
plan of action
play
system
model
principle
value
category
in the
just
at a conference
national park
banff
in
Paula Rieker
Simone La
James Derrick
Sue Ford
Kenneth Lay
Maureen Mcvicker
Sherri Sera
Rosalee Fleming
Karen Denne
Joannie Williamson
Jeffrey Mcmahon
Steven Kean
Greg Whalley
Muslim Separatists
MNLF
MNLF Renegade
Moro Islamic
Philippines
Abu Sayyaf Group
GNS1_SUR4.
receptor binding
nerve impulse
Lectin_C.
extracellular region
DUF243.
Insect_cuticle.
CI13R
structural constituent of cuticle
synaptic transmission
Endoglin/CD105.
carbohydrate binding
Chitin_bind_PerA.
membrane
CI6R
chitin metabolism
CI24R
CI5R
Title Nagarjuna
URL http://en.wikipedia.org/
Vicki
Virginia
$2,000,000
$1,000,000
$0
NY OK PA RI SD TX VA VT WI WY
Joylette
Alan
Chris
Amanda
Shane
Rikard
David
Justin
Fernando
Micaela
Steven
Matt
Jeff
Alexander
Jason
Ben
JohnAnthony
WA
MT
ND
OR
ID
MN
SD
WI
MI
WY
NE
IA
NV
UT
CO
IL
IN
OH
CA
KS
MO
KY
OK
TN
AR
NM
AZ
TX
MS
AL
GA
LA
SC
NC
VA
MD
NJ
NY
PA
VT
NH
ME
RI
Things
Artificial
Intangible
Cultural
Games
Mythological
Software
Electronic
Communication
Computer Peripherals
Computers
Home Entertainment
Personal Electronics
Tangible
Mechanical
Housewares
Instruments
Machines
Tools
Natural
Animal
Fungi
Logical
Mineral
Protist
Vegetable
Viral
Organic
Beverages
Foods
Pam
Scott
Irina
Danyel
Scott
Leila
Jeff
Humanities
Technology
Knowledge
Historical
The Millenium
Events
Generic
People
Organizations
Categories
Natural
Things
Constellations
Mohamed Atta
Hani Hanjour
South Pasadena
West Hollywood
Alhambra
Beverly Hills
Culver City
Inglewood
South Gate
Montebello



Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Chart Typologies
Excel, Google Charts

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Canvas, OpenGL, Processing

Chart Typologies

Data Sets : State Quick Facts

Uploaded By: zinggoat
Created at: Friday May 18, 3:08 PM
Data Source: US Census Bureau
Description:
Tags: people census

view as text | edit data set

	People QuickFacts	Population 2005 estimate	Population percent change April 1 2000 to July 1 2005	Population 2000	Population percent change 1990 to 2000	Persons under 5 years old percent 2004	Persons under 18 years old percent 2004	Persons 65 years old and over percent 2004
1	Alabama	4557808	0.03	4447100	0.1	0.07	0.24	0.13
2	Alaska	663661	0.06	626932	0.14	0.08	0.29	0.06
3	Arizona	5939292	0.16	5130632	0.4	0.08	0.27	0.13
4	Arkansas	2779154	0.04	2673400	0.14	0.07	0.25	0.14
5	California	36132147	0.07	33871648	0.14	0.07	0.27	0.11
6	Colorado	4665177	0.08	4301261	0.31	0.07	0.26	0.1
7	Connecticut	3510297	0.03	3405565	0.04	0.06	0.24	0.14
8	Delaware	843524	0.08	783600	0.18	0.07	0.23	0.13
9	Florida	17789864	0.11	15982378	0.24	0.06	0.23	0.17
10	Georgia	9072576	0.11	8186453	0.26	0.08	0.26	0.1
11	Hawaii	1275194	0.05	1211537	0.09	0.07	0.24	0.14
12	Idaho	1429096	0.1	1293953	0.29	0.07	0.27	0.11
13	Illinois	12763371	0.03	12419293	0.09	0.07	0.26	0.12

Choosing a visualization type for State Quick Facts

Analyze a text

Tag Cloud

How are you using your words? This enhanced tag cloud will show you the words popularity in the given set of text.

Learn more

Wordle

Wordle is a toy for generating "word clouds" from text that you provide. The clouds give greater prominence to words that appear more frequently in the source text.

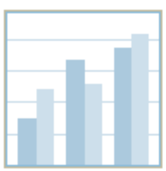
Learn more

Word Tree

See a branching view of how a word or phrase is used in a text. Navigate the text by zooming and clicking.

Learn more

Compare a set of values



Bar Chart

How do the items in your data set stack up? A bar chart is a simple and recognizable way to compare values. You can display several sets of bars for multivariate comparisons.

Learn more

Block Histogram

This versatile chart lets you get a quick sense of how a single set of data is distributed. Each item in the data is an individually identifiable block.

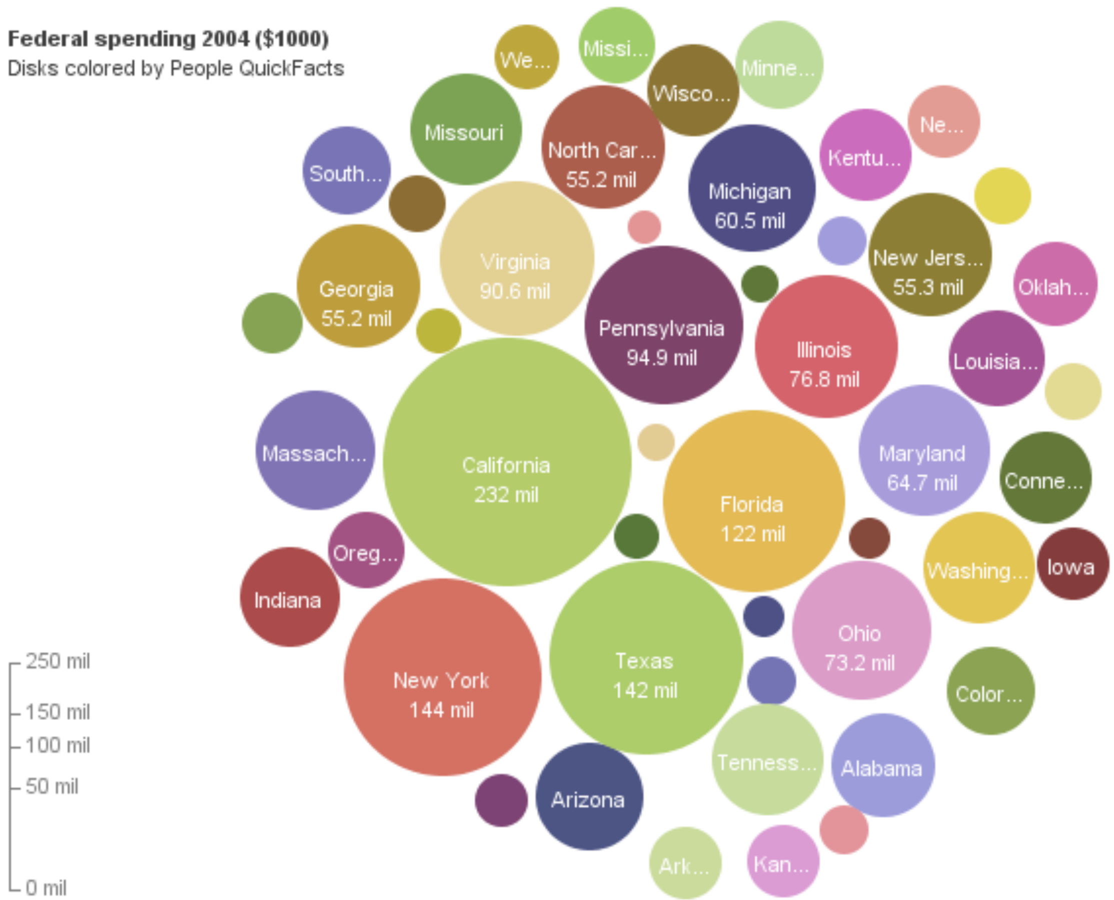
Learn more

Visualizations : Federal Spending by State, 2004

Creator: Anonymous
Tags: census people

People QuickFac...
Click to select,
Ctrl-Click: multiple
Shift-Click: range

- Alabama
- Alaska
- Arizona
- Arkansas
- California
- Colorado
- Connecticut
- Delaware
- Florida
- Georgia
- Hawaii
- Idaho
- Illinois
- Indiana
- Iowa
- Kansas
- Kentucky
- Louisiana
- Maine
- Maryland



Search>>

To highlight or find totals click or ctrl-click.

Bubble Size: Federal spending 2004 ($1000) | Label: People QuickFacts | Color: People QuickFacts

- Retail sales per capita 2002
- Minority-owned firms percent of total 1997
- Women-owned firms percent of total 1997
- Housing units authorized by building permits 2004
- Federal spending 2004 ($1000)
- Land area 2000 (square miles)
- Persons per square mile 2000
- FIPS Code

Data file ... Census Bureau

This data set has not yet been rated

full image

rate this

Comments (1)

MAD LIBS®

MY MUSIC LESSON

Every Wednesday, when I get home from school, I have a piano lesson. My teacher is a very strict house (NOUN). Her name is Hillary Clinton (CELEBRITY (FEMALE)). Our piano is a Steinway Concert tree (NOUN) and it has 88 cups (PLURAL NOUN). It also has a soft pedal and a/an Smily (ADJECTIVE) pedal. When I have a lesson, I sit down on the piano AIBERTO (NOUN) and play for 16 Minuts (PERIOD OF TIME). I do scales to exercise my cats (PLURAL NOUN), and then I usually play a minuet by Johann Sebastian washington (CELEBRITY (LAST NAME)). Teacher says I am a natural Hamted House (NOUN) and have a good musical leg (PART OF THE BODY). Perhaps when I get better I will become a concert vet (PROFESSION) and give a recital at Carnegie hospital (TYPE OF BUILDING).

[M]ost charting packages channel user requests into a **rigid array of chart types**. To atone for this lack of flexibility, they offer a kit of post-creation editing tools to return the image to what the user originally envisioned. **They give the user an impression of having explored data rather than the experience**.

Leland Wilkinson
The Grammar of Graphics, 1999

Chart Typologies

Excel, Many Eyes, Google Charts

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

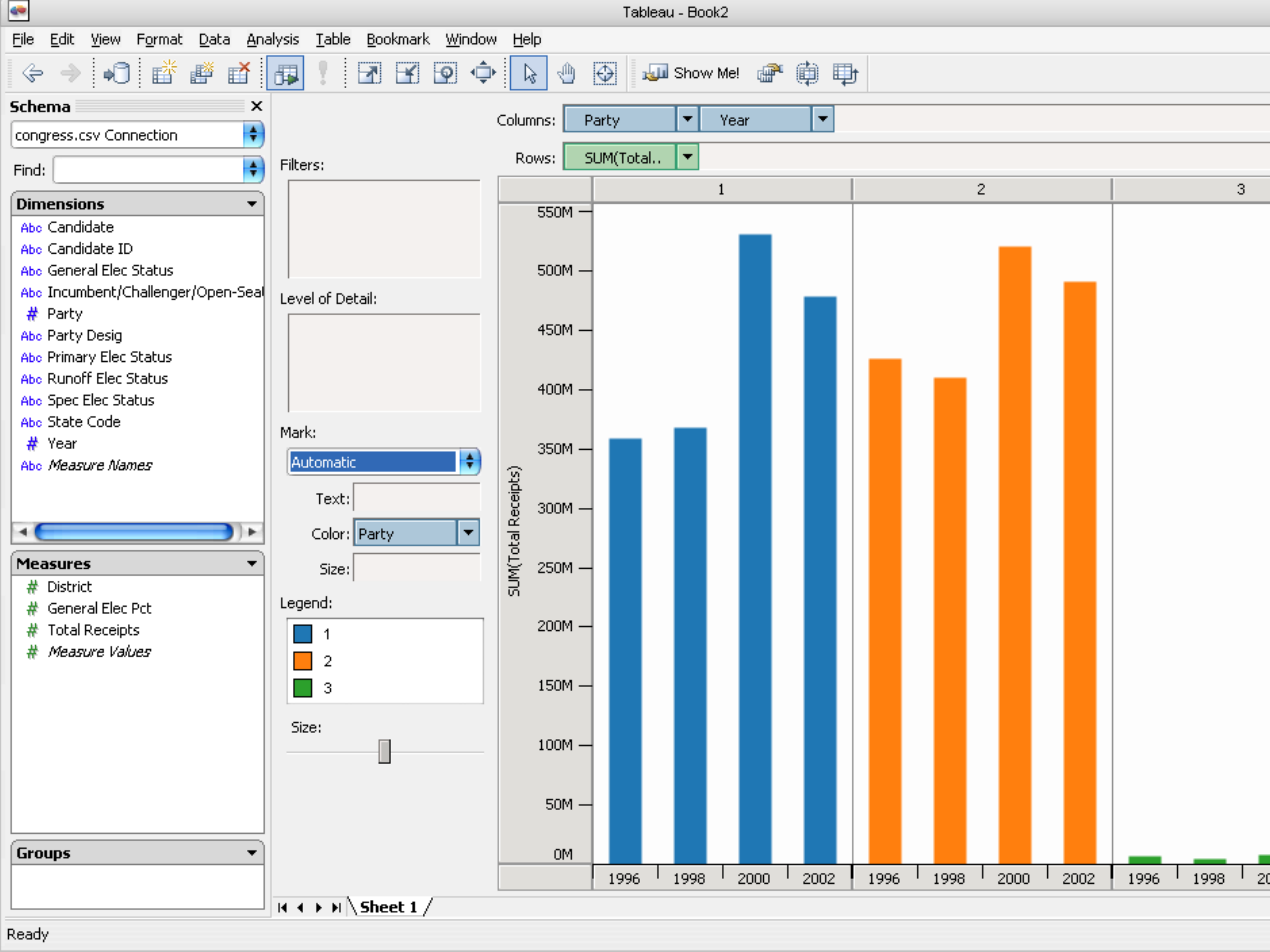
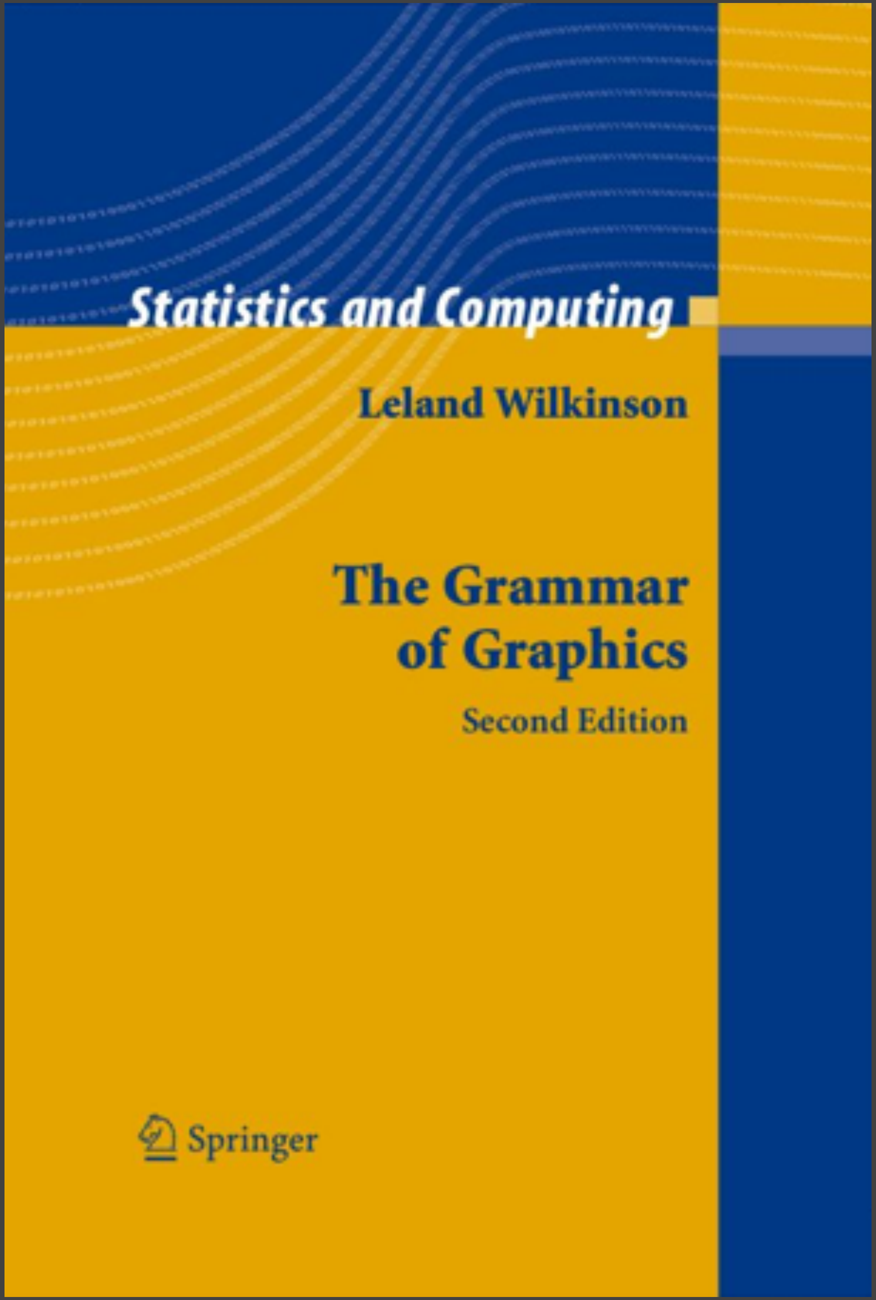
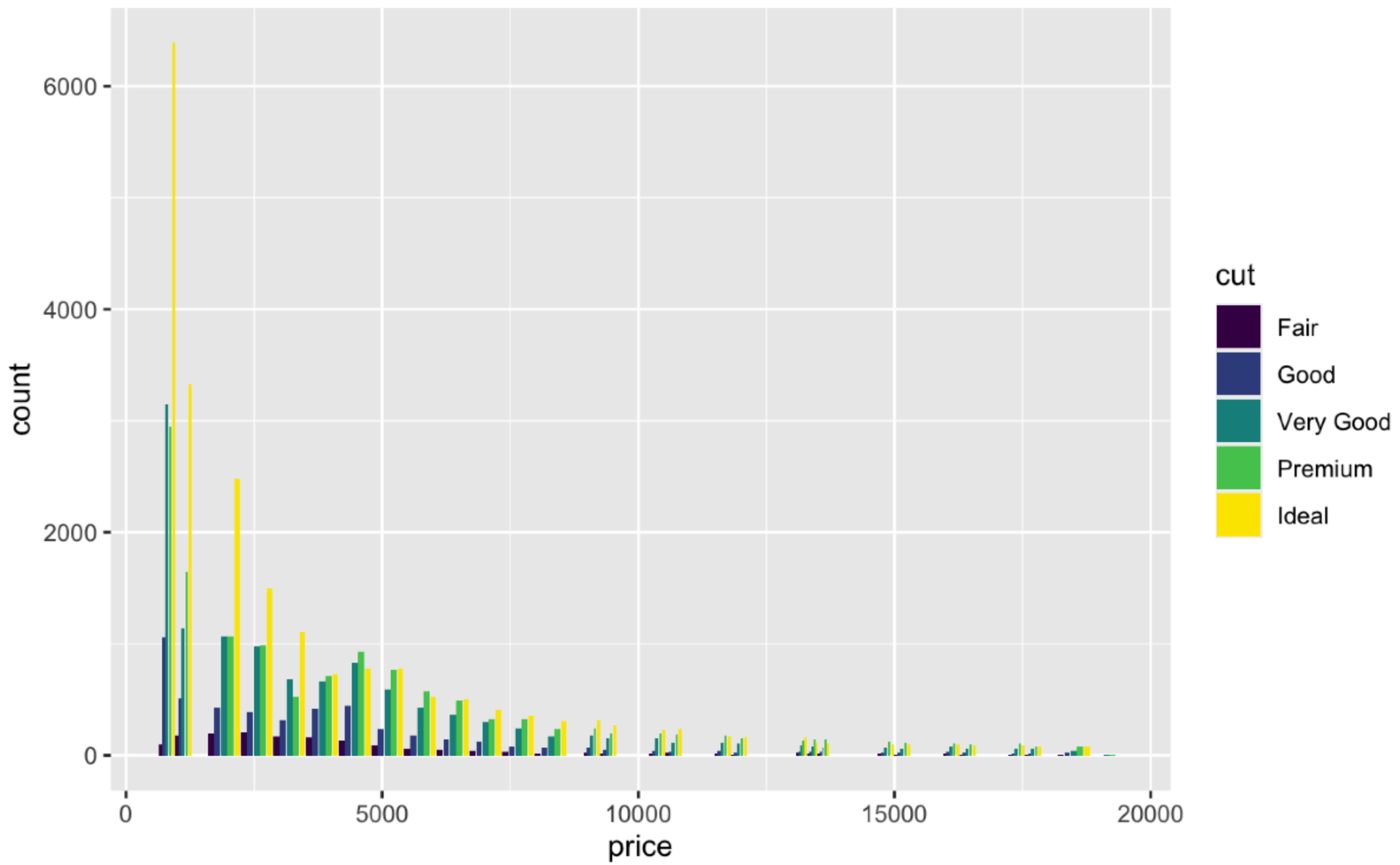


Tableau - Book2
File Edit View Format Data Analysis Table Bookmark Window Help
Show Me!
Schema
congress.csv Connection
Find:
Dimensions
Candidate
Candidate ID
General Elec Status
Incumbent/Challenger/Open-Seat
Party
Party Desig
Primary Elec Status
Runoff Elec Status
Spec Elec Status
State Code
Year
Measure Names
Measures
District
General Elec Pct
Total Receipts
Measure Values
Groups
Filters:
Level of Detail:
Mark:
Automatic
Text:
Color: Party
Size:
Legend:
1
2
3
Size:
Columns: Party Year
Rows: SUM(Total..
1
2
3
SUM(Total Receipts)
550M
500M
450M
400M
350M
300M
250M
200M
150M
100M
50M
0M
1996 1998 2000 2002 1996 1998 2000 2002 1996 1998
Sheet 1
Ready


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

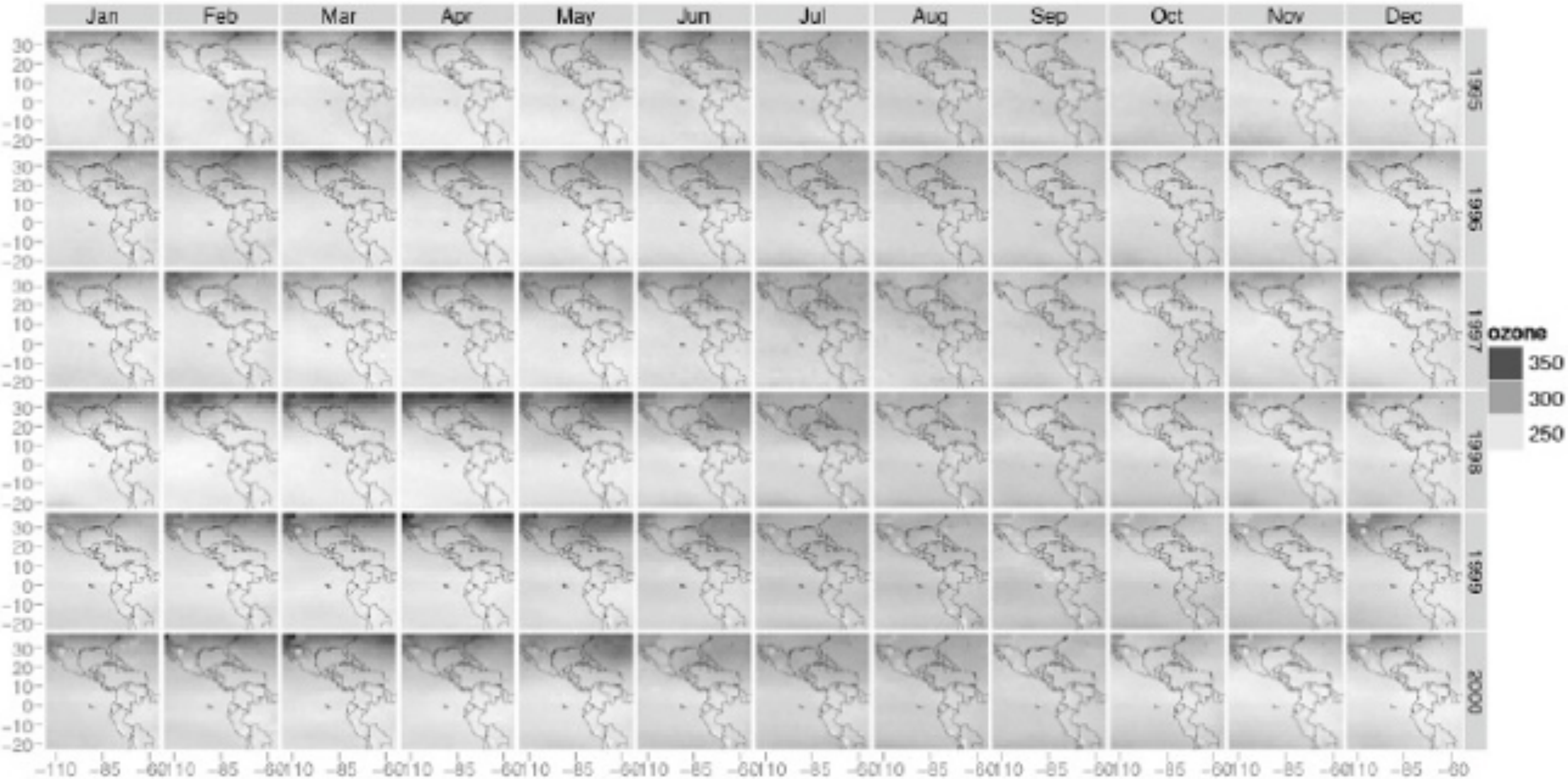
```
ggplot(diamonds, aes(x=price, fill=cut))
+ geom_bar(position="dodge")
```

ggplot2



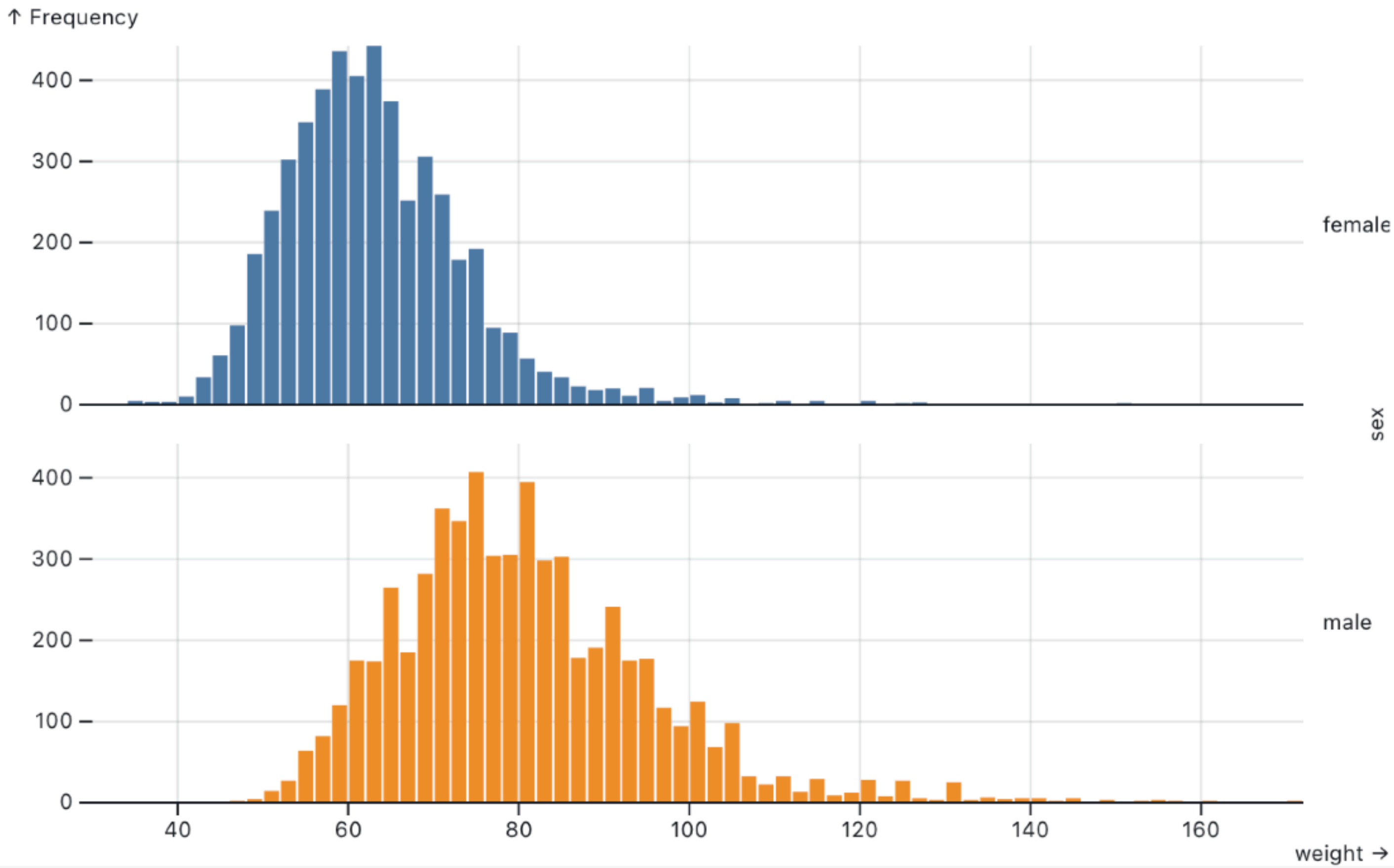

```
ggplot(diamonds, aes(x=price, fill=cut))
+ geom_bar(position="dodge")
```

ggplot2




```
qplot(long, lat, data = expo, geom = "tile", fill = ozone,
  facets = year ~ month) +
scale_fill_gradient(low = "white", high = "black") + map
```

ggplot2




```
Plot.plot({
  grid: true,
  facet: {
    data: athletes,
    y: "sex"
  },
  marks: [
    Plot.rectY(athletes, Plot.binX({y: "count"}, {x: "weight", fill: "sex"})),
    Plot.ruleY([0])
  ]
})
```

Observable Plot

Chart Typologies
Excel, Many Eyes, Google Charts

Visual Analysis Grammars
VizQL, ggplot2, Vega-Lite

Component Architectures
Prefuse, Flare, Improvise, VTK

Graphics APIs
Canvas, OpenGL, Processing

Ease-of-Use

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Ease-of-Use

Expressiveness

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Ease-of-Use

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

?

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Expressiveness

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Ease-of-Use

Expressiveness

Visualization Building Blocks

Visualization Building Blocks

Data	Input data to visualize

Visualization Building Blocks

Data	Input data to visualize
Transforms	Group, aggregate, stats, layout

Visualization Building Blocks

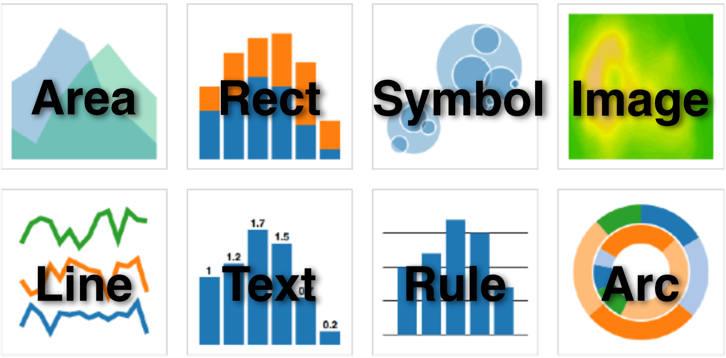
Data	Input data to visualize
Transforms	Group, aggregate, stats, layout
Scales	Map data values to visual values

Visualization Building Blocks

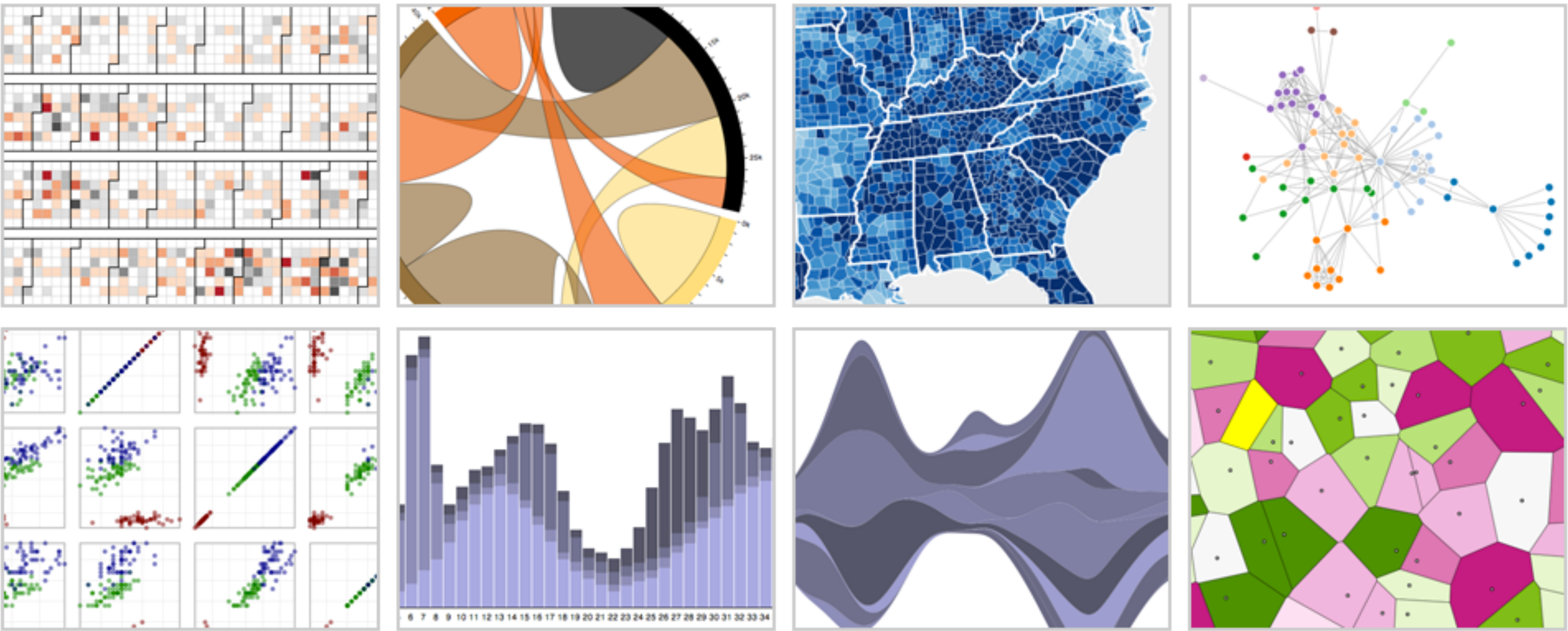
Data	Input data to visualize
Transforms	Group, aggregate, stats, layout
Scales	Map data values to visual values
Guides	Axes & legends visualize scales

Visualization Building Blocks

Data	Input data to visualize
Transforms	Group, aggregate, stats, layout
Scales	Map data values to visual values
Guides	Axes & legends visualize scales
Marks	Data-representative graphics



d3.js Data-Driven Documents



Mike Bostock, Vadim Ogievetsky, Jeffrey Heer [TVCG 2011]
+ Jason Davies (geo, 2011–13) & Philippe Rivière (2016–)

What is D3?

1. A collection of reusable visualization utilities

2. A tool for updating the browser's Document Object Model (DOM) in response to input data

What is D3?

1. A collection of reusable visualization utilities

    Data: d3.csv, d3.json, …

    Scales: d3.scaleLinear, d3.scaleLog, …

    Projections: d3.geoPath, d3.geoMercator, …

    Layout: d3.tree, d3.treemap, d3.force, …

    Interaction: d3.brush, d3.zoom, …

2. A tool for updating the browser’s Document Object Model (DOM) in response to input data

What is D3?

1. A collection of reusable visualization utilities

2. A tool for updating the browser's Document Object Model (DOM) in response to input data

   Select: query DOM content

   Join: bind input data to DOM elements

   Update: set DOM element properties

   Transition: animate changes over time

Why D3?

Enable highly custom visualization design

Support animation and dynamic displays

Support rich and varied interactions

Interoperate via web standards (HTML, SVG, CSS)

Avoid artificial limits. If a browser can do it, D3 should be able to take advantage of it.

Why D3?

"the authors have undeniably helped to bring data visualization to the mainstream. [D3] is a cornerstone contribution to this conference specifically and more generally to the success of our field as a whole"

IEEE VIS 2021 Test of Time Award

Why D3?

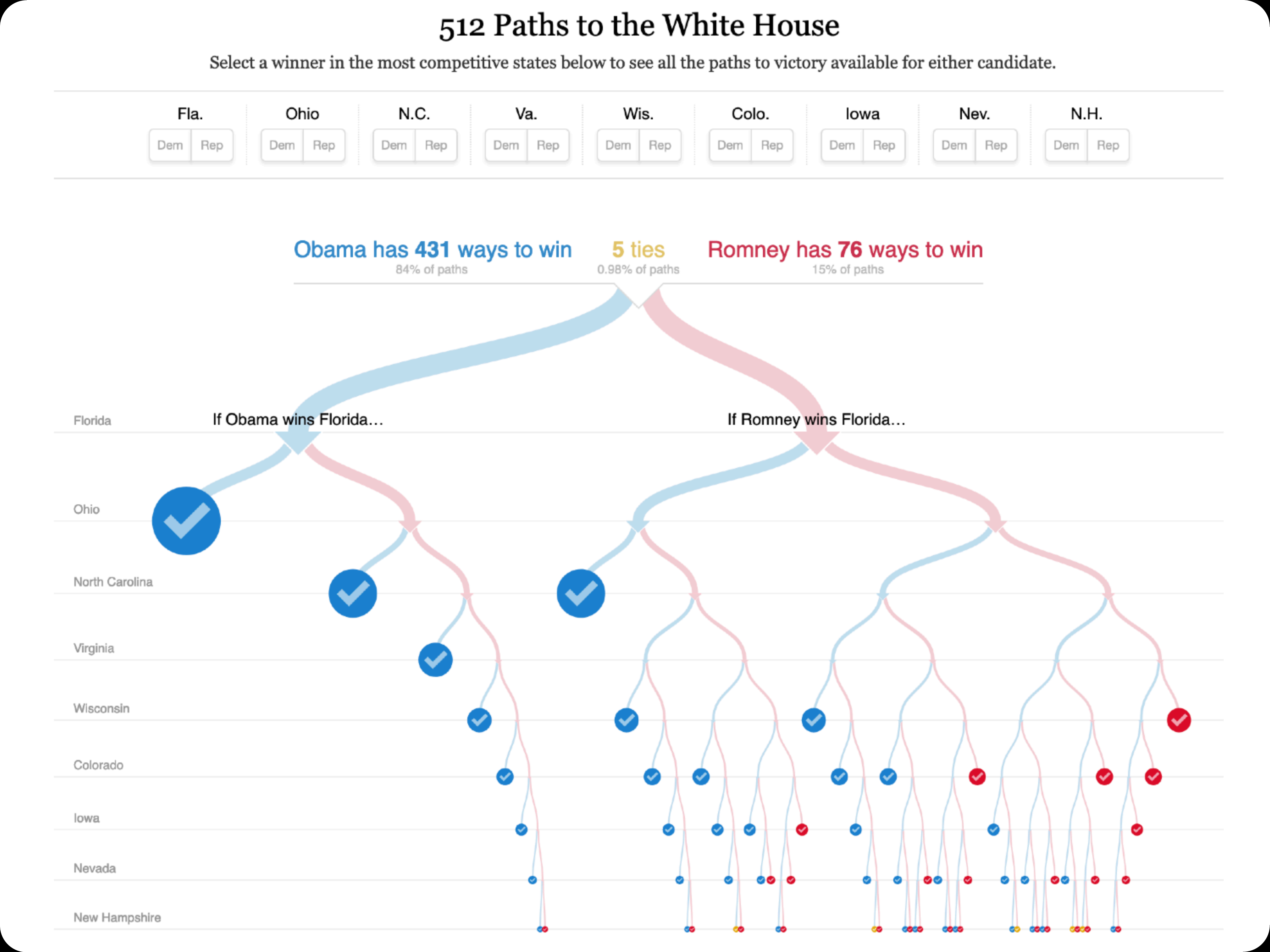
D3 “slingshotted the field into growth, diversification and creativity that has been unprecedented” and “changed how millions of data visualizations are created across newsrooms, websites, and personal portfolios”

Information is Beautiful 2022 Test of Time Award

Why D3?

“Use D3 if you think it’s perfectly normal to write a hundred lines of code for a bar chart.”

Amanda Cox, Former Graphics Editor, NY Times



512 Paths to the White House
Select a winner in the most competitive states below to see all the paths to victory available for either candidate.
Fla.
Ohio
N.C.
Va.
Wis.
Colo.
Iowa
Nev.
N.H.
Dem
Rep
Obama has 431 ways to win
84% of paths
5 ties
0.98% of paths
Romney has 76 ways to win
15% of paths
Florida
If Obama wins Florida…
If Romney wins Florida…
Ohio
North Carolina
Virginia
Wisconsin
Colorado
Iowa
Nevada
New Hampshire

D3 Selections

The core abstraction in D3 is a ***selection***.

D3 Selections

The core abstraction in D3 is a ***selection***.

```
// Add and configure an SVG element (<svg width="500" height="300">)
svg = d3.append("svg")          // add new SVG to page body
    .attr("width", 500)         // set SVG width to 500px
    .attr("height", 300);       // set SVG height to 300px
```

Data

DOM

Data	DOM
	<svg width="500" ...>
	</svg>

```
svg = d3.append("svg")
    .attr("width", 500)
    .attr("height", 300);
```

D3 Selections

The core abstraction in D3 is a ***selection***.

```
// Add and configure an SVG element (<svg width="500" height="300">)
svg = d3.append("svg")            // add new SVG to page body
    .attr("width", 500)           // set SVG width to 500px
    .attr("height", 300);         // set SVG height to 300px

// Select & update existing rectangles contained in the SVG element
svg.selectAll("rect")             // select all SVG rectangles
    .attr("width", 100)           // set rect widths to 100px
    .style("fill", "steelblue");  // set rect fill colors
```

Data

DOM

<svg width="500" …>

</svg>

Data

DOM

<svg width="500" ...>

svg.selectAll("rect")

???

</svg>

Data

DOM

<svg width="500" …>

<rect ..></rect>

<rect ..></rect>

<rect ..></rect>

<rect ..></rect>

<rect ..></rect>

</svg>

Data

svg.selectAll(“rect”)

DOM

<svg width=“500” ...>

<rect ... />

<rect ... />

<rect ... />

<rect ... />

<rect ... />

</svg>

Data

DOM

```
svg.selectAll(“rect”)
 .attr(“width”, 100)
 .style(“fill”, “steelblue”)
```

<svg width=“500” …>

<rect width=“100”
 style=“fill: steelblue;”
/>

<rect width=“100
 style=“fill: steelblue;”
/>

<rect width=“100
 style=“fill: steelblue;”

Data Binding

Selections can ***bind*** **data and DOM elements**.

values = [{…}, {…}, {…}, …]; // input data as JS objects

Data Binding

Selections can ***bind*** **data and DOM elements**.

values = [{…}, {…}, {…}, …]; // input data as JS objects

```
// Select SVG rectangles and bind them to data values.
bars = svg.selectAll("rect.bars").data(values);
```

Data

values = [
  { cat: "a", value: 5 },
  { cat: "b", value: 7 },
  { cat: "c", value: 3 },
  { cat: "d", value: 4 },
  { cat: "e", value: 6 }
];

DOM

<svg width=500 …>

</svg>

Data	DOM
values = [	<svg width=500 …>
{ cat: “a”, value: 5 },	
{ cat: “b”, value: 7 },	
{ cat: “c”, value: 3 },	? ? ? ? ?
{ cat: “d”, value: 4 },	
{ cat: “e”, value: 6 }	
];	</svg>

bars = svg.selectAll(“rect”) .data(values)

Data

values = [

{ cat: “a”, value: 5 }, ?

{ cat: “b”, value: 7 }, ?

{ cat: “c”, value: 3 }, ?

{ cat: “d”, value: 4 }, ?

{ cat: “e”, value: 6 } ?

];

DOM

<svg width=500 ...>

</svg>

bars = svg.selectAll(“rect”) .data(values)

Data Binding

Selections can ***bind*** **data and DOM elements**.

```
values = [ {…}, {…}, {…}, … ]; // input data as JS objects

// Select SVG rectangles and bind them to data values.
bars = svg.selectAll(“rect.bars”).data(values);

// What if the DOM elements don’t exist yet? The enter set represents data
// values that do not yet have matching DOM elements.
bars.enter().append(“rect”).attr(“class”, “bars”);
```

Data

```
values = [
  { cat: "a", value: 5  },   ?
  { cat: "b", value: 7 },    ?
  { cat: "c", value: 3 },    ?
  { cat: "d", value: 4 },    ?
  { cat: "e", value: 6 }     ?
];
```

DOM

```
<svg width=500 ...>

</svg>
```

```
bars = svg.selectAll("rect") .data(values)
```

Data

```
values = [
  { cat: “a”, value: 5  },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];
```

DOM

```
<svg width=500 …>
  <rect />
  <rect />
  <rect />
  <rect />
  <rect />
</svg>
```

```
bars.enter().append(“rect”)
```

Data	DOM
values = [	<svg width=500 ...>
{ cat: "a", value: 5 },	<rect class="bars" />
{ cat: "b", value: 7 },	<rect class="bars" />
{ cat: "c", value: 3 },	<rect class="bars" />
{ cat: "d", value: 4 },	<rect class="bars" />
{ cat: "e", value: 6 }	<rect class="bars" />
];	</svg>

```
bars.enter().append("rect").attr("class", "bars")
```

Data

```
values = [
  { cat: "a", value: 5  },
  { cat: "b", value: 7 },
  { cat: "c", value: 3 },
  { cat: "d", value: 4 },
  { cat: "e", value: 6 }
];
```

DOM

```
<svg width=500 ...>
  <rect x="..." />
  <rect x="..." />
  <rect x="..." />
  <rect x="..." />
  <rect x="..." />
</svg>
```

```
bars.enter().append("rect")
.attr("x", d => xscale(d.cat))
```

Data

```
values = [
  { cat: “a”, value: 5  },
  { cat: “b”, value: 7 },
  { cat: “c”, value: 3 },
  { cat: “d”, value: 4 },
  { cat: “e”, value: 6 }
];
```

DOM

```
<svg width=500 ...>
  <rect height=“...” />
  <rect height=“...” />
  <rect height=“...” />
  <rect height=“...” />
  <rect height=“...” />
</svg>
```

```
bars.enter().append(“rect”)
.attr(“height”, d => yscale(d.value))
```

Data Binding

Selections can ***bind* data and DOM elements**.

```
values = [ {…}, {…}, {…}, … ]; // input data as JS objects

// Select SVG rectangles and bind them to data values.
bars = svg.selectAll("rect.bars").data(values);

// What if the DOM elements don't exist yet? The enter set represents data
// values that do not yet have matching DOM elements.
bars.enter().append("rect").attr("class", "bars");

// What if data values are removed? The exit set is a selection of existing
// DOM elements who no longer have matching data values.
bars.exit().remove();
```

Data	DOM
values = [	<svg width=500 …>
{ cat: “a”, value: 5 },	<rect class=“bars” />
{ cat: “b”, value: 7 },	<rect class=“bars” />
{ cat: “c”, value: 3 },	<rect class=“bars” />
{ cat: “d”, value: 4 },	<rect class=“bars” />
{ cat: “e”, value: 6 }	<rect class=“bars” />
];	</svg>

Data

```
values = [
  { cat: "a", value: 5  },
  { cat: "b", value: 7 },
  { cat: "c", value: 3 },
  { cat: "d", value: 4 },
  { cat: "e", value: 6 }
];
```

DOM

```
<svg width=500 ...>
  <rect class="bars" />
  <rect class="bars" />
  <rect class="bars" />
  <rect class="bars" />
  <rect class="bars" />
</svg>
```

```
values.filter(d => !['b', 'd'].includes(d.cat))
```

Data

```
values = [
  { cat: “a”, value: 5  },

  { cat: “c”, value: 3 },

  { cat: “e”, value: 6 }
];
```

DOM

```
<svg width=500 ...>
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
</svg>
```

```
bars = svg.selectAll(“rect.bars”).data(values)
```

Data

values = [
{ cat: “a”, value: 5 },
{ cat: “c”, value: 3 },
{ cat: “e”, value: 6 }
];

DOM

<svg width=500 ...>
<rect class=“bars” />
<rect class=“bars” />
<rect class=“bars” />
<rect class=“bars” />
<rect class=“bars” />
</svg>

bars.exit()

Data

```
values = [
  { cat: “a”, value: 5  },

  { cat: “c”, value: 3 },

  { cat: “e”, value: 6 }
];
```

DOM

```
<svg width=500 ...>
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
</svg>
```

```
bars.exit().remove()
```

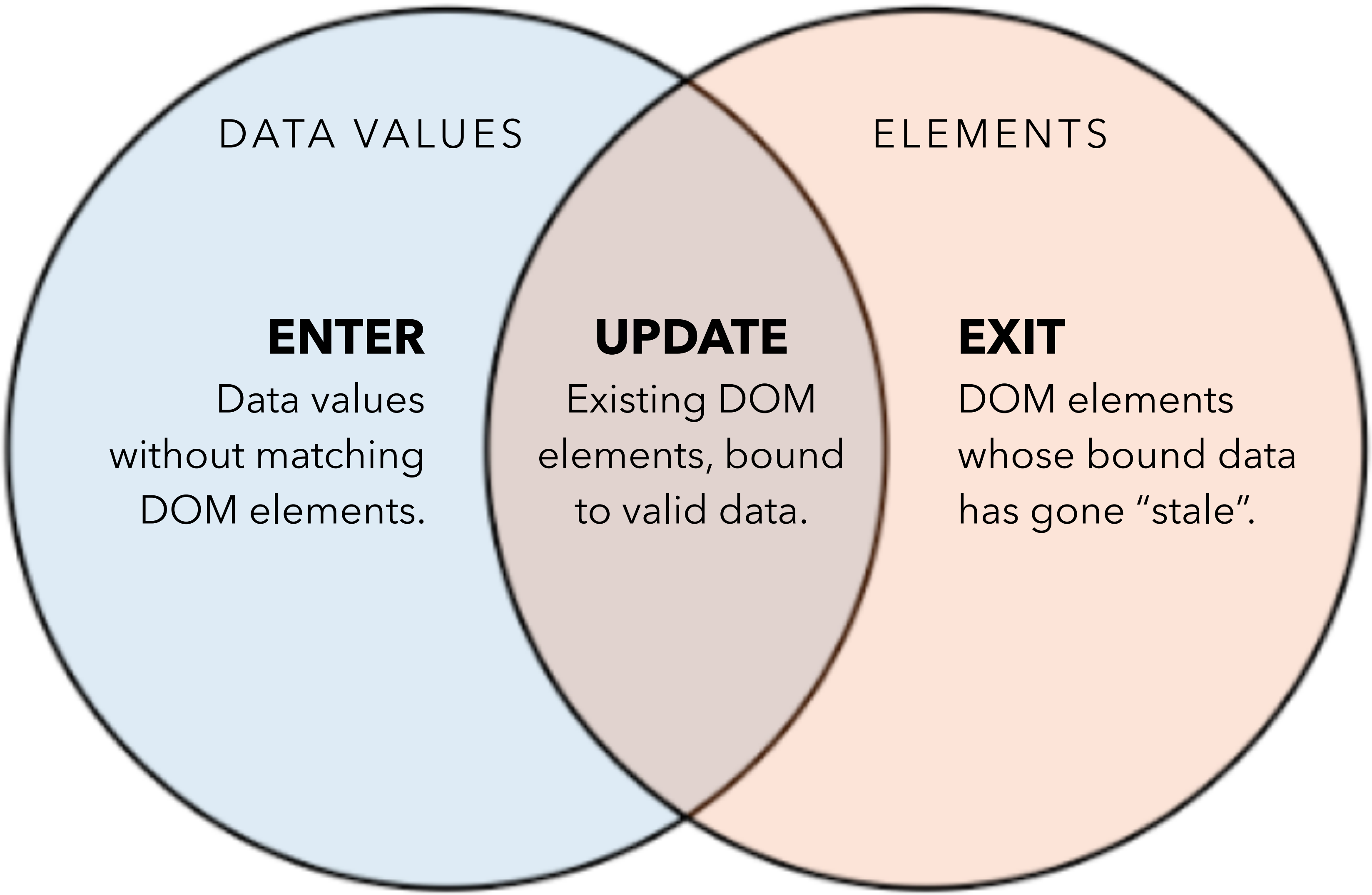
Data

```
values = [
  { cat: “a”, value: 5  },
  { cat: “c”, value: 3 },
  { cat: “e”, value: 6 }
];
```

DOM

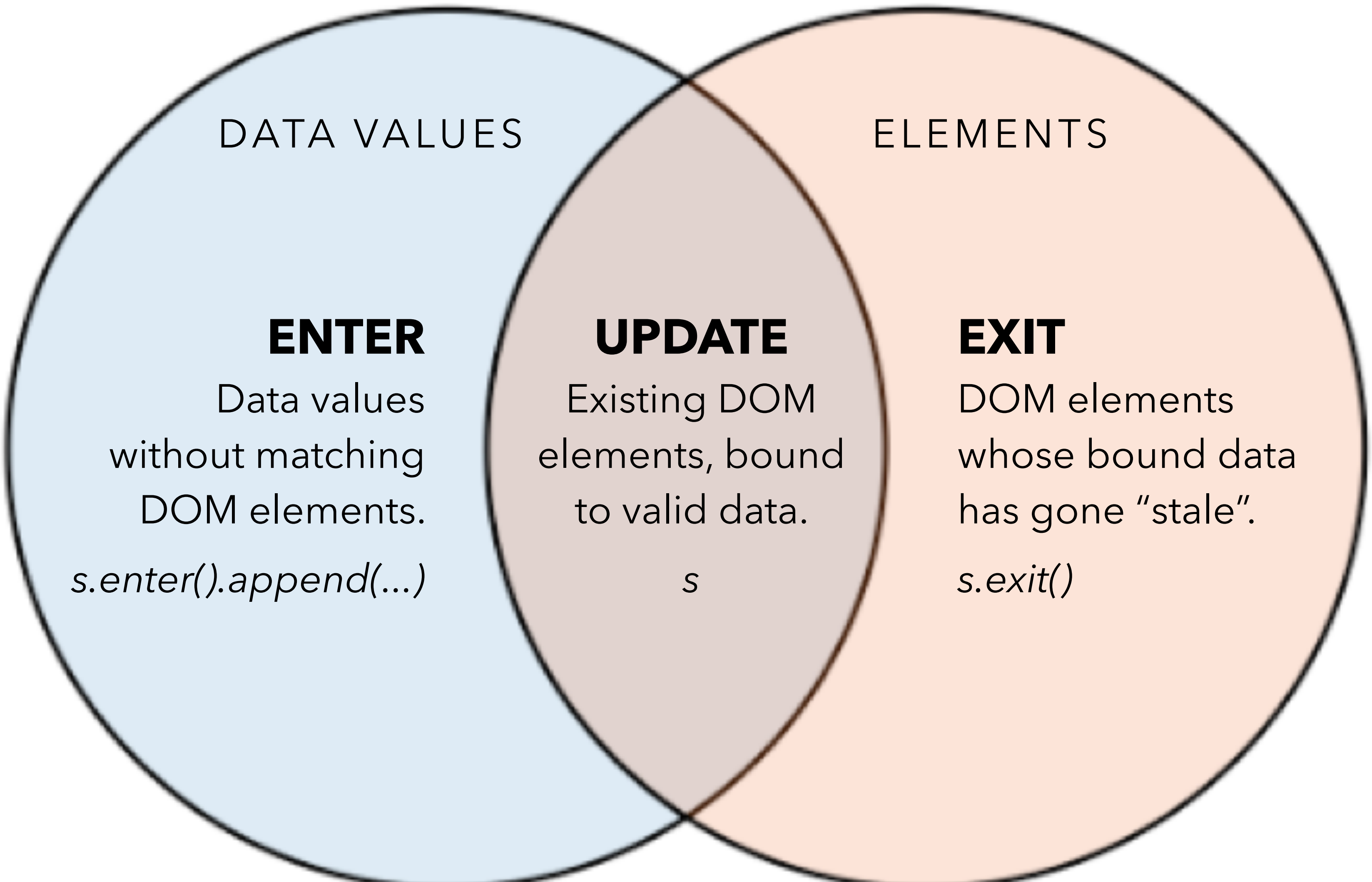
```
<svg width=500 …>
  <rect class=“bars” />
  <rect class=“bars” />
  <rect class=“bars” />
</svg>
```

The Data Join



The Data Join

var s = d3.selectAll(...).data(...)



Data Binding

Selections can ***bind*** **data and DOM elements**.

```
values = [ {…}, {…}, {…}, … ]; // input data as JS objects

// Select SVG rectangles and bind them to data values.
bars = svg.selectAll("rect.bars").data(values)
    .join(
        enter => enter.append("rect"), // create new
        update => update,              // update current
        exit => exit.remove()          // remove outdated
    )
```

D3 Modules

Data Parsing / Formatting (JSON, CSV, …)
Shape Helpers (arcs, curves, areas, symbols, …)
Scale Transforms (linear, log, ordinal, …)
Color Spaces (RGB, HSL, LAB, …)
Animated Transitions (tweening, easing, …)
Geographic Mapping (projections, clipping, …)
Layout Algorithms (stack, pie, force, trees, …)
Interactive Behaviors (brush, zoom, drag, …)

Many of these correspond to future lecture topics!

Ease-of-Use

Expressiveness

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Administrivia

A2 Peer Reviews

You have been assigned two peer A2 submissions to review. For each:

- Try to determine which is earnest and which is deceptive
- Share a rationale for how you made this determination
- Share feedback using the "I Like / I Wish / What If" rubric

Assigned reviews will be posted on the A2 Peer Review page on Canvas, along with a link to a Google Form. You should submit two forms: one for each A2 peer review.

Due by **Tue 4/23 11:59pm**.

I Like… / I Wish… / What If?

I LIKE…

Praise for design ideas and/or well-executed implementation details. *Example: "I like the navigation through time via the slider; the patterns observed as one moves forward are compelling!"*

I WISH…

Constructive statements on how the design might be improved or further refined. *Example: "I wish moving the slider caused the visualization to update immediately, rather than the current lag."*

WHAT IF?

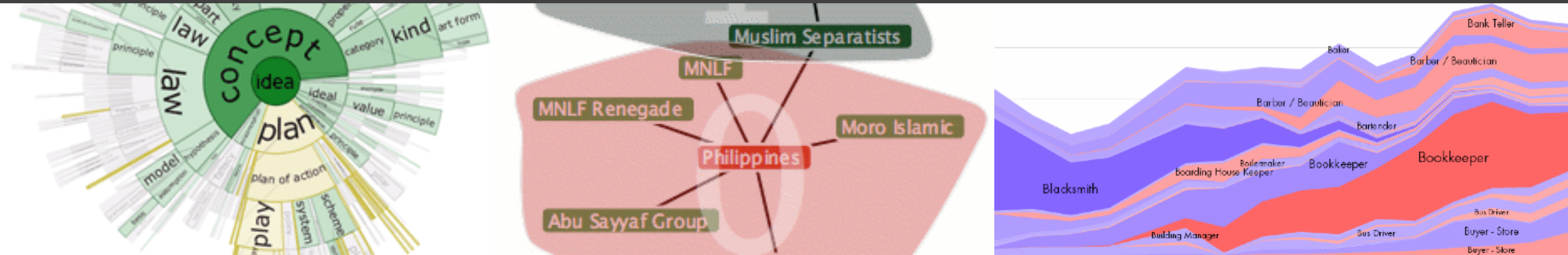
Suggest alternative design directions, or even wacky half-baked ideas. *Example: "What if we got rid of the slider and enabled direct manipulation navigation by dragging data points directly?"*

A3: Interactive Prototype

Create an interactive visualization. Choose a driving question for a dataset and develop an appropriate visualization + interaction techniques, then deploy your visualization on the web.

Due by *11:59pm* on **Monday, May 6**.

Work in project teams of 3-4 people.



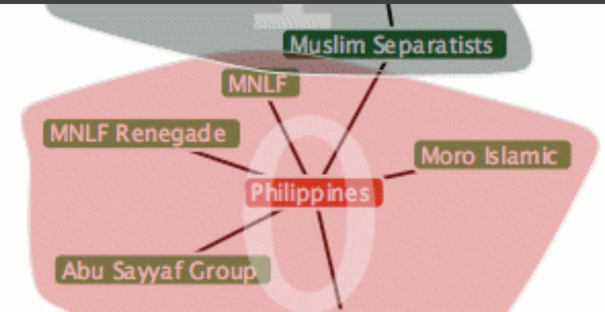
Form A3 + Final Project Team

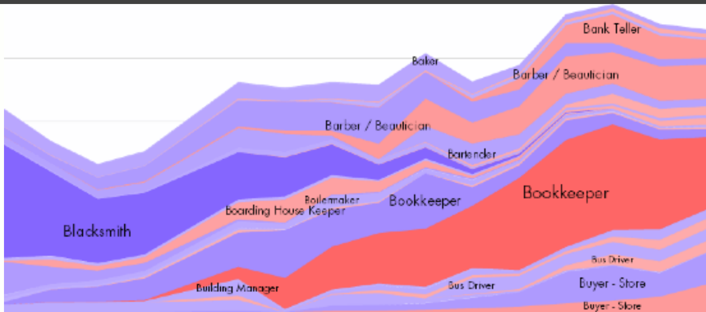
Form a **team of 3-4** for A3 and the Final Project.

Submit signup form by **Thu 4/25, 11:59pm**.

If you do not have team mates, post on Ed about your interests/skills/project ideas!

We will send out a reminder early next week.



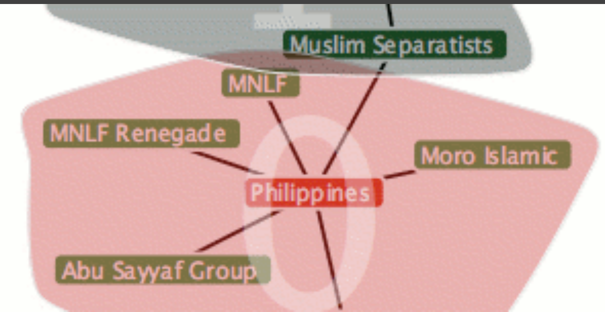


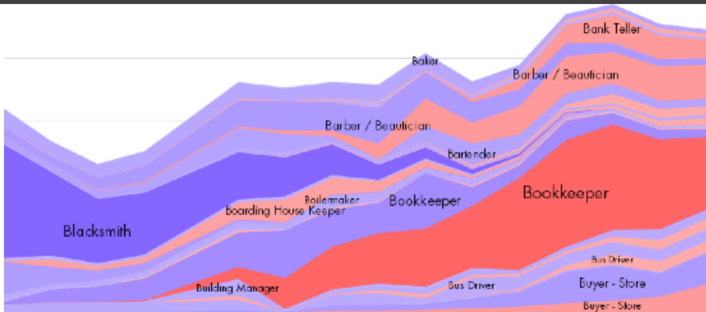
Requirements

Interactive. You must implement interaction methods! However, this is not only selection / filtering / tooltips. Also consider annotations or other narrative features to draw attention and provide additional context

Web-based. D3/Vega-Lite are encouraged, but not required. Deploy to web using GitHub pages.

Write-up. Provide design rationale.



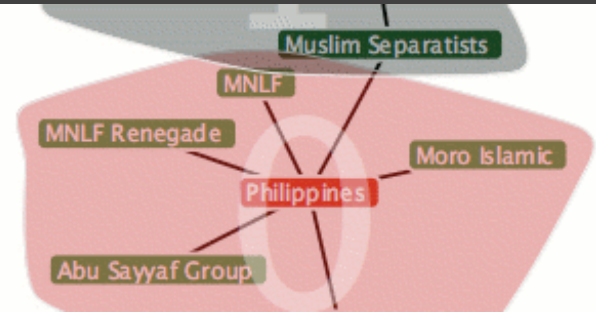


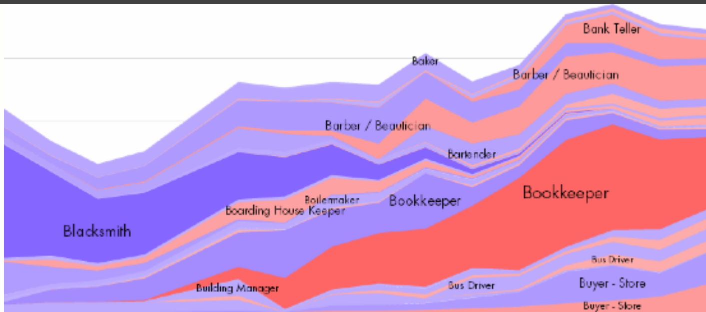
Interactive Prototype Tips

Start now. It will take longer than you think.

Keep it simple. Choose a *minimal* set of interactions that enables users to explore and generate interesting insights. Do not feel obligated to convey *everything* about the data: focus on a compelling subset.

Promote engagement. How do your chosen interactions reveal interesting observations?





D3 Tutorial - In Class Thu Apr 25

D3.js Deep Dive led by Madeleine and Luke

Be sure to read the D3, Part 1 notebook ahead of time. We'll work through Part 2 in class.

Bring your laptops and follow along in real-time.

Web Publishing Tutorial

On Zoom, led by Josh

Gain skills publishing projects to the web:

- Publish sites using GitLab pages

- Export Altair visualizations to HTML

- Learn dashboard publishing tools

A Visualization Tool Stack

Chart Typologies

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Chart Typologies

Excel, Many Eyes, Google Charts

Charting Tools

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Declarative Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Canvas, OpenGL, Processing

Programming Toolkits

Chart Typologies

Excel, Many Eyes, Google Charts

Charting Tools

Visual Analysis Grammars

VizQL, ggplot2, Vega-Lite

Visualization Grammars

D3.js, Vega

Declarative Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Programming Toolkits

Graphics APIs

Canvas, OpenGL, Processing

What is a Declarative Language?

Programming by describing *what*, not *how*

Separate **specification** (*what you want*) from **execution** (*how it should be computed*)

In contrast to **imperative programming**, where you must give explicit steps.

What is a Declarative Language?

Programming by describing *what*, not *how*

Separate **specification** (*what you want*) from **execution** (*how it should be computed*)

In contrast to **imperative programming**, where you must give explicit steps.

```
d3.selectAll("rect")
  .data(my_data)
 .join("rect")
  .attr("x", d => xscale(d.foo))
  .attr("y", d => yscale(d.bar))
```

TODAY'S PAPER | VIDEO | MOST POPULAR | TIMES TOPICS | MOST RECENT

Subscribe to The Times | Welcome, jmheer

The New York Times

Tuesday, October 26, 2010 Last Update: 3:50 PM ET

Search | ING DIRECT | Subscribe to Hom

2010 Midterm Elections

Tea Party Vow to Deter Voter Fraud Is Called Scare Tactic

By IAN URBINA 2:19 PM ET

Voting rights group say that Tea Party members' plan to question voters' eligibility at the polls is intended to suppress minority and poor voters.

Post a Comment | Read (355)

Glaxo Pays $750 Million Fine for Tainted Products

By GARDINER HARRIS and DUFF

Joshua Kristal for The New York Times

Painting at 99, With No Compromises

By ROBIN FINN

An exhibition celebrating Will Barnet's centennial year traces his evolution as a modern American artist.

OPINION »

OP-ED CONTRIBUTOR

Humans to Asteroids: Watch Out!

How to keep near-Earth objects from hitting us.

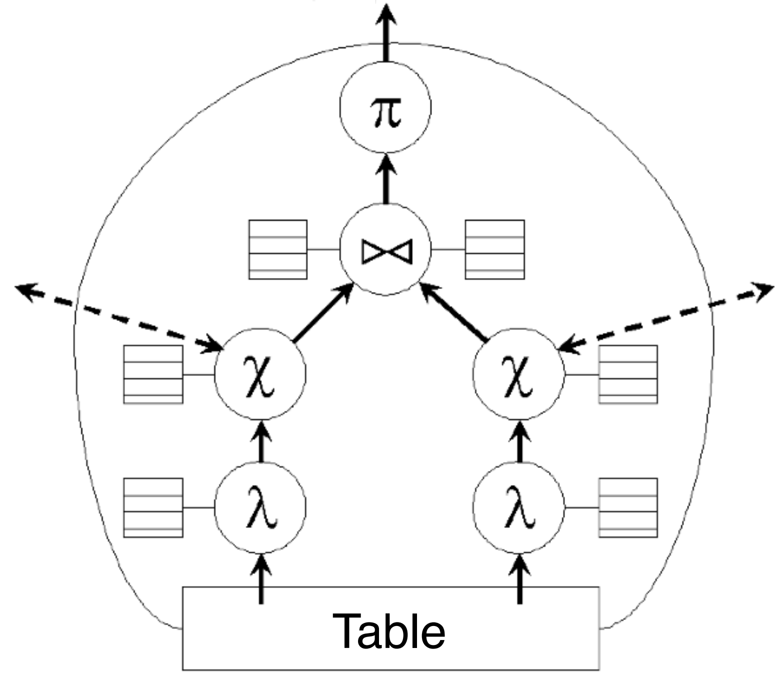
- Brooks: No Second Thoughts | Comments (200)
- Herbert: The Corrosion of America
- Cohen: Turkey Steps Out
- Editorial: Mortgage Mess
- Bloggingheads: Jon Stewart's Power

MARKETS » At 3:56 PM ET

S.&P. 500 | Dow | Nasdaq

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<!--[if IE]><![endif]-->
▼<html>
 ▶<head>…</head>
 ▼<body id="home" style="visibility: visible; ">
   <script src="http://connect.facebook.net/en_US/all.js"></scrip
   <div id="fb-root"></div>
   <a name="top"></a>
  ▼<div id="shell">
    ▶<ul id="memberTools">…</ul>
     <!-- ADXINFO classification="text_ad" campaign="nyt2010-circ-
    ▶<div class="tabsContainer">…</div>
     <!-- close .tabsContainer -->
    ▶<div id="page" class="tabContent active">…</div>
     <!--close page -->
   </div>
   <!--close shell -->
 ▶<script type="text/javascript" language="JavaScript">…</script
   <img src="/adx/bin/clientside/
   1b1a1755Q2F,.tWdC,KiQ2BCQ5C6iSM.KCYQ2FMadYYQ24xN!Q2B!xQ24WN4aN
   <script src="http://graphics8.nytimes.com/js/test/wto/wt_capi.
   charset="utf-8"></script>
   <span id="wtoScript"></span>
 ▶<script type="text/javascript">…</script>
   <img src="http://graphics8.nytimes.com/ads/blank.gif">
   <script type="text/javascript" src="http://graphics8.nytimes.c
```

HTML / CSS




```sql
SELECT customer_id, customer_name,
    COUNT(order_id) as total
FROM customers
INNER JOIN orders ON
    customers.customer_id
    = orders.customer_id
GROUP BY customer_id, customer_nam
HAVING COUNT(order_id) > 5
ORDER BY COUNT(order_id) DESC
```

SQL

Why Declarative Languages?

Faster iteration, less code, larger user base?

Better visualization. *Smart defaults.*

Reuse. *Write-once, then re-apply.*

Performance. *Optimization, scalability.*

Portability. *Multiple devices, renderers, inputs.*

Programmatic generation.
Write programs which output visualizations.
Automated search & recommendation.

Chart Typologies

Excel, Many Eyes, Google Charts

Charting Tools

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Visualization Grammars

D3.js, ***Vega***

Declarative Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Programming Toolkits

Graphics APIs

Processing, OpenGL, Java2D

Chart Typologies **?!** Charting Tools

Excel, Many Eyes, Google Charts

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Declarative Languages

Visualization Grammars

D3.js, ***Vega***

Component Architectures Programming Toolkits

Prefuse, Flare, Improvise, VTK

Graphics APIs

Processing, OpenGL, Java2D

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Visualization Grammars

D3.js, ***Vega***

Declarative Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Graphics APIs

Processing, OpenGL, Java2D

Programming Toolkits

Interactive Data Exploration

Tableau, ***Lyra, Voyager***

Graphical Interfaces

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Visualization Grammars

D3.js, ***Vega***

Declarative Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Programming Toolkits

Graphics APIs

Processing, OpenGL, Java2D

UW Interactive Data Lab PROJECTS

The Lyra Visualization Design Environment (VDE) alpha

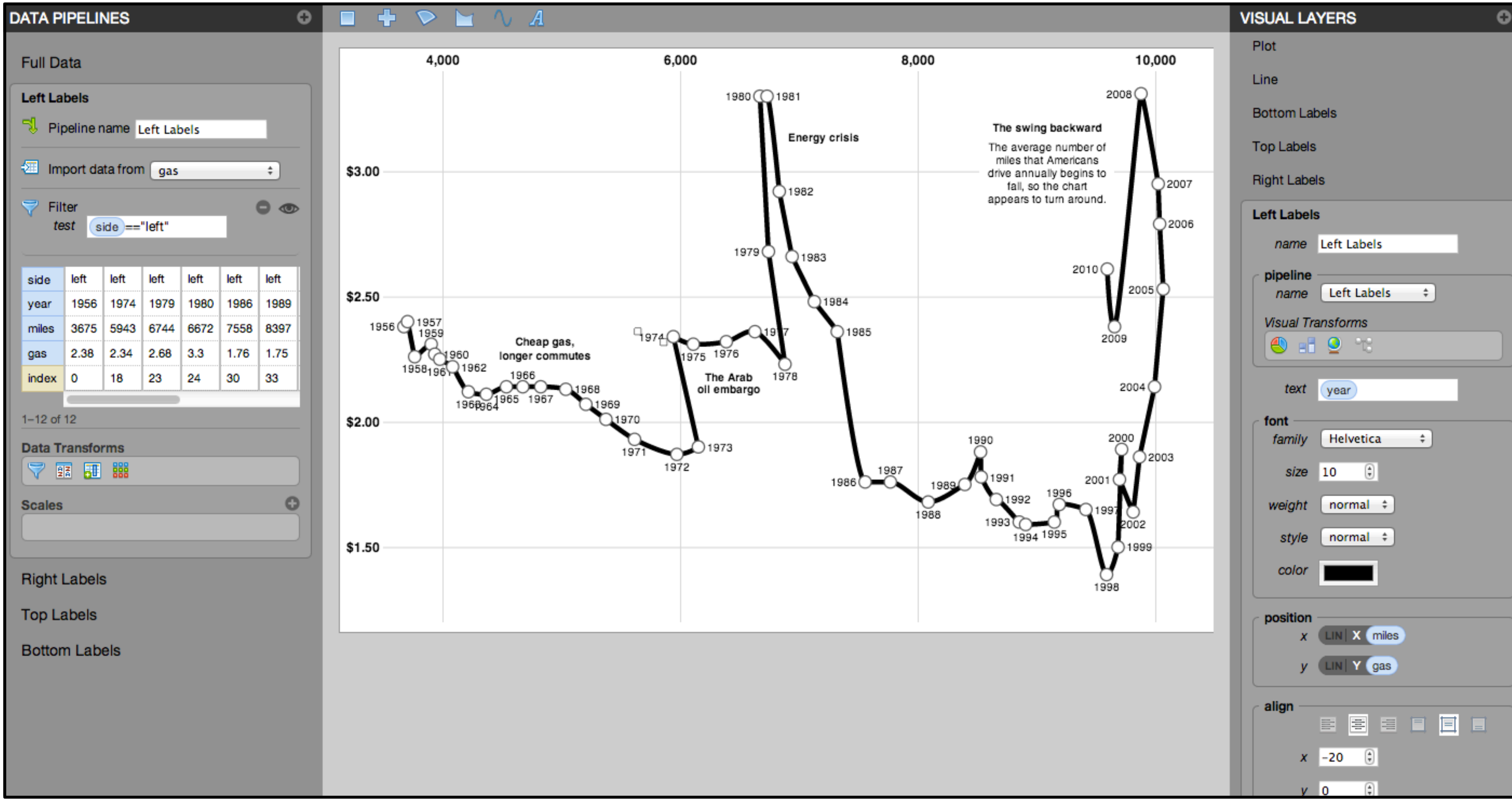
Arvind Satyanarayan, Kanit "Ham" Wongsuphasawat, Jeffrey Heer

William Playfair's classic chart comparing the price of wheat and wages in England recreated in the Lyra VDE.

idl.cs.washington.edu/projects/lyra

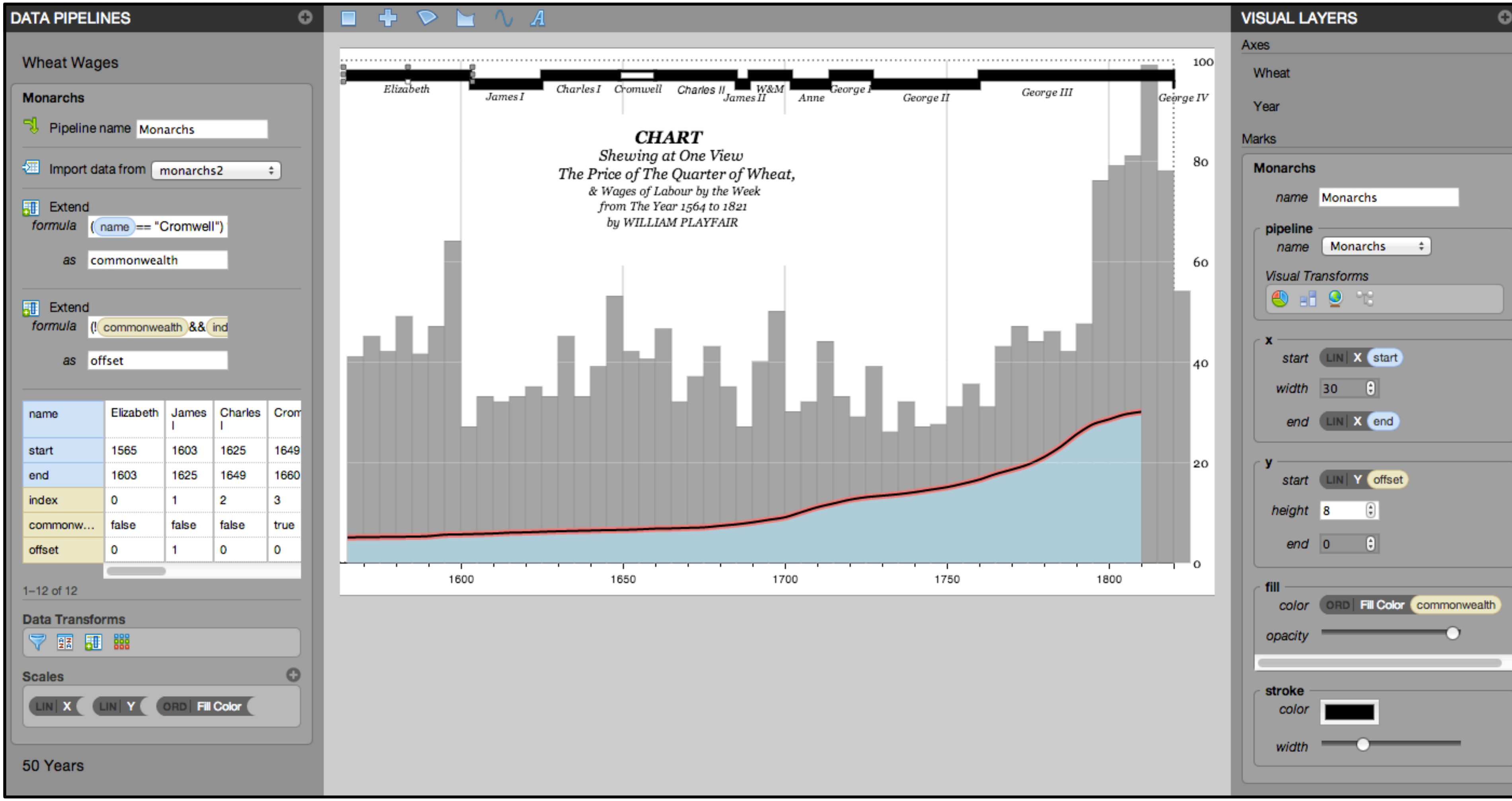
See also: Charticulator, Data Illustrator

Lyra A Visualization Design Environment



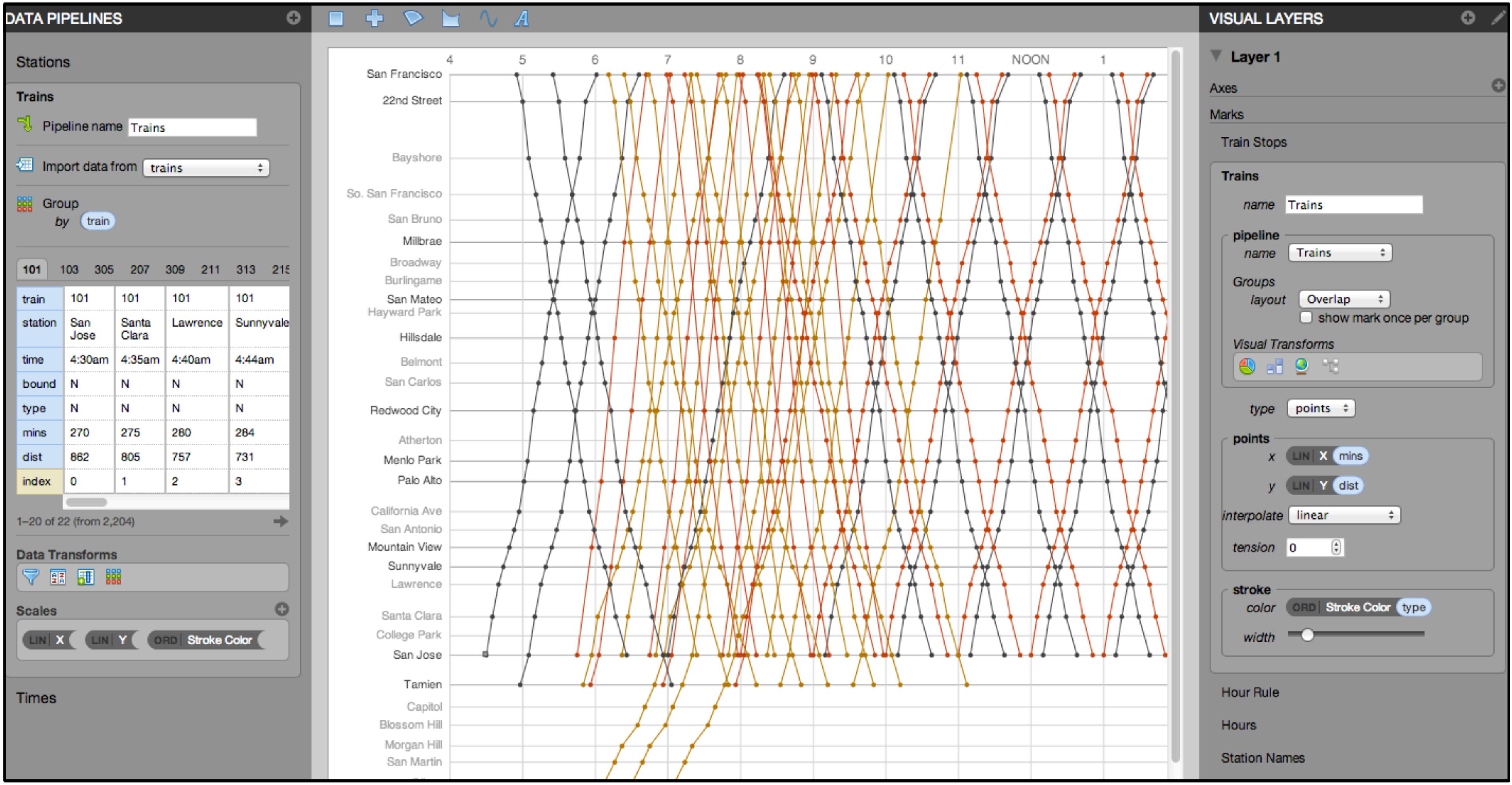
Driving Shifts into Reverse by Hannah Fairfield, NYTimes

Lyra A Visualization Design Environment



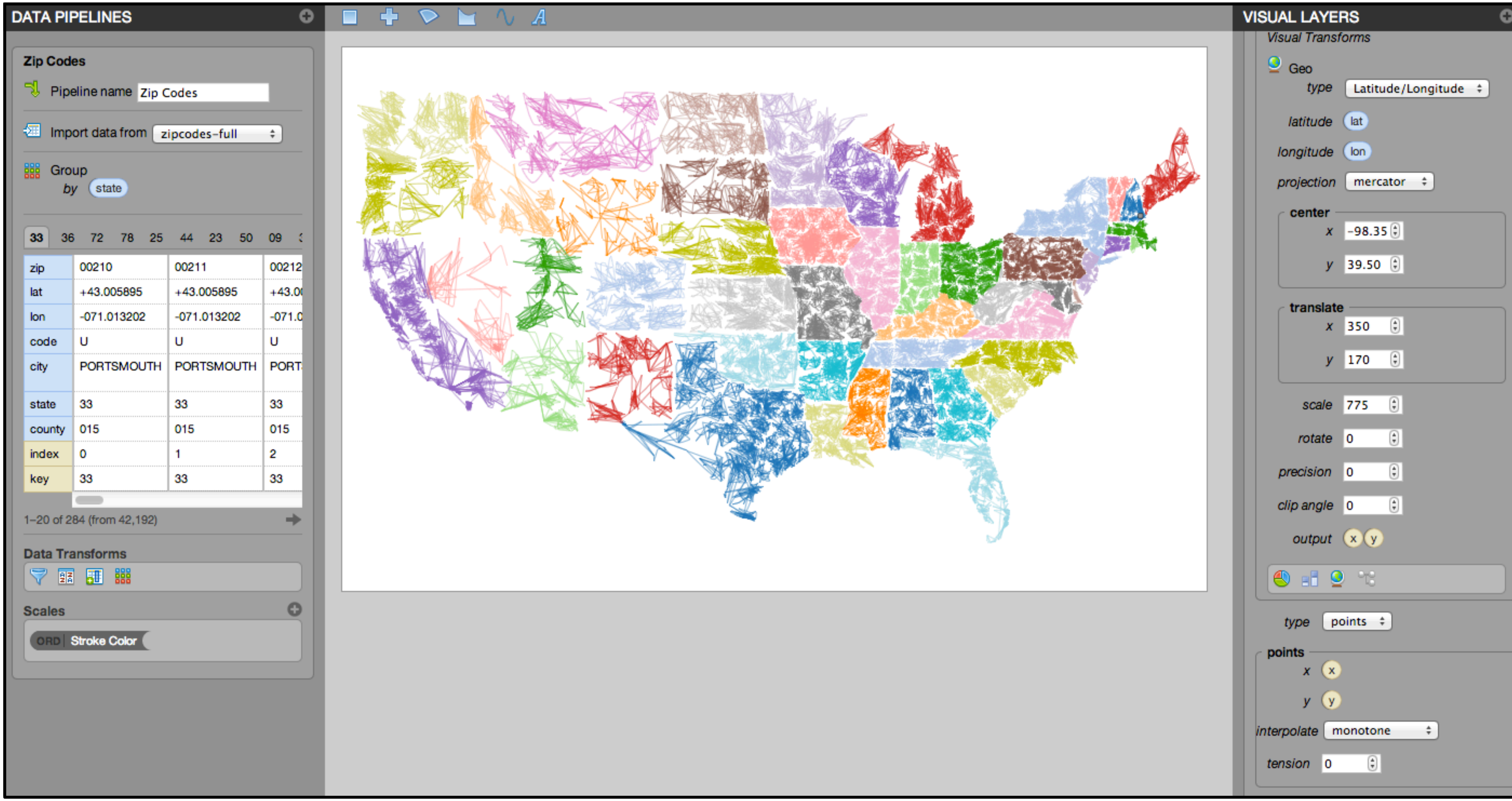
by William Playfair

Lyra A Visualization Design Environment



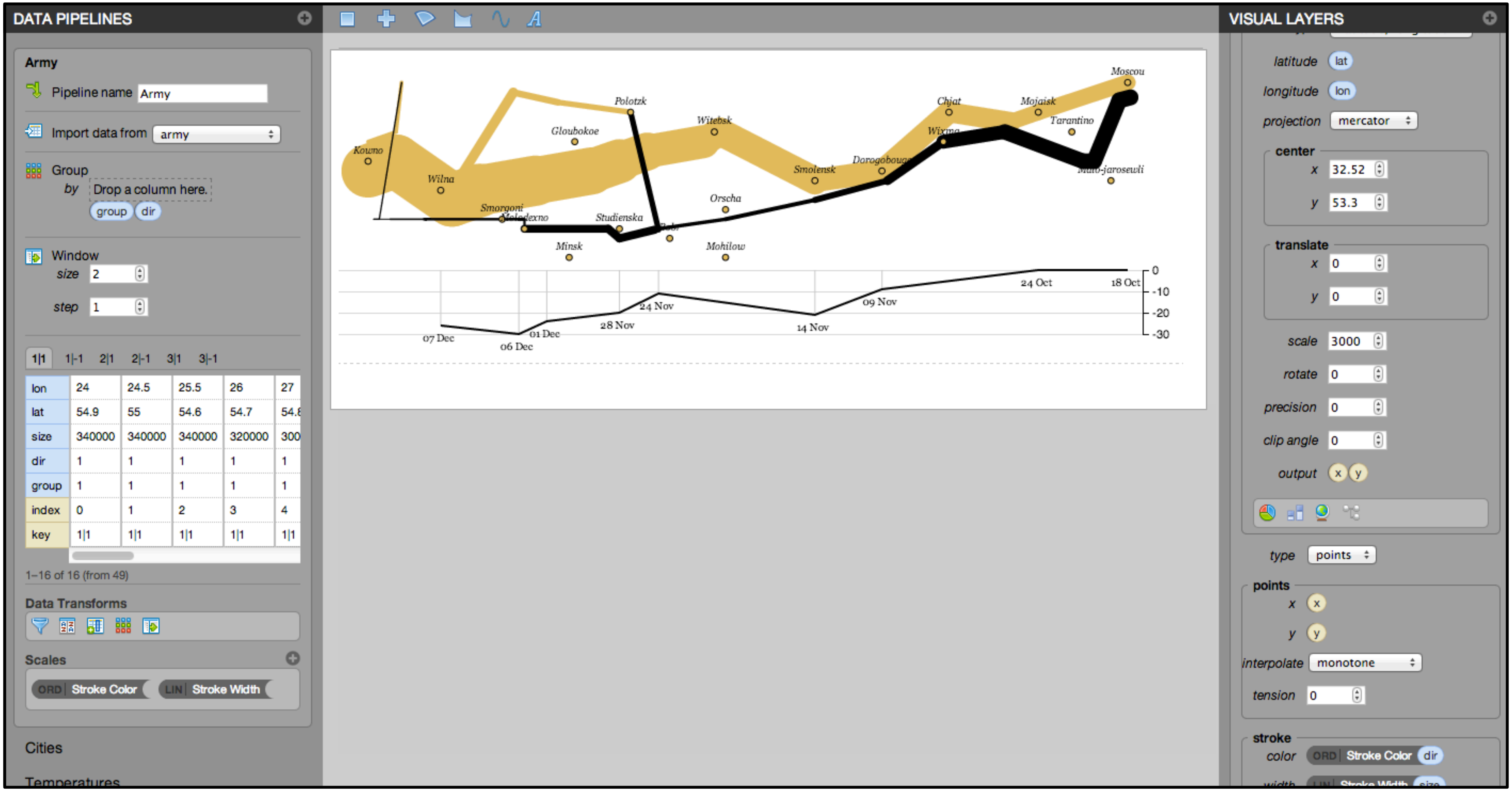
based on the **Railway Timetable** by E. J. Marey

Lyra A Visualization Design Environment

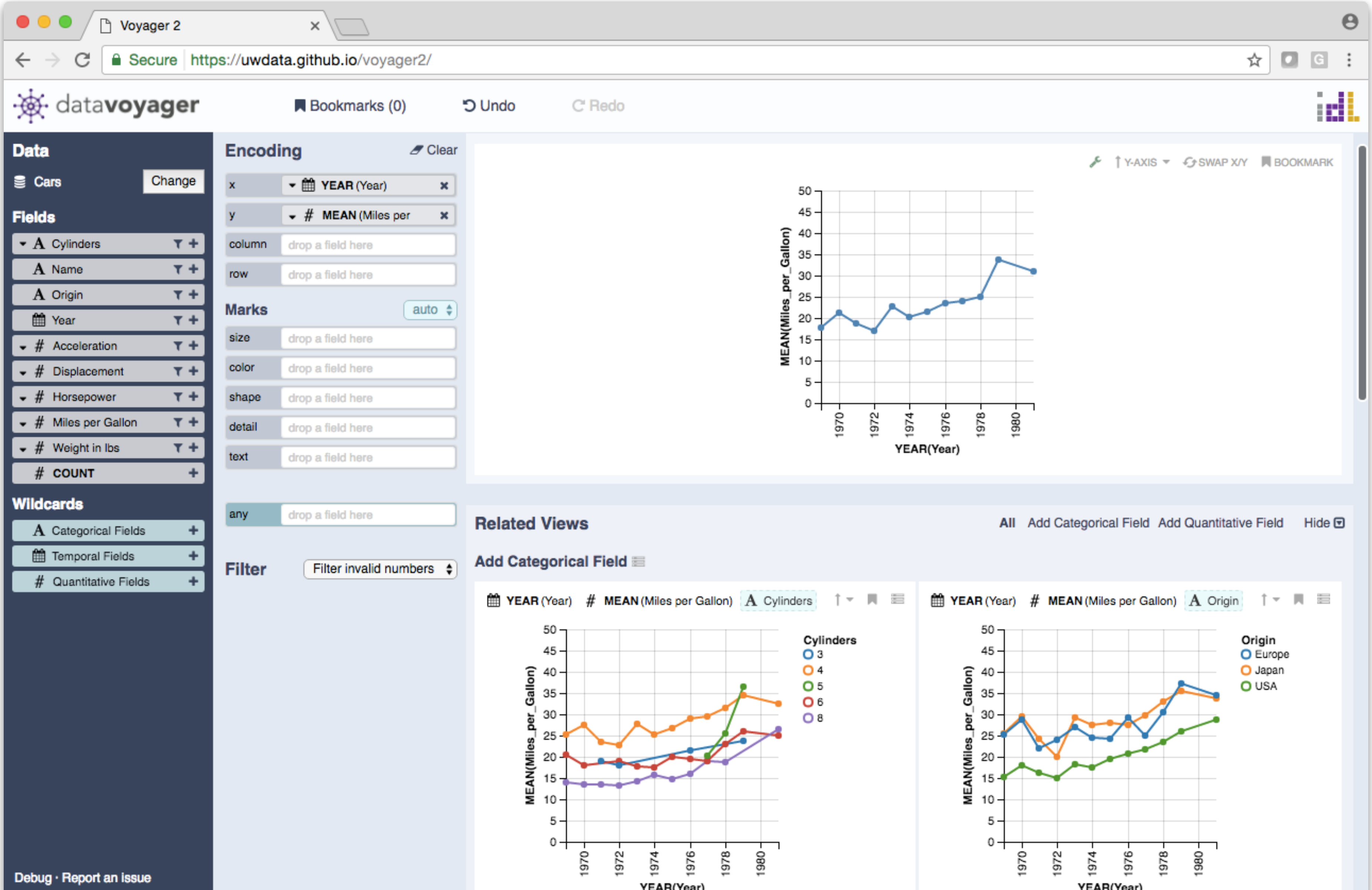


ZipScribble by Robert Kosara

Lyra A Visualization Design Environment



Napoleon's March by Charles Minard



Voyager. Wongsuphasawat et al. *InfoVis'15, CHI'17*

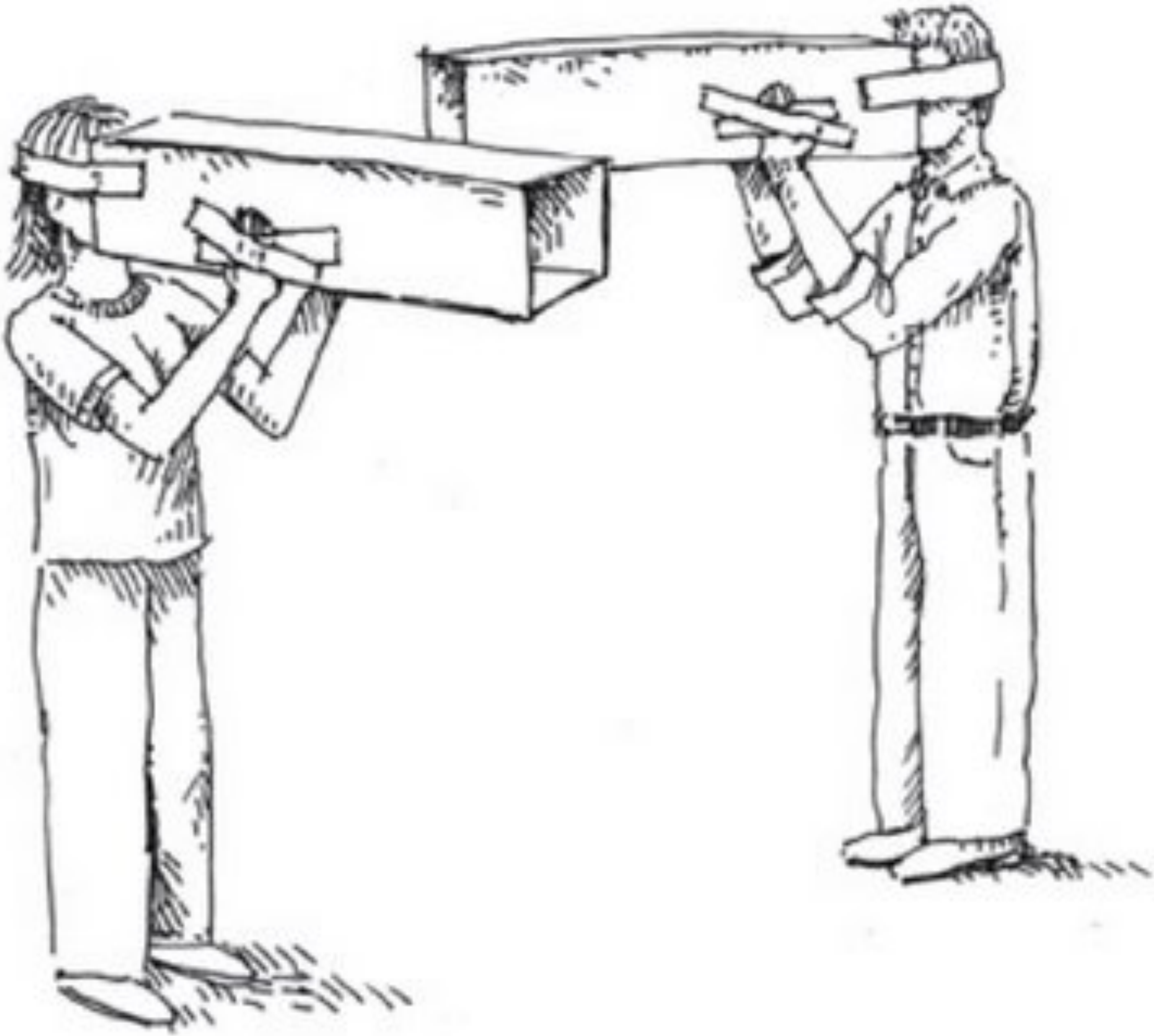
Common exploration pitfalls:

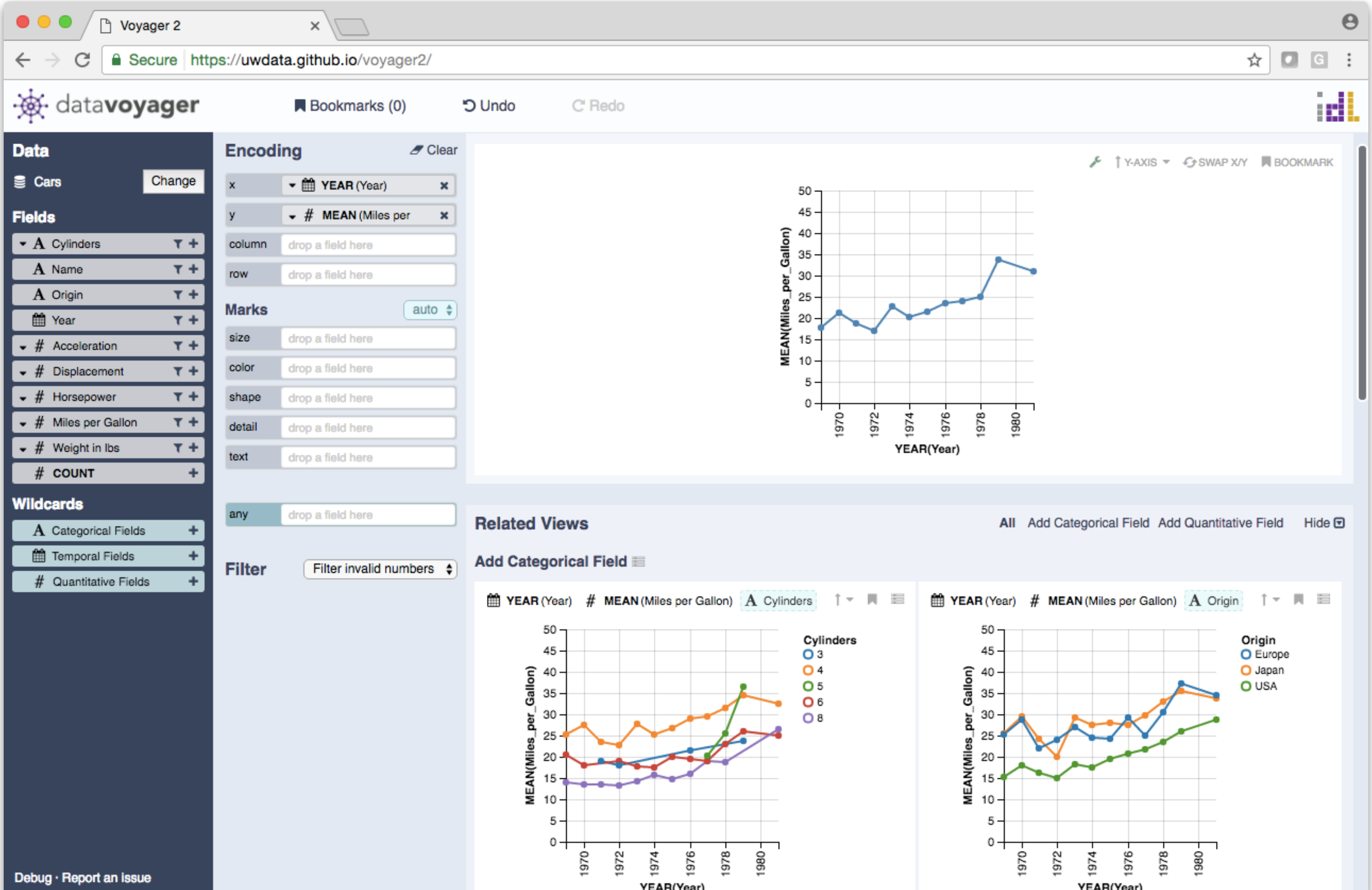
Overlook data quality issues

Fixate on specific relationships

Plus many other biases…

[Heuer 1999, Kahneman 2011, …]





Voyager. Wongsuphasawat et al. *InfoVis'15, CHI'17*

Key Idea: Augment manual exploration with visualization recommendations sensitive to the user's current focus.

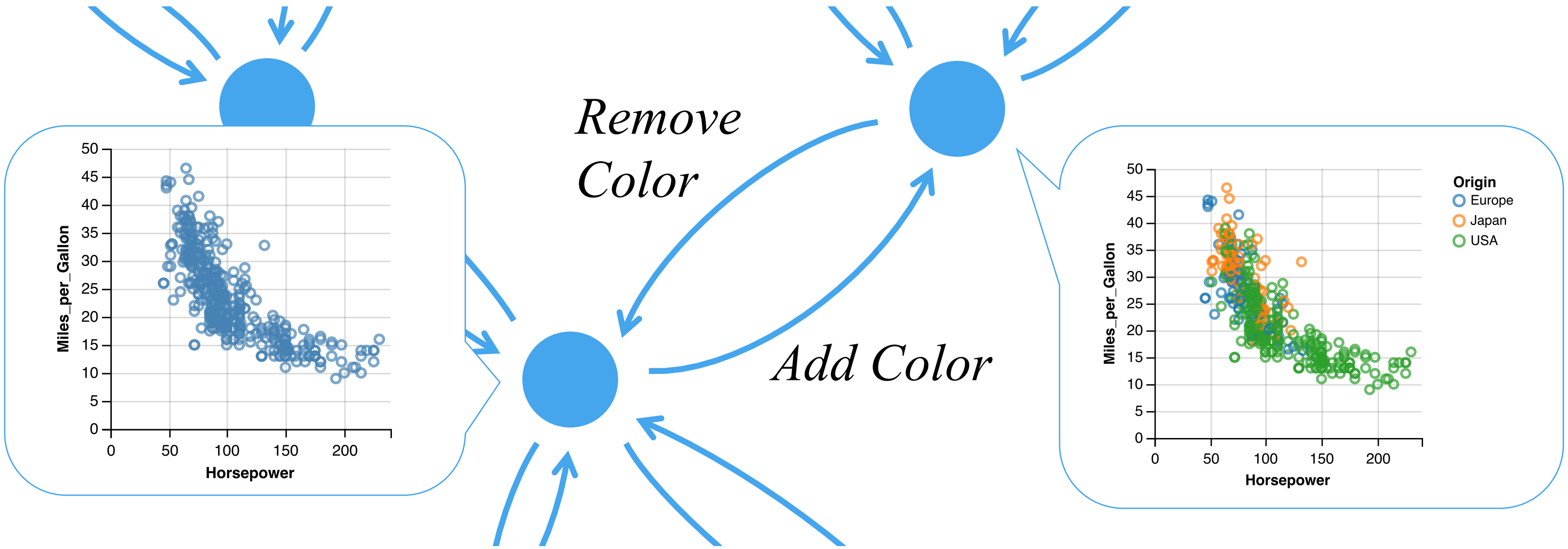
The goal is to support *systematic consideration* of the data, without exacerbating *false discovery*.

To model a user's search frontier, we *enumerate related Vega-Lite specifications*, seeded by the user's current focus.

Candidate charts are pruned and ranked using models of estimated *perceptual effectiveness*.

Voyager. Wongsuphasawat et al. *InfoVis'15, CHI'17*

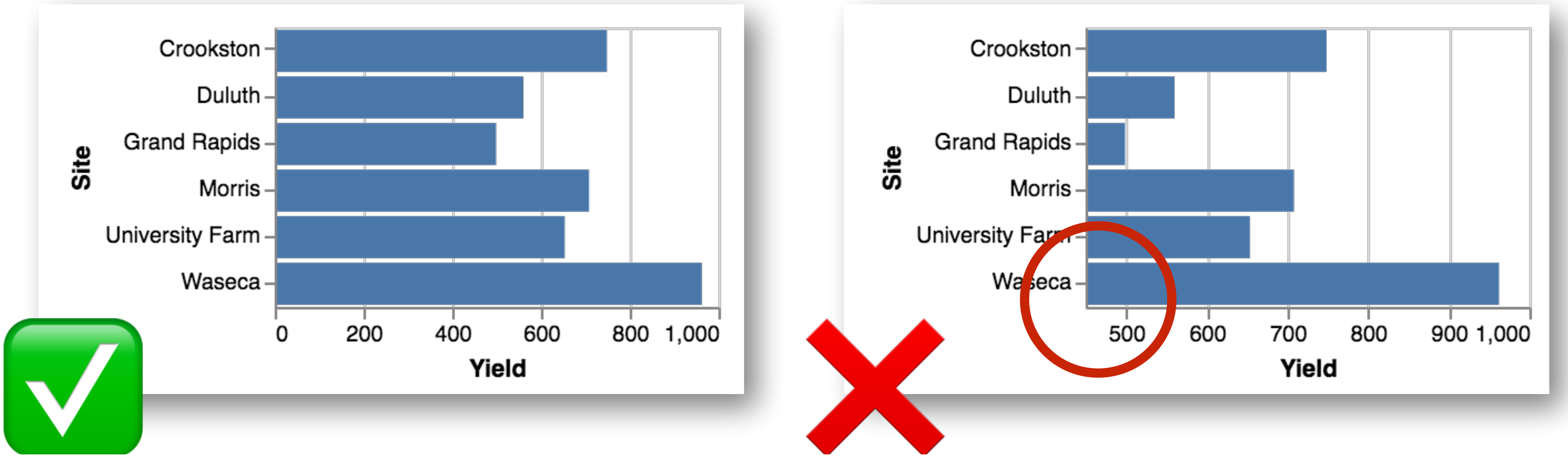
A Formal Design Space of Visualizations



Enumerate **Vega-Lite specifications** and transformations among them. Search the space using logic programming methods.

[Kim et al. 2017]

Articulate Design Constraints



"Quantitative axes should include a zero baseline"

When and how strongly should we apply this?

How to balance with other such constraints?

[Moritz et al. 2019]

Learn Design Trade-Offs from Data

Training Data

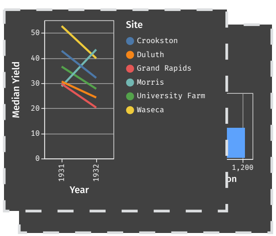
Pairs of Ranked Visualizations

Features

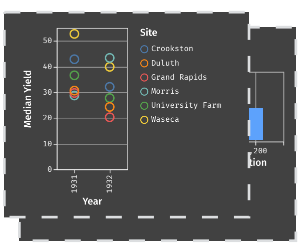
Violations of Design Constraints

Learning Algorithm

Learning to Rank with Linear SVM



👍 Positive example
$[u_1, u_2, ..., u_k]$



👎 Negative example
$[v_1, v_2, ..., v_k]$

v_i: the number of violations of constraint i.

$$\arg\max_w \sum_{i \in 0 \ldots k} w_i (u_i - v_i)$$

w is the weight vector of the soft constraints

[Moritz et al. 2019]

Compared to other tools, **over 4x more variable sets seen**, and **over 2x more interacted with**.

“related view suggestion accelerates exploration a lot.”

“I like that it shows me what fields to include in order to see a specific graph. Otherwise, I have to do a lot of trial and error and can't express what I wanted to see.”

“These related views are so good but it’s also spoiling that I start thinking less. I’m not sure if that’s really a good thing.”

Voyager. Wongsuphasawat et al. *InfoVis’15, CHI’17*

Interactive Data Exploration

Tableau, ***Lyra, Voyager***

Graphical Interfaces

Visual Analysis Grammars

VizQL, ggplot2, ***Vega-Lite***

Visualization Grammars

D3.js, ***Vega***

Declarative Languages

Component Architectures

Prefuse, Flare, Improvise, VTK

Programming Toolkits

Graphics APIs

Processing, OpenGL, Java2D