

Visualization in Practice

Laura Garrison, University of Bergen
laura.garrison@uib.no

*ICTP Workshop 2022
2. December 2022*



What?

Datasets

→ Data Types

- Items
- Attributes
- Links
- Positions
- Grids

→ Data and Dataset Types

Tables	Networks & Trees	Fields	Geometry	Clusters, Sets, Lists
Items	Items (nodes)	Grids	Items	Items
Attributes	Links	Positions	Positions	

Attributes

→ Attribute Types

- Categorical



- Ordered

→ *Ordinal*

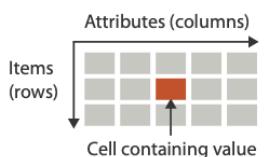


- Quantitative

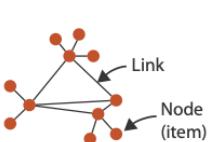


→ Dataset Types

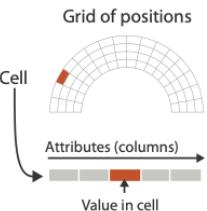
→ Tables



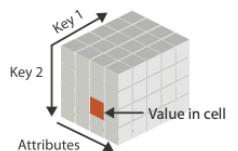
→ Networks



→ Fields (Continuous)



→ Multidimensional Table



→ Trees



→ Geometry (Spatial)



→ Ordering Direction

- Sequential



- Diverging



- Cyclic



→ Dataset Availability

→ Static



→ Dynamic



What?

Why?

Actions

Targets

→ Analyze

→ Consume

- Discover
- Present
- Enjoy

→ Produce

- Annotate
- Record
- Derive

→ Search

	Target known	Target unknown
Location known	<i>Lookup</i>	<i>Browse</i>
Location unknown	<i>Locate</i>	<i>Explore</i>

→ Query

→ Identify → Compare → Summarize

-
-
-

→ All Data

- Trends
- Outliers
- Features

→ Attributes

- One
- Many
- Distribution
- Extremes
- Dependency
- Correlation
- Similarity

→ Network Data

- Topology
- Paths
-

→ Spatial Data

- Shape

What?

Why?

How?

What?

Why?

How?

Encode

- Ⓐ Arrange
 - Express
 - Separate
- Order
- Align
- Use

- Ⓐ Map from **categorical** and **ordered** attributes
 - Color
 - Hue
 - Saturation
 - Luminance
 - Size, Angle, Curvature, ...
- Shape
 - + ● ■ ▲
- Motion
 - Direction, Rate, Frequency, ...

Manipulate

- Ⓐ Change
- Ⓐ Select
- Ⓐ Navigate

Facet

- Ⓐ Juxtapose
- Ⓐ Partition
- Ⓐ Superimpose

Reduce

- Ⓐ Filter
- Ⓐ Aggregate
- Ⓐ Embed

What?

Why?

How?

Data abstraction -> Task abstraction ->
Visual + Interaction Encoding



Data representation has an impact on interpretation

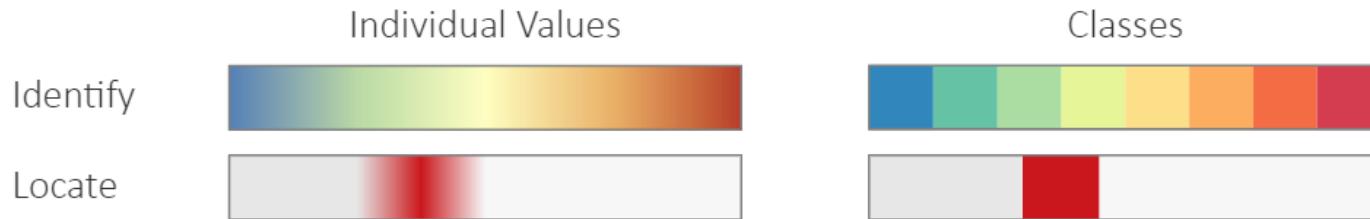
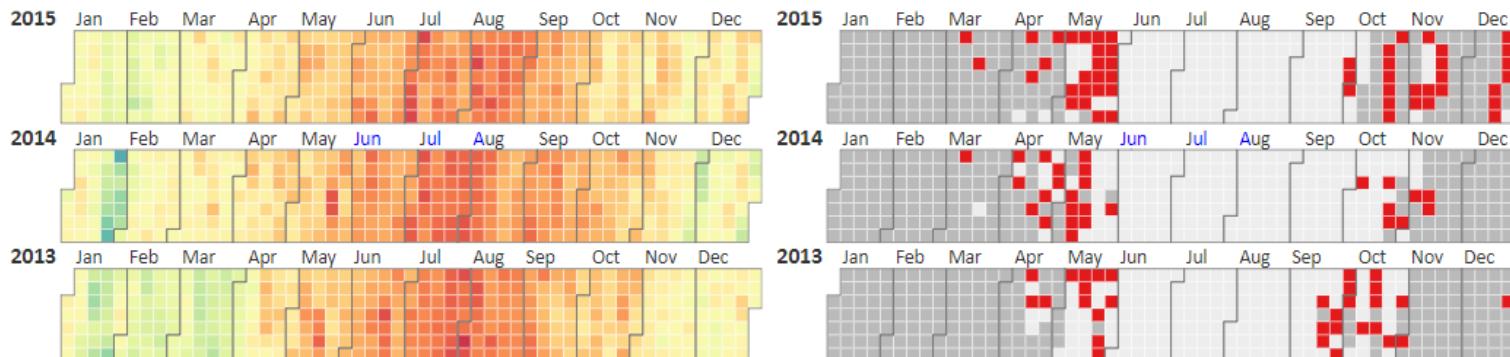


Figure 3.4 Color maps for identifying and locating values and classes.

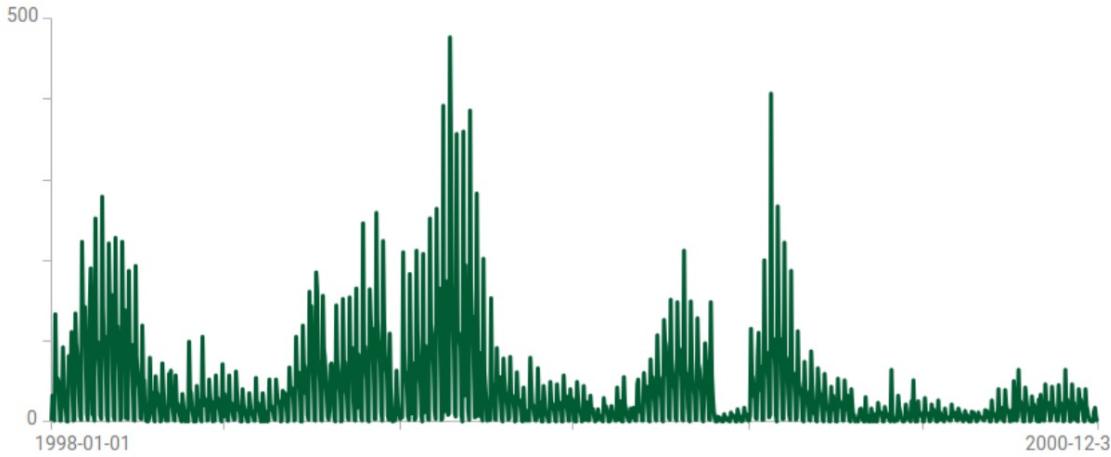


(a) Color coding for identification tasks. (b) Color coding for location tasks.

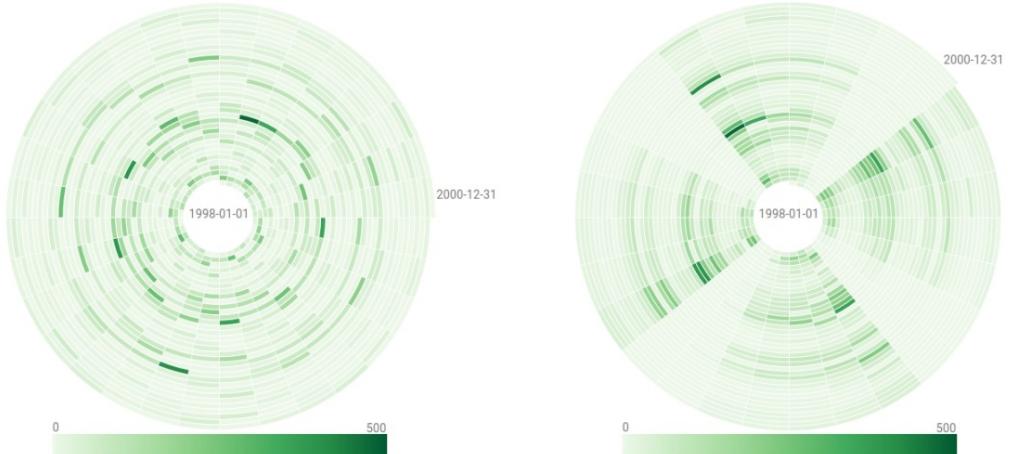
Figure 3.5 Applying the color maps from Figure 3.4 to temperature data.
Adapted from bl.ocks.org/mbostock/4063318.



Data representation has an impact on interpretation



(a) Line plot.



(b) Spiral plot (cycle length 32 days).

(c) Spiral plot (cycle length 28 days).

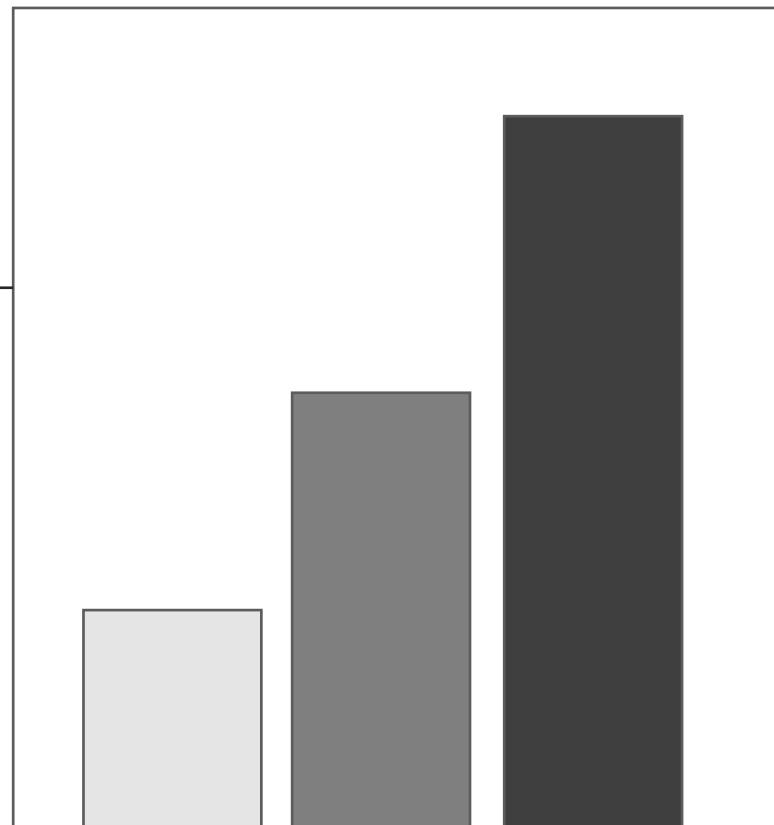


Tuning visual mappings



Mapping Relations

- 1 : 1
 - 1 : n —————
 - n : n
-
- no n : 1 (meaning of visual representation difficult)



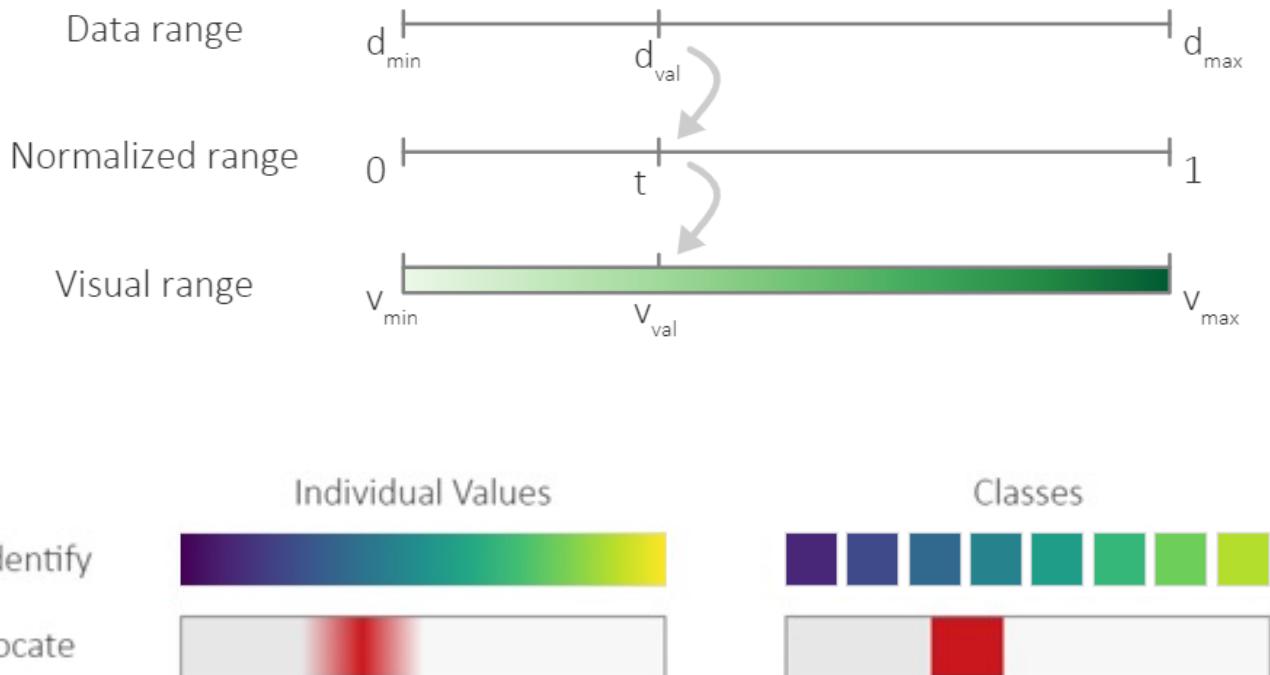
1:n = length + luminance

Sends stronger message but uses up channels



Mapping Mechanics

- Linear mapping $t = \frac{d_{val} - d_{min}}{d_{max} - d_{min}}$
 - standard approach – min-max value of data mapped to the entire spectrum of a channel
- Log mapping $t = \frac{\log(d_{val} - d_{min})}{\log(d_{max} - d_{min})}$
 - stretches out smaller values
- Individual vs aggregate values



About color...

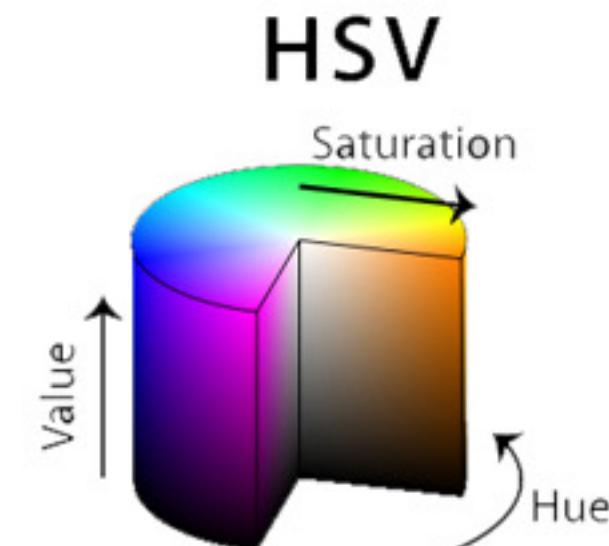
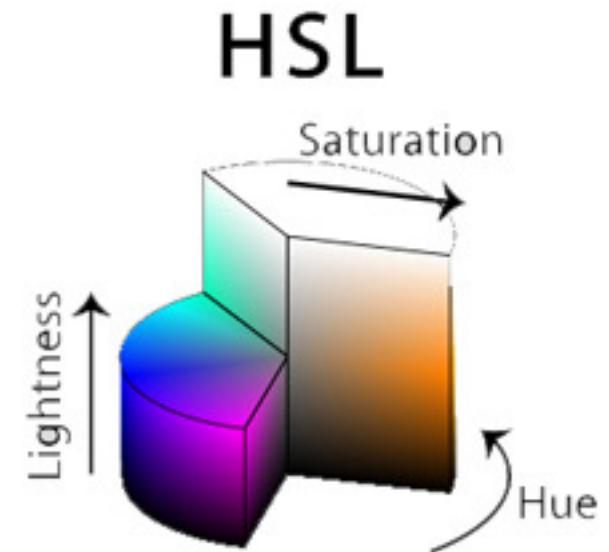
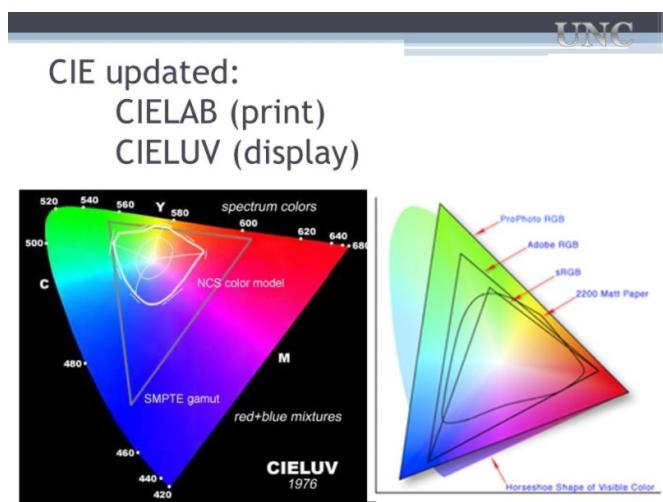
- Different color systems



CMYK - Subtractive Color



RGB - Additive Color



About color...

Decomposed into three channels:

Luminance		Ordered → Magnitude
Saturation		Ordered → Magnitude
Hue		Categorical → Identity



About color...

→ Categorical



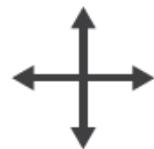
→ Ordered

→ Sequential

→ Diverging

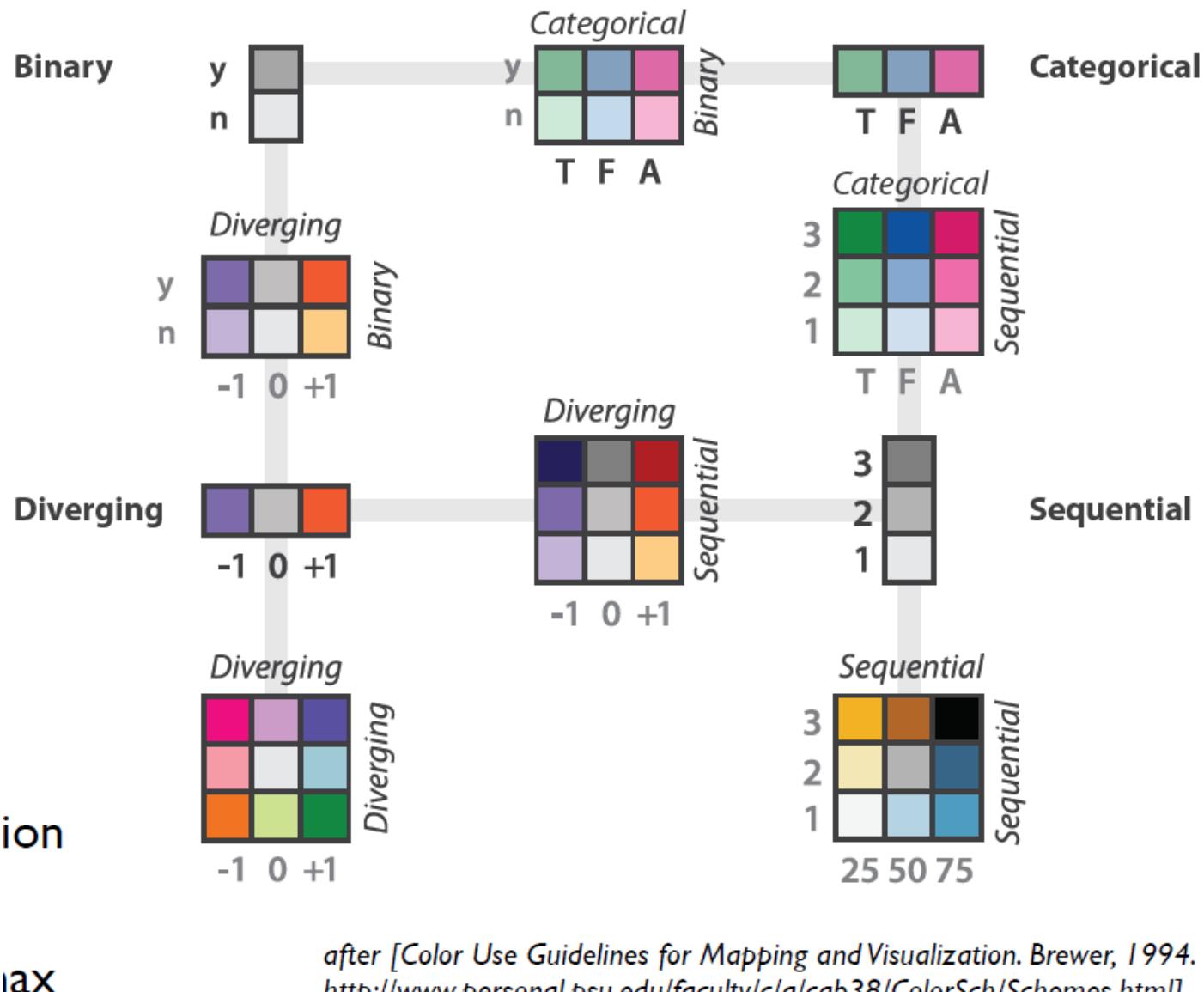


→ Bivariate



Color scheme options available with Altair
(Python version of Vega-Lite):

<https://vega.github.io/vega/docs/schemes/>



after [Color Use Guidelines for Mapping and Visualization. Brewer, 1994.
<http://www.personal.psu.edu/faculty/c/a/cab38/ColorSch/Schemes.html>]

About color...

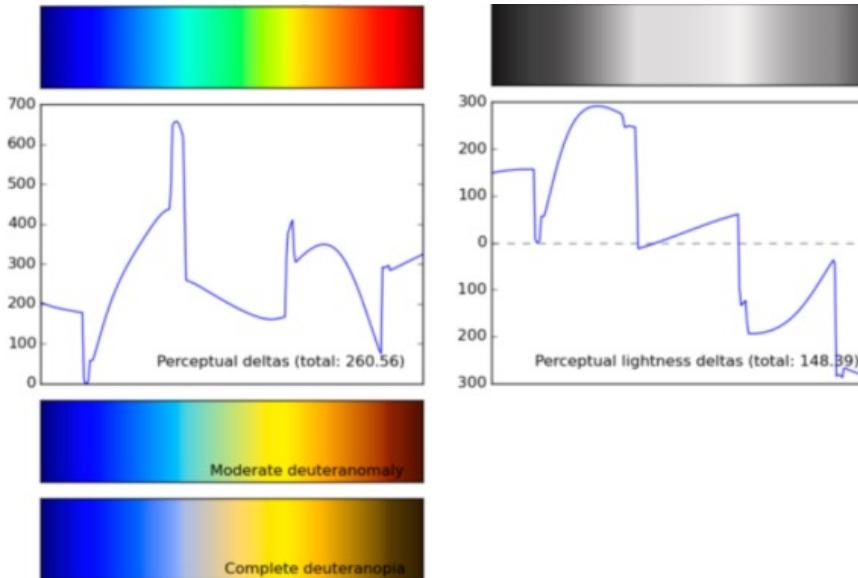
- Perceptual uniformity and ordering
 - Rainbow map introduces artificial structures ☹
- Monotonically increasing luminance

If you really want the Rainbow, consider using **Turbo**.



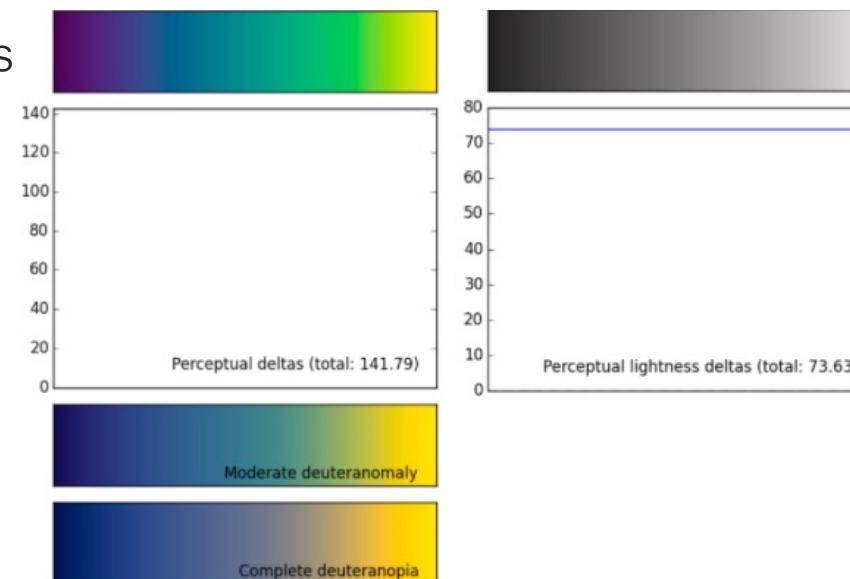
More info: <https://ai.googleblog.com/2019/08/turbo-improved-rainbow-colormap-for.html>

Rainbow



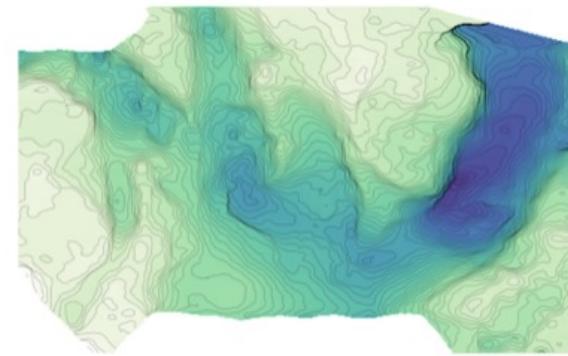
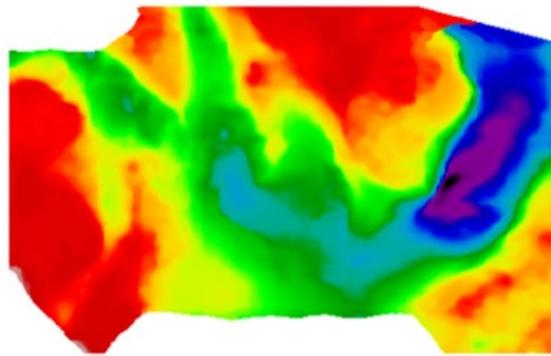
non-uniform perceptual changes between colors in rainbow (jet) map.

Viridis

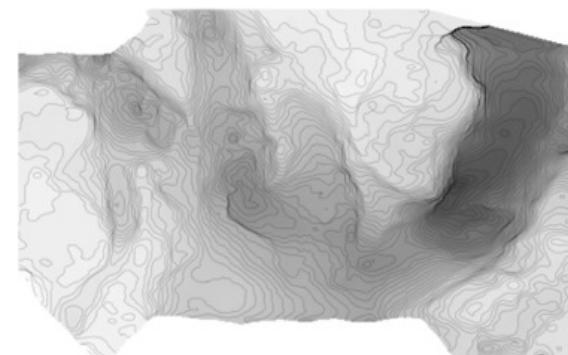
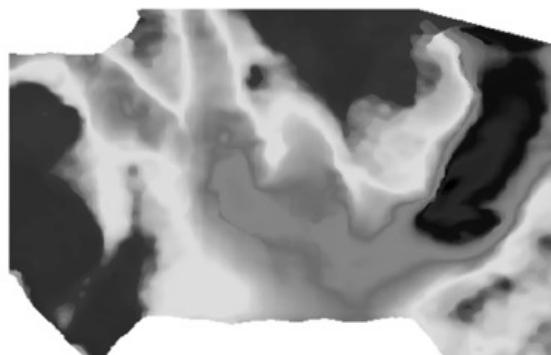
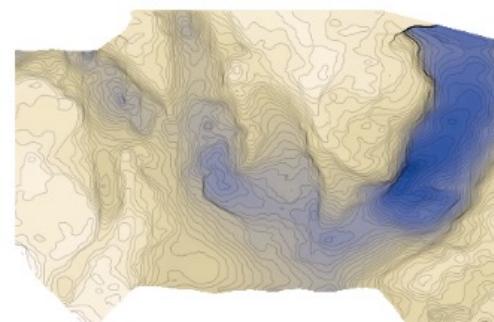
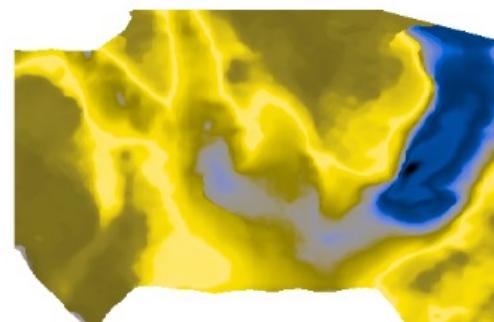


About color...

- Let's look at this with real data...
- Nova Scotia seafloor depth
 - rainbow (left) vs YIGnBu (right)



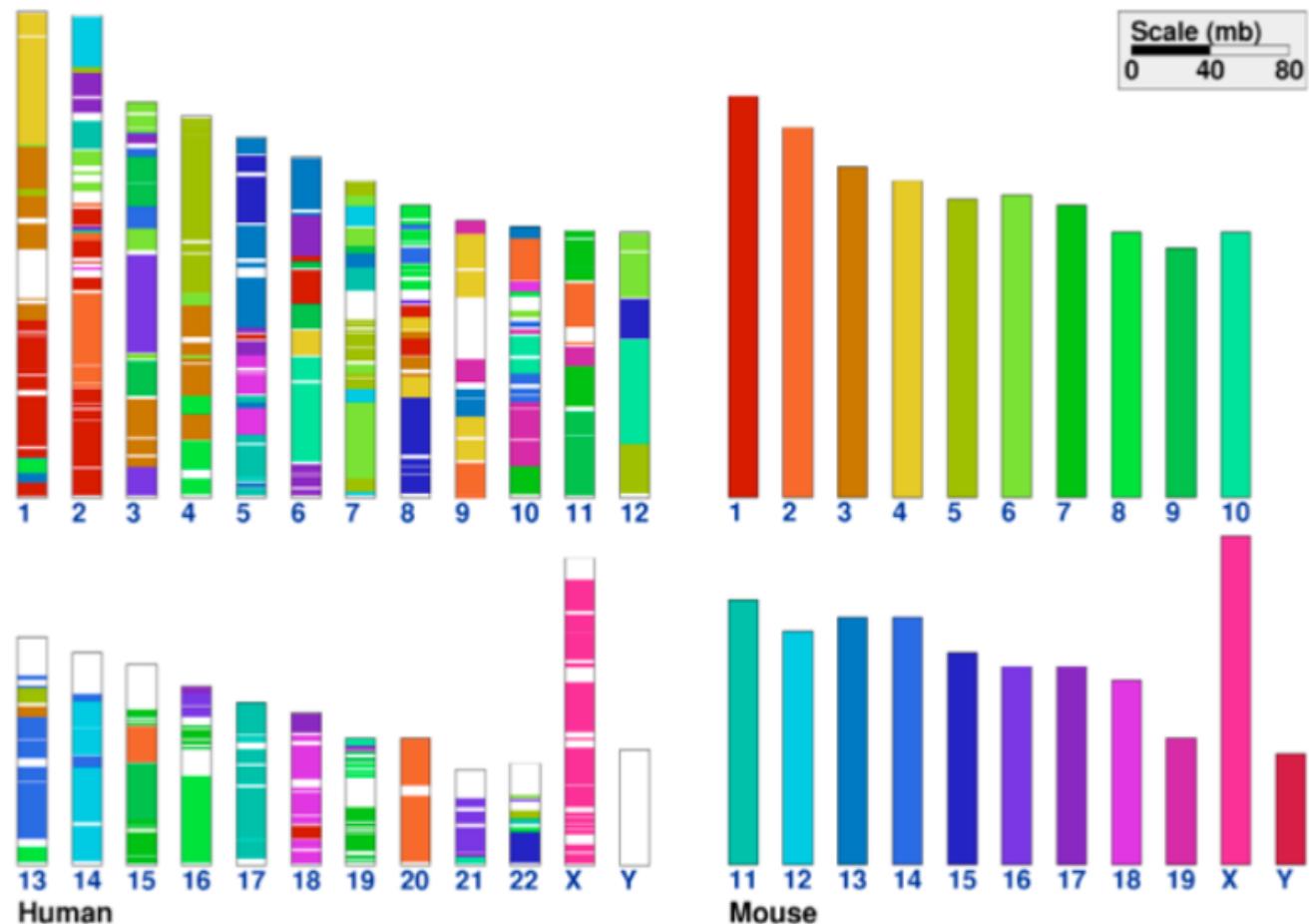
Let's check that it is indeed colourblind-safe and grey-safe:



Is this a
pit? What?

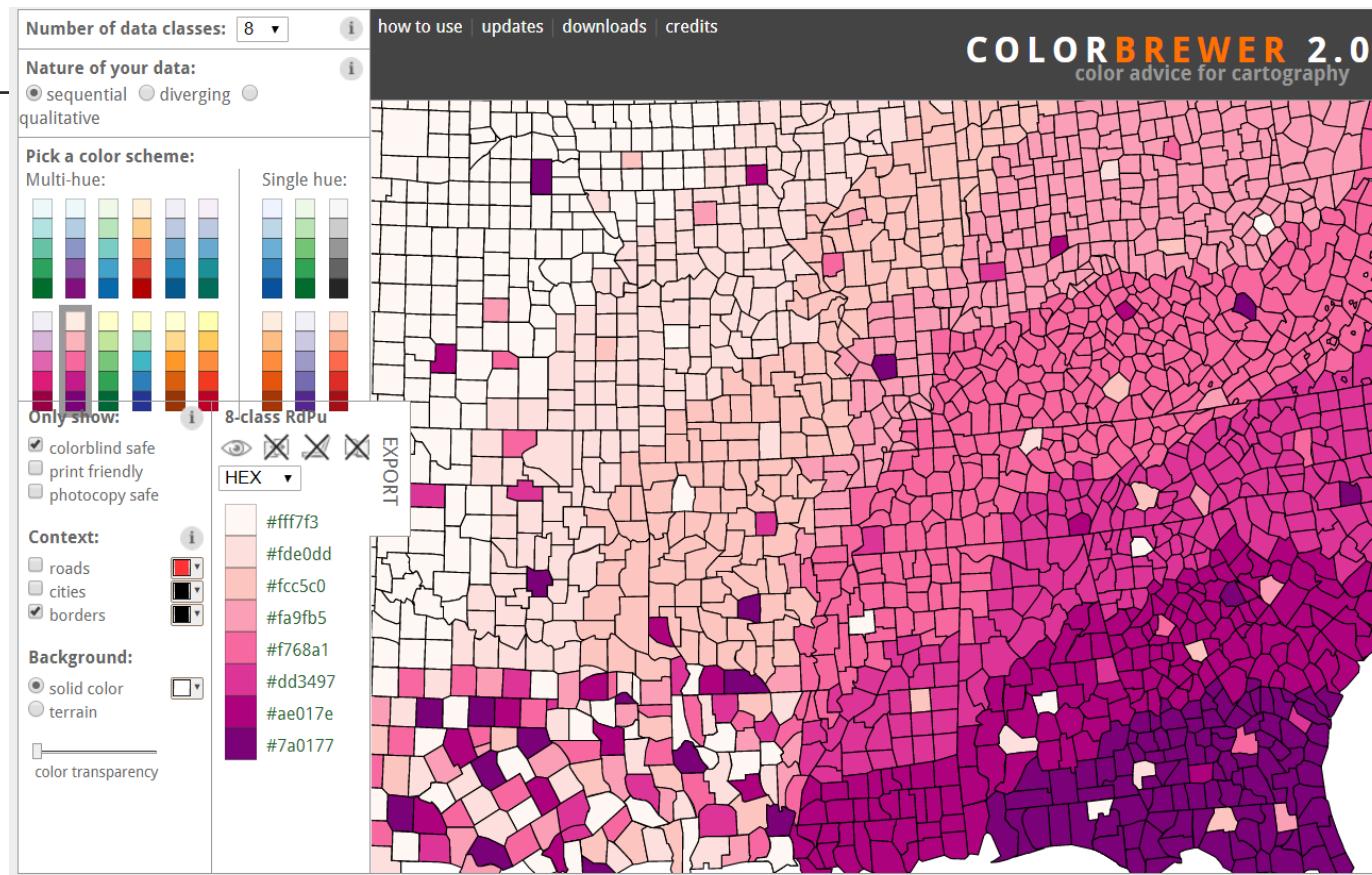
About color...

- Categorical color
 - There's a limit to number of colors that are discriminable
 - More than 5-10 bins becomes difficult



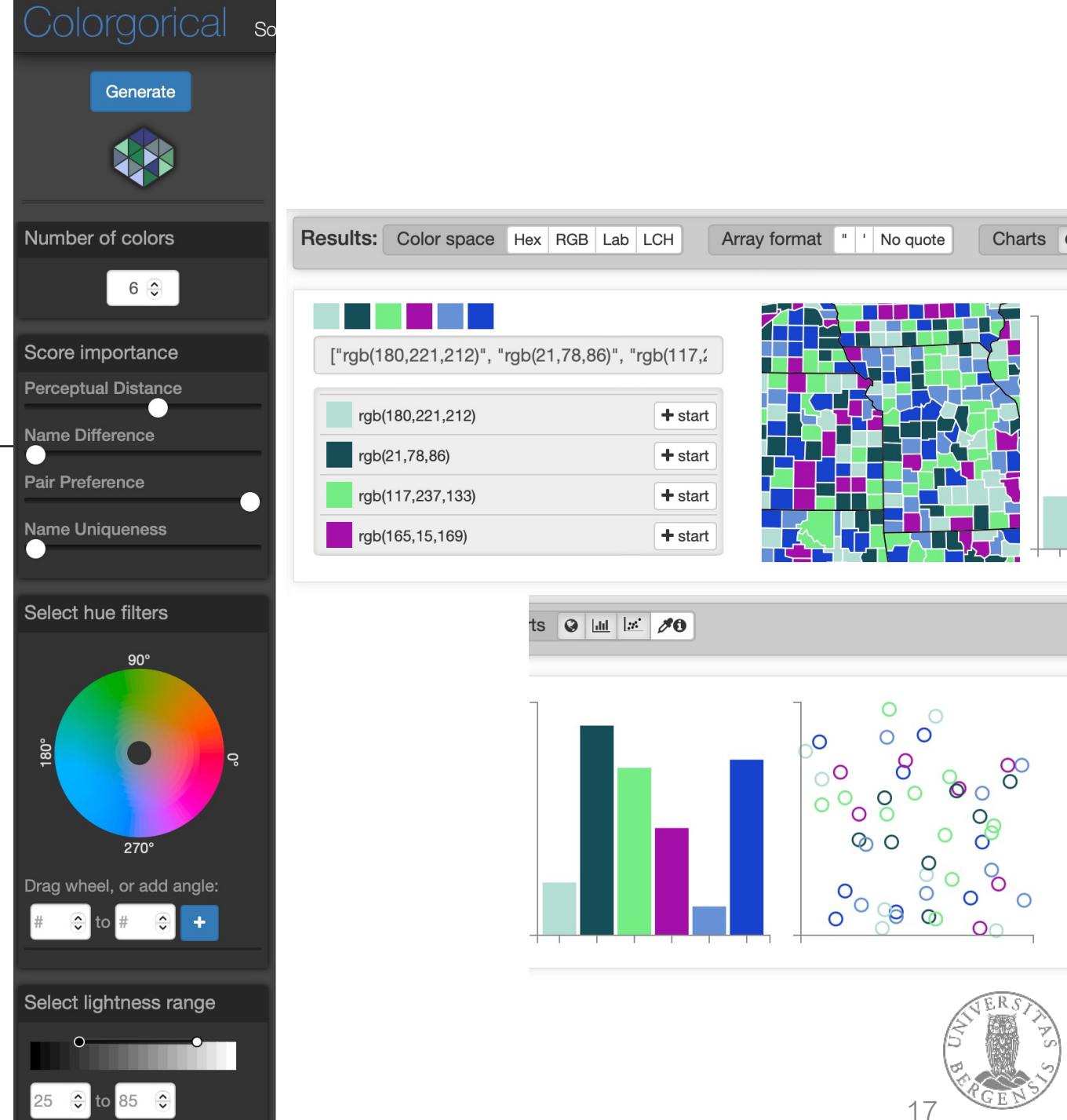
Color Tools

- ColorBrewer
 - <http://colorbrewer2.org/>
- Colorgorical
 - <http://vrl.cs.brown.edu/color>
- Sequential Color Scheme Generator
 - <http://eyetracking.upol.cz/color/>
- Colorpicker for Data
 - <http://tristen.ca/hcl-picker/>



Color Tools

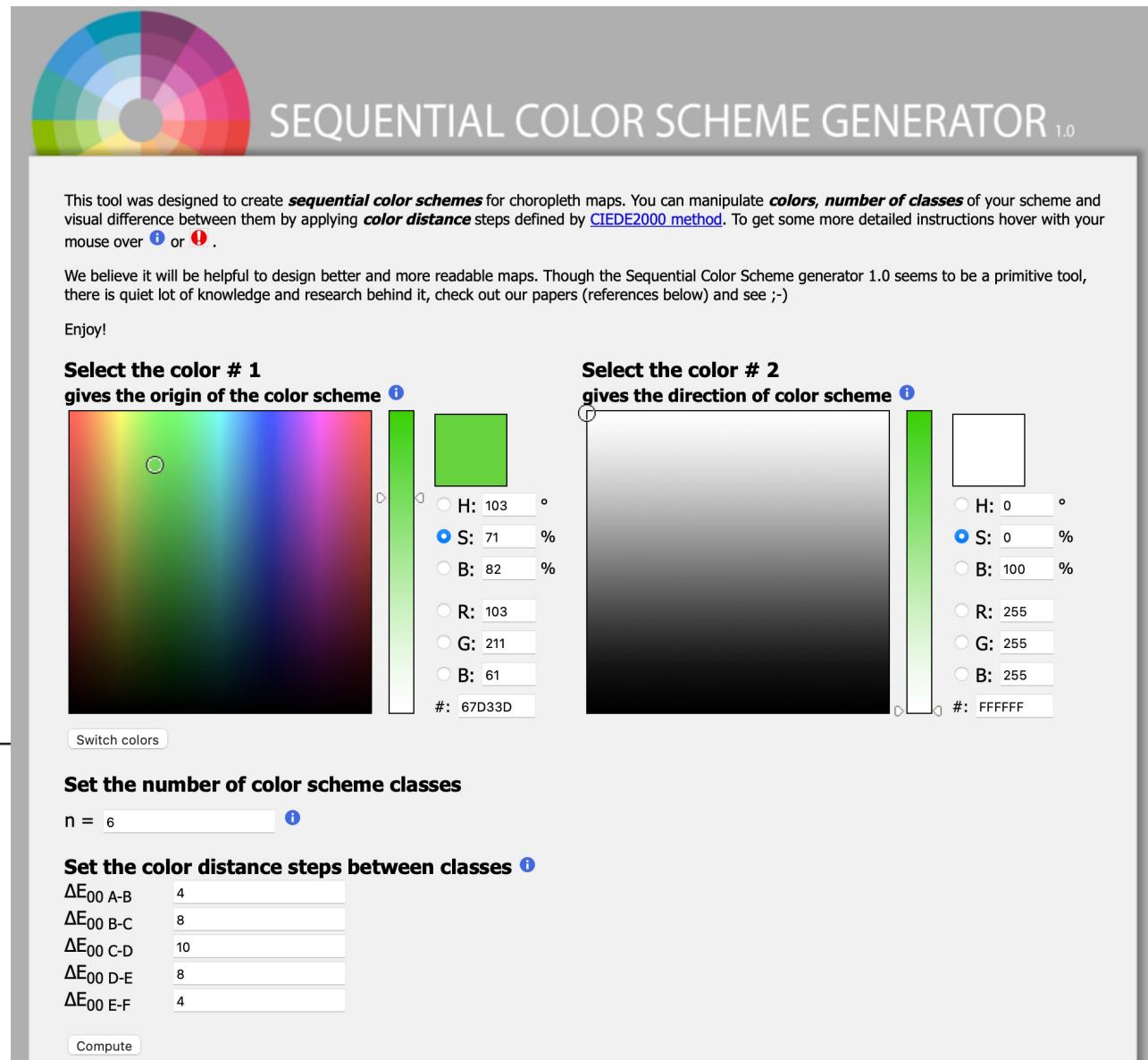
- ColorBrewer
 - <http://colorbrewer2.org/>
- Colorgorical
 - <http://vrl.cs.brown.edu/color>
- Sequential Color Scheme Generator
 - <http://eyetracking.upol.cz/color/>
- Colorpicker for Data
 - <http://tristen.ca/hcl-picker/>



Color Tools

- ColorBrewer
 - <http://colorbrewer2.org/>
- Colorgorical
 - <http://vrl.cs.brown.edu/color>
- Sequential Color Scheme Generator

 - <http://eyetracking.upol.cz/color/>
- Colorpicker for Data
 - <http://tristen.ca/hcl-picker/>



The screenshot shows the Sequential Color Scheme Generator 1.0 interface. At the top right is a color wheel with the text "SEQUENTIAL COLOR SCHEME GENERATOR 1.0". Below it is a descriptive text block: "This tool was designed to create **sequential color schemes** for choropleth maps. You can manipulate **colors**, **number of classes** of your scheme and visual difference between them by applying **color distance** steps defined by [CIEDE2000 method](#). To get some more detailed instructions hover with your mouse over ⓘ or ⓘ." A note below states: "We believe it will be helpful to design better and more readable maps. Though the Sequential Color Scheme generator 1.0 seems to be a primitive tool, there is quiet lot of knowledge and research behind it, check out our papers (references below) and see ;-)"

The interface includes two main color selection areas:

- Select the color # 1 gives the origin of the color scheme ⓘ**: A color gradient from red to purple with a circular selection point. To its right are sliders for H: 103, S: 71%, B: 82%, and color hex code #67D33D.
- Select the color # 2 gives the direction of color scheme ⓘ**: A color gradient from black to white with a circular selection point. To its right are sliders for H: 0, S: 0%, B: 100%, and color hex code FFFFFF.

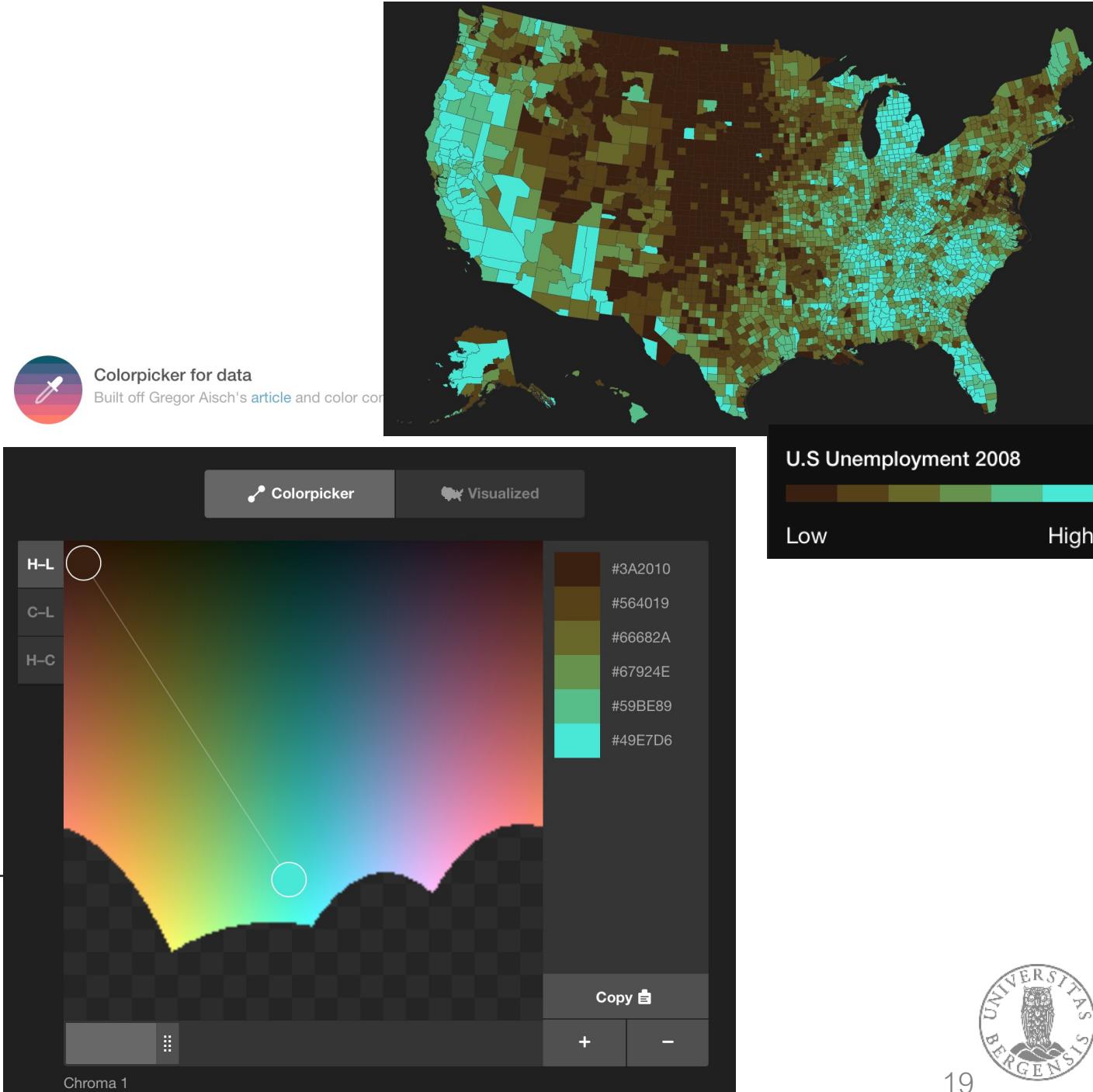
Below these are controls for the number of classes and color distance steps:

- Set the number of color scheme classes**: A text input field containing "n = 6" with a help icon ⓘ.
- Set the color distance steps between classes ⓘ**: A table with five rows for color pairs A-B, B-C, C-D, D-E, and E-F, each with a value of 4 in a dropdown menu.

A "Compute" button is located at the bottom left of the color selection area.

Color Tools

- ColorBrewer
 - <http://colorbrewer2.org/>
- Colorgorical
 - <http://vrl.cs.brown.edu/color>
- Sequential Color Scheme Generator
 - <http://eyetracking.upol.cz/color/>
- Colorpicker for Data
 - <http://tristen.ca/hcl-picker/>



Visualization Idioms

- Tabular data
- Spatial data
- Network & tree data



If you'd like to follow along...

- day_05 on ICTP 2022 GitLab
- Install modules
 - From inside repo, run:
`pip install -r requirements.txt`
- Sample visualizations
 - `./basic_charts/`
 - `./basic_charts_html_output/`

```
1 import altair as alt
2 import pandas as pd
3
4 penguins_data = pd.read_json('./basic_charts_data/penguins.json')
5 print(penguins_data.head())
6
7
8 # how many of each species of penguin are there?
9 penguin_species_bar = alt.Chart(penguins_data).mark_bar().encode(
10     alt.Y('Species:N', sort='x'),
11     x = 'count()',
12     color = 'Species'
13 ).properties(
14     title='Number of Penguins by Species'
15 )
16
17 penguin_species_bar.configure_title(
18     fontSize=20,
19     anchor='start'
20 ).save('./basic_charts_html_output/bar.html')
```

`./basic_charts/bar.py`



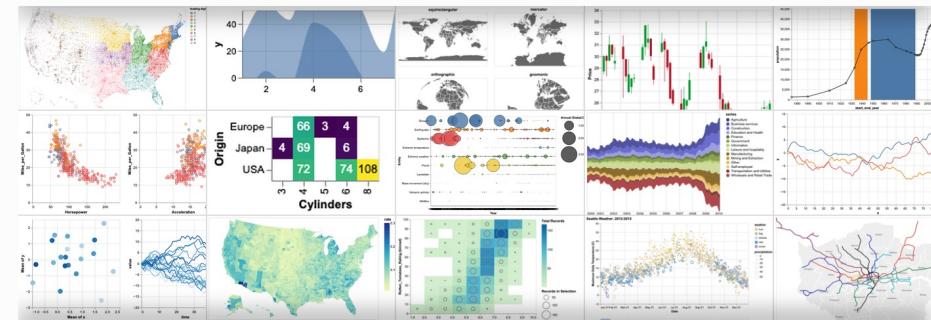
Vega-Altair

- Declarative statistical visualization library for Python
- Built on top of Vega-Lite (<https://vega.github.io/vega-lite/>)
- Directly applies marks and channels in construction of a visualization
- Output as json spec embedded in html file – no plugins/other installations required to view/interact

The screenshot shows the official Altair documentation website. At the top right is a logo for Altair with a stylized bar chart icon. Below it, the version '4.2.0' is displayed. A search bar labeled 'Search docs' is positioned at the top center. The main navigation menu includes sections for 'GETTING STARTED' (Overview, Installation, Dependencies, Development Install, Basic Statistical Visualization), 'GALLERY' (Example Gallery), and 'USER GUIDE' (Specifying Data in Altair, Encodings, Marks, Data Transformations, Bindings, Selections, Conditions: Making Charts Interactive, Top-Level Chart Configuration, Compound Charts: Layer, HConcat, VConcat, Repeat, Facet, Scale and Guide Resolution, Saving Altair Charts, Customizing Visualizations, Times and Dates in Altair).

<https://altair-viz.github.io/index.html>

Vega-Altair: Declarative Visualization in Python



Vega-Altair is a declarative statistical visualization library for Python, based on [Vega](#) and [Vega-Lite](#), and the source is available on [GitHub](#).

The Vega-Altair open source project is not affiliated with Altair Engineering, Inc.

With Vega-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.

Getting Started

- Overview
- Installation
- Dependencies
- Development Install
- Basic Statistical Visualization

Gallery

- Example Gallery



Vega Ingredients

- Data → DataFrame to visualize upon
- Mark → Notations for each row observation (line, bar, tick, point)
- Encoding → Data representation Channel (X Position, Y Position, color, size)
- Transform → Data Transformation before applying visualization (calculate, filter, aggregate,fold)
- Scale → Function which inputs data and renders it on the screen.
- Guide → Visual aids on charts (legends), ticks on x and y axis.



Arrange, Map

Encode

④ Arrange

→ Express



→ Order



→ Use



→ Separate



→ Align



④ Map

from categorical and ordered attributes

→ Color

→ Hue



→ Saturation



→ Luminance



→ Size, Angle, Curvature, ...



→ Shape



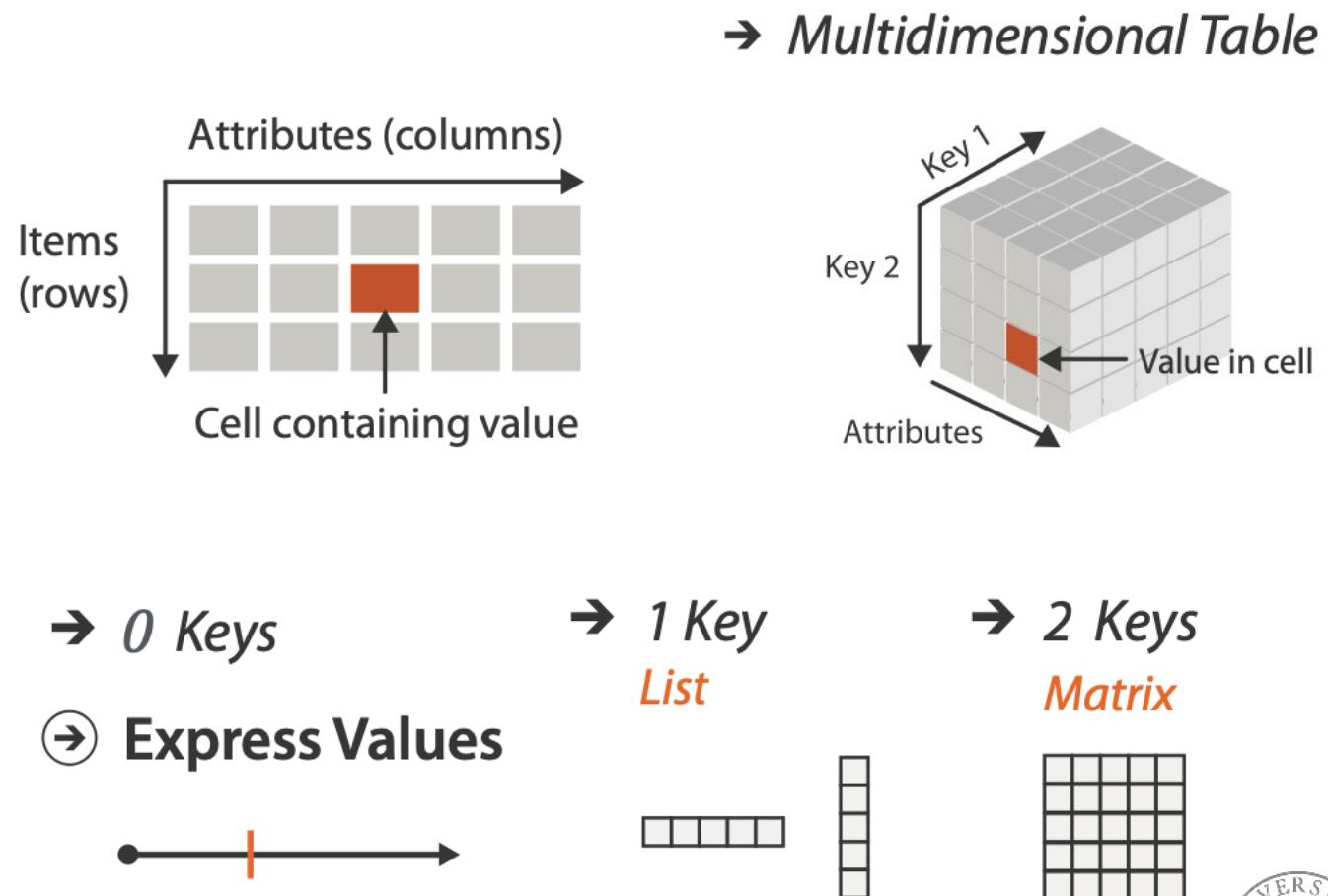
→ Motion

Direction, Rate, Frequency, ...



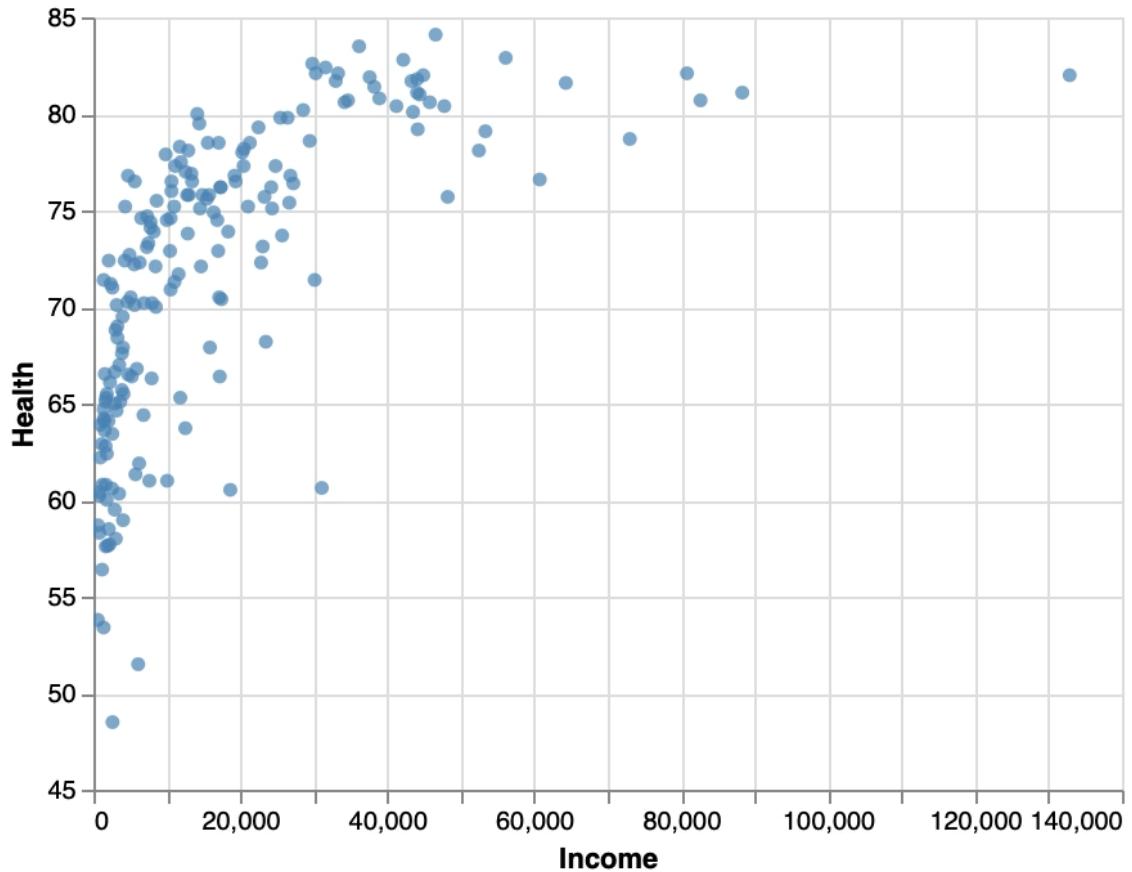
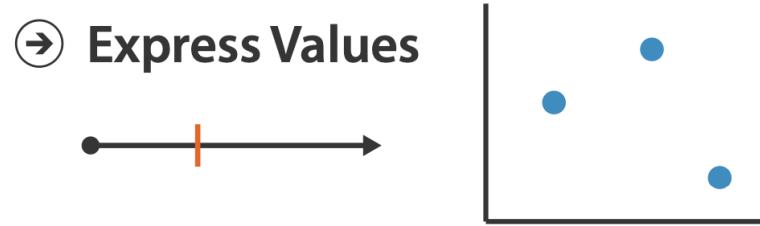
Tabular Data

- (Key, Value)
 - key = independent attribute
 - unique index for item lookup
 - can have multiple keys in multidimensional tables
 - value = dependent attribute
 - cell value
- Classify arrangement by keys used (0, 1, 2, ...)



Scatterplot

- no keys, only values
 - 2 quantitative attributes
- mark: points
- channels
 - horizontal + vertical position
- tasks
 - find trends, outliers, distribution, correlation, clusters
- scalability
 - hundreds of items

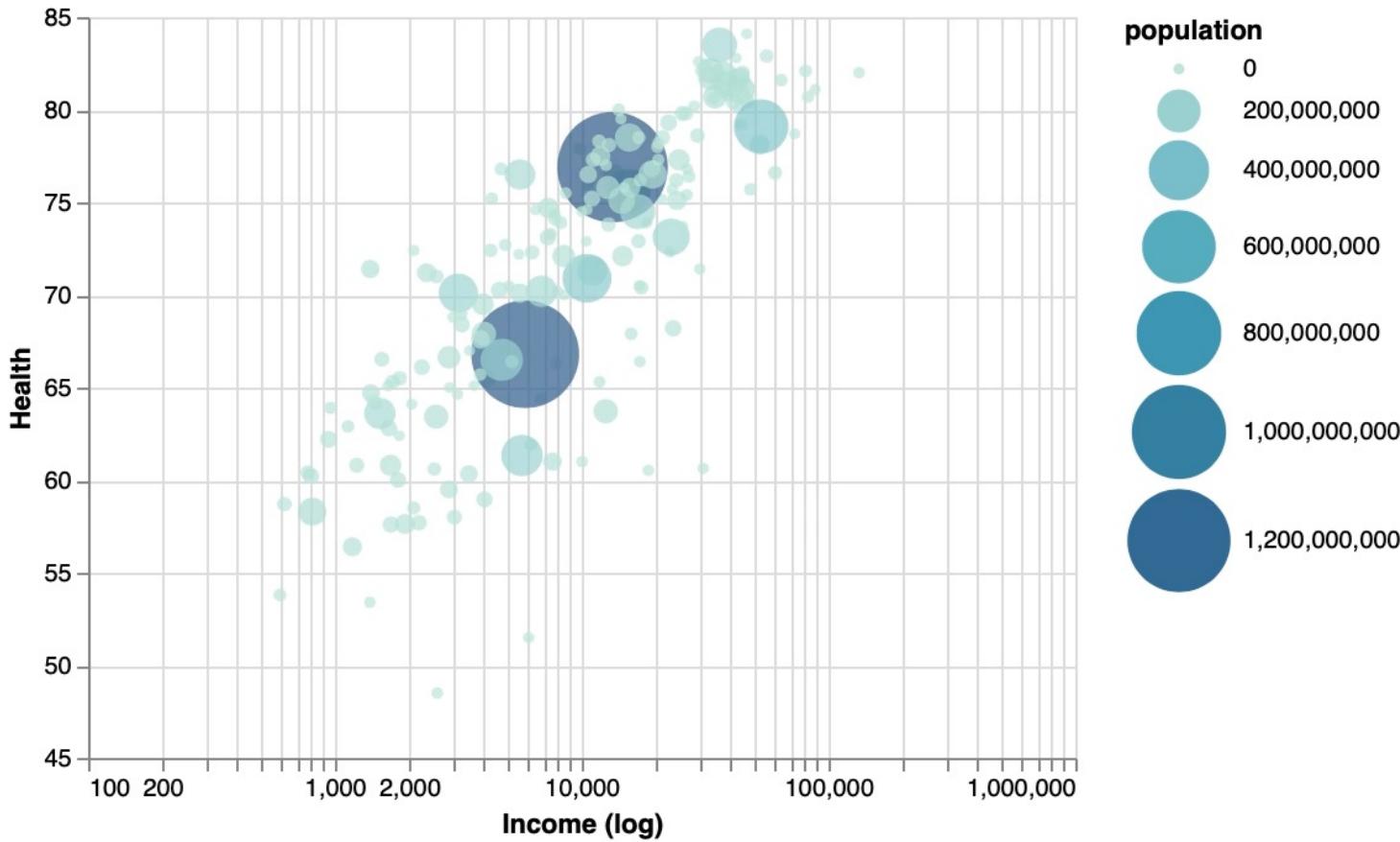


day_05/basic_charts/scatter.py

Scatterplot

- Can also encode additional channels (since using point marks)
 - Color
 - Area
 - not radius – take square root since area grows quadratically
 - Size

→ Express Values

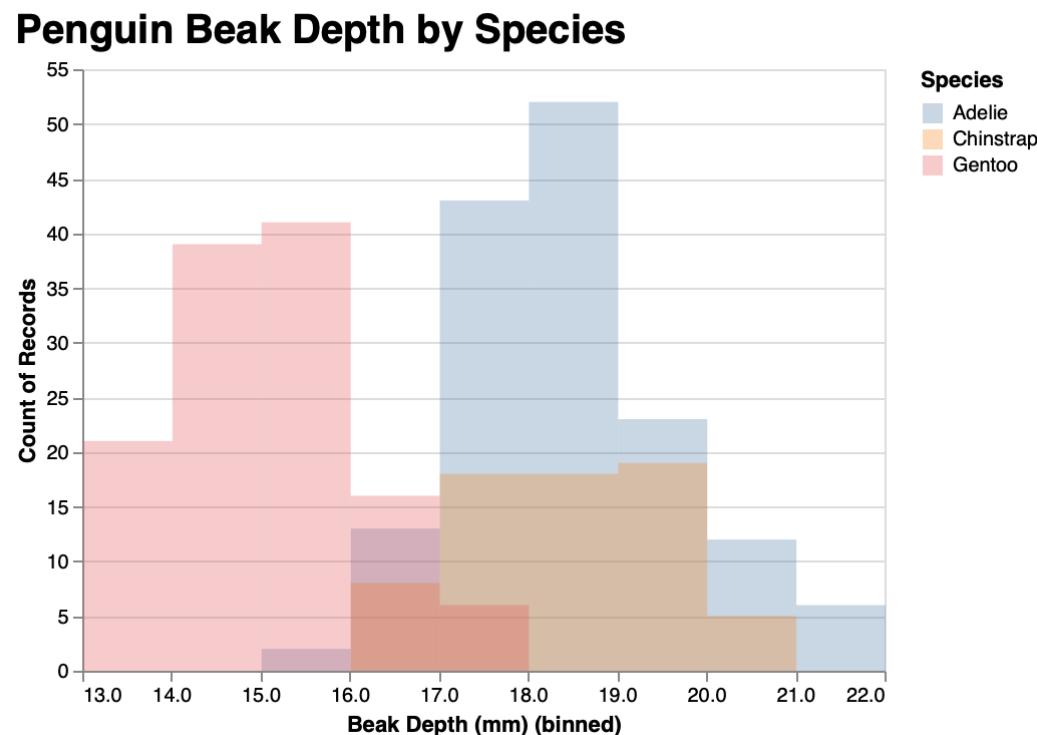
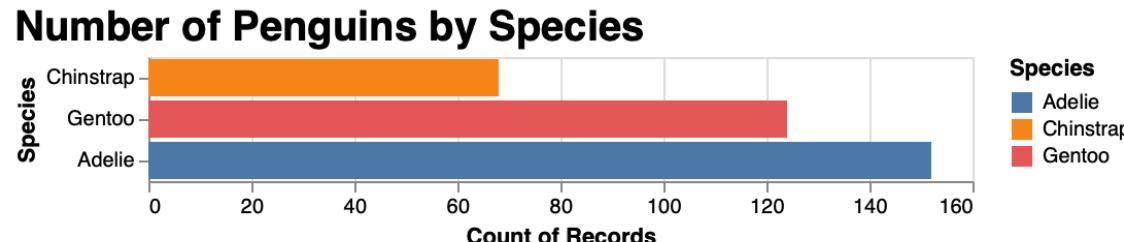


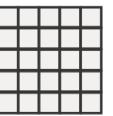
day_05/basic_charts/scatter.py



Bar Chart/Histogram

- one key, one value
 - 1 categorical attribute, 1 quantitative attribute
- mark: lines
- channels
 - length to express quantitative values
 - spatial regions: one per mark
 - ordered by label (alphabetical),
 - ordered by length attribute (data-driven)
- task
 - compare, lookup values
- scalability
 - dozens to hundreds of levels for key attribute

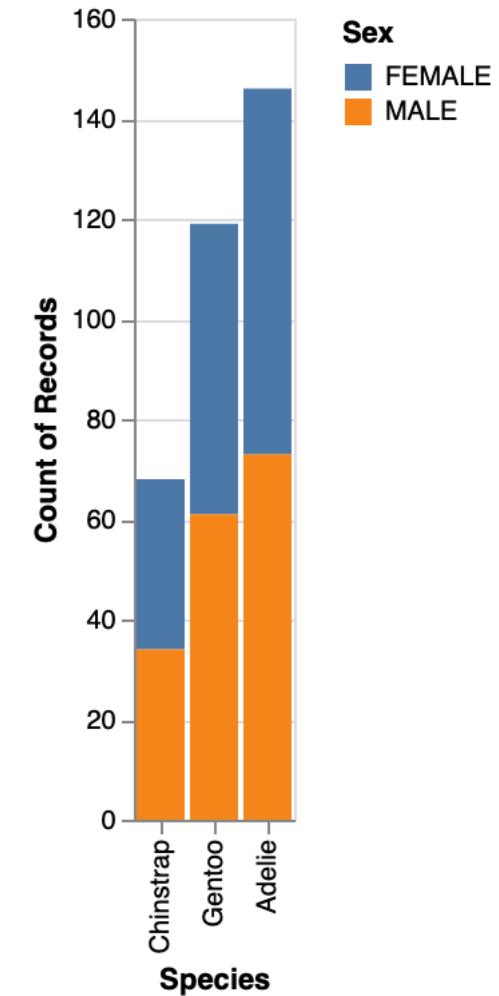


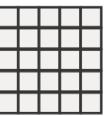


Stacked Bar Chart

- data: 2 categorical attributes, 1 quantitative attribute
- mark: vertical stack of line marks
- channels
 - length
 - color hue
 - spatial regions
 - one per stacked bar
- task
 - part-to-whole relationship
- scalability
 - several to one dozen levels for stacked attribute

Penguin Gender by Species

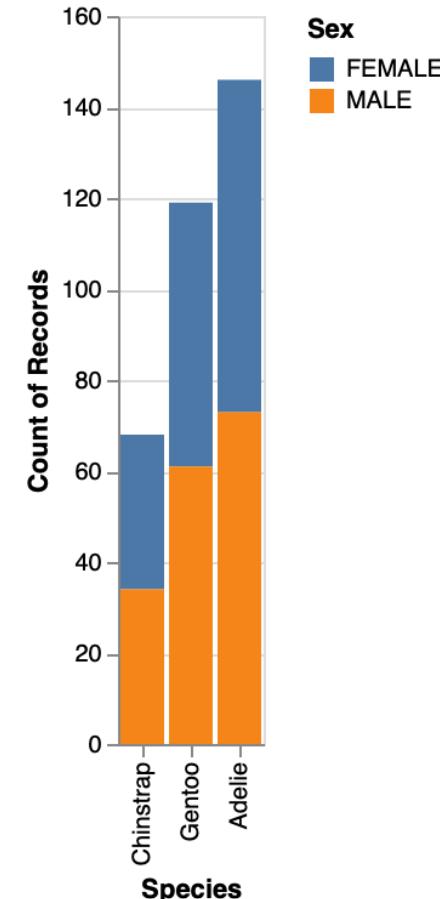




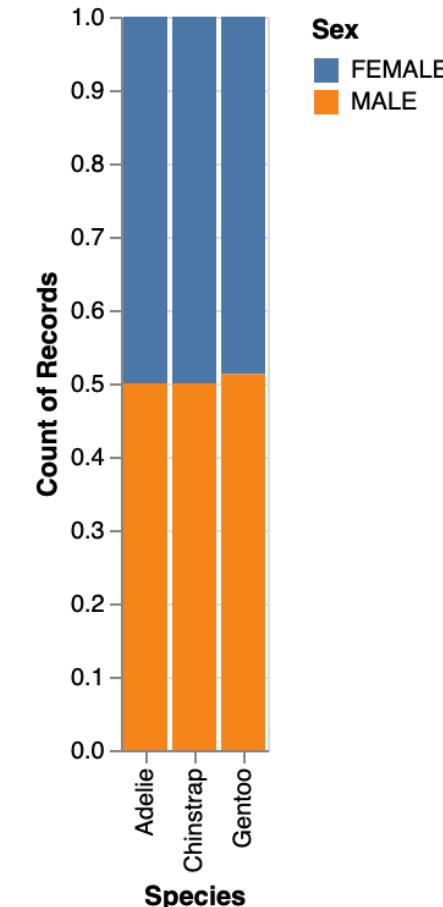
Normalized Stacked Bar Chart

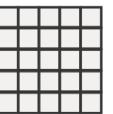
- stacked bar chart normalized to full vertical height
- comparisons between groups easier (part-to-whole relationships)

Penguin Gender by Species



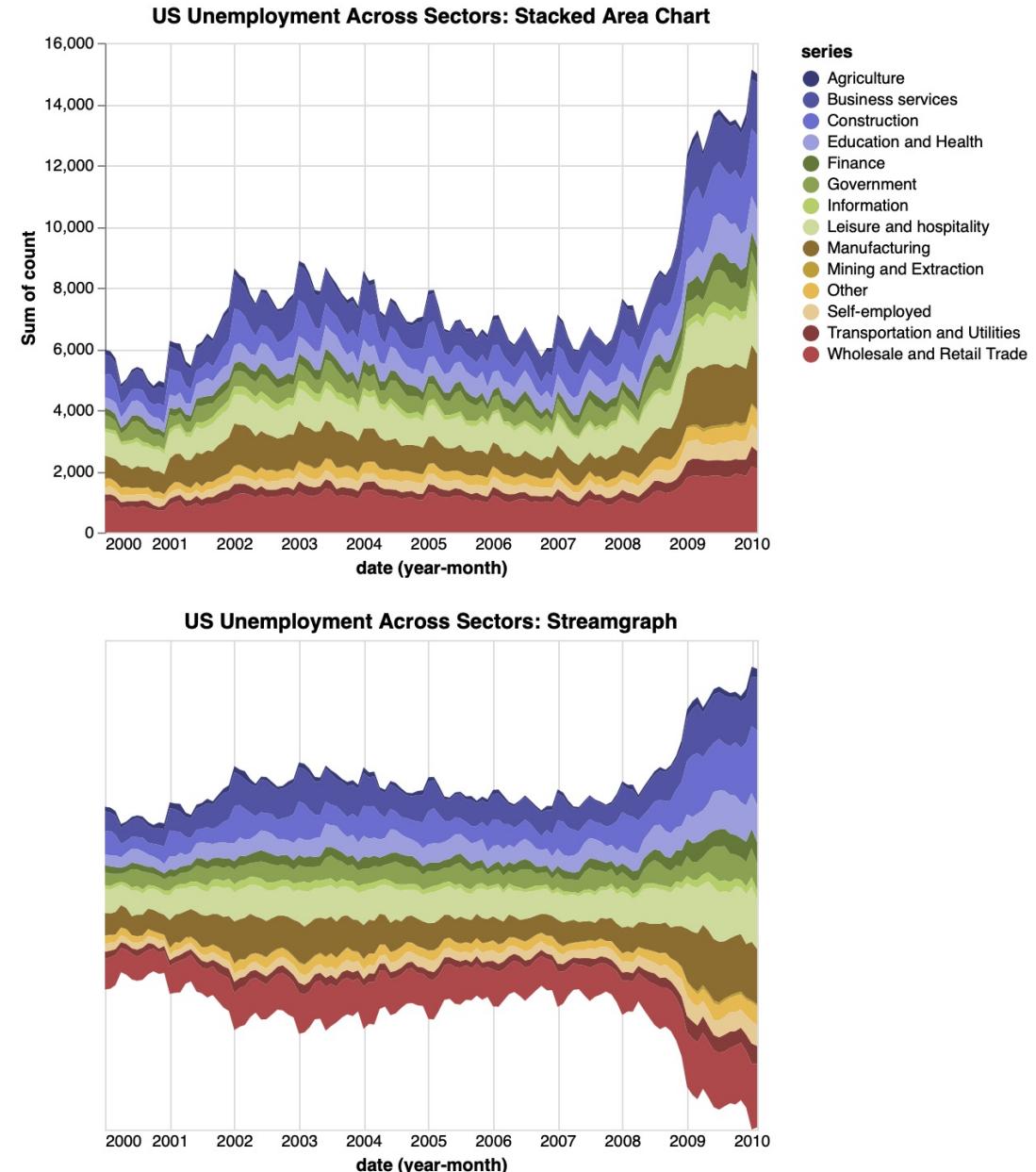
Penguin Gender by Species





Streamgraph

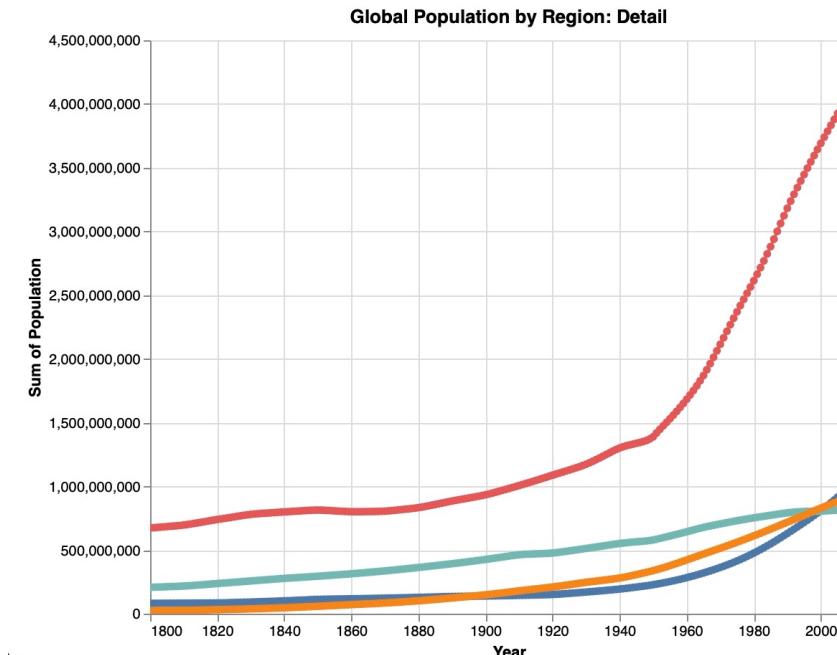
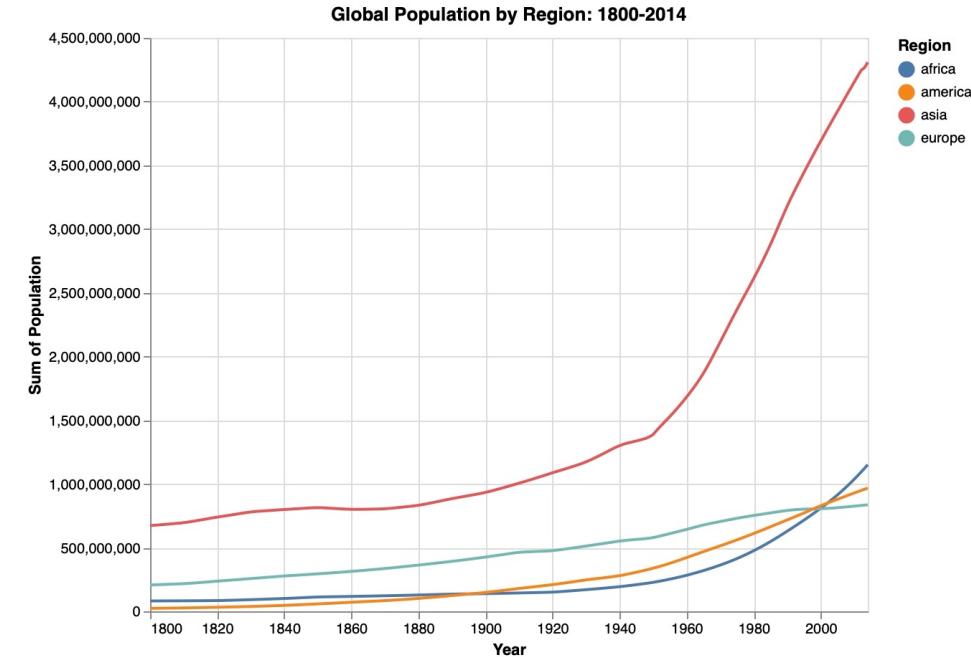
- generalized stacked graph
 - emphasizing horizontal continuity vs. vertical items
- data
 - 1 categorical key attribute (artist)
 - 1 ordered key attribute (time)
 - 1 quantitative value attribute (counts)
- derived data
 - geometry: layers -> height encodes counts
 - 1 quantitative attribute (layer ordering)
- scalability
 - hundreds of time keys
 - dozens to hundreds of artist keys
 - more than stacked bars, since most layers don't extend across whole chart



day_05/basic_charts/streamgraph.py
<https://altair-viz.github.io/gallery/streamgraph.html>

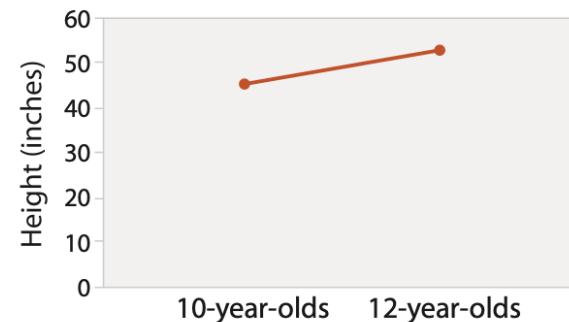
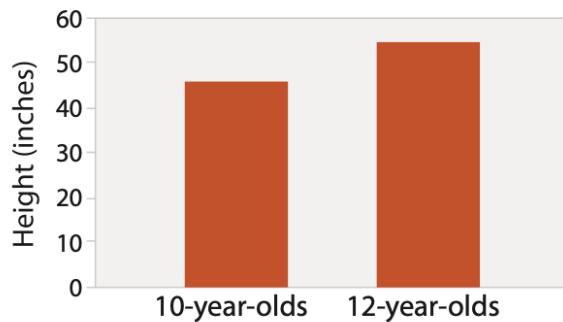
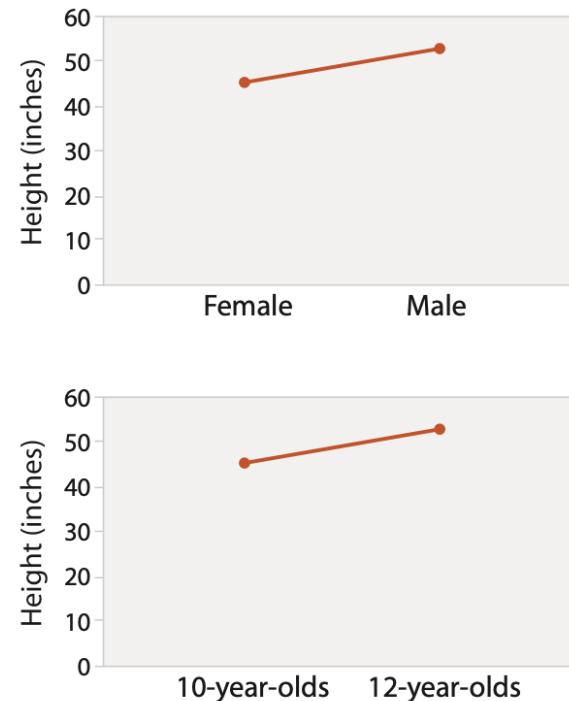
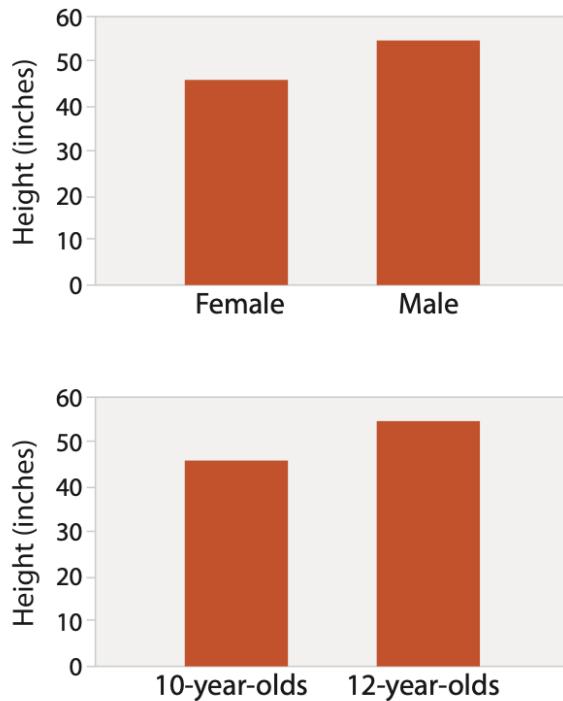
Line/Dot Chart

- one key, one value
 - 2 quantitative attributes (for each line in chart)
- mark: points + line (connection marks)
 - lines explicitly show relationship between items
- channels
 - length (position) to express quantitative values
 - separated and ordered by key attribute
- task
 - find trend



In Practice: Bar vs Line

- depends on type of key attribute
 - bar charts if categorical
 - line charts if ordered (quantitative)

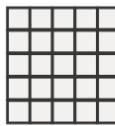


after [Bars and Lines: A Study of Graphic Communication.
Zacks and Tversky. Memory and Cognition 27:6 (1999),
1073–1079.]

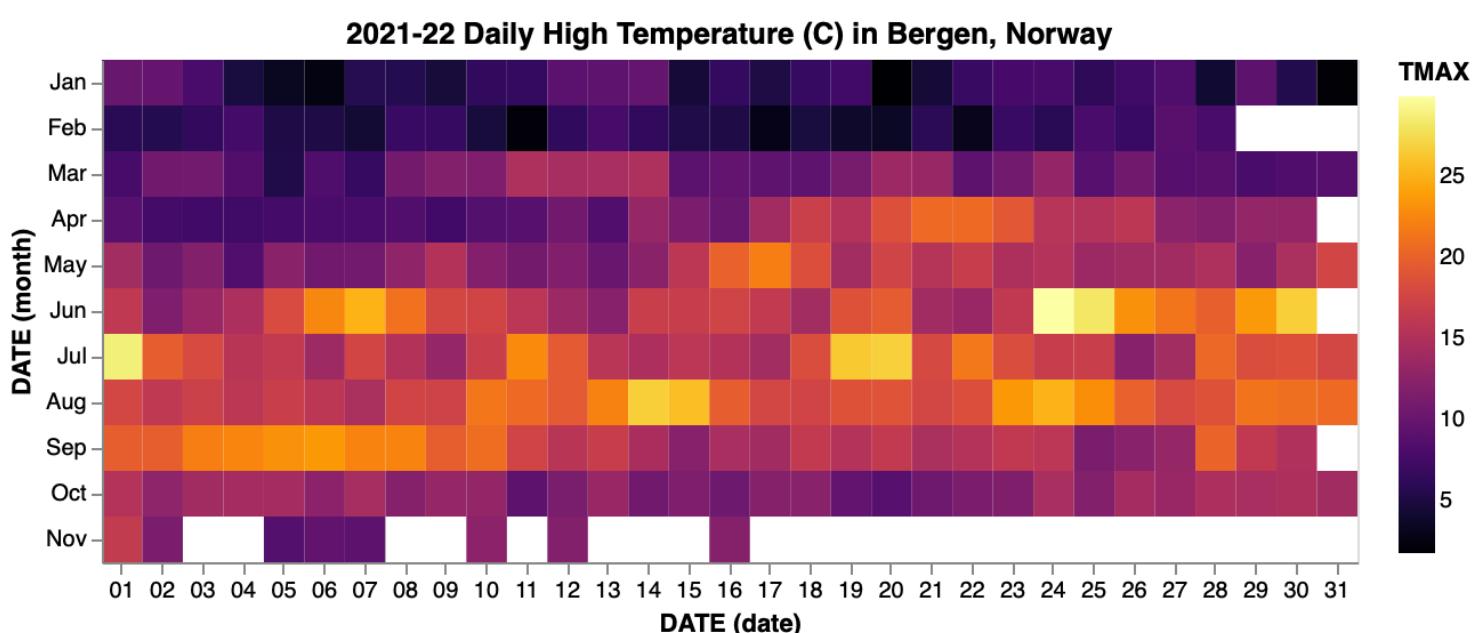
Heatmap

→ 2 Keys

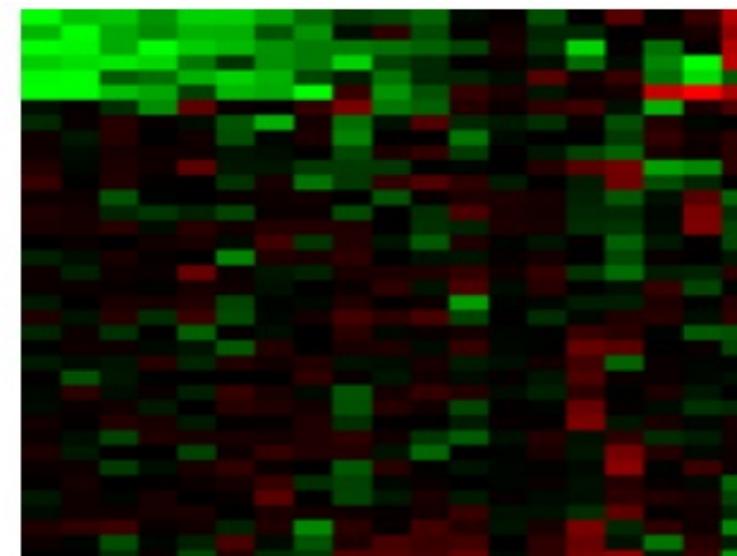
Matrix



- two keys, one value
 - 2 categorical attributes (gene, experimental condition)
 - 1 quantitative attributes (expression levels)
- marks: area
 - separate and align in 2D matrix
 - indexed (ordered) by 2 categorical attributes
- channels
 - color by quantitative attribute
 - ordered vs. diverging color map
- task
 - find clusters, outliers
- scalability
 - 1K categorical levels, 1M items; only ~10 quantitative attribute levels

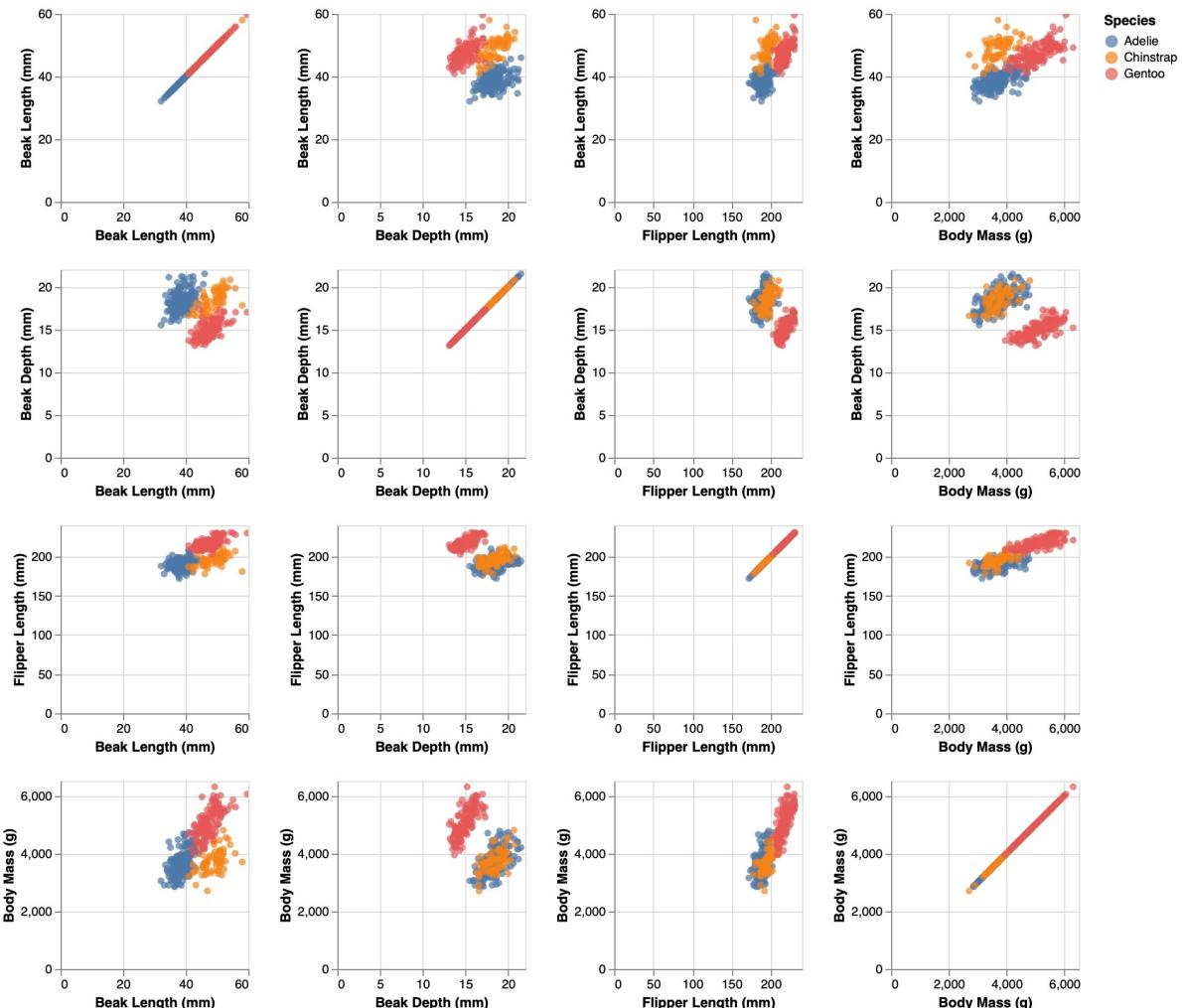


day_05/basic_charts/heatmap.py



Scatterplot Matrix

- rectilinear axes, point mark
 - all possible pairs of axes
- scalability
 - one dozen attributes
 - dozens to hundreds of items

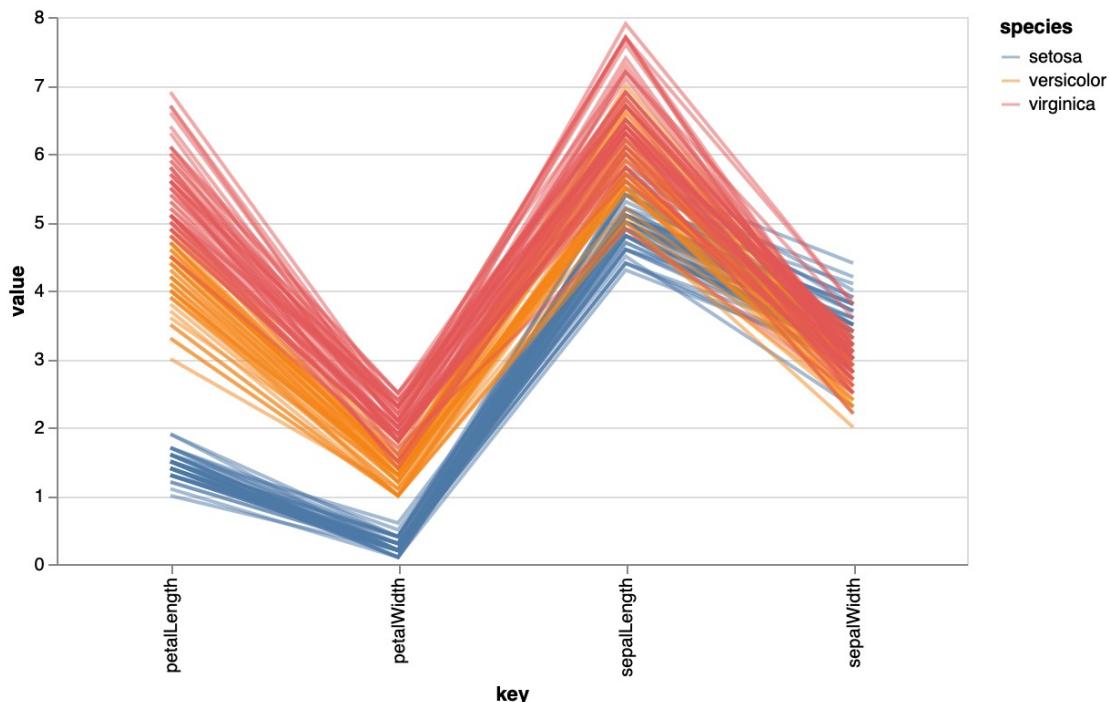


day_05/basic_charts/scatter-matrix.py

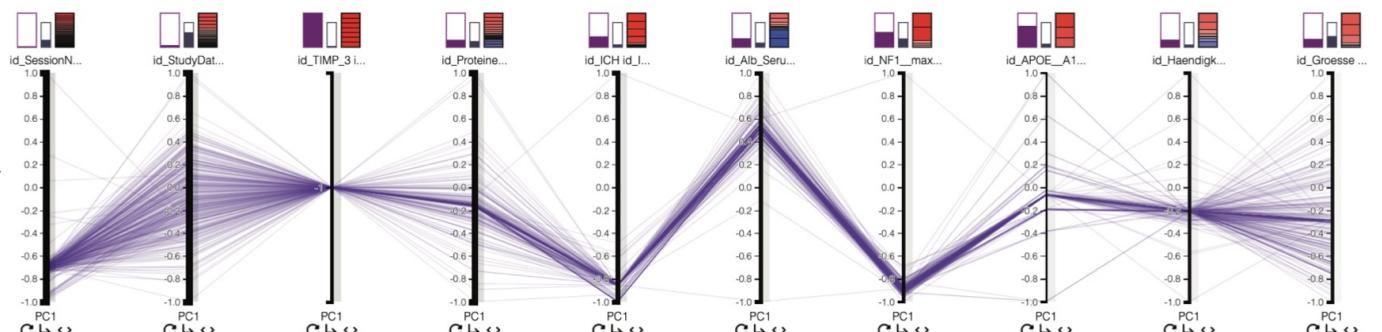


Parallel Coordinates

- parallel axes, jagged line representing item
 - parallel axes, item as point
 - axis ordering is major challenge
 - scalability
 - dozens of attributes
 - hundreds of items



https://altair-viz.github.io/gallery/parallel_coordinates.html

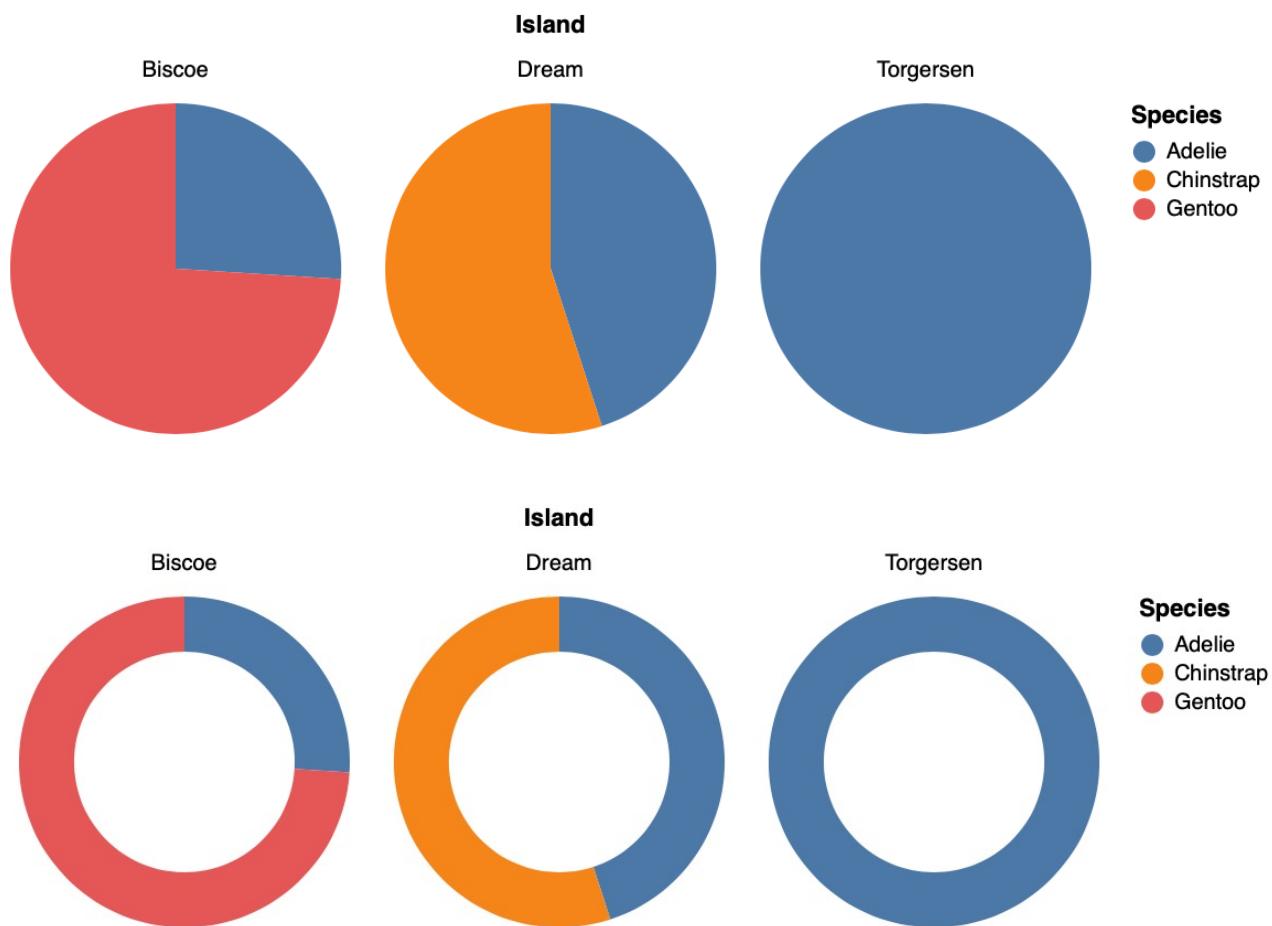


Garrison L, Muller J, Schreiber S, Oeltze-Jafra S, Hauser H, Bruckner S. DimLift: Interactive Hierarchical Data Exploration Through Dimensional Bundling. IEEE Trans Vis Comput Graph. 2021 Jun;27(6):2908-2922.



Pie/Donut Charts

- data
 - 1 categorical key attribute
 - 1 quantitative value attribute
- pie/donut chart
 - area marks with angle channel
 - accuracy: angle/area much less accurate
- task
 - part-to-whole judgements (ok for 2-4 categories)

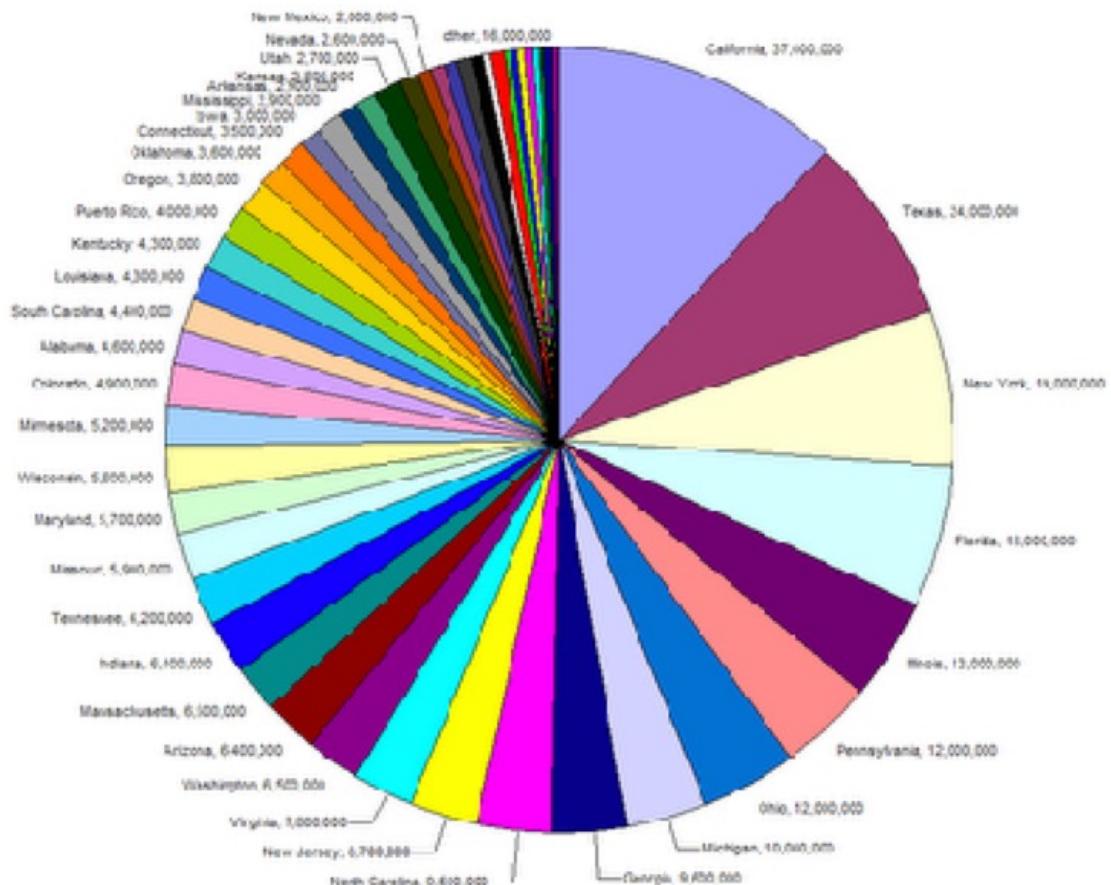


day_05/basic_charts/pie-polar.py



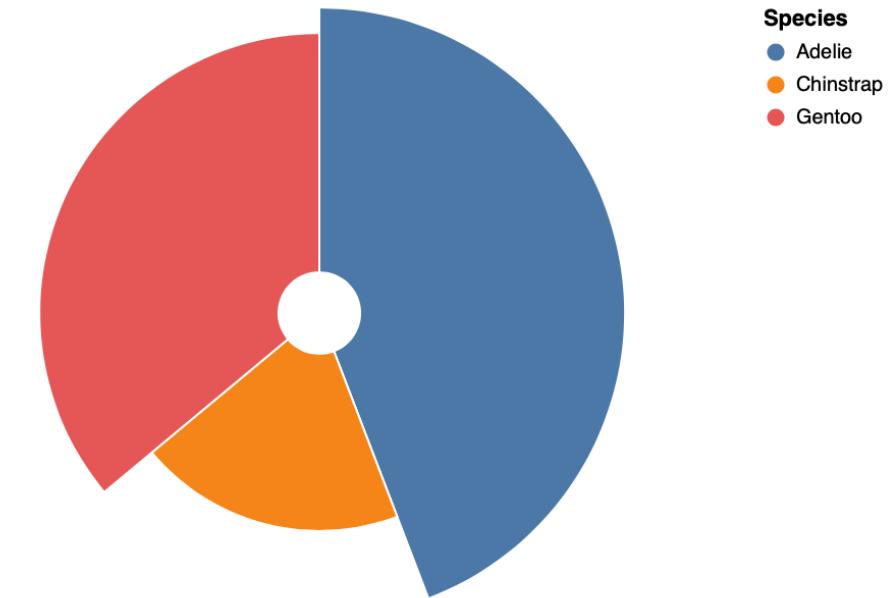
Pie/Donut Charts

- Bad for many small categories
- Consider normalized stacked bar chart as alternative for part-to-whole tasks when you have many categories
 - also take up less space

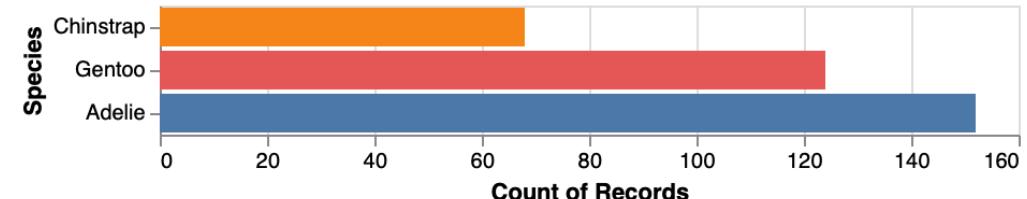


Polar Area Charts

- data
 - 1 categorical key attribute
 - 1 quantitative value attribute
- polar area chart
 - area marks with length channel
 - more direct analog to bar charts (but width increasing)
- task
 - part-to-whole judgements (ok for 2-4 categories)



Number of Penguins by Species

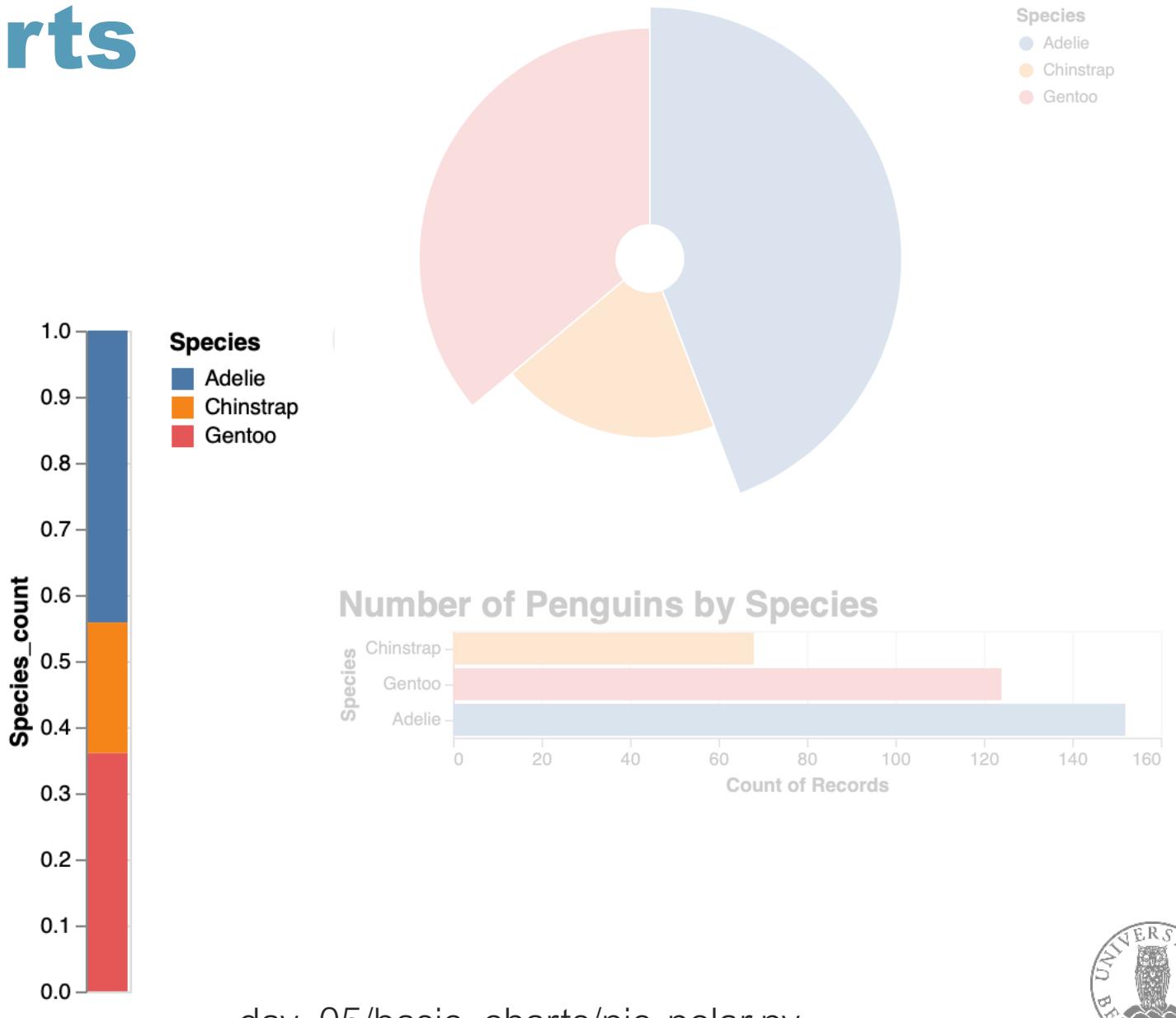


day_05/basic_charts/pie-polar.py



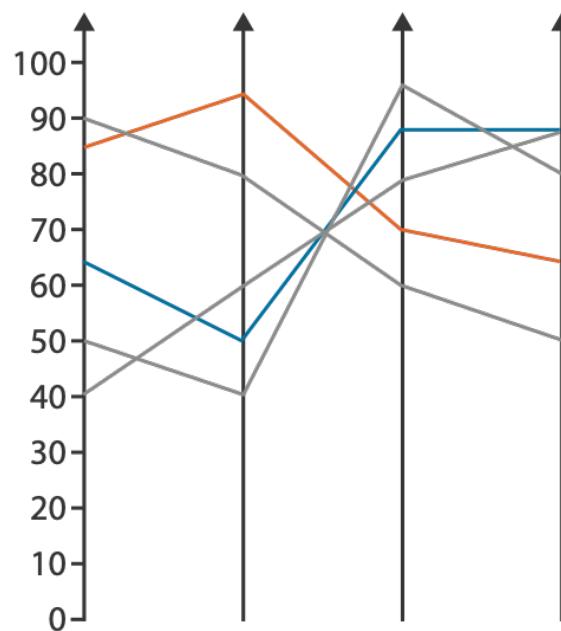
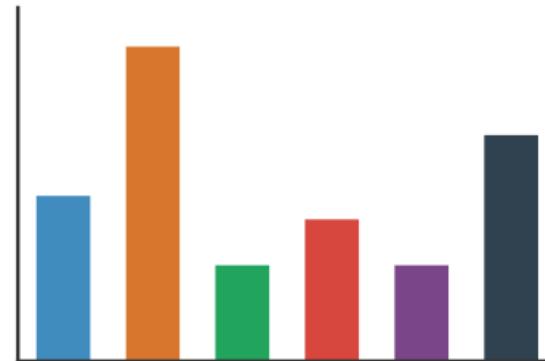
Polar Area Charts

- data
 - 1 categorical key attribute
 - 1 quantitative value attribute
- polar area chart
 - area marks with length channel
 - more direct analog to bar charts (but width increasing)
- task
 - part-to-whole judgements (ok for 2-4 categories)



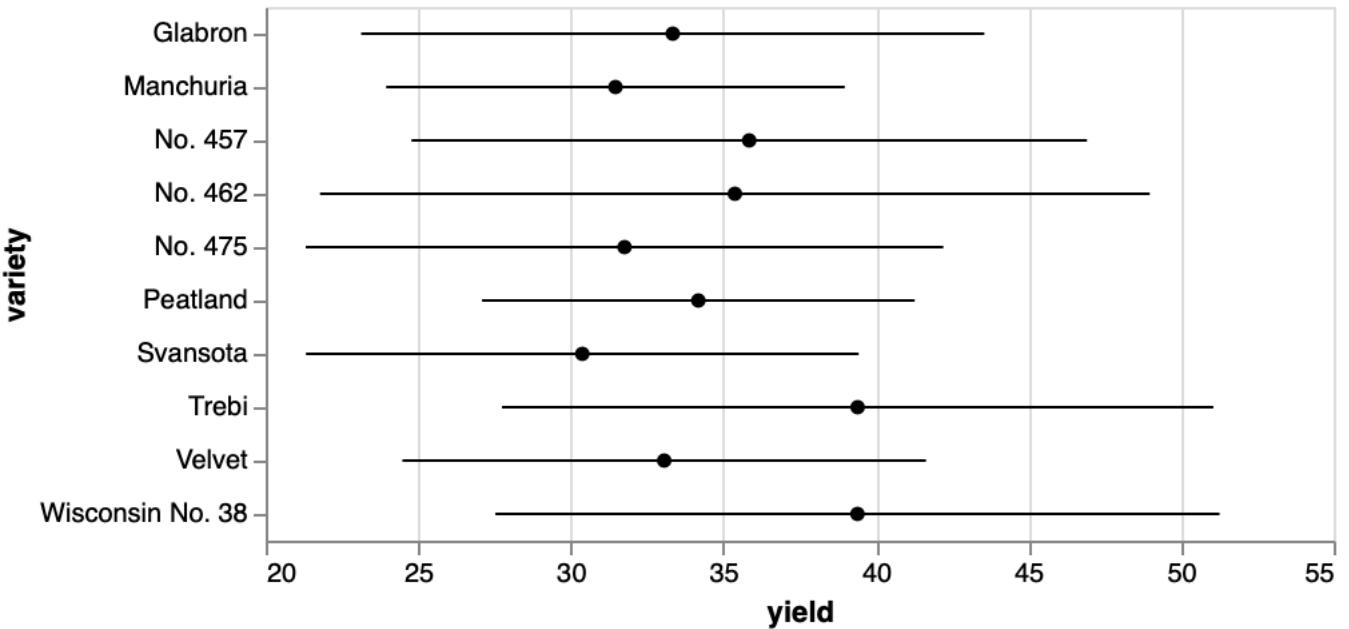
In Practice: Orientation Limitations

- Rectilinear
 - scalability (2 ideal)
- Parallel
 - unfamiliar, longer training time
- Radial
 - arc < length precision
 - length not aligned with radial layouts
 - less accurately perceived compared to rectilinear alignment



Communicating Uncertainty

- Error bars

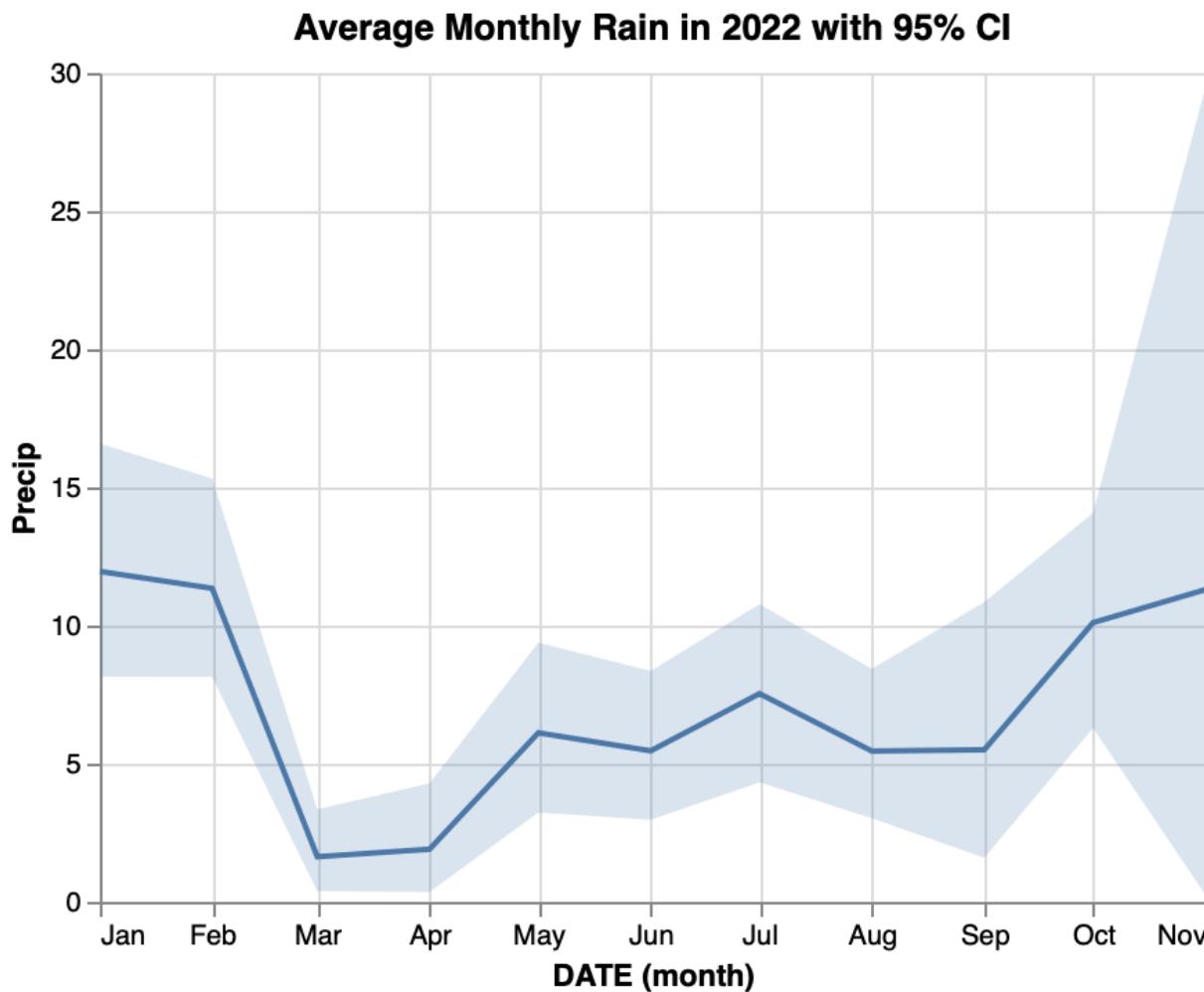


https://altair-viz.github.io/gallery/errorbars_with_std.html



Communicating Uncertainty

- Error bars
- Confidence intervals

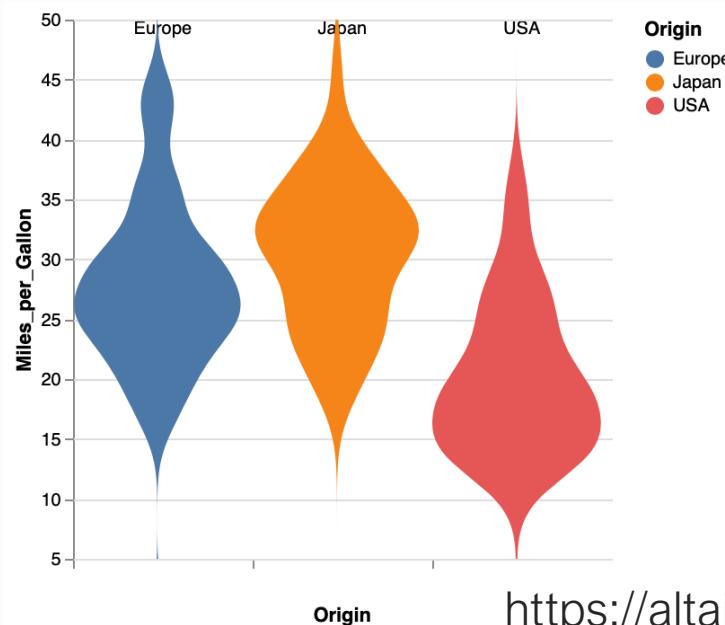


day_05/basic_charts/confidence_interval.py

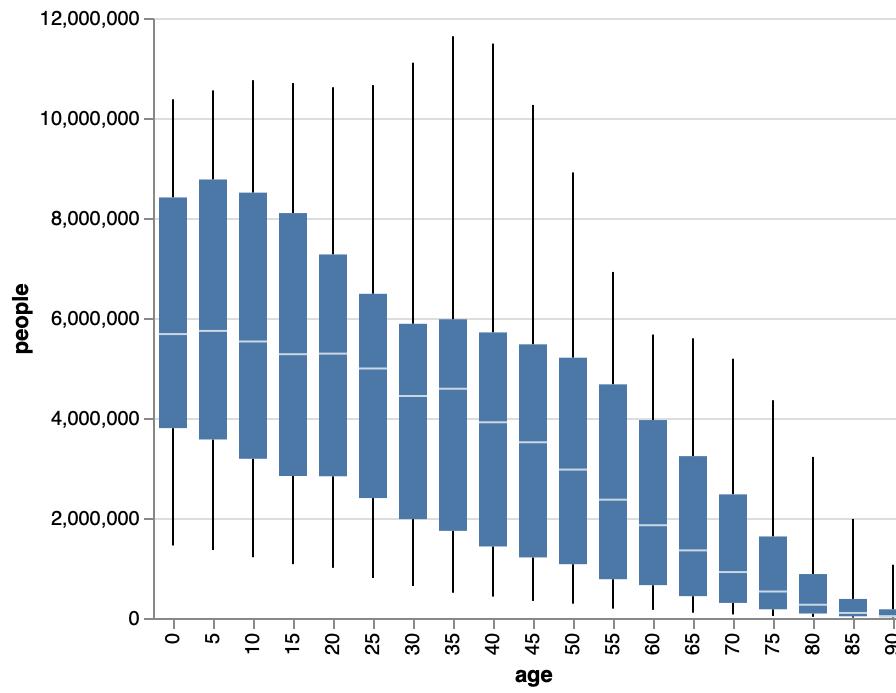


Communicating Uncertainty

- Error bars
- Confidence intervals
- Summary statistics
- Density distribution



https://altair-viz.github.io/gallery/violin_plot.html



<https://altair-viz.github.io/gallery/boxplot.html>



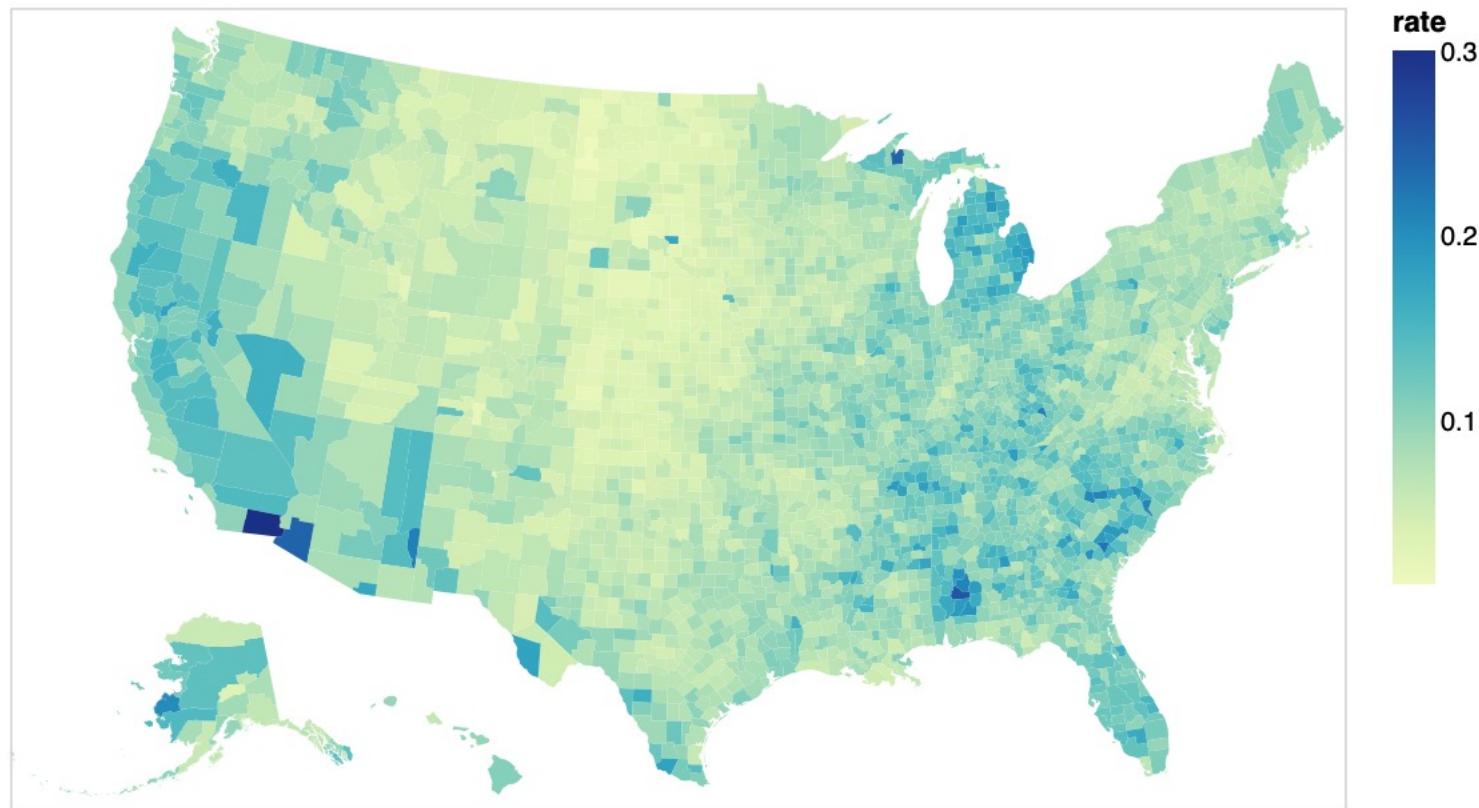
Spatial Data



Chloropleth Map

table with 1 quantitative attribute per region

- encoding
 - geometry for area mark boundaries
 - sequentially segmented color map
- task
 - understanding spatial relationships

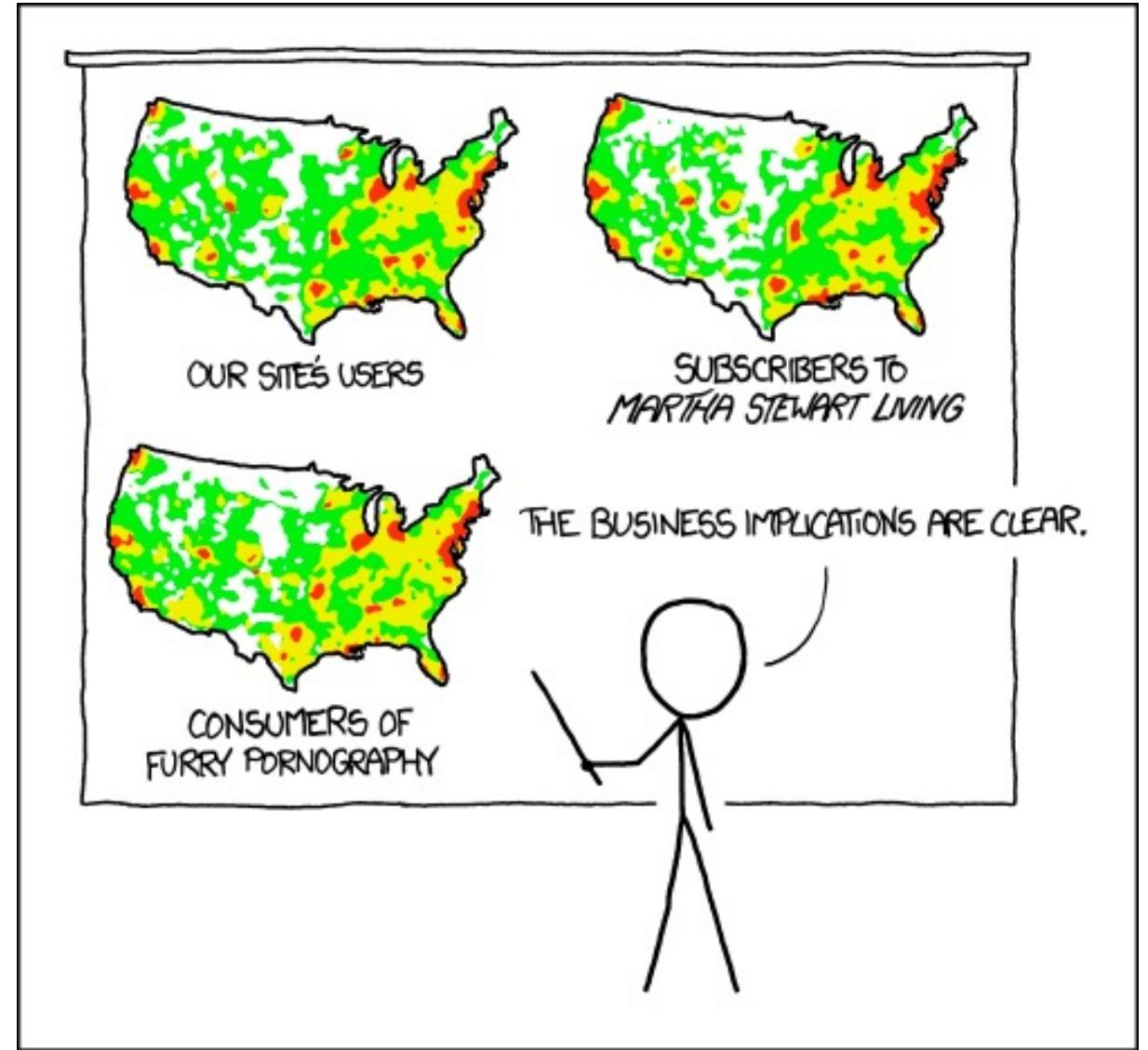


<https://altair-viz.github.io/gallery/choropleth.html>



Chloropleth Map

- Spurious correlations
 - solve by normalizing
 - e.g., Apple users per 100 people
- Large regions (e.g., Texas) emphasized when they maybe shouldn't
 - Consider granularity of color encoding

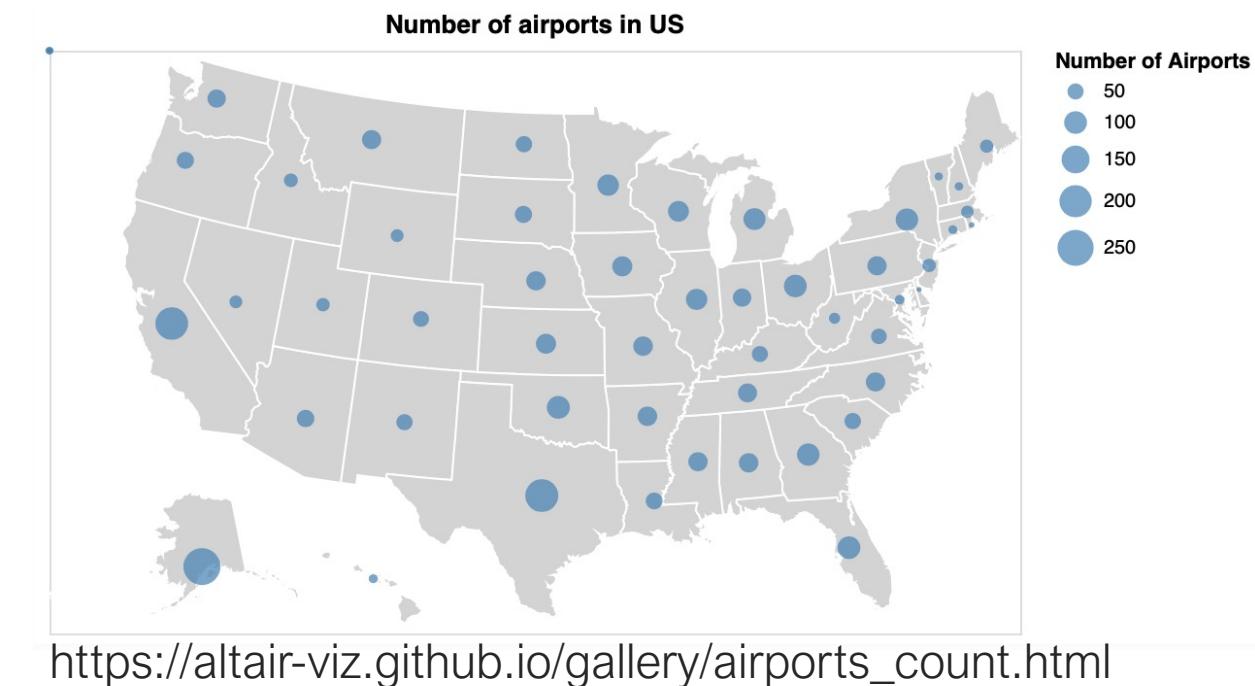


PET PEEVE #208:
GEOGRAPHIC PROFILE MAPS WHICH ARE
BASICALLY JUST POPULATION MAPS

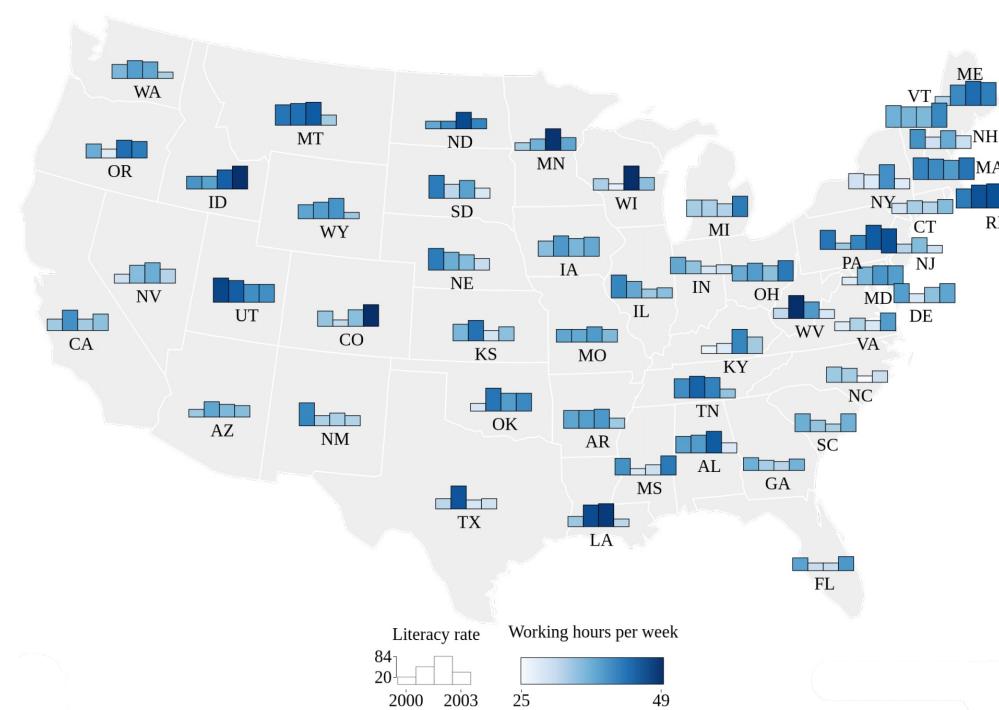
<https://xkcd.com/1138>

Symbol Maps

- Symbol represents aggregated data
- Retain spatial positions in background
- Avoids some of issues with chloropleth map

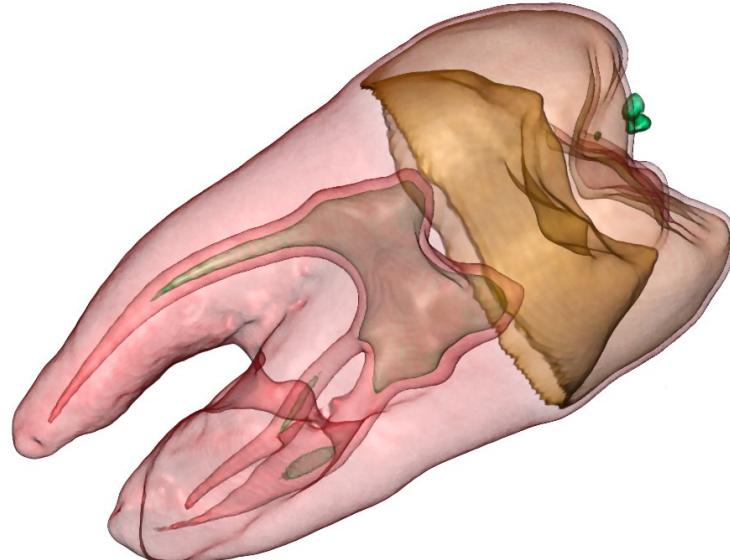


https://altair-viz.github.io/gallery/airports_count.html



Isosurfaces & Direct Volume Rendering

- data
 - scalar spatial field (3D volume)
 - 1 quant attribute per grid cell
- task
 - shape understanding, spatial relationships
- isosurface
 - derived data: isocontours computed for specific levels of scalar values

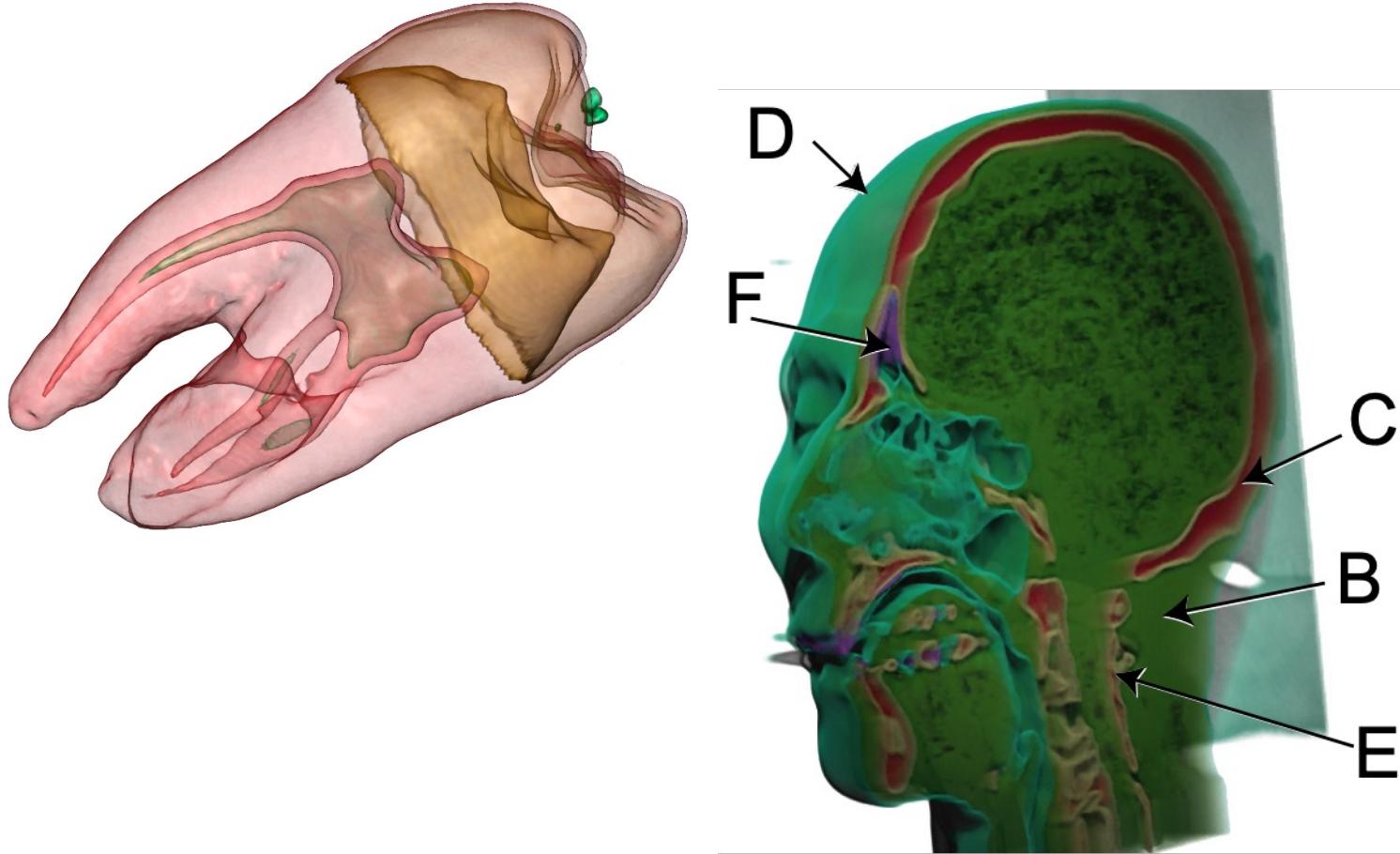


[*Interactive Volume Rendering Techniques*. Kniss. Master's thesis, University of Utah Computer Science, 2002.]

[*Multidimensional Transfer Functions for Volume Rendering*. Kniss, Kindlmann, and Hansen. In *The Visualization Handbook*, edited by Charles Hansen and Christopher Johnson, pp. 189–210. Elsevier, 2005.]

Isosurfaces & Direct Volume Rendering

- data
 - scalar spatial field (3D volume)
 - 1 quant attribute per grid cell
- task
 - shape understanding, spatial relationships
- isosurface
 - derived data: isocontours computed for specific levels of scalar values
- direct volume rendering
 - transfer function maps scalar values to color, opacity
 - no derived geometry



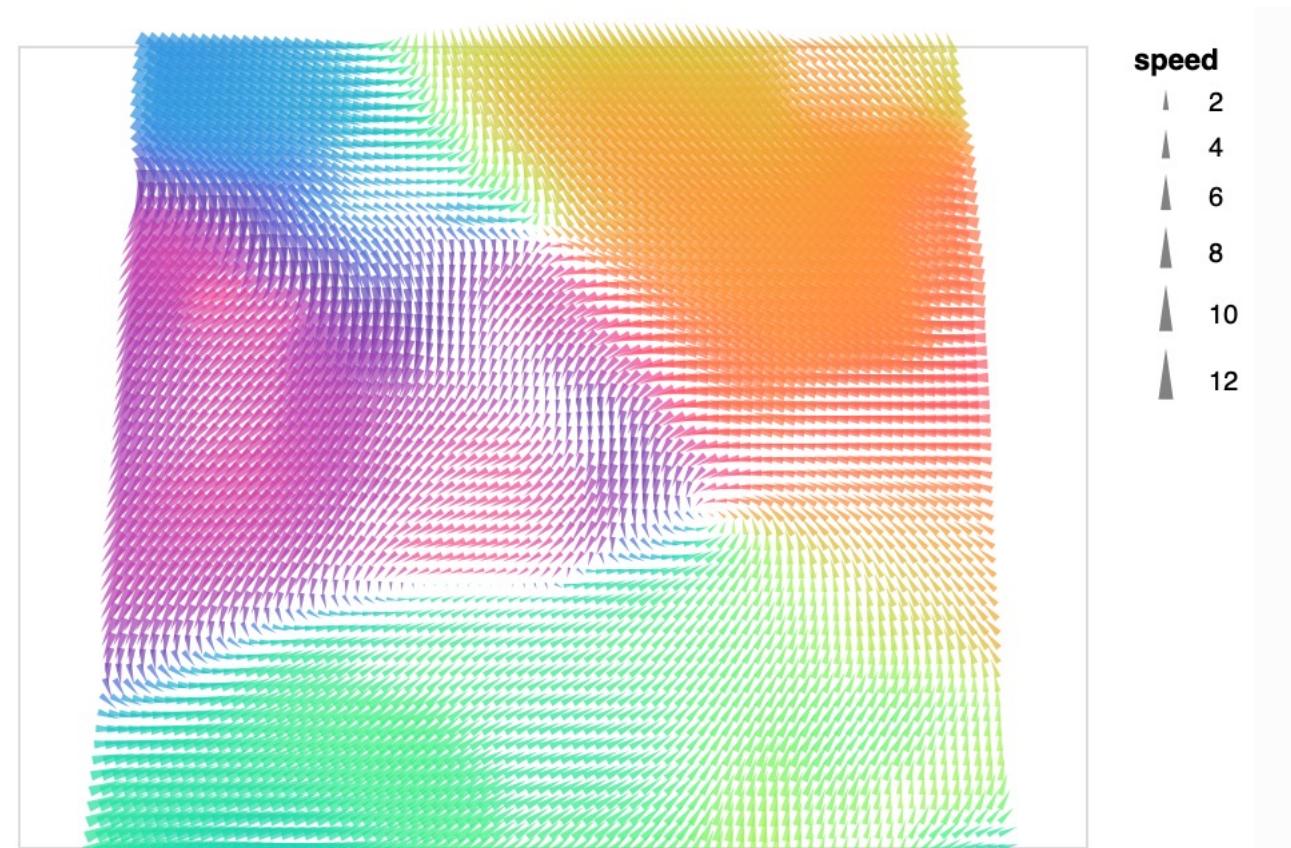
[Interactive Volume Rendering Techniques. Kniss. Master's thesis, University of Utah Computer Science, 2002.]

[Multidimensional Transfer Functions for Volume Rendering. Kniss, Kindlmann, and Hansen. In The Visualization Handbook, edited by Charles Hansen and Christopher Johnson, pp. 189–210. Elsevier, 2005.]



Vector/Tensor Fields

- data
 - multiple attrs per cell
(vector: 2)
- idiom families
 - flow *glyphs*
 - purely local
 - *geometric* flow
 - derived data from tracing particle trajectories
 - sparse set of seed points
 - *texture* flow
 - derived data, dense seeds
 - *feature* flow
 - global computation to detect features



https://altair-viz.github.io/gallery/wind_vector_map.html

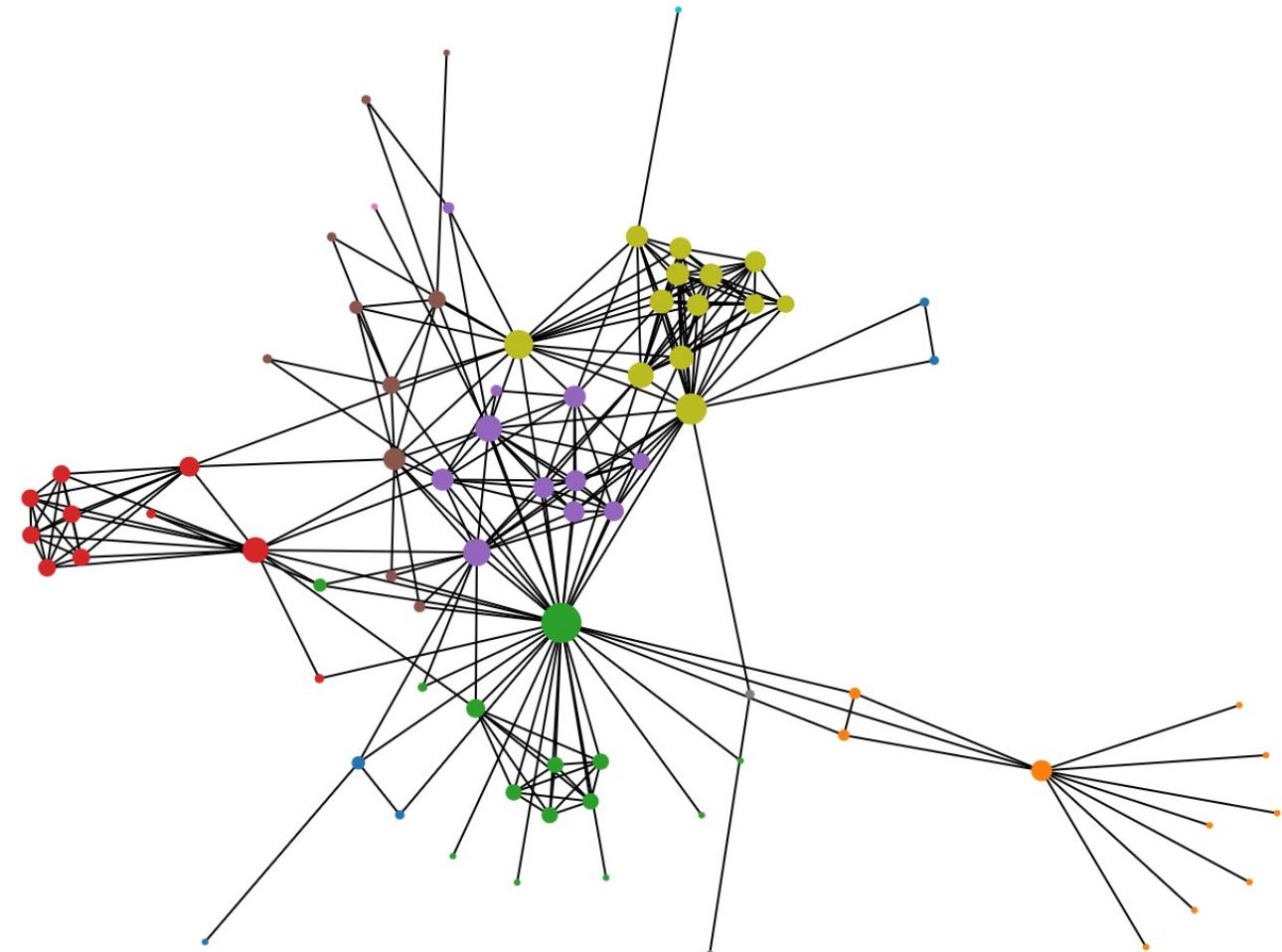
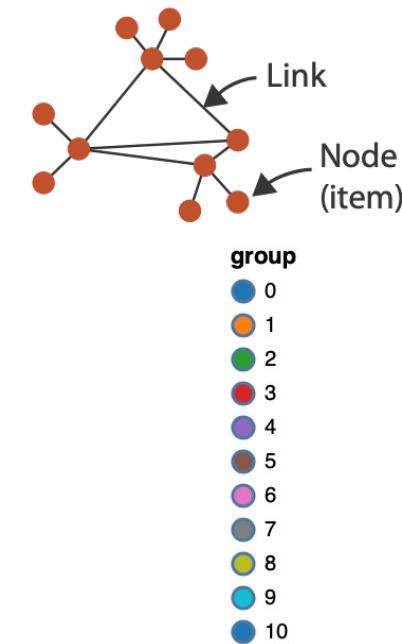


Network & Tree Data



Node-Link

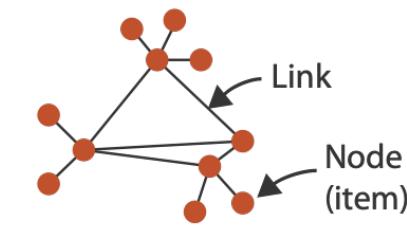
- nodes: point marks
- links: line marks
 - straight lines or arcs
 - connections between nodes
- intuitive & familiar
 - most common
 - many, many variants
- challenges
 - edge crossings
 - space optimization
 - symmetry



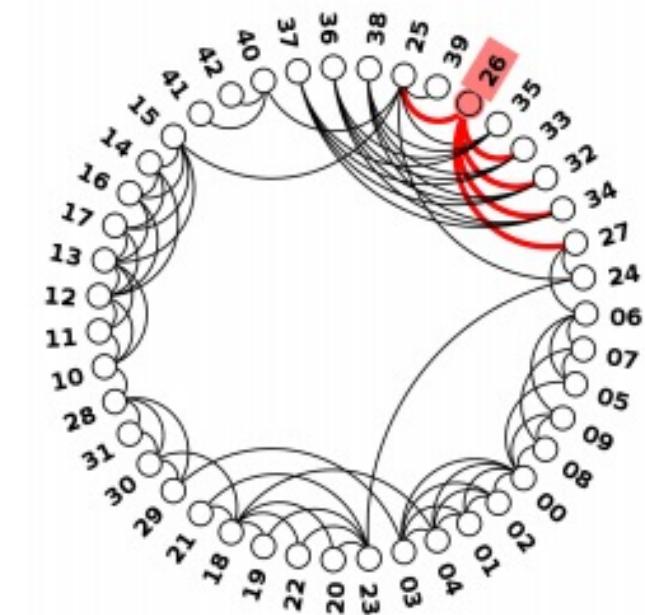
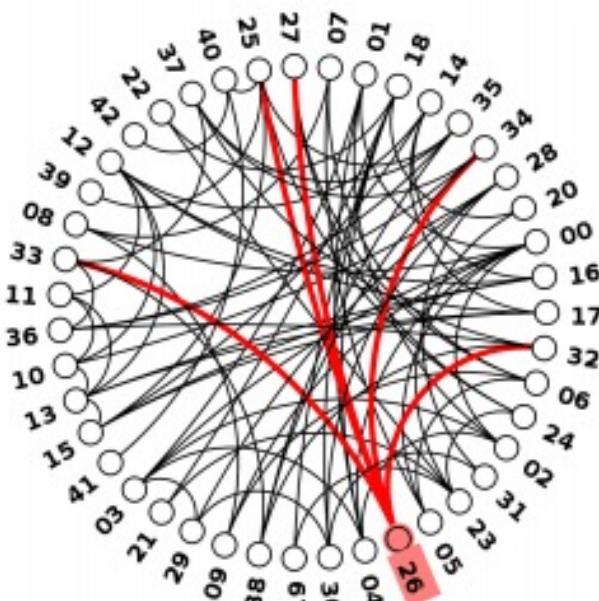
Character Relationships in *Les Misérables*

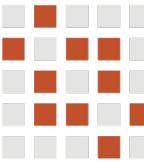
<https://infovis.fh-potsdam.de/tutorials/infovis7networks.html>
day_05/basic_charts/network.py

Node-Link



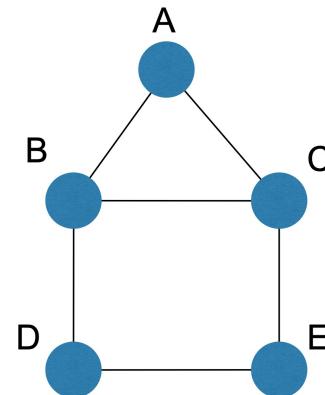
- nodes: point marks
- links: line marks
 - straight lines or arcs
 - connections between nodes
- intuitive & familiar
 - most common
 - many, many variants
- challenges
 - edge crossings
 - space optimization
 - symmetry





Adjacency Matrix

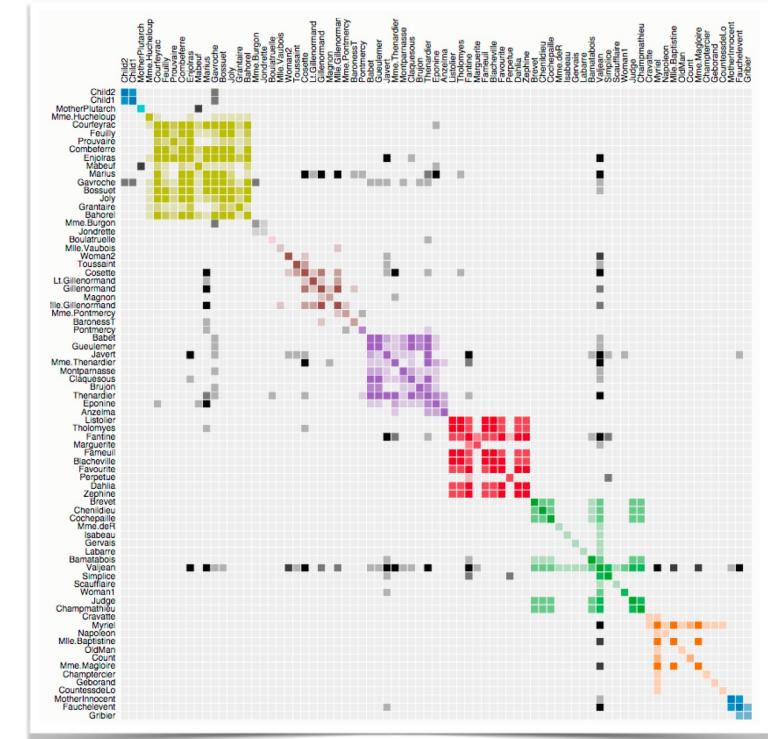
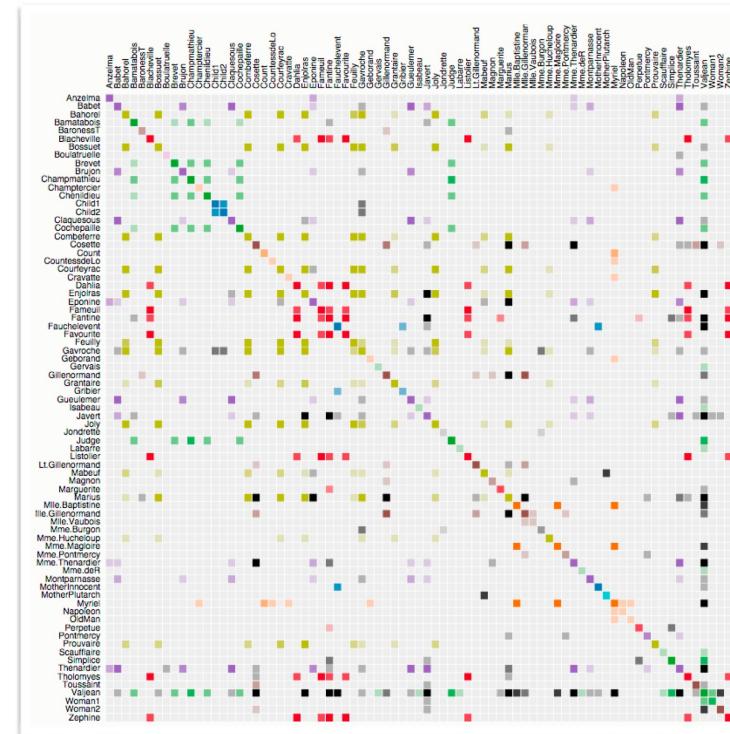
- data: network
 - transform into same data/encoding as heatmap
- derived data: table from network
 - 1 quant attrib
 - weighted edge between nodes
 - 2 categ attribs: node list x 2
- visual encoding
 - cell shows presence/absence of edge
- scalability
 - 1K nodes, 1M edges



	A	B	C	D	E
A					
B					
C					
D					
E					

Adjacency Matrix

Node ordering strategy crucial to find meaningful patterns

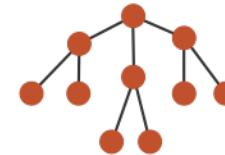


<https://bostocks.org/mike/miserables/>

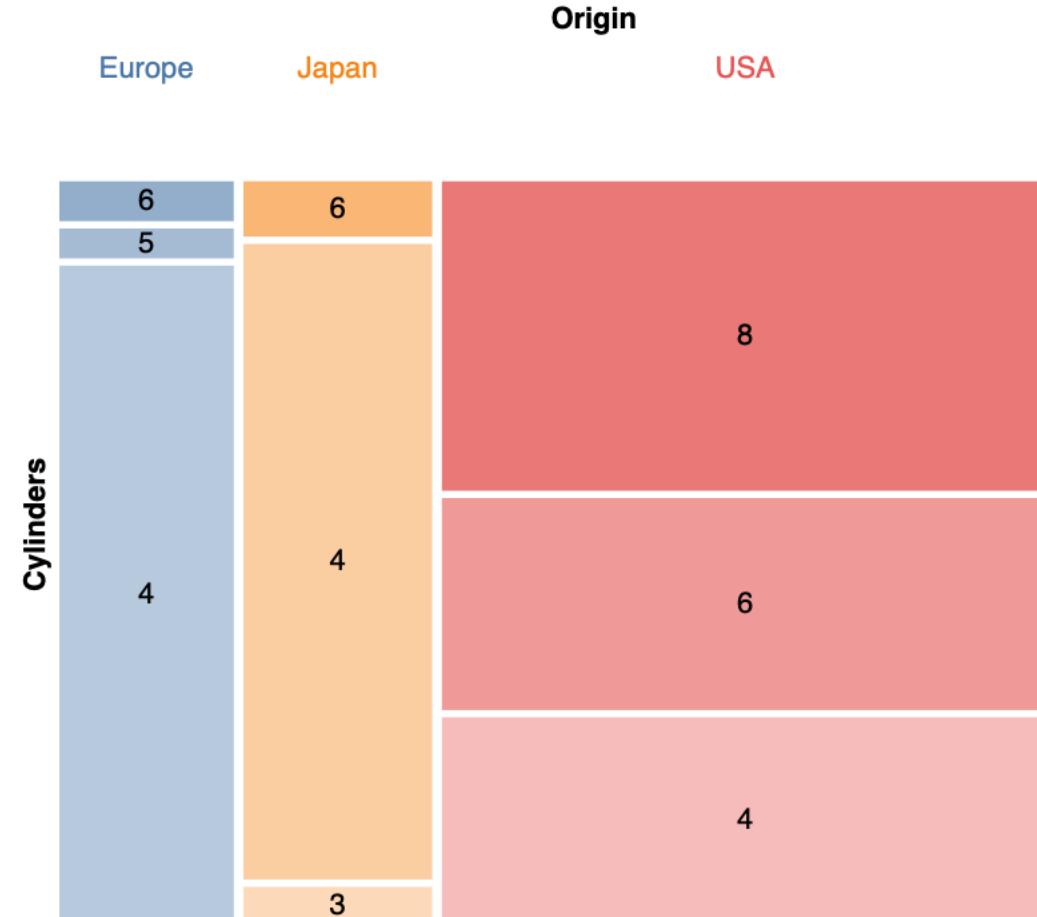


Treemap

- data
 - tree
 - 1 quant attrib at leaf nodes
- encoding
 - area containment marks for hierarchical structure
 - rectilinear orientation
 - size encodes quant attrib
- tasks
 - query attribute at leaf nodes
 - ex: disk space usage within filesystem
- scalability
 - 1M leaf nodes



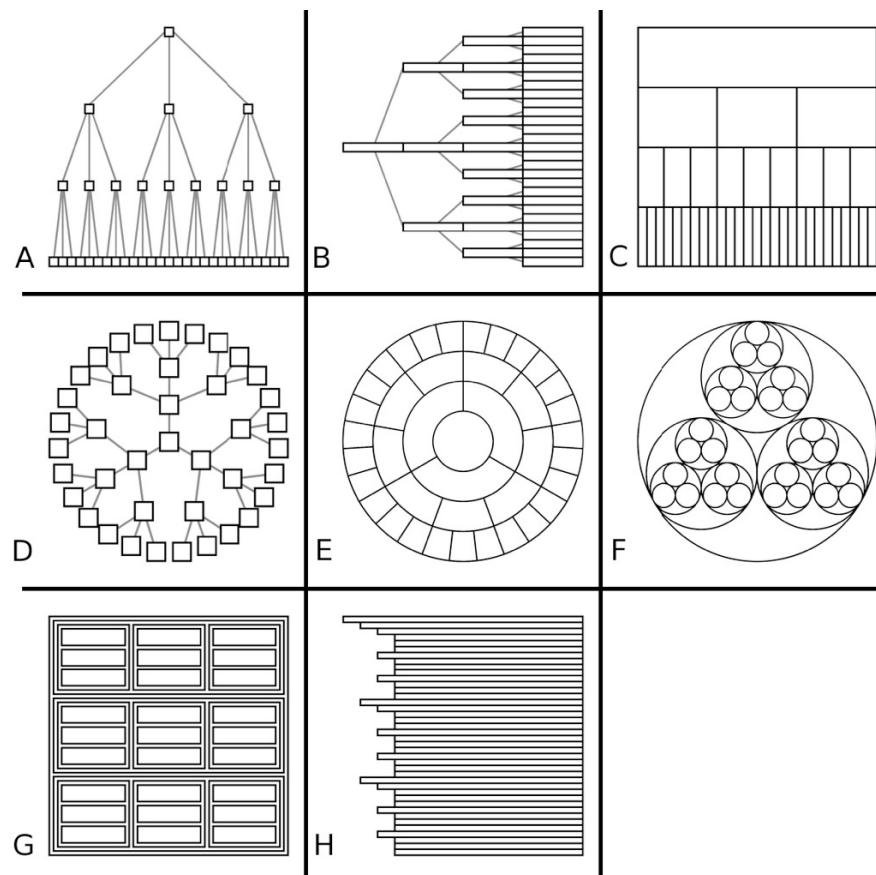
→ **Enclosure**
Containment Marks
✗ NETWORKS ✓ TREES



day_05/basic_charts/treemap.py
https://github.com/altair-viz/altair/blob/master/altair/examples/mosaic_with_labels.py



Tree Visualization



treevis.net - A Visual Bibliography of Tree Visualization 2.0 by Hans-Jörg Schulz

v.21-OCT-2014

Dimensionality: All, 2D, 3D, 4D
Representation: All, Node-Link, Matrix, Circular, Grid, Bar, Network, 3D, Heatmap, Text, Timeline, etc.
Alignment: All, Hierarchical, Radial, Circular, Grid, Bar, Network, 3D, Heatmap, Text, Timeline, etc.
Fulltext Search: x
Techniques Shown: 277

A grid of 277 tree visualization techniques, each shown as a small thumbnail image. The techniques vary greatly in their visual representation and complexity, including hierarchical trees, radial trees, circular trees, grid-based trees, and various 3D and heatmap representations.

treevis.net

Summary



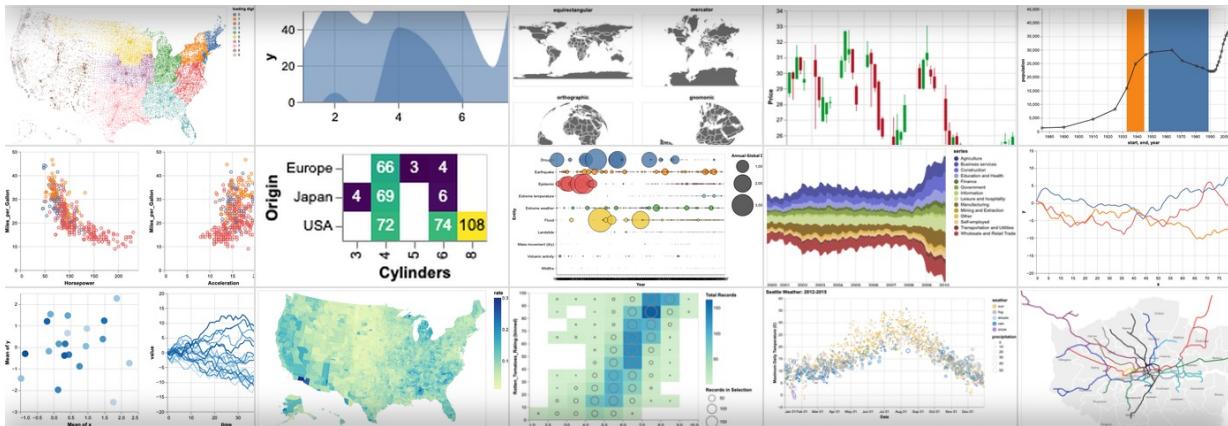
Choosing an Idiom

- Appropriate idioms for your question depend on:
 - the data
 - the task you're trying to accomplish
 - the ranking and effectiveness of marks and channels to visually encode the data
- Experiment!



Next Up

- Introduce Vega-Altair
- Case study: Bergen weather
- Mini-group project:
Weather patterns or
features in your city or
region of choice

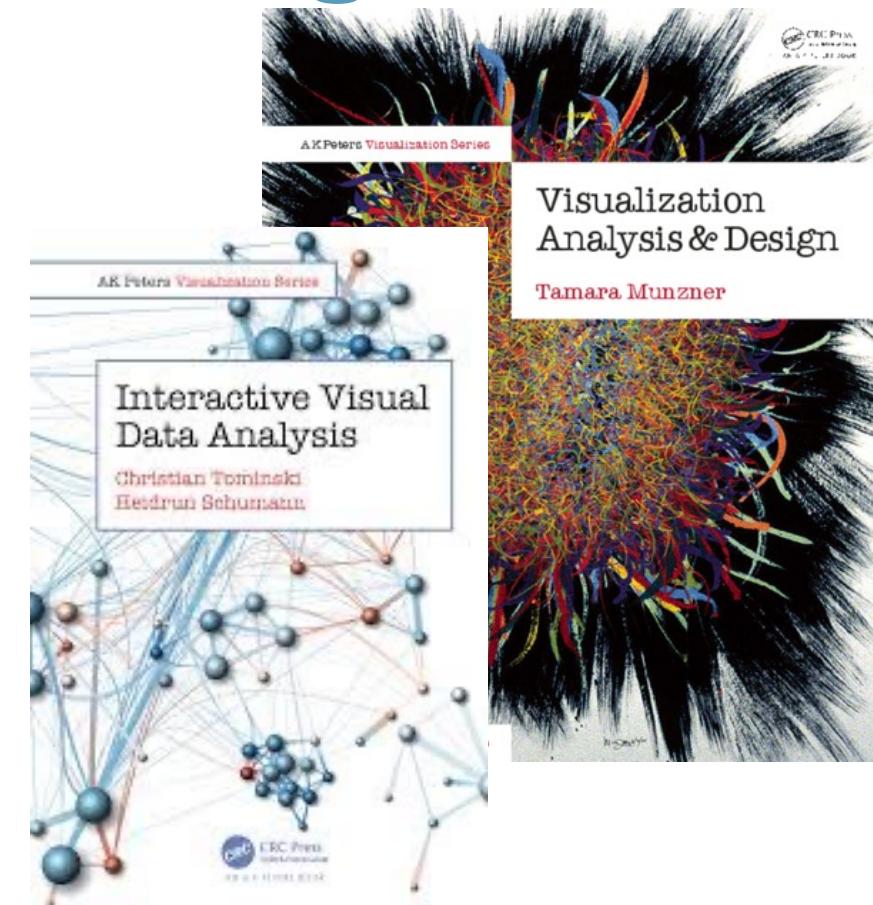


Questions?



Further Reading & Acknowledgement

- Web material for Visual Analysis & Design:
 - <https://www.cs.ubc.ca/~tmm/talks/vadbook>
(source material for many slides in this lecture)
- Interactive Visual Data Analysis
- Vega-Altair: Declarative Visualization in Python
 - <https://altair-viz.github.io/index.html>
- Exploring and Analyzing Network Data with Python
 - <https://programminghistorian.org/en/lessons/exploring-and-analyzing-network-data-with-python>



Munzner, T. (2014). Visualization analysis and design. AK Peters Visualization Series, CRC Press, Visualization Series
Tominski, T., Schumann, H. (2020). Interactive Visual Data Analysis. AK Peters Visualization Series, CRC Press, Visualization Series.

Arrange spatial data

→ Use Given

→ Geometry

→ *Geographic*

!

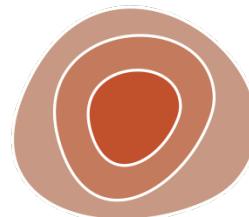


→ Spatial Fields

→ *Scalar Fields (one value per cell)*

→ *Isocontours*

→ *Direct Volume Rendering*



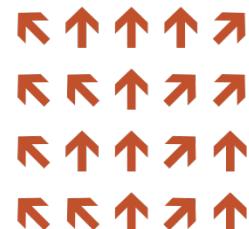
→ *Vector and Tensor Fields (many values per cell)*

→ *Flow Glyphs (local)*

→ *Geometric (sparse seeds)*

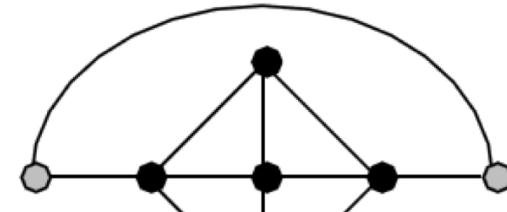
→ *Textures (dense seeds)*

→ *Features (globally derived)*



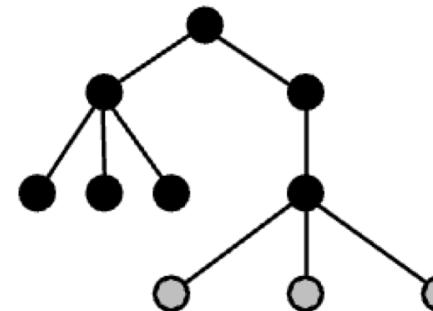
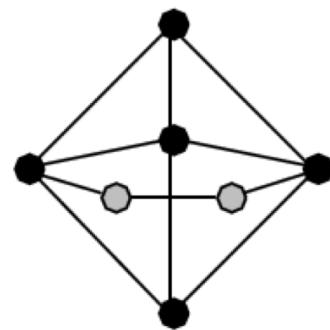
Network Vis Criteria Conflicts

Minimum number
of edge crossings



vs.

Uniform edge
length



Space utilization

vs.

Symmetry

