Ch 7/10: Tables, Color
Paper: D3

Tamara Munzner
Department of Computer Science
University of British Columbia

CPSC 547, Information Visualization
Week 4: 1 October 2019

http://www.cs.ubc.ca/~tmm/courses/547-19
News

• marks out for week 2 & 3
  – mostly 5 (full credit)
  – some 4s (comments don't show depth of understanding of material)
  – a few 0s (didn't hand in)
This Time

• wrap up Decoding exercise (from last time)
• 3 shorter in-class exercises
  – Two Numbers
  – Bars/Radial
  – Color Palettes
• paper types (carryforward from last time)
• paper: D3
  – system context
• chapters: Tables, Color
  – some new material, not just backup slides
• pitches: expectations
Paper: D3 System
Paper: D3

• paper types
  – design studies
  – technique/algorithm
  – evaluation
  – model/taxonomy
  – system

Toolkits

• imperative: how
  – low-level rendering: Processing, OpenGL
  – parametrized visual objects: prefuse
    • also flare: prefuse for Flash

• declarative: what
  – Protoviz, D3, ggplot2
  – separation of specification from execution

• considerations
  – expressiveness
    • can I build it?
  – efficiency
    • how long will it take?
  – accessibility
    • do I know how?
WebGL/OpenGL

- graphics library
  - pros
    - power and flexibility, complete control for graphics
    - hardware acceleration
    - many language bindings: js, C, C++, Java (w/ JOGL)
  - cons
    - big learning curve if you don’t know already
    - no vis support, must roll your own everything
  - example app: TreeJuxtaposer (OpenGL)

Processing / p5.js

• layer on top of Java/OpenGL, Javascript/WebGL
• visualization esp. for artists/designers

• pros
  – great sandbox for rapid prototyping
  – huge user community, great documentation

• cons
  – poor widget library support

• example app: MizBee
prefuse

• infovis toolkit, in Java
• fine-grained building blocks for tailored visualizations
• pros
  – heavily used (previously)
  – very powerful abstractions
  – quickly implement most techniques covered so far
• cons
  – no longer active
  – nontrivial learning curve
• example app: DOI Trees Revisited

• separation: abstract data, visual form, view
  – data: tables, networks
  – visual form: layout, color, size, ...
  – view: multiple renderers

InfoVis Reference Model

- conceptual model underneath design of prefuse and many other toolkits
- heavily influenced much of infovis (including nested model)
  –aka infovis pipeline, data state model

[Redrawn Fig 1.23. Card, Mackinlay, and Shneiderman. Readings in Information Visualization: Using Vision To Think, Chapter 1. Morgan Kaufmann, 1999.]
Declarative toolkits

• imperative tools/libraries
  – say exactly **how** to do it
  – familiar programming model
    • OpenGL, prefuse, ...

• declarative: other possibility
  – just say **what** to do
  – Protovis, D3
Protovis

• declarative infovis toolkit, in Javascript
  – also later Java version
• marks with inherited properties
• pros
  – runs in browser
  – matches mark/channel mental model
  – also much more: interaction, geospatial, trees,...
• cons
  – not all kinds of operations supported
• example app: NapkinVis (2009 course project)

[Fig 1, 3. Chao. NapkinVis. http://www.cs.ubc.ca/~tmm/courses/533-09/projects.html#will]
Protovis Validation

• wide set of old/new app examples
  – expressiveness, effectiveness, scalability
  – accessibility

• analysis with cognitive dimensions of notation
  – closeness of mapping, hidden dependencies
  – role-expressiveness visibility, consistency
  – viscosity, diffuseness, abstraction
  – hard mental operations

D3

• declarative infovis toolkit, in Javascript
• Protovis meets Document Object Model
• pros
  – seamless interoperability with Web
  – explicit transforms of scene with dependency info
  – massive user community, many thirdparty apps/libraries on top of it, lots of docs
• cons
  – even more different from traditional programming model
• example apps: many
D3

• objectives
  – compatibility
  – debugging
  – performance

• related work typology
  – document transformers
  – graphics libraries
  – infovis systems
    • general note: all related work sections are a mini-taxonomy/typology!

D3 capabilities

• query-driven selection
  – selection: filtered set of elements queries from the current doc
    • also partitioning/grouping!
  – operators act on selections to modify content
    • instantaneous or via animated transitions with attribute/style interpolators
    • event handlers for interaction

• data binding to scenegraph elements
  – data joins bind input data to elements
  – enter, update, exit subselections
  – sticky: available for subsequent re-selection
  – sort, filter

D3 Features

- document transformation as atomic operation
  - scene changes vs representation of scenes themselves
- immediate property evaluation semantics
  - avoid confusing consequences of delayed evaluation

- validation
  - performance benchmarks
    - page loads, frame rate
  - accessibility
  - (adoption)
    - everybody has voted with their feet by now!
Ch 7: Arrange Tables
Encode

Arrange

Express

Separate

Order

Align

Use
How?

<table>
<thead>
<tr>
<th>Encode</th>
<th>Manipulate</th>
<th>Facet</th>
<th>Reduce</th>
</tr>
</thead>
<tbody>
<tr>
<td>‣ Arrange</td>
<td>‣ Change</td>
<td>‣ Juxtapose</td>
<td>‣ Filter</td>
</tr>
<tr>
<td>‣ Express</td>
<td>→ Hue</td>
<td>‣ Partition</td>
<td>‣ Aggregate</td>
</tr>
<tr>
<td>‣ Separate</td>
<td>→ Saturation</td>
<td>‣ Superimpose</td>
<td>‣ Embed</td>
</tr>
<tr>
<td>‣ Order</td>
<td>‣ Luminance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>‣ Align</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>‣ Use</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Map from categorical and ordered attributes

- Color
  - Hue
  - Saturation
  - Luminance

- Size, Angle, Curvature, ...

- Shape
  - +
  - •
  - □
  - ▲

- Motion
  - Direction, Rate, Frequency, ...

What?

Why?

How?
Encode tables: Arrange space

Encode

Arrange

Express

Separate

Order

Align

Why?

How?

What?

Encode tables: Arrange space
Arrange tables

- Express Values
- Separate, Order, Align Regions
  - Separate
  - Order
  - Align
- Axis Orientation
  - Rectilinear
  - Parallel
  - Radial
- Layout Density
  - Dense
  - Space-Filling
- 1 Key
- 2 Keys
- 3 Keys
- Many Keys
  - List
  - Matrix
  - Volume
  - Recursive Subdivision
Keys and values

• **key**
  – independent attribute
  – used as unique index to look up items
  – simple tables: 1 key
  – multidimensional tables: multiple keys

• **value**
  – dependent attribute, value of cell

• classify arrangements by key count
  – 0, 1, 2, many...

Express Values

© 1 Key
© 2 Keys
© 3 Keys
© Many Keys

- List
- Matrix
- Volume
- Recursive Subdivision

Multidimensional Table

Tables

- Attributes (columns)
- Items (rows)
- Cell containing value

Dataset Types

- Tables
- Attributes (columns)
- Items (rows)
- Cell containing value

- Networks
- Link
- Node (item)

- Trees
- Fields (Continuous)
- Value in cell

- Geometry (Spatial)
- Position

- Grid of positions
**Idiom:** scatterplot

- **express** values
  - quantitative attributes
- no keys, only values
  - data
    - 2 quant attribs
  - mark: points
  - channels
    - horiz + vert position
- tasks
  - find trends, outliers, distribution, correlation, clusters
- scalability
  - hundreds of items

---

Some keys: Categorical regions

- **regions**: contiguous bounded areas distinct from each other
  - using space to **separate** (proximity)
  - following expressiveness principle for categorical attributes
- use ordered attribute to **order** and **align** regions

- **1 Key**
  - List
  - 1

- **2 Keys**
  - Matrix
  - 2

- **3 Keys**
  - Volume
  - 3

- **Many Keys**
  - Recursive Subdivision
  - Many
Idiom: **bar chart**

- one key, one value
  - data
    - 1 categ attrib, 1 quant attrib
  - mark: lines
  - channels
    - length to express quant value
    - spatial regions: one per mark
      - separated horizontally, aligned vertically
      - ordered by quant attrib
        » by label (alphabetical), by length attrib (data-driven)
  - task
    - compare, lookup values
  - scalability
    - dozens to hundreds of levels for key attrib
Separated and Aligned but not Ordered

LIMITATION: Hard to know rank. What’s the 4th most? The 7th?

[Slide courtesy of Ben Jones]
LIMITATION: Hard to make comparisons

[Slide courtesy of Ben Jones]
Idiom: **stacked bar chart**

- one more key
  - data
    - 2 categ attrib, 1 quant attrib
  - mark: vertical stack of line marks
    - **glyph**: composite object, internal structure from multiple marks
  - channels
    - length and color hue
    - spatial regions: one per glyph
      - aligned: full glyph, lowest bar component
      - unaligned: other bar components
  - task
    - part-to-whole relationship
  - scalability
    - several to one dozen levels for stacked attrib

Idiom: **streamgraph**

- generalized stacked graph
  - emphasizing horizontal continuity
    - vs vertical items
- data
  - 1 categ key attrib (artist)
  - 1 ordered key attrib (time)
  - 1 quant value attrib (counts)
- derived data
  - geometry: layers, where height encodes counts
  - 1 quant attrib (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

Idiom: line chart / dot plot

• one key, one value
  – data
    • 2 quant attribs
  – mark: points
    • line connection marks between them
  – channels
    • aligned lengths to express quant value
    • separated and ordered by key attrib into horizontal regions
  – task
    • find trend
      – connection marks emphasize ordering of items along key axis by explicitly showing relationship between one item and the next
  – scalability
    • hundreds of key levels, hundreds of value levels
Choosing bar vs line charts

- depends on type of key attrib
  - bar charts if categorical
  - line charts if ordered
- do not use line charts for categorical key attribs
  - violates expressiveness principle
  - implication of trend so strong that it overrides semantics!
    - “The more male a person is, the taller he/she is”

Chart axes

• labelled axis is critical
• avoid cropping y-axis
  – include 0 at bottom left
  – or slope misleads

• dual axes controversial
  – acceptable if commensurate
  – beware, very easy to mislead!

http://www.thefunctionalart.com/2015/10/if-you-see-bullshit-say-bullshit.html
Idiom: connected scatterplots

- scatterplot with line connection marks
  - popular in journalism
  - horiz + vert axes: value attribs
  - line connection marks: temporal order
  - alternative to dual-axis charts
    - horiz: time
    - vert: two value attribs

- empirical study
  - engaging, but correlation unclear


http://steveharoz.com/research/connected_scatterplot/
Idiom: **Indexed line charts**

- **data**: 2 quant attires
  - 1 key + 1 value
- **derived data**: new quant value attrib
  - index
  - plot instead of original value
- **task**: show change over time
  - principle: normalized, not absolute
- **scalability**
  - same as standard line chart
Idiom: **Gantt charts**

- one key, two (related) values
  - data
    - 1 categ attrib, 2 quant attribs
  - mark: line
    - length: duration
- channels
  - horiz position: start /end times
  - horiz length: duration
- task
  - emphasize temporal overlaps, start/end dependencies between items
- scalability
  - dozens of key levels
  - hundreds of value levels

Idiom: heatmap

- two keys, one value
  - data
    - 2 categ attrs (gene, experimental condition)
    - 1 quant attrib (expression levels)
  - marks: area
    - separate and align in 2D matrix
      - indexed by 2 categorical attributes
  - channels
    - color by quant attrib
      - (ordered diverging colormap)
  - task
    - find clusters, outliers
- scalability
  - 1M items, 100s of categ levels, ~10 quant attrib levels
Idiom: **cluster heatmap**

- **in addition**
  - derived data
    - 2 cluster hierarchies
  - dendrogram
    - parent-child relationships in tree with connection line marks
    - leaves aligned so interior branch heights easy to compare
- heatmap
  - marks (re-)ordered by cluster hierarchy traversal
Axis Orientation

- Rectilinear
- Parallel
- Radial
Idioms: **scatterplot matrix, parallel coordinates**

- **scatterplot matrix (SPLOM)**
  - rectilinear axes, point mark
  - all possible pairs of axes
  - scalability
    - one dozen attribs
    - dozens to hundreds of items

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

Task: Correlation

- scatterplot matrix
  - positive correlation
    - diagonal low-to-high
    - negative correlation
    - diagonal high-to-low
  - uncorrelated
- parallel coordinates
  - positive correlation
    - parallel line segments
  - negative correlation
    - all segments cross at halfway point
  - uncorrelated
    - scattered crossings

Idioms: **radial bar chart, star plot**

- **radial bar chart**
  - radial axes meet at central ring, line mark

- **star plot**
  - radial axes, meet at central point, line mark

- **bar chart**
  - rectilinear axes, aligned vertically

- **accuracy**
  - length unaligned with radial
    - less accurate than aligned with rectilinear

Idioms: **pie chart, polar area chart**

- **pie chart**
  - area marks with angle channel
  - accuracy: angle/area less accurate than line length
    - arclength also less accurate than line length

- **polar area chart**
  - area marks with length channel
  - more direct analog to bar charts

- **data**
  - 1 categ key attrib, 1 quant value attrib

- **task**
  - part-to-whole judgements

Idioms: **normalized stacked bar chart**

- task
  - part-to-whole judgements

- normalized stacked bar chart
  - stacked bar chart, normalized to full vert height
  - single stacked bar equivalent to full pie
    - high information density: requires narrow rectangle

- pie chart
  - information density: requires large circle

[http://bl.ocks.org/mbostock/3887235](http://bl.ocks.org/mbostock/3887235)
[http://bl.ocks.org/mbostock/3886208](http://bl.ocks.org/mbostock/3886208)
[http://bl.ocks.org/mbostock/3886394](http://bl.ocks.org/mbostock/3886394)
Idiom: **glyphmaps**

- Rectilinear good for linear vs nonlinear trends

- Radial good for cyclic patterns

![Diagram of glyphmaps]

Orientation limitations

• rectilinear: scalability wrt #axes
  • 2 axes best
  • 3 problematic
    – more in afternoon
  • 4+ impossible

• parallel: unfamiliarity, training time

• radial: perceptual limits
  – angles lower precision than lengths
  – asymmetry between angle and length
    • can be exploited!

Layout Density

- Dense


dense software overviews
Ch 10: Map Color and Other Channels
VAD Chap 10: Map Color and Other Channels

 Encode > Map

- **Color**
  - Color Encoding
    - Hue
    - Saturation
    - Luminance
  - Color Map
    - Categorical
    - Ordered
      - Sequential
      - Diverging
    - Bivariate

- **Size, Angle, Curvature, ...**
  - Length
  - Angle
  - Area
  - Curvature
  - Volume

- **Shape**
  - + ● ■ ▲

- **Motion**
  - Motion
    - Direction, Rate, Frequency, ...
Categorical vs ordered color

Decomposing color

• first rule of color: do not talk about color!
  – color is confusing if treated as monolithic

• decompose into three channels
  – ordered can show magnitude
    • luminance: how bright
    • saturation: how colorful
  – categorical can show identity
    • hue: what color

• channels have different properties
  – what they convey directly to perceptual system
  – how much they can convey: how many discriminable bins can we use?
Spectral sensitivity

![Graph showing spectral sensitivity with wavelength in nanometers (nm) on the x-axis and relative sensitivity on the y-axis, with visible, ultraviolet (UV), and infrared (IR) spectrums indicated.]
Luminance

• need luminance for edge detection
  – fine-grained detail only visible through luminance contrast
  – legible text requires luminance contrast!

• intrinsic perceptual ordering

Opponent color and color deficiency

- perceptual processing before optic nerve
  - one achromatic luminance channel (L*)
  - edge detection through luminance contrast
  - 2 chroma channels
    - red-green (a*) & yellow-blue axis (b*)
- “color blind”: one axis has degraded acuity
  - 8% of men are red/green color deficient
  - blue/yellow is rare

Color spaces

- CIE $L^*a^*b^*$: good for computation
  - $L^*$ intuitive: perceptually linear luminance
  - $a^*b^*$ axes: perceptually linear but nonintuitive
- RGB: good for display hardware
  - poor for encoding
- HSL/HSV: somewhat better for encoding
  - hue/saturation wheel intuitive
  - beware: only pseudo-perceptual!
  - lightness (L) or value (V) $\neq$ luminance or $L^*$
- Luminance, hue, saturation
  - good for encoding
  - but not standard graphics/tools colorspace
Designing for color deficiency: Check with simulator

Normal vision  Deuteranope  Protanope  Tritanope

http://rehue.net

Designing for color deficiency: Avoid encoding by hue alone

• redundantly encode
  – vary luminance
  – change shape

Color deficiency: Reduces color to 2 dimensions

Normal

Protanope

Deuteranope

Tritanope

Designing for color deficiency: Blue-Orange is safe
Bezold Effect: Outlines matter

• color constancy: simultaneous contrast effect
Color/Lightness constancy: Illumination conditions

Image courtesy of John McCann
Color/Lightness constancy: Illumination conditions

Image courtesy of John McCann
Categorical color: limited number of discriminable bins

• human perception built on relative comparisons
  – great if color contiguous
  – surprisingly bad for absolute comparisons

• noncontiguous small regions of color
  – fewer bins than you want
  – rule of thumb: 6-12 bins, including background and highlights

[Cinteny: flexible analysis and visualization of synteny and genome rearrangements in multiple organisms. Sinha and Meller. BMC Bioinformatics, 8:82, 2007.]
ColorBrewer

- [http://www.colorbrewer2.org](http://www.colorbrewer2.org)
- saturation and area example: size affects salience!
Ordered color: Rainbow is poor default

• problems
  – perceptually unordered
  – perceptually nonlinear

• benefits
  – fine-grained structure visible and nameable
Ordered color: Rainbow is poor default

• problems
  – perceptually unordered
  – perceptually nonlinear

• benefits
  – fine-grained structure visible and nameable

• alternatives
  – large-scale structure: fewer hues


Ordered color: Rainbow is poor default

• problems
  – perceptually unordered
  – perceptually nonlinear

• benefits
  – fine-grained structure visible and nameable

• alternatives
  – large-scale structure: fewer hues
  – fine structure: multiple hues with monotonically increasing luminance [eg viridis R/python]
Viridis

- colorful, perceptually uniform, colorblind-safe, monotonically increasing luminance

https://cran.r-project.org/web/packages/viridis/vignettes/intro-to-viridis.html
Ordered color: Rainbow is poor default

• problems
  – perceptually unordered
  – perceptually nonlinear

• benefits
  – fine-grained structure visible and nameable

• alternatives
  – large-scale structure: fewer hues
  – fine structure: multiple hues with monotonically increasing luminance [eg viridis R/python]
  – segmented rainbows for binned or categorical
Colormaps

- Categorical
- Ordered
  - Sequential
  - Diverging

Binary

Categorical

Diverging

Colormaps

- Categorical
- Ordered → Sequential
- Diverging
- Bivariate

Colormaps

- Categorical

- Ordered
  - Sequential
  - Diverging

- Bivariate

use with care!

Colormaps

- Categorical
- Ordered
  - Sequential
  - Diverging
- Bivariate

- color channel interactions
  - size heavily affects salience
    - small regions need high saturation
    - large need low saturation
  - saturation & luminance: 3-4 bins max
    - also not separable from transparency

Map other channels

• size
  – length accurate, 2D area ok, 3D volume poor

• angle
  – nonlinear accuracy
    • horizontal, vertical, exact diagonal

• shape
  – complex combination of lower-level primitives
    – many bins

• motion
  – highly separable against static
    • binary: great for highlighting
  – use with care to avoid irritation
Angle

Sequential ordered line mark or arrow glyph

Diverging ordered arrow glyph

Cyclic ordered arrow glyph
Next Time

• to read
  – VAD Ch. 8: Arrange Spatial Data
  – VAD Ch. 9: Arrange Networks
    • [paper type: design study]
    • [paper type: technique]

• to prepare
  – project pitches (3 min each)
Pitches

• next time (Oct 8) everybody must do a 3-min project pitch
  – slides required by 1pm in PDF format
    • submit to Canvas as "Pitch Slides" Assignment
  – if you have already made decision about teaming up
    • tell me in advance so you're back to back, coordinate so more time for detail

• goals
  – help form teams
  – give everybody (me, fellow students) situational awareness of your project ideas
    • even if not on same team, good to know who's doing similar things
      – both topic & methods
  – deadline for coming up with some concrete project idea
Pitch Hints

• think of it like an "elevator pitch"
  – explain big idea
  – convince us that it's cool/worthwhile
  – give us a sense of how fleshed out it is
    • what you've figured out
    • what's TBD

• practice in advance!
  – 3 min is both slow and fast

• I encourage you to meet with me in advance to talk through your ideas
  – 2 of you already have, and have already achieved "project signoff"
  – today's office hours is a great time for that (right after class!)
  – or make specific appointment
Projects (Reminder)

• groups of 2, 3, or 4
  – amount of work commensurate with group size
  – permission for solo project granted in exceptional circumstances, by petition

• stages
  – milestones along the way, mix of written & in-class
    • pitches (data/task), proposals, peer project reviews
    • formative feedback
  – final versions
    • final presentations
    • final reports
    • summative written feedback for both
Projects (Reminder)

• programming
  – common case (I will only consider supervising students who do these)
  – four types
    • problem-driven design studies (target specific task/data)
    • technique-driven (explore design choice space for encoding or interaction idiom)
    • algorithm implementation (as described in previous paper)
    • interactive explainer (like distill articles)

• analysis
  – use existing tools on dataset
  – detailed domain survey
  – particularly suitable for non-CS students

• survey
  – very detailed domain survey
  – particularly suitable for non-CS students
Projects: Design studies (Reminder)

• BYOD (Bring Your Own Data)
  – you (or your teammates) have your own data to analyze
    • thesis/research topic
    • personal interest
    • dovetail with another course (sometimes works, but timing may be tricky)

• FDOI (Find Data Of Interest)
  – many existing datasets, see resource page to get started
    • http://www.cs.ubc.ca/group/infovis/resources.shtml
  – can be tricky to determine reasonable task
More info

• showcase project examples
  http://www.cs.ubc.ca/~tmm/courses/547-17F/projectdesc.html#examp

• resources (detailed list from 2015)
  http://www.cs.ubc.ca/group/infovis/resources.shtml
  – inspiration
  – **data repositories**
    – data wrangling & EDA
    – visualization design
    – sharing your work

• tools directory (updated regularly)
  https://www.visualisingdata.com/resources/