Visualization Analysis & Design

Tamara Munzner
Department of Computer Science
University of British Columbia

Bedford Lab, Hutch Cancer Research Centre
July 28 2017, Seattle WA

www.cs.ubc.ca/~tmm/talks.html#vad17bedford

@tamaramunzner
Visualization (vis) defined & motivated

Computer-based visualization systems provide visual representations of datasets designed to help people carry out tasks more effectively.

Visualization is suitable when there is a need to augment human capabilities rather than replace people with computational decision-making methods.

- human in the loop needs the details
  - doesn't know exactly what questions to ask in advance
  - longterm exploratory analysis
  - presentation of known results
  - stepping stone towards automation: refining, trustbuilding
- external representation: perception vs cognition
- intended task, measurable definitions of effectiveness

more at:
Visualization Analysis and Design, Chapter 1.
Why analyze?

• imposes a structure on huge design space
  – scaffold to help you think systematically about choices
  – analyzing existing as stepping stone to designing new

SpaceTree

TreeJuxtaposer


Analysis framework: Four levels, three questions

- **domain** situation
  - who are the target users?
- **abstraction**
  - translate from specifics of domain to vocabulary of vis
- **what** is shown? **data abstraction**
  - often don’t just draw what you’re given: transform to new form
- **why** is the user looking at it? **task abstraction**
- **idiom**
  - **how** is it shown?
    - visual encoding idiom: how to draw
    - interaction idiom: how to manipulate
- **algorithm**
  - efficient computation


Why is validation difficult?

- different ways to get it wrong at each level

- **Domain situation**
  You misunderstood their needs

- **Data/task abstraction**
  You’re showing them the wrong thing

- **Visual encoding/interaction idiom**
  The way you show it doesn’t work

- **Algorithm**
  Your code is too slow
Why is validation difficult?

- solution: use methods from different fields at each level

Design Study Methodology

Reflections from the Trenches and from the Stacks

http://www.cs.ubc.ca/labs/imager/tr/2012/dsm/

Design Study Methodology: Reflections from the Trenches and from the Stacks.
Design Studies: Lessons learned after 21 of them
Methodology for Problem-Driven Work

• definitions

• 9-stage framework

• 32 pitfalls
  and how to avoid them
  - some on collaboration
  - some still apply even when designer == domain expert

<table>
<thead>
<tr>
<th>Pitfall Description</th>
<th>Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>PF-1 premature advance: jumping forward over stages</td>
<td>general</td>
</tr>
<tr>
<td>PF-2 premature start: insufficient knowledge of vis.</td>
<td>learn</td>
</tr>
<tr>
<td>PF-3 premature commitment: collaboration with wrong</td>
<td>winnow</td>
</tr>
<tr>
<td>PF-4 no real data available (yet)</td>
<td>winnow</td>
</tr>
<tr>
<td>PF-5 insufficient time available from potential</td>
<td>winnow</td>
</tr>
<tr>
<td>PF-6 no need for visualization: problem can be automated</td>
<td>winnow</td>
</tr>
<tr>
<td>PF-7 researcher expertise does not match domain problem</td>
<td>winnow</td>
</tr>
<tr>
<td>PF-8 no need for research: engineering vs. research</td>
<td>winnow</td>
</tr>
<tr>
<td>PF-9 no need for change: existing tools are good enough</td>
<td>winnow</td>
</tr>
</tbody>
</table>
Datasets

- **Data Types**
  - Items
  - Attributes
  - Links
  - Positions
  - Grids

- **Data and Dataset Types**
  - Tables
    - Items
    - Attributes
  - Networks & Trees
    - Items (nodes)
    - Grids
    - Positions
  - Fields
    - Attributes
  - Geometry
    - Items
    - Positions
  - Clusters, Sets, Lists
    - Items

- **Dataset Types**
  - Tables
  - Networks
  - Fields (Continuous)
    - Grid of positions
  - Multidimensional Table
  - Trees
  - Geometry (Spatial)

Attributes

- **Attribute Types**
  - Categorical
    - +  ●  ■  ▲
  - Ordered
    - Ordinal
  - Quantitative

Ordering Direction

- Sequential
- Diverging
- Cyclic

Dataset Availability

- Static
- Dynamic
Types: Datasets and data

Dataset Types

- Tables
- Networks
- Spatial
  - Fields (Continuous)
  - Geometry (Spatial)

Attribute Types

- Categorical
- Ordered
  - Ordinal
  - Quantitative
• \{action, target\} pairs
  – *discover distribution*
  – *compare trends*
  – *locate outliers*
  – *browse topology*
Actions: Analyze, Query

- analyze
  - consume
    - discover vs present
      - aka explore vs explain
    - enjoy
      - aka casual, social
  - produce
    - annotate, record, derive
- query
  - how much data matters?
    - one, some, all
  - independent choices
    - analyze, query, (search)
Derive: Crucial Design Choice

• don’t just draw what you’re given!
  – decide what the right thing to show is
  – create it with a series of transformations from the original dataset
  – draw that

• one of the four major strategies for handling complexity

Original Data

Derived Data
Analysis example: Derive one attribute

- Strahler number
  - centrality metric for trees/networks
  - derived quantitative attribute
  - draw top 5K of 500K for good skeleton


---

### Task 1

**What?**
- In Tree
- Quantitative attribute on nodes

**Why?**
- Derive

**Out**
- Filtered Tree
- Removed unimportant parts

---

### Task 2

**What?**
- In Tree
- Quantitative attribute on nodes

**Why?**
- Summarize
- Topology

**How?**
- Reduce
- Filter

**Out**
- Filtered Tree
Targets

- All Data
  - Trends
  - Outliers
  - Features

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

- Network Data
  - Topology
    - Paths

- Spatial Data
  - Shape
### How?

#### Encode

- **Arrange**
  - Express
  - Separate
- **Order**
  - Align
- **Use**

#### Manipulate

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

#### Maniculate

- **Change**

#### Facet

- **Juxtapose**

#### Reduce

- **Filter**
- **Facet**
- **Aggregate**
- **Embed**
How to encode: Arrange space, map channels

**Encode**

- **Arrange**
  - Express
  - Order
  - Use

- **Separate**

- **Align**

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

- **Use**

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

- **Use**

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

- **Use**

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

- **Use**

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

- **Use**
Definitions: Marks and channels

- **marks**
  - geometric primitives

- **channels**
  - control appearance of marks
Encoding visually with marks and channels

• analyze idiom structure
  – as combination of marks and channels

1: vertical position
mark: line

2: vertical position
   horizontal position
mark: point

3: vertical position
   horizontal position
   color hue
mark: point

4: vertical position
   horizontal position
   color hue
   size (area)
mark: point
Channels

Position on common scale
Position on unaligned scale
Length (1D size)
Tilt/angle
Area (2D size)
Depth (3D position)
Color luminance
Color saturation
Curvature
Volume (3D size)

Spatial region
Color hue
Motion
Shape

Same

Magnitude Channels: Ordered Attributes
Identity Channels: Categorical Attributes
Spatial region
Color hue
Motion
Shape

Length (1D size)
Tilt/angle
Area (2D size)
Depth (3D position)
Color luminance
Color saturation
Curvature
Volume (3D size)
### Channels: Matching Types

#### Magnitude Channels: Ordered Attributes
- Position on common scale
- Position on unaligned scale
- Length (1D size)
- Tilt/angle
- Area (2D size)
- Depth (3D position)
- Color luminance
- Color saturation
- Curvature
- Volume (3D size)

#### Identity Channels: Categorical Attributes
- Spatial region
- Color hue
- Motion
- Shape

- **expressiveness principle**
  - match channel and data characteristics
Channels: Rankings

**Magnitude Channels: Ordered Attributes**
- Position on common scale
- Position on unaligned scale
- Length (1D size)
- Tilt/angle
- Area (2D size)
- Depth (3D position)
- Color luminance
- Color saturation
- Curvature
- Volume (3D size)

**Identity Channels: Categorical Attributes**
- Spatial region
- Color hue
- Motion
- Shape

- **expressiveness principle**
  - match channel and data characteristics
- **effectiveness principle**
  - encode most important attributes with highest ranked channels
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**
- **Juxtapose**

Facet

- **Select**
- **Partition**
- **Superimpose**

Reduce

- **Filter**
- **Aggregate**
- **Embed**

How?
Challenges of Color

• what is wrong with this picture?

@WTFViz
“visualizations that make no sense”

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
    • saturation
  – categorical can show identity
    • hue

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

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

• intrinsic perceptual ordering
Spectral sensitivity

![Graph showing spectral sensitivity with wavelength in nm on the x-axis and relative sensitivity on the y-axis. The visible spectrum is marked by a color gradient from UV to IR.](image-url)
Opponent color and color deficiency

- perceptual processing before optic nerve
  - one achromatic luminance channel L
  - edge detection through luminance contrast
  - two chroma channels, R-G and Y-B axis
- “color blind” if one axis has degraded acuity
  - 8% of men are red/green color deficient
  - blue/yellow is rare

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

Deuteranope simulation

- Change the shape
- Vary luminance

Color deficiency: Reduces color to 2 dimensions

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

- Alternatives? This afternoon!

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

Colormaps

- Categorical
- Ordered
  - Sequential
- Bivariate

- Diverging

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

### How?

<table>
<thead>
<tr>
<th>Encode</th>
<th>Manipulate</th>
<th>Facet</th>
<th>Reduce</th>
</tr>
</thead>
<tbody>
<tr>
<td>➡️ <strong>Arrange</strong> ➡️ Express ➡️ Separate</td>
<td>➡️ <strong>Manipulate</strong> ➡️ Change</td>
<td>➡️ <strong>Facet</strong> ➡️ Juxtapose</td>
<td>➡️ <strong>Reduce</strong> ➡️ Filter</td>
</tr>
<tr>
<td>➡️ Order ➡️ Align</td>
<td>➡️ <strong>Manipulate</strong> ➡️ Select</td>
<td>➡️ <strong>Facet</strong> ➡️ Partition</td>
<td>➡️ <strong>Reduce</strong> ➡️ Aggregate</td>
</tr>
<tr>
<td>➡️ Use</td>
<td>➡️ <strong>Manipulate</strong> ➡️ Navigate</td>
<td>➡️ <strong>Facet</strong> ➡️ Superimpose</td>
<td>➡️ <strong>Reduce</strong> ➡️ Embed</td>
</tr>
</tbody>
</table>

#### Map
- **Arrange**
  - Express ➡️ Separate
  - Use ➡️ Express
- **Order**
  - Align ➡️ Separate
- **Use** ➡️ Express

#### Change
- **Color**
  - Hue ➡️ Saturation ➡️ Luminance
- **Size, Angle, Curvature, ...**

#### Select
- **Shape**
  - □ □ □ □
- **Motion**
  - Direction, Rate, Frequency, ...

#### Change
- **Map**
  - from **categorical** and **ordered** attributes
  - **Color**
    - Hue ➡️ Saturation ➡️ Luminance
  - **Size, Angle, Curvature, ...**

#### Select
- **Shape**
  - □ □ □ □
- **Motion**
  - Direction, Rate, Frequency, ...

#### Filter
- **Map**
  - **Color**
    - Hue ➡️ Saturation ➡️ Luminance

#### Aggregate
- **Select**
  - **Shape**
    - □ □ □ □
  - **Motion**
    - Direction, Rate, Frequency, ...
How to handle complexity: 3 more strategies

• change view over time
• facet across multiple views
• reduce items/attributes within single view
• derive new data to show within view
How to handle complexity: 3 more strategies

• **Manipulate**
  - **Change**
    - ![Change Illustration]
  - **Select**
    - ![Select Illustration]
  - **Navigate**
    - ![Navigate Illustration]

• **Facet**
  - **Juxtapose**
    - ![Juxtapose Illustration]
  - **Partition**
    - ![Partition Illustration]
  - **Superimpose**
    - ![Superimpose Illustration]

• **Reduce**
  - **Filter**
    - ![Filter Illustration]
  - **Aggregate**
    - ![Aggregate Illustration]
  - **Embed**
    - ![Embed Illustration]

• **Derive**

- **change over time**
  - most obvious & flexible of the 4 strategies
How to handle complexity: 3 more strategies

<table>
<thead>
<tr>
<th>Manipulate</th>
<th>Facet</th>
<th>Reduce</th>
</tr>
</thead>
<tbody>
<tr>
<td>Change</td>
<td>Juxtapose</td>
<td>Filter</td>
</tr>
<tr>
<td>Select</td>
<td>Partition</td>
<td>Aggregate</td>
</tr>
<tr>
<td>Navigate</td>
<td>Superimpose</td>
<td>Embed</td>
</tr>
</tbody>
</table>

- **Derive**

- facet data across multiple views
Idiom: **Linked highlighting**

- see how regions contiguous in one view are distributed within another
  - powerful and pervasive interaction idiom

- encoding: different
- data: all shared

**Idiom: bird’s-eye maps**

- encoding: same
- data: subset shared
- navigation: shared
  - bidirectional linking

- differences
  - viewpoint
  - (size)

- overview-detail

**System: Google Maps**

Idiom: Small multiples

- encoding: same
- data: none shared
  – different attributes for node colors
  – (same network layout)
- navigation: shared
### Coordinate views: Design choice interaction

<table>
<thead>
<tr>
<th>Encoding</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>All</td>
</tr>
<tr>
<td>Same</td>
<td>Redundant</td>
</tr>
<tr>
<td>Different</td>
<td>Multiform</td>
</tr>
</tbody>
</table>

• **why juxtapose views?**
  – benefits: eyes vs memory
  • lower cognitive load to move eyes between 2 views than remembering previous state with single changing view
  – costs: display area, 2 views side by side each have only half the area of one view
Idiom: Animation (change over time)

- weaknesses
  - widespread changes
  - disparate frames
- strengths
  - choreographed storytelling
  - localized differences between contiguous frames
  - animated transitions between states
How to handle complexity: 3 more strategies

Manipulate

- Change
- Select
- Navigate

Facet

- Juxtapose
- Partition
- Superimpose

Reduce

- Filter
- Aggregate
- Embed

- Derive

• reduce what is shown within single view
Reduce items and attributes

- **reduce/increase:** inverses
- **filter**
  - pro: straightforward and intuitive
    - to understand and compute
  - con: out of sight, out of mind
- **aggregation**
  - pro: inform about whole set
  - con: difficult to avoid losing signal
- **not mutually exclusive**
  - combine filter, aggregate
  - combine reduce, facet, change, derive
Idiom: boxplot

• static item aggregation
• task: find distribution
• data: table
• derived data
  – 5 quant attribs
    • median: central line
    • lower and upper quartile: boxes
    • lower upper fences: whiskers
      – values beyond which items are outliers
  – outliers beyond fence cutoffs explicitly shown

[40 years of boxplots. Wickham and Stryjewski. 2012. had.co.nz]
Idiom: **Dimensionality reduction for documents**

- attribute aggregation
  - derive low-dimensional target space from high-dimensional measured space

---

**Task 1**

- **What?**
  - In High-dimensional data
  - Out 2D data
- **Why?**
  - Produce
  - Derive

---

**Task 2**

- **What?**
  - In 2D data
  - Out Scatterplot
  - Out Clusters & points
- **Why?**
  - Discover
  - Explore
  - Identify
- **How?**
  - Encode
  - Navigate
  - Select

---

**Task 3**

- **What?**
  - In Scatterplot
  - In Clusters & points
  - Out Labels for clusters
- **Why?**
  - Produce
  - Annotate
A quick taste of other work!

- **technique-driven work**
- **problem-driven work**
- **theoretical foundations**
- **evaluation**
Problem-driven: Genomics

Aaron Barsky (Microbio)
Jenn Gardy (Agilent)
Robert Kincaid (Agilent)
Miriah Meyer (Harvard)
Hanspeter Pfister (Harvard)

Cerebral
MizBee
MulteeSum, Pathline
Problem-driven: Genomics, fisheries

Joel Ferstay (BC Cancer)

Cydney Nielsen (BC Cancer)

Variant View

Maryam Booshehrian (SFU)

Torsten Moeller (SFU)

Vismon
Problem-driven: Tech industry

SessionViewer: web log analysis

Heidi Lam
(Discus)

LiveRAC: systems time-series

Peter McLachlan
(IBM Research)

Stephen North
(AT&T Research)

Diane Tang
(Google)
Problem-driven: Journalism

Matt Brehmer       Stephen Ingram       Jonathan Stray
(Assoc Press)

Overview
Technique-driven: Graph drawing

TreeJuxtaposer

James Slack

Kristian Hildebrand

Daniel Archambault

David Auber
(Bordeaux)

TopoLayout
SPF
Grouse
GrouseFlocks
TugGraph
Technique-driven: Dimensionality reduction

Stephen Ingram

Glimmer

DimStiller

Glint

QSNE
Evaluation: Dimensionality reduction

Points vs landscapes for dimensionally reduced data

Guidance on DR & scatterplot choices

Taxonomy of cluster separation factors
Curation & Presentation: Timelines

**TimeLineCurator**
https://vimeo.com/123246662

**Timelines Revisited**
timelinesrevisited.github.io/

Johanna Fulda  
(Sud. Zeitung)

Matt Brehmer

Bongshin Lee  
(Microsoft)

Benjamin Bach  
(Microsoft)

Nathalie Henry-Riche  
(Microsoft)
Theoretical foundations

• Visual Encoding Pitfalls
  - Unjustified Visual Encoding
  - Hammer In Search Of Nail
  - 2D Good, 3D Better
  - Color Cacophony
  - Rainbows Just Like In The Sky

• Strategy Pitfalls
  - What I Did Over My Summer
  - Least Publishable Unit
  - Dense As Plutonium
  - Bad Slice and Dice

Papers Process & Pitfalls

Design Study Methodology

Michael Sedlmair
Miriah Meyer

Matt Brehmer

Visualization Analysis & Design
Geometry Center 1990-1995

Geomview

Charlie Gunn  Stuart Levy  Mark Phillips  Delle Maxwell

The Shape of Space  Outside In
More Information

• this talk
  www.cs.ubc.ca/~tmm/talks.html#vad17bedford

• book page (including tutorial lecture slides)
  http://www.cs.ubc.ca/~tmm/vadbook
  – 20% promo code for book+ebook combo: HVN17

  – illustrations: Eamonn Maguire

• papers, videos, software, talks, courses
  http://www.cs.ubc.ca/group/infovis
  http://www.cs.ubc.ca/~tmm

Visualization Analysis and Design.