Lectures 1&2: Manipulate & Interact

Tamara Munzner
Department of Computer Science
University of British Columbia

DSCI 532, Data Visualization 2
Week 1, Jan 2 / Jan 4 2018

www.cs.ubc.ca/~tmm/courses/mds-viz2-17

@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 & no trusted automatic solution exists
  - doesn't know exactly what questions to ask in advance
  - exploratory data analysis
    - speed up through human-in-the-loop visual data analysis
  - present known results to others
  - stepping stone towards automation
    - before model creation to provide understanding
    - during algorithm creation to refine, debug, set parameters
    - before or during deployment to build trust and monitor
Why use an external representation?

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

• external representation: replace cognition with perception

Why represent all the data?

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

- summaries lose information, details matter
  - confirm expected and find unexpected patterns
  - assess validity of statistical model

Anscombe’s Quartet

Identical statistics

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>x mean</td>
<td>9</td>
</tr>
<tr>
<td>x variance</td>
<td>10</td>
</tr>
<tr>
<td>y mean</td>
<td>7.5</td>
</tr>
<tr>
<td>y variance</td>
<td>3.75</td>
</tr>
<tr>
<td>x/y correlation</td>
<td>0.816</td>
</tr>
</tbody>
</table>

Same Stats, Different Graphs

https://www.youtube.com/watch?v=DBJyPLEmhJc
Why focus on tasks and effectiveness?

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

- Effectiveness requires match between data/task and representation
  - Set of representations is huge
  - Many are ineffective mismatch for specific data/task combo
  - Increases chance of finding good solutions if you understand full space of possibilities

- What counts as effective?
  - Novel: enable entirely new kinds of analysis
  - Faster: speed up existing workflows

- How to validate effectiveness
  - Many methods, must pick appropriate one for your context
What resource limitations are we faced with?

Vis designers must take into account three very different kinds of resource limitations: those of computers, of humans, and of displays.

• computational limits
  – processing time
  – system memory

• human limits
  – human attention and memory

• display limits
  – pixels are precious resource, the most constrained resource
  – information density: ratio of space used to encode info vs unused whitespace
    • tradeoff between clutter and wasting space, find sweet spot between dense and sparse
Nested model: Four levels of vis design

- **domain situation**
  - who are the target users?

- **abstraction**
  - translate from specifics of domain to vocabulary of vis
    - **what** is shown? data abstraction
    - **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

- **Domain situation**
  Observe target users using existing tools

- **Data/task abstraction**

- **Visual encoding/interaction idiom**
  Justify design with respect to alternatives

- **Algorithm**
  Measure system time/memory
  Analyze computational complexity

- Analyze results qualitatively
  Measure human time with lab experiment (lab study)

- **Anthropology/ethnography**
  Observe target users after deployment (field study)
  Measure adoption

- **Computer science**

- **Design**

- **Cognitive psychology**

- **Solution**
  Use methods from different fields at each level

---

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

**Dataset Types**
- Tables
- Networks & Trees
- Fields
- Geometry
- Clusters, Sets, Lists

**Attributes**
- Categorical
- Ordered
- Ordinal
- Quantitative

**Ordering Direction**
- Sequential
- Diverging
- Cyclic

**Dataset Availability**
- Static
- Dynamic
Types: Datasets and data

**Dataset Types**

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

- **Networks**
  - Node (item)
  - Link

**Attribute Types**

- **Categorical**
  - Categories

- **Ordered**
  - Ordinal
  - Quantitative

- **Ordering Direction**
  - Sequential
  - Diverging
  - Cyclic

**Spatial**

- Fields (Continuous)
- Geometry (Spatial)
- Grid of positions
- Position

**Spatial Networks**

- Networks
- Link
- Node (item)

**Spatial Tables**

- Attributes (columns)
- Items (rows)
- Cell containing value
• \{\text{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

• 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

\[
\text{trade balance} = \text{exports} - \text{imports}
\]

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
- Out Quantitative attribute on nodes

Why?
- Derive

Task 2

What?
- In Tree
- In Quantitative attribute on nodes
- Out Filtered Tree

Why?
- Summarize
- Topology
- Reduce
- Filter

How?
- Out
  - Quantitative attribute on nodes
  - Filtered Tree
  - Removed unimportant parts
Why: 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**

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?
How to handle complexity: 1 previous strategy + 3 more

Derive

• derive new data to show within view
• change view over time
• facet across multiple views
• reduce items/attributes within single view

Manipulate

- Change

Facet

- Juxtapose

Reduce

- Filter

Select

- Partition

Navigate

- Superimpose

Partition

- Aggregate

Superimpose

- Embed

Derive

- Filter
How?

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

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

**Manipulate**
- **Change**
- **Select**
- **Navigate**

**Facet**
- **Juxtapose**
- **Partition**

**Reduce**
- **Filter**
- **Aggregate**
- **Superimpose**
- **Embed**

---

**What?**

**Why?**

**How?**
Further reading

  – Chap 1: What’s Vis and Why Do It?
  – Chap 2: What: Data Abstraction
  – Chap 3: Why: Task Abstraction
  – Chap 4: Analysis: Four Levels for Validation


Manipulate / Interact
Manipulate

➡️ Change over Time

➡️ Select

➡️ Navigate

➡️ Item Reduction

➡️ Attribute Reduction

➡️ Zoom
  Geometric or Semantic

➡️ Pan/Translate

➡️ Constrained

➡️ Slice

➡️ Cut

➡️ Project
Change over time

• change any of the other choices
  – encoding itself
  – parameters
  – arrange: rearrange, reorder
  – aggregation level, what is filtered...

  – interaction entails change
Idiom: **Re-encode**

**System: Tableau**

made using Tableau, [http://tableausoftware.com](http://tableausoftware.com)
Idiom: Change parameters

- widgets and controls
  - sliders, buttons, radio buttons, checkboxes, dropdowns/comboboxes
- pros
  - clear affordances, self-documenting (with labels)
- cons
  - uses screen space
- design choices
  - separated vs interleaved
    - controls & canvas

slide inspired by: Alexander Lex, Utah

[Growth of a Nation](http://laurenwood.github.io/)
Idiom: **Change order/arrangement**

- **what:** simple table
- **how:** data-driven reordering
- **why:** find extreme values, trends

[Sortable Bar Chart](https://bl.ocks.org/mbostock/3885705)
Idiom: **Reorder**

- what: table with many attributes
- how: data-driven reordering by selecting column
- why: find correlations between attributes

[System: **DataStripes**](http://carlmanaster.github.io/datastripes/)
**Idiom:** Change alignment

- stacked bars
  - easy to compare
    - first segment
    - total bar
- align to different segment
  - supports flexible comparison

**System:** LineUp

Shiny example

• APGI genome browser
  – tooling: R/Shiny
  – interactivity
    • tooltip detail on demand on hover
    • expand/contract chromosomes
    • expand/contract control panes

https://gallery.shinyapps.io/genome_browser/
Idiom: Animated transitions

- smooth interpolation from one state to another
  - alternative to jump cuts, supports item tracking
  - best case for animation
  - staging to reduce cognitive load
- example: animated transitions in statistical data graphics

video: vimeo.com/19278444

Idiom: **Animated transitions - visual encoding change**

- smooth transition from one state to another
  - alternative to jump cuts, supports item tracking
  - best case for animation
  - staging to reduce cognitive load

[Stacked to Grouped Bars](http://bl.ocks.org/mbostock/3943967)
Idiom: Animated transition - tree detail

- animated transition
  - network drilldown/rollup

[Collapsible Tree](https://bl.ocks.org/mbostock/4339083)
Idiom: Animated transition - bar detail

• example: hierarchical bar chart
  – add detail during transition to new level of detail

[Hierarchical Bar Chart](https://bl.ocks.org/mbostock/1283663)
Interaction technology

• what do you design for?
  – mouse & keyboard on desktop?
    • large screens, hover, multiple clicks
  – touch interaction on mobile?
    • small screens, no hover, just tap
  – gestures from video / sensors?
    • ergonomic reality vs movie bombast
  – eye tracking?

*slide inspired by: Alexander Lex, Utah*
Selection

• selection: basic operation for most interaction

• design choices
  – how many selection types?
    • interaction modalities
      • click/tap (heavyweight) vs hover (lightweight but not available on most touchscreens)
      • multiple click types (shift-click, option-click, …)
      • proximity beyond click/hover (touching vs nearby vs distant)
    • application semantics
      – adding to selection set vs replacing selection
      – can selection be null?
        – ex: toggle so nothing selected if click on background
      – primary vs secondary (ex: source/target nodes in network)
      – group membership (add/delete items, name group, …)
Highlighting

• highlight: change visual encoding for selection targets
  – visual feedback closely tied to but separable from selection (interaction)

• design choices: typical visual channels
  – change item color
    • but hides existing color coding
  – add outline mark
  – change size (ex: increase outline mark linewidth)
  – change shape (ex: from solid to dashed line for link mark)

• unusual channels: motion
  – motion: usually avoid for single view
    • with multiple views, could justify to draw attention to other views
Tooltips

• popup information for selection
  – hover or click
  – can provide useful additional detail on demand
  – beware: does not support overview!
    • always consider if there’s a way to visually encode directly to provide overview
    • “If you make a rollover or tooltip, assume nobody will see it. If it's important, make it explicit.”
      – Gregor Aisch, NYTimes
Rule of thumb: **Responsiveness is required**

- **visual feedback: three rough categories**
  - 0.1 seconds: perceptual processing
    - subsecond response for mouseover highlighting - ballistic motion
  - 1 second: immediate response
    - fast response after mouseclick, button press - Fitts’ Law limits on motor control
  - 10 seconds: brief tasks
    - bounded response after dialog box - mental model of heavyweight operation (file load)

- **scalability considerations**
  - highlight selection without complete redraw of view (graphics frontbuffer)
  - show hourglass for multi-second operations (check for cancel/undo)
  - show progress bar for long operations (process in background thread)
  - rendering speed when item count is large (guaranteed frame rate)
Manipulate

- Change over Time
- Navigate
- Item Reduction
  - Zoom
    - Geometric or Semantic
  - Pan/Translate
  - Constrained
- Attribute Reduction
  - Slice
  - Cut
  - Project

Select
Navigate: Changing viewpoint/visibility

• change viewpoint
  – changes which items are visible within view
• camera metaphor
  – pan/translate/scroll
    • move up/down/sideways
Idiom: **Scrollytelling**

- how: navigate page by scrolling (panning down)
- pros:
  - familiar & intuitive, from standard web browsing
  - linear (only up & down) vs possible overload of click-based interface choices
- cons:
  - full-screen mode may lack affordances
  - scrolljacking, no direct access
  - unexpected behaviour
  - continuous control for discrete steps

https://eagereyes.org/blog/2016/the-scrollytelling-scourge
[How to Scroll, Bostock](https://bost.ocks.org/mike/scroll/)

*slide inspired by: Alexander Lex, Utah*
Scrolllytelling examples

[Graphs and charts showing data trends]


*slide inspired by: Alexander Lex, Utah*
Navigate: Changing viewpoint/visibility

• change viewpoint
  – changes which items are visible within view

• camera metaphor
  – pan/translate/scroll
    • move up/down/sideways
  – rotate/spin
    • typically in 3D
  – zoom in/out
    • enlarge/shrink world == move camera closer/further
    • geometric zoom: standard, like moving physical object
Navigate: Unconstrained vs constrained

• unconstrained navigation
  – easy to implement for designer
  – hard to control for user
    • easy to overshoot/undershoot

• constrained navigation
  – typically uses animated transitions
  – trajectory automatically computed based on selection
    • just click; selection ends up framed nicely in final viewport
Idiom: **Animated transition + constrained navigation**

- example: geographic map
  - simple zoom, only viewport changes, shapes preserved

[Zoom to Bounding Box](https://bl.ocks.org/mbostock/4699541)
Navigate: Reducing attributes

• continuation of camera metaphor
  – slice
    • show only items matching specific value for given attribute: slicing plane
    • axis aligned, or arbitrary alignment
  – cut
    • show only items on far slide of plane from camera
  – project
    • change mathematics of image creation
      – orthographic (eliminate 3rd dimension)
      – perspective (foreshortening captures limited 3D information)

Navigate: Cartographic projections

• project from 2D sphere surface to 2D plane
  – can only fully preserve 2 out of 3
    • angles: conformal
    • area: equal area
    • contiguity: no interruptions

https://www.jasondavies.com/maps/tissot/

https://www.win.tue.nl/~vanwijk/myriahedral/

[Every Map Projection](https://bl.ocks.org/mbostock/29cddc0006f8b98eff17e60dd08f59a7)
Interaction benefits

• interaction pros
  – major advantage of computer-based vs paper-based visualization
  – flexible, powerful, intuitive
    • exploratory data analysis: change as you go during analysis process
    • fluid task switching: different visual encodings support different tasks
  – animated transitions provide excellent support
    • empirical evidence that animated transitions help people stay oriented
Interaction limitations

• interaction has a time cost
  – sometimes minor, sometimes significant
  – degenerates to human-powered search in worst case

• remembering previous state imposes cognitive load
  – rule of thumb: eyes over memory
    • hard to compare visible item to memory of what you saw
    • ex: maintaining context/orientation when navigating
    • ex: tracking complex changes during animation

• controls may take screen real estate
  – or invisible functionality may be difficult to discover (lack of affordances)

• users may not interact as planned by designer
  – NYTimes logs show ~90% don’t interact beyond scrollytelling - Aisch, 2016
Further reading

  – Chap 11: Manipulate View


