**Idiom: glyphmaps**
- rectilinear good for linear vs nonlinear trends
- radial good for cyclic patterns

**Orientation limitations**
- rectilinear: scalability wrt axes
  - 2 axes best
  - 3 problematic
  - more in afternoon
- radial: possible
- parallel: unfamiliarity, training time
- radial perceptual limits
  - angles lower precision than lengths
  - asymmetry between angle and length
- can be explored

**Further reading**

---

**Idiom: choropleth map**
- use given spatial data
- when central task is understanding spatial relationships
- data: geographic geometry
- table with 1 quant attribute per region
- encoding
  - use given geometry for area mark boundaries
  - sequential segmented colormap (more later)

---

**Idiom: Bayesian surprise maps**
- use models of expectations to highlight surprising values
- confounds (population) and variance (sparsity)

---

**Idiom: force-directed placement**
- visual encoding
  - link connection marks, node points marks
- considerations
  - spatial position no meaning: directly encoded
  - cell free to minimize covering
  - proximaty semantics!
  - sometimes meaningful!
  - sometimes arbitrary artifact of layout algorithm
  - distance with length
  - edge width more visually salient than dots
- tasks
  - explore topology: locaten paths, clusters
- scalability
  - node/edge density E = 4N

---

**Idiom: adjacency matrix view**
- data: network
- same data same data/encoding as heatmap
- derived data: table from network
  - 1 quant attr
  - weighted edge between nodes
- visual encoding
  - cell shows presence/absence of edge
  - R: K nodes, M edges

---

**Idiom: radial node-link tree**
- data
- encoding
  - link connection marks
  - point node marks
  - radial axis orientation
  - angular proximity sibblings
  - distance from center: depth in tree
- tasks
  - understanding topology, following paths
- scalability
  - R: K = O(K)
How to handle complexity: 1 previous strategy + 3 more

**Idiom: Re-encode**
- data: tables with many attributes
- task: compare rankings

**System: LineUp**
- stacked bars
  - easy to compare
  - first segment
  - smooth interpolation from one state to another
  - supports flexible comparison

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

**Idiom: Animated transition - tree detail**
- animated transition
  - network drilldown/rollup

Outlines: Further reading
- Visualization Analysis and Design, Munzner: AK Peters/Visualization Series, CRC Press, 2010
- Graph Drawing, 9th International Symposium, GD’01, 2001
- How to handle complexity: 1 previous strategy + 3 more
- visualization analysis & design, munzner: AK peters/visualization series, crc press, 2010
- graph drawing, 9th international symposium, GD’01, 2001
- visual analysis of large graphs: state-of-the ART and future research challenges, von luxemburger et al., computer graphics forum 30:6 (2011), 1719-1748
- simple algorithms for network-visualization a tutorial. mcguffin, tonghui science and technology (special issue on visualization and computer graphics) 17:4 (2012), 383–398
- how to handle complexity: 1 previous strategy + 3 more
Why not animation?

- disparate frames and regions: comparison difficult
  - vs contiguous frames
  - vs small region
  - vs coherent motion of group
- safe special case
  - animated transitions

Juxtapose and coordinate views

- why juxtapose views?
  - benefits: eyes vs memory
  - lower cognitive load to move eyes between 2 views than remembering previous state with a single changing view
- costs: display area, 2 views side by side each have only half the area of one view

Idiom: Linked highlighting

- see how regions consistent in one view are distributed within another
  - powerful and pervasive interaction idiom
- encoding different
  - multiform
- data all shared

Outline

- Session 1: 10-11:30am
  - Preadmon to Amold
    - Introduction
    - Color
    - Space 2D vs 3D
- Session 2: 12-1:30pm
  - Visualization Analytics & Design, In More Depth
    - Maps and Channels, Perception
    - Arrange Tables
    - Arrange Spatial Data
    - Arrange Networks
    - Navigate: Change, Select, Navigate
    - Results: Juxtapose, Partition, Superimpose
    - Reduce: Filter-Aggregate

Facet

- Juxtapose
- Partition
- Superimpose

Why not animation?

- disparate frames and regions: comparison difficult
  - vs contiguous frames
  - vs small region
  - vs coherent motion of group
- safe special case
  - animated transitions

Further reading


Idiom: Small multiples

- encoding same
- data none shared
- navigation: shared
  - bidirectional linking
- differences
  - viewpoints
  - (size)
- overview-detail

Coordinate views: Design choice interaction

- why juxtapose views?
  - benefits: eyes vs memory
- why juxtapose views?
  - benefits: eyes vs memory
- lower cognitive load to move eyes between 2 views than remembering previous state with a single changing view
- costs: display area, 2 views side by side each have only half the area of one view

Idiom: bird’s-eye maps

- encoding same
- data shared
- navigation: shared
  - bidirectional linking
- differences
  - viewpoints
  - (size)
- overview-detail

Idiom: Improvise

- investigate power of multiple views
  - pushing into an
  view count
  interaction complexity
  - how many is best?
  - open research
  - coordinate less
  - easy linkage
  - useful when linked to other knowledge

Idiom: EDV

- see how regions consistent in one view are distributed within another
  - powerful and pervasive interaction idiom
- encoding different
  - multiform
- data all shared

System: Improvise

- investigate power of multiple views
  - pushing into an
  view count
  interaction complexity
  - how many is best?
  - open research
  - coordinate less
  - easy linkage
  - useful when linked to other knowledge

What is navigation?

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent

Partitioning: Recursive subdivision

- split by neighborhood
  - than by type
  - than by time
  - years as rows
  - months as columns
  - color by price
- neighborhood patterns
  - where it’s expensive
  - where you pay more per square meter

System: Hive

- switch order of splits
  - then by type
- switch color
  - by price variation
- type patterns
  - within specific type, which neighborhoods inconsistent
Spatial aggregation
• MAUP: Modifiable Areal Unit Problem
  – gerrymandering (manipulating voting district boundaries) is only one example!
  – zone effects
• scale effects

Further reading
  – Chap 1: Reduce Items and Attributes

More Information
• this talk
  http://www.cs.ubc.ca/~tmm/talks.html#vad17can-aft
• 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/~tmm
  http://www.cs.ubc.ca/~tmm

visualizations