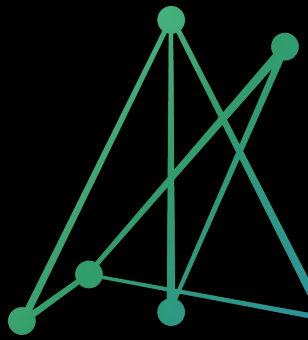


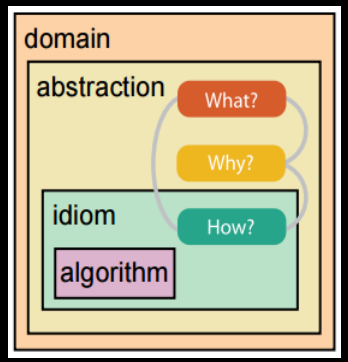
DATA SCIENCE
& ANALYTICS

University of Missouri

Data Visualization

Grammar of Graphics

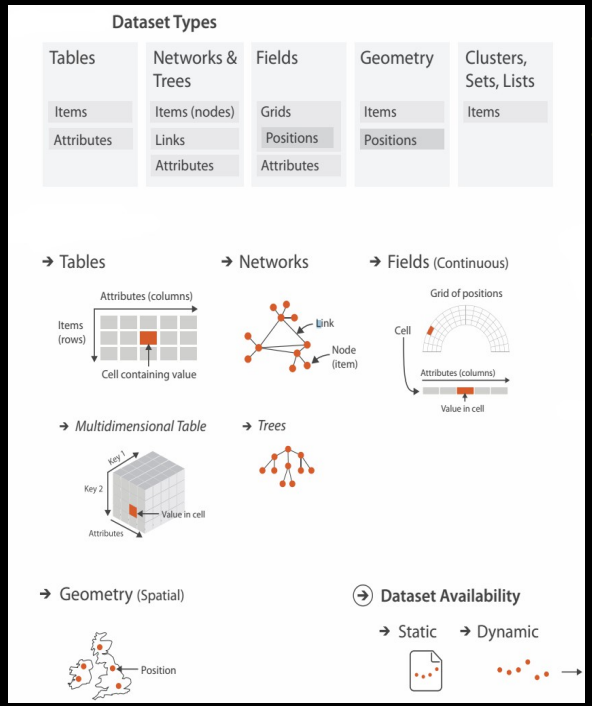
Model of Visualization Design Analysis



Tamara Munzner

- **Visual Representation: replace cognition with perception**
- Munzner's design model separates visualization into four levels:
  - Domain situation: **WHO** are the target users?
  - Abstraction: **WHAT** type of data is shown? Data abstraction
  - **WHY** is the user looking at it? Task abstraction
  - Idiom: **HOW** is the data shown ? Visual encoding, interaction
  - Algorithm: efficient computation

Model of Visualization Design Analysis



- Data Abstraction : **WHAT?**

- Dataset Types (different from data types or attributes)
  - Tables: rows and columns, familiar form.
  - Networks: specifies relationships between items
    - Trees: specific kind of network (hierarchy)
  - Fields: Each point in the dataset is a measurement from continous domain
    - Spatial Fields: structure is based on sampling spatial positions
    - Grid Types: sampling at regular intervals
  - Geometry: Shapes of items (lines,curves, regions, volumes)
  - Others: clusters, sets, lists, etc.

Model of Visualization Design Analysis

- Task Abstraction : **WHY?**
  - Consider tasks in abstract form rather than domain-specific way.
- Actions
  - Use
    - Consume, produce, enjoy, present, discover, etc.
  - Search
    - Lookup, locate, browse, explore
  - Query
    - Identify, compare, summarize
- Targets
  - All data: trends, outliers, features
  - Attributes
    - One: distribution, extremes, value
    - Many: dependency, correlation, similarity
  - Network data: topology, paths
  - Spatial data: shape

Model of Visualization Design Analysis

- Visual Encodings (idioms) : **HOW?**
  - How is the vis constructed from a set of design choices

- Encode
  - Arrange: express, separate, order, align
  - Map: color, position, size, angle,
- Manipulate
  - Change, select, navigate
- Facet
  - Juxtapose, partition, superimpose
- Reduce
  - Filter, aggregate, embed

Model of Visualization Design Analysis

- Also, four levels of validation
- Why is validation required?
  - Domain situation:
    - You misunderstood their needs
      - Observe and interview target users
  - Data/task abstraction:
    - You are showing them the wrong thing.
      - Test on users, measure utility
  - Visual encoding/interaction
    - The way you show doesn't work.
      - Test on users, measure time/error for operation
  - Algorithm
    - Your code is too slow.
      - Analyze computational complexity, measure performance

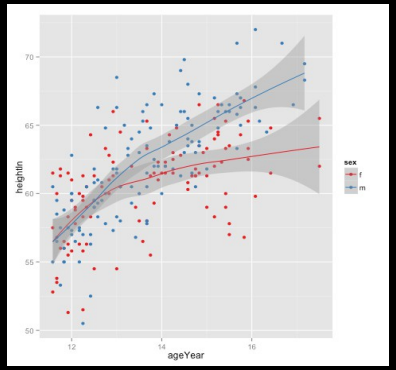
Model of Visualization Design Analysis

- Munzner's jargon:
  - Channels = visual variables (position, size, color, shape, etc.)
  - Marks = geometric primitives (points, lines), glyphs, etc.

- Marks and channels are building blocks for visual encoding design.

- The effectiveness of a channel for encoding data depends on:
  - Its type: the perceptual information it conveys (how much vs. what, where)
  - Expressiveness: e.g. ordered data should be perceived ordered by using the particular channel
  - Effectiveness: noticeability of the channel (preattentive processing!) and accuracy of the channel (Stephen's Power law!)

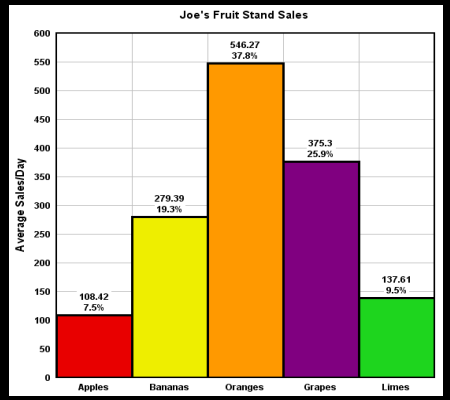
Model of Visualization Design Analysis



- An example of What-How-Why

- Idiom: Scatterplot
- What: Data
  - Data type: table, two quantitative value attributes
- How: Encoding
  - Mark: point
  - Channels:
    - express value with horizontal spatial position
    - express value with vertical spatial position
- Why: Task
  - Find trends, outliers, distribution correlation
  - Locate clusters

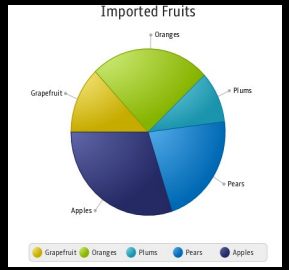
Model of Visualization Design Analysis

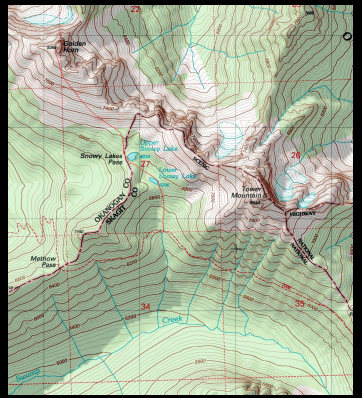


- Another example of What-How-Why

- Idiom: Bar chart
- What: Data
  - Data type: table, one quantitative value attrib, one categorical key attrib.
- How: Encoding
  - Mark: line.
  - Channels:
    - Aligned position: express value attrib.
    - Position: key attrib.
- Why: Task
  - Lookup values, find trends

Model of Visualization Design Analysis

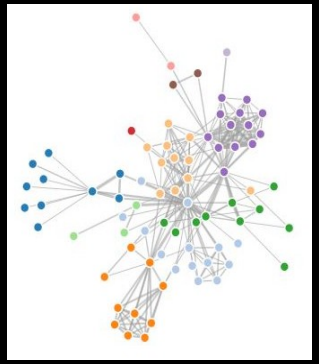




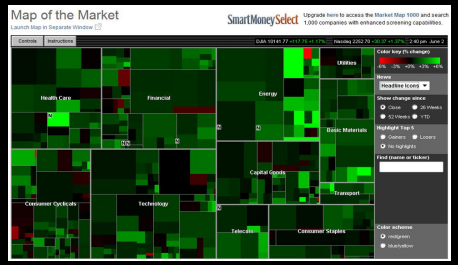
- Idiom: Pie chart
- What: Data type: table, one quant. value attrib, one categ. attrib.
- How
  - Mark: Area (wedges)
  - Channels:
    - Angle
    - Radial axis layout
- Why: Task: Part-whole relationship

- Idiom: Topographic terrain map
- What: Data type: 2D spatial field, Geometry (set of isolines)
- How: Use given geo. data of points, lines, and area marks. Use derived geometry as line marks.
- Why: Query shape

Model of Visualization Design Analysis

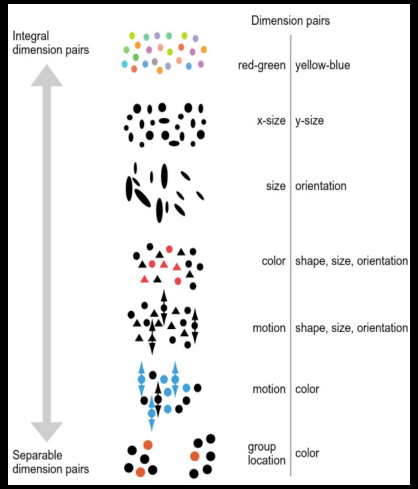


- Idiom: Force directed tree layout
- What: Data type: network
- How
  - Mark: nodes are point marks, links are connection marks.
  - Channels: position (also size, color)
- Why: Task: Explore topology, locate paths, locate clusters.



- Idiom: Treemap
- What: Data type: tree (network)
- How:
  - Mark: area, containment, rectilinear
  - Channels: position, size, color
- Why: Query attributes at leaf nodes

Glyph Design



- Glyphs are symbols to represent multivariate data
- A single glyph corresponds to one sample in the data set
- Data values are mapped to the visual properties of the glyph.

- How to design a glyph to exploit Preattention?
  - Separable channels: if they can be perceived independent of each other (size and color of a disc)
  - Integral channels: perceived holistically (rectangle:width and height)

- Separable channels are processed preattentively, integral channels require time-consuming processing.

Grammar of Graphics

An abstraction to make thinking, reasoning, and communicating graphics easier.

- Developed by Leland Wilkinson 1999/2005
- Ggplot2 implementation based on the similar concept by Hadley Wickham
- Grammar of graphics concisely describes the components of a graphic allowing us to move beyond named graphics (e.g., the “scatterplot”) and gain insight into the deep structure that underlies statistical graphics.

- What is a scatterplot?
  - Represent observations with points (geom)
  - Linear scaling of x and y axes (scales)
  - Cartesian coordinate systems (coords)
  - Converting data to physical drawing units

Good references:

- https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf
- http://docs.ggplot2.org/current/

Grammar of Graphics

- Components of layered grammar:

- Data and aesthetic mappings
- Geometric objects
- Scales
- Facet specification
- Statistical transformations
- Coordinate system

Example: Scatterplot
Geom: point
Stat: identity
Scale: linear
Coords: cartesian

Example: Histogram
Geom: bar
Stat: bin
Scale: linear
Coords: cartesian

Grammar of Graphics

Layered grammar defines the components of a plot as:

- A default data set and a set of mappings from variables to aesthetics
- One or more layers. Each layer has:
  - One geometric object
  - One statistical transformation
  - One position adjustment
  - Optionally one dataset and a set of aesthetic mappings
- One scale for each aesthetic mapping
- A coordinate system
- A facet specification

Grammar of Graphics

Layers are responsible for creating the objects. A layer has four parts:

- Data and mapping
  - Independent from the specification of the graphic. Mapping data to aesthetics creates graphic objects.
- Statistical transformation
  - Transforms the data, typically summarizes it in some manner.
- Geometric object
  - Controls the type of plot created. Every geometry has a default statistic.
- Position adjustment
  - Tweaks the position of geometric elements.

Grammar of Graphics

- Scales
  - Scales control the mapping from data to aesthetic attributes. A scale is needed for each aesthetic property used in a layer and these scales are common across layers to ensure a consistent mapping from data to aesthetics within a graphic.
- Coordinate system
  - A coordinate system maps the position of object onto the plane of the plot.
  - Coordinate systems affect all position variables simultaneously and differ from scales in that they also change the appearance of the geometric object and how the axes and grid lines are drawn.
  - The default coordinate system is Cartesian.
- Faceting
  - Faceting describes which variables should be used to split up the data, and how they should be arranged.

Grammar of Graphics

- Plot definition

```
ggplot(data, mapping) +
layer(
stat = "",
geom = "",
position = "",
geom_parms = list(),
stat_params = list(),
)
```

Grammar of Graphics

- Example

```
d <- ggplot(diamonds,
 aes(x=carat, y=price))
d + geom_point()
d + geom_point(aes(colour = carat))
d + geom_point(aes(colour = carat))
 + scale_colour_brewer()

ggplot(diamonds) +
geom_histogram(aes(x=price))
```

Data+mapping

- Data and mappings usually stay the same on a plot, so they are stored as defaults:
- ggplot(data, mapping = aes(x=x, y=y))
- aes function describes relationship, doesn't supply data

Geoms

- Geoms define the basic "shape" of the elements on the plot
- Basics: point, line, polygon, bar, text
- Composite: boxplot, pointrange
- Statistic: histogram, smooth, density

Grammar of Graphics

- Statistics
  - Separate transformation of data from its graphical representation
  - Some geoms are really statistics in disguise:
    - geom_histogram = stat_bin + geom_bar
    - geom_smooth = stat_smooth + geom_ribbon
    - geom_density = stat_density + geom_ribbon
  - Some statistics create new variables
    - stat_bin produces count and density
    - An aesthetic can be mapped to one of the new variables:
      - ggplot(diamonds, aes(x=price))
      - + geom_histogram(aes(y = ..density..))
      - + geom_histogram(aes(colour = ..count..))

Grammar of Graphics

- Scales
  - Control the mapping between data and aesthetics, and control the display of the matching guide (axis or legend)
  - ggplot automatically adds default scales as needed, but we will often need to customize.
  - Change name and range or limits
  - Position: plotting on non-linear scales, controlling breaks and space on the borders
  - Color/fill: color manipulation for discrete (hue, brewer, grey, manual) and continuous (gradient)

- Faceting
  - Drawing small multiple of subsets of data (facet_grid)

Grammar of Graphics

- Examples

Scatterplot:

```
ggplot(diamonds, aes(carat, price)) +
geom_point()
```

Scatterplot log coordinates with lm fit:

```
ggplot(diamonds, aes(carat, price)) +
geom_point() +
stat_smooth(method = lm) +
scale_x_log10() +
scale_y_log10()
```

Histogram:

```
ggplot(data = diamonds, mapping = aes(price)) +
layer(geom = "bar", stat = "bin", mapping = aes(y = ..count..))
```

Pie chart:

```
ggplot(diamonds,aes(x = "", fill=clarity)) + geom_bar(width = 1)
 + coord_polar (theta="y")
```