CS 519: Scientific Visualization

Information Visualization: Tables

Eric Shaffer

Some slides adapted Alexandru Telea, Data Visualization Principles and Practice
Scientific Visualization: “The use of computers or techniques for comprehending data or to extract knowledge from the results of simulations, computations, or measurements”  
[McCormick et al., 1987]

Information Visualization: “Visualization applied to abstract quantities and relations in order to get insight in the data”  
[Chi, 2000]
• transform raw data into insightful answers
• sequence of steps
  • data acquisition (conversion, formatting, cleaning)
  • data enrichment (transformation, resampling, filtering)
  • data mapping (produce visible shapes from data)
  • rendering (draw and interact with the shapes)
1. Confirm the known:
   • (in)validate the fit of a given model with a dataset
     • find the distribution of values over a given domain
     • find the correlation (or lack thereof) of several variables
     • answer precise (quantitative) questions
     • “show an overview of a software system’s quality”

2. Discover the unknown:
   • find support for a new model in the data
     • find which model best fits a dataset
     • find the phenomenon behind the data
     • answer more vague (qualitative) questions
     • “which anomalies does this medical image show?”

Quantitative questions:
   • “which are the strained parts of this car part?”

Qualitative questions:
   • “is this software system modular or spaghetti code?”
When is visualization useful...

1. Too much data:
   - do not have time to analyze it all (e.g. read a plot versus a table)
   - show an overview, discover which questions are relevant
   - refine search either visually or analytically

2. Qualitative / complex questions:
   - cannot capture question compactly/exactly in a query
   - question/goal is inherently qualitative: understand what is going on
   - show an overview, answer the question by seeing relevant patterns

3. Communication:
   - transfer results to different (non technical) stakeholders
   - learn about a new domain or problem
When should you not visualize

1. Queries:
   • if a question can be answered by a compact, precise query, why visualize?
   • “what is the largest value of a set”

2. Automatic decision-making:
   • if a decision can be automated, why use a human in the loop?
   • “how to optimize a numerical simulation”

Key thing to remember:
   • visualization is *mainly* a *cost vs benefits* (or value vs waste) proposal
     • cost: effort to create and interpret the images
     • benefits: problem solved by interpreting the images

✗ B. Lorensen, On the Death of Visualization, Proc. NIH/NSF Fall Workshop on Visualization Research Challenges, 2004
void ASTVisitor::traverse(ASTNode &obj) {
    ASTNodeStack stack;
    static ASTNode sentinelNode(0); // put on the bottom of the stack
    stack.push(StockItem(sentinelNode, SHOULD_IGNORE));
    stack.push(StockItem(&obj, SHOULD_VISIT)); // the node that will be visited
    while(!stack.empty()) {
        ASTNode &curNode(stack.top().cstNode);
        if (stack.top().postVisit == SHOULD_IGNORE) {
            stack.pop();
        } else if (stack.top().postVisit == SHOULD_POSTVISIT) {
            const Visit visitResult = postVisitASTNode(curNode);
            if (visitResult == VISIT_STOP) return;
            stack.pop();
            if (visitResult == VISIT_POSTPARENT)

• data is not defined on a spatial domain! $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$
• data is not always numerical $f: \mathbb{R}^n \rightarrow \mathbb{R}^m$
• data is inherently discrete!
  • no natural resampling, interpolation, reconstruction
• data structure is not a sampling artifact but a first-class property
  • relations (hierarchy, associations)
Data Attributes

- SciVis data is typically numerical ($\mathbb{R}^m$)
- InfoVis data can have more attribute types

<table>
<thead>
<tr>
<th>type</th>
<th>operations</th>
<th>examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>categorical</td>
<td>$==,\neq$</td>
<td>file type</td>
</tr>
<tr>
<td>ordinal</td>
<td>above, $&lt;$, $&gt;$</td>
<td>strings</td>
</tr>
<tr>
<td>discrete</td>
<td>above, $+, -,*,/,$</td>
<td># LOC $^{(2)}$</td>
</tr>
<tr>
<td>continuous</td>
<td>above, $*,/,$</td>
<td>metrics</td>
</tr>
</tbody>
</table>

1 relations can be seen as ordered pairs of categorical attributes
2 LOC: lines of code

- to do interpolation/resampling, we need
  - continuous attributes ($f: \mathbb{R}^n \rightarrow \mathbb{R}^m$)
  - defined on a continuous space with metric properties ($f: \mathbb{R}^n \rightarrow \mathbb{R}^m$)

- typical InfoVis attributes do not satisfy the above
  - hence some inherent difficulties
• recall the visualization pipeline:

1. Filtering:
• natural for SciVis data (sampling/reconstruction theory)
• hard for InfoVis data (problem/dataset specific)

2. Mapping:
• natural for SciVis data (draw values on given 2D/3D domain)
• explicit design decision for InfoVis data (draw values on chosen domain)
  • choose a spatial domain
  • encode some values → space attributes (layout)
  • encode other values → color/textured/shading
**Resampling:** why is this such an important issue?

- we throw away 75% of the data
- the **semantics** stays the same (a bone)

```c
#include <banking.h>

void bankCashTransfer(int amount) {
    currentBalance += amount;
}
```

- we throw away one single character
- the **semantics** becomes completely different

```c
#include <banking.h>

void bankCashTransfer(int amount) {
    currentBalance = amount;
}
```

**Note on resampling:**
- SciVis: Cauchy continuous 😊
- InfoVis: highly discontinuous 😞
Information Visualization: Filtering

However, we still can do resampling in InfoVis

- SciVis: use weighted averages

\[ f = \sum_{i=1}^{N} w_i f_i \]

- InfoVis: use semantic averages (aggregations)

\[ f = S(w_1, f_1, \ldots, w_N, f_N) \] where

- \( w_i \) = importance of item \( i \)
  - can depend on type or value of \( f_i \)
  - e.g. security-level of a function call, #bugs at a line of code, ...

- \( S \) = aggregation operation
  - task dependent
  - e.g. highlight only buggy lines in a large source code
  - e.g. group same-type relations in a graph into one relation
Semantic Resampling Example

- visualization of dynamic software activity  
  - x axis: time; y axis: memory space  
  - blocks: allocated memory fragments

- uniform spatial resampling
  - area A: appears to be empty  
  - area B: block fragmentation apparent

- outlier-emphasizing resampling
  - area A: one thin outlier block visible!  
  - area B: many thin, contiguous, blocks!
InfoVis: Mapping Data

- mapping is not ‘natural’, but reflects the problem/question to be solved

**data table**: classical view

**data table**: parallel coordinates view

**tree**: explorer view

**tree**: cushion treemap view

**source code**: classical view

**source code**: dense pixel view

**graph**: bundled view

**graph**: adjacency matrix
1. Multivariate visualization
   - tables
     - table lenses
     - parallel coordinates
   - trees / hierarchies
     - hierarchical node-link layouts
     - treemaps
   - general graphs
     - force-directed layouts
     - matrix plots
   - compound digraphs
     - bundled layouts
   - general documents
     - graph splatting
   - source code
     - dense pixel techniques
   - diagrams
     - areas of interest

2. Relational visualization
   - general graphs
     - force-directed layouts
     - matrix plots
   - compound digraphs
     - bundled layouts
   - general documents
     - graph splatting

3. Text visualization
   - general documents
     - dense pixel techniques
   - source code
     - areas of interest
   - diagrams
     - areas of interest

4. Interaction techniques
   - context and focus
   - multiple views
   - semantic zooming
one of the most ubiquitous types of InfoVis data

\[ T = \{ r_i \}_{i=1}^m, \quad r = \{ c_j \}_{j=1}^n, \quad c_j \in D_j, \quad D_j \subseteq \text{Numerical} \cup \text{Ordinal} \cup \text{Categorical} \]

functionally: \( f : \{1, \ldots, m\} \rightarrow D_1 \times \ldots \times D_n \)

compare to: \( f : \mathbb{R}^m \rightarrow \mathbb{R}^n \) (SciVis)

• columns can have different types
• rows are not uniquely ordered

Example: stock exchange data
First enhancement: map values to colors / bar lengths

- numerical data: straightforward
- ordinal data: use order as value
- categorical data: define an order (arbitrary or application-dependent)
Second enhancement: the ‘table lens’ [Rao et al, ’94]

- zoom the cells, but keep the layout
- fade out text, fade in colored / scaled bars
- → replace the entire table by a set of 1D graphs

<table>
<thead>
<tr>
<th>text opacity=1</th>
<th>text opacity ↓</th>
<th>font size=12pt</th>
<th>font size ↓</th>
<th>text not drawn</th>
<th>bar opacity↑</th>
<th>bar opacity=1</th>
<th>simplification=on</th>
</tr>
</thead>
<tbody>
<tr>
<td>min</td>
<td>zoom level</td>
<td>max</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Third enhancement: single-column sorting

- sort table on user-selected column value
- zoom out mode shows distribution and correlation of column values

Example: build cost analysis of a software system $S$

- rows = files $f$; columns = cost metrics
- build impact $(f) = \text{cost to rebuild } S \text{ if } f \text{ is touched}$
Fourth enhancement: multiple-column sorting and row grouping

- sort table on multiple user-selected column values
- emphasize same-value column ranges with cushions

- “Show stock data grouped by industry, company, and date”
- visual equivalent of SQL’s “group by” operation
- sorting: two roles
  - group same-value rows
  - show data distribution in a subset of rows
Multiple sorting: generates on-the-fly a hierarchy (tree) from the table
- one tree level per sort
- one node per group of rows

We shall see later how to visualize the tree!
Icicle Plots

• recall the tree generated by the multiple sort?
• this visualization is called an icicle plot [Kruskal and Landwehr ‘83]
  • nodes: rectangles; edges: not drawn explicitly
  • one level per vertical band (root at left, leaves at right)
  • siblings stacked vertically within a band
• compact display, no clutter

J. Kruskal, J. Landwehr, Icicle plots: Better displays for hierarchical clustering, JSTOR, 1983
Cushion design variations

- symmetric: strongly emphasize structure but occlude data bars
- asymmetric: left=transparent, right=opaque; bars are now better visible

![Symmetric cushions](image1)

![Asymmetric cushions](image2)
Take a table

- rows: car brands
- columns: car parameters (MPG, cylinders, horsepower, weight, acceleration, fabrication year)

Parallel coordinates [Inselberg and Dinsdale ‘90]

- columns: different $y$ axes
- cells: points on their corresponding axes
- rows: polylines connecting their points
- correlations: ‘bundles’ of close lines
Selection

- use mouse to select attribute ranges on axes
- highlight all rows (lines) passing through selection
- supports queries such as
  - show all cars with a low acceleration
  - find what attributes (e.g. MPG, cylinders, weight, ...) low-acceleration cars have
Parallel Coordinates

Enhancements

• permute axes (horizontally) to and swap their direction (vertically) minimize crossings
• add histograms on axes to show #lines per unit data value

swapped axes
(maximum is below)
Clutter reduction [Zhou et al. ‘08]

- define an energy function
- energy is lowest when
  - line curvature is low
  - relative ordering of curves is preserved
  - a curve is close to its neighbor curves
Parallel Coordinates

Hierarchical parallel coordinates [Fua et al. ‘99]

- reduce clutter for very large datasets ($10^6$..$10^9$ rows)
- hierarchically cluster rows $r_i$
  - create a cluster $C_i = \{r_i\}$ for each row, $S = \{ C_i \}$
  - find two most similar clusters $C_i, C_j$ using an Euclidean distance metric $d(r_i, r_j) = \sum_k (r_{ik} - r_{jk})^2$
  - build parent cluster $C = (C_i, C_j), S = S \setminus (C_i \cup C_j) \cup C$
  - repeat from step 2 until $S = \{ \text{root cluster} \}$
- select a ‘cut’ $K$ in the cluster tree at desired detail level
- visualize each cluster $C \in K$ with an opacity band which encodes cluster size and diameter

Y. Fua, M. Ward, E. Rundensteiner, Hierarchical Parallel Coordinates for Exploration of Large Datasets, IEEE InfoVis, 1999
Xmdv tool, http://davis.wpi.edu/xmdv