Data Visualization (CSCI 627/490)

Multiple Views & Filtering

Dr. David Koop
Interaction Overview

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

[Munzner (ill. Maguire), 2014]
Sorting & Slope Graphs: LineUp

[Gratzl et al., 2013]
Animation: Jump Cut vs. Animated Transitions

<table>
<thead>
<tr>
<th>♣️</th>
<th>♣️</th>
<th>♣️</th>
<th>♣️</th>
</tr>
</thead>
<tbody>
<tr>
<td>♦️</td>
<td>♦️</td>
<td>♦️</td>
<td>♦️</td>
</tr>
<tr>
<td>♠️</td>
<td>♠️</td>
<td>♠️</td>
<td>♠️</td>
</tr>
<tr>
<td>♡️</td>
<td>♡️</td>
<td>♡️</td>
<td>♡️</td>
</tr>
</tbody>
</table>

D. Koop, CSCI 627/490, Fall 2020
## Animation: Jump Cut vs. Animated Transitions

<table>
<thead>
<tr>
<th>♣️ Q</th>
<th>♣️ A</th>
<th>♣️ Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>♣️ K</td>
<td>♣️ Q</td>
<td>♣️ K</td>
</tr>
<tr>
<td>♣️ A</td>
<td>♣️ J</td>
<td>♣️ A</td>
</tr>
<tr>
<td>♣️ A</td>
<td>♣️ Q</td>
<td>♣️ J</td>
</tr>
<tr>
<td>♠️ J</td>
<td>♠️ Q</td>
<td>♠️ J</td>
</tr>
<tr>
<td>♠️ J</td>
<td>♠️ Q</td>
<td>♠️ J</td>
</tr>
<tr>
<td>♦️ A</td>
<td>♦️ Q</td>
<td>♦️ A</td>
</tr>
<tr>
<td>♦️ A</td>
<td>♦️ Q</td>
<td>♦️ A</td>
</tr>
<tr>
<td>♥️ Q</td>
<td>♥️ K</td>
<td>♥️ Q</td>
</tr>
<tr>
<td>♥️ Q</td>
<td>♥️ K</td>
<td>♥️ Q</td>
</tr>
<tr>
<td>♥️ A</td>
<td>♥️ Q</td>
<td>♥️ A</td>
</tr>
<tr>
<td>♥️ A</td>
<td>♥️ Q</td>
<td>♥️ A</td>
</tr>
</tbody>
</table>

---

D. Koop, CSCI 627/490, Fall 2020
Animation: Jump Cut vs. Animated Transitions

<table>
<thead>
<tr>
<th>♠️</th>
<th>♦️</th>
<th>♣️</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>J</td>
<td>Q</td>
<td>K</td>
</tr>
<tr>
<td>A</td>
<td>J</td>
<td>Q</td>
</tr>
<tr>
<td>Q</td>
<td>J</td>
<td>A</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>♥️</th>
<th>♠️</th>
<th>♦️</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>J</td>
<td>Q</td>
<td>K</td>
</tr>
<tr>
<td>A</td>
<td>J</td>
<td>Q</td>
</tr>
<tr>
<td>Q</td>
<td>J</td>
<td>A</td>
</tr>
</tbody>
</table>
## Animation: Jump Cut vs. Animated Transitions

<table>
<thead>
<tr>
<th>Q♣</th>
<th>A♣</th>
<th>Q♣</th>
</tr>
</thead>
<tbody>
<tr>
<td>K♦</td>
<td>Q♦</td>
<td>K♦</td>
</tr>
<tr>
<td>A♥</td>
<td>J♦</td>
<td>A♦</td>
</tr>
<tr>
<td>A♦</td>
<td>Q♠</td>
<td>J♣</td>
</tr>
<tr>
<td>Q♠</td>
<td>J♥</td>
<td>Q♥</td>
</tr>
<tr>
<td>Q♥</td>
<td>A♣</td>
<td>J♥</td>
</tr>
<tr>
<td>A♣</td>
<td>Q♠</td>
<td>J♠</td>
</tr>
<tr>
<td>K♠</td>
<td>J♣</td>
<td>K♣</td>
</tr>
<tr>
<td>K♥</td>
<td>J♠</td>
<td>K♠</td>
</tr>
<tr>
<td>A♠</td>
<td>Q♥</td>
<td>K♥</td>
</tr>
<tr>
<td>J♥</td>
<td>Q♦</td>
<td>A♦</td>
</tr>
<tr>
<td>Q♦</td>
<td>K♠</td>
<td>J♥</td>
</tr>
<tr>
<td>K♠</td>
<td>J♠</td>
<td>A♥</td>
</tr>
<tr>
<td>J♥</td>
<td>K♦</td>
<td>Q♦</td>
</tr>
<tr>
<td>J♣</td>
<td>A♥</td>
<td>K♥</td>
</tr>
<tr>
<td>J♦</td>
<td>A♦</td>
<td>K♦</td>
</tr>
</tbody>
</table>
Animated Transitions

Stacked  Grouped

[M. Bostock]
Animated Transitions

Stacked  Grouped

[M. Bostock]
Heer and Robertson Study

- User Preferences: Staged animation > animation > static transitions

- Animation improves graphical perception
- Staging is better (do axis rescaling before value changes)
- Avoid axis rescaling when possible

[Heer and Robertson, 2007]
Selection

• Selection is often used to initiate other changes
• User needs to select something to drive the next change
• What can be a selection target?
  - Items, links, attributes, (views)
• How?
  - mouse click, mouse hover, touch
  - keyboard modifiers, right/left mouse click, force
• Selection modes:
  - Single, multiple
  - Contiguous?
Highlighting

- Selection is the user action
- Feedback is important!
- How? Change selected item's visual encoding
  - Change color: want to achieve visual popout
  - Add outline mark: allows original color to be preserved
  - Change size (line width)
  - Add motion: marching ants
Highlighting

• Selection is the user action
• Feedback is important!
• How? Change selected item's visual encoding
  - Change color: want to achieve visual popout
  - Add outline mark: allows original color to be preserved
  - Change size (line width)
  - Add motion: marching ants
Interaction Latency

- The Effects of Interactive Latency on Exploratory Visual Analysis, Z. Liu and J. Heer, 2014
- Brush & link, select, pan, zoom

- 500ms added latency causes significant cost
  - decreases user activity and dataset coverage
  - reduces rate of observations, generalizations, and hypotheses
Navigation

→ Item Reduction

→ Zoom
  Geometric or Semantic

→ Pan/Translate

→ Constrained

→ Attribute Reduction

→ Slice

→ Cut

→ Project

[Munzner (ill. Maguire), 2014]
Geometric vs. Semantic Zooming

• Geometric zoom: like a camera
• Semantic zoom: visual appearance of objects can change at different scales
Project Design

- Feedback:
  - Data Manipulation?
  - Questions lead, not technique!
  - Be creative! (interaction too) https://xeno.graphics

- Work on turning your visualization ideas into designs

- Turn in:
  - Three Designs Sketches
  - One Bad Design
  - Progress on Implementation

- Due Friday, Nov. 13
Assignment 4

• Geospatial Visualizations & Treemap
  - Choose colormaps carefully
  - Add legend
• Due Nov. 2
No class Tuesday

VOTE!
Design Space of Composite Visualization

• Composite visualization views (CVVs)
  - Includes Coordinated multiple views (CMV)
  - + More!

• Design Patterns:
  - Juxtaposition: side-by-side
  - Superimposition: layers
  - Overloading: vis meshed with another
  - Nesting: vis inside a vis (recursive vis)
  - Integration: "merge" views + links

[A \otimes_{\text{op}} B = AB]
[A \otimes_{\text{int}} B = AB]
[A \otimes_{\text{nest}} B = A_B]
[A \otimes_{\text{sub}} B = AB]

[W. Javed and N. Elmqvist, 2012]
Composite Visualization Techniques

(a) Juxtaposed views.  (b) Integrated views.  (c) Superimposed views.

(d) Overloaded views.  (e) Nested views.

[W. Javed and N. Elmqvist, 2012]
Juxtaposition

[ComVis, K. Matkovic et al., 2008]
Juxtaposition

[Improvise, C. Weaver, 2004]
Juxtaposition Guidelines

• Benefits:
  - The component visualizations are independent and can be composed without interference
  - Easy to implement

• Drawbacks:
  - Implicit visual linking is not always easy to see, particularly when multiple objects are selected
  - Space is divided between the views, yielding less space for each view

• Applications: Use for heterogeneous datasets consisting of many different types of data, or for where different independent visualizations need to be combined.

[W. Javed and N. Elmqvist, 2012]
Integration

Semantic Substrates

Radial and force-directed graphs on separate visualization planes linked with visual edges.

VisLink

- Allows toggling the visibility of relational links.
- Visual links between nodes highlight citation between different cases.

ComVis

- Visual exploration of meteorology data using semantic substrates.
- User has created eight different views, each with a different visualization.
- Analyst used a single brush in the histogram view, causing all other views to update.

Improvise

- Visualization framework based on the juxtaposed views design pattern.
- Provides support for coordinated queries and interactive brushing.

Figure 5: VisLink being used for exploring a dataset of federal judicial cases on the legal issue of regulatory takings from 1978 to 2005.

Figure 4: Semantic substrates used for the exploration of a simulated ion trajectory.

Figure 2: Visual exploration of meteorology data using semantic substrates.
Integration

Figure 4: Semantic substrates used for the exploration of a simulated ion trajectory data. The substrates are designed using the node-link diagram approach, with nodes representing ion positions and links showing their movement over time.

Figure 3 shows a visual exploration of a simulated ion trajectory data using semantic substrates. The substrates are designed using the node-link diagram approach, with nodes representing ion positions and links showing their movement over time.

Figure 5 shows VisLink being used for exploring a dataset of English words based on the IS-A relation over synonym sets. VisLink creates multiple 2D planes, one for each category, and the nodes are sized proportionally to the number of data entries for the category they visualize. The individual regions are spatially non-overlapping regions that are built to hold nodes based on a user-defined semantic substrate with node-links diagram. The node-links diagram approach allows for the highlighting of one to many linking. To reduce visual clutter, the tool supports two kinds of edges: straight edges are used to show one-to-one linking, while bundled curved edges are used to show relational links only between adjacent planes.

As with semantic substrates, the VisLink relational linking is built to highlight the corresponding data items. In contrast with semantic substrates, VisLink allows users to toggle the visibility of links, or to show relational links only for selected data values. The tool supports two kinds of edges: straight edges are used to show one-to-one linking, while bundled curved edges are used to show relational links only for selected data values. The use of explicit linking in integrated views, compared to implicit linking in juxtaposed views, allows for better relational cognition.

ComVis: The user has created eight different views, each with a visualization of a dataset of court cases using semantic substrates. Figure 2 shows a visual exploration of meteorology data using ComVis. The user has created eight different views, each with a visualization of a dataset of court cases using semantic substrates. The system is highly extensible and modularized, allowing it to be adapted for virtually any type of data and visualization. To explore relational data in an interactive manner, Improvise provides support for coordinated queries, a visual abstraction language designed for relational databases. More recently, Improvise provides support for coordinated queries, a visual abstraction language designed for relational databases.

The integrated views design pattern is also based on juxtaposing (or tiling) the component visualizations (Figures 4, 5). For this reason, the visual composition for integrated views is identical to that of tiling (or tiling) the component visualizations (Figures 4, 5). The framework allows users to build integrated views, which juxtapose views, integrated views use explicit linking, normally normally.

Figure 3 shows a visual exploration of a simulated ion trajectory data using semantic substrates. The substrates are designed using the node-link diagram approach, with nodes representing ion positions and links showing their movement over time.

4.2 VisLink

VisLink [11] (Integrated Views) creates multiple 2D planes, one for each category, and the nodes are sized proportionally to the number of data entries for the category they visualize. This scheme allows users to get a quick idea about the cardinality of different categories present in the underlying dataset. Their approach is in line with the integrated view design pattern because the techniques add visual links to connect corresponding marks in the other planes. However, contrary to the implicit linking used in juxtaposed views, integrated views use explicit linking, normally.

The visual composition for integrated views is identical to that of tiling the component visualizations (Figures 4, 5). The framework allows users to build integrated views, which juxtapose views, integrated views use explicit linking, normally.

The use of explicit linking in integrated views, compared to implicit linking in juxtaposed views, allows for better relational cognition, but at the cost of added visual clutter. However, as the number of data points increases in the visualizations, the visual clutter arising from the explicit links may become a major hindrance. ComVis [24] is a multidimensional visualization system supporting the use of different visualizations while exploring the dataset. The use of explicit linking in integrated views, compared to implicit linking in juxtaposed views, allows for better relational cognition.
Integration

[Carte figurative des pertes successives de l'Armee Francaise dans la Campagne de Russie 1812-1813.](Napoleon's March to Moscow, C. J. Minard, 1869)
Integration Guidelines

• Benefits:
  - Easy to perceive one-to-one and one-to-many relations between items in components
  - Visualizations are less independent compared to juxtaposed views, but still separate

• Drawbacks:
  - Extra visual clutter added to the overall view
  - Display space is split between the views
  - Some dependencies exist between views to allow for the visual linking

• Applications: Use for heterogeneous datasets where correlation and comparisons between views is particularly important.

[W. Javed and N. Elmqvist, 2012]
Superimposition

Superimposed views overlay two or more visual spaces on top of each other (Figures 6 and 7). The resulting visualization becomes the visual combination of the component visualizations, often using transparency to enable seeing all views. Superimposed views are generally used to highlight spatial relations in the component visualizations. In other words, the spatial linking present in these views is one-to-one, i.e., all the overlay visualizations share the same underlying visual space. Line graph visualizations with several data series, where more than one graph is superimposed in a single chart (e.g., [19]), is a very commonly used example of this design pattern.

The spatial linking in the superimposed views allows for easy comparison across different datasets because the user does not have to split their attention between different parts of the visual space. Furthermore, the fact that visualizations are stacked means that they can each use the full available space in the view. However, because the composition simply adds the component visualizations together, the visual clutter may become significant, and it is also likely to cause conflicts arising from one visualization occluding another.

5.1 Mapgets
Mapgets [38] is a geographic visualization system that allows users to interactively perform map editing and querying of geographical datasets. The maps generated using Mapgets are built on an underlying presentation stack that superimposes multiple dataset layers on top of each other. The users can dynamically select the dataset to use for each layer and the total number of layers to compose. Different layers in the presentation stack allow users to independently interact with each of the associated visualization and control the layer attributes. The technique also allows the users to reorder layers in the presentation stack to achieve the desirable map result. Figure 6 shows an example of a European map generated in Mapgets. The presentation stack associated with this map consists of three layers: the bottom layer visualizes rivers, the center layer is used to depict the country borders, and the topmost layer is used to display the country labels.

5.2 GeoSpace
GeoSpace [22] allows users to interactively explore complex visual spaces using superimposed views. It permits progressively overlaying different datasets, based on the user queries, in a single view. Beyond allowing users to explore datasets through dynamic queries, GeoSpace also supports pan and zoom operations for navigation. Figure 7 shows GeoSpace system being used for exploring crime around the Cambridge, MA area. The figure shows a 2D view of the visualization, where red dots that are spatially coupled to the underlying layer show the reported crime cases in the region.

Figure 8: SPPC [45] (Overloaded Views). This tool overloads points into the region bounded by two axes in the parallel coordinate plot.

Figure 9: Links on treemaps [14] (Overloaded Views). The tool identifies a tree structure in a graph and visualizes it using a treemap.
Superimposition

Superimposed views overlay two or more visual spaces on top of each other (Figures 6 and 7). The resulting visualization becomes the visual combination of the component visualizations, often using transparency to enable seeing all views. Superimposed views are generally used to highlight spatial relations in the component visualizations. In other words, the spatial linking present in these views is one-to-one, i.e., all the overlay visualizations share the same underlying visual space. Line graph visualizations with several data series, where more than one graph is superimposed in a single chart (e.g., [19]), is a very commonly used example of this design pattern. The spatial linking in the superimposed views allows for easy comparison across different datasets because the user does not have to split their attention between different parts of the visual space. Furthermore, the fact that visualizations are stacked means that they can each use the full available space in the view. However, because the composition simply adds the component visualizations together, the visual clutter may become significant, and it is also likely to cause conflicts arising from one visualization occluding another.

5.1 Mapgets

Mapgets [38] is a geographic visualization system that allows users to interactively perform map editing and querying of geographical datasets. The maps generated using Mapgets are built on an underlying presentation stack that superimposes multiple dataset layers on top of each other. The users can dynamically select the dataset to use for each layer and the total number of layers to compose. Different layers in the presentation stack allow users to independently interact with each of the associated visualization and control the layer attributes. The technique also allows the users to reorder layers in the presentation stack to achieve the desirable map result. Figure 6 shows an example of a European map generated in Mapgets. The presentation stack associated with this map consists of three layers: the bottom layer visualizes rivers, the center layer is used to depict the country borders, and the topmost layer is used to display the country labels.

5.2 GeoSpace

GeoSpace [22] allows users to interactively explore complex visual spaces using superimposed views. It permits progressively overlaying different datasets, based on the user queries, in a single view. Beyond allowing users to explore datasets through dynamic queries, GeoSpace also supports pan and zoom operations for navigation. Figure 7 shows GeoSpace system being used for exploring crime around the Cambridge, MA area. The figure shows a 2D view of the visualization, where red dots that are spatially coupled to the underlying layer show the reported crime cases in the region.

Figure 8: SPPC [45] (Overloaded Views). This tool overloads points into the region bounded by two axes in the parallel coordinate plot.

Figure 9: Links on treemaps [14] (Overloaded Views). The tool identifies a tree structure in a graph and visualizes it using a treemap.

[GeoSpace, I. Lokuge and S. Ishizaki, 1995]
Superimposition Guidelines

• **Benefits:**
  - Allows direct comparison in the same visual space.

• **Drawbacks:**
  - May cause occlusion and high visual clutter.
  - The client visualization must share the same spatial mapping as the host visualization.

• **Applications:** In settings where comparison is common, or where the component visualization views need to be as large as possible (potentially the entire available space).

[W. Javed and N. Elmqvist, 2012]
Superimposed views overlay two or more visual spaces on top of each other (Figures 6 and 7). The resulting visualization becomes the visual combination of the component visualizations, often using transparency to enable seeing all views. Superimposed views are generally used to highlight spatial relations in the component visualizations. In other words, the spatial linking present in these views is one-to-one, i.e., all the overlay visualizations share the same underlying visual space. Line graph visualizations with several data series, where more than one graph is superimposed in a single chart (e.g., [19]), is a very commonly used example of this design pattern. The spatial linking in the superimposed views allows for easy comparison across different datasets because the user does not have to split their attention between different parts of the visual space. Furthermore, the fact that visualizations are stacked means that they can each use the full available space in the view. However, because the composition simply adds the component visualizations together, the visual clutter may become significant, and it is also likely to cause conflicts arising from one visualization occluding another.

5.1 Mapgets
Mapgets [38] is a geographic visualization system that allows users to interactively perform map editing and querying of geographical datasets. The maps generated using Mapgets are built on an underlying presentation stack that superimposes multiple dataset layers on top of each other. The users can dynamically select the dataset to use for each layer and the total number of layers to compose. Different layers in the presentation stack allow users to independently interact with each of the associated visualization and control the layer attributes. The technique also allows the users to reorder layers in the presentation stack to achieve the desirable map result. Figure 6 shows an example of a European map generated in Mapgets. The presentation stack associated with this map consists of three layers: the bottom layer visualizes rivers, the center layer is used to depict the country borders, and the topmost layer is used to display the country labels.

5.2 GeoSpace
GeoSpace [22] allows users to interactively explore complex visual spaces using superimposed views. It permits progressively overlaying different datasets, based on the user queries, in a single view. Beyond allowing users to explore datasets through dynamic queries, GeoSpace also supports pan and zoom operations for navigation. Figure 7 shows GeoSpace system being used for exploring crime around the Cambridge, MA area. The figure shows a 2D view of the visualization, where red dots that are spatially coupled to the underlying layer show the reported crime cases in the region.
Superimposition

Superimposed views overlay two or more visual spaces on top of each other (Figures 6 and 7). The resulting visualization becomes the visual combination of the component visualizations, often using transparency to enable seeing all views. Superimposed views are generally used to highlight spatial relations in the component visualizations. In other words, the spatial linking present in these views is one-to-one, i.e., all the overlay visualizations share the same underlying visual space. Line graph visualizations with several data series, where more than one graph is superimposed in a single chart (e.g., [19]), is a very commonly used example of this design pattern. The spatial linking in the superimposed views allows for easy comparison across different datasets because the user does not have to split their attention between different parts of the visual space. Furthermore, the fact that visualizations are stacked means that they can each use the full available space in the view. However, because the composition simply adds the component visualizations together, the visual clutter may become significant, and it is also likely to cause conflicts arising from one visualization occluding another.

5.1 Mapgets

Mapgets [38] is a geographic visualization system that allows users to interactively perform map editing and querying of geographical datasets. The maps generated using Mapgets are built on an underlying presentation stack that superimposes multiple dataset layers on top of each other. The users can dynamically select the dataset to use for each layer and the total number of layers to compose. Different layers in the presentation stack allow users to independently interact with each of the associated visualization and control the layer attributes. The technique also allows the users to reorder layers in the presentation stack to achieve the desirable map result. Figure 6 shows an example of a European map generated in Mapgets. The presentation stack associated with this map consists of three layers: the bottom layer visualizes rivers, the center layer is used to depict the country borders, and the topmost layer is used to display the country labels.

5.2 GeoSpace

GeoSpace [22] allows users to interactively explore complex visual spaces using superimposed views. It permits progressively overlaying different datasets, based on the user queries, in a single view. Beyond allowing users to explore datasets through dynamic queries, GeoSpace also supports pan and zoom operations for navigation. Figure 7 shows GeoSpace system being used for exploring crime around the Cambridge, MA area. The figure shows a 2D view of the visualization, where red dots that are spatially coupled to the underlying layer show the reported crime cases in the region.

Overloading

[Links on Treemaps, J.-D. Fekete et al., 2003]
Overloading Guidelines

• Benefits:
  - The client visualization does not have to share the same coordinate space as the host visualization
  - This also yields more flexibility and control over visual clutter

• Drawbacks:
  - Visual clutter is increased
  - Visual design dependencies between components are significant

• Applications: Situations where one visualization can be folded into another to yield a compact (and complex) result.

[W. Javed and N. Elmqvist, 2012]
Nesting

While previous design patterns have all operated on specific components of component visualizations, overloaded views (and also the VERLOADED.Views) are similar to superimposed views, but with some important differences. Like superimposition, the client visualization in this design pattern is overlaid on the host. However, unlike Superimposed Views, there exists no imposition, the client visualization in this design pattern is overlaid on the visual marks of the matrix to convey information about the aggregation. This visualization of the InfoVis co-authorship network.

The tool allows the user to zoom in data space, which amounts to drilling-down and rolling-up in the aggregation hierarchy. Abstract glyphs representing aggregation used in ZAME is a hierarchical aggregation of the underlying visualizations. The tool also supports controlling the visibility of various edges to reduce visual clutter, and recognizing the direction of the link. The edges are curved to highlight source and target locations. The edges through an underlying treemap and external links are visualized here, the directory structure, inherent in any website, is visualized in our terminology because the graph links are not just a separate layer on top of the treemap, but they are embedded into the visual structures themselves must be modified to combine the components. We will see examples of this below.

Figure 10: [ZAME, N. Elmqvist et al., 2008]

7.1 ZAME

The nested views pattern provides an effective way of relating visualizations across multiple dimensions due to their inherent visual clutter. Scatterplots through multidimensional scaling is efficient for visualizing multiple dimensions in a compact 2D visual representation. However, they make it hard to correlate trends in any dimension of a dataset. Combining both techniques allows for sharing their advantages.

Parallel coordinate plots do not represent all of the components. We will see examples of this below. The technique is based on converting the space between pairs of selected coordinate dimensions in a parallel coordinate plot into scatterplots on a parallel coordinates visualization (Figure 8). Yuan et al. [45] presented a system that allows overloading of 2D components. We will see examples of this below.
Nesting

One issue to discuss here is the difference between overloading and nesting. These are different design patterns because nesting requires a more careful design, while overloading does not.

Overloading requires the actual visual structures of the components, which typically takes advantage of the fact that parallel coordinate plots do not represent individual data points in the space, but are curved to highlight source and target locations. The edges are imposed views, but with some important differences. Like superposition, which specifies that the client visualizations sit on top of the host visualization, overloading is a visual layout operation to organize the views together, but the visualization system designed to explore large-scale adjacency matrix data for each cell in the matrix are nested inside the visual marks of the client visualization by nested instances of the visual marks in the host visualization. Furthermore, just like overloading, nested views compensate for their individual shortcomings. Parallel coordinates are efficient for visualizing multiple dimensions in a compact 2D visualization, called the treemap. The idea is based on the fact that it is possible to decompose a graph into a tree structure and that this space is open for being overloaded.

This design pattern characterizes compositions where one visualization, called the host, using the same spatial mapping as the client visualization, is rendered inside another, called the client, whereas overloading requires a much more integrated approach. The tool also supports controlling the visibility of various edges to reduce visual clutter, and recognizing the direction of the link. The technique is an example of overloading through overlaid edges. The overlaid edges are not straight lines, but are curved to highlight source and target locations. The edges here, the directory structure, inherent in any website, is visualized through an underlying treemap and external links are visualized through overlaid edges. The overlaid edges are not straight lines, but are curved to highlight source and target locations. The edges are imposed views, but with some important differences. Like superposition, which specifies that the client visualizations sit on top of the host visualization, overloading is a visual layout operation to organize the views together, but the visualization system designed to explore large-scale adjacency matrix data for each cell in the matrix are nested inside the visual marks of the client visualization by nested instances of the visual marks in the host visualization. Furthermore, just like overloading, nested views compensate for their individual shortcomings. Parallel coordinates are efficient for visualizing multiple dimensions in a compact 2D visualization, called the treemap. The idea is based on the fact that it is possible to decompose a graph into a tree structure and that this space is open for being overloaded.
Nesting Guidelines

• Benefits:
  - Very compact representation
  - Easy correlation

• Drawbacks:
  - Limited space for the client visualizations
  - Clutter is high
  - Visual design dependencies are high

• Applications: Situations that call for augmenting a particular visual representation with additional mapping

[W. Javed and N. Elmqvist, 2012]
Design Space

- Visualizations: the techniques or idioms used
- Spatial relation: relationship between visual structures in display space
- Data relation: visual relationship between items in different views
  - None: No relation
  - Item-item: One-to-one
  - Item-group: One-to-many
  - Item-dimension: Item in one view is a **scale** in another

[W. Javed and N. Elmqvist, 2012]
Summary

<table>
<thead>
<tr>
<th>Technique</th>
<th>Visualization A</th>
<th>Visualization B</th>
<th>Spatial Relation</th>
<th>Data Relation</th>
</tr>
</thead>
<tbody>
<tr>
<td>ComVis [24] (Figure 2)</td>
<td>any</td>
<td>any</td>
<td>juxtapose</td>
<td>none</td>
</tr>
<tr>
<td>Improvise [39] (Figure 3)</td>
<td>any</td>
<td>any</td>
<td>juxtapose</td>
<td>none</td>
</tr>
<tr>
<td>Jigsaw [36]</td>
<td>any</td>
<td>any</td>
<td>juxtapose</td>
<td>none</td>
</tr>
<tr>
<td>Snap-Together [30]</td>
<td>any</td>
<td>any</td>
<td>juxtapose</td>
<td>none</td>
</tr>
<tr>
<td>semantic substrates [34] (Figure 4)</td>
<td>node-link</td>
<td>node-link</td>
<td>juxtapose</td>
<td>item-item</td>
</tr>
<tr>
<td>VisLink [11] (Figure 5)</td>
<td>radial graph</td>
<td>node-link</td>
<td>juxtapose</td>
<td>item-item</td>
</tr>
<tr>
<td>Napoleon’s March on Moscow [37]</td>
<td>time line view</td>
<td>area visualization</td>
<td>juxtapose</td>
<td>item-item</td>
</tr>
<tr>
<td>Mapgets [38] (Figure 6)</td>
<td>map</td>
<td>text</td>
<td>superimpose</td>
<td>item-item</td>
</tr>
<tr>
<td>GeoSpace [22] (Figure 7)</td>
<td>map</td>
<td>bar graph</td>
<td>superimpose</td>
<td>item-item</td>
</tr>
<tr>
<td>3D GIS [8]</td>
<td>map</td>
<td>glyphs</td>
<td>superimpose</td>
<td>item-item</td>
</tr>
<tr>
<td>Scatter Plots in Parallel Coordinates [45] (Figure 8)</td>
<td>parallel coordinate</td>
<td>scatterplot</td>
<td>overload</td>
<td>item-dimension</td>
</tr>
<tr>
<td>Graph links on treemaps [14] (Figure 9)</td>
<td>treemap</td>
<td>node-link</td>
<td>overload</td>
<td>item-item</td>
</tr>
<tr>
<td>SparkClouds [21]</td>
<td>tag cloud</td>
<td>line graph</td>
<td>overload</td>
<td>item-item</td>
</tr>
<tr>
<td>ZAME [13] (Figure 10)</td>
<td>matrix</td>
<td>glyphs</td>
<td>nested</td>
<td>item-group</td>
</tr>
<tr>
<td>NodeTrix [17] (Figure 11)</td>
<td>node-link</td>
<td>matrix</td>
<td>nested</td>
<td>item-group</td>
</tr>
<tr>
<td>TimeMatrix [44]</td>
<td>matrix</td>
<td>glyphs</td>
<td>nested</td>
<td>item-group</td>
</tr>
<tr>
<td>GPUVis [25]</td>
<td>Scatterplot</td>
<td>glyphs</td>
<td>nested</td>
<td>item-group</td>
</tr>
</tbody>
</table>
One such example is the use of interactive hyperlinking \([6, 43]\) (or more visualizations, for example using interaction or animation. However, it is possible to envision other ways to combine two or more visualizations based on this design pattern. In the figure, datasets in the same space and using different visualizations, but also highlights the relational linking between the two datasets. This is achieved by nesting clients inside the visual marks in the host.

There is probably not a clear winner among different design patterns. While our above CVV design patterns are general in nature, they are based solely on the spatial layout of component visualizations. In contrast, designers should be able to combine any existing visualizations to generate a composite visualization view.

While our above CVV design patterns are general in nature, they are inherent limited to current designs, and more descriptive than evaluative. The selection of existing work where visual representations are combined in various ways, but also to suggest new combinations of visual representations that may significantly advance the state of the art.

We have proposed a novel framework for specifying, designing, and evaluating compositions of multiple visualizations in the same visual space that we call the composite visualization view. It is also not always straightforward to separate data visualizations from the visual structures—which is the case for overloaded and nested relations.

However, the design patterns presented in this paper are all based on evidence from the literature of how existing visualization tools have been put together. In other words, our focus has been to treat as components any technique has been presented in the literature as a standalone technique. Furthermore, this list of patterns is not necessarily comprehensive.

8.2 Delimitations

With respect to the delimitations of this study, we recognize that this list of patterns is not necessarily comprehensive. However, the list is intended to be representative of the patterns found in the literature.

8.3 Discussion

Again, situations that call for augmenting a representation that is high, and visual design dependencies are high. Limited space for the client visualizations, clutter is high, and visual design dependencies are high.

(a) Juxtaposed views. (b) Integrated views. (c) Superimposed views.

(d) Overloaded views. (e) Nested views.

[W. Javed and N. Elmqvist, 2012]
Multiple Views

• Facet (noun and verb)
  - particular aspect or feature of something
  - to split

• Partition visualization into views/layers
  - Either juxtapose (side-by-side), superimpose (layer), nest, etc.
  - Depends on data and encoding
  - Generally, superimposing does not scale as well
  - Multiple views eats display space (either large screens or small visualizations)
Multiple Views

➔ Share Encoding: Same/Different

➔ Linked Highlighting

➔ Share Data: All/Subset/None

➔ Share Navigation

[Munzner (ill. Maguire), 2014]
## Multiple Views

<table>
<thead>
<tr>
<th>Encoding</th>
<th>Data</th>
<th>All</th>
<th>Subset</th>
<th>None</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same</td>
<td>Redundant</td>
<td></td>
<td>Overview/Detail</td>
<td>Small Multiples</td>
</tr>
<tr>
<td>Different</td>
<td>Multiform</td>
<td></td>
<td>Multiform, Overview/Detail</td>
<td>No Linkage</td>
</tr>
</tbody>
</table>

[Muñzner (ill. Maguire), 2014]
Multiform
Multiform Views

• The same data visualized in different ways
• Does not need to be a totally different encoding (all choices need not be disjoint), e.g. horizontal positions could be the same
• One view becomes cluttered with too many attributes
• Consumes more screen space
• Allows greater separability between channels
Small Multiples

- Same encoding, but different data in each view (e.g. SPLOM)
Interaction with Multiform & Small Multiples

• Key interaction with multiform and small multiples: **brushing**
  - also called linked highlighting

• Want to understand correspondences between representation in the different views
Brushing
Schneiderman's Mantra

• Visual Information-Seeking Mantra [B. Schneiderman, 1996]:
  - Overview first
  - Zoom and filter (Chapter 13)
  - Details on demand

• Goal of the overview is to **summarize** all of the data

• Want specific **details** about some aspect(s) of the data, need another view/layer
  - May be permanent: side-by-side
  - May be a popup layer: often opaque or separated

• (see textbook Ch. 6.7)
Overview-Detail View
Overview—Detail (Different Encoding)
Overview-Detail (with Zoom-Filter)

• Detail involves some subset of the full dataset
• Involves user selection or filtering of some type

• How question: includes facet
• Examples:
  - Maps: partition into two views with same encoding, overview-detail
  - UC Trends: partition into multiple views, coordinated with linked highlighting, overview+detail of expenditures
Fig. 2: The Cerebral display of the TLR4 graph (V=91, E=124) with associated LPS and LPS+LL-37 time series. The small multiples show an overview of all 8 experimental conditions. The most noticeable differences between the LPS and the LPS+LL-37 condition occur at hour 4. By selecting the hour 4 conditions, the main window shows the computed difference between the two conditions.

Furthermore, the biologists' assessment of what constitutes a good layout varies depending on the nature of the biomolecules involved. In the undirected portion of the graph, which comprises protein-protein interactions that propagate a signal from membrane to nucleus, they wish to see the network structure so that they can follow the signaling cascade. Thus for this section of the graph, it is important to minimize edge crossings, even if it places interacting nodes somewhat far apart. In contrast, for the directed portion of the graph, representing the genes whose expression was altered in response to the signaling cascade, the biologists want to see the nodes grouped tightly by function, even at the expense of not being able to clearly see the interactions between them. Translating these desires into automated graph layout requires an algorithm that uses metadata associated with the nodes, in addition to the direct graph structure, for node placement. Positioning nodes according to biological meta-data defines a semantic substrate [34] so that node position reveals biological function. We wrote a simple simulated annealing-based graph layout algorithm that uses node metadata to guide node placement.

3.2 Small multiple views for multiple conditions

Cerebral uses small multiples [38] to simultaneously display multiple experimental datasets. Each small multiple contains a complete copy of the interaction graph with the same spatial layout, but with different coloring according to the experimental data it is displaying. Our design target was to handle from two to a few dozen gene expression conditions, and from 50 to 3000 nodes in the interaction graph. One obvious alternative to multiple small views would be a single changeable or animated view, where the color coding changes over time rather than being distributed over space [33, 32]. Comparing something visible with memories of what was seen before is more difficult than comparing things simultaneously visible side by side [31]. Thus, the limitations of human memory make comparing the few dozen conditions of our design goal through animation quite difficult [40]. Although small multiples would not scale to hundreds of conditions, they handle the current usage of 8-10 easily and will certainly accommodate the projected usage of few dozen conditions. A second alternative is to embed a glyph, such as a line graph or heat map, near or within the node itself [24, 32, 41]. While embedded glyphs provide good detail when zoomed in for a local view, they become indistinguishable when zoomed out for a global view of graphs larger than a few dozen nodes. The biologists often need to see such a view, as it more readily allows for the identification of interacting genes/proteins whose expression behaves similarly across several conditions. Thus, glyphs would not be appropriate in this domain.

Saraiya et al. [32] evaluated four approaches to integrating graph and time series data, comparing one versus two views and slider-controlled animation versus embedded glyphs. While they used 10 time series data points, in a good match for our problem domain, their graph contained only 50 nodes. They found many tradeoffs between task type, speed, and accuracy. Our design can be considered an attempt to combine the strengths of the four different interfaces they studied into a single interface for a problem where the tasks are complex, accuracy outweighs raw speed, and the graph is large.

3.3 Parallel coordinates and clustering for data-driven exploration

Cerebral’s main views focus on the interaction graph model of the biological system or process of interest. We also provide a data-multiform & Small Multiples (Cerebral)
Navigation across multiple views

- Often navigation in one view updates navigation in another
- Example: Maps: overview shifts as you move around in detail view
- Selections in one view may trigger selections in another
Multiple Views

Partition into Side-by-Side Views

Superimpose Layers
Partitioned Views

- Split dataset into groups and visualize each group
- Extremes: one item per group, one group for all items
- Can be a hierarchy
  - Order: which splits are more "related"?
  - Which attributes are used to split? usually categorical
Glyphs, Views, and Regions

- Glyphs are composed of multiple marks
- Views are a contiguous region of space
- A region is usually associated with a group of data
-Blurry lines of distinction between them
Example: Grouped Bar Chart

![Bar Chart Example](image-url)
Example: Small Multiples Bar Chart

Group 1
- Q108
- Q208
- Q308
- Q408
- Q109
- Q209
- Q309
- Q409

Group 2
- Orange bars

Group 3
- Green bars

Group 4
- Red bars

Example: Small Multiples Bar Chart

[M. Bostock]

D. Koop, CSCI 627/490, Fall 2020
Matrix Alignment & Recursive Subdivision

• Matrix Alignment:
  - regions are placed in a matrix alignment
  - splits go to rows and columns
  - main-effects ordering: use summary statistic to determine order of categorical attribute

• Recursive subdivision:
  - Designed for exploration
  - Involves hierarchy
  - User drives the ways data is broken down in recursive manner
Example: Trellis Matrix Alignment

[Becker et al., 1996]
Example: HiVE System

[Slingsby et al., 2009]
Example: HiVE System

[Slingsby et al., 2009]
Reducing Complexity
Reducing Complexity

- Too many items or attributes lead to visual clutter
- Interaction and Multiple Views can help, but often lose the ability to start understanding an entire dataset at first glance
- **Reduction** techniques show less data to reduce complexity
- Can reduce items or attributes (both are elements)
- **Filtering**: eliminate elements from the current view
  - "out of sight, out of mind"
- **Aggregation**: replace elements with a new element that represents the replaced elements
  - summarization is often challenging to design
- Another method is **focus+context**: show details in the context of an overview
Overview: Reducing Items & Attributes

Filter

→ Items

→ Attributes

Aggregate

→ Items

→ Attributes

[D. Koop, CSCI 627/490, Fall 2020]

[Munzner (ill. Maguire), 2014]
No class Tuesday

VOTE!