Interpreting Large Visual Similarity Matrices

Christopher Mueller, Benjamin Martin, and Andrew Lumsdaine
Open Systems Laboratory
Indiana University
Bloomington, IN 47405
{chemuell,benjmart,lums}@osl.iu.edu

ABSTRACT

Visual similarity matrices (VSMs) are a common technique for visualizing graphs and other types of relational data. While traditionally used for small data sets or well-ordered large data sets, they have recently become popular for visualizing large graphs. However, our experience with users has revealed that large VSMs are difficult to interpret. In this paper, we catalog common structural features found in VSMs and provide graph-based interpretations of the structures. We also discuss implementation details that affect the interpretability of VSMs for large data sets.


1 INTRODUCTION

Graphs (or networks) are mathematical or topological abstractions used to describe the essential relationships between objects. A graph is composed of a set of vertices, pairs of which may be connected by edges to show relationships between them. Many types of data, such as phone networks, transportation networks, and the World Wide Web, have a natural realization as a graph. In fact, any data set can be represented as a graph based on the implicit relations between its data items. If a strong enough relationship exists between two data items (e.g., using a distance metric), an edge is considered to exist between the two vertices representing the data items.

As more applications are found for graph algorithms, it is important to continue to develop methods for better visualizing graphs. Unfortunately, the traditional node-link graph visualization does not scale well to large data sets. Techniques such as graph coarsening and variable level-of-detail rendering [2, 7] can help display large graphs. But, as the size of the graph increases, so does the amount of information lost by these techniques.

The visual similarity matrix (VSM) is a technique for visualizing graphs that scales to very large graphs (Figure 1). Given a graph with a set of vertices connected by a set of edges, a VSM is generated by labeling the horizontal and vertical axes with the vertices and placing a point at each location \((u, v)\) where there is an edge between vertex \(u\) and vertex \(v\). Because they only require one point on the display for each edge, VSMs are capable of displaying very large graphs in a compact form. Using anti-aliasing algorithms, graphs with hundreds of thousands of vertices and millions of edges can be rendered on a standard display (e.g., 1280x1024).

VSMs take on many forms in different disciplines. A version of the VSM, the dot plot [5], is commonly used in bioinformatics to directly compare genomic or proteomic sequences. In contrast to most graph-based datasets, biological sequences have a natural ordering that adds additional meaning the features in the plot. Data clustering ([1,9]) and scientific tools [8] have also employed VSMs for some tasks. Our study [3] of graph ordering algorithms provides a detailed survey of literature for VSMs.

While simple to construct and very scalable, our experience visualizing real data using VSMs has revealed that they are very difficult to interpret for large data sets. The order of the vertices on the axes has a large impact on the amount of information that can be readily perceived. A well-ordered set of vertices can reveal complex structural features while poorly ordered vertices lead to images that are indistinguishable from noise. If correctly interpreted, visual similarity matrices can reveal clusters in graphs and show paths through the graph. For certain orderings, they can also suggest general properties, such as whether a graph is a small world graph or a power-law graph. But, identifying these features requires experience interpreting VSMs.

In this paper, we discuss the interpretation of visual similarity matrices by identifying common structures and providing a vocabulary for dealing with them. Similar efforts exist for dot matrix plots [6] and are useful for learning to use them. In addition to revealing structural features, we also present what we have found to be an essential collection of interactive features for VSMs.

This paper builds on previous works by cataloging features found in large visual similarity matrices and discussing their proper interpretations. Other implementations of VSMs are similar to ours, but we have identified some novel features that aid in interpretation.

2 GENERAL FEATURES

Features in visual similarity matrices are interpreted in the context of the underlying graph. Most features correspond to a relational pattern between vertices in the graph. However, some features can be artifacts of the ordering method. Thus, it is important to understand the features commonly found in VSMs and those left behind by ordering methods to properly interpret the data. In the next few
sections, we build up a vocabulary of features for visual similarity matrices, starting with the simplest elements and working towards more complex, compound features. Along the way, we discuss how to interpret each feature and identify possible misinterpretations.

**Axes** The axes of a visual similarity matrix are the basic reference system. Each axis is labeled with the data items in some order. The origin of the plot is the intersection of the two axes. The vertices on the axes start at the origin and are laid out in the same order for both the horizontal and vertical axes. Throughout this paper, the origin is assumed to be the lower-left corner of the plot. As discussed in the Introduction, the order of the vertices on the axes has a strong impact on the quality of the visualization.

**Points** The most basic features in a visual similarity matrix are the points that correspond to edges. When the number of vertices currently visible on an axis is less than or equal to the number of pixels available for the axis, each edge can be rendered using one or more pixels. However, when there are more vertices than pixels available, compression [8] or sub-pixel rendering techniques are required to show the presence of edges. To properly interpret the points in a VSM, it is important to be aware of the data to pixel ratio and also understand the technique used to combine multiple edges into a single pixel (Figure 1, right).

An important difference between points in VSMs and points in other matrix visualizations is that points in close proximity do not necessarily imply any relationship between their corresponding vertices (data items). For instance, in a dot plot, a diagonal line of points can be interpreted as a common sub-sequence in the data sets. The same line in a VSM, without other supporting points, carries no information about the relationship between the vertices that make up the line (figure 2).

Thus, visual similarity matrices should be ‘read’ along the rows and columns. Starting at any point, a three vertex path in the graph can be found by following the row for that point until a new point is reached, then following the column corresponding to the new vertex until another point is encountered (Figure 1). If the points are adjacent, an ‘L’, or elbow quickly identifies the path between the three vertices.

**The Diagonal** The main diagonal, simply referred to as the diagonal from here on, is the most important feature in a visual similarity matrix. The diagonal starts at the origin of the plot. Along with the axes, it forms a reference point for interpreting the visualization. The diagonal itself is composed of the intersection of the row and column for each vertex and represents the set of self-edges on the vertices (i.e., an edge from \(v\) to \(v\)). For most data sets, self-edges add no information and are excluded from the graph, leaving the diagonal devoid of points.

For undirected graphs, the line marked by the diagonal divides the plot area into two identical, mirrored sections. The upper and lower triangles in the plot show the same information. Figure 1 shows a VSM rendered using mirrored triangles and the upper and lower triangles. Rendering both triangles can help reinforce features along the diagonal by essentially doubling their size. It also helps reveal subtle patterns off the diagonal simply by having two copies visible on the screen. However, this effect can also make it difficult to identify smaller structures that are not amplified as much.

**Lines** Lines are the first level of features that may provide information about the underlying structure of the data. Lines fall into two main categories. Straight lines are horizontal and vertical lines that correspond to a ‘star’ pattern of edges emanating from a single vertex. For horizontal lines, the center of the ‘star’ is the vertex on the vertical (Y) axis (figure 3). Diagonal lines are any lines that are not straight lines. Diagonal lines may be composed of smaller sections of straight lines and may tend in the same direction as the main diagonal or be perpendicular to the main diagonal. Diagonal lines show paths through the graph.

Interpreting lines requires some care. Contiguous and broken straight lines, as long as they remain in the same row or column, contain the same information. Gaps in the lines are simply an artifact of the vertex order. The diagonal may change the direction of a straight line by 90 degrees. The new line is a continuation of the existing line. Straight lines thicker than one pixel carry more information and are discussed in the next section as blocks.

Diagonal lines carry information if they are more than one pixel wide or if the vertex sets from two diagonal lines intersect. For instance, in figure 4, if vertices \(a, b, c, d, w, x, y, z\) have the edges \(a - w, b - x, c - y, d - z\) and are ordered alphabetically on the axes, a diagonal line will form starting at \(a\) on the horizontal axis. In a dot plot, this would mean that the ‘words’ \(abcd\) and \(wxyz\) are the same. In a VSM, no relationship between the vertices in \((a, b, c, d)\) or \((w, x, y, z)\) can be inferred, even though the diagonal seems to connect them. However, the addition of the second diagonal line adds edges between \(x - y\) and \(y - z\), which creates paths between \(b, c,\) and \(d\). Wider diagonals and longer wider sections also show different paths through the data.

As demonstrated, paths along single pixel diagonal lines can exist if the line has supporting edges elsewhere on the plot. Most often, the supporting edges appear as diagonal lines parallel to the original line that share some common vertices. In a mirrored plot, these may be lines just on the other side of the main diagonal that are parallel to other diagonal lines. More formally, if the vertex sets from two single pixel wide diagonal lines intersect, then paths exist between the vertices in the intersection.

Given the potential for misinterpreting diagonal lines, diagonal lines represent one of the more difficult features to use effectively for data analysis in VSMs. This is particularly true for biologists who are familiar with the language of dot plots and readily confuse the two visualization methods. The main source of confusion is the difference between naturally ordered data sets, where diagonals convey meaning, and graphs, where they may convey meaning.

Diagonal lines may have steep or shallow slopes and the slopes may vary over the course of the line. If a diagonal line with a variable slope is composed of many single pixel straight line segments connected by elbows, the interior vertices (i.e., those not used to make the elbows) on the straight lines can be removed without losing any information about the path described by the diagonal line. Each point on the straight line represents a vertex that can be reached by one extra step from the path.

**Wedges and Blocks** Cliques and connected components are important structural features in graphs and generally correspond to the
Figure 4: Two different interpretations of the same diagonal line. Without the supporting edges, the vertex sequences \( wxyz \) and \( abcd \) have no intra-sequence relationships. However, the addition of the two hollow edges on the right provides paths between \( b, c, \) and \( d \). Because supporting edges may be in another area on the plot, diagonal lines are very difficult to interpret correctly.

Figure 5: A wedge (left) and a block (right) in a VSM. A wedge falls on the diagonal and indicates a clique or a cluster. A block is off the diagonal and indicates a bipartite graph. If a block is aligned vertically or horizontally with a wedge, then the block can be considered part of the cluster.

common notion of a 'cluster'. When the axes are ordered appropriately, visual similarity matrices can instantly reveal clusters in the graph. However, as with points and lines, the visual features that identify clusters must be carefully interpreted and can be deceptive.

Figure 5 shows the two structures that signal a cluster. Clusters can appear in VSMs as triangular wedges on the diagonal. Note that when the visualization is mirrored, clusters are blocks on the diagonal. Because off-diagonal blocks have a different interpretation, we use the term wedge for on-diagonal blocks, and block for off-diagonal blocks). A wedge with all the points filled in is a fully connected subgraph, or a clique. In this case, there is a direct relationship between all vertices in the wedge.

Off-diagonal blocks (i.e., block structures that do not touch the diagonal), can only be interpreted as bi-partite subgraphs. The horizontal and vertical vertices form two groups of vertices that have no intra-group edges but many inter-group edges (Figure 5, bottom). However, if the horizontal vertices also have corresponding vertical vertices on the axis, the block plus the wedges (one for each group of vertices) can be interpreted as a cluster.

In addition to being connected to distant wedges via shared vertices, blocks may be connected to other blocks in the same manner. Any blocks that fall in the same rows or columns as the main block can be interpreted as part of the main block. Simply reordering the vertices will bring them together. As with single pixel elbows, blocks can form paths through the graph.

The ordering can affect the appearance of the blocks. Blocks with square corners but gaps inside can be reordered so the gaps appear on the outside as rounded corners. Thus, sharp features should only be considered for interpretation when the blocks are completely filled.

3 ALGORITHMIC FOOTPRINTS

In addition to features found under most orderings, some ordering algorithms leave behind a characteristic footprint [3]. Footprints are often composed of common features and show the path the algorithm took through the graph. It is important to recognize footprints and to place the interpretation in the context of the ordering algorithm. There are three main types of footprints that we have identified are shown in figure 6 and include envelopes, horizons, and galaxies.

**Envelopes** Envelope footprints are characteristic of breadth-first search-based (BFS) algorithms. BFS algorithms form the foundation for many graph-theoretic (e.g. Dijkstra) and sparse matrix re-ordering algorithms (e.g. Cuthill-McKee [4]). The term envelope is borrowed from the sparse matrix community where one goal is to minimize the ‘envelope’ of a sparse matrix to reduce its memory usage. Visually, a mirrored VSM with an envelope footprint takes on the shape of a leaf. The outer edge of the leaf, or the envelope, is a diagonal line that traces the search path through the graph. All the points in the VSM are contained inside the envelope.

A disconnected graph (i.e., a graph with several clusters and no inter-cluster connections) or a graph with clusters that are loosely connected may have multiple envelopes bulging off the diagonal. Algorithms based on BFS may add additional features to the interior of the envelope, but no features will ever leave the envelope.

**Horizons** Horizons highlight the path taken by depth-first search (DFS) based algorithms. A horizon is defined by a solid diagonal line that follows the main diagonal and at some point makes a sharp turn away from the main diagonal. A line may continue to follow the main diagonal, but the majority of the edges in the graph will fall between the horizon and the horizontal axis. A graph may contain multiple horizons spreading off the main diagonal. This will occur when there are multiple disconnected clusters in the graph, but can also occur when there are loosely connected clusters and the DFS goes as deep as it can in one cluster before entering the next.

**Galaxies** Galaxy footprints can result from ordering methods that ‘probe’ the underlying graph structure for analytic properties. For instance, spectral methods and degree based orderings both yield galaxy footprints. In a mirrored VSM, a galaxy appears as a collection of points that appear to be attracted to a center of gravity on the diagonal or the diagonal itself. Their appearance is reminiscent of elliptical galaxies and head-on spiral galaxies, respectively. Galaxy footprints typically contain few general VSM features but can reveal characteristics about the graph. For instance, a head-on spiral generated using a spectral ordering that clings to the diagonal suggests a small world graph.

4 IMPLEMENTING VSMs

Visual similarity matrices are capable of displaying a large amount of information using a small amount of screen real estate. We have scaled VSMs to over 250,000 data items with 3 million edges. In order to interpret any data set with more than a trivial number of vertices and edges, interactive features and additional interpretation aids are important. Additionally, the user should have direct control over various rendering options in order to fully explore the data. In this section, we discuss the features we found to be essential for navigating large graphs using visual similarity matrices and explain how to use them to aid in interpretation.

**Color** Color can be used in two primary ways to enhance the interpretability of VSMs. First, if the edges are weighted, the points can be shaded to correspond to the edge weight. In data sets, edge
weights are often the (dis)similarity between two data items. By coloring the points using a scale that fades into the background, the relative strength of specific relations in a cluster is readily revealed. Weaker edges blend into the background, causing clusters without strong connections to appear less intense than if they were rendered with a single color. Additionally, weak outliers do not stand out and command undue attention.

The second use of color is for identifying categories of edges. In real data sets, it is not uncommon for the final graph to be built from multiple sources or for the edges to fall into distinct categories. By coloring using categories, it is possible to identify homogeneous and heterogeneous clusters and paths.

We have found that in both cases it is important to be able to toggle between multi- and single-color views of the VSM. The single-color view allows the user to view the graph from a purely structural perspective to identify interesting regions. The multi-color mode adds additional information that adds more context once an interesting region has been identified.

Interactivity The ability to navigate the plot area is essential when the number of vertices exceeds the horizontal or vertical resolution of the display. The most important interactive feature is an arbitrary aspect ratio zoom. The zoom tool serves two essential purposes for interpretation. It allows the user to examine points and lines for elbow connectors to ensure relationships between features are real. The arbitrary aspect ratio enables inspection of all edges for a single vertex or a small set of vertices against a much larger set of vertices.

In addition to a zoom tool, a selection tool is necessary to supply the user with details about the currently displayed vertices. The selection tool should support options for selecting the visible vertices or the visible edges.

Alternate Orderings Pajek [1] has long provided support for alternate orderings (partitions) of vertices. Because some orderings bring out certain features and there may exist multiple, hand-crafted orderings for the data, we have found the ability to explore multiple orderings is essential for using VSMs to interpret large data sets.

Anti-Aliasing Large graphs will not fit on the display if each vertex is assigned one row or column of pixels. Simply placing a solid point at the pixel location for each edge can quickly fill up an entire display with points. Anti-aliasing algorithms are supported by the hardware in most modern graphics cards and provide an unambiguous method for sub-pixel rendering. While it is possible to use custom bucketing algorithms to combine points, the performance overhead for large graphs can be prohibitive. Bucket algorithms also introduce another variable for the user to consider while reading the graphs. As with colors, we have found it important to provide a toggle for anti-aliasing. For very large graphs, single, remote edges can easily fade into the background and turning off anti-aliasing can reveal them. Figure 7 shows the effects of anti-aliasing on a large graph.

Linked Views In addition to the main matrix display, other visualizations can aid in the interpretation of the data. For smaller graphs, or induced subgraphs built from user selected vertices and edges, node-link diagrams can show a clearer view of the data. Matrix Zoom [2] uses a similar strategy for drilling down into a large graph.

Property plots linked to the axes can provide important details about the data items. Property plots display the values for one attribute on the data items. The size of the plot is locked to the size of the axes it is connected to and each tick on the axis matches the corresponding data item in the VSM. The other axis on the property plot corresponds to the range or category values for the feature. For instance, if the VSM is displaying a graph of relations between chemical structures, property plots may be available that show the molecular weight of the compounds and the total polar surface area. For a chemist, this additional information may provide details that help interpret the features in the VSM.

5 Conclusion Visual similarity matrices are a powerful tool for exploring very large data sets. They are an inherently interactive tool for data exploration and as a result their interpretation depends not only on a vocabulary for reading the visualization but also careful implementation of applications that use them. In this paper, we have dissected the features found in visual similarity matrices and provided the basic vocabulary for interpreting the images. We also discussed the features necessary to effectively implement VSM applications.

6 Acknowledgments This work was funded by a grant from the Lilly Endowment. The authors would like to thank Jim Costello and the Andrews-Dalkilic research group from the CGB at Indiana University for their valuable feedback and Katy Börner from SLIS at Indiana University for encouraging us to publish these results. We also thank Sun Kim for originally suggesting this project.

References