Cluster Metrics and Temporal Coherency in Pixel Based Matrices

by

Thomas Hayden

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2014 by the
Graduate Supervisory Committee:

Ross Maciejewski, Chair
Yalin Wang
George Runger
Elizabeth Mack

ARIZONA STATE UNIVERSITY

May 2014

ABSTRACT

In this thesis, the application of pixel-based vertical axes used within parallel coordinate plots is explored in an attempt to improve how existing tools can explain complex multivariate interactions across temporal data. Several promising visualization techniques are combined, such as: visual boosting to allow for quicker consumption of large data sets, the bond energy algorithm to find finer patterns and anomalies through contrast, multi-dimensional scaling, flow lines, user guided clustering, and row-column ordering. User input is applied on precomputed data sets to provide for real time interaction. General applicability of the techniques are tested against industrial trade, social networking, financial, and sparse data sets of varying dimensionality.

To my family and friends.

ACKNOWLEDGMENTS

I would like to express my gratitude to my supervisor Dr. Ross Maciejewski for his time, guidance, and seemingly endless patience throughout my thesis. I would like to thank my friends for their continued support and feedback. I would also like to express my thanks to my other committee members Dr. Yalin Wang, Dr. George Runger, and Dr. Elizabeth Mack for their time, comments, and suggestions.

Page

LIST OF FIGURES

Chapter 1

INTRODUCTION

The extraction of relevant information from temporal high-dimensional data can be challenging. However, visualizations can be a useful tool in aiding the consumption of this data. Many visualizations exist for consuming data this, but many also suffer from the curse of dimensionality. That is, these visualizations commonly decrease in usefulness as the number of dimensions grows. This work presents a novel data visualization technique, Parallel Pixel Matrix, which combines variations of the Parallel Coordinate Plot and Pixel Matrix visualizations in an attempt to provide a better tool for the consumption of temporal high-dimensional data.

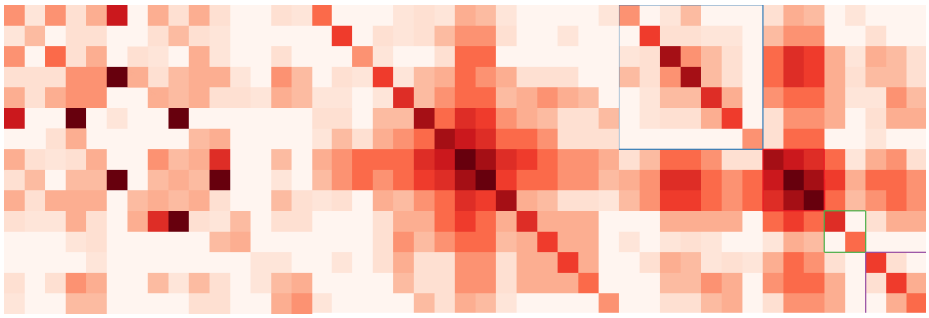


Figure 1: (left) A pixel matrix with no defined ordering. (center) A pixel matrix with color clustering using the Bond Energy Algorithm. (right) A pixel matrix with color clustering using Multidimensional Scaling with the Bond Energy Algorithm.

The first visualization, the pixel matrix, is an effective means to visualize large high-dimensional data sets at a single point in time. This is done through mapping the

value of dimension tuples onto screenspace within a grid layout into individual cells, typically by using color to signify the strength of interaction between the dimensions (Fig. 1 left). Due to the nature of human sensitivity to color each cell can be scaled down to a significantly small size while still retaining usefulness. However, without any kind of ordering of the rows and columns a user can become blind to patterns due to the large amount of information available. Using row and column ordering algorithms such as the Bond Energy Algorithm (BEA) (Fig. 1 center), patterns which were otherwise hidden to the user become easily observable. This can be combined with a dimensionality reduction algorithm such as Multidimensional Scaling to aid a user in finding patterns within groups when present [1].

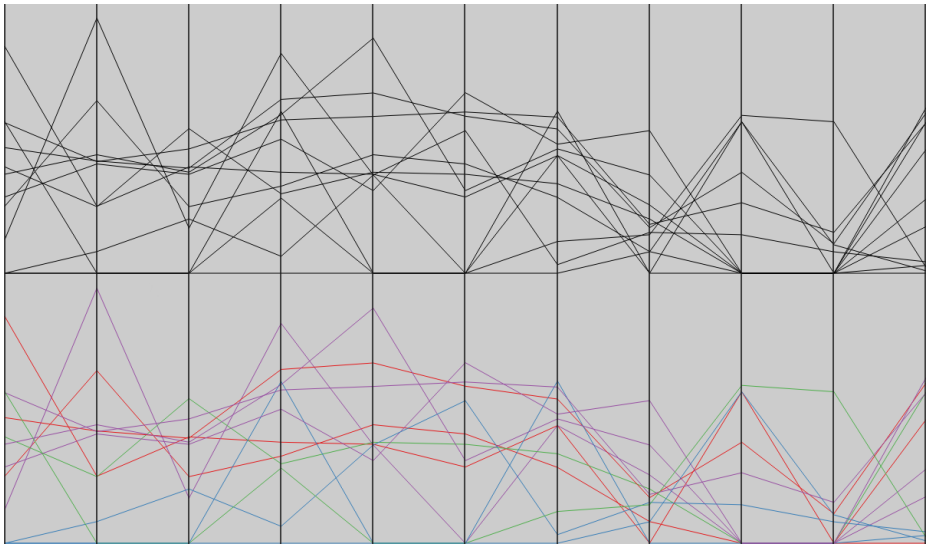


Figure 2: (top) An example of a basic parallel coordinate plot. (bottom) An example of a parallel coordinate plot with color clustering.

The second visualization, the parallel coordinate plot (Fig. 2 top), is an effective means of visualizing orthogonal coordinate systems of high dimensionality by

projecting each dimension onto parallel axes [2]. When working with time series data consisting of orthogonal data, parallel coordinate plots can be used to discover long-term behavior by projecting the time dimension onto the parallel axes, but in doing so loses the ability to display the interaction between dimensions within each time slice. This type of plot can be difficult to interpret when the number of nodes placed within it causes node lines to overlap to the point that they obscure each other. In cases where the data set is Nominal, a clustered parallel coordinate plot (Fig. 3) can be used to reduced the number of individual lines by combining them into flow lines. In cases where individual lines are required line overlap can be minimized by use of color clustering node lines (Fig. 2 bottom) with alpha transparency, which allows line clusters to be found even when a large amount of overlap is present [3].

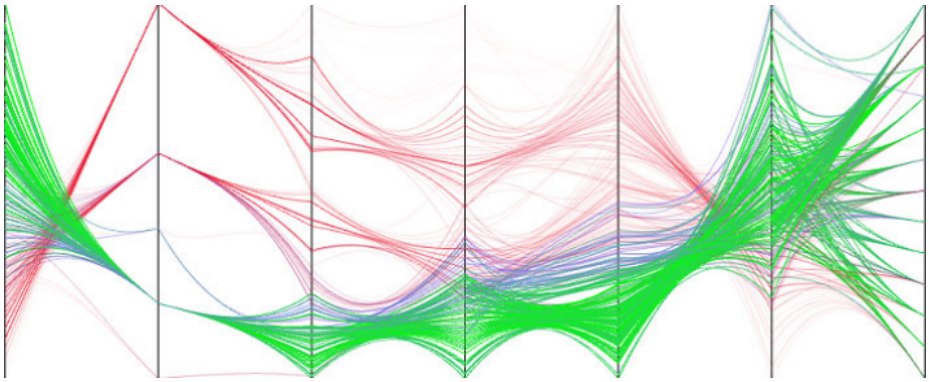


Figure 3: An example of a parallel coordinate plot with visual clustering.

Source: Visual clustering in parallel coordinates [3]

The Parallel Sets (Fig. 4) and NodeTrix (Fig. 11) visualizations are examples of significant stepping stones to the visualization presented. The Parallel Sets is a variation of the parallel coordinate plot The visualization uses a clustered (or

categorical) parallel coordinate plot with flow lines to represent nominal data [3, 4], while the NodeTrix presents clustered information within a collection of pixel matrices connected by flow lines. The visualization described in this paper attempts to combine these ideas with several others by replacing each cluster used in the parallel set visualization with a matrix representing the clustered data, and uses each parallel axes as a time slice. This resulting visualization allows for a novel solution to the problem of tracing large clustered data sets through a time series, and was possible due to the parallel sets work done by Kosara et al. [5], the NodeTrix work done by Henry et al. [6], as well as many others.

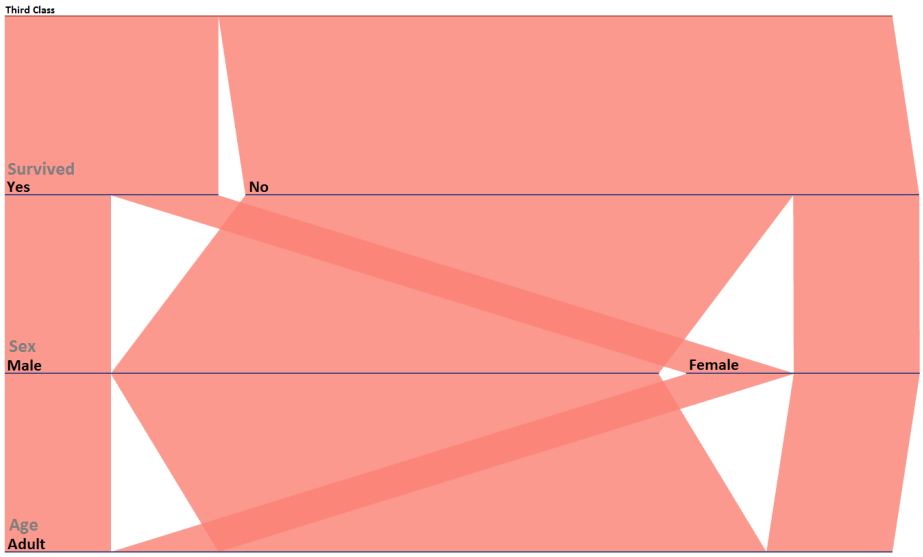


Figure 4: An example of a parallel set visualization.

Source: Parallel Sets Application [5]

This thesis is organized into six chapters. In Chapter 2, previous work is detailed in parallel coordinate plots, pixel matrices, and graphs. Chapter 3 goes

into more detail about pixel matrix organization. Chapter 4 discusses many of the visualization choices made. Chapter 5 describes the results of the work through the use of use cases. The final chapter concludes the work with a summary of finding and remaining problems for future work.

Chapter 2

RELATED WORK

Three major aims were explored in developing this visualization: the parallel coordinate plot, pixel matrix, and graphs. The following background is intended to provide the reader with a better understanding of the usefulness and problems that these visualizations attempt to solve.

1. Parallel Coordinate Plot

Parallel coordinate plots allow the visualization of high dimensional data using a series of lines in the interest of gaining insight into patterns occurring within the data. These lines are representative of data within an orthogonal coordinate systems where some set of its dimensions have been projected onto parallel axes and another set onto the horizontal axes. For example, the vertical lines within Figure 3 could represent properties of cars (acceleration, top speed, MPG, etc), while the horizontal lines could represent individual cars models (Ford Focus, Nissan Leaf, etc). The point at which a horizontal line (car) crosses a vertical line (property) represents a tuple, such as (Ford Focus, 32MPG). Given the visual simplicity of the plot, patterns such as the grouping of horizontal lines can quite easily be determined. This could then be used in this example to place these cars within a single class, while those which diverge from a group could be further inspected for additional classification.

Inselberg [2] introduced the parallel coordinate plot in 1959 as a useful tool for comparing continuous data. The parallel coordinate plot can be quite effective at displaying large data sets, but given enough data the plot can become saturated as shown by Heinrich et al. [7]. Even in this example where a smaller number of clusters are used in a parallel set, it can be difficult to understand when a large number of disperse edges exist (Fig. 5). Most attempts at solving this issue revolve around manipulating the individual lines, line bundling, or some of combination of the two.

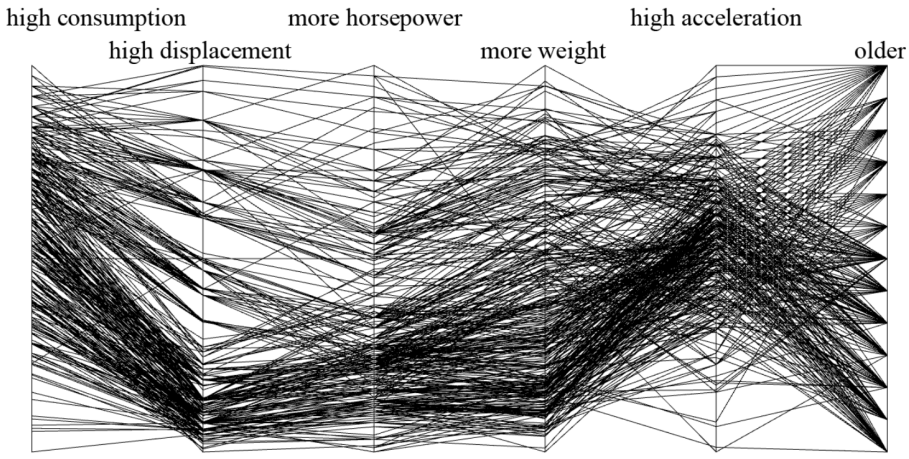


Figure 5: An example of a saturated parallel coordinate plot.

Source: Evaluation of a Bundling Technique for Parallel Coordinates [7]

For line manipulation, Wallgren et al. [8] show that in cases where line width is used to indicate significance, the reader will not confuse this with the vertical width of the line. However, they also provide a visualization in which line width is superseded by small rectangles where vertical space indicates significance. Rodrigues et al. [9]

make use of a combination of line thickness and color saturation to denote relevance while hue is used to indicate frequency in what they call a Relevance Plot, but only apply their visualization towards continuous data sets. Xu et al. [10] and Talbot et al. [11] discuss their findings of curved line use in graphs, and focus on trace-ability and perception of slop respectively.

The application of line bundling is used by Fua et al. [12] in an attempt to solve the problem of line saturation using hierarchical clustering, which allows for visualizations of varying detail. Heinrich et al. [7] evaluate several techniques in which they find that bundled curves are effective at revealing correlations between neighboring data attributes. A similar process is used by Zhou et al. [13] in their work to reduce cluster problems in visualizations. A good example of this overlap occurs in the work done by Vandewalle et al. [14] where even a small number of clusters used in a parallel set can be difficult to understand when a large number of disperse edges exist. The process used in this paper also groups edges, and, instead of using straight lines, uses curves in an attempt to reduce overlap that might obscure underlying patterns.

Another active area of research with parallel coordinate plots is how to better order or display the vertical parallel axes. Edsall [15] experiments with parallel coordinate planes injected in place of parallel lines for use with geographic data. Andrienko and Andrienko [16] attempt to combine parallel coordinate plots with geographical data by enabling selection of regions or lines in either visualization. This

allows for transformations which affect both displays, which is much like the linked widgets used within the application produced with this paper. Due to the intended use of the application being temporal, the choice was made to lay out each axis as a time slice in fashion similar to that used by Butkiewicz et al. [17].

Some existing research attempts to combine several of these ideas, such as Kosara and Ziemkiewicz [5] who make use of categorical instead of continuous data in what they call their Parallel Sets application. This work extends many of the ideas previously presented by Rodrigues et al.'s [9] several years earlier, but instead approach the problem using clustered data sets. Pilhöfer and Unwin [4] provide a write up for an R package influenced by Kosara and Ziemkiewicz's ideas that produces similar results, but replaces the frequency controlled sized categorical boxes with fixed sized labels. In this application each continuous vertical line is replaced by categorical boxes where the size of each is representative of the frequency of its occurrence, and edge significance is represented by line width. Like the work done in the parallel sets application developed by Kosara and Ziemkiewicz, the work in this paper details information about the movement of individual nodes, but bundles edges into curved flow lines. Schonlau [18] combines both categorical and continuous variables in a parallel visualization with mixed results, naming the visualization a Hammock Plot. Few [19] also makes use of both categorical and continuous variables, using circles of varying size to correlate frequency of categorical data.

In work similar to this paper, Siirtola [20] finds evidence for combining parallel

coordinates with a reorderable matrix through the use of highlighting to reduce cognitive load. However, instead of attempting to include one visualization within the other, his work focuses on the usefulness of displaying both visualizations side-by-side.

2. Pixel Matrix

The pixel matrix visualization, like the parallel coordinate plot, allows the visualization of high-dimensionality data but uses color contrast to indicate value instead of position. This is achieved by placing each dimension tuple onto the screenspace within a grid layout of individual cells, where each cell's position indicates its corresponding dimensions. For example, in Figure 6 the top-left cell represents the value of the tuple (b, d), which due to its darker color has a higher magnitude value than the tuple (c, d) represented in the top-right cell. This layout allows for a densely packed visualization compared to many other methods. However, like the parallel coordinate plot, the pixel matrix suffers from the problem of ordering choice. Due to both of these reasons, there are a number of different cell manipulation and ordering variations which exist.

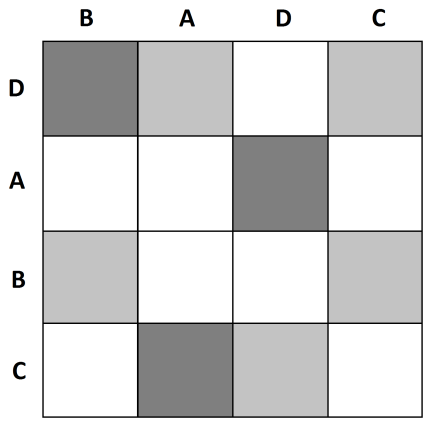


Figure 6: Pixel Matrix tuple example.

A good example of individual pixel cell manipulation is provided by Oelke et al. [21] who in several example techniques use halos, colors, distortion, and hatching, in an attempt to "boost" the recognition of significant pixels. Yi et al. [22] have introduced what they named the TimeMatrix (Fig. 7). This visualization provides a temporal display of information by replacing the normally single colored cells with a small bar graph with one axis representing time and the other value. In addition, they make use of the margins of the graph to provide similar small graphs which display a summarization of each node the column or row represents.

Much of the research on pixel matrices focuses on clustering through the use of row and column reordering. Brinton [23] in 1914 provides an early example of this row and column ordering. Bertin some years later in his book Semiology of Graphics [24] provides similar examples of matrix reordering in which he advocates

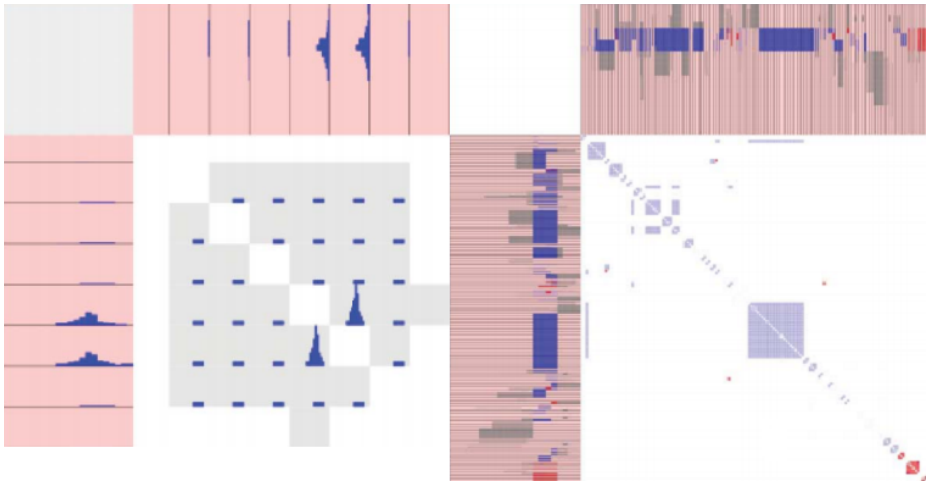


Figure 7: (left) Glyphs can be quite useful when the number of nodes is low enough to show detail. (right) Glyphs in larger data sets can sometimes be less useful than simple uses of color.

Source: Timematrix: Analyzing temporal social networks using interactive matrix-based visualizations [22]

that a simpler more logical diagram can be produced from a complex diagram. Bertin like Brinton before doesn't provide an algorithmic method for finding this ordering, but instead sorts them intuitively by hand, a method which becomes unreasonable with large data sets. McCormick et al. [25] later introduced several algorithms using reordering, including Bond Energy Algorithm, a simple to implement method used to calculate how clustered a particular permutation is. McCormick et al. expand upon these ideas on the Bond Energy Algorithm, for which an optimal $N!M!$ and quicker sub-optimal $(M^2N + N^2M)/2$ version exists where M and N are the number of rows and columns. Kusiak and Chow [26] later develop a faster method for identifying clusters in binary datasets. Wilkinson and Friendly [27] provide a useful summary of

the history of clustered heat maps, while Liiv [28] goes into more detail of reordering methods. Interesting, Henry et al. [6] decide to forgo all automated reordering in their hybrid matrix node-link visualization they name Nodetrix, due to the dense nature of the matrices produced and the authors belief that "...manipulation is key to understanding a network and its potential multiple interpretations"

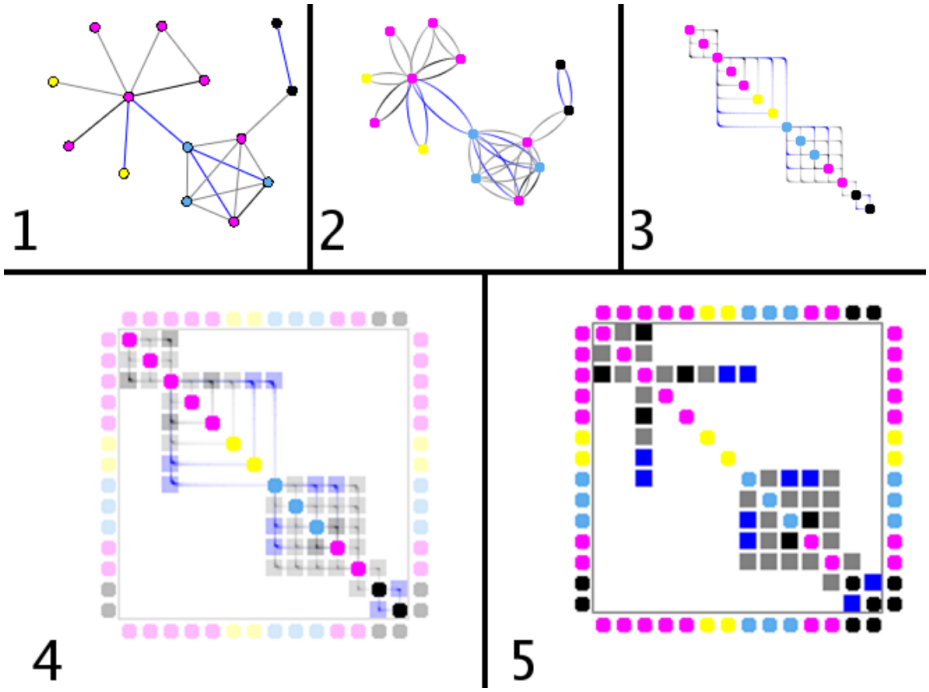


Figure 8: An example of a transformation from a node-link diagram to a matrix.

Source: NodeTrix: A Hybrid Visualization of Social Networks [6]

3. Graph

The node-link diagram is an alternative visualization of graph data to the pixel matrix (Fig. 8.1). Of the three major visualization techniques used the node-link

diagram is perhaps the oldest, with known examples dating back to at least Leonhard Euler's Seven Bridges of Knigsberg in 1736 [29]. The node-link diagram also has the distinction of being more intuitively understood than the other two diagrams when using small data sets [22]. The node-link diagram however suffers from the problem of occlusion in large dense data sets, where the pixel matrix, despite users lack of familiarity, excels (Fig. 9) [30].

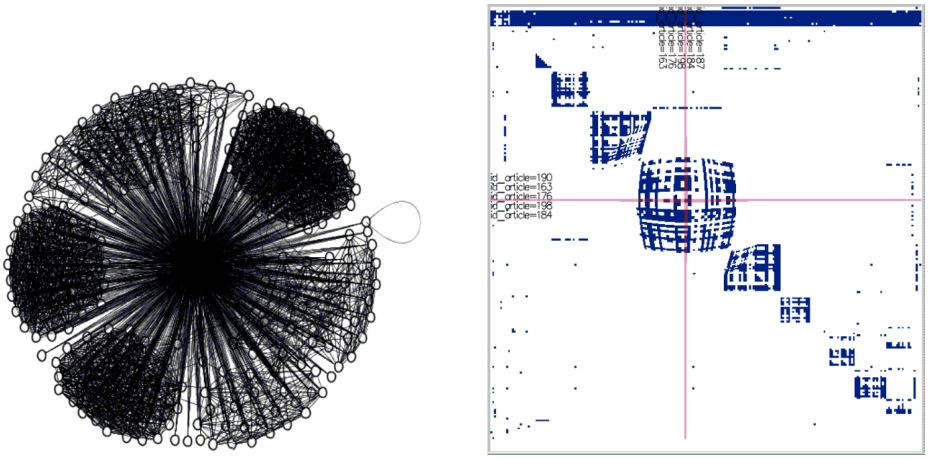


Figure 9: (left) An example of a dense node-link diagram and (right) an example of the same data within a matrix.

Source: VISEXP: visualizing constraint solver dynamics using explanations [31]

Though often used only with boolean data, variations of the node-link diagram can use modifications of edges to indicate more complex values [30]. Henry et al. [6] show in their work the lack of readability that can occur in dense node-link diagrams and make use of more complex edges with a hybridization of node-link and adjacency

matrix diagrams to minimize this problem (Fig. 10, 11). In their work, they experiment with the aggregation of links between matrices. Henry and Fekete [32] create another hybridization in which the row and column margins of an inner adjacency matrix are used as vertices and connected with links in a method somewhat similar to that used by Yi et al. [22]. Abello and Ham [33] use a combination of node-link, tree, and matrix visualizations to model hierarchical data to create a zoomable application. Krstajic et al. [34] produce a graph visualization technique through which news stories can develop and evolve over time in a method that looks quite similar to the wordle flow visualization produced with this paper.

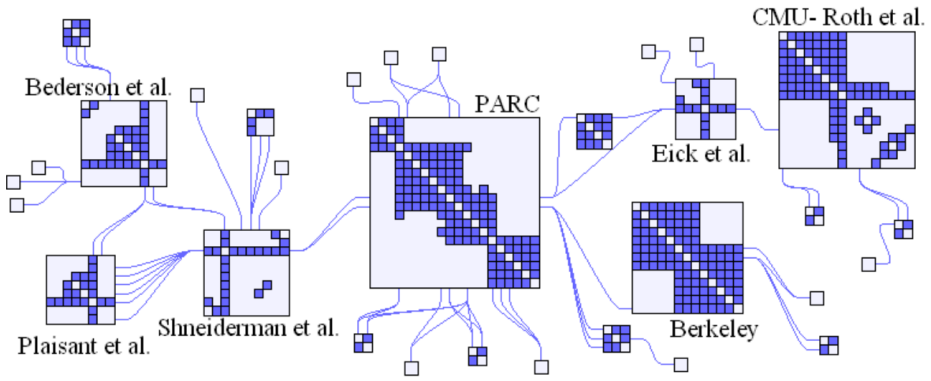
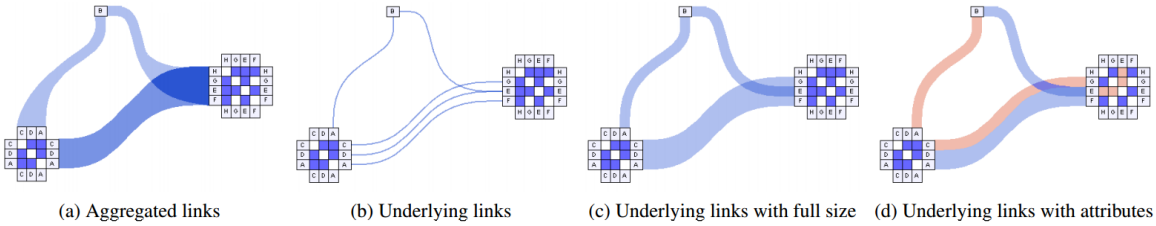


Figure 10: An example of the combined use of node-link and matrix visualization ideas.

Source: NodeTrix: A Hybrid Visualization of Social Networks [6]

While many papers focus on the importance of the algorithmic nature of graph drawing, perhaps just as important are their aesthetic quality. One method of achiev-

ing this is through the minimization of crossing edges [35]. Purchase [36] experiments with several graph drawing aesthetic ideas, concluding that while minimizing link crossings is effective, minimizing bends and maximizing symmetry provides little efficacy. Huang et al. [37] find in a user study that while attempting to maximize a single algorithmic design choice can be effective, a drawing that makes compromises in order to produce a more aesthetically pleasing drawing can be even more effective. Purchase et al [38] find common aesthetically pleasing criteria, and demonstrate the importance of automatic layout combined with manual editing in an experiment where users are given the opportunity to draw their own graphs. In an attempt to avoid the problem of edge crossings, a strictly node based diagram is provided within our application for displaying the results of Multidimensional Scaling. This takes advantage of the relative positioning that users can intuitively infer, with a lesser problem of occlusion than a visualization using links.



(a) Aggregated links (b) Underlying links (c) Underlying links with full size (d) Underlying links with attributes

Figure 11: A more advanced example showing variations of line use to indicate link values.

Source: NodeTrix: A Hybrid Visualization of Social Networks [6]

Much of the work done within the pixel matrix, parallel coordinate plot, and graphs by the previously mentioned authors have influenced in some way the work produced by this thesis. Of greatest significance is the work done by three groups: Bertin [24], Kosara and Ziemkiewicz [5], and Henry et al. [6]. While much of the later work involving row and column ordering focuses on its algorithmic aspect, Bertin [24] had the idea that proper row and column ordering is an inherently user choice. Due to this, while the work produced with this paper uses algorithmic automation where possible, an attempt was made to leave as much interaction available to the user as possible. Kosara and Ziemkiewicz's [5] work with the Parallel Sets application greatly influenced the idea of how clustered flow lines could be used to detail movement between axes. Lastly, the work done by Henry et al. [6] greatly influenced the more aesthetic qualities noticeable within the rendering of flow lines and pixel matrices.

Chapter 3

PIXEL MATRIX ORGANIZATION

A significant amount of previous research has been done into row and column ordering within pixel matrices, and for good reason [23–26, 28]. Choosing a poor ordering can make an otherwise useful visualization into a patternless eye sore. The work presented in this paper attempts to avoid this through the combined use of previously explored solutions, as well as several novel ideas. Initial clustering is supported through the use of K-Means, as well as a Multidimensional Scaling adjusted version of the algorithm. Nested clustering can optionally be performed through recursive K-Means or use of the Bond Energy Algorithm. In addition to the base and nested clustering, a variation of the K-Means algorithm is used to allow preservation of clusters over time when time series data of more than one time slice is used.

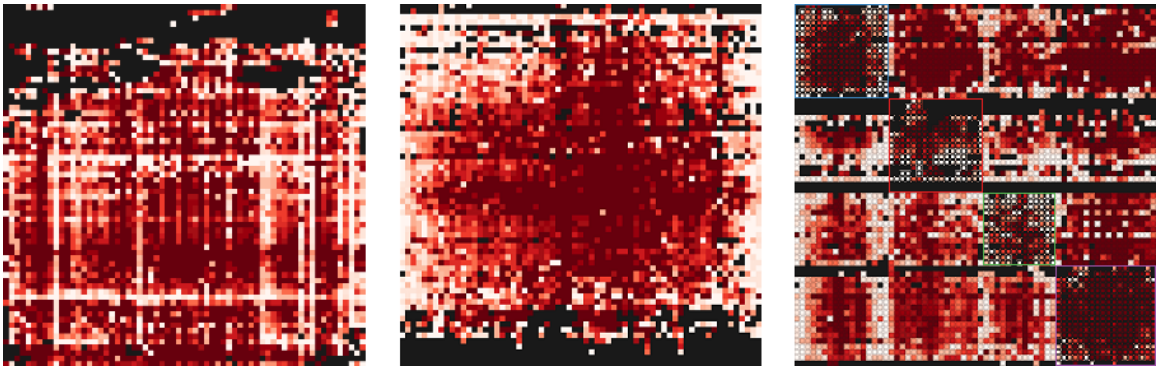


Figure 12: (left) Global BEA Results are permuted over the entirety of the time series. (center) Local BEA Results permuted over the currently selected time slice. (right) MDS K-Means BEA Results permuted within clusters local to a time slice.

Automated row and column permutation is provided using the Bond Energy Algorithm (BEA). This provides a method for bringing forward "clumpiness, denseness, and connectedness" in a fashion similar to Bertin's work but in an algorithmic fashion that might otherwise be hidden from a user [24, 39]. The algorithm permutes the rows and columns until all possible combinations have been exhausted, and at each permutation a sum is calculated of all neighbouring cells with matching values (Fig. 13). The permutation(s) with the maximum value is then determined to have the highest clumpiness.

$$max\{\sum_{i=1}^{i=M}\sum_{j=1}^{i=N} a_{ij}[a_{i,j} + a_{i,j+1} + a_{i-1,j} + a_{i+1,j}\} \tag{3.1}$$

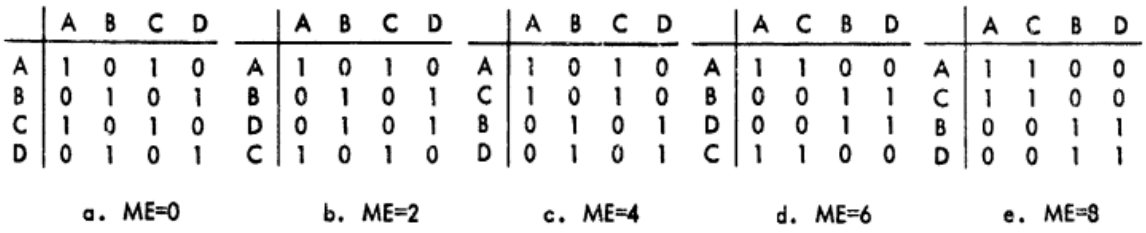


Figure 13: Example of Bond Energy Algorithm. The starting matrix at A has the lowest possible score which increases with each shown iteration until the highest score is found on iteration E. Note that not all possible iterations are shown, and that the score of each successive iteration may decrease as well.

Modified from source: Identification of Data Structures and Relationships by Matrix Reordering Techniques [39]

Owing to the optimal algorithm requiring $2N!$ permutations, a suboptimal solution requiring $\frac{M^2N+N^2M}{2}$ permutations published by McCormick et al. is instead used in tandem with in memory lookup tables to allow the process to run in real time [25]. The algorithm isn't limited to working with binary data, and is easily extended. Climer and Zhang describe several problems that may be encountered when working with such data however, but are able to produce desirable results when normalization is appropriate [40].

The BEA algorithm can be used in three modes: global, local, and within an Multidimensional Scaling (MDS) K-means clustering. When used in global mode (Fig. 12 left), the algorithm reorders the data set in a way that produces a variable number of loosely defined clusters which can be useful for finding trending relations between variables over the entirety of the data set. When used locally (Fig. 12 center) this produces one large cluster with the relations near the center being the strongest

in that year. In a similar fashion, when used with MDS and K-Means (Fig. 12 right) the algorithm again produces a single cluster, but for each block generated by the previous algorithms.

2. Multidimensional Scaling

A combination of Multidimensional Scaling (MDS) and K-Means clustering are provide as a basis for user guided clustering. The primary intended use of this visualization is to take advantage of a user's judgment of similarity as discussed by Kruskal and Wish [1] to cluster and filter the data in a way that the algorithms themselves may not be capable. This process can occur through either the customization of clustering algorithm input values, or assignment and filtering of individual nodes within the display.

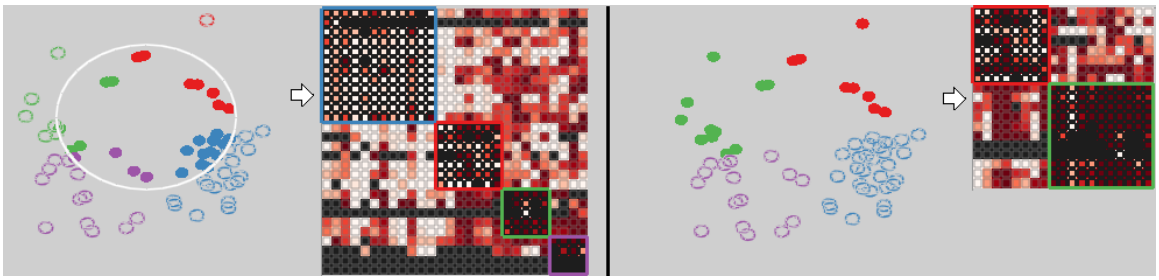


Figure 14: (left) Selection Circle Nodes from several clusters are selected using a click-and-drag selection circle. (right) Incremental Cluster Selection Two clusters are selected by double clicking each cluster while holding down the shift or ctrl key.

The MDS visualization's strength lay in its ease of use and intuitive interface,

which allows for quick filtering of large amounts of data. Due to the amount of screen space required by each node however, even with zooming this method of visualization begins to lose its effectiveness as the number of objects increases. This problem can be mitigated by undocking the visualization and using a secondary screen for display. Most interaction is provided through several mouse functions:

- Hovering the mouse over any node provides the name of the node, and highlights the node in the pixel matrix.
- Individual variables can be selected by clicking the appropriate data node, or nodes cluster by double clicking a node.
- Multiple variables can be selected through the drawing of selection circles, whereby a point is selected by holding down the mouse and the size of the circle is determined by the distance the mouse is dragged.
- The selection rules can be modified by holding down the shift key while selecting variables to add them to the current selection, while the control key can be used to provide inverse selection in a similar fashion.
- Zooming can be performed through scrolling of the mouse wheel.
- Right clicking a node brings up a context menu allowing cluster reassignment.

3. K-Means Clustering

The K-Means algorithm (Alg. 1) is included to provided for both base and recursive clustering. The base algorithm works by choosing K randomly placed centroids which are iteratively moved until a local stable solution is found. The recursive algorithm works much like its parent, but continues to partition each cluster found until a single node remains within each sub-cluster. In effect this produces results similar to those of the Bond Energy Algorithm, but instead of producing a single central grouping, produces several independent groups determined by the value of K.

Algorithm 1 K-Means distance algorithm

```
1: for all nodes do
2:     for all centroids do
3:         centroidDistance ← calculateDistance(currentNode, currentCentroid)
4:         if centroidDistance ≤ smallestDistanceFound then
5:             smallestDistanceFound ← centroidDistance
6:             nearestCentroid ← currentCentroid
7:         end if
8:     end for
9:     assignNodeCluster(currentNode, nearestCentroid)
10: end for
```

After node assignment within clusters has been chosen for each time slice permutations of the cluster indexes are performed until the minimum number of node

cluster reassignments from one time slice to the next is found (Alg. 2). Once a minimum is found, a node stability can be calculated giving the user a numerical indication of how likely a node is to stay with any given cluster. Two different node stability metrics are provided, global and local. Global stability is the percentage of time a node has spent over the entire length of the data set within its most commonly associated cluster. The local metric works likewise, but is computed only over the currently viewed time span.

Algorithm 2 Minimum cluster reassignment algorithm

```
1: for all timeslice do
2:     for all clusters do
3:         for all clusters do
4:             clusterDifference ← clusterDifference+
                difference(outterCluster, innerCluster)
5:         end for
6:         if clusterDifference ≤ smallestClusterDifference then
7:             smallestDifferenceFound ← clusterDifference
8:             bestClusterPermutation ← currentClusterPermutation
9:         end if
10:    end for
11:    reassignClusterIndexes(bestClusterPermutation)
12: end for
```

4. K-Consistency

A variation of the K-Means distance formula (Alg. 3) is provided for more control of clustering over time. This is done through the use of an epsilon value which modifies the normal node-to-centroid distance. The epsilon returned within the algorithm is user configurable between 0.0 – 1.0. At a value of 0.0 this produces the same results as a non-modified version of K-Means. At a value of 1.0 clusters obtained in the first time slice are carried over for the entirety of the time span observed. This method provides a quick and easy way to observe long term relations that might otherwise be obscured by short term variations. A more advanced use of this feature can be obtained by providing it with negative values between -1.0 – 0.0 which causes items which were not associated in a previous time slice to be more likely to associate in a future time slice.

Algorithm 3 Modified K-Means distance algorithm (line 4 has been added)

1: **for all** *nodes* **do**

2: **for all** *centroids* **do**

3: $centroidDistance \leftarrow calculateDistance(currentNode, currentCentroid)$

4: $centroidDistance \leftarrow centroidDistance - getEpsilon(currentNode, currentCentroid)$

5: **if** $centroidDistance \leq smallestDistanceFound$ **then**

6: $smallestDistanceFound \leftarrow centroidDistance$

7: $nearestCentroid \leftarrow currentCentroid$

8: **end if**

9: **end for**

10: $assignNodeCluster(currentNode, nearestCentroid)$

11: **end for**

Chapter 4

VISUALIZATIONS

The application is developed using the QT framework and makes extensive use of widgets, thus allowing many varying visualizations to be viewed and positioned within a single window or over several screens.

1. Row / Column Swapping

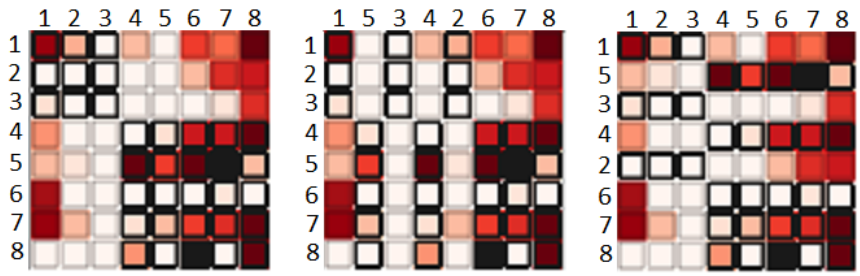


Figure 15: (left) A matrix with rows and columns in some original position. (middle) The first matrix with columns 2 and 5 swapped. (right) The first matrix with rows 2 and 5 swapped.

Rows and columns can be swapped using the mouse in a click-and-drag fashion. Two rows can be swapped by clicking and dragging in a vertical motion, while two columns can be swapped by performing a horizontal click and drag. Both columns and rows can be swapped at the same time by performing a horizontal and vertical

movement with the mouse before releasing. In practice this can be combined with the MDS visualization to select clusters in one time slice while displaying the cell values of another time slice. Rows and column ordering can then be adjusted within the resulting visualization to help associate data in a way showing a transition from one time slice to another.

2. Transformations

Several transformation functions are provided to preprocess the data set, as well as several modifying transformations. These functions and transformations can be used to present the data in a form that allows a better distribution of node values, and therefore colors, when a small number of extreme valued nodes might otherwise negatively bias the color distribution.

- Power - Transform each data point by x^y
- Log - Transform each data point by $log(x)/log(e)$
- Equal - No transformation is computed
- Normal - Each data point within a year is normalized between 1 and 0.
- Normalized Moving Average - Like Normal, but adjusted for moving average.
- Sequence Normalized - Each data point within a year is computed using (x - yearAverage) / Std, and then normalized between 1 and 0.

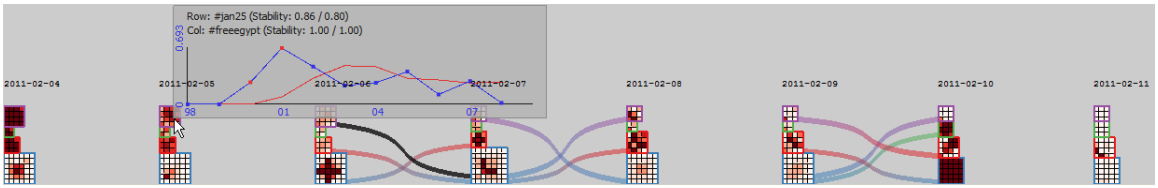


Figure 16: Normalized Example

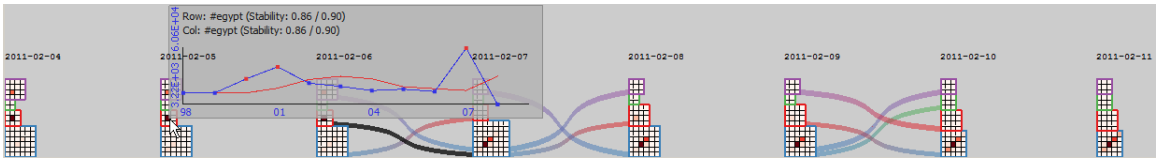


Figure 17: Absolute Example

3. Flood Fill

Figure 18: An example where the flood fill algorithm has been used to find a potential cluster of interest.

In large dense data sets it can sometimes be difficult to adequately differentiate cells. To aid in this purpose a mouse over flood fill (Figure 18) algorithm (Alg. 4) is provided which can be used to narrow down outlying values or find otherwise unseen clusters. In cases where use of the tool turns up a number of individual clusters within a cluster already determined by K-Means, a user may decide that another value of K for K-Means should be selected. Sensitivity of the algorithm is

determined by a configurable dissimilarity value, allowing smaller or larger clusters to be formed. The algorithm works by starting with a single node (chosen by mouse context) and then recursively compares each of its four direct neighbors for values within the dissimilarity range specified by the user. Each neighbor which passes this test then continues this process until no neighbors remain which pass the test.

Algorithm 4 Flood fill algorithm

```
1: if notAlreadyIncluded(currentNode) then
2:     if difference(currentNodeValue, previousNodeValue) ≤ dissimilarity
   then
3:         includedNodes ← currentNode
4:         compareNode(leftNode, currentNodeValue)
5:         compareNode(rightNode, currentNodeValue)
6:         compareNode(topNode, currentNodeValue)
7:         compareNode(bottomNode, currentNodeValue)
8:     end if
9: end if
```

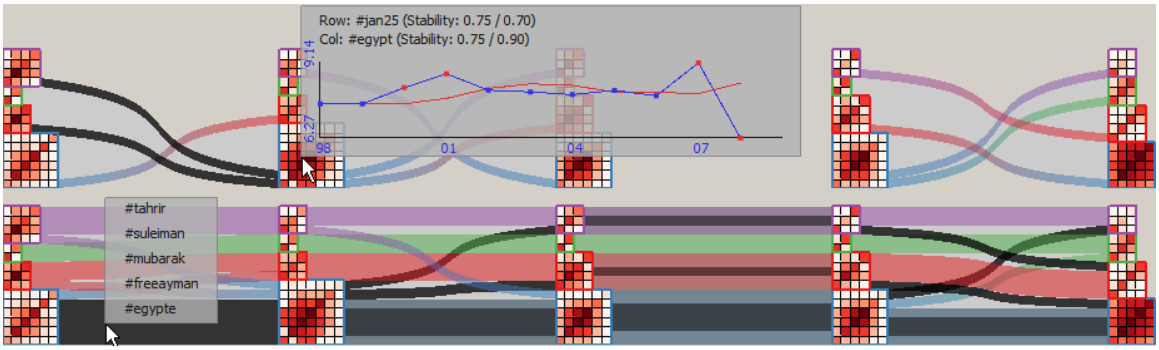


Figure 19: (top) Inspection of a cell and its two interacting nodes. (bottom) Content of a flow line, and its paths through the set.

The use of flows sets with interleaved matrix clusters is the culmination of our previous ideas. This allows the filtered visualization of the most relevant clustered data through a user determined number of time slices. This allows for inspection of individual cells, and the tracing of nodes through the time series (Fig. 19). This visualization in its current form does have some downsides. The largest problem with the visualization is the amount of screen space left unused between each time slice is quite a lot, and only only a small amount of vertical space is used compared to horizontal space. In practice, it helps to have more than one monitor when view larger data sets.

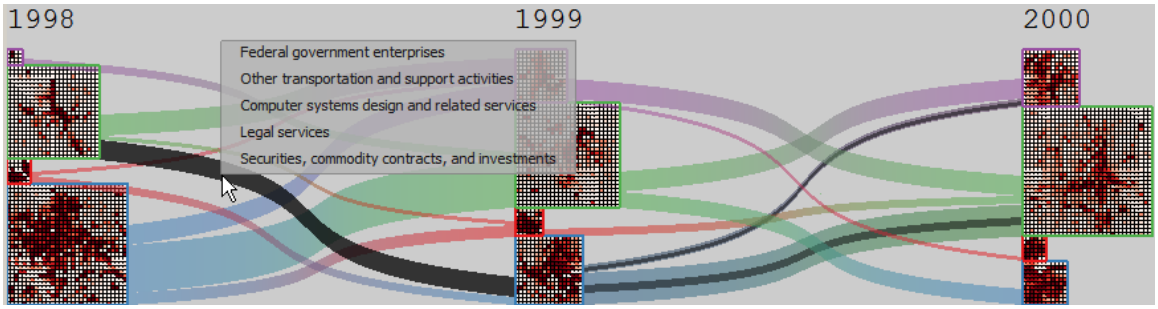


Figure 20: Example of a large matrix flow

5. Wordle Flow

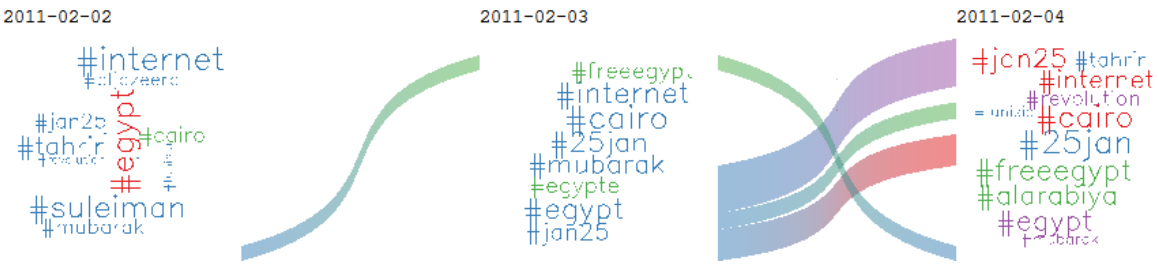


Figure 21: Wordle Experiment.

From the Flow Sets visualization was spawned the desire to better summarize the contents of each cluster. Due to the lack of available existing source code examples, an attempt to integrate an existing compiled library capable of generating Wordles was made. It was found however that most Wordle software is incapable of real time rendering of large data sets. To handle this, the rendering of the Wordles is offloaded to an Amazon Cloud application which streams in Wordle images as they become available. Ultimately while this works we felt a visualization such as this would be

useful for providing a quick overview of the data, in its current form it has several issues. Due to the nature of the flow lines, we were unable to find a reasonable way to associate them with the text placed in non-clustered positions, creating images of sufficient quality requires significant processing time, and finally non of the software available had the ability to cluster words. Alternative options such as the work done by Krstajic et al. [34] looks quite promising (fig. 24).

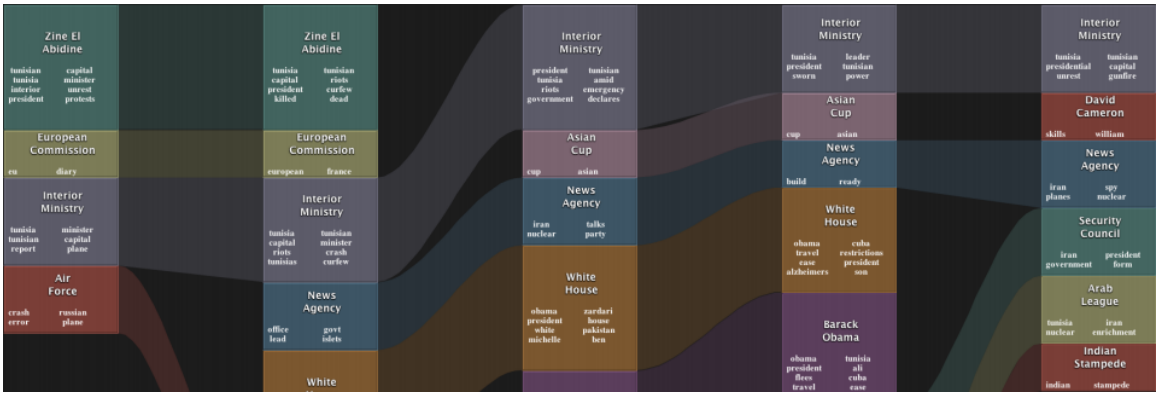


Figure 22: An example of the incremental text visualization.

Source: Incremental visual text analytics of news story development [34]

6. Cluster Labels

Due in large part to the work done by Krstajic et al. [34] addition of simple cluster labeling was added to the application. As shown in figure 23, this was done by replacing through a toggle function, the matrices normally shown within each time slice with text labels. In order to support this, each node loaded by the application must provide at least one label. Labels are then sorted by most common occurrence

within each cluster. A minimum of one label is rendered per a cluster, and the cluster enlarged if necessary to allow ample space. In practice this feature makes good use of screen real estate, as typically the larger the cluster the more labels it will contain.

In some cases where a data set containing a large number of time slices is used or where abnormally small clusters form, these clusters will expand to fit the required minimum single label. This expansion is a trade-off necessary to allow for text sizes significant enough to be readable, but can cause small disconnects in cluster size compared to flow lines which can mislead a user into believing a clusters relative size is larger than it actually is.

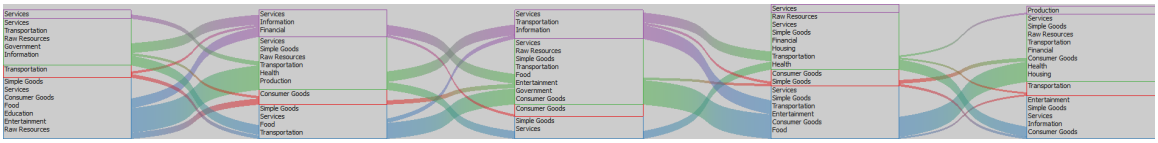


Figure 23: An example use of cluster labels.

7. Aesthetic Experimentation

A great deal of experimentation with aesthetic choices were made during the development of the application, and while many were abandoned, several had enough merit to be worth mentioning. Among them were ideas such as animation, fix or variable size time slices, and variations in pixel matrix visualization.

As a carry over from earlier versions of the application was the use of animation. While this feature was nice for presentation purposes, it was eventually abandoned due to both the code complexity and difficulty of use in larger data sets.

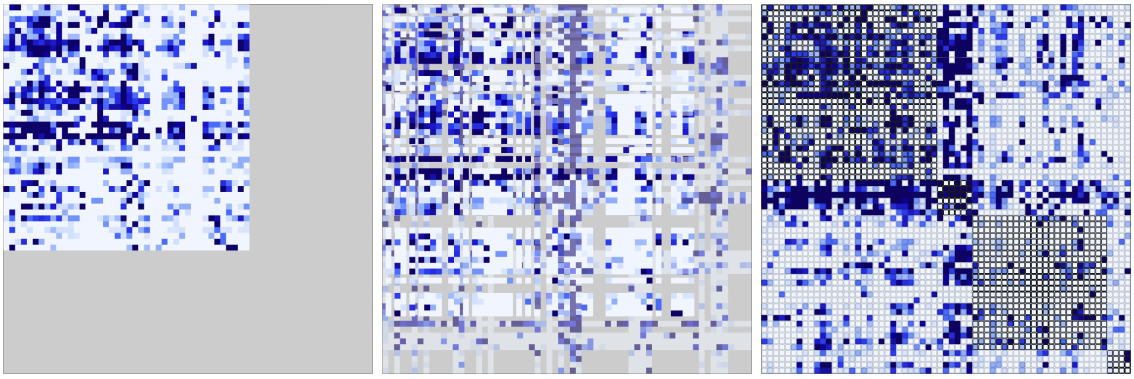


Figure 24: An example use of animation.

Fixed sized time slices were originally used to prevent flow line overlap and simplify presentation, but was replaced with a variable width system. Ultimately fixed width time slices were reintroduced with the addition of cluster labeling.

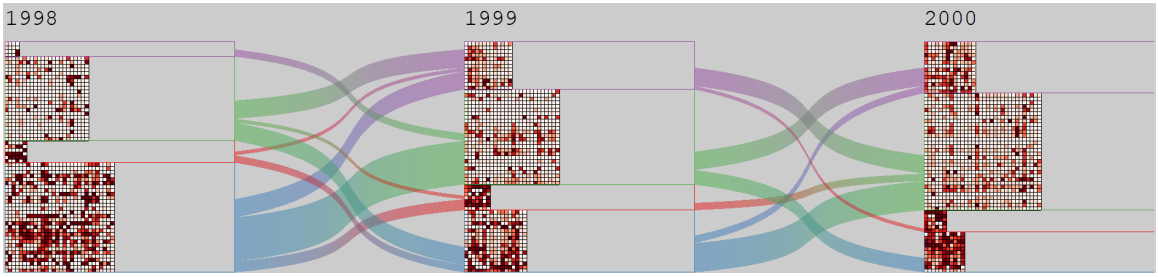


Figure 25: Fixed width time slices.

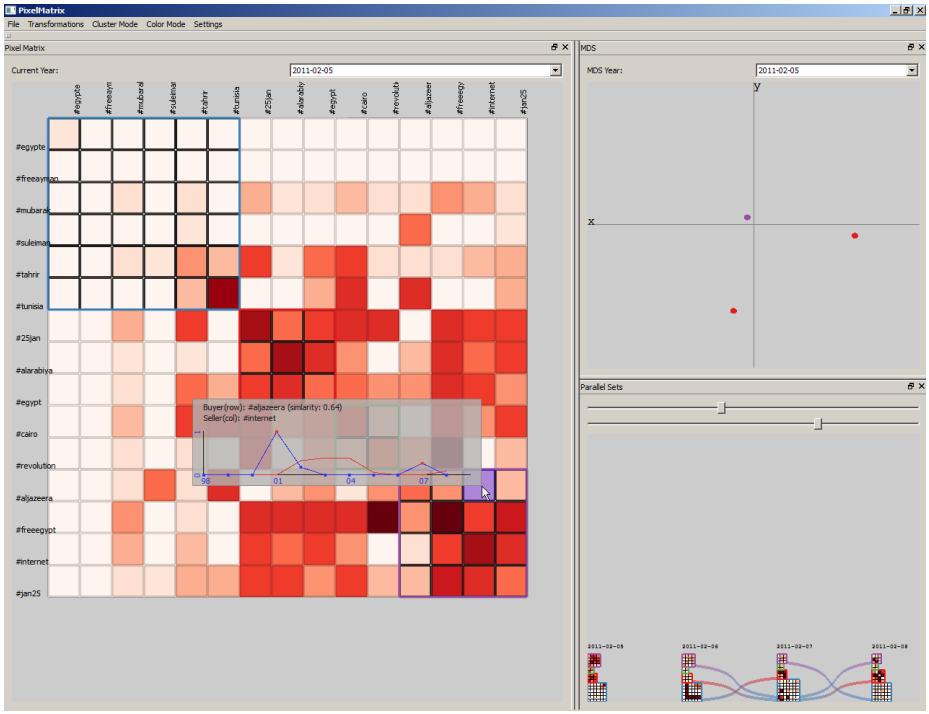


Figure 26: Full Display

Data modified or selected within one visualization is reflected within the others. For example, a user can pull up the MDS widget, and select a year while the pixel matrix widget is also open, but has a different year selected. This will cause the clustering displayed within the pixel matrix widget to be determined by the year selected by the MDS widget, but the values displayed for each node to be displayed based on the year selected within the pixel matrix widget (fig. 26). Similarly, a user can pull up the MDS widget, select a year of interest, hold shift, and double click both a purple and green node. This in turn will filter the Parallel Sets visualization

to the nodes in those clusters in that year so that only those nodes are visible over the entire time series (fig. 27).

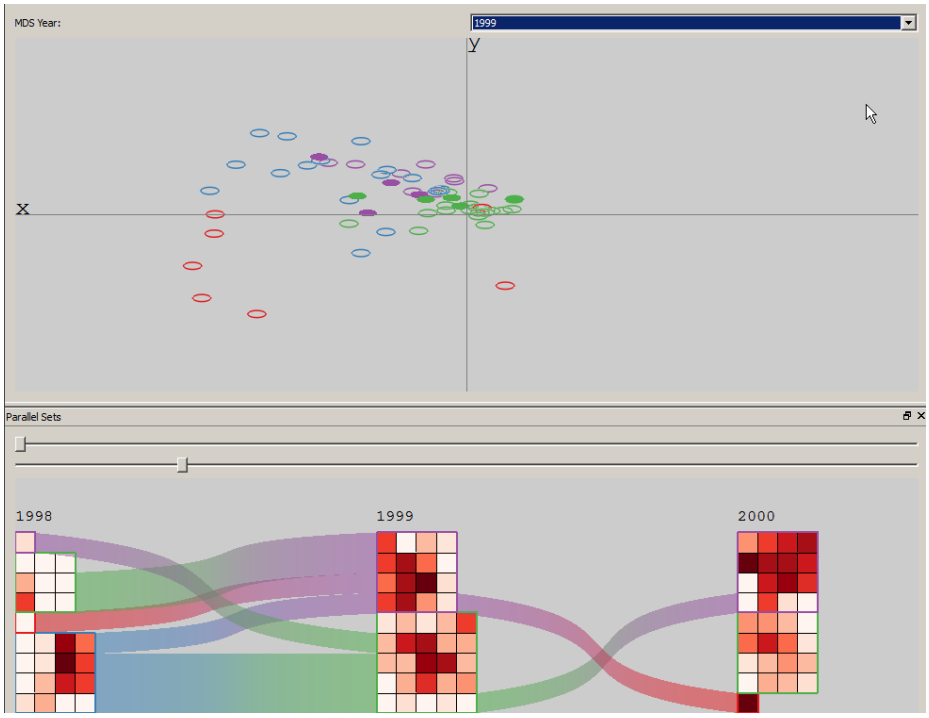


Figure 27: A good example of filtering on year cluster over a flow

Chapter 5

RESULTS

Our application and ideas were intended to support general case data and tested against real-world data sets of varying physical properties. Data tested included interaction within social media, trade of material goods, stock, and even the interaction of characters within books over chapters, consisting of binary, discrete, continuous, directed, and clustered data. Of the data tested, two case studies were focused on due to their significance. Our performance was tailored against a machine using an Intel Core i5 quad-core 3.40GHz CPU and 16GB of memory. Graphic hardware requirements are negligible; however, there is a noticeable large-multiscreen usability advantage.

1. Case Study - Industry Data Set

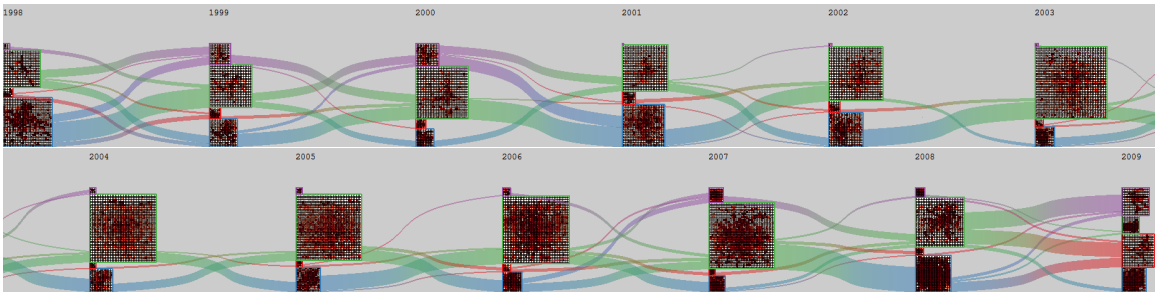


Figure 28: Initial full industry time series

Our first case study focuses on industry data provided by The U.S. Department of Commerce: Bureau of Economic Analysis, from the year 1998 and 2009. This data set contains the trade value between each of 36 industries. The data set in this example has been provided in annual time slices, and so that will be the option chosen when loading the data. After the data has been loaded a good first step is to choose a reasonable clustering method and transformation. In this case, using the Bond Energy Algorithm – Multidimensional Scaling clustering option with the logarithmic transformation creating a nice initial visualization in the Parallel Sets view (fig. 28).

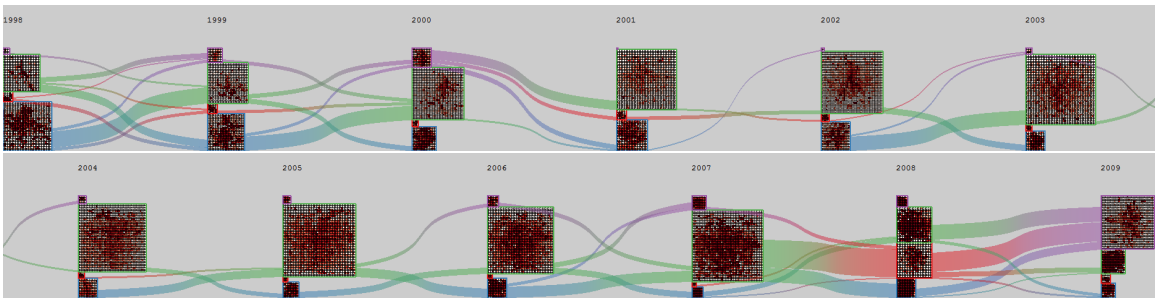


Figure 29: Full industry time series with K-Means Epsilon set to 0.15

As is visible in the initial view (and with more detail available in context selection) there is quite a bit of swapping back and forth of industries between clusters. This can be useful when wanting to see very detailed information between individual time steps, but can be distract from larger long term trends. Using the K-Means Epsilon feature with a value of 0.15 provides us with a more stable visualization to find these trends (fig. 29). After this change a noticeable difference in cluster movement between years becomes apparent. Between the years 1998 and 2007 there's relatively

little change compared to the changes that occurred in 2008 and 2009. Given a larger data set with more precise time slices, we could test if sudden cluster instability could predict market instability.

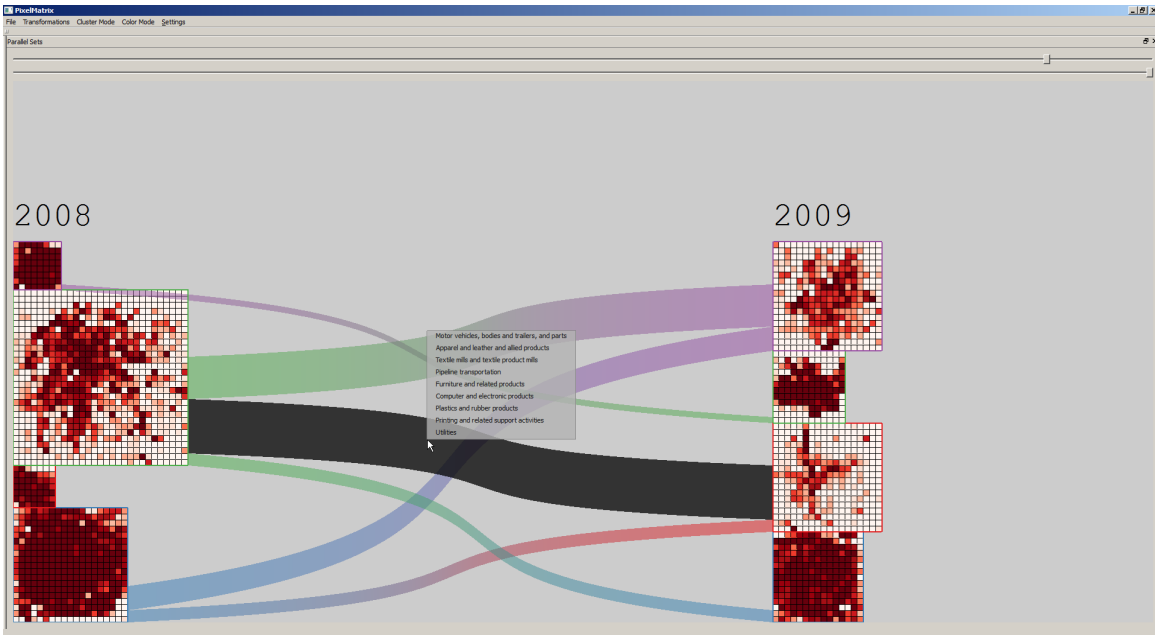


Figure 30: The movement of non-essential consumer goods from the services cluster into then also non-essential raw goods cluster between 2008 and 2009.

Due to the time span of the data provided, of specific interest in this data is the real estate industry and its relation with other industries before and during the house market crash. Real estate begins in a cluster with several industries that one might expect, the federal reserve, construction, accommodation, insurance, and many raw good industries, such as wood products, primary metals, non-metallic mineral products, chemicals, etc. Using the flow lines as a guide, this shows that the real

estate industry stays within its original cluster for some time, but eventually leaves this cluster for a more investment centric cluster in 2005. In 2001 the investment and trust industries join this new cluster, but don't produce any significance increase in trade until 2005. With a lower epsilon value selection, even with the large absolute numbers being invested in real estate in 2005 the real estate industry enters a new cluster with the transportation, raw good industries, and construction. From 2005 until 2007 the investment and trust industries purchase a large amount of real estate and related industry goods, with the trust industry typically mirroring investments with a slight delay. In 2008, the investment industry followed less so by the trust industry drops in trade with real estate and related construction and raw resource industries. Perhaps counter intuitively by 2009, the investment industry is already again purchasing real estate, but not the related construction industries; as well as the more obvious rental and government industries. Depending on the epsilon value used, roughly four stable clusters develop by the end of the time series:

- Low intensity trade among construction, real estate, and related industries
- Mild intensity trade among service industries such as food, waste management, and health industries that change little in trade over the length of the time series
- Very low intensity trade among consumer goods, transportation, and raw goods industries

- High intensity trade among Rental, Insurance, Government, and entertainment industries

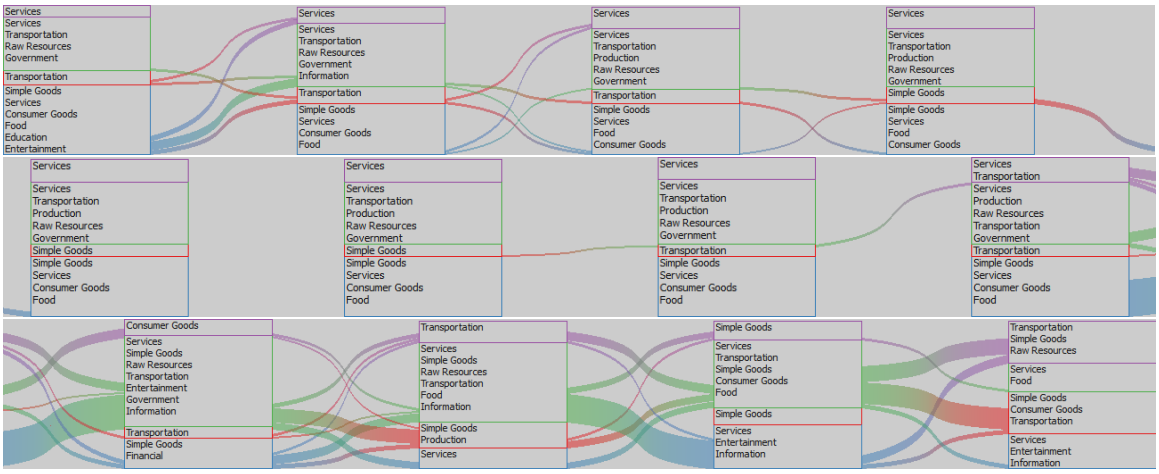


Figure 31: Industry cluster labels.

Of interest is the large change in trade of non-essential raw materials and consumer goods which quickly group together during this time (fig. 31). Due to the relatively stable nature of the data, flow lines, and epsilon adjustments, a small number of changes in clusters can easily be followed over a long period of time. In the next example a less stable and more sparse data set requires a more detailed analysis.

2. Case Study - Twitter Data Set

Our second case study focuses on Twitter data provided by Huan Liu DMML Lab. This data covers the Egyptian Revolution of 2011 through the tweets generated from February 2nd to the 11th. A preprocessing application was used to draw a

correlation between all hashtags tweeted by a user within a given time slice. For example, if a user tweeted about #freeegypte, #revolution, and #25jan within a single time slice, then those hashtags would then be more strongly correlated. Due to the short nature of the Egyptian Revolution daily time slices provide the more interesting results.

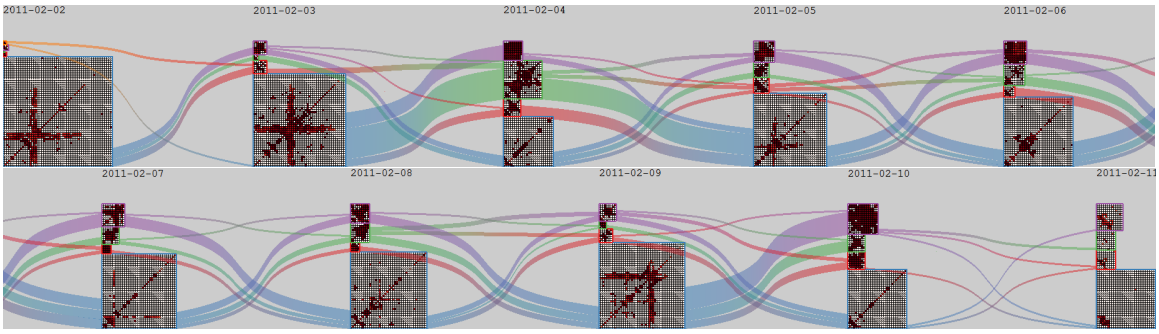


Figure 32: View after selecting a reasonable K for K-Means.

The flow set view is good place to start, due to the view over the entire time set. Using this view we can narrow down a good cluster count for K-Means. Switching the cluster mode to BEA-MDS can aid in this as additional potential clusters can be found causing more than a single clustered group of pixels within a single K-Means cluster - for example in figure 33. In this case using four clusters produces reasonable results over most of the time series, with several strong clusters forming in figure 32.

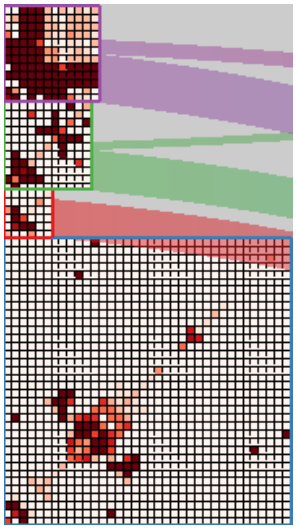


Figure 33: Possible additional cluster example. (notice the secondary groupings in the bottom and second from the top clusters)

This is usually a good time to decide if any nodes should be moved from one cluster to another. Reason for this could be found after inspecting the MDS view of a single time slice or due to the nature of movement of nodes in the Flow Sets view and the desire to better tell a story given the data. For example, in figure 35 it might make more sense to place the Tunisia node in the cluster with Algeria. Bertin discussed in his work the idea that people have an intrinsic idea of how data should be organized based on the context, a context which the algorithms used to make the initial clustering might not be aware of [24].

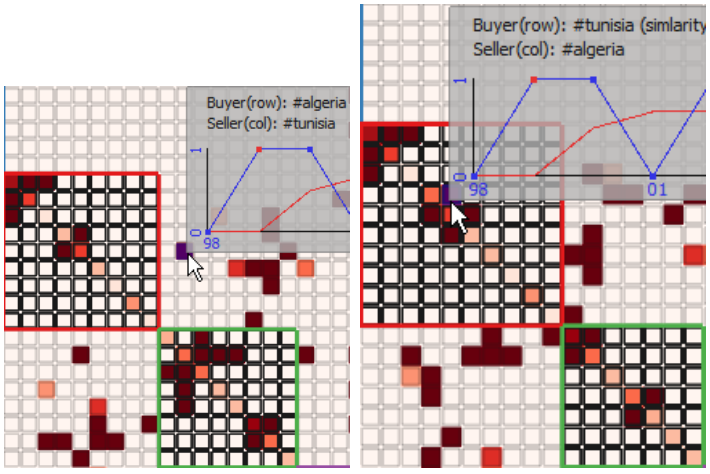


Figure 34: (left) Algeria and Tunisia in original cluster. (right) Algeria and Tunisia in new cluster.

Taking a quick scan of the cluster contents over the times series shows the following consistent clusters.

- #freeegypte - #internet
- #cairo - #revolution
- #alarabiya - #25jan
- #mubarak - #suleiman

Figure 35: Twitter data visualized with cluster labels.

Of interest is that this data set includes both #alarabiya and #aljazeera, but while #alarabiya remained stable (fig. 36), #aljazeera (fig. 36) appears to move consistently with the strongest clusters of the time. This can be verified by mousing over each node and viewing either the stability metric provided next to the node name, or by mousing over each node and through the following of flow lines through the time series. A quick search online shows that of these two news sources Alarabiya has more stringent controls placed on it than Alarabiya. One hypothesis could be that due to those constraints Alarabiya will consistently lag behind Aljazeera in covering the most up-to-date news. Something that with future data could likely be tested.

Also of interest are not just relationships, but also short lived hashtags such as #freeayman which appeared on February 6th, but then quickly disappeared after Ayman's release in the following hours. In the data, this surges between Feb. 4-5th and on Feb. 10th, which is due to the differences in time zones that may not directly correlate to significant dates recorded in Egypt. This hashtag was found not because of any strong relation with a single other hashtag, but because of its general

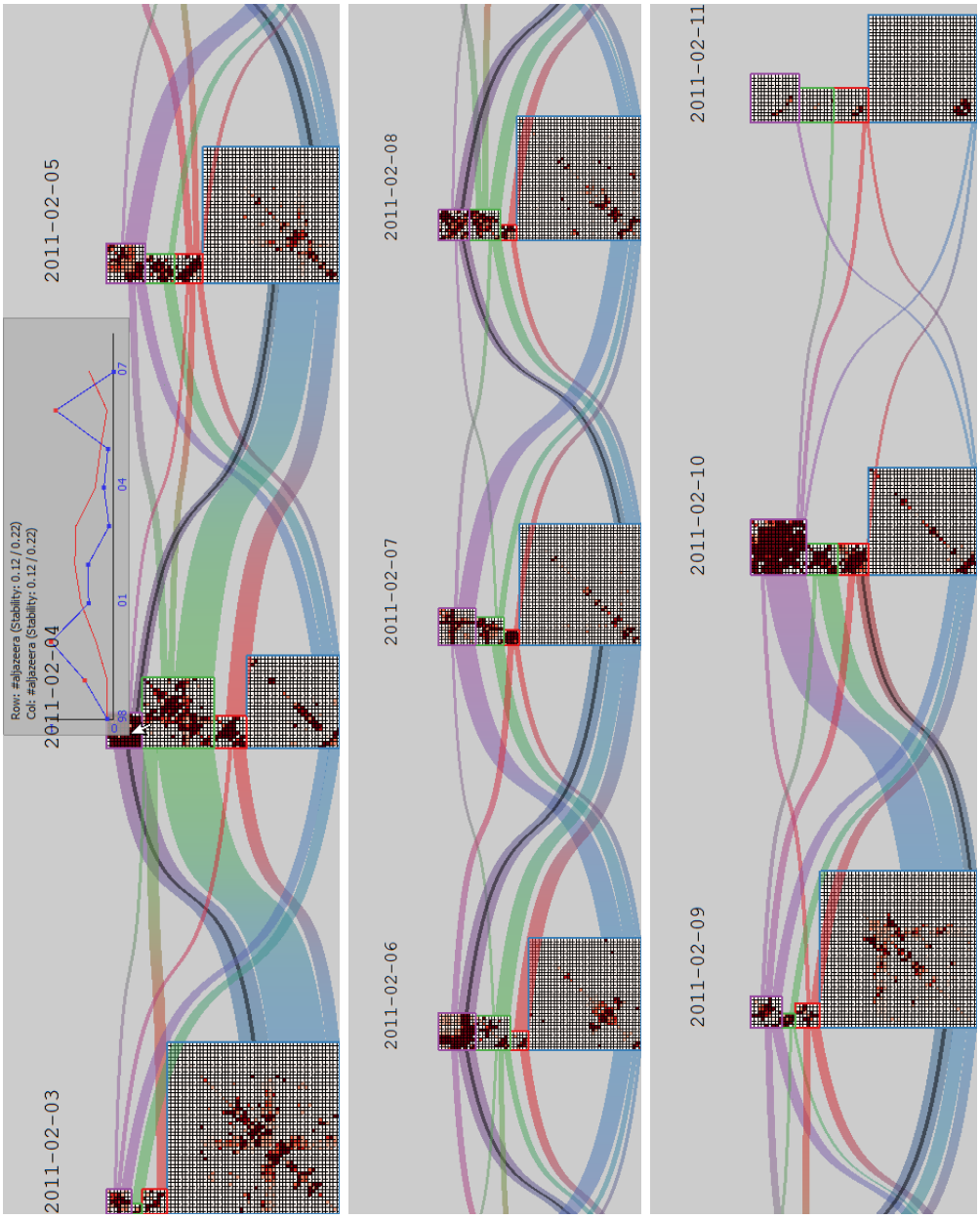


Figure 36: #Aljazeera

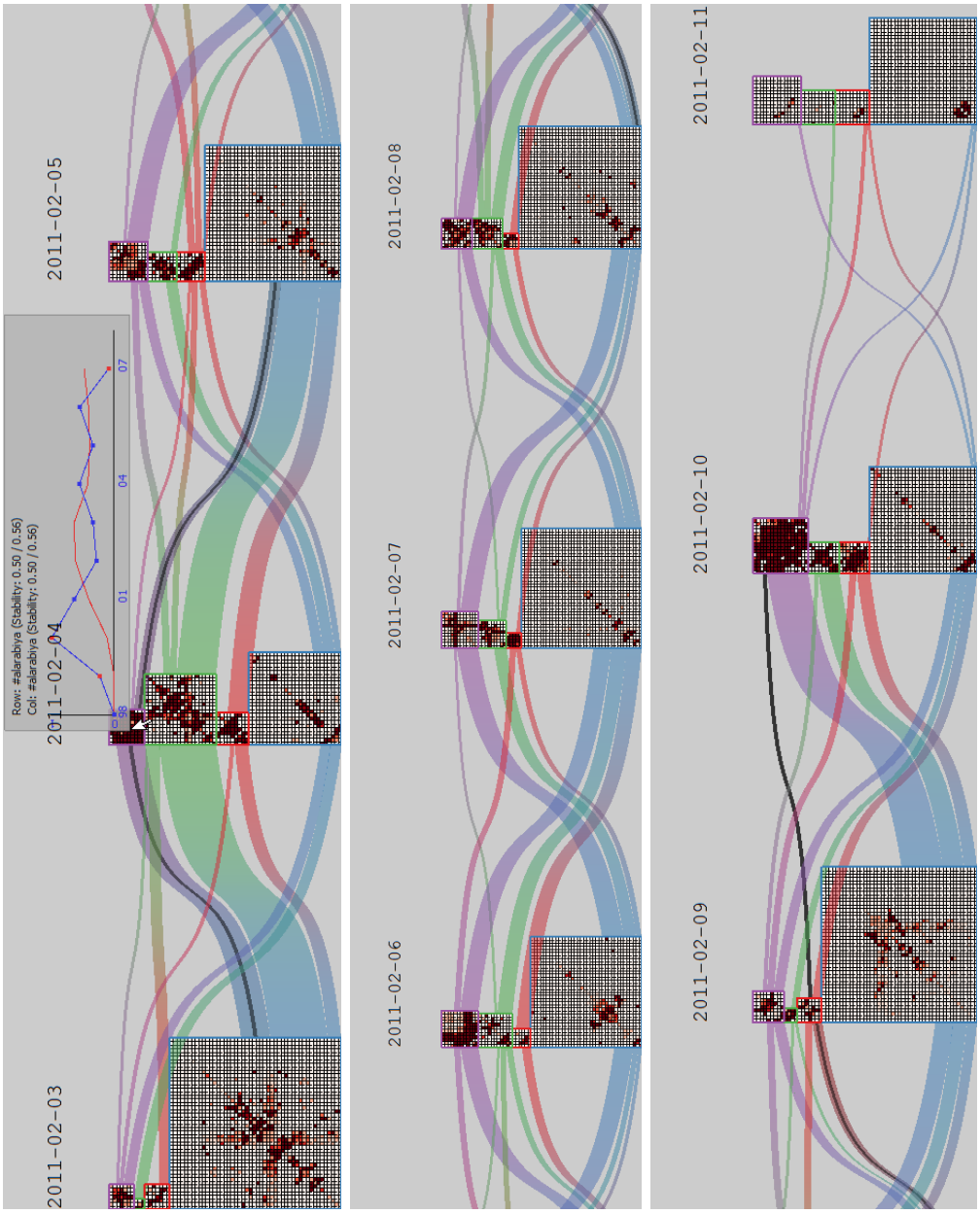


Figure 37: #Alarabiya

relation to the social group using all of hashtags popular within the time period. With additional data it may be possible to test if an application like this can spot social group specific hot topic hashtags such as this before a top ten list or something equally naive could.

Several difficulties are visible with this data set. The blue cluster at the bottom of the visualization is overwhelmingly large throughout the time series due to the number of nodes which are short lived that end up getting "abandoned" within this cluster.

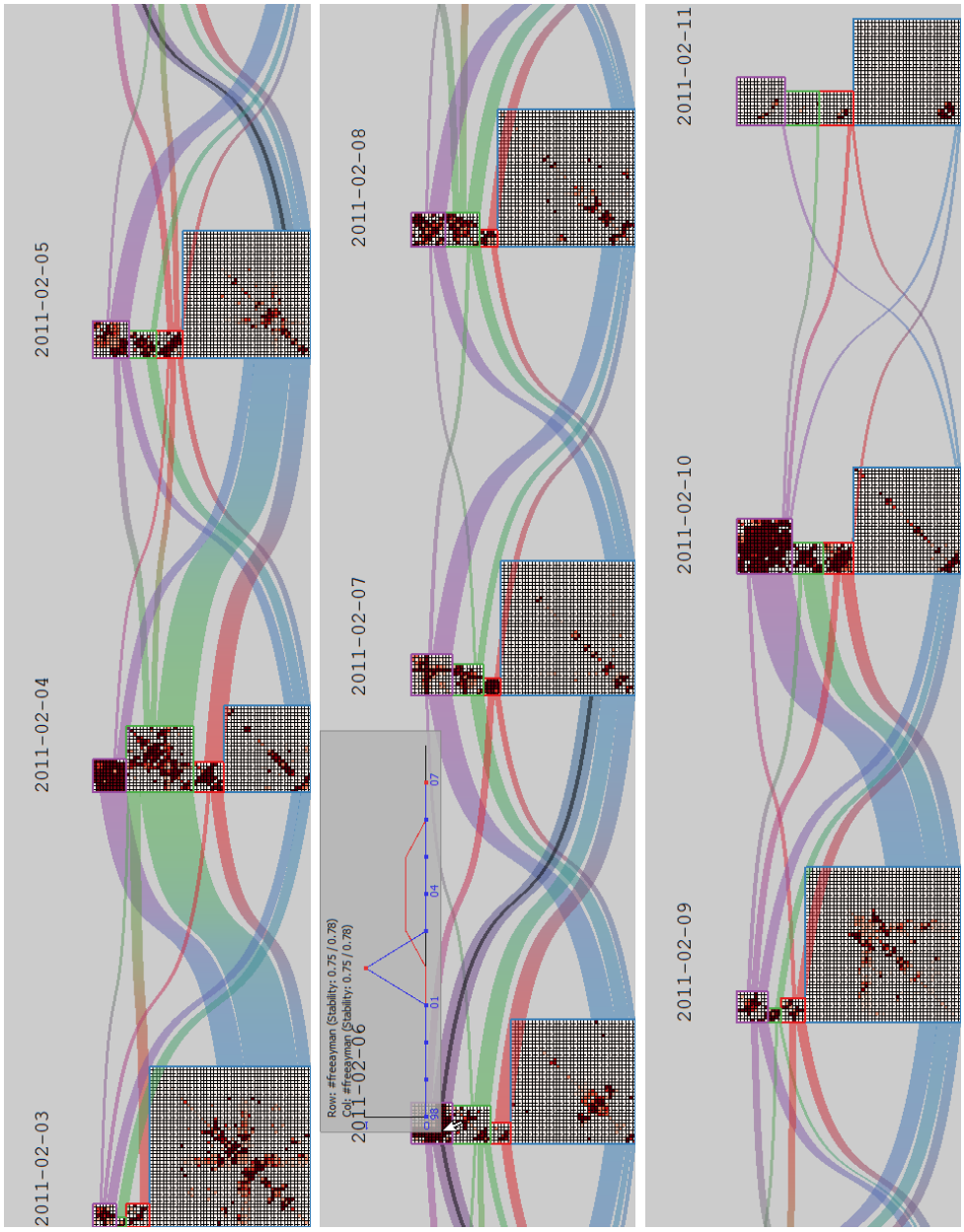


Figure 38: #Freeayman

Chapter 6

CONCLUSION AND FUTURE WORK

We have shown in our case studies how our method of combining both a pixel matrix and parallel set with more modern techniques can aid in more quickly discovering information which previously may have required cross checking multiple visualizations. This process especially works well when searching for insight into the flow and altering of significant clusters through time, however this process currently still suffers from the curse of dimensionality. Additional research into zooming and filtering support within each time slice cluster should minimize this outstanding issue. Automated cluster labeling such a that used by Krstajic et al. [34] could likely replace the need for the Wordle Flow visualization in a more useful fashion.

REFERENCES

[1] J. Kruskal and M. Wish, *Multidimensional Scaling*, ser. 07. SAGE Publications, 1978, no. no. 11. [Online]. Available: http://books.google.com/books?id=ZzmIPcEXPf0C

[2] M. Ben-Menachem, "Parallel coordinates: Visual multidimensional geometry and its applications, is written by alfred inselberg, and published by springer; (c) 2009; isbn 978-0-387-21507-5; pp. 580." *ACM SIGSOFT Software Engineering Notes*, vol. 35, no. 3, p. 39, 2010. [Online]. Available: http://dblp.uni-trier.de/db/journals/sigsoft/sigsoft35.html\#Ben-Menachem10

[3] H. Zhou, X. Yuan, H. Qu, W. Cui, and B. Chen, "Visual clustering in parallel coordinates," *Computer Graphics Forum*, vol. 27, no. 3, pp. 1047–1054, 2008. [Online]. Available: http://dx.doi.org/10.1111/j.1467-8659.2008.01241.x

[4] A. Pilhöfer and A. Unwin, "New approaches in visualization of categorical data: R package extracat," *Journal of Statistical Software*, vol. 53, no. 7, pp. 1–25, 2013. [Online]. Available: http://www.jstatsoft.org/v53/i07/

[5] R. Kosara, F. Bendix, and H. Hauser, "Parallel sets: Interactive exploration and visual analysis of categorical data," *Transactions on Visualization and Computer Graphics*, vol. 12, pp. 558–568, 2006.

[6] N. Henry, J. Fekete, and M. McGuffin, "Nodetrix: a hybrid visualization of social networks," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 13, no. 6, pp. 1302–1309, Nov 2007.

[7] J. Heinrich, Y. Luo, A. E. Kirkpatrick, H. Zhang, and D. Weiskopf, "Evaluation

of a bundling technique for parallel coordinates," *CoRR*, vol. abs/1109.6073, 2011.

[8] A. Wallgren, *Graphing statistics & data: creating better charts*. Sage Publications, 1996. [Online]. Available: http://books.google.com/books?id=tLzrAAAAMAAJ

[9] J. Rodrigues, J.F., A. J. M. Traina, and A. Traina, "Frequency plot and relevance plot to enhance visual data exploration," in *Computer Graphics and Image Processing, 2003. SIBGRAPI 2003. XVI Brazilian Symposium on*, Oct 2003, pp. 117–124.

[10] K. Xu, C. Rooney, P. Passmore, D.-H. Ham, and P. Nguyen, "A user study on curved edges in graph visualization," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 12, pp. 2449–2456, Dec 2012.

[11] J. Talbot, J. Gerth, and P. Hanrahan, "An empirical model of slope ratio comparisons," *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2012. [Online]. Available: http://vis.stanford.edu/papers/slope-ratio-comparison

[12] Y.-H. Fua, M. Ward, and E. Rundensteiner, "Hierarchical parallel coordinates for exploration of large datasets," in *Visualization '99. Proceedings*, Oct 1999, pp. 43–508.

[13] H. Zhou, P. Xu, X. Yuan, and H. Qu, "Edge bundling in information visualization," *Tsinghua Science and Technology*, vol. 18, no. 2, pp. 145–156, April 2013.

[14] T. Boogaerts, L.-C. Tranchevent, G. A. Pavlopoulos, J. Aerts, and J. Vandewalle, "Visualizing high dimensional datasets using parallel coordinates: Application to gene prioritization." in *BIBE*. IEEE Computer Society, 2012, pp. 52–57. [Online]. Available: http://dblp.uni-trier.de/db/conf/bibe/bibe2012.html\#BoogaertsTPAV12

[15] R. M. Edsall, "The parallel coordinate plot in action: Design and use for geographic visualization," *Comput. Stat. Data Anal.*, vol. 43, no. 4, pp. 605–619, Aug. 2003. [Online]. Available: http://dx.doi.org/10.1016/S0167-9473(02)00295-5

[16] G. Andrienko and N. Andrienko, "Exploring spatial data with dominant attribute map and parallel coordinates," *Computers, Environment and Urban Systems*, vol. 25, no. 1, pp. 5–15, 2001.

[17] M. Butkiewicz, T. Butkiewicz, W. Ribarsky, and R. Chang, "Integrating time-series visualizations within parallel coordinates for exploratory analysis of incident databases," in *SPIE Defense, Security, and Sensing*, vol. 7346, 2009, pp. 73 460C–73 460C–8. [Online]. Available: http://dx.doi.org/10.1117/12.818840

[18] M. Schonlau, "Visualizing categorical data arising in the health sciences using hammock plots," in *Statistical Graphics, American Statistical Association, 2003.*, 2003.

[19] S. Few, "Multivariate analysis using parallel coordinates," 2006. [Online]. Available: http://www.perceptualedge.com/articles/b-eye/parallel_coordinates.pdf

[20] H. Siirtola, "Combining parallel coordinates with the reorderable matrix," in *Proceedings of the conference on Coordinated and Multiple Views In Exploratory Visualization*, ser. CMV '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 63–. [Online]. Available: http://dl.acm.org/citation.cfm?id=937938.937951

[21] D. Oelke, H. Janetzko, S. Simon, K. Neuhaus, and D. A. Keim, "Visual boosting in pixel-based visualizations," in *Proceedings of the 13th Eurographics / IEEE - VGTC Conference on Visualization*, ser. EuroVis'11. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2011, pp. 871–880. [Online]. Available: http://dx.doi.org/10.1111/j.1467-8659.2011.01936.x

[22] J. S. Yi, N. Elmqvist, and S. Lee, "Timematrix: Analyzing temporal social networks using interactive matrix-based visualizations," *Intl. Journal of Human–Computer Interaction*, vol. 26, no. 11-12, pp. 1031–1051, 2010.

[23] W. Brinton, *Graphic Methods for Presenting Facts*, ser. Industrial management Library. Engineering Magazine Company, 1914. [Online]. Available: http://books.google.com/books?id=LbQYAAAAMAAJ

[24] J. Bertin, *Semiology of graphics*. University of Wisconsin Press, 1983.

[25] W. T. M. Jr., P. J. Schweitzer, and T. W. White, "Problem decomposition and data reorganization by a clustering technique," *Operations Research*, vol. 20, no. 5, pp. pp. 993–1009, 1972. [Online]. Available: http://www.jstor.org/stable/169162

[26] A. Kusiak and W. Chow, "An efficient cluster identification algorithm," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. 17, no. 4, pp. 696–699, July 1987.

[27] L. Wilkinson and M. Friendly, "The history of the cluster heat map," *The American Statistician*, vol. 63, no. 2, pp. 179–184, 2009. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1198/tas.2009.0033

[28] I. Liiv, "Seriation and matrix reordering methods: An historical overview," *Statistical Analysis and Data Mining*, vol. 3, no. 2, pp. 70–91, 2010. [Online]. Available: http://dx.doi.org/10.1002/sam.10071

[29] L. Euler, "Seven bridges of königsberg," *Commentarii academiae scientiarum Petropolitanae*, 1736.

[30] M. Ghoniem, J.-D. Fekete, and P. Castagliola, "On the readability of graphs using node-link and matrix-based representations: A controlled experiment and statistical analysis," *Information Visualization*, vol. 4, no. 2, pp. 114–135, Jul. 2005. [Online]. Available: http://dx.doi.org/10.1057/palgrave.ivs.9500092

[31] M. Ghoniem, , M. Ghoniem, N. Jussien, and R. A. Kastler, "Visexp: visualizing constraint solver dynamics using explanations," in *in FLAIRS'04: Seventeenth international Florida Artificial Intelligence Research Society conference, (Miami Beach, FL, 2004), AAAI.* press, 2004.

[32] N. Henry and J. daniel Fekete, "Matlink: Enhanced matrix visualization for analyzing social networks," in *Proceedings of the International Conference Interact*, 2007, pp. 288–302.

[33] J. Abello and F. van Ham, "Matrix zoom: A visual interface to semi-external graphs," in *Proceedings of the IEEE Symposium on Information Visualization*, ser. INFOVIS '04. Washington, DC, USA: IEEE Computer Society, 2004, pp. 183–190. [Online]. Available: http://dx.doi.org/10.1109/INFOVIS.2004.46

[34] M. Krstajic, M. Najm-Araghi, F. Mansmann, and D. A. Keim, "Story tracker: Incremental visual text analytics of news story development." *Information Visualization*, vol. 12, no. 3-4, pp. 308–323, 2013. [Online]. Available: http://dblp.uni-trier.de/db/journals/ivs/ivs12.html\#KrstajicNMK13

[35] J. Díaz, J. Petit, and M. Serna, "A survey of graph layout problems," *ACM Comput. Surv.*, vol. 34, no. 3, pp. 313–356, Sep. 2002. [Online]. Available: http://doi.acm.org/10.1145/568522.568523

[36] H. Purchase, "The effects of graph layout," in *Computer Human Interaction Conference, 1998. Proceedings. 1998 Australasian*, Nov 1998, pp. 80–86.

[37] W. Huang, P. Eades, S.-H. Hong, and C.-C. Lin, "Improving force-directed graph drawings by making compromises between aesthetics," in *Visual Languages and Human-Centric Computing (VL/HCC), 2010 IEEE Symposium on*, Sept 2010, pp. 176–183.

[38] H. Purchase, C. Pilcher, and B. Plimmer, "Graph drawing aesthetics: Created by users, not algorithms," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 18, no. 1, pp. 81–92, Jan 2012.

[39] W. McCormick, *Identification of Data Structures and Relationships by Matrix Reordering Techniques*, ser. Research paper. Institute for Defense Analyses, Systems Evaluation Division, 1969. [Online]. Available: http://books.google.com/books?id=HU_SYgEACAAJ

[40] S. Climer and W. Zhang, "Rearrangement clustering: Pitfalls, remedies, and applications," *J. Mach. Learn. Res.*, vol. 7, pp. 919–943, Dec. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1248547.1248579

[41] H. Siirtola, "Interaction with the reorderable matrix," in *Information Visualization, 1999. Proceedings. 1999 IEEE International Conference on*, 1999, pp. 272–277.

[42] M. Hao, U. Dayal, D. Keim, and T. Schreck, "Multi-resolution techniques for visual exploration of large time-series data," in *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization*, ser. EUROVIS'07.

Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2007, pp. 27–34. [Online]. Available: http://dx.doi.org/10.2312/VisSym/EuroVis07/027-034

[43] S. Ko, R. Maciejewski, Y. Jang, and D. S. Ebert, "Marketanalyzer: An interactive visual analytics system for analyzing competitive advantage using point of sale data," *Comp. Graph. Forum*, vol. 31, no. 3pt3, pp. 1245–1254, Jun. 2012. [Online]. Available: http://dx.doi.org/10.1111/j.1467-8659.2012.03117.x

[44] I. Hur and J. Yi, "Simulsort: Multivariate data exploration through an enhanced sorting technique," in *Human-Computer Interaction. Novel Interaction Methods and Techniques*, ser. Lecture Notes in Computer Science, J. Jacko, Ed. Springer Berlin Heidelberg, 2009, vol. 5611, pp. 684–693. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02577-8_75

[45] R. Maciejewski, S. Rudolph, R. Hafen, A. Abusalah, M. Yakout, M. Ouzzani, W. Cleveland, S. Grannis, M. Wade, and D. Ebert, "Understanding syndromic hotspots - a visual analytics approach," in *Visual Analytics Science and Technology, 2008. VAST '08. IEEE Symposium on*, Oct 2008, pp. 35–42.

[46] I. Boyandin, E. Bertini, and D. Lalanne, "A qualitative study on the exploration of temporal changes in flow maps with animation and small-multiples," *Comp. Graph. Forum*, vol. 31, no. 3pt2, pp. 1005–1014, Jun. 2012. [Online]. Available: http://dx.doi.org/10.1111/j.1467-8659.2012.03093.x

[47] J. Heer and G. Robertson, "Animated transitions in statistical data graphics," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1240–1247, Nov. 2007. [Online]. Available: http://dx.doi.org/10.1109/TVCG.2007.70539

[48] I. Boyandin, E. Bertini, P. Bak, and D. Lalanne, "Flowstrates: An approach for visual exploration of temporal origin-destination data," *Computer Graphics Forum*, vol. 30, no. 3, pp. 971–980, 2011. [Online]. Available: http://dx.doi.org/10.1111/j.1467-8659.2011.01946.x

[49] S. Anders, "Visualization of genomic data with the hilbert curve," *Bioinformatics*, vol. 25, no. 10, pp. 1231–1235, 2009. [Online]. Available: http://bioinformatics.oxfordjournals.org/content/25/10/1231.abstract

[50] D. Oelke, H. Janetzko, S. Simon, K. Neuhaus, and D. A. Keim, "Visual boosting in pixel-based visualizations," *Computer Graphics Forum*, vol. 30, no. 3, pp. 871–880, 2011. [Online]. Available: http://dx.doi.org/10.1111/j.1467-8659.2011.01936.x

[51] M. Stone, "Choosing colors for data visualization," 2006. [Online]. Available: http://www.perceptualedge.com/articles/b-eye/choosing_colors.pdf

[52] M. Ankerst, S. Berchtold, and D. A. Keim, "Similarity clustering of dimensions for an enhanced visualization of multidimensional data," in *Proceedings of the 1998 IEEE Symposium on Information Visualization*, ser. INFOVIS '98. Washington, DC, USA: IEEE Computer Society, 1998, pp. 52–. [Online]. Available: http://dl.acm.org/citation.cfm?id=647341.721216

[53] E. Mkinen and H. Siirtola, "The barycenter heuristic and the reorderable matrix," 2005.

[54] S. Ingram, T. Munzner, V. Irvine, M. Tory, S. Bergner, and T. Mo andller, "Dimstiller: Workflows for dimensional analysis and reduction," in *Visual Analytics Science and Technology (VAST), 2010 IEEE Symposium on*, oct. 2010, pp. 3 –10.

[55] E. Bertini, A. Tatu, and D. Keim, "Quality metrics in high-dimensional data visualization: An overview and systematization," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 17, no. 12, pp. 2203 –2212, dec. 2011.

[56] M. Sedlmair, A. Tatu, T. Munzner, and M. Tory, "A taxonomy of visual cluster separation factors," *Computer Graphics Forum*, vol. 31, no. 3pt4, pp. 1335–1344, 2012. [Online]. Available: http://dx.doi.org/10.1111/j.1467-8659.2012.03125.x

[57] H. Ziegler, T. Nietzschmann, and D. Keim, "Visual analytics on the financial market: Pixel-based analysis and comparison of long-term investments," in *Information Visualisation, 2008. IV '08. 12th International Conference*, july 2008, pp. 287 –295.

[58] J. Schneidewind, M. Sips, and D. Keim, "Pixnostics: Towards measuring the value of visualization," in *Visual Analytics Science And Technology, 2006 IEEE Symposium On*, 31 2006-nov. 2 2006, pp. 199 –206.

[59] J. Zhao, F. Chevalier, E. Pietriga, and R. Balakrishnan, "Exploratory analysis of time-series with chronolenses," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 17, no. 12, pp. 2422 –2431, dec. 2011.

[60] M. Hao, U. Dayal, D. Keim, and T. Schreck, "Importance-driven visualization layouts for large time series data," in *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on*, oct. 2005, pp. 203 – 210.

[61] A. Pilhofer, A. Gribov, and A. Unwin, "Comparing clusterings using bertin's idea," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2506–2515, 2012.

[62] G. Ellis and A. Dix, "Enabling automatic clutter reduction in parallel coordinate plots," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 5, pp. 717–724, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1109/TVCG.2006.138

[63] X. Zhao and B. Li, "Pattern recognition and revealing using parallel coordinates plot," *CoRR*, vol. abs/1306.1959, 2013.

[64] A. Arratia and A. Cabaña, "Tracing the temporal evolution of clusters in a financial stock market," *CoRR*, vol. abs/1111.3127, 2011.

[65] J. Heinrich, J. Stasko, and D. Weiskopf, "Eurographics conference on visualization (eurovis) (2012) m. meyer and t. weinkauf (editors) short papers the parallel coordinates matrix," 2012.

[66] J. Ma, C. Wang, and C.-K. Shene, "Flowgraph: A compound hierarchical graph for flow field exploration," 2013.

[67] M. Garcia, W. Huang, C. Seifert, and W. Wallisch, "Literature survey: The recordable matrix," may 2010. [Online]. Available: http://courses.iicm.tugraz.at/ivis/surveys/ss2010/g5-survey-reord-matrix.pdf

[68] N. Elmqvist and J. Fekete, "Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 16, no. 3, pp. 439–454, May 2010.

[69] M. Ghoniem, J. Fekete, and P. Castagliola, "A comparison of the readability of graphs using node-link and matrix-based representations," in *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, 2004, pp. 17–24.

[70] J. Bertin, *Graphics and graphic information-processing.* de Gruyter, 1981. [Online]. Available: http://books.google.com/books?id=2tlQAAAAMAAJ