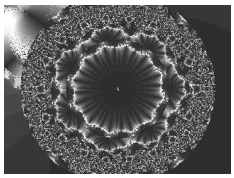
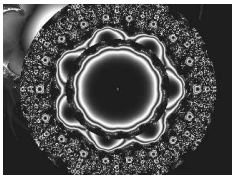


VDM@ICDM 2003



Proceedings
3rd International Workshop on Visual Data Mining

19th November, 2003, Melbourne, Florida, USA

Edited by
Simeon J. Simoff, Monique Noirhomme-Fraiture,
Michael H. Böhlen and Mihael Ankerst

---

in conjunction with

ICDM 2003 - The 3rd IEEE International Conference on Data Mining, November 19 - 22, 2003, Melbourne, Florida, USA

University of Technology Sydney
2003

Proceedings of the 3rd International Workshop on Visual Data Mining - VDM@ICDM2003, in conjunction with ICDM 2003 - The 3rd IEEE International Conference on Data Mining, November 19 - 22, 2003, Melbourne, Florida, USA

S. J. Simoff, M. Noirhomme-Fraiture, M. H. Böhlen and M. Ankerst (eds)

Workshop Web Site:
http://www-staff.it.uts.edu.au/~simeon/vdm_icdm2003/

Published by the University of Technology Sydney
ISBN

Foreword

Visual data mining is a collection of interactive methods for knowledge discovery from data, which integrates human perceptual capabilities to spot patterns, trends, relationships and exceptions with the capabilities of the modern digital computing to characterise data structures and display data. The underlying technology builds on visual and analytical processes developed in various disciplines including data mining, information visualisation, and statistical learning from data with algorithmic extensions that handle very large, multidimensional, multivariate data sets. The outcomes of the growing research and development in visual data mining includes means of visual analysis that can assist in uncovering patterns and trends and may be missed when using non-visual methods. Consequently, the data mining communities have recognised the significance of this area.

The first and second edition of this workshop series took place at the ECML/PKDD conferences in Freiburg and Helsinki in 2001 and 2002, respectively. Both workshops were supported by the 3DVDM group from Aalborg University, Denmark. The workshops offered to the participants of these European machine learning and data mining forums a mixture of presentations on state-of-art methods and techniques, with controversial research issues and applications. A report about the first workshop has been published in SIGKDD Explorations 3 (2), pp. 78-81, the report on the second workshop is on the way. A book, which includes a selection from the presentations at both workshops is under preparation and is expected to be published with Springer (in LNCS series) by the end of 2003.

Both years, the workshops brought together a number of cross-disciplinary researchers, who were pleased with the events and there was a consensus about the necessity of turning it into an annual meeting. This workshop has been initiated and organised in response to this interest. This year the workshop organisers considered the change of the workshop venue from Europe to North America, which is the reason why the workshop is collocated with ICDM2003. Being a third edition, the workshop is aiming to meet the “visualisers” and the “data miners” and create a stimulating atmosphere for open discussions of the cross-disciplinary foundations and frameworks of visual data mining, visualisation algorithms and interactive

visual mining methods. Consequently, the papers selected for presentation at the workshop are grouped in the following sessions: Frameworks and Result Interpretability in Visual Data Mining (2 papers), Visualisation Techniques for Visual Data Mining; Methodologies for Visual Data Mining; and Visual Data Mining Software Demonstrations. The works selected for presentation at this workshop form a cohesive body of work, which indicates that the field has made another step forward towards achieving some level of maturity.

We would like to thank all those who submitted their work to the workshop. As part of the ICDM conference series, the workshop follows a rigid blind peer-review process. All papers were extensively reviewed by at least three referees drawn from the program committee. Special thanks go to them. Once again, we would like to thank all those, who supported this year's efforts on all stages – from the development and submission of the workshop proposal to the preparation of the final program and proceedings.

Simeon J. Simoff
Monique Noirhomme-Fraiture
Michael H. Böhlen
Mihael Ankerst

November 2003

Workshop Chairs

Simeon J. Simoff	University of Technology Sydney
Monique Noirhomme-Fraiture	FUNDP, Belgium
Michael H. Böhlen	Free University of Bozen-Bolzano, Italy

Local Chair

Mihael Ankerst	Boeing, USA

Program Committee

Henri Briand	Universite de Nantes, France
Luca Chittaro	University of Udine, Italy
Di Cook	Iowa State University, USA
Cristina Davino	University of Naples Federico II, Italy
Alberto Del Bimbo	Università di Firenze, Italy
Chabane Djeraba	Universite Des Sciences Et Technologies De Lille, France
Alex Duffy	University of Strathclyde, UK
Eric Granum	Aalborg University, Denmark
Georges Grinstein	University of Massachusetts at Lowell, USA
Fabrice Guillet	Universite de Nantes, France
Alexander Hinneburg	Martin-Luther-University Halle-Wittenberg, Germany
Daniel Keim	University of Konstanz, Germany
Donato Malerba	Università degli Studi, Italy
Arturas Mazeika	Aalborg University, Denmark
François Poulet	Parc Universitaire de Laval-Change, France
Michael Schroeder	City University, UK
Bruce Thomas	University of South Australia, Australia
Djamel A. Zighed	Université Lumière Lyon, France

Table of Contents

Toward Visual Web Mining

Amir H. Youssefi[1], David J. Duke[2], Mohammed J. Zaki[1], and Ephraim P. Glinert[1]

[1] Department of Computer Science,
Rensselaer Polytechnic Institute,
110 Eigth Street, Troy, NY 12180
{Youssefi,Zaki,Glinert}@cs.rpi.edu
http://www.cs.rpi.edu/~youssefi/research/VWM
[2] Department of Computer Science,
University of Bath,
Bath, BA2 7AY, U.K.
D.Duke@bath.ac.uk

Abstract. Researchers analyzing web sites face two challenges of the increasingly large amount of data published online as well as web sites with more complex structures. We apply Data Mining and Information Visualization techniques to the web domain in order to benefit from the power of both computing and the human visual perception; we term this Visual Web Mining. In response to the two aforementioned challenges, a generic framework is proposed. Within this framework, we apply Data Mining techniques to large datasets and use Information Visualization methods on the results. The goal is to compare the outcomes of mining Web Usage Logs and extracted Web Structure by visually superimposing the results. We propose several new information visualization diagrams and analyze their utility, elaborate the architecture of a sample implementation, and present work in progress, along with early results and discussion of ideas for future work.

1 Introduction

Analysis of web site regularities and patterns in user navigation is getting more attention from businessmen and research community as web browsing becomes an everyday task for more people around the world. Lots of efforts has been made for exploring these regularities and some through visualization [6, 7, 10, 14, 19, 34].

To enable visual superimposition, we have designed new visual representations of site structure and user access patterns. The second part of this paper discusses the ideas behind these and the implementations.

Aspects of our representations have been motivated by metaphors developed initially within scientific visualization, in particular the representation of vector-fields in fluid dynamics. The results are interactive 3D visualizations implemented in the Visualization Toolkit [29], providing exploratory tools for supporting Visual Web Mining (VWM).

The architecture of our implementation, work in progress and early results are presented and ideas for future work are discussed. We leave further analysis and usability tests for future work.

Visualization is defined as the use of computer-supported, interactive, visual representations of data to amplify cognition [2]. Some surprising examples in which a good visualization can unlock previously hidden structure and yet not obscure the detail are given in [9]. *Information Visualization* is the use of computer-supported, interactive, visual representations of abstract data to amply cognition [2]. The field of *Information Visualization* is about creating tools that exploit the human visual system to help people explore or explain data [25] or simply visualization applied to abstract data [2]. Computer support brings the opportunity to explore radically different representations, including the use of 3-dimensional models.

The capability to interact with a visual representation is significant in allowing users to explore large-scale datasets, where it is infeasible to provide both an overview of the space plus information about points of focal interest (the so called "focus plus context problem").

Different approaches have been taken by researchers to visualize information some notable ones are [8, 11–13, 15–17]. Visual Data Mining [5] is of particular interest.

A key challenge in Information Visualization is finding a spatial mapping for an abstract data set that is cognitively useful for a specific task. To address this, information visualization draws on ideas from several intellectual traditions, including computer graphics, human-computer interaction, cognitive psychology, semiotics, graphic design, and cartography.

Although there is still disagreement on the distinction between Scientific Visualization and Information Visualization, for the purposes of this paper the distinguishing feature of information visualization is taken to be the need to find a spatial mapping of data that is not inherently spatial, whereas scientific visualization uses a spatial layout that's implicit in the data.

Web Mining as application of data mining techniques to the World Wide Web, can be devided into Web Content Mining, Web Usage Mining and Web Structure Mining. In this paper, our focus is on last two parts because of implementations available to us. We define the notion of a *user session* as a temporally compact sequence of web accesses by a user. The goal of our web mining in part is to work on these sessions for better visualization toward useful information.

A common theme underlying the use of visualization in website analysis is the graph metaphor, that is, the organization of a web site and/or patterns of access are treated as a node-link graph. There is a considerable literature on algorithms for drawing graphs however, making *aesthetically pleasing* drawings of graphs (e.g. with a minimal number of edge crossings) is computationally expensive, and for large-scale graphs new techniques have been developed, see [4] for case studies, and [26] for a survey of recent approaches. One issue is that drawing general graphs is harder than drawing trees, for which a number of efficient approaches are known, for example cone trees [8]. Where a graph is not

itself a tree, tree layout can be applied to a spanning tree of the graph, with non-tree edges either added to the resulting structure [1], or not included if the spanning structure is sufficient for a given task. For example, the structure of a web site is in general a graph, with pages corresponding to nodes, and links to edges. The high-level organization of a site is often hierarchically i.e. as a tree, with one topic branching into a number of sub-topics, as in linear printed media. However, a strength of hypermedia is the ability to cross-link topics, and also provide navigational paths back to higher-level areas, and these kind of links then form non-tree edges of the corresponding graph.

An interesting variation on the node-link representation of trees is the tree-map representation [28]. In the context of visualizing data mining results, the ability of this representation to convey large-scale datasets is an advantage. The limitation of this approach is that, while it is good at conveying properties of nodes, it does not realiy convey paths or attributes of sub-structures. Nor is it clear whether non-tree structure can be attached to the representation.

2 Overview of Visual Web Mining Framework

We propose *Visual Web Mining* (VWM) as application of Information Visualization techniques on results of Web Minining in order to further amplify perception of extracted patterns, rules and regularities or visually explore new ones on web domain.

We describe a framework i.e. *Visual Web Mining Framework* and use it in a sample implementation and there we construct useful diagrams using Information Visualization techniques on the results of Data Mining Algorithms such as Sequence Mining[36], Tree Mining [35] with input of large web access log files as well as huge Web Graphs. Web graphs are semi-static snapshots of web site structure, in contrast web access logs capture the dynamic behavior of surfers visiting a web site. Visual superimpostion of dynamic and static views enables web analyzers or even web masters to compare these two aspects and gain insight on what actually happens on a website.

The organization of large websites reflects a diverse range of design criteria and rationale; e.g., personal preferences of web designers, and the breadth and structure of the topics presented on the site. In the second part of this paper we use visualization to obtain an understanding of the structure of a particular website, as well as web surfers' behavior when visiting the site.

Due to the scale of dataset on which we work, and the structural complexity of the sites from which the data is obtained, we have decided to use 3D visual representations, with user interaction to support navigation and exploration. We have implemented this using an open source toolkit called the Visualization ToolKit (VTK) [29].

The decision to implement website visualization using a toolkit developed, at least initially, for engineering and medical applications, deserves further comment. A number of public-domain tools for the visualization of graphs do exist, and some can cope with large graphs, with up to one million nodes. However,

these tools typically provide a fixed repesentation of a graph (a node-edge diagram), and have limited modularity. In the case of visual web mining, we wish to experiment with novel combinations of metaphors, including spatial positioning, tubing of edges, node glyphs, and color mapping. These techniques are provided in a modular way within general visualization tools, such as VTK, and we have therefore chosen to extend VTK with specific support for working with node-link graphs. Further rationale and design issues underlying such an approach can be found in [3].

Although the work is at an early stage, and improvements in both mining techniques and visual representation are ongoing, early results already show different notable phenomena e.g. Cone, Funnel and Debris which corresponds to user behaviors as he/she browses the website (see figure 2). Different mappings are made from data attributes to visual attributes on each diagram. Visualization techniques are used to portray mined structures via visual features such as splines, trees, and stream tubes, with visual attributes such as color, and size used to convey interesting data features, for example users leaving the web site.

Implementation of this Visual Web Mining Framework has been applied to the website and access logs of the Computer Science Department of Rensselaer Polytechnic Institute (http://www.cs.rpi.edu).

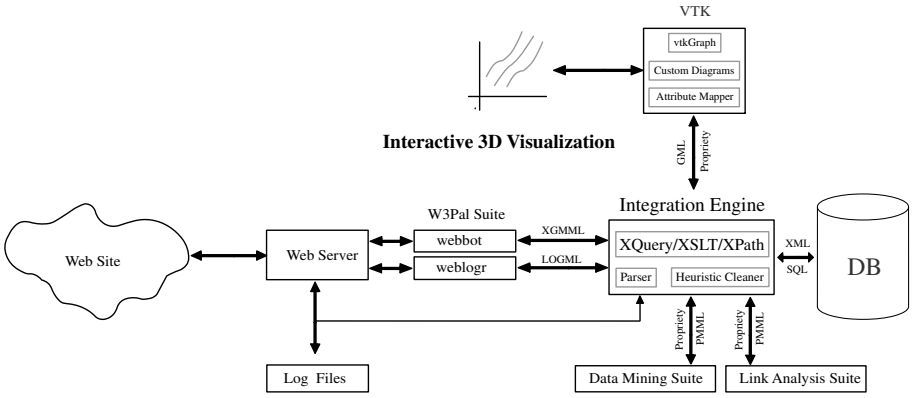


Fig. 1. Sample implementation architecture of VWM

3 Architecture

Figure 1 shows the architecture of our implementation. We target one or a group of websites for analysis. Input of the system consists of web pages and web server log files. Access to web log is either through the local file system, or by downloading it from a remote web server (possibly as a password protected resource).

A web robot (webbot) is used to retrieve the pages of the website and then construct a web graph. Our implementation uses the robot of the WWWPal Suite [32] to generate this graph, which is subsequently converted to the simple GML format used by the visualization tools.

In parallel, Web Server Log files are downloaded and processed through a sessionizer and a LOGML [30] file is generated. Web logs could be in either Common Log Format or Extended Common Log Format. Both input and output of different parts are in standard (XML) languages so output of other similar programs could be fed to the system with small changes also results can be used by other systems.

The Integration Engine is a suite of programs for data preparation i.e. cleaning, transforming and integrating data. It uses XQuery, XSLT and XPath as well as Regular Expression Parsing on XML languages also propriety text data to do the transformation. Support for PMML is readily available as well. A connection module is in charge of Bulk Loading of XML data into database and executing SQL commands against the database. Schema matching is done using external tools as well as code snippets which map different schema and import/export different XML/text files into/from relational tables in our database server. Web access log files are also imported into database for exploring and comparison of Data Mining Algorithms as well as verification of data integration.

Data Mining Suite and also *Link Analysis Suite* need special data format as input and give output in propriety formats hence the Integration Engine is required to covert data in these formats.

The visualization stage of this pipeline, which maps the extracted data and attributes into visual images, is realized through VTK extended with support for graphs. VTK[29] (and this extension) is a set of C++ class libraries, available on a range of architectures including Windows and Linux. The class library is accessible either through linkage with a C++ program, or via wrappings supported for scripting languages (Tcl, Python or Java). In the case of this work, the visualization engine is delivered in the form of a Tcl script. Results are interactive 3D/2D visualizations which could be used by analysts to compare actual web surfing patterns to expected patterns, or more generally, the intended purpose and role of the website.

Web Mining has no single recipe for satisfying requirement of analysts or business managers. Businessmen ask high level, diverse questions[20], for example:

- Is our site sticky? Which regions in it are not?
- How adept is our conversion of browsers to buyers?
- What site navigation do we wish to encourage?
- What attribute describes our best customers?
- What makes customers loyal?
- How can profiling help use cross-sell and up-sell?

These questions are semantically distant from the data available for analysis, and suitable responses are beyond the scope of either simple data mining or

visualization. What VWM can provide, however, is insight on more specific, focused questions, for example:

- What is the typical behavior of a user entering our website?
- What is the typical behavior of a user entering our website in page A from 'Discounted Book Sales' link on a referrer web page B of another web site?
- What is the typical behavior of a logged in registered user from Europe entering page C from link named "Add Gift Certificate" on page A?
- What is the typical behavior of a user who come in our website 1 day to 3 weeks before Christmas and buys something, versus one who didn't by anything?

In order to partially support this kind of analysis we take the following approach in our example implementation of the VWM framework:

- Make personalized results for targeted web surfers (users, customers) as opposed to blindly aggregating on all users.
- Use Data Mining Algorithms for extracting new insight and measures.
- Employ a database server and relational query languages as means to submit specific queries against data e.g. projection and aggregations, joins etc.
- Utilize visualization to obtain an overall picture correlating static site structure with (dynamic) access patterns. Support 'at a glance' overviews using visual aggregation and filters. Amplifying special features via interactive control.

A key issue for the data mining part of this process is how one translates the notion of the *typical behavior* of a user into actual queries on data sets to yield sufficient insight for an analyzing team to identify whether (or how) the (business) goals of the website have been satisfied relative to those targeted customers.

These points serve to emphasize how visualization complements data mining. The latter is about utilizing capabilities of the machine to find and/or compute patterns within the dataset, based on notions of pattern derived from domain knowledge or statistics. Visualization, in contrast, is about using human capabilities to detect patterns or discern trends within visual representations.

In our approach we first extract user sessions from web logs, this yields results related to a specific person as much as possible. User sessions are then converted into a special cSPADE [36] format:

```
<customer_id> <time_id> <item1> <item2>
```

Here `item1` is the smaller numerical value of SourcePageID and TargetPageID, while `item2` is the larger numerical value of TargetPageID and SourcePageID; this ordering is needed as input to cSPADE needs to be sorted. This input means that a customer (web surfer) bought `item1` and `item2` (and thus visited the source and target pages) together at her shopping (click).

For each click of user we need to have both source page and target page, not just source or target page, in order to deal with different cases in user navigation such as Hitting Browser's Back Button or finding Entry Page or Exit Page. Furthermore, Source pages (or target pages) out of our website could either be considered as a single page or for more details analysis multiple PageIDs could be assigned by a classification or clustering algorithm say on referrer URL (or Exit Page) to gain more insight.

We convert text input data into binary format, vertical format, make indices and run cSPADE algorithm using lower limit of 1 and upper limit of 1 in order to get *continues* frequent sequences with a given support. Results are imported into databases, shorter frequent sequences are removed. Later different queries are executed against this data according to some criterion e.g. support of the patterns found, length of found patterns etc. This gives the flexibility to translate *typical behavior* into different appropriate queries according to subjective requirement set forth by analysis team without going all the way down to web log analysis again.

Result is a graph rather than a simple sequence since user can go back from one page to previous ones or follow anchors in the same web page or simply follow a closed path as sequence of pages.

We attempt to present abstractions of large amounts of data in tuned diagrams in order to maximize opportunities for pattern detection via human perceptual capabilities. We have designed several visualization diagrams, some of which have been implemented. The contribution of some of those diagrams is discussed and illustrated shortly; video clips of the system in action are available from the webpage of the VWM project [31].

4 From Fluid Dynamics to Web Dynamics

Advances in scientific visualization techniques, particularly in the domain of fluid dynamics [21–23], have inspired us to apply similar techniques in information visualization on the web domain. We use an analogy between the 'flow' of user click streams through a website, and the flow of fluids in a physical environment, in arive at new representations As with most problems of information visualization, our representation of web access involves locating 'abstract' concepts (e.g. web pages) within a geometric space. With this as a base, the following ideas motivate our design of the visualization diagrams discussed in the next section:

- Iconic visualization on extracted features from Data Mining.
- Particle tracking for tracing the path of user navigation.
- Feature tracking/detection, for instance turbulence vortex detection techniques, for identifying a user cycling around a set of web pages. Note that a similar phenomena happens for a user navigating pages and this could be interpreted both as confusion and misled tracking or quite reverse a case where a user is focusing on a cluster which he/she has found interesting.

– Splatting for simplying visualization implementation in dense parts (both 'hot' and 'cold' parts of a graph could be splatted in order to focus users' attention on edges connecting these two regions.
– Fluid event detection on navigation clickstreams to capture special (interesting) events.

In [24] visualization techniques were grouped in three categories:

1. *Global Techniques* give a qualitative, global visualization of the data at a low level of abstraction.
2. *Geometric techniques* extract geometric objects (curves, surfaces, solids) from the data. They can be considered as intermediate-level representations, both with regard to locality and level of abstraction.
3. *Feature-based techniques* extract high-level, abstract entities from the data. The emphasis is on quantification for more precise evaluation and comparison.

The second and third of these are used for our visualization methods.

4.1 Structures

Graphs Much work has been done on two and tree dimensional embedding of graphs, in the form of both algorithms and tools, see for example [1, 18, 32, 33]. We have developed a library under VTK for graph visualization[3] (see `http://www.cs.bath.ac.uk/~djd/graphs.html`). Two approaches to visualizing the output of data mining have so far been implemented using this library. In the first approach, the link analysis is treated as a directed graph. This graph is drawn by obtaining a spanning tree for the underlying graph, laying out this tree (using a variation of the cone tree layout first proposed by Robertson et al [8]) and then re-introducing the non-tree edges. Further attributes obtained by data mining can then be superimposed on the underlying graph structure, for example as variable-width tubes showing particularly strong access paths.The second approach to visualization is to take a spanning tree and use this itself as the main visual element, supported by color mapping edges and glyphing nodes. In either case visual attributes of nodes and edges have different mappings/meanings for each of the diagram designed, the details of which are discussed in the next section.

Stream Tubes Variable-width tubes showing access paths with different traffic are introduced on top of the web graph structure. Here, depending on the type of visualization diagram, particular weights e.g. support of a single click-stream, total sum of support on all click-streams, support extracted from Tree Mining Algorithm [35] can be mapped onto the width (radius) of the stream tube. Color mapping could be used on the number of users leaving the website (or a cluster of these in a zoomed view), a property of the graph structure (such as the Strahler value), or simply the number of hits of some branch.

4.2 Design and Implementation of Diagrams

Building on the intuition and heuristics set out above, we have designed new visualization diagrams; these are summarized below, after which we will consider issues of utility.

- Figure 3 is a visualization of the web graph for the Computer Science department of Rensselaer Polytechnic Institute (http://wwww.cs.rpi.edu).
- Figure 4 is a 3D visualization of web usage aggregation for this site.
- Figure 2 is representation of the same site in which color mapping is used to highlight the mass of surfers scattering in any cluster making a Cone shape and later coming back to main pages of clusters and to the first page of the website making a Funnel. A user session drilling down into a cluster in a form of a semi-directed flying debris is also observed.
- Figure 10 is superimposition of web usage mining results (dynamic user access patterns) on top of the web graph (the static link structure of the web site).

A web master can find out where the load of his website and bandwidth goes by a quick look at this kind of diagrams e.g. figure 4 and click on glyphed nodes which represent web pages to see which page each node represents (see figure 9). Cube glyphs with proportional size are put on top of nodes to make them easily clickable. Glyph size adds another visual dimension that can be used for encoding attribute data. The efficiency of the underlying VTK graphics interface (in turn built on an interface to OpenGL or Mesa) is such that users can easily zoom, pan and rotate these specific diagrams, even on modest hardware (e.g. a 433Mhz Pentium II). Thus users can explore different parts of the sample website i.e. http:/www.cs.rpi.edu as extracted from the weblogs. Strategies for real-time interaction with much larger datasets are already being explored.

Web mining results in patterns of most frequent access which are visualized in 5 as white edges superimposed on the remainder of the figure. Figure 6 extends this approach by using thickness of stream tubes to add a further visual dimesion, in this case encoding how frequent those access patterns are.

In figure 10 we have visualized the static structure of website taken from our webbot (gray-coloured edges) to make a basement on which actual dynamic behavior of users is superimposed with (colored edges). There are about half a million visualized nodes. A webmaster or a web analyzer can easily see which parts of the website are 'cold' parts with low hit and which parts are 'hot' parts with high hit. This also paves the way for making exploratory changes in web site and analyze the changes in user access. For instance a webmaster can change link structure e.g. by adding a link of a cold cluster in first page of website. Another example is change of content e.g. by highlighting existing anchor text or putting it on a more visible locatoin of a web page or adding advertisements/banners and then analyze changes to the user behavior.

Although the visual images represent preliminary results from our fusion of mining and visualization, we have endeavoured to use suitable guidelines and

heuristics based on perceptual, cognitive and aesthetic criteria, for example using psychological notions of hot or cold color to guide assignment of color to nodes and edges having high/low numbers of hits, respectively. Using different diagrams we can focus on each of three link classes we find interesting: hot nodes/edges, cold nodes/edges, and edges connecting two different types of clusters i.e. hot clusters with high number of hits as opposed to cold clusters with low number of hits. This is valuable for web masters to make decisions and analyze changes of dynamics by informed or exploratory addition of edges between these two clusters. In future work we intend to use techniques such as splatting, spot noise, and/or filtering edges/nodes which are less important, in order to refine and clarify our understanding. As Hamming (1973) said,"The purpose of computation is insight not number". Likewise"The purpose of visualization is insight, not pictures" [2]. A detailed assessment of the utility of our diagrams is beyond the scope of this paper and we leave it for our future work. Recent studies lends support to our approach of discriminating different types of cluster structures [27].

5 Future Work

There is considerable room for improving the visualization of these results. First, at the algorithmic level, the scale and complexity of the graphs produced from the data mining stage still have the potential to embarrass available graph layout algorithms. There is often a tension in the design of algorithms between accommodating a wide range of data, or customizing the algorithm to capitalize on known constraints or regularities, and in the case of web log data, knowing more about the kind of graph that is to be drawn may help in simplifying the layout process. On the human side, further thought is needed on the mapping from data attributes to visual attributes, in particular where the visualization is superimposing access properties above the basic site structure. Part of this work can and should be based on known characteristics of perception and principles of visualization design, however, the ultimate utility of the representation will only become apparent once it is assessed through controlled experiments, and this will require time and a more polished version of the user interface.

A number of further tasks have already been mentioned in the text, the following could be added:

- A perceptual and logical appraisal of the visualizations relative to better understanding of specific user tasks.
- (Empirical) usability tests of the visualizations.
- Demonstrating the utility of web mining can be done by making exploratory changes to web sites e.g. adding links from hot parts of web site to cold parts and then extacting, visualizing and interpreting changes in access patterns. This may also require running our implementation on logs obtained over longer period of time.
- Visualizing output of related systems e.g. navigation predictors, recommender systems, browsing simulators, user modeling/profiling. Likewise a broad range

of AI algorithems, learners, probabilstic algorithms and more can be visually correlated with real user behavior in an unintrusive manner. Visualization of new Relational Probabilistic Models is of particular interest because of inclusion of relational structure of web sites in our techniques.
- Visualization of Link analysis and comparison of PageRank and Hub/Authority against actual web usage in line with Web Structure Mining. Also Web Content Mining can be introduced to our architecture in parralel to WWWPal suite.
- We remove web bot request in early stages of pipline, one interesting application is to visualize access path of those.

6 Acknowledgement

We would like to thank Prof. Mukkai S. Krishnamoorthy for his help on `WWWPal` [32].

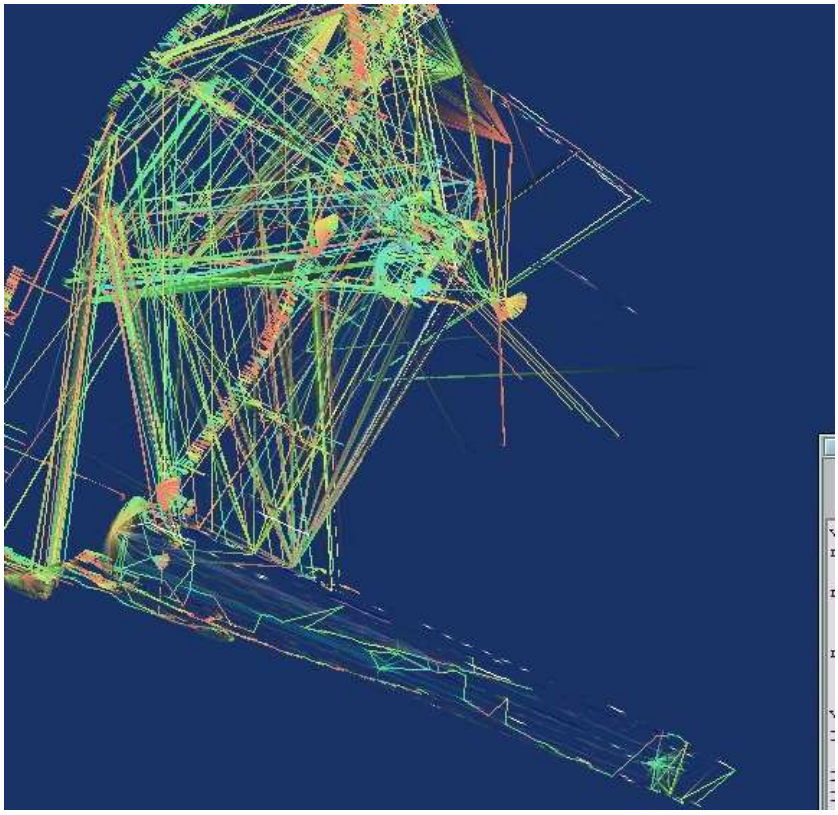


Fig. 2. Phenomena: Drilling Down, Cone and Funnel. User's browsing access pattern is amplied by coloring at the bottom of the picture.

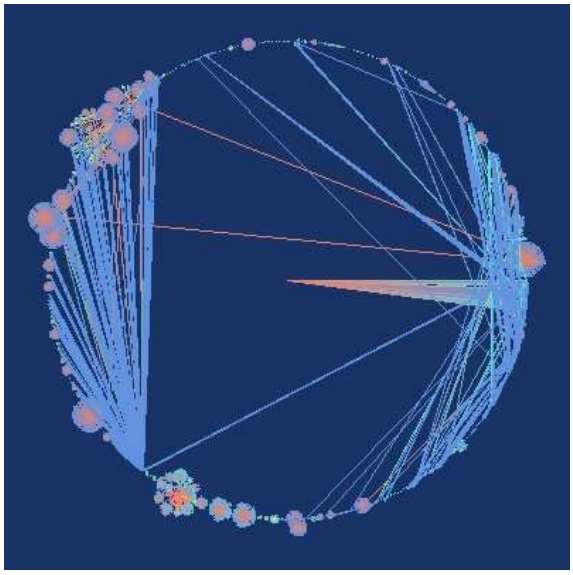


Fig. 3. 2D visualization with Strahler Coloring.

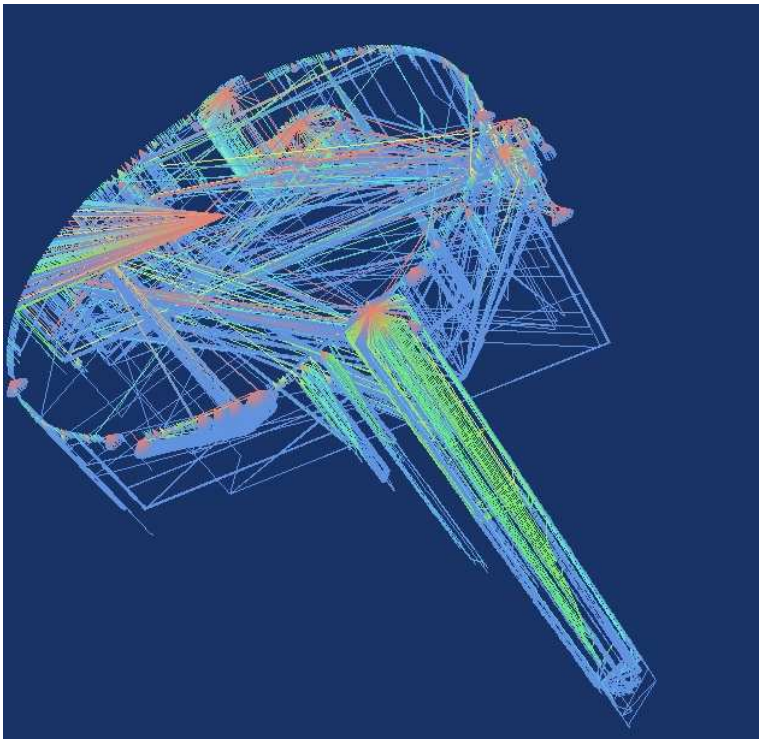


Fig. 4. Interactive 3D visualization of Drilling Down: Circular basement of the cylender like result is similar to figure 3, adding third dimension enables us visualize more information and clarifies user behavior in and between clusters.

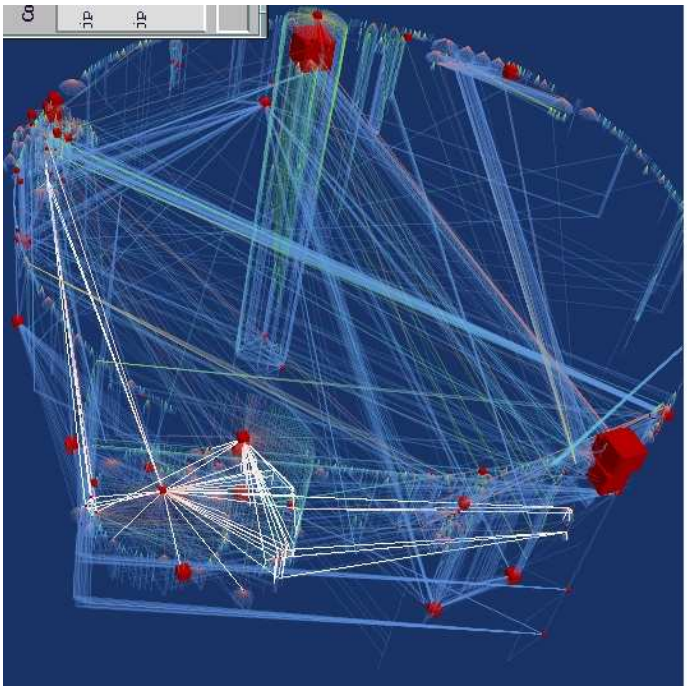


Fig. 5. Visualization of Mined Frequent Trees/Sequences: White lines are results of data mining algotithms i.e. frequent patterns.

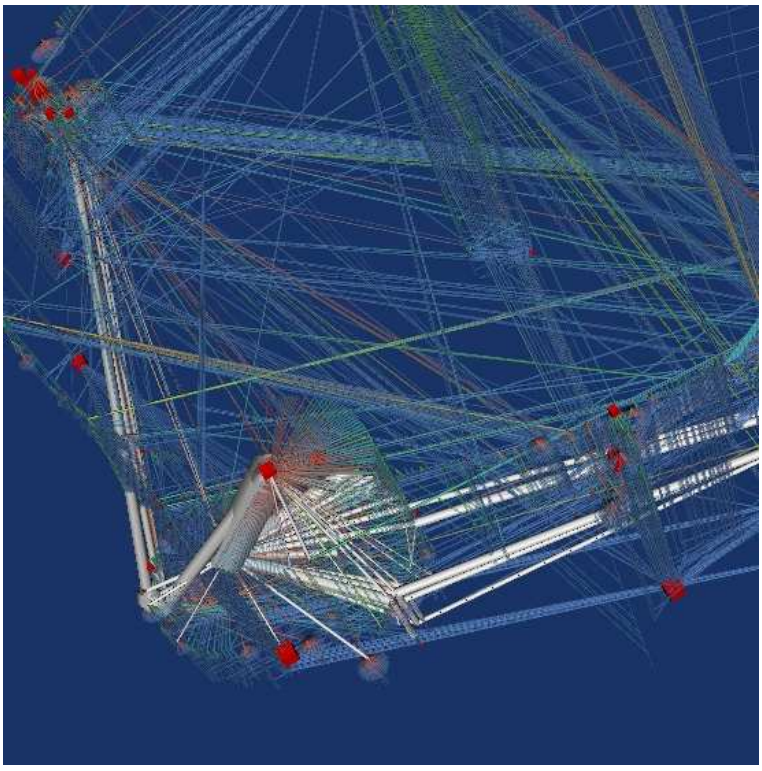


Fig. 6. Frequent Patterns visualized as Stream Tubes: Using tickness of Stream Tubes, we can add another dimension i.e. frequency of patterns to figure 5.

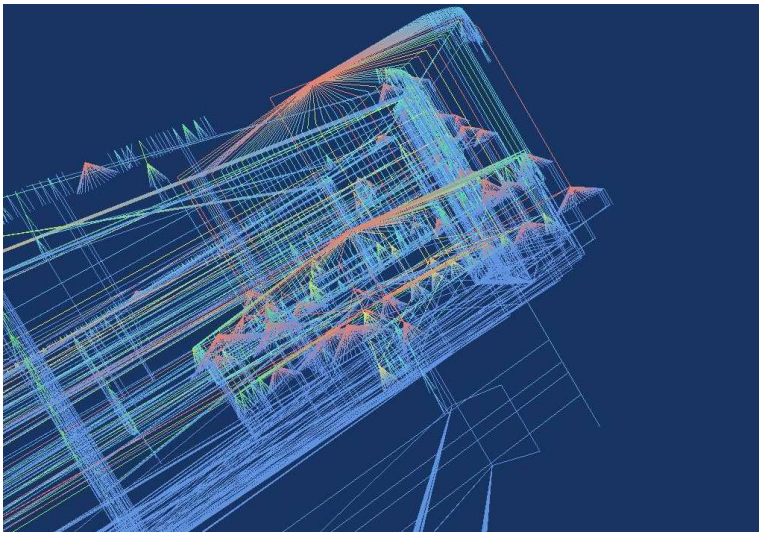


Fig. 7. Cone Trees.

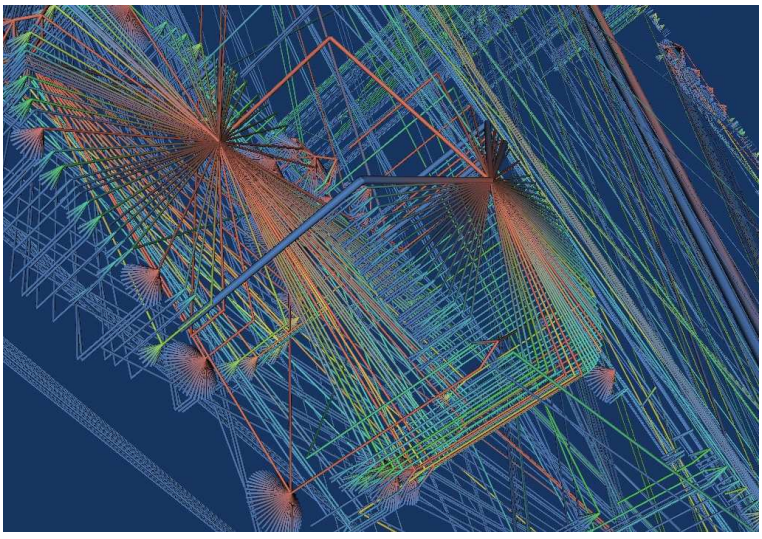


Fig. 8. Using click stream tubes on a zoom view.

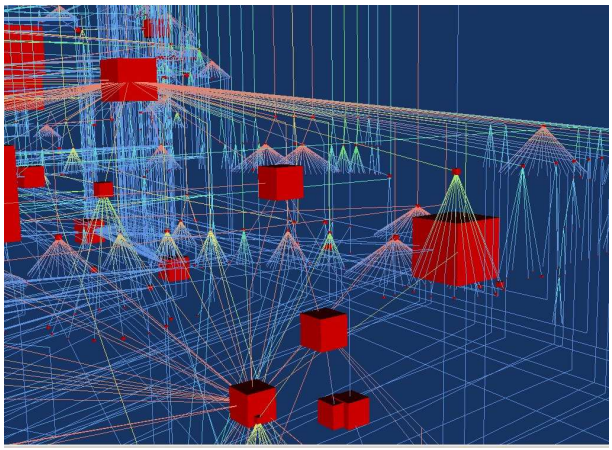


Fig. 9. Using Glyphs: Cube shaped glyphs with proportional size are put on nodes for easier selection by mouse.

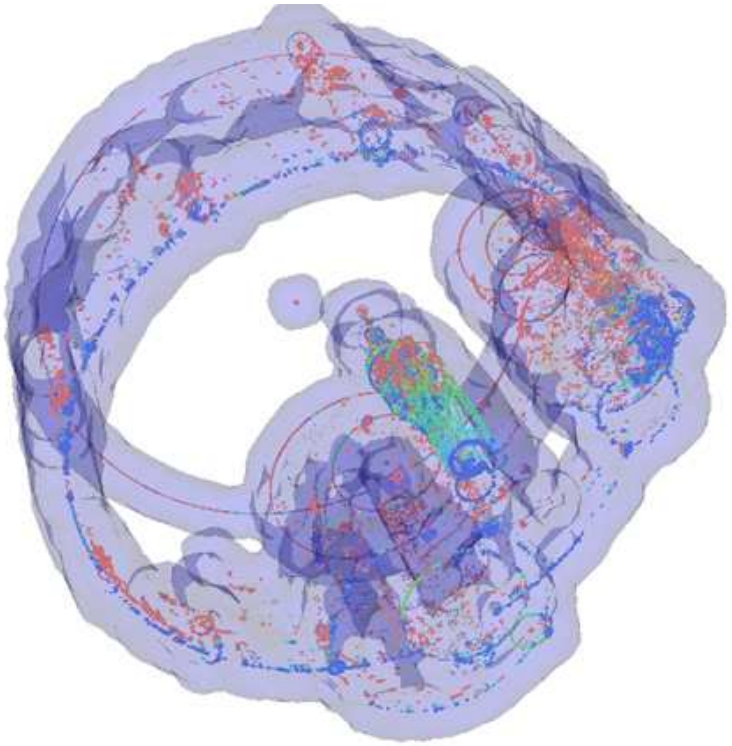


Fig. 10. Web Usage Superimposed on Web Graph with about 400,000 nodes. Semi-Static Web Graph extracted by webbot makes a gray basement for colored frequent paths extracted from Web Logs using Web Mining Algorithms. Clusters with/without frequent access are easily identified

References

1. David Auber, Using Strahler Numbers for Real Time Visual Exploration of Huge Graphs, International Journal of Applied Mathematics and Computer Science,2002.
2. S.Card, J.Mackinlay, and B.Shneiderman, Readings in Information Visualization: Using Vision to Think, Morgan Kaufmann, San Francisco, 1999.
3. D.J. Duke, Modular Techniques in Information Visualization, Proceedings of the 1st Australian Symposium on Information Visualization, Volume 9 of Conferences in Research and Practice in Information Visualization, P. Eades and T. Pattison (Eds), pp. 11-18, Australian Computer Society, 2001.
4. T. Munzner, Drawing Large Graphs with H3Viewer and SiteManager, in Proceedings of Graph Drawing'98, 1998.
5. Daniel A. Keim, Information Visualization and Visual Data Mining. IEEE Transactions on Visualization and Computer Graphics 8(1), 2002
6. C. Shahabi, A. M. Zarkesh, J. Abidi and V. Shah, Knowledge discovery from user's web-page navigation, Proc. Seventh IEEE Intl. Workshop on Research Issues in Data Engineering (RIDE), pp. 20-29, 1997.
7. B. A. Huberman, P. L.T. Pirolli, J. E. Pitkow, and R. M. Lukose. Strong regularities in World Wide Web surfing. Science, 280:96-97, April 3, 1997.
8. G.G. Robertson, J.D. Mackinlay, S.K. Card. Cone Trees: Animated 3D visualizations of hierarchical information. Proc. of ACM SIGCHI, 1991.
9. E. Tufte, Envisioning Information. Graphics Press, 1990.
10. Alan Keahey, Stephen G. Eick: Visual Path Analysis. INFOVIS, 2002.
11. I. V. Cadez, D. Heckerman, C. Meek, P. Smyth, S. White. Visualization of Navigation Patterns on a Web Site Using Model Based Clustering, SIGKDD, 2000.
12. Kirsten Risden, Mary P. Czerwinski, Tamara Munzner, and Daniel B. Cook. An initial examination of ease of use for 2d and 3d information visualizations of web content. International Journal of Human Computer Studies, 53(5):695-714, 2000.
13. John Cugini,Jean Scholtz. VISVIP: 3D Visualization of Paths through Web Sites, WebVis, 1999.
14. http://www.nist.gov/webmet/
15. Tamara Munzner, Drawing Large Graphs with H3Viewer and Site Manager, Proceedings of Graph Drawing, 1998.
16. E. H. Chi, J. Pitkow, J. Mackinlay, P. Pirolli, R. Gossweiler and S. K. Card (1998). Visualizing the Evolution of Web Ecologies. ACM Conference on Human Factors in Software (SIGCHI), 1998.
17. Ed H. Chi, Stuart K. Card. Sensemaking of Evolving Web Sites Using Visualization Spreadsheets. INFOVIS 1999.
18. Paul Mutton, Peter Rodgers, Spring Embedder Preprocessing for WWW Visualization, 2002.
19. Linda Tauscher, Saul Greenberg, Revisitation Patterns in World Wide Web Navigation (1997). Proceedings of the Conference on Human Factors in Computing Systems CHI'97
20. Jim Sterne, Invited Talk: WebKDD in the Business World, WebKDD workshop of SIGKDD 2003.
21. F. H. Post, B. Vrolijk, H. Hauser, R. S. Laramee, and H. Doleisch, "The state of the art in flow visualisation: Feature extraction and tracking," Computer Graphics Forum, vol. 22, no. 4, 2003.
22. F. Reinders, I. A. Sadarjoen, B. Vrolijk, and F. H. Post, "Vortex tracking and visualisation in a flow past a tapered cylinder," Computer Graphics Forum, vol. 21, pp. 675–682, Nov. 2002.

23. F. Reinders, F. Post, and H.J.W.Spoelder, Visualization of Time-Dependent Data using Feature Tracking and Event Detection, The Visual Computer, vol. 17(1), pp. 55-71, February 2001.
24. F. Post, W. de Leeuw, I. Sadarjoen, F. Reinders, and T. van Walsum, "Global, Geometric, and Feature-Based Techniques for Vector Field Visualization," Future Generation Computer Systems, vol. 15, February 1999.
25. Tamara Munzner, Guest Editor's Introduction, IEEE Computer Graphics and Applications Special Issue on Information Visualization, 22(1), Jan/Feb 2002.
26. I. Herman and G. Melanon and M.S. Marshall, Graph Visualization and Navigation in Information Visualization: A Survey, IEEE Transactions on Visualization and Computer Graphics, 6(1), 2000
27. Miki Nakagawa, Bamshad Mobasher, A Hybrid Web Personalization Model Based on Site Connectivity, WebKDD workshop of SIGKDD, 2003.
28. B. Johnson and B. Schneiderman, Tree-maps: a space-filling approach to the visualization of hierachical information structures, Proceedings of Information Visualization'91, pp 284-191, IEEE.
29. W. Schroeder, K. Martin and B. Lorensen, The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics, Prentice Hall, 1998. Also http://www.vtk.org.
30. John Punin, Mukkai Krishnamoorthy, Mohammed J. Zaki, LOGML – Log Markup Language for Web Usage Mining , in WebKDD Workshop of SIGKDD 2001.
31. `http://www.cs.rpi.edu/~youssefi/research/vwm`).
32. John Punin, Mukkai Krishnamoorthy, WWWPal System- A System for Analysis and Synthesis of Web Pages, Proc. WebNet 98, 1998.
33. Ivan Herman, Maylis Delest, Guy Melanon, Tree visualisation and navigation clues for information visualisation, Computer Graphics Forum, 1998.
34. M. Spiliopoulou and C. Pohle, Data mining for measuring and improving the success of Web sites, Data Mining and Knowledge Discovery, 5:85-14, 2001.
35. Mohammed J. Zaki Efficiently Mining Trees in a Forest, SIGKDD, 2002.
36. Mohammed J. Zaki, SPADE: An Efficient Algorithm for Mining Frequent Sequences, Machine Learning Journal, 2001

Towards Simple, Easy-to-Understand, yet Accurate Classifiers

Doina Caragea[1], Dianne Cook[2], and Vasant Honavar[1]

[1] Artificial Intelligence Research Laboratory,
Department of Computer Science,
Iowa State University,
226 Atanasoff Hall, Ames, IA 50011, USA
{dcaragea, honavar}@cs.iastate.edu
http://www.cs.iastate.edu/ honavar/aigroup.html
[2] Virtual Reality Applications Center,
Department of Statistics,
Iowa State University,
325 Snedecor Hall, Ames, IA 50011, USA
dicook@iastate.edu,
http://www.public.iastate.edu/ dicook/Limn/index.html

Abstract. For better interpretability of class structure in data we want to use Support Vector Machines (SVM) for exploratory data analysis. This is easier to do when data is linearly separable. However, when data is not linearly separable, the results of SVM classifiers with non-linear kernels are more difficult to understand, partly due to the mapping to a higher dimensional space. In this paper, we design a method for weighting linear support vector machine classifiers or random hyperplanes, to obtain a classifier whose accuracy is comparable to the accuracy of a non-linear support vector machine classifier, and whose results can be readily visualized. We conduct a simulation study to examine how our weighted linear classifiers behave in the presence of known structure, compared to support vector machines with non-linear kernels. We describe the results of our simulation study on 2-class non-linearly separable data, where the data sets are generated by varying the shape of the clusters, and varying the number of variables. The results show that the weighted linear classifiers might perform well compared to the non-linear support vector machine classifiers, and they are more readily interpretable than the non-linear classifiers. The normals to the separating hyperplanes are viewed using rotations or tours of the data.

1 Introduction

For many data mining tasks understanding a classification rule is as important as the accuracy of the rule itself. Going beyond the predictive accuracy to gain an understanding of the way the classifier works, provides an analyst with a deeper understanding of the processes leading to cluster structure.

Support vector machines (SVMs) [9] offer a theoretically well-founded approach to automated learning of pattern classifiers for mining labeled data sets.

The mechanism for defining the classification is intuitively appealing: maximum margin between classes, that is, put the separating hyperplane at the biggest gap between the two groups. SVMs have been shown to build accurate rules in complex classification problems, but the results often provide little insight into the class structure in the data.

The primary goal of our investigation is to determine ways in which we might de-tangle support vector machines and use visualization to obtain an understanding about the distribution of the classes in the data space. That is, to go beyond predictive accuracy to determine the nature of the separation between classes, and provide insights into the nature of the processes generating the cluster structure.

With SVM, we can understand the nature of the boundaries between classes when the data is linearly separable, by looking at the separating hyperplane or the normal to this hyperplane. In high-dimensional spaces the normal to the separating hyperplane is visualized using tour methods [4, 5]. When data is non-linearly separable, the visualization is more difficult because the boundaries between classes are non-linear.

In this paper, we have designed a classifier based on weighting linear support vector machines or random hyperplanes, whose results can be readily visualized. The proposed method outputs classifiers that have accuracy comparable to the accuracy of the support vector machines using non-linear kernels. We have shown this by conducting a simulation study on artificially generated data sets, three 2-dimensional data sets and three 5-dimensional data sets. We expect that this approach will be useful for data with a relatively low number of variables (features), less than 20, where it is possible to visually explore the space.

The paper is organized as follows: Section 2 gives more details about support vector machines and visualization methods. Section 3 introduces the weighting scheme and shows how to use it to generate weighted linear support vector machines classifiers and weighted random hyperplanes classifiers. Section 4 compares the results of the classifiers obtained using the weighting scheme to the results of the SVM classifiers with non-linear kernels. We conclude in Section 5 with a summary and directions for future work.

2 Support Vector Machines and Visualization

Let $\mathcal{E} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_l, y_l)\}$, where $\mathbf{x}_i \in \mathcal{R}^p$ and $y_i \in \{-1, 1\}$ be a set of training examples for a 2-category classifier. Suppose the training data is *linearly separable*. Then it is possible to find a hyperplane that partitions the p-dimensional pattern space into two half-spaces R^+ and R^-. The set of such hyperplanes (the solution space) is given by $f_{\mathbf{w},b}(\mathbf{x}) = \text{sign}(\mathbf{w} \cdot \mathbf{x} + b)$. SVM selects among the hyperplanes that correctly classify the training set, the one that minimizes $\|\mathbf{w}\|^2$, which is the same as the hyperplane for which the *margin* of separation between the two classes, measured along a line perpendicular to the hyperplane, is maximized.

If the goal of the classification problem is to find a linear classifier for a non-separable training set, a new set of weights, called *slack weights* (measuring the extent to which the constraints are violated) can be introduced. In this case the margin is maximized, paying a penalty proportional to the cost of constraint violation. The decision function is similar to the one for the linearly separable problem.

If the training examples are not linearly separable, the SVM works by mapping the training set into a higher dimensional *feature* space using an appropriate kernel function ψ. Therefore, the problem can be solved using linear decision surfaces in the higher dimensional space. Any consistent training set (i.e., one in which no instance is assigned more than one class label) can be made separable with an appropriate choice of a feature space of a sufficiently high dimensionality. However, in general, this can cause the learning algorithm to overfit the training data resulting in poor generalization.

The output of the SVM classifier, using linear kernels, is the normal to the separating hyperplane, which is itself a linear combination of the data. Thus the natural way to examine the result is to look at the data projected into this direction. It may also be interesting to explore the neighborhood of this projection by changing the coefficients to the projection. This is available in a visualization technique called a manually-controlled tour.

Generally, tours display projections of variables, $\mathbf{x}'\mathbf{A}$ where $\mathbf{A}$ is a $p \times d(< p)$-dimensional projection matrix. The columns of A are orthonormal. Often $d = 2$ because the display space on a computer screen is 2-dimensional, but it can be 1 or 3, or any value between 1 and p. The earliest form of the tour presented the data in a continuous movie-like manner [1], but recent developments have provided guided tours [5] and manually controlled tours [4]. Here we are going to use a $d = 2$-dimensional manually-controlled tour to recreate the separating boundary between two groups in the data space. Figure 1 illustrates the tour approach.

3 Weighted Linear SVMs

Our goal is to gain insights into the operation of the SVM algorithm (i.e., the way it chooses the separating hyperplanes). Since linear classifiers can be easily visualized, we design a method for combining linear classifiers generated using the SVM algorithm on subsamples of data, then use this method to classify non-linearly separable data. We assume that a training set E of size l is given and N hyperplanes $h_1, h_2, \cdots, h_N$ are generated using the SVM algorithm. To generate a hyperplane h_i, we randomly divide the training set into two subsets. One subset is used for generating the linear classifier and the other is used for estimating its error, $error(h_i)$.

If x is a new data example that needs to be classified, we estimate the "probability" $P(+1|x)$ that x belongs to the positive class, and the "probability" $P(-1|x)$ that x belongs to the negative class, as follows, and then we assign to

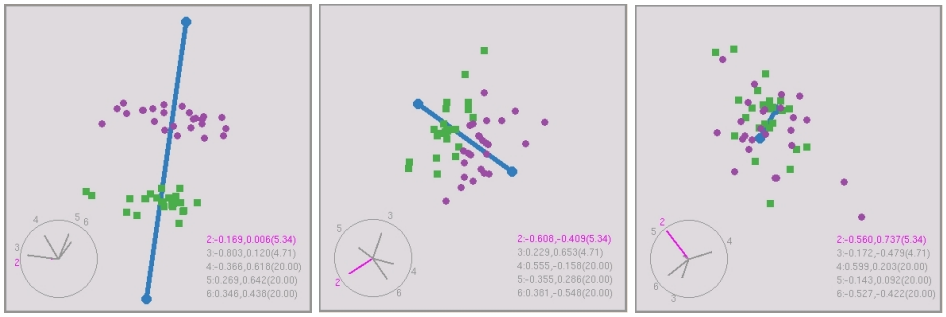


Fig. 1. Three tour projections of a (p=)5-dimensional data set where the two groups are readily separated. The normal to the separating hyperplane is shown as a vector in the space. The separating hyperplane is (p-1=)4-dimensional making it more awkward to view than the normal that defines it. The axes at bottom left and right represent the projection coordinates. The left plot shows a projection where the groups are well-separated, and the plot at right shows a projection where they are less separated. The magnitude of the projection coefficients indicate variable importance, the larger the coefficient - in the direction of separation - the more important the variable. For example, the left plot shows $\frac{-0.169}{5.34}V_1 - \frac{0.803}{4.71}V_2 - \frac{0.366}{20}V_3 + \frac{0.269}{20}V_4 + \frac{0.346}{20}V_5$ horizontally, and $\frac{0.006}{5.34}V_1 + \frac{0.120}{4.71}V_2 + \frac{0.618}{20}V_3 + \frac{0.642}{20}V_4 + \frac{0.438}{20}V_5$ vertically. The separation between groups is in the vertical direction which is primarily variables 4,5,6. The normal is effectively the direction that yields the predicted classes when the data is projected in this direction. If we were to run the SVM on multiple samples of this data we would get a set of separating hyperplanes and their respective normals. We would expect these to be distributed (like a bundle of sticks) around the normal calculated on all the data. They should be most elongated when the separation between the two groups is greatest, and oriented across the separation.

x the class with larger "probability":

$$P(+1|x) = \sum_{i=1}^{N} P(+1|x, h_i) * 2^{error(h_i)},$$

$$P(-1|x) = \sum_{i=1}^{N} P(-1|x, h_i) * 2^{error(h_i)}.$$

In order to estimate the probabilities $P(+1|x, h_i)$ and $P(-1|x, h_i)$, we use the method described in [6], which is based on binning the training examples according to their scores computed by SVM (i.e., the distances from the points to the separating hyperplane). More exactly, the training examples are ranked according to their scores with respect to a hypothesis h_i and then they are grouped in b subsets of equal size. A new test example x is placed in the corresponding subset j according to the score that h_i assigns to it. The probability that x belongs to the positive class is estimated as the fraction of positive training examples that fall into the subset j, while the probability that x belongs to the negative class is given by the fraction of negative training examples that fall into

j. Our simulation study is meant to show that the weighted linear SVM classifiers described above give results comparable to the results obtained with the SVM classifiers with non-linear kernels, while they are easier to understand and visualize. However, one may ask why we need to use hyperplanes generated with SVM algorithm instead of using randomly generated hyperplanes. The advantage of using the hyperplanes generated by SVM is that they are guided toward a reasonable solution for the given data, while the random hyperplanes can be arbitrarily good or bad. So, in general, we expect that a larger number of random hyperplanes is needed compared to the number of SVM hyperplanes, especially for higher dimensional data. To check this out, we design a similar method as the one described above for randomly generated hyperplanes. We generate N random hyperplanes and we assign scores to the test examples based on the distances from those examples to the random hyperplanes. The same subsamples as in the case of SVM hyperplanes are used for estimating the probabilities above and the error of a hyperplane. The results of the two methods are compared. We also compare them with the non-linear SVM classifier results.

4 Examining SVM Behavior Through Simulation

For the experiments in this paper, we used SVM^{light}3.50 [8] implementation (`svmlight.joachims.org`) of SVM algorithm. The parameters used were chosen by trying various values and choosing the ones that gave the best results in terms of accuracy. We also used the tour methods available in the data visualization package GGobi (`www.ggobi.org`).

The data sets used were generated by varying the shape of clusters and the number of variables. Three of the data sets (plotted in Figure 2) contain 2 classes, and 2 variables, and the other three contain 2 classes and 5 variables. There are 500 instances in each data set.

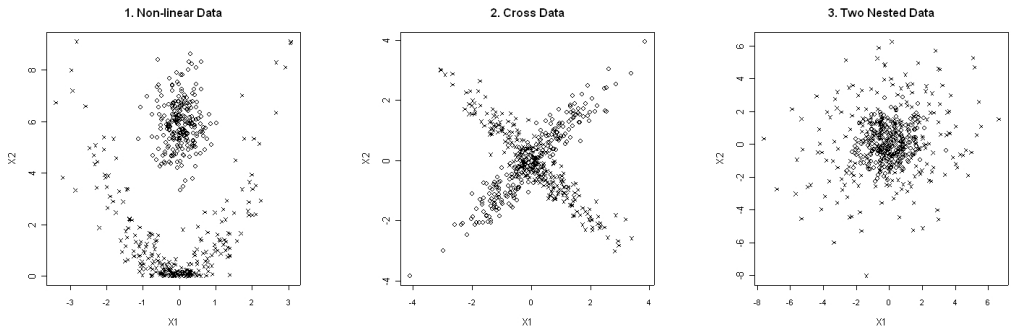


Fig. 2. Simulated Data Sets: 2-Dimensional data sets used to examine the SVM behavior under different distributions

Data set 1 contains two non-linearly separable clusters. It is motivated by previous observations [3]. Data sets 2 and 3 exhibit classes that are linearly non separable. One data set contains two ellipses that intersect in their centers (normals with the same mean). The other contains one cluster nested inside the other (generated from normal distributions with the same mean, but different variance). Data sets 4, 5 have the same first two variables and then three additional variables containing noise. Data set 6 is similar to the third data set with two nested spherical clusters but in 5 dimensions. It is very difficult to classify them using linear boundaries.

We perform two sets of experiments for all three methods we want to compare (i.e., non-linear SVM, weighted linear SVM and weighted random hyperplanes). The first set of experiments is meant to compare the performance of the three methods, while the second set is meant to show how we can visualize the results of the methods used.

For each run of the three methods we compare, we randomly divide the data set into a training set containing 400 example and a test set containing 100 examples. The non-linear SVM uses these sets as training and test sets. For the weighted linear SVM and the random hyperplanes methods, we split the training set further into a set containing 300 examples, (used to train the individual linear SVMs) and a set containing 100 examples (used to estimate the probabilities corresponding to the test examples, and also the error of each linear classifier generated). The number of bins is 5, so the size of each bin is 20. The same test set as in the case of non-linear SVM is used for the weighted linear SVM and weighted random hyperplanes methods. In what follows we denote the non-linear kernel SVM by Non-Lin SVM, weighted linear SVM hyperplanes by Lin SVM, and the weighted random hyperplanes by Rnd Hyp.

What do we expect to see? For the 5-dimensional non-linear data (data set 4) we would expect to see the best separating hyperplanes similar to the solution for the 2-dimensional non-linear data. That is, the normals would point across the separation in the first two variables. We'd expect that these separating hyperplanes would have the highest weights. For the 5-dimensional cross data (data set 5) we'd expect that results to be similar to the 2-dimensional cross data. For the 5-dimensional nested data (data set 6) there is no good single solution, the normals will be oriented in all directions.

4.1 Results

Simulation 1: The goal of the first set of experiments is to show how the three methods used in the paper perform compared to each other. To do that, we ran each of the three methods 100 times and recorded the accuracy on the test set. The average test accuracy over 100 runs for the six simulated data sets is shown in the Table 1.

We notice that in general the non-linear kernel SVM performs better than the other two methods, but the difference is not always significant. The use of the kernel brings a gain over the other two linear methods in terms of accuracy,

however it pays off in terms of the understanding of the separation between classes.

The performance of the Lin SVM and Rnd Hyp is comparable for Cross and Two Nested data sets. Rnd Hyp performs better in some cases due to the fact that there is a good chance that the Rnd Hyp finds hyperplanes that are not very accurate by themselves, but they contribute to the global accuracy when they are used in combination with other hyperplanes (in case that the number of hyperplanes generated is large enough). This is not the case with the hyperplanes obtained with SVM that are biased toward the best hyperplane given the data.

Table 1. The average test accuracy over 100 runs for the four simulated data sets

Data Set	Non-Lin SVM	Lin-SVM	Rnd Hyp
Non-Linear	0.9989	0.8915	0.9215
Cross	0.8509	0.8281	0.8201
Two Nested	0.7654	0.7044	0.7192
5d Non-Linear	0.9947	0.8828	0.8728
5d Cross	0.8038	0.8382	0.7500
5d Nested	0.8200	0.6795	0.7707

Simulation 2: The goal of the second set of experiments is to show how we can visualize the results of the methods designed. We performed 8 runs for each 2-dimensional data set, using h=2, 5, 10, 20, 25, 50, 75, 100 hyperplanes for each run, respectively. The graphical results for h=100 for the 2-dimenssional data sets are shown in Figures 3, 4, 5 for Non Linear data set, Cross data set and Two Nested data set, respectively. The training and test sets used in each run are the same for the three methods, but differ from one run to another. We generated a grid over each of the 2-dimensional data sets. The grid color corresponds to the predicted class, and thus we can see the boundaries of the prediction. The lines are the separating hyperplanes, not the normals for this simple 2-dimensional data. The (Left) plot in each of these figures shows the results for the non-linear SVM. The (Middle) plot shows the results of the Lin-SVM. The hyperplanes' width is proportional to their weights. You can see that the separating hyperplanes are concentrated in a very local region, and that together they do produce a slightly non-linear prediction. The (Right) plot shows the results of the Rnd Hyp. The first 10 most accurate random hyperplanes are shown (if we plot all 100 hyperplanes the separation boundaries cannot be easily seen). The hyperplanes' width is also proportional to their weights. You can see that these random hyperplanes together produce a more non-linear prediction that matches the class structure than the Lin-SVM method.

Figure 6 shows the results for the Non Linear 5-dimensional data set (data set 4): (Left column) SVM results, (right column) random hyperplanes. There are some big differences between the results for hyperplanes generated by SVM and for random hyperplanes. The distribution of weights (top row of plots) is

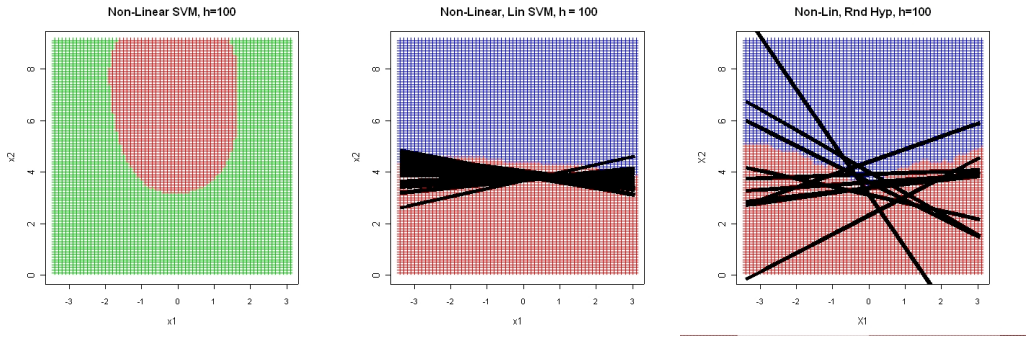


Fig. 3. Visual Results for the 2-D Non-Lin Data: In these plots there is multiple information: (1) the grid color corresponds to the predicted class, and thus we can see the boundaries of the prediction; (2) the lines are the separating hyperplanes, not the normals for this simple 2-dimensional data. (Left) The results for the non-linear SVM. (Middle) The results of the Lin-SVM. The hyperplanes' width is proportional to their weights. You can see that the separating hyperplanes are concentrated in a very local region, and that together they do produce a slightly non-linear prediction. (Right) The results of the Rnd Hyp (the first 10 most accurate random hyperplanes are shown). The hyperplanes' width is also proportional to their weights. You can see that these random hyperplanes together produce a more non-linear prediction that matches the class structure than the Lin-SVM method.

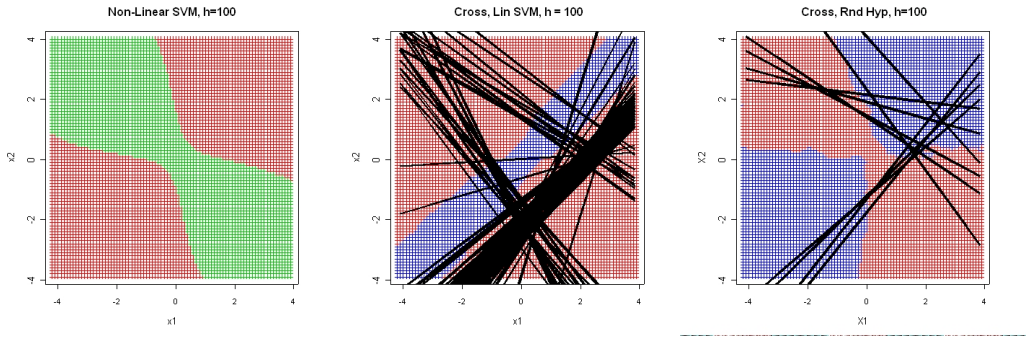


Fig. 4. Visual Results for the 2-D Cross Data: (Left) The results for the Non-Lin SVM. (Middle) The results of the Lin SVM on the grid and the SVM hyperplanes used to derive these results. (Right) The results of the Rnd Hyp on the grid and the random hyperplanes used to derive these results (the 10 most accurate hyperplanes). Again, you can see that these random hyperplanes together produce a more non-linear prediction that matches the class structure than the Lin-SVM method.

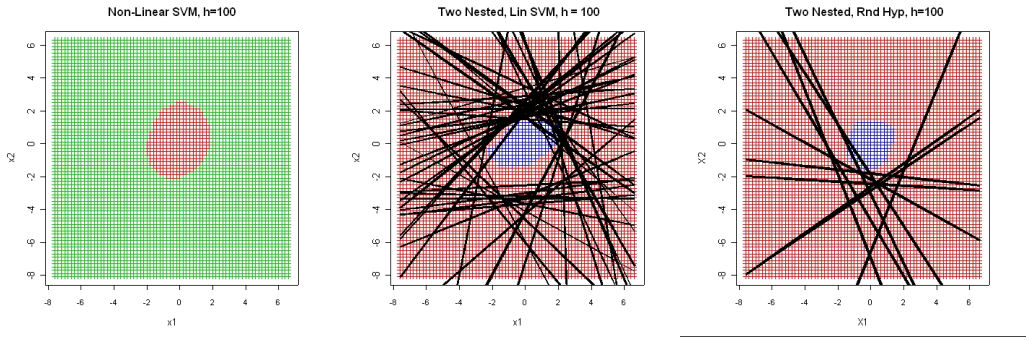


Fig. 5. Visual Results for the 2-D Two Nested Data: (Left) The results for the Non-Lin SVM. (Middle) The results of the Lin-SVM on the grid and the SVM hyperplanes used to derive these results. (Right) The results of the Rnd Hyp on the grid and the random hyperplanes used to derive these results (the 10 most accurate hyperplanes).

quite different: the weights for SVM are very similar for all hyperplanes so all hyperplanes are almost equally as good, but for the random hyperplanes the distribution of weights runs from 0 to 1, with closer to 1 being the hyperplanes with better separating power. The bottom two rows show the plots of the first two variables and one tour projection with the normals to the separating hyperplanes represented as vectors. The hyperplanes from SVM are also much less varied than for the random hyperplanes, which is as expected because SVM will produce the best separating hyperplane. For the random hyperplanes, the best are more varied than those produced by SVM, but they angle around the nonlinear boundary between the two groups. It would be possible to view the prediction areas by generating a grid over the 5 variables and predicting the class at each grid point much like was done for the 2-dimensional data.

Figure 7 shows the graphical results for the Nested 5-dimensional data (data set 6). The appropriate boundary separating the two groups is a sphere in 5 dimensions, thus the normals to the separating hyperplanes should reflect this, they should be oriented randomly in any direction. This is observed for the best random hyperplanes but not the SVM hyperplanes. It looks like there is some systematic bias to the SVM implementation. The other observation to note is that the distribution of weights is similar in both methods, and this reflects the fact that every plane is essentially as equally as good as any other plane.

5 Summary and Discussion

This paper presented new approaches to understanding class structure in high-dimensions using an ensemble algorithm building on linear separators and graphics for viewing the high-dimensional space. These approaches allow the analyst

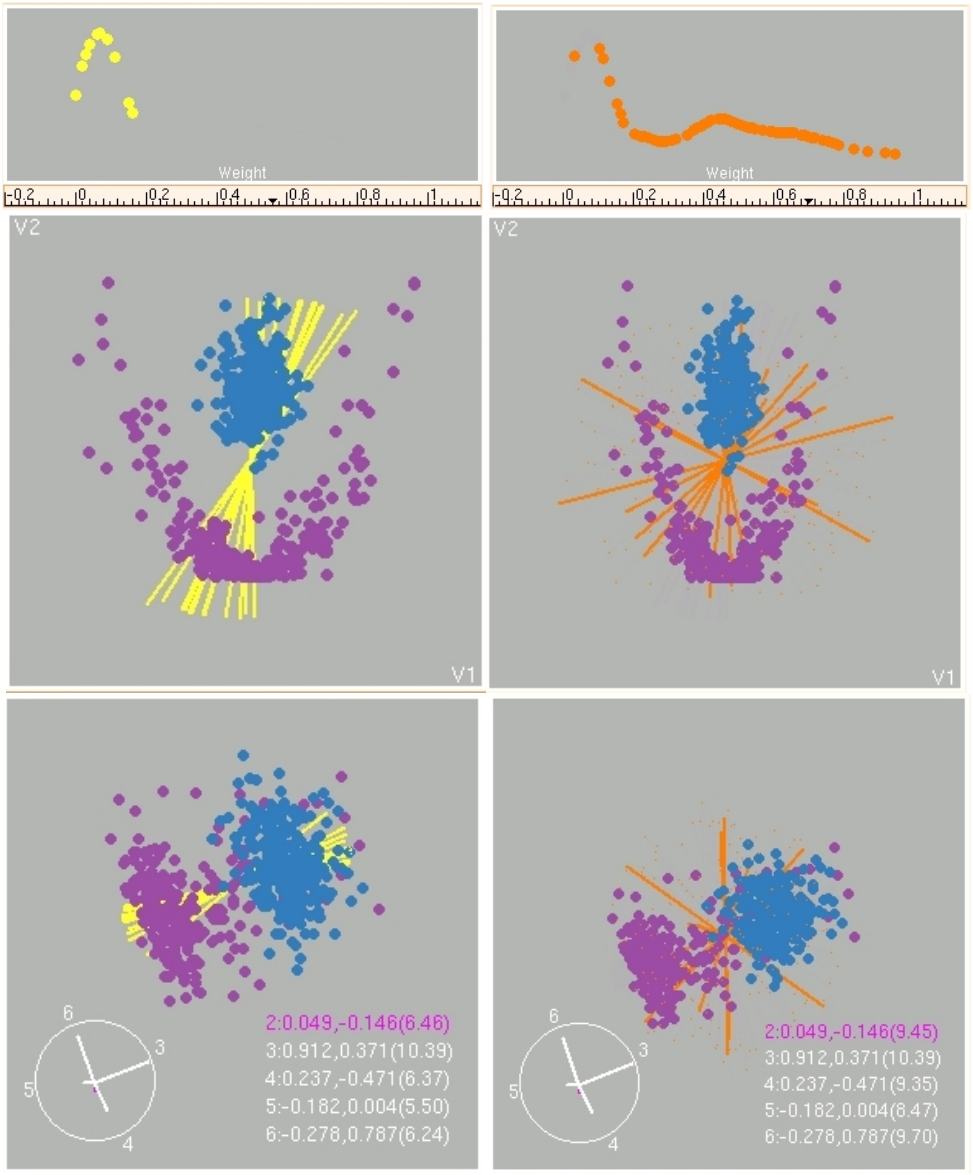


Fig. 6. Visual Results for the 5-D Non-Linear Data: (Left column) SVM Lin results, (Right column) Rnd Hyp. There are some big differences between the results for hyperplanes generated by SVM and for random hyperplanes. The distribution of weights (top row of plots) is quite different: the weights for SVM are very similar for all hyperplanes so all hyperplanes are almost equally as good, but for the random hyperplanes the distribution of weights runs from 0 to 1, with closer to 1 being the hyperplanes with better separating power. The bottom two rows show the plots of the first two variables and one tour projection with the normals to the separating hyperplanes represented as vectors. The hyperplanes from SVM are also much less varied than for the random hyperplanes, which is as expected because SVM will produce the best separating hyperplane. For the random hyperplanes, the best are more varied than those produced by SVM, but they angle around the nonlinear boundary between the two groups.

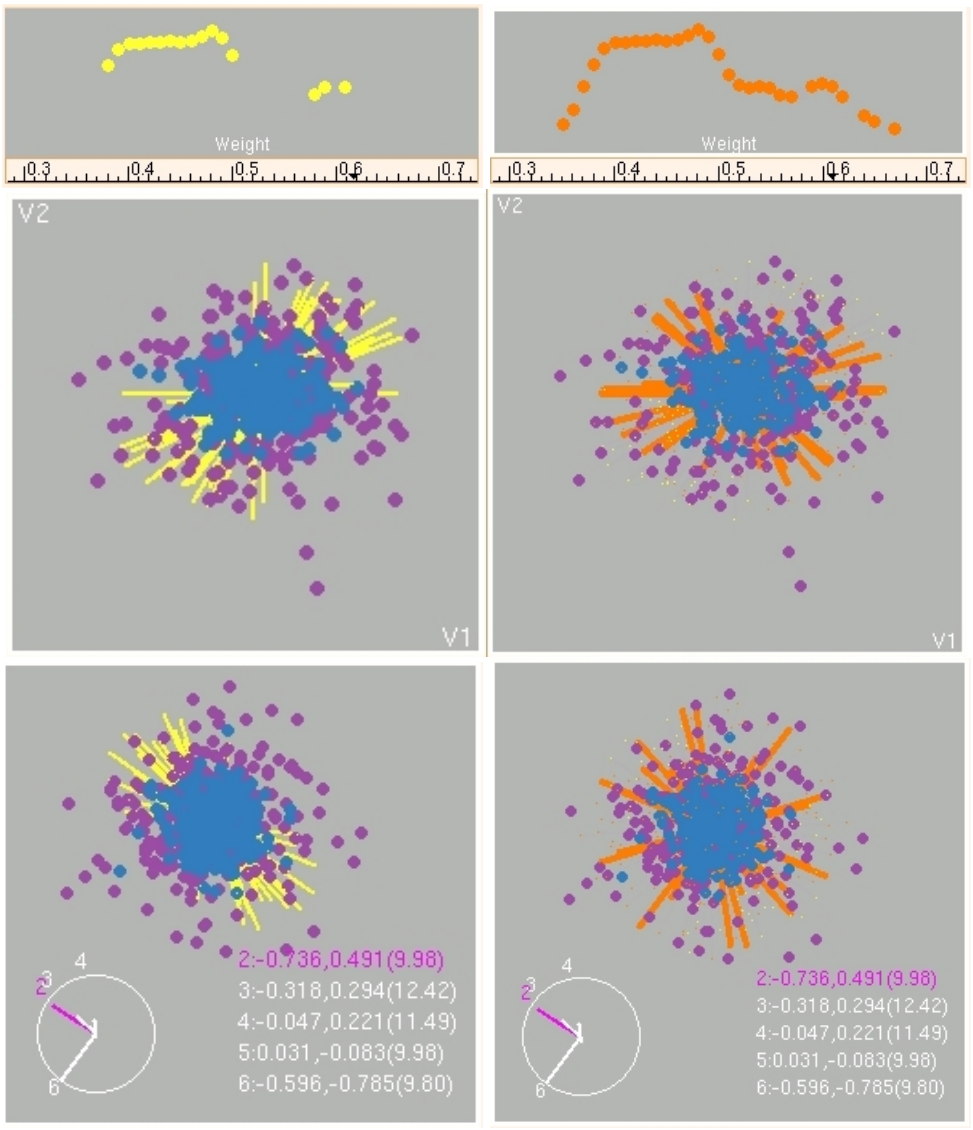


Fig. 7. Visual Results for the 5-D Nested Data: (Left column) Lin-SVM results, (Right column) Rnd Hyp. The most obvious pattern is that the SVM separating hyperplanes are not so varied, and there is no reason for this to happen. It suggests that there is some systematic bias in the implementation of the SVM algorithm. The random hyperplanes that have the highest weights are spread throughout all possible directions, which is as we'd expect. There is no one best solution for this data because the true boundary between the two groups should be a sphere in 5 dimensions.

to gain more insight into the nature of the cluster structure in relation to class, and understand the behavior of classification algorithms. The most dramatic finding we uncovered is that the SVM algorithm we used seems to have a systematic bias in its choice of separating hyperplanes. This was most obvious in the 5-dimensional nested data where every direction should be equally likely to be chosen as the best separating hyperplane, but the planes concentrated in a small area of the 5-dimensional space. So this says that the implementation of SVM we used here would not be a good candidate to build into an ensemble classifier.

The new algorithm for combining linear classifiers outperforms SVM using non-linear kernels in some cases. Even in cases where SVM with non-linear kernel outperforms the weighted combination of linear classifiers, the loss in accuracy is compensated by the better understanding of the results available through visualization.

More simulations are needed to understand better the dependence of the accuracy of the proposed methods on the number of hyperplanes generated, as well as the correlation between the number of hyperplanes needed for good accuracy and the number of dimensions of the data. The number of hyperplanes needed by Rnd Hyp to match the performance of Lin-SVM increases with the dimensionality of the space, so in general this can't be a good approach.

Various weighting schemes for combining classifiers exist in the literature [2, 7]. Many of them are similar in spirit to the scheme we used. However, our purpose was not to design another weighting scheme that would outperform existing ensemble methods, but to design/choose one particular scheme that can be easily visualized. The same visualization methods can be applied for any other weighting scheme based on linear classifier.

In conclusion, the results we present in this paper represent a first step toward achieving our goal of constructing simple and easy to understand SVM classifiers, that can be used for exploratory data analysis.

Acknowledgements

This work has been supported in part by grants from the National Science Foundation (#9982341, #9972653), the Carver Foundation, Pioneer Hi-Bred, Inc., John Deere Foundation, the ISU Graduate College, and the IBM.

References

1. D. Asimov. The Grand Tour: A Tool for Viewing Multidimensional Data. *SIAM Journal of Scientific and Statistical Computing*, 6(1):128–143, 1985.
2. L. Breiman. Bagging Predictors, In Machine Learning, Vol. 24, No. 2, 1996.
3. D. Caragea, D. Cook and V. Honavar. Gaining Insights into Support Vector Machine Pattern Classifiers Using Projection-Based Tour Methods, In Proceedings of the KDD Conference, San Francisco, CA, 2001.

4. D. Cook and A. Buja. Manual Controls For High-Dimensional Data Projections. *Journal of Computational and Graphical Statistics*, 6(4):464–480, 1997. Also see `www.public.iastate.edu/~dicook/research/papers/manip.html`.
5. D. Cook, A. Buja, J. Cabrera, and C. Hurley. Grand Tour and Projection Pursuit. *Journal of Comp. and Graphical Statistics*, 4(3):155–172, 1995.
6. J. Drish. Obtaining Calibrated Probability Estimates from Support Vector Machines. San Diego, 2001.
7. T. Dietterich. Ensemble Methods in Machine Learning. In: *Lecture Notes in Computer Science*, Vol. 1857, 2000.
8. T. Joachims. *Making Large-Scale SVM Learning Practical*. MIT Press, 1999.
9. V. Vapnik. *The Nature of Statistical Learning Theory (Statistics for Engineering and Information Science)*. Springer-Verlag, New York, NY, 1999.

Visualization of Content Information in Networks using GlyphNet

Anne Denton and Paul Juell

Department of Computer Science, North Dakota State University,
Fargo, ND, USA
{Anne.Denton,Paul.Juell}@ndsu.nodak.edu

Abstract. Visualization of information on a graph has two aspects that are equally important in many settings: Visualization of graph connectivity and visualization of node information. We introduce GlyphNet, a tool that displays node-related information graphically, using small icons or glyphs. Our goal is to assist researchers who are applying data mining techniques to relational data, and have a need to identify patterns that involve node attribute values of interconnected nodes. GlyphNet represents node data as glyphs, analogously to the symbols on a weather map. Rather than placing glyphs in a spatial context, as is the case for weather symbols, GlyphNet displays node information in its graph context. We demonstrate the use of GlyphNet for the example of a data mining task that involves yeast gene and protein properties within the corresponding protein-protein interaction network.

1 Introduction

Information visualization on a graph is important for many subject areas. Social networks, the link structure of the World Wide Web, and biological networks, such as protein-protein interaction graphs and biochemical pathways, are examples of data that are commonly represented as graphs. Many techniques exist to display such graphs [1]. Most of them do, however, focus on connectivity. Node data, if displayed at all, is usually included in textual form, such as in class diagrams and entity-relationship diagrams that are common in software engineering. Computer-based tools often also provide probe functionality to display node details in a separate window, based on selections made by the user [2]. Although this is an efficient solution to the problem of retrieving information from a network of nodes, it is not suitable to typical data mining tasks such as the identification of interesting patterns.

1.1 Data Mining Uses of Traditional Graph Visualization Tools

Visual data mining can be seen as a hypothesis generation process [3]. We will, therefore, now look at possible hypotheses that can be generated from different graph visualization tools. Traditional graph drawing techniques that display nothing but connectivity allow investigating hypotheses on the distribution of edges. Many

interesting results can come from such studies, e.g., the identification of different types of networks [4], including scale-free networks and random networks. Graph visualization tools with probe functionality allow generating hypotheses that go beyond connectivity alone. A particular node can be viewed within the nodes in its network neighborhood, which can, for example, allow generating hypothesis regarding relationships between edge distribution and node importance. Google's PageRank [5] algorithm that relates the importance of a Web page to the distribution of incoming links would be accessible to such analysis. Much work is still being done to improve on importance measures [6], and a graph visualization tool with probe functionality could be used productively in this context.

Much current work in the area of data mining involving relational data does, however, go beyond the issues addressed so far [7]. The term relational data refers to data in which a relationship exists between data records that can be represented by a graph. Typical questions of interest are how relational neighbors can or should be used in classification and clustering [8]. Such questions require access to node data of not only one node but of all nodes for which a hypothesis is to be made. In very small graphs one may try to resolve the problem by including a textual representation of the node data into the graph, as is done in class diagrams and entity relationship diagrams. When this strategy is used many problems of textual representations recur that were supposed to be addressed by the visualization: Textual information has to be interpreted and patterns are, therefore, hard to see. Text also uses up a significant amount of space, which limits the number of nodes that can be displayed simultaneously.

1.2 Data Mining Uses of GlyphNet

In this paper we introduce GlyphNet, a tool to visualize node data through glyphs, small graphical representations of the node attribute values. We thereby provide a valuable aid for the visual exploration of data that is characterized by graph connectivity as well as node data. Our tool is intended to generate hypotheses that involve attributes in multiple, connected nodes. Hypotheses may then be validated through numerical techniques. Why couldn't one find all interesting patterns numerically? For relational data the search space of interesting patterns is very large. Not only is it possible that attributes within one record affect each other, rather all the attributes of a neighbor, a neighbor's neighbor etc. can affect the properties of a given node. Although it may be hard to specify how an interesting pattern should look like, a human may still be able to detect it. There is, furthermore, still a lack of data mining techniques that are suitable to the relational setting, although some progress has been made recently [7]. Many traditional data mining techniques exist that were developed for data in a simple tabular form. Making use of the richness and sophistication of these techniques in a relational setting is possible if the characteristics of the data can be handled through preprocessing techniques, in particular, feature extraction. Feature extraction refers to the process of identifying relevant patterns that can then be included into a tabular format. A relevant feature in the relational setting could be the number of neighbors of a node for which a particular Boolean property is "true".

Section 4 discusses an example use of GlyphNet in which this type of feature extraction is used in conjunction with a classification algorithm [9].

The paper is organized as follows: Section 2 summarizes existing graph visualization techniques, and discusses the concepts and common uses of glyphs. Section 3 introduces GlyphNet and explains how graph visualization and attribute representation through glyphs are combined in this tool. Section 4 gives an example of a use of GlyphNet as a visual data mining tool for the purpose of identifying candidates for feature extraction in a classification task. Section 5 concludes the paper.

2 Previous Work

GlyphNet borrows from two existing concepts, graph visualization as discussed in section 2.1 and the concept of glyphs as discussed in section 2.2.

2.1 Graph Visualization

Graph drawing is an old topic and many techniques exist for its purpose [10]. One main goal in traditional graph drawing, planarity, consists in avoiding crossing edges. Other aesthetic rules have been formulated, including the goal that edges should be represented by straight lines, all edges should have the same length, and isomorphic structures should be displayed equivalently. Many layouts have been described in the literature, including top-down layouts like the Reingold and Tilford layout [11], the H-tree layout [12], and the balloon tree layout [13]. The layout used in GlyphNet is based on a Force-Directed Method [14], in which nodes are modeled as physical bodies with the edges representing springs that hold them together. The corresponding mathematical optimization problem is known to produce well-balanced graphs with a small number of edge crossings [15]. Time complexity of force-directed algorithms makes them mainly suitable to small graphs, which is acceptable in a graph navigation system that only displays a small part of the graph at any one time.

Many current problems involve graphs that are too large to display in such a way that all nodes can be simultaneously viewed with sufficient detail. Traditional techniques are therefore often combined with graph navigation features that allow displaying only those nodes that are within a predefined distance of a central node. Alternatively the plane may be distorted using either hyperbolic geometry [16] or a fisheye view [17] to provide detail in the vicinity of a selected node. More distant nodes are displayed with less detail and serve the purpose of providing a context. In either case, navigation is achieved by letting the user select a central node, and calculating a new layout for each new choice. GlyphNet uses the former strategy of displaying a limited number of nodes. This avoids confusion that could come from distorting glyph information.

Adding node information to graph layouts can be done in several ways. A common strategy consists in representing nodes through text boxes as in class diagrams and entity relationship diagrams. In this paper we will focus on techniques

that add node information in graphical form, since they are most closely related to GlyphNet. Some existing approaches represent one or two items of information in graphical form using node size and/or node color. These approaches commonly use textual information as well as possibly probe capability to represent the majority of node content. An example of a tree visualization that uses node size to represent one item of information in graphical form is the SGI File System Navigator [1] in which file size is represented as the size of buildings in the virtual landscape. Other tree visualizations represent two items of information. A cone tree is a two-dimensional view of a three-dimensional representation of a tree [18], for which both node size and color may be used to represent node data [19]. Tree maps have leaf nodes laid out as rectangles in a plane [20]. Internal nodes are identified either through choice of a common color for lower level nodes or through boundaries surrounding lower level nodes. In the latter case color may be used to represent node information as well as size of the rectangles. In a practical application of a tree map, a map visualizing stock performance [21], color represents performance of stock, and size represents volume. Both cone trees and tree maps are limited to trees and cannot be generalized to graphs.

In section 2.2 we discuss glyphs as a way of representing multiple items of information. A concept that is related to glyphs is used in graph clustering [22]. Complex graph structures can be made accessible by grouping related nodes into clusters. These clusters may then be represented in a summarized fashion at a higher level. The process may be iterated to result in a hierarchical structure of graph clusterings. In some cases [22] a polygon is used to represent each lower level cluster at a higher level. The shape of polygon is chosen such that it matches the shape of the lower-level cluster by some criteria. This use of glyphs differs from ours in its goal of representing connectivity information only. A summarized view of connectivity information can assist in understanding the structural properties of a graph but does not solve the problem of representing independent node information.

2.2 Visualization of Attributes Using Glyphs

A glyph is a graphical object used to display one or more dimensions of information [23]. Each glyph represents one record of data. Many or all records can be displayed simultaneously. Interesting properties of the data can then be identified as graphical patterns within the glyph without a need to refer to the actual data. Relationships between different records can be explored through comparison of different glyphs. Glyphs are designed to have several parts that can be modified individually. Each part is used to display an attribute or dimension of the information. Well known types of multidimensional glyphs are cartoon faces that use one variable to select the eyes, another to select the nose and so on [24]. These are commonly known as Chernoff faces. A motivation for using Chernoff faces is that human viewers are particularly familiar with facial features. Frequent attribute combinations can thereby be identified as a particular kind of face.

Another well-known data representation is a star glyph [25]. Star glyphs arrange attributes in a circular fashion with scales for each variable extending from a center outward. Points on neighboring scales are connected. This produces a closed pattern

that allows seeing proportional relationships between variables [26]. See Figure 1, left side, for an example of a star glyph. It is important for the usefulness of this design to place attributes in such a way that within most records attributes with a large value tend to be neighbors of attributes with a small value. This results in a particularly star-like shape. Based on guidance from Tufte [27] the display should be as simple and clean as possible, so we do not show the scaling marks or other items, just the data values.

So far, we have discussed differences between glyph patterns but have not yet looked at differences in how glyphs may be arranged. In some but not all cases the arrangement of glyphs on the plane may come naturally. Weather maps represent information that has been collected from particular spatial locations, and weather glyphs are therefore represented within spatial dimensions. Sometimes the information collection may be inherently one-dimensional, such when drilling for raw materials [25]. Finally there may be no "natural" dimensions for the organization of glyphs at all. In that case placement can be chosen to be convenient for the data mining process, such as ordering by attributes of the represented records. In our case the glyph placement is determined by the graph layout, leading to additional challenges. We will now discuss how GlyphNet combines and modifies existing techniques to allow visual data mining of content information on a network.

3 Description of GlyphNet

The design of GlyphNet is based on public domain software, Touchgraph [2] that is available from the SourceForge open source software development web site. The goal of the Touchgraph project is to provide a user interface for the exploration of web pages. The graph layout algorithm used in the project is based on a force-directed layout [14], and large graphs are made accessible through navigation [1]. Nodes that exceed a predefined number of edges from a selected central node are hidden. A new central node is selected through mouse click. The subset of nodes that are displayed is automatically adjusted. The maximum distance from the central node can be selected by the user. It is furthermore possible to move a node around by dragging the mouse. The remaining nodes follow according to the force laws that determine the layout.

3.1 Design of Suitable Glyphs for GlyphNet

Glyphs in GlyphNet are loosely based on the star glyph concept [25]. We did, however have to modify the design to avoid confusion that could come from close neighboring nodes or interference between nodes and lines that represent edges. These problems are not commonly encountered with spatially organized data such as weather data, which is usually collected at locations that are sufficiently separate. In graph layout algorithms separation of nodes and equal length of edges are a goal that may not be perfectly satisfied.

We choose to embed each glyph in a circle that clearly identifies the node. The area within the circle is reserved for the node, and edges are hidden behind it. We

found that this concept works well, provided the design of the glyph is adapted to the new setting. It can be seen in Figure 1 (left) that a star glyph does not fill the circle well, and much space is wasted. We therefore designed a glyph shape such that it fills the circle fully if all attributes have their maximum value. The circle is divided evenly into wedges or pie-slices, one for each attribute, with the radius representing the attribute value Figure 1 (right). It should be noted that the area of a wedge increases as the square of the radius. We did not use glyphs for quantitative analysis and chose the scaling of attributes entirely based on how clearly values could be distinguished. For quantitative analysis it would be necessary to take a decision on whether radius or area should be used as a measure of the attribute value and the scaling would have to be chosen accordingly.

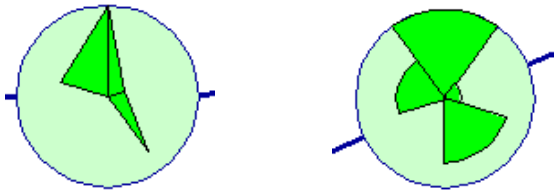


Fig. 1. Comparison between star (left) and wedge glyph (right).

The glyph design offers other degrees of freedom, in particular color, which can be included in the visualization process. For the current purpose the overall color was chosen as the only relevant item of information. Section 4 will discuss the use of color in more detail.

3.2 Adapting Navigation Features to the GlyphNet Setting

Several versions of the Touchgraph software [2] exist, featuring different aspects of the graph exploration task. All versions use textual information to identify nodes. Colors are used to distinguish nodes that have a special role in the navigation algorithm, such as the central node and the node or edge that the mouse currently points to. This is particularly important for implementations that use probe functionality, i.e., display web page content of the currently selected node in a separate window. Additional labels are attached to nodes for which neighboring nodes are omitted because of their distance to the central node. These labels show the number of omitted nodes and have yet another color in the original implementation. This gives an indication as to the intention of color use in previous implementations. Following the goal of simplifying navigation in a network of web pages, color has been exclusively used to simplify navigational tasks and summarizing structural properties of hidden parts of the graph. We will see later how the use of colors for navigational purposes had to be limited in GlyphNet so as to minimize confusion with data mining goals.

GlyphNet represents node information in graphical form for all nodes within the displayed range. Figure 2 shows a snapshot of GlyphNet for a graph that represents protein-protein interactions in yeast. Node data includes properties of yeast genes and of their corresponding proteins. Details of the underlying biological system will be given in Section 4. In the example in Figure 2 the maximum distance of nodes from the central node is two. Nodes that are further away than two edges are hidden but may be exposed by selecting a different central node or increasing the range of displayed nodes. GlyphNet was derived from other graph visualization tools by replacing node identifiers with glyphs that visualize node attributes through shape and color. Displaying node identifiers as well as glyphs for all nodes was considered too confusing. Node identifiers are, however, displayed when the mouse is placed over a node, following the probe concept. For the current implementation, the probe mechanism is limited to the display of node identifiers, but an extension to a more extensive textual display of node content would be straight-forward.

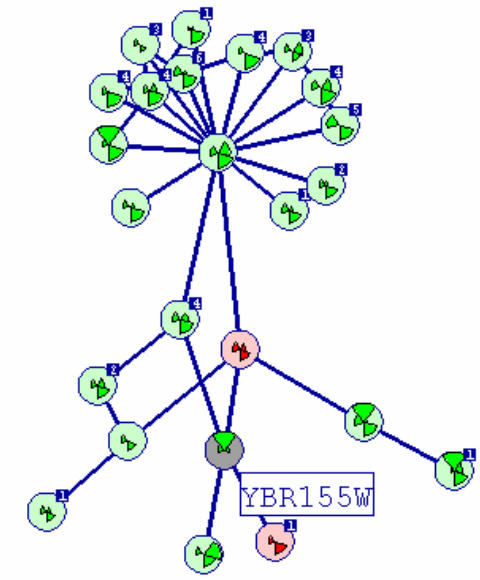


Fig. 2. Snapshot of GlyphNet for yeast gene and protein data within the graph of protein-protein interactions.

We had to reduce the use of color for navigational purposes in order to limit confusion with color choices that represent node data. We chose to use the same, blue, color for all structural information that is static, including the labels that summarize the number of hidden neighbors, such as, for example, the number 1 in the top right corner of the lower red node. If that red node is selected as central node it shows an additional edge that is currently hidden, as well as neighbors to the newly exposed node. Color changes that are related to mouse movement were kept since the user can easily get an undisturbed image by moving the mouse off the graph

display area. In particular, the highlighting of the node to which the mouse currently points was kept. Notice the gray background of the node that is identified through the node label "YBR155W". Maintaining this aspect of the navigation-related color information was possible due to the fact that the color that fills the circle is redundant in our color scheme if the color of wedges is known.

Section 4 gives details on the node and interaction data in this particular example. Six attributes are encoded in each node. One attribute of special interest determines the color, with the three color choices being green, red and yellow. Five other attributes determine the size of the pie slices. The background color within each node is chosen as a brighter version of the pie slice color. More color information could be used but didn't lend itself to the particular data mining task described in section 4. The edges don't contain directional information since the yeast interaction data is undirected. Directional information can easily be included through use of wedges for links. This representation is implemented in the TouchGraph web visualizations [2] that have to distinguish between incoming and outgoing links to web pages.

4 Example Use of GlyphNet for Yeast Data

Visualizations are as good as the insights they can give. For the purpose of visual data mining we are interested in hypotheses regarding patterns within the data. We used GlyphNet as part of a data mining task in bioinformatics.

4.1 Finding Particular Genes

The goal was the identification of yeast genes that were involved in a particular pathway, the Aryl Hydrocarbon Receptor (AHR) pathway. In data mining terms this is considered a classification task: Genes that are part of the AHR pathway were classified as "change", genes that were not part of the pathway, or for which no experiment had been performed, as "no change", and genes that responded to a control experiment as well as to the AHR-related experiment were "control" [28]. The classes were determined in a high-throughput gene deletion experiment. 5000 strains of yeast, each of which was the result of the deletion of a specific gene, were simultaneously treated in micro-array experiment. The overwhelming majority, 97%, of genes belonged to the "no change" class. In Figure 2 "change" genes are identified by their red color and "no change" genes by their green color. Yellow was picked as color for "control" genes and can be seen in Figure 5.

Figure 3 provides a summary of the information that was encoded in each node. The pie slice at the top indicates whether a gene is essential. If a gene is essential the organism will die if that gene is deleted. Gene deletion experiments can therefore not be used to gain information for essential genes. The property "essential" is Boolean, i.e., the pie slice is either filled fully or not at all. The same applies to the "pseudo gene" property in the lower left of the glyph. A pseudo gene is a gene that is known not to produce a functioning protein.

All other properties are numerical. The distance of a gene from the center of the chromosome (top right) was estimated as follows: Genes are numbered sequentially,

starting from the center of the gene, and this number is part of their name, i.e., the gene "YBR155W" in Figure 2 has the number 155, which is taken as distance from the center of the chromosome. The number of amino acids of a gene was taken as its length (top left). Finally we counted the items of information that are given for the protein that a gene produces. Items of information were functions, localizations, and some other quantities that were provided for the KDD cup 2002 [28] and are also available from the Comprehensive Yeast Genome Database at the Munich Information Center for Protein Sequences, MIPS [29].

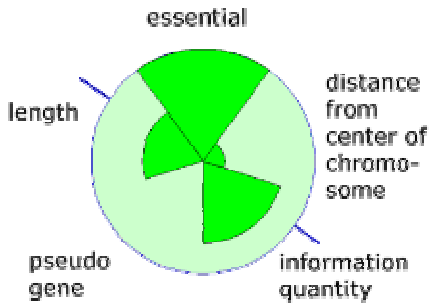


Fig. 3. Meaning of attributes for the glyphs used in Figure 2.

4.2 Use of Neighbor Information

For the evaluation of our visualization tool we will focus on the property "essential" that is displayed at the top of the glyph since it leads to the clearest hypothesis with respect to the AHR pathway property.

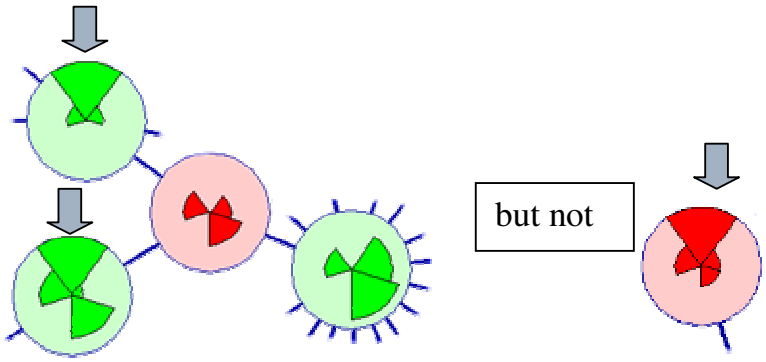


Fig. 4. Left: example of a pattern we identified; right: typical pattern that standard algorithms could find but that cannot be present in the current data.

Looking at Figure 2 it can be seen that the red, “change” gene at the center has two out of four neighbors with the property essential. The “change” gene in the bottom right only has a single neighbor which also has property “essential”. Over the entire range of nodes only four out of a total of 22 “no change” genes have the property “essential”. Note that “change” genes cannot be “essential” since the deletion experiment cannot be performed on “essential” genes. Figure 4 highlights the unusual aspect of our hypothesis.

The navigation capability of GlyphNet allows us to systematically follow patterns that could be interesting. After identifying the potential pattern of an “essential” gene that interacts with a “change” gene we may want to investigate the neighborhood of further “essential” and “change” genes to support our observation. Figure 2 shows one “essential” gene in the top half. Although that gene does not have hidden neighbors we may want to investigate its further neighborhood and therefore make it the central node. Figure 5 shows the result of that selection. It can be seen that at a distance of two hops (next nearest neighbor) there is a “control” gene (yellow) that has also an “essential” neighbor. This suggests that “essential” may be an indication of either “change” or “control”.

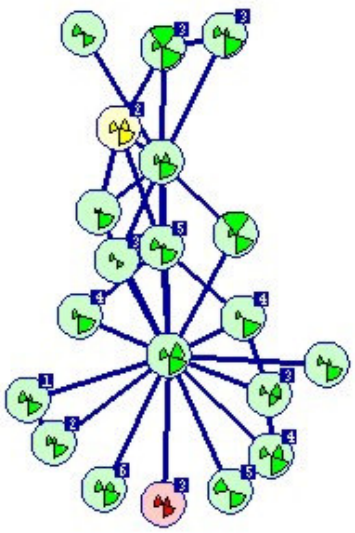


Fig. 5. Snapshot suggesting an impact of “essential” in a neighbor on the prediction of a “control” gene.

It is important to note that such hypotheses do not have to mean anything by themselves since we are only investigating a small part of the full graph of several thousand genes. Visual exploration should be considered a part of a more extensive data mining strategy that includes verification of hypotheses through quantitative numerical methods. A classification algorithm was used for this purpose as described

in [9]. The pattern in Figure 4 entered the classification algorithm in the shape of an additional attribute that represented the number of interacting "essential" genes. A genetic algorithm was used to determine attribute importance, and "essential"/"change" pattern was shown to improve classification significantly. The validity of the visual exploration results was thus verified. Improvement of classification results is particularly clear when "change" and "control" are considered together as one class and "no change" as the other class, supporting the visual exploration result that "essential" is an indication for both "change" and "control" in a neighbor.

This shows the potential of employing GlyphNet in a visual exploration process as part of a comprehensive data mining strategy. Given the prevalence of relational data in current data mining tasks and the difficulty of handling it with current data mining techniques, GlyphNet is an important tool to provide leads on interesting patterns.

5 Conclusions

We have introduced a tool, GlyphNet, that assists in the data mining of relational data by combining the concept of glyphs with graph visualization and navigation. Relational data provides particular challenges to the data mining process due to the complexity of potential patterns. Patterns in relational data can involve attributes of multiple interconnected nodes, making traditional data mining and visualization techniques hard to apply. Our tool is designed specifically to assist in the discovery of such patterns. An example of a data mining task for genomics data was presented. We demonstrated how a hypothesis involving attributes of neighboring nodes was generated through GlyphNet. The usefulness of the hypothesis was verified by a quantitative data mining algorithm. We have therefore demonstrated the potential of GlyphNet to assist in a data mining process involving relational data. Many further applications can be envisioned since pattern discovery is a central part of most areas of data mining.

References

1. Herman, I., Melancon, G., Marshall, G.S.: Graph Visualization and Navigation in Information Visualization: A survey, IEEE Transactions on Visualization and Computer Graphics. Vol. 6 No. 1 (2000) 24-43
2. Touchgraph Development: Document and Source, http://touchgraph.sourceforge.net/, (established 2001)
3. Keim, D.A.: Information Visualization and Visual Data Mining. IEEE Transactions on Visualization and Computer Graphics, Vol. 8, No. 1 (2002) 1-8
4. Barabasi, A.L., Bonabeau, E.: Scale-free Networks. Scientific American, Vol. 288, No. 5. (2003) 60-69
5. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Stanford Digital Libraries Working Paper (1998)

6. White, S., Smyth, P.: Algorithms for Discovering Relative Authority in Graphs. The Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003), Washington, DC (2003)
7. Getoor, L., Friedman, N., Koller, D., Pfeffer, A.: Learning Probabilistic Relational Models. Relational Data Mining. Dzeroski, S., Lavrac, N. (eds.), Springer-Verlag (2001)
8. Macskassy, S.A., Provost, F.: A Simple Relational Classifier. Workshop on Multi-Relational Data Mining in conjunction with KDD-2003 (MRDM-2003), Washington, DC (2003)
9. Perera, A., Denton, A., Kotala, P., Jockheck, W., Valdivia Granda, W., Perrizo, W.: P-tree Classification of Yeast Gene Deletion Data. SIGKDD Explorations, Vol. 4, No. 2, (2003) 108-109
10. Di Battista, G., Eades, P., Tamassia, R., Tollis, I.G.: Graph Drawing: Algorithms for the Visualization of Graphs. Prentice Hall (1999).
11. Reingold E.M. Tildford, J.S.: Tidier Drawing of Trees. IEEE Transactions on Software Engineering. Vol. SE-7, No. 2 (1981) 223-228
12. Shiloach, Y.: Arrangements of Planar Graphs on the Planar Lattices. PhD Thesis, Weizmann Institute of Science, Rehovot, Israel (1976)
13. Carrière, J., Kazman, R.: Research Report: Interacting with Huge Hierarchies: Beyond Cone Trees. Proceedings of the IEEE Conference on Information Visualization '95, IEEE CS Press (1995) 74-81
14. Eades, P: A Heuristic for Graph Drawing. Congressus Numeratium, Vol. 42 (1984) 149-160
15. Frick, A., Ludwig, A., Mehldau, H.: A Fast Adaptive Layout Algorithm for Undirected Graphs. Proceedings of the Symposium on Graph Drawing GD '93, Springer Verlag (1994) 389-403.
16. Lamping, L., Rao, R., Pirolli, P.: A Focus+Context Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies. Human Factors in Computing Systems, CHI '95 Conference Proceedings, ACM Press (1995)
17. Sarkar, M., Brown, M.H.: Graphical Fish-eye views of graphs. Human Factors in Computing Systems, CHI '92 Conference Proceedings, ACM Press (1992) 83-91
18. Robertson, G.G., Mackinlay, J.D., Card, S.K.: Cone Trees: Animated 3D Visualizations of Hierarchical Information. Human Factors in Computing Systems, CHI '91 Conference Proceedings, ACM Press (1991) 189-194
19. Hemmje, M., Kunkel, C., Willet, A.: LyberWorld - A Visualization User Interface Supporting Fulltext Retrieval. Proceedings of ACM SIGIR'94, ACM Press (1994)
20. Johnson, B., Schneiderman, B.: Tree-maps: a Space-filling Approach to the Visualization of Hierarchical Information Structures. Proceedings of IEEE Visualisation'91, IEEE CS Press (1991) 275-282
21. SmartMoney: Map of the Market, http://smartmoney.com/marketmap/ (last accessed 10/2003)
22. Eades, P., Feng, Q.: Multilevel Visualization of Clustered Graphs. Lecture Notes in Computer Science", Vol. 1190 (1997) 101-112
23. Hearn, D., Baker, M.P.: Computer Graphics: C Version. Prentice Hall (1997)
24. Chernoff, H.: Using Faces to Represent Points in K-Dimensional Space Graphically. J. American Statistical Association, Vol. 68 (1973) 361-368
25. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining. The MIT Press, Cambridge, MA (2001) 75
26. Jones, G.E.: How to Lie with Charts. Authors' Choice Press, Lincoln, NE (2000)
27. Tufte, E. R.: The Visual Display of Quantitative Information. Graphics Press, Cheshire, CT, (1990)
28. Craven, M.: The Genomics of a Signaling Pathway: a KDD Cup Challenge Task SIGKDD Explorations. Vol. 4, No. 2 (2003) 97-98
29. Munich Information Center for Protein Sequences: Comprehensive Yeast Genome Database, http://mips.gsf.de/genre/proj/yeast/index.jsp (last accessed 10/2003)

A Triangular Reconstruction of Density Surfaces

Michael H. Böhlen[1], Algimantas Juozapavicius[2], Eilverijus Kondratas[3], Arturas Mazeika[1], and Aleksej Struk[3]

[1] Department of Computer Science, Aalborg University, Fredrik Bajers Vej 7E, DK-9220 Aalborg, Denmark
{boehlen,arturas}@cs.auc.dk
[2] Department of Computer Science II, Vilnius University, Naugarduko 24, LT-2600 Vilnius, Lithuania
Algimantas.Juozapavicius@mif.vu.lt
[3] Department of Computer Science II, Vilnius University, Naugarduko 24, LT-2600 Vilnius, Lithuania
{eiko1275,alst1204}@uosis.mif.vu.lt

Abstract. In this paper we introduce a new, fast, simple visualization solution of Density Surfaces (DSes). We introduce the MSP (m-ary space partitioning) tree to partition the input data into slices. The organization of data allows efficient reconstruction of the triangular mesh and reduces the overall complexity of the algorithm to $O(N \log N)$. The experimental evaluation shows the complexity of the algorithm is reasonable and the algorithm is applicable in real time interactive settings. The solution supports visual data mining. It improves the visual quality and eases interpretation of DSes.

Keywords: Surface Reconstruction, Density Surfaces.

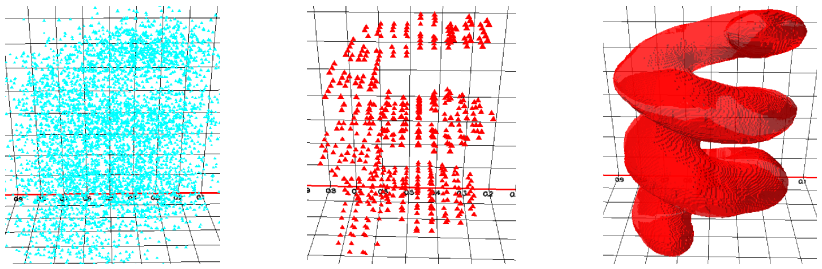
1 Introduction

The continued hard- and software advances during the last few years make it possible to employ highly advanced visualization techniques for the purpose of data mining. Specifically, it has become possible to do explorative data analysis in fully 3D immersive environments (or use advanced animations to simulate 3D immersive worlds).

Visual data mining aims to discover something new from the facts recorded in a database. Standard approach to search for interesting relationships in the database is (i) to build a model of the data, and (ii) visualize the model. Figure 1 illustrates the visual data mining process for Density Surfaces (DSes).

The DS method takes a three-dimensional dataset as an input (cf. Figure 1(a)) and calculates the non-parametric density model of the data. The result of the method is a frame of surface points, which enclose the data whose density is higher than α (cf. Figure 1(b)) [8, 10, 9]. Figure 1(c) shows the solution developed in this paper. The solution takes a frame of surface points as input and calculates a mesh of triangular surface points (triangularization).

The DS module is implemented and integrated into the 3D Visual Data Mining system (3DVDM), which is a visual data mining system used to interactively explore data on computer monitors and in advanced VR environments (Panorama, 6-sided Cave, etc). Figure 2 illustrates the DS module — the part of the 3DVDM system that deals with the calculation and visualization of DSes. First, the module takes a 3D dataset as input, estimates density of the dataset and stores the density information in an APDF



(a) Dataset

(b) Frame of Surface Points

(c) Triangularized Surface

Fig. 1. Calculation steps of a Density Surface

tree (arrow A, Figure 2). Given an APDF tree and a density level α the density surface is calculated in two steps: (i) a frame of points on the surface is identified (arrow $B1$), and (ii) a mesh of triangular surface is calculated (arrow $B2$). The user of the system can provide feedback to the system and control the current data selection (arrow $F1$), the precision of the density estimation (arrow $F2$), and the density level (arrow $F3$). The very tight integration of the calculation of the visualization of density surface solutions is essential in order to ensure interaction with the system in real time. The visualization of the triangles is done with a help of the visualization module of the 3DVDM system. The module visualizes triangles in rectangular coordinate system. The module supports standard visualization and coloring of triangles like transparency and shading.

On the technical level the paper makes the following contributions:

– We present a new, fast, simple, and general visualization solution of DSes. We organize the input frame of surface points into an MSP tree (m-ary space partitioning) – a generalization of the binary space partitioning (BSP) tree data structure [11, 12] . The partitioning decreases the exponential complexity of the triangularization (brute-force approach) to $O(N \log N)$ and enables investigation of DSes in real time interactive settings.

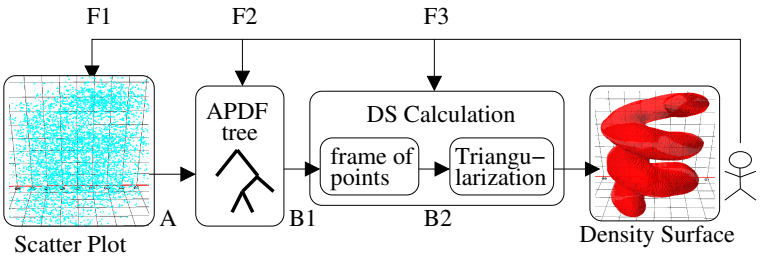


Fig. 2. The DS module

– We provide an evaluation of our triangularization. The numerical evaluation confirms that MSP is a suitable data structure for real time visualization of DSes. The visual evaluation displays selected snapshots of DSes of different shapes and forms. The analysis shows that the calculation time of the triangularization solution is almost invariant with respect to the shape of the surface.
– The solution supports interpretation of DSes. It is easier to perceive triangular surfaces than a frame of surface points (cf. Figure 1(b) and 1(c)). In addition, number of triangles in a triangular surfaces is much lower compared to the number of triangles of the same quality of the frame of surface points. It decreases the load of the visualization and rendering module as well as enables investigations of DSes at higher precisions.

The problem of surface reconstruction from a given set of points is a classical problem and occurs in many and diverse areas of computer graphics, robotics, data mining, etc. [2–7]. Many surface reconstruction algorithms were presented and implemented. Most of the algorithms (CSG, boundary representation, oct-tree) assumes the input data to be in a specific form (like vertices, edges, faces, simple geometric primitives, sets of control points) [1]. Our solution does not assume the input dataset to be in any specific form.

The problem of triangularization is investigated on theoretical level also. An example is the Delaunay algorithm with a number of its generalizations for multidimensional spaces [13]. There has been done little research in implementations and performance evaluations of such algorithms. Also, more naive methods easily outperforms theoretically better algorithms in most practical settings.

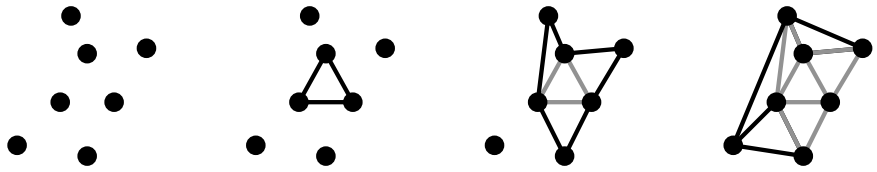
The paper is organized as follows. In Section 2 we define the MSP tree and give the triangularization algorithm. Section 3 evaluates our approach numerically and visually. Conclusions and Future Work is offered in Section 4.

2 The Surface Reconstruction

Given a 3D set of input points, and estimation error, the 3DVDM system estimates density information of the dataset and calculates a surface frame of points. The task of surface reconstruction is to calculate a triangular mesh from the surface frame of points.

Basically, the algorithm calculates set S — the mesh of triangles iteratively. It starts with a random triangle and calculates the smallest triangles having a common edge with the first one (see Figures 3(b) and 3(c) for an illustration). As this iteration is completed, at most four triangles can be in set S. The algorithm iteratively adds the smallest triangles into S having a common edge with any triangle from set S (see Figures 3(c)). The iterative process stops when no more triangles are added into set S.

Computationally, the most expensive step of the algorithm is the calculation of the smallest triangles having a common edge with a triangle from set S. To reduce the complexity of the algorithm we introduce the MSP (m-ary space partitioning) tree — a hierarchical data structure. The tree data structure partitions the input dataset into a group of sets based on the coordinates of points and speeds up the computation of the triangular mesh. Once such a tree is built, the triangularization step finishes the reconstruction surface.



(a) Input dataset (b) The initial step (c) The first iteration (d) The second iteration

Fig. 3. The idea of the Surface Recognition Algorithm

The rest of the section is organized as follows. We define the MSP tree, analyze its properties, and give examples of the data structure in Section 2.1. Surface reconstruction of the frame of points once such a tree is given, is presented in Section 2.2. The complexity of the algorithm as well as the creation of the tree data structure is given in Section 2.3.

2.1 The M-ary Space Partitioning Tree (MSP) Data Structure

M-ary pace partitioning tree is a generalization of the binary space partitioning (BSP) tree data structure. The data structure partitions three-dimensional data into non overlapping group of sets according to coordinates. Intuitively, first, all the data points (x, y, z) are partitioned into sets according to x value (the first level of the tree), then each set is further partitioned according to y value (the second level), the third level of the tree consists of the database points (cf. Figure 4). The cardinality of the root node is the number of different database values of the attribute x. The cardinality of the second level node accessed from x_0 is equal to the number of different y values of database points, whose x coordinate is x_0,

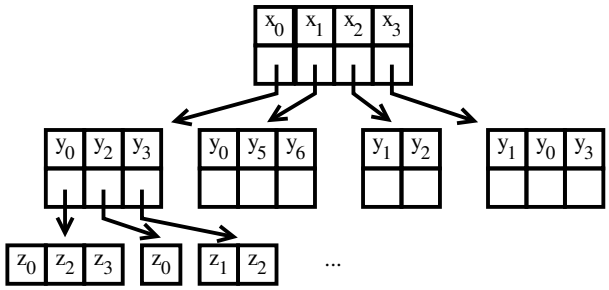


Fig. 4. MSP tree

In order to define the tree precisely, we first introduce a set of operators to partition the dataset according to axes (G_X, G_Y, G_Z).

Definition 1. *(Group $G.(A, v)$). Group $G_X(A, v)$, ($G_Y(A, v)$, $G_Z(A, v)$) for a given set of surface frame points A, is a set of points, satisfying*

$$G_X(A, v) = \{(x, y, z) \in A : x = v\}$$

$$(G_Y(A, v) = \{(x, y, z) \in A : y = v\},$$

$$G_Z(A, v) = \{(x, y, z) \in A : z = v\}).$$

The set of all possible groups we will define as $G.(A)$:

$$G.(A) = \{G.(A, v) : v \in \Re\}$$

To simplify mathematical expressions, we will allow expressions of the type: $G.(H)$ and $G.(H, v)$, where H is a group (a set of sets), and $v \in \Re$ in the following way:

$$G.(H) = \{G.(A) : A \in H\},$$

$$G.(H, v) = \{G.(A, v) : A \in H\}.$$

Example 1. (Group G_X) For simplicity lets consider a two dimensional example presented in Figure 5, with $A = \{$(1,1), (2,1), (3,1), (4,1), (5, 1), (1, 2), (5, 2), (1, 3), (3, 3), (4, 3), (5, 3), (2, 4), (3, 4)$\}$. In this case: $G_X(A, 3) = \{$(3, 1), (3, 3), (3, 4)$\}$.

For a given surface of frame points, let's define partition sets:

$$MSP_{XZY}(A) = G_X(G_Z(G_Y((A))),$$

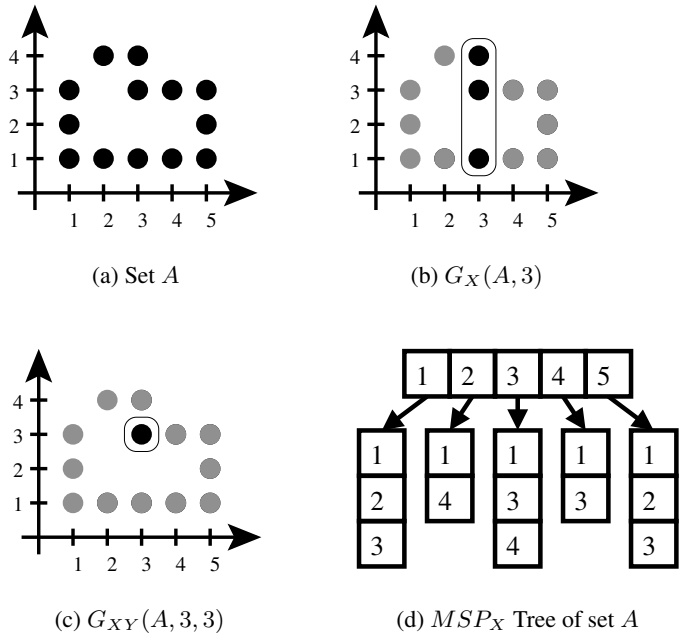
$$MSP_{YXZ}(A) = G_Y(G_X(G_Z((A))),$$

$$MSP_{ZYX}(A) = G_Z(G_Y(G_X((A))).$$

Groups MSP_{XYZ}, MSP_{YXZ}, and MSP_{ZYX} represent three different partitions of surface frame points A into MSP tree. For example, MSP_{ZYX} groups the surface frame points according to the following order of coordinates: X, Y, and then Z.

Example 2. (MSP_{YX} tree.) An example of MSP_{YX} tree for Example 1 is given in Figure 5(d). More specifically,

$$\begin{aligned} MSP_{YX} = \{ & \{(1,1),(1,2),(1,3)\}, \\ & \{(2,1),(2,4)\}, \\ & \{(3,1),(3,3),(3,4)\}, \\ & \{(4,1),(4,3)\}, \\ & \{(5,1),(5,2),(5,3)\}\}. \end{aligned}$$



(a) Set A (b) $G_X(A, 3)$

(c) $G_{XY}(A, 3, 3)$ (d) MSP_X Tree of set A

Fig. 5. Example of groups

On the implementation level, the data partitions ($MSP_{...}$) are implemented with a help of sorted arrays, i.e., the nodes are sorted (for example, $x_0 < x_1 < x_2 < x_3$ in Figure 4). We also define D_{XZY} (D_{YXZ}, D_{ZYX}) and L_{XZY} (L_{YXZ}, L_{ZYX}) look-up functions on a given tree MSP_{XZY} (MSP_{YXZ}, MSP_{ZYX}). Function D_{XZY} (address function) takes an input data point as a parameter and returns indexes of the coordinates in the sorted arrays (nodes). Function L_{XZY} (look-up function) takes a triple of indexes and returns the point the indexes are pointing to in the tree. We will use letters (i, j, k) to denote the address of point p and (x, y, z) to denote the value of p at address (i, j, k).

Example 3. (D_{ZYX}, L_{ZYX} functions). Let $p = (x_0, y_3, z_2)$, and MSP_{ZYX}, as in Figure 4. Then $D_{ZYX}(p) = (0, 2, 1)$. Also, $L_{ZYX}\big((0, 1, 0)\big) = (x_0, y_2, z_0)$.

Functions $D_{\cdot}$ and $L_{\cdot}$ allows efficient neighborhood queries. In the triangularization step we will search for the following type of queries. Given a point $p = (x, y, z)$ we will query the database for points of the form $(x \pm x^{\pm}, y \pm y^{\pm}, z \pm z^{\pm})$, where x^{+} (x^{-}) represents the smallest x database value, which is greater (smaller) than x. More specifically, we first query for the address of the point: $(d_{1\cdot}, d_{2\cdot}, d_{3\cdot}) = D_{\cdot}(p)$ and then query for its neighbors: $L_{\cdot}(d_{1\cdot} \pm 1, d_{2\cdot} \pm 1, d_{3\cdot} \pm 1)$.

Definition 2. *(Nearest Slice Neighborhood (NSN) of surface point $p \in A$). Let $p \in A$. The nearest slice neighborhood of p is set $NSN(p)$, satisfying:*

(i) $(d_{1X}, d_{2X}, d_{3X}) = D_{XZY}(p)$;
$(d_{1Y}, d_{2Y}, d_{3Y}) = D_{YXZ}(p)$;
$(d_{1Z}, d_{2Z}, d_{3Z}) = D_{ZYX}(p)$,

(ii)
$$\begin{aligned} NSN(p) = {} & L_{XZY}(d_{1X} \pm 1, d_{2X} \pm 1, d_{3X} \pm 1) \cup \\ & L_{YXZ}(d_{1Y} \pm 1, d_{2Y} \pm 1, d_{3Y} \pm 1) \cup \\ & L_{ZYX}(d_{1Z} \pm 1, d_{2Z} \pm 1, d_{3Z} \pm 1). \end{aligned}$$

We define the NSN of edge (p_0, p_1) *as the union of the NSNs of the vertexes of the edge:* $NSN\big((p_0, p_1)\big) = NSN\big(p_0\big) \cup NSN(p_1)$.

Example 4. (Nearest Slice Neighborhood). Let's continue Example1. $NSN\big((1,1)\big) = \{(1,1),(1,2),(2,1)\}$.

Note, that the NSN neighborhood of point p_0 can contain at most 27 points (cf. Figure 6).

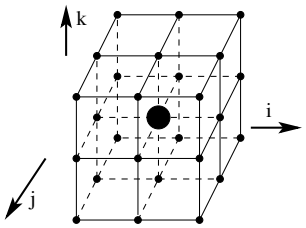


Fig. 6. The largest possible NSN of a point

2.2 Triangularization

The triangularization is done by combining closest three points into triangles based on Euclidean distance. The algorithm of triangularization consists of two steps:

- calculation of the first triangle,
- iterative addition of the remaining triangles.

The first triangle must satisfy the following three properties:

- the triangle consists of three points p_0, p_1, p_2 from surface frame of points A,
- p_0, p_1, p_2 are not on a line,
- there is no other point $p \in A$ such that $p \in \triangle(p_0, p_1, p_2)$, unless p is one of p_0, p_1, or p_2.

We find such a triangle in the following way. We choose a random surface point $p_0 = (x_0, y_0, z_0) \in A$, and scan for a set of the NSN points $NSN(p)$. Any two points $p_1, p_2 \in NSN(p)$ will form a triangle with p_0, so there will be no other points inside $\triangle(p_0, p_1, p_2)$, provided that there is not point $p' \in NSN(p)$ such that p' on the edge (p_0, p_1). Mathematically, the idea can be expressed in the following way:

1. Select a random point $p_0 \in A$.
2. Calculating the NSN set of point p_0: $N = NSN(p_0)$ (cf. Definition 2).
3. Let $p_1, p_2 \in N$ be such that p_1, p_2, p_3 are not on a line and there is no $NSN(p)$ points in between p_0 and p_1 on the line. Then $\triangle(p_0, p_1, p_2)$ is the first triangle.

During the iterative addition of the remaining triangles we maintain the list of triangles M introduced in the last step. For each edge e of a triangle t from list M we search for the candidate point of surface frame A to form a new triangle. The candidate point is the closest point in the NSN of edge e to the direction "opposite" to triangle t (cf. Figure 7). Mathematically, the idea can be expressed in the following way:

1. Let M be the list of the triangles introduced in the last step of the iterative process.
2. For each triangle $\triangle \in M$ and new edge $e \in \triangle$ do
   2.1 Calculate $NSN(e)$ (cf. Definition 2).
   2.2 Calculate the dividing plane $P = P(x, y, z)$ of $NSN(e)$ (cf. Figure 7):

$$\begin{vmatrix} x & y & z & 1 \\ x_0 & y_0 & z_0 & 1 \\ a & b & c & 0 \\ l & m & n & 0 \end{vmatrix} = 0,$$

   where: (x_0, y_0, z_0) is a point on edge e, $(l, m, n) = (x_2 - x_1, y_2 - y_1, z_2 - z_1)$ is a vector of edge e,

$$(a, b, c) = \left(\begin{vmatrix} (y_1 - y_2) & (z_1 - z_2) \\ (y_1 - y_3) & (z_1 - z_3) \end{vmatrix}, - \begin{vmatrix} (x_1 - x_2) & (z_1 - z_2) \\ (x_1 - x_3) & (z_1 - z_3) \end{vmatrix}, \right.$$

$$\left. \begin{vmatrix} (x_1 - x_2) & (y_1 - y_2) \\ (x_1 - x_3) & (y_1 - y_3) \end{vmatrix} \right),$$

   is a normal vector of the triangle. The expression of the plane enables us to divide the three–dimensional space into two sub-spaces: a sub-space of the space to the "left" of the plane ($\{(x, y, z) \in \Re^3 : P(x, y, z) < 0\}$), and the "right sub-space" ($\{(x, y, z) \in \Re^3 : P(x, y, z) > 0\}$).
   2.3 Calculate the neighborhood points, "opposite" to $\triangle$:

$$NSN^\circ = \{(x, y, z) \in NSN : \text{sgn}(P(x, y, z)) \neq \text{sgn}(P(x^o, y^o, z^o))\},$$

   where (x', y', z') is the "opposite" point of the triangle:

$$(x^o, y^o, z^o) = \{(x, y, z) \in \triangle, (x, y, z) \notin e\}.$$

   2.4 Find the closest point from NSN° to edge e. The middle point of the edge e:

$$p_M = (x_M, y_M, z_M) = \big((x_1 + x_2)/2, (y_1 + y_2)/2, (z_1 + z_2)/2\big).$$

   The closest point to edge e then is

$$p_{\min} = \min_{(x,y,z) \in NSN^\circ} \Big(\rho\big((x, y, z), (x_M, y_M, z_M)\big) \Big),$$

   where $\rho(\cdot, \cdot)$ is the Euclidean distance.

In step 2.4, the algorithm calculates all NSN points of point p with the help of the MSP tree. The computational results, therefore, are invariant to the order of the partitioning coordinates. That is, the order of partitioning coordinates: X, then Y, then Z is invariant for the computational results compared to the order, for e.g., Y, then Z, then X.

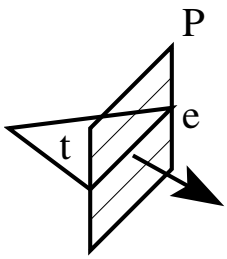


Fig. 7. The search direction for the candidate points

2.3 Complexity of the Triangularization Solution

In this section we give the worst-case complexity of the algorithm.

The triangularization algorithm consists of two steps: the data preparation step and the triangularization step, once the data is prepared. Next, we investigate the complexity of the steps separately.

In the data preparation step, three MSP tree data structures: MSP_{XZY}, MSP_{YXZ}, and MSP_{ZYX} must be created. In the database with attributes x, y, and z it can be done in $O(N \log N)$ complexity with a help of the sorting algorithm, where N is the database size. Note, that a look-up for the address of a three-dimensional point, once such the trees are given, is $O(\log N)$ (the complexity of finding an element in a sorted list).

The triangularization step consists of the search of the first triangle and the iterative process, where the list triangles M introduced in the last step is maintained. The calculation of the first triangle involves the following steps:

- random selection of a surface point p (constant time),
- calculation of neighborhood $NSN(p)$ (27 look-ups — $O(\log N)$ complexity),
- calculation of the triangle from set $NSN(p)$ (constant time).

Therefore, the complexity of the step is $O(\log N)$. The iterative process involves the following steps:

- calculation of the neighborhood for a triangle $\triangle$ ($O(\log N)$ complexity),
- calculation of a set of triangles for the triangle $\triangle$ (constant time).

The above two steps are iterated until no new triangles are produced. The number of iteration in this step is $O(N)$, since a triangle cannot be included into list M more

than once. Therefore, the overall complexity of the step is $O(N \log N)$. Hence, we just proved the following theorem:

Theorem 1. *The worst-case complexity of our triangularization algorithm is* $O(N \log N)$.

3 Experimental Section

In this section we evaluate our triangularization algorithm numerically and visually. All the experiments were run on Intel P4 2.4GHZ machine with Mandrake Linux 9.1 OS.

The algorithm is tightly integrated with the 3DVDM system: it gets a frame of surface points data as an input from the Frame of Surface Points (FSP) module and outputs the list of triangles to the visualization module (cf. Figure 2). The current implementation of the FSP module outputs frame of surface points on a three-dimensional grid. This have two implications on the evaluation of the surfaces:

- the number of triangles produced is linear with respect to the input dataset (cf. Figure 8(a)).
- the orientation of the produced triangles are limited to 0°, 45°, and 90° degrees of fixed sizes (two different sizes) of triangles. That produces locally angular surfaces (cf. Figure 9(a)).

Figure 8(b) shows experimental time complexity for different sizes of inputs sets for the spiral surface (cf. Figure 9). We use the spiral surface as a comparison point, because spiral is a visually complex structure and represents a worst-case scenario.

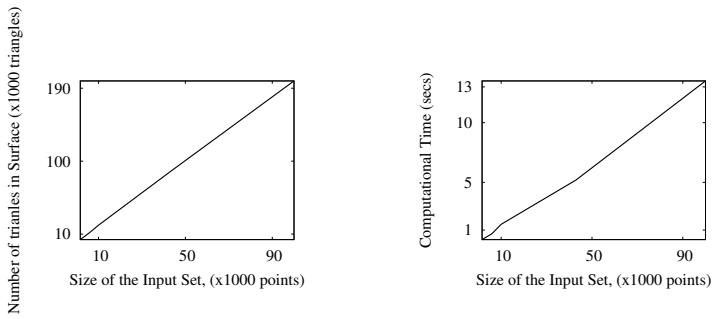
Two main parameters influence the calculated time and visual quality of the triangularization: the number of input frame of surface points and the number of triangles produced.

The experimental complexity of the algorithm is linear wrt the size of input dataset, what complies with the theoretical complexity of the algorithm for relatively small N. The evaluation also shows that the MSP tree does not deteriorate as number of input data points increases.

Figure 9 shows triangular density surfaces for different sizes of input sets. Figure 9(a) shows that it is possible to get a rough (but already informatory!) density surface very fast. As the size of the input set increases, the visual quality improves very quickly (cf. Figure 9(b) – 9(c)).

We evaluate the impact of visual complexity of density surfaces to the computational time in Figure 10. In order to get robust timings we increased the sizes of input sets to extreme values. The processed number of input points a second (cf. rate parameter in Figure 10) shows that our approach is almost invariant to the complexity of the input frame surface points.

The points of the input datasets in Figure 10 are distributed on a regular grid. Because of a very fined grid, the 0°, 45°, and 90° triangles of fixed sizes approximates the surfaces very nicely. However, the number of triangles in the surfaces can be dramatically reduced by allowing more possible angles and shapes without the trade for accuracy.



(a) Number of triangles created

(b) Computational Time

Fig. 8. Numerical Evaluation of our triangularization method

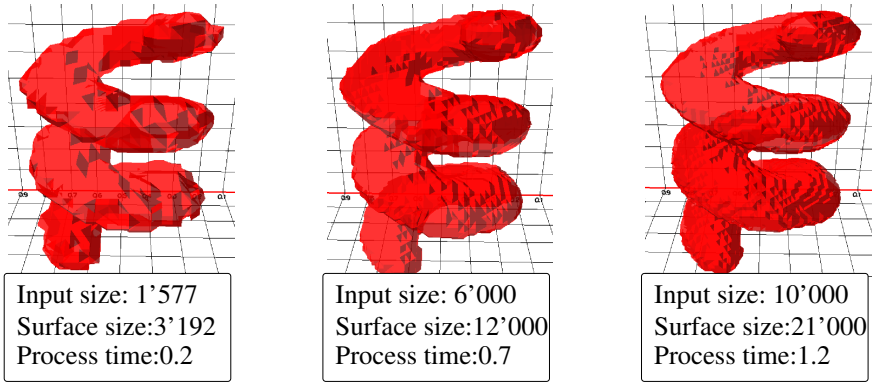
4 Conclusions

In this paper we introduce a new, fast, simple, and general triangularization solution of Density Surfaces. We organize the input set of points into the MSP tree and achieve the overall $O(N \log N)$ complexity of the method. We test the method experimentally on a number of artificially generated datasets. The experiments confirmed that the method is applicable in real-time settings. The rendering rate shows that the method is almost invariant to the visual complexity.

In the future it will be interesting to generalize the method and calculate triangular density surfaces directly from the APDF tree. This is attractive, since the APDF represents linear regions of density optimally. We expect to further decrease triangularization time and size of the surface. This will allow to investigate DSes at higher precisions and decrease the load of the visualization module.

References

1. Franz Aurenhammer. Voronoi diagrams—a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.
2. J.-D. Boissonnat and M. Teillaud. On the randomized construction of the Delaunay tree. *Theoretical Computer Science*, 112(2):339–354, 1993.
3. K. L. Clarkson, K. Mehlhorn, and R. Seidel. Four results on randomized incremental constructions. In *Symposium on Theoretical Aspects of Computer Science*, pages 463–474, 1992.
4. O. Devillers. Improved incremental randomized delaunay triangulation. In *Symposium on Computational Geometry*, pages 106–115, 1998.
5. O. Devillers. On deletion in delaunay triangulations. In *Symposium on Computational Geometry*, pages 181–188, 1999.
6. H. Edelsbrunner. *Algorithms in Combinatorial Geometry*. Springer, Heidelberg, 1987.
7. B. Joe. Construction of three-dimensional improved-quality triangulations using local transformations. *SIAM Journal on Scientific Computing*, 16(6):1292–1307, 1995.
8. A. Mazeika, M. Böhlen, and P. Mylov. Density Surfaces for Immersive Explorative Data Analyses. In *Proceedings of Workshop on Visual Data Mining in conjunction with SIGKDD*, 2001.



(a) Quality is rough

(b) Quality is good

(c) Quality is very good

Fig. 9. Visual Evaluation of the Method (different granularity of the input surface)

9. A. Mazeika, L. Bukauskas, M. Böhlen, and P. Mylov. The 3dvdm approach, a case study with clickstream data. To be published in LNAI Lecture Notes in Artificial Intelligence.
10. A. Mazeika, M. H. Böhlen, and P. Mylov. *The Density Surfaces Module*, December 2003.
11. K. Mulmuley. Randomized multidimensional search trees (extended abstract): dynamic sampling. In *Proceedings of the seventh annual symposium on Computational geometry*, pages 121–131. ACM Press, 1991.
12. R. Seidel. A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons. Technical report, 1990.
13. Peter Su and Robert L. Scot Drysdale. A comparison of sequential delaunay triangulation algorithms. In *Proceedings of the eleventh annual symposium on Computational geometry*, pages 61–70. ACM Press, 1995.

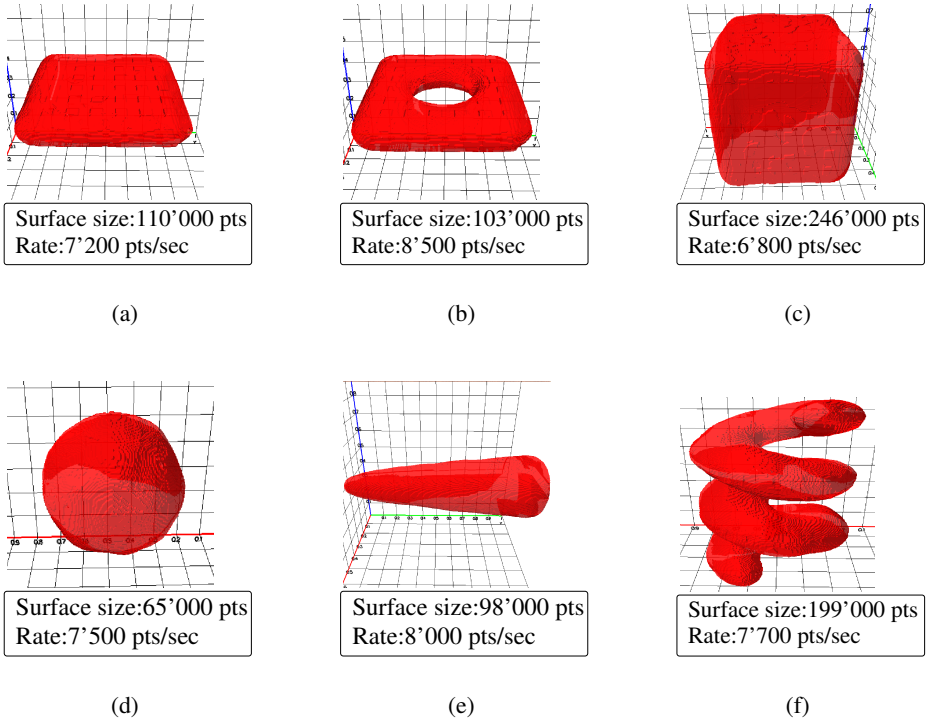


Fig. 10. Impact of the complexity of the surface on the performance

Densityplot Matrix Display for Large Distributed Data

Jing Zhang, Computer Science, Iowa State University, Ames, IA
Leslie Miller, Computer Science, Iowa State University, Ames, IA
Dianne Cook, Statistics, Iowa State University, Ames, IA
Aulia Hardjasamudra, Statistics, Iowa State University, Ames, IA
Heike Hofman, Statistics, Iowa State University, Ames, IA

Abstract

In this paper, a software application, Limn Matrix, is developed to interactively display a densityplot matrix for large, distributed data. A density plot is used to alleviate overplotting of points to more accurately represent the distribution of data points. To efficiently exchange large amounts of information through the network, we propose a hierarchical indexing system, by which the information volume transferring via network only depends on the number of plots and plot window size rather than data size. In addition, Limn Matrix provides several interactive features, including subset brushing and density transformation controls, to provide users extra insight about the large data set. An application on forest cover type data and future improvements of the software are also discussed.

1. Introduction

Scatter plots are widely used for data visualization. By plotting one variable against the other in a 2D space, the patterns of dependence and deviations from dependence between a variable pair can be easily identified. A scatter plot matrix is a neat layout of multiple scatter plots when there are more than two variables involved. The plots are arranged into a matrix format which actually matches the form of the correlation matrix. An N-dimensional data set will form a matrix of N*(N-1)/2 scatter plots. Laying out scatter plots of any two variables in a matrix helps to reveal the relationships between the variable pairs. In addition, user interactions, like brushing and subset coloring, provide users extra insights on the relationship between more than two variables (Becker and Cleveland 1987a).

A scatterplot can display a small set of data without problems. However, viewing large data sets with scatterplots brings two major challenges:

1) Massive overplotting of points
Plotting large data set into a scatterplot matrix, in which each plot has a fixed dimension, will result in many data points crowded in the same region. Displaying such a scatter plot on a computer screen is especially hard because of the limited area of screen real estate. Many close neighbors will fall in the same screen pixel. For instance, if we have random uniformly distributed data with one million cases and want to plot it into a matrix that consists of 100*100-pixel scatterplots. On average, there will be 1,00,000/10,000=100 data points each screen pixel in a scatterplot. The levels of overplotting at different pixels are varied if the data set has some other patterns. This results in some pixels having substantial overplotting points, while others have less. Such massive overplotting points and varied overplotting patterns can easily overwhelm a simple scatterplot matrix, and make trends and relationship between variables hard to identify.

2) Distributed data
A large data set requires a great amount of storage space, and frequently is distributed to several locations. Exploring and retrieving data remotely is expensive. When data sets get really large, shipping the full data around network is impossible. Moreover, even subset transferring through the network becomes infeasible for some large data set. Network traffic will definitely slow down the feedback speed and affect user interactions.

There has been some prior work on using density plots to solve the overplotting problem. Scott (1982) used a 3D bivariate histogram to represent the frequency of overlapping points in heights of 3D bars. Histograms generated in this way are usually not smooth, making it difficult to grasp the true structure of the data (Silverman 1986). Averaged shifted histograms smooth the shape of bivariate histograms (Stigler, 1986 and Scott 1992). Perspective views or contour plots of density estimation constructed from continuous kernels are also believed to provide a better picture of the data distribution (Silverman 1986). Flattened histogram with grayscale density or contour plot is another approach to solve the overplotting problem. Cleveland and McGill introduced the sunflower plot to encode the count of overplotting points in a bin to number of petals of a sunflower (Cleveland and McGill 1984). Motivated by the sunflower plot, Carr et al (1987) proposed hexagonal bins which reduce density estimate bias compared to square bins. In addition they suggested using gray scale and colors to represent data density. Huang et al (1997) proposed a similar idea in their variable-resolution bivariate plot. In 2003 Dupont and Plummer proposed a variant of sunflower plot, density distribution sunflower plot, by providing size and color controls over the sunflower (Dupont 2003). All of these approaches focus on static plots only. Huang's Varebi plots displayed 65,536 cases only in a static way. Dupont's density distribution sunflower plot provides control over sunflowers' size and color, but handled only thousands of cases.

For data analysis interaction in a plot is critical, brushing is a common interaction technique for scatterplot (Becker and Cleveland 1987a). The user paints a subset of a data and the other plots are updated accordingly. While brushing may be easy for small data sets, there are difficulties in applying this technique for large data set. Even with today's powerful computation devices, tools supporting interactively brushing of large data are still very few. We tested some traditional statistical analysis software, Microsoft Excel, JMP and R, with a simulated data set which has 500,000 cases and 5 variables. It turned out that Excel cannot load more than 65,536 rows (cases). JMP took several minutes to display the scatterplot matrix and allows no brushing interaction. R also took long time to load the data and display the scatterplot matrix. Similar to JMP, R doesn't' have brushing interaction either.

In this paper, we present a system, Limn Matrix, which interactively displays large distributed data in densityplot matrix. We use a flattened 2D histogram to solve the overplotting problem, and use gray scale to represent data density. Power transformations are applied to transform density to grayscale value. Controls over transformation parameters, *power and intercept,* provides users the flexibility to focus on different density levels.

Limn Matrix uses a hierarchical indexing technique to link scatterplot points and distributed data sets. Instead of shipping the full data set or subset of the whole data around, Limn Matrix let subservers retrieve data locally and transfer only pixel positions and density counts through network. This technique highly improves interaction performance since information volume transferred through network depends only on plot image's dimension rather than on size of data set.

Limn Matrix provides multiple user interactions, including subset brushing, and density transformation controlling. With these interactive functions, users will be able to view more details and hence acquire better understanding of the data.

2. A Solution – Density Matrix

2.1 Flattened 2D Histogram

Limn Matrix uses flattened 2D histogram to display the data density, and considers both Scott's (Scott 1979) and Freedman-Diaconis's (Freedman and Diaconis 1981) rules for choosing the default bin number. Scott proposed the optimal bin width as

$$\hat{h} = 3.5\ \hat{\sigma}\ n^{-1/3}, \qquad (1)$$

where $\hat{h}$ is the bin size, $\hat{\sigma}$ is the standard deviation of the distribution, and n is the data size. Freedman and Diaconis replaced the standard deviation of Scott's rule by the interquartile range (IQ)

$$\hat{h} = 2\ \mathrm{IQ}\ n^{-1/3}. \qquad (2)$$

Scott (1992) applied formula (1) and (2) to calculate bin number assuming that the data is standard normal, N(0, 1), and the range is (-3, 3). The bin number calculations were applied to data with size n, which was from 50 to 100,000. In this paper, we extended Scott's calculations to data with larger size, 500,000 to 10,000,000, and the result is shown in Table 1.

n	Scott Rule	Freedman-Diaconis Rule
500,000	136	177
1,000,000	171	223
10,000,000	369	479

Table 1: Number of bin size for large data set

Both Scott and Freedman-Diaconis rules suggest hundreds of bins per variable for large data set. A standard computer screen has only 1280*1024 pixels. To fully display a matrix for all variables, we can use as few as one pixel for each bin. Balancing the optimal bin number suggested by theory and the physical limitation of the screen, we chose 200 pixels (bins) as Limn Matrix's default bin number per variable.

2.2 Density Transformation

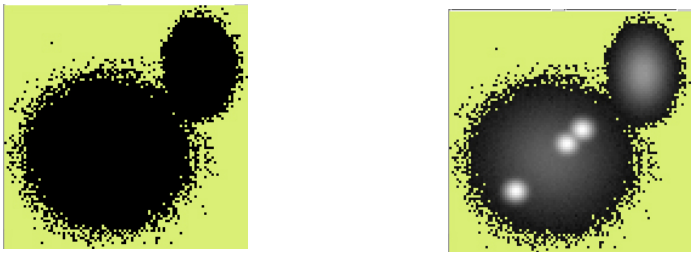
A plot window of size 200 * 200 pixels gives a total of 40,000 bins. Any data set with 40,000 cases will result in overplotting. When there is dependence between variables, substantial overplotting occurs even with small sample size. It is common to see data which has many more than 40,000 cases. An ordinary scatterplot shows only two states of a screen pixel: existence or non-existence of a data point. This strategy doesn't work so well with large data due to loss of information on overplotting amount.

Limn solves this problem by using a density plot matrix to display large data set. Density level of each bin (pixel) is transferred to a grayscale value. The transformation function we applied is

$$\text{grayscale} = \textit{intercept} \times \left(\frac{\text{overplotting count}}{\text{maximal overplotting count}} \right)^{\textit{power}}, \qquad (3)$$

where *intercept* is between 0 – 255 and *power* is between 0.0 – 1.0.

This transformation distinguishes pixels with different density counts from each other by assigning different gray values to them. For instance in a transformation with *intercept* = 255 and *power* = 0.2, pixels with low count will have a gray value as small as 0, which is almost black; whereas high count pixels' gray value will be approximately 255, which is nearly white. Figure 1 compares a scatterplot and a densityplot for a one million cases data. Densityplot in b gives a more accurate description of the data distribution compared to an ordinary scatterplot in a.



a. ordinary scatterplot gives no overlapping information

b. densityplot distinguishes low and high density pixels

Figure 1: (left) ordinary scatterplot and (right) density plot for a data with one million cases

2.3 Hierarchical Index to Handle Large and Distributed Data

It is common that large data is stored over several places. The network delay is a main factor that affects the communication speed between a plot's data points and distributed data sets. Shipping whole or even a subset of large data is practically infeasible. To solve this problem, we designed and implemented a hierarchical indexing system, which builds fast links between distributed data sets and scatterplot points, as well as points between plots of a matrix.

The goal of this system is to speed up data to plot communication by communicating only position and density count of a plot's pixels through network. Consider a remote data set with 5 variables and one million cases, the total network information amount will be 5 million numbers if the whole data set is requested. However, the information needed to construct a scatterplot matrix for these 5 variables could be less than 5 million numbers. There are only 10 different plots in the matrix. If the size of each plot is 200 by 200 pixels, the whole plot matrix can be represented by 200*200*10 = 400,000 pixels. Using two integers, pixels position and density count, for each nonempty pixel, our indexing system needs to move at most 800,000 integers through network for the entire matrix. The load is approximately 1/6 compare to the whole remote data set. This example illustrates that, with our indexing system, information volume moved through the network is not decided by data size but only number of plot and size of the window in a matrix. In other words, given a matrix with fixed plot number and size, the network transferred information will not exceed a constant amount with our indexing system.

There are two layers of indexes contained in our indexing system: a subserver layer indexes distributed data sets locally; a server layer links these subserver indexes to scatterplot points. The structure of the indexing system is illustrated in Figure 2.

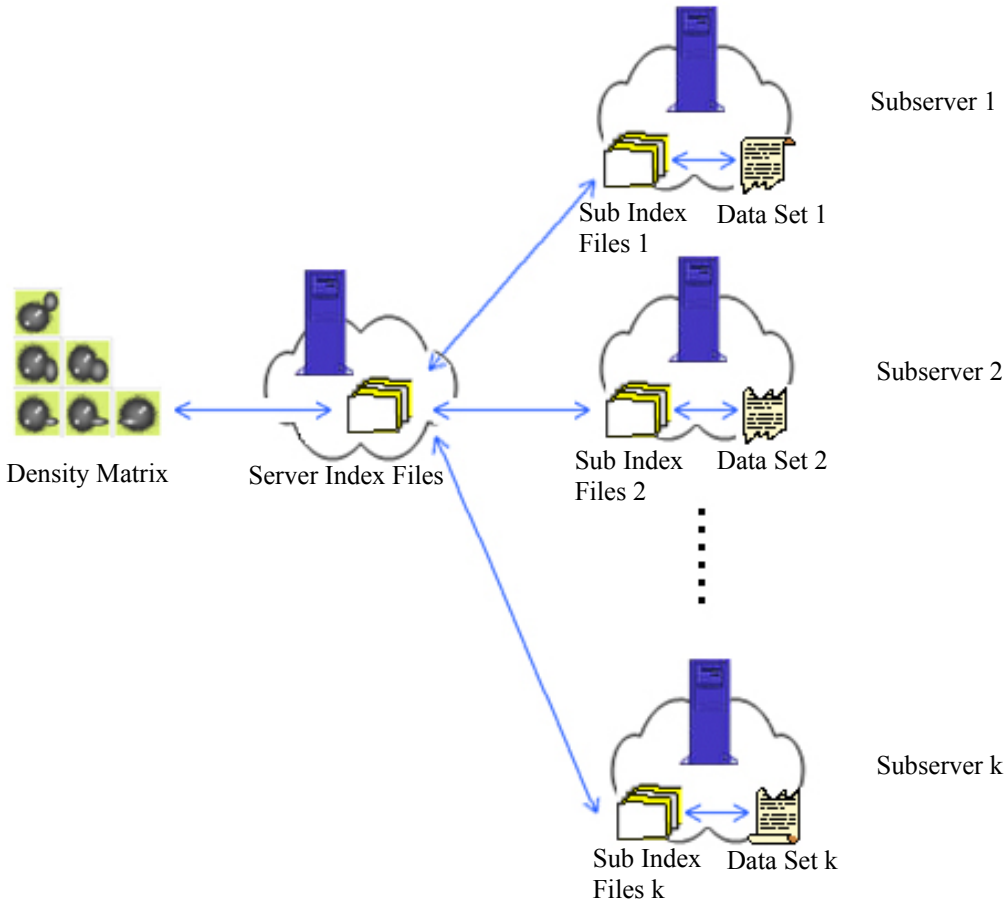


Figure 2: Structure of the hierarchical indexing system

The subserver plots its own data set into a scatterplot matrix. This data set is indexed through three files for each scatterplot in the matrix. The structure of the subserver index files is shown in Figure 3.

- Subserver X index file stores x coordinates of those non-empty screen pixels of a scatterplot. These x coordinates, denoted as SSx, are recorded in increasing order. Each Sx points to a block of y coordinate in a subserver Y index file.

- Subserver Y index file stores y coordinates, denoted as SSy, in blocks. Y coordinates of those non-empty screen pixels are also recorded in increasing order in a block if they share a same Sx. Each Sy points to a list of ID numbers in a subserver ID index file.

- Subserver ID index files stores lists of ID numbers that points to a distributed set of raw data. Raw data tuples have distinct ID numbers. Those tuples projected into the same screen pixel will have their ID numbers recorded in increasing order in the same ID list.

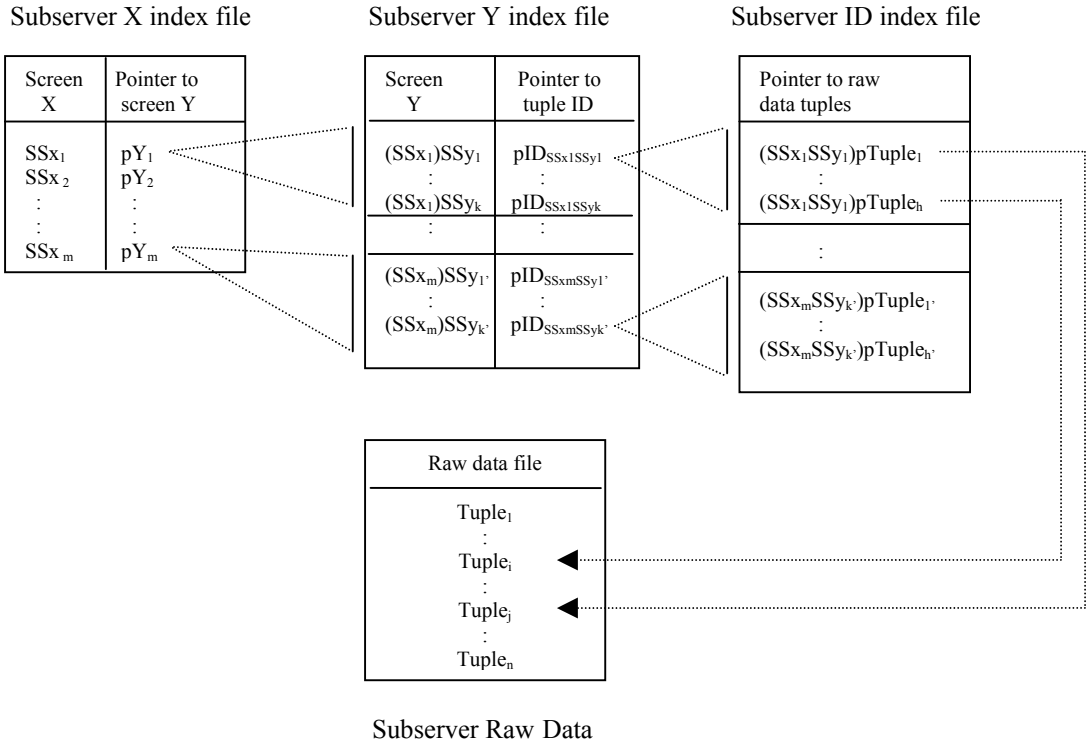


Figure 3: Structure of subserver index files

The server layer consists of index files merged from subserver. Merged index files have very similar format as those at subservers. For each scatterplot in a matrix there are also three index files in the server layer. The structure of the server index files is given in Figure 4.

- Server X index file merges corresponding subserver X files. X coordinates in server X file, denoted as Sx, represents x position of the non-empty screen pixels in a merged scatterplot. These x coordinates are recorded in increasing order. Each Sx points to a block of y coordinates in a server Y index file.

- Server Y index file merges corresponding subserver Y files by block. A block of y coordinates consists of y positions of those non-empty screen pixels having the same x position. Y

coordinates, denoted as Sy, are also recorded in increasing order in a block. Each Sy points to a block of ID numbers in a server ID index file.

- Server ID index file does not directly link to raw data sets anymore. Instead, the linking is made through subserver ID files. Each non-empty screen pixel in a scatterplot is represented by a block of ID numbers in the server ID file. These numbers are the IDs of those subservers in which some of their raw data tuples project into this pixel. These tuples' ID numbers have already been recorded as a list in the subserver ID file. Server can thus link a screen pixel to data sets by pointing the subserver ID number recorded in a block to the corresponding list of tuple IDs in a subserver ID file.

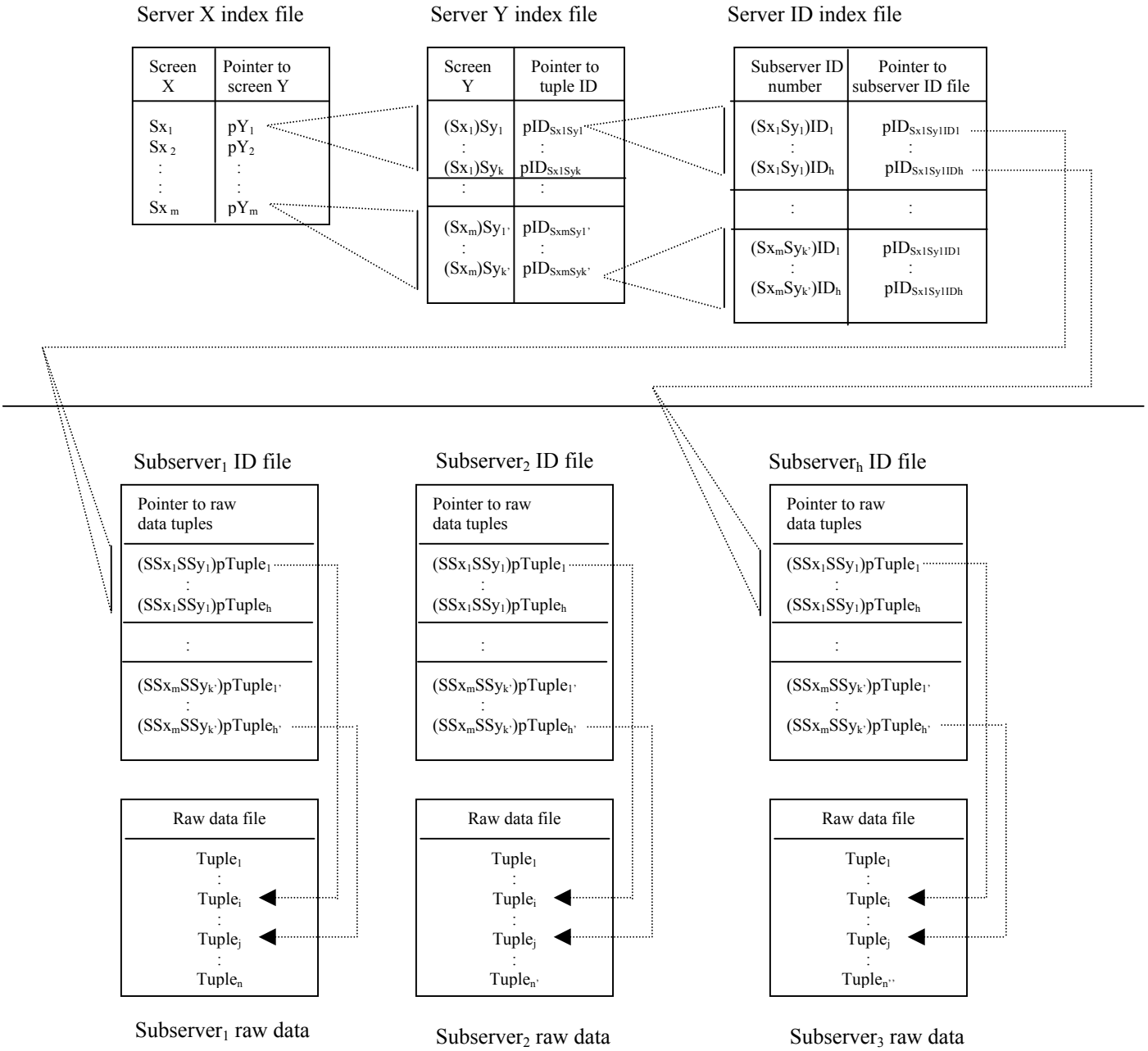


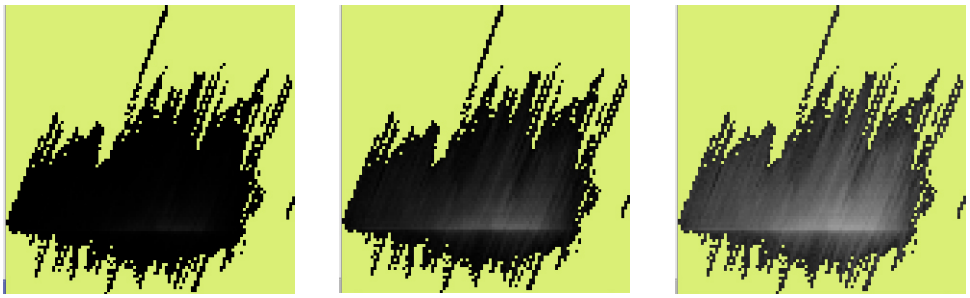
Figure 4: Structure of server index files

2.4 Interactions

Several user interactions are permitted in Limn Matrix to display more data details.

2.4.1 Transfer Function Controls

Limn Matrix uses a grayscale transformation to visually distinguish pixels with different density counts. Base on formula (3) in section 2.2, the transformation formula contains two parameters, *power* and *intercept*. Users are allowed to control these two parameters to focus on regions with different density levels. *Intercept*, ranging from 0 to 255, controls the maximal gray difference allows to display. *Power*, ranging from 0.0 to 1.0, regulates visual difference of pixels with varied overplotting amounts. Figure 5 shows a series of power value adjustments on a single plot: a linear transformation (power = 1) equally separates all pixels in different density level; and a transformation with lower power value (power = 0.2) can distinguish density pixels even for small density difference. More details of the power value adjustment will be discussed in the application section.



a. power = 1 b. power = 0.4 c. power = 0.2

Figure 5: Sequence views demonstrating power value adjusting.

In addition to controlling power and intercept values of the transfer function, users can also apply contour control on the curve display of the transfer function. By sliding the choosing bars across the function curve, users can choose a certain gray value or a gray value range from the spectrum. Data with corresponding density value or density range will be highlighted to give a contour view of the data distribution. In Figure 6 a gray value range is defined by the two bars crossing the function curve, and pixels in corresponding density level are highlighted in light blue.

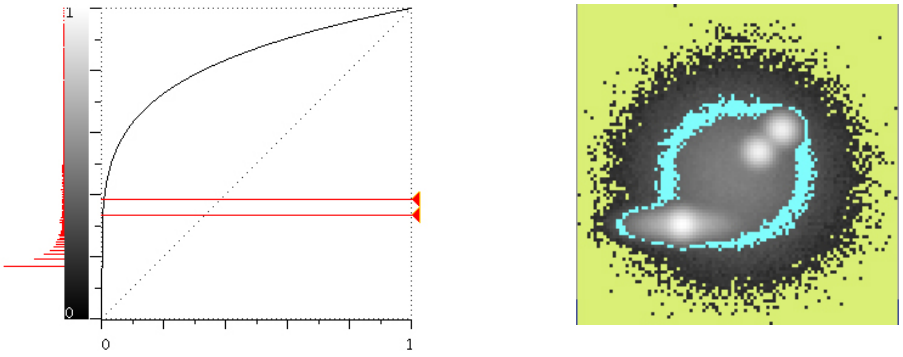


Figure 6: Contour control highlights pixels within a chosen density range. A density range is chosen by sliding two bars crossing the transfer function curve (left). Corresponding plot pixels are highlighted in light color (right). In the plot, the highlighted pixels form a contour around the most high density area.

2.4.2 Density Brushing

Subset brushing is a commonly used technique to link matrix plots. Ordinary brushing lets users select a rectangle area from a plot and have subsets drawn correspondingly on the other linked plots (Becker 1987b). This strategy displays only the shape of subset on each plot, which as Wegman (1999) pointed out is potentially confusing and misleading when there is considerable overplotting.

Limn Matrix supports real time brushing interaction and provides users two optional views to understand the whole data density and subset density. The two options are transparent shape view and proportional density view.

1. Like ordinary brushing, the shape view shows the shape of subset data on top of base plots. However, instead of obscuring the underlying image areas, Limn Matrix allows viewers to control the top subset's transparency. Users can see through top subset by adjusting its transparency level. In this way viewers can learn the shapes of brushed subsets without losing distribution information of the whole data. As an example in Figure 7, a and b show distribution of two of matrix plots, that is four of the variables. Brushing happens in the first two variables, where a rectangle area is brushed in the top plots (c, e). Plot d and f show the results of the shape representation of the subset, as opaque and transparent respectively. In the plot f both the data density and the shape of the subset can be seen.

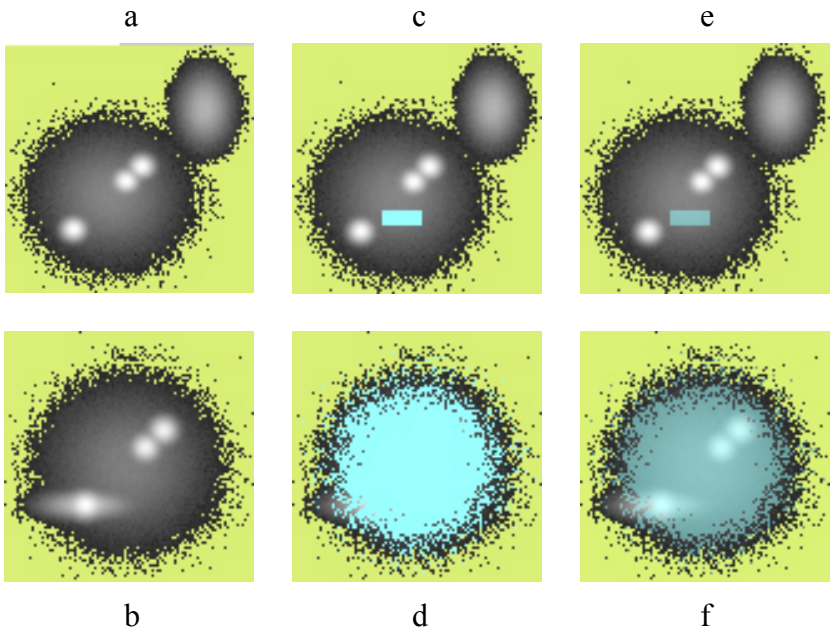
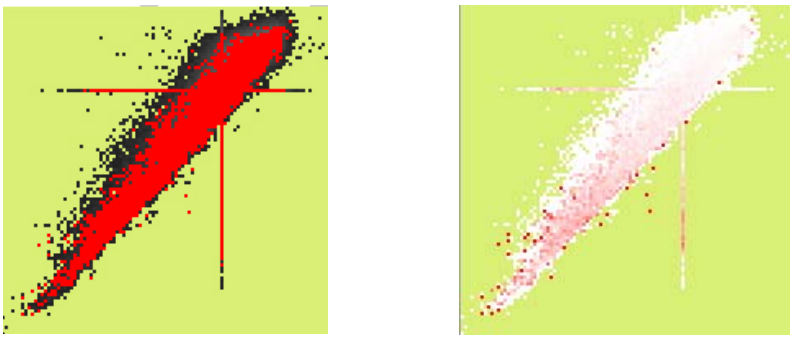


Figure 7: Transparency shape view permits users to see through the top subset, and get an understanding of both the subset shape and full data distribution.
(a) full data distribution of var0(x) and var1(y)
(b) full data distribution of var3 (x) and var4 (y).
(c), (d) subset shape is opaque
(e), (f) subset shape is transparent
Without transparent control, (d) displays only the shape of the subset. (f) displays the subset in a transparent color. High density points from the underneath data distribution can be observed by seeing through the transparent top subset.

2. Proportional density view displays more overplotting details of brushed subset. The count at each subset pixel is transferred to a transparency value based on its proportion to this pixel's whole data overplotting amount.

$$\alpha\text{-channel value} = \left(\frac{\text{subset density count}}{\text{whole data density count}}\right) \times 255 \qquad (4)$$

The transferred transparency value is between 0-255. A subset pixel containing few points will be drawn in nearly transparent, whereas a pixel covering most points will be displayed in an almost opaque color. As an example in figure 8, a shows the shape of the subset distribution, and b displays the heavy density subset area in red color darker than the light density subset area's color.



a b

Figure 8: Proportional density view displays more overlapping details of subset data by transferring overlapping proportion to transparency value.
(a) shape view displays only shape of a subset.
(b) proportional density view displays light density subset areas in light red, whereas heavy density subset areas in dark red.

3. Application

This application uses forest cover type data which is obtained from the UCI Knowledge Discovery in Databases Archive online. The data set was submitted on August 1998 by Jock A. Blackard and Colorado State University. The primary purpose of the data is to examine the relationship between forest cover type and some effective environment factors. There are 54 attributes for a given observation (30 × 30 meters spatial grid point) in the original data. For this application we chose only 7 variables which including 1 categorical variable and 6 quantitative variables:

covertype	categorical	forest cover type designation
elevation	quantitative	elevation in meters
aspect	quantitative	aspect in degrees azimuth
slope	quantitative	slope in degrees
dist.water	quantitative	vertical distance to nearest surface water features in meters
dist.road	quantitative	horizontal distance to nearest roadway in meters
dist.fire	quantitative	horizontal distance to nearest wildfire ignition points in meters

Forest cover type was determined from US Forest Service (USFS) Region 2 Resource Information System (RIS) data. Seven cover types are designated:

1	Spruce/Fir
2	Lodgepole pine
3	Ponderosa pine
4	Cottonwood/Willow
5	Aspen
6	Douglas-fir
7	Krummholz

We randomly split the 581012 cases of the data into two files with each contains 290506 cases. The two split files are stored in two different machines.

Figure 9 displays the density matrix for the forest cover type data. Two forest cover types are brushed. Cover type 4, which is cottonwood or willow, is brushed in pink; and the cover type 5, which is aspen, is brushed in light blue. In the black-and-white printout, cottonwood/willow subset is in medium gray, and aspen subset is in light gray. There is an obvious difference in elevation, where aspen always has higher elevation value than cottonwood/willow. In addition, the two subsets form different clusters in several plots. For example in the dist.road and elevation plot (5th row, 2nd plot), cottonwood/willow accumulate in a low elevation and low dist.road area, whereas aspen forms a few small clusters, especially with one having relatively high dist.road values.

Figure 5 shows a series of power value adjusting on a single plot, the plot of dist.water and elevation. Figure 5a is the plot with power value equals to 1. The high power value plot displays the shape of the data distribution and stripes of light density that persist into the center of the distribution. The power value is changed to be 0.4 in Figure 5b. With this value, high density points, which are in light color, are centered around a horizontal line with lower vertical distance water. Further checking on the gray value reveals that this distance is between -10 to +10 meters. Figure 5c is the plot with power value of 0.2. This plot shows not only the centering trend of the high density area, but also another interesting point: direction of the high density bands is consistent with the stripe direction of the data distribution.

4. Conclusion and Future Improvements

Limn Matrix is a system that interactively display density plots for large, distributed data. To solve the overplotting problem, Limn Matrix uniquely combines existing approaches from various fields. Large data is visualized in flat 2D histograms, and grey scale is used to represent the data density. Real time interactions are supported by Limn Matrix. Use of α-channel in highlighting gains Limn Matrix the capability to display data subsets with more distribution details. The novel hierarchical indexing system Limn Matrix proposed dramatically reduces the delay through network, and is used to handle large distributed data.

Future improvements on Limn Matrix's performance will focus on two aspects: multithread execution and random data sampling. Staley and Bahrami pointed out that splitting task between multiple threads of execution can speed up a task significantly (Staley and Bahrami 2003). Our future plan is to integrate multithread execution into our software, and use multiple threads to parallel process responses from distributed data. Using random sampling to reduce data cases is the other focus of our future work. The indexing system Limn Matrix used has already reduced large data cases into bins induced by screen real estate (Cook et al 2002). Our strategy is to apply random sampling on the basis of each bin. This strategy leaves us with a new question for proportional subseting. In a proportional subset view, each subset pixel's overplotting amount is transferred to a transparency value according to its proportion to this pixel's whole data overplotting amount. A fixed sampling size will make the proportion become a fraction of a constant over a pixel's whole data overplotting count. In that way a proportional subset view represents information only of the whole data but not the subset. Using a different sampling size strategy will be our future approach for applying random sampling to our software.

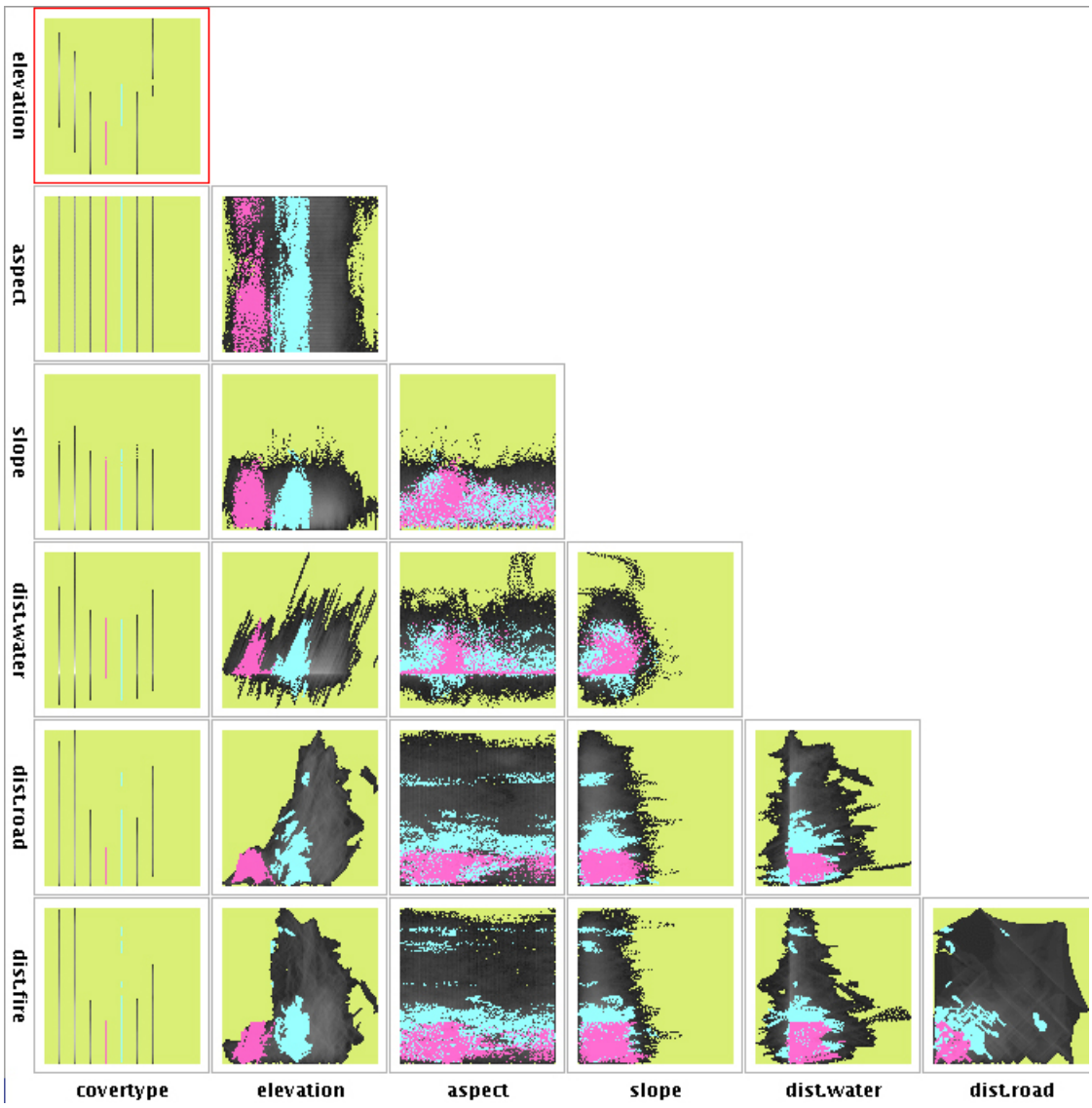


Figure 9: Density matrix with two cover types, cottonwood/willow (pink) and aspen (light blue) brushed.

Acknowledgements:

This research was funded by support from NSF grant #9982341

Reference:

Becker, R.A., and Cleveland, W. S.(1987a), *Brushing scatterplots*. Technometrics, 29(2):127-42.

Becker, R. A. Cleveland, W. S. and Wilks. A. R.(1987b), "Dynamic graphics for data analysis," *Statistical Science*, 2:355-395.

Carr, D. B., Littlefield, R.J., Nicholson, W.L., Littlefield, J.S. (1987). "Scatterplot Matrix Techniques for large N," *Journal of the American Statistical Association*, 82(398), 424-436.

Cleveland, W.S. and McGill, R. (1984), "The Many Faces of a Scatterplot," *Journal of the American Statistical Association*, **79**, 807-822.

Cook, D., Miller, L., Suarez, M., and Zhang, J.(2002), "Limn: using image video technology for visualizing a million cases of multivariate data,"

Huang, C., McDonald, J.A, and Stuetzle, W. (1997), "Variable Resolution Bivariate Plots," *Journal of Computational and Graphical Statistics*, **6**, 383-396.

Scott, D. W. (1979), **"**On optimal and data-based histograms," *Biometrika*, **66:**605-610.

Scott, D. W. (1992), *Multivariate Density Estimation*. Wiley.

Silverman, B.W.(1986), *Density Estimation for Statistics and Data Analysis,* Chapman and Hall, London, England.

Staley, S. and Bahrami. A. (2003), "Parallel Processing speeds Visualization of Scientific Data," *Scientific Computing & Instrumentation,* February.
http://www.rsinc.com/france/SC&I%20Steph%20S.%20Article.pdf

Stigler, S.M. (1986) *The History of Statistics: The Measurement of Uncertainty before 1900*, The Belknap Press of Harvard University Press: Cambridge.

Wegman, E.J. (1999), "Data mining and visualization: some strategies," *Bull Int Statist Inst*, **52**:223−226.

William D. Dupont and W. Dale Plummer Jr. (2003), "Density Distribution Sunflower Plots", *Journal of Statistical Software,* 8 (3): 1-5.

Brushed, Parallel Histograms: a Visualization Interface for Writing and Evaluating Predictive Rules in Multi-dimensional, Multi-type Feature Space

Kevin B. Pratt
Computer Science Innovations, Inc.
1235 Evans Rd., Melbourne, Florida 32904
(321) 676-2923; fax (321) 676-2355
kpratt@csi-inc.com

Abstract. We describe an interactive, graphic interface using brushed, parallel histograms that allows the user to discover patterns in multi-dimensional feature space and write rules that predict outcomes or classifications of interest. Previously generated rules, whether originated by the user, by a human expert, or by a machine learner, can be evaluated, generalized, and improved. Within a single display, the interface accommodates temporal, binary, categorical, and continuous type features, as well as outcomes after multiple temporal lags. The interface also displays conditional interdependencies of features and segments of feature values.

1 Introduction

There are many challenges in understanding and predicting behavior or classification when more than two variables are involved. Humans can work effectively in some multi-variate domains, for example intuitively applying rules to distinguish voices, where both biological evolution and life-time experience support expertise. However, humans can be overwhelmed in other multi-variate situations where the domain is relatively unfamiliar, such as economics.

Over the past 50 years, machine learners and mathematical models for prediction and classification have materially advanced understanding in multi-variate domains and the ability to write effective rules for those domains. More recently, interactive visualization techniques using computers provide a supplemental route to understanding multi-variate information.

The technique described here, brushed, parallel histograms, offers a novel interface between visualization and rule making. Specifically, it offers the ability to:

- State a rule directly by interacting with a visualization of multi-variate feature space.
- Evaluate a multi-variate rule that has been asserted by a human expert or by a machine learner such as a decision tree.

- Test the fragility of a rule with regard to any feature on which the rule depends by interactively relaxing constraints.
- Find opportunities to generalize a rule that was too narrowly stated.
- Specialize rules for sub-populations.
- Investigate reasons for rule failure.
- Track rule lifespans.
- Monitor multi-feature conditional dependencies.
- Observe time-lagged state or outcome differences.
- Work with mixed data types, including time, binary, categorical (un-ordered), and continuous types.

A primary benefit of brushed, parallel histograms is the ability to "see inside the box." The user can interact with rules and adjust their components to gain intuition into the nature of the rules and interdependencies of features, and to gain confidence that the rules are effective. This is an advantage over blackbox machine learners such as neural nets where rules are not intelligibly stated, and over greybox learners such as decision trees, where the robustness of a rule is difficult to ascertain.

A second benefit is easy recognition of dependence and independence among features without relying on sometimes fragile correlation measures. If brushing a segment of a feature results in the brushed histogram of another feature *not* looking like the population histogram of that feature, there is dependence. The dependence that is visually detected may be non-linear or non-continuous or not describable with a function (because, for example, a projection of clusters or branches in the higher dimensional space is occurring).

We demonstrate the interactive graphic interface on a set of recent stock market data detailed below. The goal of this paper is to demonstrate the capabilities of the technique, not to provide substantive rules for stock investment decisions.

Fig. 1 shows an example of the use of brushed, parallel histograms in eight dimensional feature space. The eight dimensional space is projected onto eight histograms placed on parallel axes. The histograms of the population of 12,870 points appear in red on each axis. The histograms of the brushed sub-population are shown in black to the left of the axes. The rule, stated beneath the graph, uses two features - market capitalization and institutional ownership. The outcomes of applying the rule, after four weeks and eight weeks, are shown on the two right axes. Statistics, mean and median, are shown below each axis for the population, in red, and for the brushed sub-population, in blue. The support for the rule, both in number of examples or percent of the population, is shown above the graph.

We will describe prior work in the field of high dimensional data visualization, introduce the brushed parallel histogram techniques, describe the interactive interface, and then describe demonstrative experiments on an eight dimensional set of stock market data. We conclude with thoughts on future directions.

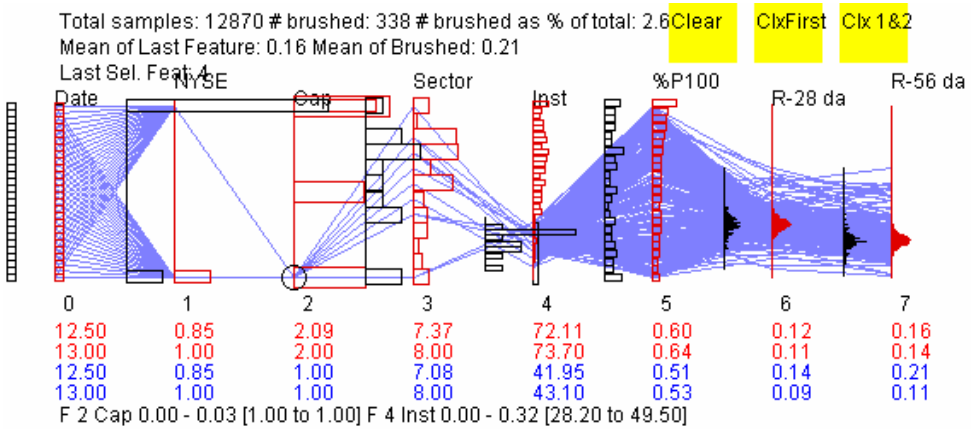


Fig. 1. Evaluation of a rule. The rule is, "Stocks with low market capitalization (axis 2) and low institutional ownership (axis 4) provide above average returns in the short run and long run (axis 6 and 7)." The rightmost axis shows the eight-week outcome: the histogram of the brushed set in black is similar to the population shown in red, the brushed mean (0.21) is higher than for the population (0.16), but the median (0.11) is lower. Conclusion: discard the rule as insignificant.

As with all interactive computer visualization tools, the capabilities of brushed parallel histograms are better understood with a live, interactive demonstration. The static figures here make dynamic changes from brushing less obvious.

2 Prior Work

There are many ways to visualize data in high dimensional feature spaces [1, 8, 9, 12]. More common approaches are the grand tour [2] of lower dimensional scatter graphs, star (also called hub and spoke) graphs, and parallel coordinate graphs. Parallel coordinate graphs originated more than 100 years ago. In 1899, d'Ocagne [16] proved the duality of a Cartesian point to a line in parallel coordinates. Friendly [8] illustrates that parallel coordinates were used in 1928 to display the relation of fourteen features of chemical concentration in blood. In the past twenty-five years, parallel coordinate graphs have re-appeared [4, 13, 14, 19], aided by computer graphics.

Interactive brushing helps identify points of interest and visualize connections of the values of different features [4, 15]. Hierarchical organization has been used to help interpret large parallel coordinate data sets [10]. When data points bundle together in visual "ropes" in a parallel coordinate graph, they can be analyzed as stating rules or clusters [21, 3].

When a parallel coordinate graph is used to display many data points, the overplotting of lines obscures the frequency and linkages represented by those lines, making analysis difficult [21]. Wegman [20] proposed creating density surfaces of parallel coordinate

graph lines to handle overplotting. The disadvantage of density surfaces is that they obscure the individual linkages and bundles of linkages between values or sets of values on adjoining axes. Berthold [3] displayed distribution using visual shading to indicate convex or plateau-type cluster centers. Our use of parallel histograms preserves the ability to see the links, while at the same time seeing the density distribution at the axis. The combination of histograms with brushing enables more complete viewing of interactions among features and relations to outcomes. [3, 11] suggested multi-dimensional brushing as a future approach to fuzzy rule discovery.

In prior work, we have proposed brushed, parallel histograms as a technique for visualizing concept drift [17], and for providing decision makers insight into how they might interevene in social processes to effect desired outcomes [18]. In this paper, we describe the more general capabilities of the technique.

3 Visualization by Brushed, Parallel Histograms

A parallel histogram graph consists of a parallel coordinate graph with a histogram superimposed on each axis. The histogram describes the frequency distribution of the points of the data set projected on that axis. Parallel coordinate graphs can be used to display up to a few hundred axes, although computer aided navigation is important where there are more than about twenty axes.

A point on a parallel coordinate graph is a set of connected line segments crossing all axes. The set of line segments is the dual of a point in Cartesian-Euclidean space of equal dimensionality. Because axes are parallel, visualization can escape the three axis constraint imposed by the orthogonality of Euclidean physical space.

The ordering of the axes in a parallel coordinate graph is arbitrary. The custom we follow puts the axis for the temporal feature to the left and axes for outcomes to the right. The user may re-order axes.

"Brushing" is the selection of a value or range of values on an axis. The user "brushes" by manipulating the mouse like a paint-brush applied to a canvas. The values included in the brushing are highlighted with color, or circled, and the user can view all linked values of all other features on the remaining axes. A sequence of brushing can constrain multiple features. Thus, it offers a visualization of conditional probabilities or nodes in a Bayesian network.

Because constraining the range of values on one (or more) features is the selection of a sub-sample of the data population, the frequency distribution of the brushed points on the other unconstrained axes may change. We display the histogram of the brushed points in black adjacent to the red histogram of the original population so that the user can see the changes that may occur.

For display, we normalize all data to the range 0 to 1(top of graph). An input mask distinguishes date, continuous, categorical, and binary data. Histograms for continuous feature values use an optimal bin width according to the normal bin width reference rule, without skew adjustment [19]. Thus, bin width = $3.5\ \sigma\ n^{-1/3}$, where σ is the standard

deviation and n is the number of points in the sample. While the brushed sub-sample often needs fewer bins under the normal bin width reference rule, users were visually confused by the variation in the bin width of the brushed histogram from the bin width of the histogram of the population. We use the same bin width for both.

4 Interactive Interface

The prototype tool containing the interactive interface is made of a settings panel (not shown here) and an interactive graphics panel. The settings panel permits re-sizing and zooming the graphics display and re-ordering axes. It also allows inspection of extensive statistics on the data, and viewing of scatter plots. Thus, the user can toggle between the brushed, parallel histogram view of the data and two and three dimensional (rotatable) brushed, scatter graphs for any pairs or triples of features.

The interactive graphics panel permits user interaction with the data. The user may brush any set of values of a feature, and any features, in any order. Simultaneously, the rule that describes the brushing appears below the graph. Statistical information is also simultaneously displayed.

The rule statement includes the axis number, the feature caption, the visual values on the axis (in the range of 0 to 1), and the underlying values from the original data. The rule states a description of a sub-population of the data set, based on an intersection of the brushed values of features. The user may remove portions of a rule, or entirely clear a rule statement. The user can apply and amend rules obtained from other sources, and thus can judge rules from subject matter experts or machine learning systems, based on the universe of historical data across all features.

In order to validate the visualization of the data, the prototype tool also displays supporting statistics.

- Population size
- Number of examples that support for the rule
- Fractional size of the brushed sub-population
- Mean and median of each feature for the population
- Mean and median of each feature for the brushed sub-population

Users find these statistics valuable in order to immediately recognize situations where the sample size is too small to be reliable, and to identify outcomes overly affected by outlying values. The stock outcomes shown below exemplify the potentially misleading nature of means. The medians shown below each mean may better indicate expected outcomes.

The parallel histogram graph tool was implemented in Java 1.4 using a data-view architecture that separates the view from the data tables. The architecture allows rapid switching from the brushed, parallel histogram graph to scatter plots or statistics. We use the scatter plots to drill down to bi- or tri-variate relationships implicated from the parallel histograms.

On a Windows XP operating system using a 1.7 GHz Pentium processor with 768 MB of RAM, window rendering after brushing takes less than one second on the stock data set of 13,000 points described below. We see similar performance on 9,000 data points in eighteen dimensions or 2,000 data points in fifty dimensions (not shown here).

5 Demonstrative Experiments

We apply brushed parallel histograms to gain insight into stock market returns and interactions of causative or descriptive features. Our goal is to demonstrate various uses of brushed parallel histograms.

We use data for the 500 Standard and Poor's listed stocks during the first half of 2003 (12,870 data points) across eight feature dimensions. The features are week, listing on the New York Stock Exchange, capitalization, market sector, institutional ownership, relative price, and return after four and eight weeks. Each feature, and its axis, is described in Table 1. The market sectors appear in Table 2.

Table 1. Axes used in graphs

Axis number	Caption	Data type	Explanation
0	Week	Temporal	Number of week of 2003. Week 0 = week ending January 3, 2003.
1	NYSE	Binary	0 = not listed on New York Stock Exchange; 1= listed on NYSE.
2	Cap	Categorical (ordered)	1 = market capitalization < $5B; 2 = between $5B and $10B; 3 = more than $10B
3	Sector	Categorical (un-ordered)	Twelve market sectors. See Table 2.
4	Inst	Continuous	% of shares owned by institutional investors at end of the 26 week period.
5	%P100	Continuous	Share price this week relative to the range of the share price over the past 100 days.
6	R-28 da	Continuous	Relative change in share price after four weeks (in percent per day units).
7	R-56 da	Continuous	Relative change in share price after eight weeks (in percent per day units).

Table 2. Market Sectors

	Sector Name		Sector Name
1	Basic Materials	7	Financial
2	Capital Goods	8	Health Care
3	Conglomerates	9	Services
4	Consumer Cyclical	10	Technology
5	Consumer Non-Cyc.	11	Transport
6	Energy	12	Utilities

We demonstrate interactive visualization of these activities:

- Rule evaluation
- Feature identification
- Hypothesis testing and rule writing
- Temporal change
- Lifespan of a rule
- Improvement of a rule for a sub-population

Fig. 1 demonstrates evaluation of a rule. The rule is, "Stocks with lower market capitalization (axis 2) and lower institutional ownership (axis 4) provide above average returns in the short run and long run (axis 6 and 7)." We brush the appropriate values on axes 2 and axis 4. After viewing the results, we conclude there is adequate support for the rule (338 examples), and the outcome means (0.14 and 0.21) are above the population means (0.12 and 0.16). Thus, our first impression is that the rule is valid and significant. However, the medians (0.09 and 0.11) are less than the population medians (0.11 and 0.14), and visual inspection of the outcome distributions on axis 6 and 7 shows no significant difference from the population distributions. Thus, we ultimately conclude the rule is not significant.

Fig. 2 demonstrates identifying features. "What are the characteristics of stocks having very low 8 week returns?" We brushed eight week returns below -0.33 (axis 7), and observe changes in the histograms on the other axes. We observe that the temporal distribution (axis 0) is strongly episodic, that the consumer sectors (Sector 4 and 5) are over-represented, the energy sector (Sector 6) is under-represented (axis 3), and that the lowest price range (axis 5) is underrepresented. We also note that institutional ownership in low return stocks is about the same as institutional ownership in all stocks. This exploration may help us write a future predictive rule in the next figure.

Fig. 3 shows testing a hypothesis derived from inspecting Fig. 2. The hypothesis is, "By selecting Sector 4 and 5 (axis 3) and high relative price (axis 5), we should obtain very low four week returns." After brushing, we conclude the hypothesis is correct. There are 417 examples supporting the conclusion that returns will be less than half the population returns, as shown by visually inspecting the outcome histograms, and comparing the outcome statistics for both mean and median below axes 6 and 7.

We can evaluate the differences of rule results across several time periods, as shown in Fig. 4. The rule is "Select stocks in the services sector (sector 9 on axis 3) that have a relative price below 30%." We condition the rule by a time frame, brushing a set of weeks on axis 0. The figure displays the variation in outcome of the rule for the first eight weeks of 2003 (Fig. 4, top), for the second eight weeks of 2003 (Fig. 4, middle) and for the third third weeks of 2003 (Fig. 4, bottom). The rule produces drastically different eight-week mean outcomes, ranging from 0.08 to 0.46 (axis 7).

Fig. 5 demonstrates visualization of the lifespan of a rule. We answer, "When did choosing stocks on the basis of relative price between 25 and 50% (axis 5) produce four week returns of over 0.76 (axis 6)?" We observe in axis 0 that the rule had a short lifespan. Its usage rose and fell during the middle weeks (axis 0) of the data set.

In Fig. 6, we find a rule that is much more effective for a sub-population. A rule that chooses stocks with relative price below 12% has mild effect on the four week return (axis 6) (Fig. 6, top). Perhaps surprisingly, the rule quadruples returns (axis 6) when applied to the non-NYSE sub-population (axis 1) (Fig. 6, bottom).

In addition to the experiments here, we have used brushed, parallel histograms to study the interaction of 18 features affecting stock market return, on a data set of 8000 points. In that instance, a year of weekly sets of approximately 140 stocks each were selected using filters on fundamental and technical information. We observed unexpected non-linear correlations among features, appearance and disappearance of modes in features during the year, and slow swings of behavior. We have also applied brushed, parallel histograms toward understanding how 44 morphological features of geologic material could predict mineral composition.

Brushed parallel histograms suffer from two common high-dimensional visualization limitations: difficulty in organizing axes and information loss by projection to lower dimensional space. We address organizing axes with options for manual axis ordering, ordering by Pearson correlation, chaining correlations, and ordering by axis modality counts, both respect to the population outcome, and the brushed outcome. Each ordering approach provides different insights about the relation of features. Information loss is unsolved. We observed new relationships in three dimensional scatter plots that we had not seen in the brushed parallel histograms.

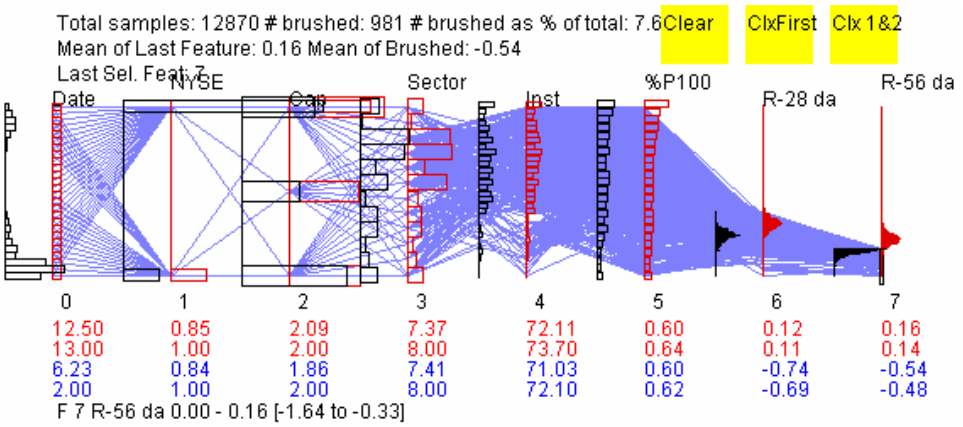


Fig. 2. Identify features. We explore, "What are the characteristics of stocks having very low 8 week returns?" We brushed returns below -0.33 on axis 7 (R-56 da), and observed changes in the histograms on the other axes. We observe that the temporal distribution (axis 0) is strongly episodic, that the consumer sectors (Sector 4 and 5) are overrepresented, the energy sector (Sector 6) is underrepresented (axis 3) and that the lowest price range (axis 5) is underrepresented. We also note that institutional ownership (axis 4) in low return stocks is about the same as institutional ownership in all stocks.

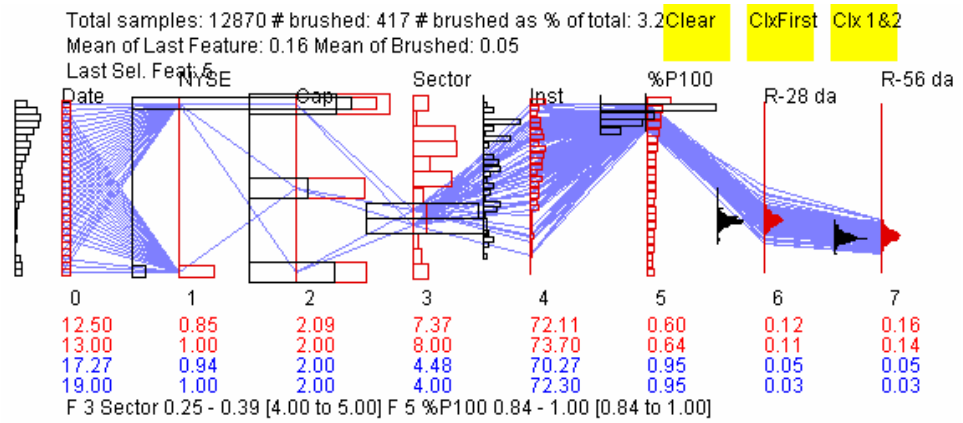


Fig. 3. Hypothesis Testing. Hypothesis: "By selecting Sector 4 and 5 (axis 3) and high relative price (axis 5), we should obtain very low four week returns." We conclude the hypothesis is correct. There are 417 examples supporting the conclusion that returns will be less than half the population returns, as shown by visually inspecting the outcome histograms, and comparing the outcome statistics for both mean and median below axes 6 and 7. (The visual comparison is largely lost in this small figure, but is clearer on the full size screen of the tool).

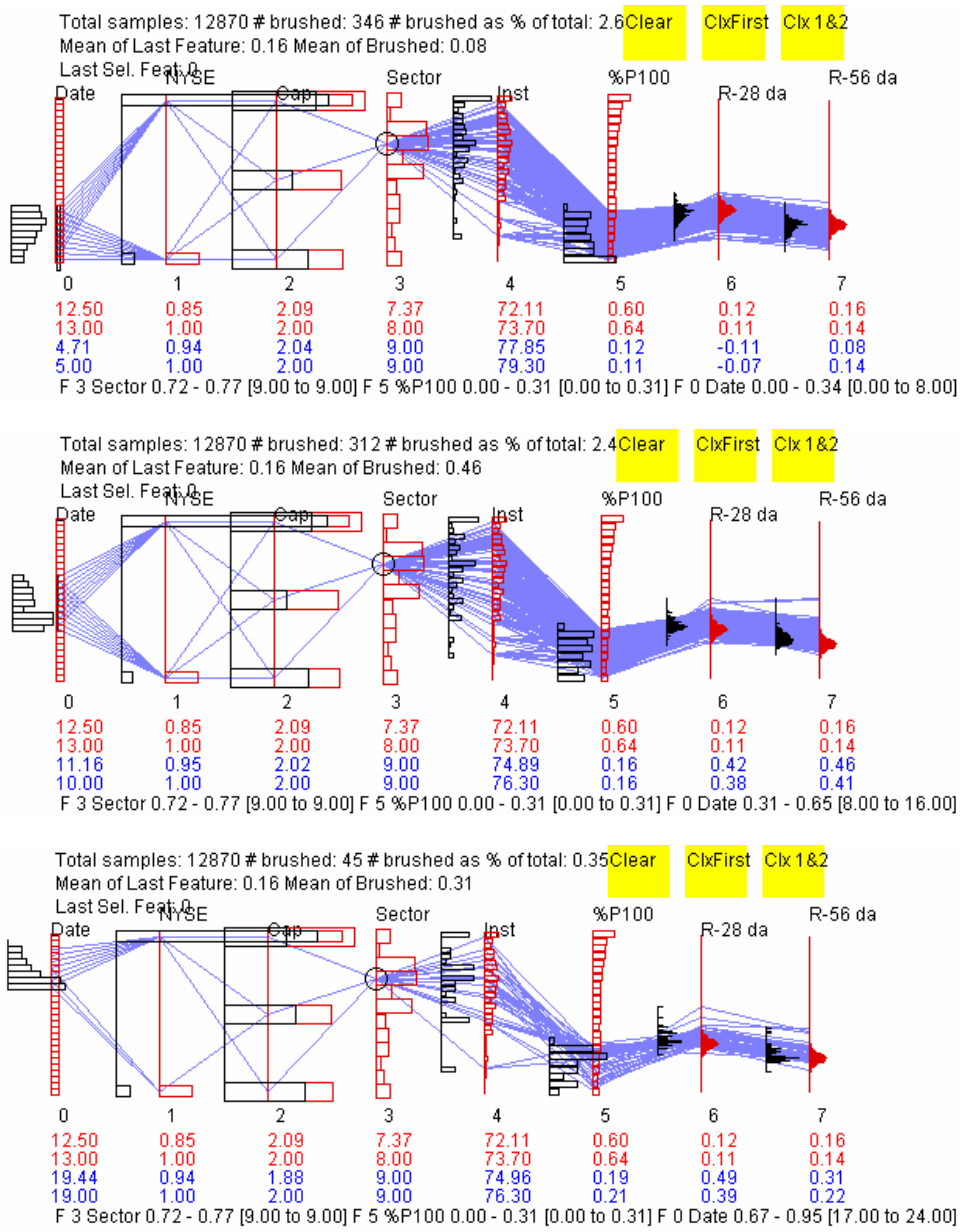


Fig. 4. Temporal change. The rule is, "Select stocks in sector 9 that have a relative price below 30%." We brush time periods on axis 0 to show the significant variation in outcome of the rule for the first 8 weeks of 2003 (top), for the second 8 weeks of 2003 (middle), for the third 8 weeks of 2003 (bottom).

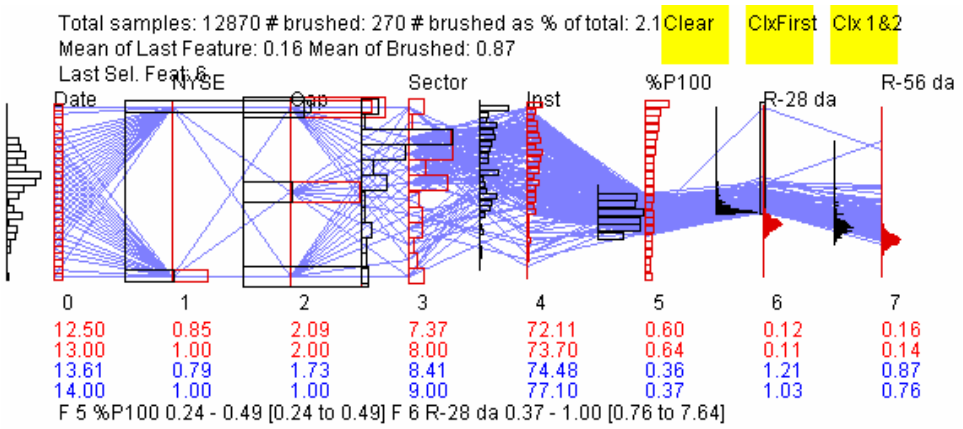


Fig. 5. Rule Lifespan. We answer, "When did choosing stocks on the basis of relative price between 25 and 50% (axis 5) produce 4 week returns of over 0.76 (axis 6)?" We observe in axis 0 that the rule was applicable only during the middle weeks of the data set.

6 Conclusion

We find that developing visualization techniques is a like sequence of turning corners into dark corridors. At each turn we find we need new tools to illuminate the path ahead. From experience with this tool, we find we need computer aid on where to brush to write the most effective rules. How to order axes in high dimensional space continues to be a common issue in all high dimensional visualization methods.

Data Sources

Stock price information was downloaded from Yahoo! Finance website. We calculated the relative price and return. We got information on membership in the Standard and Poors' 500 and market sectors from the Standard and Poors website. Information on exchange listing, market capitalization, and institutional share ownership came from Stock Investor Pro, a subscription software package obtained from the American Association for Individual Investors.

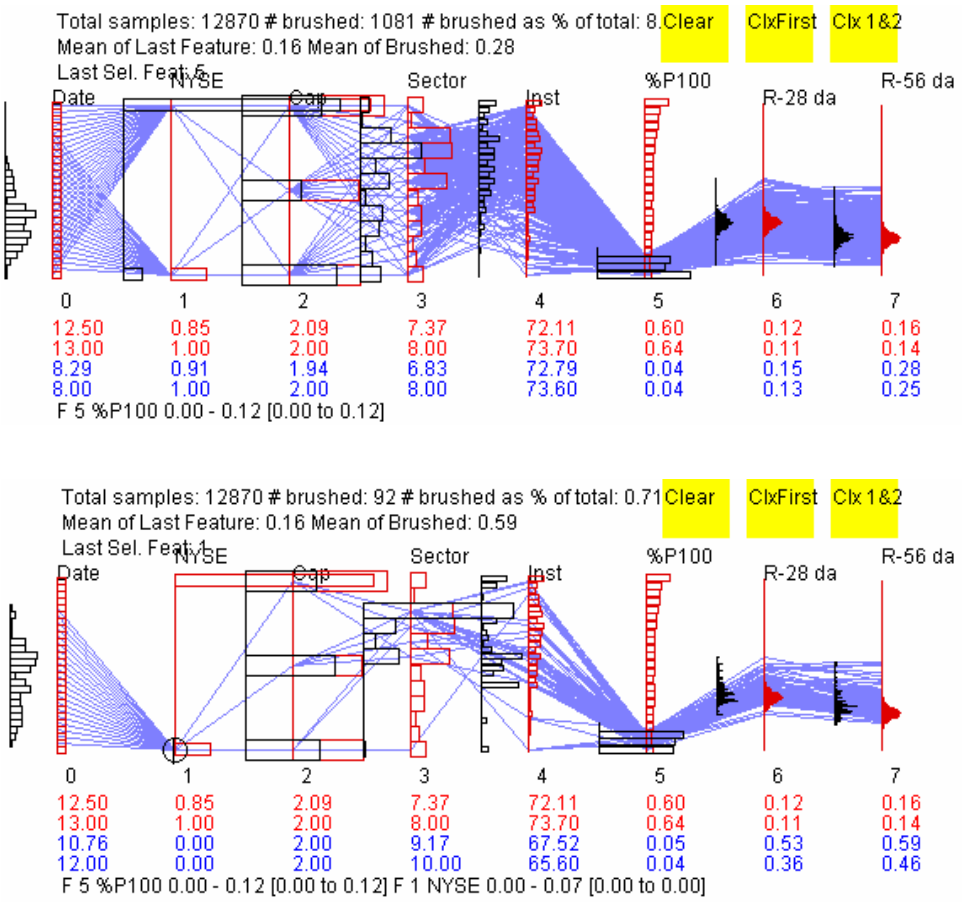


Fig. 6. Rules for sub-populations. A rule that chooses stocks with relative price below 12% has little effect on the four-week return (axis 6) (top), but quadruples returns (axis 6) for the non-NYSE sub-population (axis 1) (bottom).

References

1. Andrews, K.: Information visualization, Tutorial Notes Version 5th July 2002, IICM, Graz U. Technology, http://www.iicm.edu/ivis/, 2002.
2. Asimov, D.: The grand tour: a tool for viewing multidimensional data, DIAM J. on Scientific and Statistical Computing, v.61, pp. 128-143, 1985.
3. Chomut, T.: Exploratory Data Analysis in Parallel Coordinates, Los Angeles Scientific Center, 1987.

4. Berthold, M. R., Hall, L.O.: Visualizing Fuzzy Points in Parallel Coordinates. IEEE Transactions on Fuzzy Systems, v.11(3) pp.369-374, 2003.
5. Delmater, R., Hancock, M.: Data Mining Explained, Digital Press, Woburn MA, 2001.
6. Duda, R., Hart, P., Stork, D.: Pattern Classification, Wiley, New York, 2001.
7. Eick, S.G., Karr, A.F.: Visual scalability, J. Computational and Graphical Statistics, v. 11(1) p. 22, 2002.
8. Friendly, M., Denis, D. J.:Milestones in the history of thematic cartography, statistical graphics, and data visualization, U. York, Dept. of Mathematics, Gallery of Data Visualization, http://www.math.yorku.ca/SCS/Gallery/milestone/, 2003.
9. Friendly, M.: Corrgrams: exploratory displays for correlation matrices, Am. Statistician, v. 56(4), pp. 316-325, 2002.
10. Fua, Y., Ward, M., Rundensteiner, E.: Hierarchical parallel coordinates for exploration of large datasets, In Proceedings on Visualization '99, pp. 43-50, 1999.
11. Fua, Y-H., Ward, M. O., Rundensteiner, E. A.: Structure-based brushes: A mechanism for navigating hierarchically organized data and information spaces. IEEE Transactions on Visualization and Computer Graphics, v.6(2), pp 150-159, 2000.
12. Grinstein, G., Trutschl, M., Cvek, U.: High-dimensional visualizations, In Proceedings of Workshop on Visual Data Mining, ACM Conference on Knowledge Discovery and Data Mining, pp.1-14, 2001.
13. Inselberg, A.: The plane with parallel coordinates, Special Issue on Computational Geometry, Visual Computer 1, pp. 69-97, 1985.
14. Inselberg, A.: Data Mining and Visualization of High Dimensional Data, In Proceedings of Workshop on Visual Data Mining, ACM Conference on Knowledge Discovery and Data Mining, pp. 65-81, 2001.
15. Martin, A. R., Ward, M. O.: High dimensional brushing for interactive exploration of multivariate data, In Proceedings of IEEE Conf. on Visualization, pp. 271-278, 1995.
16. d'Ocagne, M.: Traite de nomographie: Theorie des Abaques, Applications Pratiques, Gauthier-Villars, Paris, 1899.
17. Pratt, K. B., Tschapek, G.: Visualizing Concept Drift. In Proceedings of ACM Conference on Knowledge Discovery and Data Mining (KDD 2003), pp. 735-740, 2003.
18. Pratt, K. B.: Interactive Evaluation of Social Process Intervention Options. In Proceedings of International Conference on Knowledge Engineering (IKE 2003), 2003.
19. Scott, D. W.: Multivariate Density Estimation, Wiley, NY, 1992.
20. Wegman, E. J.: Hyperdimensional Analysis Using Parallel Coordinates, J. Am. Statistical Assn, v. 85, pp. 664-675, 1990.
21. Zhao, K., Liu, B.: Visual Analysis of the Behavior of Discovered Rules, In Proceedings of Workshop on Visual Data Mining, ACM Conference on Knowledge Discovery and Data Mining, pp.59-64, 2001.

Interactive Visualization of Frequent Itemsets and Association Rules

Li Yang

Department of Computer Science, Western Michigan University
li.yang@wmich.edu

Abstract. Frequent itemsets are characterized by a border in itemset lattice which separates the frequent itemsets from infrequent ones. Visualizing this long border is a challenge. This paper introduces another border, the display border, in the itemset lattice which tells whether an itemset is displayable or not. The display border specifies another set of itemsets which have downward closure property. Only those itemsets that are within the interaction of the sets of itemsets specified by these two borders are considered to be visualized. Unlike the border of frequent itemsets, the display border can be changed by user interaction. This approach is capable of visualizing a large number of frequent itemsets and association rules by displaying only those ones whose items are interesting to the user.

1 Introduction

For years, researchers have developed many tools to visualize association rules. However, few of these tools can handle more than dozens of rules, and none of them can effectively manage many-to-many rules. Association rules are difficult to visualize. The difficulty of this problem comes from the following two fundamental challenges of association rules to information visualization:

Challenge 1 Association rules are defined on elements of the power set of the set of items that reflect many-to-many relationships among these items. With the absence of an effective visual metaphor to represent many-to-many relationships, association rules pose fundamental challenges to information visualization.

Challenge 2 Frequent itemsets, from which association rules are derived, are identified by a border of support in the itemset lattice which separates frequent itemsets from infrequent ones. Depending on the support factor and total number of items, this border could become very long. This raises the concern that the limited space of computer screen may not give enough perceptual places so that each frequent itemset or association rule can be properly displayed.

This paper presents our approaches to solve these problems and discusses topics for future research. For the first challenge, we have evaluated the major

visual metaphors and found that parallel coordinates give a good approach to handle many-to-many relationships. Parallel coordinates have been used in [1] for the visualization of sequential patterns. In a similar way, an association rule can be visualized by connecting its items, one on each parallel coordinate, with polynomial curves. In the presence of item taxonomy, furthermore, an item taxonomy tree can be used to replace each coordinate. In this way, we can visualize generalized association rules with item taxonomy.

For the second challenge, our solution is to introduce a display border in the itemset lattice. Figure 1 illustrates the borders on the itemset lattice. The same as the frequent itemsets, the set of itemsets specified by the display border is downward closed, that is, an itemset within the border implies that its subsets are also within the border. It is thus theoretically straightforward that the intersection of the set of itemsets specified by the display border and the set of frequent itemsets is also downward closed. Therefore, we can visualize only frequent itemsets that are within the display border. By changing the display border through user interaction, we can selectively visualize frequent itemsets and association rules.

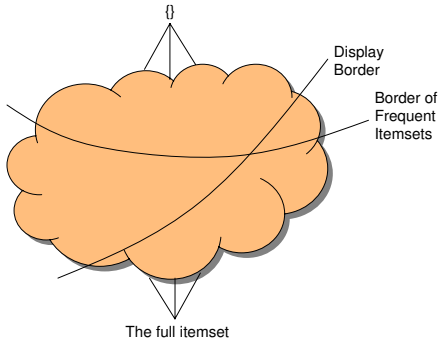


Fig. 1. The two borders in the itemset lattice.

In this paper, we show that a display border can be changed by interactive expanding or shrinking the displayed item taxonomy tree. The result 3D visualization has following features: (1) it is capable of visualizing large itemsets and many-to-many association rules; (2) it is capable of visualizing a large number of frequent itemsets and association rules by displaying only those ones whose items are selected by the user; (3) the closure properties of frequent itemsets and association rules are inherently supported by this visualization such that the implied ones are not visualized.

2 Using Parallel Coordinates for Challenge 1

Parallel coordinates [2–6] represent a simple way to visualize high dimensional data. It has also been used to visualize classification rules[7]. Although its major application so far is to visualize relational records all with equal number of attributes, it could be used to visualize data with variable lengths such as frequent itemsets and association rules. This paper proposes the visualization in the following way: first, all items are listed along a vertical coordinate; then the result coordinate is repeated evenly in the horizontal direction so that there are enough coordinates to host the longest frequent itemset or the longest association rule. An itemset or an association rule can then be visualized as a polygonal line connecting all items in the itemset or the rule.

One important feature of this visualization is that it provides a way to visualize only the maximum frequent itemsets. Their subsets are implied and not displayed. As an example, Figure 2(a) illustrates the visualization of three frequent itemsets *adbe*, *cdb* and fg as polygonal lines. Figure 2(b) illustrates the visualization of an association rule $ab \rightarrow cd$ in a similar way. An association rule is visualized as one polygonal line representing items on its left-hand side (LHS), followed by an arrow connecting another polygonal line representing items on its right-hand side (RHS).

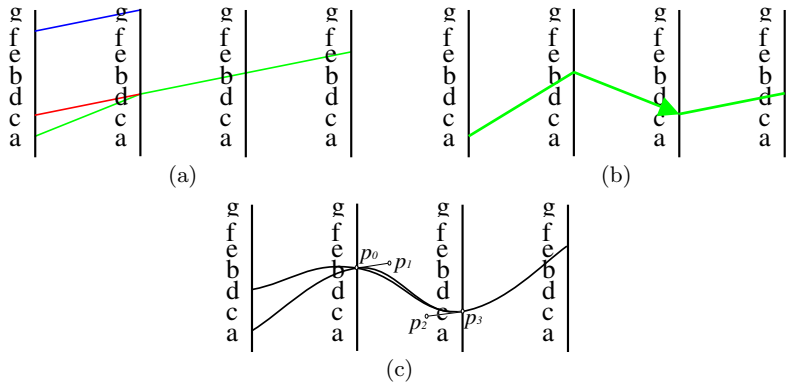


Fig. 2. Visualizing frequent itemsets(a) and an association rule(b) using polylines, and association rules using Bezier curves (c).

A problem with polygonal line representation of itemsets is that, if two or more itemsets have parts in common, there is no way to distinguish one from the other. For example, *cdb* and *cdbe* in Figure 2(a) are not distinguishable. This problem can be solved by using polynomial curves, for example, cubic Bezier curves, instead of polygonal lines to represent itemsets. Figure 2(c) visualizes the rule $ab \rightarrow ce$ by using three Bezier curves. We display these Bezier curves in such a way that they together offer C^1 continuity of the visualization at

all coordinates. Another association rule illustrated in Figure 2(c) is $db \rightarrow ce$, which shares three items with the rule $ab \rightarrow ce$. To keep C^1 continuity at the position of the first shared item b, the bc segments in the two association rules are visualized in two different Bezier curves. However, the second shared segments ce in the two rules coincide completely with each other. We think this is acceptable to distinguish two association rules with parts in common while keeping the display reasonably clean.

Another problem is that the visualization can easily become messy because these curves intersect with each other. We can minimize the intersection of these curves by organizing items along a coordinate in the following way: First, items are arranged by item groups so that the items belonging to the same group are displayed together. Since item groups do not share items, Bezier curves of itemsets in different item groups will never intersect with each other. In this way, the curves are organized into "horizontal bands" according to item groups. Second, items within the same group are arranged upward in descending order according to their supports. In the same way, items in an itemset are also arranged in descending order according to their supports. In this way, we make sure that the slopes of a polygonal line are always positive. This helps to reduce the chance of tangled intersection with other polygonal lines.

We tested this approach by using a supermarket transaction data set in the IBM DB2 Intelligent Miner software. Figure 3 visualizes discovered association rules when the minimum support is set to 2% and the minimum confidence is set to 30%. In the figure, the left coordinate represents the LHSs (one item only) of the rules, the right three coordinates represent the RHSs of the rules. The slopes of curves between the right three coordinates are always positive. There are totally 57 association rules displayed, which represent 340 association rules discovered from the data set. The visualization is interactive such that each displayed rule is selectable by clicking mouse. The selected rules and all of its implied rules will be displayed on the right side of the window together with their parameters.

3 Introducing the Display Border in the Itemset Lattice for Challenge 2

It is well known that the power set $\mathcal{P}(I)$ of the set I of all items is a lattice structure $< \mathcal{P}(I), \subseteq >$ where the subset relationship $\subseteq$ gives a partial order. Because the support factor is monotonic to the subset relationship, that is, $\text{supp}(A) \geq \text{supp}(B)$ if $A \subseteq B$, frequent itemsets are located in the upper part of the lattice whereas infrequent ones are located in the lower part of the lattice. Therefore, there is a border which separates the frequent itemsets from the infrequent ones [8]. Frequent itemsets on the border are maximum and have the property that every itemset above (and include) them is frequent, while every itemset below any of them is not. The existence of such a border is guaranteed by the lattice structure and the downward closure property. It is independent of any particular database or minimum support value.

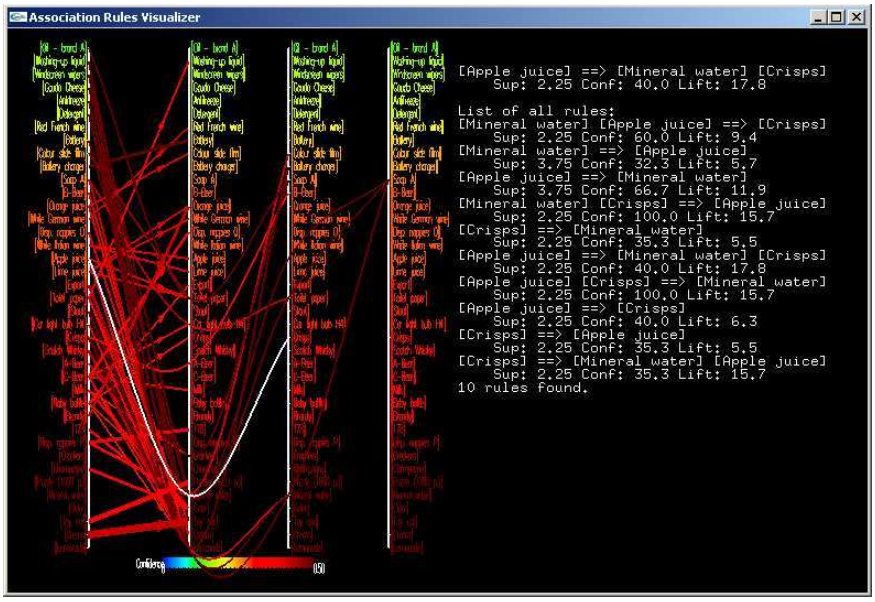


Fig. 3. Visualizing many-to-many association rules.

Depending on the minimum support value and the total number of items, the frequent itemset border may become very long and thus difficult to visualize. An idea to deal with this problem is to introduce another border in the itemset lattice, which specifies whether an itemset is displayable or not. Only those frequent itemsets that are within this border are considered for visualization. This border should be able to change by user interaction. By changing the border, we hope that we can visualize a large number of frequent itemsets and association rules by displaying only those ones that the user is interested in.

The only requirement on the border is that it must satisfy the downward closure property, that is, all subsets of an itemset within the border must also be within the border. One possible such border that we can introduce is through item taxonomy. Let I be a set of all items and IT an item taxonomy on I [9]. Define the *generalized power set* $\mathcal{GP}(I, IT)$ so that it contains all itemsets and their ancestor itemsets, i.e., $\mathcal{GP}(I, IT) = \mathcal{P}(I) \cup \{\text{all ancestor itemsets } \widehat{A} \text{ of } A | \forall A \in \mathcal{P}(I)\}$. Define a partial order $\preceq$ as: (1) $A \preceq B$ if $A \subseteq B$; (2) $\widehat{A} \preceq A$. Then $< \mathcal{GP}(I, IT), \preceq >$ is a lattice, a generalized itemset lattice. It is easy to verify that the support factor is monotonic to $\preceq$. Therefore, there is a border in $< \mathcal{GP}(I, IT), \preceq >$ which separates frequent itemsets from infrequent ones. For example, Figure 5 shows the generalized itemset lattice $< \mathcal{GP}(I, IT), \preceq >$ over the items $I = \{a, b, c, d\}$ under the item taxonomy tree IT shown in Figure 4. In Figure 5, we use straight lines to denote subset relationships and arcs to denote ancestor relationships.

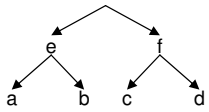


Fig. 4. A simple item taxonomy tree.

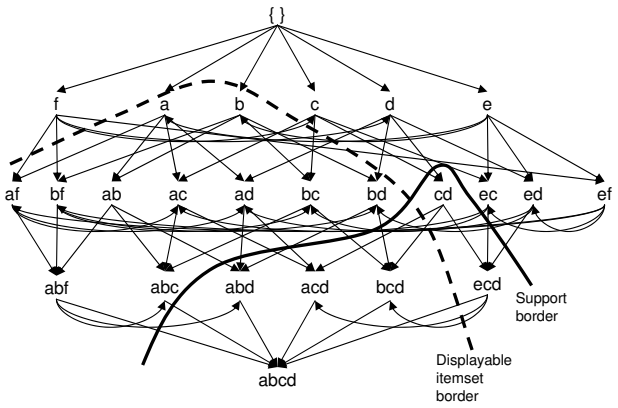


Fig. 5. A generalized itemset lattice $< \mathcal{GP}(I, IT), \preceq >$.

We can visualize frequent itemsets with item taxonomy by replacing each coordinate with a visualization of taxonomy tree. A taxonomy tree can be visualized by starting from its root and stopping at any internal nodes or leaf nodes. One possible user interaction is to expand or shrink the displayed taxonomy tree. This interaction introduces another border in the generalized itemset lattice.

An itemset is called *displayable* if all items in the itemset are shown in the visualization of the taxonomy tree. The displayable property is downward closed in the generalized itemset lattice $< \mathcal{GP}, \preceq >$: if an itemset A is displayable, any itemset B such that $B \preceq A$ is also displayable. Therefore we have now two borders in the generalized itemset lattice $< \mathcal{GP}, \preceq >$: one border separates frequent itemsets from infrequent ones; the other border separates displayable itemsets from un-displayable ones. For example, let us assume that only the items c, d, e, f are displayable in the item taxonomy in Figure 4, this specifies a border of displayable itemsets in the generalized itemset lattice, which is marked by a dashed line in Figure 5.

As we expand or shrink the displayed item taxonomy tree, the border of displayable itemsets will change. This gives us a way to selectively visualize the frequent itemsets whose items are among those that we are interested in. A visualization system is designed such that only the *maximum, displayable, frequent* itemsets are displayed. Other displayable frequent itemsets are implied by the visualized frequent itemsets. In the generalized itemset lattice $< \mathcal{GP}, \preceq >$, these maximum displayable frequent itemsets must reside on the border of intersection

of the set of frequent itemsets and the set of displayable itemsets. Taking the two borders in Figure 5 as example, ec and ed are the two displayable frequent itemsets that should be visualized. The other displayable frequent itemsets are implied by these two itemsets.

Closure Properties of Association Rules. If we consider the sub-lattice formed by a frequent itemset, all association rules generated from the frequent itemset have also a closure property. This happens when we consider association rules that have more than one items in the RHSs. For example, if $c \rightarrow ab$ is a rule that can pass support and confidence tests, then $bc \rightarrow a$ and $ac \rightarrow b$ are also association rules that can pass the same support and the same confidence tests. This means that the association rules generated from a frequent itemset are downward closed according to their RHSs in the sub-lattice formed by the frequent itemset using the subset relationship as a partial order. Figure 6 illustrates a border of RHSs of association rules generated from a frequent itemset $\{a, b, c, d\}$. The border shows that ab and ac are two maximum RHSs which means $cd \rightarrow ab$ and $bd \rightarrow ac$ are two non-redundant association rules.

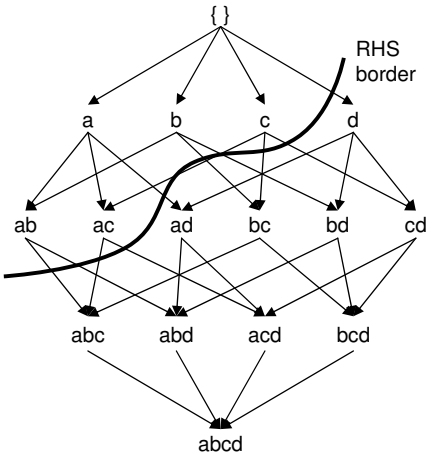


Fig. 6. The border of RHSs of association rules generated from a frequent itemset $\{a, b, c, d\}$.

4 Visualizing Frequent Itemsets and Association Rules

Frequent itemsets and association rules with item taxonomy can be visualized by replacing each parallel coordinate with an item taxonomy tree. The displayed taxonomy tree can be expanded or shrunk by user interaction. To keep the visualization clean, we choose to display only maximum frequent itemsets. Frequent

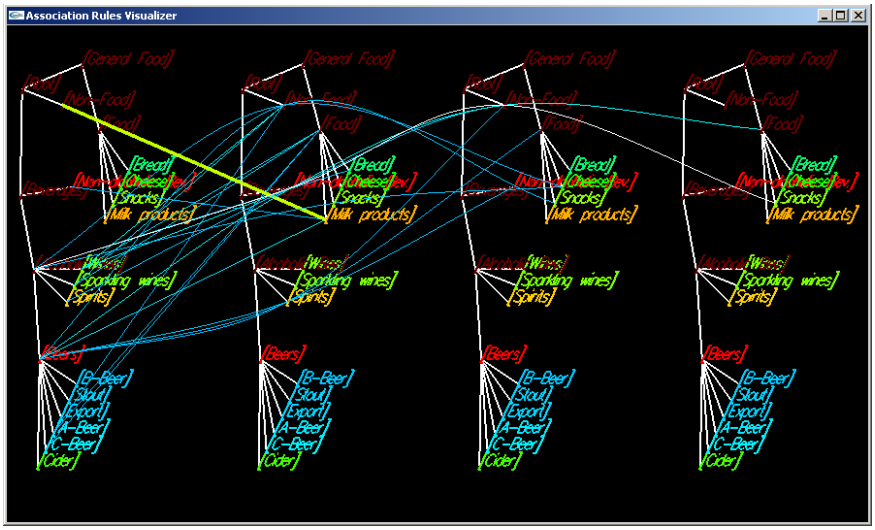


Fig. 7. Visualization of non-redundant frequent itemsets on the displayed item nodes.

itemsets are visualized so that a frequent itemset is displayed if the itemset is on the border in the generalized itemset lattice $< \mathcal{GP}(I, IT), \preceq >$ which separates displayable frequent itemsets from the rest ones. Specifically, a frequent itemset is displayed if and only if:

1. the frequent itemset is displayable; and
2. the frequent itemset is not the subset of any other displayable frequent itemsets; and
3. the frequent itemset is not the ancestor itemset of any other displayable frequent itemsets.

By interactively expanding or shrinking the displayed taxonomy tree, therefore, we can change the display border and thus visualize those frequent itemsets that we are interested in.

The test data we use contain 80 items which are leaf nodes of a 4-level taxonomy tree. 496 frequent itemsets are discovered when the minimum support is set to 5%. Figure 7 visualizes some of these frequent itemsets on the expanded taxonomy tree. The color of the name of each item or item category represents its support. Displayable frequent itemsets are visualized as smooth connections of Bezier curves. The line-width of a curve represent the support value of the corresponding itemset. Only non-redundant displayable frequent itemsets are visualized. Panning and zooming can be applied to the 3D visualization.

Similarly, a set of association rules can be visually explored by displaying only those ones whose items are shown in the displayed item taxonomy tree. However, association rules do not have those closure properties of frequent itemsets. Pruning redundant association rules has been reported in [10]. It is also

reported [10–12] that frequent itemsets are more fundamental than association rules in data mining. Instead of directly visualizing association rules, therefore, we may prefer to visualize frequent itemsets. However, we can generate and display association rules generated from a frequent itemset in a separate window if the user select the frequent itemset through interaction. Since the association rules generated from a frequent itemset have the downward closure property according to their RHSs, we may check for rule redundancy so that only those rules which have the maximum RHSs are displayed. In this way, we keep our visualization of association rules readable by reducing the number of rules displayed to only those ones that are produced from a frequent itemset and are representative of the other rules.

For example, the user can pick up an itemset shown in Figure 7 by clicking anywhere on its curve segments. The association rules generated from the itemset could then be visualized in a separate window. Figure 8 gives an example of visualizing discovered association rules with item taxonomy. Association rules are aligned according to where the RHSs separate from the LHSs. In this example, the left two coordinates represent the LHSs of the rules and the right two coordinates represent the RHSs of the rules. Support of a rule is represented by line width. Confidence of a rule is represented by color.

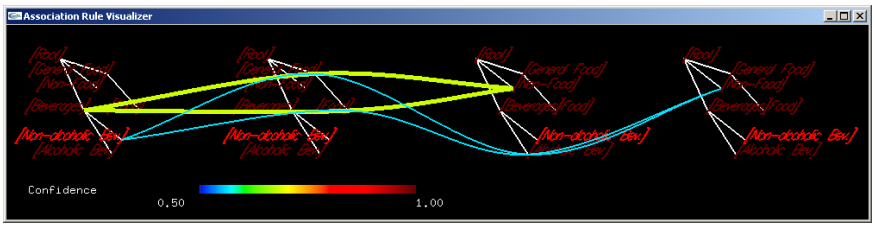


Fig. 8. Visualizing association rules.

In summary, we have introduced two kinds of interactions in our visualization: First, the displayed taxonomy tree can be expanded or shrunk by clicking a node in the tree. This interaction will change the display border in the generalized itemset lattice and, consequently, select another set of frequent itemsets to display. By interactively selecting items to display, we visualize only a minimal set of frequent itemsets whose items are among those that we are interested in. Second, when we click a displayed frequent itemset, the association rules generated from the frequent itemset could be visualized in a separate window. Among those rules, only those rules with the maximum RHSs are displayed.

An experimental software system has been developed to visualize frequent itemsets and association rules. We use IBM DB2 Intelligent Miner to generate frequent itemsets. Because Intelligent Miner does not produce association rules with multiple items on the RHS, the experimental system reads item taxonomy and frequent itemsets produced by Intelligent Miner and generates association

rules in real time as it visualizes these rules. Because each mouse click changes the displayed item taxonomy tree and the set of rules to display, the generated association rules are pruned each time the displayed item taxonomy tree is changed.

5 Discussion and Future Work

We have shown how to visualize many many-to-many association rules by taking advantage of the underlying properties of these relationships and by mapping them into visual elements of a particular visualization metaphor. Parallel coordinates offer a good support for the partial order properties of frequent itemsets and those of association rules. We further use interactively expandable taxonomy tree to deal with the problem of large number of items. Two kinds of user interactions are allowed: First, a user can expand or shrink the displayed taxonomy tree, and consequently, select frequent itemsets or rules to display. Second, each displayed frequent itemset is selectable to visualize the association rules generated from the frequent itemset. Basically, the first interaction introduces another border in the generalized item lattice, which separates displayable itemsets from non-displayable ones. Only those frequent itemsets on this border are displayed. By changing this border, we can selectively visualize frequent itemsets and association rules that we are interested in.

This approach of visualization brings topics for future research on how interaction and redundancy might be exploited. One research topic is how to explore more effective closure properties and introduce borders to selectively visualize itemsets or rules. The changes of these borders should be associated with meaningful user interaction. Another research topic is how to prune uninteresting itemsets and rules. The topic comes from the fact that some redundant itemsets or rules may be more practically interesting than their respective displayed ones.

Redundancies exist in our visualization. For example, the rules generated from A may be redundant if A is a subset or an ancestor itemset of another frequent itemset. For example, if $a \rightarrow bc$ is a rule, then $a \rightarrow b$, $a \rightarrow c$, $a \rightarrow \hat{b}c$, and $a \rightarrow b\hat{c}$ are all valid rules. These association rules are displayed by the experimental system if the user select the corresponding frequent itemsets.

This visualization approach is fundamentally different to the original use of parallel coordinates that was designed to visualize relational data. The original use of parallel coordinates enables the detection of plane patterns in high dimensional space through the structure of displayed polygonal lines based on point-line duality. It would be interesting to establish some similar dualities in the visualization of frequent itemsets and association rules. Furthermore, parallel coordinates offer several freedoms that have not been exploited yet. For example, the proposed visualization does not allow the item taxonomy tree for each coordinate be manipulated individually; the quality of the visualization of frequent itemsets depends on the ordering of items. These freedoms should be exploited to develop a better visualization technique.

Finally, this paper presents only the principles of using parallel coordinates for association rule visualization without further considering visualization details. For example, should the support and confidence values be represented by color, line width, or dashed line? Validation of the effectiveness of the proposed approach needs a usability study. In addition, usability study would be needed to have a better strategy of rule pruning because interestingness is a subjective matter. It is one thing to say that we can visualize association rules. It is another thing to confirm that the visualization is useful. Assessing the usability of visualization entails carrying out formal user studies which is part of future work.

References

1. Wong, P.C., Cowley, W., Foote, H., Jurrus, E., Thomas, J.: Visualizing sequential patterns for text mining. In: Proc. IEEE Symp. Information Visualization (InfoVis'00). (2000)
2. Inselberg, A.: The plane with parallel coordinates. The Visual Computer **1** (1985) 69–91
3. Inselberg, A., Reif, M., Chomut, T.: Convexity algorithms in parallel coordinates. Journal of the ACM **34** (1987) 765–801
4. Inselberg, A.: Parallel coordinates: A tool for visualizing multi-dimensional geometry. In: Proc. 1st IEEE Conf. Visualization, San Francisco, CA (1990) 361–375
5. Inselberg, A., Dimsdale, B.: Multi-dimensional lines: Proximity and applications. SIAM J. Appl. Math. **54** (1994) 559–596
6. Inselberg, A.: Visualizing high dimensional datasets and multivariate relations (tutorial). In: 6th ACM Conf. on Knowledge Discovery and Data Mining (KDD'00). (2000)
7. Han, J., Cercone, N.: Ruleviz: A model for visualizing knowledge discovery process. In: Proc. 6th ACM Conf. on Knowledge Discovery and Data Mining (KDD'00). (2000) 244–253
8. Toivonen, H.: Sampling large databases for association rules. In: Proc. 22th Int. Conf. Very Large Data Bases (VLDB'96). (1996) 134–145
9. Srikant, R., Agrawal, R.: Mining generalized association rules. In: Proc. 21st Int. Conf. Very Large Data Bases (VLDB'95), Zurich, Switzerland (1995) 407–419
10. Liu, B., Hsu, W., Ma, Y.: Pruning and summarizing the discovered associations. In: Proc. 5th ACM Conf. on Knowledge Discovery and Data Mining (KDD'99). (1999) 145–154
11. Bayardo, R., Agrawal, R., Gunopulos, D.: Constraint-based rule mining in large, dense databases. In: Proc. Inter. Conf. on Data Engineering (ICDE'99). (1999)
12. Morishita, S., Sese, J.: Traversing itemset lattice with statistical metric pruning. In: Proc. 19th ACM Symp. on Principles of Database Systems (PODS). (2000) 226–236

Enhancing Data Visualization Techniques

José Fernando Rodrigues Jr., Agma J. M. Traina,
Caetano Traina Jr.

Computer Science Department
University of Sao Paulo at Sao Carlos - Brazil
Avenida do Trabalhador Saocarlense, 400
13.566-590 São Carlos, SP - Brazil
e-mail: [junio, cesar, caetano, agma]@icmc.usp.br

Abstract. The challenge in the Information Visualization (*Infovis*) field is two-fold: the exploration of raw data with intuitive visualization techniques and the discover of new techniques to enhance the visualization power of well-known infovis approaches, improving the synergy between the user and the mining tools. This work pursues the second goal, presenting the use of interactive automatic analysis combined with visual presentation. To demonstrate such ideas, we present three approaches aiming to improve multi-variate visualizations. The first approach, named *Frequency Plot*, combines frequencies of data occurrences with interactive filtering to identify clusters and trends in subsets of the database. The second approach, called *Relevance Plot*, corresponds to assign different shades of color to visual elements according to their relevance to a user's specified set of data properties. The third approach makes use of basic statistical analysis presented in a visual format, to assist the analyst in discovering useful information. The three approaches were implemented in a tool enabled with refined versions of four well-known existent visualization techniques, and the results show an improvement in the usability of visualization techniques employed.

1. Introduction

The volume of digital data generated by worldwide enterprises are increasing exponentially. Therefore, together with concerns about efficient storage and fast and effective retrieval of information, comes concerns about how to get the right information at the right time. That is, companies can gain market space by knowing more about their clients' preferences, usual transactions, and trends. Well-organized information is a valuable and strategic asset, providing competitive advantage on business activities.

The Information Visualization (Infovis) techniques aim to take advantage of the fact that humans can understand graphical presentations much more easily and quickly, helping users to absorb the inherent behavior of the data and to recognize relationships among the data elements. As a consequence, such techniques are becoming more and more important during data exploration and analysis.

Besides being voluminous, the data sets managed by the information systems are frequently multidimensional. Thus, the process of analyzing that data is cumbersome, as this type of information is constituted by many features and properties that are hard to be apprehended by the use as a whole. Therefore, in order to bypass the intricate task of analyzing complex data sets, we propose an alternative course for the visualization tasks, bringing together automatic data analysis and the presentation of the results of this

analysis in intuitive graphical formats. Following this direction we implemented three ideas in a single interactive tool.

The first proposed technique, the *Frequency Plot*, intends to tackle two problems derived from the increasing in the amount of data observed in most of the existing visualization techniques. The problems are the overlapping of graphical elements, and the excessive population of the visualization scene. The first one prevents the visualizations to present information implicit in the "lost" elements overlapped in the scene. The second one is responsible for determining unintelligible visualizations since, due to the over-population, no tendency or evidence can be perceived.

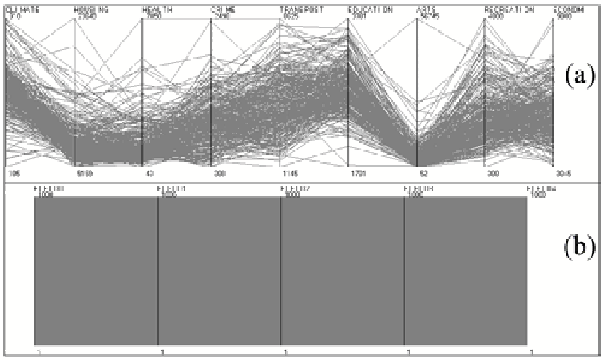


Fig. 1. - Examples of databases with too concentrated data (a), and with too spread data (b)

These cases are exemplified in figure 1, using the Parallel Coordinates visualization technique [1]. Figure 1(a) shows a common database where some ranges are so massively populated that only blots can be seen in the visualization scene. Therefore, the hidden elements cannot contribute for investigation. Figure 1(b) shows a hypothetical database where values in every dimension are uniformly distributed between the dimensions' range, so the corresponding visualization becomes a meaningless rectangle.

To deal with the issues pointed above, the *Frequency Plot* intends to increase the analytical power of visualization techniques, allowing to weight the data to be analyzed. The analysis, based on the frequency of data occurrence, can demonstrate the areas where the database is most populated, while its interactive characteristic allows the user to choose the most interesting subsets where this analysis shall take place.

The second technique proposed, the *Relevance Plot*, describes a way to analyze and to present the behavior of a data set based on a interactively defined set of properties. In this approach, the data elements are confronted with the set of properties defined by the user in order to determine their importance according to what is more relevant to the user. This procedure permits the verification, discovering and validation of hypothesis, providing a data insight capable of revealing the essential features of the data.

The third idea is the visual presentation of basic statistical analysis along with the visualization of a given technique. The statistical summarizations used are the average, standard deviation, median and mode.

The remainder of the paper is structured as follows. Section 2 gives an overview of the related work and Section 3 describes the data set used in the tests presented in this

work. Sections 4 and 5 present the *Frequency Plot* and *Relevance Plot* approaches respectively. Section 6 shows techniques to present results of statical analysis performed over visual scenes. The tool integrating the proposed techniques is described in Section 7, and the conclusions and the future works are discussed in the Section 8.

2. Background and Related Work

Nowadays, information visualization and visual data mining researchers face two facts: exhibition devices are physically limited, while data sets are inherently unlimited both in size and complexity. In this scenario, the Infovis researches must improve visualization techniques, besides finding new ones that should target large datasets. We propose improvements based on both interaction and automatic analysis, so their combination might assist the user in exploring bigger data sets, more efficiently.

According to [3], conventional multivariate visualization techniques do not scale well with respect to the number of objects in the data set, resulting in displays with unacceptable cluttering. In [4], it is asserted that the maximum number of elements that the Parallel Coordinates technique can present is around a thousand. In fact, most of the existing visualization techniques do not sustain interestingness when dealing with much more than this number, either due to space limitations inherent to current display devices, or due to data sets whose elements tend to be too spread over the data domain. Therefore, conventional visualization techniques often generate scenes with a reduced number of noticeable differences.

Many limitation from visualization methods can be considered inevitable because largely populated databases often exhibit many overlapping values, or a too spread distribution of data. These shortcomings lead many multi-variate visualization techniques to degenerate. However, some limitations have been dealt by the computer science community in many works, as explained following.

A very efficient method to bypass the limitations of overplotting visualization areas is using hierarchical clustering to generate the visualization expressing aggregation information. The work in [6] proposes a complete navigation system to allow the user to achieve the intended level of details in the areas of interest. That technique was initially developed for use with Parallel Coordinates, but implementations that comprises many other multivariate visualization schemes are available, as for example in the Xmdv Tool [3]. The drawback of this system is its complex navigation interface and the high processing power required by the constant re-clustering of data, according to user's redefinition.

Another approach, worth to mention, is described in [7], which uses wavelets to present data in lower resolutions without losing the aspects of its overall behavior. This technique takes advantage of the wavelets' intrinsic property of image details reduction. Although there is a predicted data loss that might degrade the analysis capabilities, the use of this tool can enhance the dynamic filtering activity.

However, the most known and used alternative to perceive peculiarities in massive data sets is finding ways to visually highlight subsets of data instead of the whole database. The "interactive filtering principle" claims that in exploring large data sets, it is important to interactively partition the dataset into segments and focus on interesting subsets [8]. Following this principle, many authors developed tools aiming the interactive filtering principle, as the Magic Lenses [9] and the Dynamic Queries [10]. The selective

focus on the visualization is fundamental in interaction mechanisms, since it enriches the user participation during the visualization process, allowing users to explore the more interesting partitions in order to find more relevant information.

An interesting approach in selective exploration is presented in the VisdB tool [11], which provides a complete interface to specify a query whose results will be the basis for the visualization scene. The presentation of the data items use the relevance of items to determine the color of the corresponding graphical elements. The color scheme determines the hue of the data items according to their similarity to the data items returned by the query. The multi-variate technique used is pixel oriented, and the interaction occurs based on a query form positioned alongside the visualization window. The analysis depends on the user ability to join information originating from one window per attribute, since each attribute is visualized in a separate scene.

Another approach is the direct manipulation interaction technique applied to Information Visualization [12], that is, techniques that improve the ability of the user to interact with a visualization scene in such a way that the reaction of the system to the user's activity occurs within an period short enough to the user to establish a correlation between their action and what happen in the scene [13]. Known as the "cause and effect time limit" [12], this time is approximately 0.1 second, the maximum accepted time before action and reaction seems disjointed. Our work was oriented by this principle, so that the tool developed satisfies this human-computer interaction requisite.

3. The Breast Cancer Data Set

In the rest of this work, we present some guiding examples to illustrate the presentation. The examples use a well-known data set of breast cancer exams [2], which was built by Dr. William H. Wolberg and Olvi Mangasarian from the University of Wisconsin, and is available at the University of California at Irvine Machine Learning Laboratory: (ftp://ftp.cs.wisc.edu /math-prog/cpo-dataset/machine-learn/WDBC). It comprises 457 records from patients whose identities were removed. Each data item is described by 9 fact attributes (dimensions), plus a numeric identifier (attribute 0), and a classifier (attribute 11) which indicates the tumor type (0 for benign and 1 for malign). The fact attributes (from the 2^{nd} to the 10^{th} attributes) are results of analytical tests on patients' tissue samples. They might indicate the malignity degree of the breast cancer and are respectively named *ClumpThickness, UniforSize, UniforShape, MargAdhes, SingleEpithSize, BareNuclei, BlandChromatin, NormalNucleoli and Mitoses.* These names are meaningfull in medical domain, but do not influence the visualizations interpretation. Noise data was removed from the original source.

4. The *Frequency Plot* with Interactive Filtering

Now we present a new enhancement for infovis techniques, intended not only to bypass the limits of visualization techniques previously pointed out in this text, but also to provide a way to enhance the analytical power of every visualization technique embodied in an interaction mechanism. It combines interactive filtering with direct manipulation in

an analytical-based presentation schema - that is, selective visualization enhanced by further filtering of the selected portions of the data set is followed by an automatic analysis step. We named this presentation technique as the *Frequency Plot*.

Here we describe the idea in general terms, reserving an example for later detailing. By frequency, we mean how common (frequent) is a given value inside the dataset. Formally, given a set of values V = $\{v_0, v_1, ..., v_{k-1}\}$, let q(v,V) N be a function which counts how many times v V appears in the set V. Also, let m(V) be a function that returns the statistical mode of the set V. The frequency coefficient of a value v V is given by:

$$f(v,V) = q(v,V) / q(m(V),V) \quad . \qquad (1)$$

The function $f(v,V)$ returns a real number between 0 and 1 that indicates how frequently the value v is found inside the set V. In our work, this function is applied to every value of every dimension in the range under analysis. Given a dataset C with n elements and k dimensions, its values might be interpreted as a set D with k subsets, one subset for each dimension. That is, $D = \{\{D^0\},\{D^1\},...,\{D^{k-1}\}\}$, having $|D^x| = n$. Given a k-dimensional data item $c_j = (c_j^0, c_j^1, ..., c_j^{k-1})$ belonging to set C, its corresponding k-dimensional frequency vector F_i is given by:

$$F_j = \left(f\left(c_j^0, D^0\right), f\left(c_j^1, D^1\right), ..., f\left(c_j^{k-1}, D^{k-1}\right)\right) \quad . \qquad (2)$$

Using Equation 2, we can calculate the frequency of any k-dimensional element. Once calculated the corresponding frequencies, each object is exhibited through visual effects as color and size modulated by its frequency. This is shown in our implementation in the following way. Using the Parallel Coordinates technique, the frequencies are expressed by color, and using the Scatter Plots technique, the frequencies are expressed by both color and size. This implementation uses a single color for each data item, a white background for the scene, and the high frequency values are exhibited with more saturated tones, in contrast with the low frequency ones, whose visualizations were based on less visible graphical elements determined by smooth saturations, that tend to disappear in the white background.

Combining the interactive filtering with this idea, our proposed visualization technique is not based on the whole data set, but on selected partitions specified by the user. These partitions are acquired by the manipulation stated by the interactive filtering principles, embodying AND, OR and XOR logical operators, in a way that enables the user to select data through logical combinations of ranges in the chosen dimensions. Therefore, only the data items that satisfy the visual queries are used to perform the frequency analysis. Hence, subsets of the database can be selected to better demonstrate particular properties in the dataset.

An illustrative example is given in figure 2. In this figure, the *Frequency Plot* of the complete dataset and the traditional range query approach are contrasted with a visualization comprising a range query with the *Frequency plot.* The data set under analysis is the breast cancer database described in section 3. The analysis process is illustrated intending to clarify what is the difference between malign and benign breast cancer based on laboratory tests.

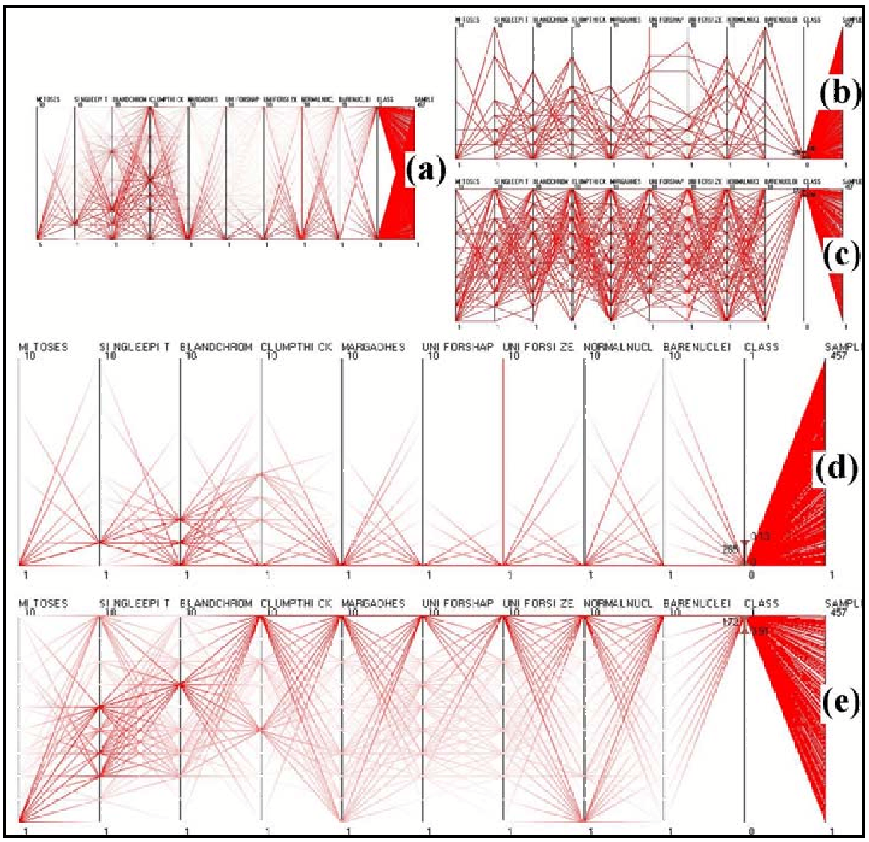


Fig. 2 - Parallel Coordinates with *Frequency Plot*. (a) The frequency analysis of the whole data set; (b) the traditional visualization of the malign class 1; (c) the benign class 0 visualization. (d) *Frequency Plot* of (b); (e) the correspondent *Frequency Plot* of (c)

In figure 2(a), the overall data distribution can be seen through the use of a global frequency analysis. The figure indicates the presence of more low values in most of the dimensions. Figures 2(b) and 2(c) presents the malign and benign records respectively, using the ordinary filtering technique, where no color differentiates the data items. It is clear that these three scenes contribute little to cancer characterization, as the visualization should do. None of them can partition nor analyze data simultaneously and, consequently, they are incapable of supporting a consistent parsing of the problem.

In contrast, figures 2(d) and 2(e) better demonstrate the characteristics relative to the laboratory tests and cancer nature. In these figures, the attribute 1, class, is used to determine frequency, considering value 0 (benign) in figures 2(d) and value 1 (malign) in figure 2(e). By highlighting the most populated areas of the selections, malign and benign cancer turns out to be identified more easily by searching for patterns alike those that are made explicit by the *Frequency Plot*. Therefore, the analyst is enabled to

conclude what results he/she needs to search for in order to make the characterization of the cancer nature more precise. The data distribution is visually noticeable and can be separately appreciated due to the interactive filtering.

Figure 3 shows the same visualizations as were presented in figure 2, but in a Scatter Plots Matrix enhanced by the *Frequency Plot* analysis. The Scatter Plots visualization corroborates what has been already seen in the Parallel Coordinates scenes and it further clarifies the fact that the "*BlandChromation*" and the "*SingleEpithSize*" attributes are the less discriminative in cancer nature classification.

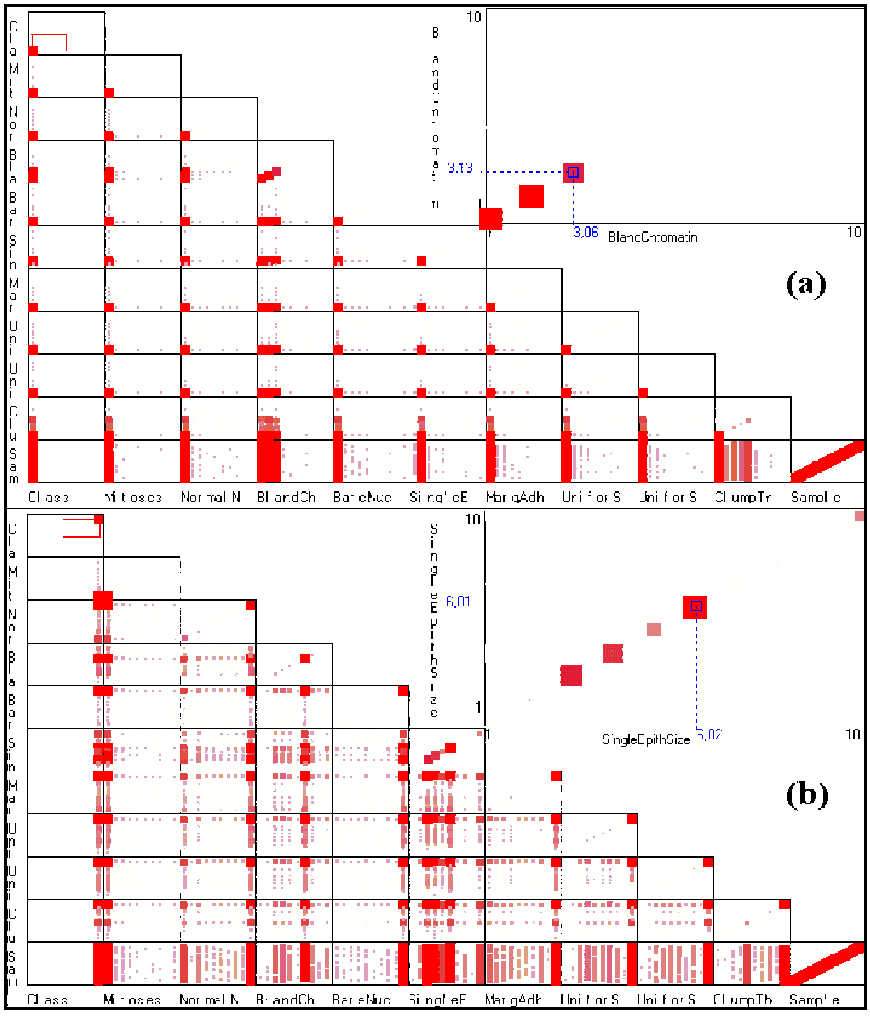


Fig. 3 - (a) The Scatter Plot Matrix with *Frequency Plot* showing the benign cancer autopsies. Note the zoom of the "BlandChromation" attribute. (b) Correspondente visualization of the malign cancer autopsies and the zoom of the "SingleEpithSize" attribute

The data presentation with frequency plot is a powerful tool because it can determine more easely the subset of interest for the analysis. Also, it points that cluster identification is not limited to the whole data set, nor to predefined manipulated data records. Instead, it might occur guided by a single value. One might choose a dense populated area belonging to one of the dimensions and questions the reason of that behavior. The frequency plot technique will present the most frequent values of the other dimensions that are correlated to the region of interest; correlation goes straight along with partial cluster identification.

5. The *Relevance Plot*

The second technique described in this work is based on the concept of data relevance, showing the information according to the user's needs or sensibility. Therefore, we intend to reduce the amount of information presented by drawing data items using visual graphic patterns in accordance to their relevance to the analysis. That is, if the data has a strong impact on the information under analysis, their visualization shall stress this fact, and the opposite must happen to data that is not relevant to the user. To this intent, the *Relevance Plot* benefits from computer graphic techniques applied to depict automatic data analysis through color, size, position and selective brightness.

The proposed interaction does not depend on a query stated on a Structured Query Language (SQL) command, thus it is not based on a set of range values, but rather, it is based on a set of values considered interesting by the user. These values are used to determine how relevant is each data item. Once the relevant characteristics are set, automatic analysis proceeds through calculation of data relevance relative to what was chosen by the user to be more interesting.

The mechanism is exemplified in Figure 4. It requires the analyst to choose values, or *Relevance Points*, from the dimensions being visualized. Hence, given a set of data items *C* with *n* elements and *k* dimensions, and assuming that these values were previously normalized so that each dimension ranges from 0.0 to 1.0, the following definitions hold:

> *Definition 1*: the *Relevance Point* (*RP*) of the *i*-th dimension, or RP^i, is the chosen *value belonging to the i-th dimension domain* that must be considered to determine the data relevance in that dimension.

For illustration purposes, let us first consider only one *RP* per dimension. Once the *Relevance Points* are set for every dimension, the data items belonging to the database must be analyzed considering their similarity to these points of relevance. That is, for each dimension with a chosen *RP*, all the values of every tuple is computed considering their Euclidean distance to the respective relevance value.

> *Definition 2:* for the *j*-th k-dimensional data record $c_j = (c_j^0, c_j^1, ..., c_j^{k-1})$, the distance of its *i*-th attribute to the *i*-th *RP*, or $D_j^i(c_j^i, RP^i)$, is given by:
>
> $$D_j^i(c_j^i, RP^i) = |c_j^i - RP^i| \quad (3)$$

We use the Euclidean distance due to its simplicity, but other distances, such as any member of the Minkowski family, can also be employed. For each dimension of the *k*-dimensional database, a maximum acceptance distance can be defined. These thresholds are called *Max Relevance Distance*s, or *MRDs*, and are used in the relevance analysis.

Definition 3: the *Max Relevance Distance* of the *i*-th dimension, or MRD^i, is the maximum distance $D_j^i(c_j^i, RP^i)$ that a data attribute can assume, before its relevance assume negative values during relevance analysis. The *MRDs* take values within the range [0.0, 1.0].

Based on the *MRDs* and on the calculated distances $D_j^i(c_j^i, RP^i)$, a value named *Attribute Relevance* (*AR*) is computed for each attribute of the *k*-dimensional data items. Thus, a total of *k ARs* are computed for each of the *n k*-dimensional data items in the database.

Definition 4: the value determining the contribution of the *i*-th attribute of the *j*-th data item, c_j^i in the relevance analysis is called the *Attribute Relevance,* and is given by:

$$AR_j^i = \begin{cases} 1 - \dfrac{(D_j^i(c_j^i, RP^i))}{MRD^i} & \text{if} \quad D_j^i(c_j^i, RP^i) \le MRD^i \\ - \dfrac{(D_j^i(c_j^i, RP^i))}{(1 - MRD^i)} & \text{if} \quad D_j^i(c_j^i, RP^i) > MRD^i \\ 0 & \text{if} \quad RP^i = -1 \end{cases} \qquad (4)$$

Equation 4 states that:

- For distances *D(c,RP)* smaller or equal the *MRD*, the equation has been settled to assign values ranging from 1 (where the distances *D*(*c*,*RP*) are null) to 0 (for distances equal the *MRD*);
- For distances *D*(*c*,*RP*) bigger than the *MRD*, the equation linearly assigns values ranging from 0 to -1;
- In dimensions without a chosen *RP,* the *AR* assumes a value 0 and does not affect analysis.

Finally, after processing every value of the dataset, each of the *k*-dimensional item will have a value computed, that is called *Data Relevance* (*DR*) and stands for the relevance of a complete information element (an tuple with *k* attributes).

Definition 5: the *Data Relevance* (*DR*) is the computed value that describes how relevant a determined data item is, based on the Attribute Relevancies. So, for a given data item, the *DR* is the average of its correspondent Attribute Relevancies. For the *j*-th *k*-dimensional element of a data set, the DR_j is given by:

$$DR_j = \frac{\left(\sum_{i=0}^{k-1} AR_j^i\right)}{\#R} . \qquad (5)$$

where #*R* is the number of *Relevance Points* that were defined for the analysis.

The *Data Relevance* value directly denotes the importance of its corresponding data element, according to the user defined *Relevance Points*. To visually explicit this fact, we use the *DRs* to determine the color and size of each graphic elements. Hence, a lower *DR* stands for weaker saturations and smaller sizes, while higher ones stand for more stressed

saturations and bigger sizes. In our implementation we benefit from the fact that only the saturation component is necessary to denote relevance, leaving the other visual attributes (colors, hue and size) available to depict other associated information. In this sense, we projected a way to denote, along with the relevance analysis, the aforementioned frequency analysis.

That is, while the saturation of color and the size of the graphical elements denote relevance, the hue component of color presents the frequency analysis of the data set. More precisely, the highest frequencies are presented in red (hot) tones, and the lowest frequencies in blue (cold) tones, varying linearly passing through magenta.

Figure 5 presents the usage of the *Relevance Plot* technique over the Parallel Coordinates technique using the breast cancer dataset. In the three scenes we have defined 9 *Relevance Points* for dimensions 2 to 10. For illustration purposes in Figure 5(a), the points are set to the smallest values of each dimension, in Figure 5(b) they are set to the maximum values of each dimension, and in Figure 5(c) they are to the middle points. However, each dimension can have its RP defined at the user's discretion.

In figure 5(a) the choice of the *Relevance Points* and their correspondent visualization leads to conclude that the lowest values of the dimensions' domains indicate class 0 (benign cancer) records. It also warns that this is not a final conclusion since the visualization reveals some records, in lower concentration, which are classified as 1 (malign cancer). It can be said that false negative cancer analysis can occur considering a clinical approach based on this condition.

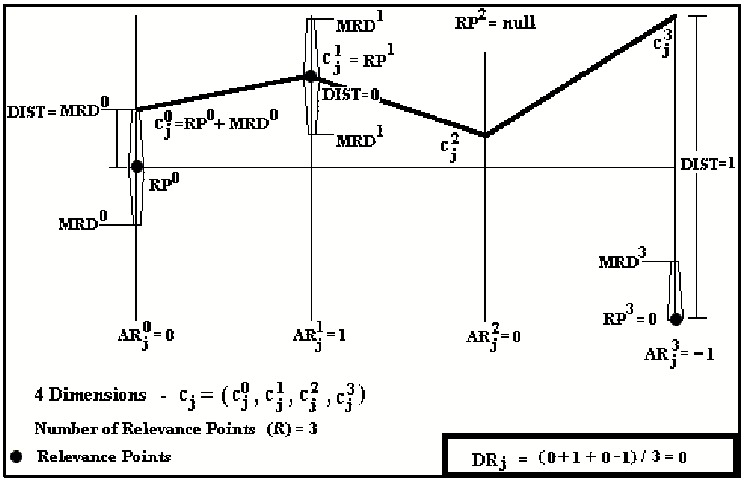


Fig. 4 - The *Relevance Plot* schema is demonstrated here through the calculus of the relevance for a 4-dimensional sample record visualized in the Parallel Coordinates technique

In figure 5(b) the opposite can be observed, as the highest values indicate the records of class 1. It can be seen that false positive cancer analysis can also occur, but they are less common that the false negative cases, since just a thin shadow of pixels heads to class 0 in the 11th (right most) dimension.

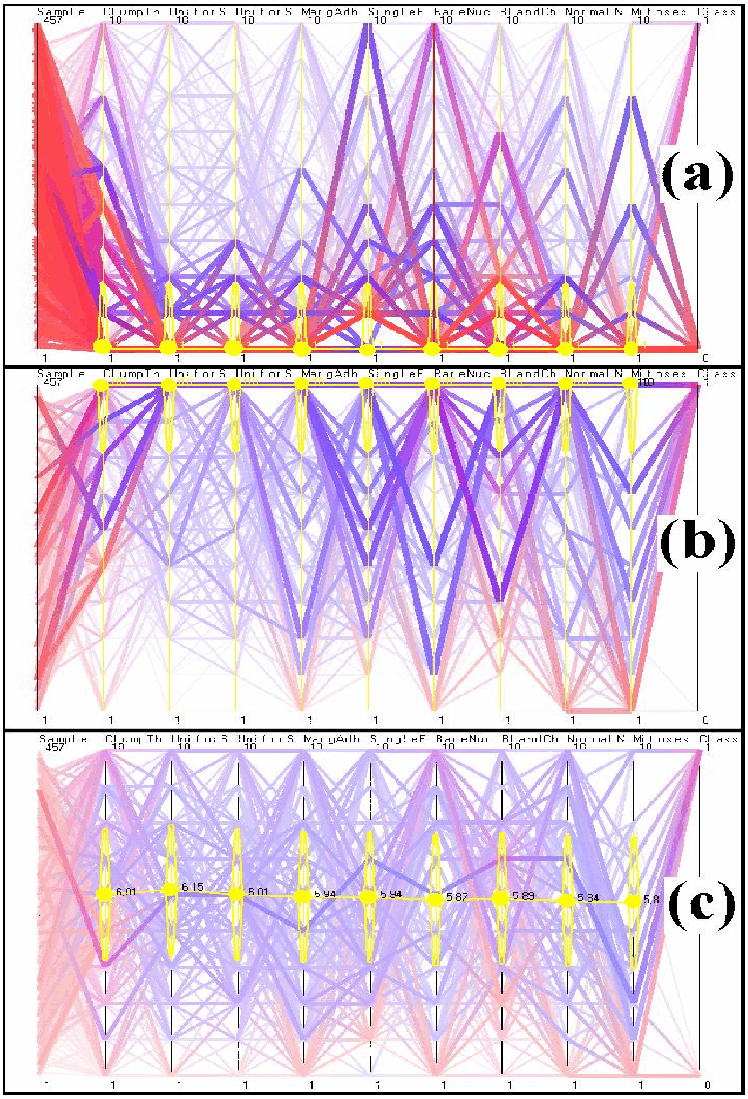


Fig 5 - The *Relevance Plot* over a Parallel Coordinates scene. In (a) all the relevance points are set to the smallest values of their dimensions. In (b) they are set to the maximum values and in (c) middle points are set

Finally, in figure 5(c) the Relevance Points were set to middle points in order to make an intermediate analysis. In this visualization, one can conclude that this kind of laboratory analysis is quite categorical, since just one record is positioned in the middle of the space determined by the dimensions' domains. But, in such cases, it is wise to classify the analysis as a malign cancer or, otherwise, to proceed with more exams.

6. Basic Statistics Presentation

The statistical analysis has been successfully applied in practically every research field and its use in Infovis naturally improves data summarization. So, defending our idea that information visualization techniques must be improved by automatic analysis, this paper describe how our visualization tool makes use of basic statistics to complement the revealing power of traditional visualization mechanisms.

The statistical properties we used take advantage of are the average, standard deviation, median and mode values, and the method for visualizing them is straightforward. The raw visualization scene is rendered, and the statistical summarizations are used to draw an extra graphical element over the image. The extra graphic elements are the summarizations which are exhibited with a different color for visual emphasis.

The four statistical summarizations are available in each of the four visualization techniques implemented in the GBDIView tool, which will be presented in the next section. As an example, figure 6 presents the breast cancer dataset drawn using the Star Coordinates technique together with a polygon over the scene indicating the median values of the malign cancer exams (figure 6.a) and of the benign cancer exams (figure

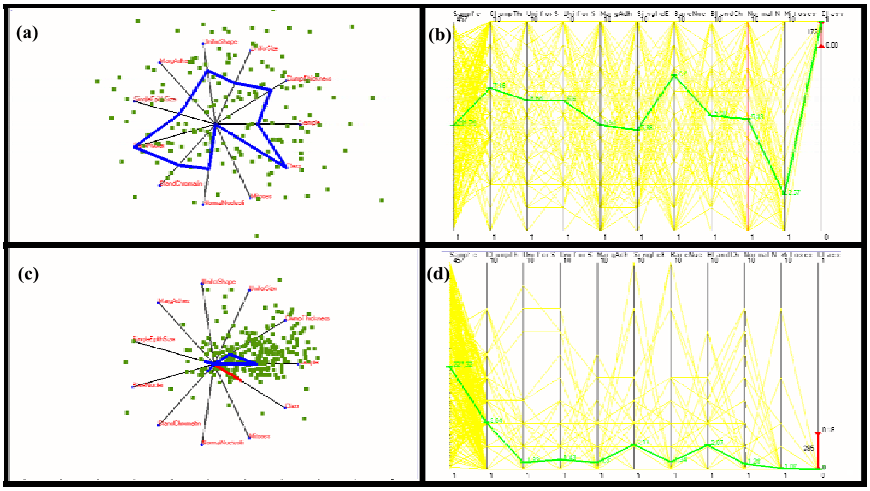


Fig. 6 - In (a) the median of the malign cancer exams presented over the Star Coordinates; in (b) the median of the benign exams. In (c) the average line of the malign cancer exams over the Parallel Coordinates and in (d) the average of the benign exams

6.b). Also in figure 6, we can see the same data set drawn using the Parallel Coordinates technique together with a polyline indicating the average values of the malign cancer exams (figure 6.c) and of the benign cancer exams (figure 6.d). The possibilities of these visualization schemes are very promising, being more powerful than their respective raw visualizations using the Star Coordinates and Parallel Coordinates techniques. In a short analysis, the presented statistical visualization stress the conclusions addressed by the examples shown in sections 4 and 5.

7. The GBDIView Tool

In order to experiment our ideas, we have implemented a tool whose snapshot is presented in figure 7, that fully encompass the theory presented in this paper. The GBDIView tool consists of 4 well-known visualization techniques enhanced by the proposed approaches we have developed. The tool is built in C++, and was designed following the software reuse paradigm, therefore, being idealized as a set of visualization methods implemented as software components that can be totally tailored to any software that uses Infovis concepts.

The techniques included in the tool are the Parallel Coordinates, the Scatter Plots Matrix [5], the Star Coordinates [14] and the Table Lens [15]. The four visual schemes are integrated by the Link & Brush [16] technique and every one is also enabled with the

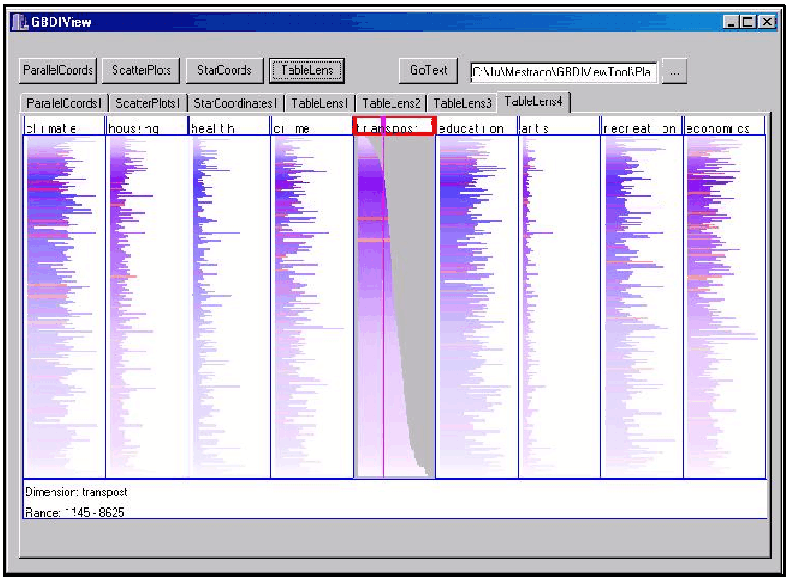


Fig. 7 - The GBDIView tool presenting a Table Lens visualization with *Relevance Plot*

Frequency Plot, with the *Relevance Plot* and with the statistical data analysis (average, standard deviation, median and mode values), what can be presented graphically over the rendered scenes.

A fully functional version of the tool, along with its user's guide and some sample datasets can be obtained at *http://gbdi.icmc.usp.br/~junio/GBDIViewTool.htm*.

8. Conclusions and Future Work

We believe that both the *Frequency Plot* and the *Relevance Plot* techniques can strongly contribute to improve the effectiveness of databases exploration, specially the relevance visualization, which is a way to focus on interesting parts of a data set without losing the overall sight. We also argue that this contribution is applicable to other multi-variate visualization techniques beyond the Parallel Coordinates, the Scatter Plots, the Star Coordinates and the Table Lens, some of which are already implemented in the GBDIView tool.

It is important to observe that the contributions of this work are not limited to the techniques themselves, but also to the innovative orientation in the development of visualization mechanisms. Here, we do not strive to discover better visualization schemes, instead, we focus on enhancing existent ones by promoting the automatic analysis of the data according to the user previous interaction.

As future works, the relevance visualization demands a certain processing power to be used interactively, and the described model does not embody optimization nor scalability. Thus, a better model should be pursued. Also, the relevance mechanism can be carried out in many different ways, as using other distance functions, defining more than one relevance point per dimension and/or setting weights to the dimensions to be analyzed. A user interface to hold all these possibilities must also be conceived, maintaining a user friendly and easy of use environment.

We considered that the frequency analysis is evaluated counting the values in the dataset. This assumption is valid for many datasets, but not always. Consequently, as a future work, improved ways should be studied to reach good results with continuous attributes or attributes that does not have categorical values. A natural option is to use the values clustered in a given way, or process the data set to define probabilistic functions, which is a more adequate alternative to deal with attributes consisting of high precision data types.

Acknowledgment

This work has been supported in part, by the Sao Paulo State Research Foundation (FAPESP) under grants 01/11287-1 and 02/07318-1, and by the Brazilian National Research Council (CNPq) under grants 52.1267/96-0, 52.1685/98-6 and 860.068/00-7.

References

1. Inselberg, A. and B. Dimsdale. Parallel Coordinates: A Tool for Visualizing Multidimensional Geometry. in IEEE Visualization. 1990: IEEE Computer Press. p. 361-370

2. Bennett, k.P. and O.L. Mangasarian, Robust linear programming discrimination of two linearly inseparable sets, in Optimization Methods and Software. 1992, Gordon & Breach Science Publishers. p. 23-34.

3. Rundensteiner, A., et al. Xmdv Tool: Visual Interactive Data Exploration and Trend Discovery of High Dimensional Data Sets. in Proceedings of the 2002 ACM SIGMOD international conference on Management of data. 2002. Madison, Wisconsin, USA: ACM Press. p. 631

4. Keim, D.A. and H.-P. Kriegel, Visualization Techniques for Mining Large Databases: A Comparison. IEEE Transactions in Knowledge and Data Engineering, 1996. 8(6): p. 923-938.

5. Ward, M.O. XmdvTool: Integrating Multiple Methods for Visualizing Multivariate Data. in Proceedings of IEEE Conference on Visualization. 1994. p. 326-333

6. Fua, Y.-H., M.O. Ward, and A. Rundensteiner, Hierarchical Parallel Coordinates for Exploration of Large Datasets. Proc. IEEE Visualization'99, 1999.

7. Wong, P.C. and R.D. Bergeron. Multiresolution multidimensional wavelet brushing. in Proceedings of IEEE Wsualization. 1995. Los Alamitos, CA: IEEE Computer Society Press. p. 184-191

8. Keim, D.A., Information Visualization and Visual Data Mining. IEEE Transactions on Visualization and Computer Graphics, 2002. 8(1): p. 1-8.

9. Bier, E.A., et al. Toolglass and Magic Lenses: The See-Through Interface. in SIGGRAPH '93. 1993.

10. Ahlberg, C. and B. Shneiderman. Visual Information Seeking: Tight coupling of Dynamic Query Filters with Starfield Displays. in Proc. Human Factors in Computing Systems CHI '94. 1994. p. 313-317

11. Keim, D.A. and H.-P. Kriegel, VisDB: Database Exploration Using Multidimensional Visualization. IEEE Computer Graphics and Applications, 1994. 14(5): p. 16-19.

12. Card, S.K., J.D. Mackinlay, and B. Shneiderman, Using Vision to Think. Readings in Information Visualization. 1999, San Francisco, CA: Morgan Kaufmann Publishers.

13. Siirtola, H. Direct Manipulation of Parallel Coordinates. in International Conference on Information Visualization. 2000.

14. Kandogan, E. Star Coordinates: A Multi-dimensional Visualization Technique with Uniform Treatment of Dimensions. in IEEE Symposium on Information Visualization 2000. 2000. Salt Lake City, Utah. p. 4-8

15. Rao, R. and S.K. Card. The Table Lens: Merging Graphical and Symbolic Representation in an Interactive Focus+Context Visualization for Tabular Information. in Proc. Human Factors in Computing Systems. 1994. p. 318-322

16. Wegman, E.J. and Q. Luo, High Dimensional Clustering Using Parallel Coordinates and the Grand Tour. Computing Science and Statistics, 1997. 28: p. 352–360.

DataJewel: Tightly Integrating Visualization with Temporal Data Mining

Mihael Ankerst, David H. Jones, Anne Kao, Changzhou Wang

Boeing Phantom Works,
P.O. Box 3707 MC 7L-70, Seattle, WA 98124-2207
[mihael.ankerst | david.h.jones | anne.kao
| changzhou.wang]@boeing.com

Abstract. In this paper we describe DataJewel, a new architecture designed for temporal data mining. It tightly integrates a visualization component, an algorithmic component and a database component. We introduce a new visualization technique called CalendarView as an implementation of the visualization component. We show how algorithms can be tightly integrated with the visualization component and that most existing temporal data mining algorithms can be leveraged by embedding them into our architecture. This integration is achieved by an interface that is used by the user and the algorithm to assign colors to events. The user assigns colors to interactively incorporate domain knowledge or to formulate hypotheses. The algorithm assigns colors based on the discovered patterns. Using the same visualization technique for both data and patterns makes it more intuitive for the user to select useful patterns from those returned by the algorithm. We also present a data structure that supports temporal mining of very large databases. In the experiments, we apply our approach to several large datasets from the airplane maintenance domain and discuss its applicability to domains like homeland security, market basket analysis and web mining.

1 Introduction

In recent years, there has been a lot of interest in the KDD community in mining temporal data. Temporal datasets have a dedicated attribute storing a time stamp for each record. This time stamp usually refers to the date an event has happened or some kind of data has been measured and collected. Examples for temporal datasets include stock market data, manufacturing or production data, maintenance data, web mining and point-of-sale records. Due to the importance and complexity of the time attribute, a lot of different kind of patterns are of interest. An overview is provided in [2]. Typically, in different domains different kind of temporal patterns are of interest. This is one aspect motivating our architecture, which provides access to many temporal data mining algorithms and an easy way to add new ones.

When dealing with temporal databases a second but very substantial aspect becomes an important challenge. In large enterprises, databases evolve as a consequence of an

organizational need. They are designed to serve a specific (e.g. operational) purpose. Often databases from different organizations can be linked together to serve a new purpose, e.g. to provide a platform for data mining. However, the task of linking databases together is far from trivial; the field of information integration deals with challenging and laborious problems of maintaining data integrity, schema mapping, and resolving duplication. Often, there is no common attribute at all except the timestamp. By linking tables together to explore a subset of the union of the attributes with respect to time, a powerful view upon the data is obtained. E.g. intelligence agencies can link tables together that correspond to news, credit card histories, travel itineraries to detect suspicious activities. An enterprise can link together helpdesk data about computer problems with a completely independent table from the procurement department and a labor database. The detected patterns might reveal insights into causes of computer problems and might form a new purchasing strategy.

In this paper, we address both aspects of temporal data mining. On the one hand, our approach is applicable to a variety of domains because it leverages existing algorithms. On the other hand, it offers a means of linking tables together that have no primary key – foreign key relationship. All they are required to have is an attribute with a timestamp.

In addition, our new architecture for temporal data mining also makes the following contribution. Traditionally, algorithmic approaches are introduced by one research community and some of them also focus on scalability aspects. However, for most of the papers, the visualization component is omitted, either because the authors do not feel comfortable in this area or because they think a graphical user interface alone should be sufficient and that is not a research task. On the other hand, the visualization community focuses most often just on the visualization aspect, i.e. how to represent data, but does not investigate algorithmic approaches [5]. With this paper, we would like to make a contribution towards a closer collaboration of these fields. We show that a system that is designed to tightly integrate components from various disciplines can substantially improve the functionality of loosely coupled components.

The rest of the paper is organized as follows. In Section 2, we summarize related work. Section 3 describes a user-centric data mining process and the DataJewel architecture. Section 4 presents the visual component of our architecture and describes in detail our new visualization technique called CalendarView. Section 5 outlines how temporal data mining algorithms can be tightly integrated into DataJewel. Section 6 reports how to handle large datasets. In section 7, we describe several experiments with large datasets. We conclude the paper with section 8 and discuss some future directions.

2 Related Work

Our main contribution is to tightly integrate a visual, an algorithmic and a database component for temporal data mining. To our knowledge no such architecture has been proposed so far. Most of the work in temporal data mining deals with either just an algorithmic approach, a way to visualize data over time or an approach to scale up to large datasets. We review these areas in the following paragraphs.

Many approaches for visualizing data over time have been proposed. Typically, visualization techniques represent temporal data either as a sequence along an axis, or as animations where data at different times is represented in different frames. A recent approach which treats data as a sequence is ThemeRiver [6]; it employs the metaphor of a current and maps histograms of document keywords to the height of a wave at a particular time. Mackinlay et. al. [11] uses a spiral for calendar visualization, however, calendar days are just used as reference points. Hierarchical pixel bar charts [8] are not aimed at visualizing temporal data but it can be used as an alternative pixel representation within a day.

Several algorithms for mining temporal datasets have been proposed. According to a recent overview [2], contributions have been made in the areas of how to model a temporal sequence, how to define a suitable similarity measure for sequences and what kind of mining operations can be performed. We will show in section 5 that many of the existing algorithms can be leveraged by our architecture.

Tightly integrated architectures have been proposed, but are only partially comparable to our approach. In [1], the authors describe an approach called cooperative classification, where the visualization and the algorithmic component are tightly integrated. This approach however, was specifically designed for decision tree classification and does not elaborate on scalability issues. Similarly, HD-eye [8] and n23Tool [15] integrate visualization with algorithms but are applicable just to clustering methods. [14] represents clusters of time series data which contain a pattern spanning one day and relating them to days with similar patterns. In contrast to our approach, it does not represent the data for each day nor does it cover scalability issues. Tightly integrating algorithms with databases or incorporating scalability considerations into data mining algorithms has been recognized and studied more extensively. A comprehensive survey is presented in [10]. Proposed ways to achieve scalability are falling in one of the three categories: design of a fast algorithm (e.g. by restricting the model space or parallelization), partitioning of the data (instance/feature selection methods) and relational representations (e.g. integration of data mining functionality in database systems). Recent approaches include the computation of sufficient statistics, like Rainforest [4] does for decision trees. [12] describes an in-depth analysis of different level of integration of an association mining algorithm into database systems. CONTROL [7] aims at a database-centric interactive analysis of large datasets focusing on online query processing. All these approaches, however, are not directly applicable to temporal data.

3 User-centric Data Mining

One design goal of our user-centric architecture is its intuitive use by a domain expert as opposed to data mining experts. As a result, the user can steer the exploration of temporal data, invoke algorithms to automatically discover patterns, incorporate his domain knowledge, hypothesize on the fly and use his perception to detect patterns of interest. In figure 1, we outline the mining process with DataJewel.

First, the user selects the tables and attributes for analysis. Then the data is loaded and

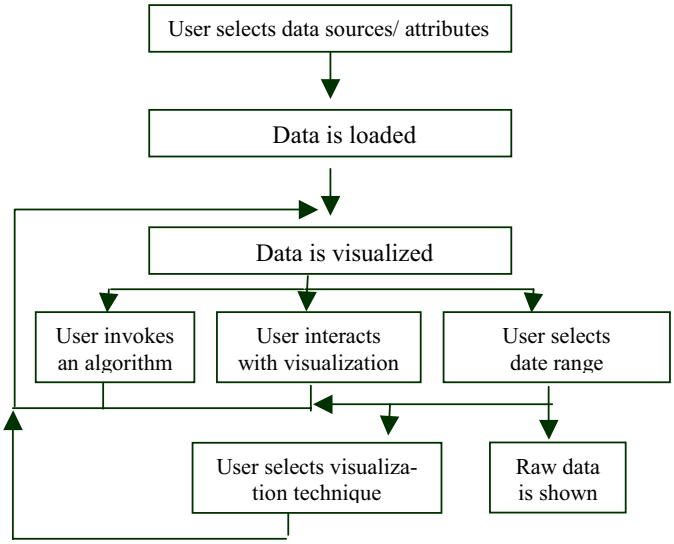


Figure 1. The mining process with DataJewel

visualized. The user has the option of invoking an algorithm and visualizing the resulting patterns using the current settings. Alternatively, the user can interact with the visualization to incorporate his domain knowledge or discover some patterns based on his perception. In either case, the user hopefully discovers some pattern of interest. Then he selects a date range of interest and visualizes it with the same or another visualization technique. Another visualization technique might be picked to represent the data in a different way or because it is more suitable due to the reduced size after the selection. After the user has iterated this loop several times, he might be interested in "drilling down" to the raw data to see all attributes. The corresponding tables are accessed, the data is retrieved and presented. Note that this approach facilitates extensions by incorporating new algorithms and visualizations.

In the following, we introduce some terminology and state assumptions for our architecture. Let us assume, the data sources consist of a set of tables. Each table contains r records, with each record consisting of d attributes $a_1,...,a_d$. At least one attribute contains a timestamp for each record. We refer to the timestamp attribute as the

event date, all categorical[1] attributes that should be incorporated in the analysis are *event attributes* and the attribute values of these event attributes are *events*. In this paper, we will focus on event attributes (categorical attributes only) for which the following assumptions hold:

a) The number of event attributes is low. (< 10)

b) The number of different events of one event attribute is moderate. (< 200)

c) The smallest time unit of interest in the event dates is one day

Assumption a) restricts the number of event attributes used during the analysis. As opposed to high-dimensional feature vectors for which some mining tasks are performed, event attributes usually have a clear meaning. Often, in one given analysis, the analyst selects a small number of event attributes, which can be associated with each other in the particular domain. Using domain knowledge, the remaining attributes are omitted from the analysis because they would just add noise.

Assumption b) limits the number of events of an event attribute to a moderate size. In case, where this is not true for the initial dataset, a concept hierarchy can be defined for the event attribute to reduce the total number of events.

With assumption c) we focus on the most common time unit of interest in business domains. Note that days are just the smallest unit of interest and the discovery of weekly or monthly patterns is also supported. Obviously, for intrusion detection systems, our proposed unit of time would have to be refined to reflect finer grained time units.

In figure 2, a simplified view of the DataJewel architecture is depicted consisting of three layer. Although the data flows from the data source to the visualization layer, we have designed our system from the opposite direction to better support a user-centric process. Corresponding to each one of these layers, we will describe the visualization, the algorithmic and the database component. We will present just one instance of the visualization and the algorithmic component, but new ones can be easily integrated.

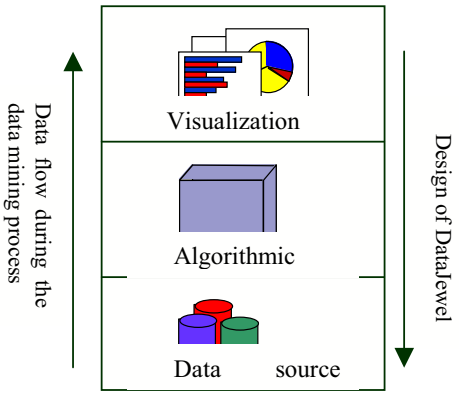


Figure 2. Data flow versus design of DataJewel

[1] Continuous attributes can be transformed into categorical attributes by discretization.

4 The Visualization Component

The visualization component contains visualization techniques suitable for representing temporal data. We present a new visualization technique, which represents temporal data on a daily basis.

4.1 CalendarView

Our architecture is primarily designed for domain experts not just for data mining experts. Thus the visualization component has to be intuitive as well as versatile. CalendarView, our new visualization technique, is motivated by what the human is already very familiar with. First, the representation of event dates is designed following the visual metaphor of a calendar. Second, the structure of the data that is represented along the event dates is the frequency of events. Its representation is based on the familiarity of humans with histograms.

In simpler linear representations, time is greatly simplified by modeling it as a sequence of dates. In contrast, we have selected the calendar metaphor because it reflects the rich temporal structure more effectively than typical simplified representations. From a calendar, the human preattentively extracts the notion of weekends, weekly repetitions, seasons, days with a special meaning in his domain, etc.

Whereas the calendar metaphor is used to represent the event dates on a daily basis, an extended version of histograms reflects the distribution of events for a single day. To enable the user to compare different event attributes with each other, each event attribute is represented by a separate calendar. In the final visualization all calendars are drawn one above the other.

Table 1. Example of a temporal dataset

Event Date	Event Attribute: Page hit	Event Attribute: Browser	Event Attribute: …
1/1/2002	Index.html	MS IE	…
1/1/2002	Dep1/contacts.htm	Netscape	…
…	…	…	…

Table 1 depicts an example of a temporal dataset. Each event has an associated event date, so we can count the frequency of this event occurring on a single day. If we do that for each event of one event attribute we can display the distribution by a histogram for this event attribute. We can initially assign a different color to each event of one event attribute. The default color map is the PBC color map [1] which has been developed to map distinct values to distinct colors. As illustrated[2] in figure 3, for each day the frequency distribution of the events is represented within the corresponding day in the calendar.

[2] Note that the color mapping in this paper is not the original color assignment. It has been changed to optimize for grayscale printing.

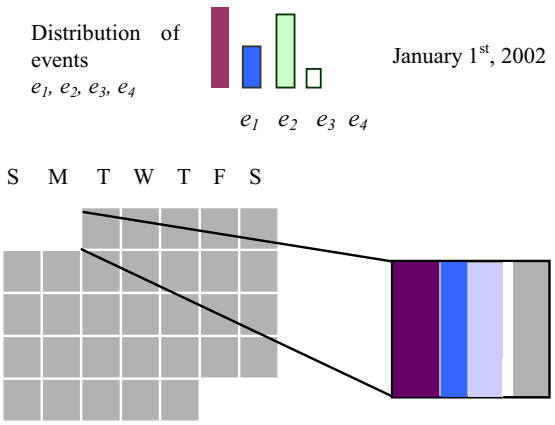


Figure 3. Illustration of CalendarView

Instead of using colored histograms where frequency is depicted by the height of the bins, the events are represented pixel by pixel to account for more categories than are usually depicted by a histogram. In particular each day is filled with pixels in the following way:

Each day is represented by a constant size square of $n \times n$ pixels. If the number of events of this event attribute on the corresponding day is less or equal to n^2, we can use one pixel per event. The pixel arrangement starts in the lower left corner of the day square. It goes up (*n-1*) times, goes one pixel to the right, then goes (*n-1*) pixel down, one pixel to the right, and so forth. Following the illustration in figure 3, let us assume we have four events e_1, e_2, e_3 and e_4. The frequency of the occurrence of e_1 on a particular day is denoted by $f(e_1, date)$. Then we draw the first $f(e_1,date)$ pixels with the color assigned to e_1. Following the described pixel arrangement, the next $f(e_2, date)$ pixels are drawn in the color of event e_2, and so forth. We can distinguish between the following three cases:

1) If $n^2 = \sum_e f(e,date)$ then we fill up the complete day square and each pixel represents one event by its color.

2) If $n^2 > \sum_e f(e,date)$ then each pixel represents one event by its color but all pixels do not fill up the entire space in the day square. The remaining pixels are drawn with a separate (background) color.

3) If $n^2 < \sum_e f(e,date)$ we fill up the complete day square by the algorithm above after substituting f(e, date) with $\left\lceil \frac{f(e,date)}{\sum_e f(e,date)} \cdot n^2 \right\rceil$. (formula 1)

The order of the events can greatly contribute to perception of their distribution. By reordering the events for each day preattentive processing can be improved. The rea-

son becomes clear with the following example. Let us assume the daily distribution of ten events over one year is very similar and even within each day the number of events does not differ largely. Let us further assume there is exactly one day where the least frequent event suddenly happens more often. At that day it is the second most frequent event. Then in addition to representing this event with more pixels than at other days, the reordering yields a better perception of this distribution change. Thus, the reordering improves the perception of distribution changes (cf. figure 4 where the 4^{th} day is reordered). Note that the daily reordering is done in real time, since computation is negligible, due to our assumptions in section 3.
Our default setting which we use throughout the paper is $n = 10$. The size of the day square n^2 is a tradeoff between representing each event by one pixel and the size of the (virtual) screen.

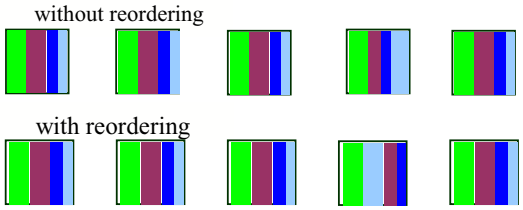


Figure 4. Ordering events daily by frequency

4.2 Interaction with CalendarView

In the following, we will describe the main interaction capabilities of CalendarView:

- Selection

As described in section 3, one essential feature of the visualization component is to select a subset of dates. The user is enabled to interactively select a set of consecutive days. The subset corresponding to the selected event dates can again be visualized following the iterative process outlined in section 3.

- Ascending/descending order

The decision if the events should be ordered ascending or descending by frequency is just important for the case where we have less pixels in a day square than events. If the frequency distribution on a particular day is very skewed, some events might not be represented at all because the drawing algorithm with formula 1 might have already filled up the complete day square. We think, in most cases the user is either interested in outlier events which happen very rarely as opposed to others or he is interested in the overall distribution of the "main" events. Therefore we enable the user to switch between ascending or descending order in real time. In case the ascending order has been selected, the drawing of the pixels starts with the rarest events and thus uncovers them possible at the expense of cutting off the largest event at the end. If the descending order is selected, the most frequent events are drawn first.

- Interactive color assignment

Initially, colors are assigned to events based on the PBC color map. Thus missing values can be elegantly treated as a distinct event and are assigned to a certain color (background by default). A dialog window enables the user to interactively assign colors to events. With manual color assignment the user can specify his domain knowledge, can formulate and test an hypothesis on-the-fly or steer the exploration in a meaningful way. The notion of color assignment is implemented as follows: If the user changes several events to have the same color, he indicates a conceptual generalization of the events. As a result, all events which are assigned to the same color are referred to as the same event when the visualization is redrawn. Thus, for each day, the events with the same color are grouped together before the drawing algorithm is invoked. For example, following the web mining dataset from table 1, let us assume we are recording web page hits on a particular day. These events can be generalized by the user by assigning color c_1 to all pages of the main website which have been visited. Color c_2 refers to the dep1/ subdirectory, c_3 to the dep2/ subdirectory and color c_4 is used for all other web pages. The interface is depicted in figure 5, also enabling the user to sort by event name or frequency. Each event attribute has a separate color assignment.

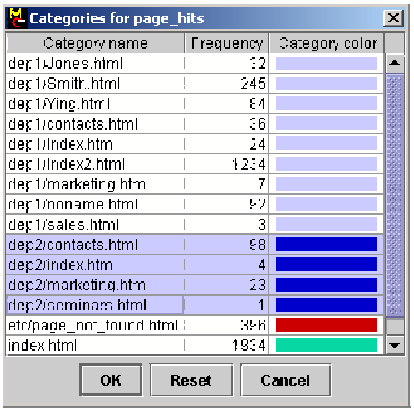


Figure 5.
Interactive color assignment

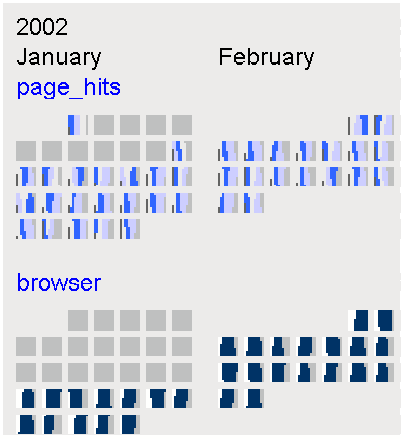


Figure 6.
Calendarview with web mining dataset

In figure 6, two event attributes from our example dataset are visualized from January 1st, 2002 to February 18th, 2002. For the "page hits" event attribute, the user has assigned colors to four different groups of web pages as described above. We see page hits on January 1st but no more until Saturday, January 12th. Maybe the web server was down for 11 days? Also the event attribute "browser" has its first event on Sunday, January 20th. Maybe the web server did not recognize the browser type until that day? Just two different browsers have been recognized. One browser has being used more often throughout the whole time period.

-Zooming

The user can zoom in or zoom out.

- Detail on Demand
The event corresponding to the pixel of the current mouse pointer position is displayed.

5 The Temporal Mining Component

Building the visualization component, we have introduced a visualization technique called CalendarView, which maps different events to distinct colors. In the case there are just a few events the visualization itself is very powerful since human's preattentive perception is very efficient in looking for variety of patterns. If the number of different events is larger, the usefulness of the default color assignment decreases because colors are not perceived as being distinct any more.

Nevertheless the visualization technique might reveal patterns since changes in the event distribution might still be perceived. With the interface for interactive color assignment, we have introduced one concept for handling a larger number of events. However, if focus is not on a certain known event, manually changing a random sequence of colors can quickly become tedious. This realization has led us to consider a tight integration of the temporal mining algorithms to the visualization. Coming from this perspective, we would like to have algorithms that discover patterns, determine the events involved in the patterns and use this information to automatically select colors based on the patterns that will be revealed. This automatic color selection can be used to compute a reasonable default color assignment or it can be invoked at any time during the exploration.

In summary, two aspects of our architecture contribute to the intuitive cooperative exploration of the data by the user and the algorithms. First, CalendarView visualizes not just the data but also the patterns. Second, the same color assignment interface is used by both the user and the algorithm. We now will focus on how the following three classes of algorithms use color assignment:

- Discover one single event of one event attribute that shows an interesting pattern
- Discover multiple events of one event attribute that show an interesting pattern
- Discover one event for each event attribute such that these events together show an interesting pattern (an extension is that the user selects one event and lets the algorithm detects events of other event attributes which show some relation to the selected event, e.g. similarity, correlation, etc.)

Discover one event of one event attribute

Many existing algorithms calculate one single event based on some measure of interest [2]. These measures can range from basic statistical methods like highest variance to more computationally expensive ones like "most interesting trend". No matter how the algorithms compute the single event of interest, our approach encapsulates it and changes the colors of the events accordingly. This means all colors but one are changed to one light color, whereas the event for which the pattern was found is

changed to have a unique dark color. Thus the user can focus on the distribution of this single event in relation to the overall frequency of all events.

We have included the following implementation of such an algorithm called *LongestStreak,* which is based upon the idea of stabilized p charts from the statistical field control charting [13]:

1. For each event *e*, compute a sequence of relative frequencies as follows: For each day, compute the percentage of occurrences of event *e* based on all events occurring on the same day.
2. Compute the weighted mean and standard deviation of each sequence. Consider just the days that are event dates.
3. Label each day where event *e* is significantly below or above its mean as *significant day* with respect to event *e*.
4. Return the event with the longest streak of consecutive significant days. Break ties by returning the first one found.

Alternatively, we could also modify step 4 to return the event with the most significant days. After the visualization is updated based on the discovered event the user can continue the exploration process.

Discover multiple events of one event attribute

Again, many algorithms have been proposed which compute this class of patterns [2], e.g. discovery of similar events. The algorithm returns a set of events which together represent a pattern. Our architecture changes the color assignment such that each event that is part of the pattern is assigned a distinct color, and all other events are assigned to one color.

Our implemented instance of this class of algorithms called *MatchingEvents* extends *LongestStreak* described above:

1. For each event, compute significant days and record a bit sequence having a '1' for each a significant day and a '0' otherwise
2. Take *LongestStreak* as the baseline event
3. Compare the bit sequence of the *LongestStreak* event with all others to find the closest match. This is determined by a bit-wise comparison and each match of a '1' in both sequences increments the match counter by one. The event whose bit sequence has the highest match counter is the *correlated event.*
4. Return the LongestStreak event and the correlated event.

Discover one event for each event attribute

The two previous algorithms have looked for patterns in one single event attribute. In contrast, this class of algorithms looks for patterns relating event attributes to each other, instead of analyzing them separately. Many proposed algorithms fall into this class, e.g. finding similar events across different event attributes. The resulting pattern is visualized by updating the color assignments of each event attribute accordingly.

We implemented an instance of this class very similar to *MatchingEvents*. But instead of comparing the *LongestStreak* of the first event attribute to other events of the same attribute, it is compared to all events of the other event attributes. The algorithm returns the *LongestStreak* of the first event attribute and for each other event attribute the event that is correlated. In the experimental section, we refer to this algorithm as *MatchingEvents2*.

6 The Database Component

In this paper, we assume the datasets reside in tables from one or more relational databases. The integration of a database component should provide access to the data, a mechanism to scale up to large datasets and the capability to access the raw data of all attributes associated with the patterns found.

The critical part of the database component is to scale up to large databases. For our architecture, scalability entails a visualization and a memory aspect. The first aspect, namely how to visualize large datasets is addressed by visualizing the relative frequency of events on a single day as described in section 4. In this section, we describe how large datasets are processed. The fundamental idea is to compute an aggregated version of the dataset such that it fits in main memory. The aggregated dataset contains sufficient statistics similar to e.g. [4] for decision trees, and we show the upper bound of the main memory requirements based on our assumptions stated in section 3.

Let us pick up our example dataset from table 1. This dataset might consist of millions of rows since each occurrence of an event is typically stored as one record. If we use the aggregation capabilities of the database, the number of records that are loaded can be significantly reduced. Instead of storing each occurrence of an event, we count for each day the number of occurrences for each event. E.g. the sufficient statistics for event attribute "page hits" can be computed by submitting the following SQL query:

```
SELECT Event_date, page_hits, count(*) as Frequency
FROM example_table
GROUP BY Event_date, page_hits
ORDER BY Event_date, page_hits;
```

The resulting table is sketched in table 2. The amount of compression achieved by aggregation depends on the number of distinct event dates, the number of distinct events and how distinct events are distributed across the dates.

Table 2. Sufficient statistics for event attribute "page_hits"

Event date	Event attribute (page_hits)	Frequency
1/1/2002	Index.html	1934
1/1/2002	Dep1/contacts.html	36
…	…	…

The memory requirement for our initial dataset is proportional to the number of entries in a relational table. For one event attribute, event dates and events of this attribute have the memory requirements mem_{init}, with

$$mem_{init} \propto number\ of\ days \cdot average\ number\ of\ events\ per\ day$$

In contrast, the memory requirements mem_{new} for the computed sufficient statistics table (table 2) is

$mem_{new} \propto number\ of\ days \cdot average\ of\ the\ number\ of\ distinct\ events\ per\ day$

The difference in memory usage is the ratio between the average number of events per day and the average number of distinct events per day. This ratio will vary with the domain and the event attribute. For example, in the aircraft maintenance domain for one airline we had:

Average number of events per day: 402

Average number of distinct events per day: 32

The ratio in this example is 12.5:1. Whereas the number of records grows linearly for the initial dataset with every new event, our new table typically just increments a counter. This is most useful in domains where the number of events per day is very high, like web page accesses, items in market baskets across departments, phone calls, etc.

Given our assumptions from section 3, the worst case memory requirements mem_{worst} for the sufficient statistics table of one event attribute can be computed for e.g. 15 years:

$mem_{worst} \propto 15 \cdot 365\ days \cdot 200\ distinct\ events = 1{,}095{,}000$

In this case, every event happens every day at least once during a period of fifteen years. We can store each event with one byte (next to a small lookup table) and the days and frequency as integers with 4 bytes. The sufficient statistics table would require: $1{,}095{,}000 \cdot (1 + 4 + 4)$ = about 9.8 Megabytes. Together with our assumption that the number of event attributes is low, we can conclude the sufficient statistics tables fit in main memory for many domains.

To summarize, the database component is integrated in two ways: First, the relevant event attributes of the original tables are compressed by computing the summary statistics offline. Second, database access is provided in a straight forward way: Since the user basically selects subsets of the initial time period during the exploration process, he can decide to retrieve the records with all attributes corresponding to the selected time period. Then a range query over the time period returns the raw data of interest. In our experiments, the computed summary statistics always fit in main memory and the computation of the proposed algorithms is efficient. Both, we believe is true for most datasets which fulfill our assumptions in section 3.

However, if more attributes are involved in an algorithmic run, or the integrated algorithms are more complex, then a tighter integration with the database component might be necessary. E.g. algorithms might be decomposed and leveraged by SQL extensions or user-defined functions could be used. If the algorithmic run is pushed back to the database, the user can continue to explore the data and get notified after the computation is finished.

7 Experiments

In our experiments, we investigate several real-world datasets from the airplane maintenance domain. We think the scenario we describe in this section is similarly applicable to many other domains like homeland security, web mining, market basket analysis or intrusion detection. The datasets are tables from a database containing maintenance events of different airlines for different airplane models[3]. Maintenance events range from negligible ones like coffee spills on the seat to major ones like problems with a landing gear. Each record has information about the date a maintenance problem has occurred, the airport where it was recorded, who discovered it, the written complaint, the maintenance action taken, the system and subsystems affected by the problem, etc… We will focus on the affected systems, which will be our event attribute. A system is a set of related parts that work together to perform a function such as communication, engine, flight control, doors, etc.

Table 3. Datasets

Dataset	Event dates	Nr. of events	Nr of records (originally)	Nr. of records (suff stat)
A	3/6/89-12/31/02	37	350,772	87,030
B	5/12/90-12/31/02	39	1,165,881	117,441
C	1/30/89-12/31/02	41	1,405,582	133,116
D	3/6/89-12/31/02	28	350,772	78,802
E	11/12/89 - 12/31/02	41	2,051,269	162,918
F	1/12/89-12/31/02	182	2,051,269	574,071
G	12/27/89 - 12/31/02	40	17,499	11,547

Table 4. Runtime (in seconds) of algorithms

Dataset	LongestStreak	MatchingEvents	MatchingEvents2
A	0.27	0.31	0.53 (with B)
B	0.31	0.30	0.62 (with C)
C	0.35	0.36	0.54 (with D)
D	0.28	0.26	0.63 (with E)
E	0.37	0.36	0.9 (with F)
F	0.71	0.68	0.87 (with G)
G	0.23	0.22	0.47 (with A)

[3] An airplane model is e.g. 747, 767, etc.

Metadata about the various datasets we explored is depicted in table 3. The recorded maintenance datasets span time periods between twelve and fourteen years. Table 4 shows the runtime of our implemented algorithms on the datasets. For the algorithm MatchingEvents2, we also indicate in brackets which other dataset has been the second event attribute. We ran all experiments on a PC with a Pentium III/ 800 Mhz processor and 1 GB main memory. For all datasets, we achieve an acceptable runtime.

7.1 Mining Airplane Maintenance Datasets

We describe a typical scenario which shows how our approach can be used. We start our investigation by selecting a dataset from one airline and one model. The chosen event attribute which we analyze over time is the system of the airplanes. Since there are a lot of different systems, we select the algorithm LongestStreak to compute one interesting system (it found engine fuel) which updates the color assignment. Figure 7 top row shows a small range of the resulting visualization. Especially during the last five days of July 2000, we perceive many events, indicating problems with engine fuel. Next, we add several datasets to compare this finding with patterns for different airlines. For each airline and the same model, we manually change the color assignment of the systems. We color every system except engine fuel with one light color and assign a dark color to all engine fuel related events. When we compare these airlines (two more airlines are shown in figure 7), we see the other airlines do not show a specific pattern. Even though just a small time range is shown, it is the case for all event dates. So we might decide to further investigate the first airline. Now we add to the first dataset another dataset which aggregates individual airplane id's of the same airline and model over time. The event attribute of the newly added dataset is the airplane id and we would like to find a correlation between the events we identified concerning engine fuel and maintenance events of individual airplanes. We run the algorithm *MatchingEvents2* to single out one airplane. This airplane is shown in figure 8 and we see e.g. that a lot of maintenance events for this single airplane have occurred on December 3rd, 1997. Note that for brevity we have omitted a screenshot of the corresponding time range of figure 7.

Finally, we select a dataset with maintenance events of just this airplane. The event attribute is again airplane systems. We run the algorithm *MatchingEvents* to see if two events frequently co-occur. A part of the resulting visualization is shown in figure 9. The two correlated events returned are fuel and communications indicated by the black and light gray color. E.g. on Monday 18th November, both events co-occur. With this knowledge we drill down to the raw data to further investigate the findings.

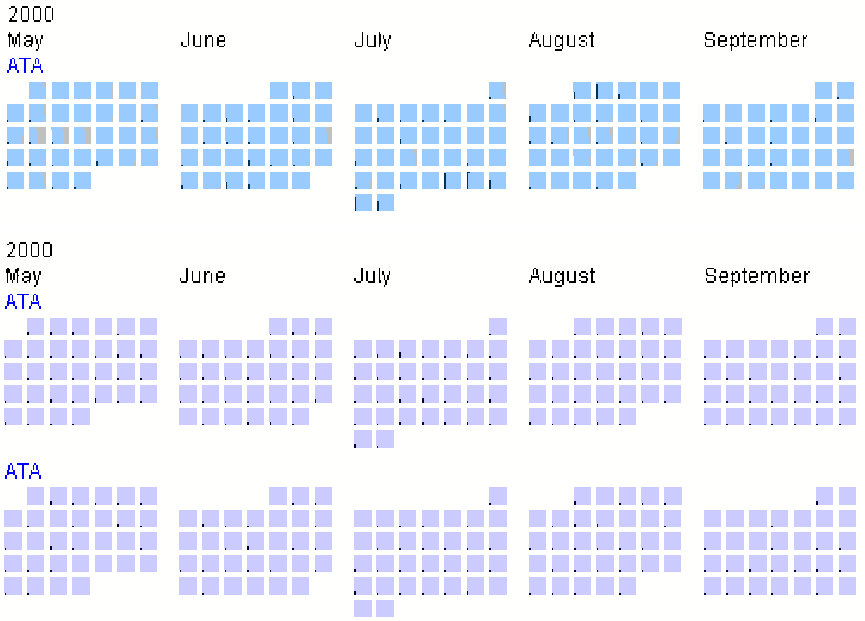


Figure 7. Focusing on maintenance events with the same subsystem for three different airlines

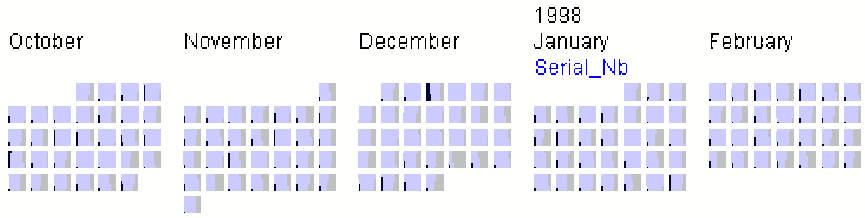


Figure 8. CalendarView focusing on maintenance events for one airplane

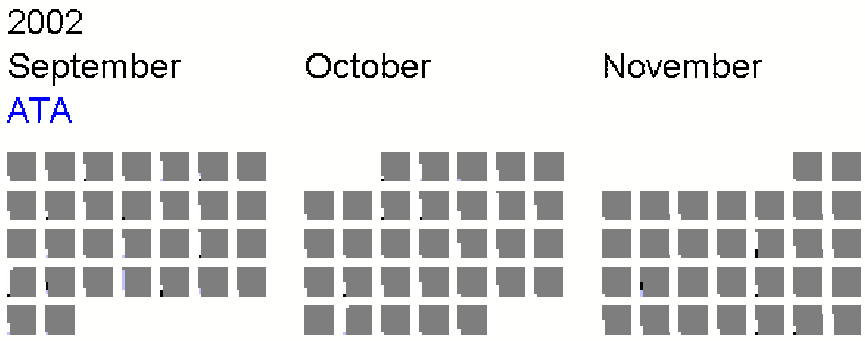


Figure 9. CalendarView focusing on maintenance events with two subsystems for one airplane

7.2 The DataJewel System

We have implemented the DataJewel system based on the architecture proposed in this paper. It can quickly be adapted to new domains since it is designed to be extensible for new visualization techniques and new algorithms. In the figure 10 two useful features are shown. First, the raw data can be accessed, displayed, saved or printed. Whereas the temporal analysis is based on just a few event attributes from possibly different tables, the user typically is interested in other attributes of the records corresponding the pattern found. As the data has been distilled and narrowed down during the exploration, the current range of event dates represents the dataset of interest. Thus just one range query is submitted against the database(s) to retrieve the attributes of interest. Second, an optional tree on the right side depicts the exploration process. The simplified temporal mining process presented in section 3, focuses on the iterative process of reducing the dataset. However, at some points the user may like to return to a previous stage, either because he found something of interest or not. Therefore, the tree on the right side shows a node for each subset of data explored so far. The user can either return to a node or annotate a node.

The algorithmic component can be used in three ways. It can be used to determine the default color mapping, it can be invoked at any time during the exploration process or it can run as a background process in parallel to the user's exploration and notify him upon discovery of some patterns. In addition to updating the color assignment after patterns have been found, the event dates not covering the patterns can be grayed out. Alternatively, the patterns can be displayed in a textual form.

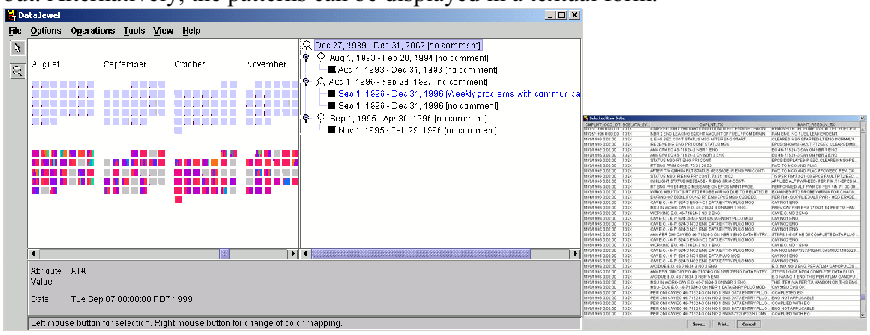


Figure 10. Screenshots of DataJewel

7.3 Discussion

We think the DataJewel architecture is also well adapted to areas like homeland security, market basket analysis or intrusion detection. Homeland security tasks like identifying suspicious behavior can be supported by our architecture in several powerful ways. For example, different event attributes can be associated with each other even though their events take place at different dates, months or possibly years. For intrusion detection, data may be aggregated hourly instead of daily, therefore an additional visualization technique would need to be added to the visualization component.

In the context of market basket analysis, many algorithms have already been proposed and successfully used to find patterns. The discovered rules look like: *If a customer buys bread and sugar then she is likely to buy beer as well*.

These algorithms look for items that are frequently bought together, however, they do not make use of the time that is associated with each transaction. Analyzing market basket databases over time can reveal a new set of patterns like: *Customers are likely to buy cereal and fruits in the beginning of the week and alcohol and candies at the end of the week*.

Note that our approach would be suitable for these datasets even though the dimensionality of market basket databases is typically very high (hundreds or thousands of items). Each item is usually modeled as one attribute and a record corresponds to all items purchased by one customer. Instead, for our approach we would map all items to different events of one attribute and store the frequency of the corresponding items bought per day. If the number of items is very large, a concept hierarchy could be used to generalize to fewer items, as outlined in section 3.

8 Conclusions

Visualization, mining algorithms and databases are main areas in the field of KDD. Most research concentrates in just one of these areas. Our work is based on an integrated approach that we believe can significantly improve the discovery of useful and understandable patterns. We present a novel user-centric architecture for temporal data mining, tightly integrating a visualization, an algorithmic and a database component. We introduce a new visualization technique called CalendarView for representing temporal data. One main contribution is the use of the same visualization for the data and for the computed patterns. In addition, we designed an interface of assigning colors to categories, which is used by both the user and the algorithms. On the one hand, the user can steer the exploration or incorporate his domain knowledge, on the other hand, the algorithm can suggest meaningful color mappings based on the pattern discovered. By precomputing sufficient statistics from the initial datasets, our approach scales up to very large databases.

In our future work, we will apply DataJewel to different areas, using the extensible architecture to add new visualization and algorithmic components. We will investigate how our approach can be extended to fit different data types like text or multimedia data.

9 References

[1] Ankerst M., Ester M., Kriegel H.-P.: Towards an Effective Cooperation of the Computer and the User for Classification, SIGKDD 2000, Boston, MA.

[2] Antunes, C.M., Oliviera A.L.: Temporal Data Mining: an overview, SIGKDD 2001 Workshop on Temporal Data Mining, San Francisco, CA.

[3] C. Daassi, M. Dumas, M-C. Fauvet, L. Nigay, P-C. Scholl : Visual Exploration of Temporal Object Databases, Proc. of BDA'00, 24-27 October 2000, Blois, France, pp. 159-178.

[4] Gehrke J., Ramakrishnan R., Ganti V.: RainForest – A Framework for Fast Decision Tree Construction of Large Datasets, Data Mining and Knowledge Discovery journal, Vol. 4, p.122-162, Kluwer, 2000.

[5] Grinstein G., Ankerst M., Keim D.A.: Visual Data Mining: Background, Techniques and Drug Discovery Applications, SIGKDD 2002, Tutorial, Edmonton, Canada.

[6] Havre S., Hetzler E., Whitney P., Nowell L.: ThemeRiver: Visualizing Thematic Changes in Large Document Collections", IEEE Transactions on Visualization and Computer Graphics, Vol.8, No. 1, January-March 2002.

[7] Hellerstein J.M., Avnur R., Raman V., Informix under CONTROL: Online Query Processing, Data Mining and Knowledge Discovery journal, Vol. 12, p.281-314, Kluwer, 2000.

[8] Hinneburg A., Keim D.A. Wawryniuk M.: HD-Eye: Visual Mining of High-Dimensional Data, IEEE Computer Graphics and Applications, Vol. 19, No. 5, 1999.

[9] Keim D.A., Hao M.C., Dayal U.: Hierarchical Pixel Bar Charts, IEEE Trans. On Visualization and Computer Graphics, Vol. 8, No. 3, pp 255-269, 2002.

[10] Kolluri V., Provost F.:A Survey for Scaling Up Inductive Algorithms, Data Mining and Knowledge Discovery journal, Vol. 2, p.131-169, Kluwer, 1999.

[11] Mackinlay J.D., Robertson G.G., deLine R.: Developing Calendar Visualizers for the Information Visualizer. Proc. UIST '94, 1994.

[12] Sarawagi S, Thomas S., Agrawal R.: Integrating Mining with Relational Database Systems: Alternatives and Implications. SIGMOD Conference 1998, 343-354.

[13] Trueblood R.P., Lovett J.N. Jr.: Data Mining and Statistical Analysis using SQL, Apress, ISBN 1893115542, 2001.

[14] Van Wijk J.J., Van Selow, E.R.: Cluster and Calendar based Visualization of Time Series Data, IEEE InfoVis '99, San Francisco, October.

[15] Yang L.: Interactive Exploration of Very Large Relational Datasets through 3D Dynamic Projections, SIGKDD 2000, pp.236-243, Boston, MA.

Exploring Non-Linear Data Relationships in VR using the 3D Visual Data Mining System

Henrik R. Nagel, Michael Vittrup, Erik Granum, and Søren Bovbjerg

Computer Vision and Media Technology Lab.
Aalborg University
Denmark
{hrn, vittrup, eg, sb}@cvmt.dk

Abstract. A software system has been developed for the study of static and dynamic data visualization in the context of Visual Data Mining in Virtual Reality. We use a specific data set to illustrate how the visualization tools of the 3D Visual Data Mining (3DVDM) system can assist in detecting potentially interesting non-linear data relationships that are hard to discover using traditional statistical methods of analysis. These detected data structures can form a basis for specification of further explanatory statistical analysis. The visualization tools are shown to reveal many interesting patterns and in particular the dynamic data visualization appears to have a very promising potential. The results encourage further developments of the current and new tools in the context of practical applications.

1 Introduction

Most Visual Data Mining (VDM) methods have been designed for PC hardware using 2D graphics or 3D graphics on a monitor. VDM in immersive 3D Virtual Reality (VR) systems using CAVE [1] or Panorama[1] arenas for visualization has so far been dependent on special and expensive hardware, in order to achieve real-time response to user interaction and navigation.

With the continuous improvements in computer technology, it is now possible with standard high-end PCs to drive VR systems. They can visualize many objects while simultaneously navigating among them and perform intensive background calculations, all in real-time.

However, while the main principle behind the design of traditional VDM methods is that e.g. 3D Scatter Plots are viewed from the "outside-in", the immersed VR users can navigate around inside a 3D Virtual World (VW), which may then also be viewed from the "inside-out". VR applications provide more comprehensive input to the human senses and can therefore make more efficient use of the human perceptual skills, when exploring large datasets.

The potential benefit of immersive VR for VDM has been debated [2]. Our hypothesis is that a complementary and valuable benefit is achievable concerning

[1] A stereo display wall with 160^0 field of view, a diameter of 7,50 meters and a height of 3,50 meters

the detection of e.g. non-linear relationships in data, which are most likely to escape traditional methods of data analysis.

To support our hypothesis, this paper presents the visualization tools for exploring data in VR, which are available in a new version of the 3D Visual Data Mining (3DVDM) system [3].

We study an example case with the visualizations tools, and we show how they can help discovering special relationships in data that may warrant further statistical analysis, by e.g. domain experts.

We will *not* derive conclusions with respect to practical nor statistical importance of the phenomena detected in the data, as this is outside the scope of this paper. We also appreciate that the benefit of immersive VR is hard to illustrate in a paper. All illustrations presented here are snapshots dumped from a mono display on a monitor.

2 Previous Work

VDM methods, such as "The Grand Tour", have been implemented in VR in several occasions [4, 5], as well as the traditional method for data exploration called "Brushing and Linking" [2].

A well known approach for visualizing multivariate data is to map data variables to visual object properties, such as position, size, shape, color, orientation, etc. Such visual objects are called glyphs [6–8], and can be in both 2D and 3D. Glyphs are efficient due to the ability of the human brain to discover patterns within and among objects, as well as to recognize objects that do not "fit" into a discovered pattern.

An example of the use of glyphs is "Chernoff Faces" [9], in which each observation in a dataset is represented by a cartoon face of which features, such as length of nose, curvature of mouth, size of eyes, and even the shape of the face itself, correspond to variables of an observation. Colored texture has also been used to visualize multivariate data elements arranged on an underlying height field [10]. Using volume visualizations of 3D scatter plots with glyphs representing data point [11], it is possible to use procedural shape generation techniques. These techniques allow from 1 to 14 additional data dimensions to be visualized using glyph shape.

It is important to note that the target is dependent on the success of the user *comprehension* of multivariate data. Thus it is not necessarily a goal in itself to visualize as many variables as possible simultaneously, but to make it possible for analysts and domain experts to get a useful impression of relationships between multiple variables.

3 The 3DVDM System and Tools

The 3DVDM system presented in this paper is the second generation of a software system for exploring data in VR. While the first generation of the software [3] pre-rendered visualizations of data in a time-consuming process, this

second generation of the software renders visualizations of data in real-time, while users are watching. This, potentially, opens up for many new possibilities, of which some are mentioned in this paper.

The second generation of the 3DVDM system is based on a separate VR software framework called VR++ [12, 13], which currently runs on Linux as well as Irix. While VR++ provides functionality for e.g. parallel-processing, communication, parameter control, and visualization of geometric objects, the 3DVDM system provides specialized VDM tasks. VDM tools are created by connecting suitable tasks from both VR++ and the 3DVDM system. Current PCs are sufficiently powerful to run the system for display on standard monitors, or if networked, drive our CAVE or Panorama visualization arenas.

3.1 Data Pipeline

Figure 1 shows the general approach adopted in this research for visualizing representations of data from databases.

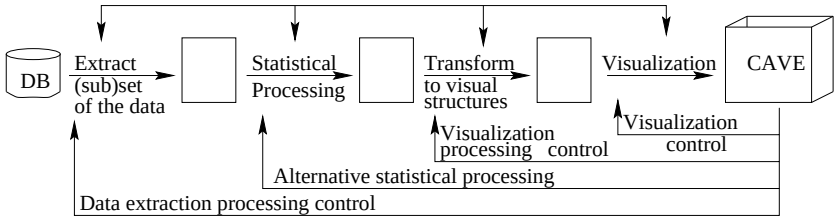


Fig. 1. The 3DVDM Data flow and interaction patterns

The system contains different data processing modules in a pipeline with the possibility of feedback from users to each module. This has been achieved by adding a control panel, which in real-time and with visual feedback allows for modification of parameters on which a visualization is based.

First, a relevant subset of the data in a database is extracted and stored as an easy accessible, internal database, which is passed on for processing to later stages of the pipeline. The 3DVDM system has specially designed software for loading "comma-separated-values" (CSV) files, in an efficient way. This is accomplished by performing a complete analysis of the data the first time they are loaded. The result of this analysis is stored along with the data in binary format, beside the original data file. When the dataset is loaded again, the result of the analysis phase is loaded instead of the original data. This allows fast loading of large datasets, once initially analyzed.

The data handling part of the 3DVDM system is designed with real-time rendering in mind, rather than with handling of large databases in mind. This means that emphasis has been put on efficient storage of data in memory (RAM), and efficient extraction of data from memory, rather than on, e.g. on-line access to data from harddisks.

The next stage in the pipeline is statistical processing. The 3DVDM system has only simple facilities for statistical computations and manipulations. While this topic is in general very important, the research reported in this paper, has emphasis on visualization facilities. Off-line statistical analysis to prepare for use of the 3DVDM system is recommended.

Data is subsequently transformed into an equivalent symbolic graphical representation. This data format is independent of specific hardware and software requirements. Last step is to transform this data to polygons, which are rendered in a 3D space for visualization.

3.2 On Explorative Data Visualizations

The 3DVDM approach to data visualization is built around the possibility to navigate around in a visual world (VW) - an "Extended 3D Scatter Plot" - to explore arbitrary view directions from arbitrary view points. "Extended" means that data points are visualized as objects, which may have different visual properties.

The ease, with which navigation occurs, highly depends on the frame-rate and therefore the number of visual objects in the VW and on the complexity of each visual object. It is an advantage to be able to visualize many objects as possible, and this means that the complexity of each visual object must be reduced as discussed in [3].

Another consequence of exploring data in the above VW is that attention must be paid to the distance between VR users and 3D visual objects currently observed. This is discussed in [14], where it is suggested that object size is kept constant to allow the perceived size to support depth perception. Object size is also the reference for a spatial distance perception in the general, and for design and evaluation of perceptual conditions. Some properties like surface texture should be observed at close range (measured in object size units) to support a perceptual grouping, object shape still works at "medium" range, while color is much more dominant perceptually and can be seen to define cluster structures also at long range distances of observation.

3.3 Dynamic Data Visualizations

Dynamic Visualization (DV) of data is a new feature of the 3DVDM system. It can be implemented in several ways, e.g. precomputed animations and real-time computed animations. For each of these two cases, there are two possibilities of interest: few frames shown for several seconds each, or many frames shown for a fraction of a second each.

VR++ is a software framework specialized for creating VR applications with real-time computed animations with many frames per second. This animation form requires data visualizations to change so frequently, in response to changes in both data and user input, that users experience a smooth animation, which also reacts appropriately to real-time interaction, such as head movement.

4 Preparing a Demonstration Case

To demonstrate the facilities and the potential of the 3DVDM system we first outline a typical procedure for its use. A specific dataset used to illustrate the 3DVDM tools is then described, followed by a pre-analysis, which allows us to directly make use of 3DVDM.

4.1 Typical use of 3DVDM

The basic principle is that statistical observations are visualized as objects in a 3D scatter plot. Three (preferably quantitative and continuous) variables are used to define the 3D coordinate system, and other statistical variables may be represented as object properties, and be encoded as surface color and texture, object shape, orientation etc.

The initial problem is to find and select one or more sets of three suitable variables (triplets) to define the coordinate system and hence the spatial layout of the objects in the 3D space.

Simple uni- and bi-variable analyses using a standard statistical package are useful for a start. Continuous variables with "regular" distributions are candidates for a triplet, and variables in a triplet should have "low" correlation in order to exploit (populate) the 3D space efficiently.

When a set of candidates is selected, a "scatter plot tour" facility of the 3DVDM system allows systematic inspection of all unique combinations of triplets. Using a control variable encoded as object color, each candidate triplet may be investigated in as much detail as desired, but given the typical number of candidate variables, a first run through may be used to eliminate the least interesting variables/triplets.

Having selected one or some triplets of interest, a more detailed 3D scatter plot analysis and visual exploration can take place. Alternative sets of the other variables can be assigned to object properties, and the user can navigate around and observe the visual world from "inside-out" or "outside-in" as he or she pleases. Being observant throughout the above process, it is very likely that interesting (sub)structures in the data will be observed.

While the above relies on a static visual world within which the user can navigate around, the system now also allows dynamic visualizations of data. Color scales can cycle with real-time feedback, and a "Macro Dynamics" facility allows a window sliding through a sorted data base, such that a sort-variable in a sense is mapped to a time scale. A "Micro Dynamics" facility allows statistical variables to control movement of all objects individually.

Another new facility in 3DVDM allowing the use of sound to support and complement the visual analysis is described in a separate paper [15].

4.2 Data Preparation

A publicly available dataset called "Forest Cover Data" is used[2]. The dataset was used to predict the forest cover type for 30×30 meter cells on the basis of cartographic variables obtained from the US Forest Service (USFS) Region 2 Resource Information System (RIS) data [16].

Data Summary

From the dataset documentation the following basic information is extracted:

Number of observations:	581012
Attribute breakdown:	10 quantitative variables, 1 categorical (wilderness areas, 4 categories) and 1 categorical (soil type comprising 40 binary variables)
Independent variable:	Forest Cover Type (7 categories)
Missing Values:	None

Name	Index	Values	Range
Elevation	1	1978	1859 – 3858 meters
Aspect	2	361	0 – 360 azimuth
Slope	3	67	0 – 66 degrees
Horizontal Hydrology Distance	4	551	0 – 1397 meters
Vertical Hydrology Distance	5	700	-173 – 601 meters
Horizontal Roadways Distance	6	5785	0 – 7117 meters
9am Hill-shade	7	207	0 – 255
Noon Hill-shade	8	185	0 – 255
3pm Hill-shade	9	255	0 – 255
Horizontal Fire Points Distance	10	5827	0 – 7173 meters
Wilderness Area	11	4	Cache la Poudre, Comanche Peak, Neota, Rawah
Soil Type	12	40	1 – 40
Forest Cover Type	13	7	Aspen, Cottonwood/Willow, Douglas-fir, Krummholz, Lodgepole Pine, Ponderosa Pine, Spruce-Fir

Table 1. Basic information about the variables in the Forest Cover dataset

The dataset is not balanced with respect to the dependent variable, where the number of observations range from 2747 to 283301 for the individual forest cover types; refer to the web-site for more information and description of the variables and basic statistics.

Data Conversion

There are 13 variables of which the last three are categorical (qualitative), that

[2] http://kdd.ics.uci.edu/databases/covertype/covertype.html

is, without any meaningful ordinal order. The categorical variable Forest Cover Type, index 13, is encoded numerically as 1 to 7 and can directly be used. Index 11 and 12 are encoded as 44 binary variables (4 mutually exclusive values from Wilderness Area and 40 from Soil Type) were converted to two numerical variables, as seen for index 11 and 12 in table 1.

The variable "Soil Type" is generally excluded from further consideration in this study due to interpretation ambiguities.

4.3 Basic Statistical Analysis

A standard statistical package is used to provide basic information of the data set. All histograms are uni-modal and most have skew distribution. Index 5 and 7 have very narrow distributions compared to the range, and may not have a lot to offer. Index 2 (Azimuth, 0^0-360^0) has a "circular" distribution peaking at 50^0 and minimum at 230^0.

Correlation Analysis

Figure 2 shows a color encoding of the correlation coefficients σ_{xy} for the 10 quantitative variables of the dataset. Keep in mind that a correlation coefficient σ_{xy} of -1 is just as strong as a correlation coefficient of 1.

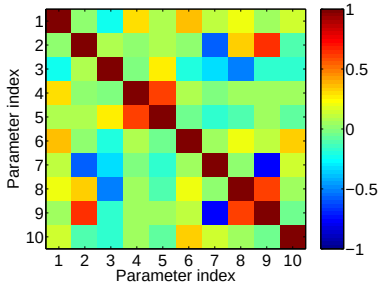


Fig. 2. Correlation coefficients σ_{xy} for the variables, index 1-10

Strongest (positive or negative) correlation coefficients are found between Hill-shades, $\sigma_{79} = -0.78$ (9am/3pm Hill-shade), and $\sigma_{89} = 0.59$ (Noon/3pm Hill-shade), while $\sigma_{78} = 0.01$ (9am/Noon Hill-shade) is surprisingly weak. Other strong correlations are $\sigma_{45} = 0.60$ (Horizontal/Vertical Hydrology Distance) and $\sigma_{29} = 0.65$ (Aspect/3pm Hill-shade).

If the estimation of the linear correlation is too strong, one of the involved variables could be discarded in order to avoid redundancy. However, non-linear relationships may underly the computed correlations.

2D Histograms

The 3DVDM system can illustrate 2D histograms for all unique combinations

of two variables of the data set, where the color is mapped to the number of records. This is shown in figure 3.

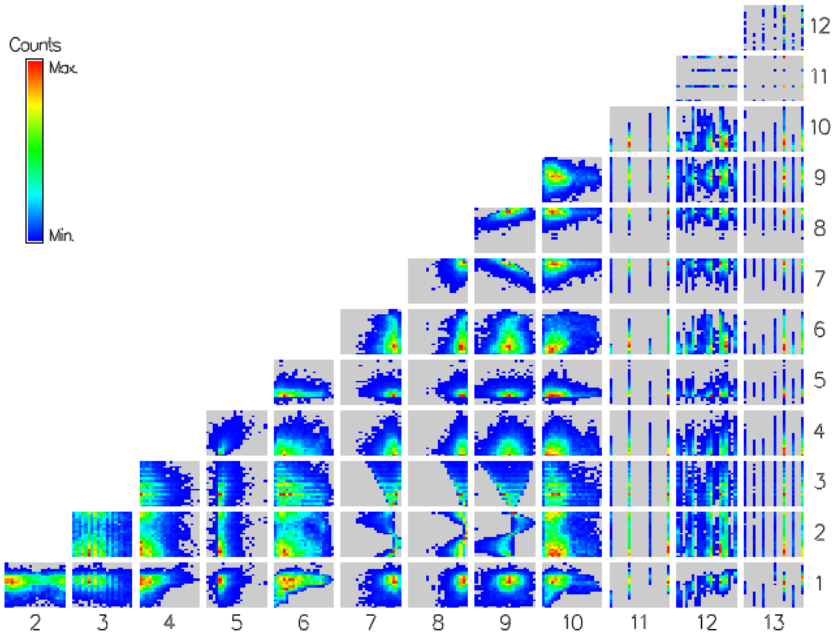


Fig. 3. An auto-scaled 2D histogram for all combinations of two variables

The 2D histograms reveals both linear and non-linear relationships. As the number of variables is relatively low, we retain all 10 quantitative variables as potential candidates for mapping to the coordinate axes in "interesting" triplets. Soil Type, index 12 is discarded from further investigation here, while index 11 and 13 may serve as "dependent" variables in our investigations below.

5 Visual Exploration of Static Worlds

In its simple form, a 3D scatter plot in 3DVDM can visualize four variables - three variables are assigned to the axes, and a fourth (dependent) variable is assigned to color representation.

We will here discuss a "tour" facility of 3DVDM, and how to use the results from this "tour" to the best of its account.

5.1 3D Scatter Plot Tour

We have retained all 10 quantitative variables as candidates for "spatial" triplets. There are, in general, many possible combinations of mappings of variables to

the axes of a 3D scatter plot. In our example, the combinatorics leaves us with the following number of unique triplets:

$$\frac{10 \times 9 \times 8}{1 \times 2 \times 3} \quad = \quad 120 \tag{1}$$

By discarding one variable we would have 84 unique combinations, and discarding two variables would leave 56 and so forth.

The "3D Scatter Plot Tour" tool displays step by step a 3D scatter plot visualization of all these triplets. With this tool one can e.g. watch a sequence of 3D scatter plot visualizations, where the mappings to the three axes of the coordinate system change at regular intervals, while the color represents the dependent variable. The tool can also be used as an "Object Property Tour" by assigning fixed mappings to variables for the three axes of the coordinate system, and letting the mappings to the other object properties change regularly.

3DVDM allows the user to rank the visualizations according to how "interesting" they look, by manually assigning a score from 0 to 9 to the individual combinations. These combinations, together with their score and basic statistical information, are stored in a log-file for easy access and analysis.

Patiently watching the "tour" with color representing Forest Cover Type we find many interesting visualizations, see figure 4 for a few interesting triplets.

All of the 10 variables were involved in "interestings" visualizations, but variables 1, 3, 4, 6, 7, 8, 9, and 10 were dominating. Conclusion of the tour is to continue with closer analysis of the triplets 1, 4, 6 (Elevation, Horizontal Hydrology Distance, and Horizontal Roadways Distance) and 7, 8, 9 (9am, Noon, and 3pm Hill-shade).

5.2 3D Scatter Plots and Object Properties

Once interesting triplets have been found, further analysis can be performed in VR. This section briefly describes navigation and how to use some of the scatter plot features of 3DVDM, e.g. the programmable dynamic color scales.

Exploring the Space

Navigation in the virtual 3D space means here to control both the viewpoint and view direction as one pleases. Hence one may "fly" around and within the visualized coordinate system and observe the objects.

The navigation interface and display of the current view are all dependent on the visualization system used. When using the CAVE stereo visualizations are provided and all view directions are available for the user, when he moves his head.

The Panorama is a popular visualization system for our VDM, and navigation is controlled according to the direction of a "Wanda", i.e. a device that is tracked with 6 degrees of freedom (position and orientation). Here view direction is fixed to the (forward) "motion" of the navigation. View direction is also fixed in the case when using a monitor, and here the arrow keys may control the movement of view point.

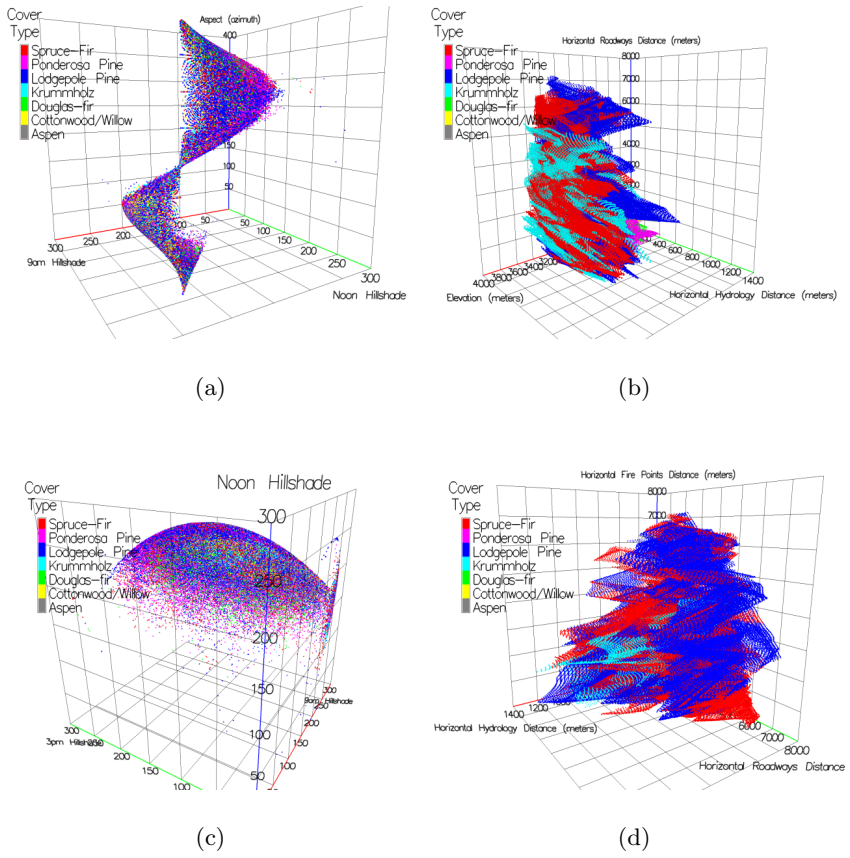


Fig. 4. Examples from 3D scatter plot tour with color representing Forest Cover Type

The most important feature in the context of VDM in VR is the real-time response to user movement while intensive background processing is being performed, - something that cannot be done with traditional desktop VDM software.

Extended 3D Scatter Plot Analysis

We choose two triplets after the tour, and starting with (figure 4(b)) we can now experiment with different use of color mappings. It may be of interest to see how the Wilderness Area variable is distributed when using this triplet, and we therefore choose to map this variable to color instead of Forest Cover Type. This situation is shown from two different viewpoints in figure 5.

For this triplet, two of them (elevation, horizontal roadways distance) show relevance for the 2D distribution of Wilderness Area, while the third (horizontal hydrology distance) drags out some specific substructures.

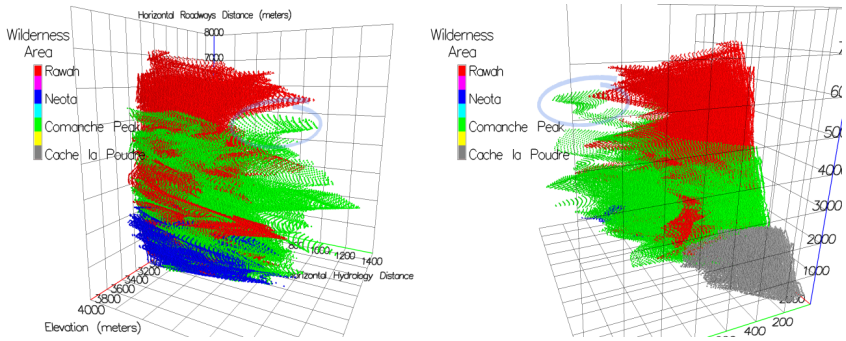


Fig. 5. Same mapping to axes as in figure 4(b), but with Wilderness Area mapped to color. The figure is seen from two different viewpoints.

Taking the visualization further, we can explore relationships between more variables, e.g. by mapping other variables to object properties like object shape, orientation, brightness, opacity, etc. Figure 6 shows a closeup on the marked area in figure 5, again seen from two different viewpoints.

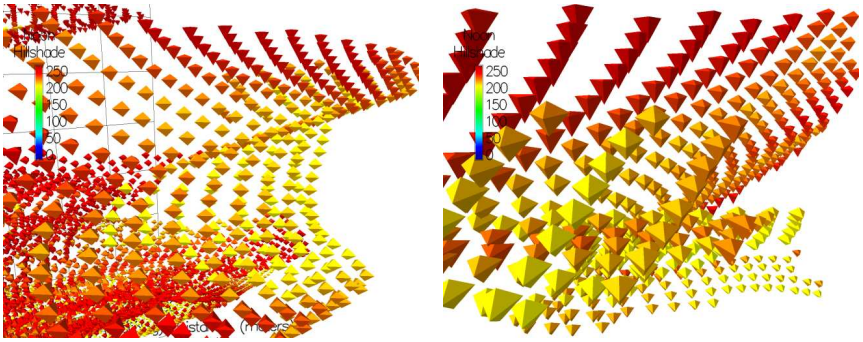


Fig. 6. 3D scatter plots with Noon Hill-shade mapped to color and slope mapped to shape. This figure is a close-up on the marked substructure in figure 5.

These sub-figures have same spatial mapping as in figure 5, but this time with Noon Hill-shade mapped to color and Slope mapped to shape. The snapshots are seen from "inside-out", in contrast to earlier figures which were viewed from an "outside-in" point of view.

Exploring Color Mappings

Now we will use figure 4(c) for further visual data exploration. Figure 7 is a scatter plot of the variables 9am, Noon and 3pm Hill-shade (index 7, 8 and 9)

mapped to the axes, now with slope (index 3) mapped to color. It is now very clear that a correlation *does* exists, although σ_{78} was computed to be only 0.01 (figure 2 in section 4.3).

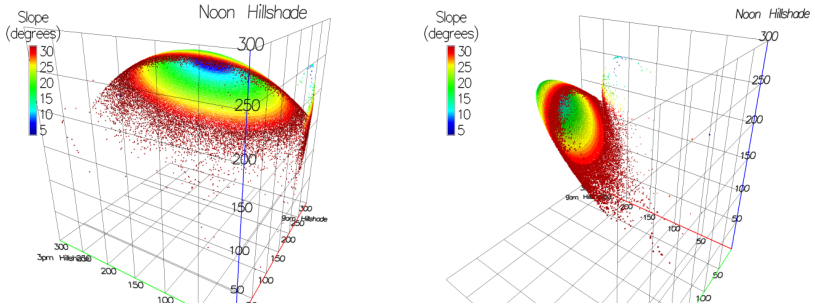


Fig. 7. Studying a selected situation using a different color scale

The scatter plots in figure 7 gives us knowledge of the dataset, which in turn can be used in a partial correlation analysis to compute the partial correlation coefficient $\sigma_{78|9}$ (that is, σ_{78} given σ_9), which yields

$$\sigma_{78|9} = \frac{\sigma_{78} - \sigma_{79}\sigma_{89}}{\sqrt{(1-\sigma_{79}^2)(1-\sigma_{89}^2)}} = 0.94 \tag{2}$$

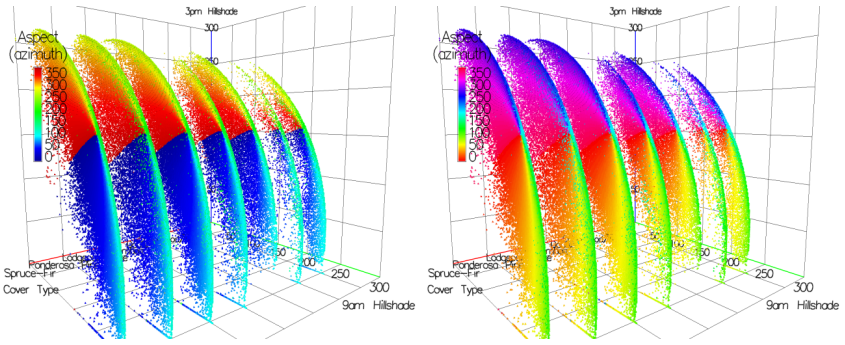
Thus, according to equation 2 there indeed exists a strong relationship between variables with index 7 and 8 (9am and Noon Hill-shade) - which is intuitively expected given their names, and seen in the plot. This clearly suggests that 3pm Hill-shade (index 9) should be taken into account.

Programmable Color Scales

Color is a very important object property, and until now we have used the "linear rainbow" color scale in two different versions - a discrete color scale (figure 6) and a continuous scale (figure 7). 3DVDM allows the user to define new color scales, or to select from a set of predefined scales - and also inversing, reversing or cycling them in order to enhance spatial structures.

The importance of using an appropriate color scale is illustrated. As an example *azimuth* (in degrees, from 0^0 to 360^0) is mapped to color, in figure 8(a). Notice that the cathegorical variable Forest Cover Type is mapped to one of the axes, which results in seven planes along this axis.

Using any of the so far presented color scales is not feasible as 0^0 and 360^0 will be mapped to widely different colors, in spite of the fact that they describe the same direction. A "wrap-around" color scale as presented in figure 8(b) solves this problem.



(a) Inappropriate color scale (b) Appropriate color scale

Fig. 8. Same figure shown using two different color scales

6 Visual Exploration of Dynamic Worlds

We consider two kinds of Dynamic Visualizations (DV), which one might distinguish between by using the terms "macro" and "micro" DV.

6.1 Macro Dynamic Visualization

In macro DV, data is sorted according to one of the variables of the database. The system then visualizes data from a "data window" which is sliding through the database. These kinds of animated visualizations therefore facilitate an alternative understanding of global trends in data, by using the time scale.

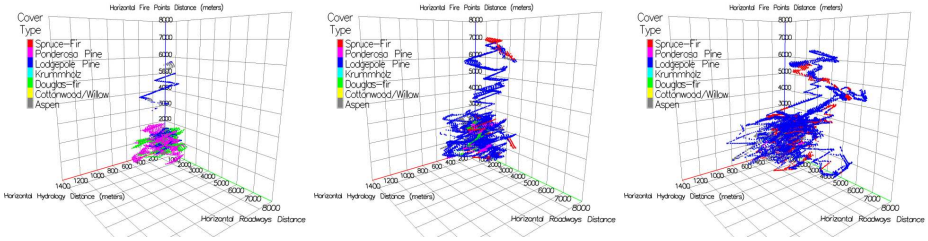


Fig. 9. Three snapshots of a macro dynamic visualization

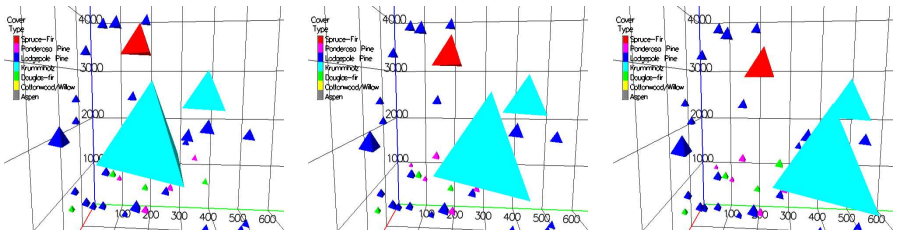
Figure 9 shows snapshots of a dynamic visualization made with the *macro* tool, provided with the 3DVDM system.

The same triplet as used in figure 4(d) is used for this illustration, and elevation is mapped to the time axis. The sub-figures show not only how Forest Cover Type (mapped to color) changes as we "walk up the mountains", but also how Fire Points Distance, Hydrology Distance and Roadways Distance change according to altitude (all horizontal). The sub-figures cannot show the smooth, continuous behavior of the "snakes", which we see in VR and real-time animation in the "real" virtual world. However, we hope the point is made that potentially interesting substructures may be detected with this facility.

If we instead choose to map azimuth to the time axis, we can walk around the mountains, giving us an impression of how Forest Cover Type is influenced by this variable given the triplet. Also, some cover types may prefer very steep slopes; this may be revealed by mapping the slope variable to the time axis, and so forth.

6.2 Micro Dynamic Visualization

In micro DV, the visual objects in a data visualization may change visual properties and/or position according to a rule that may be unique to each visual object and controlled by its statistical variable. The visual objects can e.g. change their visual properties arbitrarily with respect to position, color and shape. This makes it theoretically possible to distinguish between visual objects by their dynamic behavior. It attracts attention when a subgroup of visual objects behave similarly. Microscopic DV can be used also for detecting clusters in datasets.



(a) Frame 0/40 (b) Frame 10/40 (c) Frame 20/40

Fig. 10. Objects "vibrating" in a micro dynamics visualization

Figure 10 shows a sequence of snapshots of a micro dynamic visualization made with the 3DVDM *micro* tool. A limited number of objects are viewed with a fixed camera for illustration purposes, and the objects move individually. Variables can e.g be mapped to amplitude, frequency and phase in all three directions of the Cartesian coordinate system. It is, with this tool, for instance possible to make some objects move in circles, while other objects move in straight lines.

This tool is being developed and tested at the time of writing, so no further presentation will be given here.

7 Discussion

We aimed at demonstrating the potential benefit of immersive VR for Visual Data Mining, using a new version of our 3DVDM system and a new version of the VR++ framework upon which it is based. The system is capable of providing real-time user response and navigation as well as showing dynamic visualizations of large amounts of data.

Although we find the real-time performance of the system very adequate, we have not included any tests to support this claim.

Instead we have put emphasis on illustrative support of our hypothesis that a complementary and valuable benefit of using the system's visualization tools is achievable concerning the detection of e.g. non-linear relationships and sub-structures in data, which are most likely to escape traditional methods of data analysis.

An investigation of a data set starts with simple uni- and bi-variate analyses to become familiar with data and possibly reduce the number of variables to a reasonable number. In the case study presented we had 10 quantitative variables, and they were all forwarded to an evaluation of which combinations of three could be most interesting for defining the three spatial axes in the "Extended Scatter Plots", where more detailed visual inspection could take place.

A "Scatter Plot Tour" presented systematically the 120 unique "triplets" of the 10 variables with the dependent variable (Forest Cover Type) mapped as color of the data points (visualized as small objects). The criteria for "interesting" is entirely based on the user's subjective evaluations when observing data in 3D space. This is potentially a weakness, but the point is to complement the algorithmic methods with the power of human perception and pattern recognition.

Several interesting triplets were observed and a few were illustrated and selected for further investigation. One triplet was used to demonstrate that the objects might encode multiple visual properties representing more statistical variables simultaneously, while another demonstrated the use of flexible and controllable color scales. Eventually the Macro Dynamic visualization was demonstrated and revealed most surprising substructures in the data.

The full benefit of the 3DVDM tools assumes 3D VR visualization systems like a CAVE or a Panorama, where the user can navigate around also inside and pursue intriguing views. Such benefit can only be experienced "in situ", and for this paper we are left with the means of mono illustrations as presented on a monitor.

However, we hope that the major points of the approach and its potential do come through via the many illustrations provided and their organization as a successively progressing use of the visualization tools. Several peculiar data structures were detected through the visual inspection, and they could possibly

warrant a more detailed statistical analysis to explain the phenomena as valuable information or otherwise.

The visualizations presented are all selected on the basis of their perceptual particularities, without any claims whatsoever about practical and/or statistical significance.

8 Conclusion

Concerning the potential benefit of immersive VR for VDM, our hypothesis was that a complementary and valuable benefit is achievable concerning the detection of e.g. non-linear relationships in data, which are most likely to escape traditional methods of data analysis.

We have not presented tests that verify explicitly the benefit of navigation and real-time user response in immersive VR system, but we have illustrated the usefulness of the 3DVDM framework designed for VR through a series of examples. The VDM tools do help in discovering remarkable non-linear data relations and substructures in the dataset used, which it would have been very difficult or impossible to detect using more traditional methods of analysis. In particular the Macro Dynamic Visualization revealed unexpected substructures.

Commenting on the actual practical and statistical significance of the discovered data structures is beyond the scope of the paper. No statistical nor conclusive analysis is aimed at with the system. The output is specification of phenomena, that may warrant follow-up of proper statistical analysis.

The 3DVDM facilities for VDM seems promising, and encourage further developments of and experiments with the current and new tools in the context of various practical applications of VDM.

Acknowledgments

We gratefully acknowledge the support to the 3DVDM project from the Danish Research Councils, grant no. 9900103. We also acknowledge Jock A. Blackard and Colorado State University with thanks for making the Forest Cover database available.

References

1. Cruz-Neira, C., Sandin, D., DeFanti, T., Kenyon, R., Hart, J.: The cave: Audio visual experience automatic virtual environment. Communications of the ACM **35** (1992) 65–72
2. Nelson, L., Cook, D., Cruz-Neira, C.: Xgobi vs the c2: Results of an experiment comparing data visualization in a 3-d immersive virtual reality environment with a 2-d workstation display. Computational Statistics: Special Issue on Interactive Graphical Data Analysis **14** (1999) 39–51

3. Nagel, H.R., Granum, E., Musaeus, P.: Methods for visual mining of data in virtual reality. In: Proceedings of the International Workshop on Visual Data Mining, in conjunction with ECML/PKDD2001, Freiburg, Germany, 2nd European Conference on Machine Learning and 5th European Conference on Principles and Practice of Knowledge Discovery in Databases (2001) 13–28
4. Symanzik, J., Cook, D., Kohlmeyer, B.D., Lechner, U., Cruz-Neira, C.: Dynamic statistical graphics in the c2 virtual environment. Volume 29., Pasadena, California, USA, Second World Conference of the International Association for Statistical Computing (1997) 35–40
5. Wegman, E.J., Symanzik, J.: Immersive projection technology for visual data mining. Journal of Computational and Graphical Statistics **11** (2002) 163–188
6. Carr, D.B., Nicholson, W.L.: Evaluation of graphical techniques for data in dimensions 3 to 5: Scatterplot matrix, glyph, and stereo examples. In: Proceedings of the Section on Statistical Computing, Alexandria, VA, American Statistical Association (1985) 229–235
7. Pickett, R.M., Grinstein, G.: Iconographic displays for visualizing multidimensional data. In: Proceedings of the IEEE Conference on Systems, Beijing and Shenyang, People's Republic of China, Man and Cybernetics (1988) 514–519
8. Ribarsky, W., Ayers, E., Eble, J., Mukherjea, S.: Glyphmaker: Creating customized visualizations of complex data. IEEE Computer (1994) 57–64
9. Chernoff, H.: The use of faces to represent points in k-dimensional space graphically. Journal of the American Statistical Association **68** (1973) 361–368
10. Healey, C.G., Enns, J.T.: Large datasets at a glance: Combining textures and colors in scientific visualization. IEEE Transactions on Visualization and Computer Graphics **5** (1999) 145–167
11. Ebert, D.S., Rohrer, R.M., Shaw, C.D., Panda, P., Kukla, J.M., Roberts, D.A.: Procedural shape generation for multi-dimensional data visualization. In Gröller, E., Löffelmann, H., Ribarsky, W., eds.: Data Visualization '99. Springer-Verlag Wien (1999) 3–12
12. Nagel, H.R., Granum, E.: Vr++ and its application for interactive and dynamic visualization of data in virtual reality. In: Proceedings of the Eleventh Danish Conference on Pattern Recognition and Image Analysis, Copenhagen, Denmark (2002)
13. Nagel, H.R., Granum, E.: A software system for temporal data visualization in virtual reality. In: Proceedings of the Workshop on Data Visualization for large data sets and Data Mining, Augsburg, Germany, Department of Computer Oriented Statistics and Data Analysis University of Augsburg (2002)
14. Granum, E., Musaeus, P.: Constructing virtual environments for visual explorers. In Quotrup, L., ed.: Virtual Space: The Spatiality of Virtual Inhabited 3D Worlds. Springer-Verlag (2002)
15. Bovbjerg, S., Granum, E., Nagel, H.R., Vittrup, M.: Using dynamic soundscapes to support visual data mining in vr. In: Proceedings of the International Workshop on Visual Data Mining, in conjunction with The Third IEEE International Conference on Data Mining, Melbourne, Florida, USA (2003)
16. Blackard, J.A.: Comparison of Neural Networks and Discriminant Analysis in Predicting Forest Cover Types. PhD dissertation, Department of Forest Sciences. Colorado State University. Fort Collins, Colorado (1998)

Visual Mining of Cluster Hierarchies

Hans-Peter Kriegel, Stefan Brecheisen, Eshref Januzaj, Peer Kröger, Martin Pfeifle
University of Munich, Institute for Computer Science
http://www.dbs.informatik.uni-muenchen.de
{kriegel, brecheia, januzaj,kroegerp, pfeifle}@dbs.informatik.uni-muenchen.de

Abstract. Similarity search in database systems is becoming an increasingly important task in modern application domains such as multimedia, molecular biology, medical imaging, computer aided engineering, marketing and purchasing assistance as well as many others. In this paper, we show how visualizing the hierarchical clustering structure of a database of objects can aid the user in his time consuming task to find similar objects. We present related work and explain its shortcomings which led to the development of our new methods. Based on reachability plots, we introduce approaches which automatically extract the significant clusters in a hierarchical cluster representation along with suitable cluster representatives. These techniques can be used as a basis for visual data mining. We implemented our algorithms resulting in an industrial prototype which we used for the experimental evaluation. This evaluation is based on real world test data sets and points out that our new approaches to automatic cluster recognition and extraction of cluster representatives create meaningful and useful results in comparatively short time.

1 Introduction

In the last ten years, an increasing number of database applications has emerged for which efficient and effective support for similarity search is substantial. The importance of similarity search grows in application areas such as multi-media, medical imaging, molecular biology, computer aided engineering, marketing and purchasing assistance, etc. [10], [1],[8],[9],[2],[5],[6],[11].

Particularly, the task of finding similar shapes in 2-D and 3-D becomes more and more important. Examples for new applications that require the retrieval of similar 3D objects include databases for molecular biology, medical imaging and computer aided design.

Hierarchical clustering was shown to be effective for evaluating similarity models [12],[13]. Especially, the reachability plot generated by *OPTICS* [4] is suitable for assessing the quality of a similarity model. Furthermore, visually analyzing cluster hierarchies helps the user, e.g. an engineer, to find and group similar objects. Solid cluster extraction and meaningful cluster representatives form the foundation for providing the user with significant and quick information.

In this paper, we introduce algorithms for automatically detecting hierarchical clusters along with their corresponding representatives. In order to evaluate our ideas, we developed a prototype called *BOSS* (*B*rowsing *O*PTICS-Plots for *S*imilarity *S*earch). BOSS is

based on techniques related to *visual data mining*. It helps to visually analyze cluster hierarchies by providing meaningful cluster representatives.

The remainder of the paper is organized as follows: After briefly introducing reachability plots, we present in Section 2 the application areas of hierarchical clustering along with the corresponding requirements in the industrial and in the scientific community which motivated the development of BOSS. In Sections 3 and 4, we introduce the notions of cluster recognition and cluster representatives respectively, which form the theoretical foundations of BOSS. In Section 5, we describe the actual industrial prototype we developed and evaluate its usefulness in Section 6. The paper concludes in Section 7 with a short summary and a few remarks on future work.

2 Hierarchical Clustering

In this section, we outline the application ranges which led to the development of our interactive browsing tool, called BOSS. In order to understand the connection between BOSS and the application requirements we first introduce the reachability plots computed by OPTICS, which served as a starting point for BOSS. The technical aspects related to BOSS are described later in Section 5.

2.1 Reachability Plots

The key idea of density-based clustering is that for each object of a cluster the neighborhood of a given radius ε has to contain at least a minimum number *MinPts* of objects. Using the density-based hierarchical clustering algorithm OPTICS yields several advantages due to the following reasons.

- OPTICS is -in contrast to most other algorithms- relatively insensitive to its two input parameters, ε and *MinPts*. The authors in [4] state that the input parameters just have to be large enough to produce good results.
- OPTICS is a hierarchical clustering method which yields more information about the cluster structure than a method that computes a flat partitioning of the data (e.g. *k*-means [15]).
- There exist a very efficient variant of the OPTICS algorithm which is based on data bubbles [7], where we have to trade only very little quality of the clustering result for a great increase in performance.
- There exist an efficient incremental version of the OPTICS algorithm [13].

The reachability plots computed by OPTICS help the user to get a meaningful and quick overview over a large data set. The output of OPTICS is a linear ordering of the database objects minimizing a binary relation called *reachability* which is in most cases equal to the minimum distance of each database object to one of its predecessors in the ordering. Instead of a dendrogram, which is the common representation of hierarchical clusterings, the resulting reachability plot is much easier to analyse. The reachability values can be plotted for each object of the cluster-ordering computed by OPTICS. Valleys in this plot indicate clusters: objects having a small reachability value are closer and thus more similar to their predecessor objects than objects having a higher reachability value.

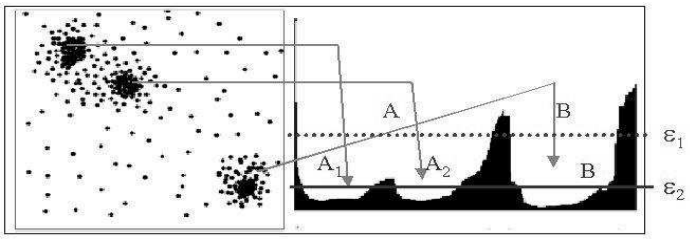


Fig.1: Reachability plots computed by optics (right) for a 2D dataset (left)

The reachability plot generated by OPTICS can be cut at any level ε_{cut} parallel to the abscissa. It represents the density-based clusters according to the density threshold ε_{cut}: A consecutive subsequence of objects having a smaller reachability value than ε_{cut} belongs to the same cluster. An example is presented in Figure 1: For a cut at the level ε_1 we find two clusters denoted as *A* and *B*. Compared to this clustering, a cut at level ε_2 would yield three clusters. The cluster *A* is split into two smaller clusters denoted by A_1 and A_2 and cluster B decreased its size. Usually, for evaluation purposes, a good value for ε_{cut} would yield as many clusters as possible.

Application Ranges . BOSS was designed for three different purposes: visual data mining, similarity search and evaluation of similarity models. For the first two applications, the choice of the representative objects of a cluster is the key step. It helps the user to get a meaningful and quick overview over a large existing data set. Furthermore, BOSS helps scientists to evaluate new similarity models.

Visual Data Mining. As defined in [3], visual data mining is a step in the KDD process that utilizes visualization as a communication channel between the computer and the user to produce novel and interpretable patterns. Based on the balance and sequence of the automatic and the interactive (visual) part of the KDD process, three classes of visual data mining can be identified.

- Visualization of the data mining result:
  An algorithm extracts patterns from the data. These patterns are visualized to make them interpretable. Based on the visualization, the user may want to return to the data mining algorithm and run it again with different input parameters (cf. Figure 2a).
- Visualization of an intermediate result:
  An algorithm performs an analysis of the data not producing the final patterns but an intermediate result which can be visualized. Then the user retrieves the interesting patterns in the visualization of the intermediate result (cf. Figure 2b).
- Visualization of the data:
  Data is visualized immediately without running a sophisticated algorithm before. Patterns are obtained by the user by exploring the visualized data (cf. Figure 2c).

The approach presented in this paper belongs to the second class. A hierarchical clustering algorithm is applied to the data, which extracts the clustering structure as an intermediate result. There is no meaning associated with the generated clusters. However, our approach allows the user to visually analyze the contents of the clusters. The clustering algorithm used in the algorithmic part is independent from an application. It performs the

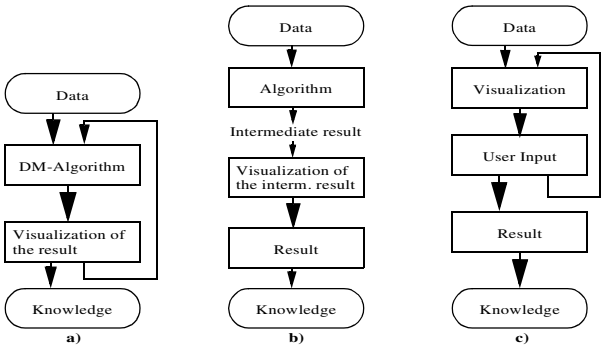


Fig.2: Different approaches to visual data mining

core part of the data mining process and its result serves as a multi-purpose basis for further analysis directed by the user. This way the user may obtain novel information which was not even known to exist in the data set. This is in contrast to similarity search where the user is restricted to find similar parts respective to a query object and a predetermined similarity measure.

Similarity Search. The development, design, manufacturing and maintenance of modern engineering products is a very expensive and complex task. Effective similarity models are required for two- and three-dimensional CAD applications to cope with rapidly growing amounts of data. Shorter product cycles and a greater diversity of models are becoming decisive competitive factors in the hard-fought automobile and aircraft market. These demands can only be met if the engineers have an overview of already existing CAD parts. It would be desirable to have an interactive data browsing tool which depicts the reachability plot computed by OPTICS in a user friendly way together with appropriate representatives of the clusters. This clear illustration would support the user in his time-consuming task to find similar parts. From the industrial user's point of view, this browsing tool should meet the following two requirements:

- The hierarchical clustering structure of the dataset is revealed at a glance. The reachability plot is an intuitive visualization of the clustering hierarchy which helps to assign each object to its corresponding cluster or to noise. Furthermore, the hierarchical representation of the clusters using the reachability plot helps the user to get a quick overview over all clusters and their relation to each other. As each entry in the reachability plot is assigned to one object, we can easily illustrate some representatives of the clusters belonging to the current density threshold ε_{cut} (cf. Figure 3) .
- The user is not only interested in the shape and the number of the clusters, but also in the specific parts building up a cluster. As for large clusters it is rather difficult to depict all objects, representatives of each cluster should be displayed. To follow up a first idea, these representatives could be simply constructed by superimposing all parts belonging to the regarded cluster (cf. Figure 4). We can browse through the hierarchy of the representatives in the same way as through the OPTICS plots.

This way, the cost of developing and producing new parts could be reduced by maximizing the reuse of existing parts, because the user can browse through the hierarchical

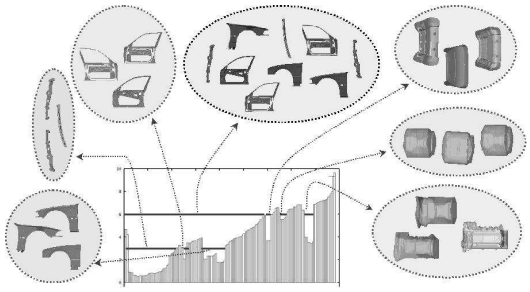


Fig.3: Browsing through reachability plots with different density thresholds ε_{cut}

structure of the clusters in a top-down way. Thus the engineers get an overview of already existing parts and are able to navigate their way through the diversity of existing variants of products, such as cars.

Evaluation of Similarity Models. In general, similarity models can be evaluated by computing k-nearest neighbor queries (k-nn queries). As shown in [14], this evaluation approach is subjective and error-prone because the quality measure of the similarity model depends on the results of a few similarity queries and, therefore, on the choice of the query objects. A model may perfectly reflect the intuitive similarity according to the chosen query objects and would be evaluated as "good" although it produces disastrous results for other query objects.

A better way to evaluate and compare several similarity models is to apply a clustering algorithm. Clustering groups a set of objects into classes where objects within one class are similar and objects of different classes are dissimilar to each other. The result can be used to evaluate which model is best suited for which kind of objects. It is more objective since each object of the data set is taken into account to evaluate the data models.

3 Cluster Recognition

In this section, we address the first task of automatically extracting clusters from the reachability plots. After a brief discussion of recent work in that area, we propose a new approach for hierarchical cluster recognition based on reachability plots.

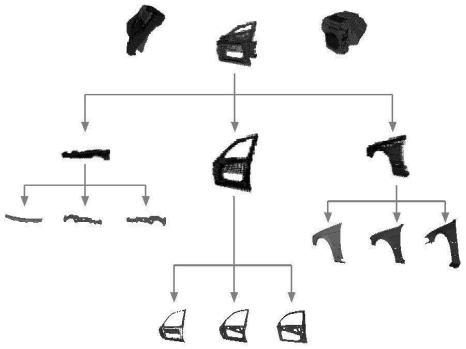


Fig.4: Hierarchically ordered representatives

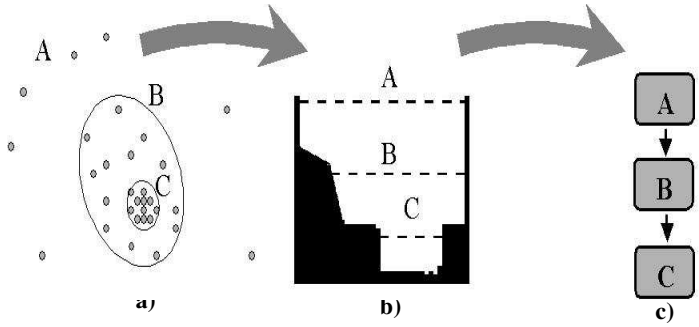


Fig.5: Sample narrowing clusters
a) data space, **b)** reachability plot and **c)** cluster hierarchy

3.1 Recent Work

To the best of our knowledge, there are only two methods for automatic cluster extraction from hierarchical representations such as reachability plots or dendrograms which are both based on reachability plots. Since clusters are represented as valleys (or dents) in the reachability plot, the task of automatic cluster extraction is to identify significant valleys.

The first approach proposed in [4] called ξ–clustering is based on the steepness of the valleys in the reachability plot. The steepness is defined by means of an input parameter ξ. The method suffers from the fact that this input parameter is difficult to understand and hard to determine. Rather small variations of the value ξ often lead to drastic changes of the resulting clustering hierarchy. As a consequence, this method is unsuitable for our purpose of automatic cluster extraction.

The second approach was proposed recently by Sander et al. [16]. The authors describe an algorithm called *cluster_tree* that automatically extracts a hierarchical clustering from a reachability plot and computes a cluster tree. It is based on the idea that *significant* local maxima in the reachability plot separate clusters. Two parameters are introduced to decide whether a local maximum is significant: The first parameter specifies the minimum cluster size, i.e. how many objects must be located between two significant local maxima. The second parameter specifies the ratio between the reachability of a significant local maximum *m* and the average reachabilities of the regions to the left and to the right of *m*. The authors in [16] propose to set the minimum cluster size to 0.5% of the data set size and the second parameter to 0.75. They empirically show, that this default setting approximately represents the requirements of a typical user.

Although the second method is rather suitable for automatic cluster extraction from reachability plots, it has one major drawback. Many real-world data sets consist of narrowing clusters, i.e. clusters consisting of exactly one smaller sub-cluster (cf. Figure 5). Since the algorithm *cluster_tree* runs through a list of all local maxima (sorted in descending order of reachability) and decides at each local maximum *m*, whether *m* is significant to split the objects to the left of *m* and to the right of *m* into two clusters, the algorithm cannot detect such narrowing clusters. These clusters cannot be split by a significant maximum. Figure 5 illustrates this fact. The narrowing cluster *A* consists of one cluster *B* which is itself narrowing consisting of one cluster *C*. The algorithm cluster_tree

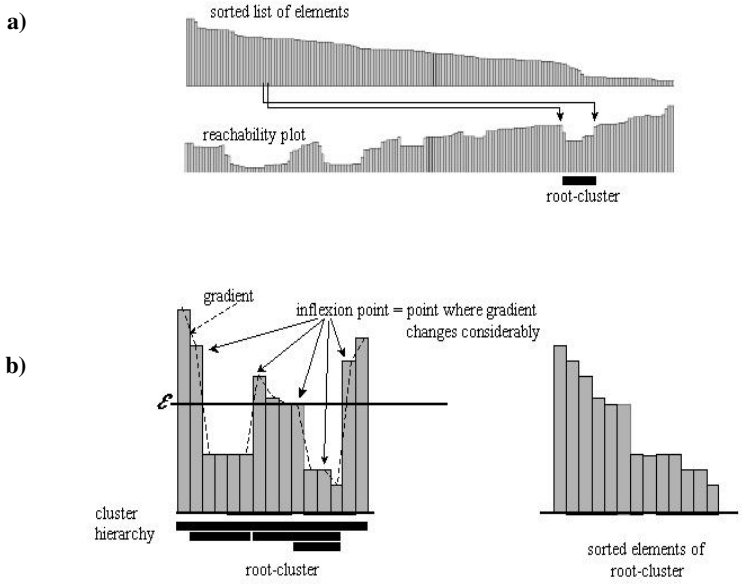


Fig.6: Drop-Down-Clustering
a) detection of root clusters, **b)** detection of subclusters

will only find cluster *A* since there are no local maxima to split clusters *B* and *C*. The ξ–clustering will detect only one of the clusters *A*, *B* or *C* dependent on the ξ–parameter but also fails to detect the cluster hierarchy.

3.2 Drop-Down Clustering

This new cluster recognition algorithm is based on the novel concept of *inflexion points* which allow the detection of narrowing subclusters. A point *o* is an inflexion point iff the gradient of the reachability values changes considerably (cf. Figure 6b).

Since our method works in a top-down fashion, we call it *Drop-Down Clustering*. The idea behind is the successive use of the visual interpretation of the cluster ordering -as described in Figure 1- which is based on the fact that the reachability plot can be cut by any level ε_{cut} to the abscissa to extract a clustering. Starting from an initial clustering we simply drop the ε_{cut}-value in order to find substructures. Since it is not practical to test each possible ε_{cut}-value, we have to extract interesting values for a cut from the reachability values of the objects.

The Drop-Down Clustering algorithm starts by generating an initial root clustering (cf. Figure 6a). This does not contain all elements of the plot, as clusters separated by noise are assumed to be not related. Basically, a set of clusters forms the basis for a set of hierarchical clusters. This initial clustering is generated as follows: The objects in the reachability plot are sorted by descending reachability distance while retaining relative order among equal elements. The sorted list is now scanned until two objects are found whose indices are more than *MinPts* apart, indicating that every element in-between these

two is smaller than either, thus constituting a dip in the graph. A top level cluster has been found, and all elements included in this cluster are removed from the sorted list. The scan can now continue until all elements have been removed or viewed.

The second part of the algorithm now separately analyzes each cluster found during the initial clustering (cf. Figure 6b). The extraction of further (sub-)clusters is a recursive procedure. The procedure starts with a set of elements from the reachability graph which is sorted by descending reachability values where elements having the same reachability value are arranged according to the cluster ordering. This list is sequentially tested for an object which is an inflexion point. An inflexion point can either indicate the start or end of a narrowing subcluster, or be responsible for two new subclusters.

Should a subcluster be found, it may be added to the resulting cluster hierarchy, after which it is then recursively processed to discover potential substructures. All discovered subclusters must conform to the following constraints:

- The minimum cluster size constraint of *MinPts* objects must be satisfied, i.e. there are at least *MinPts* objects located between the start point and the end point of the cluster (e.g. Cluster *B* in Figure 5 must contain at least *MinPts* objects).
- The current cluster has at least *MinPts* objects less than the cluster of its parent node in the hierarchy (e.g. Cluster *B* in Figure 5 must have at least *MinPts* objects less than cluster *A*).

Let us note, that we could also claim a minimum ratio of reachabilities at the boundary of a cluster and inside a cluster as postulated in [16] or increment/decrement the required minimum cluster size.

Obviously, the Drop-Down algorithm is able to extract narrowing clusters. Experimental comparisons with the methods in [16] and [4] are presented in Section 6.

4 Cluster Representatives

In this section, we present different approaches to determine representatives for clusters computed by OPTICS. A simple approach could be to superimpose all objects of a cluster to build the representative as it is depicted in Figure 4. However, this approach has the huge drawback that the representatives on a higher level of the cluster hierarchy become rather unclear. Therefore, we choose real objects of the data set as cluster representatives.

In the following, we assume that *DB* is a database of multimedia objects, $dist$: $DB \times DB \rightarrow IR$ is a metric distance function on objects in *DB* and $N_\varepsilon(o) := \{q \in DB \mid dist(o,q) \leq \varepsilon\}$ where $o \in DB$ and $\varepsilon \in IR$. A cluster $C \subseteq DB$ is represented by a set of k objects of the cluster, denoted as $REP(C)$. The number of representatives k can be a user defined number or a number which depends on the size and data distribution of the cluster C.

4.1 The Extended Medoid Approach

Many partitioning clustering algorithms are known to use medoids as cluster representatives. The medoid of a cluster C is the closest object to the mean of all objects in C.

The mean of C is also called centroid. For $k > 1$ we could choose the k closest objects to the centroid of C as representatives.

The choice of medoids as cluster representative is somehow questionable. Obviously, if C is not of convex shape, the medoid is not really meaningful.

An extension of this approach coping with the problems of clusters with non-convex shape is the computation of k medoids by applying a k-medoid clustering algorithm to the objects in C. The clustering using a k-medoid algorithm is rather efficient due to the expectation that the clusters are much smaller than the whole data set. This approach can also be easily extended to cluster hierarchies. At any level we can apply the k-medoid clustering algorithm to the merged set of objects from the child clusters or -due to performance reasons- merge the medoids of child clusters and apply k-medoid clustering on this merged set of medoids.

4.2 The Minimum Core-Distance Approach

The second approach to choose representative objects of hierarchical clusters uses the density-based clustering notion of OPTICS. To compute the reachability, OPTICS determines for each object the so called *core-distance*:

Definition 1 (*Core-distance*).

Let o be an object from a database DB, let ε be a distance value, let $N_\varepsilon(o)$ be the ε-neighborhood of o, let *MinPts* be a natural number and let *MinPts-distance*(o) be the distance from o to its *MinPts*-nearest neighbor. Then, the *core-distance* of o is defined as:

$$\textit{core-distance}(o) = \begin{cases} \infty, \textit{ if } |N_\varepsilon(o)| < \textit{MinPts} \\ \textit{MinPts-distance}(o), \textit{ otherwise} \end{cases}$$

The core-distance of an object indicates the density of the surrounding region. The smaller the core-distance of an object o, the denser the region surrounding o. This observation led us to the choice of the object having the minimum core-distance as representative of the respective cluster. Formally, $REP(C)$ can be computed as:

$REP(C) := \{o \in C \mid \forall x \in C : \textit{core-distance}(o) \leq \textit{core-distance}(x)\}$.

We choose the k objects with the minimum core-distances of the cluster as representatives.

The straightforward extension for cluster hierarchies is to choose the k objects from the merged child clusters having the minimum core-distances.

4.3 The Maximum Successor Approach

Based on the core-distance, the reachability-distance (or short: reachability) is defined as:

Definition 2 (*Reachability-Distance*).

Let $o \in DB$, let $N_\varepsilon(o)$ be the ε-neighborhood of o, let *MinPts* be a natural number. Then, the *reachability-distance* of $p \in DB$ with respect to o is defined as:

reachability-distance $(p,o) = \max$ (*core-distance*(o), *distance*(p,o))

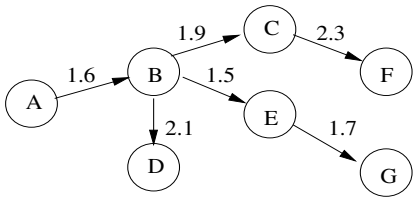


Fig.7: Sample successor graph for a cluster of seven objects

The result of OPTICS is an ordering of the database minimizing the reachability relation. At each step of the ordering, the object p having the minimum reachability wrt. the already processed objects occurring before p in the ordering is chosen. Thus, if the reachability of object p is not ∞, it is determined by *reachability-distance*(p,o) where o is an object located before p in the cluster ordering. We call o the *predecessor* of p.

Definition 3 (*Successors*).
Let $o \in DB$. Then, the set of successors is defined as $S(o) := \{s \in DB \mid o \text{ is predecessor of } s\}$.

Let us note, that objects may have no predecessor, e.g. each object having a reachability of ∞ does not have a predecessor, including the first object in the ordering. On the other hand, some objects may have more than one successor. In that case, some other objects have no successors.

We can model this successor-relationship within each cluster as a directed *successor graph* where the nodes are the objects of one cluster and a directed edge from object o to s represents the relationship $s \in S(o)$. Each edge (x,y) can further be labeled by *reachability-distance* (x,y). A sample successor graph is illustrated in Figure 7.

For the purpose of computing representatives of a cluster, the objects having many successors are interesting. Roughly speaking, these objects are responsible for the most density-connections within a cluster. The reachability values of these "connections" further indicate the distance between the objects.

Our third strategy selects the representatives of clusters by maximizing the number of successors and minimizing the according reachabilities. For this purpose, we compute for each object o of a cluster C, the *S*um of the *I*nverse *R*eachability distances of the successors of o within C, denoted by $SIR_C(o)$:

$$SIR_C(o) = \begin{cases} 0, & \text{if } S(o) = \varnothing \\ \sum_{\substack{s \in S(o) \\ s \in C}} \frac{1}{1 + reachability - distance(s, o)}, & \text{otherwise} \end{cases}$$

We add 1 to *reachability-distance* (s,o) in the denominator to weight the impact of the number of successors over the significance of the reachability values. Based on $SIR(o)$, the representatives can be computed as follows:

$REP(C) := \{o \in C \mid \forall x \in C : SIR_C(o) \geq SIR_C(x)\}$.

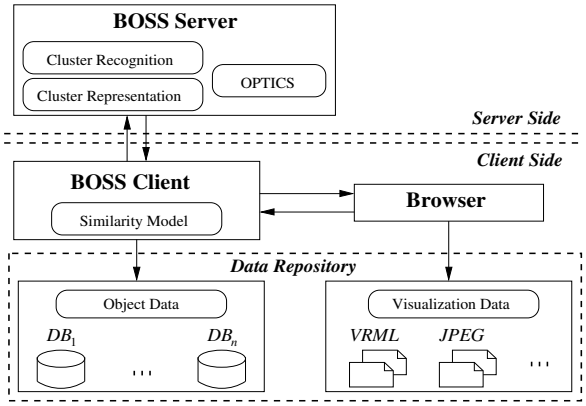


Fig.8: BOSS distributed architecture

If we want to select k representatives for C we simply have to choose the k objects with the maximum SIR_C values.

5 System Architecture

The development of the industrial prototype BOSS is a first step towards developing a comprehensive, scalable and distributed computing solution designed to make the efficiency of OPTICS and the analytical capabilities of BOSS available to a broader audience. BOSS is a client/server system allowing users to provide their own data locally, along with an appropriate similarity model (cf. Figure 8).

The data provided by the user will be comprised of the objects to be clustered, as well as a data set to visualize these objects, e.g. VRML files for CAD data (cf. Figure 9) or JPEG images for multi-media data. Since this data resides on the user's local computer and is not transmitted to the server heavy network traffic can be avoided. In order for

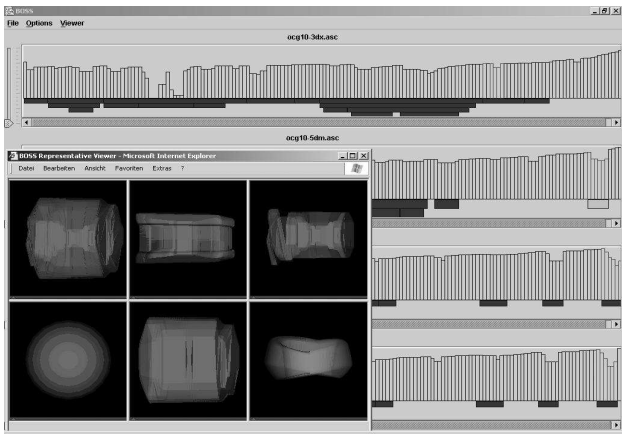


Fig.9: BOSS screenshot

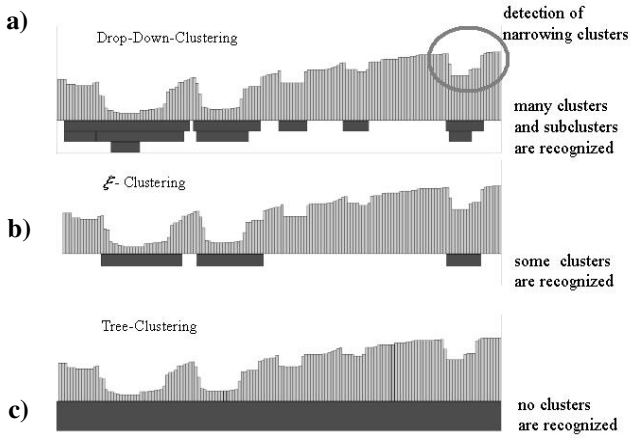


Fig.10: Sample Clustering of CAR Parts
a) *Drop-Down-Clustering*, **b)** ξ-*Clustering* and **c)** *Tree-Clustering*

BOSS to be able to interpret this data, the user must supply his own similarity model with which the reachability data can be calculated.

The independence of the data processing and the data specification enables maximum flexibility. Further flexibility is introduced through the support of external visual representation. As long as the user is capable of displaying the visualization data in a browser, e.g. by means of a suitable plug-in, the browser will then load web pages generated by BOSS displaying the appropriate data. Thus, multimedia data such as images or VRML files can easily be displayed (cf. Figure 9). By externalizing the visualization procedure we can resort to approved software components, which have been specifically developed for displaying objects which are of the same type as the objects within our clusters.

6 Evaluation

We evaluated both the effectiveness and efficiency of our approaches using two real-world test data sets. The first one contains approximately 200 CAD objects from a German car manufacturer, the second one 5000 CAD objects from an American aircraft producer. We tested on a workstation with a 1.7 GHz CPU and 2 GB RAM.

In the following, three cluster recognition algorithms will vie among themselves, after which the three approaches for generating representatives will be evaluated.

Cluster Recognition. Automatic cluster recognition is clearly very desirable when analyzing large sets of data. In this case, we will be looking at a subset of a database of CAD objects representing car parts. The results are depicted in Figure 10.

This data exhibits the commonly seen quality of unpronounced but nevertheless to the observer clearly visible clusters. The *Tree-Clustering* algorithm does not find any clusters at all, whereas the ξ-clustering approach successfully recognizes some clusters while missing out on significant subclusters. On the other hand, our new Drop-Down-Algorithm detects many clusters. Furthermore, it detects a lot of meaningful cluster hierar-

	CAR (200 parts)	PLANE (5000 parts)
x-clustering	0.348 s	9.714 s
cluster_tree	0.130 s	3.019 s
Drop-Down clustering	0.104 s	0.707 s

Fig.11: CPU time for cluster recognition

chies, consisting of narrowing subclusters. To sum up, in all our tests the Drop-Down-algorithm detected much more clusters than the other two approaches, without producing any redundant and unnecessary cluster information.

The overall runtime for the three different cluster recognition algorithms is depicted in Figure 11. Our new Drop-Down clustering algorithm does not only produce the most meaningful results, but also in the shortest time. It seems to be the only algorithm which is suitable for interactive use, if the data sets contain several thousand elements.

Cluster Representation . After a cluster recognition algorithm has analyzed the data, algorithms for cluster representation can help to get a quick visual overview of the data. With the help of representatives, large sets of objects may be characterized through a single object of the data set. We extract a sample cluster from the plot depicted in Figure 10a in order to evaluate the different approaches for cluster representatives. In our first tests, we set the number of representatives k to 1.

The objects of one cluster are displayed in Figure 12. The three annotated objects are the representatives computed by the respective algorithms. Both the *Maximum Successor* and the *Minimum Core Distance* approaches give good results. Despite the slight inhomogeneity of the cluster, both representatives sum up the majority of elements within this cluster. This cannot be said of the representative computed by the commonly used medoid method, which selects an object from the trailing end of the cluster.

Summary. The results of our experiments show, that our new approaches for the automatic cluster extraction and for the determination of representative objects outperform

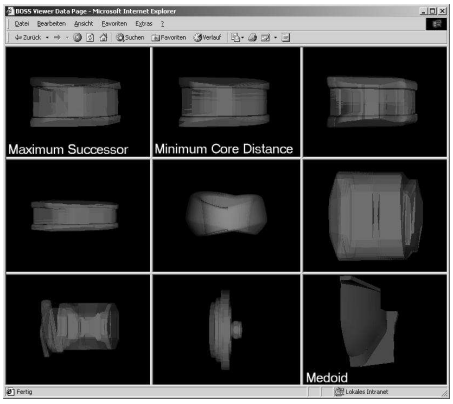


Fig.12: Representatives displayed by the BOSS object viewer

existing methods. It theoretically and empirically turned out, that our Drop-Down-Clustering algorithm seems to be more practical than recent work for automatic cluster extraction from hierarchical cluster representations. We also empirically showed that our approaches for the determination of cluster representatives is most likely more suitable than the simple (extended) medoid approach.

7 Conclusions

In this paper, we proposed hierarchical clustering combined with automatic cluster recognition and selection of representatives as a promising visualization technique. Its areas of application include visual data mining, similarity search and evaluation of similarity models. We surveyed three approaches for automatic extraction of clusters. The first method, ξ-*clustering*, fails to detect some clusters present in the clustering structure and suffers from the sensitivity concerning the choice of its input parameter. The algorithm *cluster_tree* is obviously unsuitable in the presence of narrowing clusters. To overcome these shortcomings, we proposed a new method, called *Drop-Down-Clustering*. The experimental evaluation showed that this algorithm is able to extract narrowing clusters. The cluster hierarchies produced by the Drop-Down-Algorithm are similar to the clustering structures which an experienced user would manually extract.

Furthermore, we presented three different approaches to determine representative objects for clusters. The commonly known medoid approach is shown to be unsuitable for real-world data, while the approaches minimizing the core-distance and maximizing the successors both deliver good results.

Finally, we described our industrial prototype, called BOSS, that implements the algorithms presented in this paper.

In our future work we will concentrate on efficient algorithms for incremental hierarchical clustering. These new cluster algorithms should allow an easy determination of the cluster hierarchy together with an easy extraction of meaningful representatives.

References

1. Agrawal R., Faloutsos C., Swami A. *Efficient Similarity Search in Sequence Databases*. Proc. 4th. Int. Conf. on Foundations of Data Organization and Algorithms (FODO .93), Evanston, ILL, volume 730 of Lecture Notes in Computer Science (LNCS), pages 69-84. Springer, 1993.
2. Agrawal R., Lin K.-I., Sawhney H., Shim K.: *Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases*. Proc. 21th Int. Conf. on Very Large Databases (VLDB 95), pages 490-501, 1995.
3. Ankerst M.: *Visual Data Mining.* PhD thesis, Institute for Computer Science, University of Munich, 2000.
4. Ankerst M., Breunig M. M., Kriegel H.-P., Sander J.: "*OPTICS: Ordering Points To Identify the Clustering Structure*", Proc. Int. Conf. on Management of Data (SIGMOD), Philadelphia, PA, 1999, pp. 49-60.
5. Berchtold S., Keim D. A., Kriegel H.-P.: *Using Extended Feature Objects for Partial Similarity Retrieval.* VLDB Journal, 6(4):333-348, 1997.

6. Berchtold S., Kriegel H.-P.: *S3: Similarity Search in CAD Database Systems*. Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD 97), Tucson, AZ, pages 564-567, 1997.
7. Breuning M., Kriegel H.-P., Kröger P., Sander J.: *Data Bubbles: Quality Preserving Performance Boosting.* Proc. ACM SIGMOD 2001 Int. Conf. on Management of Data, Santa Barbara, CA, 2001
8. Faloutsos C., Equitz M., Flickner M., Niblack W., Petkovic D., Barber R.: *Efficient and Effective Querying by Image Content*. Journal of Intelligent Information Systems, 3:231-262, 1994.
9. Faloutsos C., Ranganathan M., Manolopoulos Y.: *Fast Subsequence Matching in Time-Series Databases*. Proc. ACM SIGMOD Int. Conf. on Management of Data, Minneapolis, MN, pages 419-429, 1994.
10. Jagadish H. V.: *A Retrieval Technique for Similar Shapes*. Proc. ACM SIGMOD Int. Conf. on Management of Data, 208-217,1991.
11. Keim D.: *Efficient Geometry-based Similarity Search of 3D Spatial Databases*.Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD 99), Philadelphia, PA, pages 419-430, 1999.
12. Kriegel H.-P., Brecheisen S., Kröger P. , Pfeifle M., Schubert M.: *Using Sets of Feature Vectors for Similarity Search on Voxelized CAD Objects*. Proc. ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'03), San Diego, CA, 2003.
13. Kriegel H.-P., Kröger P., Gotlibovich I.: *Incremental OPTICS: Efficient Computation of Updates in a Hierarchical Cluster Ordering*. Proc. 5th Int. Conf. on Data Warehousing and Knowledge Discovery (DaWaK'03), Prague, Czech Republic, 2003.
14. Kriegel H.-P. , Kröger P., Mashael Z., Pfeifle M., Pötke M., Seidl T.: *Effective Similarity Search on Voxelized CAD Objects*. Proc. 8th Int. Conf. on Database Systems for Advanced Applications (DASFAA'03), Kyoto, Japan, 2003.
15. McQueen J.: *Some Methods for Classification and Analysis of Multivariate Observations*. 5th Berkeley Symp.Math. Statist. Prob., volume 1, pages 281–297, 1967.
16. Sander J., Qin X., Lu Z., Niu N., Kovarsky A.: *Automatic Extraction of Clusters from Hierarchical Clustering Representations*. Proc. 7th Pacic-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2003), Seoul, Korea, 2003.

Using Dynamic Soundscapes to Support Visual Data Mining in VR

Søren Bovbjerg, Erik Granum, Henrik R. Nagel, and Michael Vittrup

Computer Vision and Media Technology Lab.
Aalborg University
Denmark
{sb, eg, hrn, vittrup}@cvmt.dk

Abstract. Data Mining in Virtual Reality (VR) has, for good reasons, been focused on facilitating visual inspection and analysis. As the amount and complexity of data is overwhelming, it is worthwhile to consider a further exploration of the human perceptual faculties, and it seems natural to consider sound as a possible perceptual cue. Current technology enables us to create advanced real-time 3D soundscapes, which may prove useful since the human ears' field of hearing is larger than the eyes' field of view, and thus is able to inform us on events happening in areas that we do no see at a given time. This paper presents methods for generating 3D soundscapes from statistical information. Through a series of investigations, we present and discuss the effectiveness of these methods in situations where sound acts as support for visual cues, as well as the use of sound as a separate cue for analyzing data in VR.

1 Introduction

The 3D Visual Data Mining system (3DVDM) presented by Nagel et al. [1] and [2] provides a flexible system for doing explorative Visual Data Mining (VDM) in VR using super computers. The optimal VR system for this purpose (such as the six-sided CAVE[1]) provides technology allowing us to create an immersive environment of surrounding 3D visual cues, as well as the possibility of creating immersive 3D soundscapes.

The 3DVDM task is perceptual representation of statistical data and the reference is a virtual visual world, where statistical observations are represented in a 3D scatter plot. Each data point in the plot is shown as a visual object, such that statistical variables may also control various visual object properties like object shape, orientation, color, and surface texture. The aim is to present data from within and to allow the human observer as a visual explorer to navigate around in the visual world, in order to inspect data in arbitrary view directions and from arbitrary viewpoints. To supplement what ordinary statistical analysis of data can provide, the observer will look for special structures in data. He may search systematically and/or hope for the talent of serendipity [3] to help

[1] http://www.ncsa.uiuc.edu/VR/VR/VRHomePage.html

detect the interesting and unexpected. The technological challenge is to provide facilities as convenient as possible for such search to be successful. Part of this challenge is to enable us to represent simultaneously as many statistical variables as possible to exploit the perceptual bandwidth, while avoiding perceptual "overflow".

When we choose to visualize large amount of data in a 3D Cartesian coordinate system we create an artificial world that is an artificial world in more than one sense: It is "an artificially created artificial world". The user will find himself floating in infinite space surrounded by the graphical objects representing entries in a database, and may loose sense of orientation.

1.1 Motivation

The intentions of using sound in data mining is derived from the underlying idea of the 3DVDM project, which is to encode as much information as the human perceptual system can cope with [3]. The idea is to create a 3D sound interpretation of the data that can serve either as a support for a visualization parameter such as color or as an independent audio representation of one or more data dimensions. A main motivation for this approach is that the human ears' field of hearing is larger than the eyes' field of view, and thus is able to inform us on events happening in areas that we do not see at a given time. It may then be possible to create sound cues that will draw the listener's attention towards part of the visualization that is out of visual range. In VR, the user will then be able navigate using sound cues and perform a more detailed investigation of areas of interest. Likewise a soundscape may help the user to maintain a sense of orientation while navigating in arbitrary directions.

Our aim is to create a soundscape that in some way represents the surrounding visual world, and may be of assistance to the user in several ways. This paper will describe some examples of how sound can be helpful for performing VDM, and in addition support orientation and navigation in the artificial worlds that we deal with.

The examples in this paper are created using a "Forest Cover Type" database, see table 1. The database is available with description online[2].

1.2 Sonification of Statistical Values

The methods described in this paper only aim at encoding one level of statistical information into the soundscape. It is possible to add more levels using e.g. different types of instruments/timbres or different persons speaking, much like using different shapes, orientations and textures on graphical objects. It is important to note that the target is dependent on the success of the user *comprehension* of a multivariate soundscape. Thus it is not necessarily a goal it self to sonificate as many variables as possible simultaneously, but to make it possible for analyst

[2] http://kdd.ics.uci.edu/databases/covertype/covertype.data.html

and domain experts to get a useful impression of relationships between multiple variables.

Listening to a soundscape rendering based on statistical data can be an overwhelming sensation for beginners, and it may be difficult to extract just one level of information. It takes some time to learn how to interpret the information, and before this is mastered the soundscape should be kept simple. If the possibility of encoding more level is made available, it should be up to the user to add more levels to the soundscape, as he/she gets used the sensation. Still, adding more levels may just deteriorate the the information held in the existing level(s).

2 Previous Work

It is well known that color provides a powerful tool for categorizing visual objects, e.g. when performing visual data mining[1]. Some research have been focused on using auditory patterns to present complex information and combining visual and auditory information to facilitate the processing of information [4]. It is concluded that for some types of information, the auditory modality works just as well as the visual modality. In particular, auditory information can be used as effectively as visual information for a visual search task when speed is not crucial. Furthermore it is suggested that humans can rapidly extract more than one aspect of information from a sound, and then act on the information.

These observations indirectly suggest that it will be possible to create a useful 3D soundscape, that in some way represent a visual world and yields further information about this.

Much research has been focused on creating auditory displays using different artificial cues for simulating direction and distance. [5] provides a useful overview of things to consider when implementing a 3D sound system, especially regarding distance perception and implementation. Also [6] is a useful source when dealing with the human auditory system.

Some specific research regarding data analysis has previously been done [7]. This article describes different aspects and methods for rendering data as sound and provides a good overview of different possibilities within this scope.

3 Rendering Value as Sound in Virtual Spaces

The 3DVDM system provides tools for using 3D sound to perform general and detailed investigation of data visualized in a 3D scatter plot. A sound engine is designed and integrated into the VR++ system (VR++ is the software framework, which hosts the 3DVDM software [8]). With this, it is possible to place sampled sounds or synthesized sounds at any point in the virtual world. It is thus possible to attach sounds representing statistical values to selected objects, or add sound representation to groups of objects, such as density clusters or measurements of density in solid angles around the users current position. Adding sound to this infinite visual space with thousands of generators and with no major solid structures to influence with reverberation, as in the natural world, has

appeared to be a rather complicated task with a large number of possibilities and constraints, respectively.

Databases of concern typically have many thousands of observations, and our current software can handle 32 simultaneous sound generators (voices). Even if this may seem insufficient for creating an interpretation of thousands of observations, we need to consider that a soundscape created from thousands of simultaneous sounds will not necessarily yield useful information (it is more likely that it will not). One possible solution to this problem is to create a soundscape that changes in time (and space), allowing the listener to be able to pinpoint locations in space, which may be interesting to explore even further. For this purpose, the number of voices is sufficient. The change in time can either be an automated process, or controlled by the user in ways that will be described in more detail later.

The limitation of 32 voices is not a finite number, but can easily be expanded to whatever number of voices the host computer that runs the sound engine is capable of.

The sound system can produce an audible 3D sound field generated with 2, 4, 5.1 (Dolby Digital positions) and 8 speakers in different configurations. The optimal 3D sound field is generated with 8 speakers placed in each corner of a cube, because this creates an even placement of the speakers in all directions of the listener, and represents all three directions in the 3D world. The 8 speaker configuration is widely used with the CAVEs.

4 Sound Schemes

The human auditory system is capable of pinpointing and maintaining focus of perception on a single sound source in an otherwise complicated soundscape. This is referred to as the "cocktail party effect"[6]. This means that we can encode much information into the soundscape and expect the listener to be able to decode a single source, as long as it is distinguishable from the rest of the sound sources. This may be achieved by dynamicly changing the soundscape - a side effect that will happen automaticly if the data base samples are selected as a function of time (only requiring that the database contains different observations).

Likewise [6] speaks of the "ventriloquism effect". This phenomenon indicates that, if there is a visual cue and a sound is located close to this object, the sound is automatically perceived as being where the visual object is. This means that if we visualize the currently sampled objects, the listener will perceive the sound as originating from this object. This may compensate for possible shortcomings in the 3D sound rendering system.

When placing sound cues at different distances one should consider that the human auditory system perceives distance far more accurate when using familiar sounds rather than unfamiliar sounds. [9].

Based on this we propose two ways of mapping statistical values to sound events: Samples and synthesized sound.

– Using sampled sounds where an exchangeable sound bank consists of selected samples that may (or may not) be related in a way that makes them yield some kind of numerical information. The samples should be short (i.e. less than 500ms) to avoid a clouded soundscape. The sounds used in the soundscape should (ideally) represent some kind of numerical information making the listener able to distinguish between these. It is clear that rendering a few categorical (Nominal or Ordinal) values such as the four *Wilderness Areas* or the seven *Cover Types* in the Forest Cover Type database (see table 1) will create a more interpretable soundscape than *9am Hill-shade* which ranges from 0 to 255. With a few n categorical values it is then suitable to use a sound bank with a few easy distinguishable sounds, e.g. recordings of n different musical instruments or a spoken voice saying the numbers from 1 to n.
– Using synthesized sounds generated with a simple sawtooth waveform with musically tuned pitch. Low value can correspond to a low pitch and visa versa. When using the synthesized scales the soundscape will most likely have some kind of musical content. Since we have chosen pitch as value it is our hope that listeners with a minimum of musical training will be able to hear and interpret the steps in pitch as values. The sounds are created with a relatively short envelope time consisting of a short attack around 30ms and decay around 100ms. The resulting sounds are short musical sounds with a noticeable onset, that should create sufficient attention for the listener in order to locate it's position and perceive its encoded value. In the non-categorical case it may be more difficult to define the ideal sound sample bank. For this purpose it may be more suitable to use these synthesized sounds and achieve value information by mapping the values to musical pitch. The pitches are chosen in predefined scales, which have different spacing between the pitches and different tonal content. The software has a few of these scales to which values can be mapped. The scales are: Pentatonic, Aeolic, Major Chord, Chromatic, Melodic (Ionic) and Tritone (Boolean). In some scales the tones are close to each other (Melodic/Chromatic) while the others have more or less spread (3-5 semi-tones between them). The Tritone scale only has two tones placed in the distance of six semi-tones. All scales are placed musically in the mid range (not too low, nor too high notes).

Our wish to encode easy understandable numerical information into the soundscape introduces a problem that we need to consider. Under normal conditions the auditory system is exposed to many *different* sound sources. In our case the soundscape will be created from many *similar* sources. This means that events are more sensible to masking, which means that sounds with higher intensity "eats" the weaker ones. We may overcome this to some degree by choosing many different sounds or by lowering the number of simultaneous sounds.

In practice the software has control options to adjust such parameters. These settings can be applied in real-time depending on what the listener may find useful in the specific situation.

4.1 Tools for Rendering a Soundscape

The 3DVDM sound tools consist of two major methods which both operate on visual 3D scatter plots:

- A render tool, which can be used for rendering a 3D soundscape around the user representing either statistical values or density measured in a user definable cubic grid. The soundscape rendering can be of the whole data set with random sampling, sliding windowed data sampling, or rendering a sweeping plane along a specified axis in time, where sound is triggered when the plane intersects with objects. The database is sampled several times each predefined time interval, e.g. each 100ms, and if the threshold conditions for a sample are met a sound will play. Each sampling period a specified number of voices (1-32) will be triggered on this basis.
- A torch tool, which is placed in the hand of the user. In VR, the user can point this virtual torch in arbitrary directions towards objects in space and get sound responses from these representing statistical values of objects or local area densities (see Figure 1).

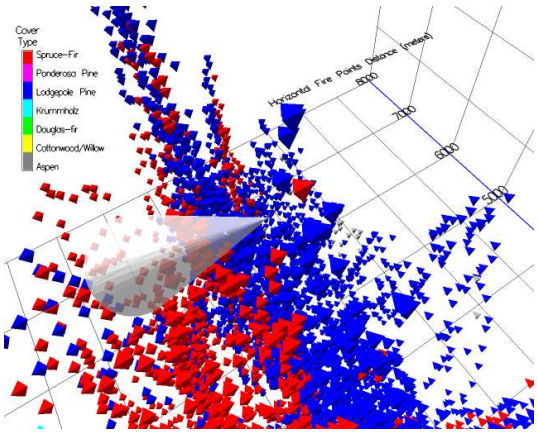


Fig. 1. Auditory data mining with the "virtual torch". In VR the torch cone originates from the hand of the observer, and can be pointed in any direction.

For all render methods the same rule apply: When a data sample is set to play it is rendered at its actual position in 3D space. Ideally the listener will be able to locate the origin of the sound (position and distance). For all methods it is possible to apply density thresholds in order to concentrate the attention on higher populated areas. Likewise the user may define a distance threshold and discard records placed in virtual space beyond this threshold.

5 A Case Study: Forest Cover Type

The following presents three example series using the Forest Cover Type dataset mentioned in section 5.

The dataset contains the forest cover type for 30×30 meter cells obtained from the US Forest Service (USFS) Region 2 Resource Information System (RIS) data.

Independent variables were derived from data originally obtained from US Geological Survey (USGS) and USFS data. Data is in raw form (not scaled) and contains binary (0 or 1) columns of data for qualitative independent variables *wilderness area* and *soil type*).

The dataset is not balanced; refer to the web-site for description of the variables and basic statistics[3]. .

Data summary

From the dataset documentation the following basic information was obtained:

Number of observations:	581012
Number of Attributes:	54
Attribute breakdown:	12 measures, but 54 columns of data (10 quantitative variables, 4 binary wilderness areas and 40 binary soil type variables)
Missing Values:	None

Name	Index	Values	Range
Elevation	1	1978	1859 – 3858 meters
Aspect	2	361	0 – 360 azimuth
Slope	3	67	0 – 66 degrees
Horizontal Hydrology Distance	4	551	0 – 1397 meters
Vertical Hydrology Distance	5	700	-173 – 601 meters
Horizontal Roadways Distance	6	5785	0 – 7117 meters
9am Hill-shade	7	207	0 – 255
Noon Hill-shade	8	185	0 – 255
3pm Hill-shade	9	255	0 – 255
Horizontal Fire Points Distance	10	5827	0 – 7173 meters
Wilderness Area	11	4	Cache la Poudre, Comanche Peak, Neota, Rawah
Soil Type	12	40	1 – 40
Forest Cover Type	13	7	Aspen, Cottonwood/Willow, Douglas-fir, Krummholz, Lodgepole Pine, Ponderosa Pine, Spruce-Fir

Table 1. Basic information about the variables in the Forest Cover dataset

[3] http://kdd.ics.uci.edu/databases/covertype/covertype.data.html

This dataset, see table 1, was originally used in a classification exercise, with 12 independent variables for classifying the cover type, and one control variable, *Forest Cover Type*. It was also used in [10] where the cover type was classified using different methods.

Data conversion

There are 13 variables of which the last three are categorical; that is, without any meaningful ordinal order. The categorical variable *Forest Cover Type*, index 13, can directly be used, but it is difficult to use the two categorical variables *Wilderness Area* and *Soil Type* in a 3D visualization as they are stored as mutually exclusive binary values.

These 44 binary variables (4 mutually exclusive values from *Wilderness Area* and 40 from *Soil Type*) were therefore converted to two quantitative variables; see index 11 and 12 in table 1.

5.1 The Examples

We will now present three different investigations:

- The first case will investigate usage of soundscapes in a situation where sound acts as a support for color, which represents *Forest Cover Type*.
- The second will investigate the same dataset but with sound used to represent *Wilderness Area* (categorical data).
- In the third example we will attempt to map *Vertical Hydrology Distance* to sound (continuous data).

For *all* cases the axes in the coordinate system indicate *Elevation*, *Horizontal Roadways Distance* and *Horizontal Hydrology Distance*. This combination does not receive particular high score when we calculate the partial correlation coefficient (see [2]). Interesting shapes appear as "tongues" stretching towards two of the axes' higher ends, *Horizontal Hydrology Distance* in particular. Figure 2 shows a 3D scatter plot of the dataset used for these examples. The two pictures show the same scatter plot from different viewpoints.

As it is difficult to present a soundscape in time in a document of this type, we will try to make a description of the soundscapes that occur. Black cubes in screen shots indicate currently sampled data entries.

The investigation of the dataset is not meant to be conclusive in any way, but should only serve as an example of how these tools can be used. Some of the information revealed by these examples are obvious and can be found simply by looking at the data. These are used for cross-reference.

During the tests the distance threshold is adjusted to investigate how this function affects the soundscape and the listeners perception of the content. We also try different rendering methods: Rendering the whole set and rendering a sweeping plane along each of the three axes. When applicable we bring out the virtual torch to test this on areas on high interest.

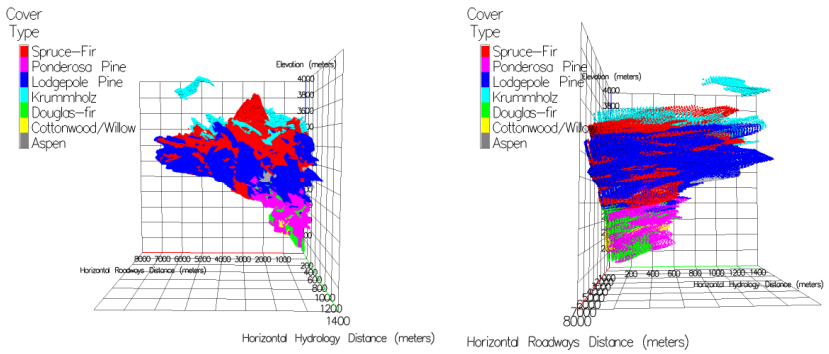


Fig. 2. 3D scatter plot from different viewpoints. *Elevation*, *Horizontal Roadways Distance* and *Horizontal Hydrology Distance* are mapped to the axes. Color shows *Cover Type*.

5.2 Sound Supporting Color

The main objective in this test is to investigate how the use of dynamic 3D soundscapes works as support for a visual parameter, in this case: Color. We will also investigate if it is possible to navigate the soundscape, so that we can locate areas of high concentration. Finally, we will will investigate the possibility of locating interesting areas that will not be visible to the eye.

The database is sampled every 120ms, and 8 entries are randomly picked and mapped to sound samples of spoken numbers 1 to 7 representing the seven *Cover Types*. Initially, all thresholds are set to maximum values, so that all observations are potential sources. The listener is placed in the middle of the coordinate system, and starts navigating the soundscape from there. It is allowed to adjust distance threshold.

– The immediate overall impression is a soundscape consisting of the numbers 5 and 7, which are the two dominant types of cover type: *Lodgepole Pine* and *Spruce Fir*. It is difficult to hear other types.

Distance threshold is lowered to 10 (graphical) units[4]. This allows the listener to investigate close range areas further by navigating through the data.

– It does not reveal anything straight away, but closing in on the area around the middle of the elevation axis increases the number of *Aspen* (Type 1) to a noticeable level. It reveals what can be seen from the color: That the number of *Aspen* are few compared to *Lodgepole Pine* and *Spruce Fir* at that elevation point (and they are close to roads in general).

[4] For reference: The coordinate system is $100 \times 100 \times 100$ graphical units

Next, the *Elevation* Axis is rendered in time. Data sampling is done using a sliding window (Figure 3) and distance threshold reset to maximum. The objective is then to navigate around in the soundscape and listen for interesting things.

– This gives the impression of gradually changing *Cover Type* as the *Elevation* increases. After several passes it also reveals that there still are types that we cannot see in the scatter plot when *Lodgepole Pine* and *Spruce Fir* become visually dominant (primarily *Aspen*).

Rendering the other axes does not reveal anything that can not be seen from the colors. Using the torch to make close-up investigation of statistical values does not really make sense in this case where sound acts as support for color.

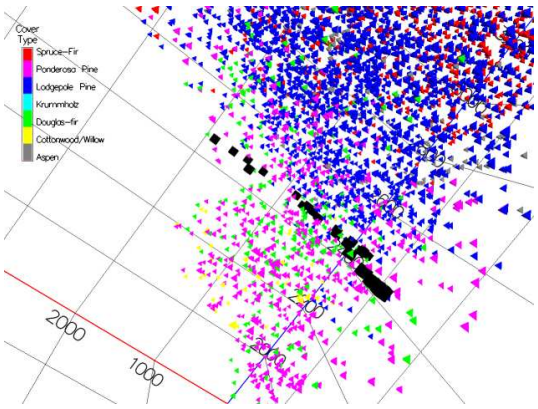


Fig. 3. Sampling along the *Elevation* axis. Black cubes mark the current samples.

5.3 Sound Representing Categorical Data

The main objective in this test is to investigate how the use of dynamic 3D soundscapes works for data mining when the there is no visual reference, and the data variable that is used for rendering the soundscape consists of a few (four) categorical values. We will see if it is possible to get an idea of how the selected variable is distributed. We will also investigate if it is possible to navigate through the soundscape, letting us locate areas of interest.

The database is sampled every 120ms, and 8 entries are randomly picked and mapped to sound samples of spoken numbers 1 to 4 representing the four *Wilderness areas*. Color still shows *Cover Type*. Initially all thresholds are set to maximum values, so that all observations are potential sources. The listener is placed in the middle of the coordinate system, and starts navigating the soundscape from there. It is allowed to adjust distance threshold. Doing a bit of "cheating" by changing color to represent *Wilderness Area* shows the layout (See figure 4).

– The initial experience when sampling the whole set randomly is an overweight of area 2 and 4. This is expected from the layout of the database.

We choose to render the 3 axes one by one while navigating the dataset to get a picture of the Wilderness Area distribution.

– It becomes clear that the large tongue stretching out the *Horizontal Roadways Distance* axis is area 4. About half the maximum distance area 2 gradually increases. The slim tongues that reach out the *Horizontal Hydrology Distance* axis are primarily area 2 with some representation of area 4. Data from area 3 seems to be located at high *Elevation* with low *Roadway Distance* and area 1 is located at low *Elevation* and low *Hydrology* and *Roadway Distances*.

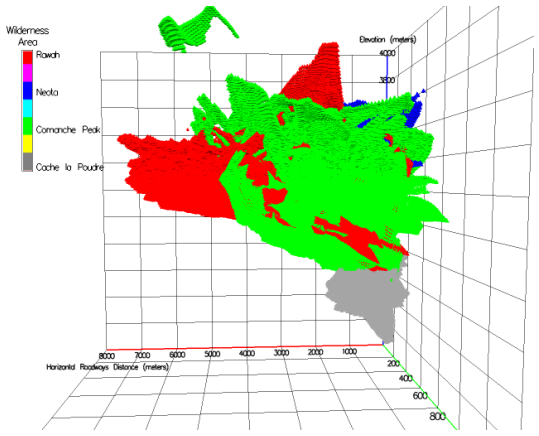


Fig. 4. A "sneak peak" at *Wilderness Area* distribution using color.

We then choose to try to identify what kind of *Forest Cover* that is in different *Wilderness Areas*. By looking at the colors one can see that *Cottonwood/Willow, Douglas fir* and *Ponderosa Pine* are grouped in one corner of the scatter plot. Distance threshold is set to 10 and the area is investigated further (by navigating into this area).

– This reveals that all these *Cover Types* are in area 1 until the *Elevation* reaches a certain level.
– This area also has a few *Lodgepole Pines*.

It now seems feasible to bring out the torch and point it where area 1 seems to stop (figure 5) to see how these *Cover Types* are distributed in the *Wilderness Areas*.

– Closer investigation with the torch reveals that area 1 has a few observations around *Elevation* 2600 where it seems to stop. Areas 2 and 4 take over and have a few *Douglas-fir* and *Ponderosa Pine* but no *Cottonwood/Willow*.
– Moving a bit upwards along the *Elevation* axis with the torch aimed at *Aspen* confirms that this only exist in area 4.

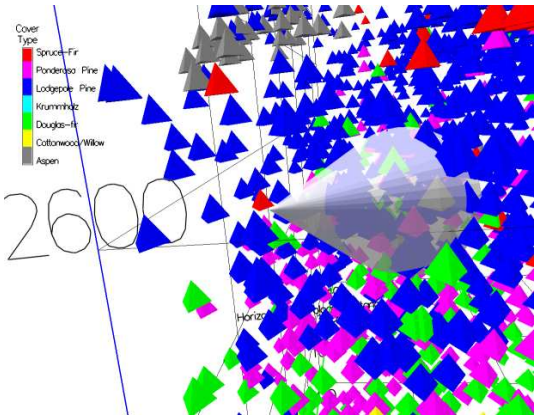


Fig. 5. Investigating area 1 maximum elevation area.

5.4 Sound Representing Continuous Data

The main objective in this test is to investigate how the use of dynamic 3D soundscapes works for data mining when the there is no visual reference, and the data variable that is selected for rendering the soundscape consists of many different observations. We will see if it is possible to get an idea of how the selected variable is distributed. We will also investigate if it is possible to navigate through the soundscape, so that we can locate areas of interest.

The database is sampled every 120ms, and 8 entries are randomly picked and mapped to synthesized waveforms spaced on a Ionic scale over two octaves. The values in the chosen scale (*Vertical Hydrology Distance*) are normalized and mapped to this scale so that low values become low pitch and visa versa. Color still shows *Cover Type*. Initially all thresholds are set to maximum values so that all observations are potential sources. The listener is placed in the middle of the coordinate system starts navigating the soundscape from there. It is allowed to adjust distance threshold.

– The immediate overall impression is a soundscape with many different values but with high concentration of mid range values and very few in the ultimate high and low region.

Distance threshold is again lowered to 10 units allowing closer inspection of local areas by navigation.

- There is a noticeable change in the soundscape along the *Horizontal Hydrology Distance* axis. At low distance the values seem concentrated on a value in the middle of the lower octave (i.e. around 1/4th of the maximum *Vertical Distance*, i.e. around 0 meters, since this data variable has both negative and positive values). There does not seem to be any very low or high values.
- Moving in the direction of high *Horizontal Hydrology Distance* creates a more distributed soundscape with many notes of different pitch. Sounds seem concentrated around middle values with a larger spread than initially, but there are definitely some very low and high values in this area.

It seems feasible to try to render along the three axes, especially *Horizontal Hydrology Distance* should be interesting.

- This confirms what was indicated rendering the whole scatter plot. There is a strong representation of a level about 1/4th as mentioned above
- This spreads out as *Horizontal Hydrology Distance* increases.
- When *Horizontal Hydrology Distance* is 0 *Vertical Hydrology Distance* is also 0 provided that the pitch value we hear as about 1/4th of the total tonal range is equal to the value of 0. (This correlation between the two distances would be expected for e.g. trees close to lakes and rivers).

Mapping *Vertical Hydrology Distance* to color (Figure 6) gives a clearer view of how the distribution of this value changes as *Horizontal Hydrology Distance* increases. The red tongue was not detected during the test, but was only hearable as a few high pitched sounds, which was difficult to locate.

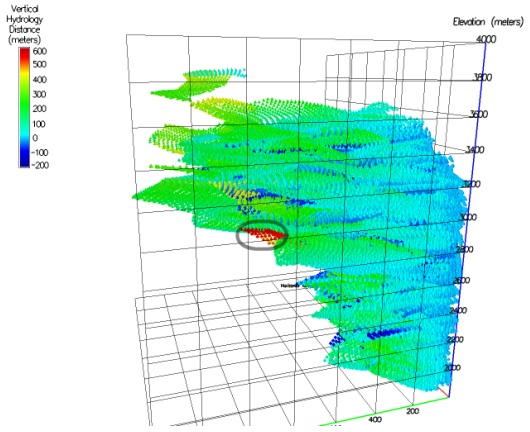


Fig. 6. Mapping *Vertical Hydrology Distance* to color. Notice the red tongue (marked).

6 Discussion

The tools and methods presented above are only few of many possibilities of this system. There are virtually unlimited ways to construct a soundscape using different sounds and settings.

In general, sound seems to support a visual parameter like color quite well. It does not reveal much new information; but can be useful if a data structure occludes other interesting observations. If certain clusters are out of visual range it will create sound cluster in that direction, provided that there is sufficient data in that cluster.

Mapping statistical values to a few different sound events works with success when the data values are a few categorical values. If the data values are continuous it is more difficult to suggest a soundscape with the same informational value, though using the synthesized sounds may still yield some information about relative values.

Sampling of the whole 3D scatter plot randomly and especially along the three axes in time gives us a soundscape that is much similar to the one we will get by choosing color rather than sound. The real strength of this method appears when comparing color and sound information to investigate the correlation between these.

The virtual torch proves quite useful for finding the direct statistical value of a given observation as long as the statistical values are few and preferably categorical. When this is not the case it will still provide information on relative values.

An important parameter to consider is the distance threshold that enables the listener to concentrate on the local area, because this also seems to eliminate most of the potential background noise created from distant objects. However, this also eliminates the ability to navigate towards distant areas on basis of the auditory cue from these. This was especially true for the last test where the synthesizer was used. In the second test it was possible to work with a higher distance threshold, probably because of the few different sounds.

6.1 Navigation Using Sound

When the user becomes familiar with the current properties of a 3D soundscape it may become possible to navigate supported by sound cues, given that there are clusters that provide sufficient positional cues. In cases where there are no apparent clusters or other patterns in the soundscape it may just confuse the user. In this case it is probably a better solution not to use the sound tools.

In any case, using soundscapes does hold enough information to give a strong indication of the distribution of a given statistical value. This information is also significant enough to trigger a closer investigation in most cases.

7 Conclusion and Future Work

This project intended creating software tools that allowed us to use sound to assist us in performing VDM in VR. This paper presented two basic sound tools developed for this purpose, and our aim was to present and test these tools, in various ways.

Our work has shown that it is possible to use sound for data mining in VR as either a support for visual parameters or as a stand-alone method, and especially using different exchangeable sample banks to represent statistical values proves to be a useful way of locating data values in VR. The success will depend on which type of data we wish to investigate, since keeping a simple soundscape is crucial for precise perception of value. Still it is possible to encode some kind of information about level for data that is more complicated.

Future tests should try to investigate the threshold of complexity for soundscapes that are useful for data mining (i.e. holds some kind of numerical value). This is important in order to avoid listener fatigue and information overload which was a common problem during this work.

Acknowledgments

We gratefully acknowledge the support to the 3DVDM project from the Danish Research Councils, grant no. 9900103. We also acknowledge Jock A. Blackard and Colorado State University, who are the donors and copyright holders of the Forest Cover Type database.

References

1. Nagel, H.R., Granum, E., Musaeus, P.: Methods for visual mining of data in virtual reality. In: Proceedings of the International Workshop on Visual Data Mining, in conjunction with ECML/PKDD2001, Freiburg, Germany, 2nd European Conference on Machine Learning and 5th European Conference on Principles and Practice of Knowledge Discovery in Databases (2001) 13–28
2. Nagel, H.R., Vittrup, M., Granum, E., Bovbjerg, S.: Exploring non-linear data relationships in vr using the 3d visual data mining system. In: Proceedings of the International Workshop on Visual Data Mining, in conjunction with The Third IEEE International Conference on Data Mining, Melbourne, Florida, USA (2003)
3. Granum, E., Musaeus, P.: Constructing virtual environments for visual explorers. In Quotrup, L., ed.: Virtual Space: The Spatiality of Virtual Inhabited 3D Worlds. Springer-Verlag (2002)
4. Brown, M.L., Newsome, S.L., Glinert, E.P.: An experiment into the use of auditory cues to reduce visual workload. In: CHI '89 Proceedings, New York, USA (1989)
5. Mutanen, J.: Perception of sound source distance, Nokia Research Center (2003)
6. Baluert, J.: Spatial hearing: the psychophysics of human sound localization, The MIT Press, ISBN 0-262-02413-6 (1997)
7. Hermann, T., Ritter, H.: Listen to your data: Model-based sonification for data analysis. In: Proceedings of ISIMADE 1999, Baden-Baden, Germany (1999)

Interactive Decision Tree Construction for Interval and Taxonomical data

François Poulet

ESIEA Recherche
38, rue des Docteurs Calmette et Guérin
Parc Universitaire de Laval-Changé
53000 Laval, France
poulet@esiea-ouest.fr

Abstract. Visual data-mining strategy lies in tightly coupling the visualizations and analytical processes into one data-mining tool that takes advantage of the assets from multiple sources. This paper presents two graphical interactive decision tree construction algorithms able to deal either with (usual) continuous data or with interval and taxonomical data. They are the extensions of two existing algorithms: CIAD [1] and PBC [2]. Both CIAD and PBC algorithms can be used in an interactive or cooperative mode (with an automatic algorithm to find the best split of the current tree node). We have modified the corresponding help mechanisms to allow them to deal with interval-valued attributes. Some of the results obtained on interval-valued and taxonomical data sets are presented with the methods we have used to create these data sets.

1 Introduction

Knowledge Discovery in Databases (or KDD) can be defined [3] as the non-trivial process of identifying patterns in the data that are valid, novel, potentially useful and understandable. In most existing data mining tools, visualization is only used during two particular steps of the data mining process: in the first step to view the original data, and in the last step to view the final results. Between these two steps, an automatic algorithm is used to perform the data-mining task. The user has only to tune some parameters before running the algorithm and waiting for its results.

Some new methods have recently appeared [4], [5], [6], trying to involve more significantly the user in the data mining process and using more intensively the visualization [7], [8], this new kind of approach is called visual data mining. In this paper we present some tools we have developed, which integrate automatic algorithms, interactive algorithms and visualization tools. These tools are two interactive classification algorithms. The classification algorithms use both human pattern recognition facilities and computer calculus power to perform an efficient user-centered classification. This paper is organized as follows.

In section 2 we briefly describe some existing interactive decision tree algorithms and then we focus on the two algorithms we will use for interval-valued data and

taxonomical data. The first one is an interactive decision tree algorithm called CIAD (Interactive Decision Tree Construction) using support vector machine (SVM) and the second is PBC (Perception Based Classifier).

In section 3 we present the interval-valued data: how they can be sorted, what graphical representation can be used and how we perform the graphical classification of these data with our decision tree algorithms.

The section 4 presents the same information as section 3 but concerning the taxonomical data. Then we present some of the results we have obtained in section 5 before the conclusion and future work.

2 Interactive decision tree construction

Some new user-centered manual (i.e. interactive or non-automatic) algorithms inducing decision trees have appeared recently: Perception Based Classification (PBC) [9], Decision Tree Visualization (DTViz) [10], [11] or CIAD [12]. All of them try to involve the user more intensively in the data-mining process. They are intended to be used by a domain expert and not the usual statistician or data-analysis expert. This new kind of approach has the following advantages:

- the quality of the results is improved by the use of human pattern recognition capabilities,
- using the domain knowledge during the whole process (and not only for the interpretation of the results) allows a guided search for patterns,
- the confidence in the results is improved, the KDD process is not just a "black box" giving more or less comprehensible results.

The technical part of these algorithms are somewhat different: PBC and DTViz use an univariate decision tree by choosing split points on numeric attributes in an interactive visualization. They use a bar visualization of the data: within a bar, the attribute values are sorted and mapped to pixels in a line-by-line fashion according to their order. Each attribute is visualized in an independent bar (cf. fig.1). The first step is to sort the pairs ($attr_i$, class) according to attribute values, and then to map to lines colored according to class values. When the data set number of items is too large, each pair ($attr_i$, class) of the data set is represented with a pixel instead of a line. Once all the bars have been created, the interactive algorithm can start. The classification algorithm performs univariate splits and allows binary splits as well as n-ary splits.

Only PBC and CIAD provide the user with an automatic algorithm to help him choose the best split in a given tree node. The other algorithms can only be run in a 100% manual interactive way.

CIAD is a bivariate decision tree using line drawing in a set of two-dimensional matrices (like scatter plot matrices [13]). The first step of the algorithm is the creation of a set of $(n-1)^2/2$ two-dimensional matrices (n being the number of attributes). These matrices are the two dimensional projections of all possible pairs of attributes, the color of the point corresponds to the class value. This is a very effective way to graphically discover relationships between two quantitative attributes. One particular matrix can be selected and displayed in a larger size in the bottom right of the view (as shown in figure 2 using the Segment data set from the UCI repository [14], it is

made of 19 continuous attributes, 7 classes and 2310 instances). Then the user can start the interactive decision tree construction by drawing a line in the selected matrix and performing thus a binary, univariate or bi-variate split in the current node of the tree. The strategy used to find the best split is the following. We try to find a split giving the largest pure partition, the splitting line (parallel to the axis or oblique) is interactively drawn on the screen with the mouse. The pure partition is then removed from all the projections. If a single split is not enough to get a pure partition, each half-space created by the first split will be treated alternately in a recursive way (the alternate half-space is hidden during the current one's treatment).

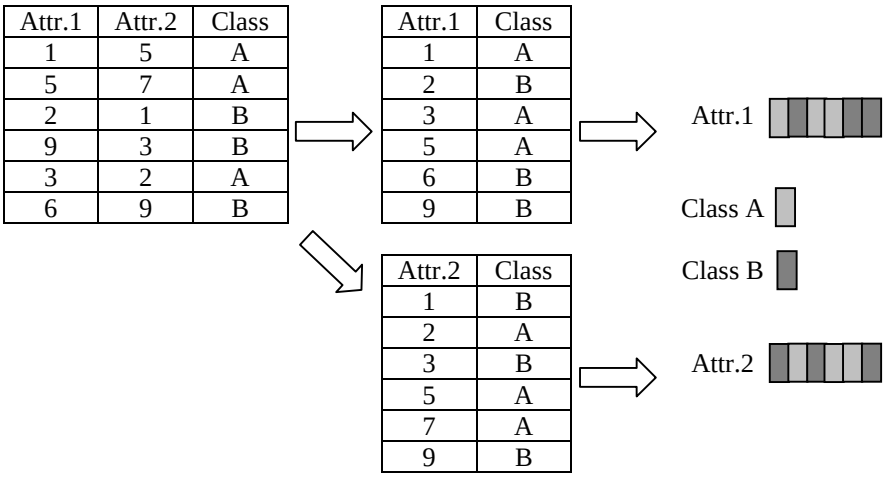


Fig. 1. Creation of the visualization bars with PBC

At each step of the classification, some additional information can be provided to the user like the size of the resulting nodes, the quality of the split (purity of the resulting partition) or overall purity. Some other interactions are available to help the user: it is possible to hide, show or highlight one class, one element or a group of elements.

A help mechanism is also provided to the user. It can be used to optimize the location of the line drawn (the line becomes the best separating line) or to automatically find the best separating line for the current tree node or for the whole tree construction. In the latter case, the line is drawn on the screen (for each step of the algorithm if the help mechanism is used for the whole tree construction). They are based on a support vector machine algorithm, modified to find the best separating line (in two dimensions) instead of the best separating hyperplane (in n-1 dimensions for a n-dimensional dataset). The SVM algorithms are only able to deal with two classes, when we have more than two classes we use the once-against-all approach.

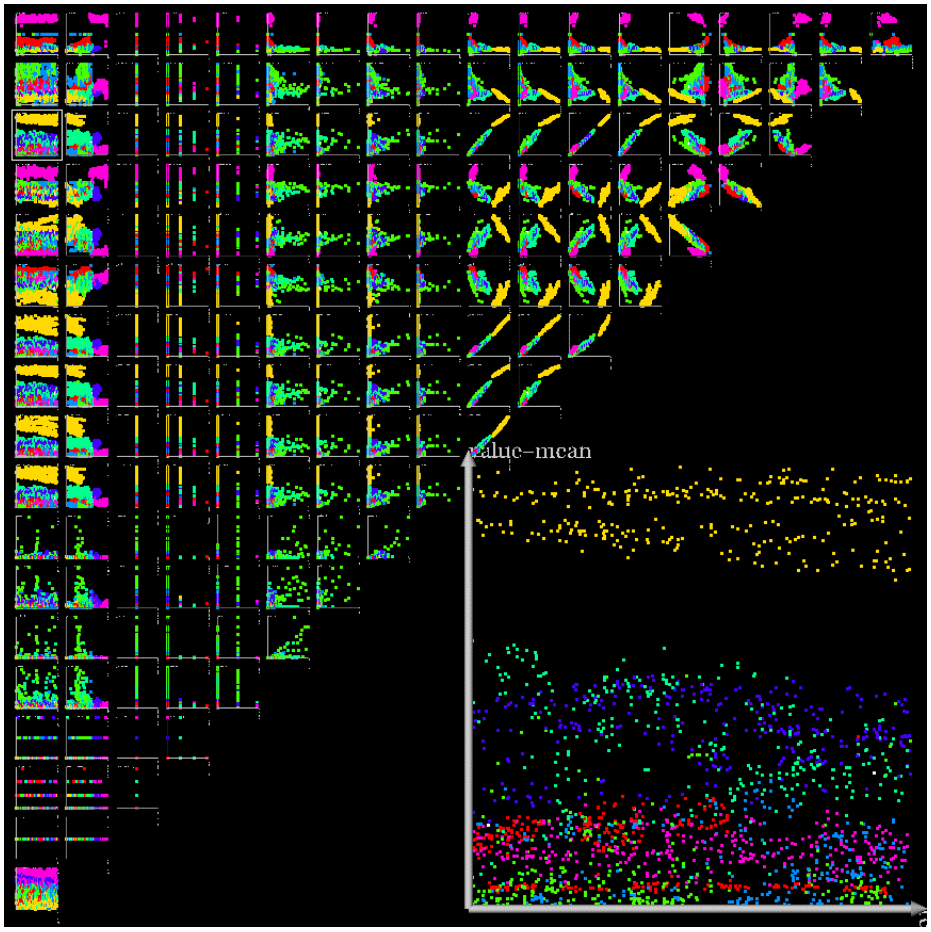


Fig. 2. The Segment data set displayed with CIAD

3 Interval data

Decision trees usually deal with qualitative or quantitative values. Here we are interested in interval-valued data. This kind of data is often used in polls (for example for income or age). We only consider the particular case of finite intervals.

3.1 Ordering interval data

To be able to use this new kind of data with PBC, we need to define an order on these data. There are mainly three different orders we can use [15]: according to the minimum values, the maximum values or the mean values. Let us consider two interval data: $I_1=[l_1,r_1]$ (mean=m_1) and $I_2=[l_2,r_2]$ (mean=m_2).

If the data are sorted according to the minimum values, then:
if $l_1 = l_2$, then $I_1 < I_2 <=> r_1 < r_2$; if $l_1 \neq l_2$, then $I_1 < I_2 <=> l_1 < l_2$.

If the data are sorted according to the maximum values, then:
if $r_1 = r_2$, then $I_1 < I_2 <=> l_1 < l_2$; if $r_1 \neq r_2$, then $I_1 < I_2 <=> r_1 < r_2$.

And finally, if the data are sorted according to the mean values, then $I_1 < I_2 <=> m_1 < m_2$.

We can choose any of these three functions to create the bar in the first step of the PBC algorithm in order to sort the data according to the values of the current attribute.

3.2 Graphical representation of interval data

In order to use interval data with CIAD+, we must find what kind of graphical representation can be used in the scatter plot matrices for two interval attributes and for one interval attribute with a continuous one. In the latter case, a segment (colored according to the class) is an obvious solution.

To represent two interval attributes in a scatter plot matrix, we need a two dimensional graphical primitive allowing us to map two different values on its two dimensions, the color being the class. Among the possible choices, there are a rectangle, an ellipse, a diamond, a segment or a cross as shown in figure 3. To avoid occlusion, we must use the outline of the rectangle, the diamond and the ellipse.

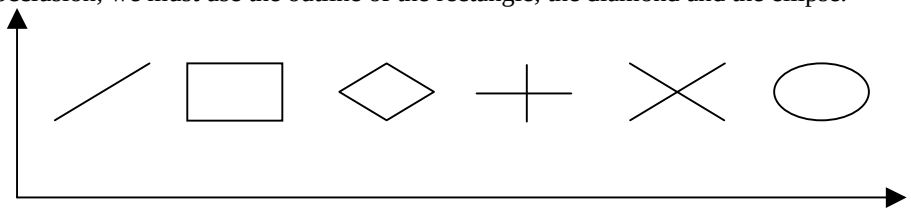


Fig. 3. How to visualize interval x interval data in 2 dimensions?

The rectangle and the diamond will introduce some bias when two rectangles (diamonds) are overlapping, it is impossible to know if there are two or three rectangles (diamonds) drawn as shown in figure 4.

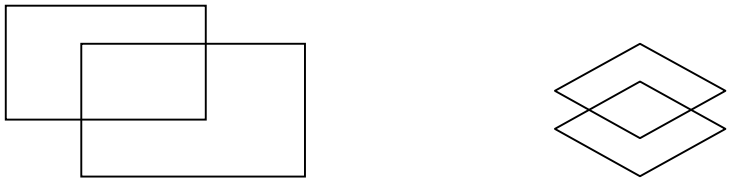


Fig. 4. Are there three or two rectangles and diamonds?

And this can become considerably more complicated if we increase the number of overlapping rectangles or diamonds. For example, in the figure 5, we have drawn 3 rectangles and three diamonds, but it is possible to see between 3 and 6 diamonds and between 3 and at least 19 rectangles!

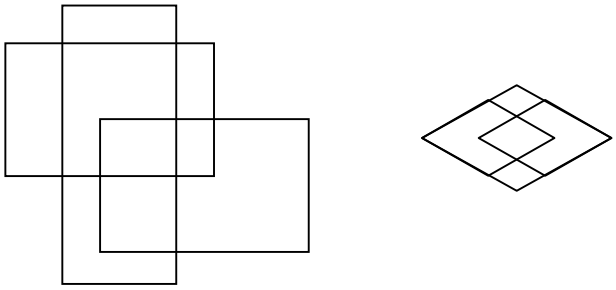


Fig. 5. Three rectangles and three diamonds

So we cannot use rectangles or diamonds; the ellipse, segment and cross do not have the same drawbacks. Concerning the segments, they have another kind of disadvantage: they are drawn from the minimum to the maximum of the two intervals, so they are all in the first quadrant (or third one). When using such a representation, people always try first to find a separating line in the same quadrant, even if a larger pure partition exists but requires a cut in the second (or fourth) quadrant. That is why we have rejected this choice. The only remaining graphical representations are the ellipses and the crosses. The cross being of a lower cost to display, it is the graphical primitive we have chosen.

3.3 Classifying interval data with PBC

We have explained how the interval data can be sorted in section 3.1. This method is used in the first step of the PBC algorithm to create the bar charts. Once this task has been performed for each attribute, the classification algorithm is exactly the same as for continuous data (when it is used in its 100% manual mode).

3.4 Classifying interval data with CIAD

As explained in section 2, the first step of CIAD is to display a set of two-dimensional matrices being the two-dimensional projections of all possible pairs of attributes, the color corresponding to the class value. This first step will be the same for the interval data, but using crosses instead of points. Once all the matrices have been drawn, the algorithm is exactly the same as the continuous version. We try to find the best pure partition, etc.

For the time being, the help mechanism is the same too. We use a SVM algorithm based on the cross centers (a point equal to the middle of the interval).

4 Taxonomical data

A taxonomical variable [16] can be defined as a mapping of the original data on a set of ordered values. It is equivalent to a structured or hierarchical variable. For example, a geographical description can be made with the town or with the county or

the country. The taxonomical variable describing the location will use any level of the description (town, county or country). In the data set we can find items with a location given by a town name and other ones with a county or country name. From the hierarchical description, we get a set of ordered values by using a tree traversal (either depth-first or breadth-first). Let us show the results on a very simple example of geographical location. The location is defined by the binary tree described in figure 6. The leaves correspond to town, and the upper levels to county and country.

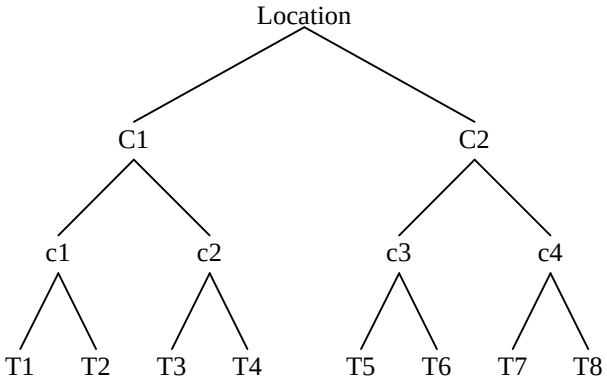


Fig. 6. Hierarchical description of the location

In the data set, the location attribute can take any value of this tree (except the root value). An example of such a data set is given in table 1, the *a priori* class has two possible values: 1 and 2. The two columns on the left correspond to the original data, the two columns in the middle are the same data set sorted according to a depth-first traversal of the tree (T1, c1, T2, C1, T3, c2, T4, etc.) and the two columns on the right are the same data set sorted according to a breadth-first traversal of the tree (C1, C2, c1, c2, c3, c4, T1, etc.).

Location	Class	Location (depth-1st)	Class	Location (breadth-1st)	Class
T1	1	T1	1	C1	2
T2	2	c1	2	c1	2
T3	1	T2	2	c3	2
T3	1	C1	2	T1	1
c1	2	T3	1	T2	2
C1	2	T3	1	T3	1
T5	1	T5	1	T3	1
c3	2	c3	2	T5	1
T7	1	T7	1	T7	1

Table 1. An example of taxonomical data set

4.1 Graphical representation of a taxonomical variable

Once the data have been sorted (whatever the tree traversal is), a taxonomical variable can be seen as an interval variable. When the variable is not a leaf of the tree (for example C1 or c3 in figure 6), it is graphically equivalent to the interval made of all the leaves of the corresponding sub-tree (C1=[T1,T4] and c3=[T5,T6]). In a two dimensional representation, we will use exactly the same graphical primitive as for the interval data: a cross for (taxonomical x taxonomical) or (taxonomical x interval) representation and a segment for (taxonomical x continuous) or (interval x continuous) representation.

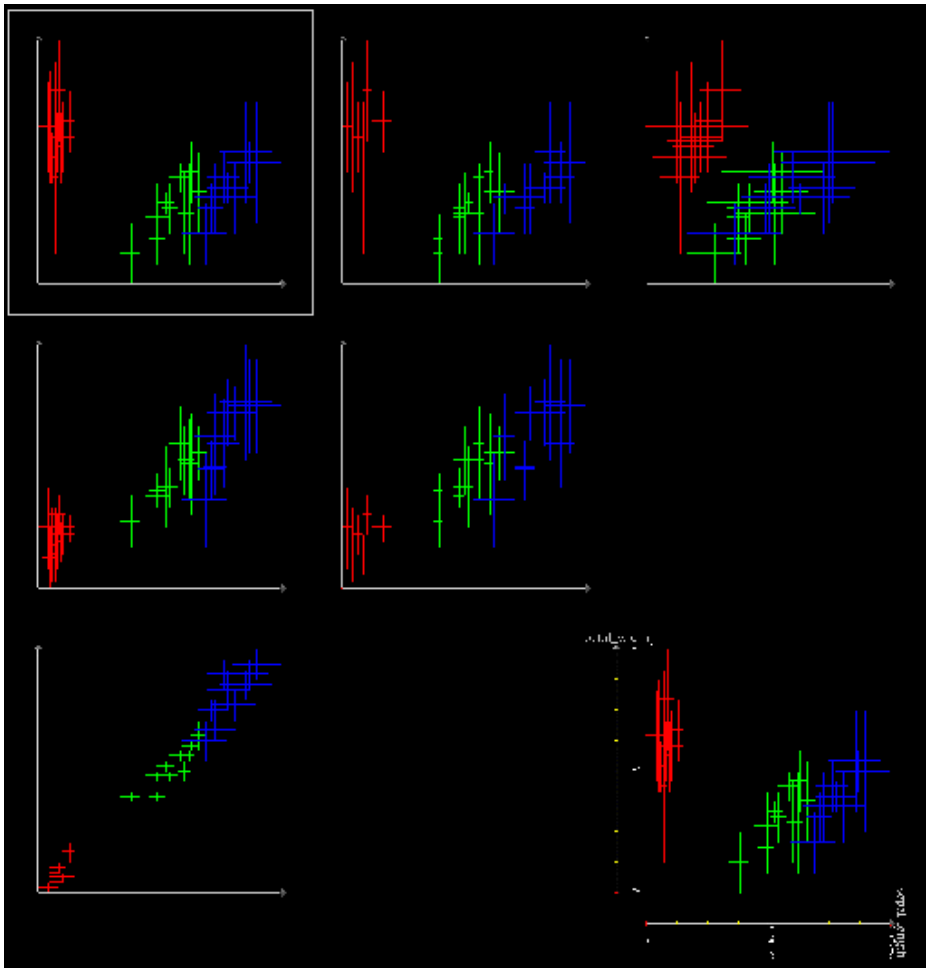


Fig. 7. Interval-valued version of the iris data set

4.2 Interactive taxonomical data classification

Here again, the way PBC is used is exactly the same as for interval or continuous data (when it is used in 100% manual mode). There is an order for the taxonomical data, it is used in the first step of the PBC algorithm, to sort the data according to the attribute value.

CIAD is also used the same way as for interval data. The help mechanisms can be used if we consider only the center of the interval (corresponding to the taxonomical data treated).

5 Some results

First of all, we must underline that we have not found any result of a decision tree algorithm dealing with interval and taxonomical data and there was no available interval-valued data sets in existing machine learning repositories when we wrote this paper (the only one we knew was unavailable [17]). We present in this section some of the results we have obtained and we start with the description of the data sets we have created. We have used existing data sets with continuous variables to create the interval-valued data sets. The first data set used is the well-known iris data set from the UCI Machine Learning Repository [12]. First, a new attribute has been added to data set: the petal surface. Then the data set has been sorted according to this new attribute (it nearly perfectly sorts the iris types) and we have computed (for each attribute) the minimum and the maximum values of each group of five consecutive items. And so we obtain a data set made of four interval-valued attributes and 30 items (10 for each class).

The resulting display with the CIAD set of 2D scatter plot matrices is shown in figure 7 (the petal surface attribute has been removed). The original data set has one class linearly separable from the other two, and the other two not linearly separable from each other. The interval data have three linearly separable classes. Some of the data set items are represented with a segment because the minimum and maximum have the same value according to the second attribute used in the matrix.

The second data set we have created is an interval-valued version of the shuttle data set (which also comes from the UCI Machine Learning repository). This data set is made of nine continuous attributes, 43500 items and seven classes. Four of them have very few items and for these four classes we have only created one interval-valued item with the average value plus and minus the standard deviation for each attribute. The three remaining classes have more items, we wanted to have nearly one hundred interval-valued items, so we created a number of interval-valued items in proportion to the original number of continuous items. These items have been computed by a clustering algorithm (k-means) to get similar continuous elements in an interval-valued one (then the average value plus and minus the standard deviation are computed in each cluster for all the attributes). The same method has been used for both the training set and the test set. The graphical display of the shuttle-interval training set is shown in figure 8.

Once these data are displayed, we perform the interactive creation of the decision tree. The accuracy obtained on the training set is 100% with a 10-leaf tree (99.7% on the test set with a ten-fold cross-validation). On the continuous data set, the accuracy

obtained with CIAD was 99.9% on the test set with a tree size of nine leaves. As we can see, the tree size of the interval-valued data set is larger than the continuous one and the accuracy is lower. This is because the interval-valued data set has only one hundred elements compared to the 43500 elements of the original data set. One misclassified element has an accuracy loss of 1% on the training set and 3% on the test set in the first case and 0.002% (and 0.006%) in the later one. To get a similar accuracy, the interval-valued tree needs more splits (and more leaves) to avoid any misclassification when the continuous version allow tens of misclassification errors while keeping the accuracy greater than 99.9%.

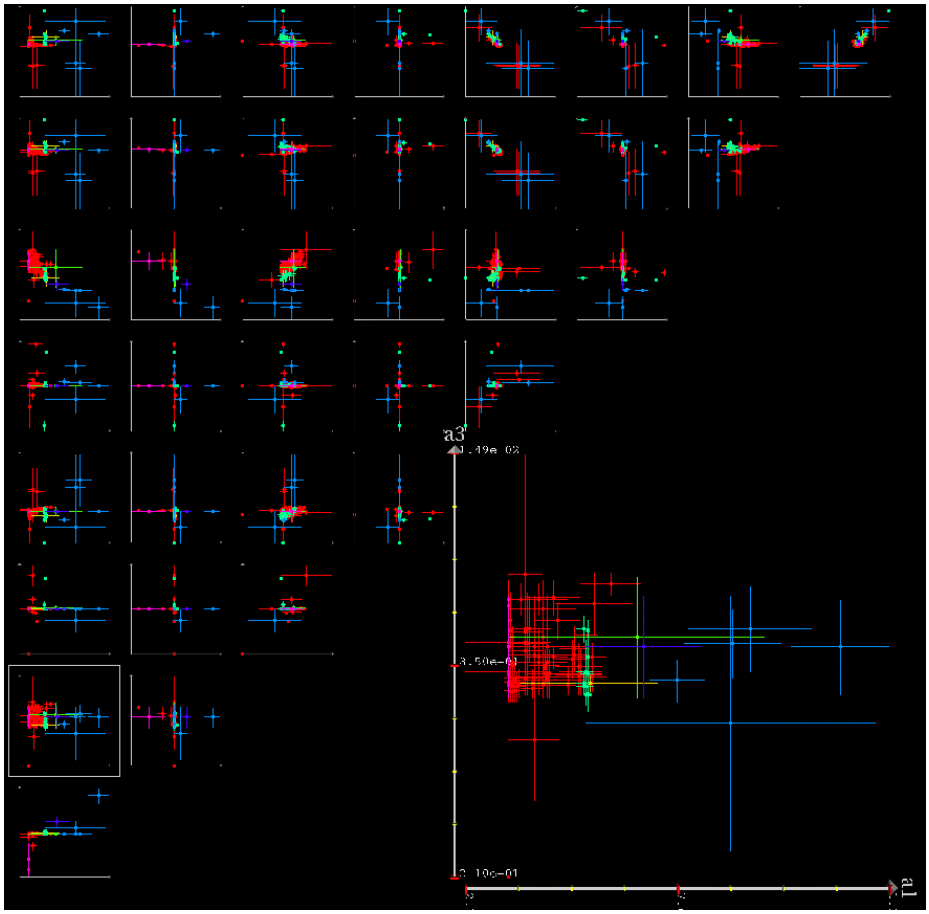


Fig. 8. Interval version of the shuttle data set

To evaluate the accuracy on the test set, our decision tree algorithm gives as output the source code of a C program we only need to compile and run it to compute the accuracy on the test set. We had not enough time to manage other large datasets, we are working on a software program being able to automatically create an interval-valued data set from a continuous one.

6 Conclusion and future work

Before concluding, some words about the implementation. All these tools have been developed using C/C++ and three open source libraries: OpenGL, Open-Motif and Open-Inventor. OpenGL is used to easily manipulate 3D objects, Open-Motif for the graphical user interface (menus, dialogs, buttons, etc.) and Open-Inventor to manage the 3D scene. These tools are included in a 3D environment, described in [18], where each tool can be linked to other tools and be added or removed as needed. Figure 9 shows an example with CIAD set of 2D scatter plot matrices, PBC bar charts and parallel coordinates. The element selected in a bar chart appeared selected too in the set of scatter plot matrices and in the parallel coordinates (in bold white). The software program can be run on any platform using X-Window, it only needs to be compiled with a standard C++ compiler. Currently, the software program is developed on SGI O2 and PCs with Linux.

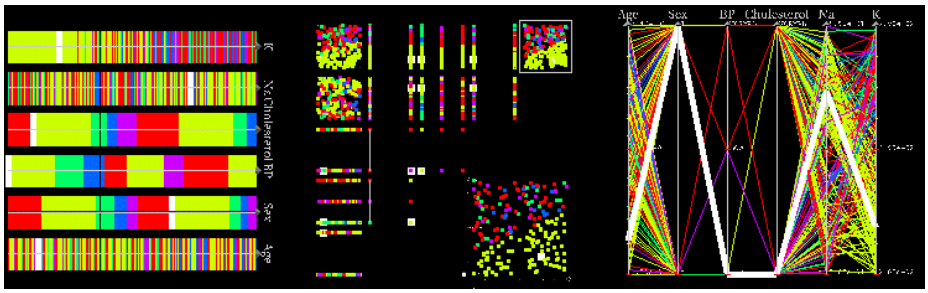


Fig. 9. Three linked representation of the drug data set

In this paper we have presented two new interactive classification tools able to deal with interval-valued and taxonomical data. The classification tools are intended to involve the user in the whole classification task in order to:

- take into account the domain knowledge,
- improve the result comprehensibility, and the confidence in the results (because the user has taken part in the model construction),
- exploit human capabilities in graphical analysis and pattern recognition.

A forthcoming improvement will be to use the method we have described to create interval-valued data sets as a pre-processing to reduce the number of item in very large data sets.

References

1. F.Poulet, "Cooperation Between Automatic Algorithms, Interactive Algorithms and Visualization Tools for Visual Data Mining" in proc. of VDM@ECML/PKDD'2002, International Workshop on Visual Data Mining, Helsinki, Finland, 67-80, 2002.
2. M.Ankerst, C.Elsen, M.Ester, H-P.Kriegel: "Perception-Based Classification", in Informatica, An International Journal of Computing and Informatics, 23(4), 493-499, 1999.
3. U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, R. Uthurusamy, Eds, "Advances in Knowledge Discovery and Data Mining", AAAI Press, 1996.

4. P.Wong, "Visual Data Mining", in IEEE Computer Graphics and Applications, 19(5), 20-21, 1999.
5. F. Poulet, "Visualization in data mining and knowledge discovery," in Proc. HCP'99, 10th Mini Euro Conference "Human Centered Processes" ed. P. Lenca (Brest, 1999), 183-192.
6. M.Ankerst, M.Ester, H-P.Kriegel, "Toward an Effective Cooperation of the Computer and the User for Classification" in proc. of KDD'2001, 179-188.
7. C.Aggarwal, "Towards Effective and Interpretable Data Mining by Visual Interaction", in SIKDD Explorations 3(2), 11-22, accessed from www.acm.org/sigkdd/explorations/.
8. B.Schneiderman, "Inventing Discovery Tools: Combining Information Visualization with Data Mining", in Information Visualization 1(1), 5-12, 2002.
9. M.Ankerst, "Visual Data Mining", PhD Thesis, Ludwig Maximilians University of Munich, 2000.
10. J.Han, N.Cercone, "Interactive Construction of Decision Trees" in proc. of PAKDD'2001, LNAI 2035, 575-580, 2001.
11. M.Ware, E.Franck, G.Holmes, M.Hall, I.Witten, "Interactive Machine Learning: Letting Users Build Classifiers", in International Journal of Human-Computer Studies (55), 281-292, 2001.
12. F.Poulet, "CIAD: Interactive Decision Tree Construction" in proc. of 8th Conf. of the French Classification Society, Pointe-à-Pitre, 275-282, 2001 (in french).
13. J.Chambers, W.Cleveland, B.Kleiner, P.Tukey, "Graphical Methods for Data Analysis", Wadsworth (1983).
14. C.Blake, C.Merz, UCI Repository of machine learning databases, [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science, (1998).
15. C.Mballo, F.Gioia, E.Diday, "Qualitative Coding of an Interval-Valued Variable", in proc. of the 35th Conference of the French Statistical Society, June 2003, Lyon, France (in french).
16. H.H.Bock, E.Diday, "Analysis of Symbolic Data: Exploratory Methods for Extracting Statistical Information from Complex Data", Springer-Verlag, Berlin-Heidelberg, 2000.
17. "The SODAS Project", accessed from http://www.ceremade.dauphine.fr.
18. F.Poulet, "FullView: A Visual Data-Mining Environment", in International Journal of Image and Graphics, 2(1), 127-144, 2002.

Author Index