Animated Interval Scatter-plot Views for the Exploratory Analysis of Large Scale Microarray Time-course Data

Paul Craig

A thesis submitted in partial fulfilment of the requirements of Napier University for the degree of Doctor of Philosophy

January, 2006

Abstract

Microarray time-course data relates to the recorded activity of large numbers of genes (~8,000) recorded in parallel over a number of time points (~20). This data contains a massive amount of potentially valuable information. In order to unlock that information, a biologist needs to find patterns of changing activity within variable (normally large) proportions of the data. Established techniques are, however, limited in the quantity and range of valuable patterns that they can allow a biologist to find. For example they cannot uncover patterns such as those where smaller numbers of genes have common activity over an interval of the data. This thesis presents an alternative approach where an animated display allows biologists to find these potentially valuable patterns.

The type of animated visualisation developed is unique in that it both presents abstract data which has no spatial attributes and maps time in the data to time in the display (to animate across time). This animation is formed by presenting an interval of the data and allowing that interval to be progressively re-specified with the display updated. Here, the perception of spatial motion in the display can be related to changes in the data. This allows the user to pre-attentively perceive patterns in the data and, as the display changes over time, perceive a greater variety of patterns that may be of relevance to their analysis. The main issue involved in developing the animated visualisation was the need to configure an effective display for frames of an animation using abstract qualities. This had to be effective, revealing enough information for patterns to be detected, and expressive so that meaning could be derived from motion and patterns could be properly interpreted. In addition to this, the interface needed to accommodate the inability of humans to absorb information when it is only presented for a brief time. These issues are dealt with by matching the user's conceptualisation of changes in the data to motion in the display, giving the user direct control over the pace and direction of the animation, interpolating the data for a smooth animation, and coordinating animated/static views.

Table of Contents

Chapter 1 Introduction ***1***
1.1 Motivation 2
1.2 Hypothesis 5
1.3 Contribution to Knowledge 7
Chapter 2 Information Visualisation ***9***
2.1 Origins and Related Fields 9
2.2 Models for Information Visualisation 12
2.2.1 The Reference Model for Visualisation: Interactive Mappings from Data to Visual Form. 12
2.2.2 The Knowledge Crystallisation Task Model 17
2.3 Guidelines for Information Visualisation Design 19
2.3.1 Methodologies for Application Development 19
2.3.2 Guidelines for Interface Design 21
2.3.3 Guidelines for Data Representations 24
2.4 Information Visualisation Techniques 33
2.4.1 Starfield Displays 33
2.4.2 Zooming, Dynamic Query Filters and Details on Demand 35
2.4.3 Multiple Coordinated Views 40
2.5 Animated Information Visualisations 41
2.5.1 Expressive Animations 42
2.5.2 Animated View Transformations 45
2.5.3 Animated Visual Mapping Transitions 50
2.5.4 Mapping Time to Time 53
2.6 Conclusion 56
Chapter 3 Microarray Time-course Data ***58***
3.1 Biological Background 58
3.2 Microarrays 61
3.3 Microarray Data 64
Chapter 4 Exploratory Analysis of Microarray Time-course Data ***67***
4.1 Rescaling 68
4.2 Clustering 70
4.2.2 Principle Component Analysis 72
4.2.3 Similarity and Dissimilarity Measures 73
4.2.4 Singular Value Decomposition 75
4.2.5 Radviz 76
4.2.6 The Similarity Matrix 78
4.2.7 Relevance Networks Visualisation 78
4.2.8 Multidimensional Scaling 79
4.2.9 Hierarchical Clustering 80
4.2.10 K-means Clustering 85
4.2.11 Self-Organizing Maps 88
4.2.12 Gene Shaving 90
4.2.13 Plaid Models 91
4.2.14 Evaluating Clustering Methods 92
4.3 Interactive Clustering 96
4.3.1 Basic Interaction 97
4.3.2 Drill-down Techniques 98
4.4 Non-clustering Interactive Techniques 99

4.4.1 Graph View Techniques....100
4.4.2 Scatter-plot Matrices....104
4.4.3 Scatter-plot Variations....107
4.5 Combined Techniques....108
4.6 Conclusion....109
Chapter 5 Developing an Information Visualisation Technique for Microarray Time-course Data: Requirements Analysis....110
5.1 User Requirements....110
5.2 Relation of User Requirements to General Biological Questions....115
5.3 Evaluation of Existing Techniques....117
5.3.1 Finding dominant trends....117
5.3.2 Finding Less Dominant Patterns....117
5.3.3 Finding Interval Patterns....118
5.3.4 Finding Complex and Subtle Patterns....124
5.3.5 Finding Relationships between Patterns, Predefined-groupings and Experimental Conditions....125
5.3.6 Store, Restore and Export Results....125
5.4 Conclusion....125
Chapter 6 Prototype 1: Animated Scatter-plots to convey Changes in Activity128
6.1 Design Rationale....130
6.2 Scatter-plot Layout....132
6.3 Animation....135
6.4 Interaction....137
6.5 Rescaling....140
6.6 Evaluation....140
6.6.1 Procedure....141
6.6.2 Results....142
Chapter 7 Prototype 2: Animated Scatter-plots to find Activity Patterns....145
7.1 Design Rationale....147
7.2 Interaction....149
7.3 Evaluation....153
7.3.1 Results....153
Chapter 8 Final Prototype: Animated Scatter-plots to Support Exploratory Analysis....157
8.1 Design Rationale....157
8.2 Rescaling and Distortion....158
8.3 Colour Coding....162
8.4 Parallel Views....164
8.5 Selecting Genes and Combining Selections....168
8.6 Evaluation....171
8.6.1 Procedure....172
8.6.2 Results....174
8.6.3 Conclusions....180
Chapter 9 Conclusions....181
9.1 Further Work....184
Appendices....186
Appendix 1: A selection of applications developed to support the exploratory analysis of microarray data....186
Appendix 2: Image buffering for scatter-plot and graph views in the final version187
References....189

List of Figures

Figure 1.1. Microarray time-course data. 2
Figure 2.1. Charles Minard's illustration of Napoleon's Russian campaign: The graph shows the size of the army by the width of the band across the map of the campaign on its outward and return legs, with temperature on the retreat shown on the line graph at the bottom (to be found in [27]). 10
Figure 2.2. The London underground map: Geographical perspective is disregarded in favour of graphic order and usability (to be found in [28]). 10
Figure 2.3. Reference model for visualisation (taken from [24]). 13
Figure 2.4. The knowledge crystallisation task model with subtasks (italics) and suggested roles for information visualisation (far LHS and RHS). Diagram taken from [24] 17
Figure 2.5. The prototype life cycle model 20
Figure 2.6. Rose et al.'s iterative model for developing and generalising visual interfaces (taken from [43]). 21
Figure 2.7. Chartjunk, reproduced in [27] 25
Figure 2.8. Proximity principle used to group rows (LHS) and columns (RHS) 27
Figure 2.9. Similarity principle used to group items in the same column by their shape 28
Figure 2.10. Similarity principle used to group items in the same column by shape and rows by colour 28
Figure 2.11. Area (Smallness) principle: Smaller areas tend to be seen as figures against a larger background. 29
Figure 2.12. Closure principle: The whole image of a circle can be made from partial visual data. 29
Figure 2.13. Continuity principle: Smooth contours (a-b and c-d) dominate irregular, abruptly changing contours. 29
Figure 2.14. Pragnänz / Simplicity Principle: Humans have an instinctive preference for the simplest, the most elegant and most stable interpretation with the figure appearing as an overlapping triangle and square rather than any other combination of shapes. 29
Figure 2.15. Ranking of perceptual task by effectiveness. Columns are for different types of information and tasks higher in the chart are perceived more accurately than those lower in the chart. Tasks shown in grey are not relevant to that columns type of information and bold lines are used to link tasks using the same graphical property for different types of information (reproduced from [57]). . 30
Figure 2.16. Change blindness: It is difficult to observe the difference between the two images. This is because the area of the picture that changes is not an area that would naturally draw the attention of the observer (taken from [62]). 33
Figure 2.17. FilmFinder: A Starfield display with details on demand and dynamic query sliders (taken from [33]). 35
Figure 2.18. Single value slider 37
Figure 2.19. Alphaslider (taken from [33]). 37
Figure 2.20. Range-slider (taken from [33]). 38
Figure 2.21. Excentric labelling (taken from [69]) 40
Figure 2.22. The expressive use of motion in a visualisation: the windows file copy pop-up window. 42

Figure 2.23. A supervisory control and data acquisition (SCADA) interface used for the real-time control and monitoring of industrial machinery: In this case, fermentation tanks (taken from[77]). 43
Figure 2.24. An animated presentation graphic: The 'swingometer' (taken from [79]). 45
Figure 2.25. The camera metaphor used to describe zooming in information visualisations. 46
Figure 2.26. Zooming: Zooming in and out (LHS), and panning (RHS) 46
Figure 2.27. The camera metaphor used to describe two dimensional projections of a thee-dimensional display with axes about which that display may be rotated to convey structure. 49
Figure 2.28. Cone-trees: An animated technique for hierarchical data (taken from [17]). 52
Figure 2.29. The animated exploration of dynamic graphs with radial layout (taken from [19]). 53
Figure 2.30. Animation of a thunderstorm simulation: The red isosurfaces represent columns of high vorticity, so noted because these areas potentially spawn tornadoes. The ribbons show wind velocity at two imaginary planes (taken from [21]). 55
Figure 3.1. The double helix structure of DNA with complimentary bindings C to G and A to T 59
Figure 3.2. Gene expression: Genes transcribe into RNA and RNA translate into proteins 59
Figure 3.3. Process of single cell-sample microarray experiment. 62
Figure 3.4. Process of a two cell-sample microarray experiment. 63
Figure 3.5. A microarray scanner 64
Figure 3.6. Transformation from microarray images to normalised microarray data (adapted from [106]). 65
Figure 3.7. Portion of a data table from a microarray time-course experiment 66
Figure 4.1. Mapping of recorded expression values to n-dimensional expression space: top) The three recorded expression values of a single gene are used to position it in a three dimensional expression space and bottom) multiple genes positioned in expression space. 72
Figure 4.2. Two representations of principle component analysis (PCA) results: left) PC1 is plotted against PC2 in a two-dimensional visual representation (the blue colour gradient indicates the density of gene representations in the display) and right) PC1 is plotted against PC2 and PC3 in a three-dimensional representation (taken from [117]). 73
Figure 4.3. The results of singular value decomposition: left) Eigengenes summarise trends in expression across all genes and right) genes are plotted according the correlation between their recorded expression and the patterns of the eigengenes (genes are colour coded according to classifications by Spellman et.al [111], images are taken from [119]). 76
Figure 4.4. Radviz: points around the circumference of the circle are known as dimensional anchors. Points within the circle correspond to data items. Representations of data items are positioned (In this case items correspond to genes colour-coded according to predefined gene classifications and anchors relate to samples for which those genes activity has been recorded, image taken from [122]). 77
Figure 4.5. Direct visualisation of a similarity matrix: left) Genes are unordered. right) Genes are ordered so that genes and groups of genes with similar recorded

expression are close together. This causes large blocks of light coloured cells along the top-left to bottom-right diagonal to appear. This shows large groups of genes with similar recorded expression (taken from [123]). 78
Figure 4.6. Relevance networks visualisation of a similarity matrix (taken from [118]). 79
Figure 4.7. Multidimensional scaling of genes. Pairs of genes for which similarity is above a certain threshold are linked by an edge (taken from [127]). 80
Figure 4.8. Heat-map/dendrogram visualisation of gene expression (taken from [132]). 83
Figure 4.9. Horizontal distance tree representations of hierarchical clustering results with symbols representing gene classifications (taken from [130]). 84
Figure 4.10. Radial distance tree representations of hierarchical clustering results with five main visual clusters colour coded (taken from [95]). 84
Figure 4.11. The results of K-means clustering: left) Separate heat-maps for each cluster and right) Separate graph views for each cluster (taken from [132]). 87
Figure 4.12. Self-organizing Maps visualisation of microarray data using U-distance matrix and colour coded so that the lighter the cell the higher the similarity: left) The overview shows clusters of genes (light coloured) and right) a detail view with centroid cells (with genes labelled or a dot to indicate they are empty) surrounded by cells indicating levels of inter-centroid similarity (taken from [139]). 89
Figure 4.13. A collection of screenshots from the GEDI visualisation of animated Self-organizing Maps. Each square is a frame of the animation and the arrows indicate the order of frames. Within frames, red indicates high expression and blue indicates low expression (taken from [140]). 90
Figure 4.14. Plaid modelling display of microarray data: left) The original data and right) the modelled data (taken from [143]). 92
Figure 4.15. A graph of gene activity over time for a single gene (time-points are labelled according to the biologists own notation, image created using [151]). 101
Figure 4.16. Parallel-coordinate plot: A simple example where five items (colour coded) have different values for four attributes (x1, x2, x3 and x4). 101
Figure 4.17. Parallel-coordinate plots before and after dimensional brushing (images generated using [158]). 103
Figure 4.18. Composite brushing with axes range sliders for brush manipulation (image generated using [158]). 103
Figure 4.19. Selections based on graph view mappings of microarray time-course data: (LHS query and RHS query results): an acceptable range of values over a given interval (top), an acceptable change in values between time points (middle) and a profile that the expression of genes must adhere to (bottom). Images created using [151]. 104
Figure 4.20. Scatter-plot matrix of Fisher's iris data [160] (taken from [161]). 105
Figure 4.21. Triangular scatter-plot matrix (a lower triangular matrix) of Fisher's iris data [160] (taken from [161]). 105
Figure 4.22. Scatter-plot matrix with brushing and linking (taken from [161]) 106
Figure 4.23. Three-dimensional scatter-plot 107
Figure 5.1. An interval pattern that would not be revealed by clustering the data (image created using [151]). 119
Figure 5.2. Heat-map clustering display of microarray time-course data with arrows at the bottom of the figure highlighting genes which contribute to a pattern

occurring exclusively over an interval of the data (see Figure 5.1, image created using [151]). 119
Figure 5.3. Example of types of relationships between patterns in microarray time-course data. 123
Figure 6.1. Basic design for Time-series Explorer 129
Figure 6.2. Layout of dimensional anchors in the first prototype Time-series Explorer: for an interval with three time-points (LHS) and an interval with four time-points. 133
Figure 6.3. Values required for linear interpolation (for unknown v_x in red). 136
Figure 6.4. Interpolation smoothes animation by gradually changing the pattern of activity for a gene to be displayed in the scatter-plot. 137
Figure 6.5. Screenshot of Prototype 1 interface: a) interval controller panel, b) selection mode panel, c) selected groupings panel, d) scatter-plot view and e) graph view. 138
Figure 6.6. General layout of gene representations according to their activity over an interval in the first prototype. 143
Figure 7.1. Basic design for second prototype Time-series Explorer 146
Figure 7.2. Attributes of a gene's time-series over an interval (P) used to determine the gene's scatter-plot coordinates. 146
Figure 7.3. Two genes with dissimilar activity over an interval and the same positions in the scatter-plot. 147
Figure 7.4. Deviation from mean attribute used to filter gene representations in the second prototype 149
Figure 7.5. Deviation from change attribute used to filter gene representations in the second prototype 149
Figure 7.6. Screenshot of Prototype 2 interface: a) interval controller buttons, b) graph view, c) dynamic query sliders, d) scatter-plot view , e) selected genes list f) groupings panel and g) query composition panel. 150
Figure 7.7. The query composition panel after three box-lassos 152
Figure 8.1. Attributes of a gene's time-series over an interval (P) to be mapped to the scatter-plot representation. 160
Figure 8.2. The spread of data in the distorted scatter-plot view. 161
Figure 8.3. The colour mapping used for the Time-series Explorer: a) The percentage of each hue corresponding to the number of overlaid points or crossing lines, b) the scale applied to the scatter-plot view and c) the scale applied to the graph view. 163
Figure 8.4. Change-in-activity graph view (bottom) can be used when significant trends in change-in-activity (from V12 to P1 and P3 to P8.5) are obscured in the activity graph view (top). 166
Figure 8.5. A split pane allows the user to adjust the relative proportions of the scatter-plot view and graph views: Scatter-plot bigger (top) and graph views bigger (bottom). 167
Figure 8.6. Time-series Explorer final prototype interface: a) toolbar, b) graph view, c) change-graph view, d) selected genes list and e) scatter-plot view and f) grouping panel. 168
Figure 8.7. Excentric labelling of the Scatter-plot 171
Figure 8.8. The identification of genes associated with the production of milk proteins with high activity at the early stages of lactation: a) Selecting relevant interval, b) observing an outlying group of genes with a high level of activity over this interval, c) labelling the gene symbols and d) highlighting the gene activity patterns for the remainder of the time-course. 175

Figure 8.9. Animating the scatter-plot to reveal general trends in activity for all genes over the entire time-course: Static frames from the animated scatter-plot. 178
Figure 8.10. An unexpected pattern of temporal activity: a) Animating the scatter-plot reveals a group of outlying genes with rising then falling activity over a small interval of the time-course, b) moving the mouse over the gene representations in the scatter-plot view allows them to be labelled and c) have their activity patterns over the entire time-frame highlighted in the graph view. 179
Figure 9.1. Different layers for image buffering 188

List of Tables

Table 2.1. Form of a data table in the reference model for visualisation (taken from [24]). 13
Table 2.2. Visual Variables (taken from [35])....15
Table 4.1. Summary of clusters produced by hierarchical clustering algorithms. 94
Table 4.2. Summary of display methods for hierarchical clustering. 95
Table 4.3. Summary of clustering methods that allow genes to belong to multiple clusters. 95
Table 4.4. Summary of methods that allow genes to be clustered across two or three dimensions....96
Table 8.1. Functionality of final prototype toolbar buttons.....169
Table 8.2. Results of the user evaluation....177

List of Equations

Eq. 4.1 Rescaling 68
Eq. 4.2 Centering transform 69
Eq. 4.3 Linear rescaling transform 69
Eq. 4.4 Log rescaling transform 70
Eq. 4.5 The general form of a similarity measure 74
Eq. 4.6 Euclidean distance 74
Eq. 4.7 Pearson's correlation coefficient 75
Eq. 8.1 Linear rescaling 159
Eq. 8.2 Logarithmic distortion 159
Eq. 8.3 Hyperbolic transform distortion 161

Acknowledgements

I would like to thank my supervisors Jessie Kennedy and Andrew Cumming, my thesis panel chair Alison Crerar and the biologists who helped contribute to the development and evaluation of the Time-series Explorer. Especially Edward K. Wagner at The Center for Virus Research (CVR) the University of California at Irvine; Peter Ghazal, Thorsten Forster and Paul Dickinson at the Scottish Centre for Genomic Technology and Informatics; and Torsten Stein at the Division of Cancer Sciences and Molecular Pathology Glasgow University. I would also like to thank my parents and all those at the School of Computing at Napier who have helped me out with constructive criticism throughout my period of PhD study. Especially the Object Database Systems group who were particularly helpful and supportive.

Contributing Publications

Craig, P., J. B. Kennedy, and Cumming, A. (2005). "Animated Interval Scatter-plot Views for the Exploratory Analysis of Large Scale Microarray Time-course Data." Information Visualisation **4**(3): 2005.

Craig, P., Kennedy, J. and Cumming, A. (2005). Coordinated Parallel Views for the Exploratory Analysis of Microarray Time-course Data. In, 3rd International Conference on Coordinated & Multiple Views in Exploratory Visualization . London: IEEE Computer Society.

Craig, P., Kennedy, J. and Cumming, A. (2005). Time-series Explorer: An Animated Information Visualisation for Microarray Time-course Data. BMC Bioinformatics, 6(Suppl 3), P8.

Craig, P. and J. B. Kennedy (2003). Coordinated Graph and Scatter-Plot Views for the Visual Exploration of Microarray Time-Series Data. IEEE Symposium on Information Visualization, Seattle WA, IEEE Computer Society.

Craig, P., J. B. Kennedy, and Cumming, A. (2002). Towards Visualising Temporal Features in Large Scale Microarray Time-series Data. 6th International Conference on Information Visualisation - IV2002, University of London, London, GB, IEEE Press.

Chapter 1 Introduction

Over the last five years their have been a series of significant advances within the field of genetics. First, initiatives such as those carried out by the Human Genome Project [*1*] involved scientists 'breaking the genetic code' by determining the biological sequence of large proportions of the human genome and identifying units of that sequence as genes. With this structural information known, biologists were then able to develop automated technologies capable of recording the activity of genes over time and in different conditions. Arguably the most significant of these, and certainly the most widely used, are Microarray technologies. These allow biologists to indirectly monitor the activity of large numbers of genes in parallel for different conditions and/or at different points in time. The main challenge for biologists using these technologies is to use the data produced to determine the functions of genes and uncover details of how they interact to realise the higher level functions of an organism. These biologists need to be able to explore their data and, because of its scale and complexity, new techniques are required for them to be able to do so effectively [*2-4*]. This thesis presents the Time-series Explorer, a new technique that supports the exploratory analysis of microarray data and overcomes many of the limitations of established techniques using animated representations of the data.

The genome is the genetic material of an organism and is primarily made up of DNA. DNA exists physically as long strands of nucleotides (chromosome) within the nucleus of cells and the *sequence* of the DNA is the sequence of different nucleotide nitrogen bases which are either adenine (A), thymine (T), guanine (G) or cytosine (C). The sequence of nitrogen bases is identical within every cell of an individual organism, varies little between individuals and is still predominantly the same between species. When the term DNA is related to an individual or grouping of individuals it normally refers to the sequence rather than any physical entity. For many organisms much of this sequence is known and, from rules derived from experimentation, it can be subdivided into genes which are the functional units of physical DNA. While genes indirectly combine to realise a number of higher level biological functions, the underlying function of a gene (and the function that defines a gene as a functional unit) is the production of RNAs. These RNAs are biological

entities that carry the code of the gene from which they where produced and the rate at which RNAs specific to a gene are produced can be used as a measure of that gene's activity.

To determine the functions of genes and uncover details of how they interact to realise the higher level functions of the organism, structural information must be supplemented by information about the activity of genes. Specifically, how that activity can be related to biological processes and how the activity of different genes can be interrelated. This can be achieved, in part, by performing microarray time-course experiments. These automated high-throughput experiments typically involve around 8,000 genes (that's about 40% of all known human genes) with activity recorded over about 20 time points. Figure 1.1 shows an activity against time plot for a single gene.

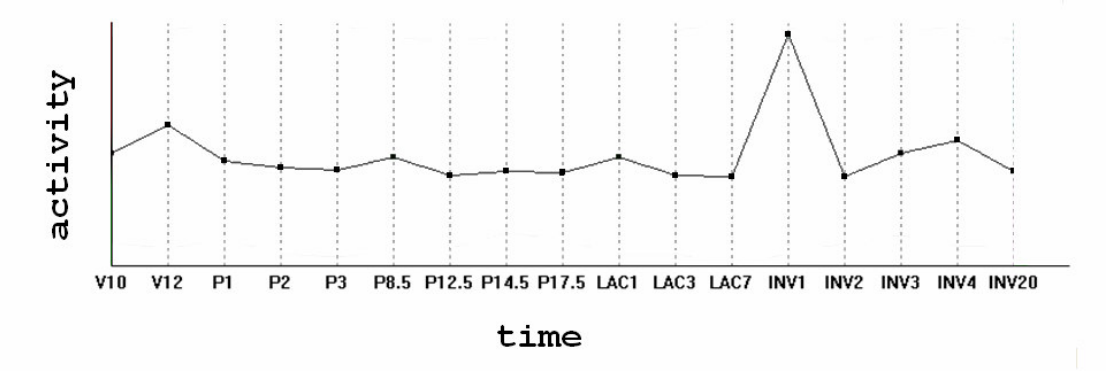

Figure 1.1. ***Microarray time-course data.***

While occasionally the analysis of microarray data involves the combination of data from multiple unrelated experiments where each gene has multiple series of activity levels, this thesis focuses on the analysis of data where each gene is associated with a single series. This data can come from a single experiment or a combination of experiments replicating the same conditions.

1.1 Motivation

The types of knowledge a biologist would wish to extract from microarray time-course data can be summarised by the following high-level biological questions [*5*]:

- What are the functional roles of different genes and in what cellular processes do they participate?
- How is the activity of genes regulated, how do genes and gene products interact, what are these interaction networks?
- How does gene expression differ in various cell types and states, how is gene expression changed by various diseases or compound treatments?

There are two general approaches to answering these questions; hypotheses testing and exploratory analysis. Hypotheses testing uses existing knowledge about the function of small numbers of gene groupings to make assertions about a particular cell sample. Exploratory analysis is more focused on the generation of knowledge relating to gene function. The usefulness of hypothesis testing analysis is obvious for applications such as disease diagnosis where, for example, if a biologist knows a group of genes to be active in a particular disease state and then finds those genes to be active in the cell sample being surveyed, they can presume that the patient from which the cell sample came also has the disease. The results of exploratory analysis are less immediately practical but it is often considered as being a more relevant procedure because the gene function information required to perform hypothesis testing can only be generated by performing exploratory analysis. The relevance of exploratory analysis stems from the fact that there is a significant dearth of functional information with only a small proportion of genes having any known function [*6*]. So, in order for hypotheses testing to be effective, exploratory analysis is required to bolster existing functional information. Another important aspect of exploratory analysis is that it can allow biologists to gain insight into their data by revealing processes they did know of, or even expect, before their analysis began [*7, 8*]. This allows them to increase their knowledge not only of interactions at the genomic level but also of higher level processes which can often be directly related to patterns of gene activity.

Exploratory analysis essentially relies on biologists working with general assumptions about how genes are supposed to interact. The most useful of these assumptions is ‘guilt by association’ [*9*] where common activity is considered to indicate that a group of genes are more likely to participate in the same process and be functionally related. To use guilt by association, a biologist performing

exploratory analysis would want to be able to find patterns of genes with common activity. This could be over a small or large interval of the data and be shared by different numbers of genes. The main factors that might make such a pattern significant are the timing of common activity (with regard to known or suspected processes) and the functional groupings to which they belong. If the activity is over an interval of the data, its timing can be related to processes that are known (or suspected) to occur. This information can be used by a biologist to determine which processes the activity is likely to be related to, toward determining the function of the active genes. The functions of the active genes that are already known and recorded in functional groupings are also useful when trying to determine unknown functions. This is because genes known to one type of function are thought of as being more likely to share other functions and if a gene has multiple functions those functions are often related. The discovery of patterns of activity involving genes with known function can also allow a biologist to become more aware of the general interactions within the cell sample and direct them to focus their analysis onto related groupings or general timings that might lead them to the discovery of further significant patterns.

Existing techniques that support the exploratory analysis of microarray time-course data do so either by clustering representations of genes with similar activity [*2*] or allowing the user to query the data with regard to the existence of specific patterns of activity over intervals of the data [*10-12*]. First, from a clustering view of the data biologists can detect general trends where large numbers of genes have common activity over the entire time-frame. Other patterns where smaller numbers of genes have common activity or common activity is over a smaller interval of the data are not represented. This is essentially because there is such a large quantity of such patterns in the data and it would be impossible to represent all these clearly in a single static overview. In a clustering display more dominant patterns naturally take precedence and the representation of less dominant patterns is diluted so that they cannot be detected. Second, other techniques used for the exploratory analysis of microarray time-course data allow the user to select genes according to the profile of their activity over an interval of the data. These require the biologist to interact with a graphical representation where graphs of gene activity (such as that shown in Figure 1.1) are overlaid onto one another. This interaction allows the user to specify

the form of a pattern of activity over an interval of the data. This could be defined by a range of values [*10*] that activity must lie between, an allowable range for the change in activity [*11*] or some other more sophisticated condition relating to the activity of genes over that interval [*12*]. After the activity form is specified, the display is refreshed with only the genes having that form remaining. This type of interaction allows the user to find various types of patterns that associate genes with the condition that the form of the activity must be known (or guessed) before a relevant pattern can be found. The number of potential pattern forms in the data is, however, still massive and even if information related to the timing of patterns or the types of gene groupings is known, this number is not sufficiently reduced. This makes it impractical to visually sift through the data hoping to find all the relevant patterns and limits the technique according to the number and types of pattern the user attempts to look at. Indeed, the quantity of potentially interesting patterns (particularly those where the common activity of genes is restricted to an interval of the data) is problematic for both types of technique employed for the exploratory analysis of microarray time-course data; both those that use a static overview to convey patterns and those that rely on queries to allow the user to find patterns. The primary motivation of the research described in this thesis is to investigate the feasibility of developing an alternative type of information visualisation technique for the exploratory analysis of microarray time-course data.

1.2 Hypothesis

The technique proposed in this thesis uses animated views of the data to avoid the aforementioned limitations of relying on queries or static overviews. As an animated overview is capable of representing more data by utilising a third display dimension (time), it is conceived that such a view will be capable of representing a larger number of potentially interesting (and unsuspected) patterns therefore increasing the users capacity to extract relevant knowledge from the data [*13*].

Another predicted advantage of using this type of representation for microarray time-course data is that, as the data has an intrinsic temporal quality (being as it is *time*-course data), time in the data can be logically mapped to time in the display to reduce the distance between the user's conceptualisation of the data and its representation. A less obvious advantage is that when observing an animation the user is compelled to

utilise their low-level visual system (known as pre-attentive perception) and can rapidly perceive complex patterns from motion [*14*] as well as those that might be represented in individual frames of the animation.

There are also a number of potential drawbacks associated with developing an animated representation for microarray time-course data. The foremost of these is the fact that the rate at which information can be committed to longer term memory is far less than the rate at which it can be displayed [*15*]. While the human eye is capable of absorbing information at a rate of approximately 10,000,000 bits per second, information can only be transferred to long term memory at a rate of 40 bits per second. It is, however, conceived that this limitation can be avoided by giving the user tight control over the rate and direction of the animation allowing them to play, fast forward, rewind, slow motion, pause and stop the animation to examine, and re-examine, the more interesting features in the data as and when required.

Other potential pitfalls are that animations are computationally expensive and the way in which people perceive motion is more difficult to predict with the potential for undesirable effects such as undue perception of structure from motion or disorientation. The issue of computational expense, which might make an animation too slow to have the proper effect, is the lesser of these issues. Due to the increased performance of relatively inexpensive computer hardware, an effective animation of microarray time-course data should be feasible as long as the software developed to generate such an animation is coded to take advantage of hardware performance. As effects caused by how people perceive animations are largely unpredictable, these are more problematic and need to be dealt with as they occur. This makes it preferable to develop the software iteratively so that successive prototypes can be evaluated and amended to remove such unwanted artefacts before the development of a final version.

With specific relation to the research undertaken for this thesis to develop a new type of animated technique, there have already been various related studies performed concerning the use of animation in information visualisation. These include studies of:

a) The general potential of animation as an extra display dimension to increase the amount of information that can be displayed in a visualisation [*13*],
b) Animation used expressively to draw a users attention to specific areas of a visualisation [*14*].
c) The use of animations to convey structure from motion in three dimensional mappings of abstract data [*16*].
d) The use of animation to switch between different representations of abstract data [*17-19*].
e) Animation being used to convey changes in values over time in relation to spatial attributes (e.g. animated maps) [*20, 21*].

There are, however, no equivalent studies of the use of animation to convey time variant attribute values for abstract data with no spatial quality.

1.3 Contribution to Knowledge

This thesis describes two main contributions to knowledge. These are:

a) The Time-series Explorer [*22, 23*]: A new technique for the exploratory analysis of microarray time-course data which overcomes the limitations of established techniques by allowing the user to find and investigate more (less dominant though potentially significant) patterns in their data [*22*].
b) The results of a general study of the issues relating to how animation can be used to assist the exploratory analysis of large scale data with an intrinsic temporal quality.

The thesis also describes some minor contributions more specifically related to the problems associated with the development of the animated technique for this particular type of data. These include:

a) A method for displaying the activity of large number of genes over intervals of the data, rescaling the data and distorting the display so that patterns of activity can be detected [*22*] (see Section 8.2).

b) A colour mapping to allow the user to interpret the density of large numbers of overlaid gene representations in a scatter-plot and graph views while still being able to detect outliers [*22*] (see Section 8.3).
c) A complementary change-in-activity graph view that allows the user to detect more subtle patterns of activity within a selected subset of genes from the original data [*23*] (see Section 8.4).

Of these contributions, the first has the potential to be generalised for similar temporal data where values have a log-normal distribution. The third might be generalised for use with other time-series data and the second has the potential to be generalised for any other visual display where there are a high density of overlaid item representations and the user requires to observe the general distribution of values as well as outliers.

Chapter 2 Information Visualisation

The field of information visualisation is primarily concerned with the study and design of techniques that use interactive graphical interfaces to make large or complex datasets more accessible. One generally accepted definition of information visualisation is;

> *The use of computer-supported, interactive, visual representations of abstract data to amplify cognition* [*24*].

Here, cognition is the mental faculty or process of acquiring knowledge by the use of reasoning, intuition or perception [*25*]. Information visualisation applications (often just called information visualisations) are basically interactive computer-supported representations of data designed to help users gain knowledge from that data. The primary advantage of information visualisations over other analysis techniques is that they are able to exploit the capacity of humans to rapidly interpret large amounts of visually presented information to extract knowledge from that data.

2.1 Origins and Related Fields

Information visualisation research has its origins in the more established fields of presentation graphics and scientific visualisation, starting to become a distinct field of research in the early 1990s [*26*]. While information visualisations and presentation graphics are both essentially visual representations of information, there are some fundamental differences in the type of information each type of graphic is intended to convey. Presentation graphics tend to use a representation of data to illustrate a point already known by the graphic designer. Normally this point will be outlined in accompanying text. Information visualisations are designed to uncover unknown phenomena. In a more practical regard, presentation graphics tend to be static and are normally in printed format while information visualisations are computerised and interactive. Regardless, as both types of graphic essentially involve the visual presentation of information, many of the principles applied to presentation graphics also apply to the design of information visualisations. Indeed, persons involved in the study of information visualisation often draw inspiration from classic presentation

graphics such as Charles Minard's illustration of Napoleon's Russian campaign (Figure 2.1) or the London underground map (Figure 2.2).

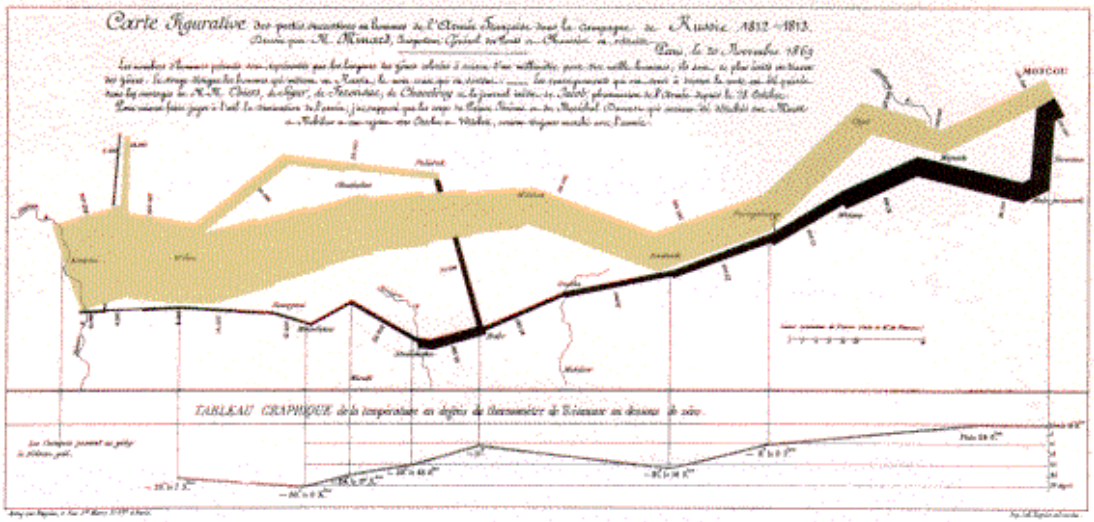

Figure 2.1. ***Charles Minard's illustration of Napoleon's Russian campaign: The graph shows the size of the army by the width of the band across the map of the campaign on its outward and return legs, with temperature on the retreat shown on the line graph at the bottom (to be found in [27]).***

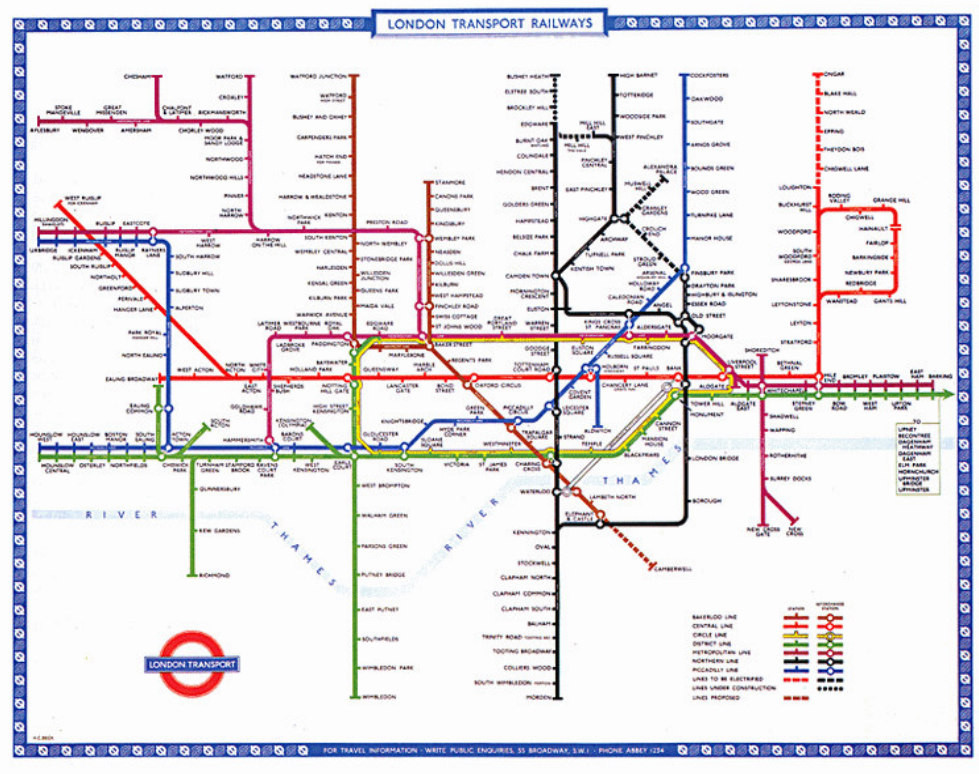

Figure 2.2. ***The London underground map: Geographical perspective is disregarded in favour of graphic order and usability (to be found in [28]).***

The differences between information visualisation and scientific visualisation are more subtle. The data represented in a scientific visualisation tends to have some relation to physical space that is directly represented in a view of the data. In information visualisation the data visualised tends to be of a more abstract nature and not related to physical space [*24*]. Another distinction is that the users of scientific visualisations tend to be application area specialists and the knowledge they wish to extract from such data tends to be more specific and related to a small number of well established objectives. The users of information visualisation systems tend to be less specialised and have more general objectives such as browsing or exploring their data [*29*]. Despite these differences, there is a lot of common ground between the two fields (harnessing human perception, supporting interaction etc) and they often come under the shared banner of *visualisation* or *data visualisation*.

Other research areas associated with information visualisation are human computer interaction (HCI) and branches of psychology concerned with the study of human perception. As HCI is concerned with the study and design of human-centric interactive computer systems, and information visualisations are a specific type of such systems, information visualisation can be considered as something of a subfield of HCI (although information visualisation has developed as a strong field in its own right with its own journals and conferences). The relationship between information visualisation and the study of human perception is that information visualisation relies on exploiting the properties of human perception and an understanding of those properties is therefore fundamental to the effectiveness of any information visualisation technique. This is also the case with scientific visualisation and presentation graphics. In general information visualisation tends to borrow ideas and concepts from its fields of origin and other related disciplines when and wherever it is appropriate to the technique being developed or studied.

Other fields that have developed an association with information visualisation include those that employ information visualisation techniques to analyse specific types of data. These include Bioinformatics which is concerned with the analysis of large-scale biological data, the study of graphical information systems (GIS) for geographic data and data mining dealing with the extraction of information from

large databases having the subfield of visual data mining specialised toward visual methods.

2.2 Models for Information Visualisation

As stated above, the process of information visualisation can be described as the use of interactive visual representations of abstract data to amplify cognition. In order to design an application or technique to support this process one must consider two things. How visual representations can be generated from the raw data and how those visual representations can be used in the process of knowledge extraction to amplify cognition. The first of these issues can be dealt with by considering the *reference model for visualisation* [*24*] which generalises information visualisation applications as a series of mappings from raw data to visual form with automatic or user controlled transformations between mappings. Each mapping and transformation within this basic reference model can be further analysed toward a more complete understanding of the design of information visualisation systems.

The knowledge crystallisation task model [*24*] on the other hand is a reference model which compliments the reference model for information visualisation by dealing with the issues of how knowledge might be extracted from data to amplify cognition. Here, the process of knowledge extraction is broken down into knowledge crystallisation tasks. These are characterised as involving large amounts of information, ill-structured problem solving and well-defined goals. Visual representations are potentially valuable at various stages of such a task and a view of how they fit into the greater process of knowledge extraction helps indicate how an information visualisation application might best be configured to amplify cognition.

2.2.1 The Reference Model for Visualisation: Interactive Mappings from Data to Visual Form.

Card et al. introduced their *reference model for visualisation* [*24*] (Figure 2.3) to establish a formal way of making structured visualisations of data. The model essentially generalises the transformations from raw data to visual forms in an information visualisation application.

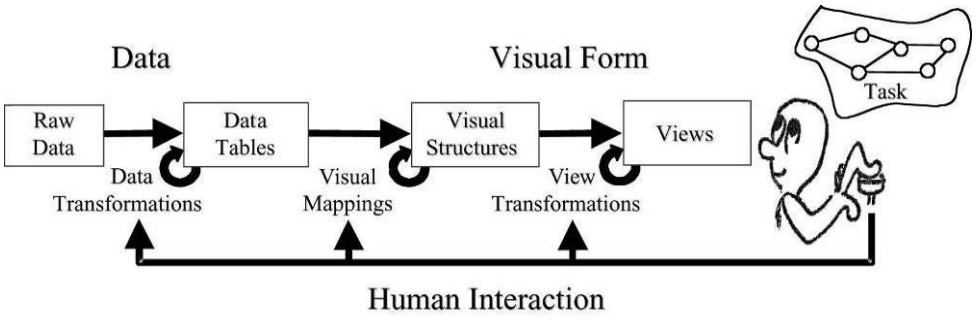

Figure 2.3. ***Reference model for visualisation (taken from [24]).***

In the reference model for visualisation *raw data* are in an idiosyncratic format peculiar to the data source, have no spatial arrangements and no obvious relationship to one another. *Data tables* are categorised data put into tables of the form shown in Table 2.1. For microarray time-course data, the cases of data tables would correspond to time-points, variables to genes and values to gene activity levels at time-points.

Table 2.1. Form of a data table in the reference model for visualisation (taken from [*24*]).

	Case$_i$	**Case$_j$**	**Case$_k$**	**…**
Variable$_x$	Value$_{ix}$	Value$_{jx}$	Value$_{kx}$	…
Variable$_y$	Value$_{iy}$	Value$_{jy}$	Value$_{ky}$	…
…	…	…	…	…

The next stage in the framework for information visualisation indicates a mapping from data-tables to visual structures. Bertin [*30*] states that these visual structures can be generated from three basic components; a spatial substrate, marks and the marks' graphical properties. The spatial substrate is the space formed by the selection of axes for display. It is possible to combine up to three dimensions in a spatial substrate and there are four elementary types [*24*]:

- **Unstructured -** No axes;
- **Nominal -** Region is divided into nominal sub-regions;
- **Ordinal -** Region is divided into ordered sub-regions;
- **Quantitative -** The region is subdivided based on a metric.

Spatial substrate is often highlighted as the most fundamental aspect of visual mapping due to the perceptual dominance of space. Other features of the spatial substrate involve the interaction of axes. Different axes configurations are:

- **Composition -** The orthogonal placement of axes (e.g. a scatter plot).
- **Alignment -** The repetition of an axis at a different position in space (e.g. scatter-plot matrices [*31*]).
- **Folding -** The continuation of an axis in an orthogonal dimension (e.g. wrap around displays and SeeSoft [*32*]).
- **Recursion -** A repeated subdivision of space (e.g. scatter-plot matrices [*31*] or interactive zooming [*33*]).
- **Overloading -** Reuse of the same space for the same data table (e.g. worlds within worlds [*34*]).

While spatial substrate defines the space of the display, marks are visible things that occur in that space. There are four elementary types of marks, these are: points (0 dimensional marks), lines (1 dimensional), areas (2 dimensional) and volumes (3 dimensional)[1].

The graphical properties of marks are defined by visual variables. These variables include position and a set of *retinal* variables, so-called due to their association with features of visual space to which the human retina is particularly sensitive. Each visual variable can be characterized according to their relationship with five different types of perception. These are:

- **Selective -** Can a change in this variable make it easier to select the changed mark among other unchanged marks?
- **Associative -** Can marks that have like values of this variable be grouped in the presence of other marks with different values?
- **Quantitative -** Can a quantitative value be read from this variable?
- **Order -** Do changes in this variable support ordering of the data?

[1] In graphics, unlike mathematics, points and lines take up space and become visible entities.

- **Length -** How many values of this variable can be used without losing its functionality?

An original list of seven visual variables were proposed by Bertin in 1983 [*30*]. This list was later extended [*35*] adding variables which are especially relevant to interactive visualisations. Table 2.2 describes the extended list of visual variables with regard to the types of perception listed above.

Table 2.2. Visual Variables (taken from [*35*])

Visual Variable	Selective	Associative	Quantitative	Order	Length
Position	Yes	Yes	Yes	Yes	Dependent on resolution
Size	Yes	Yes	Approximate	Yes	Association: 5; Distinction: 20;
Shape	With effort	With effort	No	No	Theoretically Infinite
Value (brightness)	Yes	Yes	No	Yes	Association: 7; Distinction: 10;
Hue (colour)	Yes	Yes	No	No	Association: 7; Distinction: 10;
Orientation	Yes	Yes	No	No	4
Grain	Yes	Yes	No	No	5
Texture	Yes	Yes	No	No	Theoretically Infinite
Motion	Yes	Yes	No	Yes	Unknown

There are numerous options for mapping data tables to visual structures. An effective mapping can be considered as one which is faster to interpret, conveys more distinctions, or leads to fewer errors than some other mapping. Ultimately, these metrics for the quality of a visual mapping are dependent on the underlying properties of human perception and are dependent on the ability of marks to be perceived in different ways.

The visual variable *position* is unique in that a mark can have more than one value for that variable (ordinarily two for a two dimensional display). All other variables, except motion, are retinal and the retina of the eye is sensitive to them regardless of their position. A mark may also have a temporal property where there is a variation in any visual variable over time. Motion can be described as such a temporal property where the variable changing over time is position.

As each mark of a static view is limited by the range of visual variables and static views are limited in the amount of information they can convey, view transformations changing the spatial substrate of a display according to user interaction are often needed to allow the user to extract the required amount of information from their data. In the reference model for visualisation view transformations transform visual structures into views. In information visualisation there are three common types of view transformation [*24*], these are:

- **Location Probes -** Transformations that show details (often, but not always, in a separate window) of the data set at specific points in the visual structure chosen by the user. These transformations are often described as details-on-demand [*36*].
- **Viewpoint Controls -** Transformations that zoom, pan and clip the viewpoint.
- **Distortions -** Transformations that distort the spatial substrate. Often so that a detail view and overview are combined in the same space, creating a so-called focus + context view [*37*].

View transformations can be either instantaneous or animated to smooth the transition from one view to another. Typically transformations such as zooming, panning and rotation around a three-dimensional visual structure are animated to allow users to track objects between views and relate between alternate views without becoming disorientated. An animation where time is seen to pass can also be considered as a type of view transformation where the viewpoint is moved along a time axis and can be animated for a similar effect.

Visual mapping transformations can also be controlled by the user in an information visualisation application. Here the mapping from data tables to visual structures can be changed with the possibility of a new configuration better exploiting the properties of human perception and allowing the user to extract more relevant knowledge from the data. This might involve changing the spatial substrate of the display by allowing the user to assign different variables to its axes or changing the properties that are conveyed by the retinal variables of marks. Interactive *data* transformations are also possible within the framework for information visualisation.

These adjust the way in which raw data is transformed into data tables before they are visualised. This could involve a change in the values from the raw data that are included in the data table or the way in which those values are transformed to derive values for the data table.

2.2.2 The Knowledge Crystallisation Task Model

While the reference model for visualisation can be used to describe the form of an information visualisation application, the knowledge crystallisation task model [*24*] (Figure 2.4) acts as a reference model for the general process of knowledge extraction as it is likely to be supported by such an application. Here, the process is deconstructed into knowledge crystallisation tasks. These are characterised by large amounts of information, ill-structured problem solving and a relatively well-defined goal. During each task the user searches for a schema within some defined area of the data. This is essentially a representation (pattern, cluster etc) with the potential to help the user to solve some problem or define another schema. Once an appropriate schema is found (i.e. one that allows the user to solve the problem associated with their task), the user can use the knowledge gained either by extracting their results to share them with other users or using the result to make a decision or initiate some other action.

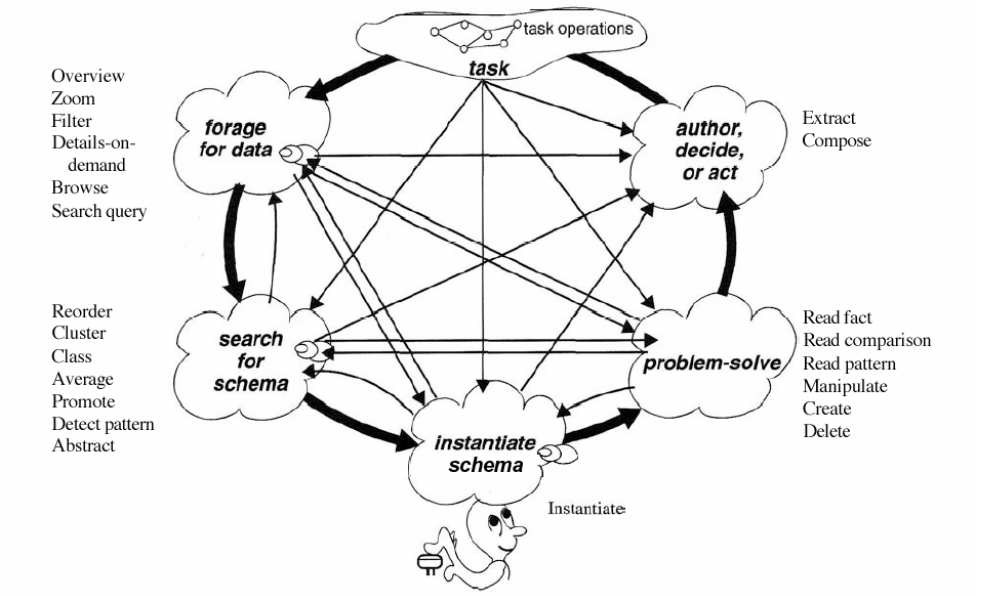

Figure 2.4. ***The knowledge crystallisation task model with subtasks (italics) and suggested roles for information visualisation (far LHS and RHS). Diagram taken from [24].***

An interesting feature of the knowledge crystallisation model is that it represents an iterative process of data exploration. This would occur if the outcome of problem solving was an action or decision that generates a new task. The knowledge crystallisation model also reflects the fact that a visual representation might not be suitable to solve a particular problem and the user may require to configure an alternative representation of their data.

Visual representations of data are involved at different stages of a knowledge crystallisation task. Firstly, they can aid the search for data at the 'forage for data' step by providing an overview of the data and allowing the user to interact with that representation by zooming, filtering or otherwise restricting the data displayed. Next, a visual representation can allow the user to search for a schema within the data by detecting patterns or clusters within that representation. If the visual representation of the data contains no schema then a user might return to the 'forage for data' stage. If a schema is found then it can be used to problem solve. This might involve foraging for additional information and generating an alternative visual representation for a better schema (for example, labelling the original pattern found). Once the problem is solved the user can author, decide or act. This might involve exporting their findings to share with other users or deciding that another knowledge crystallisation task is required in order to satisfy their wider objectives.

The process of knowledge crystallisation has three requirements; a well-defined goal, data and a schema. If this process is to be supported in an information visualisation application there are also a number of requirements for such an application. The first of these is for the designer of such an application to characterise the user's objectives by breaking them down into well-defined task types. Also, a mechanism must be put in place so that the user can forage for the data defining the types of area of the data that they are likely to want to consider for any given task. In addition to this an appropriate visual representation (or group of representations) must be configured so that the user is capable of finding a schema within the foraged data. If no schema can be found the user must be able to return to the 'forage for data' subtask. The type of schema the representation allows the user to find must be related to the user tasks so that they can solve the required problem and there also must be a mechanism in place so that the user can author, form a decision or act on their results. In cases where a

user is likely to require iterative exploration of the data it would also be convenient if processes could be linked seamlessly so if a decision at the end of a task generates another task the appropriate options are presented.

2.3 Guidelines for Information Visualisation Design

There are numerous methodologies and guidelines available to assist in the design and development of information visualisation systems. These can be roughly divided into methodologies concerned with development of a system (eliciting user requirements, prototypes, evaluation etc), guidelines concerned with interface design and guidelines concerned with the individual views. Here, the methodologies concerned with development of a system tend to be related to the development of computer applications in general rather than specifically relating to information visualisation systems. Guidelines for interface design focus on how the user interacts with such an interface. Guidelines concerned with individual views are often adapted from principles for designing presentation graphics or theories of human perception.

2.3.1 Methodologies for Application Development

Methodologies for developing software applications tend to focus on the type of objectives of the user rather than the form the software should take. As the type of tasks information visualisations are generally designed to support are of an exploratory nature, for the sake of adopting a software design methodology, information visualisations can be classed as exploratory systems [*38*]. A feature of this type of system is that the details of user requirements are likely to change during the process of software development [*39*]. This makes it appropriate to adopt an iterative strategy for the development of information visualisation systems [*39, 40 41*]. The prototype life-cycle model (Figure 2.5) fits this strategy allowing for each prototype to progressively refine the user's initial requirements. To save time, initial prototypes may consist of pen and paper mock ups with later computer prototypes integrating more functionality as the user requirements take form.

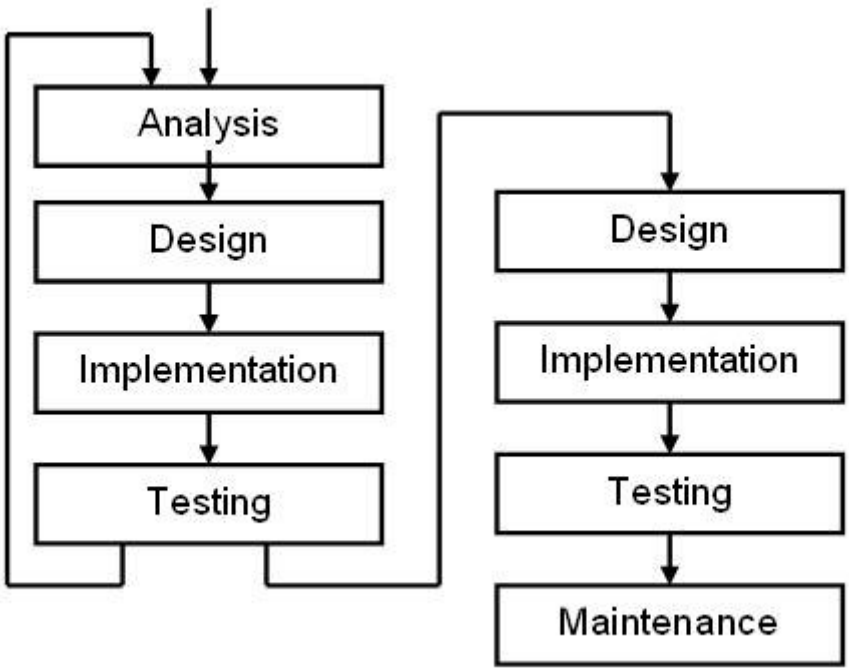

Figure 2.5. ***The prototype life cycle model***

Rose et al. also prescribe an iterative life-cycle model for the development of information visualisation systems and demonstrate the effectiveness of such a strategy with regard to the development of a number of information visualisation applications [*40*, *42*]. Figure 2.6 illustrates their particular model, which specifies that prototypes should be generalised so that other researchers can benefit from any progress made. A potential limitation of this approach is that the path from the initial problem statement to the development of prototypes is unidirectional indicating that the problem will be fixed and user requirements cannot change. This approach has the potential to work well for simple problems that are unlikely to change. The model is somewhat less appropriate when user requirements are less well defined and likely to evolve while prototypes are being developed.

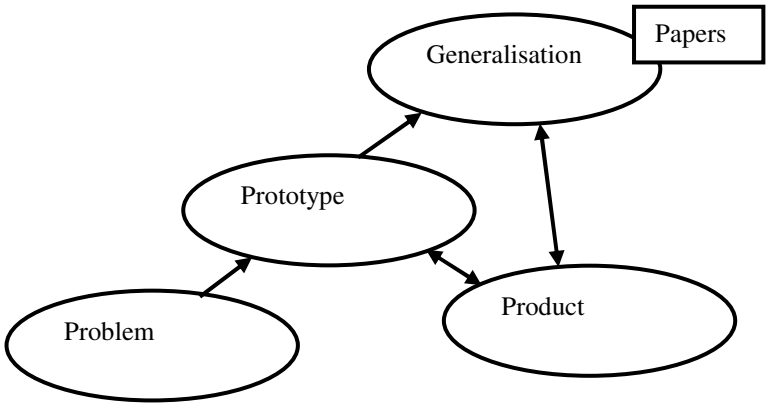

Figure 2.6. ***Rose et al.'s iterative model for developing and generalising visual interfaces (taken from [43]).***

The nature of user requirements for an information visualisation is also problematic when considering the evaluation of information visualisation systems. While there are many methodologies designed to test the usability of an interface [*44, 45*], tests to determine how well users can achieve specific well defined tasks [*46, 47*] and metrics for the effectiveness and expressiveness of displays [*27, 48*], none of these can be used to completely determine the actual effectiveness of an interface for extracting relevant knowledge from the data. Indeed, many of the measures involved in other types of testing may yield results that are misleading when one tries to assess the ability of an interface to support exploration. For example, a user may actually take longer to execute a particular task while using an effective interface because an interesting pattern has made them stop and think to translate the information into knowledge [*49*]. So, when evaluating the quality of an interface for exploratory analysis, it is necessary for a developer to consider more abstract concepts such as insight [*50*] which can only be quantified by the actual real-world users of the interface.

2.3.2 Guidelines for Interface Design

By far the most widely used of set of guidelines for the design of information visualisation interfaces is the *Visual Information-Seeking Mantra* [*36*]. The mantra summarizes several design guidelines as:

Overview first, zoom and filter, then details on demand.

The mantra is supported by task by data type taxonomy compiled to guide researchers to new opportunities and help categorize prototypes. The taxonomy contains seven low-level tasks and seven common data types. The tasks are:

- **Overview** - Gain an overview of the entire collection.
- **Zoom** - Zoom in on items of interest.
- **Filter** - Filter out un-interesting items.
- **Details-on-demand** - Select an item or group and get details when needed.
- **Relate** - View relationships among items.
- **History** - Keep a history of actions to support undo, replay and progressive refinement.
- **Extract** - Allow extraction of sub-collections and of the query parameters.

Data types described in the task by data type taxonomy are:

- **1 dimensional** - Including textual documents, alphabetical lists and other things that are organised in a sequential order.
- **2 dimensional** - Planar or map data, e.g. geographical maps or floor plans.
- **3 dimensional** - Including real world objects such as molecules, the human body and buildings. Abstract data can also be 3-dimensional if the application designer chooses to organise the data into a 3-dimensional structure. An example of a 3-dimensional structure constructed from abstract data is the information landscape generated from the Themescape application [*51*].
- **Temporal** - Relating to entities with start and end times such as processes or historical events.
- **Multi-dimensional** – A data collection with *n* (more than three) variables. Multi-dimensional-data include relational and statistical databases.
- **Tree structures** - Collections of data with each item is capable of having a link to one parent item.

- **Network structures** – Items are linked to an arbitrary number of other items.

Another set of guidelines applicable to the design of information visualisation interfaces are the principles of *direct manipulation* [*52*]. These principles are concerned with the level of control a user has with objects represented by the computer. They tell us that an interface should include the following features:

- **Visual representations of the world in action including both actions and objects.**
- **Rapid, incremental and reversible actions.**
- **Selection by pointing (not typing).**
- **Immediate and continuous display of results**

Overall direct manipulation is thought of as giving the user the feeling of being directly engaged with control of the objects rather than with the program or a computer [*38*]. This acts towards removing any consideration of how the computer should operate and allows the user to focus on their specific domain-level objectives.

Tight coupling [*33*] is a principle concerned with making interfaces more usable by maximising the relevant information displayed at any point in time, guiding users toward useful interactions and discouraging or preventing users from performing actions that are not sensible (such as performing a query that returns no items). Some important aspects of tight coupling are:

- **Comprehensible consistent affordances to guide users**: For example, if aspects of a query result could possibly be used to formulate a further query then that follow-on query should be possible. This is known as the principle of output-is-input.
- **Rapid, incremental, and reversible interactions among components**: The values permitted for selection from components are continuously updated to avoid the user being able to perform selections that return no items.

- **Constraint on permissible operations to preserve display invariants and prevent errors:** For example, a scroll bar thumb position must track the position in the document.
- **Progressive refinement:** Users can incrementally adjust query parameters after they are set so they can reformulate their goals in order to seek a more restrictive (more specific) result.
- **Allow users to select details-on-demand:** Reduces clutter while a query is being formulated and allows access to supplementary information as-and-when required.

It is interesting to note that various aspects of tight coupling, principles of direct manipulation and components of the information seeking mantra are closely related. For example, details-on-demand are integral to both the information seeking mantra and the principle of direct manipulation. Indeed, it is often the case that these three guidelines go hand in hand to guide the development of information visualisation applications. The information seeking mantra describes a framework for the flow of user interaction, the rules of direct manipulation describe how the user should be given control of an interface and the principle of tight coupling helps a designer to increase its usability.

2.3.3 Guidelines for Data Representations

Guidelines for the design of individual views within an information visualisation interface tend to have their origins in the field of presentation graphics and/or be derived from findings related to the study of human perception. One such set of guidelines prescribed by Tufte [*27*] for presentation graphics states that "an interface designer should aim to maximise the information density of the display" and that "the data graphic should draw the viewer's attention to the sense and substance of the of the data, not something else". The first of these statements conflicts with the *information seeking mantra* where it is suggested that details should be available to the user only when they are asked for. This conflict is likely to be due to the fact that visual interfaces can be interactive while presentations graphics (for which the guideline is prescribed) are largely static requiring data to be presented in the overview or not at all. The second statement, however, can be seen as applying

equally to presentation graphics and visualisation interfaces. The five principles associated with this proposal are:

1. **Above all else show the data.**

   Maximise the data-ink ratio (for computer graphics data-ink translates to salient pixels)
2. **Erase non-ink-data**
3. **Erase redundant data-ink**
4. **Revise and edit**

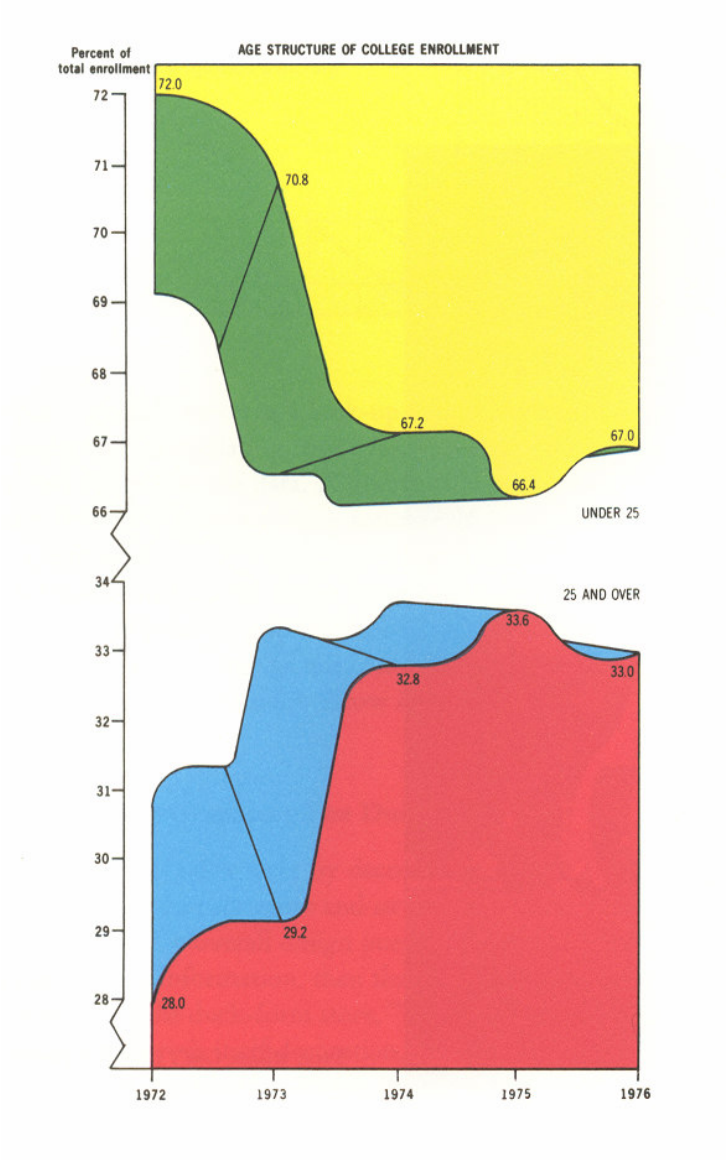

Figure 2.7. ***Chartjunk, reproduced in [27]***

Those items that do not add value to the representation are described using the derogatory term "chartjunk". Figure 2.7 is an example of a graphic with excessive chartjunk. In this display graphic there are actually only five values being displayed but the chartjunk (the curvature of the line between values, the 3d raised effect and a mirror image of the plot) may actually mislead the user into thinking more is being displayed. The notion of chartjunk is also addressed by Bertin who states that "any non-differential element is useless and reduces the visibility of the image" [*30*]. Bertin and Tufte also advise strongly against the overuse of colour (also apparent in Figure 2.7), which can overload the user's capacity to interpret their meaning or mislead users into assuming unordered data is ordered.

Other guidelines applicable to visualisation interface design focus more on the immediate usability of an interface. Here, Mullet and Sano [*53*], for instance, stress the importance of elegance and simplicity. Simple visualisations are said to benefit the user by being approachable, recognisable, immediate and usable. The goal of simplicity is said to be one of achieving maximum meaning with minimum means. Tufte also discusses the element of elegancy [*27*] described as "simplicity of design and complexity of data" stating that graphical elements should carry information as well as perform a design. To accomplish elegant and simple design, Mullet and Sano [*53*] describe three basic techniques. These are:

- **Reducing:** A design should be reduced to its essential elements and each element should be reduced to its essential form.
- **Regularization:** This involves the reduction of information by repeating elements in order for human perception to operate more *efficiently*.
- **Leverage:** This involves combining elements wherever a single graphic element can do the job of two.

Another important aspect of immediate usability of a visual interface is its *attractiveness*. Norman [*54*] claims that the positive affect of an attractive design makes people more tolerant of minor difficulties, and more flexible and creative in finding solutions. For many people elegancy and simplicity are linked with attractiveness. Other aspects of a display that can be linked with attractiveness are organization and balance [*53*]. The organization of a visual interface depends largely

on the relationship between visual elements and is not only important from a purely aesthetic point of view but also as a means of communicating information [*53*].

Numerous guidelines for visual interface design stress the importance of information drawn from the overall relationship of the entire set of visual objects [*30, 38, 53*]. Here, scale contrast, and proportion should be adjusted to make relationships appropriate. Gestalt theory states that patterns take precedence over elements and have properties that are not inherent in the elements themselves. This idea is expanded using the Gestalt Principles of Perceptual Organization (in [*55*]). The most important of these principles, with regard to information visualisation, are those which allow visual items to be grouped. These are the proximity principle, the similarity principle and the common fate principle.

The proximity principle is concerned with the visual variable *position* stating, rather obviously, that features close together are associated. For example, in Figure 2.8 on the left-hand-side dots in the same row are closer together and likely to be associated. On the right hand side dots in the same column are closer together and likely to be associated. The similarity principle rules that similar items connect or group. Here, similarity covers retinal visual variables such as size, shape and hue. An example of the similarity grouping principle is shown in Figure 2.9 where columns contain a similar shape of item and are likely to be associated. Another example (Figure 2.10) shows the similarity principle applied twice to the same items where columns are grouped by shape and rows are grouped by colour. The common fate principle is concerned with the movement of items, ruling that items moving together (in the same direction) are associated.

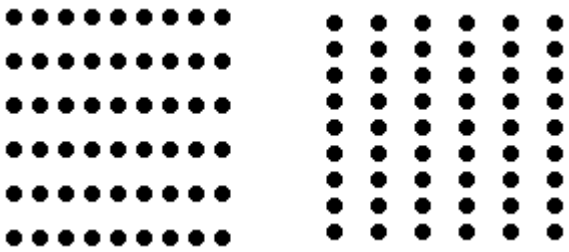

Figure 2.8. ***Proximity principle used to group rows (LHS) and columns (RHS).***

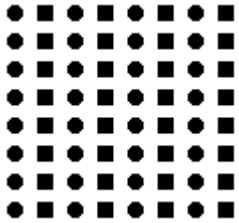

Figure 2.9. ***Similarity principle used to group items in the same column by their shape***

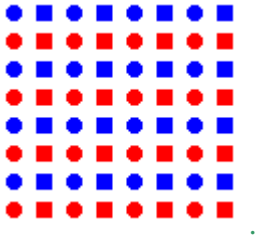

Figure 2.10. ***Similarity principle used to group items in the same column by shape and rows by colour***

Other gestalt principles that are not concerned with grouping are also worth considering when designing an information visualisation interface. While the utility of these principles is not as obvious as those involved in grouping items, they can be used in information visualisations to convey different types of (i.e. non-grouping) information or organize visual elements. Moreover, if these principles are ignored there is a greater likelihood that they might have a negative effect on the visualisation.

The non-grouping gestalt principles are the area (or smallness) principle, the closure principle, the continuity principle and the Pragnänz (simplicity) principle. The area principle rules that smaller areas tend to be seen as figures against a larger background. For example, in the Figure 2.11 we are more likely to see a black cross rather than a white cross within the circle. The closure principle denotes that whole images can be made from partial visual data. For example, we see a circular figure with small gaps in it as a full or closed circle (Figure 2.12). Similarly, we perceive a whole figure even if a part of the image of the figure falls on the blind spot of the retina. The continuity principle describes a tendency for smooth contours to dominate irregular, abruptly changing contours. For example, in Figure 2.13 we would tend to identify lines a-b and c-d than a-c and d-b or a-d and c-b. Notably this principle has been employed in information visualisation to avoid the ambiguity of

crossing lines by curving the lines when they would normally be straight [*56*]. The Pragnänz (simplicity) principle describes our instinctive preference for the simplest, the most elegant and most stable interpretation. For example, in Figure 2.13 there are seven triangles, one square, one rectangle, plus several polygons. But the whole image is more likely to be perceived as the overlapping of one square and one triangle. This is because, as the Simplicity Principle states, it is the simplest and most stable interpretation.

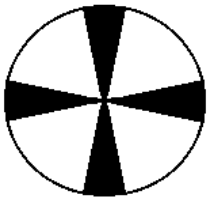

Figure 2.11. ***Area (Smallness) principle: Smaller areas tend to be seen as figures against a larger background.***

Figure 2.12. ***Closure principle: The whole image of a circle can be made from partial visual data.***

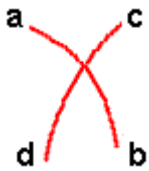

Figure 2.13. ***Continuity principle: Smooth contours (a-b and c-d) dominate irregular, abruptly changing contours.***

Figure 2.14. ***Pragnänz / Simplicity Principle: Humans have an instinctive preference for the simplest, the most elegant and most stable interpretation with the figure appearing as an overlapping triangle and square rather than any other combination of shapes.***

Information visualisations may employ the combinations of different gestalt grouping principles or, as in the case of Figure 2.10, different variations of the same grouping principle to create the impression of overlapping groupings within a set of items. Here, the relative dominance of groupings in such combinations is related to the perceptual qualities of the particular visual variables involved. For, example in Figure 2.10 it can be seen that it is easier to group rows by colour than group columns by shape. This is because humans are more sensitive to changes in colour-hue than to changes in shape. In other terms, colour-hue is more *perceptually dominant* or effective than shape.

While Bertin describes the expressiveness of visual variables [*30*] in absolute terms by noting the types of information that can be encoded into each variable (see Table 2.2), the actual effectiveness of visual variables is somewhat more difficult to quantify. MacKinley [*57*] describes effectiveness as the degree of accuracy with which people accomplish the perceptual tasks associated with the interpretation of graphical presentations (visual variables) and orders tasks according to their effectiveness for communicating three different types of information; quantitative, ordinal and nominal (Figure 2.15).

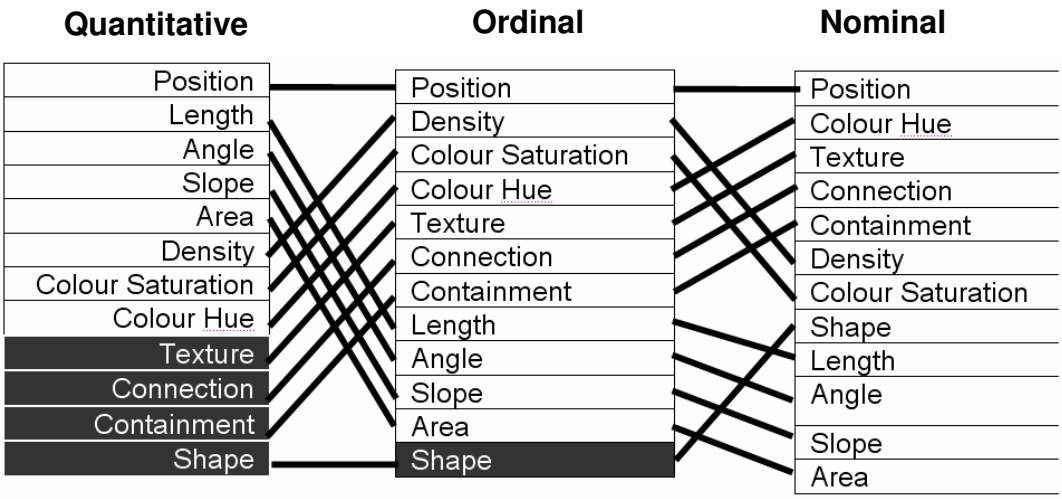

Figure 2.15. ***Ranking of perceptual task by effectiveness. Columns are for different types of information and tasks higher in the chart are perceived more accurately than those lower in the chart. Tasks shown in grey are not relevant to that columns type of information and bold lines are used to link tasks using the same graphical property for different types of information (reproduced from [57]).***

Other guidelines applicable to information visualisation are concerned with the immediacy of different graphical properties. Here, certain properties can be thought of as activating a low-level visual system, known as pre-attentive perception [*58*], which is capable of recognising features of a display in less than 200 or 250 and typically around 100 milliseconds. Properties cited as being capable of triggering pre-attentive perception include position, size, colour-hue, angle and motion (for a more complete list see [*59*]) with position, colour and motion being particularly salient. Pre-attentive response theory can be used to explain why, for example, in Figure 2.10, it is easier to group by colour, which triggers pre-attentive perception, than properties of the shapes, which do not.

The information that can be received through pre-attentive perception is nominal, rather that ordinal or quantitative, and allows users to associate items rather than compare their values [*58*]. This means that the types of visual information seeking tasks that can be supported by exploiting this type of perception are those associated with binary or nominal qualities of the data. These include target detection, boundary detection, region tracking, counting and estimation. Target detection involves users detecting the presence or absence of an item within a field of distracters, boundary detection involves the detection of a boundary between items of different types and region tracking is the tracking of one or more unique items as they move through space. Counting and estimation involve counting or estimating the number of unique items.

The number of item attribute values which can be communicated pre-attentively depends on the types of graphic property used. The number of graphic properties that can be used together to effectively exploit pre-attentive processing depends on the degree to which those properties can be separated within the part of the brain responsible for such perception. The study of these factors is known as *feature integration theory* [*60*]. Here mutually exclusive properties for a single stimulus dimension are known as dimensions and separable pre-attentive dimensions that can be used in parallel are;

- Colours and sufficiently distinct different levels of contrast.
- Line curvature

- Line tilt or misalignment
- Quantitative values such as length, number or proximity
- Terminators and closure
- Direction of movement and stereoscopic disparity

These different types of graphical property can be thought of as using different perceptual channels exploiting pre-attentive processing across these channels almost simultaneously [*60*].

To compliment theories of pre-attentive perception, theories of post-attentive processing [*15*] deal with the persistence of visual representations by considering what happens when the user stops looking at the graphic and starts looking at something else. While intuition might lead us to the belief that we accumulate information from visual representations as they are studied over time, theories of post-attentive vision tell us that this is not the case. This is explained by describing relationship between vision and memory. A visual impression only impacts on short-term memory, when the stimulus (image) is removed the memory is erased almost immediately. It is only when items are committed to long-term memory (an item at a time) that information can be retrieved by the user for further use after the stimulus has been removed. So, while the user may be convinced that they remember an entire scene (envisaging themselves as "taking a mental photograph"), they are indeed only committing a few of the more important items to long-term memory. Once information is committed to long-term memory it can be recalled almost simultaneously but committing items to long-term memory takes a longer time and substantially more effort (i.e. it requires *attention*). Naturally, the persistence of long-term memory and reluctance of people to overload that memory makes them selective about what gets transferred from short-term visual memory to long-term memory. This is demonstrated by a phenomenon known as *change-blindness* [*61*] where people become blind to substantial changes between two images when these changes do not draw their attention (Figure 2.16).

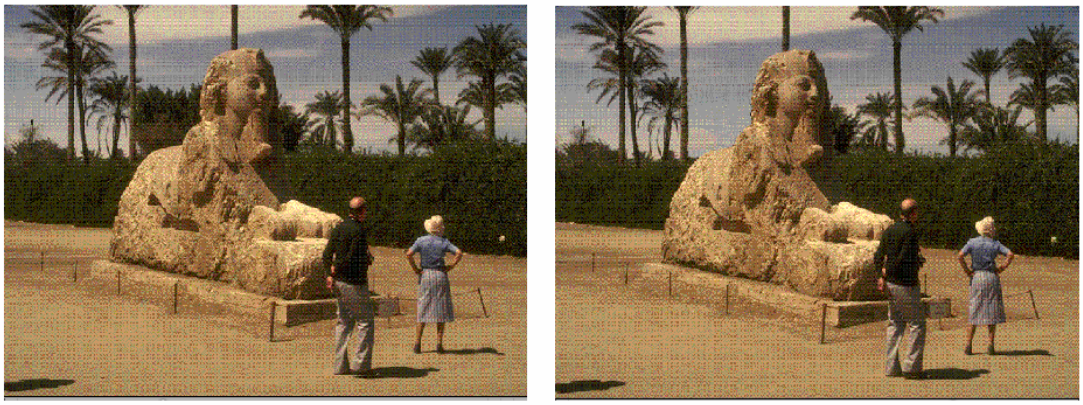

Figure 2.16. ***Change blindness: It is difficult to observe the difference between the two images. This is because the area of the picture that changes is not an area that would naturally draw the attention of the observer (taken from [62]).***

One likely explanation of the poor performance of post-attentive vision is that users would prefer to utilise a graphic itself as if it where long-term memory (i.e. referring back to the graphic as and when required) rather than commit features of that graphic to their own long term memory. This makes it desirable to allow them to do so by keeping the most relevant views accessible at all times and reducing the time taken to extract information from a graphic so that it is comparable to the time taken to extract information from long term memory (i.e. as short as possible). A more general guideline that can be derived from the theory of post-attentive processing is that users should not be required to rely on long-term memory to extract knowledge from a visualisation.

2.4 Information Visualisation Techniques

Starfield-displays [*33*] are a classic example of how various information visualisation guidelines have come to be applied to support the analysis of multidimensional data. Techniques commonly used with starfield displays are zooming, dynamic query filters [*33*], dynamic-labelling [*63*], details-on-demand pop-up windows [*64*] and multiple coordinated views [*65*]. These techniques are detailed further to provide an example of how the principles of information visualisation are put into practice.

2.4.1 Starfield Displays

The Starfield display [*33*] is a type of interactive scatter-plot where points encode additional information by having different colours, sizes or shapes. The term

Starfield is used to convey the feeling of flying through a galaxy of stars and planets that a user might experience when zooming into an area of the initial overview to view that area with more detail. In information visualisation terms starfield displays can be described as an interactive visualisation where data items are represented as points with the mapping of two selected attributes to orthogonal axes and the mapping of a selection of remaining attributes to other appropriate visual variables (e.g. colour, shape or size).

Normally Starfield type scatter-plot displays are configured according to two basic rules. Firstly, the most important data attributes are mapped to the visual variables which humans are more perceptually sensitive to (such as position and colour) [*24*] and secondly, where possible, the properties one would wish to draw from an attributes representation *match* the properties that attribute is capable of conveying (see Table 2.2 for a list of visual variables and their properties). For example, in the Starfield display of the FilmFinder application [*33*] (Figure 2.17) the most important attributes of the data, with regard to the overview required, are *Year of Production*, *Popularity*, and *Genre*. Each of these attributes is mapped to a more dominant visual variable. *Year of Production* and *Popularity* are ordered and mapped to visual variables from which order can be and perceived (position, x and y axes respectively). *Genre* is not ordered but it has nine possible values. This means that, in order to distinguish between genres, genre must be mapped to a visual variable with a length of nine or more for distinction. Genre is mapped to the visual variable hue (i.e. colour), which has a length for distinction of ten. While it would also be possible to map *genre* to the visual variables size or position with each of these variables having a length for distinction greater than nine, these variables are ordered so cannot be said to match the properties of *genre*. If *genre* were mapped to an ordered variable it would use up a visual variable that might be used better for another attribute and, possibly, mislead the user into the notion that the visual attribute used to convey genre is either being used to convey some other ordered attribute or make them confused as to why there would be an ordering to film genres. In later implementations of the Filmfinder application the size of film markers is, appropriately, used to indicate the length of a film.

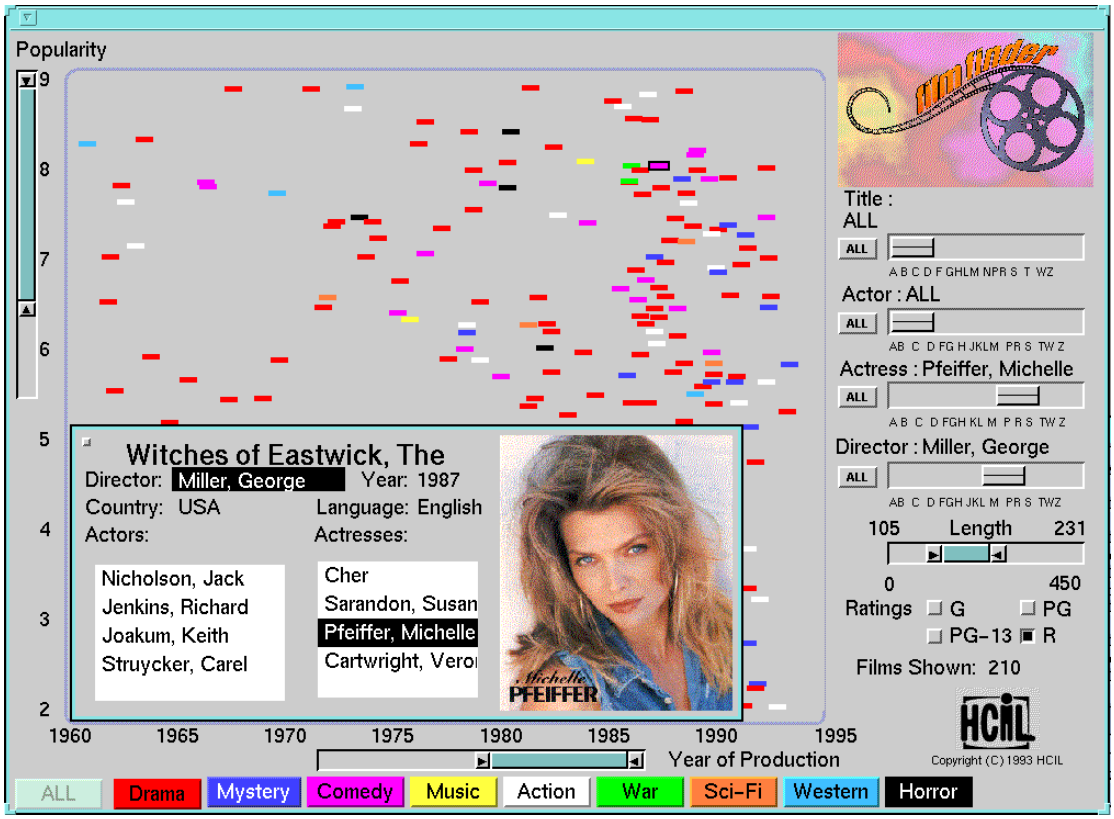

Figure 2.17. ***FilmFinder: A Starfield display with details on demand and dynamic query sliders (taken from [33]).***

2.4.2 Zooming, Dynamic Query Filters and Details on Demand

As well as presenting the most appropriate attributes of a data-table in an efficiently mapped overview of the data, Starfield displays can also adopt the *visual information seeking mantra* approach of revealing more information as and when the user requires it. The first phrase of the mantra, *overview first*, is adhered to by offering the user an initial overview of the data. The second phrase, zoom and filter, has two components; zooming allows the user to focus in on an area of the overview and filtering allows them the specify constraints that determine which items are displayed. Details-on-demand reveals more information about selected items whenever it is appropriate to do so without degrading the main display or when they are specifically requested by the user.

Zooming is integral to the Starfield display technique. Here, the user can select an area of the main display which is enlarged to replace the main display. Often this is combined with an element of *details-on-demand* where more information is revealed in the less crowded zoomed-in display. For example, when a user zooms into the Starfield display of the FilmFinder application and the number of marks in the

display drops bellow a certain threshold, the display is augmented to include text indicating the names of films beside their respective marks.

The act of filtering is to specify a condition and have data items that do not satisfy that condition removed. With regard to information visualisation, filtering removes items from the display and should be rapid, incremental and reversible. This type of filtering is known as dynamic querying and is supported by interface components known as widgets (sliders, buttons etc). Each widget is capable of being used to filter according constraints associated with a single attribute and, in general, the type of widget will match the type of attribute. For example, for a categorical attribute with no ordering (such as the genre of films) each possible attribute may be represented by a checkbox switch with the state of the checkbox used to determine whether or not items with that attribute value are displayed or not. To remove items with a certain value from a display the user would simply click once on the checkbox with that value labelled.

The established widget for filters based on the values of ordered attributes is the slider. Basic single value sliders have a single line or elongated rectangle, normally horizontal, representing the range of values for an attribute and a thumb (usually a triangle) positioned over the line to represent a selected value within the range of that attribute's values (Figure 2.18). Normally this type of slider also has ticks (small vertical lines) and/or text labels at selected regular intervals along the range of values line to indicate the relationship between attribute values and points on the line. Moving the mouse cursor over a slider thumb, clicking and dragging curser, moves the thumb so that it remains at the same vertical displacement as the mouse cursor until the mouse button is released or the mouse cursor is moved so that it would put the thumb outside the range of the attribute values. This allows the user to change the slider value. Depending on the specific coding of a slider it can be used to filter the information displayed according to any one of three conditions. Either the display will retain all items with a value greater than, less than or equal to the value indicated for the attribute in the data related to the slider. The last of these cases is often applicable when a slider is used to select a value from an attribute range with a fixed number of unordered string values. In this case values are assigned to equidistant points along the slider range, ordering is determined alphanumerically and the option

all is appended to the list of possible values (selected by moving the thumb to the far left-hand-side giving the user the option to avoid setting any filter according to the attribute associated with the slider).

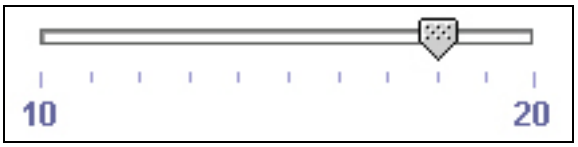

Figure 2.18. ***Single value slider***

For attributes with a larger number of alphanumeric values the single value slider becomes impractical due to the reduced spacing between values and the difficulty a user would have in moving the slider to their desired position. In this case an Alphaslider [*66*] (Figure 2.19) can be used. Alphasliders have their range annotated using only the first letter from selected values and allow rapid selection using two levels of granularity for interaction.

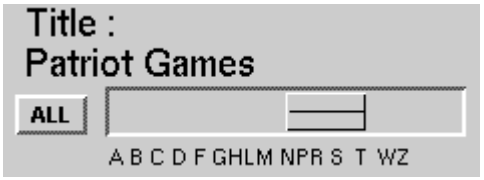

Figure 2.19. ***Alphaslider (taken from [33]).***

Range sliders (Figure 2.20) are used to filter items according to a range of acceptable values for a given attribute. The appearance of a range-slider is similar to that of a single value slider with the notable exception that it has two thumbs positioned at different points on the attribute range line (normally triangles pointing in toward one another) separated with a rectangular bar which indicates the selected range. The default positions of the thumbs are at opposite ends of the slider range and thumbs are moved in the same way as those of a standard slider. The condition of a filter set by a range slider is that only items with values (for the attribute related to the slider) within the selected range will remain in the display. Multi-rage sliders are common in information visualisation being either combined with Starfield displays occupying an adjacent panel of the same application interface, superimposed onto the data which is being visualised [*67*] or, in the case of data visualisation sliders [*68*], having

the attribute range line transformed into an elongated rectangle and having a representation of information enclosed.

Figure 2.20. ***Range-slider (taken from [33]).***

Dynamic query widgets normally operate according to the principles of direct manipulation [*38*]. This means for example that, when a value is to be selected it can be selected using a slider rather than having to type in its value using the keyboard and while a slider is being moved the display is continuously and rapidly updated to reflect its current state. The principle of tight coupling is also applied so that after a slider thumb is released and a query is performed the slider thumb can be adjusted to refine the query. Rapid, incremental, and reversible interactions among components ensures that while a slider is adjusted the values permitted for selection on any other sliders are continuously updated to avoid the user being able to perform selections that return no items. These compliances with principles of direct manipulation and tight coupling act to maintain the users feeling of direct control over objects and enhance the usability of the interfaces in general.

Another extension of Starfield displays that involves tight coupling is details-on-demand. While zooming already incorporates an aspect of details-on-demand, this feature is also implemented as pop-up details-on-demand windows, combined details-on-demand views and dynamic labelling [*63*]. Pop-up details-on-demand windows are the most basic form of details-on-demand an as such they are commonly referred to simply as *details-on-demand*. These details-on-demand are used to encode information, normally in plain text format, that cannot be accommodated in, or is not relevant to, the main data display. To initiate details-on-demand a user will move the mouse cursor over and click on an item of interest. This causes a window to pop up over the main display to reveal information about that item alone (see Figure 2.17 for an example). Once the user is finished with the pop-up details-on-demand window it can be closed with the interface view restored to its prior state. Combined details-on-demand views are the same as their pop-up

counterparts with the exception that the details are displayed in a different panel of the interface that does not obscure the main display. This means that interaction with the main display can continue while the details-on-demand are visible and they can remain visible until the details of another item are required.

Dynamic labelling [*63*] is a different type of details-on-demand that acts to display a textual attribute for a selected number of items. This type of technique is particularly useful for visualisation when data-sets have a fundamental textual identifier for each element (e.g. a person's name, a gene's symbol, a movies title etc). These identifiers tend to be particularly difficult to accommodate in a visual representation of data. This is especially the case when there are large numbers of items as there is a limited amount of screen space and characters need to be large enough on the display space to be legible. The basic interaction to initiate dynamic labelling is for the user to move the mouse cursor over the top of the visual representation of a data item. On this action a label appears somewhere next to the representation of the item. This label is a plain text representation of the textual attribute value for the item and is normally overlaid onto a coloured rectangle to distinguish it from the background. When the mouse cursor is moved away from the items representation, the label is removed and the display is restored to its state prior to the interaction.

For denser displays of data where a user may require to label multiple items simultaneously interface designers can implement a technique known as excentric labelling [*69*] (Figure 2.21). Whenever excentric labelling is in effect the mouse cursor is replaced with, or augmented by, the outline of a circle or diamond shape and all items within the bounds of the shape are joined to labels appearing on either side of the shape. When the mouse cursor shape is moved the labelled items change accordingly.

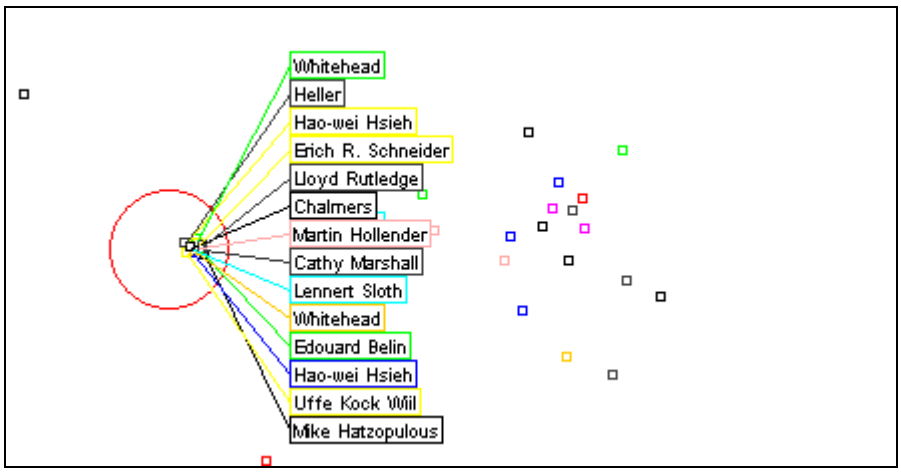

Figure 2.21. ***Excentric labelling (taken from [69])***

A limitation of Starfield displays is that the number of data attributes represented in the overview is restricted by the properties of visual variables. The qualities of attributes to be represented are also limited by the properties of visual variables and the perceptual qualities of different variables might affect the predominance of attributes disproportionately with their relative importance to the user. This might be problematic if the data has a high dimensionality, with a large number of attributes, or attributes require an equivalent weighting in the display. While Starfield displays are appropriate for applications, such as the FilmFinder, where only a few attributes of the data are required for an effective overview, Starfield displays are inappropriate in situations where users require to find patterns associated with larger numbers of attributes and these attributes cannot all be encoded effectively using appropriate visual variables.

2.4.3 Multiple Coordinated Views

An information visualisation interface need not be restricted to a single view of the data. Indeed, the majority of interfaces include two or more views relating to the same data. These views are often *linked* [*70*] so that interaction with one view will transform the data representation in an other. Here multiple combined views have a variety of functions [*65*]. Normally, however, they can be considered as different representations that give the user a better understanding of the underlying data [*65*], or allow the user to select and manipulate objects more easily [*71*]. Different views either represent different mappings of the same data or different subsets of the data. In the former case the different representations can serve different user requirements

and combine to allow the user to form a better understanding of the data. When views relate to different subsets of the data, an overview might be linked to a detail view to provide context, allowing the user to determine which subset of the data they are looking at.

2.5 Animated Information Visualisations

Computer animations exploit a perceptual illusion known as the Phi phenomenon which allows humans to perceive motion without actual physical motion of the stimulus (initially described in [*72*]). This phenomenon, also used in television and cinema projections, involves the successive presentation of briefly visible still images that concatenate into motion. For a display to be animated it must be redrawn at least every 0.1 seconds [*73*]. This allows the animation to be smooth and support object consistency so that the user can track objects throughout the animation without having to reassimilate different displays. With regard to information visualisation, the perception of dynamic patterns through animation is less well understood than the perception of static patterns [*74*]. Nonetheless, humans are particularly sensitive to patterns in motion and animation is shown to be effective and appropriate for the analysis of various types of data.

There are two primary motivations for information visualisation designers to develop animated interfaces. These are related to the expressiveness of motion and the capacity of animations to allow visualisations to encode large amounts of information. Those techniques that exploit the capacity of animations to encode large amounts of information do so by exploiting the third display dimension of time. Here, animation is used to smooth the transition between different viewpoints or different mappings of the data to visual form. This allows the user to track items between representations and reduces the likelihood of them becoming disorientated as successive representations can be quickly and effectively related to those that precede them. These types of animation can also be expressive, exploiting the capacity of animation to convey structure from motion, group items according to similar motion or draw the user's attention to motion.

2.5.1 Expressive Animations

Animated visualisations can be considered as being expressive on two counts. Firstly, they compel the user to utilise their low-level visual system using their pre-attentive perception to rapidly perceive patterns from motion [*14*]. Second, an animated representation can often be expressive in that it is closest to a user's conceptualisation of their data. A basic example of both these facets is the moving file pop-up window that appears when a computer file is copied in the Microsoft Windows operating system. Here, when a file is being copied, icon images of two paper files are shown side by side with an animated paper icon flying from one folder to another (Figure 2.22). This represents data being copied between *computer* files. To compliment these icons a horizontal bar is coloured progressively from the left-hand-side to indicate how much of the file has been copied. The animation effectively conveys three items of information. Firstly, the icons tell a user that a file is being copied. Moreover, their attention is specifically drawn to this fact by the motion of the flying paper icon. The progress bar conveys two items of information. By looking at how much of the horizontal bar has been colourised the user can tell how much of the file has been copied. By observing the *motion* of the division between coloured and non-coloured sections of the bar, the user can also assess how quickly the file is being copied. This use of motion to indicate additional information in the progress bar is both expressive and elegant using the minimum amount of graphics to convey the maximum amount of relevant information.

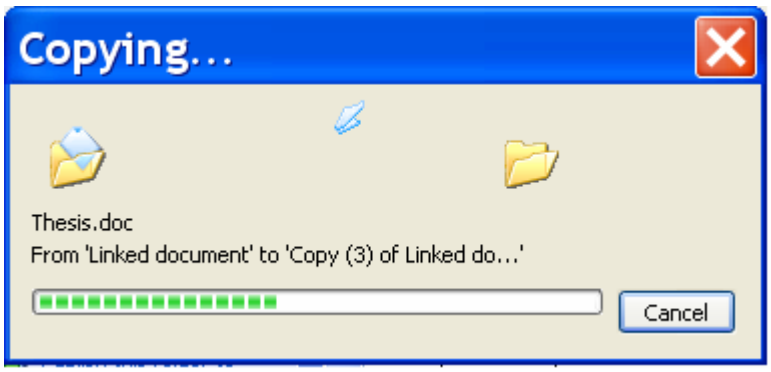

Figure 2.22. ***The expressive use of motion in a visualisation: the windows file copy pop-up window.***

In the case of the moving files pop-up window, animation is appropriate because the data being presented is in a state of flux and the display's main purpose is that of a live representation of the data. This type of animation, which displays change as it

occurs, is particularly useful for supervisory control and data acquisition (SCADA) interfaces used for remote control and monitoring of industrial machinery. An example of a typical SCADA interface is shown in Figure 2.23. A key issue in the development of these types of interface is annunciation and ensuring that users notice, comprehend and respond appropriately to events. Here, motion can be used to convey live events as they occur but great care must be taken not to overload the user's perception when an interface can include a number of signals of variable importance. A general rule prescribed is that smooth motion and motion on the periphery of a display will draw a users attention gracefully while course motion, or flashing, will draw attention abruptly with attention remaining fixed for a considerable time thereafter [*75*]. So, messages displayed in the direct line of vision and those using abrupt motion should be used sparingly for such an interface to be effective. Other less critical real-time visualisations involve more basic animation where it is the fact that something is moving or not that conveys information rather than the nature of the motion. These include interfaces such as computer drawing packages that animate icons [*76*].

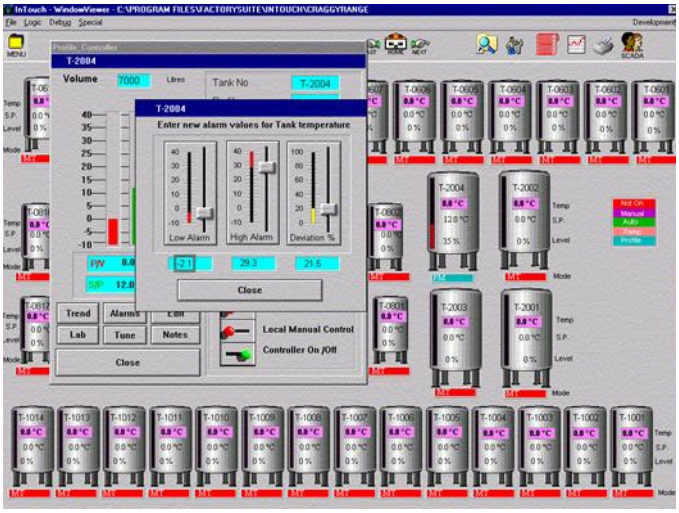

Figure 2.23. ***A supervisory control and data acquisition (SCADA) interface used for the real-time control and monitoring of industrial machinery: In this case, fermentation tanks (taken from[77]).***

In the case of SCADA and other real-time interfaces, the objectives of a user interpreting data from a live animation is normally somewhat different from that of a user performing exploratory analysis. A user viewing live data will generally be most interested in a relatively small amount of information related to the current state of the system and have a small number of well defined objectives related to that

information. Exploratory analysis generally involves looking at a large proportion of a data-set and if the data has a temporal quality, the user is likely to want to look at a larger range of time points together. While it is also, no doubt, the case that the types of interface that can support exploratory analysis need to be expressive, they also need to able to display a lot of data and allow a user to navigate effectively through that data.

Animation can also be used to make certain types of presentation graphics more expressive. Such animated presentation graphics are common in television news programs where they are used to convey a small amount of information and illustrate a specific point being made in the news-piece. Animations of this form are also appropriate for slideshow and internet presentations although the time and creative energy needed to produce such graphics can be seen as a significant deterrent to their application (see [*78*] for some examples of animated presentation graphics). Animated presentation graphics tend to come in one of two common forms; either exploiting the ability of motion to signify change or to facilitate the comparison of attribute values by altering the scale of the representation (particularly to demonstrate a significant disparity between values). An example of a presentation graphic that signifies change is the 'swingometer' popularised by the election broadcaster Peter Snow (Figure 2.24). The swingometer moves to indicate a shift in the popular vote between political parties and relates that swing to a number of seats to be won or lost in the British parliament. Here, a shift is a *change* in the popular vote and that change is represented using the motion of a swinging pendulum. Other animated presentation graphics illustrate changes in geographical coordinates showing, for example, the path of a hurricane or an advancing army. A presentation graphic using animation to facilitate the comparison of values might show the national surplus as a pile of banknotes then zoom out to show the national dept as a pile of bank notes beside it [*78*]. Here, the user can comprehend the rate at which the fist pile shrinks together with the time taken to zoom onto the second pile in order to assess how much one value is greater than the other. This works even if the first pile of notes is too small for the user to comprehend its size when the second pile is in full view. These animated graphics are not only expressive and informative but they can also make a presentation more compelling as the user is drawn to the motion.

Figure 2.24. ***An animated presentation graphic: The 'swingometer' (taken from [79]).***

2.5.2 Animated View Transformations

As well as being expressive, animations can be used in information visualisations to smooth the transition between alternative views of the same visual structure. These animations are generally user controlled and, as such, can be thought of as types of viewpoint control view transformation. Common animated viewpoint controls are zooming, panning and rotation. The animation of these transformations ensures that the transition from one viewpoint to another is smooth. This has a number of benefits. Firstly, it allows users to track objects between views without becoming disorientated. Moreover, it increases their capacity to relate between alternate views. This allows the user to build a mental image of the structure from their perception of different projections. This mental image can be used as an overview that acts as a summary of the data as a whole or, often more crucially, allows the user to assess which part of the data they are looking at.

Animated zooming and panning are generally applied to the exploration of two-dimensional visual structures such as maps or floor plans. These transformations are best understood using the concept of a fixed display space that is observed through the viewfinder of a moving viewpoint camera [*80*]. Here, the image displayed by the visualisation is what would be observed through a fixed rectangular aperture with a fixed displacement from the viewpoint (Figure 2.25). The viewpoint for an overview is the nearest point from which the entire display space can be observed and the basic actions of zooming are moving the viewpoint toward the display space to zoom-in, away from the display space to zoom-out and perpendicular to the display space to pan across (Figure 2.26). These actions are initiated either by the user interacting with an image (e.g. selecting a portion of the image to zoom in to), operating direct

controls (e.g. pressing buttons to pan up, down, left, right, and zoom-in, zoom-out) or by selecting an item in a coordinated view that is to become the focus of the new image (this might cause zooming, panning or combined zoom/pan).

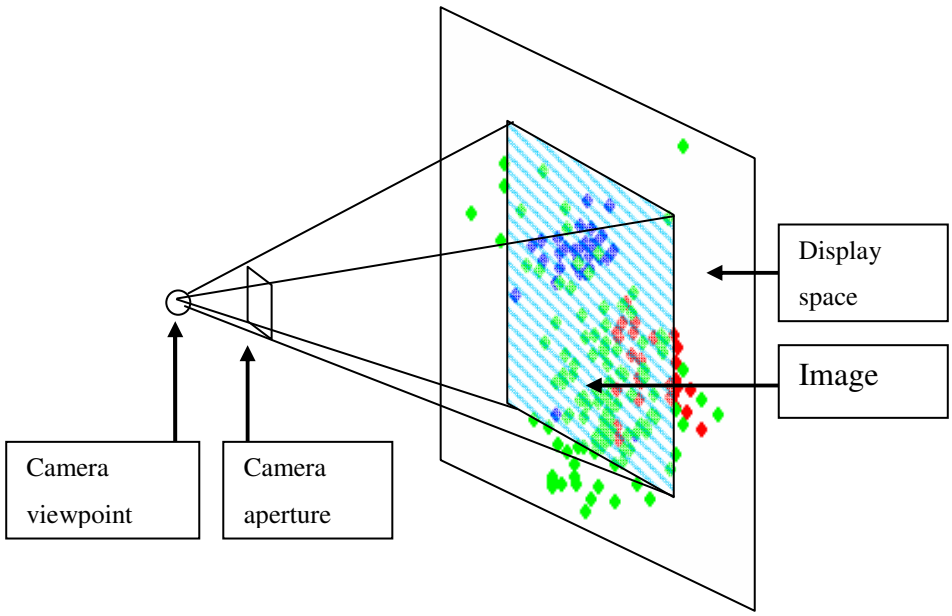

Figure 2.25. ***The camera metaphor used to describe zooming in information visualisations.***

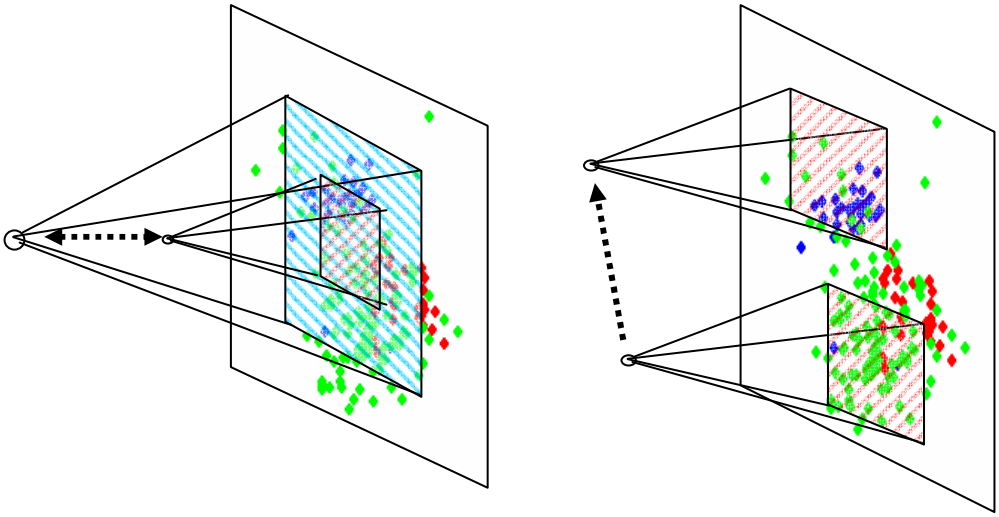

Figure 2.26. ***Zooming: Zooming in and out (LHS), and panning (RHS)***

Instantaneous (non-animated) zooming and panning can be thought of as a viewpoint camera moving instantly from one viewpoint to another. This might cause the user to become disorientated as they find it difficult to relate between the views. After each

zoom the user is forced to reassess the scale of the display and will find it difficult to keep track of items. If zooming is animated the camera moves gradually between viewpoints so that the transition between viewpoints is smooth. Here the viewpoint can move along a simple path [*81*] or a curved optimal path [*80*]. If zooming is along a simple path the viewpoint will be shifted along a straight line to its new location. Optimal paths are arcs configured to make zooming more smooth and efficient. In the case of a pan operation, this arc might cause the camera to zoom out before zooming back in to a second viewpoint. This would allow the user to view the primary subject of both viewpoints simultaneously in order to better assess their relative proportions and displacement. In any case animated zooming can give an indication of the scale of the current view. This can be assessed by relating the perceived rate of viewpoint movement to perception of scale at different viewpoints as interpreted from the relative displacement of known items. Animated transitions also allow users to track items between views providing object consistency so that users do not have to search for objects each time the viewpoint changes. This improved understanding of scale and the relative displacement of items made possible by the use of animation in zoomable displays, generally allows users to improve their understanding of the context of the data they are viewing allowing them to form a better understanding of the structure of the data.

Animated view transformations can also be used to explore three-dimensional visual structures. These types of visual structures are commonly used in scientific visualisations, where real world three-dimensional objects are represented in the display, or information visualisations of abstract data when a third dimension is required to organise the data. Of course, most physical displays are, in reality, only two dimensional with computer monitors having flat screens and projection equipment projecting an image onto a flat surface. Indeed, human vision is only capable of a limited version of three dimensions with flat images falling on each eye and human brain utilising visual clues, known as depth cues, to perceive the spatial orientation of objects. Visualisations of three-dimensional visual structures replicate these depth cues so that users can have a similar perception of three-dimensional visual structures. Depth cues include stereopsis (stereo vision), kinetic depth, perspective, occlusion, texture gradients, size gradients, shape from shading, cast shadows and focus.

Stereopsis, also known as stereovision, is the view of the world perceived by two eyes. Here, each eye has a slightly different perspective and the disparity can be used to gauge the distance of objects from the observer. Other depth cues are monocular, meaning that they can be perceived without stereopsis. The majority of these require only a single static image. These include occlusion, where an object that partially obscures another will appear closer, and perspective, where larger objects appear closer. These cues tend to be less reliable than steriopsis, as a single image has more scope for ambiguity than two images being compared. For example, when using perspective a larger object may appear closer because it is actually physically larger. Moreover, certain monocular visual cues might actually be detrimental to conveying information from a visual structure. For example, if an object is partially obscured, as a number of items must be for occlusion to be used, then the user will be less aware of that object and less able to draw information from its representation.

Kinetic depth is a monocular depth cue that works with a moving image. This cue works on the assumption that all the objects being observed are physically static. When the observer's viewpoint moves around these objects, different perspectives are available. Human perception and memory can combine these perspectives to interpret a three dimensional structure. Given that this interpretation is combined from an array of images it has less chance of it being ambiguous and there is less likelihood that objects will be obscured. Indeed, studies have shown that animated representations using a multitude of images are more effective at conveying structure in data than representations that use only two images to replicate stereopsis [*16*].

Normally in visualisations kinetic depth cues are activated when a viewpoint is rotated around a visual structure. As with zooming, a camera metaphor can be used to describe how viewpoints are used to generate images (Figure 2.27). In this example, the camera is considered as remaining static with the visual structure being rotated about mutually perpendicular X, Y or Z axes of the display space. The Z axis lies along the line of vision and, the X and Y axes are perpendicular to the Z axis and each other. The x-axis is horizontal, the y-axis vertical and the axes origin lies somewhere inside the three dimensional display space. For the display to be

animated, the rotation about axes must be graduated over time with intermediate projections rapidly re-displayed during rotation.

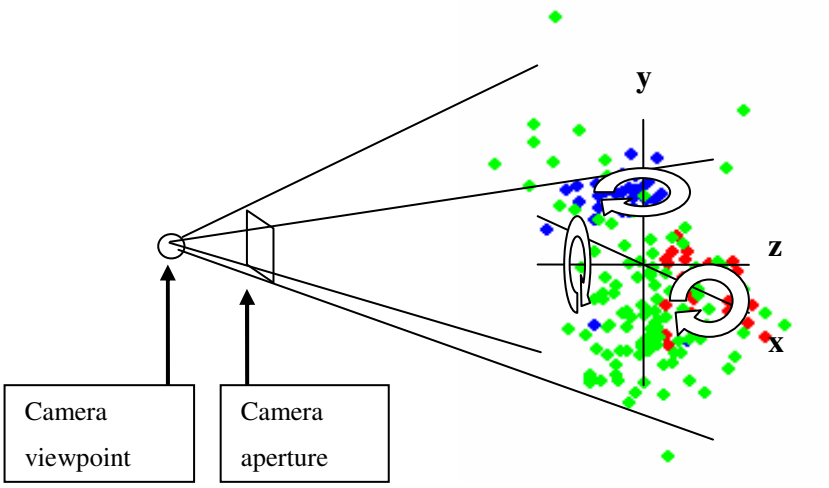

Figure 2.27. ***The camera metaphor used to describe two dimensional projections of a thee-dimensional display with axes about which that display may be rotated to convey structure.***

The primary advantage of animated rotation is that it allows the user to perceive structure from motion. There are, however, other significant advantages of animated rotation, particularly when the user is given control over the direction of rotation and can stop the rotation when an informative static image is found. Alternative viewpoints on the data can have more or less value to the observer as they highlight different groupings and outliers. The user can continue to rotate the visual structure until a useful viewpoint is found then stop the rotation to investigate that view in more detail. In this case the user might also use their mental image of a three dimensional structure (built from the structure from motion effect) to assess which part of the data they are looking at. The overall process of investigating the data in this manner can be thought of as being similar to the manner in which a museum visitor might investigate a model of an unusual animal. The visitor might first walk around the model to gauge its three-dimensional shape then stop at a position from which they have some interesting perspective to, for example, count the animal's legs or see how long its neck is compared to the rest of its body.

Another popular use of viewpoint transformations is to allow the user to 'fly-through' or 'walk-through' a virtual three dimensional environment [*26*]. Here the

path of the viewpoint camera is controlled by the user so they have the feeling of moving through the data. When the data is abstract it is normally represented using some real world metaphor, such as landscapes [*82, 83*] or buildings [*26*], that allows the user to relate their experience to movement in the real world. View transformations are also involved when a three dimensional representation can be manipulated using specialised hardware such as virtual reality gloves. Here the observer is given the impression of being able to grab an object and move it directly using their hand. This type of operation has even been used to facilitate the exploration of four dimensional objects [*84*]. Here the object is projected to three dimensions with sound cues supplementing or replacing visual cues to help the user build a clearer mental model of the original object being represented.

2.5.3 Animated Visual Mapping Transitions

Animation can also facilitate the exploration of large-scale data with a more complex structure when it is used to smooth transitions between alternative visual mappings of that data [*17-19, 85*]. These visual mapping transitions differ from view transformations in that it is the visual structure of the data that changes and not merely the point from which that structure is viewed. This means that the nature of the motion is generally less predictable. Aspects of the motion caused by viewpoint transformations are predictable in so far as the user will able to build a mental model of the visual structure and items will largely move in accordance with that model. When the visual structure of a visualisation changes, building a mental model of a single visual structure is less useful and the comprehension of multiple visual structures may be beyond the user's capacity (especially if there are a large amount of data). Instead, visual mapping transitions tend to rely on allowing the user to better understand structured data by building a mental model of that data's structure itself rather than the visual structure of a single mapping. This motion caused by mapping transitions will generally be less natural than that caused by viewpoint transformations as it cannot necessarily be related to any aspect of real-world motion. As a general rule a visual mapping transition should take about one second [*86*]. If the transition were to take much less time, object consistency might be lost and if it were to take more time the user might become disinterested waiting for a response.

An example of animated visual mapping transitions is the use of animation to switch between various TreeMap [*87*] and scatter-plot views to explore large-scale multidimensional data [*18*]. Here, animation allows users to track items, and groups of items, between representations and perceive patterns that cannot be made apparent from any single static mapping by observing the motion of item representations during transition. Another example of animated visual mapping transitions used to facilitate the exploration of structured data, is the use of animation in the cone-trees visualisation [*17*] (Figure 2.28). Here, animation is applied to facilitate the exploration of hierarchical data. This type of data includes a number of items, each of which can be connected upwards to one other item and downwards to any number of items. In the cone-tree display each item is represented with a labelled horizontal rectangle. If there is more than one downward connection from such a rectangle, these are represented using a three-dimensional cone connected at its point to that item with downwardly connected item representations equally spaced around its rim. All such connections in the data are represented in this manner with all cones, representing downward connections, orientated the same way pointing either to the left or upward.

It can be seen from Figure 2.28 that cone-trees are effective at conveying the overall structure of the data, thus providing an overview. However, all of the detail cannot be represented in such a static display as a proportion of item representations are obscured. In order to uncover information that is encoded in obscured representations, the user must interact with the representation to rotate selected cones nodes until those representations appear. Here when cones are rotated the visual mapping of the data changes. This rotation is animated so that the user can track elements and does not become disorientated as the display changes. Animation in the cone-trees visualisation can also be thought of as conveying element of structure from motion where the user's interpretation of the 3D data representation is strengthened as they observe the structure moving and changing.

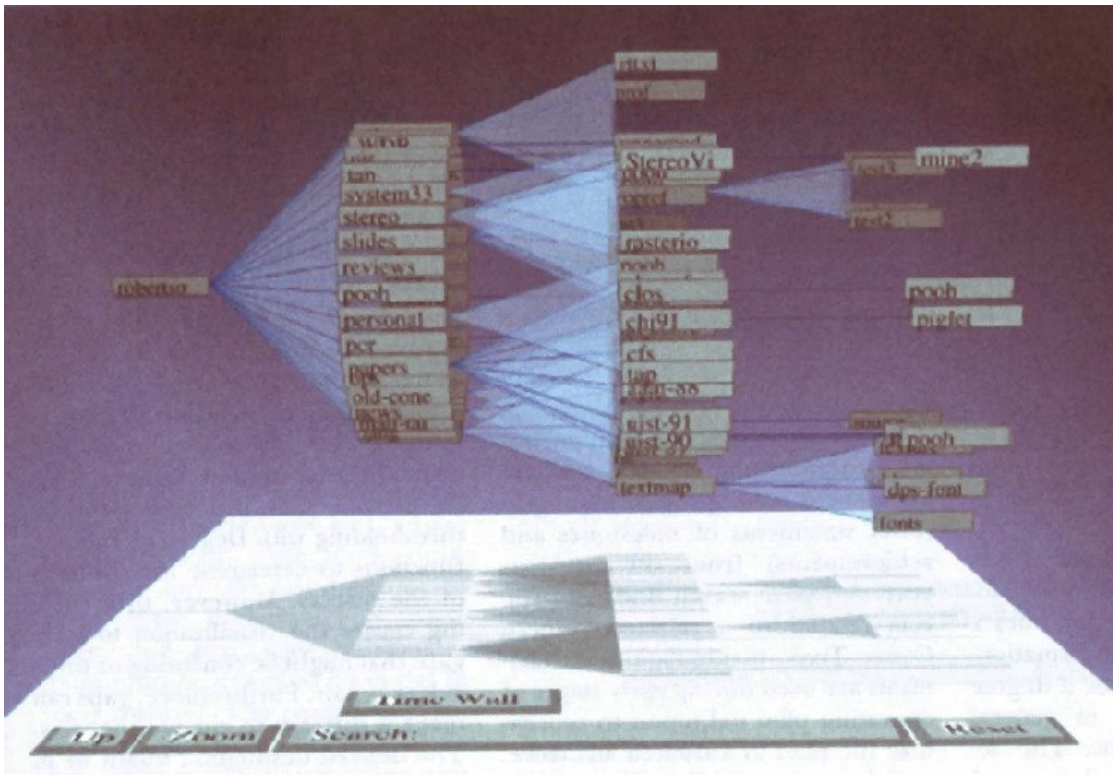

Figure 2.28. ***Cone-trees: An animated technique for hierarchical data (taken from [17]).***

The animated exploration of dynamic graphs with radial layout [*19*] (Figure 2.29) is another example of animation being used to facilitate the exploration of large-scale structured data. Here the data being visualised is graph data[2]. This type of data is similar to hierarchical data but connections are not directed and any item can be connected to any number of other items. In visualisations of graph data, item representations are known as nodes and connection representations as links. Figure 2.29 shows a static frame of the dynamic graphs with radial layout display. Here the graph layout is organised around a single graph node so that that node and its relationship with other nodes are the main focal point of the display. When the user selects another node, the display is reorganised around that node so that it becomes the new focus of the display. This transition is animated so that when a new focus is selected, nodes move smoothly form one position to another. This animation allows the user to track items and item groups maintaining context while exploring the graph. This use of animation can be thought of as allowing users to interpret structure from motion, but rather than interpreting a three-dimensional spatial structure, as is

[2] Not to be confused with the data that is displayed in a traditional graph which indicates the relationship between two or more variables.

normally the case with structure from motion, the user is able to directly interpret structure in the actual data being displayed by switching focus while maintaining context.

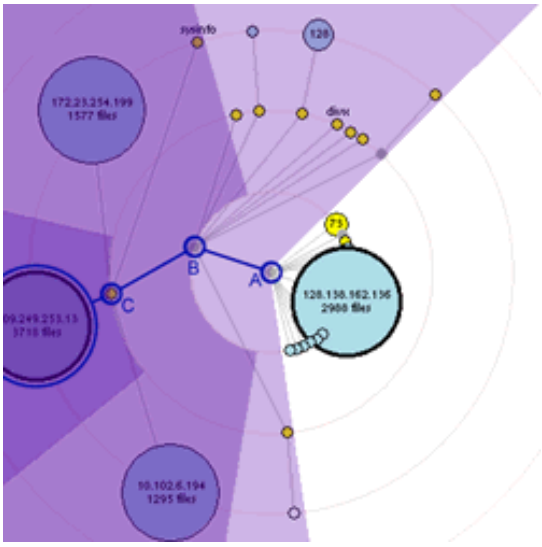

Figure 2.29. ***The animated exploration of dynamic graphs with radial layout (taken from [19]).***

2.5.4 Mapping Time to Time

The mapping of time in a data-set to time in an animated representation of that data can be thought of as specialised type of animated view transformation. Here the visual structure has time as one of its axes and the viewpoint camera moves along this axis. This type of visualisation is normally used with scientific and geographical data where spatial qualities are also represented. These include visualisations such as animated maps showing urban growth [*20*] and three-dimensional representations of simulated thunderstorms animated to illustrate the likely nature of their propagation (Figure 2.30) [*21*].

In animations where time is mapped to time, the user is normally able to control the animation - playing the animation until interesting patterns are found and pausing the animation to take the time to view those patterns in more detail. If time is considered as a dimension of the visual structure of the display then these controls can be considered as viewpoint control view transformations where a viewpoint camera moves along the time axis. Alternatively, changing the temporal context of a display can be considered as changing the visual mapping of the display where the current

time being viewed is a parameter of the visual mapping. In either case, the movement of item representations will be less predictable than with a normal viewpoint transformation as humans find it more difficult to build a mental model of a structure with a temporal dimension. The motion does however relate to motion in the real world as real world changes are represented as changes in the visualisation. In this regard time mapped to time visualisations can be considered as being expressive showing changes as they occur (i.e. *over time*).

A problem with displaying data over time is that the focus in time is normally determined by the rate of the animation rather than being directly controlled by the user. This makes it difficult to provide overviews and detail views or allow the user to focus on a portion of the data while being aware of the context. Partial solutions to these problems are normally applied in the area of video footage visualisation [*88-90*]. Approaches to providing an overview of an animation are to fast forward through the animation [*88*] or summarise the data using small images of frames either taken from regular intervals throughout the animation [*88*] or key points where the image displayed changes abruptly [*89, 90*]. The former approach has the disadvantage that the user might not have time to extract the relevant information from the overview. The latter approaches have the disadvantage that key information might be left out of the overview. Another approach to this problem relies on the majority of the image being constant throughout the animation (for example footage from a security camera) and only displays what differs from the normal background [*89*]. A number of different strategies are suggested for providing detail views for an animation. The most obvious of these is to pause the frame of the animation containing the detail [*88*] but this relies on all of the detail being contained in one frame. Alternatively an interval can be viewed as a detail view by either providing more space to view frames, slowing down the animation to spend more time on detail frames or cycling through the detail frames [*89*]. Here cycling through frames is often more appropriate as it allows the user to spend more time looking at the details. Focus with context in animation often requires linking a detail view to an overview and highlighting the detail view time in the overview [*88-90*].

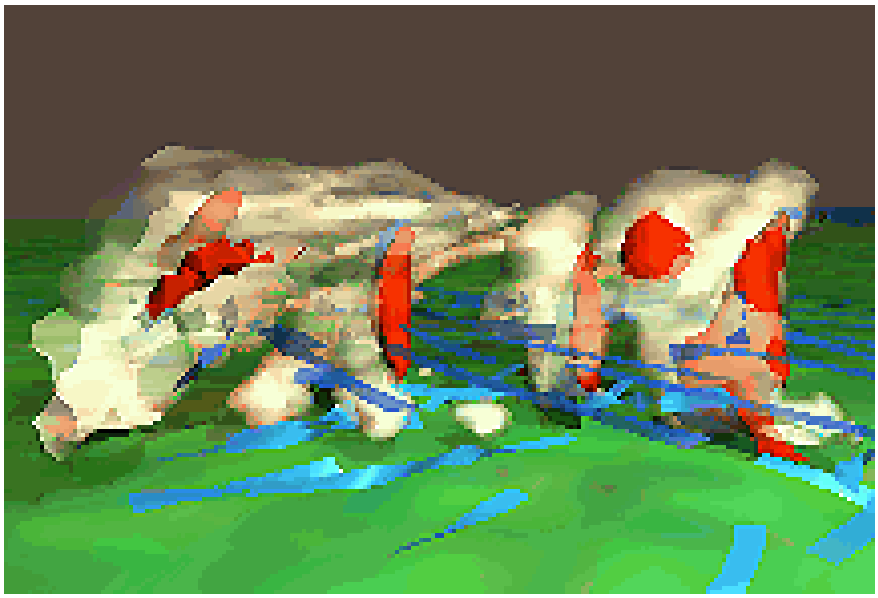

Figure 2.30. ***Animation of a thunderstorm simulation: The red isosurfaces represent columns of high vorticity, so noted because these areas potentially spawn tornadoes. The ribbons show wind velocity at two imaginary planes (taken from [21]).***

Animations where time is mapped to time are rarely employed in information visualisations (representing abstract data without intrinsic spatial qualities). There are a few possible explanations of why this might be the case. These relate to the types of abstract data that are analysed, general problems associated with the nature of abstract data and the possibility of false interpretation of motion in the display as actual physical motion.

Humans are naturally tuned to the perception of the real world motion of real world objects. These objects adhere to laws of physics and tend to move in a smooth and predictable fashion of the type users are known be particularly adept to interpreting. Animated representations of related to real world objects, including simulations, can aim to directly replicate this type of motion and appear natural allowing the user to utilise elements of their natural perceptual machinery. Animated representations of abstract data tend to have no such relation to real-world physical motion and can, as a consequence, be difficult to interpret. This might be because the observer misinterprets the motion as presenting movement in physical space or the motion itself is difficult to interpret because it is unnatural, with occurrences such as large objects disappearing or objects changing direction suddenly without their motion being regulated by the actual laws of physical motion.

An example of something that happens with animated representations of abstract data that does not generally happen in the real world is the disappearing of large objects. Change blindness can perhaps be explained by the fact that large items do not tend to disappear instantaneously so there has been no need for humans to perceive the disappearance of large objects in the past and the ability to detect such changes has not yet evolved. This could be problematic when, for example, an animated map attempts to portray a group of events with a fixed duration. As well as such obvious rules that govern the physical world there are other general laws of physics that a physical body will adhere to. A body will have properties such as inertia, momentum and mass that relate to its motion. Physical bodies are also compelled not to overlap in the same physical space. As these tend to regulate motion and abstract representations do not adhere to these rules, it is quite possible that abstract motion would appear more erratic and therefore be more difficult to interpret. While there are indeed some parallels to physical motion properties in abstract terminology for some data types (for example, steadily rising share price is thought of as having momentum while one that moves slowly and sporadically is thought of as having high inertia) these are normally based on conjecture rather than absolute laws of how something will behave so they are unlikely to have such a regulatory effect on the motion.

Time mapped to time animations of abstract data without a spatial aspect are particularly problematic because of the necessity to map non-spatial attributes to position in the display without motion being misleading. If, however, an appropriate mapping can be found, the additional information that could be displayed and the need to display larger amounts of information when analysing certain types of data is likely to make such an animation an appropriate option. Examples of types of data without a spatial aspect and with a temporal aspect (i.e. data where a time mapped to time information visualisations are possible) are stock price data, economic data and microarray time-course data.

2.6 Conclusion

To summarise, there are numerous issues that need to be considered when developing information visualisation software. The main challenge of information

visualisation is to find the best way of representing the data and allowing a user to interact with that representation so that they can extract the most valuable information. A good information visualisation will address perceptual issues, use the right display techniques and allow interaction that best serves the objectives of the user.

For this thesis the objective is to design an information visualisation technique to support the exploratory analysis of microarray time-course data. This type of analysis is recognised as being particularly challenging because of the scale and complexity of the data involved. As the data includes large numbers of values changing over time and a lot of this data would need to be previously displayed to the user in order for them to extract valuable knowledge from the data, the most relevant information techniques to be considered before designing a new technique are those that deal with large scale data and data that changes over time. As the biologists also use existing visualisation techniques to analyse their data, it is also worth considering what the specific requirements of the biologists are and why those techniques fail to meet those requirements.

Chapter 3 Microarray Time-course Data

The development of microarray technologies [*91, 92*] has revolutionized biological and biomedical research, specifically in the area of gene expression analysis. Where previous technologies allowed biologists to monitor the activity of only a few genes at a time, microarray experiments allow them to monitor the activity of thousands of genes in parallel. This can be done for different conditions [*93*] or at multiple stages of a biological process [*94-96*]. As the expression of genes can be correlated with a variety of higher level biological processes, there are a large number of applications for microarray experimentation in various areas of biological and biomedical research. Here, the more complete view of gene expression made available by microarray experiments has a number of distinct advantages particularly since the functional roles of a large proportion of genes are still, as yet, unknown and the proper analysis of microarray has the potential to identify these functions [*6*]. This thesis chapter begins by outlining the biological basis of a microarray experiment. This is followed by a brief description of how microarray technologies operate and a fuller description the data produced.

3.1 Biological Background

Microarray experiments record the activity of genes within a sample of cells. Cells are the essential structural and functional units of all living organisms. While some organisms, such as bacteria, are unicellular with only one cell, other organisms have large numbers of cells. For example, humans have an estimated 100,000 billion. These cells come in a number of different varieties (blood cells, skin cells, brain cells etc) and collaborate to realise a multitude of sophisticated biological processes within the organism. Regardless of their particular function, each cell contains an identical copy of the organism's genetic material. Here a set of chromosomes has genetic information represented in the chemical structure of particular deoxyribonucleic acid (DNA) molecules. These molecules are composed of building blocks called nucleotides that consist of; a deoxyribose sugar, a phosphate group and one of four nitrogen bases – adenine (A), thymine (T), guanine (G) or cytosine (C). The familiar double helix structure of DNA with complimentary bindings, i.e. C to G and A to T, is illustrated in Figure 3.1. In general the code of a portion of DNA is described

using a string of nitrogen base initials from a single strand of the DNA. For example the portion of DNA in Figure 3.1 is described using "…*AGTGCCCATTGAC*…".

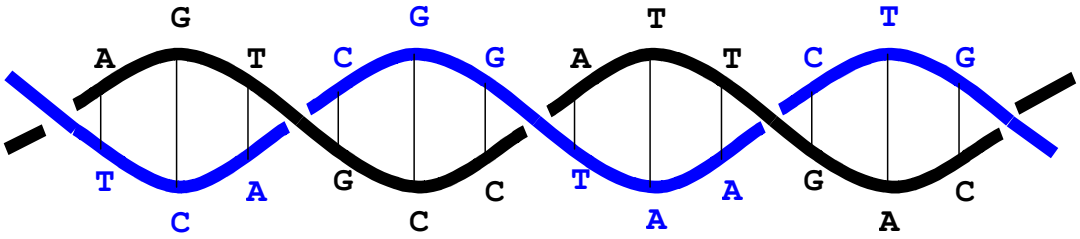

Figure 3.1. ***The double helix structure of DNA with complimentary bindings C to G and A to T***

Genes are biological entities that consist of proximate broken segments of DNA. Specifically, these are the segments of DNA that cells can transcribe into different ribonucleic acids (RNAs) and, at least in part, translate into proteins with each different gene normally transcribing then translating into a unique protein (Figure 3.2). As each cell of an individual has structurally identical DNA, each cell will also have a structurally identical set of genes.

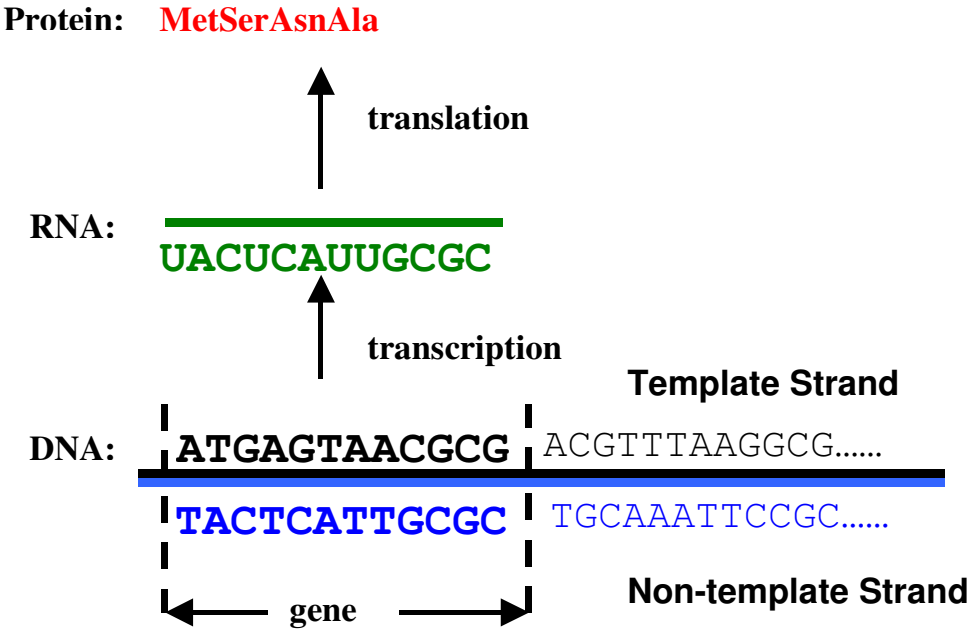

Figure 3.2. ***Gene expression: Genes transcribe into RNA and RNA translate into proteins***

RNA is considered as the primary product of a gene. In information terms it is essentially a copy of the strand of DNA from which it is transcribed. Here, the RNA

is a nucleotide sequence that maps from the sequence of the template strand with nitrogen bases guanine (G) replaced by cytosine (C), cytosine (C) replaced by guanine (G), thymine (T) replaced by adenine (A) and adenine (A) replaced by uracil (U). Uracil (U) is the only acid that is *not* already present in DNA.

After transcription, RNAs translate into proteins which can be considered as the secondary products of a gene. These different proteins combine both inside and outside the cell to fulfil a majority of the structural and functional roles within an organism. They also interact with a variety of other biomolecules to directly or indirectly regulate the expression of other genes [*76*]. This constitutes a network where the activity of genes combines to indirectly control both the production of proteins and the activity of other genes. While aspects of this network are extremely complex, it is often possible to correlate the activity of genes and groups of genes with a number of higher level biological processes in order to reveal previously unsuspected aspects of biological functioning at different levels of detail. The process of transcribing DNA into RNA then translating RNA into proteins is commonly known as protein expression, gene expression or simply just expression and can be considered as the conversion of genetic information into the structures and functions of the organism.

In recent years, initiatives such as the human genome project [*1*] have allowed biologists to identify and ascertain the structure of an increasingly large number of genes in human DNA and the DNA of several modal organisms. While in many cases this structural information is considered as virtually complete, information related to the function of genes is relatively sparse. Humans are generally accepted as having around 20,000 to 25,000 genes [*97, 98*] and bacterium around 6,000. Only about 6,000 (~25-30%) of human genes and about 1,500 (25%) of *Saccharomyces cerevisiae*, which is perhaps the most intensively studied model organism[3], have any known biological function [*99, 100*].

[3] A model organism is a species that is extensively studied to understand particular biological phenomena, with the expectation that discoveries made in the model organism will provide insight into the workings of other organisms.

3.2 Microarrays

Microarrays allow biologists to simultaneously measure the relative concentrations of large numbers of different RNAs within a given cell sample. This allows them to extract information about the function of genes or use existing information about the function of genes to reveal aspects of biological processes in the cell sample. The primary advantage of microarrays over other technologies that allow biologists to monitor the activity of genes [*101-103*] is that expression is recorded in parallel and for a much larger number of genes. Where other technologies only allow biologists to monitor the expression of a few genes at a time, microarrays allow them to monitor the expression of thousands of genes. This more complete view of gene expression has a number of significant benefits. Firstly, the ability to monitor large number of genes in parallel means that the technology can be used to investigate biological processes without the necessity of prior information indicating that a particular gene or group of genes are involved. Secondly, a near complete view of expression can reveal biological processes where there has been no prior indication that that process might exist. In effect, where previous technologies allowed biologists to test limited hypotheses involving a few genes at a time, microarrays provide biologists a with a view that allows them to test hypotheses involving large numbers of genes or even form new hypotheses as patterns in the data reveal previously unknown or unsuspected phenomena [*6, 104*].

The two types of microarrays in common use are customized cDNA microarrays [*92, 93*] and commercially available high-density arrays [*105*]. While there are fundamental differences in the manner by which these types of array are manufactured and distributed [*105*], both rely on the same basic principle of preferential binding which rules that any single stranded sequence of nucleotides (RNAs or DNAs) will bind to (or "hybridize" to) its complementary sequence (i.e. C to G and A to T or U). Microarrays (also known as gene chips or slides) are manufactured by attaching nucleotide sequences corresponding to the known sequences of genes to fixed locations on a solid support (usually a glass microscope slide). Each point at which a sequence is attached is called a probe and different types of microarrays use different types of nucleotides as probes.

Regardless of the type of nucleotide used in the probes, the general process of microarray experimentation is the same. Firstly, RNA is extracted from a cell sample and labelled with a fluorescent dye. Next, the RNA is washed over the microarray. Complementary binding means that RNA relating to different genes in the sample will hybridize to their corresponding probes on the microarray. After hybridization, the array is scanned and image processing is used to quantify the amount of dye deposited on each probe to give an indication of the relative quantity of RNAs corresponding to each gene in the original cell sample. This process is repeated with different cell samples under different conditions or at different stages of a biological process depending on the specific objectives of the biologist performing the experiment. It is also common to repeat the process with similar cell samples (known as replicates) in order that results can be statistically validated. With regard to microarray experimentation, the only practical difference between different technologies is that commercial high-density arrays can only survey a single sample on the one array while cDNA microarrays are generally capable of surveying two. Figure 3.3 and Figure 3.4 show the processes of microarray experimentation for singular and dual sample arrays.

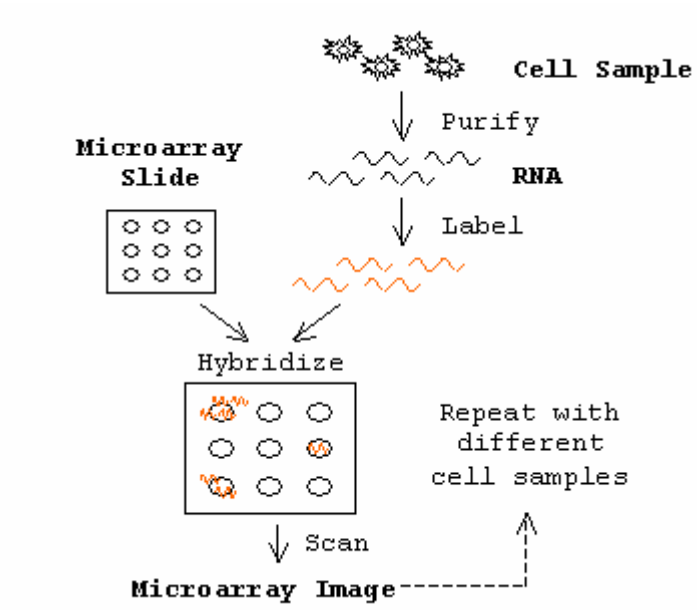

Figure 3.3. ***Process of single cell-sample microarray experiment.***

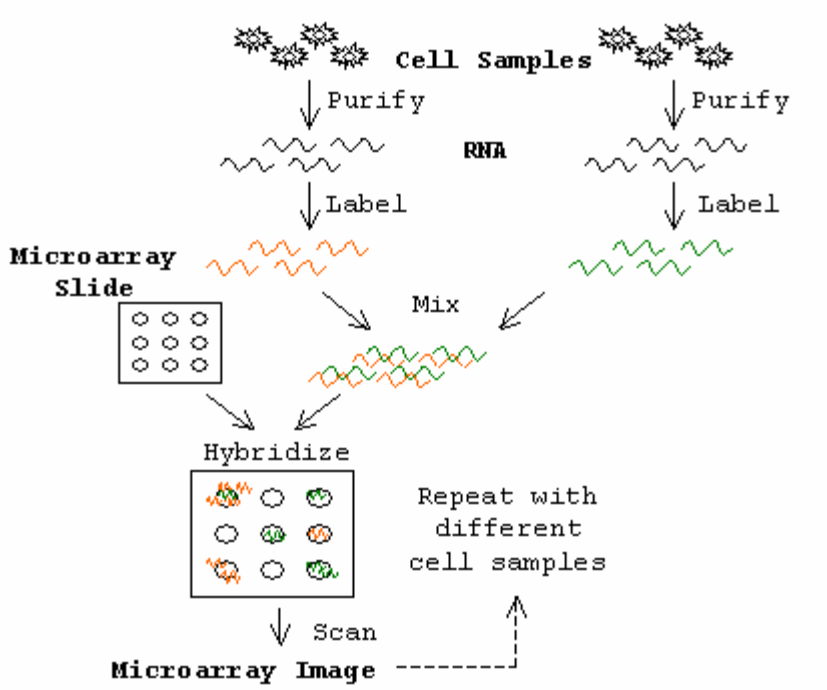

Figure 3.4. ***Process of a two cell-sample microarray experiment.***

The two types of machinery involved in microarray experimentation are microarray printers and microarray scanners. A microarray printer is used to attach nucleotides to the microarray and can operate using a spotting pin or an inkjet. Biologists also have the option to purchase prefabricated arrays from a commercial source. The main part of a microarray experiment, where samples are washed over the prepared microarray, is a manual procedure. This involves the microarray and sample being placed together in test tube and shaken. Microarray scanners (see Figure 3.5) are then used to generate data from the microarray in the form of array images. Although commercial scanning services are available it is more common for biologists to have a scanner within the laboratory where an experiment is performed. This scanner will normally be connected to a computer with software capable of processing the images to generate data-tables.

Figure 3.5. ***A microarray scanner***

3.3 Microarray Data

The raw data produced by microarray experiments are the scanned images of the microarray slides. Before these data can be considered as suitable for analysis they must undergo a number of transformations (Figure 3.6). Firstly, image-processing software is required to quantify the scanned images. This involves identifying spots related to probes on the array and quantifying the fluorescence intensity of spots, background intensity and other variables such as spot size and shape. The quantification matrices for each set of microarrays and their replicates are then processed to combine the quantities relating to each gene with the resulting matrix normalised to account for and remove sources of variation obscuring the underlying variation of interest which are variations in the level of gene expression [21]. Normalization adjusts for differences in labelling, detection efficiencies for fluorescent labels, and differences in the quality of RNA from the samples examined in the assay. While RNA populations cannot be quantified due to the nature of the experimentation, which deals with gross cell populations, normalization makes values relative across samples.

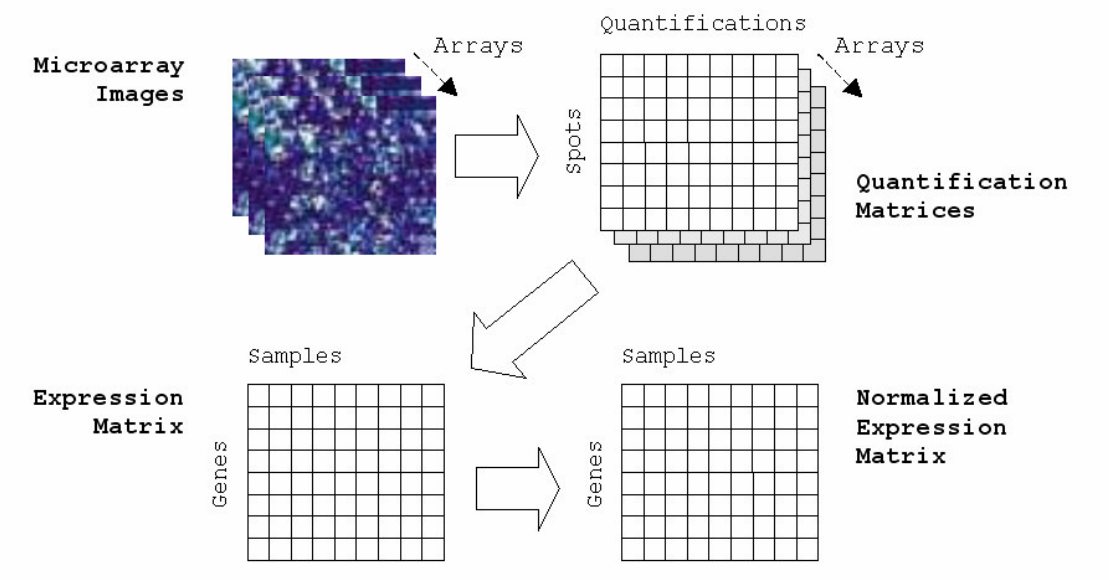

Figure 3.6. ***Transformation from microarray images to normalised microarray data (adapted from [106]).***

Normalised gene expression data can be viewed abstractly as a single table of data with rows representing genes, columns representing samples and cells giving measurements for genes in particular samples. In the case of microarray time-course data these samples relate to time-points and the data can be considered as large-scale time-series with a large number of distinct elements (genes) that have values recorded at each of a relatively small number of shared time-points. Figure 3.7 shows a portion of such data with labelled time-points represented as column headers and gene symbols as row headers. A typical microarray time-course experiment records the expression of around 8,000 genes over twenty or so time-points so normalised time-course data will contain around 160,000 values. These values have a log-normal distribution with a large number of small values and a small number of very high outliers. While errors can be associated with each of these values, for practical reasons they are normally disregarded during exploratory analysis. In general errors are only used to filter out genes prior to exploratory analysis or statistically validate any patterns found.

	V10	V12	P1	P2	P3	P8.5	P12.5	P14.5	P17.5	Lac1
Cd3g	33.9	26.47	34.5	26.5	12.43	18.27	11.63	13.4	20.5	6.5
Itih3	10.17	16.4	11.8	7.43	15.43	6.57	4.57	7.1	5.83	11.27
Ryr1	83.5	52.57	42.37	45.67	27.7	50.17	21.3	22.53	14.37	16.4
5930412E2	33.43	44.2	36.2	38.6	42.3	36.63	32.3	36.67	41.13	43.73
Traf4	81.07	113.33	90	110.13	149.4	93.73	103.5	127.03	241.3	292.93
Cdh11	9.4	25.43	16.5	16.97	19.13	16.47	10.3	10.97	10.77	11.17
6330414O(	101.57	141.77	100.97	140.8	178.17	102	127.47	128.87	151.27	158.23
Sox2	2.73	1.83	5.53	6.7	2.97	2.87	4.13	4.4	4.63	6.47
Klf3	32.87	31.67	23.8	20.9	40.67	27.33	28.43	29.3	53.77	45.4
Klf3	22.93	53.17	21.7	31.93	64	24.6	17.27	19.27	41.13	60.03
Laptm5	344.33	553.03	346.97	350.6	469.53	234.83	167.9	189.97	229.57	128.67
2010008K1	82.33	109.4	136.5	111.3	103.43	90.13	106.37	117.37	115.97	155
Tlk2	36.7	45.63	40.8	37.37	38.5	53.3	43.93	39.83	34.07	21.97
Yes	65.47	51.5	49.93	66.3	52.03	79.33	69.63	77.2	56.43	63.5
Mmp11	29.53	30.73	34.87	34.53	38.17	24.87	17.17	19.83	18.93	20.43
Mybph	37.33	37.83	28.1	44	46.3	28.2	23.93	23.73	35.07	60.93
Mtf1	33.97	33.3	23.03	40.93	46.37	35.77	41.1	43.93	49.87	53.23
Cspg2	43.23	70.23	110.1	166.67	163.07	119.67	112.93	55.37	29.33	7.73
Slc4a2	119.47	136.93	108.87	126.67	160.5	87.3	103.3	102.47	144.8	220.53
Chrna1	3.3	6.03	2	4.33	6.2	4.93	4.9	2.27	5.97	5.47
Cish	81.57	77.8	76.03	124.57	162.37	93.93	135.9	202.3	255.2	197.87
Mybl2	8.37	8.37	4.03	8.57	13.67	6.87	8	6.67	9.77	14.5
shrm	67.33	78.43	89.57	112.4	137.17	149.47	143.73	166.33	158.03	125.23
Bcat1	57.57	58.73	54.2	71.37	93.3	57.17	63	48.3	27.6	23.53
Pex14	273.77	287.8	202.33	252.63	311.57	224.47	211.63	191.87	231.63	236.97
Pex14	35.27	21.6	16.3	20.57	29.77	17.3	17.5	16.7	37.97	42.63

Figure 3.7. ***Portion of a data table from a microarray time-course experiment***

While microarray time-course data can, in general terms, be described as temporal where temporal is defined as having some relation to time [*25*], the data cannot be categorised as temporal data for information visualisation. This is because temporal data relates to entities with start and end times and microarray time-course data describes the changing activity of all genes over the same period. If one were to attribute start and end times to the activity of genes it would require pre-processing to assign some arbitrary cut-off dictating when a gene's activity was sufficient to deem that gene to be 'switched on'. This approach would necessarily reduce some of the essential information content of the data as to consider a gene to be on or off is a gross oversimplification of gene functioning.

The correct classification for microarray time-course data in information visualisation terms is that of large scale multidimensional data [*107*] where items correspond to genes and dimensions correspond to time points. While time itself is a physical dimension it is important to note that it cannot be used as a multidimensional data dimension. This is because the multidimensional data model works with fixed dimensional values and the value of time is constantly changing (in parallel) for all entities. This is without even considering gene activity which is, after all, the actual unknown variant of interest.

Chapter 4 Exploratory Analysis of Microarray Time-course Data

There are two basic approaches to the analysis of microarray time-course data; hypothesis testing and exploratory analysis [*6, 7, 108*]. Hypothesis testing is a 'top down' approach where the perturbation of a biological process will lead to predicted results (although often revealing unknown phenomena). Exploratory analysis is a 'bottom up' process that involves visualisation of the data and has the potential to lead to unpredicted results and generate entirely new hypothesis. While the type of analysis applied in any given situation will often depend on the particular objectives of biologists performing the analysis, it is exploratory analysis that is better suited to exploiting the potential of microarray data to reveal unsuspected phenomena and uncover gene functions. At present a large majority of gene functions are still unknown and the knowledge about gene functioning is recognised as having impact on a number of significant applications in different areas of biological and biomedical research. It is exploratory type analysis of microarray time-course data that holds the greatest potential to advance scientific knowledge and influence these areas of research [*6*]. This thesis therefore focuses on techniques applied to support this type of analysis.

As an initial pre-process to the exploratory analysis of microarray time-course data it is often rescaled to make the expression of genes comparable across multiple samples and ensure that analysis is not overly dominated by outlying values. This is normally followed by a procedure known as clustering. Clustering displays rescaled microarray data so that genes with similar patterns of activity and groups of genes with similar activity are positioned closer to each other. These representations are often interactive so that a biologist can manipulate the display to switch between alternate overviews of their data, relate between overviews and drill-down to examine interesting features in more detail. Other techniques do without clustering and allow biologists to interact with more abstract representations of the data. This chapter reviews methods for the rescaling of microarray time-course data and various clustering and interaction techniques for the exploratory analysis of microarray time-course data. This includes techniques that are not commonly used with microarray time-course data but are used with equivalent large-scale multidimensional data and could be adapted to be used with microarray time-course data.

4.1 Rescaling

Rescaling is a pre-process, common to most types of microarray data analysis. Eq. 4.1 describes how the data is transformed on a per-gene basis with rescaled gene values R being calculated as a function f of recorded expression values E. This transformation facilitates the comparison of recorded expression between genes across time-points. Qualities of the data that would normally retard such a comparison if the data were not rescaled, are between-gene variations in expression amplitude and the log-normal distribution [*109*] of values across the data.

$$R \leftarrow f(E)$$

Eq. 4.1 Rescaling

The most significant factor which impedes between-gene comparisons is the variation in recorded expression amplitude. This variation is due to a combination of variations in the sensitivity of microarray probes and variations in the actual levels of RNA produced by different genes. As these variations are normally of little interest to a biologist investigating microarray data, it is desirable for their influence to be removed before gene-gene comparisons are made.

The simplest form of rescaling is known as centering. This accounts for between-gene variations in expression amplitude by subtracting a measure specific to the recorded values of each gene (Eq. 4.2) so that rescaled values are either negative to indicate above normal expression or positive to indicate below normal expression[4]. The most common forms of centering are mean and median centering, which subtract either the mean or median of each genes recorded expression from each of its recorded expression values. While each of these measures employs a different concept of what is to be considered the 'normal' expression of a gene, in statistical terms the median measure can be thought of as more appropriate as it goes some way to account for the log-normal distribution of values.

[4] In the terminology of microarray data analysis, expression above and below normal expression is normally referred to as up and down regulation or over and under expression.

$$R(t) \leftarrow E(t) - f(E)$$

Eq. 4.2 Centering transform

Another group of transforms which rescale microarray data to account for between-gene variations in expression amplitude are linear transforms which divided gene values by a measure specific to the recorded values of each gene (Eq. 4.3). The measure by which a gene's expression values are divided can be the mean value of expression for that gene, its standard deviation, its root mean square or its typical expression in a control experiment. These, as with the measures used in centering, can be considered as representing different concepts of what can be considered the 'normal' expression of a gene. With linear rescaling, rescaled values are either below or above one to indicate activity below or above normal expression. Rescaled values of x (where x>1) relate to above normal activity conversely equivalent to below normal activity corresponding to rescaled values of 1/x. The main limitation of linear rescaling is that it fails to account for either the relative significance of changes in expression being dependent on absolute values or the logarithmic distribution of values across the data. Indeed, linear recalling will not only retain outlying high values that are symptomatic of a logarithmic distribution, but it will also introduce a set of high outliers where $f(E) \to 0$ making $R(t) \to \infty$. While these outliers can be suppressed by adding an arbitrary constant to *f(E)*, this will have the undesirable effect of altering the relationship between above/below expression values in the rescaled data as well as altering the level of expression for which above/ below normal expression is inferred by a rescaled value above or below one. Typically, when rescaling is required for an analysis technique that cannot handle outliers, *E(t)* and *f(E)* are incremented by the same small number. This means that the relationship between above/below expression in the rescaled data is approximately preserved and above/below normal expression is always indicated by a rescaled value above or below one.

$$R(t) \leftarrow E(t) / f(E)$$

Eq. 4.3 Linear rescaling transform

Logarithmic rescaling [*2, 110*] (Eq. 4.4) is the most commonly applied rescaling method for microarray data [*11, 12, 111-115*]. The advantage of log-rescaling is its

ability to quantify the significance of changes in expression values while accounting for the log-normal distribution of values across the data. Log rescaling makes each value in the rescaled time-series of a gene equivalent to its fold change from its normal value in the original data. This allows for comparisons across genes from which recorded expression can be related to biological activity and shifts the majority of the data towards normal distribution. A disadvantage of this form of rescaling is that, while suppressing outliers at the high end of the scale which are a symptom of log-distribution, it also introduces a new set of outliers where $E(t) \to 0$ makes $R(t) \to -\infty$. This can, however, be corrected in a similar manner to the outlier suppression correction of the linear rescaling scaling transforms (i.e. by incrementing *E(t)* and *f(E)* by the same small factor) so that the indication of above/below normal expression is preserved and the indication of the degree to which over/under expression occurs is approximate.

$$R(t) \leftarrow \log_2(E(t)/f(E))$$

Eq. 4.4 Log rescaling transform

In general there is no single correct way to rescale microarray time-series and the type of rescaling that is appropriate in any given scenario will depend largely of the type of analysis that is to be applied. Different analysis procedures require different aspects of the data to be accounted for and often analysis procedures will account for problematic aspects of the data themselves. Moreover, as rescaling essentially distorts the data and reduces its information content, the wrong type of rescaling may indeed render a particular type of analysis less effective.

4.2 Clustering

Clustering [*2*] is the most popular operation employed in the analysis of microarray data and the basis of most exploratory analysis of microarray time-course data. While the term clustering is often more specifically used to describe the procedure of applying algorithmic methods to partition the data into subsets (clusters) of genes, it can also be more broadly defined to include any procedure that provides a visual representation of the results from which groupings can be interpreted. These include principle component analysis, singular value decomposition and self organising maps.

A conceptualisation common to many forms of clustering is the notion of a multidimensional expression space where cell samples (time-points) relate to different dimensions and the position of any gene along a dimensional axes corresponds to its recorded expression in that sample (Figure 4.1). This is often described as an n-dimensional expression space where n denotes the number of dimensions. As an extension to this concept genes are often described as data having a position in expression space with the recorded expression of a gene described as its vector. Some clustering algorithms also use vectors that either belong to hypothetical genes or are not assigned to genes at all. In general, techniques for the clustering of microarray data tend to employ a significant level of multidimensional geometry which is, at times, difficult to relate to the actual objectives of the data analysis until the results are viewed.

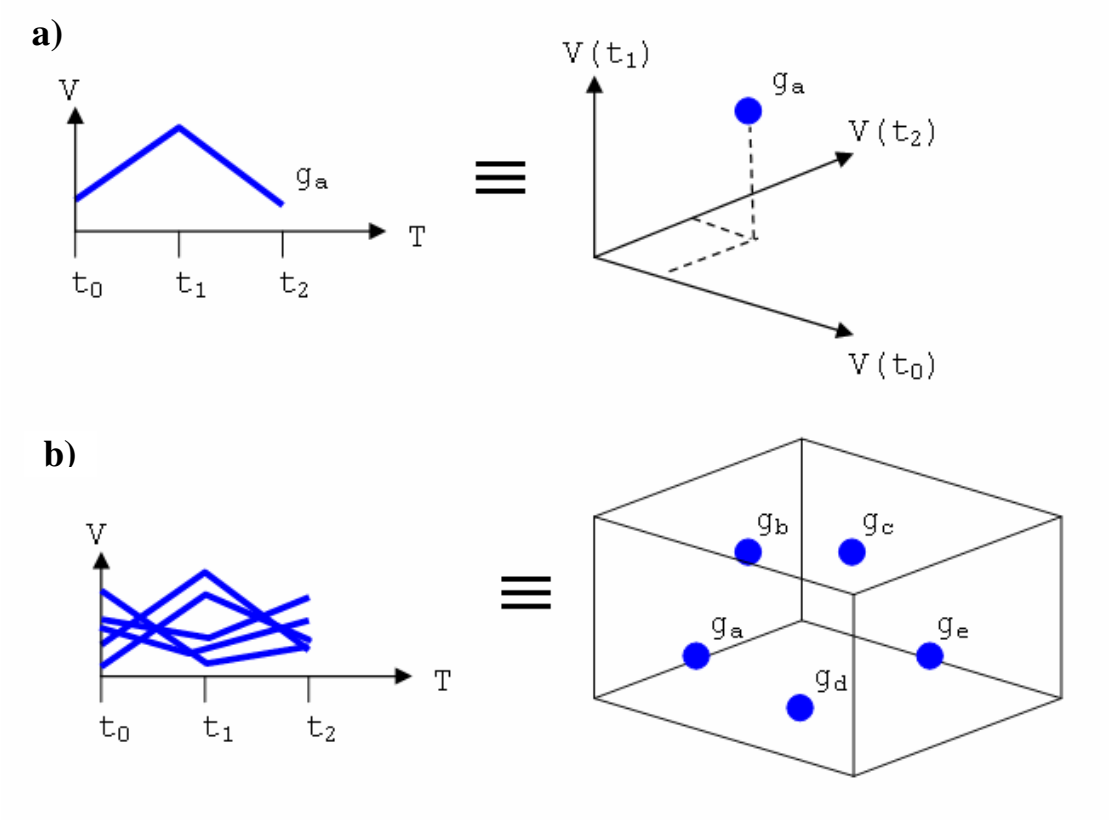

Figure 4.1. ***Mapping of recorded expression values to n-dimensional expression space: top) The three recorded expression values of a single gene are used to position it in a three dimensional expression space and bottom) multiple genes positioned in expression space.***

4.2.2 Principle Component Analysis

Principle component analysis[5] (PCA) [*116*] is one of the most basic forms of clustering and relies heavily on the concept of n-dimensional expression space. In the case of microarray data analysis [*113*] PCA describes the data in terms of its variance by performing a linear mapping of data points (genes) from n-dimensional expression space to a d-dimensional display space. The PCA algorithm begins by finding the intersection through expression space with the maximum variation – this is principal component 1 (PC1) which can be thought of as describing most of the data. Principal component 2 (PC2) is the intersection of maximum variation that lies perpendicular to PC1. This can be thought of as describing most of the rest of the data. If a third principle component is required then it will be the line with maximum variation lying perpendicular to both PC 1 and PC2. Typically PC1 is plotted against PC2 in a two-dimensional visual representation of the results (Figure 4.2 left) or PC1

[5] Occasionally known as the (discrete) Karhunen-Leove transform or the Hotelling transform.

is plotted against PC2 and PC3 in a three-dimensional representation (Figure 4.2 right). This technique works well to group genes only if all natural groupings are dispersed along the principle components. If features are not dispersed along the principle components, they are discounted regardless of any potential biological significance.

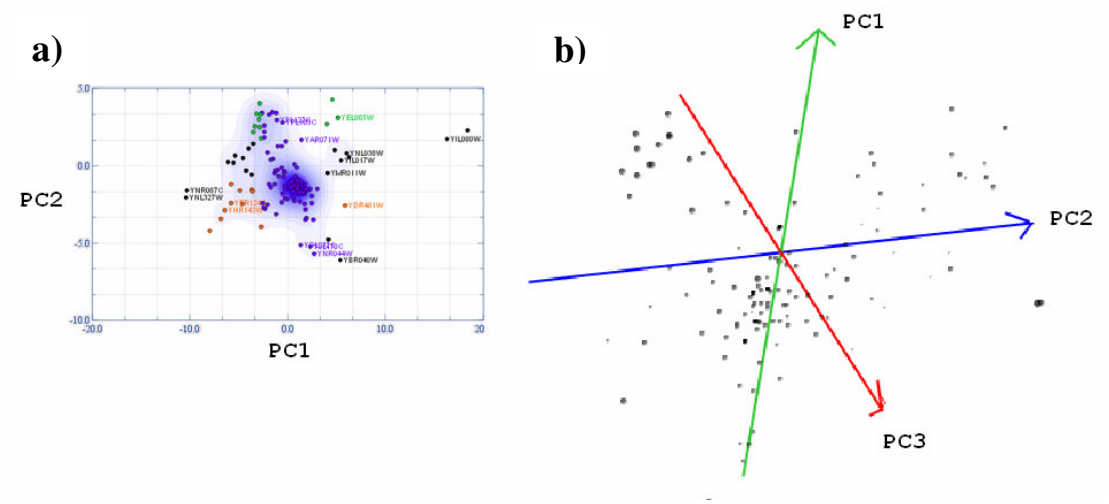

Figure 4.2. ***Two representations of principle component analysis (PCA) results: left) PC1 is plotted against PC2 in a two-dimensional visual representation (the blue colour gradient indicates the density of gene representations in the display) and right) PC1 is plotted against PC2 and PC3 in a three-dimensional representation (taken from [117]).***

4.2.3 Similarity and Dissimilarity Measures

A feature common to most other types of clustering is that they group genes according to some pre-defined measure of similarity or dissimilarity [2]. These measures are of the form of Eq. 4.5 combining the recorded expression values of two genes, *i* and *j*, for *n* samples (time-points) to return a single real number quantifying their similarity or dissimilarity. While dissimilarity and similarity are complimentary concepts and it is normally easy to convert a measure from one form to another, it is dissimilarity measures, and in particular distance measures which are a subgroup of dissimilarity measures, that are more commonly used in the analysis of microarray data. To be classified as a dissimilarity measure a measure must satisfy the condition that $f(i,j)>f(k,l)$ if, and only if, *i* is less similar to *j* than *k* is to *l*. To be classified as a distance measure, a measure must satisfy four conditions [2]. These are:

1. The distance must be positive definite, $f(i,j) \geq 0$ (i.e. greater or equal to zero).
2. The distance must be symmetric, $f(i,j)=f(j,i)$, so that the distance from *i* to *j* is the same as that from *j* to *i*.
3. An object must have zero distance from itself, $f(i,j)=0$.
4. When considering three objects i, j and k, the distance from *i* to *k* is always less than or equal to the sum of the distances from *i* to *j*, and the distance from *j* to *k*, $f(i,k) \leq f(i,j) + f(j,k)$ (this is often referred to as the triangle rule).

Most of the measures that are used with microarray data satisfy only the first three of these conditions. These are classified as semi-metric distance measures while measures that satisfy all four are classified as metric.

$$f(i, j) \rightarrow \mathbf{R} \text{ where } i, j \in \mathbf{R}^n$$

Eq. 4.5 The general form of a similarity measure

One of the most common similarity measures used with microarray data is the metric distance measure Euclidean distance (Eq. 4.6). This is a generalisation of the familiar Pythagorean theorem applied in n-dimensional expression space to return the straight-line distance between two genes. Another similar metric distance measure that yields similar results to the Euclidian distance measure is Manhattan distance (also known as taxi-cab or city block distance) which returns the sum of all differences between values in the different samples. Both these measures can be thought of as providing a basic indication of positive correlation.

$$d(i, j) = \sqrt{\sum_{x=1}^{n} (i_x - j_x)^2}$$

Eq. 4.6 Euclidean distance

The most common semi-metric distance measure in use with microarray data, and probably the most common of all similarity measures, is Pearson's correlation coefficient (Eq. 4.7). This is similar to Euclidean distance with the exception that it accounts for variable expression amplitude between genes and calculates the similarity between the *shapes* of recorded expression patterns rather than the similarity between absolute values. In this respect, the output of Pearson's correlation

coefficient can be thought of as equivalent to the output of the Euclidean distance measure after the data has been subject to some form of linear rescaling (see section 4.1).

$$d(i,j) = \frac{\sum_{x=1}^{n}(i_x - \bar{i})(j_x - \bar{j})}{\sqrt{\sum_{x=1}^{n}(i_x - \bar{i})^2}\sqrt{\sum_{x=1}^{n}(j_x - \bar{j})^2}}$$

Eq. 4.7 Pearson's correlation coefficient

Other (dis)similarity measures attempt to communicate a number of different aspects of similarity. These include phase shifted similarity, common binary patterns, mutual information [*118*], covariance , similar maximum or minimum differences etc. Indeed, there are a large number of (dis)similarity measures currently in use with microarray data and most of these carry their own notion of similarity and attempt to bring out a different aspects of similarity. In general it is accepted that each different (dis)similarity measure will have its own advantages and the type of measure used will largely depend on the objectives of the analysis being performed.

4.2.4 Singular Value Decomposition

Singular value decomposition (SVD) [*119*] is a mathematical procedure which uses similarity measures in the display of its results. In the case of microarray data SVD generates a group of eigengenes that summarise the major trends in expression over the entire time-course. If an eigengene correlates with a suspected artefact in the data then that artefact can be deduced from the dataset and genes whose expression correlates strongly with the eigengene can be associated with that artefact. Typical representations of SVD results include value against time graphs for different eigengenes (Figure 4.3 left) and scatter-plots where genes are represented as single points with their translation along the X and Y axes corresponding to the similarity of their recorded expression to the patterns described by two of the most significant eigengenes (Figure 4.3 right).

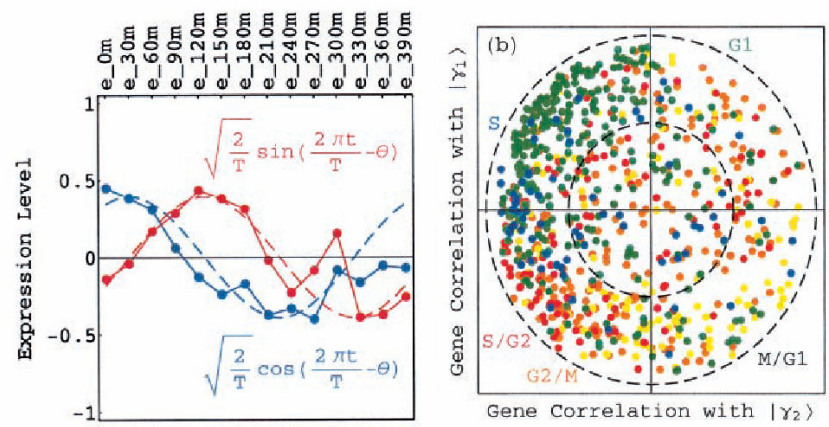

Figure 4.3. ***The results of singular value decomposition: left) Eigengenes summarise trends in expression across all genes and right) genes are plotted according the correlation between their recorded expression and the patterns of the eigengenes (genes are colour coded according to classifications by Spellman et.al [111], images are taken from [119]).***

4.2.5 Radviz

The Radviz dimensional reduction technique [*120, 121*] (see Figure 4.4) uses graphic primitives known as dimensional anchors spaced evenly round the circumference of a circle. Each anchor corresponds to an attribute (dimension) of the data and data items are represented as points within the circumference of the circle. Item positions are calculated using a spring algorithm. Here, an item is positioned at the centre of the circle by default and each dimensional anchor exerts a force to attract each item with the force's magnitude proportional to the value the item has for the dimensional anchor's corresponding attribute (as if springs were attached between each item and each of the anchors). The spring algorithm adjusts the position of items until the forces on any given item are resolved. This tends to move items towards anchors corresponding to attributes for which they have a relatively high value. Used with microarray time-course data, dimensions and dimensional anchors would relate to time-point and item points would describe the activity of genes.

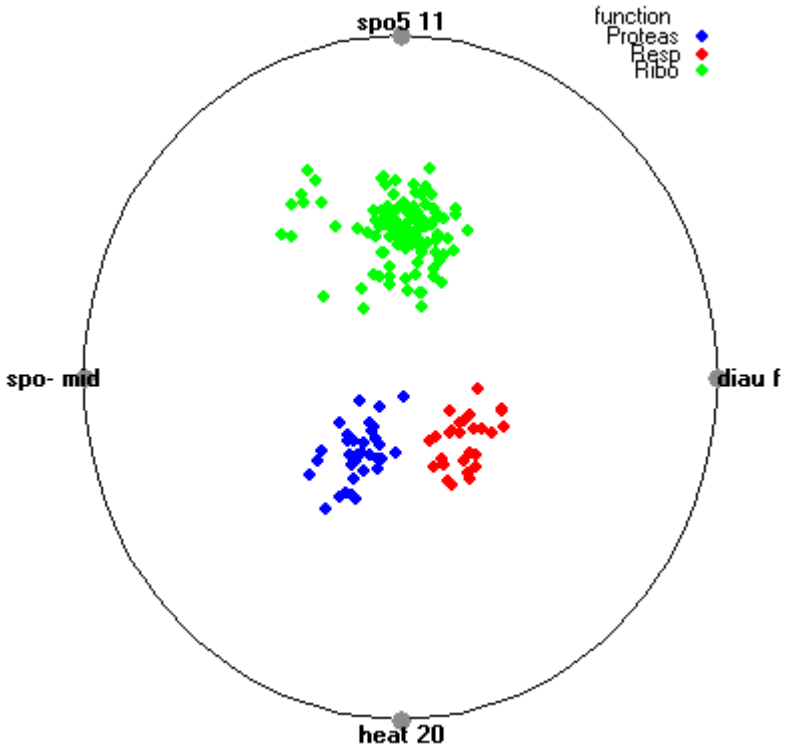

Figure 4.4. ***Radviz: points around the circumference of the circle are known as dimensional anchors. Points within the circle correspond to data items. Representations of data items are positioned (In this case items correspond to genes colour-coded according to predefined gene classifications and anchors relate to samples for which those genes activity has been recorded, image taken from [122]).***

The main problem of the Radviz display is that results may be ambiguous, especially when the dimensionality of the data is high. There are two main types of cases where this might occur. Firstly, if two attributes, *i* and *j*, have their corresponding anchors opposite each other across the centre-point of the circle and two items, *a* and *b*, have a similar ratio of *i* to *j* then they will be at a similar position in the display regardless of their actual values of *i* and *j*. Other types of ambiguity comes into play if the data set has a higher dimensionality where it will be unclear as to which degree each anchor acts to determine the position of each item. While similar attributes' anchors can be placed together to make this ambiguity less of a problem it cannot reduce the difficulty associated with differentiating between the attributes of different items and this problem will necessarily worsen when more unrelated dimensional anchors are introduced. In reality Radviz is only really effective at presenting clusters of rough ratios between smaller numbers of groups of tightly correlated dimensions and

cannot be effective when used with microarray time-course data with larger numbers of attributes.

4.2.6 The Similarity Matrix

A more common use of (dis)similarity measures is the creation of a similarity matrix. Here both rows and columns correspond to the full list of genes while cells reflect inter-gene (dis)similarities calculated according to a single (dis)similarity measure. Figure 4.5 left shows a direct visualisation of a similarity matrix [*123*] with the colour of the cells corresponding to the level of similarity (red for high and green for low). While Figure 4.5 right where the genes are ordered according to similarity reveals a general pattern in the data and could be of some interest to a biologist involved in the analysis of that data, this type of visualisation is rare and the similarity matrix is more commonly used to support other forms of cluster analysis.

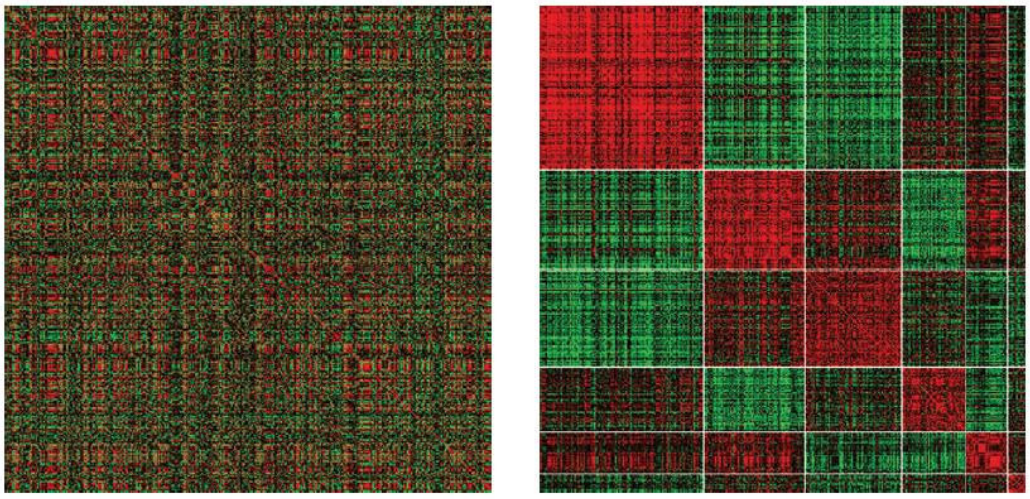

Figure 4.5. ***Direct visualisation of a similarity matrix: left) Genes are unordered. right) Genes are ordered so that genes and groups of genes with similar recorded expression are close together. This causes large blocks of light coloured cells along the top-left to bottom-right diagonal to appear. This shows large groups of genes with similar recorded expression (taken from [123]).***

4.2.7 Relevance Networks Visualisation

The Relevance Networks technique [*118*] (Figure 4.6) specifically employs a shared information content similarity measure and conceptualises the resultant similarity matrix as a network structure where genes correspond to nodes and inter-gene similarities correspond to links. This network is visualised using standard graph

layout algorithms to represent a subset of nodes and only those links where the relevant similarity is above a certain threshold. A limitation of this representation is that there are no axes and no way to relate the position of gene representations to recorded expression values.

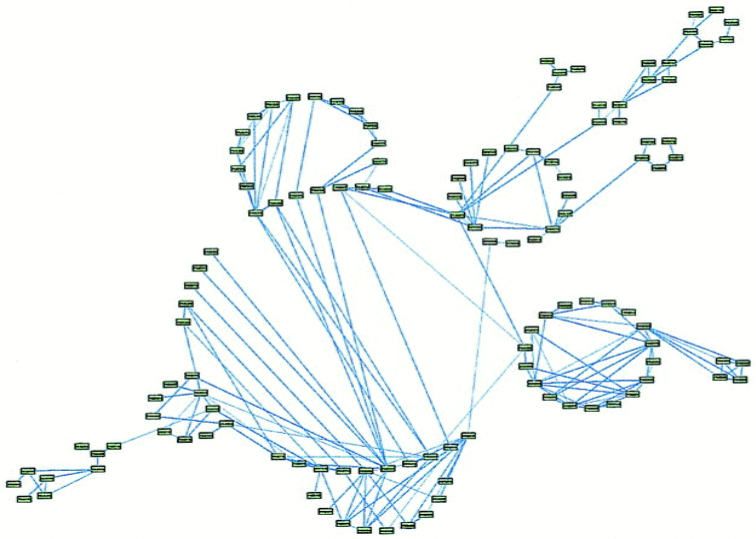

Figure 4.6. ***Relevance networks visualisation of a similarity matrix (taken from [118]).***

4.2.8 Multidimensional Scaling

Another technique that uses the similarity matrix to visualise microarray data is multidimensional scaling (MDS, Figure 4.7). This represents genes as dots and places them within a 2d or 3d display space so that their proximity corresponds to their calculated similarity. As the similarity matrix exists in a higher dimensional space than the display space it is impossible to make proximity proportional to similarity in all cases. Instead, multidimensional scaling produces a representation where the proximity of any two genes is approximate to their calculated similarity. As with relevance networks, the disadvantage of multidimensional scaling displays is that there are no axes and no way to relate the position of gene representations to expression values. This is perhaps why the multidimensional scaling of microarray data is more commonly employed in the clustering of samples [*124-126*] and rarely for the clustering of genes [*127, 128*].

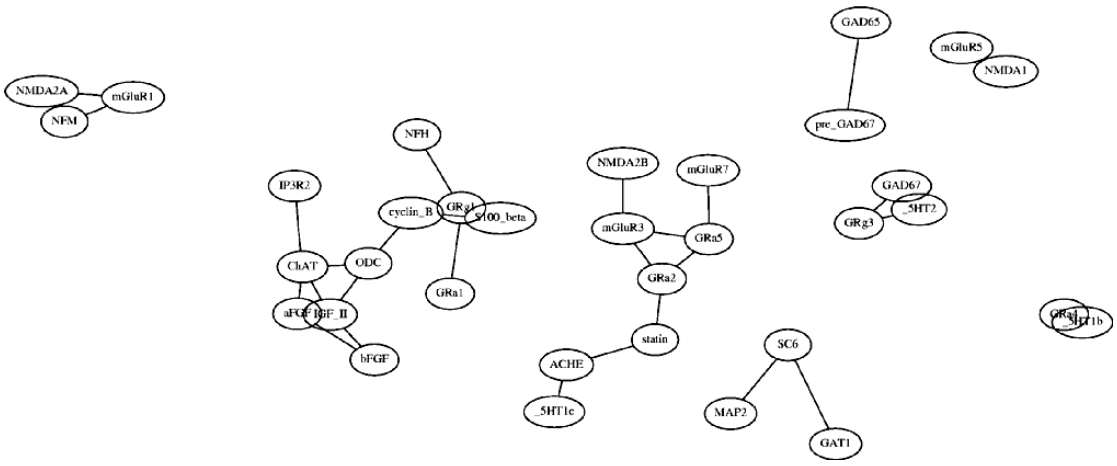

Figure 4.7. ***Multidimensional scaling of genes. Pairs of genes for which similarity is above a certain threshold are linked by an edge (taken from [127]).***

4.2.9 Hierarchical Clustering

By far the most common form of cluster analysis is hierarchical clustering [*111*]. This is an algorithmic method which partitions the data to produce nested clusters adhering to a hierarchical structure. At the base of the hierarchy are individual genes while at the top is a single super-cluster containing all genes. Higher levels of the hierarchy include clusters with higher numbers of genes and clusters on the same level always include the same number of genes. All clusters are conglomerates of lower level clusters with no two clusters on the same level including the same gene. Hierarchical clustering algorithms are either divisive (constructed from the top layer down) or, more commonly, agglomerative (constructed from the bottom layer up).

The basic process of agglomerative clustering begins by considering each gene as a cluster and successively combining the most similar cluster to form a cluster containing multiple genes. This continues until all the genes are combined in a single cluster. After this the clusters formed at each iteration of the cycle can be combined to classify genes at different levels of a hierarchical tree structure. When a cluster contains multiple genes the distance from that cluster to any other cluster can be calculated in a number of different ways. These include:

1. Single linkage which rules that the similarity between two clusters is the maximum similarity between any gene in the first cluster and any gene in the second.

2. Complete linkage which is the opposite of single linkage and rules that the similarity between two clusters is the minimum similarity between any gene in the first cluster and any gene in the second.
3. Average linkage (sometimes referred to as unweighted pair-group average linkage) which rules that the similarity between two clusters is the average of all the similarities between genes in the first cluster and genes in the second.
4. Weighted pair-group average linkage. This is similar to the average linkage method with the exception that the size of respective clusters is used as a weight.
5. Pair-group centroid linkage. As with Euclidean distance, each sample of the experiment is conceptualised as a dimension of n-dimensional space with the value for a gene in any given sample considered as its translation along the relevant dimensional axes. The similarity of two clusters is calculated as the similarity between their geometric centroids [*109*] within the n-dimensional space.
6. Weighted pair-group centroid linkage. Similar to the centroid linkage method with the exception that the size of respective clusters is used as a weight.
7. Ward's method. This method attempts to minimise the sum of squares of any two (hypothetical) clusters formed at each iteration of the clustering cycle.

These methods have their various advantages and disadvantages. For example, single linkage does well at detecting long chained clusters of genes but suffers from the disadvantage that the close similarity of two genes will force the amalgamation of two, otherwise dissimilar, clusters. Complete linkage avoids this disadvantage but is unable to detect chained clusters. Average linkage is the most popular option. Despite being more computationally expensive it avoids undesirable phenomena caused by genes with outlying patterns of expression dominating the output. Centroid linkage produces similar results with the weighted versions being more effective at revealing clusters of irregular size. Ward's method is more efficient at revealing tightly grouped small sized clusters.

Divisive methods for hierarchical clustering include Tree Structural Vector Quantization, the MacNaughton Smith Algorithm and 2-means clustering. These begin with a group containing all genes which is successively broken down into

subsidiary groups until the remaining groups are populated by a single gene. In general these methods are more computationally expensive but are more effective at preserving the higher-level structure of the data and finding larger natural groupings.

The most common visual representation of microarray data uses the results of hierarchical clustering to produce a combined heat-map/dendrogram display [*111*] (Figure 4.8). In the heat-map (also known as a colour mosaic), expression levels are colour-coded and displayed in a grid with rows corresponding to genes and columns corresponding to conditions or time-points (depending on the type of microarray experiment). Genes are ordered so that the groups defined by hierarchical clustering are unbroken. The dendrogram is a type of binary tree which is attached to the left-hand-side of the colour-mosaic to illustrate the groupings defined.

Distance trees are an alternative display of hierarchical clusters which disregard the colour mosaic representation in preference to a more informative dendrogram. These employ tree structure layout algorithms that allow branches to have variable lengths and, in the display of microarray data, they can be configured so that the length of the branches joining endpoints (genes) can be made approximately proportional to their dissimilarity (Figure 4.9). The two types of distance trees in use with microarray data are horizontal [*129, 130*] (Figure 4.9a) and radial [*95, 131*] (Figure 4.9b) While radial trees are more flexible and allow for a more accurate relationship between branch distance and inter-gene dissimilarity, horizontal distance trees accommodate gene labels which can be placed parallel to each other at the endpoint of each branch.

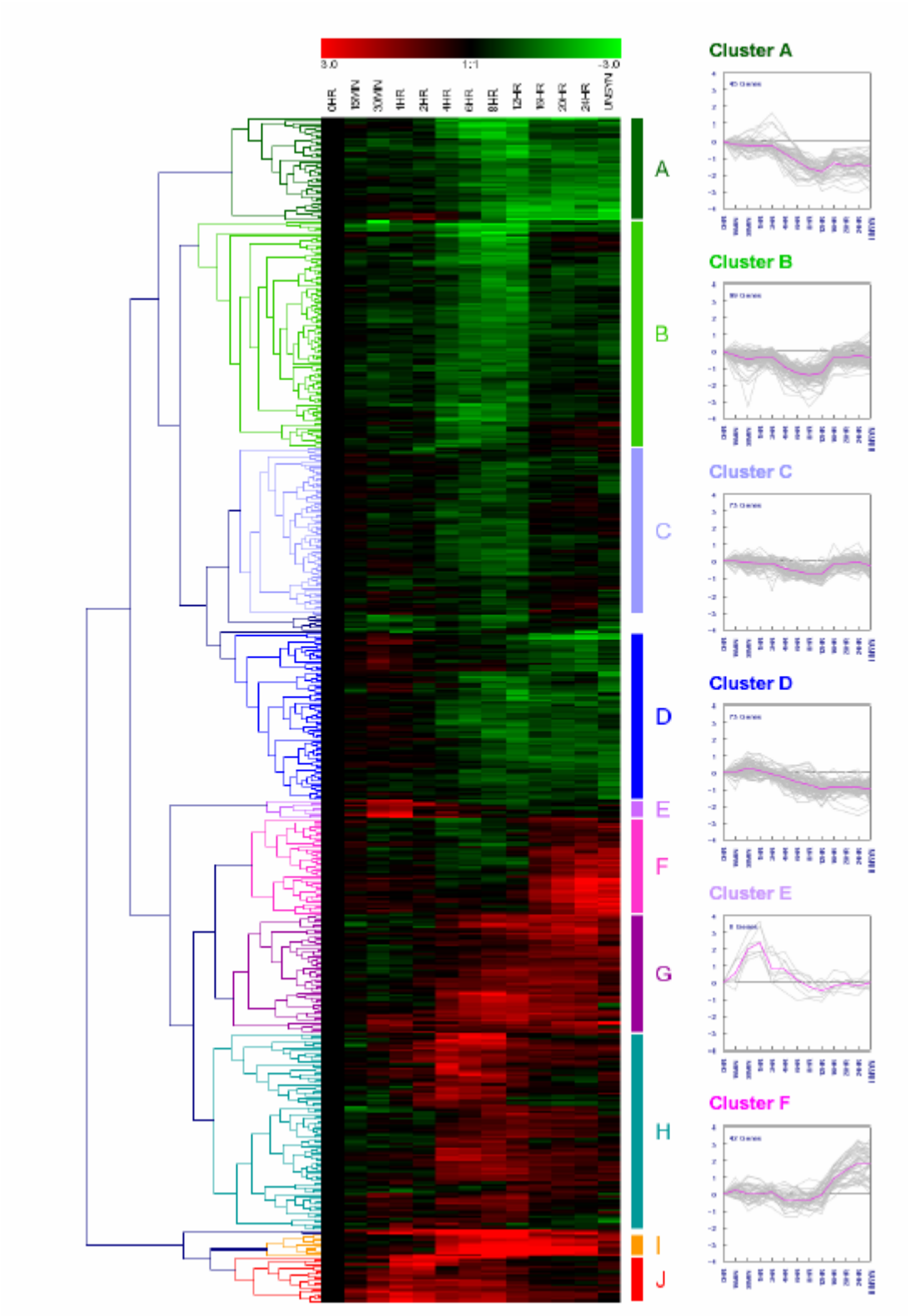

Figure 4.8. ***Heat-map/dendrogram visualisation of gene expression (taken from [132]).***

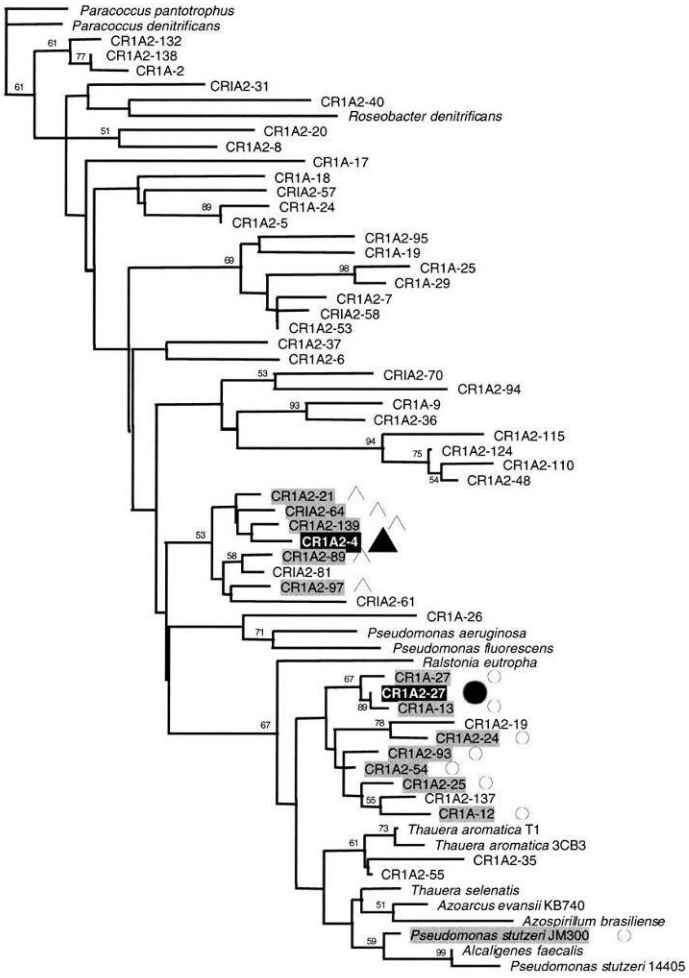

Figure 4.9. ***Horizontal distance tree representations of hierarchical clustering results with symbols representing gene classifications (taken from [130]).***

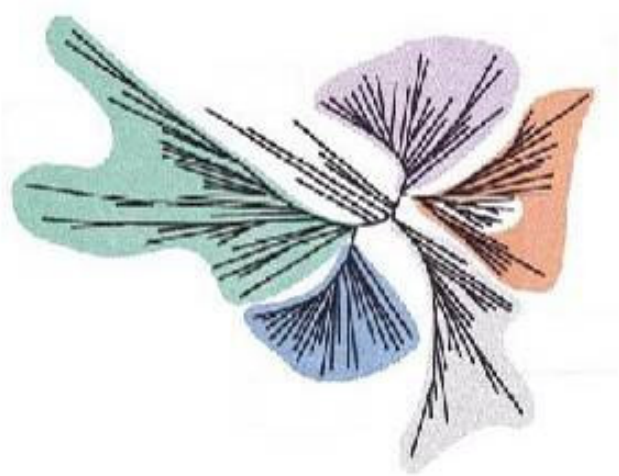

Figure 4.10. ***Radial distance tree representations of hierarchical clustering results with five main visual clusters colour coded (taken from [95]).***

4.2.10 K-means Clustering

The next most popular method of clustering for microarray data is K-means clustering. This is a divisive form of clustering which requires the user to indicate the number of clusters (K) to be formed and assign an initial centroid (i.e. a typical gene or position in n-dimensional expression space) to each cluster. As the number of natural clusters and typical genes belonging to these clusters would almost certainly not be obvious from the raw data, another method such as agglomerative hierarchical clustering can be used to give an indication of these details. Once these centroid genes are chosen, the k-means algorithm can be run to execute the following steps:

Step 1. Each non-centroid gene is assigned to the cluster whose centroid it is most similar to.

Step 2. Each centroid is recalculated according to the genes in its cluster.

Step 3. Genes are reassigned to a different cluster if they are more similar to the centroid of that cluster than to that of their own.

Step 4. Repeat stages 2 and 3 until no more genes require to be reassigned.

The advantages of k-means clustering is that it less computationally expensive than hierarchical clustering and, if the correct number of natural clusters are selected, these clusters will be more accurate and relevant than those created by hierarchical clustering. The disadvantage of k-means clustering is that it returns a limited number of clusters and results are highly dependent on the user's initial selection of cluster centroids.

A useful extension to K-means clustering is fuzzy K-means clustering [*133*]. When applied to microarray data (as in [*134*]) this allows genes to be assigned to multiple clusters. The general process of fuzzy k-means clustering is as follows:

Step 1. K/3 centroids are calculated as the first three principle components of the data (see PCA above).

Step 2. Centroids are refined by iteratively recalculating the gene-cluster membership and updating centroid positions until no more genes require to be reassigned.

Step 3. Each gene is recorded as belonging to either one of the K/3 clusters.

Step 4. Genes that with a similarity above a certain threshold are removed from the data.

Step 5. Steps 1 to 3 are repeated for two more rounds of fuzzy clustering.

The advantage of fuzzy clustering is that the assignment of genes to multiple clusters accounts for the fact that a gene can be associated with multiple biological processes and it may be associated with multiple processes in the data. There is, however, still the disadvantage that results are dependent on the user's specification of the number of centroids and there is the additional difficulty that a larger number of results will be more difficult to display effectively.

In general the results of k-means and fuzzy k-means clustering are visualised using separate heat-maps and genes ordered within clusters according to a hierarchical clustering algorithm (Figure 4.11 left). It is also common to find the results displayed using separate overlaid value versus time plots of each cluster (Figure 4.11 right).

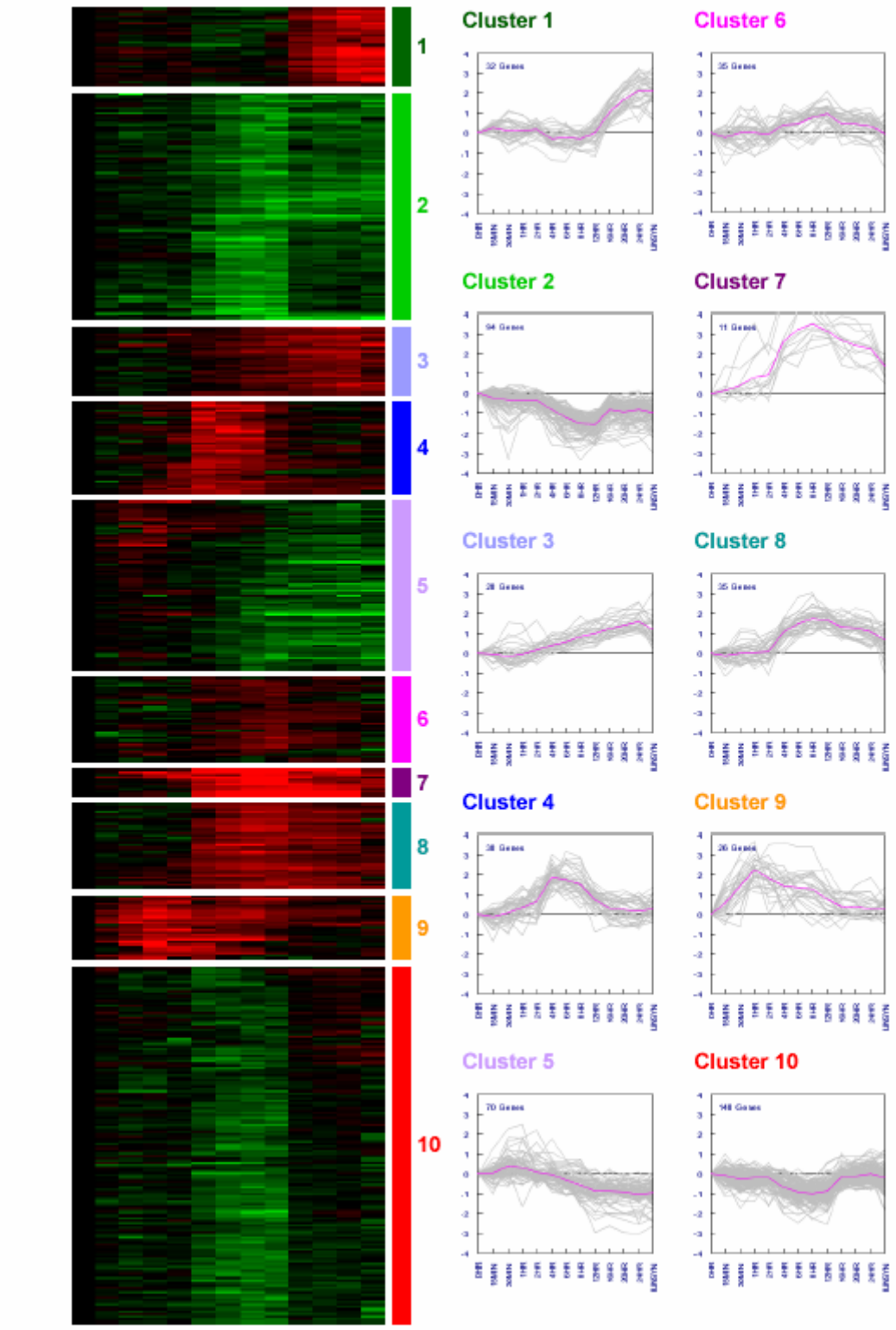

Figure 4.11. ***The results of K-means clustering: left) Separate heat-maps for each cluster and right) Separate graph views for each cluster (taken from [132]).***

4.2.11 Self-Organizing Maps

Self-Organizing Maps (SOM) [*135*, *112*] is a clustering method similar to K-means clustering that imposes a structure on the data by having centroids positioned in a two dimensional grid. This grid contains a predefined number of cells (sometimes called neurons) laid out in a fixed arrangement. Cells are normally hexagonal or square and the number of cells determines the granularity of the resulting mapping. This affects the accuracy and the generalisation capability of the SOM. The process of generating a Self Organizing Map is as follows:

Step 1. Initialisation: Centroids are assigned to nodes (These are normally random points in n-dimensional expression space or the position of random genes but there are more ordered methods for assigning centroids).

Step 2. A random gene (X) is chosen and the most similar centroid (this is called the best-matching unit or BMU) is selected.

Step 3. The BMU and its topological neighbours (i.e. the centroids assigned to map cells neighbouring that of the BMU) are moved closer to X in n-dimensional expression space.

Step 4. Steps 2 and 3 are repeated for 20,000 to 50,000 iterations (Initially BMU and BMU neighbours are shifted by large amounts then smaller amounts to fine tune the output). The process, known as the training phase, causes neighbouring centroids to become close together in expression space and the centroids, as a collective, to approximate a dimensionally reduced summary of the original data.

Step 5. Finally, each gene is assigned to the centroid to which it is most similar and the results are visualised so that clusters can be interpreted.

As a cluster can be defined as group of genes that are similar to each other but dissimilar to all other genes, SOM visualisations can communicate clusters by displaying the cells of a mapping (which will contain similar genes) together with some indication of inter-centroid similarity. The most common way this is done uses the Unified Distance Matrix (U-matrix) [*136*]. This contains the dissimilarities between all centroids and the centroids of neighbouring map units. In SOM visualisations this matrix is overlaid onto the map structure by colour coding the edges of cells (or new cells placed between the edges of centroid cells) according to

centroid dissimilarity of relevant neighbours and colour coding the body of cells according to the mean value of its edges (Figure 4.12). The Kraaijveld method [*137*] is a somewhat less complicated solution where only a single attribute is plotted for each cell. This is the maximum dissimilarity between the cell's centroid and the genes that are associated with that centroid.

SOMs can be thought of as being similar to MDS as both algorithmically map points in n-dimensional expression space to d-dimensional display space. SOMs are, however, more effective at grouping similar items while MDS is effective in preserving the structure of clusters [*138*]. This makes SOMs the preferred alternative in microarray data analysis where it is often the former objective that has the higher priority.

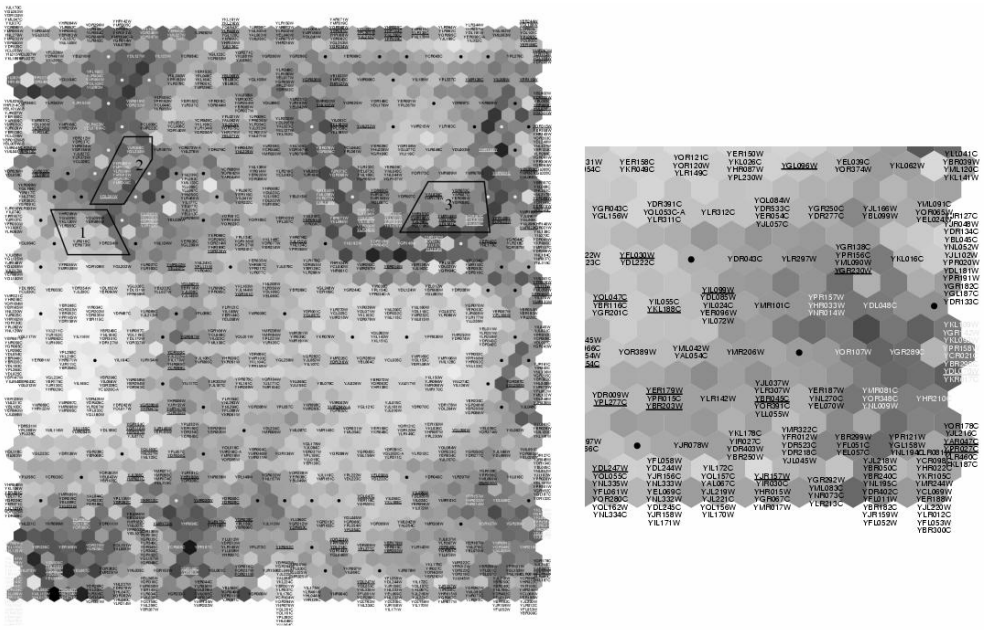

Figure 4.12. ***Self-organizing Maps visualisation of microarray data using U-distance matrix and colour coded so that the lighter the cell the higher the similarity: left) The overview shows clusters of genes (light coloured) and right) a detail view with centroid cells (with genes labelled or a dot to indicate they are empty) surrounded by cells indicating levels of inter-centroid similarity (taken from [139]).***

An interesting extension to static displays of SOM results is the animated SOM display of the Gene Expression Dynamics Investigator application (GEDI) [*140*]. This allows biologists to animate through a time-course with cells of the SOM colour-coded according to their value (that is, the value of the centroid if it were mapped back to a pattern of expression) at each point in time (Figure 4.13). The idea

behind such an animation is that it should allow for an easy association to be made between clusters and the patterns of expression they represent. Figure 4.13 shows some a collection of screenshots from GEDI illustrating cyclic patterns of gene expression during the HeLa cell cycle [*141*].

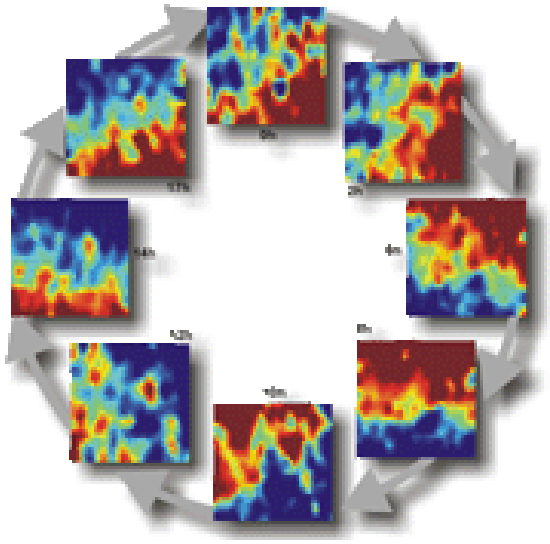

Figure 4.13. ***A collection of screenshots from the GEDI visualisation of animated Self-organizing Maps. Each square is a frame of the animation and the arrows indicate the order of frames. Within frames, red indicates high expression and blue indicates low expression (taken from [140]).***

4.2.12 Gene Shaving

Gene Shaving (also known as Principle Component Shaving) [*142*] is similar to Fuzzy K-means clustering in that it allows genes to be assigned to multiple clusters. The primary goal of gene shaving is to find small blocks of genes that may overlap. These blocks are chosen progressively to be the most dissimilar groupings which have high inter-gene similarities. The procedure of gene shaving is as follows:

Step 1. The first principle component of the data is found (see above, Principle Component Analysis).

Step 2. 10% of the genes with the smallest translation along this component are removed (shaved) from the data.

Step 3. Stages one and two are repeated until only a single gene is left. The result of these iterations is a group of nested gene clusters,
$S_n \supset S_{k1} \supset S_{k2} \supset S_{k3} \supset S_{k4} \ldots$ where S_k contains k genes.

Step 4. From the nested gene clusters, a cluster (S_k) with both high-variance and high inter-gene similarities (the best cluster is determined using a measure known as the gap statistic) is selected for output.

Step 5. Each gene in the original data set is orthogonalised with respect to the average position of genes in S_k. Put simply; this transforms the position of genes in n-dimensional expression space so that subsequent clusters formed in stages 1 to 4 will be uncorrelated with the average position of genes in the output clusters.

Step 6. Steps 1 to 5 are repeated until no more interesting output clusters are found.

The overlapping clusters product of Gene Shaving is, in terms of data structure, very similar to that of Fuzzy K-means clustering with the exception that clusters are ordered. It is, therefore, appropriate that the results are displayed in a similar manner to those of K-means clustering (Figure 4.11). As discussed above, the disadvantage of this display is that a larger number of results as overlapping clusters mean that the expression of some genes are required to be displayed twice.

4.2.13 Plaid Models

The Plaid Models technique [*143*] is similar to Fuzzy K-means clustering and Gene Shaving in that it allows genes to belong to multiple clusters. The advantage of the Plaid Models technique is that it does not require the expression of any genes to be displayed more than once. Instead, the recorded expression of genes is modelled and the output of the data model is presented as a heat-map so that multiple gene-cluster membership can be interpreted from the resultant display (Figure 4.14). The advantage of displaying modelled data, rather than the original data, is that the structure of the data is preserved and noise is reduced. This means that patterns of similar expression are more easily detected and the effort required to perceive associations between genes that are separated in the display is reduced. As individual genes can be associated with multiple separated patterns of activity, it is therefore

possible to associate genes with multiple visual clusters without their expression patterns being displayed more than once.

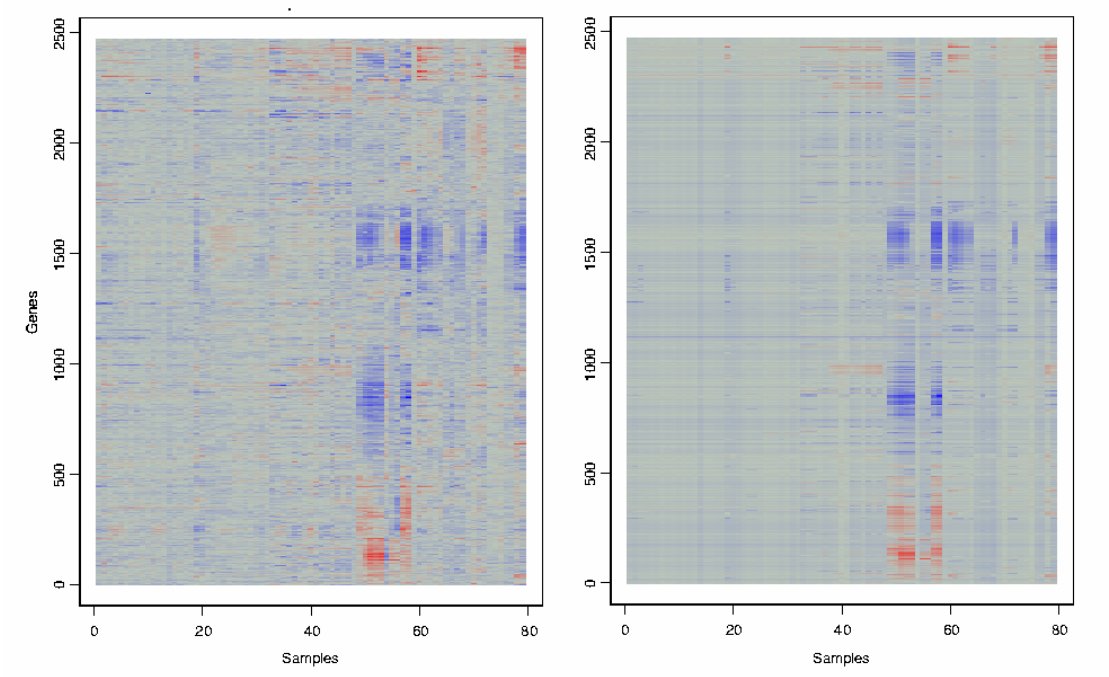

Figure 4.14. ***Plaid modelling display of microarray data: left) The original data and right) the modelled data (taken from [143]).***

4.2.14 Evaluating Clustering Methods

The relative effectiveness of different clustering algorithms is difficult to assess for a number of reasons. The first of these are the subjectivity of significance in visually represented results and the variation in the type of result produced by different techniques. As different techniques present clusters of genes with varying degrees of distinction it is often difficult to assess what actually constitutes a cluster in any given technique. Besides this, even when clusters are absolutely distinct, as they are with K-means clustering or Gene shaving, there is no definitive concept of what constitutes a good or bad cluster. Indeed, the notion of a good or bad cluster becomes redundant when one considers that the type of clusters that a particular biologist wishes to find will be largely determined by their own particular objectives and it is likely that different biologists will want to use different types of clustering at different stages of their analysis. This tendency to switch between different types of clustering is evident by the number of academic papers which describe multiple analysis techniques applied to the same data, for example [*94, 144*], and software tools designed to support this process [*12, 114, 115, 123, 132*]). So, while there are

techniques which attempt to quantify the effectiveness of different clustering techniques (these either attempt to statistically assess the quality of clusters [*145*] or evaluate their correlation with pre-defined classifications [*146, 147*]) it is more appropriate, at least in the context of this thesis, to consider the relative functionality of different techniques and assess their effectiveness with regard to other related techniques with similar functionality. Here, clustering techniques are classified as belonging to one of three distinct groups. These are; clustering methods that place genes within a hierarchy of nested clusters (hierarchical clustering), methods that allow genes to belong to multiple clusters (fuzzy k-means, gene shaving and plaid models) and methods that allow genes to be clustered visually across two or three dimensions of an actual or virtual display space (MDS, SOM, PCA and SVD).

The effectiveness of any given hierarchical clustering method will depend on the particular type of clusters the clustering algorithm finds, how the results are displayed and, ultimately, how well the perceived clusters match the type of cluster the biologist wants to find. The relative advantages and disadvantages of the different hierarchical clustering methods with regard to the type of clusters they produce are summarised in Table 4.1.

The three basic methods for displaying the results of hierarchical clustering are heat map/dendrogram displays, horizontal distance trees and radial distance trees. The relative advantages and disadvantages of each of these methods are summarised in Table 4.2.

Methods that allow genes to belong to multiple clusters are fuzzy k-means, gene shaving and plaid models. The general advantage of these techniques is that with genes being allowed to belong to multiple clusters, they can be seen to contribute to multiple patterns in the data to reflect the assumption that a gene can participate in more than one biological process that can be inferred from the data. The relative advantages and disadvantages of each of these methods are summarised in Table 4.3.

Methods that allow clusters to form over two or three dimensions are PCA, SVD, MDS and SOM. The general advantage of this type of method is that genes can be communicated as having less rigid associations with more clusters. For example, if a

gene (x) is positioned on a one dimensional surface it can be placed between two clusters (A and B) to be associated with these two clusters. If a third cluster (C) is added then it becomes unclear as to the degree by which the position of gene x is governed by its association between clusters A, B or C. If, however, the same gene is placed on a two dimensional surface one can determine, to a greater extent, its relative degree of association with three clusters A, B and C by observing its distance from each cluster. The relative advantages and disadvantages of each two or three dimensional clustering method are summarised in Table 4.4.

Table 4.1. Summary of clusters produced by hierarchical clustering algorithms.

Type	Advantages	Disadvantages
Agglomerative (General)	More suitable for finding smaller natural groupings and computationally inexpensive.	Can destroy the higher level structure of the data.
Agglomerative - Single linkage	Can find long chained clusters.	Close similarity of two genes will force the amalgamation of two, otherwise dissimilar, clusters. This is known as the 'chaining phenomenon'.
Agglomerative - Complete linkage	Close similarity of two genes will not force the amalgamation of dissimilar clusters.	Sensitive to outliers which will prevent similar clusters from amalgamating.
Agglomerative - Average linkage and pair-group centroid linkage	Avoids the chaining phenomenon without being sensitive to outliers.	Computationally expensive.
Agglomerative - Weighted pair-group average linkage and weighted pair-group centroid linkage	Effective at revealing clusters of irregular size	Computationally expensive.
Agglomerative - Ward's method	More efficient at revealing even small sized clusters.	Computationally expensive.
Divisive – Tree Structural Vector Quantization, MacNaughton Smith Algorithm and 2-means clustering	Preserves the higher-level structure of the data and good at finding larger natural groupings.	Less suitable for finding smaller natural groupings and very computationally expensive.
Divisive – K-means clustering	User can specify the number of clusters to form at a given level of the hierarchy	Presumes some pre-knowledge of the higher-level structure and requires some other type of clustering to build other levels (the levels other than that specified) of the hierarchy.

Table 4.2. Summary of display methods for hierarchical clustering.

Type	Advantages	Disadvantages
heat map/dendrogram	Provides a direct indication of gene activity and can include gene names or identifiers.	Does not indicate the degree of similarity between genes/clusters and must take up a lot of display space in order for the activity of individual genes (or their names) to be distinguishable.
horizontal distance trees	Indicates the degree of similarity between genes/clusters. Includes gene names or identifiers.	Does not provide a direct indication of gene activity.
radial distance trees	Indicates the degree of similarity between genes/clusters.	Does not provide a direct indication of gene activity and cannot include gene names or identifiers.

Table 4.3. Summary of clustering methods that allow genes to belong to multiple clusters.

Type	Advantages	Disadvantages
fuzzy k-means gene shaving	Provides a direct indication of gene activity and can include gene names or identifiers.	A larger set of results make it difficult to perceive less dominant patterns unless they are specifically defined by the clusters.
plaid models	Can include gene names or identifiers. Does not require the activity of genes to be displayed more than once	Patterns that are characterised over a limited interval of the experiments time frame will be dispersed along the gene-axis of the display and if a pattern is contributed to by a small proportion of the monitored genes then it is very unlikely the pattern will detected.

Table 4.4. Summary of methods that allow genes to be clustered across two or three dimensions.

Type	Advantages	Disadvantages
PCA, SVD	Display can be used to infer aspects of a gene's recorded activity by relating its position to the measure of variation communicated in the axes	Variations in activity that are not represented in the axes will not be communicated in the display
MDS	Capable of communicating more subtle variations in the data than PCA and SVD. More effective at preserving the structure of clusters than SOM.	No axes from which to relate the position of a gene's representation to any aspect of its recorded activity.
SOM	Capable of communicating more subtle variations in the data than PCA and SVD. More effective at grouping similar items than MDS.	No axes from which to relate the position of a gene's representation to any aspect of its recorded activity.
2D clustering (General)	Less occlusion than a 2D mapping of 3D clustering.	
3D clustering (General)	Extra axis can be used to communicate more of the variations in the data.	Higher likelihood of gene representations being occluded. Actual 3D displays are not generally available and 2D representations of virtual 3D displays may be ambiguous.

4.3 Interactive Clustering

In essence, clustering provides biologists with an overview of their data. This is a general summary of the data from a single experiment that reveals dominant trends and general structure within the data. Whenever appropriate, the results of clustering can be displayed to communicate these qualities. If, however, a biologist requires to investigate more subtle aspects of their data that are not revealed by such a general overview, a static representation is no longer sufficient and the biologist will need to interact with representations their data.

A large number of interactive techniques have been specifically developed to facilitate the exploration of microarray data. The majority of these are either adapted from or have been specifically designed to complement existing clustering techniques. In the former case there are two main subgroups of techniques; those that allow biologists to interact with a single clustering view and those that link multiple clustering views. Alternative techniques designed to complement existing clustering techniques tend to use a view of the data that is less effective as an overview but can

be used to query the data or display a subset of genes with similar activity. In general a range of different techniques are implemented in any one microarray analysis application.

4.3.1 Basic Interaction

In general most applications designed to assist in the analysis of microarray data support at least one mechanism by which a biologist can interact with a single static clustering view of the data. The most basic mechanisms allow biologists to highlight specific genes in the display, highlight groups of genes or select genes to have them labelled additional information. The advantage of allowing biologists to highlight genes or groups of genes in a clustering display is that before an experiment begins a biologist may have a particular interest in such a gene or gene grouping. If they are then allowed to highlight the gene or grouping in a cluster display it can help enable them to verify or reject any preconceived notions of how the activity of those genes changed and, by deduction, how the genes themselves are likely to have behaved.

Finding a gene typically involves a unique identifier being typed into a dialogue box or selected from list to highlight a genes representation in the relevant clustering display. Normally this unique identifier is the gene's symbol (a short string of alphanumeric characters which act as a unique abbreviation of the genes name) but usually some other identifier, such as references to the actual microarray probes used to record activity levels, can also be used. The highlighting of a selected gene can either be through some form of colour coding (i.e. the selected gene will have a different colour from the other genes), the outlining of the gene's representation or labelling of the genes identifier at an appropriate position (i.e. close to the genes representation) on the display. These methods are more or less effective for different types of clustering display. For example, while colour coded highlighting as appropriate for single point representations such as SOM or PCA, it would detract from or obscure the original colour coding in a hierarchical clustering colour mosaic display (see Figure 4.8) where outline highlighting is more appropriate.

Highlighting a group of genes requires a similar action to finding a gene. The difference between the two operations is that in order to highlight a gene grouping a grouping identifier is required rather than a gene identifier and that gene groupings

must be loaded from some external data-store. Once a grouping of genes is selected the individual genes which constitute the grouping are highlighted in the clustering display. This highlighting is generally in the form of colour coding or outlining of gene representations with the labelling of selected genes inappropriate due to the large number of genes that may be selected and the subsequent likelihood that gene labels could not be accommodated in a limited display space without overlapping and obscuring one another or other features of the display.

Interactive labelling of individual gene representations is also supported in a number of microarray data analysis tools. This is generally applied to clustering displays of microarray data and useful because of the typically massive number of genes surveyed in any single experiment and the amount of screen space that would be required if all these genes were to be labelled at the same time. In most applications a gene can be labelled by moving the mouse pointer over its representation. After the mouse cursor is moved the gene label disappears. If the mouse button is clicked while the label is still in place the gene selection is stored and/or supplementary information is displayed. Alternatively, gene labelling can require the mouse button to be clicked when the pointer is over a gene representation. Here labels are removed when the user clicks on another gene or an unoccupied area of the display.

4.3.2 Drill-down Techniques

Another mechanism by which biologists can interact with a single clustering view of their data is by using a drill-down type technique where they are able to focus in on seemingly more interesting clusters made apparent from the initial cluster overview [*12, 111, 148, 149*]. Here clusters are selected by either dragging the mouse pointer over an interesting region of a colour mosaic display [*12*, *149*] or clicking on the root node of a hierarchical clustering dendrogram [*148-150*]. Whenever a selection is made the selected cluster is displayed in an adjacent frame of the application window [*111, 149*] or allocated more screen space with the remainder of the data discarded from the display [*12*]. As the display of a selection's results are of an equivalent format to that of the original display, further selections can be made using the results of a selection that has already been made. This allows biologists to follow one selection with another, viewing an increasingly smaller number of genes, until only a few of the most interesting genes according to their selections are left on the display.

This allows the biologists to navigate the structure imposed on the data by the original clustering algorithm and affords them the opportunity to reveal some of the detail within the data that would otherwise be condensed into an all-encompassing cluster overview.

An alternative approach to drilling down in to a cluster view is to display selected clusters using overlaid time versus activity graph representations [*114, 115, 151-155*] such as those to the right of the heat-map in Figure 4.8. The advantage of this approach is that graph views are a more explicit representation of gene activity and it is easier to compare activity levels when activity is represented as displacement along an axis rather than encoded using colour. This is because position is perceptibly dominant [*57*] and can be used to represent more values [*30*]. While graph views do not work well for larger numbers of dissimilar time-series where the activity of individual genes become illegible due to crossing lines, they are effective at revealing extreme values and the extent of outlying values at individual time-points. As such they can be used for communicating general trends within groupings of genes with predominantly similar activity patterns such as those that are likely to be combined in the same cluster. So, the combination of a cluster view as an overview and a graph view, as a detail view to display a selection of genes where activity is coherent, seems quite logical. In general, graph views and clustering views are positioned in adjacent frames of an application and linked so that selections in the clustering view are represented in the graph view. This combination of views allows the graph to be used whenever an appropriate selection is made and ignored whenever the genes selected have incoherent activity and the display is inappropriate.

4.4 Non-clustering Interactive Techniques

A smaller group of techniques for the exploratory analysis of microarray time-course data forgo clustering altogether in favour of less abstract representations of the data. Here, rather than having features of the data revealed in an overview, the onus is largely on the biologist analysing the data to interact with representations for features to be uncovered. The two most common types of techniques for the analysis of microarray time-course data that do not use clustering are those that use overlaid graph views of the data and those that combine scatter-plots in a matrix. Techniques

that combine scatter-plots in other ways are generally not used with microarray time-course data but are used with other types of large-scale multidimensional data and are worth considering due to the potential that they might be adapted for use with time-course.

4.4.1 Graph View Techniques

In the context of information visualisation, overlaid graphs[6] (Figure 4.15) can be considered as a specialised type of parallel coordinate plot [*156, 157*] (Figure 4.16). Parallel coordinate plots and overlaid graphs both accommodate a multitude of attribute axes which are orientated to be parallel rather than orthogonal. In either case, vertical or horizontal line axes of a uniform length are placed at regular intervals along an orthogonal plane of the display. Lines are also used to represent items. Here, the value for any given item on any given axes is linked to the values for that same item on any adjacent axes. This means that lines will link the values of an item so that that items attribute values are represented by a continuous chain of lines linked at their endpoints. While parallel coordinate plots accommodate a range of different types of multidimensional data, overlaid graphs are specialised in that the data must have continuous scalar attributes which are ordered and relate to values with the same units. The layout of graph views differs from that of parallel coordinate plots in that the maximum and minimum of each attribute are *not* normally scaled to the upper and lower points of their respective axes. This is to ensure that the user can compare values across axes which are related to values with the same units.

[6] Using the more general definition of a graph as a diagram used to indicate relationships between two or more variable quantities with the quantities measured along two axes at right angles. Not to be confused with the information visualisation definition of a graph as a node-link diagram.

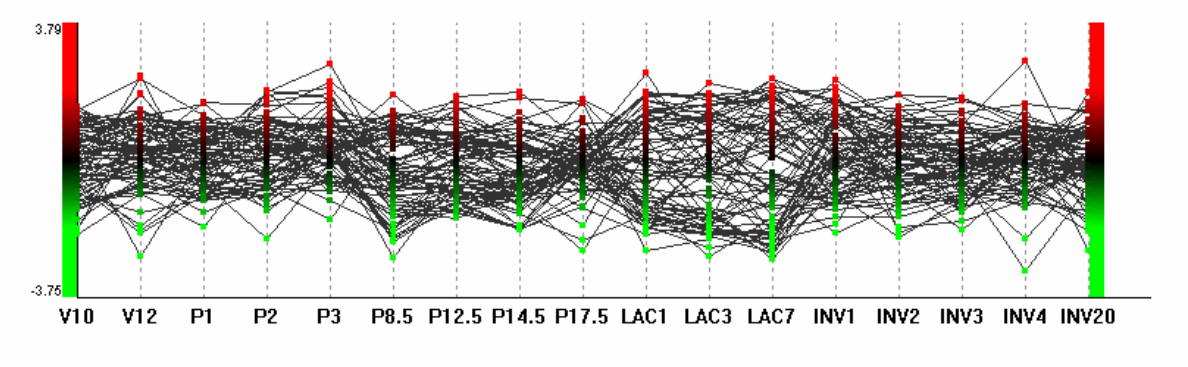

Figure 4.15. ***A graph of gene activity over time for a single gene (time-points are labelled according to the biologists own notation, image created using [151]).***

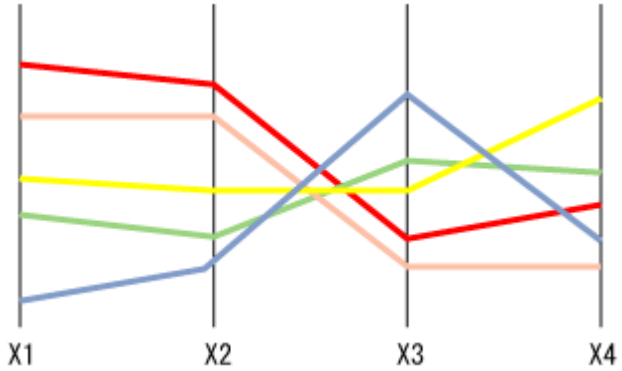

Figure 4.16. ***Parallel-coordinate plot: A simple example where five items (colour coded) have different values for four attributes (x1, x2, x3 and x4).***

The problem with graph views, and parallel-coordinate plots in general, is that when there are larger numbers of dissimilar items, lines associated with different items cross each other and it becomes difficult to distinguish between items. Moreover, when the number of dissimilar item is larger than about 20 it becomes difficult to interpret anything other than an indication of the range of values for each attribute. If, however, the lines which represent items are colour coded (as they are in Figure 4.16), with the ten hues that are distinguishable [*30*] it becomes possible to distinguish the proportions of a parallel plot relating to up to ten dissimilar items or predefined item groupings. Although, in the latter of these case, the parallel coordinate plot would be effectively displaying the general range of attribute values for predefined item groupings rather than the values of item attributes themselves. Hierarchical parallel coordinate plots use colour coding in a similar manner but use

the clusters produced by hierarchical clustering, rather than predefined groupings, in order to distinguish item groupings based on similarity.

The limitation of parallel plot displays is that they can only provide an overview of multidimensional data that effectively conveys the attribute values of up to ten dissimilar items. If a user requires to derive knowledge from information that is encoded by the attributes of more items they will require some means through which they can manipulate the representation to reveal more information. Typically this involves some interaction with the display that reduces the number of items or highlights a subgroup of items. Dimensional brushing [*67*] (Figure 4.17) is an example of such a filter mechanism. Here, the user clicks the mouse button while dragging the mouse cursor over a sub-portion of a dimensional axis to specify an acceptable range of values for the attribute corresponding to that axis. When the user releases the mouse button, items whose value for the relevant attribute do not fall within the specified range will be excluded from further selections and removed from the display or greyed out. Composite brushing [*67*] allows a multiple of selections of this type to be combined by one selection following another and the user specifying some logical operation rule (AND or OR) to determining how the selection is to filter the data. Brush manipulation allows a selection, once made, to be refined. After each selection the axis on which the selection was made becomes a virtual multi-range slider which can be adjusted to alter the condition associated with that axis. If an axes has multiple conditions specified for its attribute, which would necessarily be OR type conditions if any items are to remain selected, each of these would have its own set of adjustable multi-range slider thumbs. Composite brushing and axes range sliders for brush manipulation are shown in Figure 4.18.

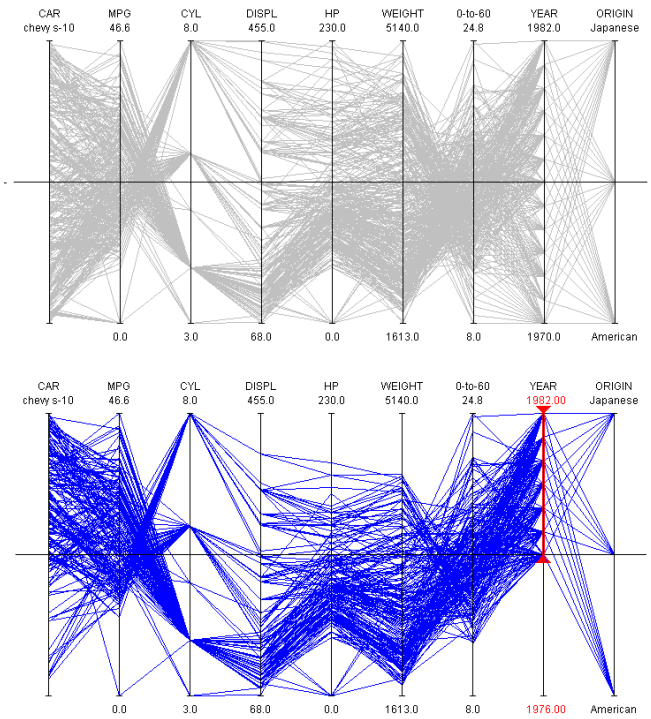

Figure 4.17. ***Parallel-coordinate plots before and after dimensional brushing (images generated using [158]).***

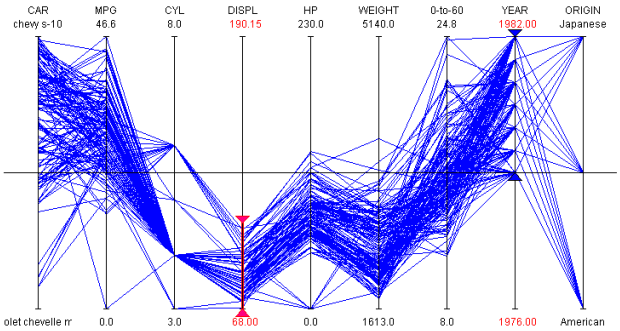

Figure 4.18. ***Composite brushing with axes range sliders for brush manipulation (image generated using [158]).***

Angular brushing [*159*] (Figure 4.19 middle) is a technique similar to dimensional brushing that allows users to select items according to an acceptable range for the ratio of attributes values between adjacent axes. Like dimensional brushing, angular brush selections can be combined using different rules and (Composite brushing) and adjusted after their initial formation (Brush manipulation). As the ordering of axes in graph type parallel-plots is related to the data itself and all axes represent scalar attributes, ratios of adjacent axes attribute values tend to be more meaningful which means that angular brushing is often appropriate for this type of display. Other techniques which are more specific to graphs are timeboxes, which allow the user to

specify an range of attribute values over a number of adjacent axes [*10, 11*] (Figure 4.19 top), and hierarchical-cluster explorer which allows the user to specify a profile that values must adhere to [*12*] (Figure 4.19 bottom).

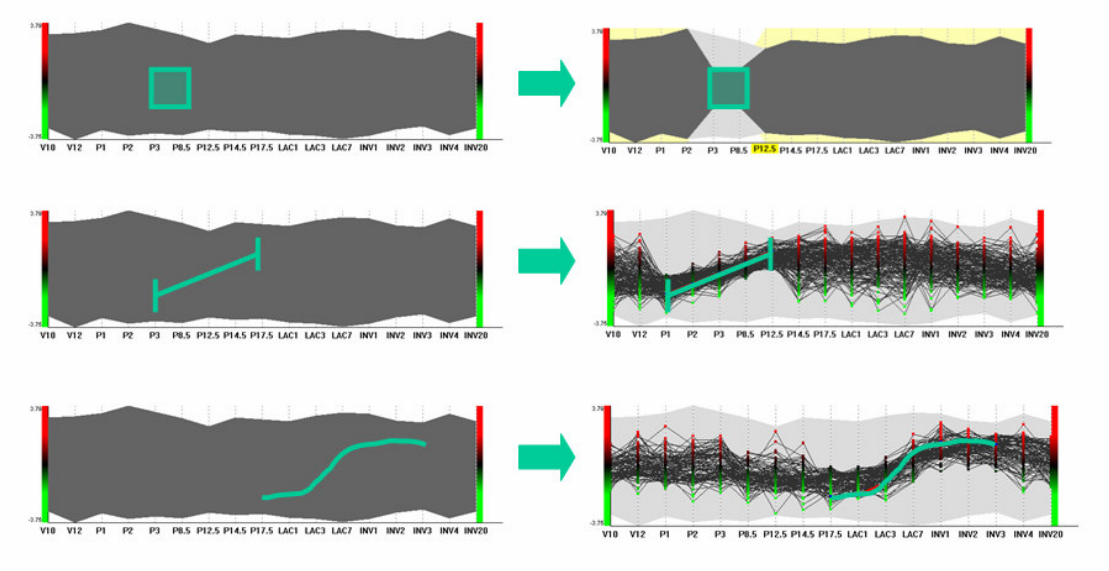

Figure 4.19. ***Selections based on graph view mappings of microarray time-course data: (LHS query and RHS query results): an acceptable range of values over a given interval (top), an acceptable change in values between time points (middle) and a profile that the expression of genes must adhere to (bottom). Images created using [151].***

4.4.2 Scatter-plot Matrices

Scatter-plot matrices [*31*] (Figure 4.20) display multiple attributes of multidimensional data using a recursive spatial-substrate. Here a standard matrix contains *n* by *n* individual scatter-plots where n is the number of attributes of the data table. Each row of scatter-plots has a different attribute for their y-axes and each column of scatter-plots a different attribute for their x-axes with the ordering of attribute axes the same for rows and columns and both rows and columns having the full set of attributes. In the case of microarray time-course data the attributes are gene activity levels at different time points and attributes are generally ordered according to their order in time.

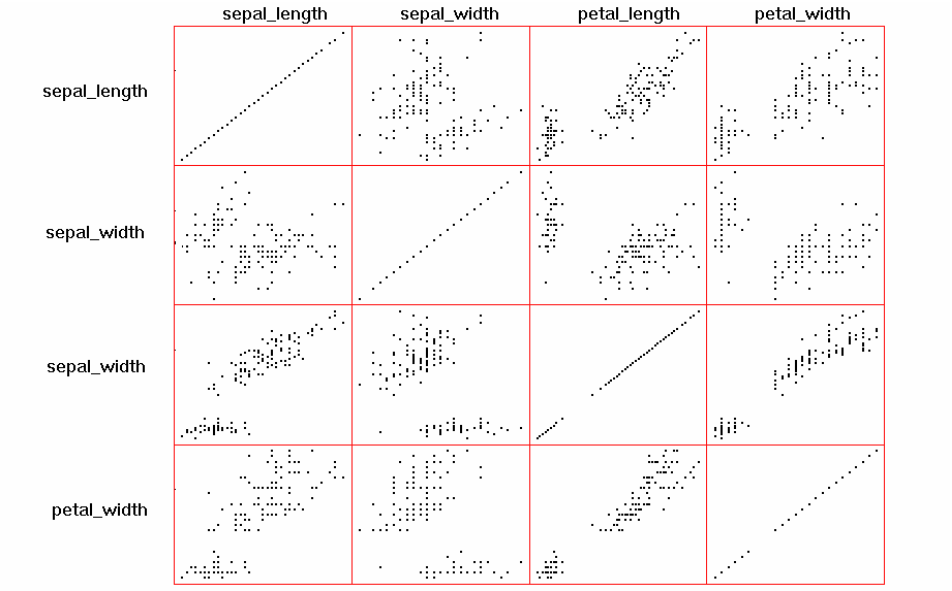

Figure 4.20. ***Scatter-plot matrix of Fisher's iris data [160] (taken from [161]).***

The triangular scatter-plot matrix (Figure 4.21) is a variation of the standard scatter-plot matrix which saves screen space by including only the plots that contain unique pairs of attributes. This is done by missing out the top-left to bottom-right diagonal plots and any other plots above and to the right (a lower triangular matrix) or below and to the left (an upper triangular matrix).

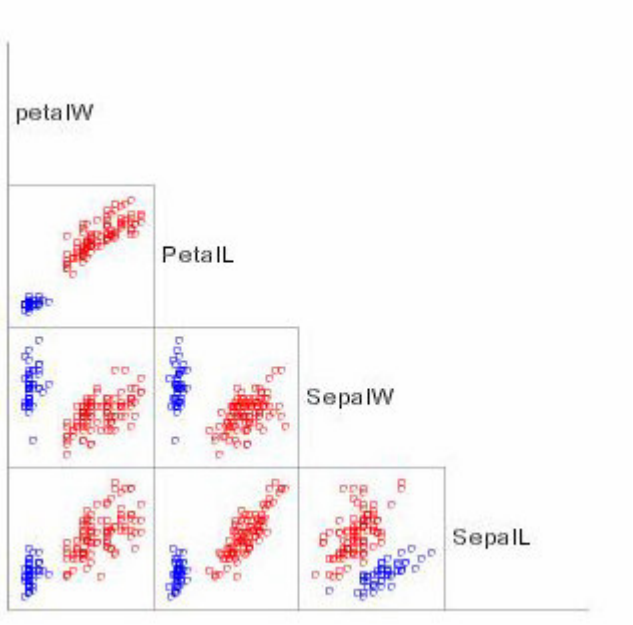

Figure 4.21. ***Triangular scatter-plot matrix (a lower triangular matrix) of Fisher's iris data [160] (taken from [161]).***

Brushing and linking [*70*] is a technique commonly used with scatter-plot matrices to allow users to determine how interesting features exhibited by the an attribute pairing in a single plot are reflected in the plots of other attribute pairs. Brushing means selecting a subset of the data items. Linking is when a selection of items made in one view causes those items to be highlighted in one or more other views. An example of such brushing and linking is shown in Figure 4.22. Here, the user has interacted with the plot outlined in black by brushing a group of items near the centre of the plot (the brushed area is outlined with a red box). The selected items are highlighted, using red crosses, in all plots of the matrix.

A major limitation of scatter-plot matrixes is that the overall relationship of the attribute values for the entire dataset is not immediately apparent [*24*] and it is difficult to see patterns that are present only when less dimensions are taken into account [*162*]. In this respect scatter-plot matrices are probably best described as an overview aimed at communicating relationships between pairs of attributes rather than finding more sophisticated patterns in the data. While this might be appropriate for multiple condition microarray data without a time-series it is less appropriate for the types of pattern one might want to find in time-course data where the associations between different multiples of attribute values are likely to be of interest.

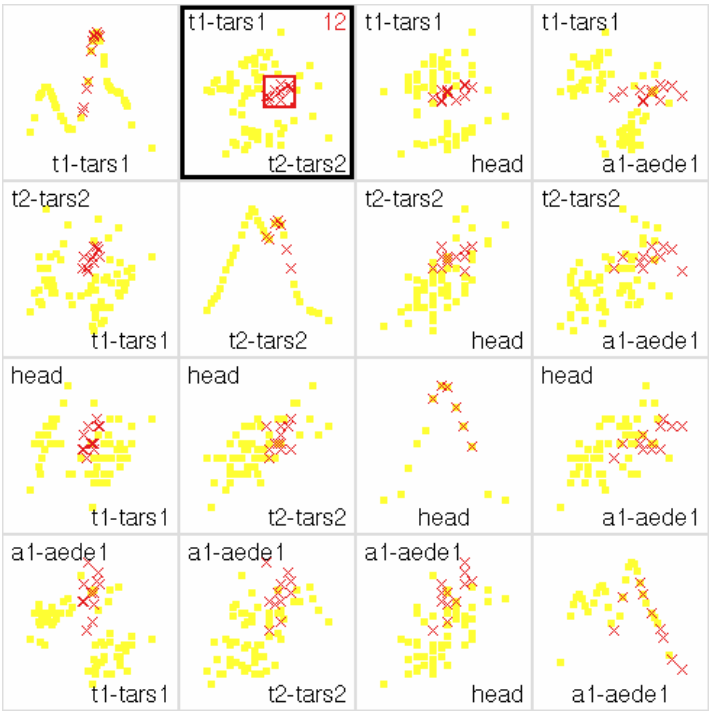

Figure 4.22. ***Scatter-plot matrix with brushing and linking (taken from [161])***

4.4.3 Scatter-plot Variations

If a data-table contains three attributes that require to be mapped to position for an overview of the data, an interface designer may consider supporting such an overview using a three-dimensional (3d) scatter-plot (Figure 4.23). While the advantage of a three dimensional scatter-plot is that more attributes can be mapped to spatial dimensions, there are also certain limitations of 3d scatter-plots. Firstly, special equipment is required to display data in true 3d. While most computers can, however, display a 2d projection of 3d scatter-plot, in such a projection it is probable that certain items will be occluded (obscured) by other items and that there will a degree of ambiguity in the degree to which axes determine the position of each data item representation. This means that users will often require to be allowed to rotate the representation about its axes in order to resolve ambiguities, avoid the occlusion of items of interest and utilise their ability to determine structure from motion [*163*]. In general microarray time-course data has more than three time-points and cannot be displayed using a three dimensional scatter-plot unless some form of dimensional reduction such as principle component analysis (section 4.2.2), singular value decomposition (section 4.2.4) or multidimensional scaling (section 4.2.8) are applied.

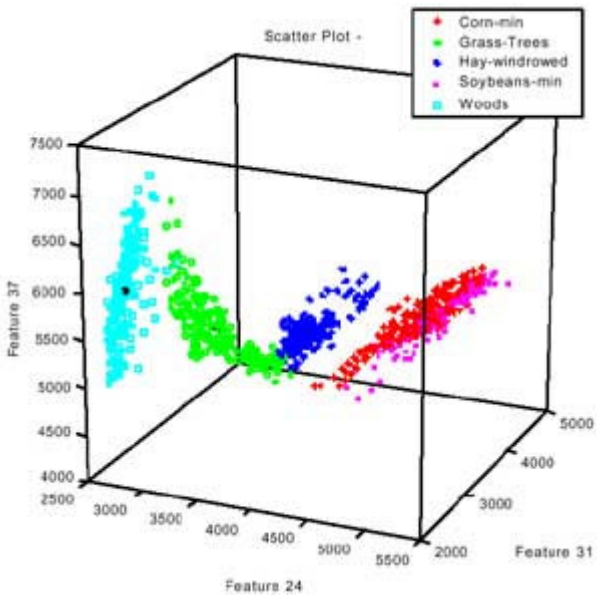

Figure 4.23. ***Three-dimensional scatter-plot***

Grand tours and projection pursuit display high-dimensional data by animating through different 2d or 3d linear projections. In the classic grand tour a multidimensional space filling curve is defined and a plane is moved perpendicular

to this curve with the data projected. Projection pursuits are grand tours adjusted so that the curve through multidimensional space is guided by some particular goal, such as differentiating between two classes of data, and projections bring out the data deviating from normal distribution as much as possible. In general this type of technique works best when the user wishes to discriminate between two or more classes of the data and insight relating to their objectives can be gained from some irregular linear projection of the data. The problem with grand tours and projection pursuit techniques is that when a user finds a useful projection it may be difficult for them to extract useful information from the axes configuration of that projection.

4.5 Combined Techniques

A number of software applications for the exploratory analysis of microarray time-course data allow biologists to combine different clustering overviews of their data or combine clustering views with complimentary non-clustering views (a characteristic selection of these are described in Appendix 1). The combination of clustering views reflects the fact that different types of clustering displays are more effective at allowing biologists to view different types of dominant trends in their data and for any given biologist analysing microarray data from a single experiment it is likely that more than one clustering technique will be appropriate [*4, 108*]. The obvious advantage combining clustering techniques in a single application is that the biologist is not required to switch between applications or load their data into a multiple of applications. It is also the case that the most appropriate clustering method for any given biologist at any given stage of their analysis may not always be immediately apparent and the ability to switch between different clustering views allows a biologist to try out different views of their data before they find out which leads them to the most valuable results. Occasionally clustering views are not just combined in a single application but linked together so that a selection in one of the views highlights genes in another [*115, 117, 164*]. This increases the capacity of biologists to compare cluster views of their data by reducing the time taken to relate representations of gene activity in one view to those in another. In these applications whenever a biologist finds a pattern in their data which appears to have significant value, the constituents of that pattern can be selected and immediately highlighted in an alternative cluster view in order for the biologist to confirm or challenge their initial finding.

4.6 Conclusion

There are many techniques designed to support the exploratory analysis of microarray data in general but few of these are specialised toward the analysis of microarray time-course data. This is despite the fact that microarray time-course data is fundamentally different from other (multi-sample) microarray data. Indeed, the clustering algorithms which act to generate an overview of microarray data and support the different types of visualisation (either to complement other visualisation techniques or generate a cluster view) would provide the same results if the time-points were reordered. As the ordering of biological events is of fundamental importance in the analysis of the data and this ordering is necessarily disregarded when processing the data to gain an overview, this seems unreasonable. It is therefore conceived that the development of a new method for analysis of microarray data specialised toward the analysis of time-course data (i.e. one preserves the order of time-points) might be more suitable at better satisfying the objectives of biologists analysing that data.

Chapter 5 Developing an Information Visualisation Technique for Microarray Time-course Data: Requirements Analysis

Requirements analysis involves the investigation, scoping and definition of a new software system prior to the design and implementation of any prototypes or versions. In the case of the research project described in this thesis, this involved identifying target users and working together with those users to agree on a set of requirements for a new microarray time-course data information visualisation technique. Common problems associated with requirements analysis are finding the right people to be involved and ensuring that the requirements generated are appropriate [*165*].

A successful requirement analysis procedure should involve potential users with adequate experience and expertise [*166*]. The biologists involved in the initial stages of this project, from the Scottish Centre for Genomic Technology and Informatics (SCGTI), were familiar with microarray time-course experimentation and had experience of using many of the more established analysis techniques. Moreover, as the problems associated with the analysis of microarray data were recognised by these biologists as being a major bottleneck in exploiting microarray technologies to their full potential, they were keen to collaborate in a research project aimed at alleviating this bottleneck in order to improve the efficiency of their own research.

5.1 User Requirements

The first user requirements meeting at the SCGTI took place on the 11th September 2001 and involved; an experienced biologist who was primarily involved in overseeing microarray experiments from their conception to the analysis of the data, a biologist who was involved in the practical side of experimentation and a statistician who helped the biologists to quantify the results of their experiments. The purpose of this meeting was to establish whether or not the biologists would benefit from the development of a new information visualisation technique for the analysis of their data. This began with the biologists outlining the basic procedure for experimental design, experimentation and data analysis.

First, it was established that microarray experiment design normally begins with an initial hypothesis. Next, technical aspects of experiment design are considered so that the results of the experiment can be used to test the hypothesis. These aspects might include which organism(s) will be used in the experiment, where the cell samples will be taken from, under what conditions the organism will be subjected and at which times or stages of development gene activity will be recorded. The initial hypothesis may also considerably affect the analysis procedure if it is related to a particular group of similarly classified genes. Here, when analysing the data it is likely that a biologist will initially focus on that group of genes. If, however, the hypothesis does not relate to a group of genes or the initial hypothesis is not conclusively proven, the biologist is likely to want to consider large numbers of genes and explore the data to try and generate further hypothesis. Hence the exploratory stage of analysis is often known as hypothesis generation. After a suitable hypothesis is generated it can be tested using statistical analysis and/or further experiments.

The biologists did not believe that the analysis techniques available to them at that time allowed them to explore their data in a manner that allowed them to find the most relevant patterns and form the most relevant hypothisese. They complained that the existing techniques relied on them disregarding valuable information either by; filtered out genes so that they would be left with a manageable number or compressing the entire dataset in a single display. During the meeting one of the biologists exclaimed; "*we do these experiments that allow us to record the activity of thousands of genes then most of the data is thrown away during analysis*". A literature review performed after the meeting proved that this type of sentiment was not uncommon. Biologists were found to express concerns about "a flood of data creating a need and demand for *new* tools" [*3*] and "the bottleneck in biological investigation shifting from data generation, to data analysis" [*4*].

Subsequent informal meetings with various biologists at the SCGTI allowed for some familiarisation with their data analysis methods. The next formal meeting with the biologist took place on 29th April 2002. This was attended by the participants from the previous formal meeting as well as an experienced virologist (a biologist specialising in the study of viruses) who had a particular interest in a relatively small

data-set relating to the early activation of the herpes simplex virus [*167*]. The aim of this meeting was to try to determine the nature of patterns that the biologists wanted to find in their data and could not find using existing techniques. It was decided that the herpes simplex virus dataset could be used to demonstrate examples of such patterns. This data-set was already well characterised so the biologists already new about a number of significant patterns in the data-set. The majority of these patterns where of a consistent general form relating to numbers of genes associated by common activity over limited intervals of the data. For example, one pattern was of 'early rising' activity where at the time of the virus moving from its dormant to active state a small group of genes had a sharp rise in activity. After this time the activity of the genes was neither correlated nor did the activity have any particular significance to the biologists. Other examples included intermediate and late rising activity where genes where linked by common activity over other periods of time. The other biologists in attendance agreed that these types of pattern where important in other microarray time-course data-sets and they where often missed by existing analysis techniques. The specific type of pattern was subsequently termed as an interval pattern and loosely defined as patterns of gene activity correlated activity over an interval but not necessarily over any other time.

When asked to be more specific about how they defined an interval pattern the biologists where initially reluctant to answer. This was justified by a belief that in constraining what was defined as an interval pattern and designing a tool capable of only finding those types of patterns, they might miss out on patterns that later prove to be valuable to their research. After reassuring the biologists that our approach would aim not be too restricted by a tight definition of a single pattern type and avoid other common restrictive approaches such as filtering out genes at an early stage of analysis they were more forthcoming. A summary of the following question and answer session is summarised as follows:

Q: Define a significant pattern in microarray time-course.

A: The correlated activity of a number of genes.

Q: How many genes?

A: This depends on other information about genes, such as known gene function, and the nature of the activity that is correlated. In general, if the

activity of the genes is more tightly correlated or the correlation is over a longer period of time then the correlation itself is more significant and not so many genes need be involved.

Q: How is activity correlated?

A: It can be rising, falling, high or low for different genes at the same time or, indeed, one group of genes can have converse activity to that of another or they can have two similar patterns of activity with a time lag between them. These patterns can be simple involving a single common change/activity level or complex with activity changing over different times.

Q: Is their any levels at which rising, falling, high or low activity can be considered to be significant?

A: No. In general the difference in activity from a baseline or across time, as a ratio, is a good marker for the significance of the activity but other factors such as the timing of the activity, its duration and information about function of the genes (i.e. predefined functional groupings) has some bearing. Some biologist like to restrict analysis to genes with a fold change above two or under a half, either from the baseline or over time, but there is no strong biological precedent for this and by doing this they run the risk of losing information that might otherwise be valuable.

Q: Without being able to strictly quantify what a pattern is, how can a pattern be found?

A: In general we use methods that cluster the data then inspect the clusters using overlaid graph views to look at the shape of the activity patterns. We aren't sure which clustering algorithms work best or exactly what they're doing to the data so we use different algorithms and use the best groupings made. The concern is that genes we would potentially be interested in are not included in these groupings.

Q: How significant are factors such as the granularity of the data, margins of error and other factors that might affect the quality of the data.

A: Before analysis begins genes with a high margin of error, i.e. those that cannot be used to form any meaningful assertions, are removed from the data. During the exploratory analysis errors are not considered until a particularly interesting pattern is found. At this stage we attempt to quantify the pattern by performing statistical analysis which takes errors into account. This allows

us to consider whether or not the pattern is still significant. The granularity of the data is less problematic as the time points at which expression is recorded are chosen both to produce the most useful data and minimise uncertainty.

Q: Are different timings more or less significant.

A: Yes. A biologist often relates the data to processes known or suspected to occur and a number of the times at which activity is recorded are often planned to synchronize with biological events. The relation of activity to a biological event can be very significant.

This questioning provided valuable information about what the biologist needed to be able to do in order to explore their data. So that the answers could be formally evaluated and used more efficiently to design a new technique for microarray time-course data analysis, they were organised into a more structured list of user requirements:

Requirement 1. Find basic temporal patterns in the data: These patterns are of the following types:

a) **Dominant trends:** These patterns associate a large number of genes with common activity levels and changes in activity over the majority of time points.
b) **Less dominant patterns:** Patterns that associate smaller numbers of genes with common activity levels and changes in activity over time points.
c) **Interval patterns:** Patterns that associate genes with common activity levels and changes in activity over smaller intervals of the data.
d) **Complex patterns:** Patterns that associate genes with numerous different common activity levels and changes in activity over different intervals.
e) **Subtle patterns:** Patterns that associate genes with a common subtle gradual change in activity.

Requirement 2. Find relationships between patterns: A relationship between any of the above patterns associating genes. This might be, for example, that a change in activity for one pattern precedes that of another or one pattern is the converse of the other with rising activity concurrent with falling activity.

Requirement 3. Find relationships between patterns and predefined-groupings: If genes related by common activity belong to the same predefined grouping or related groupings then a greater significance might be drawn from their common activity.

Requirement 4. Find relationships between patterns and the experimental conditions: If genes have common activity and an aspect of that common activity can be somehow related to the timing of a known biological event or state then a greater significance can be drawn from the commonality. For example, if a group of genes have a high level of activity during a state or a group of genes have falling activity before, during or after an event, this might make their common activity more significant.

Requirement 5. Store, Restore and Export results: So that findings can be compared and shared with other biologists.

5.2 Relation of User Requirements to General Biological Questions

The next stage of requirements analysis was to evaluate the users' requirements by considering if they were representative of microarray time-course analysis at large. A meeting was held on the 27th of May 2003 to discuss this issue. This was attended by the three biologists involved in the first project meeting and began by them describing a set of high level biological questions of the general form documented by Brazma and Vilo [*5*]:

Question 1. What are the functional roles of different genes and in what cellular processes do they participate?

Question 2. How is the activity of genes regulated, how do genes and gene products interact, what are these interaction networks?

Question 3. How does gene expression differ in various cell types and states, how is gene expression changed by various diseases or compound treatments?

The biologists suggested a number of ways in which a tool that allows a biologist to perform the tasks listed in section 5.1 could be used to help them answer these questions. Firstly, detecting different types of trends and patterns that associate genes

(Requirement 1) can allow a biologist to form assumptions relating to the function of those genes (Question 1). This is done using a process of deduction known as 'guilt by association' [*9*] where common activity is thought of as indicating that a group of genes are more likely to participate in the same process and be functionally related. Here, it is normally a larger number of genes associated by common activity over a larger proportion of the data's time frame that indicate an increased likelihood of shared function. This type of pattern could be considered as a general trend (Requirement 1a). When such a pattern is found a biologist can consider the timing of the pattern and the known functions of the genes involved in order to suppose some hypothetical process to be associated with those genes. Other types of pattern that associate genes with functions might involve smaller numbers of genes and genes with shared activity over smaller intervals (Requirement 1b-e). Here the degree of confidence with which a biologist could assume that multiple genes share function would also tend to rely on other factors such as the functional groupings which the genes already belong to and the timing of their shared activity (hence Requirements 3 and 4). In the first case, genes that are already known to share a function are thought of as being more likely to share some other function, so their common activity is more likely to relate to shared function. The timing of shared activity can also influence the process of 'guilt by association'. Here, the timing of the activity can often be related to that of other processes in such a manner that the activity is more likely to be symptomatic of a real process rather than just being coincidental. The processes that activity is associated with can either be known to occur, because of how the experiment is set up or observations made during the experiment, or be supposed to occur because of the relative timing of activity patterns.

Once patterns are found and related to processes, finding relationships between the timing of those patterns (Requirement 2) might allow a biologist to find relationships between processes. For example, if a group of genes associated with a process are seen to be active before another group associated with another process, it might be reasonable to presume that the first process precedes the other. This, in turn, allows biologists to consider how genes and their products interact (Question 2) and, if the data relates to different states of a cell sample, how that activity changes in different states. Relating these patterns and processes to the overall experimental conditions

(e.g. diseases or compound treatments) would also allow the biologist to analyse how the processes of the cell sample are changed under those conditions (Question 3).

5.3 Evaluation of Existing Techniques

The biologist involved in requirements analysis were experienced in using a number of analysis techniques and adamant that these techniques did not properly satisfy their full requirements for exploratory analysis. There was however still a possibility that they either were not using the right techniques to analyse their data, they were not using the techniques in the most appropriate manner or that an existing technique might fulfil their requirements if it underwent some minor alteration. This made it necessary to perform an evaluation of existing techniques before considering whether or not it would be appropriate to develop an entirely new technique. The evaluation involved an assessment of the degree to which the users' requirements could be met through the application of established techniques for the exploratory analysis of microarray time-course data as well as techniques that could feasibly be adapted for such analysis. This differs from the general review of techniques presented in Chapter 4 as it relates directly to the requirements stated in section 5.2.

5.3.1 Finding dominant trends

The first of the biologists' requirements was to be able to find dominant trends in their data. They did not, however, feel that this requirement was not met by any existing techniques as dominant patterns are easy to find when the data is clustered. As previously mentioned, techniques which cluster the data are by far the most popular group of techniques for the exploratory analysis of microarray time-course data (section 4.2). The basic idea of clustering is that genes with similar activity, and groups of genes with similar activity, are placed closer together in the same display. Clustering associates genes according to their activity over the majority of the time-frame and dominant trends, which involve large numbers of genes, appear as the most predominant features of a clustering display.

5.3.2 Finding Less Dominant Patterns

The biologists' second requirement was to be able to find less dominant patterns involving smaller numbers of genes. This was not properly satisfied by any existing

techniques for two reasons. Firstly, in their initial presentation of the data the majority of techniques such as clustering assign an equal proportion of the display to each gene. As there are typically massive numbers of genes to be considered in the data this means that even if the representations of a smaller number of genes are placed closer together, their representation might not occupy enough screen space for it to be apparent to the viewer. The second reason that a less dominant pattern cannot necessarily be found is that the representations of genes involved might not even be closer together in the display. This would be likely to happen if the less dominant pattern where also an interval pattern where genes have common activity over a limited interval of the data (see section 5.3.3).

5.3.3 Finding Interval Patterns

According to the biologists involved in requirements analysis the main limitation of existing techniques for the exploratory analysis of microarray time-course data is their inability to allow for the detection of interval patterns. Interval patterns cannot be detected from clustering displays because genes are grouped according to the similarity of their activity over the entire time frame and common activity over an interval of the data will not be evident if the genes with common activity have different activity over the remainder of the time-frame. This would cause the genes to be more closely associated with other diverse clusters. In a display where each gene is assigned to a single cluster, genes associated with different clusters will be dispersed throughout the display making extremely difficult impossible to infer any association between them.

The inability of clustering to reveal interval patterns is perhaps best demonstrated using an example of a pattern that cannot be detected using Hierarchical Clustering [*111*]. This pattern is illustrated in Figure 5.1 where a rise then fall in activity found over a particular interval (P1 to P3) could suggest that a group of genes are related to a particular biological process and that that process is associated with the experimental conditions. In this case, when the data is clustered, different patterns of activity before or after the interval cause the related genes to be assigned to different groupings with the significance of their common activity over the relevant interval lost. This is illustrated in Figure 5.2. The arrows at the bottom of the figure show the columns that relate to genes with the characteristic rising and falling activity from P1

to P3. It can be seen that it would be impossible to detect this pattern using only this type of clustering display.

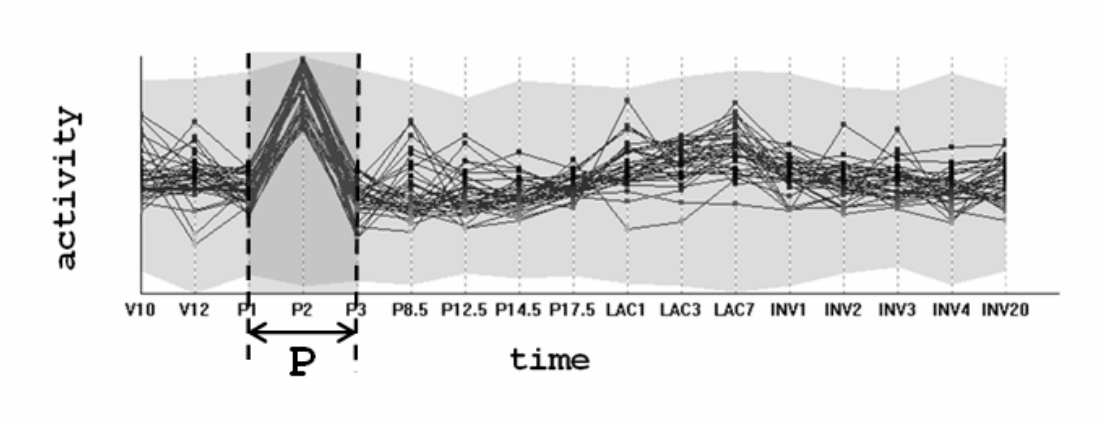

Figure 5.1. ***An interval pattern that would not be revealed by clustering the data (image created using [151]).***

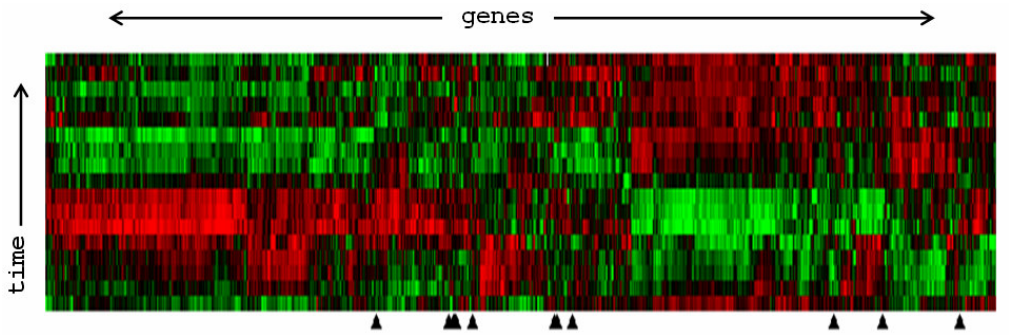

Figure 5.2. ***Heat-map clustering display of microarray time-course data with arrows at the bottom of the figure highlighting genes which contribute to a pattern occurring exclusively over an interval of the data (see Figure 5.1, image created using [151]).***

Other clustering techniques such as Fuzzy k-means clustering [*133*,*134*], Gene Shaving [*142*] and Plaid Modes [*143*] allow genes to be assigned to multiple clusters. The first two of these techniques produce groups of distinct clusters that can be represented as separate heat-maps and allow genes to be clustered according to different aspects of their activity with a limited scope for genes to be clustered according to interval patters. There is however, a significant disadvantage in that, as genes can belong to multiple clusters, there will be a larger set of results and it will be even more difficult to perceive less dominant significant patterns unless they are specifically defined by the clusters. Given the scale of the data and subsequent high number of potentially 'interesting' patterns, for any given less-dominant pattern this will be extremely unlikely. So, these types of clustering are perhaps only appropriate

for the detection of dominant patterns and interval patterns that involve larger numbers of genes.

The Plaid Models clustering technique does not require the activity of genes to be displayed more that once for genes to belong to multiple clusters. Instead, the data is processed so that the bands of colour that link genes in a hierarchically clustered display are more predominant. This means that patterns of similar activity are more easily detected and the effort required to perceive associations between genes that are separated in the display is significantly reduced. It is, however, still the case that a pattern that is characterised over a limited interval of the experiment's time frame will be dispersed along the gene-axis of the display and, if such a pattern is also contributed to by small proportion of the monitored genes then it is still unlikely that the pattern will be detected.

The only types of technique that allow the user to focus in on intervals to view interval patterns are those techniques that allow the user to select genes from a graph overview of the data. Here the biologist can set different types of query parameter to select genes with a particular activity profile over a specific interval of the data. The limitation of these techniques is that they do not provide an overview from which unknown or unsuspected patterns can be revealed. Instead the user is not only required to specify an interval but also the type of activity (e.g. rising, falling, high or low) before a pattern can be found. While the biologist's knowledge might relate to the timing of processes which can be related to the timing of interesting patterns, this knowledge is unlikely to relate to a specific timing and will rarely relate to the type of activity. This means that if a technique of this type were to be used in an attempt to find unsuspected interval patterns a biologist would, almost certainly, be required to search for patterns by performing a series of speculative queries specifying different types of activity over different intervals around an interval of some interest. Given the scale of the data and the variety of potentially interesting patterns in the data, this type of iterative search would be excessively time consuming.

In reality, a more appropriate use for graph views is for them to be combined with a clustering view. This type of combination provides both an overview and a means of querying the data to look at interval patterns. Neither of these techniques is, however,

capable of allowing a biologist to *find* interval patterns and their combination does little to counter the limitation of either technique in this respect. Likewise, when different clustering views are linked they can be used to compare dominant trends but there is still no scope to find the patterns that each clustering view would be incapable of finding if applied in isolation. So, the interactive techniques that supplement clustering do *not* allow biologists to find interval patterns in their data.

The inability of existing techniques to reveal temporal patterns was explained by the biologists as being due to the scale and complexity of the data with the massive number of potentially interesting patterns of this type making it incredibly difficult to find a pattern that is actually interesting without adequate opportunity for the user to guide their search and, at the same time, have adequate feedback from the interface. To gain an understanding of the scale of the problem, it is worth considering how many interval patterns are likely to exist in a typical set of microarray data. This is done by analysing a typical data-set with seventeen time points and 12,489 genes [*94*]. To begin with the count of interval patterns is limited to changes where activity halves or doubles between adjacent time-points and the concurrence of two or more changes of either type. There were found to be 3,353 such patterns, 1,143 associate five or more genes, 581 associate ten or more and 276, twenty or more. Table 5.1 shows the distribution of these patterns according to how many adjacent time-point pairings they involve and the number of genes associated. It can be seen from this table that there tends to be more patterns associating smaller numbers of genes and patterns involving medium numbers of adjacent time-point pairs (around seven, eight and nine). Nonetheless, there are a considerable number of patterns that associate a significant number of genes involving all different numbers of time point pairs. Moreover, this analysis only considers a limited subset of the patterns that are potentially interesting. From requirement 1, a biologist will need to be able to associate genes according to different degrees of changing activity (not just those with a halving or doubling in activity) and indeed different levels of high and low activity over intervals. This can only increase the number of patterns that need to be displayed.

Table 5.1. Number of patterns by number of adjacent time-point pairs and number genes associated

	Number of Patterns								
Pairs of adjacent time points	2-4 genes	5-9 genes	10-19 genes	20-29 genes	30-49 genes	50-99 genes	100-199 genes	200+ genes	Total
2	2	3	4	6	5	2	13	5	40
3	17	12	29	15	15	23	8	2	121
4	62	67	51	27	17	13	2	3	242
5	166	105	57	22	13	12	2	1	378
6	286	101	52	12	7	4	1		463
7	343	89	36	12	5	2			487
8	348	67	25	8	3	4			455
9	303	46	22	6	2				379
10	205	31	6	2					244
11	191	17	8						216
12	145	11	10		1				167
13	67	10	2						79
14	33	1	1						35
15	29	2	1						32
16	14		1						15
Total	2211	562	305	110	68	60	26	11	3353

If these amounts of information are to be properly accessible from an information visualisation with a static overview of the data it must include an overview that not only acts as a summary, but can also facilitate exploration of the data by allowing the user to interact to find the most relevant patterns in the data. This would be relatively straightforward if data were to have a well defined structure or the users' tasks were more strictly defined. Unfortunately, for the exploratory analysis of microarray data this is not the case. At any point in their analysis a biologist is likely to want to consider a large number of patterns and there is little in the way of formal structure that can be used to organise views of the data.

It would be easy to misconstrue the patterns in microarray time-course data to have a simple hierarchical structure from the fact that patterns that occur over smaller intervals are characterised by trends in activity that also partially characterise patterns that occur over larger intervals. If such a structure were to exist, a visualisation could allow biologists to select patterns at a higher level (i.e. those that occur over a larger number of time-points) in an overview of the data and successively drill down to patterns at a lower-level. This is not possible because not all patterns at a lower-level are necessarily associated with a pattern at a higher level. This is illustrated in Figure 5.3 which conceptualises the data as each gene having rising, falling or non-changing activity between any two adjacent time-points and

links genes with the same sequence over the same time points. It can be seen from this figure that genes associated in a pattern can be associated either by a variety of patterns at the same higher level, no patterns at higher level, a number of patterns at a lower level or no patterns at a lower-level. So, if interval patterns are to be explored in organised visualisations from either the top down (as they are with interactive clustering) or from the bottom up (as they are with visual queries) there is the distinct possibility of a number of significant patterns being missed. This is without even considering that patterns can be more subtle than merely presuming that a gene can have one of three states of activity between two time points (i.e. rising significantly, falling significantly or not changing significantly either way) or that a biologist may want to explore the activity of genes as events and relate genes with activity at different intervals.

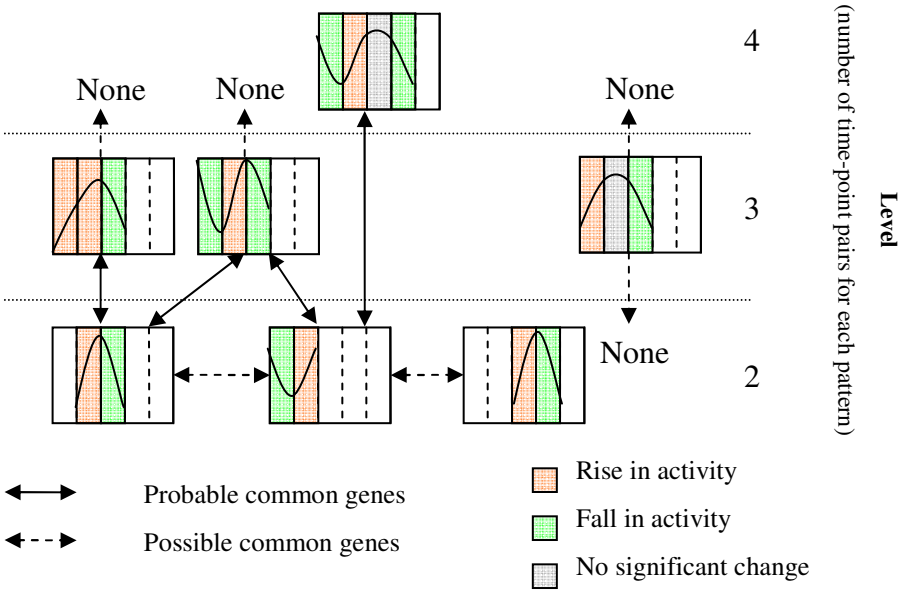

Figure 5.3. ***Example of types of relationships between patterns in microarray time-course data.***

GEDI is the one animated visualisation which maps time to time in the display and allows the user to perceive changes in the data over time. The problem with GEDI, and the reason that it cannot be used to find interval patterns, is that the mapping of data to display variables is inappropriate. While the user conceptualises genes as rising and falling over time and will be able to find the patterns they require to find if they can see this, in GEDI the positioning of genes is static. The changing colour of gene representations (or, rather, cells that represent groupings of genes) cannot draw the attention of the user to significant patterns as the overall impression of spatial

motion is perceptually dominant. Moreover, static gene representations with changing colour cannot be grouped as motion grouping does not work with common changes in hue (see section 2.3.3).

Radviz is another technique that might seem to have the potential to reveal interval patterns in the data. This is a dimensional reduction technique that is *not* commonly applied in the analysis of microarray time-course. A possible advantage of using a Radviz *type* approach with microarray time-course would be that, as with parallel coordinate plots, attributes that are ordered in the data can be ordered in the display. Here one might imagine attribute anchors to relate to time points and gene representations to gravitate toward certain anchors according to interval patterns. Unfortunately this would not occur because the display is likely to be ambiguous with larger numbers of attributes, and microarray time-course has sufficiently large number of attributes (time-points) to make the Radviz technique impractical.

5.3.4 Finding Complex and Subtle Patterns

Complex and subtle patterns can be found in microarray data using established techniques if the patterns are also dominant with large numbers of genes associated over the majority of the time frame and an appropriate similarity measure is used to cluster the data. The distance measure used in a clustering algorithm attempts to quantify the similarity of the activity of genes over the time course and common similarity measures such as the Euclidian distance measure can quantify the similarity of genes sharing complex and subtle activity equally as well as the similarity of genes sharing other types of activity. This however depends on shared activity being over the majority of the time frame. If similar activity is only over an interval of the time frame the genes will not be measured as being similar and the pattern cannot be found. Other techniques such as visual queries allow complex and subtle patterns to be investigated but not found as very little information is apparent from their initial overview of the data (see Section 5.3.3 for an explanation of why interval patterns cannot be found using visual queries).

5.3.5 Finding Relationships between Patterns, Predefined-groupings and Experimental Conditions

Relationships between patterns are easy to covey in an information visualisation if the associated activity is shown. This can be either in a clustering view such as a heat map where all activity levels are shown (and any patterns that are evident can be compared) or from a view where multiple patterns can be selected with pop-up detail views of their activity shown. Showing the activity of genes also allows patterns to be related to experimental conditions as the user can relate activity to time and time to the conditions. Relating patterns to predefined groupings is less straightforward. Here, either a limited number of groupings can be represented by colour coding gene representations or group membership can be queried once genes are selected. The former option has the advantage that groupings are immediately obvious and the known function of genes can be used to assess the significance of a pattern before genes are selected. It does, however, generally rely on there being no overlap between the groupings represented. This is due to the difficulty in associating one gene with two groupings and having that one gene with two colours. If predefined groupings are only apparent once genes are selected, the known function of genes cannot be used to inform the user prior to making a selection.

5.3.6 Store, Restore and Export Results

Most established techniques for the analysis of microarray time-course allow results to be stored, restored and exported. Normally the data will be exported to a text file or there will be some direct connection to a database. Some techniques also provide the user with the option to export a static frame of a visualisation as an image file. These can be used to illustrate findings when they are published on websites and/or in academic journals or conference proceedings.

5.4 Conclusion

Requirement analysis began with a group of biologists describing a set of requirements for a tool to support the exploratory analysis of microarray time-course data. The biologists then justified the requirements by describing their relation to biological objectives and citing relevant published materials. After this, an evaluation of existing techniques for the analysis of microarray time-course data was performed.

This showed them to be incapable of properly satisfying the users' requirements. The main problem is that existing techniques do not allow users to find less dominant patterns that associate smaller numbers of genes or interval patterns that associate genes according to common activity over an interval.

Clustering only allows the user to find more dominant patterns that associate larger numbers of genes over the majority of the time frame. Techniques that allow the user to perform queries based on a graph view of the data rely on the user already having knowledge of both the timing and the type of common activity that associates genes. While knowledge relating to the timing of patterns that are likely to be of interest to a biologist might exist, it is extremely unlikely that the biologist will have any idea of a particular type of common activity that is likely to be shared by genes and there is unlikely to be any particular type of activity that is of special significance. This means that actually finding a pattern, as opposed to investigating a pattern that is already found, might involve a time consuming iterative search over all the types of patterns that are potentially interesting until such a pattern is found to exist. This is impractical, especially given the massive scale of the data. When clustering and graph queries are combined in a single interface, neither technique can be used to find anything other than dominant patterns. Other techniques for multidimensional data are inappropriate for the analysis of microarray time-course data either because of the scale of the data (parallel-plots) or the fact that the number of data attributes makes the representation of item qualities too ambiguous (Radviz). The animated representation of microarray time-course data, GEDI, uses a layout based on clustering and suffers from the same limitation of clustering in that it can only be used to detect more dominant trends.

In order to support the exploratory analysis of microarray data, an information visualisation must be capable of providing representations from which the user is able to find certain types of pattern that are capable of being read to generate new knowledge. While at any given time the data from which any such representation needs to be formed can be reduced according to particular timing or predefined groupings of particular interest, this still constitutes a large amount of data and the number of potentially interesting patterns in the data might still be very large -

particularly if the user's interest is in a specific time in the data and they are likely to be interested in patterns of common activity over, before and after that time.

As animated representations are capable of utilising an additional display dimension and are capable of displaying more information, it is hypothesised that an animated representation may be better suited to supporting the users' requirements. Such animated representations of microarray time-course can, naturally, map time to time in the display and represent changes in a display over time. GEDI is the one existing animated representation of microarray time-course which does this but without allowing interval patterns to be revealed because of an inappropriate mapping of data to display variables. If, however, a new visualisation technique mapping time to time were developed with a more appropriate mapping to other display variables (and animation used to represent changes in such a manner that the motion of objects draws the users attention to the most relevant patterns), there is a distinct possibility that that type of animation would be able to satisfy the full set of user requirements as listed. This type of visualisation could use time to display more of the data and exploit the expressive qualities of motion so the users' attention would be drawn to the most relevant patterns in the data.

Chapter 6 Prototype 1: Animated Scatter-plots to convey Changes in Activity

From the user requirements and evaluation of existing visualisation techniques it was decided that an animated information visualisation technique would be capable of supporting the effective exploratory analysis of microarray time-course data. The tool developed to demonstrate and evaluate this new technique is called the Time-series Explorer as it allows biologists to *explore* their microarray *time-series* data. Development of the software proceeded in accordance with the prototype life-cycle model. This model was chosen partly because it has already been used successfully for the development of other exploratory information visualisation applications [*39*] and partly because it allows for functionality to be built in gradually so that different aspects of that functionality can be tested separately with the tool being progressively modified to fit user feedback. This was important because of the potential for fundamental aspects of the technique not matching user requirements and development proceeding unchecked toward an unsuitable final product.

The functionality of each prototype in the development of the Time-series Explorer technique is as follows:

> **Prototype 1:** The first prototype tests the ability of an animated representation to allow a biologist to relate motion to changes in gene activity over time for large numbers of genes. This prototype is evaluated according to its ability to allow a biologist to detect basic interval patterns such as a single period of rising, falling, high or low activity.
>
> **Prototype 2:** The second prototype builds on this functionality, focusing on the user's requirement to be able to find different types of patterns. These types of pattern including those that associate genes according to more subtle changes in their activity and more complex patterns including different successive changes and levels.
>
> **Prototype 3:** Three is the final version. This prototype is designed to support, and is evaluated with respect to, the full set of user requirements.

As well as building on the functionally of the previous prototype each later prototype also refines aspects of earlier implementations. This is naturally in accordance with

user feedback and analysis of how the users operate and interpret the displays of those earlier prototypes.

Figure 6.1 shows the design for the first prototype. This is in a general form which also relates to the design of the later prototypes. The idea of the design is to allow the user to select and adjust an interval of the data which is displayed in a linked scatter-plot view. When the interval is shifted over time the genes representations will move to reflect their changing position for different intervals. Here, the two attributes that define an interval are its start time and length. Each of these is measured by a number of time-points, those being time-points for which the activity of genes has been recorded, rather than an actual physical measure of time.

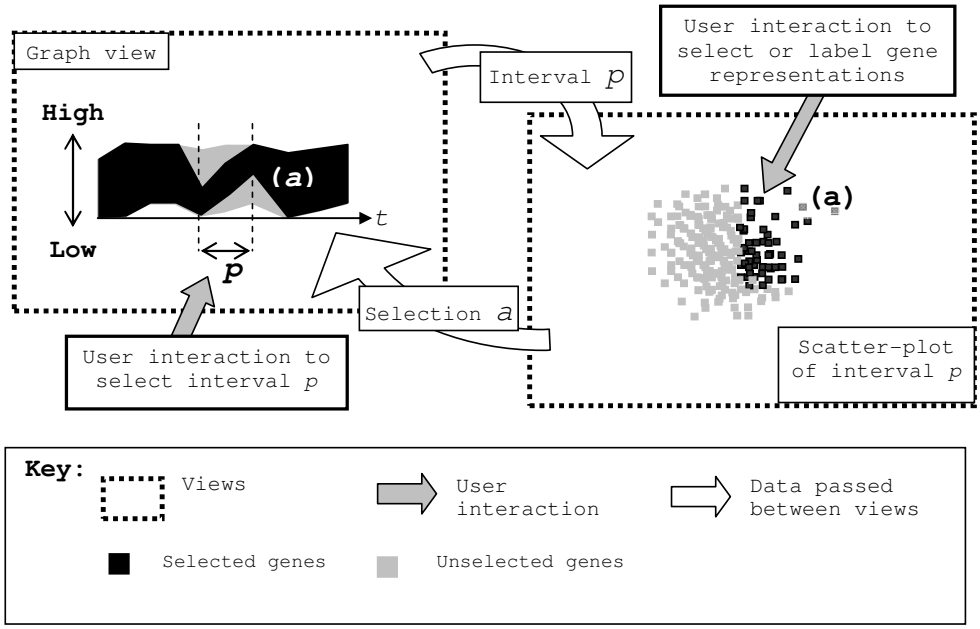

Figure 6.1. ***Basic design for Time-series Explorer***

The fact that displays are configured according to numbers of time-points rather than measures of actual time reflects the way a biologist is likely to set up a microarray experiment. This is generally done to record the activity of genes over the most relevant points of time and these points of time can be predicted to warrant approximately equal attention in the analysis of the data. While it would be possible to configure the display to have a regular time scale, this would mean that adjacent time-points with a relatively small time between them would have less display space and display time contributing to their representation. The biologists preferred this not to be the case and wanted instead to have all time points equally spaced (both in a

graph view and through time in an animation) using time point labels to interpret their actual physical timing. As the display scales the data to time-points for which data exists and the smallest useful interval on this scale has two time-points to include a single change in activity for each gene, it is useful to consider this to be a basic unit of time which can be referred to as a 'unit interval'.

6.1 Design Rationale

The design of the first Time-series Explorer prototype is based on two assumptions. The first of these is that time can be used as an extra display dimension [*13*] and an animated representation will be able to reveal more of the detail and, therefore, more of the less dominant patterns of the type that cannot be found using other techniques. The second assumption is that, as the data is temporal, it is appropriate to present such an animation across time so that the user will be able to intuitively relate the visualisation to the data with changes over time in the data represented as changes over time in the visualisation. Moreover, if the mapping of data to display variables is appropriate, the motion of the display might be able to communicate information more effectively by exploiting pre-attentive processing [*58*] and artefacts such as the common fate principle [*55*] where items moving in the same direction tend to appear as a distinct grouping. There are also a number of potential disadvantages of developing such an animated visualisation. These are:

1) The increased delay in *seeing* the data as it is animated [*168*].
2) The delay in patterns being committed to long-term memory and potential for patterns to be lost [*15*].
3) The inability to compare data that is presented at different points in time.
4) The animation having coarse motion due to a lack of data.
5) The time taken for a user to become familiar with a new alternative view of their data.

In the first prototype these problems are accounted for in a number of different ways. Firstly, to reduce the delay in seeing the data as it is animated, the user is given tight control over the direction and pace of the animation so that they can animate slowly over intervals where interesting patterns appear and quickly over the remainder of

the time-course. In this regard the first prototype can be thought of as a kind of genomic video cassette player where the user can play, fast forward, rewind, slow motion, pause and stop the animation of gene activity to examine, and re-examine, the more interesting intervals of the data. While the animation will still take some time, the time taken by the interface will not necessarily reduce the efficiency of the user. Indeed, it is likely that time taken to complete the task will be significantly reduced as the cognitive load of the task is switched to the human perceptual system [*17*].

Tight control over the direction and pace of the animation also makes the second listed potential disadvantage of an animated display, the delay in patterns being committed to long-term memory, somewhat less of a problem. If the animation is played at a fixed rate from start to finish then it would be possible for the user to first see a pattern and then, waiting for the animation to finish and viewing other patterns in the data, forget what they saw. With control over the pace and direction of the animation, as soon as a pattern is seen the animation can be stopped, rewound and the pattern can be viewed again or the user can select and store the relevant gene identifiers for further reference. Here, the animated view is only intended for the user to detect a pattern. For a pattern to be read, the genes with common activity can be selected and analysed using other coordinated views. Here the coordinated graph view and a list of selected gene names would come into play, allowing the user to read the pattern for problem solving and extracting knowledge.

The third potential disadvantage of an animated time-course scatter-plot is the inability to compare data at multiple time points simultaneously. While a comparison of all time-points simultaneously (for the activity of all genes) would be impossible for the biologist to digest, there are certain situations where it is of particular value to compare a number of time-points. Specifically, these are situations where recorded activity at a smaller number of adjacent time points defines a pattern which occurs over a limited interval of the experiment's time frame. To better facilitate the finding of such patterns the first prototype is designed so that each frame of the animation represents an interval of the time-course rather than an instantaneous time-point.

Next, a lack of data, specifically a lack of data between time-points, means that gene representations may shift drastically between time-points without the user being able to relate between successive frames of the animation. To account for this potential pitfall, the activity of genes is interpolated between the time points in the data and the interval selection is incremented by quantities independent of (and smaller than) the space between time-points for which expression is recorded. This allows for an animation where the motion of gene representations is smooth and they can be tracked by the user to reveal more sophisticated patterns. To prevent the user inferring undue significance from interpolated values, the start and end of the interval selection automatically moves to the nearest time-points, for which activity is recorded, whenever the user is not adjusting the interval selection. In effect, the interval selection scale is only continuous when the selected interval is in motion, so that genes can be tracked to discern more complex patterns of activity, and discrete when the selected interval is fixed for the user to interact with the static scatter-plot by making selections and forming queries.

The fifth and final potential weakness of an animated time-course scatter-plot is the time taken for a user to familiarise themselves with a new alternative view of the data. The fact that the user will require some amount of time to familiarise themselves with a new type of representation is inevitable, but the time taken can be reduced by accommodating the process of familiarisation in the design of the interface. Here, the graph view representation will already be familiar to the user and can be used to aid their interpretation of the unfamiliar scatter-plot. This is done by linking the views [*70*] so that the user can select gene representations in the scatter-plot to have them highlighted in both views. To enforce the idea that the scatter-plot is only representing an interval of the data at any given time and to show what that interval is, the graph view also has an overlaid transparent rectangle covering that interval.

6.2 Scatter-plot Layout

The layout of the scatter-plot in the first prototype can be seen as an adaptation of the Radviz layout for multidimensional data specifically designed for smaller intervals of time-course. While Radviz has dimension anchors that are each related to a single data attribute, the anchors in the layout of the first prototype Time-series Explorer

each relate to a characteristic pattern of activity. Figure 6.2 shows the layout for intervals with three time-points beside the layout for intervals with four time-points.

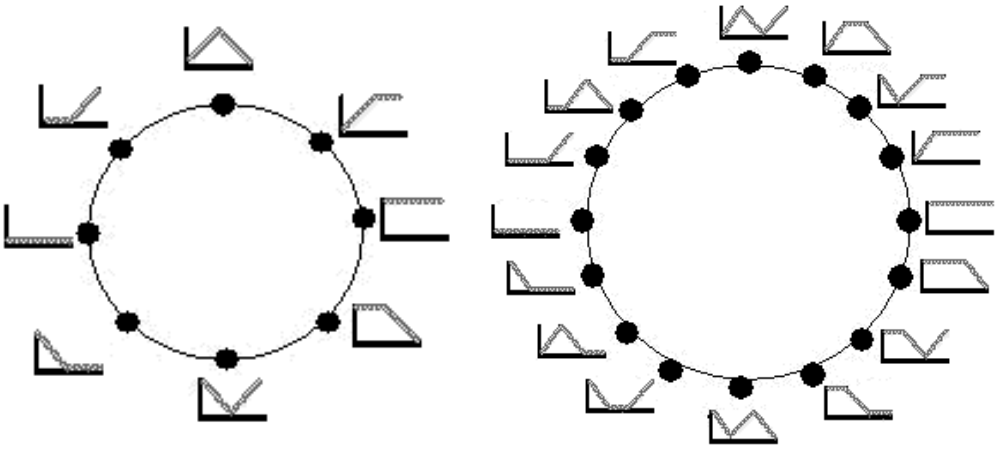

Figure 6.2. ***Layout of dimensional anchors in the first prototype Time-series Explorer: for an interval with three time-points (LHS) and an interval with four time-points.***

The idea behind this type of layout is based on the fact that if an entire set of microarray time-course data can be considered as a set of points within an n-dimensional expression space (see section 4.2) and rescaled expression is equivalent to gene-activity, then an interval of rescaled time-course can be considered as a set of points within an n-dimensional gene-activity space. Here, n is the number of time-points within the interval with each time-point relating to an axis. The displacement of gene-points along each axis in activity space is equivalent to their gene's activity at that time point. As the range of gene activity across time-points is equivalent, interval gene-activity space can be bounded by a hypothetical n-dimensional hypercube[7]. The characteristic activity patterns associated with anchors of the prototype layout relate to the corners of this multi-dimensional cube with single-point gene representations positioned according to the similarity of their activity to each characteristic pattern calculated using the Euclidean distance similarity metric. The anchors are spaced around the circumference of the circle in the layout according to two rules. Firstly, to ensure that genes with diverse activity are displaced accordingly, patterns related to opposite corners of the hypercube are at opposite points round the rim of the circle. Also, to allow for genes to be tracked

[7] Also known as a multidimensional cube, a hypercube is a regular polytope with mutually perpendicular sides. Hypercubes can have any number of dimensions. In 1D it's a line, in 2D a square, in 3D a regular cube and in 4D a tesseract [*153*].

over time, the layout is configured so that when the start of the time interval is shifted (with its size remaining fixed) the displacement of a gene representation will be minimal and in a constant rotational direction regardless of their change in activity.

As a different anchor layout was required for each different interval size an algorithm for configuring the anchor layout was devised. Here, each characteristic pattern was reduced to a binary number with n-digits (n being the number of time-points in the interval) with 0 signifying minimum activity and 1 maximum activity. The number of permutations of such a binary number, the number of n-dimensional hypercube corners and the number of characteristic patterns in the layout is 2^n. First anchor place-marks are mapped equidistant around the circumference of a circle. Next, the pattern of minimal activity at all time points (e.g. for interval size 3, 000) is associated with the place-mark at the far left-hand-side. Moving anticlockwise around the rim, each subsequent place-mark is associated with a pattern incremented by one (e.g. 001, 010, 011…) until the place-marks at the far right hand side is reached. This place-mark is associated with the pattern of maximum activity at all time-points (e.g. 111). Again moving anticlockwise around the rim, each successive place-mark is associated with a pattern one less than the previous (e.g. 110, 101, 100…) until all the place-marks are associated with patterns.

The advantage of using a Radviz type layout for an animation of time-course rather than adapting any other type of dimensionally reduced display, is that it has a structured spatial substrate of sorts that can be used to relate between different views as the display is animated. The vast majority of clustering techniques do not have such a spatial substrate and the position of items is based purely on their relationships with other items in the data. If such a display were to be used to display an interval of microarray time-course data and then a subsequent interval of the same size (as in the case of such a display being animated across time), genes with constant activity might move or genes with changing activity might remain in the same position because of the reorganisation of gene groupings. Moreover, these displays rely on the user referring to the activity of individual genes rather than their position to determine the general activity of a group of genes. With a rapid succession of displays of this type being presented in an animation this would not be

possible. The corner anchors of the first Time-series explorer prototype define its spatial substrate and can be used to relate gene representations to the activity of their genes over an interval. These remain static for intervals of a fixed size so that the user can easily interpret a gene's activity and rapidly relate between successive representations when the start time of such an interval is incremented.

The advantage of using the corners of a hypercube as anchors (rather than dimensions of the data like Radviz), is that hypercube corners lend themselves better to a configuration that encourages smooth animation of the gene representations. This is because hypercube corners relate to patterns of activity and can be positioned to encourage changing patterns of activity to appear smooth using a general prediction of how patterns are likely to change over time (i.e. rising activity is more likely to be followed by falling or high activity than more rising activity, low activity is less likely to be followed by falling activity etc). This cannot be done with dimensional anchors as the position of items is predominantly influenced by high attribute values and changes in activity over time involve high and low activity at different time-points. Another advantage of using hypercube corner anchors in the scatter-plot of the first prototype is that they can be related directly to patterns of activity over intervals and labelled accordingly. This helps the user relate the positions of gene representation directly to those patterns of activity.

6.3 Animation

As the layout of the anchors in the first prototype is different for intervals of a different size, the interval size must remain fixed during animation for the users to track gene representations across time. To ensure that such an animation is smooth the interval start time must advance by a fraction of a unit interval with between time-point activity values interpolated. This is done using simple linear interpolation with between time-point values approximated using values at the next highest and lowest real time-points. Linear rescaling is preferable to other forms of interpolation in this application because of its simplicity. This allows interpolated values to be calculated as they are required for the display without unduly degrading the performance of the animation. Figure 6.3 shows the values required for linear interpolation (Eq. 6.1).

Figure 6.4 begins to show how interpolation smoothes the animation in the first prototype by showing how interpolation gradually changes the shape of a gene's activity pattern over the selected interval. As genes are positioned in the scatter-plot according to this shape, it follows that a gene representation will move gradually as the selected interval changes and motion in the scatter-plot will be smooth. In reality, where the figure only describes three points where activity is interpolated between time-points with real data, in the actual prototype there are about ten. This ensures that motion is still smooth whenever the activity of genes changes dramatically and the resolution of the display is high.

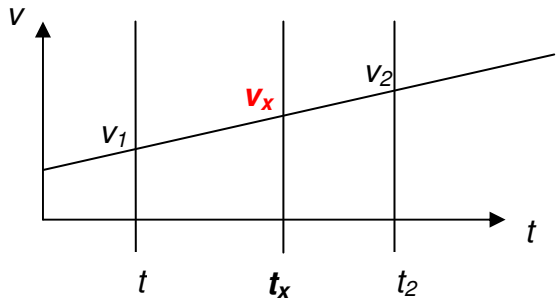

Figure 6.3. ***Values required for linear interpolation (for unknown v_x in red).***

$$v_x = v_1 + \frac{t_x - t_1}{t_1 - t_2}(v_2 - v_1)$$

Eq. 6.1 Formula for linear interpolation

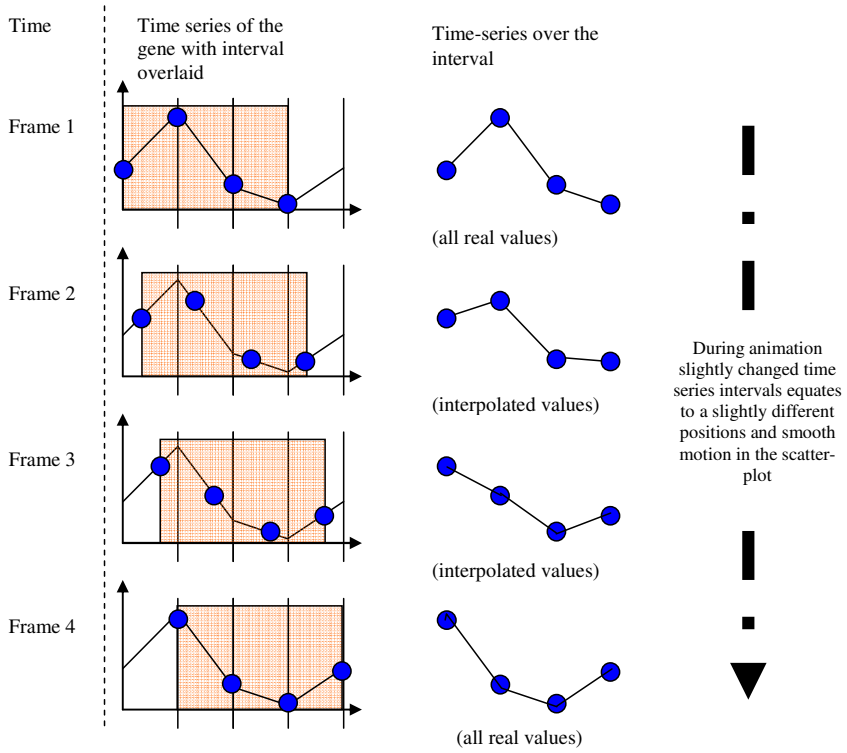

Figure 6.4. ***Interpolation smoothes animation by gradually changing the pattern of activity for a gene to be displayed in the scatter-plot.***

As there is no natural rate at which the animation could be thought of as running to communicate the optimal amount of information, it was decided to allow the user to configure the rate of animation themselves and to their own preferred rate. This can be done by selecting speed options from the menu-bar (slow, medium, fast or very fast) which adjust the amount of time each frame of the animation is displayed.

6.4 Interaction

Figure 6.5 shows a screenshot of the first prototype interface. The interface is comprised of five separate panels. These are the scatter-plot view, graph view, interval controller panel, selection mode panel and selected groupings panel. These panels are combined and work together toward supporting the requirements of the user.

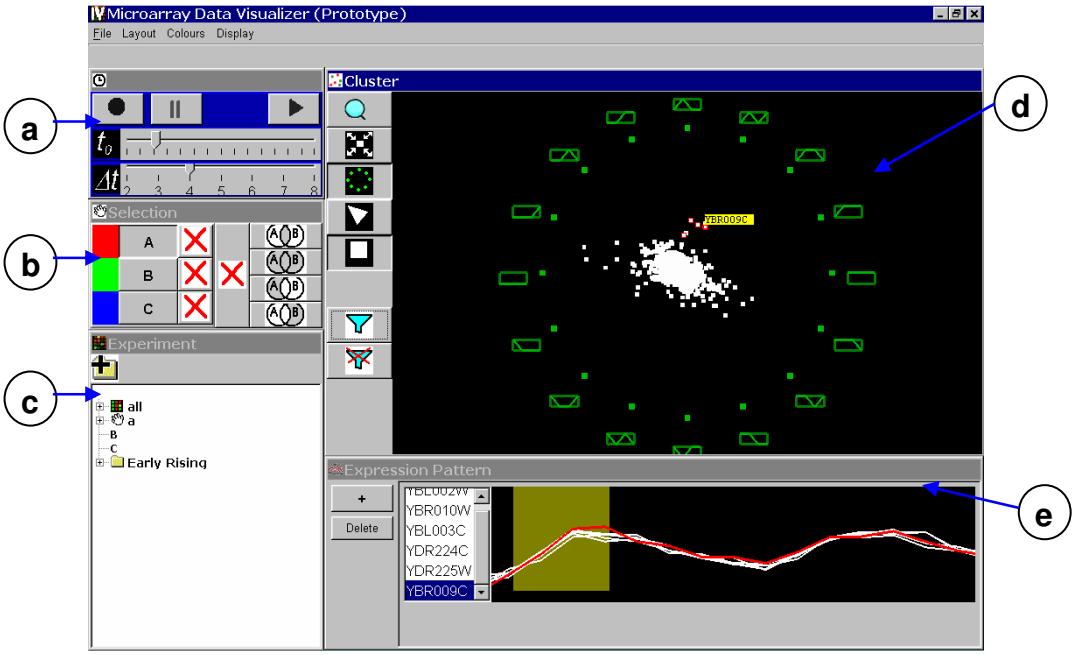

Figure 6.5. ***Screenshot of Prototype 1 interface: a) interval controller panel, b) selection mode panel, c) selected groupings panel, d) scatter-plot view and e) graph view.***

The scatter-plot view displays the activity of genes over an interval of the data. When this interval is shifted with a fixed size, the view animates with genes moving smoothly to their new positions. When the scatter-plot is not in motion, it can be used to select genes or groups of genes. This is done either by clicking on a single gene representation or lassoing a group of genes. When a single gene is selected with a single click, the representation of the selected gene is labelled with its gene symbol. When genes are lassoed, the selected genes' representations become differently coloured (the colour depending on the current selection mode) and remain that colour until another selection of this type is made.

The graph view of the first prototype has two main functions. Firstly, it gives an indication of the interval displayed in the scatter-plot. This is done by having the entire width of the graph view represent the entire time-frame of the experiment (i.e. the maximum interval of the data) with a green shaded rectangle representing the portion of that time frame that comprises the current selected interval. When the selected interval is adjusted, this representation moves accordingly. Another function of the graph view is to display overlaid graph representations of the activity of any groups of selected genes. If genes have been selected in the scatter-plot view by lassoing, their graphs are displayed in white. If individual genes have been selected

with a single click, their representations are coloured red. This use of different colours is designed to help the user distinguish between selections made by either selection type if more than one type of selection has been made and both are active at the same time.

The interval controller panel of the first prototype interface allows the user to adjust the interval of the data to be displayed in the scatter-plot view. This panel has two sets of components. These are VCR style control buttons (play, stop and pause) and direct manipulation sliders. The sliders allow the user to select the size and starting time of the interval, and are tightly coupled so that when an interval is selected only valid start times for that interval size are available on the interval start slider. This ensures that no portion of the selected interval will lie outside the time range of the data. The play, stop and pause, VCR style control buttons allow the user to animate the scatter-plot view by setting the current interval start time to increment by small portions of the interval size at regular intervals of time. Here the interval size remains fixed. The play button triggers the animation to start, the pause button stops the animation with the interval start time being reset to the nearest time-point for which there is data and the stop button stops the animation with the interval start reset to zero. The animation also stops before the selected interval leaves the data when the sum of the interval size and interval start equal to the number of time points.

The selection mode panel of the interface allows the user to have selected genes assigned to any number of three different groups. Each of these has its own colour-coding and toggle buttons that allow the user to select either of the groups as the current selected group. The colours for groups are red for group A, green for group B and blue for group C. Brushed genes are assigned to the current selected group (replacing the contents of that group) and their representations in the scatter-plot coloured accordingly. If a gene is assigned to more than one group it's the colour of its representation is determined by combining the colours of the relevant groups. For example if the gene belongs to the green and blue groups B and C, its representation in the scatter-plot will be coloured cyan. The selection grouping panel also allows genes to be assigned to groups according to different logical rules [*169, 170*]. This is controlled by clicking on the buttons labelled with Venn diagrams on the right hand side of the panel. These either select the intersect of groups A and B, all genes in A

and B, genes only in A or genes only in B. When one of these buttons is pressed the resulting selection is stored in the current selected group, A, B or C, replacing its content.

The selected groupings panel allows the user to store and name selections. These can be exported or reselected at any time. This panel also contains expandable retractable folder icons that can be used to group and name groupings. This feature can be used if, for example, different sets of groupings relate to different patterns of activity over different intervals of the data and the user wants to refer back to them within such a context.

6.5 Rescaling

In the first prototype of the Time-series Explorer, if the data is displayed without being rescaled, it is dominated by low values and gene representations tend to cluster around the anchor related to low activity over all time points. This display of un-scaled data is only capable of communicating the fact that the data is skewed toward smaller values and the extent of outliers over different intervals of the data. At first it was considered that as rescaling is often referred to as a pre-process to be performed prior to analysis 'proper', it should be left up to the biologist to rescale the data before it is loaded into the software. While this worked to improve the distribution of gene representations in the scatter-plot, it prevented the users from referring back to values in the original data without having to switch between applications on their computer. To allow the users to do this a rescaling option is included in the menu bar. Here, the user can select to have the data un-scaled, have each gene rescaled to a proportion of its mean value or each gene rescaled to a proportion of its range of values. When the rescaling mode is changed, the representation of gene activity changes accordingly with any genes which were selected remaining so. This allows the user to relate patterns found using rescaled values to un-scaled values whenever necessary.

6.6 Evaluation

The primary objective of the first prototype evaluation was to assess its ability to allow users to relate motion to different types of changes in activity. In addition to this there was also a need to take on board any user feedback and perform an analysis

of how the users operated and interpreted the displays of the prototype. This would help guide the development of the next prototypes so that they could not only build on but also effectively refine the functionally of the first prototype.

6.6.1 Procedure

The evaluation of the first prototype involved two experienced biologists analysing familiar data. These were an experienced virologist who was responsible for designing the experiment from which the data was gathered and a statistician who had a role in statistical analysis of discoveries already made using the data. At this stage of development it was important to involve biologists familiar with the data as these were the persons most likely to be capable of relating motion in the scatter-plot to changes in activity. As inexperienced users would not be familiar with the fundamental characteristics of the data, it would be unlikely for them to be able to interpret any significance from the motion at all. While an evaluation with more participants would have been preferable in order to cross-validate the results, the nature of the research area (microarrays being a relatively new technology) meant that a limited number of persons were familiar with the data and available to take part.

In order for the evaluation to be productive it was necessary to encourage the natural processes of exploration, these being the processes that would allow the users to make discoveries and find unsuspected patterns. In order that this type of exploration might occur naturally, it was necessary to minimise any factors that could distract the biologists. It was, therefore, inappropriate to restrict the participants by instructing them to follow pre-defined tasks or operate in an alien environment. Instead, the main session of the evaluation procedure was to be relatively informal. Operating in their normal workspace the users were encouraged to operate the tool in a manner that was appropriate to their own working objectives with minimal disruption.

The evaluation proceeded in three stages. The first of these was a training session took place on the 18th of June 2003 involving a short tutorial guiding the users through the basic functionality of the technique. This was followed two days later on the 20th of June by a session where the users were asked to explore their data as per their normal working procedures. The data used in the experiment was the herpes

simplex data-set [*167*]. This was a relatively small data-set with activity recorded for 70 genes over ten time points (700 values) with each gene belonging to one of three functional groupings. Lasting approximately one hour, users were encouraged to think-aloud so that any findings could be noted. Four days later on the 24th of June in the final session of the evaluation the findings of the previous session were discussed with the user in order to determine which of their requirements were satisfied and how the technique could be developed to better satisfy these existing requirements or accommodate further requirements.

6.6.2 Results

Overall the impression of the biologists involved in the evaluation was quite positive indicating that they were indeed able to use such an animated technique to perceive changes in activity in microarray time-course data. Selecting individual gene representations in the scatter-plot view and having the activity of those genes highlighted in the graph view allowed them to familiarise themselves with the general layout of genes in the scatter-plot (Figure 6.6) within a short period of time. After this, the users tended to select smaller intervals (normally a unit interval, but occasionally two or three) and animate the scatter-plot by changing the start time of the interval using the interval controller slider. From this animated view they could tell which genes had rising and falling activity over different intervals of the data by relating the motion of gene representations to changes in activity. Once a biologist managed to find what they believed to be a potentially significant pattern in the data, they were often keen to select the genes associated by that pattern to uncover their names and view their characteristic pattern of activity in the graph view. Here the users found the lasso method for selecting genes to be satisfying as they could select genes by directly interacting with their representations in the scatter-plot without too much consideration of how the interface operated.

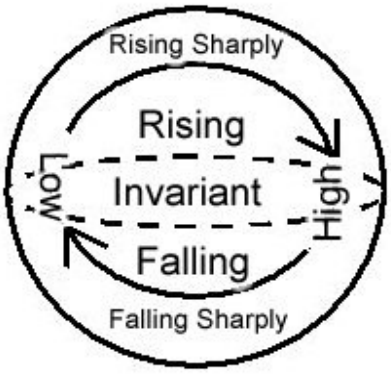

Figure 6.6. ***General layout of gene representations according to their activity over an interval in the first prototype.***

The users found longer time intervals less useful. This was mainly because, with larger intervals of data, there were a larger number of anchors acting on a gene representation and the user could not determine the degree to which any particular gene or group of genes were associated with any of the patterns. Here, the biologist would have preferred to be able to assess the degree to which genes were rising, falling, higher or low over the interval rather than try to determine the existence on any more complex patterns that could be detected otherwise by animating using a smaller interval. This would have allowed them to detect more subtle interval patterns that could not already be found with the existing scatter-plot layout.

Other issues uncovered by the evaluation of the first prototype were concerned with the usability of the interface. Here, the selection mode panel was described as overcomplicated. While the biologist had a need to combine their selections using different rules, the different colouring of selections was confusing. The biologists also felt that as they could directly interact with gene representations in the scatter-plot, that they should also be able to directly interact with the representation of the selected interval parameters in the graph view by clicking and dragging that representation to animate the scatter-plot.

Another issue raised by the evaluation related directly to the biologists interpretation of motion in the scatter-plot. While after a time they could properly interpret the motion as relating to changes in activity, their initial impression was that the display conveyed structure from motion as points representing genes appeared to form a three dimensional structure being rotated. This was despite the biologists' knowledge

that no such structure could exist. While it would be overambitious to try to assign and describe a specific cause for this phenomenon (especially since, on closer examination, the motion cannot be equivalent to that of the rotation of a static structure), it can be assumed that it is something to do with the scatter-plot layout being what is essentially a lower dimensional projection of a higher dimensional structure, the periodic activity of certain genes or, most likely, a combination of both factors. If this is the case then the phenomenon of structure from motion can be thought of as an unfortunate side effect of the layout used in the first prototype. In any case, it is a prime example of the unpredictable effects of motion but in this case it is not so detrimental since the user quickly realises that the motion does not represent structure. Once the illusion of structure from motion is broken it ceases to have an effect on the user's interpretation of the scatter-plot.

Chapter 7 Prototype 2: Animated Scatter-plots to find Activity Patterns

The first prototype of the Time-series Explorer demonstrates the ability of an animated scatter-plot to communicate changes in activity in microarray time-course data but it is limited in the types of patterns it can be used to reveal. Specifically, it cannot be used to find more subtle patterns where genes are associated according to a gradual change in activity or more complex patterns where genes share a number of common changes at different points in time. The design of the second prototype aimed to overcome this limitation so that the technique could be used to find a wider range of significant interval patterns. This was achieved primarily by replacing the original scatter-plot layout with less abstract design that directly represents the degree by which the genes' activity is changing and the levels of their high/low activity over the selected interval.

There are two main advantages to this more direct representation of gene activity over an interval. Firstly, being able to directly relate a genes position to its activity removes any ambiguity as to whether a gene's position relates to its activity or any unwanted artefact of the display mechanism. This makes it easier for the user to interpret more subtle changes in activity and detect genes with common changes in activity when those changes are more gradual over time. Secondly, a more explicit representation of gene activity allows the user to place a greater degree of confidence in the display. This makes them more likely to select genes and combine those selections to find more complex patterns in the data.

Figure 7.1 shows the design for the second prototype. This design is the same as that of the first prototype (Figure 6.1) with the addition of axes relating gene representations in the scatter-plot directly to levels of rising, falling, low and high activity over an interval. Figure 7.2 illustrates specifically how the positions of gene representations in the scatter-plot are calculated, by describing how the activity of a single gene over an arbitrary interval (P) is used to generate its coordinates. Here, the y-axis translation of a gene summarises its average activity over all time points of the selected interval. This average is calculated as the area under the gene's rescaled time-series enclosed by the bounds of the selected interval divided by the number of unit intervals. This method of calculating the average activity ensures that as the

interval selection is shifted by increments less that the space between two adjacent time-points, the Y-axes translation of a gene can be recalculated so that that gene's representation in the scatter-plot moves gradually along the y-axis. The translation of a gene representation along the x-axis is calculated as its activity at the end of the selected interval minus its activity at the start to give a measure of the change in activity. In this case when the interval is shifted by small increments, values are recalculated using linear interpolation so that gene representations shift gradually along the x-axis as well as the y-axis.

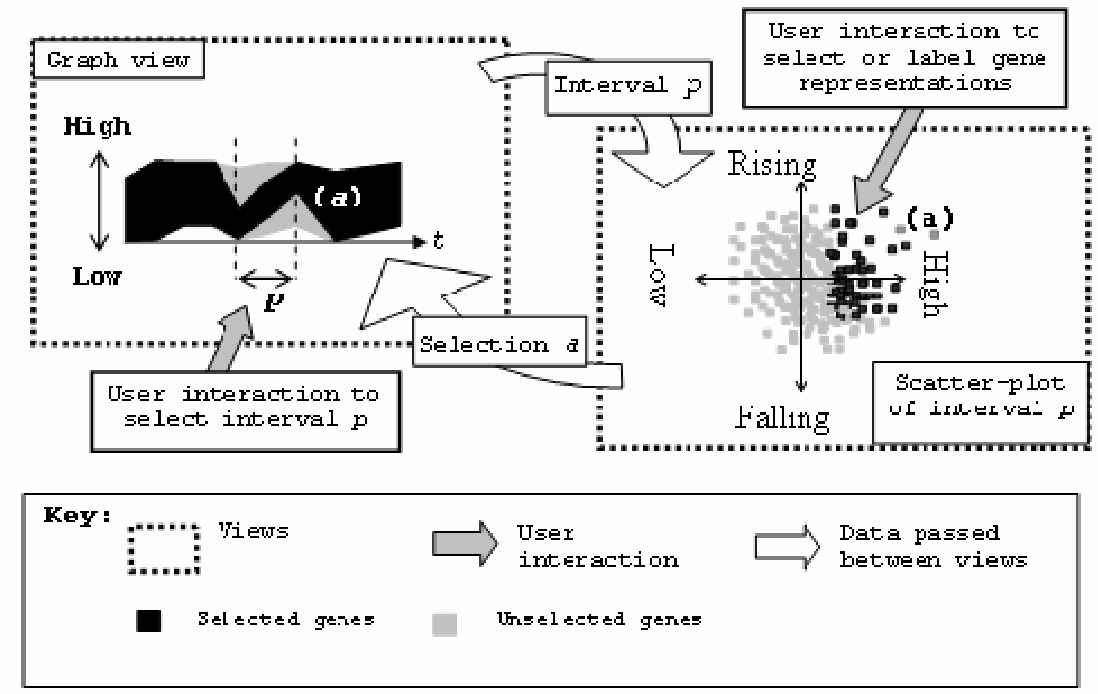

Figure 7.1. ***Basic design for second prototype Time-series Explorer***

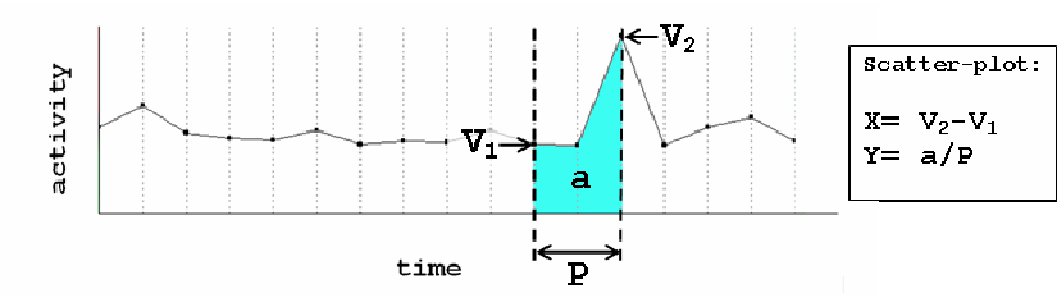

Figure 7.2. ***Attributes of a gene's time-series over an interval (P) used to determine the gene's scatter-plot coordinates.***

7.1 Design Rationale

The design of the second prototype Time-series Explorer was largely guided by the findings of the first prototype evaluation. Here, the users were able to perceive changes in activity from an animated representation of their data to find interval patterns but were unable to perceive more subtle patterns and were unwilling to combine selections over different intervals of the data to find more complex patterns. As more subtle patterns associate genes over larger intervals of the time-course, the second prototype is designed to convey these patterns using static interval views of such intervals. Specifically, the layout of the scatter-plot used in the second prototype summarises the activity of each gene over the selected interval using two attributes, its average activity and its change in activity. This allows for a simpler axes oriented scatter-plot layout, but it also creates a certain degree of ambiguity when relating the position of a gene to its activity over the interval. An example of two different gene activity patterns that generate the same axes coordinates is shown in Figure 7.3. This ambiguity will however be resolved if the interval is shifted as there is a very small likelihood of any two genes with different patterns of activity over an interval being in the same position in two successive intervals. So while the position of genes with different activity can be the same, their motion will separate them and genes can be grouped by motion rather than their position in static frames.

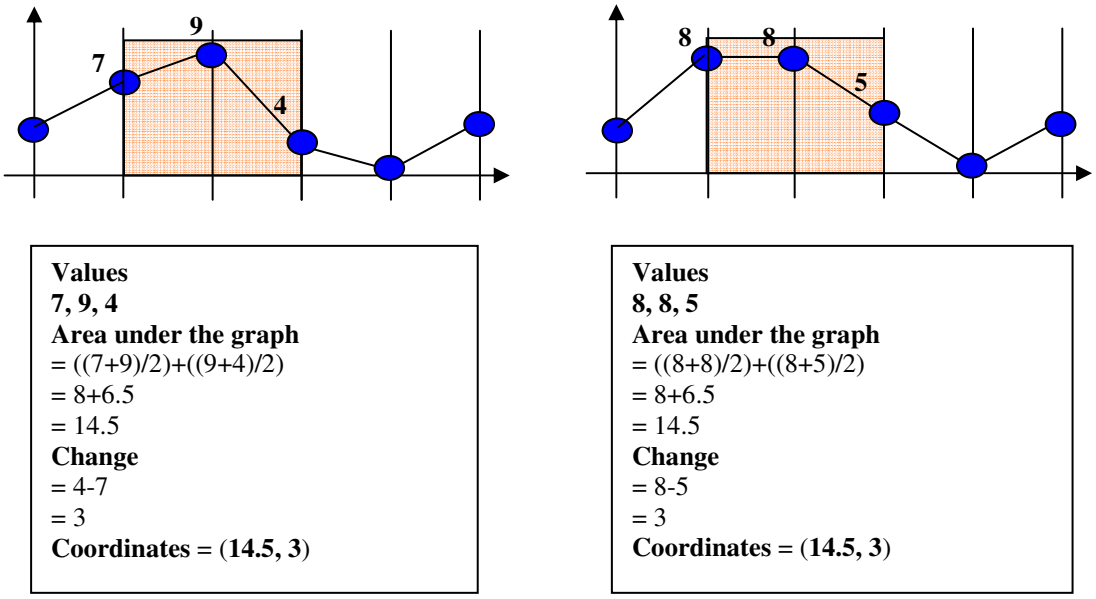

Figure 7.3. ***Two genes with dissimilar activity over an interval and the same positions in the scatter-plot.***

To support the user's efforts to resolve ambiguous gene positioning in the scatter-plot, especially for longer intervals, the prototype is designed to allow the user to filter gene representations according to additional attributes of their activity over the selected interval. These are the standard deviation of the activity over the interval, the deviation from the mean and the deviation from the change in activity, each calculated on a per gene basis:

- Standard deviation: This is a common statistical measure of how much values deviate from their mean.
- Deviation from average: The measure of *deviation from average* (Figure 7.4) is contrived for this particular scatter-plot layout and aims to indicate how well the average value, as calculated for a gene's representation in the scatter-plot represents its activity over the interval. This is done by calculating the area between the gene's rescaled time-series enclosed by the bounds of the selected interval and a line representing its average and dividing by the number of time points. The higher the value, the worse the genes average value is at representing its interval activity.
- Deviation from change: The measure of *deviation from change* (Figure 7.5) is a similar metric used to indicate how well the change value calculated for a gene's representation in the scatter-plot represents its activity over the interval. This is calculated as the area between the gene's rescaled time-series enclosed by the bounds of the selected interval and a line representing its change over the interval and dividing by the number of time points. Here the higher the value, the worse the gene's change value represents its interval activity.

It is imagined that using these attributes to filter gene representations a user can ensure that only genes whose activity over the entire interval can be effectively modelled are displayed.

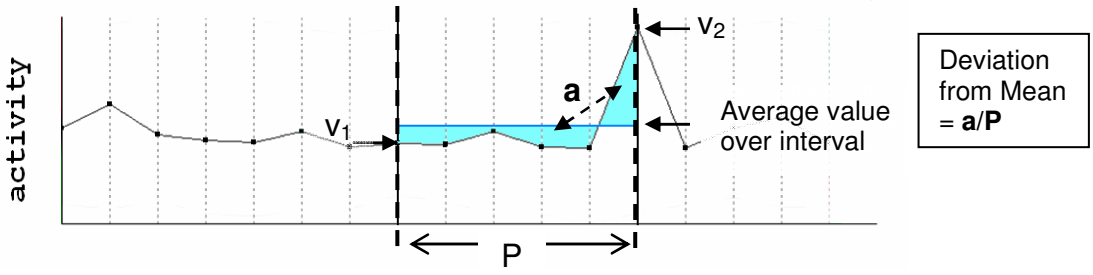

Figure 7.4. ***Deviation from mean attribute used to filter gene representations in the second prototype***

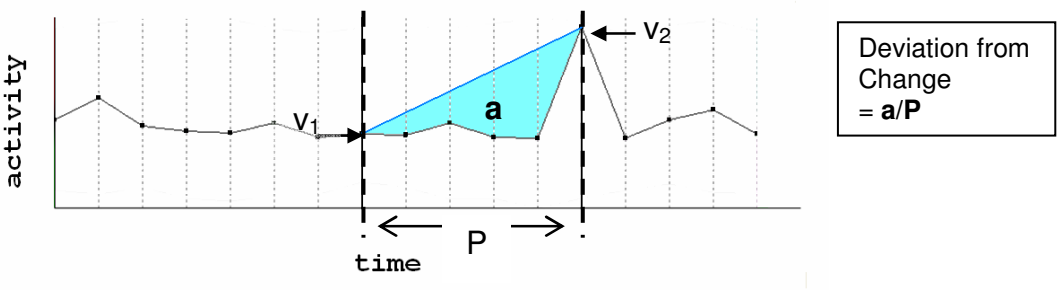

Figure 7.5. ***Deviation from change attribute used to filter gene representations in the second prototype***

Another advantage of the more explicit representation of gene activity in the second prototype is that it allows the user to place a greater degree of confidence in the display. This makes them more likely combine selections to find more complex patterns in the data. Here the user might detect a pattern over an interval of the data where a number of genes have similar activity. When those genes are selected, the user can use the graph view or an animation to find a more complex pattern where the genes have similar activity over another interval of the data. The scatter-plot of that other interval can then be used to select those genes for them to be identified and the pattern to be read. The existence of two intervals where activity is similar is likely to be more significant than similar activity over a single interval as it is less likely to be coincidental and more likely to relate to a real process.

7.2 Interaction

A screenshot of the second prototype interface is shown in Figure 7.6. The interface consists of seven panels containing; the time interval controller buttons, the graph view, dynamic query sliders to filter gene representations, the animated interval

scatter-plot, a list of selected genes, a list of defined groupings and a query construction panel.

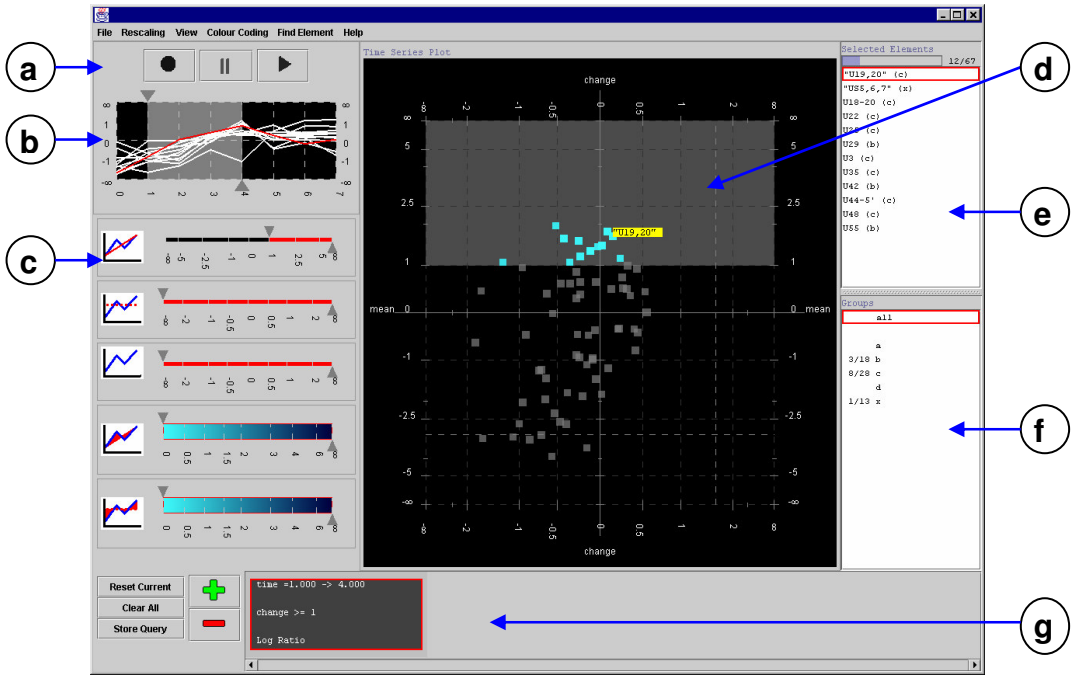

Figure 7.6. ***Screenshot of Prototype 2 interface: a) interval controller buttons, b) graph view, c) dynamic query sliders, d) scatter-plot view , e) selected genes list f) groupings panel and g) query composition panel.***

The interval controller play, pause and stop buttons (Figure 7.6a) are unchanged from the first prototype. These allow the user to animate the scatter-plot view by setting the current interval start time to increment by a small portion of the interval size at regular intervals of time. The interval control sliders from the first prototype are however replaced with a mechanism that allows the user to directly interact with the graph view to change the interval (Figure 7.6b). This functionality is essentially identical to that of a multi-range dynamic query slider [*171*] utilizing the internal slider space for a visual representation of data in a similar manner to that of data-visualisation sliders [*68*]. Dragging the edges of a vertical bar overlaid onto the graph-view allows the user to adjust its start and end times independently. Dragging the centre of the bar changes the start and end times with the duration remaining constant to shift the selected interval. The vertical bar also acts to represent the selected interval's timing, telling the user what portion of the data they are currently observing in the scatter-plot view.

The dynamic query sliders in the second prototype (Figure 7.6c) allow the user to filter gene representations in both the graph and scatter-plot views. This filtering is according to; any of the measures of deviation described in the previous section, the genes' mean value over the interval or the genes' range of values over the interval. As these attributes are dependent on the current selected interval, when the interval is adjusted the attribute values for genes change accordingly. This means that if a filter parameter is set, genes may become filtered or unfiltered whenever the current interval is adjusted.

The scatter-plot of the second prototype (Figure 7.6d) appears surprisingly similar to that of the first prototype with the absence of anchor labels and the addition of axes labels so that the positions of gene representations can be directly related to qualities of their activity over the selected interval. The procedure for selecting genes is also similar to that of the first prototype. Here the user can move the mouse over a gene representation to have that representation labelled or lasso a number of representations to have those genes selected with their identifiers listed. In the second prototype, however, genes are lassoed by dragging a rectangle rather a freehand shape and the lasso deselects genes outwith its bounds rather than selecting genes within its bounds. This is called a box-lasso and appears similar to a time-box [*10*] which is used to select items from a graph view. To create a box-lasso the user clicks on one point in the scatter-plot and drags the cursor to another point. These become the opposite corners of an unfilled rectangle drawn with horizontal and vertical edges. When the mouse button is released, gene representations outwith the bounds of the rectangle become deselected. The idea of making the user drag a rectangle rather than a freeform shape to select genes was to make those selections explicitly measurable so that they're associated parameters can be exported along with the identifiers of the genes that are selected. Deselecting genes already selected rather than selecting genes allows multiple lassos to be combined to find more complex patterns in the data.

The query composition panel (Figure 7.6g) allows users to manage box-lassos and adjust their parameters. When all genes are selected, either before the user performs any selection or after they have pressed the 'clear selection' button, the query

composition panel is empty. After the first box-lasso is formed a box appears on the far left-hand side of the query composition panel. This describes the parameters of the box-lasso (i.e. the conditions set and the interval selected when the lasso was formed) and characterises the genes that remain selected. When another lasso is formed a further box describing that lasso's parameters is added to the right hand side of the first box. Now the two boxes characterise the genes that remain selected. As more lassos are formed more boxes describing their parameters are added (Figure 7.7). These continue to characterise the selected genes and allow for the user to recall the steps taken in their selection. When a box in the query composition panel is clicked on, the relevant box-lasso can be refined. Here, the tool reverts to the interval where the lasso was originally formed and a representation of the lasso becomes visible (appearing as a transparent rectangle). When the edges of that representation are clicked on and dragged this changes the parameters associated with the original lasso and, whenever appropriate, the set of genes that remain selected. The user can also delete a box relating to box-lasso to remove that lassos effect on the gene selection.

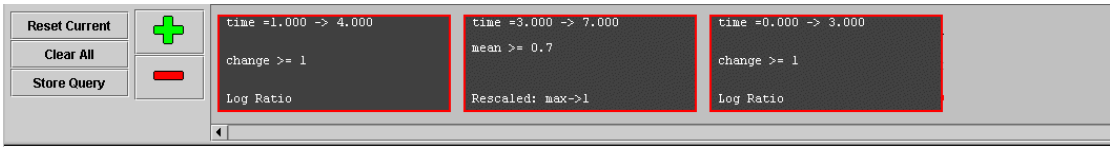

Figure 7.7. ***The query composition panel after three box-lassos***

Whenever a query is stored the user is asked to provide a name for the query and that name is added to the grouping panel (Figure 7.6f). This panel provides a reference to all query names together with the number of genes they select. If a grouping is selected using a query, clicking on that grouping's name allows the query to be restored. This causes the genes selected by that query to be reselected (with all other genes unselected) and the query composition panel to revert back to the state it was in when that query was last stored. If a grouping is *not* related to a query then only a list of genes is stored and when the grouping is selected the interface will only select the genes without altering the query composition panel. These types of grouping are made by importing a plain text file that list's the users selection of pre-defined groupings in comma separated value (CSV) format.

7.3 Evaluation

The objective of the second prototype evaluation was to assess its ability to allow users to explore their data to find different types of activity patterns. Here the procedure for evaluation was the same as that for the first prototype involving a small number of experienced biologists analyzing familiar data with feedback provided in a follow-up meeting. The biologists involved in this evaluation were a statistician who were involved in statistical analysis of discoveries already made using this and other data, a biologist with experience in microarray experimentation and a bioinformatics expert with experience of analysing similar data. The data used in the evaluation was the herpes simplex data [*167*] used in the evaluation of the previous prototype. As with the previous evaluation the procedure involved three session. These took place on the 27th, 28th and 30th of June 2003. During the first session the biologists where trained to use the software. In the second session the users where recorded using the software to explore the data. Initially this was planned to involve the use of a Dictaphone to record audio as the biologists operated the tool but they felt uncomfortable about being recorded this way and ultimately only written notes where taken. After each user had spent about an hour exploring the data they where asked to discuss their use of the software in more detail. These discussions where written up and detailed further during the final session of the evaluation. This took the form of a general meeting involving myself and all the biologists involved in the evaluation discussing the issues raised in the evaluation together.

7.3.1 Results

The main outcome of the second prototype evaluation was to show the potential of a more direct axis oriented scatter-plot representation to better support the biologists' process of extracting knowledge from microarray time-course data. The evaluation also revealed some incongruities related to the specific scatter-plot layout and the mechanisms of the interface in general. These primarily related to the orientation of the scatter-plot axes, the configuration of the *change in activity* axes, the manner in which gene selections were made and the manner in which those selections could be combined to find more complex patterns in the data.

As with the evaluation of the first prototype the users were still able to interpret changes from motion. Indeed, in order to do so the operation of the second prototype was largely similar to that of the first. The main difference in operation was that the users felt more willing to select larger intervals of the data. With the second prototype, this allowed them to find some of the more subtle patterns of common activity. This was not, however, done in the manner envisaged during the prototype's design. Instead of using the dynamic query sliders to differentiate between subtle and complex patterns, users rapidly switched between the graph view and scatter-plot combining a multiple of small interval queries to gradually refine their original query toward finding either type of pattern. Indeed, most of the users chose to forgo the dynamic query sliders altogether after a relatively short period of time. This was down to a number of factors. Firstly, the attributes by which the user could filter were not easily understood and they preferred to accept that a view of a larger interval would have a certain proportion of genes with complex rather than subtle patterns of activity. Next, whenever a query parameter was set the user would often forget it was set and as the interval selection was adjusted genes would become selected and de-selected. This caused a number of gene representations to flash on and off in a rather distracting manner. Finally, despite any limitation of the query sliders themselves, the users felt that their own method for distinguishing between complex and subtle patterns (i.e. animating across different intervals) was more effective.

Although the users were able to select genes and combine selections to find both subtle and complex patterns, there were certain issues with the way in which the interface allowed them to do this. The user felt that having to drag a rectangle to select any more than one gene was too restrictive. Lassoing to select genes in the first prototype gave the users a feeling of direct manipulation which was lost in the second prototype. Quite often the users wanted to select a visible cluster of gene representations in the scatter-plot rather than query according to attributes. This action was far simpler when dragging a freeform shape rather than a rectangle, especially if the cluster had some irregular form. Also, the users found the labelling operation (where genes under the mouse cursor are labelled) useful but not as powerful as they would have liked. Users would select different numbers of genes within a visual cluster and try and build up an idea of its composition using the gene

list panel when it would obviously be more appropriate if they were able to label multiple genes in a cluster.

In combining selections the users often felt the query composition panel to be over-elaborate. While appreciating the ability to select genes according to measurable attributes, the users did not want the query attributes to be recorded and presented textually. Here they would have preferred to simply refer to a graph view of the activity of the selected genes. This made the query composition panel redundant suggesting that it should not be included in any further prototypes. Another minor irritation was that selections being made would always refine the previous selection. The users felt that a new selection should replace earlier selections by default and if a selection is to refine the results of a previous selection then this should be because they have implicitly requested such an action. Here the users also wanted the option to add further to a selection already made by *summing* selections.

In general the users felt that the scatter-plot layout of the second prototype to be far more useful than that of the first prototype. There were, however, a few suggestions made as to how the layout could be improved. The first of these related to how the change in activity for a gene over an interval was calculated. Here, the biologists would have preferred the tool to use the ratio between values than the difference between values. The users considered this to be a more realistic indication of biological activity as absolute values are already discounted by the rescaling and the proportion that the recorded value changes by is more significant than any absolute change. So, for example, it would be more appropriate to associate genes with an approximate halving or doubling of their activity over time than base any association on absolute or rescaled values. It was also suggested that the data should be rescaled to a proportion of the gene's median value rather than its mean value or its maximum/minimum values. This was because the recorded values for a gene have a log normal distribution and a small number of larger values have an undue influence on the mean. Using the median of values would rectify this problem as a median is significantly less sensitive to outliers [*172*].

Another suggestion for the next prototype was to adjust the orientation of the scatter-plot axes. While, for some unknown reason, the biologists preferred the clockwise

motion of genes (perhaps relating this motion to the hands of a clock as time progressed), they felt uncomfortable with the average activity axis being on the horizontal as this was perpendicular to the activity axis of the graph view. Here the biologists felt that activity was more related to mean activity than change in activity and if these axes were made parallel it would be easier to relate between the graph view and the scatter-plot. Other suggestions were that the user should be able to cross-reference predefined grouping with their selections and refer back to the original (un-scaled) data whenever a group of genes are selected without adjusting the scatter-plot.

Over and above these suggestions relating to how the scatter-plot might be improved, the evaluation also highlighted another factor that might limit its usefulness. This happened whenever the tool was being used to look at larger numbers of genes. This resulted in the display being saturated with considerable overplotting. Here large numbers of gene representations clustered around the centre of the scatter-plot and representations being clumped in such a manner that there would be no space between them and thereby no way of determining their density within any particular area. The biologists felt that this would impact on their ability to perceive general trends either over an interval or as the scatter-plot was animated over time.

Chapter 8 Final Prototype: Animated Scatter-plots to Support Exploratory Analysis

The first Time-series prototype demonstrated the ability of an animated representation to allow biologists to effectively convey changes in activity over time for large-scale microarray time-course data. The second prototype used a more direct axes oriented representation of activity over intervals of the data which could be animated to allow the users to find a greater variety of significant patterns. There were, however, certain issues regarding the specifics of the representations and the ways in which users were allowed to interact with these representations that detracted from this functionality and prevented the technique from being used to its full potential. The third and final prototype is designed to resolve these issues and improve the technique's general usability toward supporting the full process of extracting knowledge from microarray time-course data (as described in Chapter 5).

In making the changes necessary to better support the process of extracting knowledge from microarray time-course data, the design of the final prototype remained unchanged from that of the first and second prototypes. Indeed the diagram describing the second prototype (Figure 7.1) still describes the final prototype with the minor difference that the axes have switched places. The important differences between the second and final prototype are more subtle. These are; the rescaling of the data, the way in which gene displacement along the change axis is calculated (which requires the display to be distorted along that axis), the colour coding of the scatter-plot to communicate the density of gene representations, the way in which genes are selected and the addition of a change in activity against time graph.

8.1 Design Rationale

Most aspects of the final prototype's design directly relate to assertions made by the biologists during the follow up meeting for the evaluation of the second prototype. Here the amount of redevelopment varied from changes that merely required minor adjustments to the existing code, such as switching the scatter-plot axes, to those that required new features to be developed and integrated, such as the colour coding to communicate the density of gene representations in the scatter-plot. Other aspects of the final prototype design were features incorporated to further improve the way

users could explore the data. These included changing the way users could select genes and the introduction of an additional graph view to provide an explicit representation of activity change over time.

In reality the final prototype would probably be better be described as a sixth or seventh prototype as there were a number of different minor prototypes developed after the second prototype. Each of these prototypes successively refined the functionality of the second prototype toward the functionality of the final prototype. This helped to develop the interaction mechanisms of the final prototype which were tuned to the users' requirements as changes were made in accordance with the feedback provided. As the changes made to the design normally related to separate aspects of the technique's overall design, the rationale for each change is best described alongside the aspect of the design to which it relates.

8.2 Rescaling and Distortion

From the evaluation of the second prototype it was found that a better representation of biological activity could be achieved by changing the way in which the data was rescaled and using a different measure of change in activity for the scatter-plot. This is because in order to effectively communicate biological activity from microarray data it is necessary to account for the fact that the significance of a gene's activity is more closely related to rational changes in expression (either between time-points or from some base level) than absolute values. While rational changes can be derived from absolute values, the fact that changes in expression tend to be proportional to expression levels causes the data to have log-normal skewed distribution with a large number of low values and a small number of high outliers. As low values with low changes may hold valuable biological information it is necessary to adjust the spread of the data in any visual representation so that the significance of these features is appropriately communicated.

The majority of existing techniques (for example [*11, 12, 111-115*]) account for this characteristic by rescaling the data using a log-transform [*2, 110*] (Eq. 4.4). This makes rational-changes comparable across genes and the distribution of values close to normal distribution [*110*] with the disadvantage that as $v(t)$ tends to zero $LS(t)$ tends to negative infinity. This means that the rescaled data cannot be completely

displayed using a regular finite scale, such as the axes of a graph or scatter-plot, without grossly overemphasizing small changes in very small values.

As an alternative to logarithmic rescaling, linear rescaling is used and the axes onto which the values are represented are distorted. While linear rescaling makes values comparable across time-series, distortion improves their distribution in the display and represses the dominance of large outlying values. Eq. 8.1 describes the linear rescaling as it is applied to each gene's time-series with *V* the original time-series of a gene, *median(V)* the median of all values for that gene and *MS* the rescaled time-series.

$$MS(t) \leftarrow V(t) / median(V)$$

Eq. 8.1 Linear rescaling

The median of a time-series can be considered as a statistical measure of what can be considered as a *normal* value accounting for a skewed distribution of values. Each value in the rescaled time-series of a gene is equal to the proportion of its corresponding value in the original time series to the median of all its values. This means that, in the rescaled time-series, anything below 1 is below normal and anything above 1 is above normal.

Eq. 8.2 describes the distortion applied to the graph view y-axis for rescaled values. Here *V* is a value, *Ydisp* the position for plotting the value on the axis and C_1, C_2 and C_3 are derived so that the maximum and minimum values are at the top and bottom of the allowed display space with *V*=1 (the *normal* value) at its mid-point.

$$Ydisp \leftarrow \log_2(V \times C_1 + 1) \times C_2 + C_3$$

Eq. 8.2 Logarithmic distortion

An advantage of combining linear rescaling and logarithmic distortion in this manner is that when two values for a gene are divided to calculate the rational change in value between time-points, the factors used to rescale the data (both being the genes median value) cancel out. This means that the rescaled data can be used to calculate values for the X-axes of the scatter-plot which is used to display rational change.

After this the distortion of axes improves the distribution of data in the display without actually changing the values to be displayed.

In the final prototype of the Time-series explorer scatter-plot each gene is represented as a single point with its translation along the y-axis corresponding to its change from normal expression over the selected interval and its translation along the x-axis corresponding to its change in expression from the first time point to the second. Figure 8.1 illustrates how these values are calculated for a single gene over a given interval (P). As the selected interval may contain a multiple of time-points for which expression has been recorded, the y-axis translation summarizes a gene's change from normal expression as its average change from normal expression over all these time points. This average is calculated as the area under the gene's rescaled time series enclosed by the bounds of the selected interval divided by the number of time points enclosed. This ensures that as the interval selection is shifted by amounts less a unit interval, the y-axis translation of a gene can be recalculated so that that gene's representation in the scatter-plot moves gradually along the y-axis. In order to calculate the translation of a gene representation along the x-axis the value at the end of the selected interval is divided by the value at the start to give a measure of the rational change in expression. In this case when the interval is shifted by small increments, values are recalculated using linear interpolation so that gene representations shift gradually along the x-axis as well as the y-axis.

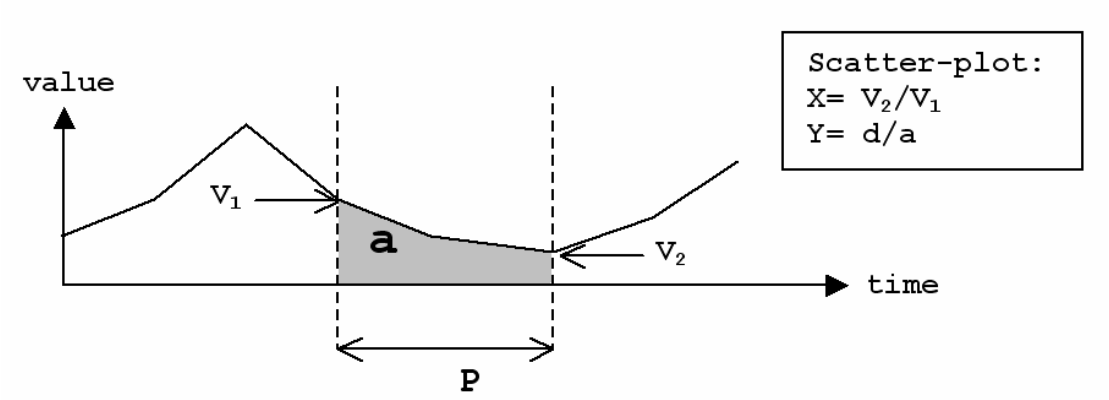

Figure 8.1. ***Attributes of a gene's time-series over an interval (P) to be mapped to the scatter-plot representation.***

As the distribution of values along the y-axis of the scatter-plot is roughly equivalent to the distribution of values in the rescaled data, in order to make the distribution normal it is appropriate to use the same transform as that used for the y-axis of the

graph view (3). While the distribution of values along the x-axis is also similar to the distribution of values in the rescaled data, the same transform cannot be used. This is because when V_1 tends to zero *Xdisp* will tend to infinity and logarithmic transforms cannot translate values tending to infinity onto a finite range. Instead, an alternative distortion transform was constructed using the hyperbolic tangent function. This function is similar to the logarithmic function with the notable exception that as x tends to infinity the hyperbolic tangent of x tends to one. The transform as applied is described in Eq. 8.3 where X is the derived value of change in expression over the selected interval, *Xdisp* is the position for plotting on the axis and C_1, C_2 and C_3 are derived so that the value for no change (X=1) is in the centre of the display space and the values for biologically significant halving or doubling of expression (X=0.5 and X=2) are one and three quarters along the display space.

$$Xdisp \leftarrow TanH(X \times C_1 + 1) \times C_2 + C_3$$

Eq. 8.3 Hyperbolic transform distortion

The resultant spread of data in the scatter-plot view is illustrated in Figure 8.2.

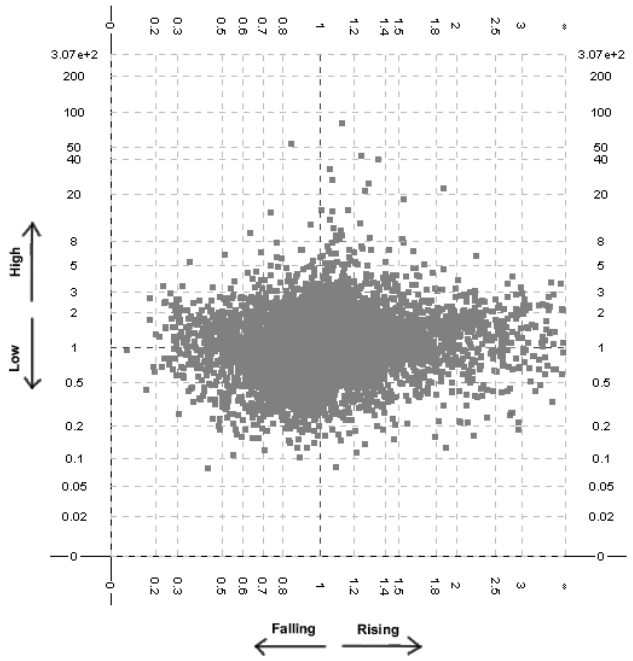

Figure 8.2. ***The spread of data in the distorted scatter-plot view.***

8.3 Colour Coding

While rescaling and distortion improve the spread of the data in an interval scatter-plot view allowing for better detection of outlying patterns of temporal activity, the sheer volume of gene representations saturates the display with overplotted points creating a general grey mass of in the centre that makes it difficult to perceive more general trends (Figure 8.2). While a transparency composite [*18, 173*] would allow a user to more accurately perceive general trends by indicating the relative density of overlaid gene representations, it would also make it harder to distinguish outliers from the background as smaller numbers of overlaid elements are represented with their colour closer to that of the background. As an alternative, the final version of the Time-series explorer includes a colour mapping composite that communicates the density of genes through a colour-scale where outliers are significantly different in colour from the background and, therefore, easy to distinguish from the background.

The colour composite used in the final version is similar to a standard transparency composite in that each pixel of the display has an alpha value that is increased by the alpha value of overlaid elements. The essential difference is that alpha values are translated into a colour scale rather than used to combine the colours of overlaid elements. The scale used (shown in Figure 8.3a) ranges from dark-blue, for small numbers of overlaid genes, through blue, cyan and green to yellow for larger numbers of overlaid gene representations. This composite attempts to utilize as much of the visible spectrum as possible without using reds or greys, which are more appropriately used to represent highlighted and deselected genes. The ordering of colours is such that light colours represent a high density of genes and dark colours represent a low density of genes. This ensures that dark colours surround light colours, which would be otherwise hard to distinguish from the background. Colour-coded graph and scatter-plot views are shown in Figure 8.3b and Figure 8.3c.

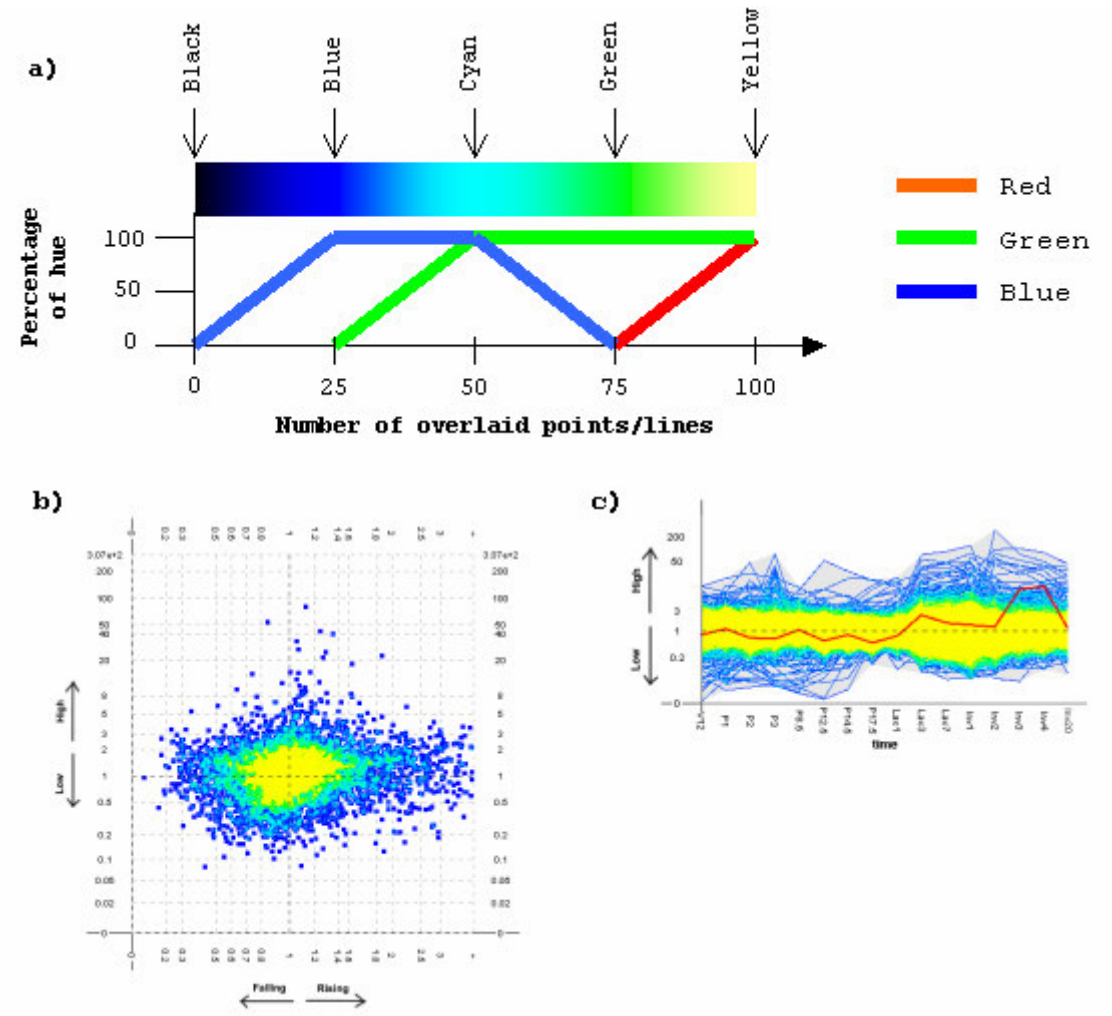

Figure 8.3. ***The colour mapping used for the Time-series Explorer: a) The percentage of each hue corresponding to the number of overlaid points or crossing lines, b) the scale applied to the scatter-plot view and c) the scale applied to the graph view.***

A problem with the implementation of the colour coding for overlaid gene representations is that it takes significantly more time to render a display with colour coding on than it would with colour coding off. This was particularly problematic for the graph views which required a graph of activity against time to be drawn for each selected gene. While image buffering (see Appendix 2) can be used to ensure that the graph views do not need to be redrawn when different genes are highlighted, these views still need to be redrawn when the user selects a different group of genes to be displayed or the display panel holding the graph views is resized. Here the amount of time required to redraw is proportional to number of genes selected. If the processor is completely dedicated to rendering the image for such a representation whenever a large number of genes are selected the time for which the interface is unresponsive will be unacceptable. To avoid this, the graph view is redrawn initially without

colour coding and the representation is only of the range of values at different time-points (as with the first two prototypes). After this a separate program thread redraws the graph views *with* colour coding to an image cache using a small enough proportion of the computers processing power so that the interface remains responsive. While this image is being rendered a status bar informs the user of the progress made using a horizontally elongated rectangle that gradually fills with colour from the left hand side. Once rendering is complete the more informative colour coded image replaces the initial representation.

Restricting allocation of processor power to the rendering of colour-coded graphs to avoid the interface becoming unresponsive has a number of advantages. Firstly, an unresponsive interface is generally inadvisable as the user is likely to attribute unresponsiveness to a program error and desist from using an application that they perceive to be faulty. Secondly, after a number of genes are selected to be colour coded it is conceivable that the user would not necessarily want to use the graph view immediately and, even if they did, they could perform some other actions using the views available and should not have to wait to do so. Lastly, the progress bar gives the user continuous feedback [*52*] to keep the user informed of the situation where limits in the capabilities of the processor make it impossible to deliver a rapid display of results as prescribed by the rules of direct manipulation [*52*]. A potential drawback of the progress bar is that its motion might distract the user's attention and that it is not immediately apparent that it relates to the rendering of graph views. If further time can be allocated to development it is conceivable that the status bar might be replaced with a less distracting indicator more directly related to the graph views (e.g. an animated hour glass icon beside each view).

8.4 Parallel Views

One of the suggestions made during the evaluation of the second prototype was to adjust the orientation of the scatter-plot axes so that the average activity axis was parallel to the activity axis of the graph view. This was to make it easier to relate between the views when finding patterns in the data. The user would tend to use the scatter-plot view when they wanted to look for patterns over an interval of the data, specifically when a large number of genes with dissimilar activity were selected and those patterns could not be perceived in the graph view. When a number of genes

were selected and highlighted in both views, if it was conceivable that patterns could be visible in the graph view, the user would switch to the graph view. If no patterns (or no patterns of any interest) were visible in the graph view, the user would quickly revert back to the scatter-plot. Making the average activity axis of the scatter-plot parallel with the activity axis of the graph view makes it easier for the user to relate between these views and easier for them to switch between views. Observing the technique being used in this manner (in an intermediate prototype) also prompted two additions to the design. These were an additional graph view directly representing changes in activity over time and a split pane [*174*] containing both graph and scatter-plot views so that the relative screen space allocated to either view could be adjusted.

The change-in-activity graph view of the final prototype (bottom of Figure 8.4) overlays change-in-activity against time graphs to indicate the range of rising or falling activity of selected genes between each pair of adjacent time points. Change-in-activity is defined as the relative change in recorded activity between time points for a gene. This is approximate to the quality of a gene's activity that is displayed using the x-axis of the scatter-plot for smaller intervals. In effect the change-in-activity graph is to the x-axis what the original activity graph is to the y-axis, giving a summary of an equivalent quality over time. This graph is also used in a similar manner to the activity graph, being used whenever genes are selected and a pattern can be perceived most appropriately using that representation. The utility of the change-in-activity graph over that of the activity graph is demonstrated in Figure 8.4. Here there are certain patterns over time that can be detected using the change-in-activity graph but *not* the activity graph. Specifically, significant trends in change-in-activity (from V12 to P1 and P3 to P8.5) are evident only from the change-in-activity graph view.

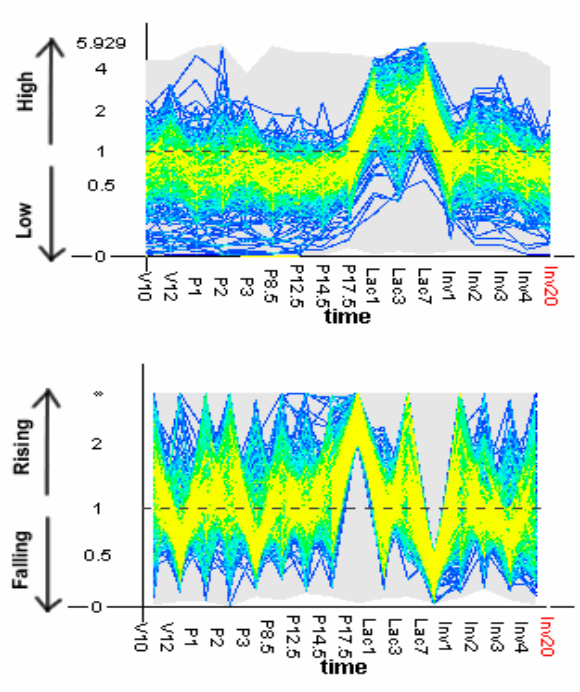

Figure 8.4. ***Change-in-activity graph view (bottom) can be used when significant trends in change-in-activity (from V12 to P1 and P3 to P8.5) are obscured in the activity graph view (top).***

In another attempt to better support the user's process of switching between views in order to find patterns, a split pane [*174*] was introduced to allow the user to adjust the relative amount of screen space allocated to either scatter-plot or graph views (Figure 8.5). Here, both the scatter-plot and graph views occupy the same panel and are divided by a vertical bar which is roughly in the middle of the panel. If the user clicks on this bar it can be dragged along the horizontal to indicate the relative proportion of the panel each view should take up. When the mouse button is released, each view is resized so that it remains on its original side of the bar. As axes labels remain in a consistent size of font (i.e. one that of a reasonable size to be readable), this occasionally means axis labels need to be removed from a view or additional labels can be inserted for further clarity. The axis labels that are included with either view are dictated using an algorithm that attempts to include the most useful numbers (i.e. the round numbers that are easiest for the biologist to remember).

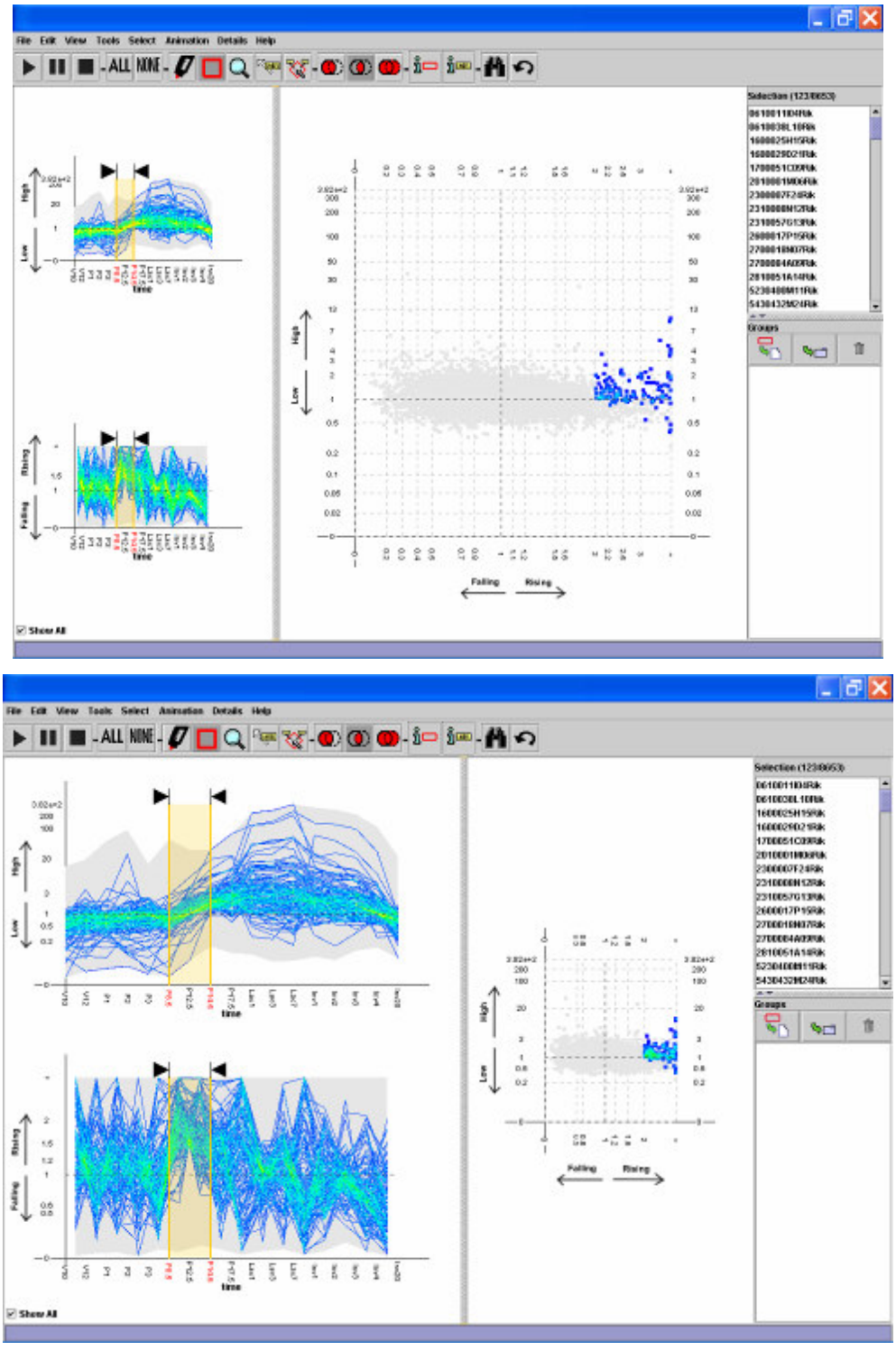

Figure 8.5. ***A split pane allows the user to adjust the relative proportions of the scatter-plot view and graph views: Scatter-plot bigger (top) and graph views bigger (bottom).***

Allowing views to be resized recognises the fact that as the user switches between views and their analysis evolves to look at smaller or larger number of genes, the relative importance of either view will change. Changing the screen space allocated

to either view allows the user to look at the most relevant view in more detail without necessarily loosing any utility that the other views might still have.

8.5 Selecting Genes and Combining Selections

In the final version there are a number of ways in which a user can select genes and have those selections combine to reveal different types of patterns. These are best described with reference to a screen-shot of the final version interface (Figure 8.6). This contains five main panels with which the user can interact in order to manipulate the representations of their data. These are the toolbar, graph view, scatter-plot, gene list and grouping panel.

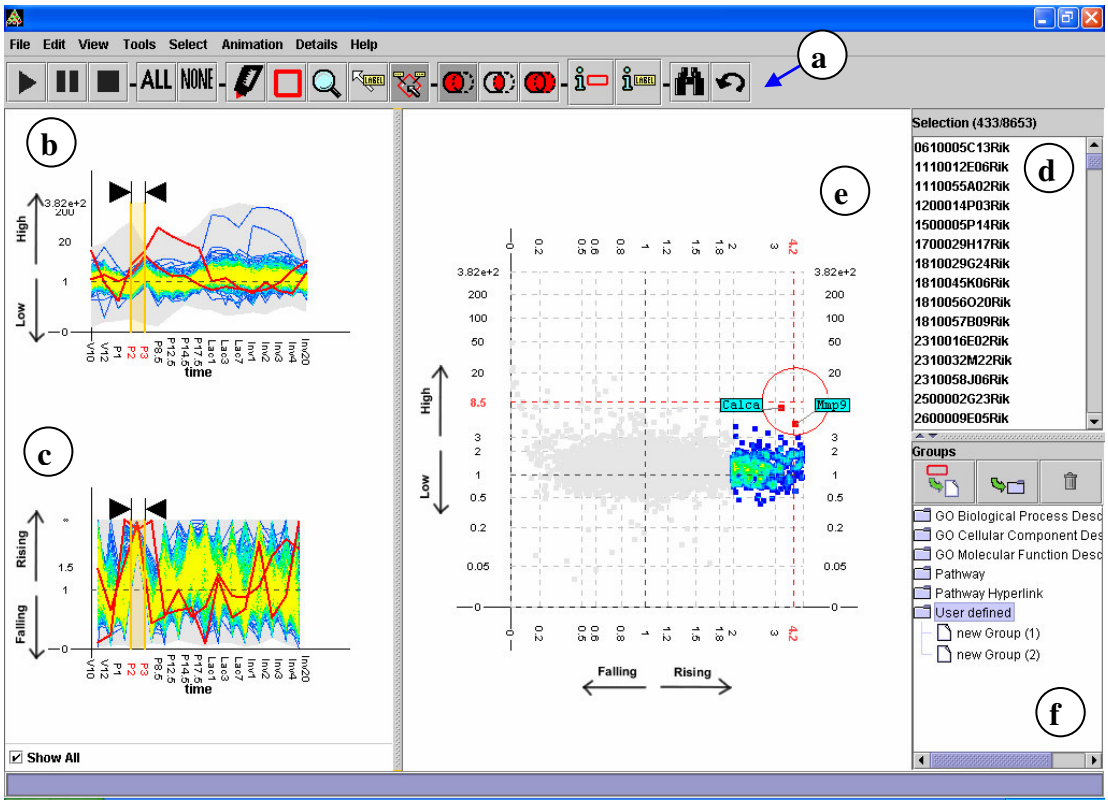

Figure 8.6. ***Time-series Explorer final prototype interface: a) toolbar, b) graph view, c) change-graph view, d) selected genes list and e) scatter-plot view and f) grouping panel.***

The toolbar (Figure 8.6a) contains 17 buttons in five groups with various different functions such as animating the scatter-plot view, changing the selection mode on the scatter-plot and viewing details for a selection (see Table 8.1).

Table 8.1. Functionality of final prototype toolbar buttons.

Category	Icon: Name: Action
Animation	**Play:** Animates the scatter-plot by increasing the start and end times of the selected interval at regular intervals of time.
	Pause: Pauses the animation.
	Stop: Stops the animation.
Selection	**Select all:** Selects all genes.
	Select none: Deselects all genes.
Scatter-plot tools	**Lasso selection:** When selected allows the user to select genes by lassoing their representations in the scatter-plot.
	Box-lasso selection: When selected allows the user to select genes by dragging a box around their representations in the scatter-plot.
	Zoom tool: When selected left clicking on the scatter-plot zooms in, right clicking zooms out and double right clicking zooms out fully.
	Labelling tool: When selected moving the mouse over gene representations in the scatter-plot causes them to be labelled and have their activity highlighted (over the entire time-course) in the graph view.
	Excentric labelling tool: Similar function to that of the labelling tool (above) with the exception that all gene representations within the bounds of a circle are labelled. Right clicking increases the size of the circle and left-clicking decreases its size.
Selection mode	**Replace selection mode:** Successive selections replace the previous selection.
	Refine selection mode: Successive selections refine the previous selection (equivalent to combining the two results using a logical AND operation).
	Add-to selection mode: Successive selections add to the previous selection (equivalent to combining the two results using a logical OR operation).
Details	**Selected gene details:** Activates a pop-up window with details-on-demand for the selected genes including a cross-reference with other groupings and a list of the genes selected with reference to groupings.
	Labelled gene details: Activates a pop-up window with details-on-demand for the labelled genes including a list of the groupings to which the gene belongs and it's original recorded activity values.
Other	**Find gene:** Activates a pop-up window that allows the user to find a gene by typing its name.
	Undo: Undoes the previous selection.

The graph view (Figure 8.6b) allows users to adjust the selected interval to focus [*70*] in on a specific interval or animate the interval scatter-plot to reveal general trends and outliers across the time-course. The interaction mechanism of the graph view is essentially the same as that of the second prototype allowing the user to select and adjust the interval displayed in the scatter-plot. The change-graph view (Figure 8.6c) is positioned below the activity graph view and linked so the greying out and highlighting of genes is consistent. The change-graph can also be used to select or adjust an interval of the data. This is the same interval that is selected in the graph view and the vertical bar representing the interval is consistent in both graphs.

Once the user ceases interacting with the graph view there are a number of different options for interacting with the scatter-plot (Figure 8.6c). Most of these interactions employ standard brushing and linking [*70*] information visualisation operations. If the labelling tool is activated from the interface toolbar, moving the mouse over gene representations in the scatter-plot view causes them to be labelled and have their

activity patterns highlighted in the graph view. The functioning of the excentric labelling tool (adapted from [*69*]) is similar to that of the labelling tool with the exception that all gene representations within the bounds of a visible circle are labelled and highlighted (Figure 8.7). The inclusion of excentric labelling was largely a reaction to the results of the second prototype evaluation where the users wanted to identify large numbers of genes without having to use a lasso. Here, the additional information revealed by labelling and the subsequent coordination between scatter-plot and graph views allows the user to rapidly perform a more informed assessment of a pattern's significance. If the user is interested in a smaller number of genes or wishes to investigate a sample of selected genes in more detail, double-clicking on gene representations in the scatter-plot allows them to view a pop-up details-on-demand [*36*] window describing the un-scaled recorded intensities for the subject gene and a summary of the groupings to which the gene belongs. This, again, will lead to a more informed assessment of a pattern's significance.

As an alternative to labelling, genes can also be selected when the lasso or box-lasso tools in the toolbar are activated. These options were also available in the first and second prototypes and feedback from the users after the second prototype evaluation indicated that both options should be incorporated into the final version. The procedure for box-lasso selection is, however, changed from that of the second prototype. In the final version, whenever a box-lasso selection is made the parameters of that selection are not stored or presented to the user. This is because normal lasso selections cannot be quantified in a similar manner and to quantify a selection only according to box-lasso selections would be misleading. When either type of selection is made, the representations of un-selected genes are greyed out in both graph views and scatter-plot views allowing users to focus in on selections which are colour-coded, labelled, animated and selected again (using logical AND or OR rules) independent of the un-selected data. This allows the user to find groupings within groupings and combine selections to uncover more complex patterns in the data. If an erroneous or undesirable selection is made it can be reversed by pressing the *undo* button located in the toolbar.

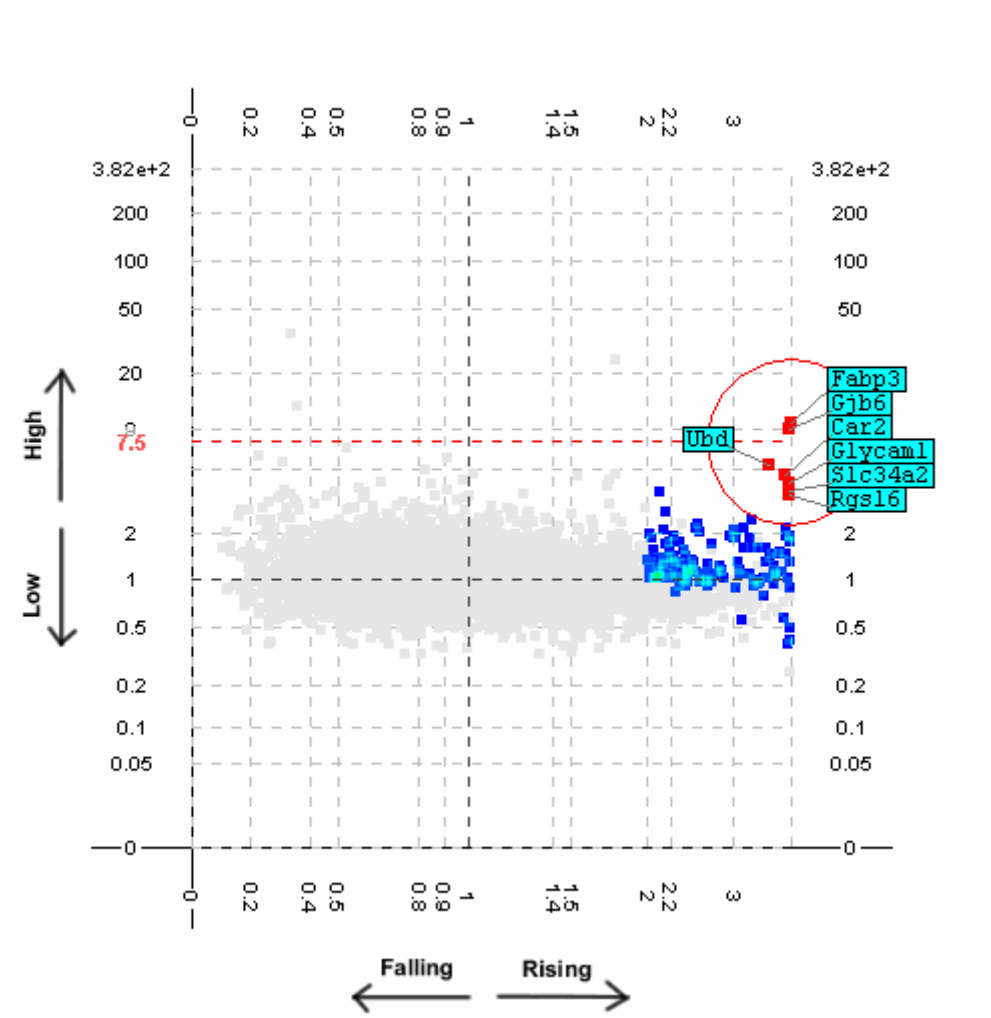

Figure 8.7. ***Excentric labelling of the Scatter-plot***

The interface of the second prototype also allows selections to be stored in the grouping panel (Figure 8.6f), saved selections to be restored, pre-defined groupings to be imported as stored selections and stored selections to be cross referenced in a pop-up table. In all, these actions increase the user's ability to relate the patterns they find to each other and to existing biological knowledge. In the latter case this allows them to make more informed judgements as to the biological significance and relevance of their discoveries. This increases the user's capacity to extract knowledge from the data that can be used in turn to find further interesting patterns the data.

8.6 Evaluation

The objective of the final prototype evaluation was to asses the full projected functionality of the technique by assessing its ability to allow a biologist to extract

relevant biological knowledge from their data. There was also a need to assess the practical utility of the Time-series Explorer within the wider context of microarray time-series analysis and, in particular, the extent to which Time-series Explorer could be considered as a specialised microarray time-course data exploration tool which complements and adds to the functionality of existing clustering techniques. To do this it was necessary to discover if the technique was capable of overcoming the limitations of existing techniques to reveal previously unsuspected interval patterns of temporal activity and, to a lesser degree, assess its ability to uncover certain suspected patterns of temporal activity in order to highlight the areas where it may be advantageous for a biologist to use the Time-series Explorer in preference to other more established techniques.

With reference to the objectives stated the questions posed in the evaluation were:

1. Is the Time-series Explorer capable of allowing the biologist to find previously unsuspected patterns of temporal activity?
2. Which of the patterns found can be revealed using other existing techniques, or combinations thereof, and what are the patterns that can only be found using the Time-series Explorer?
3. Of the patterns that can be found using other techniques what are the advantages, if any, of using the Time-series Explorer?
4. Are the patterns that can be found by the Time-series Explorer of sufficient significance to justify its use?

.

8.6.1 Procedure

The evaluation of the final prototype took place on the 6th of January 2005. This involved only a single experienced biologist who was *not* involved in the development of the technique analyzing familiar data from an experiment he himself had designed [*94*]. While results from a single biologist could not be statistically validated, it has been shown that domain experts will be significantly more motivated to find patterns in data of this type [*50*]. Moreover, it is only specific domain experts that are capable of assessing the relevant biological significance of the patterns found in microarray data or indeed finding any number of patterns with any substantial biological significance (this became clear after a few preliminary evaluation sessions

with domain novices). The participating biologist also had extensive experience of analyzing data using a range of established software tools such as Time-Searcher [*11*], Hierarchical Clustering Explorer [*12*] and GeneSpring [*114*]. As these tools implement a range of existing clustering and visual query techniques and have been evaluated as being most effective at revealing patterns from microarray time-course data [*50*], this qualified the biologist to properly assess the relative advantages, and disadvantages, of using the Time-series Explorer as an alternative.

The data used in this evaluation recorded the expression of around 8,500 genes over 17 time points belonging to 4 distinct stages of mouse development: virgin (days 10 and 12), pregnancy (days 1, 2, 3, 8.5, 12.5, 14.5 and 17.5), lactation (days 1, 3 and 7) and involution (days 1, 2, 3, 4 and 20) [*94*]. While the original data contained three replicates (separate runs of the experiment under identical conditions allowing results to be statistically verified), the data used in the evaluation combines these replicates (using the average recorded value for each gene at each time-point) to optimally exploit the existing functionality of the Time-series Explorer. This data was being used to investigate the immune involution of breast tissue and identification of genes involved in an immune response, with applications to the study of breast cancer in humans. This data contained over 144,500 values. This is a considerably larger amount of data than the 700 values contained in the data for the prior two evaluations and was considered by the biologists to be a moderately large data-set with regard to those generated by a typical microarray time-course experiments.

The evaluation proceeded in three stages, all completed in the same day. The first of these was a training session involving a short tutorial guiding the user through the basic functionality of the technique and allowing the biologist to become familiar with its interaction mechanisms and data representations. This was followed by a session where the user was asked to explore the data in order to find new patterns as per normal working procedures. Lasting approximately one hour, interactions were recorded and user was encouraged to think-aloud so that patterns revealed could be identified for further analysis in the third and final stage of the evaluation where he was asked to compare the results obtained using the Time-series Explorer with those obtained using other techniques.

8.6.2 Results

The first pattern found in the final prototype evaluation is illustrated in Figure 8.8. In order to find this pattern the biologist began by selecting the interval relating to the early stages of lactation (days 1 to 3) already knowing that certain predefined groupings of genes (and, specifically, the genes associated with the production of milk proteins) are likely to have a high level of expression over this period of time. The scatter-plot of this interval then allowed the biologist to observe an outlying group of genes with a high level of activity, use the labelling tool to identify them by name and, as their activity patterns were highlighted in the graph view, relate their high activity over this interval to their activity over the remainder of the time-course. This revealed that the genes with the pattern of activity specified did indeed belong to the expected groupings and that their activity over the remainder of the time-course was as predicted (low before lactation and falling after).

The second pattern found was a combination of general trends in activity for all genes over the entire time-course. Here the biologist selected an interval fixed at its minimum value (a *unit* interval, constrained by two time-points for which activity is recorded) and shifted it across the entire time frame of the experiment to animate the scatter-plot view. Selected frames of this animation are illustrated in Figure 8.9. At various stages of the animation the spread of gene representations in the scatter-plot became horizontally elongated. This occurred primarily during transitions between stages of development (i.e. virgin to pregnancy, pregnancy to lactation and lactation to involution) and indicated large numbers of genes with significant changes in their activity. The majority of these trends were unsurprising to the biologist as they reflected changes in the essential functioning of cells within the sample that would largely be detected by the observing the general activity patterns of groupings formed by clustering. Somewhat more interesting were the more subtle trends, such as the increased number of genes with changes in activity during pregnancy in relation to lactation. It was later verified that these particular trends would not be revealed by clustering.

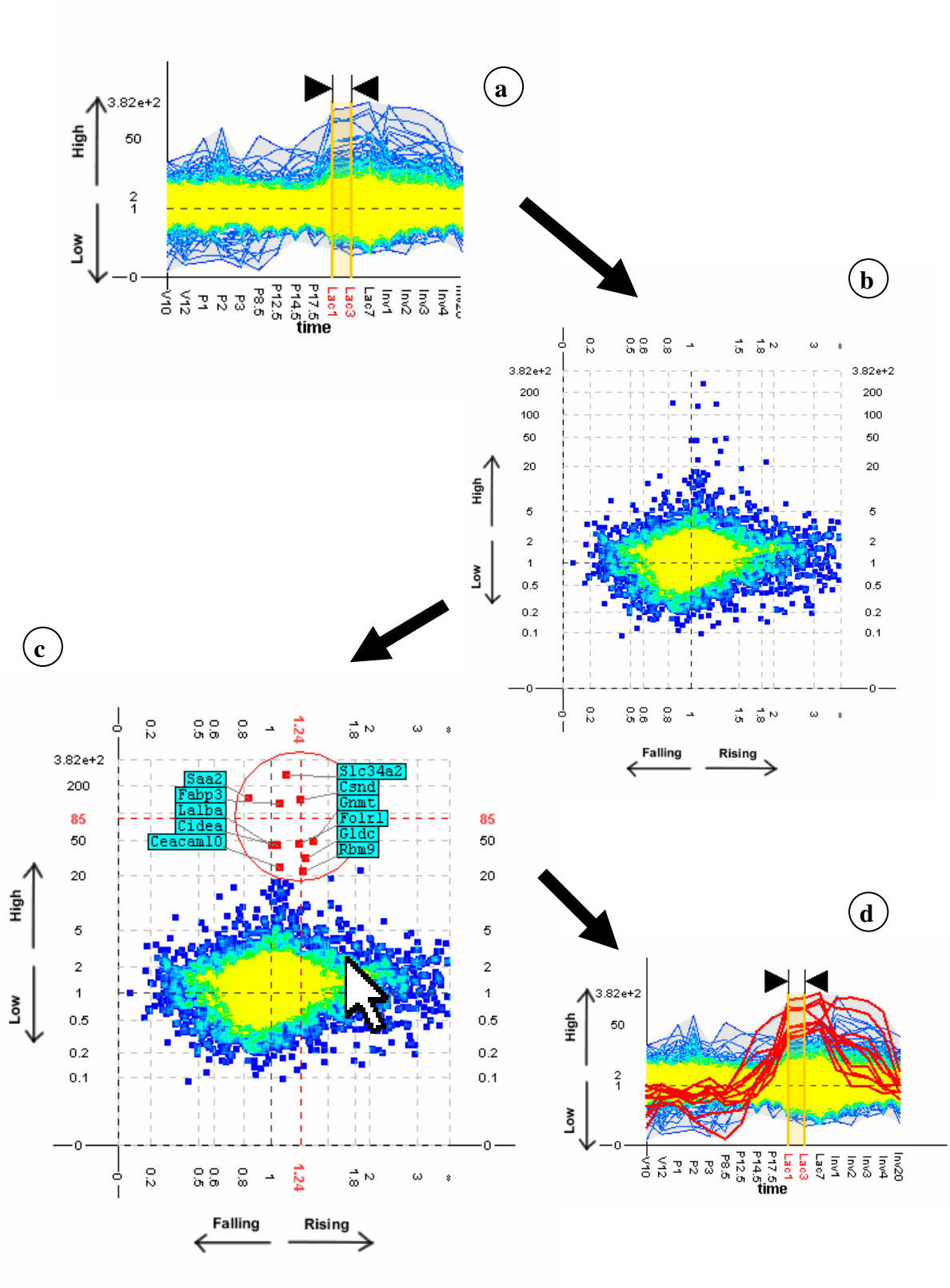

Figure 8.8. ***The identification of genes associated with the production of milk proteins with high activity at the early stages of lactation: a) Selecting relevant interval, b) observing an outlying group of genes with a high level of activity over this interval, c) labelling the gene symbols and d) highlighting the gene activity patterns for the remainder of the time-course.***

The next pattern found involved a combination of two distinct patterns of activity over intervals. Here the user wished to find which genes and groups of genes have rising activity at the start of lactation and falling activity at the end of lactation. The first stage in finding this pattern was to use the graph view to select the interval at the start of lactation. After this the scatter-plot was used to select all the genes with high rising activity over this interval. Next, the interval at the end of lactation was selected and the query was refined by selecting all the genes with falling activity over this interval. The results of the selections were then viewed over the entire time-course in the graph view. This revealed two groups of outliers that either had significant high or low activity over different periods of the pregnancy stage. This prompted the biologist, who was concerned that the activity of these genes would not adhere to the required profile, to investigate further by adjusting the selected interval so that the representations of these genes could be labelled and their activity patterns over the entire time-course highlighted. While the genes with low activity still appeared to fit the original profile, those with high activity did not and the original query was adjusted (by making the required level of activity at the start of lactation higher) to exclude them from the results. Finally the resultant gene listing was cross-referenced with existing predefined gene classifications in a pop-up window so that biological significance of the pattern could be properly assessed. This indicated a number of interesting genes and gene groupings which the biologist stored to be cross referenced with the results of another related experiment.

The final, and most significant, pattern found in the evaluation was discovered, in part, when investigating general trends across the entire time-period. As the scatter-plot animated through days 1 to 3 of the pregnancy stage (an interval for which there are three time-points for which expression is recorded) an outlying group of gene representations showed significant rising then falling expression. To investigate this further the relevant interval was animated again and then stopped so that the outlying genes could be labelled by moving the mouse over their representations in the scatter-plot. This revealed the majority of the genes also shared low activity over the remainder of the time-course. Next the genes were selected and cross-referenced with pre-defined gene classifications. Significantly the selection was found to contain

a high proportion of Keratin associated genes. Figure 8.10 illustrates this pattern showing selected frames of the original animation from interval P1 to P2 through to interval P2 to P3, the labelled scatter-plot at P1 to P2 and the effect of labelling in the coordinated graph view where the genes are highlighted.

The final stage of the evaluation was a follow up meeting where the biologist was asked to assess the degree of relevant biological significance to which each of the patterns revealed could be attributed and identify the extent to which the patterns could be uncovered using other techniques. These results were combined to produce a summary (Table 8.2) describing the specific areas and extent to which the utility of the technique contributes to the support of microarray time-course analysis. Here the patterns in the data were categorized, rated from one to ten according to the extent to which existing techniques are already capable of uncovering them and assigned a measure of biological significance. The measures of biological significance indicated were: high (biological significance relevant to the specific objectives of the experiment), medium (biological significance but not relevant) or low (not significant).

Table 8.2. Results of the user evaluation

Patterns found using the Time-series Explorer	**Type**	**Suspected**	**Can be found using alternative technique (marked out of 10)**		**Significance**
			Clustering	Visual queries	
Genes associated with Milk Proteins with very high expression from L1 to L3.	Outliers over an interval.	✔	0	9	Medium
Large changes in gene activity during known transitional phases.	General trend over the entire time-course.	✔	7	0	Medium
309 genes belonging to various interesting groupings with expression rising at L1 and falling L7.	General trend over an interval.	✔	6	0	High
Keratin associated genes with expression rising sharply at P2 and falling sharply at P3.	Outliers over an interval.	✘	0	0	High

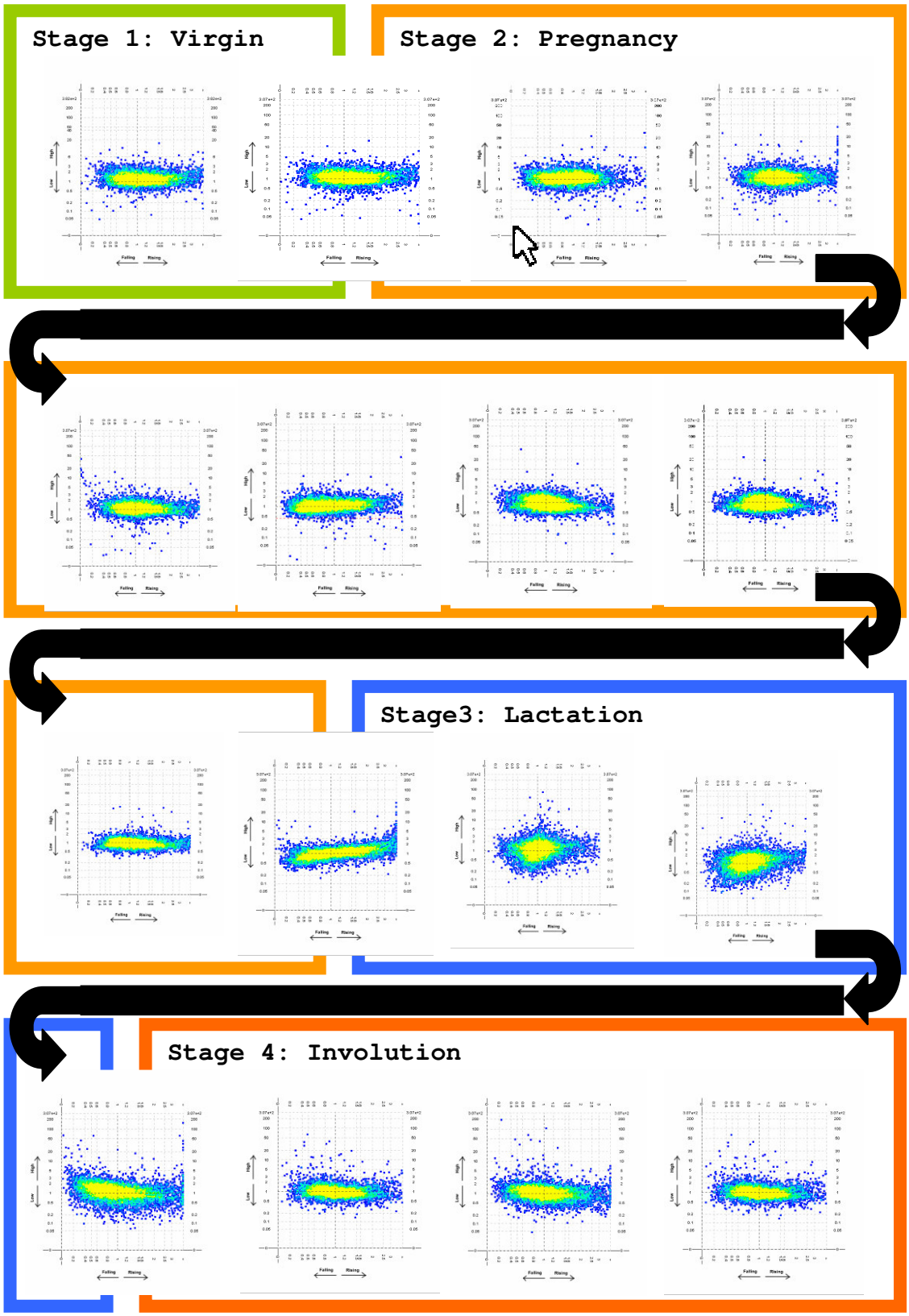

Figure 8.9. ***Animating the scatter-plot to reveal general trends in activity for all genes over the entire time-course: Static frames from the animated scatter-plot.***

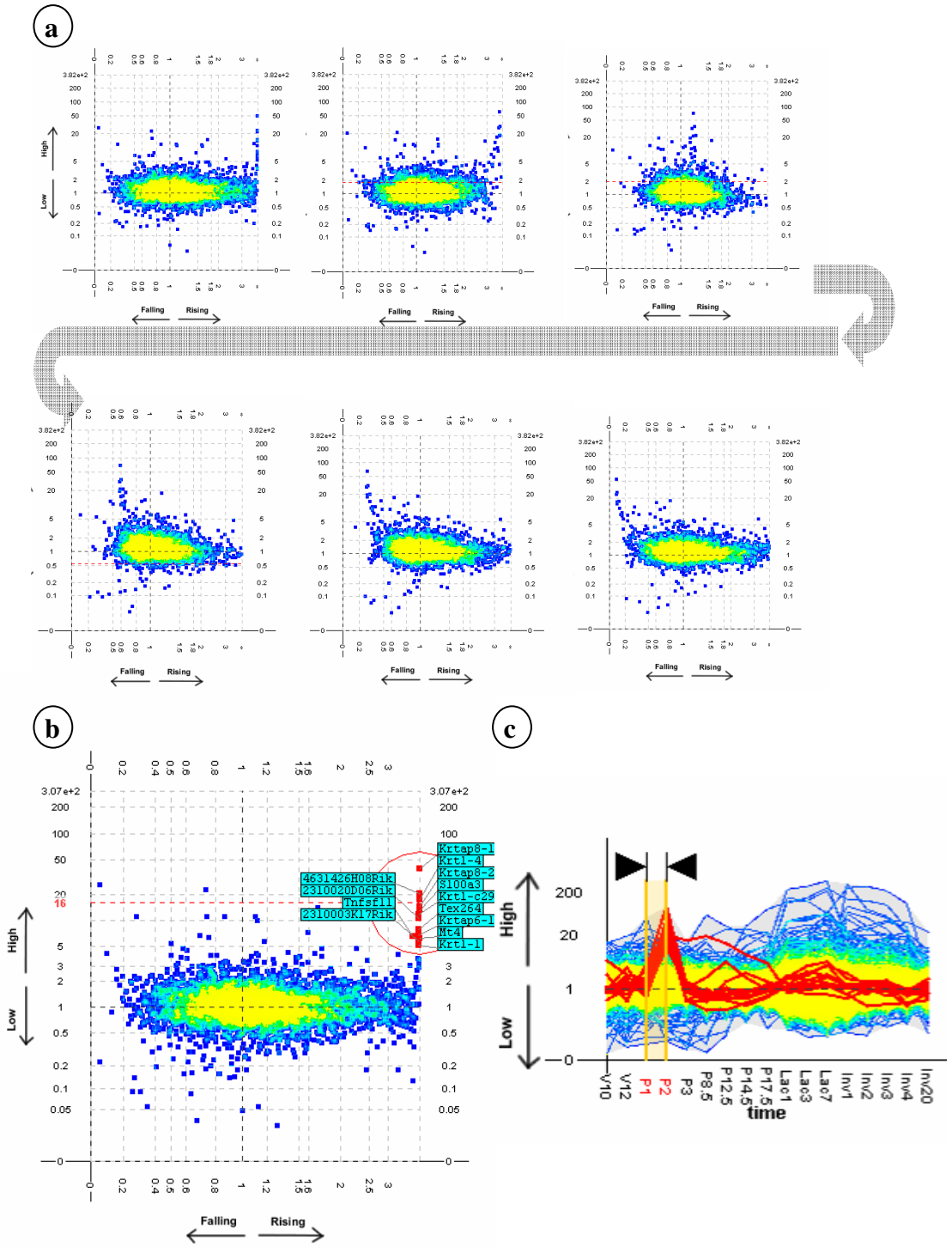

Figure 8.10. ***An unexpected pattern of temporal activity: a) Animating the scatter-plot reveals a group of outlying genes with rising then falling activity over a small interval of the time-course, b) moving the mouse over the gene representations in the scatter-plot view allows them to be labelled and c) have their activity patterns over the entire time-frame highlighted in the graph view.***

8.6.3 Conclusions

The main outcome of the user evaluation was to verify that the Time-series Explorer is uniquely capable of revealing previous unsuspected interval patterns, where a number of genes have similar activity over an interval of the data, and that the patterns found were of sufficient relevant biological significance to encourage a biologist to use the technique in the analysis of their data (positively answering questions 1, 2 and 4 posed in the evaluation). Moreover, the technique also proved capable of revealing suspected patterns of temporal activity and the evaluation uncovered significant advantages in using the Time-series Explorer over other more established techniques (positively answering questions 3 and 4). Specifically, when the technique was used to uncover general trends occurring over limited periods of the time-course the user had the advantage (over clustering techniques that would also allow biologists to find such patterns) of being able to quickly identify interesting sub-groupings, when identifying suspected outliers over smaller intervals the technique offered the biologists the ability to perceive distinct groupings of outliers and when looking for general trends across the entire time-course the biologists found it easier to assess more subtle patterns of general activity.

In general the biologist felt the tool to be more flexible than the other techniques used. This stemmed from the fact a large number of the valuable patterns in microarray data are combinations of patterns of temporal activity and that a biologist can use the technique to investigate different types of pattern without having to readjust between different tools and multiple unrelated representations of the data. Indeed, it was often found that when using the tool biologists would aim to find multiple unrelated types of pattern simultaneously. An example of this from the case study is when the biologist is assessing general trends for all genes over the entire time-course but also finds an interesting group of outliers over a smaller interval. Indeed, this was the discovery of an entirely unsuspected pattern which, when selected and cross-referenced with pre-defined gene classifications, proved of sufficient relevant biological significance for the biologist involved to be encouraged to investigate their finding further by cross-referencing the result with findings from other experiments.

Chapter 9 Conclusions

The main outcome of the research undertaken for this thesis was the development of a new technique using animated views to support the exploratory analysis of microarray time-course data. The main benefit of animation was that it allowed the display to be capable of conveying large amounts of information by utilising a third display dimension (i.e. time). The display also mapped time in the data to time in the display in a manner that ensured that motion could be related to meaningful qualities of the data. This allowed the users to detect and interpret a quantity of valuable patterns in their data and, specifically, find patterns of common activity over intervals of the data that could not be found using established techniques. Once a pattern was detected, the user could interact with a static frame of the animation and relate that view with complementary linked views, to read the pattern, in order to extract valuable knowledge.

It has already been established that animation can be used in scientific visualisation [*20, 21*] (i.e. for data with a special quality) or, in information visualisation, to smooth the translation between views [*17-19*] or draw a users attention [*14*]. This study builds on this work to demonstrate the ability of animations over time to convey large amounts of information and for the motion to carry sufficient meaning for valuable knowledge to be acquired. There are a number of conclusions that can be drawn from this thesis to help guide further research. The first of these relates to the appropriateness of an animated visualisation of this type with regard to data types and user objectives. In this study, factors that made the development of an animated technique appropriate were the large scale of the data, the large amounts of potentially valuable information that *could* be found in the data, a need to explore the data and the fact that the users could relate some form of motion to their own conceptualisation of the data. The benefits of animation for exploring large scale data are already well established for scientific visualisations [*20, 21*] (i.e. for data with a reference to physical space) but not for information visualisation. Such animation was, however, found to be useful and have largely the same benefits of a spatial animation in that it allowed users to exploit an extra display dimension to view more information. In this study, this was largely down to the animation matching the user's

conceptualisation of the data. Here, it was the fact that motion in the visualisation could have some meaning to the biologist that made an animated display particularly appropriate. This was primarily because the data had a temporal quality so that time in the data could be mapped to time in an animation, but also because the users had a concept of the activity of genes rising and falling which could be translated literally to a display where gene representations rise and fall.

An important aspect of the visualisation was that it utilised static views in order to help the user to become familiar with the animated view. These also allowed the user to read a pattern once it was found in the animation. Indeed, the animated scatter-plot would act itself as a static view once the animation was paused with the user rapidly switching attention between views as their relative utility changed with the user's immediate objectives. The utility of the scatter-plot as both an effective static and animated view was, in part, down to the axes used. For genes, translation along the y-axes represents activity and translation along the x-axes is approximate to the rate of change for that activity. During animation the y-axis communicates activity and the rate of motion along the y-axis indicates rate of change for activity (with the x-axis ensuring that items are dispersed about the display and their paths are consistently anticlockwise). When the animation is paused these two primary attributes of activity and change in activity remain visible for the interval selected. During the evaluation the users needed both types of view, static and animated, to find patterns in their data.

Another finding of the evaluation was the user's preference for direct control over the pace and direction of an animation. Here, the users would animate over intervals where they expected they might find a valuable pattern. Then, if a potentially interesting pattern was found, quickly rewind and animate the period again. Without direct control over the animation, the users would not absorb enough of the information presented and become frustrated waiting for the animation to end. In effect the animation from start to finish was used almost exclusively as a one-off overview. While direct manipulation is recognised as giving users a feeling of satisfaction and control over a representation, for an animation with large amounts of information it is imperative that the user has direct control over its pace and direction.

Another important finding of this research was that, in order for the technique to reach its optimal functionality, the animated display needed to be adjusted so that motion was smooth and predictable. It is already established that erratic and unpredictable motion can have a detrimental effect [*75*]. In this study it was found that in order to make an animation of non-spatial data effective, it is necessary to make the motion smooth by interpolation of values over time. Motion was made predictable by careful selection of axes (to make motion anticlockwise) and distortion of the display space. Without a natural spatial reference to configure the display it is likely that most animated information visualisations would require some similar form of smoothing and regulation so that the motion is palatable.

With regard to the findings presented in this thesis, a set of draft guidelines for developing animated visualisations to support the exploratory analysis of large-scale data can be prescribed:

1. Animated views should be configured so that the motion has relevant meaning.
2. The pace and direction of the animation should be controlled by the user.
3. The motion of objects should be smoothed and regulated to avoid the undesirable effects of erratic or unpredictable motion (e.g. interpolation across time and distortion of the display space).
4. Static views (coordinated and linked) should be available to help the user read a pattern once it is detected.
5. Static views (coordinated and linked) can be used to help the user interpret the animation.
6. The animated view can be used as a static view (see 4 and 5) when the animation is paused.

A general conclusion to be drawn from this thesis is that animation, where time in the data is mapped to time in the data, has enormous potential to be used to great effect in an information visualisation. While the effects of animation on human perception is less well understood than the effects of a static display, this should not discourage a developer from considering an animated visualisation if the benefits of expressiveness and an extra display dimension are required. Many of the problems

with the perception of animation encountered during this project were easy to detect during user evaluations and relatively easy to eliminate after they were detected. In this case, at least, the benefits of animation were found to outstrip its disadvantages with the technique developed using animation allowing biologists to find significant patterns they could not previously find using any other techniques. It is hoped that this thesis will guide other developers toward considering the use of animation in their own information visualisation applications to achieve similar results.

9.1 Further Work

There are two main types of opportunities for further research that could be based on the study presented in this thesis. First, the Time-series Explorer technique could be adapted to support the analysis of other similar types of data. These would be other types of large scale time-course data such as stock price data or economic data (e.g. economic indicators over time). In the first case the data is likely to have a finer granularity and include significant variations over a greater number of time points. In the second case, it is likely that some of the variables that change over time will have a more concrete association with each other. These types of data are, however, very similar to microarray time-course data in respect to the fact that large numbers of variables change over time and analysts of these data are likely to have similar requirements (e.g. associating common values and finding trends).

Second, further research could involve the further development of the technique to improve the analysis of microarray time-course data. Here the technique could be developed to accommodate the comparison of data from multiple experiments. Microarray experiments can be performed to determine time-course gene expression profiles under different experimental conditions. These experiments allow biologists to investigate how a process and associated gene expression is changed under different conditions, for example, disease state versus healthy or drug-treated versus untreated. Existing tools do not adequately provide the fundamental requirement to facilitate comparison between sets of time-series experiments performed under different experimental conditions. The ability to easily explore multiple complex data sets, and interpret the statistical significance of differences between data sets, presents a significant challenge. The representations could also be modified to account for, or represent, margins of uncertainty incurred by combining data from

replicate experiments or rescaling. None of the existing microarray visual analysis techniques exploit the measures of statistical confidence that are calculated in association with normalised expression values and if given access to this additional information, the biologists could use these measures to variably pre-filter (remove) data that is unreliable or insignificant, or to inform the data analysis process, for example, by verifying that final selected results are statistically valid.

Another such modification to improve the analysis of microarray time-course data would be to improve the integration of existing biological information (pre-defined groupings, pathway information etc). Extensive biological information resources are available through the Internet, which hold information classifying and annotating genes with respect to their families, functions, relationships, expression pathways, the proteins they are known to encode etc. and which can be used to sort and group sets of genes for analysis or to inform the interpretation of result sets of candidate genes of interest. If this metadata about genes could be integrated with the visual tools for exploring the microarray data it might be used as an additional query mechanism to guide the biologists' data exploration. For example, known temporal patterns of gene expression (e.g. during a given developmental process) could be compared with the patterns of expression observed in the microarray experiment.

Appendices

Appendix 1: A selection of applications developed to support the exploratory analysis of microarray data

Software	Company/ Academic Institution	Analysis techniques	Features
AMADA [*150*]	Hong Kong University	Hierarchical Clustering (HC), PCA	Interactively explore cluster display
Avadis	Strand Genomics	HC, K-means, PCA, SOM	
BRB ArrayTools [*128*]	National Cancer Institute Biometric Research Branch Division of Cancer Treatment and Diagnosis	HC, MDS	
Cleaver 1.0	Stanford Biomedical Institute	K-means, PCA	Web based interface
ChipST2C	Pearson Lab at Baylor College of Medicine	HC, K-means	
Cluster [*111*]	Michael Eisen's lab; Lawrence Berkeley National Lab	HC, K-means, PCA, SOM	Interactively explore cluster display
Engene [*153*]	Computer Architecture department, Univereidad de Malaga	HC, K-means, PCA, SOM	Web based interface.
Expression Profiler	European Bioinformatics Institute	HC	Web based interface.
Expressionist	GeneData	HC, PCA	
Gene Linker Gold	Molecular Mining Corporation	HC, K-means, PCA, SOM	
GeneMaths [*148, 155*]	Applied Maths	HC, PCA, SOM	Interactively explore cluster display
GeneSight	BioDiscovery	HC, PCA, SOM	
GeneSpring [*114*]	Silicon Genetics	HC, SOM, K-means, PCA, Graph view queries	
Genesis	Graz University	HC, K-means, PCA, SOM	
GEPAS [*152*]	National Spanish Cancer Centre	HC, SOM	Web based interface.
Hierarchical Cluster Explorer [*12*]	HCIL, University of Maryland	HC, K-means, Graph view queries	Interactively explore cluster display, Views linked
J-Express 2.1 [*117*]	MolMine	HC, K-means, PCA, SOM	Views linked
MAExplorer [*154*]	NCI Laboratory of Experimental and Computational Biology	HC, K-means	
Parteck Discoverer	Parteck	HC, PCA, MDS (of experiments)	
SpotFire Decision Site for Functional Genomics [*115*]	Spotfire	HC, K-means, PCA	Interactively explore cluster display, Views linked
Time Searcher [*10*]	HCIL, University of Maryland	Graph view queries	
TreeView [*149*]	Center for Computational Genomics and Bioinformatics, University of Minnesota	HCE	Interactively explore cluster display

Appendix 2: Image buffering for scatter-plot and graph views in the final version

The image buffering in the final prototype (Figure 9.1) ensures that when the user interacts with the interface only parts of the display that absolutely need to be redrawn are redrawn. Typical scenarios are as follows:

- **A view is resized:** This means that the axes gridlines need to be reconfigured to include the optimal values for that view size. Here the background image is redrawn, then the middle-ground image is redrawn using the background image and the resultant image is sent to the screen. The middle-ground and background images are then stored in a cache.
- **The scatter-plot is animated or the set of genes to be colour coded changes:** Here the background need not be redrawn only the middle-ground still needs to be redrawn. This image is drawn using the stored background image then sent to the screen. The new middle-ground image is then stored in a cache.
- **Genes are labelled:** Presuming the scatter-plot is not animating only the foreground image needs to be redrawn. This is drawn using the stored middle-ground image and is *not* stored to any cache.

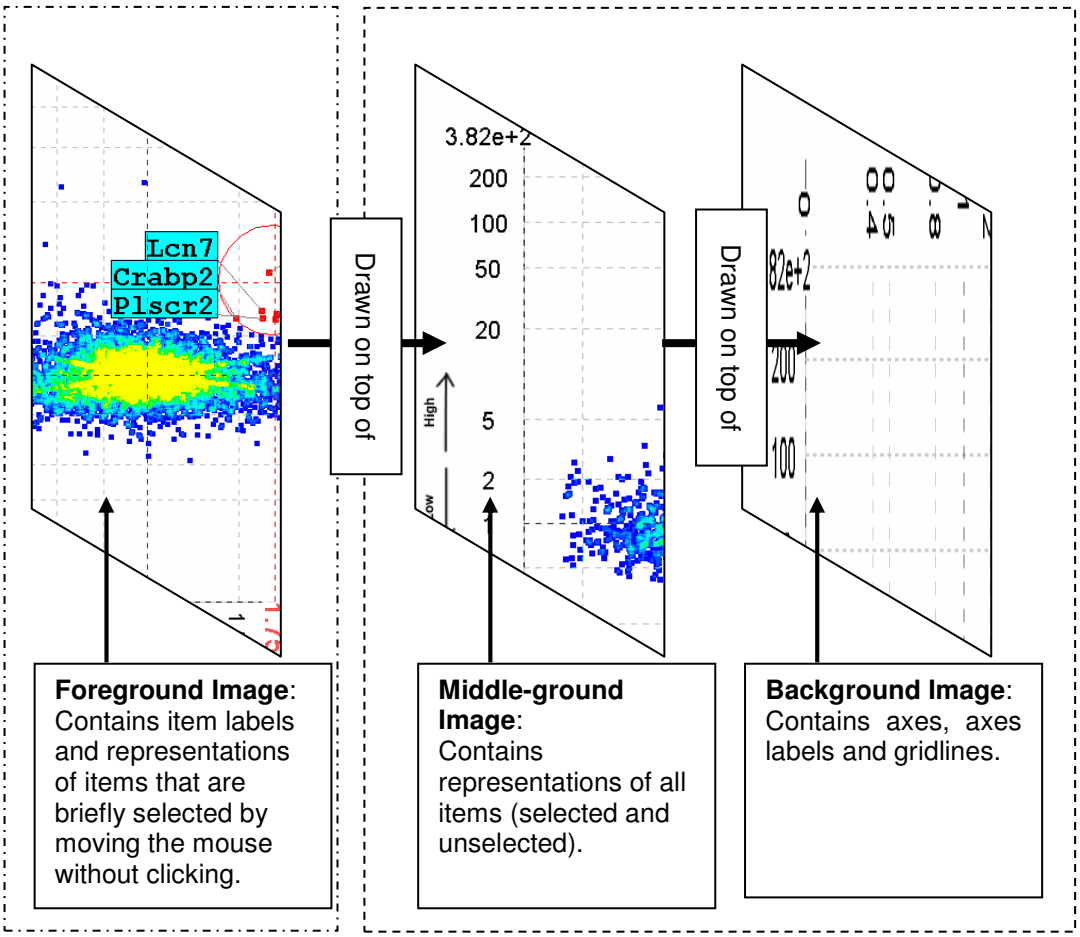

Figure 9.1. ***Different layers for image buffering***

References

1 *the Human Genome Project Website*. 2004.

2 Quackenbush J. Computational Analysis of Microarray Data*. Nature Reviews Genetics* 2001; **2**(6): 418- 427

3 Searls DB. Bioinformatics Tools for Whole Genomes*. Annu. Rev. Genomics Hum. Genet.* 2000; **1**: 251-279

4 Sherlock G. Analysis of large-scale gene expression data. *Brief Bioinform.* 2001; **2**(4): 350-62.

5 Brazam A and Vilo J. Gene expression data analysis*. FEBS letters* 2000; **480**(1): 17-24

6 Brown PO and Botstein D. Exploring the new world of the genome with DNA Microarrays*. Nature Genetics* 1999; **21**(1 Suppl): 33-37

7 Komorowski J, Hvidsten TR, Jenssen T-K, Tjeldvoll D, Hovig E, Sandvik AK, and Lægreid A. Towards Knowledge Discovery from cDNA Microarray Gene Expression Data. *Principles of Data Mining and Knowledge Discovery* 2000; 470-475.

8 Bassett DE, Eisen MB, and Boguski MS. Gene expression informations- it's all in your mine*. Nature Genetics* 1999; **21**(1 Suppl): 51- 55

9 Quackenbush J. Genomics. Microarrays--guilt by association. *Science* 2003; **302**(5643): 240-241

10 Hochheiser H, Shneiderman, B., *Visual Specification of Queries for Finding Patterns in Time-Series Data*, in *Proceedings of Discovery Science 2001*. 2001, University of Maryland, Computer Science Dept.

11 Hochheiser H and Shneiderman B. Dynamic query tools for time series data sets: Timebox widgets for interactive exploration*. Information Visualisation* 2004; **3**(1): 1-18

12 Seo J and Shneiderman B. Interactively Exploring Hierarchical Clustering Results*. IEEE Computer* 2002; **35**(7): 80-86

13 Bartram L. Can motion increase user interface bandwidth in complex systems? *Systems, Man, and Cybernetics* 1997 (Orlando, FL, USA); 1686-1692.

14 Bartram L, *Perceptual and Interpretative Properties of Motion for Information Visualization*. 1997, School of Computing Science, Simon Fraser University.

15 Wolfe JM, Klempen N, and Dahlen K. Postattentive vision*. Journal of Experimental Psychology: Human Perception & Performance* 2000; **26**(2): 693–716

16 Ware C and Franck G. Evaluating stereo and motion cues for visualising information nets in three dimensions*. ACM Transactions on Graphics* 1996; **15**(2): 121-140

17 Robertson GG, Mackinlay JD, and Card SK. Cone Trees: Animated 3D Visualizations of Hierarchical Information. *CHI '91 : Human Factors in Computing Systems* 1991 (New Orleans, Louisiana, USA), ACM Press; 189-194.

18 Fekete J and Plaisant C. Interactive Information Visualization of a Million Items. *IEEE Symposium on Information Visualization* 2002 (Boston, Massachusetts, USA), IEEE Computer Society; 117-126.

19 Yee K-P, Fisher D, Dhamija R, and Hearst M. Animated Exploration of Dynamic Graphs with Radial Layout. *IEEE Symposium on Information Visualization* 2001 (San Diego, California); 43-50.
20 Hansen HS. A time-series animation of urban growth in Copenhagen metropolitan area. *8th Scandinavian Research Conference on Geographical Information Science (ScanGIS)* 2001 (Ås, Norway); 225-235.
21 Droegemeier KK, Xue M, Johnson K, Mills K, and O'Keefe M. Experiences with the scalable-parallel ARPS cloud/mesoscale prediction model on massively parallel and workstation cluster architectures. *5th Workshop on the Use of Parallel Processors in Meteorology, European Center for Medium Range Weather Forecasts* 1992 (Reading, England).
22 Craig P, Kennedy JB, and Cumming A. Animated Interval Scatter-plot Views for the Exploratory Analysis of Large Scale Microarray Time-course Data*. Information Visualisation* 2005; **4**(3)
23 Craig P. Coordinated Parallel Views for the Exploratory Analysis of Microarray Time-course Data. *3rd International Conference on Coordinated & Multiple Views in Exploratory Visualization* 2005 (London), IEEE Computer Society; 3-14.
24 Card SK, Mackinlay JD, and Shneiderman B, eds. *Readings in Information Visualization: Using Vision to Think*. 1 ed. The Morgan Kaufmann Series in Interactive Technologies. Morgan Kaufmann: San Francisco, 1999; 686pp.
25 *Encarta World English Dictionary*. St Martin's Press (New York), 1999
26 Card SK, Robertson GG, and Mackinlay JD. The Information Visualizer, an Information Workspace. *ACM CHI '91* 1991 (New Orleans, Louisiana, USA), ACM Press; 181-188.
27 Tufte ER, *The Visual Display of Quantitative Information*. Graphics Press, 1983
28 Garland K, *Mr. Beck's Underground Map*. Capital Transport Publishing, 1994
29 Gershon N and Eick SG. Information Visualization*. IEEE Computer Graphics & Applications* 1997; **17**(4): 29-31
30 Bertin J, *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press.: Madison, WI, 1983
31 Andrews DF. Plots of High-Dimensional Data*. Biometrics* 1972(28): 125-136
32 Eick SG, Steffen JL, and Sumner EE, Jr. Seesoft - A Tool For Visualizing Line Oriented Software Statistics*. IEEE Transactions on Software Engineering* 1992; **18**(11): 957-968
33 Ahlberg C and Shneiderman B. Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. *ACM Conference on Human Factors in Software, CHI '94* 1994 (Boston, MA, USA), ACM Press; 313-317.
34 Feiner S and Beshers C. Worlds within Worlds: Metaphors for Exploring n-Dimensional Virtual Worlds. *ACM UIST '90* 1990 (Snowbird, Utah, USA), ACM Press; 76-83.
35 Carpendale MST, *Considering Visual Variables as a Basis for Information Visualisation*. 2003, The University of Calgary: Calgary.

36 Shneiderman B. The eyes have it: A task by data type taxonomy for information visualizations. *Proceedings IEEE Visual Languages* 1996: 336-343

37 Furnas GW. Generalized Fisheye Views. *ACM CHI '86* 1986 (Boston, Massachussetts, USA), ACM Press; 16-23.

38 Shneiderman B, *Designing the User Interface*. Addison Wesley Longman: Reading, MA, 1998

39 Graham M, Kennedy JB, and Benyon D. Towards a methodology for developing visualisations. *International Journal of Human-Computer Studies* 2000; **53**(5): 789-807

40 Ellis JB, Rose A, and Plaisant C. Putting visualization to work: ProgramFinder for youth placement. *CHI '97: Conference proceedings on Human Factors in Computing Systems* 1997 (Atlanta, Georgia, USA), ACM Press; 502-509.

41 Glaser DC, Tan R, Canny J, and Do EY-L. Developing Architectural Lighting Representations. *IEEE Symposium on Information Visualization* 2003; 241-248.

42 Plaisant C, Rose A, Milash B, Widoff S, and Shneiderman B. LifeLines: Visualizing Personal Histories. *ACM CHI'96 Conference* 1996; 221-227.

43 Rose A, Ellis J, Plaisant C, and Greene S, *Life Cycle of User Interface Techniques: The DJJ Information System Design Process.* 1996, University of Maryland HCIL: Maryland MD.

44 Freitas CMDS, Luzzardi PRG, Cava RA, Winckler MAA, Pimenta MS, and Nedel LP. Evaluating Usability of Information Visualization Techniques. *IHC2002, V Simposio sobre Fatores Humanos em Sistemas Computacionais* 2003 (Fortaleza, Brazil).

45 Hix D and Hartson HR, *Developing User Interfaces : Ensuring Usability Through Product & Process*. Professional Computing. Wiley, 1993

46 Chen C and Czerwinski M. Empirical Evaluation of Information Visualizations: An Introduction. *International Journal of Human-Computer Studies* 2000; **53**: 631-635

47 Chen C and Yu Y. Empirical studies of information visualization: a meta-analysis. *International Journal of Human-Computer Studies* 2000; **53**(5): 851-866

48 Mackinlay JD. Automating the design of graphical presentations of relational information. *ACM Transaction on graphics* 1986; **5**(2): 110--141

49 Frøkjær E, Hertzum M, and Hornbæk K. Measuring usability: are effectiveness, efficiency, and satisfaction really correlated? *CHI 2000 conference on Human factors in computing systems* 200 (The Hague, The Netherlands,); 345-353.

50 Saraiya P, North C, and Duca K. An Evaluation of Microarray Visualization Tools for Biological Insight. *IEEE Symposium on Information Visualization (INFOVIS'04)* 2004 (Austin Texas); 1-8.

51 Wise JA, Thomas JJ, Pen-nock K, Lantrip D, Pottier M, and Schur A. Visualizing the non-visual: Spatial analysis and interaction with information from text documents. *Information Visualization Symposium* 1995, IEEE Computer Society Press; 51-58.

52 Shneiderman B. Direct Manipulation for Comprehensible, Predictable and Controllable User Interfaces. *International Conference on Intelligent User Interfaces* 1997 (Orlando, Florida); 33-39.
53 Mullet K and Sano D, *Designing Visual Interfaces - communication oriented techniques*. Sun Press, 1995
54 Norman DA, *Emotion and design: Atrractive things work better*, in *Interactions Magazine*. 2002. p. 36-42.
55 Chang D, Dooley L, and Tuovinen JE. Gestalt theory in visual screen design: a new look at an old subject. *Seventh world conference on computers in education conference on Computers in education: Australian topics* 2002 (Copenhagen, Denmark); 5-12.
56 Graham M and Kennedy J. Using Curves to Enhance Parallel Coordinate Visualisations. *Information Visualisation* 2003 (London, UK), IEEE Computer Society Press; 16-18.
57 Mackinlay J. Applying a theory of graphical presentation to the graphic design of user interfaces. *Symposium on User Interface Software and Technology* 1988 (Alberta, Canada); 179 - 189.
58 Healey CG, Booth KS, and Enns JT. High-speed Visual Estimation using Pre-attentive processing. *ACM Transactions on Human-Computer Interaction* 1996; **3**(2): 107-135
59 Healey CG, *Effective Visualization of Large Multidimensional Datasets*. 1996, The University of British Columbia: Vancouver, B.C.
60 Treisman A and Gormican S. Feature analysis in early vision: Evidence from search asymmetries. *Psychological Review* 1988(95): 15-48
61 Nowell L, Hetzler E, and Tanasse T. Change Blindness in Information Visualization: A Case Study. *IEEE Symposium on Information Visualization 2001* 2001; 15.
62 Healey CG and Enns JT. Perception and Painting: A Search for Effective, Engaging Visualizations. *IEEE Computer Graphics and Applications*; **22**(2): 10 - 15
63 Chalmers M, Ingram R, and Pfranger C. Adding Imageability Features to Information Displays. *(ACM) Symposium on User Interface Software and Technology* 1996; 33-39.
64 Shneiderman B. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. *IEEE Visual Languages '96* 1996 (Boulder, Colorado, USA), IEEE Computer Society Press; 336-343.
65 Roberts JC. Multiple-View and Multiform Visualization. *Visual Data Exploration and Analysis VII, Proceedings of SPIE* 2000; 176-185.
66 Ahlberg C and Shneiderman B, *The Alphaslider: A Compact and Rapid Selector*. 1998, University of Maryland: Maryland.
67 Martin AR and Ward MO. High Dimensional Brushing for Interactive Exploration of Multivariate Data. *6th conference on Visualization* 1995; 271.
68 Eick SG. Data Visualization Sliders. *UIST '94* 1994 (Marina del Ray, California, USA), ACM Press; 119-120.
69 Fekete J and Plaisant C. Excentric Labeling: Dynamic Neighborhood Labeling for Data Visualization. *Conference on Human Factors in Computing Systems* 1999 (Pittsburgh, Pennsylvania, United States), ACM Press New York, NY, USA; 512 - 519.

70 Buja A, McDonald JA, Michalak J, and Stuetzle W. Interactive Data Visualization Using Focusing and Linking. *IEEE Visualization '91* 1991 (San Diego, California, USA), IEEE Computer Society Press,; 156-163.

71 Roberts JC. Issues of Dataflow and View Presentation in Multiple View Visualization. *CISST Annual Conference, Workshop: Fundamental Issues of Visualization* 2001 (Las Vegas, NV); 177-183.

72 Wertheimer M, *Experimental studies on the seeing of motion*. Classics in psychology, ed. Shipley T. Philosophical Library: New York, 1961pp.(Original published in 1912 as Experimentalle Studien über das Sehen von Bewegung. Zeitschrift für Psychologie, 61, 161-265.)

73 Card SK, Moran TP, and Newell A, *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates: Hillsdale, NJ, 1983

74 Ware C, *Information Visualization: Perception for Design*. The Morgan Kaufmann Series in Interactive Technologies. Morgan Kaufmann, 2000; 580

75 Ware C, Bonner J, Knight W, and Cater R. Moving Icons as a Human Interrupt. *International Journal of Human-Computer Interaction* 1992; **4**(4): 341-348

76 Bryant B, Milosavljevic A, and Somogyi R. Gene Expression and Genetic Networks (Session Introduction). *Pacific Symposium on Biocomputing* 1998; **3**: 3-5

77 *scada.jpeg*. 2005, Transfield Worley (NZ) Ltd. (www.transfieldworley.co.nz).

78 Holmes N, *Explanation Graphics (http://www.nigelholmes.com/)*. 2005.

79 *http://news.bbc.co.uk/*. 2005, BBC News.

80 Wijk JJv and Technische WAAN. Smooth and efficient zooming and panning. *INFOVIS'03* 2003 (Seattle, WA); 15 - 22.

81 Bederso BB, Meyer J, and Good L. Jazz: an extensible zoomable user interface graphics toolkit in Java. *User Interface and Software Technology* 2000, ACM Press.

82 Andrews K. Visualising Cyberspace: Information Visualisation in the Harmony Internet Browser. *Information Visualization* 1995 (Los Alamitos, CA); 97-104.

83 David K. Modjeska. View vs. Overview - Visualizing Hierarchical Data in Desktop Virtual Reality. *NordiCHI 2000* 2000 (Stockholm, Sweden).

84 Hanson AJ and Zhang H. Multimodal Exploration of the Fourth Dimension. *IEEE Visualization* 2005 (Minneapolis, MN, USA), IEEE Computer Society; 34.

85 Mackinlay JD, Robertson GG, and Card SK. The perspective wall: detail and context smoothly integrated. *ACM CHI '91* 1991 (New Orleans, Louisiana, USA), ACM Press; 173-176.

86 Robertson GG, Card SK, and Mackinlay JD. Information Visualization using 3D interactive animation. *Communications of the ACM* 1993; **36**(4): 57-71

87 Johnson B and Shneiderman B. Treemaps: A Space-Filling approach to the visualization of hierarchical information structures. *IEEE Visualization '91* 1991 (San Diego, California, USA), IEEE Computer Society Press; 284-291.

88 Nakakoji K, Takashima A, and Yamamoto Y. Cognitive effects of animated visualization in exploratory visual data analysis. *Information Visualisation 2001* 2001 (Los Alamos, CA), IEEE Computer Society; 77- 84.

89 Daniel G and Chen M. Video visualization. *IEEE Visualization 2003* 2003 (Seattle, WA); 409-416.

90 Yamamoto Y, Nakakoji K, and Takashima A. The Landscape of Time-based Visual Presentation Primitives for Richer Video Experience. *Human-Computer Interaction - INTERACT 2005* 2005 (Rome, Italy); 795-808.

91 Schena M, Shalon D, Davis RW, and Brown PO. Quantitative monitoring of gene-expression patterns with a complementary-DNA microarray. *Science* 1995; **270**(5235): 467-470

92 Schena M, Shalon D, Heller R, Chai A, Brown PO, and Davis RW. Parallel human genome analysis: microarray-based expression monitoring of 1000 genes. *Proc. Natl. Acad. Sci. U.S.A. 93,* 1996; **93**(20): 10614-10619

93 Duggan DJ, Chen M, Meltzer P, and Trent J. Expression profiling using cDNA microarrays. *Nature Genetics* 1999; **21**: 10-14

94 Stein T, Morris J, Davies C, Weber-Hall S, Duffy M-A, Heath V, Bell A, Ferrier R, Sandilands G, and Gusterson B. Involution of the mouse mammary gland is associated with an immune cascade and an acute-phase response, involving LBP, CD14 and STAT3. *Breast Cancer Research* 2004; **6**(2): R75 - R91

95 Wen X, Fuhrman S, Michaels GS, Carr DB, Smith S, Barker JL, and Somogyi R. Large-Scale Temporal Gene Expression Mapping of CNS Development. *Proc Natl Acad Sci USA* 1998; **95**(1): 334-339

96 Hughes TR, Marton MJ, Jones AR, Roberts CJ, Stoughton R, Armour CD, Bennett HA, Coffey E, Dai H, and He YD. Functional discovery via a compendium of expression profiles. *Cell* 2002; **102**(1): 109- 126

97 Pennisi E. A Low Number Wins the GeneSweep Pool. *Science* 2003; **300**: 1484

98 Stein LD. Human genome: End of the beginning. *Nature* 2004; **431**: 915-916

99 Goffeau A, Barrell BG, Bussey H, Davis RW, Dujon B, Feldmann H, Galibert F, Hoheisel JD, Jacq C, Johnston M, Louis EJ, Mewes HW, Murakami Y, Philippsen P, Tettelin H, and Oliver SG. Life with 6000 Genes. *Science* 1996; **274**: 563-567

100 Deloukas P, Schuler GD, Gyapay G, Beasley EM, and Soderlund C. A physical map of 3,000 human genes. *Science* 1998; **282**: 744-746

101 Gibson U, Heid CA, and Williams PM. A novel method for real-time quantitative RT-PCR. *Genome Res.* 1996(6): 995-1001

102 Heid CA, Stevens J, Livak K, and Williams PM. Real time quantitative PCR. *Genome Res.* 1996; **6**: 986-994

103 Higuchi R, Fockler C, Dollinger G, and Watson R. Kinetic PCR: Real time monitoring of DNA amplification reactions. *Biotechnology* 1993(11): 1026-1030

104 Lander E. Array of Hope. *Nature Genetics* 1999; **21**: 23-32

105 Bowtell DD. Options available--from start to finish--for obtaining expression data by microarray. *Nature Genetics* 1999; **21**(2): 25-32

106 Brazma A, Hingamp P, Quackenbush J, Spellman PT, Stoeckert C, Aach J, Ball CA, Causton HC, Gaasterland T, Glenisson P, Holstege FCP, Kim IF, Markowitz V, Matese JC, Parkinson H, Robinson AJ, Sarkans U, Schulze-Kremer S, Stewart J, Taylor R, Vilo J, and Vingron M. Minimum information about a microarray experiment (MIAME) toward standards for microarray data. *Nature Genetics* 2001; **29**(4): 365-371

107 Tao Y, Friedman C, and Lussier Y. Information Visualization Techniques in Bioinformatics during the Postgenomic Era. *Biosilico* 2004; **2**(6): 237-245

108 Leung YF CD. Fundamentals of cDNA microarray data analysis. *Trends Genet* 2003; **19**(11): 649-59

109 Weisstein EW, *"Geometric Centroid." From MathWorld--A Wolfram Web Resource.* http://mathworld.wolfram.com/GeometricCentroid.html. 2005.

110 Nadon R and Shoemaker J. Statistical issues with microarrays: processing and analysis. *TRENDS in Genetics* 2002(18): 265-271

111 Eisen MB, Spellman PT, Brown PO, and Bostein D. Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci. USA* 1998; **95**(25): 14863-14868

112 Tamayo P, Slonim D, Mesirov J, Zhu Q, Kitareewan S, Dmitrovsky E, Lander E, and Golub T. Interpreting patterns of gene expression with self-organizing maps. *Proceedings of the National Academy of Sciences of the United States of America* 1999; **96**(6): 2907-2912

113 Raychaudhuri S, Stuart JM, and Altman RB. Principal Components Analysis to Summarize Microarray Experiments: Application to Sporulation Time Series. *Pacific Symposium on Biocomputing* 2000; 452-463.

114 *GeneSpring*. 2004, Silicon Genetics www.silicongenetics.com.

115 *Spotfire Decisionsite for Functional Genomics*. 2002, Spotfire.

116 Kendall M, *Multivariate Analysis*. Charles Griffin&Co, 1975

117 Dysvik B and Jonassen I. J-Express: exploring gene expression data using Java. *Bioinformatics* 2001; **17**(4): 369-370

118 Butte AJ and Kohane IS. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Proceedings of the Pacific Symposium on Biocomputing* 2000: 418-429

119 Alter O, Brown PO, and Botstein D. Singular value decomposition for genome-wide expression data processing and modeling. *PNAS* 2000; **97**(18): 10101–10106

120 Hoffman P, Grinstein G, Marx K, Grosse I, and Stanley E. DNA visual and analytic data mining. *8th conference on Visualization '97* 1997 (Phoenix, Arizona, United States); 437 - ff.

121 Ankerst M, Keim DA, and Kriegel HP. Circle Segments: A Technique for Visually Exploring Large Multidimensional Data Sets. *Visualization '96* 1996 (San Francisco, CA); 279.

122 Mramor M, Leban G, Demsar J, and Zupan B. Conquering the Curse of Dimensionality in Gene Expression Cancer Diagnosis: Tough Problem, Simple Models. *Artificial Intelligence in Medicine* 2005 (Aberdeen, UK); 514-523.

123 Sharan R, Maron-Katz A, and Shamir R. CLICK and EXPANDER: a system for clustering and visualizing gene expression data. *Bioinformatics* 2003; **19**(14): 1787-1799

124 Luo J, Duggan DJ, Chen Y, Jurga Sauvageot, Ewing CM, Bittner ML, Trent JM, and Isaacs WB. Human Prostate Cancer and Benign Prostatic Hyperplasia: Molecular Dissection by Gene Expression Profiling. *Cancer Research* 2001(61): 4683-4688

125 Bittner M, X PM, X YC, Jiang Y, Seftor E, Hendrix M, Radmacher M, Simon R, k ZY, Ben-Dork A, k NS, Dougherty E, I EW, I FM, Gooden C, Lueders J, Glatfelter A, Pollock P, Carpten J, Gillanders E, Leja D, Dietrich K, Beaudry C, Berens M, Alberts D, Sondak V, Hayward N, and Trent J. Molecular Classification of Cutaneous Malignant Melanoma by Gene Expression Profiling. *Nature* 2000; **406**(6795): 536-540

126 Jazaeri AA, Yee CJ, Sotiriou C, and al. e. Gene expression profiles of BRCA1-linked, BRCA2-linked, and sporadic ovarian cancers. *J Natl Cancer Inst.* 2002(94): 990-1000

127 D'haeseleer P, Wen X, Fuhrman S, and Somogyi R. Mining the Gene Expression Matrix: Inferring Gene Relationships from Large Scale Gene Expression Data. *Information Processing in Cells and Tissues* 1998, Plenum Publishing; 203-212.

128 Simon R and Lam AP, *BRB ArrayTools*. 2005, National Cancer Institute Biometric Research Branch Division of Cancer Treatment and Diagnosis (linus.nci.nih.gov).

129 Taroncher-Oldenburg G, Griner EM, Francis CA, and Ward BB. Oligonucleotide Microarray for the Study of Functional Gene Diversity in the Nitrogen Cycle in the Environment. *Applied and Environmental Microbiology* 2003; **69**(2): 1159-1171

130 Baud S, Vaultier M-N, and Rochat C. Structure and expression profile of the sucrose synthase multigene family in Arabidopsis. *Cell and Molecular Biology, Biochemistry and Molecular Physiology* 2003; **55**(396): 397-409

131 D'Haeseleer P, Wen X, Fuhrman S, and Somogyi R. Linear Modeling Of mRNA Expression Levels During CNS Development And Injury. *Pacific Symposium on Biocomputing* 1999: 41-52

132 Sturn A, Quackenbush J, and Trajanoski Z. Genesis: cluster analysis of microarray data. *Bioinformatics* 2000; **18**(1): 207-208

133 Gath I and Gev AB. Unsupervised Optimal Fuzzy Clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1989; **11**(7): 773 - 780

134 Gasch AP and Eisen MB. Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. *Genome Biology* 2002; **3**(11): research0059.1–research0059.22.

135 Kohonen T. The Self-Organizing Map. *Proceedings of the IEEE* 1990; **78**(9): 1464-1480

136 Ultsch A and Siemon HP. Kohonen's self organizing feature maps for exploratory data analysis. *INNC'90, Int. Neural Network Conf* 1990 (Dordrecht, Netherlands).

137 Jain JMMAKaAK. A nonlinear projection method based on kohonen's topology preserving maps. *IEEE Transactions on Neural Networks* 1995; **6**(3): 548-559

138 Flexer A. Limitations of Self-Organizing Maps for Vector Quantization and Multidimensional Scaling, In: al. MMCe (Ed). *Advances in Neural Information Processing Systems 9*. MIT Press/Bradford Books. 1997. 445-451.
139 Kaski S, Nikkilä J, Oja M, Venna J, Törönen P, and Castrén E. Trustworthiness and metrics in visualizing similarity of gene expression. *BMC Bioinformatics* 2003; **4**(1): 48
140 Eichler GS, Huang S, and Ingber DE. Gene Expression Dynamics Inspector (GEDI): for integrative analysis of expression profiles. *Bioinformatics* 2003; **19**(17): 2321-2322
141 Whitfield ML, Sherlock G, Saldanha AJ, Murray JI, Ball CA, Alexander KE, Matese JC, Perou CM, Hurt MM, Brown PO, and Botstein D. Identification of Genes Periodically Expressed in the Human Cell Cycle and Their Expression in Tumors. *MBC* 2002; **13**(6): 1977-2000
142 Hastie T, Tibshirani R, Eisen MB, Alizadeh A, Levy R, Staudt L, Chan WC, Botstein D, and Brown P. Gene shaving' as a method for identifying distinct sets of genes with similar expression patterns. *Genome Biology* 2000; **1**(2): research0003.1-0003.21
143 Lazzeroni L and Owen A, *Plaid Models for Gene Expression Data*. 2000, Stanford University.
144 Wang J, Delabie J, Aasheim HC, Smeland E, and Myklebost O. Clustering of the SOM easily reveals distinct gene expression patterns: results of a reanalysis of lymphoma study. *BMC Bioinformatics* 2002; **3**(36)
145 Jain AK and Dubes RC, *Algorithms for clustering data*. Englewood Cliffs, N.J., 1988
146 Gibbons FD and Roth FP. Judging the Quality of Gene Expression-Based Clustering Methods Using Gene Annotation. *Genome Research* 2002; **12**(10): 1574-1581
147 Costa IG, Carvalho FdATd, and Souto MCPd. Comparative analysis of clustering methods for gene expression time course data. *Genet. Mol. Biol.* 2004; **27**(4): 623-631
148 *GeneMaths*, Applied Maths http://www.applied-maths.com.
149 Saldanha AJ. Java Treeview—extensible visualization of microarray data. *Bioinformatics* 2004; **20**(17): 3246-3248
150 Xia X. AMADA: analysis of microarray data. *Bioinformatics* 2001; **17**(6): 569-570
151 Shneiderman B and Seo J. Interactively Exploring Hierarchical Clustering Results. *IEEE Computer 35* 2002; **7**: 80-86
152 Herrero J, Al-Shahrour F, Díaz-Uriarte R, Mateos Á, Vaquerizas JM, Santoyo J, and Dopazo J. GEPAS: a web-based resource for microarray gene expression data analysis. *Nucleic Acids Research* 2003; **31**(13): 3461-3467
153 García de la Nava J, Santaella DF, Cuenca Alba J, María Carazo J, Trelles O, and Pascual-Montano A. Engene: the processing and exploratory analysis of gene expression data. *Bioinformatics* 2003; **19**(5): 657-8
154 Lemkin PF, Thornwall G, and Hennighausen L. MicroArray Explorer - A Java-based Tool For Data Mining Microarrays. *AMS-IMS-SIAM Summer Conference on Statistics in Functional Genomics* 2001.
155 *GeneMath (http://www.applied-maths.com/genemaths)*. 2004.

156 Inselberg A. The plane with parallel coordinates. *The Visual Computer* 1985(1): 69--91
157 Inselberg A and Dimsdale B. Parallel Coordinates: A Tool for Visualizing Multidimensional Geometry. *IEEE Visualization 1990* 1990 (San Francisco, California, USA), IEEE Computer Society Press; 361-378.
158 Siirtola H, *Parallel Coordinate Explorer. Java 1.1 Applet at* http://www.cs.uta.fi/~hs/pce/. 2000.
159 Hauser H, Ledermann F, and Doleisch H. Angular Brushing of Extended Parallel Coordinates. *IEEE Symposium on Information Visualization (InfoVis'02)* 2002 (Boston, Massachusetts, USA).
160 Fisher RA. The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics* 1936(7): 179 - 188
161 Voigt R, *An Extended Scatterplot Matrix and Case Studies in Information Visualization*. 2002, Hochschule Magdeburg-Stendal.
162 Ware C, *Information Visualization*. Morgan Kaufmann, 2000
163 Siegel RM and Andersen RA. Perception of three-dimensional structure from two-dimensional motion in monkey and man. *Nature* 1988; **331**: 259-261
164 Seo J and Shneiderman B. Understanding Hierarchical Clustering Results by Interactive Exploration of Dendrograms: A Case Study with Genomic Microarray Data. *IEEE Computer Special Issue on Bioinformatics* 2002; **35**(7): 80-86
165 Rosson MB and Carroll JM, *Usability engineering: scenario-based development of human computer interaction*. Morgan Kaufmann: Redwood City, CA, 2001
166 Nielsen J. Guerrila HCI: Using Discount Usability Engineering to Penetrate the Intimidation Barrier, In: Mayhew DJ (Ed). *Cost-Justifying Usability*. Academic Press Professional. 1994. Chapter 11.
167 Wagner EK, Ramirez JJ, Stingley SW, Aguilar SA, Buehler L, Devi-Rao GB, and Ghazal P. Practical approaches to long oligonucleotide-based DNA microarray: lessons from herpesviruses. *Prog Nucleic Acid Res Mol Biol* 2002(71): 445-491
168 Bederson BB and Boltman A. Does Animation Help Users Build Mental Maps of Spatial Information? *InfoVis '99* 1999 (San Francisco, California, USA), IEEE Computer Society Press; 28-35.
169 Chen H. Compound Brushing. *IEEE Symposium on Information Visualization* 2003 (Seattle); 181 - 188.
170 Wills GJ. Selection: 524, 288 Ways to say 'This is Interesting'. *Information Visualization* 1996 (San Francisco, California); 54-61.
171 Shneiderman B. Dynamic Queries for Visual Information Seeking. *IEEE Software* 1994; **11**(6): 70 -77
172 Weisstein EW, *"Statistical Median." From MathWorld--A Wolfram Web Resource.* http://mathworld.wolfram.com/StatisticalMedian.html. 2005.
173 Colby G and Scholl L. Transparency and blur as selective cues for complex visual information. *SPIE Image Handling and Reproduction Systems Integration* 1991 (San Jose, California, USA), SPIE; 114-125.
174 *Sun Microsystems: Java Swing (http://java.sun.com)*. 2005.