

Coordinated Graph and Scatter-Plot Views for the Visual Exploration of Microarray Time-Series Data

Paul Craig
School of Computing, Napier University
p.craig@napier.ac.uk

Jessie Kennedy
School of Computing, Napier University
j.kennedy@napier.ac.uk

Abstract

Microarrays are a relatively new, high-throughput data acquisition technology for investigating biological phenomena at the micro-level. One of the more common procedures for microarray experimentation is that of the microarray time-course experiment. The product of microarray time-course experiments is time-series data, which subject to proper analysis has the potential to have significant impact on the diagnosis, treatment, and prevention of diseases. While existing information visualization techniques go some way to making microarray time-series data more manageable, requirements analysis has revealed significant limitations. The main finding was that users were unable to uncover and quantify common changes in value over a specified time-period. This paper describes a novel technique that provides this functionality by allowing the user to visually formulate and modify measurable queries with separate time-period and condition components. These visual queries are supported by the combination of a traditional value against time graph representation of the data with a complementary scatter-plot representation of a specified time-period. The multiple views of the visualization are coordinated so that the user can formulate and modify queries with rapid reversible display of query results in the traditional value against time graph format.

CR Categories: H.5.2. [Information Storage and Retrieval]: User Interfaces -- Screen design ; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval -- Query formulation

Keywords: Bioinformatics, Microarrays, Information Visualization, Time-Series, Multiple-views

1 Introduction

Time-series can be described as "...a sequence of observations ordered by a time parameter". These observations are typically measured at regular discrete intervals [Yaffee 2000] and constitute an observed realization of an underlying event, or combination of events. It is often the case that a large collection of related time-series will combine to constitute an observed realization of an underlying system. Examples of such time-series collections are numerous, and include stock market data, economic data, medical data and microarray data.

The specific data we have been working with is that produced by microarray time-course experiments. Microarrays are a relatively new, high-throughput data acquisition technology for investigating biological phenomena at the micro-level. The product of microarray time-course experiments is time-series data where the origin of time-series are genes and the recorded value is gene-expression. Individual time-series are often referred to as expression patterns or expression profiles. While microarray technology is credited for producing massive amounts of data, which have the potential to have significant impact on the diagnosis, treatment, and prevention of diseases [Debouck and Goodfellow 1999; Jagota 2001; Noordewier and Warren 2001; Schena et al. 1996; Somogyi 1999], the analysis and handling of the data is becoming a major bottleneck in the utilization of the technology towards its full potential [Brazam and Vilo 2000; D'Haeseleer et al. 1999].

The two main approaches to the visualization of microarray data are clustering techniques that group time-series according to some predefined similarity measure and visual querying methods that allow the user to interactively explore the data through the formulation of visual queries. Common pre-processes to the visualization of microarray data are normalization and rescaling. Normalization [Duggan et al.1999] is a statistical preprocess that seeks to account for deficiencies in the experimentation, which may obscure the underlying expression levels. Rescaling is an optional preprocess which transforms individual time-series to improve inter-time-series comparisons. Another, less common, preprocess to the visualization of microarray data is the interpolation of missing values.

The most common of approach to the visualization of microarray time-series is clustering. The rationale behind clustering is obvious. Clustering together similar elements allows the data analyst to generalize across groups of elements rather than having to consider individual elements. For this reason, clustering can be seen as making the data more manageable and lessening the problems associated with the massive scale of the data (microarray experiments typically record the expression of around 6,000 genes).

The first stage of clustering is the creation of a similarity matrix. In the similarity matrix both rows and columns correspond to the full list of genes - cells reflect inter-time-series similarities calculated according to some predefined similarity measure [Quackenbush 2001]. The direct visualization of a similarity matrix is a grid with the brightness of cells representing degrees of similarity. Other methods of visualizing the similarity matrix, such as self organizing maps (SOM) and multidimensional scaling (MDS), represent genes as individual points with the distance between representations approximate to inter-expression-pattern similarities. The most common method of clustering is based on the result of hierarchical agglomerative clustering [Eisen et al.

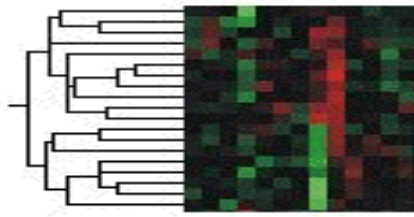


Figure 1: Clustering by color mosaic/ dendrogram display.

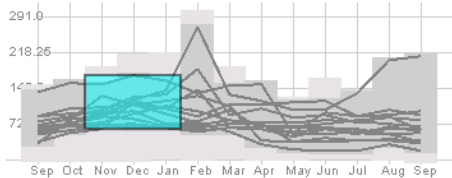


Figure 2: Visual querying (TimeSearcher).

1998; Seo and Shneiderman 2002] which uses the similarity matrix to produce a type of binary tree known as a dendrogram. In the most common visualization of a dendrogram (shown in Figure 1) the end points of outer branches represent genes with the similarity groupings represented by common branches in the tree structure. In this display gene expression patterns are color-coded and stacked vertically beside their respective gene nodes. This part of the display is known as a color mosaic. An alternative method of visualizing the results of hierarchical agglomerative clustering is presented by Fua et al. [1999]. In this visualization a variation on parallel coordinates is used to convey aggregation information for clusters. The display is multiresolutional allowing users to navigate the dendrogram structure until the desired focus region and level of detail is reached. While this visualization is suggested as a method for displaying multidimensional data, it is also suitable for time series which data can be considered as a specialized subtype of multidimensional data where the ordering of dimensions (time points) is a fundamental quality of the data.

While clustering is useful for revealing certain natural groupings and providing a global view of the data, it is also highly subjective and the significance of what one sees is not quantified [Jagota 2001]. Moreover, individual time-series may contain multiple interesting features that cannot all be accounted for by the single similarity measure on which a clustering visualization must ultimately rely. The abstraction of the data, based on similarity, leads to the loss of important information. For example, if a biologically significant feature occurs exclusively over a limited period of the overall time frame its representation may be diluted by the representation of less significant features occurring over the remainder of the time frame.

The primary alternative to the clustering of microarray time-series data is visual querying. Existing visual query tools for time-series present an overview of the data as overlaid value versus time graphs (Figure 2). Visual queries are formulated by clicking and dragging to draw boxes on top of the overview to specify an acceptable range of values over a given period of time. Successive querying filters the data, causing the initial overview to be replaced by overlaid value against time graphs of a filtered subset. The TimeSearcher technique [Hochheiser and Shneiderman 2001] extends this basic model by allowing the user to modify queries, by clicking and dragging edges or corners of visual query representations while viewing a rapid reversible update of query

results (Figure 2). The extended parallel coordinate plots of Hauser et al. [2002] offer similar functionality to that of TimeSearcher with the most notable extension of allowing queries specifying a permitted range of gradients between two adjacent time-points.

The main advantage of visual querying over clustering is that queries are measurable and it is easy for the user to quantify their results. An additional advantage is that queries can be confined to a specified period of time. This allows for the identification of biologically significant features that occur exclusively over a limited period of the overall time frame. The disadvantage of existing visual querying techniques is that the user will generally have a less effective overview of the data than that offered by clustering. The overlaid value against time graph representation of time-series is unable to accommodate large numbers of dissimilar time-series without individual time-series being illegible due to crossing lines. This problem is illustrated in Figure 2, with a relatively small number of overlaid time-series. It can be seen that this representation is only effective in revealing extreme values, and the extent of outlying values, at individual time-points.

Commercial microarray visualization tools often combine clustering and visual querying views of the data in multiple coordinated windows. This allows the user to benefit from the overview provided by clustering while having the ability to perform measurable queries using visual querying views. However, these tools were unable to satisfy all the requirements of our test users who were a typical group of biologists and bioinformaticians working toward the analysis of microarray time-series data. This paper describes the development of a novel visualization technique, which addresses these particular visual exploration and querying requirements.

2 Evaluation of existing systems

Initial user requirement analysis [Craig et al. 2002] indicated that existing systems did not allow users the desired functionality for microarray time-series analysis. In order to properly understand the limitations of existing systems a more detailed requirements analysis and evaluation were undertaken. This process guided us in the development of the system described in this paper.

The personnel involved in requirements analysis had a wide range of abilities; some were expert biologists who were unfamiliar with computing concepts, others were practicing bioinformaticians who were more familiar with computing concepts but less familiar with the particular goals of the biologists. The existing processes of microarray time-series analysis often involved a high level of cooperation between these two groups. The biologists would design and carry-out experiments while the bioinformaticians would apply 'informatics' techniques (derived from disciplines such as applied maths, computer science, and statistics) to organize and assist in the understanding of the information produced. Often the groups of users were not distinct, with biologists employing an increasing amount of bioinformatics and bioinformaticians becoming more tuned to the biology behind the experiments. For this reason it was important that the tool developed would be accessible to users unfamiliar with computer concepts as well as supporting the high level information seeking processes required for proper analysis of the data.

The main product of requirements analysis was a list of high-level tasks that a microarray time-series tool should facilitate. These are summarized as follows:

1. Show which genes are switched-on or switched-off over a specified period of the overall time frame.
2. Show which genes have rising or falling expression over a specified time-period.
3. Allow for the identification of similar expression patterns.
4. Relate features revealed to existing functional classifications.
5. Allow for the export of results.

Considering each of the items in our list of requirements in detail allowed us to gain a better understanding of what was required from the visualization tool.

2.1 Requirement 1: switched-on or switched-off?

The first thing to notice about requirement 1 is the ambiguity of the terms switched-on and switched-off. Consultation with the biologists revealed that these terms related to the level of gene expression. As different genes have different sensitivities, the level of expression at which a gene may be considered switched-on or switched-off will be different. A common approach to determining whether a gene is switched-on or switched-off at a time-point is to rescale individual time-series according to their maximum value [Heath and Ramakrishnan 2002; Quackenbush 2001]. As the sensitivity of a gene is often related to the maximum value of its time-series, this form of rescaling should give an indication of activity that accommodates for variable sensitivities. This form of rescaling is known as percent to maximum rescaling because rescaled time-series values are effectively percentages of the maximum value at any point in the time-series. A major advantage of this rescaling is that it makes values comparable across time-series. The problem with this approach is that, if the expression of a gene is low throughout the time-frame, rescaling will have the effect of amplifying noise or insignificant variations in expression making the gene appear to be switched on at certain time-points when it is in fact switched off at all time points. In order to assess whether a gene is switched on or switched off over a period of time, it is necessary to have both a view of rescaled value and absolute value over that period of time.

2.2 Requirement 2: falling or rising?

Requirement 2 deals with falling and rising expression. In order to calculate the change in value over a period of time it is necessary to subtract the initial value from the final value. This can be expressed as:

$$\Delta v = v(t_f) - v(t_o), \text{ where } t_f > t_o$$

where $v(t_o)$ is the initial value, $v(t_f)$ is the final value and Δv is the change in value. If the user is to be able to assess the significance of change values it is again necessary to rescale the time-series. Calculating change according to percentage relative to maximum rescaled values gives percentage relative to maximum change. While this method of calculating change will make rise or fall in value comparable across time-series, it will not allow a rise to be comparable with a fall. In order that rise or fall in value are comparable it is necessary to consider log fold-change which makes an x factor change the converse equivalent of a $1/x$ factor change [Heath and Ramakrishnan 2002]. In order to calculate log fold change it is necessary to rescale the values according to the transform:

$$LS(t) = \log_2 [V(t)/\text{mean}(V)]$$

where V is the original time-series and LS is the rescaled time-series. This rescaling is known as log scaling [Jagota 2001]. In microarray data analysis log-scaled values are often used as an

indication of the degree to which a gene is turned on or off but the major advantage of log-scaling microarray time-series is to relate changes in expression. A useful feature of log-scaling is that the change between log-scaled values equates to the log fold change between values. As with percent to maximum rescaling, large-scaling has the disadvantage that, if the expression of a gene is low throughout the time-frame rescaling will have the effect of amplifying noise or insignificant variations in expression. Therefore, to properly assess changing expression over a period of time, it is necessary to have both a view of log-scaled value and absolute value over that period of time.

2.3 Requirement 3: Similarity

Requirement 3 states that the user should be able to find similar expression patterns. Having already stated that the features of interest are expression level and change in expression level over periods of time, it is logical that a measure of similarity should be able to reflect one or more of these features. Some descriptions of similarity measures were given by the biologists. These included things like “expression rising at the start then falling away” and “expression rising later on and staying high”. In order that the biologist can find similar expression patterns, it is necessary to allow for the grouping of time-series according to multiple value and change features over multiple time-periods.

2.4 Requirement 4: existing functional classes

The fourth requirement of the biologists was that existing functional classifications should be reflected in the visualization. Further investigation revealed that it is often the case that the most interesting features of the data will run across functional classifications and that considering functional classes separately is not an option.

2.5 Requirement 5: export of queries and results

The final requirement of the biologists was that the visualization technique should allow for the export of queries and query results so that they can be reused and shared between collaborators. This requirement infers that the user should be able to extract sub-collections of the data that are the result of measurable queries and that query parameters match the goals of the analysis.

2.6 Limitations of existing techniques

In order to assess the limitations of existing techniques we considered the extent to which they fulfilled our requirements analysis.

Clustering techniques satisfy item 3 of our requirements analysis by grouping time-series according to similarity. However, due to the abstraction of the data based on similarity measures, these techniques are unable to adequately reveal features that exist over a limited period of the overall time frame and therefore cannot satisfy requirements 1 or 2. Moreover, the subjective nature of the displays produced makes it inappropriate for results to be exported and reused so clustering techniques cannot be seen to satisfy requirement 5. Requirement 4 can be satisfied by clustering techniques if an indication of existing functional classifications is overlaid into the display.

Existing techniques that allow for visual querying of time-series data allow for the grouping of time-series according to a specified acceptable range of values over a specified period of time

[Hochheiser and Shneiderman, 2001]. This allows the user to find out when genes are switched on or off as stated in requirement 1. If the user combines queries that specify different value ranges over successive periods of time the visualization will reflect change in value over a period of time. However, while queries that reflect change in value over a period of time can be quantified according to the different value ranges and time-periods specified they are not quantifiable according to the actual degree of change. While extended parallel coordinates [Hauser et al. 2002] allow queries specifying an allowed gradient between two adjacent dimensions, querying gradient between non adjacent dimensions requires a reordering of dimensions that is inappropriate for time-series data where the ordering of dimensions is a fundamental quality. Requirements 2 and 5 are only partially satisfied by visual query techniques. Requirement 3 is also only partially satisfied – while the user is able to find similar time-series where similarity is based on multiple value and change conditions over subsequent time-periods, change cannot be quantified in the desired manner for non-adjacent time-points. Requirement 4 cannot be satisfied by existing visual query techniques by overlaying an indication of existing functional classifications because individual time-series are not distinct in the visualization due to the problem of crossing lines. The alternative method of filtering to examine individual functional classifications would make it difficult to identify features running across classifications.

Requirements 1 and 2 both state the need to view value and change in value for both original and rescaled time-series. In the analysis of microarray time-series data rescaling is often seen as a process that precedes visualization. While some tools incorporate rescaling none give the option of switching between rescaled versions of the data during the analysis process. The need to refer to both the original and rescaled data during the analysis process is highlighted in our requirements analysis.

While clustering techniques provide an overview of microarray time-series data that may be useful for revealing natural groupings, techniques that employ visual query mechanisms allow the user to constrain result sets more precisely and quantify their results. As our users had a requirement to quantify their results, so that queries and results could be shared and reused, the visual query approach was preferred. The major limitation of existing visual querying techniques is that queries can only specify an acceptable range of values over a period of time and not an acceptable range of change in value over an extended period of time. If we summarize the existing visual queries available as:

$$Qv_{min} \leq v \leq Qv_{max}, \text{ where } Qt_{min} \leq t \leq Qt_{max}$$

where v is the time-series value at time t and Qv is a specified value range over time-period Qt . The queries that the user requires to perform are:

$$Qv_{min} \leq v \leq Qv_{max}, \text{ where } Qt_{min} \leq t \leq Qt_{max}$$

$$Q\Delta v_{min} \leq (v(t_{max}) - v(t_{min})) \leq Q\Delta v_{max}$$

where $Q\Delta v$ is an allowable change in value. Accounting for the fact that the user also requires rescaled views of the data, and that queries based on such rescaled views will also be required, gives us two additional types of query to consider:

$$QPM_{min} \leq PM \leq QPM_{max}, \text{ where } Qt_{min} \leq t \leq Qt_{max}$$

$$QALS_{min} \leq (LS(t_{max}) - LS(t_{min})) \leq QALS_{max}$$

where QPM is a specified percentage max rescaled value range, PM is percentage max value, $QALS$ is an allowable change in log-scaled value (change in log-scaled value \equiv fold change) and LS is log-scaled value. Each of these query types can be broken down into a condition component (Qv , $Q\Delta v$, QPM and $QALS$), a time context (Qt) and a rescaling (none, percentage to max or log-scale). This breakdown of query types is presented in Table 1.

Query	Conditions	Context	Rescaling
Value	Qv_{min}, Qv_{max}	Qt_{min}, Qt_{max}	None
Change	$Q\Delta v_{min}, Q\Delta v_{max}$	Qt_{min}, Qt_{max}	None
Percentage to max value	QPM_{min}, QPM_{max}	Qt_{min}, Qt_{max}	Percentage to max
Log fold change	$QALS_{min}, QALS_{max}$	Qt_{min}, Qt_{max}	Log-scaling

Table 1. Queries that require to be supported according to requirement analysis.

The only types of query that are supported in existing visual querying tools are value queries or, if the data is rescaled beforehand, percentage to max value queries. Clustering tools support none of the required queries. In order to support all required queries it was necessary to develop a new visualization technique.

3 Our approach

Given that our users wish to investigate, study, or analyze their data with the outcome of one stage of analysis often leading to the formulation of another, it is easy to classify our tool as an exploratory system. The design strategy prescribed by Shneiderman [1998] for such systems is: *designers should pursue the goal of having the computer vanish as users become completely absorbed in their task domain*. This approach is particularly appropriate in our case where a significant proportion of our users are computer novices who are highly knowledgeable in the task domain. Shneiderman continues by stating that this goal is most effectively met with a direct manipulation representation of the data.

Given the exploratory nature of microarray time-series analysis and the fact that our users' initial idea of a query could not necessarily be quantified according to any specific parameters it was felt preferable that our visualization not only support direct manipulation but also visual queries where visual query components are directly overlaid onto a visualization of the data. Examining the queries that required support in order to satisfy our users' requirements (listed in Table 1) allowed us to extract parameters that needed to be represented measurably in our visualization for visual querying. These are listed, along with functional dependencies, in Table 2

Parameter	Functional dependencies
Time (t)	none
Value (v)	$v = F(t)$
Change (Δv)	$\Delta v = F(t_1, t_2)$, where $t_1 < t_2$
Percentage max rescaled value (PM)	$PM = F(t)$
Fold change (ΔLS)	$\Delta LS = F(t_1, t_2)$, where $t_1 < t_2$

Table 2. Parameters to be represented measurably in the visualization in order to support visual querying.

It is useful to note that while value and percentage max rescaled value can be derived from a single value of time, change and fold-change are derived for a given time-period requiring two values of time; the start and end of the time-period.

After deciding on the parameters that we needed to represent measurably in our visualization, we employed Shneidermans [1996] task by data type taxonomy of information visualizations to categorize the types of data we needed to visualize. Our time-

series data can be most accurately classified as temporal data. However, as we are to support visual querying of change and fold-change, which are properties of the data that are derived for a given time-period, we can also consider the data to have a two-dimensional aspect with dimensions Δv and ΔLS for each time-period. For time-series with n observations there are 2^n possible time-periods. This makes it be infeasible to display all change and fold-changes for all possible time-periods for all time-series due to the massive amounts of data that would require to be displayed. In order to present measurable views of all parameters it is necessary to employ multiple views; an overview that presents the qualities of the data dependent on a single value of time and a detail view that presents qualities of the data dependent on a time-period. While the overview displays the temporal aspect of the data the detail view displays derived two-dimensional subsets for selected time-periods.

The interface of our microarray time-series visualization tool is illustrated in Figure 3. The tool comprises of several components, a graph overview for time-period specification, a scatter-plot detail view of a specified time-period, a textual list of gene names, a set of range sliders for query composition, a combination query panel and a panel for revealing predefined groups. Each of these will be considered in more detail along with the effects of coordination to realize the user requirements.

3.1 Overview

The logical first stage in designing a visualization tool is to consider the overview. Overviews are commonly used to give the user a ‘feel’ for the entire data set allowing them to assess the general spread of the data. Overviews can also be used to provide a visual representation of the context for additional detail views. It makes sense that our initial overview should deal with the actual data rather than derived qualities presented to aid data manipulation. In this case the data we wish to visualize can be classified as temporal data

The two main strategies for the visualization of temporal data are static and animated representations. Static representations, of which there are many examples [Plaisant et al. 1996; Kumar et al. 1998], present time as a fixed axis of the overall display.

Animations allow the user to observe all given attributes of the data at any given value of time while time is increased at a steady rate. Animations of temporal data are somewhat less common than static representations due to the fact that users find it hard to compare between concurrent frames. This problem makes the use of animation most inappropriate for providing an overview as, in order to assess the spread of the data, the user may need to compare qualities of the data at all time-points. As discussed earlier the abstraction of the data necessary for clustering results in an inability to explore the data in a temporal manner. Therefore we have adopted the value against time graph representation of time-series data, which is an example of the standard static display where time is represented as a fixed axes and is the most familiar representation of this type of data.

The overlaid value against time graph representation of multiple time-series is a measurable display and can be used in the formulation of certain visual queries. The value and time axes of the graph display support the specification of times and values. This allows for the specification of allowable value ranges over periods of time in existing visual exploration techniques. The time axis of the value against time representation can also allow users to specify a time-period context for change and fold change queries.

Given that all the users’ queries (listed in Table 1) have a time-period context, the specification of a time-period would be the primary operation associated with the value against time graph display. It is also apparent, as value queries are not the only queries that require the specification of a time-period, that the users may wish to specify a time-period without specifying an acceptable range of values. For this reason it was decided to restrict interaction with the value against time graph display to the specification of a time-period so that this operation would be more efficient for all queries. This functionality was realized by the introduction of a time-slider. The design of the time-slider component is based on Eick’s [1994] data visualization sliders where the internal space of a range-slider [Ahlberg and Shneiderman 199] is used to display some aspect of the data. In our case the internal space of the slider is used to display a value against time graph representation of our time-series data.

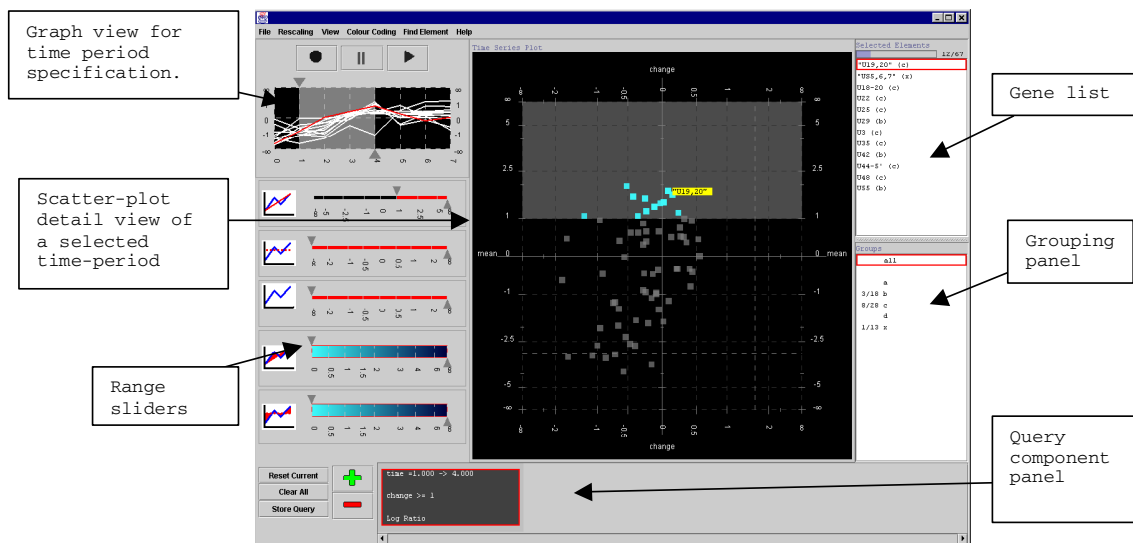


Figure 3: Interface of the microarray time-series visualization tool.

A minor alteration to the basic range slider design is that rather than allowing both thumbs to move independently, movement in the t_{\min} thumb will cause the t_{\max} thumb to move so that the actual time-period changes but the length of the time-period remains constant. Only movement of the t_{\max} thumb can alter the length of the time-period. This accommodates for the fact that the user is more likely to be interested in comparing time-periods of equal duration. Figure 4 shows the data overview provided by the time-slider while Figure 5 shows the slider being used to specify a time-period.

Once the time-slider has been used to specify a time-period, standard multi-range sliders can be used to specify acceptable ranges of value and change in value. As with existing visual query tools, the initial overview is replaced with the filtered subset after query formulation and visual representations of value queries are overlaid so that users can view the effect of their queries in a familiar format. Given that the result of queries will be a subset of the original data the problem of crossing lines in this display may be reduced. At the very least the overlaid time-series will have a period over which they are similar and it will be easy to recognize the general pattern over the period of similarity. The value query representation, overlaid on the corresponding filtered result set, is illustrated in Figure 6. In order to query percentage to max value and fold change, the appropriate rescaling mode is selected via a menu bar option and the data is rescaled accordingly. This approach matched the users' expectations that they would be able to switch between multiple rescalings to verify their results and accommodated for the fact that having multiple rescalings presented at the same time may confuse users who were used to rescaling as a preprocess of visualization. If however, further user testing reveals this process of switching between rescaling modes to be cumbersome it is conceivable that multiple rescalings will be incorporated into the views of later prototypes.

Combining the time-slider with multi-range query parameter sliders allows the users to perform all the queries they require with query results displayed as value against time graphs.

A significant limitation of this combination is that the visualization cannot support visual queries of change in value so, unless the user's initial idea of a change query is quantified according to change itself the formulation of such a query will involve a great deal of trial and error. To compound this problem, individual time-series and groups of time-series are not visually distinct (due to crossing lines) which makes it hard to directly assess the effects of any queries. The introduction of a suitable detail view to complement the value against time overview goes some way to solving these problems.

3.2 Detail view

While the traditional value against time graph representation of time-series is capable of measurably presenting features of the data dependent on a single time, (value and percentage max rescaled value) there is also the need for an additional view of the data that presents features of the data dependent on a time-period. These features are change and fold-change. These attributes are represented in a detail view as a scatter-plot as shown in Figure 7 with the y axis being either change or fold change depending on the rescaling mode and the x axis being the mean value. The mean value is calculated for each individual gene over the specified time-period and should not be confused with the mean value of all genes over the time-period. While a measurable display of change in expression allows the user to query according to rising

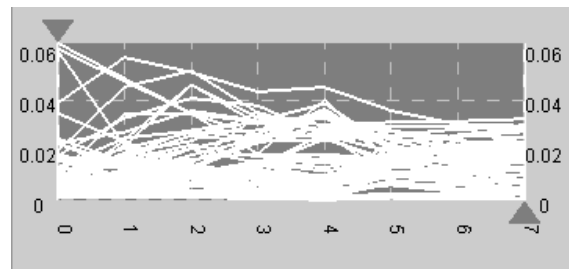


Figure 4: Data overview provided by the graph view time-slider.

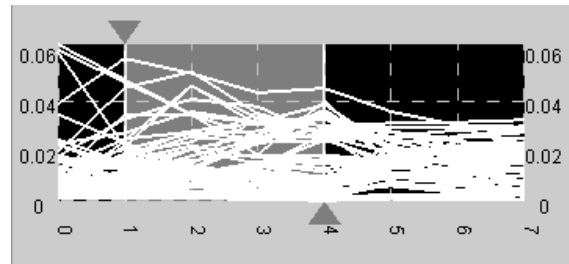


Figure 5: Graph view time-slider used to specify a time-period.

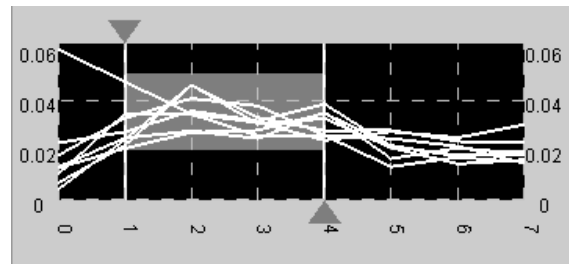


Figure 6: Graph view time-slider used to specify a value range query.

or falling expression, it is also conceivable that a measurable indication of mean value will allow the user to query as to whether or not a group of genes are switched on or off.

Another advantage of the scatter-plot display of a selected time-period is that the compact single point representation of genes allows for the overlay of functional classifications by color-coding as shown in Figure 7. If the number of functional classifications exceeds the number of distinguishable colors, or the user has trouble distinguishing between colors, color coding can be switched off allowing the user to highlight different groupings by clicking on their name in the grouping panel. In color-coding mode this panel acts as a key to indicate which colors correspond to which functional classifications.

Change queries and mean-value queries are performed by dragging a box which defines an allowable range of mean value and change in value. Once a query is formulated clicking and dragging the edges or corners of the visual representation can modify the query. Query results are highlighted with a view of all the data (for the selected time-period) perpetually displayed so that the user can modify their query with the same overview that was used in its initial formulation (see Figure 8). The query

shown in Figure 8 selects a group of genes with high yet falling expression.

3.3 Coordination of views

If the scatter-plot is used to visualize the entire observed time-period it can be thought of as a measurable, though somewhat rudimentary, overview of the entire data. If a time-period is selected then the scatter-plot can be thought of as an overview of that time-period. The real power of the visualization comes when the multiple views are used together to interactively explore the data. This functionality is supported by multiple levels of coordination between views.

One way in which the graph and scatter-plot views are coordinated is that, when a user specifies a time-period using the graph view, features of that time-period are displayed in the scatter-plot view. As there may be a need to view how the time-series evolve over successive periods of time, and to relate the positions of representations between different time-periods, the derived parameters presented in the scatter-plot are calculated using interpolated values. This means that the thumbs of the slider can be moved with a granularity far less than that of the original time-series while the representations of genes in the scatter-plot display move by small increments. This makes it possible for the user to animate the scatter-plot view to gain a perspective of the change in expression for individual, or groups of genes, over time. Unlike traditional animated views, direct manipulation over the specified time-period allows the user to easily compare different frames.

While linear interpolation is not generally considered appropriate for microarray data, in the case of our tool we make an exception for the following reasons. Firstly, linear interpolation is consistent with the value versus time graph representation of the data in the time slider component. Secondly, linear interpolation will not be used to approximate mixing values (which we consider a pre-process), or form queries, as the time slider will click to recorded values when not being manipulated. The sole purpose of linear interpolation within the tool is to allow the user to relate between different gene representations in the scatterplot while the time period is adjusted using the time slider is being moved.

Clicking on a gene representation in the scatter-plot view or the gene list will cause it to be labeled in the scatter-plot and highlighted in the graph view. This action allows the user to interpret the scatter-plot view of the data with a more familiar graph view. Genes may also be labeled in the scatter-plot and highlighted in the graph view by searching for and selecting their name using a 'find gene' menu option. A similar coordination between views is in the presentation of query results. When a query is formed, or in the process of being formed, results are presented in several ways. While replacement is used in the graph view, highlighting is used in the scatter-plot. As queries are incrementally adjusted the result display is updated rapidly in each display. This again will help the user comprehend the scatter-plot view with the more familiar graph representation.

Coordination of views is also required in order to combine multiple queries to reveal patterns of similarity (item 3 of our requirements analysis). During the formulation and modification of an individual query the user can incrementally adjust any of the query parameters, including the time-period context, by manipulating slider thumbs or visual query components. To add a query, whose result will filter the result of any existing queries, the user presses the 'add' button (marked with a plus symbol) on a

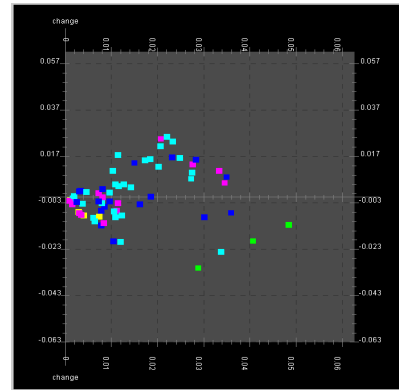


Figure 7: Scatter-plot detail view with functional classifications color-coded.

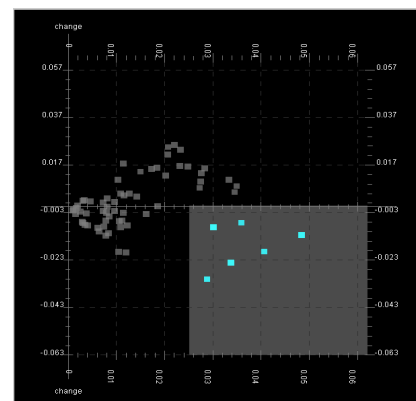


Figure 8: Representation of a query overlaid onto the scatter-plot detail view.

query combination panel. When a new query is added it only has a time-period and a rescaling mode. These parameters will not filter the results of existing queries. This makes it possible to adjust the rescaling and specified time-period to explore the effect on the existing query result. For example, a user may select genes with early-rising expression with log-scaling in an initial query. After adding a new blank query they can alter the specified time-period to examine how these genes behave at a later period. Alternatively, after specifying the initial query, they may switch to no-rescaling mode to assess whether or not any of these genes may have rising log-scaled expression due to the amplification of noise or insignificant variations. The user can even use a blank query to animate the scatter-plot representation of query results viewing how the selected genes expression changes over time. Clicking on the graphical representation of a query within the query combination panel restores that query's rescaling and parameters to the graph view and scatter-plot so that an old query can be modified in much the same way as a new one. Regardless of which query is selected the highlighted selection is always that of the original data filtered by all queries. Clicking between different queries without query modification simply changes the visual query representation and, where appropriate, the time-period or rescaling. In order to remove a query and cancel its filter on the overall selection the user first selects the query then presses the 'remove' button (marked with a minus symbol) on the query combination panel. The graphical representations of queries in the query combination panel are simply boxed textual representations

of the query parameters, time-period context and rescaling outlined with a red border if the query is currently selected.

Finally, users are able to save and restore the measurable queries, combined queries and results for reuse and sharing using simple 'save' and 'open' menu options.

4 Conclusion and further work

Employing requirement analysis to evaluate the limitations of existing microarray time-series visualization techniques led us to the conclusion that a new technique was required in order to support the particular exploration and querying requirements of our users. The most striking limitation of existing tools was the omission of any mechanism by which the user could measurably query according to change in value over a period of time. Further analysis of this and other limitations directed us to the development of a tool that facilitated the exploration of microarray time-series through coordinated graph and scatterplot views. This new visualization technique allows the user to perform and combine a number of queries, including measurable change in value over a period of time queries, through an intuitive direct manipulation interface. The technique also gives the users a unique directly manipulated animated view of microarray time-series that allows them to explore patterns over time for the entire data set and selected subsets. Combining a measurable query mechanism with an intuitive direct manipulation interface will guide the biologists toward query formulation and allow them to quantify query results. Initial testing of the tool proved positive with our users finding it easy to familiarize themselves with the different views of the data and identify specific features of interest in their data. The system is currently undergoing more detailed user evaluation in order to validate the results returned from a variety of different data sets and evaluate specific usability issues.

References

- AHLBERG, C., AND SHNEIDERMAN, B. 1994. Visual Information Seeking: Tight Coupling of Dynamic Query Filters with Starfield Displays. In *Proceedings of the Conference on Human Factors in Computing Systems (CHI'94)*, pages 313-317. ACM Press.
- BRAZAM, A., AND VILO, J. 2000. "Gene expression data analysis," *FEBS letters*, vol. 480, pp. 17-24.
- CRAIG, P., KENNEDY, J. B., AND CUMMING, A. 2002. "Towards Visualising Temporal Features in Large-scale Microarray Time-series Data," presented at 6th International Conference on Information Visualisation - IV2002, University of London, London, GB.
- DEBOUCK, C., AND GOODFELLOW, P. N. 1999. "DNA microarrays in drug discovery and development," *Nat Genet*, vol. 21, pp. 48-50.
- D'HAESELEER, P., WEN, X., FUHRMAN, S., AND SOMOGYI, R. 1999. "Linear Modeling Of mRNA Expression Levels During CNS Development And Injury," *Pacific Symposium on Biocomputing*, pp. 41-52.
- DUGGAN, D. J., CHEN, M., MELTZER, P., AND TRENT, J. 1999. "Expression profiling using cDNA microarrays," *Nature Genetics*, vol. 21, pp. 10-14.
- EICK, S. G. 1994. "Data Visualization Sliders," presented at UIST '94, Marina del Ray, California, USA.
- EISEN, M. B., SPELLMAN, P. T., BROWN, P. O., AND BOSTEIN, D. 1998. "Cluster analysis and display of genome-wide expression patterns," *Proc. Natl. Acad. Sci. USA*, vol. 95, pp. 14863-14868.
- FUA, Y., WARD, M. O., AND RUNDENSTEINER, E. A. 1999. "Hierarchical Parallel Coordinates for Visualizing Large Multivariate Data Sets," presented at IEEE Conference on Visualization.
- HAUSER, H., LEDERMANN, F., AND DOLEISCH, H. 2002. "Angular Brushing of Extended Parallel Coordinates," presented at IEEE Symposium on Information Visualization (InfoVis'02), Boston, Massachusetts, USA.
- HEATH, L. S., AND RAMAKRISHNAN, N., 2002. "The Emerging Landscape of Bioinformatics Software Systems," *IEEE Computer* 35, vol. 7, pp. 41-45.
- HOCHHEISER, H., AND SHNEIDERMAN, B. 2001. "Visual Specification of Queries for Finding Patterns in Time-Series Data," *Proceedings of Discovery Science 2001*.
- JAGOTA, A. 2001. *Microarray Data Analysis and Visualization*. Santa Cruz CA: Bioinformatics by the Bay Press.
- KUMAR, V., FURUTA, R., AND ALLEN, R. B. 1998. Metadata Visualization for Digital Libraries: Interactive Timeline Editing and Review. In *Proc. of ACM Digital Libraries, Pittsburgh, USA*, pp. 126-133.
- NOORDEWIER, M. O., AND WARREN, P. V. 2001. "Gene expression microarrays and the integration of biological knowledge," *Trends in Biotechnology*, vol. 19, pp. 412-415.
- QUACKENBUSH, J. 2001. "Computational Analysis of Microarray Data," in *Nature Reviews*, vol. 2, pp. 418-427.
- SCHENA, M., SHALON, D., HELLER, R., CHAI, A., BROWN, P. O., AND DAVIS, R. W. 1996. "Parallel human genome analysis: microarray-based expression monitoring of 1000 genes," *Proc. Natl. Acad. Sci. U.S.A.* 93, pp. 10614-10619.
- SEO, J., AND BEN SHNEIDERMAN. 2002. "Interactively Exploring Hierarchical Clustering Results," *IEEE Computer*, Volume 35, Number 7, pp. 80-86.
- SHNEIDERMAN, B. 1996. "The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations," presented at IEEE Visual Languages '96, Boulder, Colorado, USA.
- SHNEIDERMAN, B. 1998. *Designing the User Interface*. Reading, MA: Addison Wesley Longman.
- SOMOGYI, R. 1999. "Making sense of gene-expression data," *Pharmainformatics (a Trends Guide)*, pp. 17-24.
- YAFFEE, R. 2000. Introduction to time-series analysis and forecasting with applications of SAS and SPSS. San Diego, CA: Academic Press.