ModVis: an Information Visualization Tool for Gene Module Discovery

Justin Molineaux[1], Jianhua Xuan[1, 2], Ting Gong[2], Yitan Zhu[2], Eric Hoffman[3], Robert Clarke[4], Yue Wang[2]
[1]Dept of EECS, the Catholic University of America, Washington, DC, USA, {05molineaux, xuan}@cua.edu
[2]Dept of ECE, Virginia Polytechnic Institute and State University, Arlington, VA, USA, {tinggong, yitanzhu, yuewang}@vt.edu
[3]Research Center for Genetic Medicine, Children's National Medical Center, Washington, DC, USA, ehoffman@cnmcresearch.org
[4]Lombardi Cancer Center, Georgetown University, Washington, DC, USA. clarker@georgetown.edu

Abstract

The repeated grouping of genes across module sets produced by different module discovery methods may prove to be biologically significant in the area of functional genomics. To ease in the exploration of these overlapping groups of genes, we introduce ModVis, a software tool that allows for the interactive visualization of module discovery data, with an emphasis on the overlap between data sets derived from different methods. Our tool provides a high-level overview of multiple module sets, displaying the modules in each set, and the location and magnitude of the overlap between sets. The tool's more-detailed visualization displays a heat map for a particular gene module, and organizes the heat maps for its overlapping modules so that the individual genes involved in the overlap become readily apparent.

Keywords

Gene clustering, gene regulatory networks, information visualization

1 INTRODUCTION

Recent work in the field of Bioinformatics has focused on the applications of microarray technologies in understanding functional genomics related to human disorders and diseases. Microarray data is a simultaneous measurement of the expression levels of thousands of genes in a single tissue sample. Using this expression data, genes that appear to be coregulated and coexpressed can be assembled into groups known as *modules*. Each module performs a particular biological function, and will often show similar expression patterns under different conditions. The discovery of gene modules such as these could prove to be very useful in the science of dissecting the gene regulation mechanism in pathway signaling and networking.

Many different approaches for gene module discovery have been introduced. While none has become a clear standard, there are several interesting methods that are worth mentioning in brief. Examples include VIsual Statistical Data Analyzer (VISDA), which uses hierarchical Standard Finite Normal Mixtures while exploiting the human gifts for pattern recognition [1]. Tamayo et al used Self-Organizing Maps (SOM) to find gene clusters with different change patterns of gene expression level [2]. CLICK, developed by Sharan and Shamir, is a graph-theoretic and statistical algorithm used for this same purpose [3]. Additionally, Lee and Batzoglou accomplished this task using independent component analysis [4]. Similarly, Gong et al. offered non-negative ICA (nICA) as an improvement on ICA [5].

Each of these techniques produces a set of modules that are believed to be biologically relevant and interesting. While this is certainly true, we believe that the overlap – the repeated clustering of a few genes – across different clustering methods will also prove to be quite interesting. If several genes regularly appear to be grouped together, we can believe with more confidence that they indeed share some true biological function.

To ease the exploration of "inter-method" overlap, we introduce ModVis (short for Module Visualization), a software tool that provides an interactive visualization of gene module discovery data and emphasizes the overlap between module sets. Section II will present an overview of the software's data structure, as well as an explanation of the different visualization techniques it offers. Some experimental results will be presented and discussed in section III. Section IV will explore the strengths and weaknesses of this approach.

2 METHODOLOGY

2.1 Data Structuring and Preparation

When dealing with mass quantities of information, such as microarray data, it is often difficult to design software in a way that allows the user to perform complicated queries while maintaining an interactive level of performance. This subsection will discuss the design concepts we have implemented to allow for high functionality and enable fast performance.

For ModVis to perform its job, three input files are necessary. We refer to the first as a *reference table* which matches each gene identifier (i.e. probe set ID) with one or more expression values. The second and third files are referred to as *module sets*, each containing several columns of gene identifiers. Each column in each of these module sets represents a module. Although we will be demonstrating the features of ModVis with two module sets produced by different discovery methods, it is not necessary that each module set be the result of a

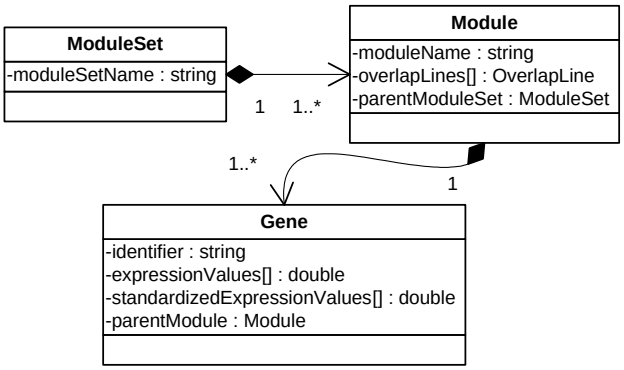


Figure 1: UML class diagram describing module set data structure. Methods and data members relating to the visualization of the object have been omitted for the sake of brevity.

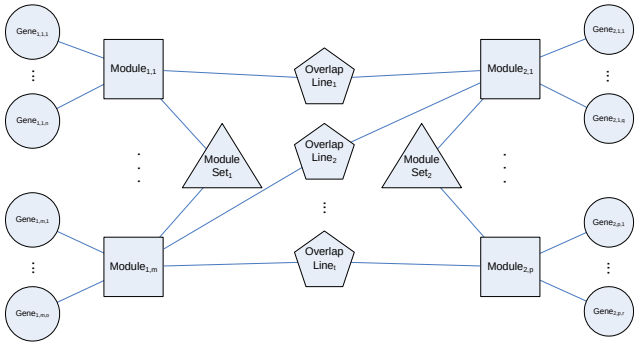


Figure 2: When data preparation is complete, the data objects are organized in this fashion. Each line is traversable in both directions.

different method, or that the number of module sets be limited to two.

From these input files, all of the necessary information is collected in a data structure with two distinct parts: the module set data and the overlap data.

A Module Set Data

Our input module set data follows a natural tree-like pattern in which a module set contains multiple modules and each module contains multiple genes. We have modeled this structure as closely as possible in ModVis and have found that it performs well as a data structure when some pre-processing of the data is done.

In our software, each module set is stored in its own tree-like structure. A *module set node* acts as the root of the tree, storing the module set's name and an array of references to all of the *module nodes* contained in the module set. Each module node contains specific information relating to the module it represents, including the module's name (usually just an index), an array of *overlap lines* (which will be discussed in the next subsection), and an array of references to the *gene nodes* contained in the module. Each gene node contains specific information relating to the gene it represents. This information includes a gene identifier, an array of raw expression values, and an array of standardized expression values. The standardization process for the gene expression values sets the mean expression across all samples for each gene to 0, allowing us to measure variations in the expression values in terms of standard deviation. Because performing the standardization calculation for every query will pose a great hindrance to performance, we perform the standardization process only once when the files are loaded, and store the standardized values along with the non-standardized values.

Each node in the tree also keeps a reference to its parent node (i.e. a gene node will keep a reference to its parent module node) to provide for two-directional traversal of the tree. A UML class diagram describing this structure is provided in Figure 1.

All of the pre-processing of the data takes place when a module set is loaded, and is updated as necessary when each additional module set is loaded. This allows for faster query times and an interactive level of performance for the user. The tradeoff however, is that the user will have to wait for this data processing to take place at load time, and depending on the amount of data, a large chunk of memory could be consumed by the data structure.

B Overlap Data

When a module set is loaded, each module is compared against all other modules to check for overlap. If two modules overlap (meaning one or more genes appear in both modules), an *overlap line* is created to represent this relationship. These overlap lines exist outside of the module set trees, but each module node contains an array of references to its respective overlap lines. The overlap line objects, in turn, contain references to the modules that are involved in the overlap, furthering the ease of tree traversal and hastening query times. The visual properties of these lines will be discussed in the next subsection.

A general description of the organization of the module set trees and overlap lines is presented in Figure 2.

2.2 View #1: Finding the Location and Magnitude of the Overlaps

The first view that ModVis provides is an abstracted representation emphasizing the overlap between modules in different module sets. This abstraction of the data allows the user of the software to not be overwhelmed by the mass amount of data, yet still recognize the trends that he or she is looking for –

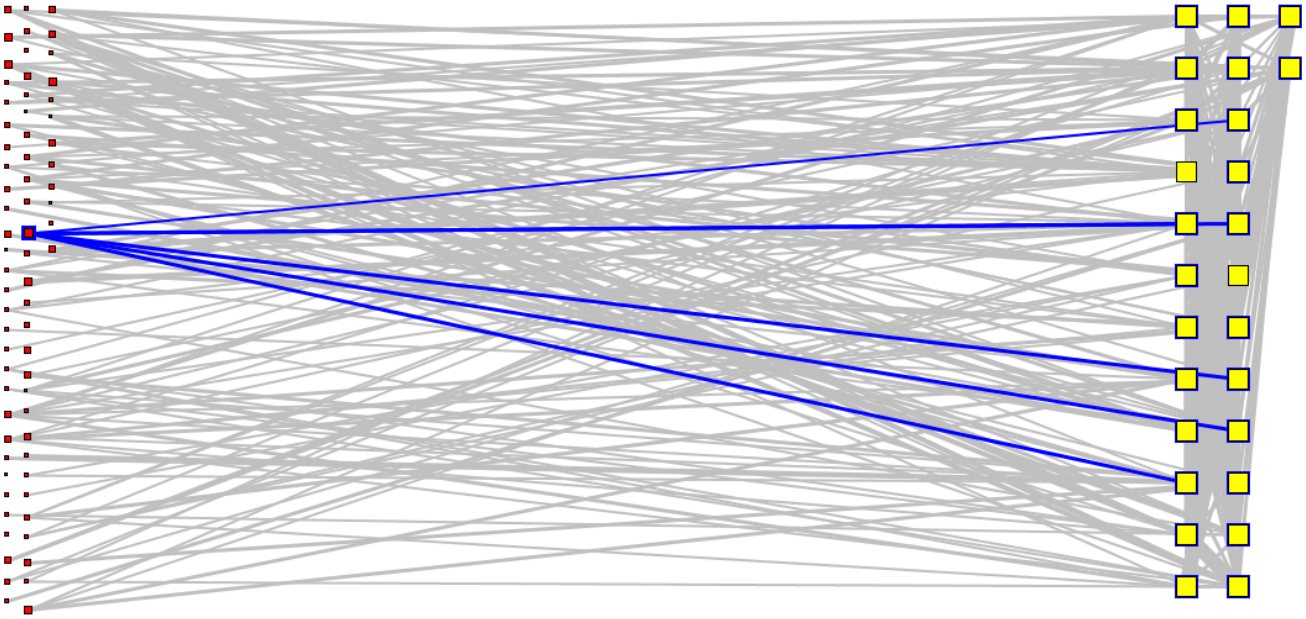


Figure 3: Visual representation of the location and magnitude of the overlap between two module sets. The modules on the left have been created from the VISDA module discovery process, and the modules on the right have been created using the ICA process. In this case, VISDA module #39 has been selected, causing it and the modules it overlaps with to be outlined in blue. The overlap lines that extend from VISDA module #39 have also been highlighted in blue. Using the line-thickness threshold, lines representing overlaps of fewer than 20 genes have not been drawn so as to make the overlaps of larger magnitude (greater than 20 genes) readily apparent.

namely, large groups of overlapping genes. The finer-grained details of the data will become visible in the second, more module-specific representation.

A The Method of Abstraction

To visually summarize the data without sacrificing its accuracy, we represent each module as a square box, where the width w and height h of the box correspond to the number of genes in the module, and the color of the box indicates the module set to which the module belongs. Because the number of genes contained in a module may be quite large, and real estate on the screen is precious, we use the following function to calculate the width and height of each box:

$$w = h = \sqrt{N} \times X$$

where N is the number of genes in the module and X is a constant multiplier used to scale the size of the boxes in the event that a module set has very small modules.

When more than one data set is loaded, each module of each data set is compared to every other module of every other data set. In the case that a given gene appears in two modules, an overlap line (as introduced in the previous section) is drawn to connect those two modules. It is quite common, and in fact desired, that two modules will share many genes. To handle this, we borrow a technique from Van Ham and Van Wijk's work on small world graphs [6]. Instead of drawing a new line for each gene shared by the two modules, we simply increment the thickness of the existing line for each subsequent shared gene. Thus, the thickness of the line is relative to the number of genes shared by the two modules. As before, we must conserve screen space, so we scale the thickness of the line with the following equation:

$$thickness = \sqrt{N} \times X$$

where N is the number of shared genes and X is a constant multiplier used to further scale the thickness of the lines if necessary.

It may be the case that thinner lines (indicating an overlap of just a few genes) are not of interest. If this is true, the user can specify a line thickness threshold that will prevent the smaller lines from being drawn on the screen, removing clutter from the visualization.

The use of these lines provides us with an effective means of identifying where the overlap between two data sets occurs (location), and how much of an overlap exists (magnitude). Although we don't know exactly which genes are involved in the overlap just yet, we do know which modules to explore further. Figure 3 provides an example of how this representation might look when two module sets are loaded. A module-specific visualization of the overlap will be discussed in subsection C.

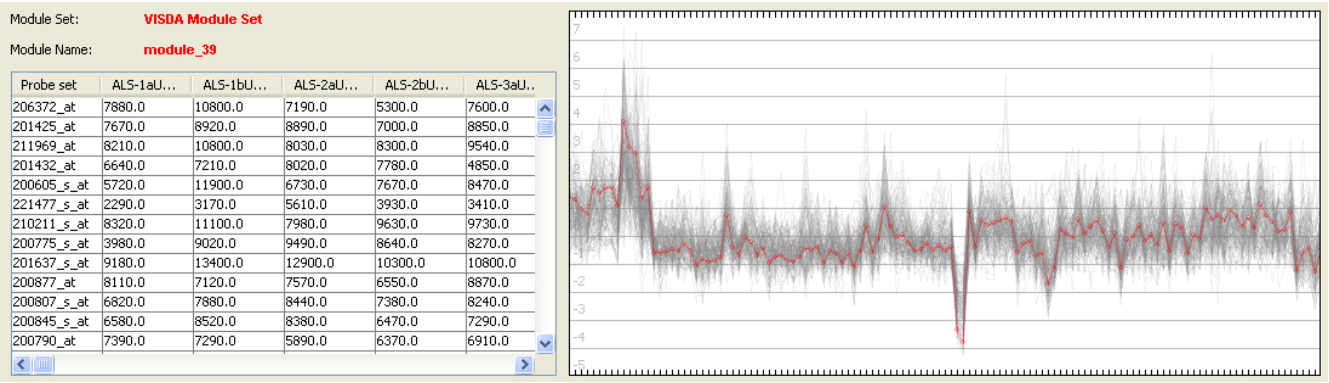


Figure 4: This information panel appears below the diagram presented in figure 3 and provides more-detailed information regarding the current selection. The names of the selected module and the module set to which it belongs are both displayed, as well as a table of gene identifiers and expression values for the genes belonging to the selection. On the right, a parallel coordinates graph provides a general overview of the expression pattern across all of the samples in the reference table (125 in our case).

B Interactivity

Although the above technique does successfully abstract the data and emphasize overlap, it is much easier for the user to explore the data if he or she can interact with it. Realizing this, we have incorporated a few interactive features into our visualization.

Perhaps most importantly, a module can be selected (by clicking once inside its square), allowing the user to further inspect it. When a single module is selected, it becomes highlighted along with all of its overlapping modules and all of the lines that connect the overlapping modules to the selected module. This makes the overlapping modules readily identifiable, the connecting lines easier to follow and their thicknesses easier to compare. In figure 3, VISDA module #39 is selected, allowing us to see which other modules overlap with it.

In addition to the highlighting that occurs in the representation of the data, the information panel shown in figure 4 springs to life when a module is selected. The genes in the module and their corresponding expression values are displayed in the table on the left side of this panel. The data in this table is also displayed more-generally in the parallel coordinates graph to the right [7]. In this graph, each gray line represents a gene, and each vertical axis represents a sample. Where the line representing a gene intersects an axis representing a sample, we can read an expression value for that gene. The red line on the graph displays the average expression value for each sample. All of the data in the graph is standardized for the purpose of emphasizing abnormalities.

To emphasize the overall "shape" of the expression pattern, we have made two important alterations to our graph, which cause it to vary from the traditional parallel coordinates approach. The first is that we have not drawn the vertical axes on the graph, but have instead used small tick marks to represent their location. This compels the user to read the graph as an overall presentation of the expression pattern for the selected genes, rather than a plot of the expression value of each gene in each sample. The second alteration is that we have drawn each of the lines that represent a gene at only 25% opacity. Thus, where multiple lines are plotted on top of each other, we see a darker color. This alteration does not remove any data from the graph, but rather focuses the user's attention on the general expression pattern, which will appear darker. The parallel coordinates graph in figure 4 shows how the outlying data points appear much lighter, while the data points that fall close to the mean appear much darker, emphasizing their participation in the general pattern.

More than one module can be selected at a time by holding the CTRL key and clicking additional modules. When this is done, only the selected modules and the lines connecting them are highlighted, emphasizing only the overlap that exists within the selection. In addition, the gene table and the parallel coordinates graph change to show only the genes that are shared by the modules in the selection.

This allows us to determine which modules overlap and to what degree. If we wish to explore these overlapping modules in more detail, we must use a module-specific visual representation.

2.3 View #2: Exploring Overlap at the Module Level

When the user has selected one or more modules that he or she wishes to explore further, a second visualization technique can be used to examine them in more detail. This second view employs the Eisen et al. heat map technique for visualizing microarray data [8].

A heat map is drawn for each of the selected modules, and is organized so that common genes are aligned horizontally. In the heat maps, each row of pixels corresponds to a gene, and each column of pixels

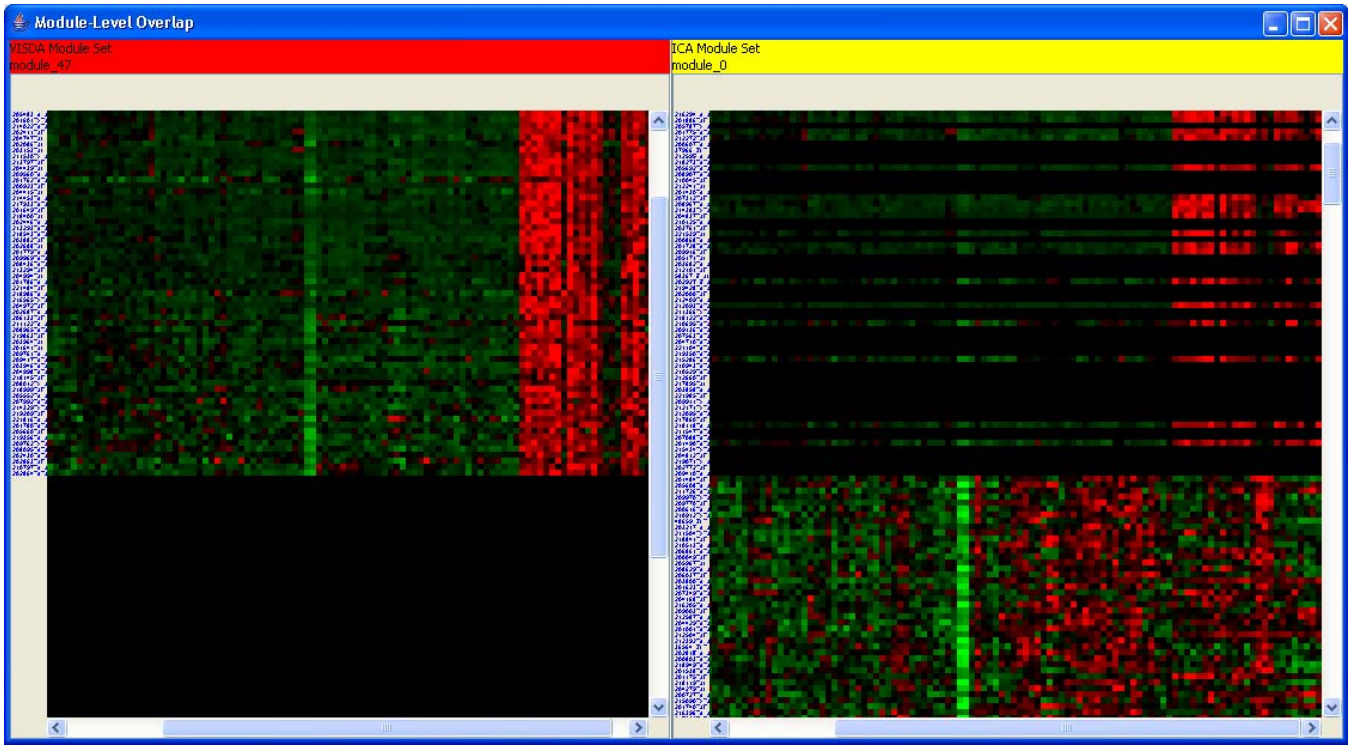


Figure 5: With view #2, the user is able to examine the overlap in more detail by placing the heat map for each of the selected modules side by side. This heat map is laid out so that each row represents a different gene, and each column represents a different sample. Red is used to represent an up-regulated expression value, and green is used represent a down-regulated value. The intensity of the color corresponds to the degree of up- or down-regulation. More than two heat maps can be displayed at one time, allowing the user to see which genes appear in each of the selected modules. In this figure, we can see that the genes appearing in VISDA module #47 tend to follow a more uniform expression pattern than those appearing in ICA module #0.

corresponds to a sample. The rows of the leftmost heat map are organized in exactly the order that the probe IDs appear in the input module data, and all subsequent heat maps are organized according to the order of genes in this first heat map. This causes any overlapping genes to appear in the same row across all heat maps, making them readily identifiable. Genes that do not overlap are placed at the bottom of each map, below the last row of the leftmost heat map. An example of this representation is presented in figure 5.

To make it easier to read the heat map, we allow the user to adjust the width and height of the pixels, and to choose the colors that represent up- and down-regulated expression values. By default, red is used to indicate an up-regulated expression value, and green is used to indicate a down-regulated expression value.

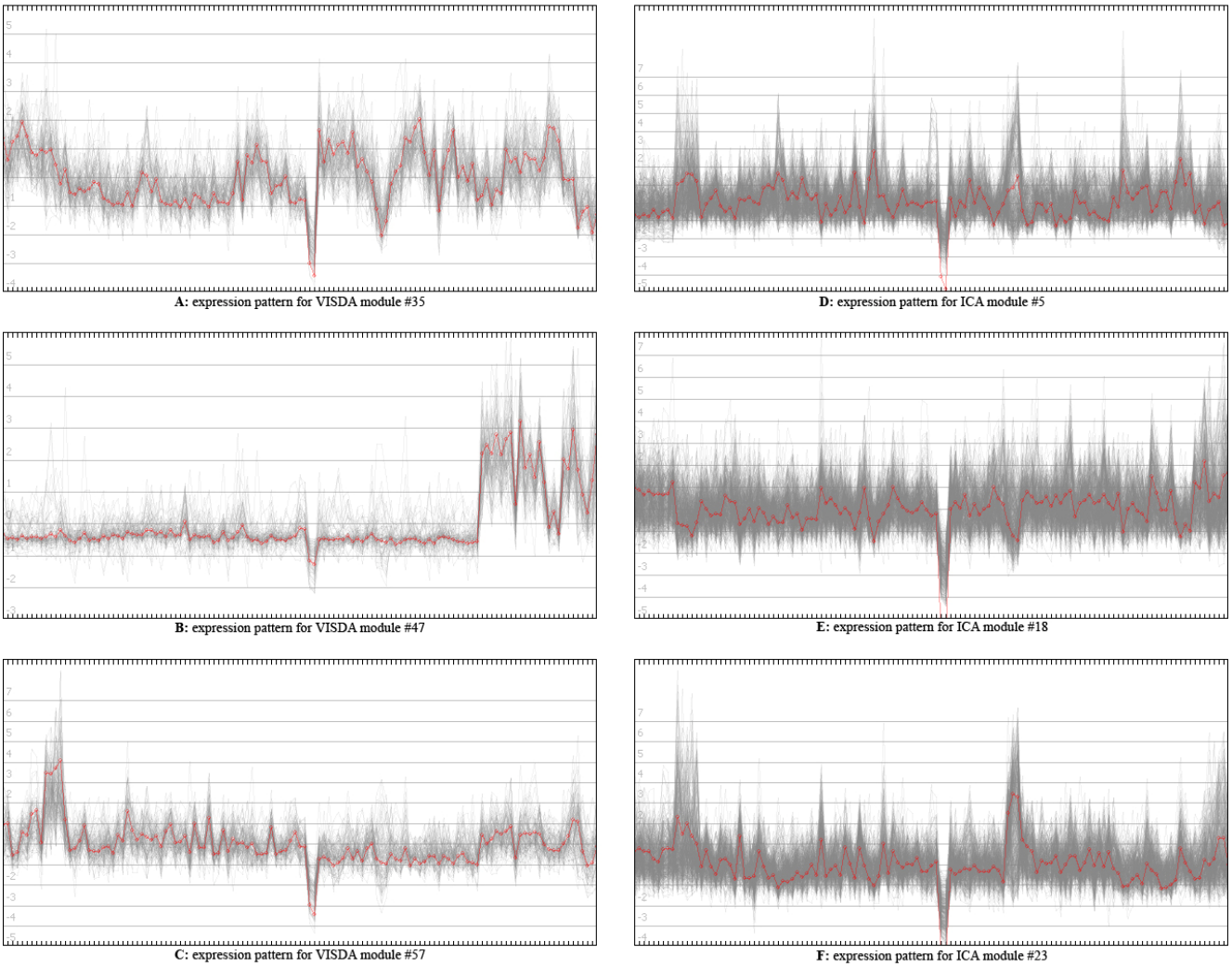
We believe that using heat maps to represent modules in more detail will make visible the overall expression trends in the module, while accenting the commonalities between overlapping modules.

3 EXPERIMENTAL RESULTS

We used ModVis to explore the overlap between a module set produced by the VISDA module discovery method and a second module set produced by the ICA method. Our VISDA module set has 69 gene modules of varying size, and the ICA set has 26 modules of 450 genes each. Our reference table, from which these module sets were created, is a muscular dystrophy data set that includes samples from 125 different patients. Each of these samples belongs to one of 13 different phenotypes which, in turn, each correspond to a specific type of muscular dystrophy.

Using view #1, we find that a great deal of overlap exists between the two sets of modules, but that the magnitude of the overlap in most cases is quite small: 5 genes or fewer. These small overlaps are not of interest, and also make the overlaps of significant size somewhat difficult to detect. By increasing the overlap line thickness threshold to 20 genes, we are able to filter out much of the noise from our display, and clearly see where overlaps of significant size occur. The resulting representation is shown in figure 3.

Immediately noticeable is the relatively small size of a VISDA module when compared to that of an ICA module. It is also easy to notice that the VISDA modules are mutually exclusive, while the ICA module set appears to have quite a bit of overlap within itself. Upon further inspection, we see that the general expression pattern for each VISDA module, when displayed across all 125 samples, seems to have a rather



A: expression pattern for VISDA module #35

D: expression pattern for ICA module #5

B: expression pattern for VISDA module #47

E: expression pattern for ICA module #18

C: expression pattern for VISDA module #57

F: expression pattern for ICA module #23

Figure 6: Our inspection of the parallel coordinates representation for each of the modules in our two module sets shows us that the VISDA modules tend to have unique expression patterns where the genes are tightly coexpressed across all samples. Contrastingly, the ICA modules all tend to have very similar expression patterns with a weaker coexpression, and in some samples, no coexpression at all. The drastic down-regulation of a couple samples in each module is due to a defect in the microarray data. Because of this, we must ignore those two samples.

unique rise and fall, unlike the ICA modules which all have similar patterns. Additionally, the genes in a VISDA module tend to be more tightly coexpressed across the 125 samples than the genes in an ICA module. Figure 6 displays the parallel coordinates representations for some of the modules from each set side-by-side, making the differences in expression pattern apparent. These differences are most likely due to the nature of the discovery methods themselves. For example, the VISDA method attempts to cluster genes based on their coexpression, and includes human input that governs this process. The ICA method is purely statistical, and does not use coexpression as a criterion for creating modules.

Also immediately noticeable from observing the parallel coordinates graphs is the fact that all genes in two of the samples in the middle of our data set are extremely down-regulated. This is due to a defect in the experiment that produced the muscular dystrophy data set. Because these measurements are erroneous, we must ignore them. When we observe modules using the heat map view, we will see this defect in the form of two columns of bright green pixels. It is unfortunate that some of the data is not usable, but this does confirm that our visualization techniques can successfully point out such abnormalities in the data.

Some modules, such as VISDA module #47 (shown in figure 6B), appear to have very interesting expression patterns. This particular module is slightly yet consistently down-regulated in the first 100 samples of our data set. In the last 25 samples, however, the expression of the genes in this module becomes drastically up-regulated and much less consistent, causing us to believe that this module bears some relevance to Juvenile Dermatomyositis (JDM), with which all of these 25 patients have been diagnosed. Analyzing this module with Ingenuity's Pathways Analysis software (IPA) reveals that it does have some biological significance, but not quite enough for us to be confident in claiming

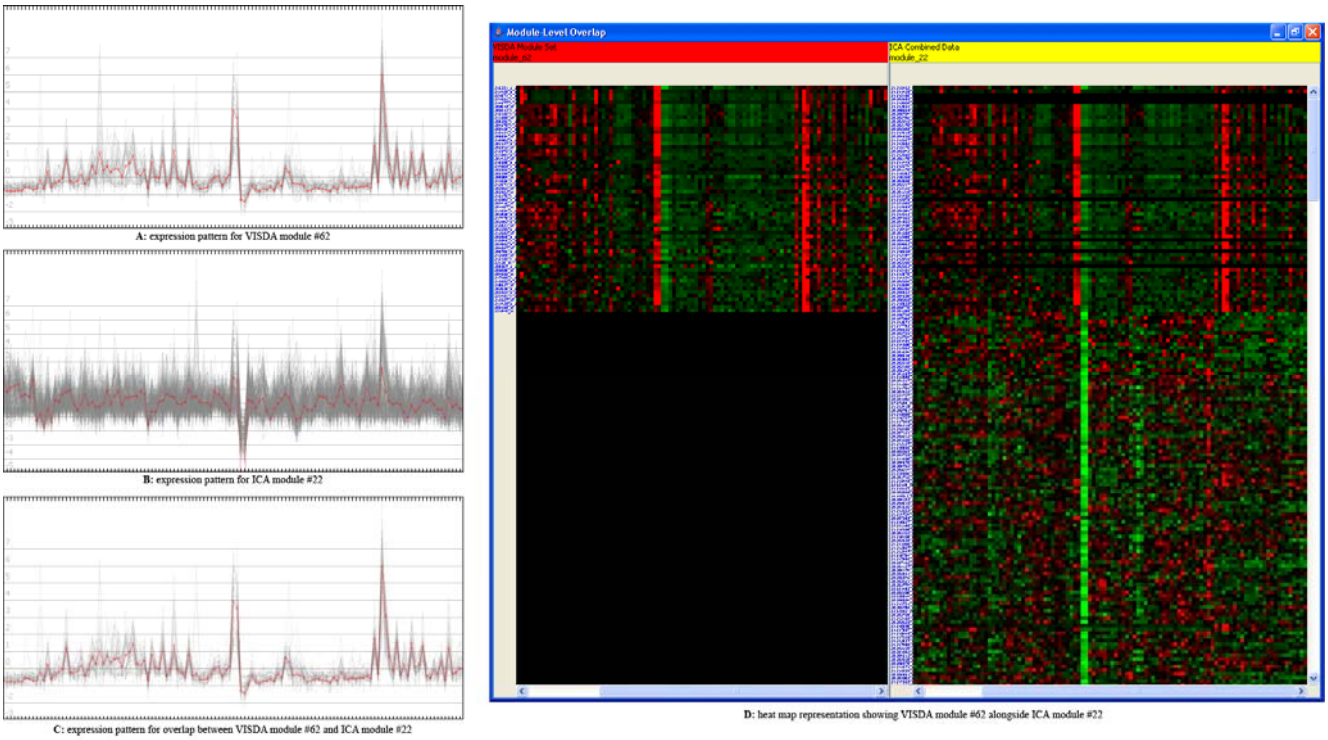


Figure 7: The exploration of the overlap between VISDA module #62 and ICA module #22 shows that nearly all genes that appear in the VISDA module also appear in the ICA module.

anything more than that it simply has an interesting expression pattern [9].

Our hypothesis has been that the overlap between two modules will bear an even higher significance to the biological function of its contributing modules. In the overlap between VISDA module #62 and ICA module #22, this hypothesis becomes realized.

Upon first observation, VISDA module #62 does not appear to have a particularly exciting expression pattern. However, its expression does seem to follow loosely with the different phenotypes in our reference table. When we check the biological significance of this

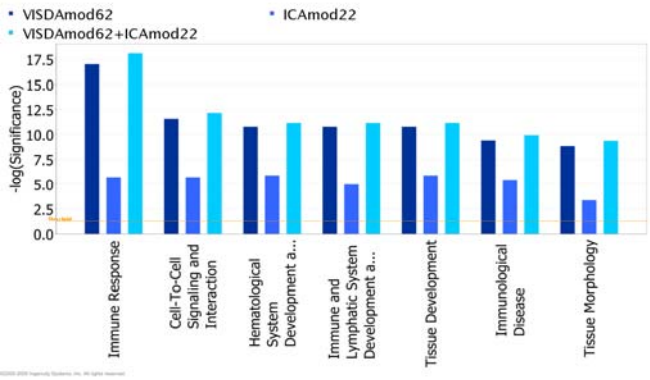


Figure 8: Functional significance of VISDA module #62 (represented by dark blue), ICA module #22 (represented by medium blue), and the overlap of these two modules (represented by light blue). This figure shows that the significance of the overlap is slightly higher than either of the two contributing modules.

module using IPA, we see that it has a very high significance to several immunological and tissue development functions. These functions are known to have a biological relationship to the JDM phenotype. A further study of this disease tells us that JDM is a rather severe childhood autoimmune disorder which is believed to be associated with viral infections that stimulate muscle destruction by inflammatory cells and ischemic processes.

From the display in view #1, it is easy to see that VISDA module #62 has a large overlap with ICA module #22. An analysis of ICA module #22 finds that it also has some significance to many of the same functions as VISDA module #62. Our exploration of the overlap between the two modules shows that it shares the same general expression pattern as the VISDA module, due to the fact that all but a few genes from the VISDA module are included in the overlap. In figure 7D, we can see that this pattern and the inclusiveness of the overlap are immediately noticeable in view #2 – the heat map view. This display shows the heat map for the VISDA module on the left, and the heat map for the ICA module on the right. The heat map for the ICA module is arranged in a way that will force each row of pixels for each gene that is involved in the overlap to appear toward the top, aligned with that same gene's row of pixels in the VISDA module. Normally, the number of genes involved in the overlap does not approach the total number of genes in either module, but in this case it appears as though we have an exception. In the ICA module's heat map, we see only a few empty rows toward the top, meaning that only

a few of the genes in the VISDA module do not appear in the ICA module. We can also see that the expression pattern of the genes involved in the overlap is very different from the general expression pattern for the ICA module. Figure 7 shows the expression patterns of VISDA module #62, ICA module #22, and the overlap between these two modules.

In order to determine if this particular case follows our hypothesis (that the overlap will be of greater significance than either of its contributing modules), we must use IPA once more. This analysis software allows us to produce a side-by-side display of the significance for certain biological functions of each of the modules and their overlap. This graph, presented in figure 8, shows that both modules bear significance to several common biological functions. We immediately notice that the VISDA module has a very strong significance on its own, and the significance of the ICA module appears to be much lower that that of the VISDA module. This is not to say that the ICA module is insignificant, but rather that it contains many "extra" genes that are not relevant to any of these biological functions. Our hope is that the overlap between the two modules will essentially filter out these extra genes, yielding a smaller cluster of genes with a higher biological significance.

The third, light blue bar for each function is most interesting to us because it effectively supports our hypothesis. This bar represents the biological significance for the overlap between the two modules, and for each function shows that the significance of the overlap is greater than that of the VISDA module and the ICA module. In addition to having a higher significance, the overlap also contains fewer genes than either of its two contributing modules, meaning that it is a more "pure" cluster of genes.

4 DISCUSSION

Although our approach is still quite new, we have already seen some very promising results. The ability to refine a module by weeding out its extra, irrelevant genes may prove to be very valuable to future research.

Aside from the biological significance of the overlaps that we are able to find with ModVis, the visualization process itself seems to be very valuable. From the displays produced by the software, we are immediately able to pinpoint abnormalities in the data, such as the defect in two of the samples in our microarray data set. We can also use this visualization to verify that the module discovery methods worked as they should. For example, we know that the VISDA method assembles modules based on common expression patterns, and can very easily confirm this by inspecting the modules with our software.

Currently ModVis is a standalone application, but we believe that it would be very useful as an integrated component of the existing VISDA software. This would allow VISDA users to compare their modules to known modules as the gene clustering process takes place, perhaps offering some guidance in the clustering process. Although the true benefits of this approach are still unknown, it would not be a terribly difficult task to accomplish, and so is worth exploring.

Additionally, we plan to develop a feature that will allow ModVis to interface with databases, similar to that of Ingenuity, allowing the user to measure the significance of a module or overlap without ever having export the data out of ModVis.

5 ACKNOWLEDGEMENT

This work is supported by the National Institute of Health under grants CA109872, NS29525, and by the Department of Defense under grant BC030280.

6 REFERENCES

[1] Y. Zhu, et al., "Phenotypic-Specific Gene Module Discovery using Diagnostic Tree and VISDA," *IEEE EMBS Annual Int. Conf*, 2006.

[2] P. Tamayo, et al., "Interpreting patterns of gene expression with self-organizing maps: methods and application to hematopoietic differentiation," *Proc. Natl. Acad. Sci. USA*, vol. 96, pp. 2907-12, March 1999

[3] R. Sharan and R. Shamir, "CLICK: A clustering algorithm with applications to gene expression analysis," in *Proc.8th Int. Conf. Intelligent Systems for Molecular Biology*, 2000, pp. 307-16

[4] S. Lee and S. Batzoglou, "Application of independent component analysis to microarrays," *Genome Biology*, vol. 4, issue 11, pp. R76.1-21, 2003

[5] T. Gong et al., "Composite Gene Module Discovery using Non-negative Independent Component Analysis," *LSSA Bi-annual Int. Workshop*, 2006.

[6] F. van Ham and J. van Wijk, "Interactive Visualization of Small World Graphs," *IEEE Symposium on Information Visualization*, October 2004

[7] A. Inselberg and B. Dimsdale, "Parallel Coordinates: A Tool for Visualizing Multidimensional Geometry," *in Proc. of IEEE Conf. on Vis. '90*, 361-378. IEEE Comp. Soc., 1990

[8] M. Eisen et al., "Cluster analysis and display of genome-wide expression patterns, " *Proc. Natl. Acad. Sci. USA* 95 14863-14868, 1998

[9] http://www.ingenuity.com