1

Data Mining in Computational Biology

Mohammed J. Zaki
Rensselaer Polytechnic Institute

Karlton Sequeira
Rensselaer Polytechnic Institute

1.1 Introduction

Data Mining is the process of automatic discovery of valid, novel, useful, and understandable patterns, associations, changes, anomalies, and statistically significant structures from large amounts of data. It is an interdisciplinary field merging ideas from statistics, machine learning, database systems and data-warehousing, high-performance computing, as well as visualization and human-computer interaction. It has been engendered by the phenomenal growth of data in all spheres of human endeavor, and the economic and scientific need to extract useful information from the collected data.

Bioinformatics is the science of storing, extracting, analyzing, interpreting, and utilizing information from biological data such as sequences, molecules, pathways, etc. Genome sequencing projects have contributed to an exponential growth in complete and partial sequence databases. The structural genomics initiative aims to catalog the structure-function information for proteins. Advances in technology such as microarrays have launched the subfields of genomics and proteomics to study the genes, proteins, and the regulatory gene expression circuitry inside the cell. What characterizes the state of the field is the flood of data that exists today or that is anticipated in the future; data that needs to be mined to help unlock the secrets of the cell. Data mining will play a fundamental role in understanding

0-8493-8597-0/01/$0.00+$1.50

these rapidly expanding sources of biological data. New data mining techniques are needed to analyze, manage and discover sequence, structure and functional patterns/models from large sequence and structural databases, as well as for structure prediction, gene finding, gene expression analysis, biochemical pathway mining, biomedical literature mining, drug design and other emerging problems in genomics and proteomics.

The goal of this chapter is to provide a brief introduction to some data mining techniques, and to look at how data mining has been used in some representative applications in bioinformatics, namely three-dimensional (3D) or structural motif mining in proteins and the analysis of microarray gene expression data. We also look at some issues in data preparation, namely data cleaning and feature selection via the study of how to find normal variation in gene expression datasets. We first begin with a short introduction to the data mining process, namely the typical mining tasks and the various steps required for successful mining.

1.2 Data Mining Process

Typically data mining has the two high-level goals of prediction and description. The former answers the question "what", while the latter the question "why". That is, for prediction the key criteria is that of accuracy of the model in making future predictions; how the prediction decision is arrived at may not be important. For description, the key criteria is that of clarity and simplicity of the model describing the data, in human-understandable terms. There is sometimes a dichotomy between these two aspects of data mining in the sense that the most accurate prediction model for a problem may not be easily understandable, and the most easily understandable model may not be highly accurate in its predictions. It is crucial that the patterns, rules, and models that are discovered are valid not only in the data samples already examined, but are generalizable and remain valid in future new data samples. Only then can the rules and models obtained be considered meaningful. The discovered patterns should also be novel, and not already known to experts; otherwise, they would yield very little new understanding. Finally, the discoveries should be useful as well as understandable.

1.2.1 Data Mining Tasks

In verification-driven data analysis the user postulates a hypothesis, and the system tries to validate it. The common verification-driven operations include querying and reporting, multidimensional analysis, and statistical analysis. Data mining, on the other hand, is discovery-driven, i.e., it automatically extracts new hypotheses from data. The typical data mining tasks include:

- Association Rules: Given a database of transactions, where each transaction consists of a set of items, association discovery finds all the item sets that frequently occur together, and also the rules among them. For example, a rule could be as follows: 90% of the samples that have high expression levels for genes $\{g_1, g_5, g_7\}$ have high expression levels for genes $\{g_2, g_3\}$, and further 30% of all samples support this rule.
- Sequence Mining: The sequence mining task is to discover sequences of events that commonly occur together. For example, 70% of the DNA sequences from a family have the subsequence $TATA$ followed by ACG after a gap of, say, 50 bases.

- Similarity Search: An example is the problem where we are given a database of objects and a "query" object, and we are then required to find those objects in the database that are similar to the query object. Another example is the problem where we are given a database of objects, and we are then required to find all pairs of objects in the databases that are within some distance of each other. For example, given a query 3D structure, find all highly structurally similar proteins.
- Deviation Detection: Given a database of objects, find those objects that are the most different from the other objects in the database, i.e., the outliers. These objects may be thrown away as noise, or they may be the "interesting" ones, depending on the specific application scenario. For example, given microarray data, we might be able to find a tissue sample that is unlike any other seen, or we might be able to identify genes with expression levels very different from the rest of the genes.
- Classification and Regression: This is also called supervised learning. In the case of classification, we are given a database of objects that are labeled with predefined categories or classes. We are required to learn from these objects a model that separates them into the predefined categories or classes. Then, given a new object, we apply the learned model to assign this new object to one of the classes. In the more general situation of regression, instead of predicting classes, we have to predict real-valued fields. For example, given microarray gene expression data, with tissues from cancerous and non-cancerous patients, a classification model might be able to predict for a new tissue sample, whether it is cancerous or not.
- Clustering: This is also called unsupervised learning. Here, we are given a database of objects that are usually without any predefined categories or classes. We are required to partition the objects into subsets or groups such that elements of a group share a common set of properties. Moreover the partition should be such that the similarity between members of the same group is high and the similarity between members of different groups is low. For example, given a set of protein sequences, clustering can group them into similar (potentially homologous) families.

1.2.2 Steps in Data Mining

Data mining refers to the overall process of discovering new patterns and building models from a given dataset. There are many steps involved in the mining enterprise such as data selection, data cleaning and preprocessing, data transformation and reduction, data mining task and algorithm selection, and finally post-processing and interpretation of discovered knowledge:

- Collect, clean and transform the target dataset: Data mining relies on the availability of suitable data that reflects the underlying diversity, order, and structure of the problem being analyzed. Therefore, the collection of a dataset that captures all the possible situations that are relevant to the problem being analyzed is crucial. Raw data contain many errors and inconsistencies, such as noise, outliers, and missing values. An important element of this process is to remove duplicate records to produce a non-redundant dataset. Another important element of this process is the normalization of data records to deal with the kind of pollution caused by the lack of domain consistency.

- Select features, reduce dimensions: Even after the data have been cleaned up in terms of eliminating duplicates, inconsistencies, missing values, and so on, there may still be noise that is irrelevant to the problem being analyzed. These noise attributes may confuse subsequent data mining steps, produce irrelevant rules and associations, and increase computational cost. It is therefore wise to perform a dimension reduction or feature selection step to separate those attributes that are pertinent from those that are irrelevant.
- Apply mining algorithms and interpret results: After performing the pre-processing steps apply appropriate data mining algorithms—association rule discovery, sequence mining, classification tree induction, clustering, and so on—to analyze the data. After the algorithms have produced their output, it is still necessary to examine the output in order to interpret and evaluate the extracted patterns, rules, and models. It is only by this interpretation and evaluation process that we can derive new insights on the problem being analyzed.

1.3 Mining 3D Protein Data

Protein structure data has been mined to extract structural motifs [29, 22, 31], analyze protein-protein interactions [55], protein folding [65, 39], protein thermodynamic stability [5, 38], and many other problems. Mining structurally conserved motifs in proteins is extremely important, as it reveals important information about protein structure and function. Common structural fragments of various sizes have fixed 3D arrangements of residues that may correspond to active sites or other functionally relevant features, such as Prosite patterns. Identifying such spatial motifs in an automated and efficient way, may have a great impact on protein classification [13, 3], protein function prediction [20] and protein folding [39]. We first describe in detail a graph-based method for mining structural motifs, and then review other approaches.

1.3.1 Modeling Protein Data as a Graph

An undirected graph $G(V, E)$ is a structure that consists of a set of vertices $V = \{v_1, \cdots, v_n\}$ and a set of edges $E \subseteq V \times V$, given as $E = \{e_i = (v_i, v_j) | v_i, v_j \in V\}$, i.e., each edge e_i is an unordered pair of vertices. A weighted graph is a graph with an associated weight function $W : E \rightarrow \Re^+$ for the edge set. For each edge $e \in E$, $W(e)$ is called the weight of the edge e. Protein data is generally modeled as a connected weighted graph where each vertex in the graph may correspond to either a secondary structure element (SSE) (such as α-helix or β-strand) [22, 66], or an amino acid [29], or even an individual atom [31]. The granularity presents a trade-off between computation time and complexity of patterns mined; a coarse graph affords a smaller problem computationally, but runs the risk of oversimplifying the analysis. An edge may be drawn between nodes/vertices indicating proximity, based on several criteria, such as:

- Contact Distance (CD) [29, 66]: Given two amino acids a_i and a_j along with the 3D co-ordinates of their α-Carbon atoms (or alternately β-Carbon), (x_i, y_i, z_i) and (x_j, y_j, z_j), resp., define the Euclidean distance between them as follows:

$$\delta(a_i, a_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$$

  We say that a_i and a_j are in contact, if $\delta(a_i, a_j) \leq \delta^{\max}$, where $\delta^{\max}$ is some maximum allowed distance threshold (a common value is $\delta^{\max} = 7\mathring{A}$). We can

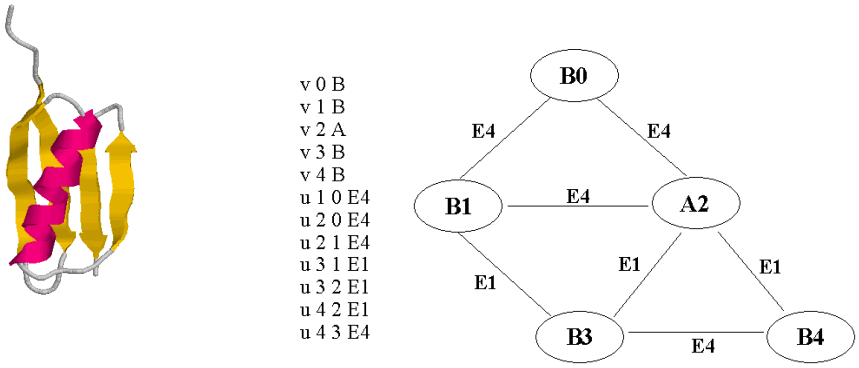
add an edge between two amino acids if there are in contact. If the vertices are SSEs then the weight on an edge can denote the number of contacts between the two corresponding SSEs.

- Delaunay Tessellation (DT)[29, 54, 62]: the Voronoi cell of $x \in V(G)$ is given by

$$\mathcal{V}(x) = \{y \in \Re^3 | d(x,y) \leq d(x',y) \quad \forall x' \in V(G) \backslash \{x\}\}$$

  An edge connects two vertices if their Voronoi cells share a common face.
- Almost Delaunay (AD) [29]: to accommodate for errors in the co-ordinate values of the points in the protein structure data, we say that a pair of points $v_i, v_j \in V(G)$ are joined by an almost-Delaunay edge with parameter ϵ, or AD(ϵ), if by perturbing all points by at most ϵ, v_i, v_j could be made to have Voronoi cells sharing a face.

1.3.2 Graph-based Structural Motif Mining



(a) 2IGD Structure (b) 2IGD Graph

FIGURE 1.1: (a) 3D structure for protein G (PDB file 2IGD, Length 61). (b) Graph for 2IGD: graph input format (left) and graph representation (right). The input format lists the vertices ('v') followed by the edges ('u'). A vertex is a tuple (v n l), where v denotes a vertex, n gives the vertex number and l its label. An edge is a tuple (u n_1 n_2 l), where u denotes an edge, n_1, n_2 are the two vertices making the edge, and l is the edge label.

We now describe our graph-based structural motif mining approach. We represent each protein as an undirected graph; the vertices correspond to the secondary structures, α-helix or β-strands. The vertex numbers increase in the sequence order from N-terminus to C-terminus. The edges correspond to the contacts between two structures. If two SSEs have no contacts between them, there is no edge between them. Each one of the vertices and edges has a label associated with it. The labels on the vertices correspond to the type of secondary structure, and the labels on the edges correspond to the contact types which are defined by the number of contacts between the two SSEs. We use four types of edges $E1$, $E2$, $E3$ and $E4$ depending on the number of contacts between two structures:

$E1 \in [1,5), E2 \in [5,10), E3 \in [10,15), E4 \in [15,+\infty)$. The graph representation of the protein 2IGD with contact cutoff $7\mathring{A}$ is shown in Figure 1.1. There are 5 secondary structures in 2igd: $\beta 0$, $\beta 1$, $\alpha 2$, $\beta 3$, $\beta 4$ and 7 edges between $\beta 1 - \beta 0$, $\alpha 2 - \beta 0$, $\alpha 2 - \beta 1$, $\beta 3 - \beta 1$, $\beta 3 - \beta 2$, $\beta 4 - \beta 2$, and $\beta 4 - \beta 3$.

Discovering Frequent Subgraphs

To mine the frequent structural motifs from the database of proteins represented as graphs we used the FSG algorithm for frequent subgraph discovery[44]. The input is a database D of protein graphs and a minimum support threshold σ. The output is the set of all connected subgraphs that occur in at least $\sigma\%$ of the proteins.

There are three considerations when FSG are applied to the database of protein structures. Firstly, we are interested in subgraphs that are composed of all secondary structures in contact, whether or not they are consecutive. A protein sequence or subsequence is naturally connected from N-terminal to C-terminal, thus any two successive structures are connected with each other. Contacts between two non-successive structures can form tertiary structure as well. Secondly, we used labeled graphs, where different graphs can contain vertices and edges with the same label. We classified edges in four types according to the range of contacts. We chose this classification range heuristically to allow us to find patterns containing multiple occurrences of the same structures and contact ranges. A finer classification of edge types will decrease the frequency of common patterns and make the running time extremely slow. Thirdly, we are interested in the subgraphs with at least $\sigma\%$ frequency in the database. This makes sure that the generated subgraphs are the dominant motifs.

FSG uses a level-by-level approach to mine the connected subgraphs. It starts with the single edges and counts their frequency. Those that meet the support threshold are extended by another frequent edge to obtain a set of candidate edge pairs. The counting proceeds by extending a frequent edge-set by one more edge. The process stops when no frequent extension is possible. For a detailed description of FSG see [44].

We ran the FSG algorithm on a dataset of 757 protein graphs obtained from the non-redundant proteins in PDBselect database [27]. The results with different frequency thresholds ranging from 10% to 40%, and with contact cutoff $7\mathring{A}$ are shown in Table 1.2. Figure 1.3 shows some mined subgraphs with different number of edges using support 10%.

We mapped the mined graph patterns back to the PDB structure to visualize the mined results. Two such frequent tertiary motifs are shown in Figure 1.4. The top one has 6 edges, and frequency 157. This motif shows four alpha helices and three beta strands. Two proteins where this pattern occurs are also shown: PDB files $1AD2$ and $1KUH$. The bottom motif has 8 edges and frequency 153. It is an all alpha motif. Two occurrences in PDB files $1BG8$ and $1ARV$ are shown. The results obtained from graph mining are quite encouraging. We found that most of the highest scoring patterns match part of the whole protein structures, and we were able to find remote interactions as well.

Comparison with SCOP Database

The SCOP protein database[50], categorizes proteins into all-α, all-β, α and β (interspersed), α plus β (isolated), and multi-domain according to their structure and functional similarity. We applied the graph-mining method to protein families classified in SCOP database. In order to find out whether our method matches with SCOP in mining and categorizing common domains, several protein families were chosen randomly from SCOP and represented as graphs. Within one protein family, we used 100% support to find the largest frequent patterns that appear in every protein in that family.

	10%	20%	30%	40%
1-edge graphs	12	10	8	4
2-edges graphs	34	23	8	6
3-edges graphs	144	42	17	6
4-edges graphs	420	72	22	2
5-edges graphs	1198	142	23	0
6-edges graphs	2920	289	4	0
7-edges graphs	6816	32	0	0
8-edges graphs	14935	114	0	0

FIGURE 1.2: Frequent subgraphs

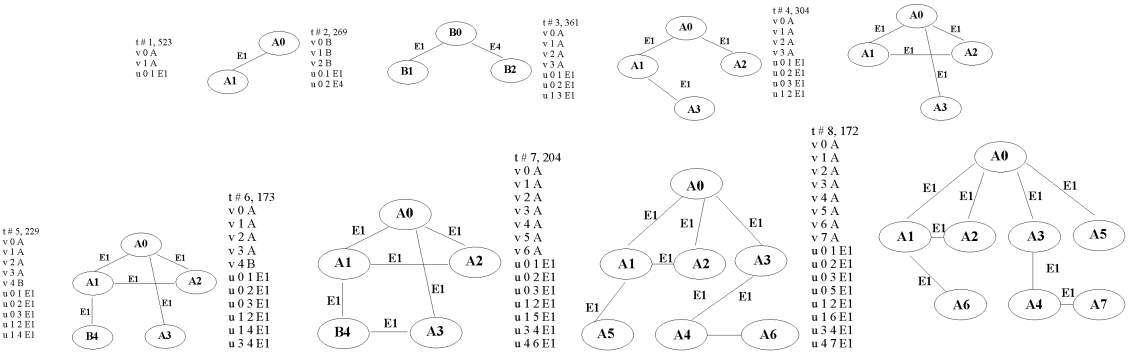


FIGURE 1.3: Highest Frequency Tertiary Motifs:Edge Size 1 to 8. The notation "t #1, 523" in the graph input format denotes edge size = 1, support = 523.

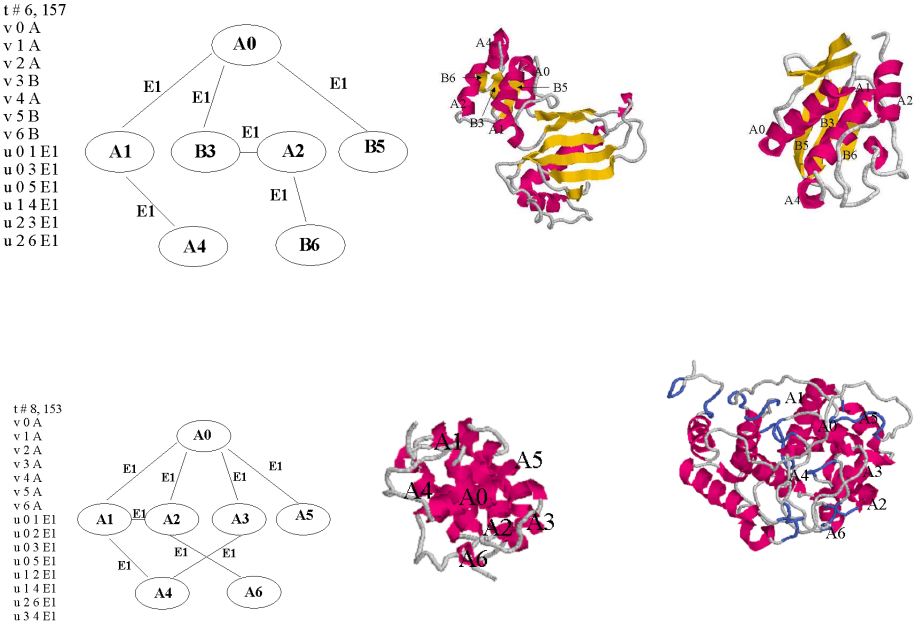


FIGURE 1.4: Top: Alpha-Beta Motif and its Occurrence in PDB files: 1ad2, 1kuh. Bottom: All Alpha Motif and its Occurrence in PDB files: 1bg8, 1arv

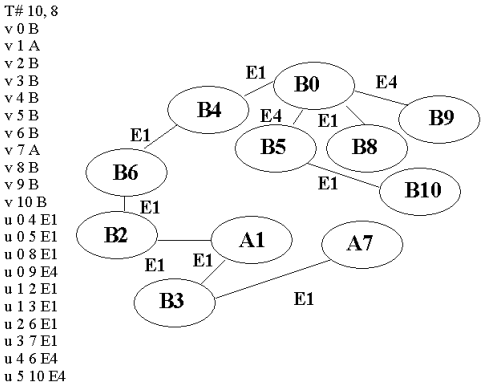
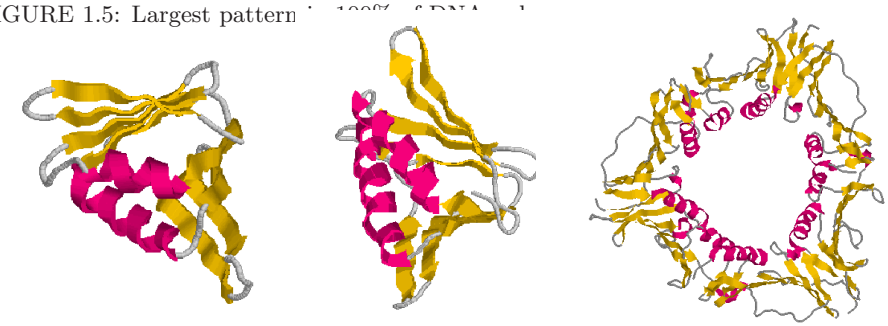


FIGURE 1.5: Largest pattern [illegible] of DNA [illegible]



(a) 1PLQ (b) 1CE8 (c) 1CZD

FIGURE 1.6: Some occurrences of the pattern in PDB files: 1PLQ, 1CE8, 1CZD

To test retrieval rate, we can create a mixed database from several families within the same superfamily. Mining this database, frequent patterns, could only be found with less than 100% support. If we add to the database some other proteins outside of the superfamily, the maximum size pattern was only found at very small support and it is not frequent any more. This demonstrates that our graph mining method could be used to classify proteins into families based on their shared motifs. For example, consider the DNA clamp superfamily, which is made up of two families: DNA polymerase III, beta subunit and DNA polymerase processivity factor. DNA polymerase III beta subunit family has three E. coli proteins: 2POL, 1JQL and 1JQJ. The DNA polymerase processivity factors family has eight proteins: Bacteriophage RB69 (1B77, 1B8H) and Bacteriophage T4 (1CZD), human herpes virus type 1 (1DML), proliferating cell nuclear antigens (1PLQ, 1PLR, 1AXC, 1GE8). The following pattern, shown in Figure 1.5 and its corresponding occurrences in PDB files, shown in Figure 1.6, appears in every protein of DNA polymerase processivity factors. The graph-based method has the potential to become a powerful tool in structure analysis. It is a rapid, intuitive method to identify and find proteins that display structure similarities at the level of residues and secondary structure elements.

1.3.3 Alternate Approaches for Structural Motif Mining

Another graph based approach to motif discovery was proposed in [29], where they search for frequent subgraphs in proteins belonging to the same structural and functional family in the SCOP database; they propose these subgraphs as family-specific amino acid residue

signatures of the underlying family. They represent the protein structures as graphs, using CD, DT and AD representations (see section 1.3.1). They represent each protein graph by an adjacency matrix in which each entry is either a 0 (if there is no edge), a vertex label (if it is a diagonal element), or an edge label (if an edge exists). They define the code of this adjacency matrix as the sequence of the entries in the lower triangular matrix read going left to right and then top to bottom. They use the lexicographic order to impose an ordering on the codes obtained by rearranging the rows of the matrix. Using a novel graph representation such as this, they construct a rooted, ordered tree for each graph and then search for frequent graphs. They then discard those which have a low mutual information score. They conclude that in order to achieve the highest accuracy in finding protein family specific signatures, AD graphs present the best choice both due to their relative computational efficiency and their robustness in taking into account possible experimental errors in determining protein atomic coordinates.

In a different approach, Jonassen et al. [35] represent the neighborhood of each residue r as a neighbor string NS_r of which r is called the *anchor*. NS_r encodes all residues within a d angstrom radius of r. Each residue is encoded by its amino acid type, secondary structure type and a coordinate set (x, y, z) calculated as the mean of the r's side chain atoms. The order of residues in NS_r is governed by their order along the protein's backbone. They then define a packing pattern against which the neighbor strings are matched. A packing pattern consists of a list of elements where each element defines a match set (a set of allowed amino acids), a set of allowed SSE types and one set of coordinates. $NS_r = r_1, \ldots, r_k, \ldots, r_l$, where r_k is the anchor residue, is said to match packing pattern $P = p_1, \ldots, p_l, \ldots, p_n$ if NS_r contains a subsequence $r_{i1}, \ldots, r_{in}$, such that residues have amino acids and SSE types included in the match sets of the corresponding pattern elements and the anchors of the neighbor string and pattern string are aligned. Also, NS_r is said to structurally match P within ϕ, if it is possible to superimpose the coordinates of NS_r onto those of P with a root mean square deviation of maximum ϕ. A neighbor string that structurally matches a packing pattern with a threshold ϕ describes an occurrence of the pattern. A pattern having occurrences in k structures is said to have support k. Thus, they seek packing patterns having support k in a dataset of N structures. For each of the N structures, a packing pattern is generated for the generalization of each neighbor string. There may exist many generalizations of the match sets and hence pruning based on geometrical constraints is used to constrain the length of the neighbor strings. The packing pattern inherits its coordinates and SSE types from the neighbor string. Further, neighbor strings with fewer than 4 elements are discarded. Using depth-first search, they search for all generalizations of a neighbor string having support k moving from simple (short generalizations) to complex (long ones) as the depth of the search increases. For every pattern thus found, they compute a score measuring the pattern's information content divided by its maximum root mean square deviation.

The 3D conformation of a protein may be compactly represented in a symmetrical, square, boolean matrix of pairwise, inter-residue contacts, or "contact map". The contact map provides a host of useful information about the protein's structure. In [28] Hu et al. describe how data mining can be used to extract valuable information from contact maps. For example, clusters of contacts represent certain secondary structures, and also capture non-local interactions, giving clues to the tertiary structure. They focus on two main tasks: 1) Given the database of protein sequences, discover an extensive set of non-local (frequent) dense patterns in their contact maps, and compile a library of such non-local interactions. 2) Cluster these patterns based on their similarities and evaluate the clustering quality. To enumerate all the frequent dense patterns they scan the database of contact maps with a 2D sliding window of a user specified size $W \times W$. Across all proteins in the database,

any sub-matrix under the window that has a minimum "density" (the number of contacts) is captured. The main complexity of the method stems from the fact that there can be a huge number of candidate windows. Of these windows only relatively few will be dense, since the number of contacts is a lot less than the number of non-contacts. They propose a fast hash-based approach to counting the frequency of all dense patterns. Finally they use agglomerative clustering to group the mined dense patterns to find the dominant non-local interactions.

Several methods for secondary level motif finding have also been proposed. SPASM can find the motifs consisting of arbitrary main-chain and/or side-chains in a protein database[40]. An algorithm based on subgraph isomorphism was proposed in[49]; it searches for an exact match of a specific pattern in a database. Search for distantly related proteins using a graph to represent the helices and strands was proposed in[42]. An approach based on maximally common substructures between two proteins was proposed in[23]; it also highlights areas of structural overlap between proteins. SUBDUE [15] is an approach based on Minimum Description Length and inexact graph matching for finding patterns in proteins. Another graph based method for structure discovery, based on geometric hashing, was presented in [61]. Most of these methods either focus on identifying predefined patterns in a group of proteins, or find approximate/inexact matches.

1.3.4 Finding Sites of Non-bonded Interaction

Sites of non-bonded interaction are extremely important to find, as they cannot be determined from primary structure and give clues about the higher-order structure of the protein. Side chain clusters in proteins aid in protein folding and in stabilizing the three-dimensional structure of proteins [25]. Also, these sites may be occurring in structurally similar proteins with very low sequence homology [37]. Spectral methods have been gaining favor in finding clusters of non-bonded interaction as they use global information of non-bonded interactions in the protein molecule to identify clusters.

In [11], Brinda et al. construct a graph, where each vertex is a residue and they connect vertices if they are in contact (using cut-off $4.5\mathring{A}$). They represent the protein graph in terms of an adjacency matrix A, where $a_{p,q} = 1/d_{p,q}$, if $p, q \in V(G)$ are connected and 1/100, otherwise, $d_{p,q}$ = distance between p and q. The degree matrix, D, is a diagonal matrix obtained by summing up the elements of each column. Then, the Laplacian matrix, $L = D - A$, is of dimension $|V| \times |V|$. Using eigen decomposition on L, they get the eigenvalues and the eigenvectors. The Fiedler eigenvector, corresponding to the second lowest eigenvalue, gives the clustering information [24]. The centers of the clusters can be identified from the eigenvectors of the top eigenvalues. The cluster centers identified correspond to the nodes with the highest connectivity (degree) in the cluster, which generally correspond to the geometric center of the cluster. They consider clusters with at least three residues. The residues with the same vector component in the second lowest eigenvalue form a cluster. The residue with the highest magnitude of a vector component in the corresponding top eigenvalue is the center of the cluster.

The α/β barrel proteins are known to adopt the same fold in spite of very low sequence similarity. This could be possible only if the specific stabilizing interactions important in maintaining the fold are conserved in topologically equivalent positions. Such stabilization centers are usually identified by determining the extent of long-range contacts made by the residue in the native structure. In [36], Kannan et al. use a data set of 36 (α/β) barrel proteins having average pair-wise sequence identity less than 10%. They represent each protein by a connected graph using contact distance approach to connect vertices/residues with $\delta = 6.5\mathring{A}$. The representation chosen causes high connectivity among the vertices of

the graph and hence operating on the Laplacian of the graph is ineffective. Hence they operate on the adjacency matrix corresponding to the proteins. On eigen decomposition, they get a set of eigenvalue-eigenvector pairs. They sort this set based on the eigenvalues. They consider the set of largest eigenvalue-eigenvector pairs. Those residues having a large vector component magnitude in the direction of any of these eigenvectors are believed to belong to the cluster corresponding to that eigenvector. Thus, they cluster the residues. In each cluster, the residue having largest vector component magnitude in the direction of the corresponding eigenvector is the center of that cluster. Using eigenvalue analysis, they infer the degree of each vertex. The residues with the largest degree (typically the cluster centers) correspond to the stabilization centers. They found that most of the residues grouped in clusters corresponding to the higher eigenvalues, typically occur in the strand regions forming the β barrel and were found to be topologically conserved in all 36 proteins studied.

1.4 Mining Microarray Gene Expression Data

High-throughput gene expression has become an important tool to study transcriptional activity in a variety of biological samples. To interpret experimental data, the extent and diversity of gene expression for the system under study should be well characterized. A microarray [59] is a small chip (made of chemically coated glass, nylon membrane or silicon), containing a (usually rectangular) grid into which tens of thousands of DNA molecules (probes) are attached. Each cell of the grid relates to a fragment of DNA sequence.

Typically, two mRNA samples (a test sample and a control sample) are reverse-transcribed into cDNA (targets) and labeled using either fluorescent dyes or radioactive isotopes. They are then hybridized by base-pairing, with the probes on the surface of the chip. The chip is then scanned to read the signal intensity that is emitted from the labeled and hybridized targets. The ratio of the signal intensity emitted from the test target to that emitted from the control target is a measure of the gene expressivity of the test target with respect to the control target.

Typically, each row in the microarray grid corresponds to a single gene and each column, to either an experiment the gene is subjected to, or a patient the gene is taken from. The corresponding gene expression values can be represented as a matrix of real values where the entry (i, j) corresponds to the gene expression value for gene i under experiment j or from patient j. Formally, the dataset is represented as $\mathbf{Y} = \{y(i, j) \in \Re^+ | 1 \leq i \leq n, 1 \leq j \leq m\}$ where n, m are the number of rows(genes) and columns(experiments) and $\Re^+$ is the set of positive real numbers. We represent the vector of expression values corresponding to gene i by $y(i, \cdot)$ and that corresponding to experiment/patient j by $y(\cdot, j)$. A microarray dataset is typically tens of thousands of rows (genes) long and as many as one hundred columns (experiments/patients) wide. It is also possible to think of microarray data as the transpose of $\mathbf{Y}$, i.e., where the rows are experiments and the columns are genes.

The typical objectives of a microarray experiment is to:

1. Identify candidate genes or pathological pathways: We can conduct a microarray experiment, where the control sample is from a normal tissue while the test sample is from a disease tissue. The over-expressed or under-expressed genes identified in such an experiment may be relevant to the disease. Alternatively, a gene A, whose function is unknown, may be similarly expressed with respect to another gene B, whose function is known. This may indicate A has a function similar to that of B.

2. Discovery and prediction of disease classes: We can conduct a microarray experiment using genes from patients known to be afflicted with a particular disease and cluster their corresponding gene expression data to discover previously unknown classes or stages of the disease. This can aid in disease detection and treatment.

1.4.1 Challenges of Mining Microarray Data

One of the main challenges of mining microarray data is the high dimensionality of the dataset. This is due to the inherent sparsity of high-dimensional space. It has been proven [8], that under certain reasonable assumptions on the data distribution and for a variety of distance functions, the ratio of the distances of the nearest and farthest points to a given point in a high-dimensional dataset, is almost 1. The process of finding the nearest point to a given point is instrumental in the success of algorithms, used to achieve the objectives mentioned above. For example, in clustering, it is imperative that there is an acceptable contrast in distances between points within the same cluster and distances between points in different clusters.

Another difficulty in mining microarray data arises from the fact that there are often missing or corrupted values in the microarray matrix, due to problems in hybridization or in reading the expression values. Finally, microarray data is very noisy. A large number of the genes or experiments may not contribute any interesting information, but their presence in the dataset makes detection of subtle clusters and patterns harder and increases the motivation for highly scalable algorithms.

1.4.2 Association Rule Mining

In order to achieve the objective of discovery of candidate genes in pathological pathways, there has been an attempt to use association rules [2]. Association rules can describe how the expression of one gene may be associated with the expression of a set of genes. Given that such an association exists, one might easily infer that the genes involved participate in some type of gene network.

In order to apply association mining, it is necessary that the data be nominal. In [16], Creighton et al. first discretize the microarray data, so that each $y(a, b)$ is set to $\{high,\ low,\ moderate\}$ depending on whether it is up-regulated, down-regulated and neither considerably up nor down regulated, respectively. Then, the data corresponding to each experiment, i.e., $(y(\cdot\,, a))$ can be thought of as a transaction from the market-basket viewpoint. They then apply the standard Apriori algorithm to find the association rules between the different genes and their expression levels. This yields rules of the form $g_1(\uparrow) \wedge g_3(\downarrow) \Rightarrow g_2(\uparrow)$, which means that if gene g_1 is highly expressed and g_3 is under-expressed, then g_2 is over-expressed. Such expression rules can provide useful insight into the expression networks.

1.4.3 Clustering Algorithms

In order to achieve the objective of discovery of disease classes, there are three types of clustering algorithms:

1. Gene-based clustering: the dataset is partitioned into groups of genes having similar expression values across all the patients/experiments.

2. Experiment-based clustering: the dataset is partitioned into groups of experiments having similar expression values across all the genes.
3. Subspace clustering: the dataset is partitioned into groups of genes and experiments having similar expression values.

If the algorithm searches for clusters, which have elements which are similar across all dimensions, it can be called a "full dimensional" one.

Similarity Measures

Before delving into the specific clustering algorithms used, we must discuss the measures used to express similarity between the expression values. Although, formulae mentioned in this section describe similarity between rows of the gene expression matrix, they can, without loss of generality be applied, to describe similarity between the columns of the gene expression matrix as well.

One of the most used classes of distance metrics is the L_p distance where

$$||y(a,\cdot) - y(b,\cdot)||_{p\in\Re^+} = \left(\sum_{t=1}^{m} |y(a,t) - y(b,t)|^p\right)^{1/p}$$

In this family, the L_1, L_2 and L_∞ metrics, also called the Manhattan, Euclidean and Chebyshev distance metrics, respectively, are the most studied; the Euclidean distance metric is the most commonly used [21, 47, 12]. However, these measures do not perform too well in high-dimensional spaces. Note that this class of metrics treats all dimensions equally, irrespective of their distribution. This is remedied to some extent, by standardizing the data, i.e. normalizing the data in each row to the range [0,1] having mean 0 and standard deviation 1 [17, 58]. Note that standardization assumes the underlying data is multivariate normal.

Alternatively, researchers [18, 19, 12, 63] have used Pearson's correlation coefficient as a similarity measure, where

$$r(y(a,\cdot), y(b,\cdot)) = \frac{\sum_{t=1}^{m}(y(a,t) - \mu(y(a,\cdot)))(y(b,t) - \mu(y(b,\cdot)))}{\sqrt{(\sum_{t=1}^{m}(y(a,t) - \mu(y(a,\cdot)))^2)(\sum_{t=1}^{m}(y(b,t) - \mu(y(b,\cdot)))^2)}}$$

where $\mu(y(a,\cdot)) = \sum_{t=1}^{m} \frac{y(a,t)}{N}$ and $\mu(y(b,\cdot)) = \sum_{t=1}^{m} \frac{y(b,t)}{N}$.

Note that r assumes the two vectors are approximately normally distributed and jointly bivariate normal. In this case there is a strong relationship between r and standardized Euclidean distance, since if $y(a,\cdot)$ and $y(b,\cdot)$ are standardized,

$$||y(a,\cdot) - y(b,\cdot)||_2 = \sqrt{2m(1 - r(y(a,\cdot), y(b,\cdot)))}$$

If $\mu(y(a,\cdot)) = \mu(y(b,\cdot)) = 0$, i.e. the vectors are translated so their mean is 0, then r is identical to the cosine similarity between the vectors, which is a similarity measure known to be highly effective in high-dimensional spaces. Other distance measures tested in microarray data analysis include Kullback-Leibler distance or mutual information [47].

Gene-based Clustering

A number of traditional clustering algorithms like k-means [58] and hierarchical clustering [19] have been used to find full-dimensional clusters in microarray data.

K-means(Y,sim,k)
1. select k points(genes) from Y randomly as cluster centers
2. repeat until convergence
3. for s=1 to n
4. assign $y(s,\cdot)$ to the cluster whose center is most similar to it using sim
5. for j=1 to k
6. recalculate center of cluster c_j as mean of all rows(genes) assigned to it

The k-means algorithm, shown above, is a partitional, iterative clustering algorithm. It takes as input the $n \times m$ dataset $\mathbf{Y}$, the similarity measure sim, and the number of clusters to mine k. K-means runs very quickly in $O(nmt)$ time (t is the number of iterations, and k is a small constant), but suffers from the disadvantage that it requires a parameter k to be supplied, indicating the number of clusters to be found. This parameter is hard to set. Also, k-means is highly sensitive to noise and outliers, because it assigns every point in the dataset to some cluster. K-means converges to a local optima and theoretical guarantees of its accuracy are yet to be proven. The practical accuracy of k-means is noted to improve considerably, if the initial assignment of points to clusters is not so arbitrary [10].

A Self-Organizing Map (SOM)[43] is based on a single-layered neural network which maps vectors(rows/genes) in the microarray data to a two-dimensional grid of output nodes. Each input node is a row from the microarray dataset and each output node corresponds to a vector in the high-dimensional space.

SOM(Y,sim, k, g, r)
1. select k vectors(genes) from Y randomly as output nodes
2. repeat until convergence
3. For s=1 to n
4. find output node most similar to $y(s,\cdot)$ using sim
5. update all nodes in the r-neighborhood of that output node

As shown above, a SOM trains on the input microarray dataset to adjust its output nodes (line 5), so that they move toward the denser regions of the high-dimensional feature space. The algorithm uses a number of user-specified parameters like the learning rate (g) and the neighborhood size(r). Also, SOM converges slower than k-means but is far more robust than k-means. SOM [56] and related algorithms [26] have been used for gene-based clustering.

Hierarchical clustering has two flavors: agglomerative (bottom-up) and divisive (top-bottom). In agglomerative gene-based clustering (shown below), each gene is initially assigned to its own cluster (line 1). In each iteration the two clusters with the highest inter-cluster similarity (line 3) are merged to form a single one (line 4), until some convergence criterion is satisfied e.g., the desired number of clusters remain, only one cluster remains, etc. The inter-cluster similarity may be computed by a number of methods such as:

- single linkage, where $Sim(\mathbf{a},\mathbf{b}) = \max_{i\in\mathbf{a},j\in\mathbf{b}} sim(i,j)$.
- complete linkage, where $Sim(\mathbf{a},\mathbf{b}) = \min_{i\in\mathbf{a},j\in\mathbf{b}} sim(i,j)$.
- average linkage, where $Sim(\mathbf{a},\mathbf{b}) = \frac{\sum_{i\in\mathbf{a},j\in\mathbf{b}} sim(i,j)}{|\mathbf{a}||\mathbf{b}|}$.
- average group linkage, where $Sim(\mathbf{a},\mathbf{b}) = \frac{\sum_{i\in\mathbf{a}\cup\mathbf{b},j\in\mathbf{a}\cup\mathbf{b}} sim(i,j)}{|\mathbf{a}\cup\mathbf{b}|^2}$.

Here Sim is the inter-cluster similarity of clusters a and b, each having $|\mathbf{a}|$ and $|\mathbf{b}|$ genes assigned to them respectively and sim is the similarity measure between two genes. This merging of clusters gives rise to a tree called a dendrogram. This algorithm is greedy and susceptible to noise. Also, it has time complexity $O(n^2 \log n)$ [32] implying it converges slower than K-means.

Agglomerative(Y,Sim)
1. Assign each $y(1, \cdot), i \in [1, n]$ to its own cluster to form set of clusters C
2. repeat until convergence
3. $\{\mathbf{a}^*, \mathbf{b}^*\} = argmin_{(\mathbf{a},\mathbf{b})\in\mathbf{C}\times\mathbf{C}} Sim(\mathbf{a}, \mathbf{b})$
4. $\mathbf{a}^* = \mathbf{a}^* \cup \mathbf{b}^*, \mathbf{C} = \mathbf{C} \backslash \mathbf{b}^*$

In divisive clustering, all the genes are initially assigned to the same cluster. Then iteratively, one of the clusters is selected from those existing, and is split into two clusters based on some splitting criterion. This continues, until some convergence criterion is achieved e.g., each gene has its own cluster, desired number of clusters remain, etc.

Experiment-based Clustering

For experiment-based clustering, the dimensionality (i.e. the number of genes) of the space is extremely high. Solutions proposed to remedy the failure of distance metrics in such high-dimensional spaces include, designing new distance metrics [1] and dimensionality reduction [4, 52]. Dimension reduction techniques, such as the Karhunen-Loeve tranformation (KLT), and singular value decomposition (SVD) [4, 52] are applied as a preprocessing step to reduce the number of dimensions prior to the application of a clustering algorithm. In such dimension reduction techniques, the entire database is projected onto a smaller set of new dimensions, called the principal components, which account for a large portion of the variance in the dataset. These new dimensions are mutually uncorrelated and orthogonal. Each of them is a linear combination of the underlying dimensions. Once the data has been projected to a lower-dimensional space, any of the clustering algorithms can be applied.

Subspace Clustering

The strategy of dimension reduction using KLT may be inappropriate as the clusters involving the transformed dimensions may be hard to interpret for the user. Also, data is only clustered in a single subspace. [2] cites an example, in which KLT does not reduce the dimensionality without trading off considerable information, as the dataset contains subsets of points which lie in different and sometimes overlapping lower dimensional subspaces. Hence, the focus of much recent microarray data analysis focuses on subspace clustering [6, 14, 21, 45, 57, 60, 63].

Ben-Dor et al.[6], seek to identify large order-preserving submatrices (OPSMs) in Y. A submatrix is order-preserving if there is a permutation of its columns under which the sequence of values in every row is strictly increasing. They prove that the problem is NP-hard. In that paper, they discuss two methods to discover clusters: a complete model and a partial model. The complete model is simply to enumerate every combination, which is unacceptable in reality. The partial model is a stochastic model.

Getz et al.[21] alternate between gene-based and experiment-based clustering. They cluster using super-paramagnetic clustering (SPC), a divisive hierarchical clustering based on the analogy to the physics of inhomogeneous ferromagnets [9], which is robust against noise and searches for "natural" stable clusters. They partition the microarray dataset into

gene-based and experiment-based clusters. The gene(experiment)-based clusters specify the group of genes(experiment) which are similar. They then cluster the set of genes reported by each gene-based clustering over the set of experiments specified by each experiment-based cluster. This continues until SPC clustering produces no new robust clusters.

Cheng et al.[14] proposed the biclustering algorithm which seeks to group subsets of microarray data rows and columns having a high similarity score. They use the mean squared residue score as a similarity measure for a cluster. If $O \subseteq [1,n], C \subseteq [1,m]$, the mean square residue of (O,C) is defined as :

$$H(O,C) = \frac{1}{|O||C|} \sum_{i \in O, j \in C} (y(i,j) - \mu_C(y(i,\cdot)) - \mu_O(y(\cdot,j)) + \mu_{O,C}(y(\cdot,\cdot)))$$

where, $\mu_C(y(i,\cdot)) = \frac{1}{|C|}\sum_{j\in C} y(i,j)$, $\mu_O(y(\cdot,j) = \frac{1}{|O|}\sum_{i\in O} y(i,j)$, $\mu_{O,C}(y(\cdot,\cdot)) = \frac{1}{|O||C|}\sum_{i\in O, j\in C} y(i,j)$. If $H(O,C) \leq \delta$, a user-specified threshold, the cluster (O,C) is retained. This definition imposes a constraint on the variation of gene expression values in the genes in O across the experiments in C. Their algorithm aims to greedily find multiple δ-biclusters, one in each iteration. After discovering a cluster, the values in that cluster are replaced by random data, so that partially overlapping biclusters may be found. However, if clusters naturally overlap, such random data may obstruct the detection of the overlap [60].

The p-clustering algorithm [60] is designed to solve this problem. Unlike biclustering, it is a deterministic algorithm. It defines a pScore of a 2×2 matrix $X = \left(\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right)$ as: $\text{pScore}(X) = |(a-b)-(c-d)|$. A sub-matrix (O,C) is a δ-pCluster iff $\forall X_{2\times 2} \in (O,C)$ $\text{pScore}(X) \leq \delta$, a predefined threshold. The intuition is that δ constrains the variation in the gene expression values $((a-b),(c-d))$, across the conditions (columns in X) over which the genes (rows in X) cluster. They use the implicit recursivity in the definition of a δ-pCluster to design a depth-first algorithm that searches for maximal δ-pClusters. The pCluster algorithm can discover every δ-pCluster in a data set, no matter whether the clusters are overlapping or not. It has time complexity $O(nm^2log(n) + mn^2log(m))$.

1.5 Determining Normal Variation in Gene Expression Data

Having looked at the application of data mining to the problems of motif discovery and microarray gene expression analysis, we now highlight some issues in data pre-processing. We illustrate this with the problem of finding normal variation in gene expression data.

An ubiquitous and under-appreciated problem in microarray analysis is the incidence of microarrays reporting non-equivalent levels of an mRNA or the expression of a gene for a system under replicate experimental conditions. In ideal conditions, the gene expression values for each gene should be the same across all array experiments. But due to the technical limitations the data contains lot of inherent noise, which could also be due to normal variation in expression of the genes across the genetically identical male mice. Our goal is to extract those genes which are contributing to the noise due to their biological variance. This kind of analysis should be done prior to mining, since otherwise, we might come to a wrong conclusion about co-expressed genes or genes that correlate well with a pathological condition.

We try to capture the genes which show variance among the identical mice by trying to eliminate the variations which come in due to experimental errors and fluctuations. We use a very robust method to exclude genes, which would eliminate any considerable variance in the replicates. Our approach is based on the following steps: 1) Calculation of fold-change

ratio and discretization of expression levels for each gene, 2) Elimination of experimental noise, 3) Constructing an expression profile for each gene, and 4) Calculating and raking by gene variability via entropy calculation. We describe the steps in more detail below.

1.5.1 Fold-Change Ratio

We assume that we have n genes, in m mice, with r replicates for each mice, for a given tissue. We denote gene i as g^i. Let S_t^i denote the expression level for gene g^i in the test sample and S_r^i the expression of g^i in the reference microarray samples. We define fold-change ratio as the log-odds ratio of the expression intensities of the test sample over the reference sample, given as $\log_2(\frac{S_t^i}{S_r^i})$. To analyze the variability, we discretize the fold-change into k bins ranging from very low expression levels to very high expression levels. The data is normalized in such a way that the median of the deviation from the median was set to the same value for the distribution of all the log-ratios on each array [51]. Similar analysis was done for the Affymetrix data. The raw data containing the intensities was median centered and scaled by the standard deviation. This normalization technique was chosen after experimenting with other methods like linear regression and mean centering. Though none of these methods yielded a normal distribution for the histogram plot of the gene expression values of all clones in a sample, the median centered normalization technique performed the best and also provided a uniform distribution for our binning method.

	Rep1	Rep2	Rep3	Rep4
Mouse1	$\{g^1_{VH}, g^2_{VL}, g^3_{VH}, g^4_L\}$	$\{g^1_{VH}, g^2_{VL}, g^3_{VH}, g^4_N\}$	$\{g^1_{VH}, g^2_{VL}, g^3_{VH}, g^4_N\}$	$\{g^1_{VH}, g^2_{VL}, g^3_{VH}, g^4_N\}$
Mouse2	$\{g^1_{VH}, g^2_{VL}, g^3_L, g^4_N\}$	$\{g^1_{VH}, g^2_{VL}, g^3_L, g^4_N\}$	$\{g^1_{VH}, g^2_{VL}, g^3_L, g^4_N\}$	$\{g^1_{VH}, g^2_{VL}, g^3_H, g^4_N\}$
Mouse3	$\{g^1_{VH}, g^2_N, g^3_{VH}, g^4_N\}$	$\{g^1_{VH}, g^2_N, g^3_{VH}, g^4_N\}$	$\{g^1_{VH}, g^2_N, g^3_{VH}, g^4_N\}$	$\{g^1_{VH}, g^2_N, g^3_{VH}, g^4_L\}$
Mouse4	$\{g^1_{VH}, g^2_N, g^3_{VL}, g^4_L\}$	$\{g^1_{VH}, g^2_N, g^3_{VL}, g^4_N\}$	$\{g^1_{VH}, g^2_N, g^3_{VL}, g^4_N\}$	$\{g^1_{VH}, g^2_N, g^3_{VL}, g^4_N\}$
Mouse5	$\{g^1_{VH}, g^2_H, g^3_L, g^4_L\}$	$\{g^1_{VH}, g^2_H, g^3_L, g^4_L\}$	$\{g^1_{VH}, g^2_H, g^3_L, g^4_L\}$	$\{g^1_{VH}, g^2_H, g^3_L, g^4_L\}$
Mouse6	$\{g^1_{VH}, g^2_H, g^3_{VH}, g^4_L\}$	$\{g^1_{VH}, g^2_H, g^3_{VH}, g^4_L\}$	$\{g^1_{VH}, g^2_H, g^3_{VH}, g^4_N\}$	$\{g^1_{VH}, g^2_H, g^3_{VH}, g^4_N\}$

If the number of bins for expression level discretization is too small or too high, then it leads to problems in analysis. In coarse binning, the information about the values is ignored, and in a very fine binning, the patterns are lost. We tried several values of k and found that $k = 5$ works well. The bin intervals are determined using the uniform frequency binning method. Other popular methods like discriminant discretization, boolean reasoning based and entropy based discretization can be considered [48]. In frequency binning method we discretized the relative expression (fold-change) into 5 levels depending on their expression value. The values of -1.5, -0.5, 0, 0.5, and 1.5 for the fold change ratio were taken as thresholds for very low (VL), low (L), normal (N), high (H), and very high (VH) expression, respectively. That is, $VL \in (-\infty, -1.5]$, $V \in (-1.5, -0.5]$, $N \in (-0.5, 0.5)$, $H \in [0.5, 1.5)$, and $VH \in [1.5, +\infty)$. We use the notation g_e^i to denote the expression level for gene g^i in a given replicate, where $e \in \{VL, L, N, H, VH\}$. Table 1.1 shows an example of the expression of 4 genes in six mice with 4 array replicates for each mouse.

1.5.2 Elimination of Experimental Noise

In order to eliminate the noise due to experimental fluctuations, we process the data taking one mouse at a time. For each mouse the genetic expression signature is obtained and compared across all r replicates. Only those genes which show consistent expression signature in

all r replicates are chosen and the ones which show even a slight deviation in any of the replicates are eliminated. This methodology takes a very stringent approach toward eliminating even the slightest errors due to technical noise. One shortcoming of this approach is that it would not eliminate any genes which show high fluctuations in the range $(-0.5, 0.5)$. In our study of normal variance to identify genes which have been falsely reported as differentially expressed, the genes which we might fail to eliminate do not contribute to the databank anyway, because they lie in the normal expression range. So, our approach would eliminate most of the noise which comes due to technical/experimental issues. This operation is done on all m mice, as a result of which we have gene expression signatures in all the mice with minimal experimental noise.

	Gene Expression
Mouse1 (F_1)	$\{g^1_{VH}, g^2_{VL}, g^3_{VH}\}$
Mouse2 (F_2)	$\{g^1_{VH}, g^2_{VL}, g^4_{N}\}$
Mouse3 (F_3)	$\{g^1_{VH}, g^2_{N}, g^3_{VH}\}$
Mouse4 (F_4)	$\{g^1_{VH}, g^2_{N}, g^3_{VL}\}$
Mouse5 (F_5)	$\{g^1_{VH}, g^2_{H}, g^3_{L}, g^4_{L}\}$
Mouse6 (F_6)	$\{g^1_{VH}, g^2_{H}, g^3_{VH}\}$

Table 1.2, illustrates this process on our example data. For example consider Mouse1. Since gene g^4 is differentially expressed as L in replicate 1, but as N in the other three replicates, we eliminate g^4 from further consideration. The resulting expression signatures for Mouse1 and other mice from our example are shown in Table 1.2.

1.5.3 Gene Expression Profile

Let F_j represent the gene expressions of the j-th mice after the elimination of experimental noise. The F_j's contain the expression level (very high, high, normal, low, very low) information of each gene in each of the m mice in our example. Some values could be missing due to elimination in the first stage. The F_j values, for our example of six mice, are shown in Table 1.2.

From the F_j values we construct a frequency table, which contains the number of occurrences of each gene for each discretized expression level (VH, H, N, L, VL). The frequency of every distinct (gene g^i, expression level e) pair across all F_j, is used to populate the frequency table. The frequency for gene g^i and expression level e is given as $f^i_e = \sum_{j=1}^{m} \delta^i_e(j)$, where m is the number of mice, and $\delta^i_e(j)$ is a characteristic function that notes the presence/absence of gene g^i at level e in mouse j, defined as: $\delta^i_e(j) = 1$, if $g^i_e \in F_j$, and $\delta^i_e(j) = 0$, if $g^i_e \notin F_j$. The frequency table obtained for our example is shown in Table 1.3. As an example, g^2, has expression level VL in mice 1 and 2, level N in mice 3 and 4, and level H in mice 5 and 6. Thus the expression profile for g^2 is given by the vector $(0, 2, 2, 0, 2)$, as shown in the table.

	f^i_{VH}	f^i_H	f^i_N	f^i_L	f^i_{VL}
Gene g^1	6	0	0	0	0
Gene g^2	0	2	2	0	2
Gene g^3	3	0	0	1	1
Gene g^4	0	0	1	1	0

1.5.4 Entropy-based Variability Ranking

The genes that show presence in more than one discrete level are of interest to us. The frequency table is analyzed further to identify those genes which show considerable variance by their presence in more than one state. To capture the variance in a gene's expression level, the entropy measure was used. Entropy gives us the amount of disorder in the expression values of a gene, and thus is a measure of the normal variance, since the noise due to experimental variation is eliminated prior to this step. The entropy measure for a gene g^i is given as follows, $E(g^i) = -\sum_{e=1}^{k} p_e^i \log_2(p_e^i)$, where k is the number of discrete expression levels, and p_e^i is the probability of gene g^i having expression level e, which is given as $p_e^i = \frac{f_e^i}{\sum_{j=1}^{k} f_j^i}$.

By definition of entropy, if a gene has only one expression level (say j), then $p_j^i = 1$ and $E(g^i) = 0$. On the other hand, if a gene has the most variance (i.e., equal occurrence at each expression level), then $P_j^i = 1/k$ for all expression levels j, and $E(g^i) = -\sum_{j=1}^{k} 1/k \log_2(1/k) = -\log_2(1/k) = \log_2(k)$. In our approach genes with entropy 0, i.e., those having no variance in expression across the mice, are discarded, and the remaining genes are ranked in descending order of their entropy (and thus variance). The entropy ranking for the four genes (along with the probability of each expression level) in our example are shown in Table 1.4. Gene g^2 and g^3 are of most interest to us because they show variation in expression states across the six mice. On the other hand gene g^1 is always high in all six mice, showing no variance.

	p^i_{VH}	p^i_H	p^i_N	p^i_L	p^i_{VL}	Entropy
Gene g^2	0	0.33	0.33	0	0.33	1.59
Gene g^3	0.6	0	0	0.2	0.2	1.37
Gene g^4	0	0	0.5	0.5	0	1
Gene g^1	1.0	0	0	0	0	0

1.5.5 Weighted Expression Profiles

In our approach to experimental noise elimination, any gene with varying expression level among the replicates is considered experimental noise, and eliminated. Instead of such a stringent approach, we can choose to retain a gene provided it has the same expression level in a given fraction of the replicates. For instance, gene g^4 has expression level N in three out of the four replicates for Mouse1. If we set our threshold to 75%, then we would retain g^4_N in the gene expression for Mouse1 in Table 1.2.

Another approach is to construct a weighted expression signature, as follows: For every gene we record the fraction of the replicates in which it takes a particular value. For instance, for Mouse1, gene g^1 always takes the value VH, so its weighted expression is $g^1_{VH(1.0)}$. On the other hand, gene g^4 is N in three and L in one out of the four replicates; we record its weighted expression as $g^4_{N(0.75),L(0.25)}$. We denote by $w_e^i(j)$ the weight of gene g^i at expression level e in Mouse j. Table 1.5 shows the weighted expression signatures for all the six mice (note: if the weight is 1.0 we omit the weight; we write g^1_{VH} instead of $g^1_{VH(1.0)}$).

From the weighted gene expressions, we can construct a weighted profile using the approach in Section 1.5.3. The weighted frequency for gene g^i and expression level e is given

	Gene Expression
Mouse1 (F_1)	$\{g^1_{VH}, g^2_{VL}, g^3_{VH}, g^4_{N(0.75),L(0.25)}\}$
Mouse2 (F_2)	$\{g^1_{VH}, g^2_{VL}, g^3_{H(0.25),L(0.75)}, g^4_N\}$
Mouse3 (F_3)	$\{g^1_{VH}, g^2_N, g^3_{VH}, g^4_{N(0.25),L(0.75)}\}$
Mouse4 (F_4)	$\{g^1_{VH}, g^2_N, g^3_{VL}, g^4_{N(0.75),L(0.25)}\}$
Mouse5 (F_5)	$\{g^1_{VH}, g^2_H, g^3_L, g^4_L\}$
Mouse6 (F_6)	$\{g^1_{VH}, g^2_H, g^3_{VH}, g^4_{N(0.5),L(0.5)}\}$

as $f^i_e = \sum_{j=1}^{m} w^i_e(j)$, where m is the number of mice. The weighted frequency table obtained for our example is shown in Table 1.6. As an example, g^4, has expression levels $N(0.75)$ in Mouse1, $N(1.0)$ in Mouse2, $N(0.75)$ in Mouse3 and Mouse4, and $N(0.5)$ in Mouse6. Thus $f^4_N = 0.75 + 1.0 + 2 \times 0.75 + 0.5 = 3.75$, and similarly $f^4_L = 2.25$. Thus the weighted expression profile for g^4 is given by the vector $(0, 0, 3.75, 2.25, 0)$, as shown in the table.

	f^i_{VH}	f^i_H	f^i_N	f^i_L	f^i_{VL}
Gene g^1	6	0	0	0	0
Gene g^2	0	2	2	0	2
Gene g^3	3	0.25	0	1.75	1
Gene g^4	0	0	3.75	2.25	0

From the weighted expression profile, we can derive the entropy-based variability ranking for each gene as shown in Table 1.7. Comparing with Table 1.4, we find that g^3 is ranked higher in terms of variability than g^2, but the overall trend is similar.

	p^i_{VH}	p^i_H	p^i_N	p^i_L	p^i_{VL}	Entropy
Gene g^3	0.5	0.04	0	0.29	0.17	1.64
Gene g^2	0	0.33	0.33	0	0.33	1.59
Gene g^4	0	0	0.62	0.38	0	0.95
Gene g^1	1.0	0	0	0	0	0

1.5.6 Application Study

We applied our entropy-based method to detect normal variance in gene expression for the two datasets taken from [51] and [46]. We used three datasets of kidney, liver and testis provided by [51]. Six genetically identical male C57BL6 mice were used to compare the expression values of 5,406 unique mouse genes. Four separate microarray assays were conducted for each organ from each animal, for a total of 24 arrays per organ. Also the dataset of [46] was used which contained the expression values for three mice across the four organs of liver, heart, lung and brain. Each experiment was replicated three times. Affymetrix oligo-chips were used in these experiments.

Using our entropy-based approach, in kidney tissue around 3.5% of the 3088 genes showed considerable variance across the six mice. As reported by [51] we found several immune modulated and stress responsive genes. In liver tissue, 23 out of 2513 genes show significant variation in their expression levels among the six mice. Out of the 3252 genes analyzed in the testis tissue, 63 showed differential expression levels across the six mice. Importantly,

many of the genes that we found to vary normally have been reported previously to be differentially expressed because of a pathological process or experimental intervention. One recent study used microarrays to investigate the differential gene expression patterns during pre-implantation mouse development [41]. Rpl12 was reported to be differentially expressed while we found it to be normally varying in the testis tissue. PUFA (polyunsaturated fatty acids) feeding can influence Protein Kinase C (PKC) activity [7]. Itrp1 is another gene which has been reported as differentially expressed [30] in papillary thyroid carcinoma, while we found this gene to be normally varying in kidney tissues. Another study investigated the effects of acetaminophen on gene expression in the mouse liver [53]. Eight of the genes reported to differ in response to acetaminophen, including CisH2, and Hsp40, were genes we found to vary normally.

Principal Component Analysis (PCA) [34] is a classical technique to reduce the dimensionality of the data set by transforming to a new set of variables (the principal components). It has been used in the analysis of gene expression studies. Principal components (PC's) are uncorrelated and ordered such that the k-th PC has the k-th largest variance among all PC's. The k-th PC can be interpreted as the direction that maximizes the variation of the projections of the data points such that it is orthogonal to the first $k-1$ PC's. PCA is sometimes applied to reduce the dimensionality of the data set prior to clustering. Using PCA prior to cluster analysis may aid better extraction of the cluster structure in the data set. Since PC's are uncorrelated and ordered, the first few PC's, which contain most of the variations in the data, are usually used in cluster analysis. Unless external information is available, [64] recommend cautious interpretation of any cluster structure observed in the reduced dimensional subspace of the PC's. They observe no clear trend between the number of principal components chosen and the cluster quality. [33] use PCA analysis for extracting tissue specific signatures.

We used PCA to analyze how well the genes we have extracted capture the normal variance between the mice, eliminating the variance due to any other sources to the maximum possible extent. Projection on to a 3-dimension space (the top three PCs) allows for better visualization of the entire data set. Figure 1.7 shows the arrangement of the samples by plotting them on the principal components derived from: 1) PCA analysis of all the genes in the dataset, and 2) PCA analysis of only those genes which were found to have normal variance across mice. These plots are shown for all the three tissues under study. Kidney and testis show non-random arrangement of the assay points while liver has less discernible patterns. In the case of the kidney tissue for genes with normal variance, the assays arrange into two clusters. One of the clusters has assays which include the replicates from four mice (M1, M2, M5, M6), while the other cluster has mice M3 and M4. This indicates that there is a high similarity among these mice in kidney tissue. In the testis, the first two mice are systematically different from the last four mice. No pattern was observed in liver. PCA can be additionally used as a platform to compare the performance of different methodologies to determine normal variance. The performance can be judged visually on the basis how well the replicates cluster together or measure the goodness of the clusters. We observe that 1) the experimental replicates belonging to any single mice cluster close to each other, and 2) the mice (biological replicates) are also grouped into visible clusters. Pathologically similar mice are clustered together.

1.6 Summary

The goal of this chapter was to provide a brief introduction to some data mining techniques, and to look at how data mining has been used in some representative applications

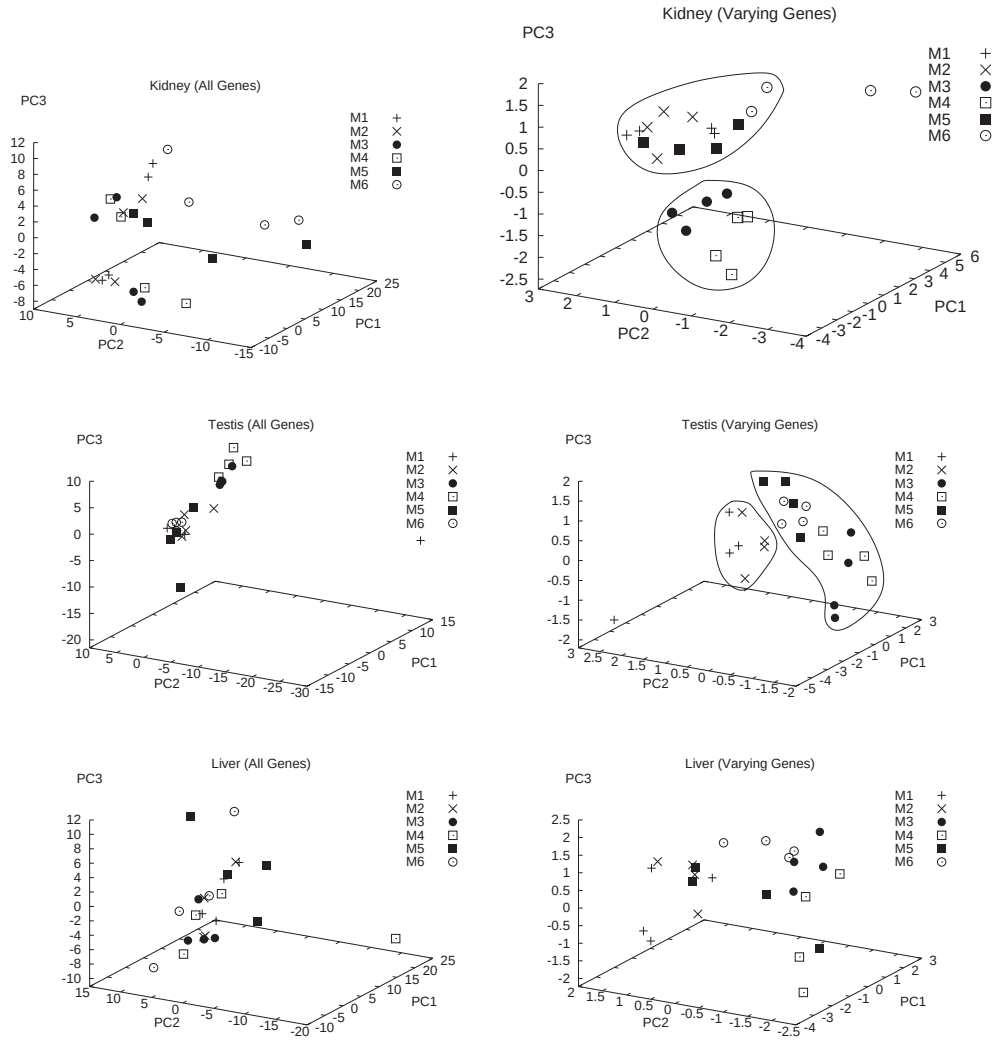


FIGURE 1.7: Principal component analysis of all genes (left column) and genes with normal variance (right column). Three tissues were studied: Kidney (top row), Testis (middle row), and Liver (bottom row). Results for all 6 mice and 4 replicates are shown

in bioinformatics, namely three-dimensional (3D) or structural motif mining in proteins and the analysis of microarray gene expression data. We also looked at some issues in data preparation, namely data cleaning and feature selection via the study of how to find normal variation in gene expression datasets.

It is clear that data mining is playing a fundamental role in understanding the rapidly expanding sources of biological data. It is equally clear that new data mining techniques are needed to analyze, manage and discover sequence, structure and functional patterns/models from large sequence and structural databases, as well as for structure prediction, gene finding, gene expression analysis, biochemical pathway mining, biomedical literature mining, drug design and other emerging problems in genomics and proteomics.

Acknowledgment

This work was supported in part by NSF CAREER Award IIS-0092978, DOE Career Award DE-FG02-02ER25538, and NSF grant EIA-0103708. We thank Jingjing Hu and Vinay Nadimpally for work on contact map mining and determining normal variations in gene expression data, respectively.

References

[1] C. Aggarwal. Towards systematic design of distance functions for data mining applications. In 9th ACM SIGKDD Conference, 2003.

[2] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In ACM SIGMOD Conference, pages 94–105, 1998.

[3] N. Alexandrov and N. Go. Biological meaning, statistical significance and classification of local spatial similarities in non-homologous proteins. Protein Sci., 3:866–875, 1994.

[4] O. Alter, P. Brown, and D. Botstein. Singular value decomposition for genome-wide expression data processing and modeling. Proceedings of the National Academy of Sciences, 97:10101–10106, 2000.

[5] I. Bahar, A. Atilgan, and B. Erman. Direct evaluation of thermal fluctuations in proteins using a single parameter harmonic potential. Folding and Design, 2:173–181, 1997.

[6] A. Ben-Dor, B. Chor, R. Karp, and Z. Yakhini. Discovering local structure in gene expression data: The order-preserving submatrix problem. In 6th Annual International Conference on RECOMB, pages 49–57, 2002.

[7] A. Berger, D.M. Mutch, J. Bruce, G. Matthew, and A. Roberts. Dietary effects of arachidonate-rich fungal oil and fish oil on murine hepatic and hippocampal gene expression. Lipids in Health and Disease, 1:2–10, 2002.

[8] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When is nearest neighbors meaningful? In ICDT Conference, 1999.

[9] M. Blatt, S. Wiseman, and E. Domany. Data clustering using a model granular magnet. Neural Computation, 9:1805–1842, 1997.

[10] P. Bradley and U. Fayyad. Refining initial points for kmeans clustering. In 15th International Conference on Machine Learning, pages 91–99. Morgan Kaufmann, 1998.

[11] K. Brinda, N. Kannan, and S. Vishveshwara. Analysis of homodimeric protein interfaces by graph-spectral methods. Protein Engineering, 15:265–77, April 2002.

[12] Tang C., Zhang L., Zhang A., and Ramanathan M. Interrelated two-way clustering: An unsupervised approach for gene expression data analysis. In 2nd IEEE International Symposium on Bioinformatics and Bioengineering, pages 41–48, November 2001.

[13] S. Chakraborty and S. Biswas. Approximation algorithms for 3-d common substructure identification in drug and protein molecules. In Workshop on Algorithms and Data Structures, pages 253–264, 1999.

[14] Y. Cheng and G. Church. Biclustering of expression data. In 8th International Conference on Intelligent Systems for Molecular Biology, volume 8, pages 93–103, 2000.

[15] D.J. Cook, L.B. Holder, R. Maglothin S. Su, and I. Jonyer. Structural mining of

molecular biology data. IEEE Engineering in Medicine and Biology, 20(4):67–74, 2001.

[16] C. Creighton and S. Hanash. Mining gene expression databases for association rules. Bioinformatics, 19:79–86, 2003.

[17] F. DeSmet, J. Mathys, K. Marchal, G. Thijs, B. DeMoor, and Y. Moreau. Adaptive quality-based clustering of gene expression profiles. Bioinformatics, 18:735–746, 2002.

[18] P. D'haesleer, X. Wen, S. Fuhrman, and R. Somogyi. Information Processing in Cells and Tissues. Plenum Press, New York, 1998.

[19] M. Eisen, P. Spellman, P. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. Proceedings of the National Academy of Sciences, 95:14863–8, 1998.

[20] D. Fischer, H. Wolfson, S. Lin, and R. Nussinov. Three-dimensional, sequence order-independent structural comparison of a serine protease against the crystallographic database reveals active site similarities: potential implication to evolution and to protein folding. Protein Science, 3:769–778, 1994.

[21] G. Getz, E. Levine, and E. Domany. Coupled two-way clustering analysis of gene microarray data. Proceedings of the National Academy of Sciences, 97:12079–12084, October 2000.

[22] H. Grindley, P. Artymiuk, D. Rice, and P. Willet. Identification of tertiary structure resemblance in proteins using a maximal common subgraph isomorphism algorithm. Journal of Molecular Biology, 229:707–721, 1993.

[23] H.M. Grindley, P.J. Artymiuk, D.W. Rice, and P. Willett. Identification of tertiary resemblence in proteins using a maximal common subgraph isomorphism algorithm. J. of Mol. Biol., 229(3):707–721, 1993.

[24] K. Hall. An r-dimensional quadratic placement algorithm. Management Sciences, 17:219–229, November 1970.

[25] J. Heringa and P. Argos. Side-chain clusters in protein structures and their role in protein folding. Journal of Molecular Biology, 220:151–171, 1991.

[26] J. Herrero, A. Valencia, and J. Dopazo. A hierarchical unsupervised growing neural network for clustering gene expression patterns. Bioinformatics, 17:126–136, 2001.

[27] U. Hobohm and C. Sander. Enlarged representative set of protein structures. Protein Science, 3(3):522–524, 1994.

[28] J. Hu, X. Shen, Y. Shao, C. Bystroff, and M.J. Zaki. Mining protein contact maps. 2nd BIOKDD Workshop on Data Mining in Bioinformatics, July 2002.

[29] J. Huan, W. Wang, D. Bandyopadhyay, J. Snoeyink, J. Prins, and A. Tropsha. Mining spatial motifs from protein structure graphs. In 8th Annual International Conference on RECOMB, 2004.

[30] Y. Huang, M. Prasad, W.J.. Lemon, H. Hampel, F.A. Wright, K. Kornacker, V. LiVolsi, W. Frankel, R.T. Kloos, C. Eng, N.S. Pellegata, and A. de la Chapelle. Gene expression in papillary thyroid carcinoma reveals highly consistent profiles. PNAS, 98:15044–15049, October 2001.

[31] D. Jacobs, A. Rader, L. Kuhn, and M. Thorpe. Graph theory predictions of protein flexibility. Proteins: Struct. Funct. Genet., 44:150–155, 2001.

[32] A. Jain, M. Murty, and P. Flynn. Data clustering: a review. ACM Computing Surveys, 31:254–323, September 1999.

[33] M. Jatin, W. Schmitt, D. Hwang, L.-L. Hsiao, S. Gullans, G. Stephanopoulos, and G. Stephanopoulos. Interactive exploration of microarray gene expression patterns in a reduced dimensional space. Genome Res, 12:1112–1120, 2002.

[34] I.T. Jolliffe. Principal Component Analysis, Springer Series in Statistics. Springer Verlag, New York, 1986.

[35] I. Jonassen, I. Eidhammer, D. Conklin, and W. Taylor. Structure motif discovery and mining the pdb. Bioinformatics, 18:362–367, 2002.

[36] N. Kannan, S. Selvaraj, M. Michael Gromiha, and S. Vishveshwara. Clusters in α/β barrel proteins: Implications for protein structure, function and folding: A graph theoretical approach. Proteins: Struct., Funct., Genet., 43:103–112, May 2001.

[37] N. Kannan and S. Vishveshwara. Identification of side-chain clusters in protein structures by graph spectral method. Journal of Molecular Biology, 292:441–464, September 1999.

[38] N. Kannan and S. Vishveshwara. Aromatic clusters: a determinant of thermal stability of thermophilic proteins. Protein Engineering, 13:753–761, November 2000.

[39] G. J. Kleywegt. Recognition of spatial motifs in protein structures. Journal of Molecular Biology, 285:1887–1897, 1999.

[40] G.J. Kleywegt. Recognition of spatial motifs in protein structures. J. Mol. Biol, 285:1887–1897, 1998.

[41] M.S.H Ko, J.R. Kitchen, X. Wang, T.A. Threat, A. Hasegawa, T. Sun, M.J. Kargul, M.K. Lim, Y. Cui, Y. Sano, T. Tanaka, Y. Liang Y, S. Mason, P.D. Paonessa, A.D. Sauls, G.E. DePalma, R. Sharara, L.B. Rowe, J. Eppig, C. Morrell, and H. Doi. Large-scale cdna analysis reveals phased gene expression patterns during preimplantation mouse development. Development, 127:1737–1749, 2000.

[42] I. Koch, T. Lengauer, , and E. Wanke. An algorithm for finding maximal common subtopologies in a set of protein structures. J. of Comp. Biol., 3(2):289–306, 1996.

[43] T. Kohonen. Self-Organization and Associative Memory. Spring-Verlag, Berlin, 1988.

[44] M. Kuramochi and G. Karypis. Frequent subgraph discovery. 1st IEEE Int'l Conf. on Data Mining, November 2001.

[45] L. Lazzeroni and A. Owen. Plaid models for gene expression data. Statistica Sinica, 12:61–86, 2002.

[46] P.D. Lee, R. Sladek, C. Greenwood, and T. Hudson. Control genes and variability: Absence of ubiquitous reference transcripts in diverse mammalian expression studies. Genome Research, 12(2):292–297, February 2002.

[47] G. Michaels, D. Carr, M. Askenazi, S. Fuhrman, X. Wen, and R. Somogyi. Cluster analysis and data visualization of large-scale gene expression data. In Pacific Symposium on Biocomputing, volume 3, pages 42–53, 1998.

[48] H. Midelfart, J. Komorowski, K. Norsett, F. Yadetie, A.K. Sandvik, and A. Legreid. Learning rough set classifiers from gene expression and clinical data. Fundamenta Informaticae, 53:155–183, November 2002.

[49] E.M. Mitchell, P.J. Artymiuk, D.W. Rice, and P. Willett. Use of techniques derived from graph theory to compare secondary structure motifs in proteins. J. Mol. Biol., 212:151–166, 1990.

[50] A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. Scop: a structural classification of proteins database for the investigation of sequences and structures. J. of Mol. Biol., 247:536–540, 1995.

[51] C.C. Pritchard, L. Hsu, J. Delrow, and P.S. Nelson. Project normal: Defining normal variance in mouse gene expression. PNAS, 98:13266–13271, 2001.

[52] S. Raychaudhuri, J. Stuart, and R. Altman. Principal components analysis to summarize microarray experiments: Application to sporulation time series. In Pacific Symposium on Biocomputiing, pages 455–66, 2000.

[53] T.P. Reilly, M. Bourdi, J.N. Brady, C.A. Pise-Masison, M.F. Radonovich, J.W. George, and L.R. Pohl. Expression profiling of acetaminophen liver toxicity in mice using microarray technology. Biochem. Biophys. Res. Commun, 282:321–328, 2001.

[54] R. Singh, A. Tropsha, and I. Vaisman. Delaunay tessellation of proteins. J. Comput. Biol., 3:213–222, 1996.

[55] M. Sternberg, H. Gabb, and R. Jackson. Predictive docking of protein-protein and protein-dna complexes. Current Opinion in Structural Biology, 8:250–256, 1998.

[56] P. Tamayo, D. Solni, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E. Lander, and T. Golub. Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. Proceedings of the National Academy of Sciences, 96:2907–2912, March 1999.

[57] A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. Bioinformatics, 18(suppl.1):S136–S144, 2002.

[58] S. Tavazoie, D. Hughes, M. Campbell, R. Cho, and G. Church. Systematic determination of genetic network architecture. Nature Genet, pages 281–285, 1999.

[59] A. Tefferi, M. Bolander, S. Ansell, E. Wieben, and T. Spelsberg. Primer on medical genomics. part iii: Microarray experiments and data analysis. Mayo Clin Proc., 77:927–40, September 2002.

[60] H. Wang, W. Wang, J. Yang, and P. S. Yu. Clustering by pattern similarity in large data sets. In ACM SIGMOD Conference, 2002.

[61] X. Wang, J.T.L. Wang, D. Shasha, B.A. Shapiro, I. Rigoutsos, and K. Zhang. Finding patterns in three-dimensional graphs: Algorithms and applications to scientific data mining. IEEE Transactions on Knowledge and Data Engineering, 14(4):731–749, July/August 2002.

[62] L. Wernisch, M. Hunting, and S. Wodak. Identification of structural domains in proteins by a graph heuristic. Proteins, 35:338–352, 1999.

[63] J. Yang, W. Wang, H. Wang, and P. Yu. δ-cluster: Capturing subspace correlation in a large data set. In 18th International Conference on Data Engineering, pages 517–528, 2002.

[64] Yeung and Ruzzo. Principal component analysis for clustering gene expression data. Bioinformatics, 17:763–774, 2002.

[65] M. J. Zaki, S. Jin, and C. Bystroff. Mining residue contacts in proteins using local structure predictions. IEEE Transactions on Systems, Man and Cybernetics – B, 33(5), October 2003.

[66] M.J. Zaki, V. Nadimpally, D. Bardhan, and C. Bystroff. Predicting protein folding pathways. 12th Int'l Conference on Intelligent Systems for Molecular Biology, July 2004.

Index