Functional bioinformatics of microarray data: from expression to regulation

Yves Moreau, Frank De Smet, Gert Thijs, Kathleen Marchal, and Bart De Moor
Department of Electrical Engineering, Katholieke Universiteit Leuven
Kasteelpark Arenberg 10, B-3001 Leuven, Belgium

Abstract— **Microarrays are a powerful technique to monitor the expression of thousands of genes in a single experiment. From series of such experiments, it is possible identify the mechanisms that govern the activation of genes in an organism. Short DNA patterns (called binding sites) in or around the genes serve as switches that control gene expression. As a result similar patterns of expression can correspond to similar binding site patterns. We integrate clustering of coexpressed genes with the discovery of binding motifs. We overview several important clustering techniques and present a clustering algorithm (called adaptive quality-based clustering), which we have developed to address several shortcomings of existing methods. We overview the different techniques for motif finding, in particular the technique of Gibbs sampling, and we present several extension of this technique in our Motif Sampler. Finally, we present an integrated web tool called INCLUSive (`http://www.esat.kuleuven.ac.be/~dna/BioI/Software.html`) that allows the easy analysis of microarray data for motif finding.**

I. Introduction

Unraveling the mechanisms that regulate gene activation in an organism is a major goal of molecular biology. In the past few years, microarray technology has emerged as an effective technique to measure the level of expression of thousands of genes in a single experiment. Because of their capacity to monitor many genes, microarrays are becoming the workhorse of molecular biologists studying gene regulation. However, these experiments generate data in such amount and of such a complexity that their analysis requires powerful computational and statistical techniques. As a result, unraveling gene regulation from microarray experiments is currently one of the major challenges of bioinformatics.

Starting from microarray data, a first major computational task is to cluster genes into biologically meaningful groups according to their pattern of expression [25]. Such groups of related genes are much more tractable for study by biologists than the full data itself. Classical clustering techniques such as hierarchical clustering [14] or K-means [55] have been applied to microarray data. Yet the specificity of microarray data (such as the high level of noise or the link to extensive biological information) have created the need for clustering methods specifically tailored to this type of data [19]. We overview both the first generation of clustering methods applied to microarray data as well as second generation algorithms, which are more specific to microarray data. In particular, we address a number of shortcomings of classical clustering algorithms with a new method called adaptive quality-based clustering [10] in which we look for tight reliable clusters.

In a second step, we ask what makes genes belong to the same cluster. A main cause of coexpression of genes is that these genes share the same regulation mechanism at the sequence level. Specifically, some control regions (called the *promoter* region) in or around the genes will contain specific short sequence patterns, called *binding sites*, which are recognized by activating or repressing proteins, called *transcription factors*. In such a situation, we say that the genes are transcriptionally regulated. Switching our attention from expression data to sequence data, we consider algorithms that discover such binding sites in sets of DNA sequences from coexpressed genes. We analyze the upstream region of those genes to detect patterns, also called *motifs*, that are statistically overrepresented when compared to some random model of the sequence. The detection of overrepresented patterns in DNA or amino-acid sequences is called *motif finding*; here we address only the problem of motif finding in DNA sequences (as opposed to motif finding in protein sequences). Two classes of methods are available for motif finding: word-counting methods and probabilistic sequence models. Word-counting methods are string-matching methods based on counting the number of occurrences of each DNA word (called *oligonucleotide*) and comparing this number with the expected number of occurrences based on some statistical model. Probabilistic sequence models build a likelihood function for the sequences based on the motif occurrences and a model of the background sequence. Probabilistic optimization methods, such as Expectation Maximization and Gibbs sampling, are then used to search for the best configurations (motif model and positions). After briefly presenting the word-counting methods and the method based on Expectation Maximization, we discuss the basic principles of Gibbs sampling for motif finding more thoroughly. We also present our Gibbs sampling method, called the Motif Sampler, where we have introduced a number of extensions to improve Gibbs sampling for motif finding, such as the use of a more precise model of the sequence background based on higher-order Markov chains. This improved model increases the robustness of the method significantly.

These two steps, clustering and motif finding, are interlocked and specifically dedicated to the discovery of regulatory motifs from microarray experiments. In particular, clustering needs to take into account that motif finding is sensitive to noise. Therefore, we need clustering methods that build conservative clusters for which coexpression can be guaranteed in an attempt to increase the proportion of coregulated genes in a cluster. This is one of the requirements that warranted the development of our adaptive quality-based clustering algorithm. Also the motif finding algorithms are specifically tailored to the discovery of transcription factor binding motifs (while related algorithms can be developed for slightly different problems in protein sequence analysis). These tight links mandate our integrated presentation of these two topics in this paper. Furthermore, the same links call for integrated software tools to handle this task in an efficient manner. Our INCLUSive web tool (`http://www.esat.kuleuven.ac.be/~dna/BioI/Software.html`) supports motif finding from microarray data. Starting with the clustering of microarray data by adaptive quality-based clustering, it then retrieves the DNA sequences relating to the genes in a cluster in a semi-automated fashion, and finally performs motif finding using our Motif Sampler (see Figure 1). Integration is paramount in bioinformatics as, by optimally matching the different steps of the data analysis to each other, the total analysis becomes more effective than the sum of its parts.

This paper is organized as follows. In Section II, we briefly described microarray technology while in Section III we summarize the basic concepts of molecular biology relevant to motif finding. In Section IV, we overview current clustering algorithms for microarray data, in particular our adaptive quality-based clustering algorithm. We also discuss methods to preprocess the microarray data to make it suitable for clustering and methods to assess the quality of clustering results from statistical and biological standpoints. Next, we describe in Section V the problem of motif finding and overview several of the methods available for this problem. We then explore, in Section VI, the basic principles of Gibbs sampling for motif finding and describe the extensions necessary for its efficient practical application. In Section VII, we describe our INCLUSive web tool for the integration of adaptive quality-based clustering and Gibbs sampling for motif finding. Finally, we summarize our presentation and briefly conclude. Our presentation does not aim at being exhaustive or at presenting all methods in full technical details, but rather at giving a comprehensive and coherent view of the complex problem of motif finding from microarray data.

II. Measuring gene expression profiles

Cells produce the proteins they need to function properly by (1) *transcribing* the corresponding genes from DNA into messenger RNA (mRNA) transcripts and (2) *translating* the mRNA molecules into proteins. Microarrays obtain a snapshot of the activity of a cell by deriving a measurement from the number of copies of each type of mRNA molecule (which also gives an indirect and imperfect picture of the protein activity). The key to this measurement is the double-helix *hybridization* properties of DNA (and RNA). When a single strand of DNA is brought in contact with a complementary DNA sequence, it will anneal to this complementary sequence to form double-stranded DNA. For the four DNA bases, Adenine is complementary to Cytosine and Guanine is complementary to Thymine and the complementary sequence is produced by complementing the bases of the reference sequence starting from the end of this sequence and proceeding further upstream. Hybridization will therefore allow a DNA probe to recognize a copy of its complementary sequence obtained from a biological sample.

An array consists of a reproducible pattern of different DNA probes attached to a solid support. After RNA extraction from a biological sample, fluorescently labeled complementary DNA (cDNA) or cRNA is prepared. This fluorescent sample is then hybridized to the DNA present on the array. Thanks to the fluorescence, hybridization intensities (which are related to the number of copies of each RNA species present in the sample) can be measured by a laser scanner and converted to a quantitative readout. In this way, microarrays allow simultaneous measurement of expression levels of thousands of genes in a single hybridization assay. This data can be further analyzed by data-mining techniques as explained in the next sections.

Two basic array technologies are currently available: cDNA microarrays and gene chips. cDNA microarrays [12] are small glass slides on which double-stranded DNA is spotted. These DNA fragments are normally several hundred base pairs in length and are often derived from reference collections of expressed sequence tags (which are subsequences from an mRNA transcript that uniquely identify this transcript) extracted from many sources of biological materials so as to represent the largest possible number of genes. Usually each spot represents a single gene. cDNA microarrays use *two* samples: a reference and a test sample (e.g., normal versus malignant tissue). A pair of cDNA samples is independently copied from the corresponding mRNA populations with the reverse transcriptase enzyme and labeled using distinct fluorescent molecules (green and red). These labeled cDNA samples are then pooled and hybridized to the array. Relative amounts of a particular gene transcript in the two samples are determined by measuring the signal intensities detected at both fluorescence wavelengths and calculating the ratios (here, only relative expression levels are obtained). A cDNA microarray is therefore a differential technique, which intrinsically normalizes for part of the experimental noise. An overview of the procedure that can be followed with cDNA microarrays is given in Figure 2.

GeneChip oligonucleotide arrays (Affymetrix, Inc., Santa Clara, CA) [32] are high-density arrays of oligonucleotides synthesized using a photolithograpic technology similar to microchip technology. The synthesis uses in situ light-directed chemistry to build up hundreds of thousands of different oligonucleotide probes (25-mers). Each gene is represented by 15-20 different oligonucleotides, serving as unique sequence-specific detectors. In addition mismatch control oligonucleotides (identical to the perfect match probes except for a single base-pair mismatch) are added. These control probes allow estimation of cross-hybridization and significantly decrease the number of false positives. With this technology, absolute expression levels are obtained (no

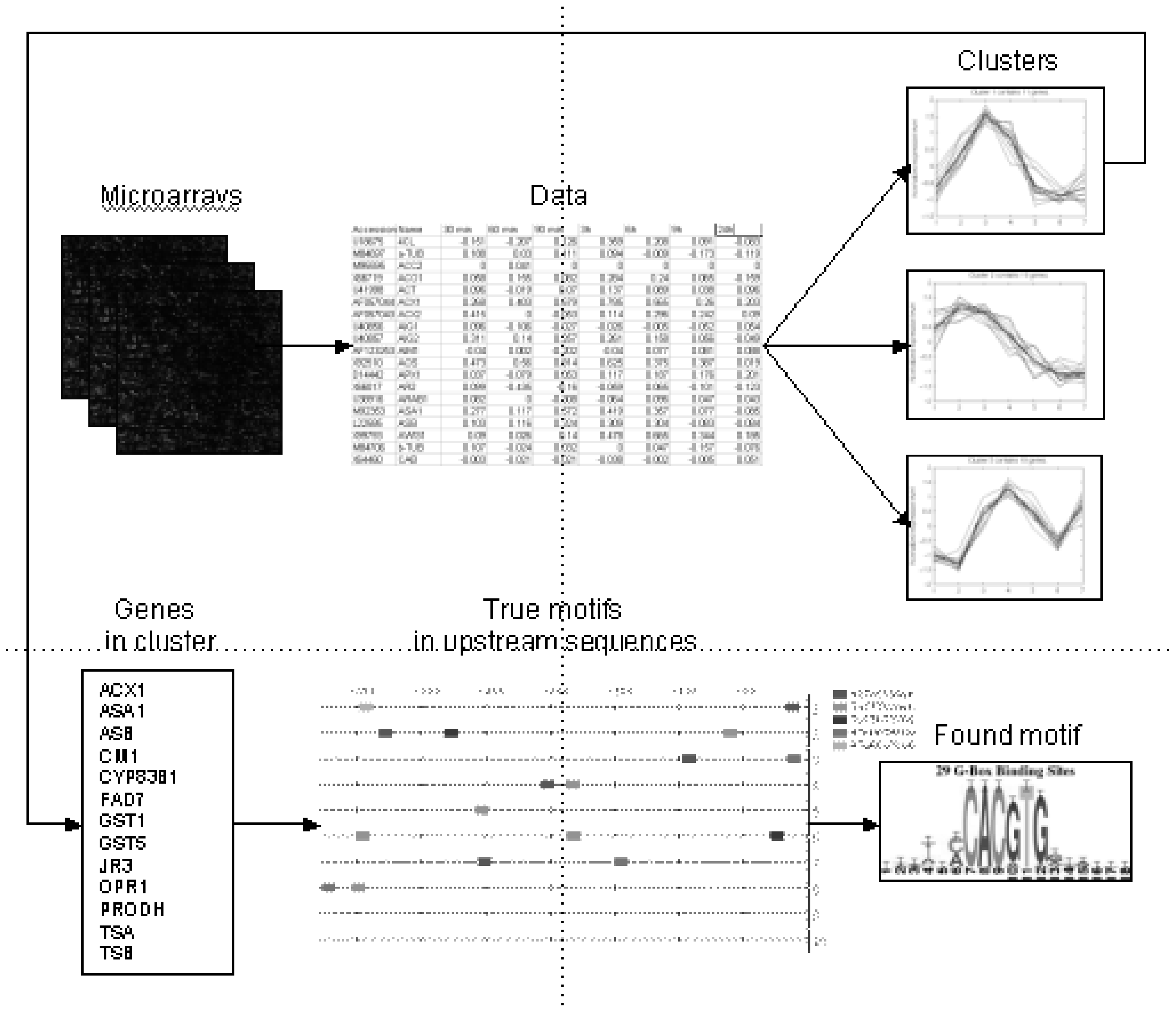


Fig. 1. A high-level description of data analysis for motif finding from microarray data. The analysis starts from scanned microarray images. After proper quantification and preprocessing, the data is available for clustering in the form of a data matrix. Clustering then determines clusters of potentially coregulated genes. Focusing on a cluster of genes of interest, motif finding analyzes the sequences of the control regions of the genes in the cluster. A number of true motifs are present in those sequences but they are unknown. Motif finding analyzes those sequences for statistically overrepresented DNA patterns. Finally, candidate motifs are returned by the motif finding algorithm and are available for further biological evaluation.

ratios).

Both techniques have their own advantages. GeneChip technology claims to be more sensitive and allows absolute measurements. This is mainly because, for each gene, more probes are present on the array (including the mismatch probes). cDNA microarray technology is more flexible as users can produce customized array designs without depending on an external vendor. Further, new variants of these technologies are appearing at a regular pace and the technology can be expected to evolve significantly in the coming years.

III. Introduction to transcriptional regulation

In this section, we present concisely the main concepts from biology relevant to our discussion of motif finding in DNA sequences.

A. *Structure of genes*

Genes are segments of DNA that encode for proteins through the intermediate action of messenger RNA (mRNA). A gene and the genomic region surrounding it consists of a transcribed sequence, which is converted into an mRNA transcript, and of various *untranscribed* sequences. The mRNA consists of a coding sequence that is translated into a protein and of several *untranslated* regions (UTRs). The untranscribed sequences and the UTRs play a major role in the regulation of expression. Notably, the *promoter region* in front of the transcribed sequence contains the *binding sites* for the *transcription factor* proteins that start up transcription. Moreover, the region upstream of the transcription start contains many binding sites for transcription factors that act as *enhancers* and *repressors* of gene expression (although some transcription factors can

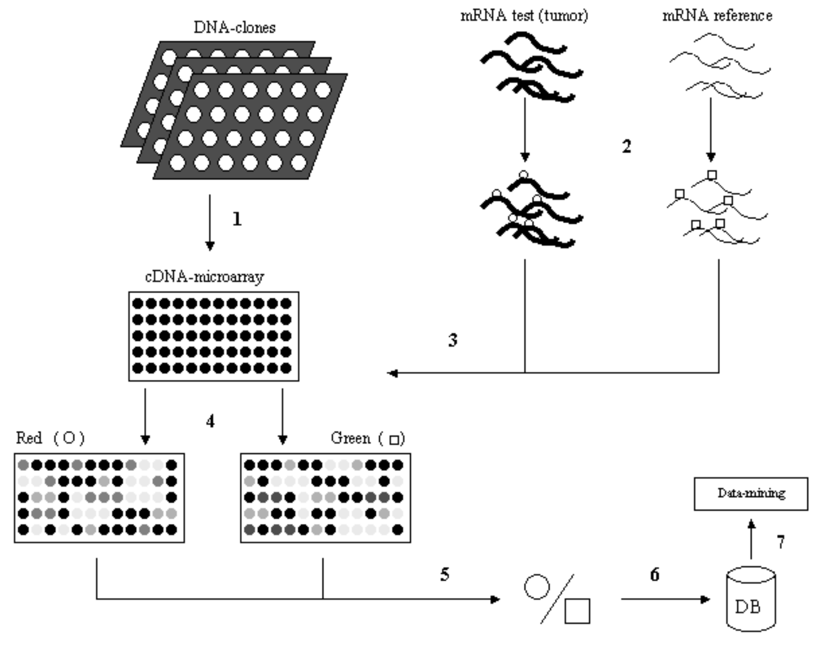


Fig. 2. Schematic overview of an experiment with a cDNA microarray. (1) Spotting of the presynthesized DNA-probes (derived from the genes to be studied) on the glass slide. These probes are the purified products from PCR-amplification of the associated DNA-clones. (2) Labeling (via reverse transcriptase) of the total mRNA of the test sample (red channel ◯) and reference sample (green channel □). (3) Pooling of the two samples and hybridization (4) Readout of the red and green intensities separately (measure for the hybridization by the test and reference sample) in each probe. (5) Calculation of the relative expression levels (intensity in the red channel / intensity in the green channel). (6) Storage of results in a database. (7) Data mining.

bind outside this region).

B. Transcription

Transcription means the assembly of ribonucleotides into a single strand of mRNA (messenger RNA). The sequence of this strand of mRNA is dictated by the order of the nucleotides in the transcribed part of the gene. The transcription process is initiated by the binding of several transcription factors to regulatory sites in the DNA, usually located in the promoter region of the gene. The transcription factor proteins bind each other to form a complex that associates with an enzyme called RNA polymerase. This association enables the binding of RNA polymerase to a specific site in the promoter. In Figure 3, the initiation of the transcription process is shown.

Together, the complex of transcription factors and the RNA polymerase unravel the DNA and separate both strands. Subsequently, the polymerase proceeds down on one strand while it builds up a strand of mRNA complementary to the DNA, until it reaches the terminator sequence. In this way, an mRNA is produced that is complementary to the transcribed part of the gene. Then, the mRNA transcript detaches from the RNA polymerase and the polymerase breaks its contact with the DNA. In a later stage, the mRNA is processed, transported out of the nucleus and translated into a protein.

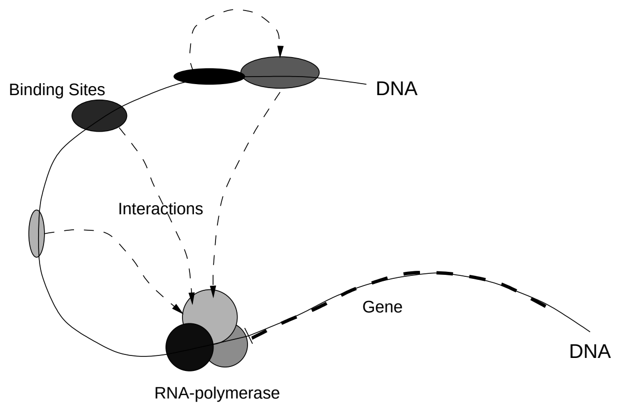


Fig. 3. Initiation of the transcription process by the association of the complex of transcription factors (gene regulatory proteins), the RNA-polymerase, and the promoter region of a gene

C. Transcription factors

Transcription factors are proteins that bind to regulatory sequences on eukaryotic chromosomes thereby modifying the rate of transcription of a gene. Some transcription factors bind directly to specific sequences in the DNA (promoters, enhancers, and repressors), others bind to each other. Most of them bind both to the DNA as well as to other transcription factors. It should be noted that the transcription rate can be positively or negatively affected by the actions of transcription factors. When the transcription factor significantly decreases the transcription of a gene, it is called a repressor. If, on the other hand, the expression of a gene is upregulated, biologists speak of an enhancer.

The protein structure of a transcription factor can be divided into several domains, such as a DNA-binding domain, a transcription activation domain. The DNA-binding domain determines the target genes of the transcription factor. This domain determines the binding specificity to a certain DNA motif. The transcription activation domain is required for binding to the polymerase and actually starting the transcription process. Transcription activation domains presumably function by direct interaction with elements of the transcription initiation complex (e.g., TFIID) or by interaction with intermediary proteins that in turn recognize the initiation complex [56].

D. Regulatory elements on the web

Regulatory elements play a central role in the study of biological sequences and many databases are available to explore known regulatory elements. Table I give a list of database of promoters and gene regulation that are accessible online. Most of these sites are also portals to specific tools for the analysis of regulatory mechanisms.

TABLE I
Databases on transcriptional regulation

Database	URL
EPD	`www.epd.isb-sib.ch/`
TRANSFAC	`www.gene-regulation.de/`
PlantCARE	`sphinx.rug.ac.be:8080/PlantCARE`
PLACE	`www.dna.affrc.go.jp/htdocs/PLACE`
TRRD	`www.bionet.nsc.ru/`
SCPD	`cgsigma.cshl.org/jian/`
HPD	`zlab.bu.edu/~mfrith/HPD.html`
COMPEL	`compel.bionet.nsc.ru/compel/`

IV. Clustering of gene expression profiles

Using microarrays, we can measure the expression levels of thousands of genes simultaneously. These expression levels can be determined for samples taken at different time points during a certain biological process (e.g., different phases of the cycle of cell division) or for samples taken under different conditions (e.g., cells originating from tumor samples with a different histopathological diagnosis). For each gene, the arrangement of these measurements into a (row) vector leads to what is generally called an expression profile. These expression profiles or vectors can be regarded as data points in a high-dimensional space.

Because relatedness in biological function often implies similarity in expression behavior (and vice versa) and because several genes might be involved in the process under study, it will be possible to identify subgroups or clusters of genes that will have similar expression profiles (i.e., according to a certain distance function, the associated expression vectors are sufficiently 'close' to one another). Genes with similar expression profiles are said to be coexpressed. Conversely, coexpression of genes can thus be an important observation to infer the biological role of these genes. For example, coexpression of a gene of unknown biological function with a cluster containing genes with known (or partially known) function can give an indication of the role of the unknown gene. Also, coexpressed genes are more likely to be coregulated, see Section V.

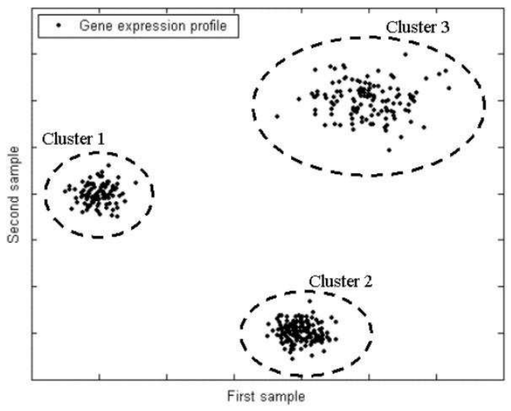


Fig. 4. Visualization of 375 (simulated) gene expression profiles (each expression profile contains two expression levels measured in two different samples—data not standardized). It is clear that, in this case, cluster analysis will result in the identification of three well-separated clusters (representing three classes of genes, possibly associated with specific biological pathways).

Cluster analysis in a collection of gene expression profiles aims at identifying subgroups (= clusters) of such coexpressed genes which thus have a higher probability of participating in the same pathway. In pattern recognition clustering is also called unsupervised learning since these methods are designed to detect unknown classes in the data (this is in contrast with supervised learning where the classes are known in advance). An idealized example in two dimensions is shown in Figure 4.

Note that cluster analysis of expression data is only a first rudimentary step preceding further analysis which includes motif finding [53], [44], [59], functional annotation, genetic network inference, and class discovery in the microarray experiments or samples themselves [5], [19]. Moreover, clustering often is an interactive process where the biologist or medical doctor has to validate or further refine the results and combine the clusters with prior biological or medical knowledge. Full automation of the clustering process is here still far away.

The first generation of cluster algorithms (e.g., direct visual inspection [9], K-means [55], Self-Organizing Maps (SOMs) [48], hierarchical clustering [14]) used to cluster gene expression profiles were developed for purposes other than biologically related research. Although it is possible to obtain biologically meaningful results with these algorithms, some of their characteristics often complicate their use for clustering expression data (these methods lack fine-tuning for biological problems) [46]. They require, for example, the predefinition of one or more user-defined parameters that are hard to estimate by a biologist (e.g., the predefinition of the number of clusters in K-means and SOM—this number is almost impossible to predict in advance). Moreover, changing these parameter settings will often have a profound impact on the final result. These methods therefore need extensive parameter fine-tuning, which means that a comparison of the results with different parameter settings is almost always necessary—with the additional difficulty that comparing the quality of the different clustering results is hard. Another problem is that forcing every data point into a cluster is another problem often occurring with the first generation of clustering algorithms. In general, a considerable number of genes included in the microarray experiment do not contribute to the biological process studied and these genes will therefore lack coexpression with other genes (they will have seemingly constant or even random expression profiles). Including these genes into one of the clusters will contaminate their content (these genes represent noise) and make these clusters less suitable for further analysis. Finally, the computational and memory complexity of some of these algorithms often limit the number of expression profiles that can be analyzed at once. Considering the nature of our data sets (number of expression profiles often running up into thousands), this constraint is often unacceptable.

Recently, many new clustering algorithms have started to tackle some of the limitations of earlier methods (e.g., Self-Organizing Tree Algorithm or SOTA [20], quality-based clustering [23], adaptive quality-based clustering [10], model-based clustering [17], [63], simulated annealing [37], gene shaving [19], CAST [4]). Also, some procedures were developed that could help the biologist to estimate some of the arbitrary parameters needed for the first generation of algorithms (such as the number of clusters present in the data [17], [37], [63]). We will discuss a selection of these clustering algorithms in the following sections. Note that many of these methods can be used with different distance measures, which can also have serious implications for the final result.

An important problem that arises when performing cluster analysis of gene expression profiles is the preprocessing of the data. Clustering implies more than just submitting the raw microarray data to the cluster algorithm of choice. A correct preprocessing strategy is almost as important as the cluster analysis itself. Firstly, it is necessary to normalize the hybridization intensities within a single array experiment. In a two-channel cDNA microarray experiment for example, normalization adjusts for differences in labeling, detection efficiency, and in the quantity of initial RNA within the two channels [25]. Normalization is necessary before one can compare the results from different microarray experiments. Secondly, transformation of the data using a nonlinear function (often the logarithm is used, especially for two-channel cDNA microarray experiments where the values are expression ratios) can be useful [25]. Thirdly, expression data often contains numerous missing values and many clustering algorithms are unable to deal with them [57]. It is therefore imperative either to use appropriate procedures that can estimate and replace these missing values or to adapt existing clustering algorithms, enabling them to handle missing values directly (without actually replacing them [10], [26]). Fourthly, it is common to (crudely) filter the gene expression profiles (removing the profiles that do not satisfy a simple criterion) before proceeding with the actual clustering [14]. A fifth and final customarily used preprocessing step is standardization or rescaling of the gene expression profiles (e.g., multiplying every expression vector with a scale factor so that their lengths are one [25]). This makes sense because the aim is to cluster gene expression profiles with the same relative behavior (expression levels go up and down at the same time) and not only the ones with the same absolute behavior. Some of these preprocessing steps will be discussed in more detail in the following sections.

Validation is another key issue when clustering gene expression profiles. When using existing algorithms or developing new ones it is not merely enough to submit the data to the algorithm and wait for the results. Cluster analysis is more than just producing clusters. The biologist using the algorithm is of course mainly interested in the biological relevance of these clusters and wants to use the results to discover new biological phenomena. This means that we need methods to (biologically and statistically) validate and objectively compare the results produced by new and existing clustering algorithms. Some standard methods for doing cluster validation have recently emerged (Figure of merit [63], (adjusted) Rand index [65], and looking for enrichment of functional categories [50]) and will be discussed below. Note that one of the reasons that there are so many different clustering methods (sometimes giving very different results) is that, from a biological point of view, multiple and equally valid solutions are possible and that these different algorithms often expose different aspects present within the data. Note also that no real benchmark data set exists to unambiguously validate novel algorithms (however the results produced by Cho *et al.* [9] on the cell cycle of yeast are often used for this purpose).

A. Clustering algorithms

As stated in the introduction, many clustering methods (first and second generation algorithms) are available and we will discuss some of the most important of them in more detail below.

A.1 First generation algorithms

Not withstanding some of the disadvantages of these early methods, it must be noted that many good implementations of these algorithms exist ready for use by biologists (which is not always the case with the newer methods).

A.1.a *Direct visual inspection.* This is of course the most simple and direct approach, which was used by many biologists in early work on gene expression analysis [9]. This method is best suited where the patterns of interest are known in advance, but does not work for larger data sets (high number of dimensions or data points) or when one tries to discover unexpected patterns.

A.1.b *Hierarchical clustering.* Hierarchical clustering [14], [25], [46] is the most widely used method for clustering gene expression data and can be seen as the *de facto* standard. Hierarchical clustering has the advantage that the results can be nicely visualized (see Figure 5). Two approaches are possible: a top-down approach (divisive clustering, see [1] for an example) and a bottom-up approach (agglomerative clustering, see [14]). The latter is the most commonly used and will be discussed here. In the agglomerative approach each gene expression profile is initially assigned to a single cluster. The distance between every couple of clusters is calculated according to a certain distance measure (this results in a pairwise distance matrix). Iteratively (and starting from all singletons as clusters), the two closest clusters are merged giving rise to a tree structure where the height of the branches is proportional to the pairwise distance between the clusters. Merging stops if only one cluster is left. Finally, clusters are formed by cutting the tree at a certain level or height. Note that this level corresponds to a certain pairwise distance which in its turn is rather arbitrary (it is difficult to predict which level will give the best biological results). Finally, note that the memory complexity of hierarchical clustering is quadratic in the number of gene expression profiles which can be a problem when considering the current size of the data sets.

Several implementations of hierarchical clustering are available (often together with K-means and self-organizing maps, see next sections) (see Table II).

A.1.c *K-means clustering.* K-means clustering [50], [55] results in a partitioning of the data (every gene expression profile belongs to exactly one cluster) using a predefined number K of partitions or clusters (see Figure 6). K-means starts by assigning at random all the gene expression profiles to one of the N clusters. Iteratively, the center (which is nothing more than the average expression vector) of each cluster is calculated, followed by a re-assignment of the gene expression vectors to the cluster with the closest cluster center. Convergence is reached when the cluster centers remain stationary.

Note that the predefinition of the number of clusters by the user also is rather arbitrary (it is difficult to predict the number of clusters in advance). In practice, this makes it necessary to use a trial-and-error approach where a comparison and biological validation of several runs of the algorithm with different parameter settings are necessary.

A.1.d *Self-organizing maps.* In Self-Organizing Maps (SOMs) [28], [48], the user has to predefine a topology or geometry of nodes (e.g., a two-dimensional grid—one node for each cluster), which again is not straightforward. These nodes are then mapped into the gene expression space, initially at random and iteratively adjusted. In each iteration, a gene expression profile is randomly picked and the node that maps closest to it is selected. This selected node (in gene expression space) is then moved into the direction of the selected expression profile. The other nodes are also moved into the direction of the selected expression profile but to an extent proportional to the distance from the selected node in the initial two-dimensional node topology.

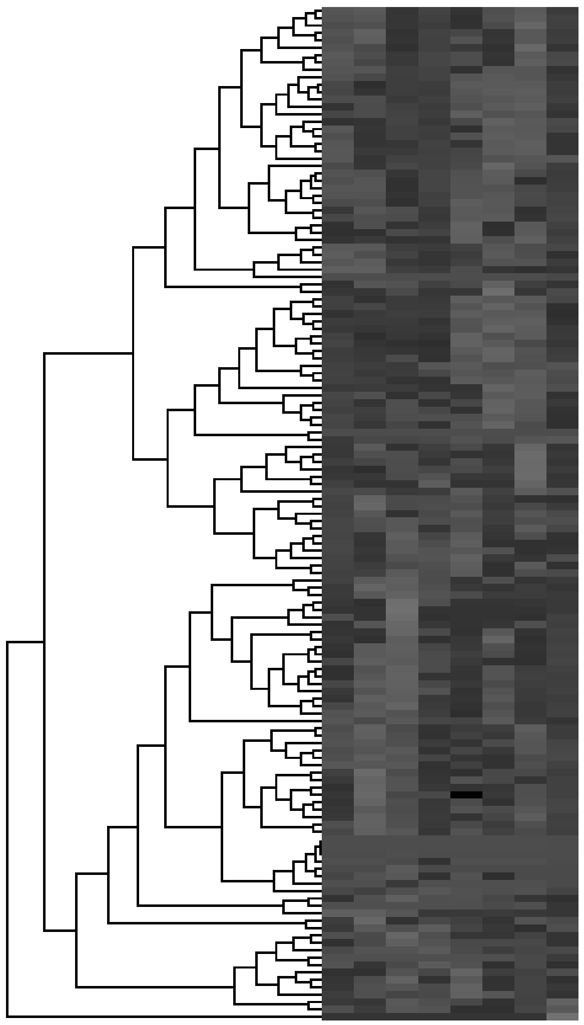


Fig. 5. Typical result from an analysis using hierarchical clustering using 137 expression profiles of dimension 8. The left side of the figure represents the tree structure. The terminal branches of this tree are linked with the individual genes and the height of all the branches is proportional to the pairwise distance between the clusters. The right side of the figure (also called a heat map) corresponds with the expression matrix where each row represents an expression profile, each column a microarray experiment and the individual values are represented on a color (green to red) or grey scale.

A.2 Second generation algorithms

In this section we describe several of the newer clustering methods that have specifically been designed to cluster gene expression profiles.

A.2.a *Self-organizing tree algorithm.* The Self-Organizing Tree Algorithm (SOTA) [20] combines both self-organizing maps and divisive hierarchical clustering. The topology or node geometry here takes the form of a dynamic binary tree. Similar to self-organizing maps, the gene expression profiles are sequentially and iteratively presented to the terminal nodes (located at the base of the tree—these nodes are also called cells). Subsequently, the gene expression

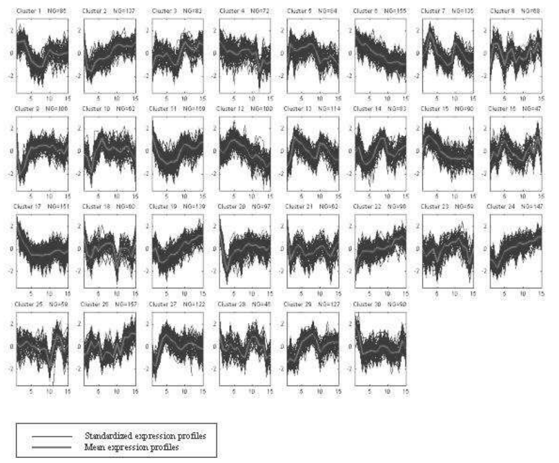


Fig. 6. Typical result from an analysis using K-means clustering with 30 clusters using 3000 standardized expression profiles of dimension 15 (yeast cell cycle, see Cho *et al.* (1998)). The plots represent the 30 different clusters. For each cluster, the normalized expression measurement (y-axis) of each gene across is plotted for the 15 experiments (x-axis). The bold line in each cluster is the average expression profile for this cluster. Note that the sum of the number of genes in each cluster equals the total number of genes submitted to the algorithm ($= 3000$). NG = number of genes.

profiles are associated with the cell that maps closest to it and the mapping of this cell plus its neighboring nodes are updated (moved into the direction of the expression profile). The presentation of the gene expression profiles to the cells continues until convergence. After convergence the cell containing the most variable population of expression profiles (variation is defined here by the maximal distance between two profiles that are associated with the same cell) is split in two sister cells (causing the binary tree to grow) where after the entire process is restarted. The algorithm stops (the tree stops growing) when a threshold of variability is reached for each cell. To obtain a statistical definition for this threshold a randomized version of the entire data set is used (for each expression profile all its expression values are randomly and independently shuffled—this operation destroys the actual correlation between expression profiles) and the distances between all possible pairs of gene expression profiles in this version of the data are calculated. This results in the probability distribution of the distances that could occur by chance (i.e., the distribution that describes the probability that two unrelated expression profiles have a certain distance). The threshold of variability can now be defined by choosing a confidence level α (e.g., $\alpha = 5\%$), so that only a fraction α of the randomized gene expression profiles have a distance smaller than this threshold. Using this threshold ensures that the fraction of wrong assignments (unrelated profiles assigned to the same cluster) in the actual cluster result is limited by the α-value.

The approach described in [20] has some properties that make it potentially useful for clustering gene expression profiles:

- The clustering procedure itself is linear in the number of gene expression profiles (compare this with the quadratic complexity of standard hierarchical clustering).
- The number of clusters does not have to be known in advance. Moreover, the procedure provides for a statistical procedure to stop growing the tree. Therefore, the user is freed from choosing a (arbitrary) level where the tree has to be cut (like in standard hierarchical clustering).

Also, a server running the program is available (see Table II.

To our opinion, this method, however, also has some disadvantages:

- The procedure for finding the threshold of variability is time-consuming since it involves the actual construction of a randomized data set and the calculation of the distances between all possible pairs of randomized expression profiles. The entire process described in [20] is thus in fact quadratic in the number of gene expression profiles.
- No biological validation was provided showing that this algorithm indeed produces biologically relevant results.

A.2.b *Model-based clustering.* Model-based clustering [15], [17], [63] is an approach that is not really new and has already been used in the past for other applications outside bioinformatics. In this sense it is not really a true second generation algorithm. However its potential use for cluster analysis of gene expression profiles has only been proposed recently and we thus we treat it in this text as a second generation method.

Model-based clustering assumes that the data is generated by a finite mixture of underlying probability distributions. Usually, multivariate normal distributions are used for these probability distributions. In this case, each cluster C_i is represented by a multivariate Gaussian model p_i in d dimensions:

$$p_i(y_j|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{\frac{d}{2}}|\Sigma_i|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(y-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)\right)$$

where y_{j} is an expression profile and μ_i and Σ_i the mean and covariance matrix of the multivariate normal distribution respectively.

The covariance matrix Σ_i can be represented by its eigenvalues decomposition, which in general takes the following structure:

$$\Sigma_i = \lambda_i D_i A_i D_i^T,$$

where D_i is the orthogonal matrix of the eigenvectors of Σ_i, A_i is a diagonal matrix whose elements are proportional to the eigenvalues of Σ_i and λ_i is the constant of proportionality. This decomposition implies a nice geometric interpretation of the clusters: D_i controls the orientation, A_i controls the shape, and λ_i controls the volume of the cluster. Note that simpler forms for the covariance structure can be used (e.g., by having some of the parameters take the same values across clusters), thereby decreasing the number of parameters that have to be estimated but also decreasing the model flexibility (capacity to model more complex data structures).

The mixture model p itself takes then the following form:

$$p(y_j) = \sum_{i=1}^{K} \pi_i . p_i(y_j|\mu_i, \Sigma_i)$$

where K is the number of clusters and π_i is the prior probability that an expression profile belongs to cluster C_i so that

$$\sum_{i=1}^{K} \pi_i = 1.$$

In practice we would like, given a collection of expression profiles $y_j (j = 1, \ldots, n)$ to estimate all the parameters (π_i, μ_i, $\Sigma_i (i = 1, \ldots, K)$, and K itself) of this mixture model. In a first step π_i, μ_i, $\Sigma_i (i = 1, \ldots, K)$ are estimated with an EM algorithm using a fixed value for K and a fixed covariance structure. This parameter estimation is then repeated for different values for K and different covariance structures. The result of the first step is thus a collection of different models fitted to the data and all having a specific value for K and a specific covariance structure. In a second step the best model in this group of models is selected (i.e., the most appropriate number of clusters and a covariance structure is chosen here). This model selection step involves the calculation of the Bayesian Information Criterion (BIC) [45] for each model, which is not further discussed here.

A good implementation for model-based clustering (called MCLUST [15]) is available (see Table II). Yeung *et al.* [63] reported good results using this software on several synthetic data sets and real expression data sets. They claimed that the performance of MCLUST on real expression data was at least as good as could be achieved with a heuristic cluster algorithm (CAST [5], not discussed here).

A.2.c *Quality-based clustering.* In [23], a clustering algorithm (called QT_Clust) is described that produces clusters that have a quality guarantee which ensures that all members of a cluster should be coexpressed with all other members of this cluster (this property is called transitivity). Heyer *et al.* define the quality of a cluster C as a diameter (equal to $1 - \min_i, j \in \{s_{ij}\}$ where s_{ij} is the jackknife correlation between expression profile i and j) but the method can be easily extended to other definitions. The quality guarantee itself is defined as a fixed and user-defined threshold for the quality of each cluster.

Briefly said, the aim of QT_Clust is to find clusters, with a quality guarantee, containing a maximum number of expression profiles. It considers every expression profile in the data set as a cluster seed (one could also call this a cluster center) and iteratively assigns the expression profiles to these clusters that cause a minimal increase in diameter until the diameter threshold (= quality guarantee) is reached. Note that at this stage every expression profile is made available to every candidate cluster and that there are as many candidate clusters as there are expression profiles. At this point, the candidate cluster that contains the most expression profiles is selected as a valid cluster and removed from the data set where after the whole process starts again. The algorithm stops when the number of points in the largest remaining cluster falls below a prespecified threshold. Note that this stop criterion implies that the algorithm will terminate before all expression profiles are assigned to a cluster.

This approach was designed with cluster analysis of expression data in mind and has some properties that could make it useful for this task:

- By using a stringent quality guarantee it is possible to find clusters with tightly related expression profiles (containing highly coexpressed genes). These clusters might therefore be good ‘seeds’ for further analysis.
- Genes not really coexpressed with other members of the data set are not included in any of the clusters.

There are, however, also some disadvantages:

- The quality guarantee of the clusters is a user-defined parameter that is hard to estimate and too arbitrary. This method is therefore, in practice, hard to use by biologists and extensive parameter fine-tuning is necessary.
- This algorithm produces clusters all having the same fixed diameter not optimally adapted to the local data structure.
- The computational complexity is at least quadratic in the number of expression profiles.

Furthermore, no ready to use implementation is available.

A.2.d *Adaptive Quality-Based Clustering.* Adaptive quality-based clustering [10] was developed starting from the principles of quality-based clustering (finding clusters with a quality guarantee containing a maximal number of members) but was designed to circumvent some of its disadvantages.

Adaptive quality-based clustering is a heuristic iterative two-step approach. In the first step a quality-based approach is followed. Using an initial estimate of the quality of the cluster (here, quality is defined as a radius and not as a diameter), a cluster center is located in an area where the density of gene expression profiles is locally maximal. Contrary to the original method [23], the computational complexity of this first step is only linear in the number of expression profiles.

In the second step, called the adaptive step, the quality of the cluster—given the cluster center, found in the first step, that remains fixed—is re-estimated as to assess that the genes belonging to the cluster are, in a statistical sense, significantly coexpressed (higher coexpression that could be expected by chance—according to a significance level S; e.g., $S = 95\%$). To this end, a bimodal and one-dimensional probability distribution (the distribution consists of two terms: one for the cluster and one for the rest of the data) is fitted to the data using an EM algorithm. Note that, the computational complexity of this step is negligible with respect to the computational complexity of the second step.

Finally, step one and two are repeated, using the re-estimation of the quality as the initial estimate needed in the first step, until the relative difference between the initial and re-estimated quality is sufficiently small. The cluster is subsequently removed from the data and the whole procedure is restarted. Note that only clusters whose size exceeds a predefined number are presented to the user.

We feel that adaptive quality-based clustering approach has some additional advantages over standard quality-based clustering that make it suited for the analysis of gene expression profiles:

- In adaptive quality-based clustering the user has to specify a significance level S. This parameter has a strict statistical meaning and is therefore much less arbitrary (contrary to the quality guarantee used in standard quality-based clustering). It can be chosen independently of a specific data set or cluster and it allows for a meaningful default value (95%) that in general gives good results. This makes this approach user-friendly without the need for extensive parameter fine-tuning.
- Adaptive quality-based clustering produces clusters adapted to the local data structure (the clusters do not have the same radius).
- The computational complexity of the algorithm is linear in the number of expression profiles.
- Adaptive quality-based clustering was extensively biologically validated.

Furthermore, a server running the program is available (see Table II). Note also that this implementation has an integrated approach for missing values without the necessity to replace them.

However, the method also has some limitations:

- It is a heuristic approach not proven to converge in every situation.
- Because of the model structure used in the second step, some additional constraints have to be imposed. They include the fact that only standardized expression profiles are allowed and that the method has to be used in combination with the Euclidean distance and cannot directly be extended to other distance measures.

As a conclusion to this overview of clustering algorithms, Table II gives an overview of some clustering methods for which the software is available for download or can be accessed online.

TABLE II
Availability of clustering algorithms.

Package	URL
Cluster	`http://rana.lbl.gov/EisenSoftware.htm`
J-Express	`http://www.molmine.com`
Expr. Profiler	`http://ep.ebi.ac.uk/`
SOTA	`http://bioinfo.cnio.es/sotarray`
MCLUST	`http://www.stat.washington.edu/fraley/mclust`
INCLUSive	`www.esat.kuleuven.ac.be/~dna/BioI/Software.html`

B. Preprocessing of the data

As stated at the start of this section, clustering also implies performing some additional operations on the data, preparing it for the actual cluster analysis. Below, we will discuss some of the most common preprocessing steps. The order of the different preprocessing steps can somewhat vary between specific implementations, we present them in their most frequent order.

B.1 Normalization

The first step is the normalization of the hybridization intensities within a single array experiment. In a two-channel cDNA microarray experiment, several sources of noise (such as differences in labeling, detection efficiency, and in the quantity of initial RNA within the two channels) create systematic sources of biases. The biases can be computed and removed to correct the data. As many sources can be considered and as they can be estimated and corrected in a variety of ways, many different normalization procedures exist. We therefore do not cover this topic further here and refer to [25] for more details.

B.2 Nonlinear transformations

It is common practice to pass expression values through a nonlinear function. Often the logarithm is used for this nonlinear function. This is especially suited when dealing with expression ratios (coming from two-channel cDNA microarray experiments, using a test and reference sample) since expression ratios are not symmetrical [25]. Upregulated genes have expression ratios between 1 and infinity, while downregulated genes have expression ratios squashed between 1 and 0. Taking the logarithms of these expression ratios results in more symmetry between expression values of up- and downregulated genes.

Other, but uncommonly used transformations, include square, square root, and inverse transformations.

B.3 Missing value replacement

Microarray experiments often contain missing values (measurements absent because of technical reasons). The inability of many cluster algorithms to handle such missing values, necessitates the replacement of these values. Simple replacements such as a replacement by 0 or by the average of the expression profile often disrupt these profiles. Indeed replacement by average values relies on the unrealistic assumption that all expression values are similar across different experimental conditions. Because of an erroneous missing value replacement genes containing a high number of missing values can be assigned to the wrong cluster. More advanced techniques of missing value replacement have been described [57] that take advantage of the rich information provided by the expression patterns of other genes in the data set.

The K-nearest neighbor method selects the K genes with an expression profiles most similar (e.g., based on an Euclidean distance) to the gene of which the missing value needs to be replaced. The missing value is then substituted by the similarity-weighted average of the corresponding values in the profiles of the K closest genes. This algorithm seems relatively robust against the setting of the value K (the number of closest genes selected for inference of the missing value) and is independent of the data structure. An alternative method uses an iterative technique based on singular value decomposition (SVD) iteratively to identify the most important eigenvectors in the data set (often called eigengenes and eigenarrays in the case of microarray data). These constitutes a set of mutually orthogonal expression patterns that can, when linearly combined, reconstruct the expression levels of all genes in the data set. In each iteration the profile of the gene containing a missing value is regressed against a selection of the k most important eigengenes. Based on a linear combination of the eigengenes determined by the readily determined regression coefficients, the missing value is reconstructed. Because of the properties of SVD, this technique is optimally adapted to data sets containing an expression profile with a pronounced variation.

Remark that when a cluster algorithm is used based on the same procedure, as is used by the algorithm for missing value replacement, faint expression patterns in the data set can be artificially enhanced. Inferred values should be flagged to avoid drawing unwarranted conclusions.

Finally note that some implementations of algorithms only make use of the measured values to derive the clusters and as such obviate the need for missing value replacement [10].

B.4 Filtering

As stated in the overview section, a set of microarray experiments, generating gene expression profiles, frequently contain a considerable number of genes that do not really contribute to the biological process that is being studied. The expression values of these profiles often show little variation over the different experiments (they are called constitutive with respect to the biological process studied). Moreover, these constitutive genes will have seemingly random and meaningless profiles after standardization (division by a small standard deviation resulting in noise inflation), which is also a common preprocessing step (see further). Another problem with microarray data sets is given by the fact that they regularly contain highly unreliable expression profiles with a considerable number of missing values. Due to their number, replacing these missing values in these expression profiles is not possible within the desired degree of accuracy.

The quality of the clusters would significantly degrade, if these data sets would be passed to the clustering algorithms as such. Most clustering algorithms assign every expression profile in the data to one of the clusters, even the ones of poor quality, corrupting the content and the average profile of these clusters making them less suitable for further analysis. A solution to this problem is to use clustering algorithms that do not assign every profile to a cluster (e.g., quality-based clustering). Another, more simple solution (that can also be used in combination with the previous solution), is to remove at least a fraction of the undesired genes from the data. This procedure is in general called filtering [14]. Filtering involves removing gene expression profiles from the data set that do not satisfy a one or possibly more simple criteria. Commonly used criteria include a minimum threshold for the standard deviation of the expression values in a profile (removal of constitutive genes) and a threshold on the maximum percentage of missing values. Another similar method for filtering takes a fixed number or fraction of genes best satisfying one criterion (like the criteria stated above).

B.5 Standardization or rescaling

Biologists are mainly interested in grouping gene expression profiles that have the same relative behavior; i.e., genes that are up- and downregulated together. Genes showing the same relative behavior but with diverging absolute behavior (e.g., gene expression profiles with a different base line and/or a different amplitude but going up and down at the same time) will have a relatively high Euclidean distance. Cluster algorithms based on this distance measure will therefore wrongfully assign these genes to different clusters.

This result can largely be prevented by applying standardization or rescaling to the gene expression profiles. Gene expression profiles showing the same relative behavior will have a small(er) Euclidean distance after rescaling [25].

Consider a gene expression profile $g(g_1, g_2, \ldots, g_n)$ with average expression level μ and standard deviation σ. Rescaling is commonly done by replacing every expression level g_i in g by

$$\frac{g_i - \mu}{\sigma}$$

and repeating this for every expression vector in the data set. This operation results in a collection of expression profiles all having average zero and standard deviation 1 (i.e., the absolute differences in expression behavior have been largely removed). Note that the division by the standard deviation is sometimes omitted (rescaling is then called mean centering).

C. Cluster validation

As mentioned before, clustering will produce different results. Even random data often produces clusters depending on the specific choice of preprocessing, algorithm, and distance measure. Even random data can be fed into many clustering algorithms and deliver some clusters. Therefore validation of the relevance of the cluster results is of utmost importance. Validation can be either statistical or biological. Statistical cluster validation can be done by assessing cluster coherence, by examining the predictive power of the clusters, or by testing the robustness of a cluster result against the addition of noise.

Alternatively, the relevance of a cluster result can be assessed by a biological validation. Of course it is hard, not to say impossible, to select the best cluster output since 'the biologically best' solution will only be known if the biological system studied is completely characterized. Although some biological systems have been described extensively, none such completely characterized benchmark system is now available. A common method to biologically validate cluster outputs is to search for enrichment of functional categories within a cluster. Detection of regulatory motifs (see [10]) is also an appropriate biological validation of the cluster results. Some of the recent methodologies described in literature to validate cluster results will be highlighted in the following.

C.1 Testing cluster coherence

Based on biological intuition, a cluster result can be considered reliable if the within cluster distance is small (i.e., all genes retained are tightly coexpressed) and the cluster has an average profile well delineated from the remainder of the data set (maximal intercluster distance). Such criteria can be formalized in several ways, such as the sum-of-squares criterion of K-means [55] , silhouette coefficients [26], or Dunn's validity index [2]. These can be used as stand-alone statistics to mutually compare cluster results. They can also be used as an inherent part of cluster algorithms, if their value is optimized during the clustering process.

C.2 Figure of merit

The Figure Of Merit (FOM) [64] is a simple quantitative data-driven methodology that allows comparisons between outputs of different clustering algorithms. The methodology is related to the jackknife and leave-one-out cross-validation. The method goes as follows. The clustering algorithm (for the genes) is applied to all experimental conditions (the data variables) except for one left-out condition. If the algorithm performs well, we expect that if we look at the genes in a given cluster, their values for the left-out condition will be highly coherent. Therefore, we compute the FOM for a clustering result by summing, for the left-out condition, the squares of the deviations of each gene relative to the mean of the genes in its cluster *for this condition*. FOM measures the within-cluster similarity of the expression values of the removed experiment and therefore reflects the predictive power of the clustering. It is expected that removing one experiment from the data should not interfere with the cluster output if the output is robust. For cluster validation, each condition is subsequently used as a validation condition and the aggregate FOM over all conditions is used to compare cluster algorithms. Although simple at first glance, application of the FOM procedure to real biological data is not straightforward. At first the FOM as a measure of the cluster predictive power tends to increase proportionally to the number of clusters. To compensate for this an adjusted FOM was described. Secondly, outputs of cluster algorithms can only be compared if they consist of the same number of clusters and if all genes were retained during clustering.

C.3 Sensitivity analysis

Gene expression levels are the superposition of real biological signals and experimental errors. A way to assign confidence to a cluster membership of a gene consists in creating new *in silico* replicas of the microarray data by adding to the original data a small amount of artificial noise (similar to the experimental noise in the data) and clustering the data of those replicas. If the biological signal is stronger than the experimental noise in the measurements of a particular gene, adding small artificial variations (in the range of the experimental noise) to the expression profile of this gene will not drastically influence its overall profile and therefore will not affect its cluster membership. In this case, the cluster membership of that particular gene is robust with respect to sensitivity analysis and a reliable confidence can be assigned to the clustering result of that gene. However, for genes with low signal to noise ratios, the outcome of the clustering result will be more sensitive to adding artificial noise. Thus, sensitivity analysis will let us detect which clusters are robust within the range of experimental noise and therefore trustworthy for further analysis.

The main issue in this method is to choose the noise level for sensitivity analysis. Bittner *et al.* [6] perturb the data by adding random Gaussian noise with zero mean and a standard deviation that is estimated as the median standard deviation for the log-ratios for all genes across the experiments (as was mentioned earlier in most cDNA data sets, the log-ratio is used as an estimate of the differential expression level). This estimate of the noise level does not only account for the experimental noise but to a certain extent includes variation due to differential expression and thus is an overestimate of the experimental error. Moreover, using Gaussian noise is based on the strong assumption that errors on the log-ratio are normally distributed. This implicitly assumes that ratios are unbiased estimators of relative expression. Reality shows often otherwise.

To compare the clustering results of the different replicas, Bittner *et al.* [6] developed a statistics that counts the frequency with which pairs of genes, clustered together in the original data set are clustered together in distinct replicas (see above). Stable clusters of genes should emerge if each pair of genes clusters together reliably. Their statistics is reminiscent of the RAND index [65].

The bootstrap analysis methods described by Kerr *et al.* [27] to identify statistically significant expressed genes or to assess the reliability of a clustering result, offers a more statistically founded basis for sensitivity analysis and overcomes some of the problems of the method described by Bittner *et al.* [6]. Bootstrap analysis uses the residual values of a linear ANOVA model as an estimate of the measurement error. By using an ANOVA model nonconsistent measurement errors can be separated from variations caused by alterations in relative expression or by consistent variations in the data set. These errors are assumed to be independent with mean 0 and constant variance σ^2 but no explicit assumption on their distribution is made. The residuals are subsequently used to generate new replicates of the data set by bootstrapping (adding residual noise to estimated values). Newly generated data sets are clustered and cluster results are compared.

C.4 Use of different algorithms

Just as clustering results are sensitive to adding noise, they are sensitive to the choice of clustering algorithm used and the specific parameter settings of a particular algorithm. Many clustering algorithms are available, each of them with different underlying statistics and inherent assumptions about the data. The best way to infer biological knowledge from a clustering experiment is to use different algorithms with different parameter settings. Clusters detected by most algorithms will reflect the pronounced signals in the data set. Biologists tend to prefer algorithms with a deterministic output since this gives the illusion that what they find is 'right'. However, nondeterministic algorithms offer an advantage for cluster validation since their use implicitly includes a form of sensitivity analysis.

Performing sensitivity analysis and repeating cluster experiments is of course only useful if statistical measures can be developed to compare the outcome of all these different analyses. Analyzing a microarray experiment does not only involve the development of statistical data-mining algorithms to cluster the information but also to statistically compare the output of these algorithms.

C.5 Enrichment of functional categories

One way to (biologically) validate results from clustering algorithms is to compare the gene clusters with existing functional classification schemes. In such schemes, genes are allocated to one or more functional categories [17], [50] representing their biochemical properties, biological roles, and so on. Finding clusters that have been significantly enriched for genes with similar function is proof that a specific clustering technique produces biologically relevant results.

As stated in the overview section, the results of the expression profiling experiment of Cho *et al.* [9] studying the yeast cell cycle (*Saccharomyces cerevisiae*) in a synchronized culture is often used as a benchmark data set. It contains 6220 expression profiles taken over 17 time points (measurements over 10-min intervals, covering nearly two cell cycles, also see `http://cellcycle-www.stanford.edu`). One of the reasons that this data is so frequently used as benchmark data for the validation of new clustering algorithms is because the majority of the genes included in the data have been functionally classified [40] (MIPS database, see `http://mips.gsf.de/proj/yeast/catalogues/funcat/index.html`) making it possible to biologically validate the results.

Assume that a certain clustering method finds a set of clusters in the Cho *et al.* data. We could objectively look for functionally enriched clusters as follows: suppose that one of the clusters has n genes where k genes belong to a certain functional category in the MIPS database and suppose that this functional category in its turn contains f genes in total. Also suppose that the total data set contains g genes (in the case of Cho *et al.* [9], g would be 6220). Using the cumulative hypergeometric probability distribution, we could calculate the probability or P-value that this degree of enrichment could have occurred by chance; i.e., what is the chance of finding at least k genes in this specific cluster from this specific functional category by chance:

$$P = 1 - \sum_{i=0}^{k-1} \frac{\binom{f}{i}\binom{g-f}{n-i}}{\binom{g}{n}} = \sum_{i=k}^{\min(n,f)} \frac{\binom{f}{i}\binom{g-f}{n-i}}{\binom{g}{n}}.$$

These P-values can be calculated for each functional category in each cluster. Since there are about 200 functional categories in the MIPS database, only clusters where the P-value is smaller than 0.0003 for a certain functional category, are said to be significantly enriched (level of significance 0.05). Note that these P-values can also be used to compare the results from functionally matching clusters identified by two different clustering algorithms on the same data. As an example of cluster validation and as an illustration of our adaptive quality-based clustering, we compare K-means and adaptive quality-based clustering on the Cho *et al.* data. We performed adaptive quality-based clustering [10] using default parameters (in particular, the significance level is set at 95%) and compare these results with those for K-means reported by Tavazoie *et al.* [50]. The genes in each cluster have been mapped to the functional categories in the MIPS database and the negative base-10 logarithm of the hypergeometric P-values (representing the degree of enrichment) have been calculated for each functional category in each cluster. In Table III, we compare enrichment in functional categories for the three most significant clusters. To compare K-means and adaptive quality-based clustering, we identified functionally matching clusters manually. The first column ("Cl. #, AC") gives the index of the cluster identified by adaptive quality-based clustering. The second column ("Cl. #, KM") gives the index of the matching cluster for K-means as described in Tavazoie *et al.* [50]. The third column ("# Gene, AC") gives the number of genes of in the cluster for adaptive quality-based clustering. The fourth column ("# Gene, KM") gives the number of genes of in the cluster for adaptive quality-based clustering. The fifth column ("MIPS functional category")lists the significant functional categories for this cluster. The sixth column ("In cat., AC") gives the number of genes of the corresponding functional category in the cluster for adaptive quality-based clustering. The seventh column ("In cat., KM") gives the number of genes of the corresponding functional category in the cluster for K-means. The eighth column ("P-val.") gives the negative logarithm in base 10 of the hypergeometric P-value for adaptive quality-based clustering. The eighth column ("P-val.") gives the negative logarithm in base 10 of the hypergeometric P-value for K-means (NR = not reported.) Although we do not claim to draw any conclusion from this single table, we observe that the enrichment in functional categories in stronger for adaptive quality-based clustering than for K-means. This result and several others are discussed extensively in [10].

TABLE III

Comparison of functional enrichment on the yeast cell cycle data of Cho *et al.* for adaptive-quality based clustering and K-means.

Cl. # AC	Cl. # KM	# Gene AC	# Gene KM	MIPS functional category	In cat. AC	In cat. KM	P-val. AC	P-val. KM
1	1	302	164	ribosomal proteins	101	64	80	54
				organization of cytoplasm	146	79	77	39
				protein synthesis	119	NR	74	NR
				cellular organization	211	NR	34	NR
				translation	17	NR	9	NR
				organization of chromosome structure	4	7	1	4
2	4	315	170	mitochondrial organization	62	32	18	10
				energy	35	NR	8	NR
				proteolysis	25	NR	7	NR
				respiration	16	10	6	5
				ribosomal proteins	24	NR	4	NR
				protein synthesis	33	NR	4	NR
				protein destination	49	NR	4	NR
5	2	98	186	DNA synthesis and replication	20	23	18	16
				cell growth and division, DNA synthesis	48	NR	17	NR
				recombination and DNA repair	12	11	8	5
				nuclear organization	32	40	8	4
				cell-cycle control and mitosis	20	30	7	8

V. Searching for common binding sites of coregulated genes

In the previous section, we described the basic ideas underlying several clustering techniques together with their advantages and shortcomings. We also discussed the preprocessing steps necessary to make microarray data suitable for clustering. Finally, we described methodologies for validating the result of a clustering algorithm. We can now make the transition towards looking at the groups of genes generated by clustering and study the sequences of these genes to detect motifs that control their expression (and cause them to cluster together in the first place).

Given a cluster of genes with highly similar expression profiles, the next step in the analysis is the search for the mechanism that is responsible for their coordinated behavior. We basically assume that coexpression frequently arises from transcriptional coregulation. As coregulated genes are known to share some similarities in their regulatory mechanism, possibly at transcriptional level, their promoter regions might contain some common motifs that are binding sites for transcription regulators. A sensible approach to detect these regulatory elements is to search for statistically overrepresented motifs in the promoter region of such a set of coexpressed genes [7], [43], [44], [50], [66].

In this section we describe the two major classes of methods to search for overrepresented motifs. The first class of methods are string-based methods that mostly rely on counting and comparing oligonucleotide frequencies. The second class of methods are based on probabilistic sequence models. For these methods, the model parameters are estimated using maximum likelihood or Bayesian inference.

Table IV gives an overview of some of the methods described in the section that can be accessed online or where the software is available for download.

TABLE IV

Availability of motif finding algorithms.

Package	URL
RSA tools	`www.ucmb.ulb.ac.be/bioinformatics/rsa-tools/`
YMF	`abstract.cs.washington.edu/~blanchem/cgi-bin/YMF.pl`
Consensus	`ural.wustl.edu/ softwares.html`
MEME	`meme.sdsc.edu/ meme/website/`
Gibbs Sampler	`bayesweb.wadsworth.org/gibbs/gibbs.html`
AlignACE	`atlas.med.harvard.edu/`
BioProspector	`bioprospector.stanford.edu/`
INCLUSive	`www.esat.kuleuven.ac.be/~dna/BioI/Software.html`

In this section, we first start with a discussion of the important facts that we can learn by looking at a realistic biological example. Prior knowledge about the biology of the problem at hand will facilitate the definition of a good model. Next, we discuss the different string-based methods, starting from a simple statistical model and gradually refining the models and the statistics to handle more complex configurations. Then we switch to the probabilistic methods and we introduce Expectation Maximization for motif finding. In the Section VI, we discuss Gibbs sampling for motif finding. This method is less well known than Expectation Maximization and, yet, it is more effective for motif finding in DNA sequences. We therefore explain the basic ideas underlying this method and overview the extensions, including our own work, that are necessary for the practical use of this method.

A. Realistic sequence models

To search for common motifs in sets of upstream sequences a realistic model should be proposed. Simple motif models are designed to search for conserved motifs of fixed length, while more complex models will incorporate variability like insertions and deletions into the motif model. But not only the model of the binding site itself is important also the model of the background sequence in which the motif is hidden and the number of times a motif occurs in the sequence play important roles. For instance, there are some major differences in transcriptional regulation between prokaryotes (organisms without cell nucleus) and eukaryotes (organisms whose cells contain a nucleus). In higher eukaryotes there are typically multiple copies of a binding site present in the upstream region of a gene to enhance the activity of a binding factor.

To illustrate this complexity, we look at an example in baker's yeast (*S. Cerevisiae*). Figure 7 gives a schematic representation of the upstream sequences from 11 genes in *S. Cerevisiae* which are regulated by the Cbfl-Met4p-Met28p complex and Met31p or Met32p in response to methionine [58]. The consensus (which is the dominant DNA pattern describing the motif) for these binding sites is given by `TCACGTG` for the Cbfl-Met4p-Met28p complex and `AAAACTGTGG` for Met31p or Met32p [58]. A logo representation of the aligned instances of the two binding sites is shown in Figure 8. Such a logo represents the frequency of each nucleotide at each position, the relative size of the symbol represent the relative frequency of each base at this position while the total height of the symbols represent the magnitude of the deviation from a uniform (noninformative) distribution. Figure 7 shows the locations of the two binding sites in the region 800 base pairs upstream of translation start. It is clearly from this picture that there are several possible configurations of the two binding sites present in this data set. First of all, it is important to note that motifs can occur on both strands. Transcription factors indeed bind directly on the double-stranded DNA and therefore motif detection software should take this fact into account. Second, sequences could have either zero, one, or multiple copies of a motif. This example gives an indication of the kind of data that come with a realistic biological data set.

A.1 Palindromic motifs

Palindromic motifs are a special type of transcription factor binding site from a computational point of view. Remember that the DNA double-helix is created by the hybridization of the complementary bases A-T and G-C. A palindromic binding site is a subsequence that is exactly the same as its own reverse complement. As an example: if we take the complement of the sequence `TCACGTGA`, this returns `AGTGCACT` and if we read this string in the reverse

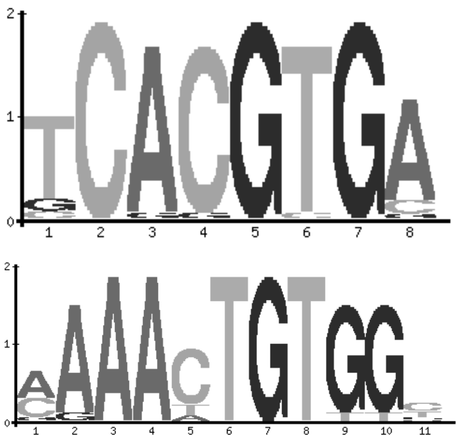


Fig. 8. Logo representation of the transcription factor binding sites present in the MET data set.

order (as the two strands of the DNA helix have opposite orientations) it will be exactly the same as the original sequence. The first motif in Figure 8 is an example of a motif with a palindromic core.

A.2 Gapped motifs

A second class of special motifs are *gapped* motifs or spaced dyads. Such a motif consists of two smaller conserved sites separated by a ga+p or spacer. The spacer occurs in the middle of the motif because the transcription factors bind as a dimer. This means that the transcription factor is made out of two subunits that have two separate contact points with the DNA sequence. The parts where the transcription factor binds to the DNA are conserved but are typically rather small (3-5bp). These two contact points are separated by a non-conserved gap or spacer. This gap is mostly of fixed length but might be slightly variable. Figure 9 shows a logo representation of the FNR binding site in bacteria.

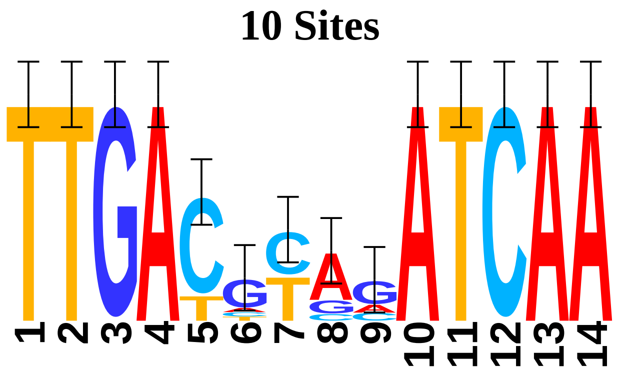


Fig. 9. Logo representation of the FNR binding site. The motif consists of two conserved parts `TTGAy` and `ATCAA` separated by a spacer of length 4.

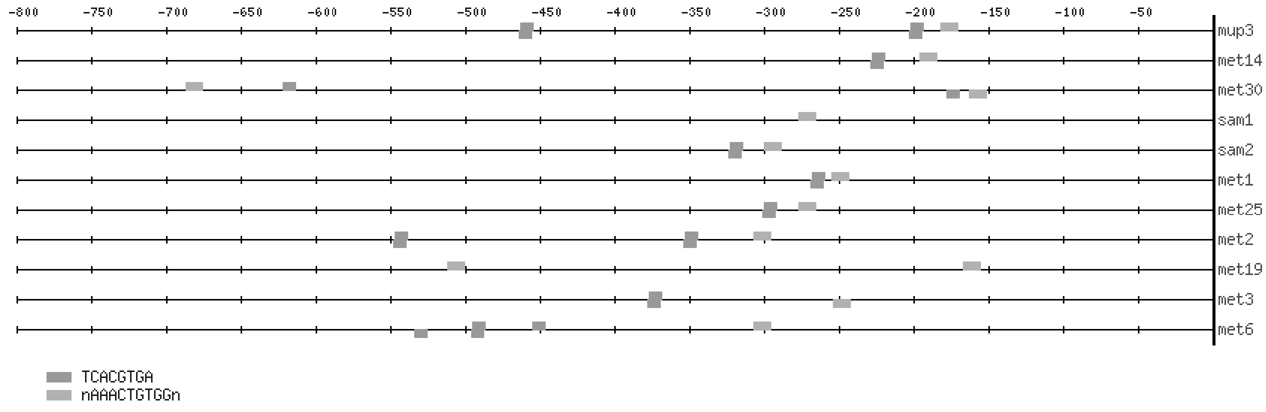


Fig. 7. Schematic representation of the upstream region of a set of coregulated genes. Several possible combinations of the two motifs are present: (1) motifs occur on both strands, (2) some sequences contain one or more copies of the two binding sites, or (3) some sequences do not contain a copy of a motif.

A.3 Cooperatively binding factors and modules

Currently another important research topic is the search for cooperatively binding factors [61]. When only one of the transcription factors binds there is no activation but the presence of two or more transcription factors activates the transcription of a certain gene. If we translate this to the motif finding problem we could either search for individual motifs and try to find, among the list of possible candidates, motifs that tend to occur together. Another possibility is to search for multiple motifs at the same time.

B. Oligonucleotide frequency analysis

The most intuitive approach to extract a consensus pattern for a binding site is a string-based approach, where typically overrepresentation is measured by exhaustive enumeration of all oligonucleotides. The observed number of occurrences of a given motif is compared to the expected number of occurrences. The expected number of occurrences and the statistical significance of a motif can be estimated in many ways. In this section we give an overview of the different methods.

A basic version of the enumeration methods was implemented by van Helden *et al.* [58]. They presented a simple and fast method for the identification of DNA binding sites in the upstream regions from families of coregulated genes in *S. cerevisiae*. This method searches for short motifs of five to six base pairs long. First, for each oligonucleotide of a given length, we compute the expected frequency of the motif from all the non-coding, upstream regions in the genome of interest. Based on this frequency table, we compute the expected number of occurrences of a given oligonucleotide in a specific set of sequences. Next, the expected number of occurrences is compared to the actual, counted, number of occurrences in the data set. Finally, we compute a significance coefficient that takes into account the distinct number of oligonucleotides. A binomial statistic is appropriate in the case where there are non-overlapping segments.

Later, van Helden *et al.* [59] extended their method to find spaced dyads, these are motifs consisting of two small conserved separated by a fixed spacer. The spacer can be different for distinct motifs and therefore the spacer is systematically varied between 0 and 16. The significance of this type of motif can be computed based on the combined score of the two conserved parts in the input data or based on the estimated complete dyad frequency from a background data set.

The greatest shortcoming of this method is that there are no variations allowed within an oligonucleotide. Tompa [54] addressed this problem when he proposed an exact method to find short motifs in DNA sequences. Tompa used a different measure than van Helden *et al.* to calculate the statistical significance of motif occurrences. First, for each k-mer s with an allowed number substitutions, the number of sequences in which s is found is calculated. Next, the probability p_s of finding at least one occurrence of s in a sequence drawn from a random distribution is estimated. Finally, the associated z-score is computed as

$$z_s = \frac{N_s - Np_s}{\sqrt{Np_s(1-p_s)}}.$$

z_s gives a measure of how unlikely it is to have N_s occurrences of s given the background distribution. Tompa proposed an efficient algorithm to estimate p_s from a set of background sequences based on a Markov chain

Sinha and Tompa [47] extended this method to include spacers in the motif, which allows more variability of the motif. Since they were working with prokaryotes, they simplified the model by stating that there is only one copy of the binding site in each sequence. Again they computed the z-score based on the counted number of instance N_s of oligonucleotide s and the expected number of occurrences X_s in a random data set X. The problem is to find good estimates of $E(X_s)$ and $\sigma(X_s)$.

An elaborate description of the solution to this problem is presented by Nicodème *et al.* [42]. They lay out the mathematical foundations build around *generating functions* to quantify the expected occurrence of an unrestricted regular expression in a random text. Such a generating function encodes *exactly* all the information relative to the frequency of occurrence of a regular expression in random texts. Nicodème *et al.* show that those generating functions are rational and computable at reasonable cost for most patterns.

Another interesting string-based approach is based on the representation of a set of sequences with a suffix tree [38], [60]. Sagot *et al.* [60] have used suffix trees to search for single motifs in whole bacterial genomes. Marsan *et al.* [38] later extended the method to search for combinations of motifs. The proposed configuration of a structured motif is a set of p motifs separated by a spacer that might be variable. The variability is limited to ± 2bp around an average gap length. They also allow for variability within the binding site. The representation of upstream sequences as suffix trees resulted in an efficient implementation despite the large number of possible combinations.

C. Probabilistic methods

While in the previous section a binding site was modeled as a set of strings, the following methods are all based on a representation of the motif model with a position weight matrix.

C.1 Probabilistic model of sequence motifs

In the simplest model, we have a set of DNA sequences where each sequence contains a single copy of the motif of fixed length. (For the sake of simplicity, we will consider here only models of DNA sequences, but the whole presentation applies directly to sequences of amino-acids.) Except for the motif, a sequence is described as a sequence of independent nucleotides generated according to a single discrete distribution $\theta_0 = (q_0^A, q_0^C, q_0^G, q_0^T)^T$ called the *background model*. The motif θ_W itself is described by what we call a *position weight matrix*, which are W independent positions generated according to different discrete distributions q_i^b:

$$\theta_W = \begin{pmatrix} q_1^A & q_2^A & \cdots & q_W^A \\ q_1^C & q_2^C & \cdots & q_W^C \\ q_1^G & q_2^G & \cdots & q_W^G \\ q_1^T & q_2^T & \cdots & q_W^T \end{pmatrix}.$$

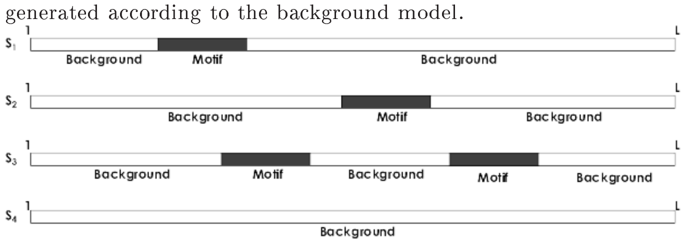
If we known the location a_i of the motif in a sequence S_i, the probability of this sequence given the motif position, the motif matrix, and the background model is

$$P(S_i|a_i, \theta_W, \theta_0) = \prod_{j=1}^{a_i-1} q_0^{S_{ij}} \prod_{j=a_i}^{a_i+W-1} q_{j-a_i+1}^{S_{ij}} \prod_{j=a_i+W}^{L} q_0^{S_{ij}}.$$

Wherever appropriate, we will pool the motif matrix and the background model into a single set of parameters $\theta = (\theta_0, \theta_W)$. For a set of sequences, the probability of the whole set $S = \{S_1, \ldots, S_N\}$ given the *alignment* (i.e., the set of motif positions), the motif matrix, and the background model is

$$P(S|A, \theta) = \prod_{i=1}^{N} P(S_i|a_i, \theta). \qquad (1)$$

Fig. 10. In this basic sequence model, each sequence contains one and only one copy of the motif. The first part of the sequence is generated according to the background model θ_0, then the motif is generated by the motif matrix θ_W, after which the rest of the sequence is again generated according to the background model.



The sequence model is illustrated in Figure 10. The idea of the Expectation-Maximization algorithm for motif finding is to find simultaneously the motif matrix, the alignment position, and the background model that maximize the likelihood of the sequence. Gibbs sampling for motif finding extends Expectation Maximization in a stochastic fashion by not looking for the maximum likelihood configuration but by weighting alignments according to their likelihood.

C.2 Expectation Maximization

One of the first implementation to find a matrix representation of a binding site was a greedy algorithm by Hertz *et al.* [21] to find the site with the highest information content (which is the entropy of the discrete probability distribution represented by the motif matrix). This algorithm was capable of identifying a common motif that is present once in every sequence. This algorithm has been substantially improved over the years [22]. In their latest implementation, CONSENSUS, Hertz and Stormo have provided a framework to estimate the statistical significance of a given information content score based on large deviation statistics.

Within the maximum likelihood estimation framework, Expectation Maximization (EM) is the first choice of optimization algorithm. EM is a two-step iterative procedure for obtaining the maximum likelihood parameter estimates for a model of observed data and missing values. In the expectation step, the expectation of the data and missing values is computed given the current set of model parameters. In the maximization step, the parameters that maximize the likelihood are computed. The algorithm is started with a set of initial parameters and iterates over the two described steps until the parameters have converged.

Since EM is a gradient ascent method, EM strongly depends on the initial conditions. Poor initial parameters may lead EM to converge to a local minimum.

EM for motif finding was introduced by Lawrence and Reilly [30] and was an extension of the greedy algorithm of Hertz *et al.* [21]. It was primarily intended for searching motifs in related proteins the described method could also be applied to DNA sequences. The basic models assumption is that each sequence contains exactly one copy of the motif, which might be reasonable in proteins but is too strict in DNA. The starting position of each motif instance is unknown and is considered as being a missing value from the data. If the motif positions are known then the observed frequencies of the nucleotides at each position in the motif are the maximum likelihood estimates of model parameters. To find the starting positions, each subsequence is scored with the current estimate of the motif model. These updated probabilities are used to re-estimate the motif model. This procedures is repeated until convergence.

Since assuming there is exactly one copy of the motif per sequence is not really biological sound, Bailey and Elkan proposed an advanced EM implementation for motif finding called MEME [3]. Although MEME was also primarily intended to search for protein motifs, MEME can also be applied to DNA sequences. MEME operates in three different modes which correspond to the number of copies of a motif in a sequence. These modes are: exactly one copy of the motif in each sequence, zero or one copy of the motif in each sequence or any number of copy of the motif in each sequence.

To overcome the problem of initialization and getting stuck in local minima, MEME propose to initialize the algorithm with a motif model based on a contiguous subsequence that gives the highest likelihood score. Therefore, each substring in the sequence set is used as a starting point for a one-step iteration of EM. Then the motif model with the highest likelihood is retained and used for further optimization steps until convergence. The corresponding motif positions are then masked and the procedure is repeated.

Finally, Cardon and Stormo proposed an EM algorithm to search for gapped motifs [8].

VI. Gibbs sampling for motif finding

The goal of motif finding is to determine whether the sequences in a set of DNA sequences or proteins share some unspecified patterns called motifs. This task is in contrast with motif searching where a set of sequences is screened for some known motifs.

A. Bayesian inference

In Bayesian inference, we use the likelihood function and observations to infer the probability of different sets of model parameters. Let us suppose we have a data set D whose likelihood can be computed according to some model $\mathcal{M}$ with parameters ω:

$$P(D|\omega, \mathcal{M}).$$

Working with a single fixed model, we drop the notation for the model. Using Bayes' rule $P(A|B) = P(B|A)P(A)/P(B)$, we can write the following to describe the relationship between data and parameters:

$$p(\omega|D) = \frac{P(D|\omega)p(\omega)}{P(D)}.$$

(We use the notation p to describe a continuous probability distribution versus the notation P for a discrete distribution and we assume here that the data has discrete values while the parameters have continuous values). $p(\omega)$ is called the prior distribution and we set it up to contain the *a priori* knowledge that is available about the modeling task (for example, we can choose a prior $p(\omega)$ that penalizes complex models to prevent overfitting). $P(D|\omega)$ is the likelihood function such as the one we described above. $P(D)$ is the *a priori* distribution of the data, which we can actually compute from the prior and the likelihood as $P(D) = \int_\eta P(D|\eta)p(\eta)d\eta$. Finally, $p(\omega|D)$ is the posterior distribution of the parameters after incorporating the data D and it gives the probability distribution of the parameters based on the data *and on the prior*.

B. Sampling methods for optimization

The idea behind sampling methods for optimization is the following. In maximum likelihood, we choose a set of parameters to describe our data by

$$\omega^* = \text{argmax}_\omega P(D|\omega).$$

However, the likelihood function $P(D|\omega)$ contains much more information about the data than just the point estimate $P(D|\omega^*)$. In fact, the posterior distribution $p(\omega|D) = P(D|\omega)P(\omega)/P(D)$ provides a more accurate representation of which parameter values are good candidates to describe our data. For example, if $p(\omega|D)$ is multimodal, the modes provide very different models that describe the data well. Also, we can construct confidence intervals for the parameters based on this distribution while we do not get this information from an optimal point estimate. Thus it is advantageous to work with the full probability distribution instead of limiting ourselves to a point estimate.

In some cases, it is possible to describe the posterior distribution analytically. However, for more complex models such as our sequence model, it is impossible to handle the probability distributions analytically. In that case, several methods are available to generate data according to a complex probability distribution. These are methods such as the Metropolis-Hasting algorithm [41] (which is well-known as the foundation of the simulated annealing algorithm for global optimization), the hybrid Markov-Chain Monte-Carlo method [11], and Gibbs sampling [49].

Here we will describe the general Gibbs sampling method and how it can be applied to motif finding. If we assume that we can generate samples $w^{(i)}$ according to the posterior distribution $p(\omega|D)$, we can use these samples to approximate quantities of interest (possibly using Monte-Carlo integration). For example, we can approximate a *global* solution with maximum posterior probability by tracking the sample with the highest posterior probability if we draw enough sample from the posterior distribution. Further, we can approximate the posterior mean solution

$$\omega^{\text{PME}} = \int_\omega \omega P(\omega|D)d\omega$$

by averaging the samples drawn from the posterior distribution.

C. *The missing data problem and data augmentation methods*

Before proceeding to the description of the general Gibbs sampling method, we need to explicit further how sampling can be applied to motif finding. *The idea is to generate plausible motifs and alignments by drawing samples* $(\theta^{(i)}, A^{(i)})$ *from the posterior* $p(\theta, A|S)$. From these samples, we can then track a best motif matrix or alignment or compute an average motif matrix or alignment.

However, we need to make an important semantic distinction. Indeed, the alignment A is a property of the data, not of the model. But, while the set of sequences S is available, the alignment is unknown. If the alignment was available in the form of sequence labels, our task of estimating the motif matrix would be greatly facilitated. So, when we set up the likelihood function $P(S|A, \theta)$, the alignment is in fact missing from our sequence data. Therefore, recovering the alignment is called the *missing data problem* [3]. Moreover, recovering the alignment is often less important than estimating the model parameters θ. We could thus try to set up directly the likelihood $P(S|\theta)$. But writing down this likelihood function directly is next to impossible. It is only by introducing the alignment that we get a simple expression for our likelihood. Simplifying the likelihood by introducing new variables is called the *data augmentation method*.

D. *Markov-chain Monte-Carlo methods*

We can now go into how to sample from the posterior distribution. Actually, Markov-chain Monte-Carlo methods are methods to sample from any distribution $p(x)$ provided it has an appropriate structure. While sampling from an arbitrary one-dimensional probability density can be achieved simply using a uniform random generator on the [0,1] interval and using the cumulative density function, sampling from high-dimensional probability densities is hard. Markov-chain Monte-Carlo method exploit an important property of Markov chains, which is that data generated by a Markov chain will eventually (after a number of samples going to infinity) be generated according to a fixed probability distribution called the equilibrium distribution of the Markov chain.

A Markov chain is a stochastic process that generates a sequence of samples s_i according to the following relationship $p(s_i|s_{i-1}, s_{i-2}, \ldots, s_1) = p(s_i|s_{i-1})$. This relationship means that any value in the sequence depends only on the previous one and is independent of all the earlier ones. This property is called the *Markov property*.

Markov chains have the remarkable property that, starting from an arbitrary initial condition, the distribution of the samples of the Markov chain will converge to an equilibrium distribution called a *stationary* distribution. Thus in practice, after a sufficient number of transient samples (called *burn-in* period), the Markov chain will draw approximately from this stationary distribution. Specifically, if we have that

$$\pi_j T_{jk} = \pi_k T_{kj},$$

then π is a stationary distribution. This condition is called the condition of *detailed balance*.

E. *Gibbs sampling*

Gibbs sampling is a Markov-chain Monte-Carlo method that was introduced by Geman and Geman [16]in the context of image restoration. Tanner and Wong [49] introduced its use for data augmentation problems. *The idea is to describe a complex probability distribution in terms of a Markov chain built with the simpler marginals of the distribution.* Suppose we have only three (possibly continuous) variables described by the probability distribution $P(x_1, x_2, x_3)$, Gibbs sampling will consists of sampling $x_1^{(i+1)}$ according to $P(x_1|x_2^{(i)}, x_3^{(i)})$, then sampling $x_2^{(i+1)}$ according to $P(x_2|x_1^{(i+1)}, x_3^{(i)})$, and then sampling $x_3^{(i+1)}$ according to $P(x_3|x_1^{(i+1)}, x_2^{(i+1)})$. We denote the fact that this Markov chain converges to the joint distribution by the *chain operator*:

$$P(x_1, x_2, x_3) = P(x_1|x_2, x_3) \bigotimes P(x_2|x_1, x_3) \bigotimes P(x_3|x_1, x_2).$$

To prove that Gibbs sampling draws from the joint distribution $P(x_1, \ldots, x_d)$, we would have to show detailed balance (which is easy, see [13]).

F. *The collapsed Gibbs sampler*

For motif finding, we thus want to build a Gibbs sampler to sample from $P(\theta, A|S)$. However, many variables are now involved, which leaves a great deal of leeway in how the exact sampling is set up. In the basic Gibbs sampler with three variables as an illustration, we have

$$P(x_1, x_2, x_3) = P(x_1|x_2, x_3) \bigotimes P(x_2|x_1, x_3) \bigotimes P(x_3|x_1, x_2).$$

But in some cases, we maybe able to *group* variables together, for example, we may have

$$P(x_1, x_2, x_3) = P(x_1|x_2, x_3) \bigotimes P(x_3, x_2|x_1)$$

with $P(x_3, x_2|x_1) = P(x_3|x_2, x_1)P(x_2|x_1)$. Or we maybe able to *collapse* one of the variables

$$P(x_1, x_2, x_3) = P(x_3|x_1, x_2)(P(x_1|x_2) \bigotimes P(x_2|x_1)).$$

Liu [33] showed that the collapsed Gibbs sampler converges faster than the grouped Gibbs sampler, which itself converges faster than the basic Gibbs sampler.

For motif finding, Liu then proposed to set up a collapsed Gibbs sampler as

$$P(\theta, A|S) = P(\theta|A, S)\left(\bigotimes_{i=1}^{N} P(a_i|a_1, \ldots, a_{i-1}, a_{i+1}, a_N, S)\right),$$

where the chain directly approximate $P(A|S)$.

G. *The basic algorithm for Gibbs sampling for motif finding*

In the previous subsections, we have discussed the main ideas behind Gibbs sampling for motif finding. The derivation of the exact algorithm as was presented by Lawrence [29] and by Liu [35] is more technical; we do not cover it here and refer to the publication by Liu for the technical details. Shortly said, the algorithm is basically the Markov chain described above, but the computation of the probability distributions involves the use of multinomial probability distributions (for the probability of the data based on the likelihood function presented in Section V-C.1 and on the motif matrix and the background model) and of Dirichlet probability distributions (for the probability of the parameters of the motif matrix). The derivation of the collapsed Gibbs sampler involves several properties of integrals of Dirichlet distributions and a number of approximations is used to speed up the algorithm further. To be concrete, we present the resulting algorithm in Table V.

H. *Extended Gibbs sampling methods*

Several groups proposed advanced methods to fine-tune the Gibbs sampling algorithm for motif finding in DNA sequences.

A first version of the Gibbs sampling algorithm that was especially tuned towards finding motif in DNA sequences is AlignACE [44] and this version was later refined [24]. This algorithm was first reported to be used for the analysis of gene clusters. Several modifications were made in AlignACE with respect to the original Gibbs sampling algorithm. First, one motif at the time was retrieved and the positions were masked instead of simultaneous multiple motif searching. Second, AlignACE was implemented with a fixed single nucleotide background model based on base frequency in the sequence set. Also, both strands were included in the search. Finally, in the latest version, the *maximum a priori* likelihood score (MAP) was used to judge different motifs. The MAP was approximated by $N \log R$ where N is the number of aligned sites and R is the degree of overrepresentation.

BioProspector [36] also uses a Gibbs sampling strategy to search for common motifs in the regulatory region of coexpressed genes. In this implementation various extensions are proposed. First, BioProspector uses zero to third-order Markov background models. The predictive update formula is changed in such a way that the probability of the instance being generated by the background model is given by this higher-order background model:

$$\begin{aligned} P_{\mathrm{b}}(x) &= P(b_l)P(b_{l+1}|b_l)P(b_{l+2}|b_{l+1}b_l) \\ &\quad \ldots P(b_{l+W-1}|b_{l+W-2}b_{l+W-3}b_{l+W-4}). \end{aligned}$$

Second, the core sampling step was replaced by a *threshold sampler*. This threshold sampling step was incorporated to estimate the number of copies of a motif in a sequence. The program defines two threshold T_L and T_H. Instances with a score W_x higher than T_{H} will be automatically selected while there will be one motif sampled from those motifs that have a score between T_{L} and T_{H}. T_{H} is set proportional to the product of the average length of the input sequences and the motif width. T_{L} is initialized at 0 and linearly increase till it reaches the value of $T_{\mathrm{H}}/8$. This threshold sampling step ensures faster convergence. As another modification, BioProspector proposes two possible alternative motif models. The first possibility is to search for palindromic motifs. The second possibility is to search for a gapped version of the motif model, where the motif consists of two blocks separated by a gap of variable length. The gapped version searches for two motifs at the same time that occur within a given range.

Within the INCLUSive framework [53], we designed the Motif Sampler [51], [52] specifically to search in sets of upstream sequences from groups of coexpressed genes. Such groups typically come from a cluster analysis of the expression profiles. Since the results of clustering is known to be subject to noise, only a subset of the set coexpressed genes will be actually coregulated and have one or more copies of the binding. Therefore it is important to have an algorithm that can cope with this noise. Motif Sampler uses the framework of the probabilistic sequence model to estimate the number of copies of a motif in a sequence. For each sequence in the data set the number of copies of the motif is estimated, which is more accurate than earlier methods. Furthermore, we demonstrated [52] that the use of a higher-order background model build from an independent data set significantly improves the performance of the Gibbs sampling algorithm. On our web site (see Table IV), we also provide precompiled background models for several organisms (*A. thaliana, S. cerevisiae, E. coli, H. pylori, C. elegans*). To exemplify the improvements obtained by further refinements of the Gibbs sampling strategy, we report here briefly the use of higher-order background models on a data set of coregulated genes from plants. The data set consists of 33 genes known to be regulated in part by the G-box transcription factor, which is linked to the light response of plants. Additionally, noisy sequences not suspected to contain an active motif are added gradually. Figure 11 presents the total number of times (out of 10 runs) that the algorithm finds the correct motif for three different background models when adding from 0 to 50 noisy sequences. The background models are order 0 and order 3 based on a reference set or on the data only. A background model of order 0 corresponds to the classical background model of earlier versions of Gibbs sampling for motif finding. We can observe that the performance of the higher-order algorithms is more robust to the addition of noisy sequence than that of the zero-order algorithm. The improved robustness of the method thanks to the higher-order background model is discussed extensively in [51].

Ann_spec [62] has a slightly different approach to model the motif. The motif model is represented with a sparsely encoded perceptron with one processing unit. The weights of the perceptron resemble the position weight matrix. This model is based on the approximation of the total protein binding energy by the sum of partial binding energies at the individual nucleotides in the binding sites. The use

TABLE V

Basic Gibbs sampling algorithm for motif finding.

```
INPUT: A set of sequences $S$ and the length $W$ of the motif to search.

1. Compute the background model $\theta_0$ from the nucleotide frequencies observed in $S$
2. Initialize the alignment vector $A = \{a_i | i = 1, \ldots, N\}$ uniformly at random
3. For each sequence $S_z, z = 1, \ldots, N$,
  (a) Create subsets $\tilde{S} = \{S_i | i \neq z\}$ and $\tilde{A} = \{a_i | i \neq z\}$
  (b) Compute $\theta_W$ from the segments indicated by $\tilde{A}$
  (c) Assign to each possible alignment start $(x_{ij}; i \neq z, j = 1, \ldots, L_i - W + 1)$ in $S_z$ a weight $W(x_{ij})$ given
      by the probability that the corresponding segment is generated by the motif versus the background:

      $$W(x_{ij}) = \frac{P((S_{ij}, \ldots, S_{i(j+W-1)})|\theta_W)}{P((S_{ij}, \ldots, S_{i(j+W-1)})|\theta_0)} = \prod_{k=1}^{W} \frac{q_i^{S_i(j+k-1)}}{q_0^{S_i(j+k-1)}}$$

  (d) For $i \neq z$, draw new alignment positions $a_i$ according to the normalized probability distribution
      $W(x_{ij}) / \sum_{k=1}^{L_i-W+1} W(x_{ij})$
4. Repeat Step 3 until the Markov chain reaches convergence (fixed number of iterations)

OUTPUT: A motif matrix $\theta_W$ and an alignment $A$.
```

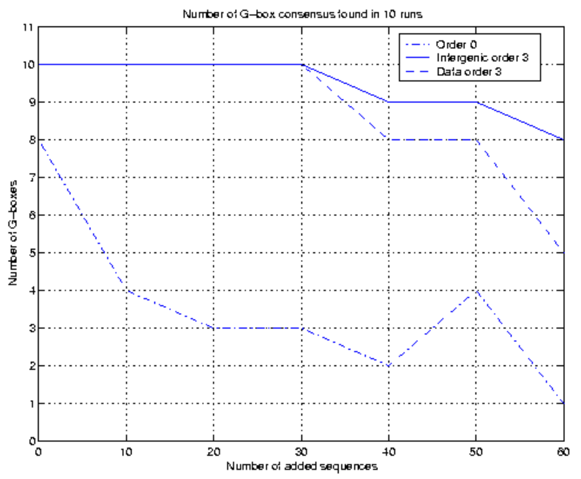


Fig. 11. Total number of times the G-box motif is found in 10 repeated runs of the tests for three different background models. The data set consists of the 33 G-box sequences and a fixed number of added noisy sequences.

of a perceptron is also justified by the fact that it can be used to approximate posterior probability distributions. A gradient descent training method is used to find the parameters of the perceptron. For the training set for the perceptron, positive examples are selected using a Gibbs sampling procedure. Negative examples can be either constructed from random sequence or from genomic data. To improve the specificity of the motif model, a background model based on an independent data set is preferred.

Ann_spec was recently extended to search for co-operatively acting transcription binding factors by GuhaThakurta and Stormo [18]. Co-Bind searches for two motifs simultaneously by combining the weights that optimize the objective functions of the two individual perceptrons. The identification of two motifs simultaneously improved significantly the detection of the true motifs compared to the classical methods searching for one motif at the time.

McCue *et al.* have used a Gibbs motif finding algorithm for phylogenetic footprinting [39]. They also proposed a motif model that accounts for palindromic motifs. Their most important contribution lies in the use of a position-specific background model estimated with a Bayesian segmentation model [34]. This model accounts for the varying composition of the DNA upstream of a gene.

VII. INCLUSive: Integrated CLUstering, Upstream sequence retrieval, and motif Sampling

Analysis of microarray experiment is not restricted to a single cluster experiment. Inferring "biological knowledge" from a microarray analysis usually involves a complete analysis going from preprocessing, sequential use of distinct data preparation steps to the use of different complex procedures that make predictions on the data. Clustering predicts whether genes behave similarly while motif finding aims at retrieving the underlying mechanism of this similar behavior. These data-mining procedures make thus predictions about the same biological system. These predictions are in the best case consistent with each other but they can also contradict each other. Combining these methods into a global approach therefore increases their relevance for biological analysis. Moreover, this integration also allows the optimal matching of the different procedures (such as the quality requirements in adaptive quality-based clustering that reduce the noise level for Gibbs sampling for motif finding). Furthermore, such global approaches require extensive integration at the information technology level. Indeed, as is often underestimated, the collection of data from multiple data sources and transformation of the output of one algorithm to the input of the next algorithm are often tedious tasks.

To make such an integrated analysis of microarray data possible, we have developed and made publicly available our INCLUSive web tool (INtegrated CLustering, Upstream sequence retrieval, and motif Sampling; `http://www.esat.kuleuven.ac.be/~dna/BioI`). In the first step, starting preprocessed measurements, the tool lets the user perform adaptive quality-based clustering on the microarray data. This clustering results in lists of genes that are coexpressed. These genes are identified by their gene names and accession numbers (which are references to the databases where gene sequences have been deposited). In the second step, the tool collects, based on the gene identifiers, upstream (promoter) sequences for selected clusters from several publicly available data sources. In the third step, the tool lets the user perform motif finding (using our Motif Sampler) on the DNA sequences retrieved after clustering. As an output, the tool provides the different motifs discovered as motif matrices and logos, it also provides the candidate binding sites identified by the algorithm as well as a series of scores assessing the quality of the candidate motifs. Finally, it also represent the different candidate motifs graphically for easy visualization of the results.

As an illustration of the results obtained by combined adaptive quality-based clustering and the Motif Sampler, we show the results of motif finding on a microarray experiment in plants. The data is a microarray experiment on the response to mechanical wounding of the plant *Arabidopsis Thaliana*. The microarray consists of 150 genes related to stress response in plants. The experiment consists of expression measurements for those 150 genes at 7 time points following wounding (after 30 min, 60 min, 90 min, 3h, 6h , 9h, and 24 h). The expression data was clustered using adaptive quality-based clustering with a significance level of 95%. Four clusters where identified that contained at least 5 genes and those were selected for motif finding. The Motif Sampler was used to search for 6 motifs of length 8bp and for 6 motifs of length 12bp. A background model of order 3 was selected as it gave the most promising results. The analysis was repeated 10 times and only the motifs identified in at least 5 runs were retained. Table VI presents the motifs found. In the first column, the cluster is identified together with the number of genes it contains. The second column gives the consensus of the motif found. The consensus of a motif is the dominant DNA pattern in the motif described using a degenerate alphabet (e.g., r = A/G); capitals are for strong positions while lower letters are for degenerate positions. The third column gives the number of times this motif was found in the 10 runs. The fourth column gives matching known motifs found in the PlantCARE database [31], if any. Finally, the last column gives a short explanation of the matching known motifs.

VIII. Conclusion

We have presented algorithmic methods for the analysis of microarray data for motif finding. Microarrays are a powerful technique to monitor the expression of thousands of genes and they have become a key techniques for biologists attempting to unravel the regulation mechanisms of genes in an organism. After reviewing the basics of microarray technology, we introduced some concepts from molecular biology to describe how transcription factors recognize binding sites to control gene activation. We then introduced the strategy of integrating clustering (to detect groups of potentially coregulated genes) with motif finding (to detect the DNA motifs that control this coregulation). We presented several clustering techniques (such as hierarchical clustering, K-means, self-organizing maps, quality-based clustering, and our adaptive quality-based clustering) and discussed their respective advantages and shortcomings. We also discussed the preprocessing steps necessary to prepare microarray data for clustering: normalization, nonlinear transformation, missing value replacement, filtering, and rescaling. We also presented several strategies to validate the results of clustering biologically as well as statistically. Turning to motif finding, we described the two main classes of methods for motif finding: word counting and probabilistic sequence models. We focused on the particular technique of Gibbs sampling for motif finding. After reviewing the basic ideas underlying this Markov-chain Monte-Carlo method, we discussed several extensions that improve the effectiveness of this method in practice. We introduced our Motif Sampler, which in particular includes the use of higher-order background models that increase the robustness of Gibbs sampling for motif finding. Finally, we briefly presented our integrated web tool INCLUSive that allows the easy analysis of microarray data for motif finding. Furthermore, we illustrated the different steps of this integrated data analysis at the hand of several practical examples.

As a conclusion, we emphasize that a major endeavor of bioinformatics is to develop methodologies that integrate multiple types of data (here expression data together with sequence data) to obtain robust and biologically relevant results in an efficient and user-friendly manner. Only such powerful tools can deliver the necessary support for 21st-century molecular biology.

References

[1] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, and A.J. Levine, *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays*, Proc. Natl. Acad. Sci. USA **96** (1999), 6745–6750.

[2] F. Azuaje, *A cluster validity framework for genome expression data*, Bioinformatics **18** (2002), 319–320.

[3] T. L. Bailey and C. Elkan, *Unsupervised learning of multiple*

TABLE VI

Results of the motif search in four clusters from a microarray experiment on mechanical wounding in *A. thaliana* for the third-order background model.

Cluster	Consensus	Runs	PlantCARE	Descriptor
1	TAArTAAGTCAC	7/10	TGAGTCA	tissue specific GCN4-motif
(11 seq.)			CGTCA	MeJA-responsive element
	ATTCAAATTT	8/10	ATACAAAT	element associated to GCN4-motif
	CTTCTTCGATCT	5/10	TTCGACC	elicitor responsive element
2	TTGACyCGy	5/10	TGACG	MeJa responsive element
(6 seq.)			(T)TGAC(C)	Box-W1, elicitor responsive element
	mACGTCACCT	7/10	CGTCA	MeJA responsive element
			ACGT	Abcissic acid response element
3	wATATATATmTT	5/10	TATATA	TATA-box like element
(5 seq.)	TCTwCnTC	9/10	TCTCCCT	TCCC-motif, light response element
	ATAAATAkGCnT	7/10	-	-
4	yTGACCGTCCsA	9/10	CCGTCC	meristem specific activation of H4 gene
(5 seq.)			CCGTCC	A-box, light or elicitor responsive element
			TGACG	MeJA responsive element
			CGTCA	MeJA responsive element
	CACGTGG	5/10	CACGTG	G-box light responsive element
			ACGT	Abcissic acid response element
	GCCTymTT	8/10	-	-
	AGAATCAAT	6/10	-	-

motifs in biopolymers using expectation maximization, Machine Learning **21** (1995), 51–80.

[4] A. Ben-Dor, N. Friedman, and Z. Yakhini, *Class discovery in gene expression data*, Proceedings Recomb 2001, 2001, pp. 31–38.

[5] A. Ben-Dor, R. Shamir, and Z. Yakhini, *Clustering gene expression patterns*, J. Comput. Biol. **6** (1999), 281–297.

[6] M. Bittner, P. Meltzer, Y. Chen, Y. Jiang, E. Seftor, M. Hendrix, M. Radmacher, R. Simon, Z. Yakhini, A. Ben-Dor, N. Sampas, E. Dougherty, E. Wang, F. Marincola, C. Gooden, J. Lueders, A. Glatfelter, P. Pollock, J. Carpten, E. Gillanders, D. Leja, K. Dietrich, C. Beaudry, M. Berens, D. Alberts, and V. Sondak, *Molecular classification of cutaneous malignant melanoma by gene expression profiling*, Nature **406** (2000), 536–540.

[7] P. Bucher, *Regulatory elements and expression profiles*, Curr. Opin. Struct. Biol. **9** (1999), 400–407.

[8] L.R. Cardon and G.D. Stormo, *Expectation maximization for identifying protein-binding sites with variable lengths from unaligned DNA fragments*, J. Mol. Biol. **223** (1992), 159–170.

[9] R.J. Cho, M.J. Campbell, E. A. Winzeler, L. Steinmetz, A. Conway, L. Wodicka, T. G. Wolfsberg, A. E. Gabrielian, D. Landsman, D. J. Lockhart, and R. W. Davis, *A genome-wide transcriptional analysis of the mitotic cell cycle*, Mol. Cell **2** (1998), 65–73.

[10] F. De Smet, J. Mathys, K. Marchal, G. Thijs, B. De Moor, and Y. Moreau, *Adaptive quality-based clustering of gene expression profiles*, Bioinformatics (2002), in press.

[11] A. D. Duane, S. amd Kennedy, B. J. Pendleton, and D. Roweth, *Hybrid Monte Carlo*, Physics Letters B **195** (1990), 216–222.

[12] D. J. Duggan, M. Bittner, Y. Chen, P. Meltzer, and J. M. Trent, *Expression profiling using cDNA microarrays*, Nat. Genet. **21** (1999), no. 1 Suppl., 10–14.

[13] R. Durbin, S. Eddy, A. Krogh, and G. Mitchison, *Biological sequence analysis: probabilistic models of proteins and nucleic acids*, Cambridge University Press, 1998.

[14] M.B. Eisen, P.T. Spellman, P.O. Brown, and D. Botstein, *Cluster analysis and display of genome-wide expression patterns*, Proc. Natl. Acad. Sci. USA **95** (1998), 14863–14868.

[15] C. Fraley and E. Raftery, *MCLUST: Software for model-based cluster analysis*, Journal of Classification **16** (1999), 297–306.

[16] D. Geman and S. Geman, *Stochastic relaxation, Gibbs distribution and Bayesian restoration of images*, IEEE Trans. PAMI **6** (1984), no. 6, 721–741.

[17] D. Ghosh and A.M. Chinnaiyan, *Mixture modelling of gene expression data from microarray experiments*, Bioinformatics **18** (2002), 275–286.

[18] D. GuhaThakurta and G.D. Stormo, *Identifying target sites for cooperatively binding factors*, Bioinformatics **17** (2001), 608–621.

[19] T. Hastie, R. Tibshirani, M.B. Eisen, A. Alizadeh, R. Levy, L. Staudt, W.C. Chan, D. Botstein, and P. Brown, *'gene shaving' as a method for identifying distinct sets of genes with similar expression patterns*, Genome Biology (2000), research0003.1-0003.21.

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

[21] G.Z. Hertz, G.W. Hartzell, and G.D. Stormo, *Identification of consensus patterns in unaligned DNA sequences known to be functionally related*, Comput. Appl. Biosci. **6** (1990), 81–92.

[22] G.Z. Hertz and Gary D. Stormo, *Identifying DNA and protein patterns with statistically significant alignments of multiple sequences*, Bioinformatics **15** (1999), no. 7/8, 563–577.

[23] L.J. Heyer, S. Kruglyak, and S. Yooseph, *Exploring expression data: Identification and analysis of coexpressed genes*, Genome Res. **9** (1999), 1106–1115.

[24] J.D. Hughes, P.W. Estep, S. Tavazoie, and G.M. Church, *Computational identification of cis-regulatory elements associated with groups of functionally related genes in Saccharomyces cerevisiae*, J. Mol. Biol. **296** (2000), 1205–1214.

[25] Quackenbush J., *Computational analysis of microarray data*, Nat. Rev. Genet **2** (2001), 418–427.

[26] L. Kaufman and P.J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*, Wiley, New York, 1990.

[27] M.K. Kerr and G.A. Churchill, *Bootstrap cluster analysis: Assessing the reliability of conclusions from microarray experiments*, Proc. Natl. Acad. Sci. USA **98** (2001), 8961–8965.

[28] T. Kohonen, *Self-organizing maps*, Springer-Verlag, Berlin, 1997.

[29] C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wootton, *Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment*, Science **262** (1993), 208–214.

[30] C.E. Lawrence and A.A. Reilly, *An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences*, Proteins **7** (1990), 41–51.

[31] M. Lescot, P. Déhais, G. Thijs, K. Marchal, Y. Moreau, Y. Van de Peer, P. Rouzé, and S. Rombauts, *PlantCARE, a database of plant cis-acting regulatory elements and a portal to tools for in silico analysis of promoter sequences*, Nucleic Acids Res. **30** (2002), 325–327.

[32] R.J. Lipschutz, S.P.A. Fodor, T.R. Gingeras, and D.J. Lockheart, *High density synthetic oligonucleotide arrays*, Nature Genet Suppl. **21** (1999), 20–24.

[33] J.S. Liu, *The collapsed Gibbs sampler in Bayesian computations with applications to a gene regulation problem*, J. Am. Stat. Assoc. **89** (1994), no. 427, 958–966.

[34] J.S. Liu and C.E. Lawrence, *Bayesian inference on biopolymer models*, Bioinformatics **15** (1999), 38–52.

[35] J.S. Liu, A.F. Neuwald, and C.E. Lawrence, *Bayesian models for multiple local sequence alignment and Gibbs sampling strategies*, J. Am. Stat. Assoc. **90** (1995), no. 432, 1156–1170.

[36] X. Liu, D.L. Brutlag, and J.S. Liu, *BioProspector: discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes*, Proc. Pacific Symposium on Biocomputing, vol. 6, 2001, pp. 127–138.

[37] A.V. Lukashin and R. Fuchs, *Analysis of temporal gene expression profiles: clustering by simulated annealing and determining the optimal number of clusters*, Bioinformatics **17** (2001), 405–414.

[38] L. Marsan and M.-F. Sagot, *Algorithms for extracting structured motifs using a suffix tree with application to promoter and regulatory site consensus identification*, J. Comp. Biol. **7** (2000), 345–360.

[39] L.A. McCue, W. Thompson, C.S. Carmack, M.P. Ryan, J.S. Liu, V. Derbyshire, and C.E. Lawrence, *Phylogenetic footprinting of transcription factor binding sites in proteobacterial genomes*, Nucleic Acids Res. **29** (2001), 774–782.

[40] H.W. Mewes, D. Frishman, C. Gruber, B. Geier, D. Haase, A. Kaps, K. Lemcke, G. Mannhaupt, F. Pfeiffer, C. Schuller, S. Stocker, and B. Weil, *MIPS: a database for genomes and protein sequences*, Nucleic Acids Res. **28** (2000), 37–40.

[41] R. M. Neal, *Bayesian learning for neural networks*, Lecture Notes in Statistics, no. 118, Springer, New York, 1996.

[42] P. Nicodème, B. Salvy, and Ph. Flajolet, *Motif statistics*, Theor. Comp. Sci. (2002), in press.

[43] U. Ohler and H. Niemann, *Identification and analysis of eukaryotic promoters: recent computational approaches*, Trends Genet. **17** (2001), no. 2, 56–60.

[44] F.P. Roth, J.D. Hughes, P.W. Estep, and G.M. Church, *Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole genome mRNA quantitation*, Nature Biotech. **16** (1998), 939–945.

[45] G. Schwarz, *Estimating the dimension of a model*, Ann. Stat. **6** (1978), 461–464.

[46] G. Sherlock, *Analysis of large-scale gene expression data*, Curr. Opin. Immunol. **12** (2000), 201–205.

[47] S. Sinha and M. Tompa, *A statistical method for finding transcription factor binding sites*, Proc. 8th Intl. Conf. Intelligent Systems for Molecular Biology (San Diego, USA), vol. 8, 2000, pp. 37–45.

[48] P. Tamayo, D. Slonim, J. Mesirov, Q. Zhu, S. Kitareewan, E. Dmitrovsky, E.S. Lander, and T. R. Golub, *Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation*, Proc. Natl. Acad. Sci. USA **96** (1999), 2907–2912.

[49] Martin A. Tanner and Wing Hung Wong, *The calculation of posterior distributions by data augmentation. with discussion and with a reply by the authors*, Journal of the American Statistical Association **82** (1987), no. 398, 528–550. MR 88h:62050

[50] S. Tavazoie, J.D. Hughes, M.J. Campbell, R.J. Cho, and G.M. Church, *Systematic determination of genetic network architecture*, Nature Genet. **22** (1999), no. 7, 281–285.

[51] G. Thijs, M. Lescot, K. Marchal, S. Rombauts, B. De Moor, P. Rouzé, and Y. Moreau, *A higher-order background model improves the detection of promoter regulatory elements by Gibbs sampling*, Bioinformatics **17** (2001), no. 12, 1113–1122.

[52] G. Thijs, K. Marchal, M. Lescot, S. Rombauts, B. De Moor, P. Rouzé, and Y. Moreau, *A Gibbs sampling meyhod to detect over-represented motifs in the upstream regions of co-expressed genes*, J. Comput. Biol. **9** (2002), no. 2, 447–464.

[53] G. Thijs, Y. Moreau, F. De Smet, J. Mathys, M. Lescot, S. Rombauts, P. Rouzé, B. De Moor, and K. Marchal, *INCLUSive: INtegrated CLustering, Upstream sequence retrieval and motif Sampling*, Bioinformatics **18** (2002), no. 2, 331–332.

[54] M. Tompa, *An exact method for finding short motifs in sequences, with application to the ribosome binding site problem*, Proc. 7th Intl. Conf. Intelligent Systems for Molecular Biology (Heidelberg, Germany), vol. 7, August 1999, pp. 262–271.

[55] J.T. Tou and R.C. Gonzalez, *Pattern classification by distance functions*, Pattern recognition principles, Addison-Wesley, Reading, MA, 1979, pp. 75–109.

[56] S. J. Triezenberg, *Structure and function of transcriptional activation domains*, Curr. Opin. Genet. Dev. **5** (1995), no. 2, 190–196.

[57] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R.B. Altman, *Missing value estimation methods for DNA microarrays*, Bioinformatics **17** (2001), 520–525.

[58] J. van Helden, B. André, and L. Collado-Vides, *Extracting regulatory sites from upstream region of yeast genes by computational analysis of oligonucleotide frequencies*, J. Mol. Biol. **281** (1998), 827–842.

[59] J. van Helden, A.F. Rios, and J. Collado-Vides, *Discovering regulatory elements in non-coding sequences by analysis of spaced dyads*, Nucleic Acids Res. **28** (2000), no. 8, 1808–1818.

[60] A. Vanet, L. Marsan, A. Labigne, and M.F. Sagot, *Inferring regulatory elements from a whole genome. an analysis of helicobacter pylori sigma^{80} family of promoter signals.*, J. Mol. Biol. **297** (2000), no. 2, 335–353.

[61] T. Werner, *Models for prediction and recognition of eukaryotic promoters*, Mamm. Genome **10** (1999), 71–80.

[62] C.T. Workman and G.D. Stormo, *Ann-spec: a method for discovering transcription binding sites with improved specificity*, Proc. Pacific Symposium on Biocomputing (Honolulu, Hawai), vol. 5, 2000, pp. 464–475.

[63] K.Y. Yeung, C. Fraley, A. Murua, A.E. Raftery, and W.L. Ruzzo, *Model-based clustering and data transformations for gene expression data*, Bioinformatics **17** (2001), 977–987.

[64] K.Y. Yeung, D.R. Haynor, and W.L. Ruzzo, *Validating clustering for gene expression data*, Bioinformatics **17** (2001), 309–318.

[65] K.Y. Yeung and W.L. Ruzzo, *Principal component analysis for clustering gene expression data*, Bioinformatics **17** (2001), 763–774.

[66] J. Zhu and M.Q. Zhang, *Cluster, function and promoter: analysis of yeast expression array*, Proc. Pacific Symposium on Biocomputing, vol. 5, 2000, pp. 467–486.