Supplementary Materials

Methods

1 MOTIF EXTRACTION ALGORITHM (MEX)

MEX is a motif extraction algorithm that extracts statistically significant motifs from sequential data. MEX is a data driven unsupervised algorithm, hence does not need any preprocessing of the data or additional information apart from the data set itself. Furthermore, MEX finds motifs that are not necessarily over-represented in the data.

MEX was originally developed in a linguistic context, as a distillation tool for extracting words from corpora of natural language. As more intuitive, let us first describe the algorithm in its original context.

Consider a corpus of sentences, whose word delimiters have been removed (such as spaces, capital letters, punctuations, etc.). The problem at hand is to uncover the words that have originally constructed the sentences. MEX receives as an input such corpus, consisting of many sequences of a given finite alphabet of size N (e.g. N=26 letters in the English alphabet, N=20 amino acids in proteins and N=4 nucleic acids in the case of DNA). The algorithm uses a directed graph, whose vertices, V, are composed of the letters of the given alphabet, in addition to a 'begin' and an 'end' vertices. A set of ordered pairs of vertices (directed edges) represent the order in which the letters appear in the corpus. For example, the edge e(t,h), represents a connection from the vertex 't' to the vertex 'h', which means that the letter 'h' appears at some point along the corpus after the letter 't'. MEX loads the given corpus onto a directed graph, one sentence after the other. The edges representing each sentence are built, starting with the 'begin' vertex, followed by the letters composing the sentence, one after the other, and ending with the 'end' vertex. This way, ordered paths are created in the graph, such that each sentence is represented by a path. Each path is saved by MEX and will be used as a search path for patterns. This procedure is demonstrated in figure 1.1.

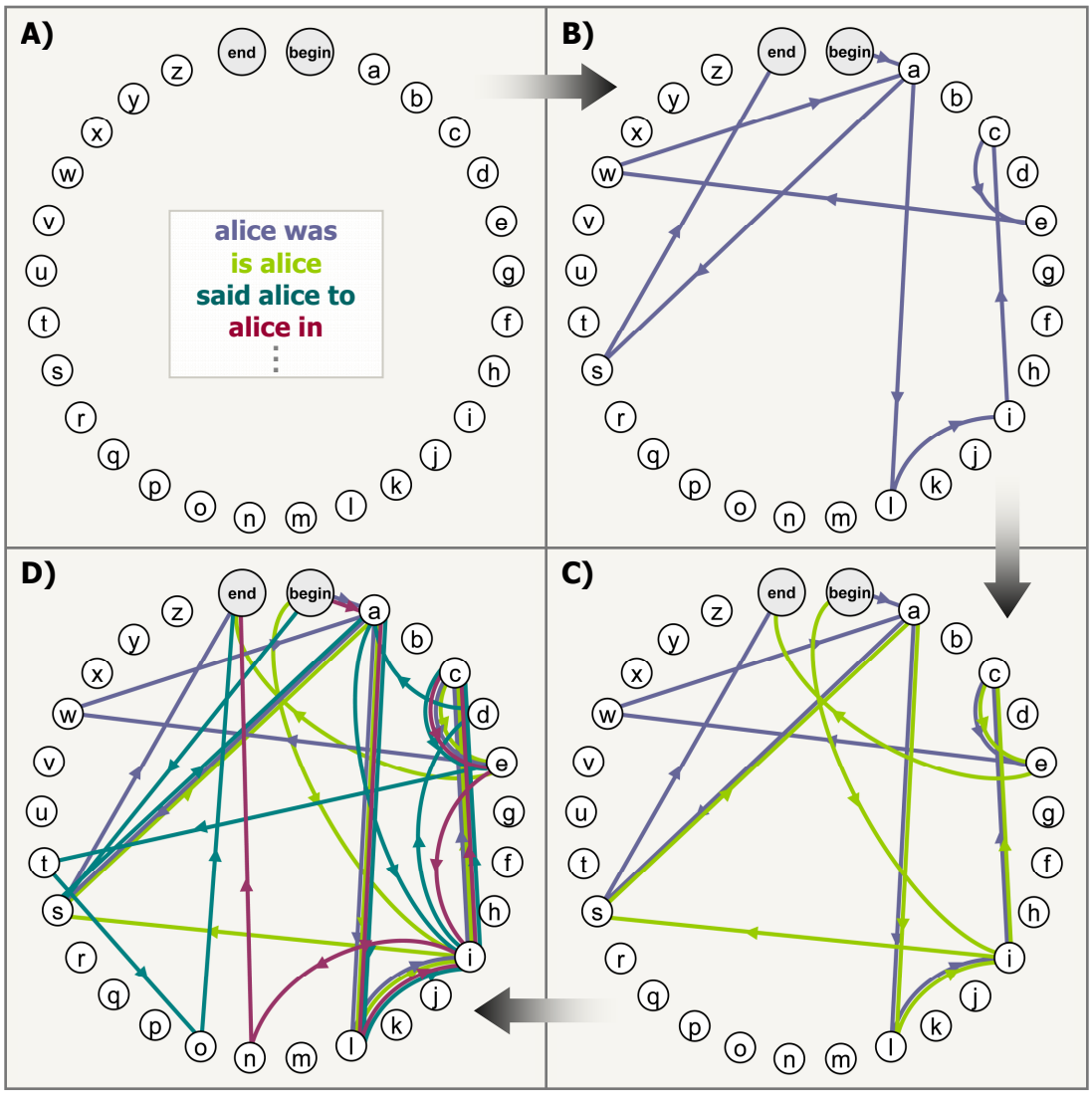


Figure 1.1 MEX loads the corpus onto a directed graph, one sentence after the other. The graph is composed of vertices representing the letters of the given alphabet, in addition to two vertices representing the beginnings and the endings of sentences (A). One sentence at a time, directed edges are added to the graph, representing the order in which letters appear in each sentence (B-D). The ordered edges composing a sentence are considered a path along the graph. In this example, four paths are loaded onto the graph: 'alicewas' (blue path), 'isalice' (light green path), 'saidalice' (turquoise path) and 'alicein' (red path), one after the other.

Once the entire corpus has been represented as search paths on a directed graph, the algorithm starts searching for statistically significant patterns. Intuitively, for each search path MEX looks for sub-paths that may be considered as candidates for being significant patterns. A sub-path that represents a significant pattern is expected to be shared by other paths throughout the graph, such that these paths will converge into the sub-path at its first vertex, form a bundle along the sub-path and scatter after the sub-path's last vertex. This follows the assumption that at different instances of a given word throughout the corpus, after the word ends, it is likely to find many different possible words following it. In such a case many paths will form a bundle along the sub-path representing the word and scatter immediately after it ends. The vertex after which such a divergence occurs may be considered as the last vertex of the pattern. A similar notion underlies the way MEX searches the start points of patterns, by looking for a divergence of a bundle while going leftwards through a search path. Figure 1.2 demonstrates this idea. The four paths in figure 1.2 converge and form a bundle along the sub-path 'a→l→i→c→e', after which they diverge.

This can be rephrased into a probabilistic language; for each search path (sentence) that is to be explored for patterns, two probability functions are defined, based on information inheres in the complete graph. The first one, P_{Right}, is the right moving ratio of the through-going flux of paths to the incoming flux of paths, which varies along the search path. Starting at the vertex e_1 we define P_{Right} at e_2 as:

$$P_{Right}(e_1, e_2) = p(e_2 \mid e_1) = \frac{\text{total no. of paths passing from } e_1 \text{ to } e_2}{\text{total no. of paths entering } e_1}$$

At e_3 P_{Right} becomes:

$$P_{Right}(e_1, e_3) = p(e_3 \mid e_1 e_2) = \frac{\text{total no. of paths passing from } e_1 \text{ through } e_2 \text{ to } e_3}{\text{total no. of paths passing from } e_1 \text{ to } e_2}$$

And generally:

$$P_{Right}(e_i, e_j) = p(e_j \mid e_i e_{i+1} e_{i+2} \cdots e_{j-1}) = \frac{\text{total no. of paths passing from } e_i \text{ up to } e_{j-1} \text{ and continue to } e_j}{\text{total no. of paths passing from } e_i \text{ up to } e_{j-1}}$$

Similarly, a second function, P_{Left}, is defined as we proceed leftward from some vertex e_j down the search path towards the vertex e_i and examine the left-going ratio of the through-going flux of paths to the incoming flux of paths:

$$P_{Left}(e_j, e_i) = p(e_i \mid e_{i+1} e_{i+2} \cdots e_{j-1} e_j) = \frac{\text{total no. of paths passing from } e_i \text{ to } e_j}{\text{total no. of paths passing from } e_{i+1} \text{ to } e_j}$$

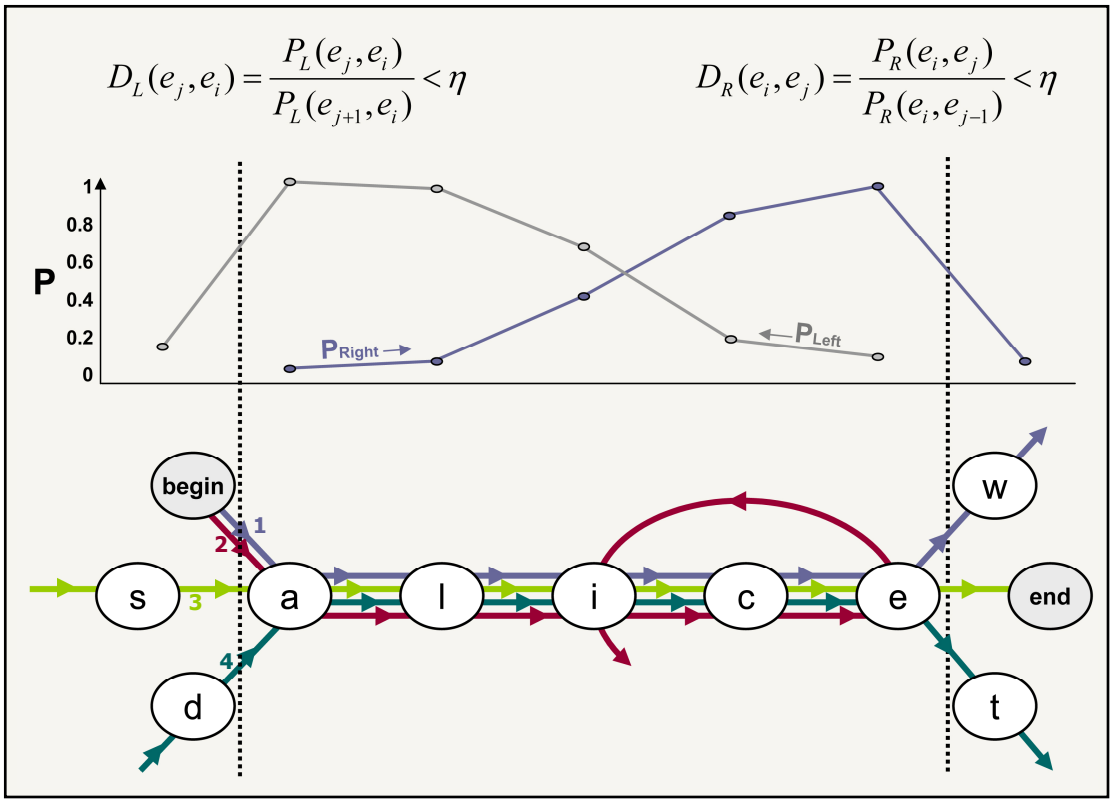


Figure 1.2 A partial view of the graph used by MEX. The search path no. 1, 'alicewas' (blue line), shares the sub-path 'a→l→i→c→e' with three other paths: 'isalice' (2), 'saidalice' (3) and 'alicein' (4). The four paths form a bundle that may constitute a significant pattern. The conditional probabilities P_{Right} and P_{Left}, originating at the vertices 'a' and 'e', respectively, are illustrated for the example shown here. A sharp drop in the right moving probability, P_{Right}, indicates that the paths constructing the bundle have scattered, thus may denote the end of the pattern. Similarly, a sharp drop in P_{Left} may indicate the beginning of the pattern, hence reveal the pattern 'alice'.

Vertex	Conditional Probability Expression	P_{Right}
a	P(a) = 8770 / 109625	0.08
l	P(al | a) = 1046 / 8770	0.12
i	P(ali | al) = 486 / 1046	0.45
c	P(alic | ali) = 397 / 486	0.85
e	P(alice | alic) = 397 / 397	1
w	P(alicew | alice) = 48 / 397	0.12
a	P(alicewa | alicew) = 21 / 48	0.44
s	P(alicewas | alicewa) = 17 / 21	0.81
b	P(alicewasb | alicewas) = 2 / 17	0.12
e	P(alicewasbe | alicewasb) = 2 / 2	1
g	P(alicewasbeg | alicewasbe) = 2 / 2	1
.	.	.
.	.	.
.	.	.

Table 1.1 Calculating right-going conditional probebilities for the search path 'alicewasbegining...'. Probablities are calculated for a given serach path, based on information inheres in the entire graph. The corpus used in this example was the sentences from Alice in wonderland, by Lewis Carroll.

MEX calculates P_{Right} from different starting points to each vertex down the search path. Going rightwards through a sub-path that represents a significant pattern, it is expected that P_{Right} will first increase since other paths join the search path to form a coherent bundle, and then decrease as many paths leave the search path.

In order to demonstrate this, let us examine as a toy problem the corpus of Alice in wonderland, by Lewis Carroll. MEX has received as an input the sentences within Alice in wonderland, after all word delimiters have been removed. Going through the first search path 'alicewasbeginningtogetverytired...' MEX calculates the rightward-going probabilities, P_{Right}, along the path, as demonstrated at table 1.1. MEX starts at the first vertex 'a' and calculates the probability of its appearance in the corpus, $P_{Right}(a)$; as 'a' appears in 8770 cases out of the total of 109625 letters in the corpus, $P_{Right}(a)=\frac{8770}{109625}=0.08$. MEX continues to the next vertex 'l', calculating the probability of its appearance after the previous vertex, i.e. $P_{Right}(al|a)$; in this case, 'l' appears 1046 times after the 8770 instances of 'a', hence $P_{Right}(al|a)=\frac{1046}{8770}=0.12$. MEX continues calculating the rightward-going probabilities $P_{Right}(ali|al)$, $P_{Right}(alic|ali)$ and so on, up to the end of the search path. As can be seen in table 1.1, the rightward-going probabilities initially rise and then drop sharply. Such a dramatic drop may occur owing to the sudden divergence of a coherent bundle, and will be considered as a candidate for terminating a pattern.

We will define the end of a motif as the vertex after which a dramatic drop in the right-moving probabilities is apparent (expressing the divergence of edges from that vertex), and the beginning of a motif as a dramatic drop in the left moving probabilities (expressing the convergence of edges to that vertex).

Formally, let us define a "decrease ratio":

$$D_{Right}\left(e_i, e_j\right)=\frac{P_{Right}\left(e_i, e_j\right)}{P_{Right}\left(e_i, e_{j-1}\right)}$$

$$D_{Left}\left(e_j, e_i\right)=\frac{P_{Left}\left(e_j, e_i\right)}{P_{Left}\left(e_j, e_{i+1}\right)}$$

We will declare e_{j-1} as a candidate end point of the pattern if $D_{Right}(e_i,e_j)$ is smaller then a preset cutoff parameter $\eta<1$. Similarly, e_{i+1} will be declared as candidate start point of a pattern if $D_{Left}(e_j,e_i)<\eta$.

The statistical significance of the decreases in P_{Right} and P_{Left} must be evaluated. P_{Right} and P_{Left} can be regarded as variable-order Markov probability functions. We can define their significance in terms of a null hypothesis stating that $P_{Right}(e_i,e_j) \geq \eta P_{Right}(e_i,e_{j-1})$ and $P_{Left}(e_j,e_i) \geq \eta P_{Left}(e_j,e_{i+1})$, and require that the p-values of both $D_{Right}(e_i,e_j)<\eta$ and $D_{Left}(e_j,e_i)<\eta$ be, on average smaller than a preset threshold parameter $\alpha<1$.

A bundle of coinciding paths whose end-points obey these significance conditions is declared as a possibly significant pattern. Given a search path, we calculate both P_{Right} and P_{Left} from all of the possible starting points, traversing each path leftward and rightward, correspondingly. This technique defines many search-sections, which may be candidates for significant patterns. The most significant ones of these candidates are returned as the outcome patterns for the search path in question.

2 EXPRESSION COHERENCE (EC) METHOD

Rational

The Expression coherence (EC) score is a measure of how clustered a set of genes is in expression space. It may be defined for any gene set for which expression profiles are available. Given an expression profile of N time points for M genes, each gene can be thought of as a point in an N dimensional space, where the ith dimension has the expression level of the gene at the ith time point (Figure 2.1). Given a set of genes, one wishes to determine whether they are tightly clustered, or rather spread "all over the place". One way to accomplish this could be to calculate the "center of mass" of the cloud of genes and then sum over distances of each gene from it (other variations may be to sum over squares of such distances, take standard deviation around that mean etc). Yet, this measure has a clear shortcoming - in cases where the gene cluster is split, say to two, equally sized very tightly clustered subsets, that are yet remote from one another, any deviation-from-mean score will be low. This fails to capture the unique substructure of this gene set, which is composed of two tight sub-clusters. The intuitive reason why this gene set is 'impressive' is that out of P=M*(M-1)*0.5 gene pairs in it p=(M/2)*((M/2)-1) pairs are close (defined below) to each other. So the ratio p/P is a good measure for how tight the cluster is . This is the expression coherence score (see figure 2.1 for illustration).

Biological relevance

Applying the EC score to a set of genes that share a given motif in their promoters, gives a measure of the extent to which the motif may influence expression. Moreover it allows to functionally annotate the motif, by describing the biological conditions in which it governs coherent expression, along with the regulatory effect it exerts (e.g. increased expression in response to a particular stress, or peak in expression at a specific pint during cell cycle). Such analyses can be performed online for the S. cerevisiae genome, via the Motif Analysis Workbench (Lapidot and Pilpel 2003) at http://longitude.weizmann.ac.il/services.html

Algorithm

Given a gene set S , we compute the Euclidean distances between the centered and variance-normalized expression profiles of each of its P=|S|*(|S|-1)*0.5 gene pairs. The EC score is defined as p/P where p is the number of gene pairs whose Euclidian distance is smaller than a threshold distance (D). D is determined based on the distribution of pair-wise distances between expression profiles of all genes in the genome (or more precisely of all genes for which expression level was measured). The original definition of the EC score (Pilpel et al Nat. Genet. 2001) used the 5th percentile as the cutoff for defining “close” expression profiles, but other cutoffs may be applied.

$$EC(S) = \frac{\left|\left\{g_i, g_{j\neq i} \in S\right\}: ExpDist(g_i, g_j) < D\right|}{|S| * \left(|S| - 1\right) \div 2}$$

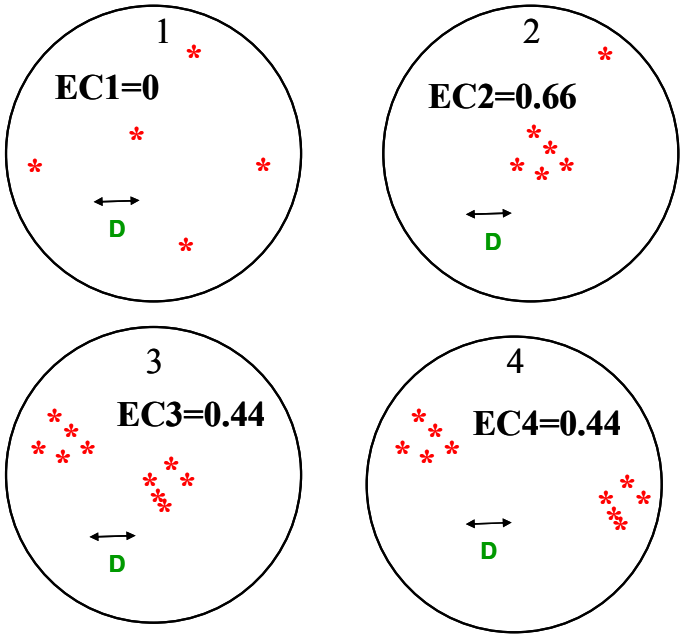


Figure 2.1 EC scores of different scenarios. To illustrate how the EC score is used to measure the extent to which genes are clustered in expression space, we show four scenarios; each disc displays gene sets embedded in a putative expression space. In the first scenario the genes are evenly spread in expression space, no two genes are closer than the threshold distance D, and thus the EC sore is 0. In the second scenario, all genes, but one are tightly clustered and thus the EC score is high, 10/15 possible gene pairs are closer than the threshold D. In scenarios 3 and 4, two tight clusters are observed. The EC score is the same for both cases, despite the fact that the clusters in scenario 3 are closer to one another. This is because in both cases only genes within each of the two subsets are closer than the threshold (20/45 gene pairs) whereas any two genes belonging to different subsets are further away than the threshold. Measures based on distance from the center of mass would not capture the sub-cluster structure depicted in scenarios 3 and 4.

Estimation of EC Score Significance

The significance of an EC score calculated for a set of genes, relies on the set size and on the analyzed condition. Thus for each of the expression time series experiments and for each gene set sizes (varying from 3-100 genes), we selected 100,000 random gene sets and computed an EC score for each such set at each cutoff definition. We define the p-value of a given EC score as the fraction of random sets (of the same size and in the same condition) that scored similarly or higher (Note that this sets a lower bound of 10-5 on the significance that can be assigned to a given EC score). Since we assume that for a given EC score the probability to get the same score for random sets of genes drops with the set size, gene sets larger than 100 are assigned an upper bound approximated p-value, using the randomly sampled sets of size 100.

For large sets of genes even a small deviation from an EC score of 0.05 (the mean value for random sets) can be statistically significant, whereas for very small sets large deviations from an EC score of 0.05 can be expected purely by chance, as demonstrated in (http://bioportal.weizmann.ac.il/~lapidotm/rMotif/html/doc/ECscore.html).

3 EXPRESSION DATA

Table 3.1

Experiment short name	Experiment name[1]
Cell cycle (1)	ExpressDB Cho - cell cycle
Cell cycle (2)	ExpressDB Spellman - cell-cycle alpha
Cell cycle (3)	ExpressDB Spellman - cell-cycle cdc15
Cell cycle (4)	ExpressDB Spellman - cell-cycle cdc28
Cell cycle (5)	ExpressDB Spellman - cell-cycle eluteration
Sporulation	ExpressDB Chu - sporulation
MapK	ExpressDB - MapK
Diaux shift	ExpressDB Gasch environmental response - diaux shift
YPD (1)	ExpressDB Gasch environmental response - YPD1
YPD (2)	ExpressDB Gasch environmental response - YPD2
X media vs. car1	ExpressDB Gasch environmental response - x media vrs car1
YPx media vs. car2	ExpressDB Gasch environmental response - YPx media vrs car2
Nitrogen depletion	ExpressDB Gasch environmental response - Nitrogen Deplation
Amino acid starvation	ExpressDB Gasch environmental response - Amino Acid starv
Acid	ExpressDB Environmental response - Acid
Alkali	ExpressDB Environmental response - Alkali
Diamide	ExpressDB Gasch environmental response - diamide
Hydrogen Peroxide (H2O2)	ExpressDB Environmental response - Peroxide
Constant H2O2	ExpressDB Gasch environmental response - constatnt h2o2
Menadione	ExpressDB Gasch environmental response - Menadione
NaCl	ExpressDB Environmental response - NaCl
Hypo-osmotic	ExpressDB Gasch environmental response - Hypo-osmotic
DTT (1)	Eisen - dtt
DTT (2)	ExpressDB Gasch environmental response - DTT1
DTT (3)	ExpressDB Gasch environmental response - DTT2
Sorbitol (1)	ExpressDB Environmental response - Sorbitol
Sorbitol (2)	ExpressDB Gasch environmental response - sorbitol
DNA damage	Jelinsky - DNA Damage
Cold	Eisen - cold
Heat shock (1)	ExpressDB Environmental response - Heat
Heat shock (2)	Eisen - heat
Heat shock (3)	ExpressDB Gasch environmental response - 37-25 shock
Heat shock (4)	ExpressDB Gasch environmental response - Heat Shock 1
Heat shock (5) & sorbitol	ExpressDB Gasch environmental response - hs 29-33 1m sorbitol
Heat shock (6)	ExpressDB Gasch environmental response - hs 29-33
Heat shock (7)	ExpressDB Gasch environmental response - hs 29-33 No sorbitol
Heat shock (8)	ExpressDB Gasch environmental response - Heat Shock2 (3 time zero)
Heat shock (9)	ExpressDB Gasch environmental response - hs various temp to 37c
Various temp growth	ExpressDB Gasch environmental response - various temp growth
Various temp steady state	ExpressDB Gasch environmental response - var temp steady state

[1] Whole-genome mRNA expression data of 40 time series experiments in *S. cerevisiae,* were obtained from ExpressDB (http://arep.med.harvard.edu/cgi-bin/ExpressDByeast/EXDStart). These time series represent a wide range of natural (e.g. cell cycle) and perturbed conditions. This set of conditions was utilized by us before (Garten et al. Nucleic Acids Res 2005) and a complete list of the conditions is available at http://longitude.weizmann.ac.il/TFLocation/conditions_explist.html

4 CLUSTERING ALGORITHM

We have formulated an iterative method for clustering motifs, according to their sequences and EC scores information. We first initiate clusters by gathering motifs that share some building blocks, or `seeds'. Then, a series of iterations improves the clusters, using various procedures detailed below. The clusters refinement steps include the addition and removal of motifs from existing clusters and splitting and merging of clusters. We have quantified the quality of clusters using several criteria associated with sequential patterns and EC score patterns of the motifs. The metrics as well as the refinement steps are listed below.

The clustering algorithm may be used to cluster any type of sequential data that are linked to numerical data. The input to the algorithm is a set of sequences of a given alphabet (e.g. motifs) and a complementary set of vectors (e.g. EC vectors), holding an additional information that needs to be taken into account in the clustering process.

Our clustering method may be considered 'fuzzy' in the sense that single motifs may belong to several clusters. Additionally, not all motifs must be clustered and may be left as singletons.

Initiating clusters by seeds

Our set of motifs was scanned to find short strings of nucleotides (of length 6) that appear within at least three motifs, to be called `seeds'. Selecting all motifs that contain a given seed defines a preliminary cluster.

Pruning clusters to increase EC tightness

For each motif one defines an EC vector of length 40 whose entries specify the p-values of significantly successful EC experiments (that had passed the FDR criterion). Let us define the space of all these vectors as EC space and define an *EC divergence* measure for a cluster of motifs as the average distance of all pairs of its EC vectors. In order to decide whether to eliminate a motif from a given cluster, we ask whether its presence increases the divergence of the cluster. To decide whether a motif m should be eliminated from a cluster M_C, we compare the EC divergence of M_C with the empirical distribution of EC divergence scores resulting from replacing m with every one of the motifs that lie outside the cluster M_C (that is, with a background sample M_B). The motif will be pruned from the cluster if it does not significantly reduce the cluster's EC divergence, in comparison to motifs from the random background. The deletion of motifs from a cluster occurs after all motifs have been tested, thus the order of tested motifs does not affect their chances of remaining in the cluster. A pseudo code describing this procedure is available in box 4.1. An example is shown in Figure 4.1.

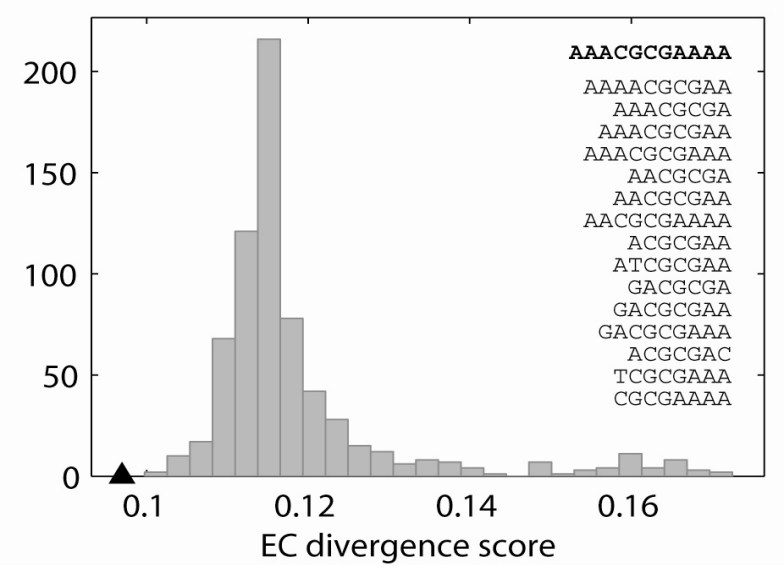


Figure 4.1 An example for testing the contribution of a specific motif to the cluster's tightness. The EC-divergence score of the cluster including the motif AAACGCGAAAA (black triangle) is compared to the empirical distribution of EC-divergence of clusters, in which the motif in question has been replaced with random motifs (histogram). Our null hypothesis claims that the motif does not reduce EC-divergence of the group (which is equivalent to saying that the motif harms the tightness of the cluster). In this example, however, the divergence-score of the cluster with the motif included in it is very small. Hence, we can reject the null hypothesis with a probability value of 0.001 and include the motif in the cluster.

Box 4.1: Pruning clusters to increase EC tightness

Given a set M of motifs, their EC scores vectors, $EC_{i\in M} \in \Re^{40}$, and two disjoint subsets, $M_C, M_B \subset M$ (the cluster in question and a background subset of motifs, respectively), we wish to eliminate from cluster M_C motifs that increase its EC divergence. We will test the contribution of M_C's motifs to its EC divergence, by comparing them to M_B's motifs contribution to M_C's EC divergence.

Pseudo code:

1. Calculate the EC distance between every pair of *EC* vectors in $M_C \bigcup M_B$:
$$ECdist_{ij} = avg(|\, EC_i - EC_j \,|)\ ;\ i,j \in M_C \bigcup M_B$$
2. Calculate M_C's divergence score: $DivScore_{M_C} = avg(ECdist_{ij})\ ;\ i < j \in M_C$.
3. Create a new subgroup M_{ij} by replacing the *i*'th motif of M_C with the *j*'th motif of M_B.
4. Calculate M_{ij}'s divergence score.
5. Repeat steps 3, 4 for all $i \in M_C$, $j \in M_B$ in order to examine the effect of each motif *i* on M_C's tightness.
6. For each motif $i \in M_C$, generate the empirical distribution of divergence scores, as found in the replacement of motif *i* with every motif $j \in M_B$.
7. For each motif $i \in M_C$, calculate the p-value of getting the divergence score of M_C by chance.
8. Compare each p-value to a preset significance value α.
9. Eliminate motifs that are not significantly reducing the divergence of the group in comparison to the randomly sampled motifs.

Expanding clusters

We search for new motifs to be added to the cluster without increasing its EC divergence. To decide whether a motif *m* should be added to a cluster M_C we compare the EC divergence resulting from its addition (M_C+*m*) with the empirical EC divergence distribution resulting from additions of each of the motifs lying outside M_C (that is, in a background sample M_B), one at a time.

At the same time we also require sequential similarity of the new motif to the ones that belong to the cluster. The sequential distance between motifs is defined as the edit distance of their best alignment, not allowing gaps. The sequential distance score, *D*, is normalized between 0 and 1, such that *D*=0 if the short motif is fully contained in the long one and *D*=1 if the motifs have no match at all.

A cluster will be expended by motifs that keep its tightness, as well as being strongly similar to the cluster by sequence. The addition of motifs to a cluster occurs after all motifs in M_B have been tested, thus the order of tested motifs does not affect their chances of being added to the cluster. A pseudo code is available in box 4.2.

Box 4.2: Expending clusters

Given a set *M* of motifs, their EC scores vectors, $EC_{i\in M} \in \Re^{40}$ and two disjoint subsets, $M_C, M_B \subset M$, we wish to expend M_C by similar motifs from a background set M_B that do not increase its EC divergence.

Pseudo code:

1. Find candidates motifs for addition, $\mathrm{M_{cand}} \subset \mathrm{M_B}$, that show strong similarity by sequence to at least one motif in M_C.
2. Calculate the EC distance between every pair of *EC* vectors in $M_C \bigcup M_B$:
$$ECdist_{ij} = avg(|\, EC_i - EC_j \,|)\ ;\ i, j \in M_C \bigcup M_B .$$
3. Create a new candidate subgroup $M_{C,cand}$ by adding M_C a single motif from M_{cand}.
4. Calculate $M_{C,cand}$'s divergence score: $DivScore_{M_{C,cand}} = avg(ECdist_{ij})\ ;\ i < j \in M_{C,cand}$.
5. Create a new test group $M_{C,j}$ by adding M_C a single motif from $M_B \neq M_{cand}$
6. Calculate $M_{C,j}$'s divergence score.
7. Repeat steps 5, 6 for all $j \in M_B \neq M_{cand}$ to generate the empirical distribution of divergence scores of the test groups.
8. Calculate the probability value for getting the divergence score of $M_{C,cand}$ by chance.
9. Expend the group by the current motif candidate if it produces a significantly low divergence score (lower than some preset significance value, α).
10. Repeat steps 3-9 for every motif candidate in respect to the original cluster M_C.

Fusion of clusters

Clusters will be merged if they share a minimum percentage of motifs and are also found to be similar in EC. EC distance between two clusters A and B is defined by a *Fisher* criterion, as the distance between the centers of the clusters, divided by the sum of their standard deviations:

$$F_{A,B} = \frac{\|\mu_A - \mu_B\|}{\|\sigma_A\| + \|\sigma_B\|}$$

μ_A and μ_B are the mean EC vectors of the two EC matrices (the center of each cluster). For each cluster we define σ as the vector of the 40 standard deviations corresponding to the 40 EC experiments. Clusters will be merged if their *Fisher distance*, F, is smaller than some threshold, as long as they also obey the sequential similarity criterion.

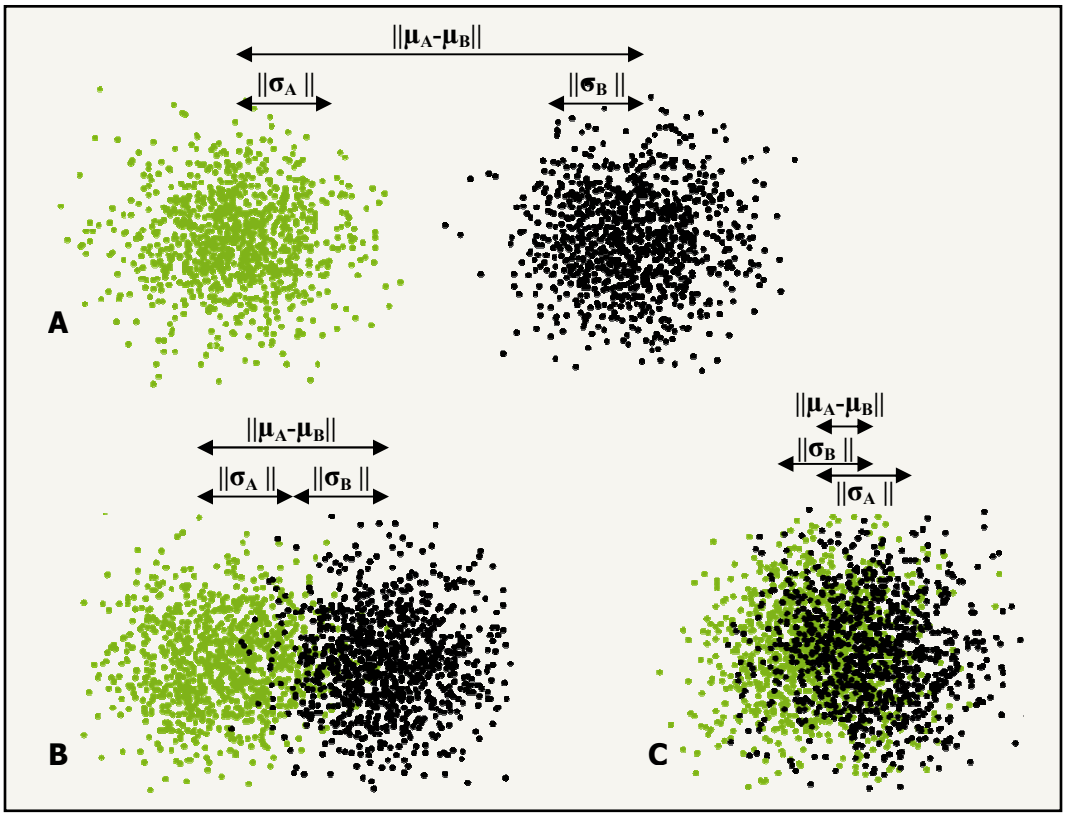


Figure 4.2 A demonstration of the Fisher criterion. The fisher distance for distant clusters exceeds the value 1 (A). The smaller the fisher distance is, the more difficult it gets to distinguish between the clusters (B, C).

Splitting of clusters

Clusters will be split into K smaller clusters if they exceed a given size. Splitting is done using the K-means algorithm on the EC space of the cluster. After applying this indiscriminative step, however, a fusion step is applied, so that unnecessary splitting will be reversed. In this work we have used K=3.

Fine refinement of clusters

The former procedures are applied iteratively in a preset order, to generate clusters that are rather tight in EC and in sequence and differ from each other in sizes, EC patterns and motif sequences. In a final pruning step finer parameters are used. Then the improvement of each cluster is tested with respect to a *cluster score*, assessing the quality of the cluster, and the pruning is accepted or rejected accordingly. The clusters are given a *cluster score*, a heuristic function encapsulating the various measures used in the analysis:

$$ClusterScore = \frac{\sqrt{\frac{SeqDivScore_{cluster}}{SeqDivScore_{others}} \cdot \frac{DivScore_{cluster}}{DivScore_{others}}}}{\left(MC \cdot F_{cluster,others}\right)^2}$$

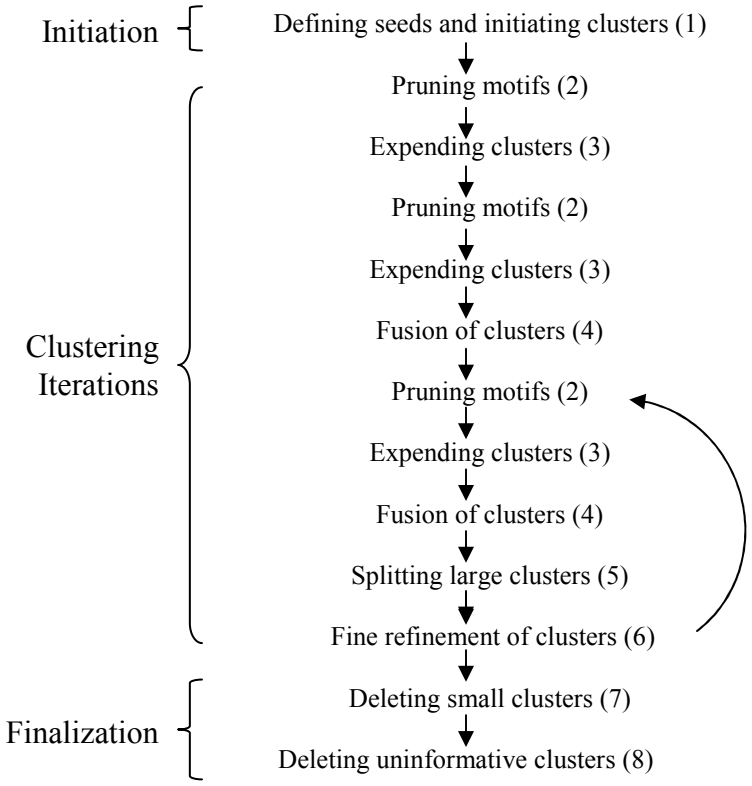
$SeqDivScore_{cluster}$, $DivScore_{cluster}$ are the cluster's sequential and EC divergence scores, respectively (the former is defined similarly to the later, as the average sequential distance of all pairs of motifs within the cluster). $SeqDivScore_{others}$, $DivScore_{others}$ are the sequential and EC divergence scores of all the motifs outside the cluster, respectively. $F_{cluster,others}$ is the EC fisher distance between the cluster and the rest of the motifs, and MC is the number of motifs within the cluster. The smaller the cluster score is, the better the quality of the cluster is considered.

The cluster score quantifies the quality of a cluster in terms of its internal tightness relatively to the background. As affected by many different factors, the cluster score is sensitive to noise. Hence it is only used at a late stage along the algorithm, when clusters are already coherent to a great extent.

Flow of the algorithm

After initiation, cycles of the various iterations occur, gradually improving the clusters with respect to their sequences and EC patterns. The algorithm stops when the rate of change of the clusters falls below a certain cutoff (a stopping criterion) or if no clusters are found. Clusters that are too small (below a preset threshold) are disregarded.

The order of iterations used for clustering our motifs is described below:



Results

5 CLUSTERS

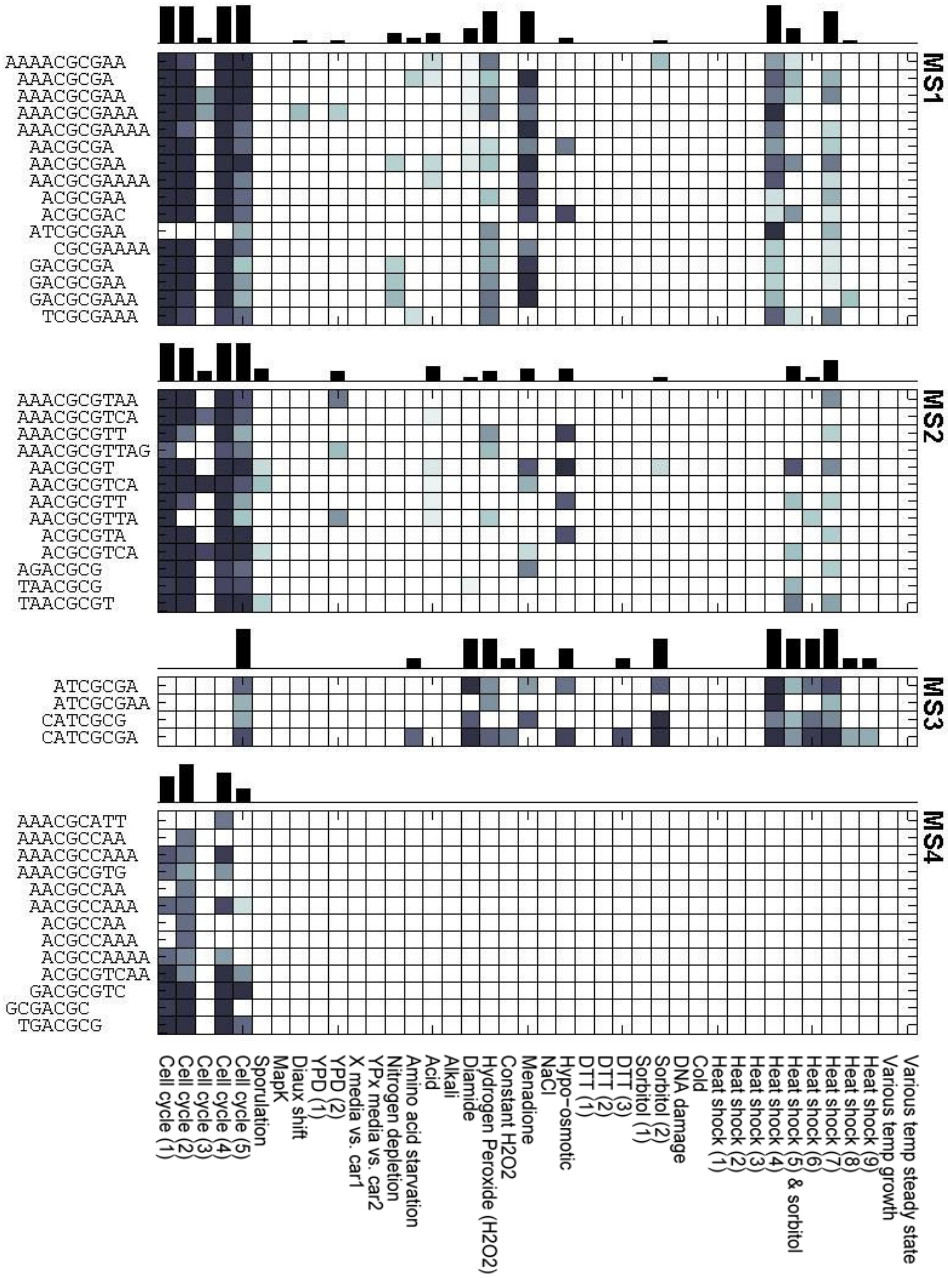
We have clustered a distilled set of motifs, consisting of 694 motifs that have passed the FDR criterion and have also had at least one EC success with a p-value of 0.001 or lower. Our algorithm finds 20 clusters, covering a total of 182 motifs. 14 of our clusters have large overlaps with known clusters.

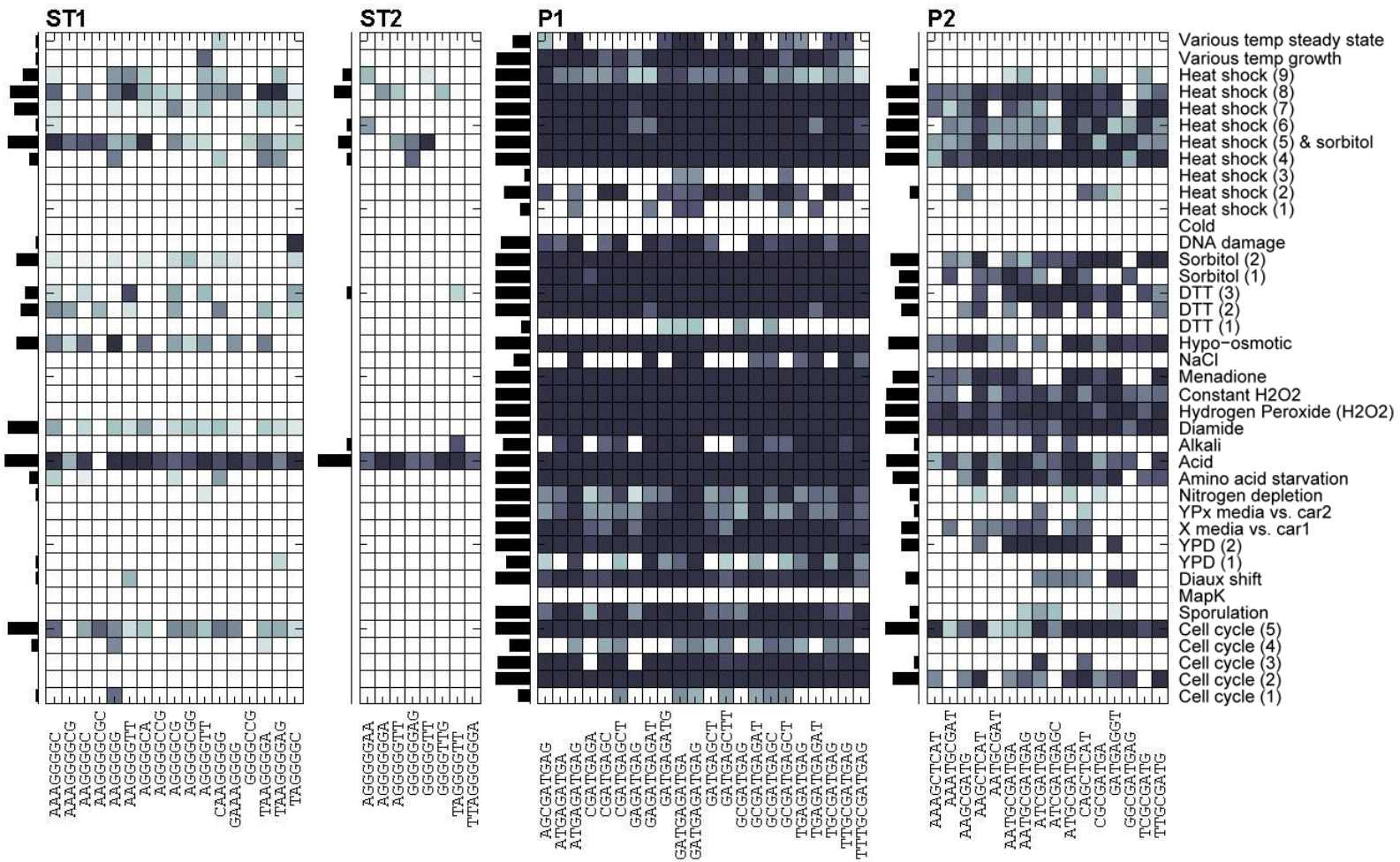
Table 5.1

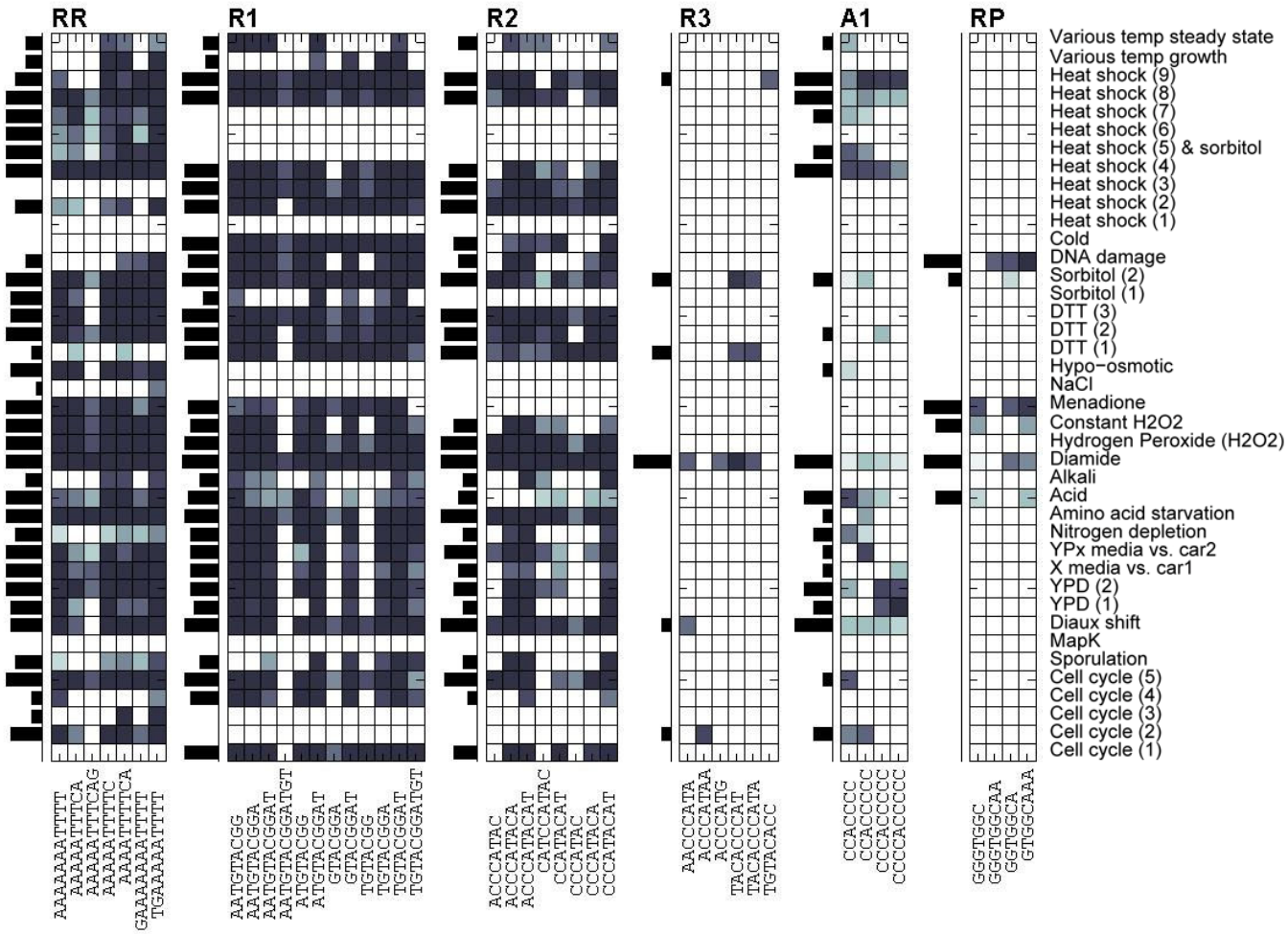
Cluster name	Identified as
MS1	MBF / SBF
MS2	MBF / SBF
MS3	MBF / SBF
MS4	MBF / SBF
ST1	STRE
ST2	STRE
P1	PAC
P2	PAC
RR	RRPE
R1	RAP1
R2	RAP1
R3	RAP1
A1	ADR1 / STRE
RP	RPN4
C15	Unknown
C16	Unknown
C17	Unknown
C18	Unknown
C19	Unknown
C20	Unknown

6 EC PATTERNS OF CLUSTERS

Figure 6.1




ST1
ST2
P1
P2
AAAGGGGC
AAAGGGGCG
AAGGGGC
AAGGGGCGC
AAGGGGG
AAGGGGTT
AGGGGCA
AGGGGCCG
AGGGGCG
AGGGGCGG
AGGGGTT
CAAGGGGG
GAAAGGGG
GGGGCCG
TAAGGGGA
TAAGGGGAG
TAGGGGC
AGGGGGAA
AGGGGGGA
AGGGGGTT
GGGGGGAG
GGGGGTT
GGGGTTG
TAGGGGGTT
TTAGGGGGGA
AGCGATGAG
ATGAGATGA
ATGAGATGAG
CGATGAGA
CGATGAGC
CGATGAGCT
GAGATGAG
GAGATGAGAT
GATGAGATG
GATGAGATGA
GATGAGATGAG
GATGAGCT
GATGAGCTT
GATGAGAG
GCGATGAG
GCGATGAGAT
GCGATGAGC
GCGATGAGCT
TGAGATGAG
TGAGATGAGAT
TGCGATGAG
TTGCGATGAG
TTTGCGATGAG
AAAGCTCAT
AAATGCGAT
AAGCGATG
AAGCTCAT
AATGCGAT
AATGCGATGA
AATGCGATGAG
ATCGATGAG
ATCGATGAGC
ATGCGATGA
CAGCTCAT
CGCGATGA
GATGAGGT
GATGAGAG
GGCGATGAG
TCGCGATG
TTGCGATG
Various temp steady state
Various temp growth
Heat shock (9)
Heat shock (8)
Heat shock (7)
Heat shock (6)
Heat shock (5) & sorbitol
Heat shock (4)
Heat shock (3)
Heat shock (2)
Heat shock (1)
Cold
DNA damage
Sorbitol (2)
Sorbitol (1)
DTT (3)
DTT (2)
DTT (1)
Hypo−osmotic
NaCl
Menadione
Constant H2O2
Hydrogen Peroxide (H2O2)
Diamide
Alkali
Acid
Amino acid starvation
Nitrogen depletion
YPx media vs. car2
X media vs. car1
YPD (2)
YPD (1)
Diaux shift
MapK
Sporulation
Cell cycle (5)
Cell cycle (4)
Cell cycle (3)
Cell cycle (2)
Cell cycle (1)

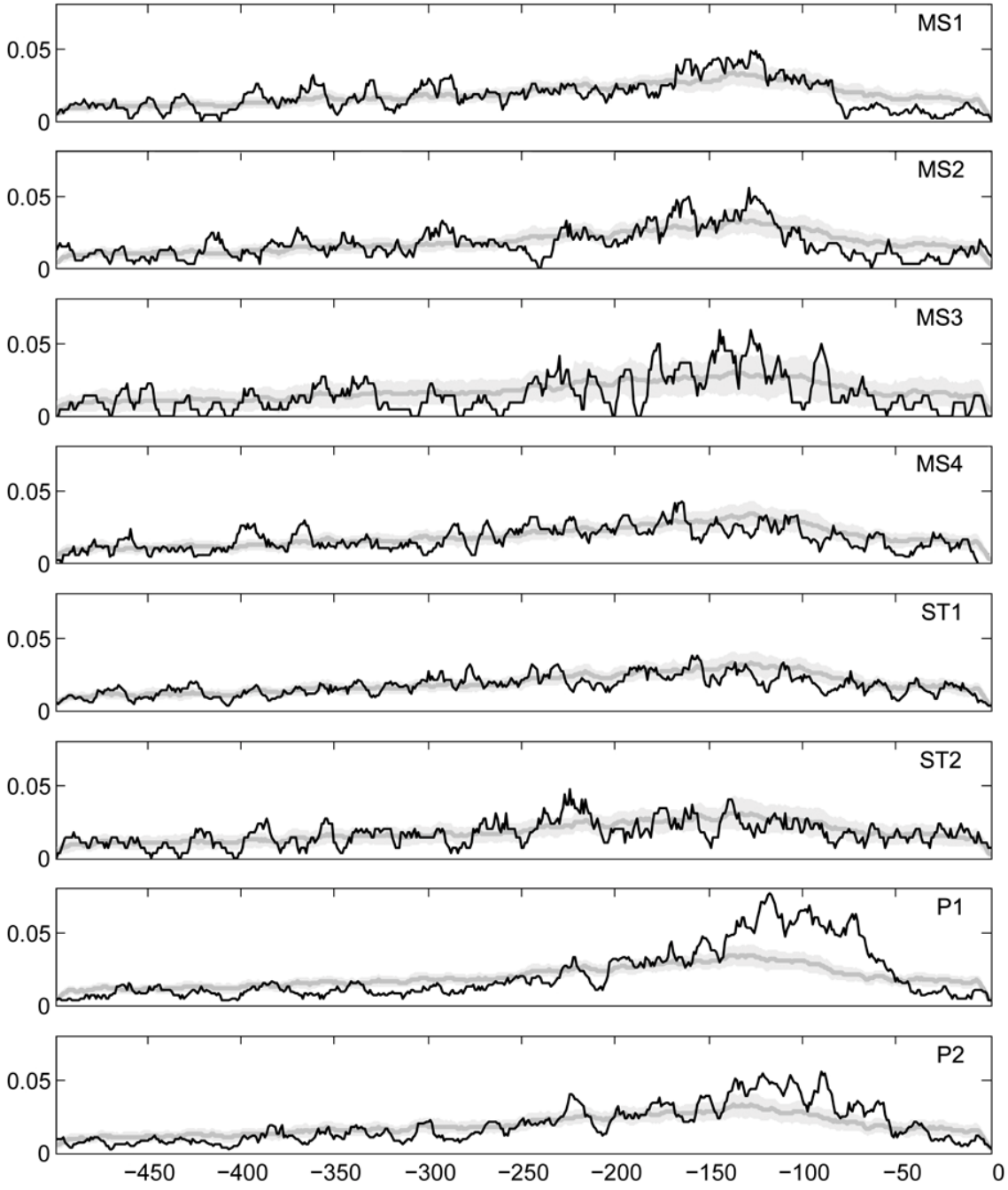

RR
R1
R2
R3
A1
RP
AAAAAAATTTTT
AAAAATTTTCA
AAAAATTTTCAG
AAAAATTTTC
AAAATTTTTCA
GAAAAAAATTTT
TGAAAAATTTTT
AATGTTACGG
AATGTTACGGA
AATGTTACGGAT
AATGTTACGGATGT
ATGTTACGG
ATGTTACGGA
GTTACGGAT
TGTTACGG
TGTTACGGA
TGTTACGGAT
TGTTACGGATGT
ACCCATAC
ACCCATACA
ACCCATACAT
CATCCATAC
CCATACAT
CCCATAC
CCCATACA
CCCATACAT
AACCCATA
ACCCATAA
ACCCATG
TACACCCAT
TACACCCATA
TGTACACC
CCACCCC
CCACCCCC
CCCACCCCC
CCCCACCCCC
GGGTGGC
GGGTGGCAA
GGTGGCA
GTGGCAAA
Various temp steady state
Various temp growth
Heat shock (9)
Heat shock (8)
Heat shock (7)
Heat shock (6)
Heat shock (5) & sorbitol
Heat shock (4)
Heat shock (3)
Heat shock (2)
Heat shock (1)
Cold
DNA damage
Sorbitol (2)
Sorbitol (1)
DTT (3)
DTT (2)
DTT (1)
Hypo−osmotic
NaCl
Menadione
Constant H2O2
Hydrogen Peroxide (H2O2)
Diamide
Alkali
Acid
Amino acid starvation
Nitrogen depletion
YPx media vs. car2
X media vs. car1
YPD (2)
YPD (1)
Diaux shift
MapK
Sporulation
Cell cycle (5)
Cell cycle (4)
Cell cycle (3)
Cell cycle (2)
Cell cycle (1)

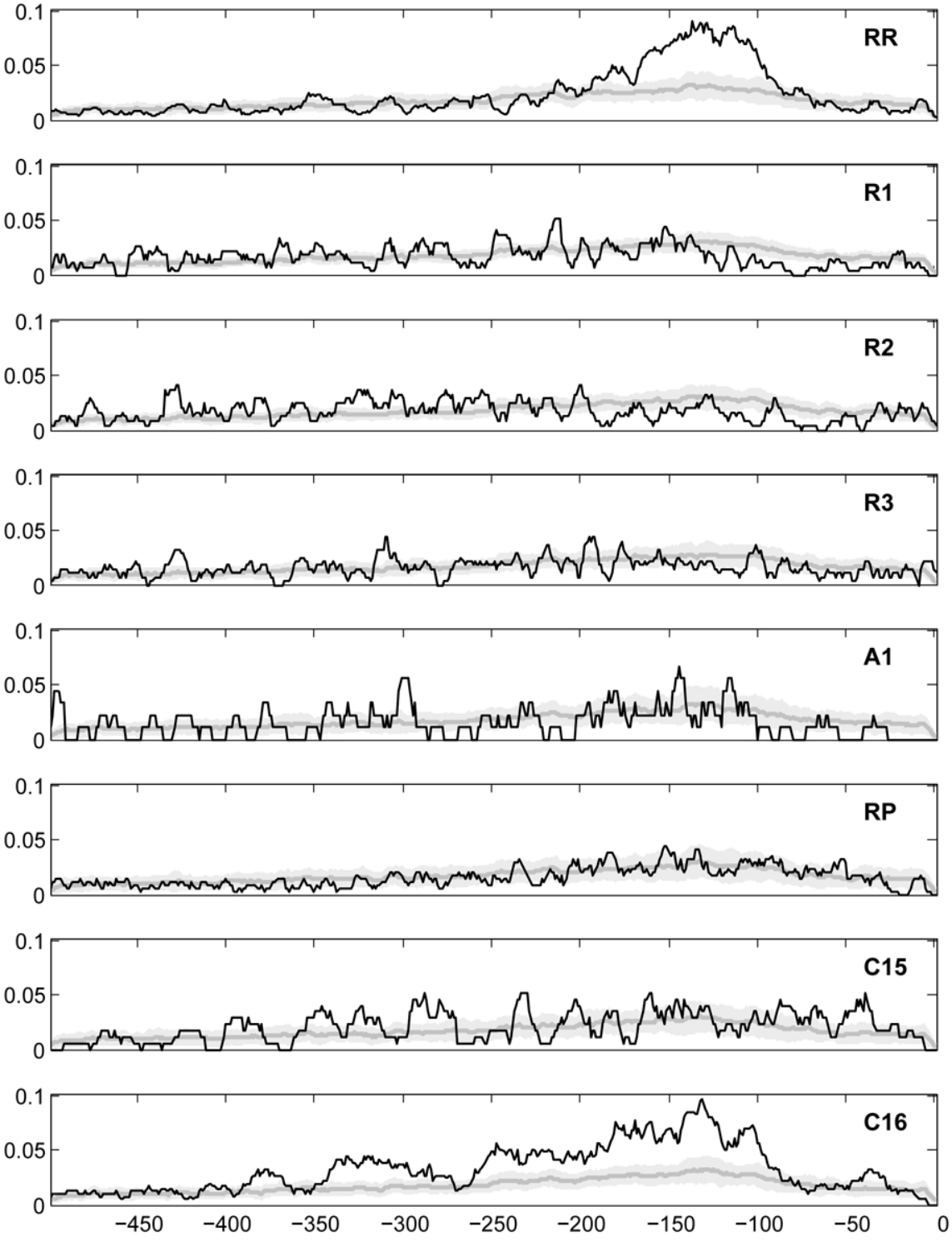
C15
AAAAAAAACCG
AAAAAAGAAAAAGAA
AAAAAGAGAG
ACAAAATCA
C16
AAAAAAAAAAAAAAGAA
AAAAAAAAAAAGAAA
AAAAAAGAAAAGA
AAGGAAAAAAAAA
CAGTGAAAAA
GAAAAAAAAAAAAAA
GAAAAAAAAAAAAAAA
C17
AAAAATGCGA
AAAAATGCGAT
AAAATGCGA
AAAATGCGAT
AAATGCGAT
AATGCGAT
GATGCGAT
TTTGCGAT
C18
AGGGAGAG
AGGGAGC
AGGGAGCA
AGGGAGT
C19
AAAAACGTA
AAAAACGTAT
AAAACAAAAAC
AAAACAAAAACA
AAACAAAAGA
AAACCTAAAA
ACAAAACAAA
C20
AAAAAATCGATGA
AAAAATGCGAT
AAAATGCGAT
ATCGATGA
ATGAGCTC
CGCGATG
GCGATGC
GGCGATGA
TCGCGAT
TGAGCTG
Cell cycle (1)
Cell cycle (2)
Cell cycle (3)
Cell cycle (4)
Cell cycle (5)
Sporulation
MapK
Diaux shift
YPD (1)
YPD (2)
X media vs. car1
YPx media vs. car2
Nitrogen depletion
Amino acid starvation
Acid
Alkali
Diamide
Hydrogen Peroxide (H2O2)
Constant H2O2
Menadione
NaCl
Hypo−osmotic
DTT (1)
DTT (2)
DTT (3)
Sorbitol (1)
Sorbitol (2)
DNA damage
Cold
Heat shock (1)
Heat shock (2)
Heat shock (3)
Heat shock (4)
Heat shock (5) & sorbitol
Heat shock (6)
Heat shock (7)
Heat shock (8)
Heat shock (9)
Various temp growth
Various temp steady state

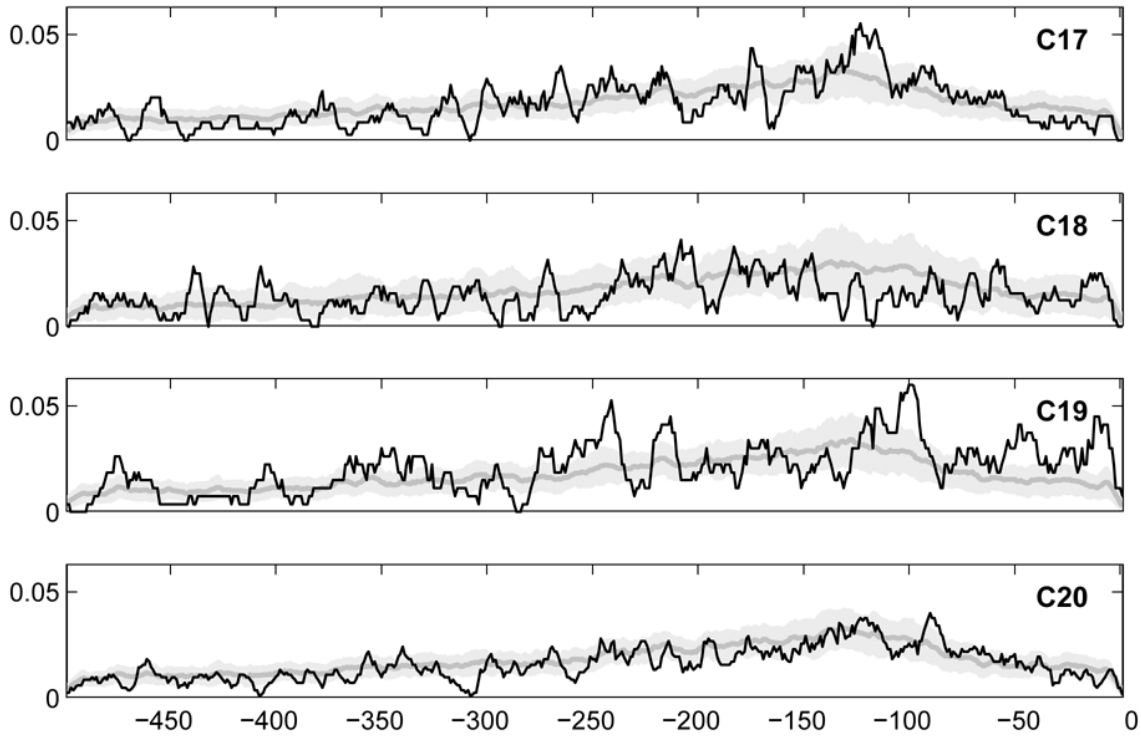
7 LOCALIZATION OF MOTIFS ALONG PROMOTERS

Black lines indicate, for each position upstream to the genes (up to -500bp), the percentage of promoters on which the cluster's sequences have been found. This can be compared to the localization of sampled groups of motifs (of the same sizes as those of the clusters in question) out of the set of 694 motifs. For each cluster, the dark gray line shows the mean motif occurrence per position over 1000 such sampled groups, while the light gray area represents the samples' standard deviation of occurrences per position.

Figure 7.1




0.1
0.05
0
RR
R1
R2
R3
A1
RP
C15
C16
−450
−400
−350
−300
−250
−200
−150
−100
−50
0


0.05
0
C17
0.05
0
C18
0.05
0
C19
0.05
0
C20
−450
−400
−350
−300
−250
−200
−150
−100
−50
0

8 FISHER DISTANCES BETWEEN CLUSTERS

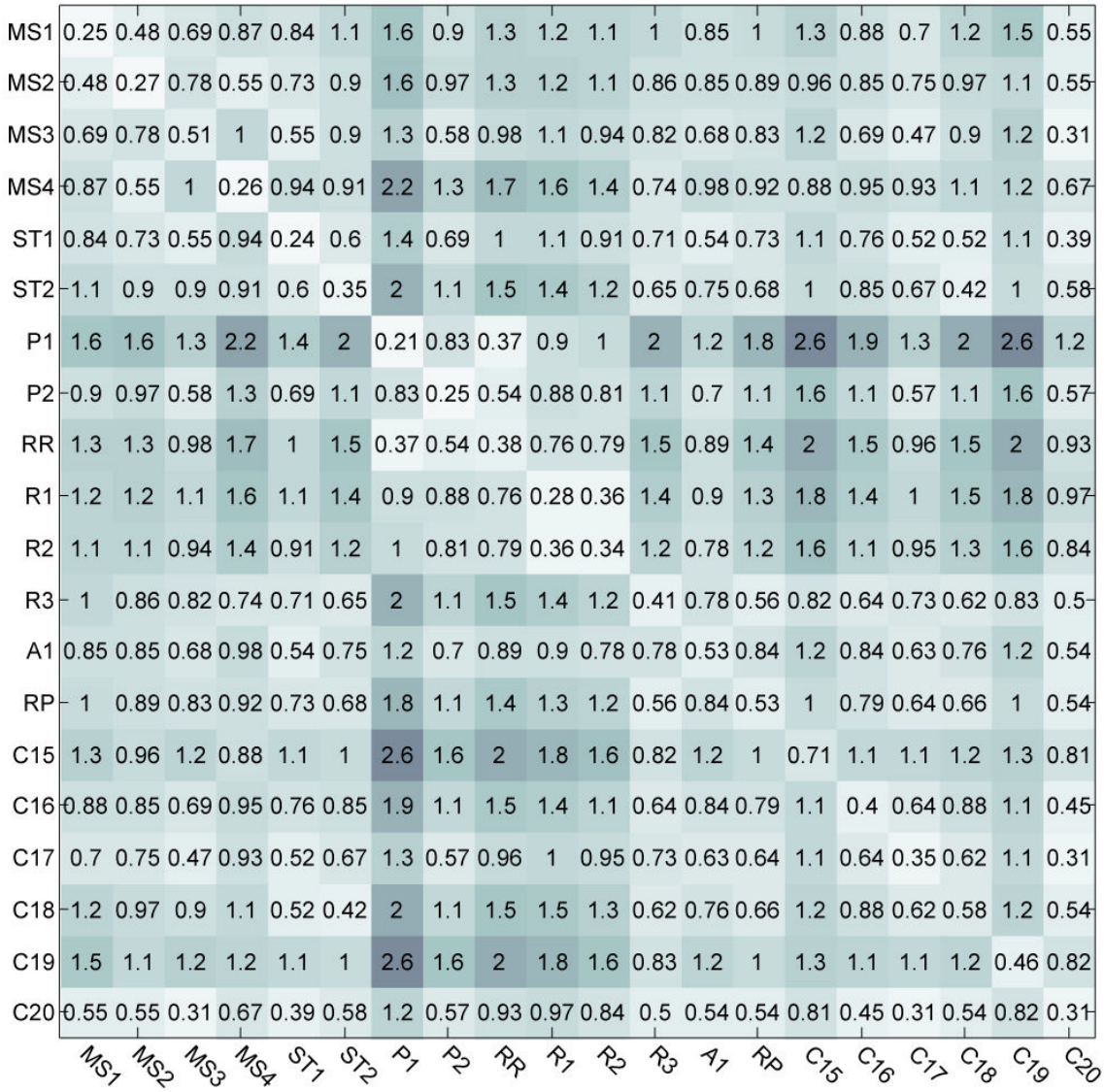


Figure 8.1 Fisher distances between clusters. On the diagonal (where F=0) we have added the mean F-values obtained by randomly dividing each of the clusters into two arbitrary ones (mean over 100 random divisions for each cluster).

9 INTERSECTIONS BETWEEN SETS OF GENES OF COUPLES OF CLUSTERS

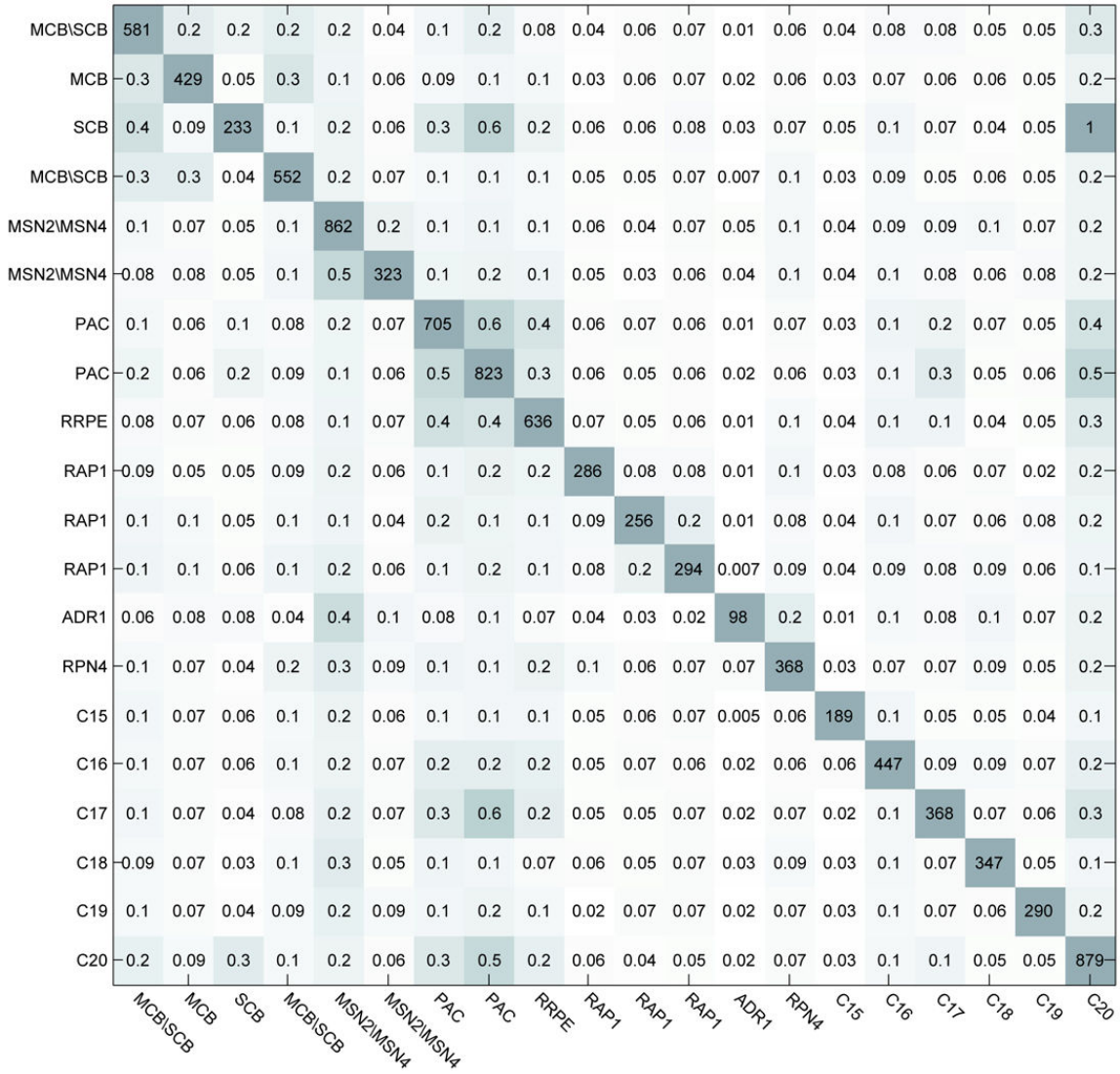


Figure 9.1 Intersections between sets of genes of couples of clusters. The percentage of common genes between cluster i and cluster j, out of cluster i (i in rows, j in columns).

i=j → The number of genes on the promoter of which the clusters motifs are found (the color on the diagonal is set as 100%, in respect to the rest of the matrix).

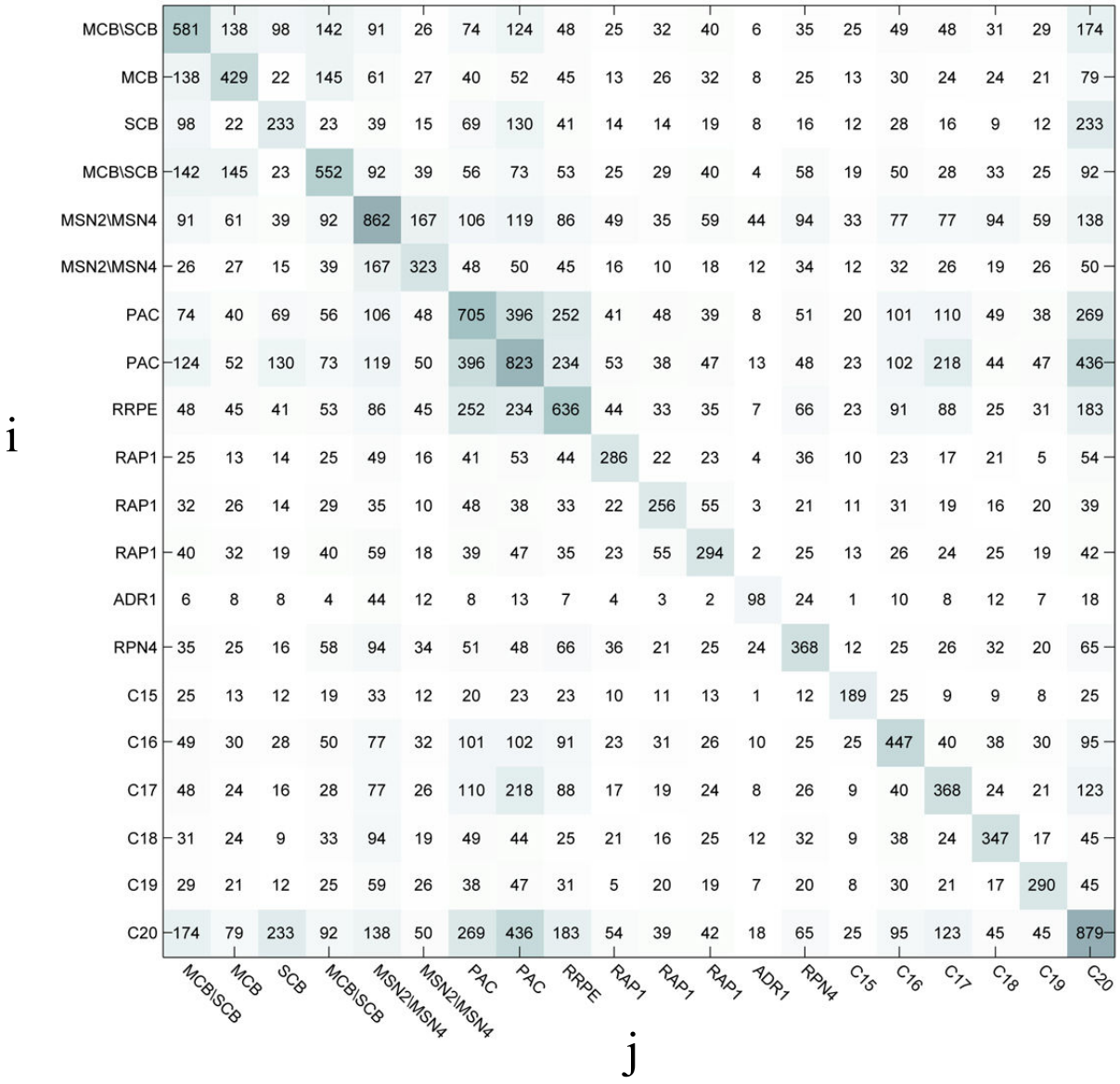


Figure 9.2 Intersections between sets of genes of couples of clusters.

i≠j → The number of common genes between cluster i and cluster j.

i=j → The number of genes on the promoter of which the clusters motifs are found.

10 COMPARING MEX TO ALTERNATIVE APPROACHES

Comparison of applying the MEX algorithm, followed by EC to an exhaustive k-mer enumeration followed by EC

1,873/8,498 (22 %) of MEX's extracted motifs had a significant EC score (passed FDR of 0.1) in at least one of the examined biological conditions. For comparison, in an exhaustive enumeration of all k-mers of length 7-11 residing in yeast promoters (Shalgi et al 2005, Lapidot & Pilpel in prep), we produced 1,510,057 hypotheses, only 8,610(0.6%) of which scored significantly under the same FDR cutoff. In other words we see a striking enhancement in the probability of a k-mer to pass an EC test if it was pre-selected by MEX as a motif that obeys some syntactic rules.

There was an overlap of 849 motifs between the motif sets obtained by the two approaches. 1024 (55%) of MEX's motifs were not discovered by the exhaustive approach (Figure 10.1A). These are mostly weaker motifs, that could not be identified within a very noisy background; MEX provides an enrichment in signal which relaxes the p-value thresholds set by FDR, allowing for weaker motifs to be detected as significant. In addition, MEX extracted sequence motifs of length up to 19 nucleotides. 57 of the unique MEX motifs were longer than 11 bases, and thus were not even scanned by the exhaustive approach (Figure 10.1B). Expanding the exhaustive enumeration to larger sequence lengths is extremely expensive computationally, whereas MEX is easily scalable to longer sequences and to larger genomes. Motifs that were detected by the exhaustive approach, but not by MEX most likely do not obey the inherent position dependencies, selected for by MEX (figure 10.1C). It has been reported that some, but not all functional TFBS display such position dependencies (Tomovic et al. Bioinformatics 2007). The relative success of MEX in identifying high scoring motifs suggests however that there are some syntactic rules that characterize functional binding sites. This thought is intriguing because it implies that we may be able to identify at least some of the TF binding sites based on their sequence context alone.

Coverage of the Harbison PSSM set

To estimate the comprehensiveness of our MEX-extracted motif set, we examined its coverage of the well accepted reference PSSM set published by Harbison et al. We scanned each of the 1,873 motifs of the MEX-extracted set, as well as each of the 8,610 motifs of the exhaustive enumeration set against all 102 Harbison PSSMs. We applied a scoring method that assesses how likely a given k-mer is to be generated by a given PSSM. For each pair of known PSSM and k-mers we summed up the frequencies corresponding to the nucleotides observed in the k-mer, over all PSSM relevant positions. This score was then scaled to the range [0-100] by subtracting the minimal possible score that may be obtained from the PSSM (that is, the score obtained for a k-mer that corresponds to the least frequent position in each column of the PSSM) and dividing by the range of possible scores (obtained after additionally identifying the maximal possible scoring k-mer from that PSSM). Applying a cutoff of 99% identity, 16% of the exhaustive set provide a coverage of 91% of Harbison's PSSMs, and 45% of the MEX set, cover 66% of the Harbison set. Namely MEX is not as comprehensive as the exhaustive set, but it is enriched in signal and contains less false positives. Table 10.1

summarizes the coverage of the Harbison set by the MEX-extracted motifs when applying different cutoffs.

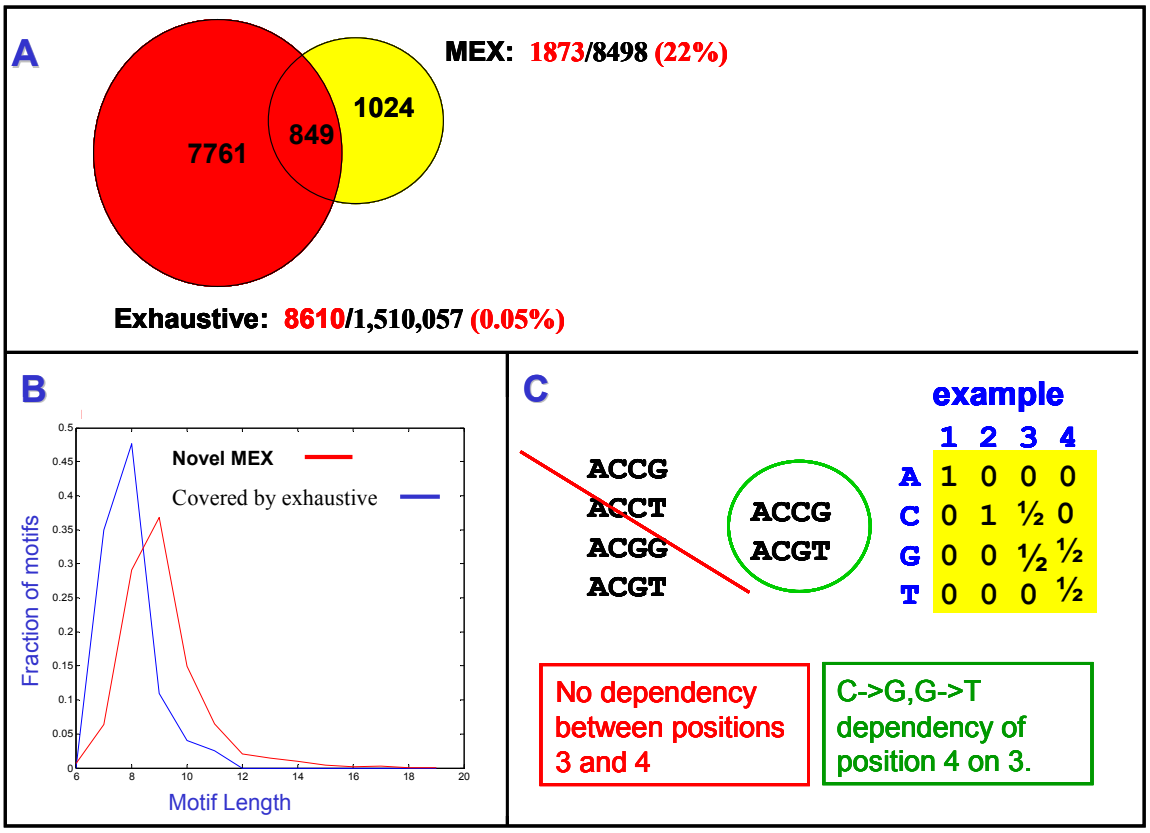


Figure 10.1 Comparison between motif sets obtained by the exhaustive approach versus the syntax based approach (MEX). A. Overlap between motif sets obtained by exhaustive enumeration and by MEX. B. Length distribution of MEX motifs that are covered by the exhaustive search versus novel MEX motifs. MEX has a clear advantage in identifying longer motifs, for which the exhaustive search is computationally too demanding. There are also motifs in the length range of 7-11 which scored significantly when pre-selected by MEX, and not in the exhaustive dictionary. These motifs are weaker and when embedded in a very noisy background (of all possible k-mers), their score is not high enough to pass the threshold set by FDR.C. Motifs in which there are no clear inter-position dependencies will be missed by MEX. MEX learns simple syntax rules from the promoter sequences and searches for motifs that obey these rules.

Comparison score cutoff	Coverage of MEX motif set	Coverage of known	Unique known clusters
99%	837/1873=45%	63/102=61%	51/77=66%
98%	866/1873=46%	68/102=67%	56/77=72%
97%	904/1873=48%	73/102=72%	60/77=78%
95%	996/1873=53%	79/102=77%	65/77=84%

Table 10.1 Coverage of the Harbison PSSM set by the significant motifs extracted by ME. A scoring method was devised to assess how likely a given string is to be generated from a given PSSM. The score is on a scale of 0 to 100. It is computed by summing up the frequencies corresponding to the observed nucleotides over all motif positions, and normalizing this score to a scale of 0-100. The scaling is done by subtracting the minimal possible score and dividing by the range of possible scores. For example for the PSSM [A: 0.0191 0.0191 0.9733 0.9733 0.0120, C:0.9500 0.9500 0.0074 0.0074 0.0074, G: 0.0117 0.0117 0.0074 0.0074 0.0074 T:0.0191 0.0191 0.0120 0.0120 0.9733] the lowest possible score 0.0455 is obtained for the string GG(C/G)(C/G)(C/G), the highest possible score 4.8198 is obtained for the string CCAAT. After scaling GGCCC will score 0%, CCAAT will score 100% and CCATT will score 79.9% ((3.8585-0.0455)/(4.8198-0.0455)). We computed this score for each of the significant (based on the EC score analysis) motifs extracted by MEX versus all 102 Harbison PSSMs. The coverage of Harbison's motif set was assessed for several different score cutoffs. Note that one motif may match more than one Harbison PSSM, because of redundancy in Harbison's dataset. For comparison 16% to 25% of the exhaustive set provide a coverage of 91% to 97% of the Harbison set, depending on the cutoff employed. There are two very long (17 and 18 positions) gapped motif in Harbison's set, for which we have no match, because the dictionary only covers motifs of length 7-11.