Iterative signature algorithm for the analysis of large-scale gene expression data

Sven Bergmann,* Jan Ihmels,† and Naama Barkai‡
Department of Molecular Genetics, Weizmann Institute of Science, Rehovot 76100, Israel
(Received 11 October 2002; published 11 March 2003)

We present an approach for the analysis of genome-wide expression data. Our method is designed to overcome the limitations of traditional techniques, when applied to large-scale data. Rather than alloting each gene to a single cluster, we assign both genes and conditions to context-dependent and potentially overlapping *transcription modules*. We provide a rigorous definition of a transcription module as the object to be retrieved from the expression data. An efficient algorithm, which searches for the modules encoded in the data by iteratively refining sets of genes and conditions until they match this definition, is established. Each iteration involves a linear map, induced by the normalized expression matrix, followed by the application of a threshold function. We argue that our method is in fact a generalization of singular value decomposition, which corresponds to the special case where no threshold is applied. We show analytically that for noisy expression data our approach leads to better classification due to the implementation of the threshold. This result is confirmed by numerical analyses based on *in silico* expression data. We discuss briefly results obtained by applying our algorithm to expression data from the yeast *Saccharomyces cerevisiae*.

DOI: 10.1103/PhysRevE.67.031902 PACS number(s): 87.10.+e

I. INTRODUCTION

DNA microarray experiments monitor the expression levels of thousands of genes simultaneously [1–4]. Using this technology, large sets of genome-wide expression data have been accumulated [5]. For example, the expression levels of the entire yeast genome (comprising ~6200 genes) have been measured for more than 1000 different experimental conditions [6]. A large number of DNA-chip experiments have also been carried out for higher eukaryotes, such as the nematode *C. elegans* and the fruit fly *Drosophila*, as well as for a variety of both normal and malignant human tissues.

While large-scale expression data have the potential to reveal new insights into the transcriptional network that controls gene expression, they also give rise to a major computational challenge: How can one make sense of the massive expression data containing millions of numbers? The classification of the genes and the experimental conditions is an essential first step in reducing the complexity of such data. However, while standard tools, such as clustering algorithms [7–14] (see Refs. [15,16] for reviews) and singular value decomposition (SVD) [17,18], provide interesting results when applied to relatively small data sets, typically containing tens of experimental conditions and at most several hundred genes, these methods are of limited use for the analysis of large data sets. In particular, a well-recognized drawback of commonly used clustering algorithms is the fact that they assign each gene to a single cluster, while in fact genes that participate in several functions should be included in multiple clusters [19–24]. Moreover, both in standard clustering methods and SVD, genes are analyzed based on their expression under *all* experimental conditions. This is problematic, since cellular processes are usually affected only by a small subset of these conditions, such that most conditions do not contribute relevant information but rather increase the level of background noise.

In a recent paper [25] we introduced a new method for the analysis of large-scale gene expression data that was designed to overcome the above-mentioned problems (see Refs. [21–24] for other recent approaches). A central idea of this work was to integrate prior biological information, such as the function or sequence of known genes, into the analysis of the gene expression data. In the present paper we present a complementary method for the analysis of large-scale data that does not require any prior knowledge beyond the expression data. We start by providing a rigorous definition of the type of information we aim to extract from the expression data by introducing the notion of a *transcription module* (TM). A TM contains both a set of genes and a set of experimental conditions. The conditions of the TM induce a coregulated expression of the genes belonging to this TM. That is, the expression profiles of the genes in the TM are the most similar to each other when compared over the conditions of the TM. Conversely, the patterns of gene expression obtained under the conditions of the TM are the most similar to each other when compared only over the genes of the TM. The degree of similarity is determined by a pair of threshold parameters. The gene threshold constrains the gene set, while the condition threshold constrains the condition set. Importantly, distinct transcription modules may share common genes and conditions.

The precise definition of a TM as the object to be retrieved from the expression data allows us to establish an efficient algorithm that searches for the modules encoded in the data. Starting from a set of randomly selected genes (or conditions) one iteratively refines the genes and conditions until they match the definition of a TM. Using a sufficiently large number of initial sets it is possible to determine all the modules corresponding to a particular pair of thresholds. Scanning through a range of thresholds decomposes the data into modules at different resolutions.

*Email address: Sven.Bergmann@weizmann.ac.il
†Email address: Jan@weizmann.ac.il
‡Email address: Naama.Barkai@weizmann.ac.il

This paper is organized as follows. In Sec. II we provide a mathematical definition of a transcription module. In Sec. III we introduce our algorithm that searches for such modules and compare our method with SVD. In Sec. IV we discuss the normalization of the expression data. In Sec. V we present analytical insight into the role of the threshold in our algorithm. We show that for noisy expression data the application of a threshold improves significantly the identification of transcription modules. We provide an estimate for the maximal amount of noise for which a successful identification is still possible. In Sec. VI we compare our method with other standard tools using *in silico* expression data. In Sec. VII we discuss briefly results obtained by applying our algorithm to real expression data from the yeast *Saccharomyces cerevisiae*. We conclude in Sec. VIII.

II. FORMALISM

A. The expression matrix

We consider data from microarray experiments given in terms of a gene expression matrix $\boldsymbol{E}$. The matrix element E^{cg} denotes the log-fold expression change of gene $g \in G \equiv \{1,\ldots,N_G\}$ at the experimental condition $c \in C \equiv \{1,\ldots,N_C\}$, where N_G and N_C refer to the total number of genes and conditions, respectively. The matrix $\boldsymbol{E}$ may be viewed as a collection of N_C row vectors:

$$\boldsymbol{E} = \begin{pmatrix} \boldsymbol{g}_1^T \\ \boldsymbol{g}_2^T \\ \vdots \\ \boldsymbol{g}_{N_C}^T \end{pmatrix}. \tag{1}$$

Each vector $\boldsymbol{g}_c^T = (g_c^{(1)}, g_c^{(2)}, \ldots, g_c^{(N_G)})$ describes the *gene profile* for condition c, containing the expression levels $g_c^{(g)} = E^{cg}$ of all the genes that were monitored under this condition. Alternatively, the expression matrix can be viewed as a collection of N_G column vectors:

$$\boldsymbol{E} = \boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_{N_G}. \tag{2}$$

Here each vector $\boldsymbol{c}_g = (c_g^{(1)}, c_g^{(2)}, \ldots, c_g^{(N_C)})^T$ describes the *condition profile* for gene g, containing the expression levels $c_g^{(c)} = E^{cg}$ of this gene under all the conditions of the data set.

We define two normalized expression matrices (cf. Sec. IV)

$$\boldsymbol{E}_G \equiv \begin{pmatrix} \hat{\boldsymbol{g}}_1^T \\ \hat{\boldsymbol{g}}_2^T \\ \vdots \\ \hat{\boldsymbol{g}}_{N_C}^T \end{pmatrix} \tag{3}$$

and

$$\boldsymbol{E}_C \equiv (\hat{\boldsymbol{c}}_1, \hat{\boldsymbol{c}}_2, \ldots, \hat{\boldsymbol{c}}_{N_G}). \tag{4}$$

The rows of $\boldsymbol{E}_G$ and the columns of $\boldsymbol{E}_C$ are given in terms of the normalized gene and condition vectors,

$$\hat{\boldsymbol{g}}_c \equiv \frac{\boldsymbol{g}_c - \langle \boldsymbol{g}_c \rangle_{g \in G}}{|\boldsymbol{g}_c - \langle \boldsymbol{g}_c \rangle_{g \in G}|} \quad \text{and} \quad \hat{\boldsymbol{c}}_g \equiv \frac{\boldsymbol{c}_g - \langle \boldsymbol{c}_g \rangle_{c \in C}}{|\boldsymbol{c}_g - \langle \boldsymbol{c}_g \rangle_{c \in C}|}, \tag{5}$$

respectively. These vectors have zero mean ($\langle \hat{\boldsymbol{g}}_c \rangle_{g \in G} = \langle \hat{\boldsymbol{c}}_g \rangle_{c \in C} = 0$) and unit length ($|\hat{\boldsymbol{g}}_c| = |\hat{\boldsymbol{c}}_g| = 1$). This normalization implies that $\Sigma_g \hat{E}_G^{cg} = 0$, $\Sigma_g (\hat{E}_G^{cg})^2 = 1$ for each condition c and $\Sigma_c \hat{E}_C^{cg} = 0$, $\Sigma_c (\hat{E}_C^{cg})^2 = 1$ for each gene g. Centering and rescaling the rows in $\boldsymbol{E}_G$ allows for a meaningful comparison between any two conditions c and c' through their associated gene profiles $\hat{\boldsymbol{g}}_c$ and $\hat{\boldsymbol{g}}_{c'}$. Similarly, centering and rescaling the columns in $\boldsymbol{E}_C$ allows for the comparison of any two genes g and g' through their associated condition profiles $\hat{\boldsymbol{c}}_g$ and $\hat{\boldsymbol{c}}_{g'}$. Note that the normalized matrices $\boldsymbol{E}_G$ and $\boldsymbol{E}_C$ in general are not equal.

B. Transcription modules

Our goal is to find sets of coregulated genes $G_m \subset G$, together with the relevant experimental conditions $C_m \subset C$ that induce their coregulation. We refer to such a combined set, $M_m = \{G_m, C_m\}$, as a TM. Here the index m ranges between one and the number of transcription modules, N_M. Biologically a TM may be associated with a particular cellular function. Ideally each TM would correspond to a transcription factor that regulates the genes in G_m and that is activated under the conditions in C_m. Of course, a one-to-one correspondence between transcription modules and transcription factors is an over-simplification, but it can still provide useful insight into the nature of the expression data. First, the total number of transcription factors, N_{TF}, is much smaller than the number of genes: $N_{TF} \ll N_G$. Thus we expect also the number of transcription modules, and therefore the effective dimensionality of the expression matrix to be relatively small: $N_M \ll N_G$. Second, the number of genes activated by a single transcription factor, $N_G^{(m)}$, is known to be limited: $N_G^{(m)} \ll N_G$. Third, different transcription factors can regulate the same gene and can be activated under the same experimental conditions. Hence distinct modules may share common genes and conditions.

Mathematically, a TM can be defined as follows:

$$\exists (T_C, T_G): \begin{cases} C_m(G_m) = \{c \in C : \langle E_G^{cg} \rangle_{g \in G_m} > T_C\}, \\ G_m(C_m) = \{g \in G : \langle E_C^{cg} \rangle_{c \in C_m} > T_G\}, \end{cases} \tag{6}$$

where T_C and T_G are two threshold parameters. The above definition states that for each condition c in the TM the average expression level of the genes in the TM, $\langle E_G^{cg} \rangle_{g \in G_m}$, is above a certain threshold T_C. Conversely, for each gene g in the TM the average expression level over the conditions of the TM, $\langle E_C^{cg} \rangle_{c \in C_m}$, is also above some threshold T_G. This reciprocal dependence between the genes and the conditions associated with a TM implies that, considering only the genes of the module, the conditions of the module are exactly those for which the coexpression is the most stringent.

Similarly, considering only the conditions of the module, the genes of the module are the most tightly coregulated. Note that our definition of a TM is symmetric with respect to genes and conditions, such that no preference is given to either of them. In particular, we use the expression matrix $\boldsymbol{E}_G$ (normalized with respect to genes) in order to specify the conditions of the module (C_m), given the genes of the module (G_m). Similarly we use $\boldsymbol{E}_C$ (normalized with respect to conditions) to specify the genes in G_m, given the conditions in C_m.

We would like to reformulate and somewhat generalize the definition of a TM in Eq. (6) by introducing vector notation. To this end we represent the genes and the conditions of a TM by a pair of a gene vector $\boldsymbol{g}_m=(g_m^{(1)},g_m^{(2)},\ldots,g_m^{(N_G)})^T$ and a condition vector $\boldsymbol{c}_m=(c_m^{(1)},c_m^{(2)},\ldots,c_m^{(N_C)})^T$. A nonzero component $g_m^{(g)}$ ($c_m^{(c)}$) implies that the gene g (condition c) is associated with the module m. Consider the linear transformations

$$\boldsymbol{c}_m^{proj}\equiv\boldsymbol{E}_G\cdot\boldsymbol{g}_m=\begin{pmatrix}\hat{\boldsymbol{g}}_1^T\boldsymbol{g}_m\\ \hat{\boldsymbol{g}}_2^T\boldsymbol{g}_m\\ \vdots\\ \hat{\boldsymbol{g}}_{N_C}^T\boldsymbol{g}_m\end{pmatrix}\quad\text{and}$$

$$\boldsymbol{g}_m^{proj}\equiv\boldsymbol{E}_C^T\cdot\boldsymbol{c}_m=\begin{pmatrix}\hat{\boldsymbol{c}}_1^T\boldsymbol{c}_m\\ \hat{\boldsymbol{c}}_2^T\boldsymbol{c}_m\\ \vdots\\ \hat{\boldsymbol{c}}_{N_G}^T\boldsymbol{c}_m\end{pmatrix}. \tag{7}$$

The resulting vectors contain the projections of the vectors $\boldsymbol{g}_m$ and $\boldsymbol{c}_m$, which specify the TM, onto the set of the (normalized) gene profiles $\{\hat{\boldsymbol{g}}_c\}$ and condition profiles $\{\hat{\boldsymbol{c}}_g\}$, defined in Eq. (5), that describe the expression data. For a binary vector $\boldsymbol{g}_m$ the components of $\boldsymbol{c}_m^{proj}$ are just the expression levels summed over the genes of the TM for each condition in the data set. Likewise for a binary vector $\boldsymbol{c}_m$ the components of $\boldsymbol{g}_m^{proj}$ are the expression levels summed over the conditions of the module for each gene.

The consistency requirement in Eq. (6) can then be written as

$$\exists(t_C,t_G):\begin{cases}\boldsymbol{c}_m=f_{t_C}(\boldsymbol{c}_m^{proj}),\\ \boldsymbol{g}_m=f_{t_G}(\boldsymbol{g}_m^{proj}),\end{cases} \tag{8}$$

where t_C and t_G are the condition threshold and the gene threshold, related to T_C and T_G, respectively. The threshold function

$$f_t(\boldsymbol{x})\equiv\begin{pmatrix}w(x_1)\Theta(\tilde{x}_1-t)\\ \vdots\\ w(x_{N_x})\Theta(\tilde{x}_{N_x}-t)\end{pmatrix} \tag{9}$$

acts separately on each of the N_x components x_i of the vector $\boldsymbol{x}$ and yields the products of a weight function $w(x)$ and a step function $\Theta(x)$ as output. The arguments of the step function, $\tilde{x}_i=[x_i-\mu(\boldsymbol{x})]/\sigma(\boldsymbol{x})$, have been centered and rescaled. We use the mean as center, $\mu(\boldsymbol{x})=\langle\boldsymbol{x}\rangle$, and the expected or measured standard deviation, $\sigma(\boldsymbol{x})=\sqrt{\Sigma_i^{N_x}(x_i-\langle\boldsymbol{x}\rangle)^2/N_x}$, as scale factor. The step function sets to zero all elements of the vector $\boldsymbol{x}$ that do not exceed $\mu(\boldsymbol{x})$ by at least $t\sigma(\boldsymbol{x})$. [Down-regulation can be captured by replacing $\tilde{x}_i\rightarrow|\tilde{x}_i|$ in Eq. (9).] Using $w(x)=1$ as weight function all the significant elements are set to unity. This binary formulation corresponds to the consistency requirement in Eq. (6). [To capture down-regulation one uses $\mathrm{sgn}(x)$ as weight function.] It is straightforward to extend our formalism using different weight functions. In this case the entries of the gene vector and condition vector become continuous, and their value determines the significance of a particular gene or condition, respectively. As we shall see, a particularly relevant choice is $w(x)=x$ in which case $f_t(\boldsymbol{x})$ is semilinear.

The compact definition of a TM in Eq. (8) can be understood as follows: Applying the threshold function f_{t_C} to $\boldsymbol{c}_m^{proj}$ results in a nonzero component $c_m^{(c)}$ of the module's condition vector $\boldsymbol{c}_m$, if the corresponding gene profile $\hat{\boldsymbol{g}}_c$ is sufficiently aligned with the gene vector $\boldsymbol{g}_m$ of the module. Biologically this means that a significant fraction of the genes in the module are coregulated under condition c. Similarly, the application of f_{t_G} to $\boldsymbol{g}_m^{proj}$ results in a nonzero component $g_m^{(g)}$ in the module's gene vector $\boldsymbol{g}_m$, if the corresponding condition profile $\hat{\boldsymbol{c}}_g$ is sufficiently aligned with the condition-vector $\boldsymbol{c}_m$ of the module. Biologically this implies that a significant fraction of the conditions in the module induce a coregulated expression of gene g.

It is important to note that the content of a particular module $M_m=\{G_m,C_m\}$ depends on the pair of thresholds (t_G,t_C). In many cases for slightly larger thresholds there exists a related module M_m^{up}, such that $M_m^{up}\subset M_m$. Similarly, for somewhat smaller thresholds there usually exists a module M_m^{down}, such that $M_m\subset M_m^{down}$. Thus there are nested sets of modules, $M_m^{top}\subset\cdots\subset M_m^{bottom}$ that persist over a finite range of the thresholds. This hierarchical structure resembles the tree structures obtained from clustering. However, in our case distinct branches may share common genes or conditions.

III. THE ITERATIVE SIGNATURE ALGORITHM

The rigorous definition of a transcription module, in principle, allows us to determine the modules encoded in the expression matrix by testing all possible sets $\{G_m,C_m\}$ for their compliance with Eq. (8). However, since the number of such sets scales exponentially with the number of genes and conditions, such an approach is completely infeasible computationally. We therefore suggest a different approach. Our principle idea is to search for solutions of the consistency equation in Eq. (8) through the map defined by

$$\boldsymbol{c}^{(n+1)} = f_{t_C}(\boldsymbol{E}_G \cdot \boldsymbol{g}^{(n)}), \tag{10}$$

$$\boldsymbol{g}^{(n+1)} = f_{t_G}(\boldsymbol{E}_C^T \cdot \boldsymbol{c}^{(n+1)}). \tag{11}$$

The first equation assigns a condition vector $\boldsymbol{c}^{(n+1)}$ to a given gene vector $\boldsymbol{g}^{(n)}$. We refer to the component $c_c^{(n+1)}$ of this vector as a *condition score*. This score is nonzero only if the corresponding gene profile $\hat{\boldsymbol{g}}_c$, defined in Eq. (5), is sufficiently aligned with the gene vector $\boldsymbol{g}_m^{(n)}$. In the subsequent step in Eq. (11) the component (or *gene score*) $g_g^{(n+1)}$ of the gene vector $\boldsymbol{g}_m^{(n+1)}$ is assigned a nonzero value only if the corresponding condition profile $\hat{\boldsymbol{c}}_g$ is sufficiently aligned with the condition vector $\boldsymbol{c}_m^{(n+1)}$. In a recent work [25] we have applied the map in Eqs. (10) and (11) to a variety of biologically motivated input sets $\{\boldsymbol{g}_i^{(0)}\}$ assembled according to prior knowledge of the regulatory sequence or function of the genes. Sets of coregulated genes and coregulating conditions were constructed from recurrent realizations of the output sets defined by $\boldsymbol{g}^{(1)}$ and $\boldsymbol{c}^{(1)}$. In this work we pursue a different strategy, namely, we apply the maps in Eqs. (10) and (11) iteratively by reusing the gene vector $\boldsymbol{g}^{(1)}$ as input for Eqs. (10) and (11) in order to obtain new output sets defined by $\boldsymbol{c}^{(2)}$ and $\boldsymbol{g}^{(2)}$. Repeating this procedure we obtain $\{\boldsymbol{g}^{(3)},\boldsymbol{c}^{(3)}\}$ from $\boldsymbol{g}^{(2)}$ and so on. In general, the series $\{\boldsymbol{g}^{(0)},\boldsymbol{g}^{(1)},\boldsymbol{g}^{(2)},\boldsymbol{g}^{(3)},\ldots\}$ rapidly converges and we can define a ''fixed point'' gene vector $\boldsymbol{g}^{(n^*)}$ that satisfies

$$\frac{|\boldsymbol{g}^{(*)} - \boldsymbol{g}^{(n)}|}{|\boldsymbol{g}^{(*)} + \boldsymbol{g}^{(n)}|} < \varepsilon \tag{12}$$

for all n above a certain number of iterations. The parameter ε determines the accuracy of the fixed point. $\boldsymbol{g}^{(*)}$ depends both on the ''seed'' $\boldsymbol{g}^{(0)}$ and the thresholds t_G and t_C, which are fixed parameters. Together with the associated condition vector $\boldsymbol{c}^{(*)}$ it defines a TM, since $(\boldsymbol{g}^{(*)},\boldsymbol{c}^{(*)})$ by definition solve Eq. (8). We call this procedure the *iterative signature algorithm* (ISA).

Although the set of possible input seeds is huge, usually there exist only a rather limited number of fixed points for a given set of thresholds (t_G,t_C). Therefore, in general the ISA is applied as follows: (1) generate a (sufficiently large) sample of input seeds $\{\boldsymbol{g}_m^{(0)}\}$, (2) find the fixed points $(\boldsymbol{g}_m^{(*)},\boldsymbol{c}_m^{(*)})$ corresponding to each seed through iterations, and (3) collect the distinct fixed points in order to decompose the expression data into modules. The structure of this decomposition depends on the choice of thresholds (t_G,t_C). Scanning over different values for (t_G,t_C) reveals the modular structure at different resolutions: Lower thresholds yield larger units whose coregulation is relatively loose, while higher thresholds lead to smaller, tightly coregulated modules. Each fixed point $(\boldsymbol{g}_m^{(*)},\boldsymbol{c}_m^{(*)})$ has its ''basin of attraction,'' i.e., the set of seeds that converge to it under the iterative scheme in Eqs. (10) and (11). The size of this set is a measure of the ''convergence radius,'' while the average number of iterations, which is needed until Eq. (12) is satisfied, characterizes the ''depth'' of this basin.

The computation time of any algorithm, designed for the analysis of large-scale expression data, is of crucial importance. For algorithms that require the full correlation matrices (such as clustering or SVD), already the computation of these two matrices can be very intensive, since its computation time scales like $t_{comp}^{corr} \propto N_G^2 N_C + N_C^2 N_G$. However, the ISA is not based on this kind of information. Rather than squaring the expression matrix, only multiplications of the expression matrix with *sparse* matrices (of size $N_G \times N_I$ or $N_C \times N_I$), where N_I is the number of input sets, have to be performed. Due to the sparseness, the computation time of the ISA goes like $t_{comp}^{ISA} \propto N_{iter} N_I (N_C \tilde{N}_G + N_G \tilde{N}_C)$, where $\tilde{N}_G$ and $\tilde{N}_C$ refer to the average number of genes and condition, respectively, whose scores are above the threshold, and N_{iter} is the number of iterations until convergence. Thus the computation time of the ISA scales linearly with N_G and N_C. In general, only very few iterations N_{iter} are needed to find the fixed points. A large number of input sets N_I increases the chances to find the fixed points with a small convergence radius. However, for practical purposes it is useful to accumulate progressively sets of fixed points by running the ISA repeatedly with a moderate value for N_I, thus increasing gradually the accuracy of the fixed point decomposition. Importantly, $\tilde{N}_G$ and $\tilde{N}_C$ are much smaller than N_G and N_C as long as the respective thresholds are high enough. Finally, we note that t_{comp}^{ISA} can be further improved by choosing the input seeds not completely at random, but using the information of previous runs (e.g., starting from the sets obtained at different thresholds).

Comparison with singular value decomposition

For $w(x)=x$, in the absence of thresholds and neglecting the two different normalizations of the expression data, the iterative scheme reads

$$\hat{\boldsymbol{c}}^{(n)} = \frac{\boldsymbol{E} \cdot \hat{\boldsymbol{g}}^{(n-1)}}{|\boldsymbol{E} \cdot \hat{\boldsymbol{g}}^{(n-1)}|}, \tag{13}$$

$$\hat{\boldsymbol{g}}^{(n)} = \frac{\boldsymbol{E}^T \cdot \hat{\boldsymbol{c}}^{(n)}}{|\boldsymbol{E}^T \cdot \hat{\boldsymbol{c}}^{(n)}|}. \tag{14}$$

The fixed points of the above equations correspond to the pairs of vectors $(\hat{\boldsymbol{g}}_m,\hat{\boldsymbol{c}}_m)$, where $\hat{\boldsymbol{g}}_m = \boldsymbol{g}_m/|\boldsymbol{g}_m|$ and $\hat{\boldsymbol{c}}_m = \boldsymbol{c}_m/|\boldsymbol{c}_m|$ are the normalized eigenvectors of $\boldsymbol{E}^T \cdot \boldsymbol{E}$ and $\boldsymbol{E} \cdot \boldsymbol{E}^T$, respectively. Both eigenvectors are associated with the common eigenvalue $\mu_m^2 = |\boldsymbol{E} \cdot \hat{\boldsymbol{g}}_m|^2 = |\boldsymbol{E}^T \cdot \hat{\boldsymbol{c}}_m|^2$. It is interesting to note that a SVD of the expression matrix yields exactly those eigenvectors and eigenvalues [28,29] (see section 1 of the Appendix for a brief review of SVD). This decomposition is usually performed in a sequential manner. In this case one determines first the pair $(\hat{\boldsymbol{g}}_1,\hat{\boldsymbol{c}}_1)$ associated with the largest eigenvalue μ_1^2. In fact, this pair emerges as a fixed point of the above equations for any seed $\boldsymbol{g}^{(0)}$ that is not perpendicular to $\hat{\boldsymbol{g}}_1$. It can be shown that the matrix

$$\boldsymbol{E}_1 = \mu_1 \hat{\boldsymbol{c}}_1 \cdot \hat{\boldsymbol{g}}_1^T \tag{15}$$

provides the best rank-1 approximation to $\boldsymbol{E} = \boldsymbol{E}_1 + \boldsymbol{R}_1$, where $\boldsymbol{R}_1$ denotes the residual term. A subsequent diagonalization of $\boldsymbol{R}_1$ yields the (orthogonal) pair $(\hat{\boldsymbol{g}}_2, \hat{\boldsymbol{c}}_2)$ associated with the second largest eigenvalue μ_2. Continuing this procedure eventually decomposes the expression matrix into a sum

$$\boldsymbol{E} = \sum_m^{N_M} \boldsymbol{E}_m + \boldsymbol{R}_{N_M} \tag{16}$$

of the rank-1 matrices $\boldsymbol{E}_m = \mu_m \hat{\boldsymbol{c}}_m \hat{\boldsymbol{g}}_m^T$ with $\mu_m = |\boldsymbol{c}_m||\boldsymbol{g}_m|$. These matrices can be viewed as a special kind of transcription modules.

One of the advantages of SVD is that the significance of each modular component $\boldsymbol{E}_m$ can be determined simply according to the magnitude of the associated eigenvalue. The components associated with small eigenvalues are likely to reveal no real information and to contain only noise. Thus the spectrum of eigenvalues can give some indication of the dimensionality of the data: The existence of N_M eigenvalues that are significantly larger than the remaining eigenvalues suggests that there are N_M dominant components. Similar to SVD, the lengths of the fixed point vectors of the ISA provide a measure of the relative importance of the associated TM. Specifically, $|\boldsymbol{g}_m^{(*)}|^2 = \Sigma_{g \in G_m} (g_g^{(*)})^2$ reflects the size of the gene set and [for $w(x) = x$] the strength of its coregulation, while $|\boldsymbol{c}_m^{(*)}|^2 = \Sigma_{c \in C_m} (c_c^{(*)})^2$ reflects the size of the condition set and the strength of the coregulation induced by this set.

While the similarity between the ISA and SVD is instructive, there are several important differences.

Applying the threshold functions in Eqs. (10) and (11) yields a different spectrum of fixed points. Sets of genes that are fixed points of the iterative scheme for a particular choice of the threshold, in general do not correspond to the eigenvectors of the expression matrix.

The thresholds affect the stability of the fixed points. While the iterations in Eqs. (13) and (14) have only a single *stable* fixed point $(\hat{\boldsymbol{g}}_1, \hat{\boldsymbol{c}}_1)$, the ISA in Eqs. (10) and (11) usually possesses several stable fixed points. This is essentially because the thresholds induce an ''effective orthogonality'' by setting the small scalar products in Eq. (7) to zero. Consequently input sets that are almost (but not exactly) orthogonal to the strongest fixed point, do not flow towards this point under the iterations, but converge to a different fixed point.

SVD is very sensitive to the (unavoidable) noise in the expression data. This noise induces mixing between modules that would be orthogonal to each other in the absence of noise. In the ISA the threshold function provides an efficient way to deal with such noise. Excluding the bulk of the genes and conditions from the expression data at each step of the iterative procedure allows to pick up coregulated units that would otherwise be masked by the noise.

For SVD distinct eigenvectors $\hat{\boldsymbol{g}}_m$ and $\hat{\boldsymbol{g}}_{m'}$ as well as $\hat{\boldsymbol{c}}_m$ and $\hat{\boldsymbol{c}}_{m'}$ are orthogonal to each other, since they diagonalize a symmetric matrix. The constraint of orthogonality is not present in the ISA.

SVD only reveals one single decomposition of the expression matrix into modules. As for the ISA, changing the values of the thresholds allows to analyze the modular structure recorded in the expression matrix at different resolutions.

For SVD the expression data have to be normalized either according to genes or conditions. The choice of data normalization, in general, follows from the interpretation of the data. Demanding maximal variance among the principal components, one is led to center the data either as in $\boldsymbol{E}_G$ or $\boldsymbol{E}_C$ (see section 1 of the Appendix on SVD for details). Thus the symmetry between the genes and the conditions is explicitly broken when committing to either $\boldsymbol{E}_C$ or $\boldsymbol{E}_G$. In contrast, the ISA avoids this bias by alternating between the two possible normalizations at each step of the iterative procedure in Eqs. (10) and (11).

A connection between SVD and other algorithms for multivariate analysis was pointed out in Ref. [24]. ''Semidiscrete decomposition'' [26], ''non-negative matrix factorization,'' [27] and *k*-means clustering can be viewed as SVD with additional constraints on the eigenvalues and eigenvectors. Other algorithms, like the ''Plaid model'' [24], extend SVD by introducing additional parameters that allow for an improved decomposition of the expression data into potentially overlapping sets of genes and conditions. In ''gene shaving'' [23] a cluster is formed by removing iteratively genes whose expression profiles are the least similar to the principal component. The optimal cluster is determined *a posteriori*, by demanding both high-variance clusters and high coherence between the genes in the cluster. Subsequently, a new cluster is obtained from the data orthogonal to the first cluster, and so on. Our approach differs from these algorithms in two important aspects. First, our method relies on a formal definition of a module, which is independent of the features of the other modules. In contrast, in all other approaches, individual clusters are defined only with respect to the other clusters in the data. Second, our algorithm avoids the accumulation of noise by applying the threshold function already in each step of the iterations in Eqs. (10) and (11) that would otherwise lead to the principal component of the expression matrix.

IV. THE PROPER DATA NORMALIZATION

Given the ''raw'' expression data contained in $\boldsymbol{E}$ it is difficult to compare two experiments ($\boldsymbol{g}_c$ and $\boldsymbol{g}_{c'}$) or two genes ($\boldsymbol{c}_g$ and $\boldsymbol{c}_{g'}$). This is because different experiments may affect the expression levels at a different scale. For example one condition may change the expression of many genes by a very large factor ($\gg 1$) while another condition affects mainly the same genes, but shifts their expression level by a much smaller amount. Although the two conditions are related, this relation is not explicit in the expression data. Moreover, recording the expression levels with different microarray techniques as well as variations in the sample prepa-

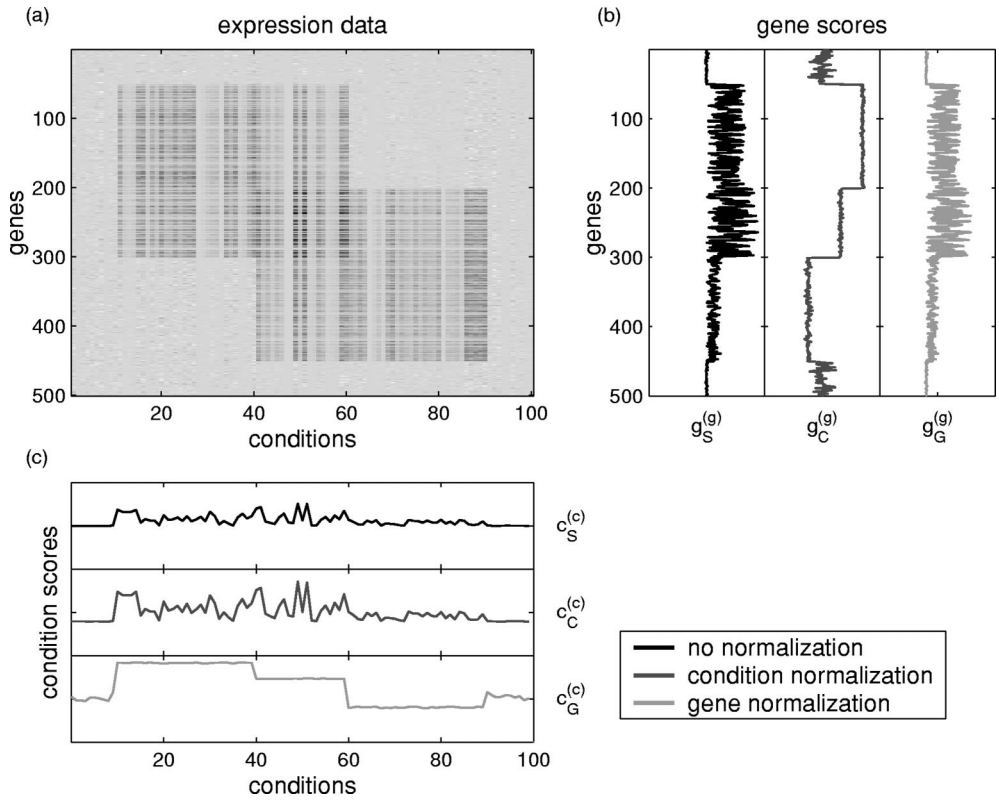

FIG. 1. How to properly normalize the expression matrix. (a) An *in silico* expression matrix, corresponding to two overlapping modules of equal size and strength, was generated according to the model described in the text. The elements of the original expression matrix E^{cg}, were scaled to $E_S^{cg} \equiv E^{cg} s_g s_c$, where $s_g \in [0,1]$ and $s_c \in [0,1]$ are random scale factors selected from a uniform distribution for each gene g and condition c. From $\boldsymbol{E}_S$ we calculated the normalized expression matrices $\boldsymbol{E}_G$ and $\boldsymbol{E}_C$ according to Eqs. (3) and (4). (b) From the vector $\boldsymbol{c}_1$, whose nonzero components $c_1^{(c)}$ specify the conditions of the upper-left module in (a) we calculated the vectors $\boldsymbol{g}_S = \boldsymbol{E}_S^T \boldsymbol{c}_1$, $\boldsymbol{g}_C = \boldsymbol{E}_C^T \boldsymbol{c}_1$, and $\boldsymbol{g}_G = \boldsymbol{E}_G^T \boldsymbol{c}_1$. We plot their components (horizontal axes) $g_S^{(g)}$ (black), $g_C^{(g)}$ (dark gray), and $g_G^{(g)}$ (light gray) as functions of the gene index (vertical axis). Only for $\boldsymbol{g}_C$, obtained according to normalization used in the ISA, *all* the components associated with the genes of the module are significantly larger than the others. (c) From the vector $\boldsymbol{g}_1$, whose nonzero components $g_1^{(g)}$ specify the genes of the upper-left module in (a) we calculated the vectors $\boldsymbol{c}_S = \boldsymbol{E}_S \boldsymbol{g}_1$, $\boldsymbol{c}_C = \boldsymbol{E}_C \boldsymbol{g}_1$, and $\boldsymbol{g}_G = \boldsymbol{E}_G \boldsymbol{c}_1$. We plot their components (horizontal axes) $c_S^{(c)}$ (black), $c_C^{(c)}$ (dark gray), and $c_G^{(c)}$ (light gray) as functions of the condition index (vertical axis). Only for $\boldsymbol{c}_G$, obtained according to normalization used in the ISA, *all* the components associated with the conditions of the module are significantly larger than the others.

ration can change the scale of the results. Similarly the dynamic range of two distinct genes could differ greatly even though the shape of their condition profiles might be similar. To overcome this difficulty we have introduced the normalized matrices $\boldsymbol{E}_G$ and $\boldsymbol{E}_C$ [cf. Eqs. (3) and (4)].

In order to study the impact of the normalization on our algorithm we generated an *in silico* expression matrix $\boldsymbol{E}$ corresponding to two overlapping modules of equal size and strength [see Sec. VI for more details on the model used to generate these data]. We selected random scale factors $s_g, s_c \in [0,1]$ for each gene g and condition c from a uniform distribution and transformed the elements of the expression matrix according to $E^{cg} \rightarrow E_S^{cg} \equiv E^{cg} s_g s_c$. Unlike the original expression matrix $\boldsymbol{E}$, the rescaled expression matrix $\boldsymbol{E}_S$ [shown in Fig. 1(a)] corresponds to the realistic scenario where the entities of the expression data have been recorded at different scales. From $\boldsymbol{E}_S$ we calculated the normalized matrices $\boldsymbol{E}_C$ and $\boldsymbol{E}_G$.

The question we ask is which normalization has to be employed in order to reveal the ''correct'' genes from the conditions associated with the underlying module, and which normalization leads to the ''correct'' conditions, given the genes of the module. To answer this question we defined the vectors $\boldsymbol{g}_1$ and $\boldsymbol{c}_1$ by assigning nonzero components only for the genes and conditions of one of the modules, respectively. Using these vectors we computed $\boldsymbol{c}_S = \boldsymbol{E}_S \cdot \boldsymbol{g}_1$, $\boldsymbol{c}_C = \boldsymbol{E}_C \cdot \boldsymbol{g}_1$, and $\boldsymbol{c}_G = \boldsymbol{E}_G \cdot \boldsymbol{g}_1$ as well as $\boldsymbol{g}_S = \boldsymbol{E}_S^T \cdot \boldsymbol{c}_1$, $\boldsymbol{g}_C = \boldsymbol{E}_C^T \cdot \boldsymbol{c}_1$, and $\boldsymbol{g}_G = \boldsymbol{E}_G^T \cdot \boldsymbol{c}_1$. The components of the resulting gene and condition vectors are plotted in Figs. 1(b) and 1(c), respectively.

One can see that only for $\boldsymbol{g}_C$ and $\boldsymbol{c}_G$ [corresponding to the the ''correct'' normalizations as used in the ISA, cf. Eqs. (10) and (11)] *all* the components associated with the genes and conditions of the module [specified by $(\boldsymbol{g}_1, \boldsymbol{c}_1)$] are significantly larger than the others. For missing or ''wrong'' normalization there are large fluctuations among the vector components. Hence applying a threshold would only capture part of the relevant genes or conditions in this case. Thus $\boldsymbol{E}_C$ is best suited to identify the genes of a module from a set of conditions that is a good approximation of C_m, while $\boldsymbol{E}_G$ is the proper normalization to obtain the conditions of a module from a set of genes close to G_m. Note that using these ''correct'' normalizations, it is even possible to distinguish the

genes and conditions associated exclusively with the specified module from those that belong also to the other module, because the latter obtain a somewhat lower score.

V. ANALYSIS OF THE ISA

The fundamental issue is how well the ISA can reveal relatively small, noisy, and possibly overlapping modules from the expression matrix. In this section we address this question by considering a simple model where the expression matrix corresponds to a *single* transcription module. Our idea is to consider the gene vector that undergoes iterations as a stochastic entity and to study how its distribution evolves under the iterations. This approach allows us to quantify how the efficiency of our algorithm depends on the size of the module and the noise in the expression data.

A. Linear recursions

In the following we consider a slightly simplified iterative scheme, where no threshold function is applied to the condition vector. In this case one can write an iterative equation that depends only on the gene vector. If, moreover, no gene threshold is applied the iterations are defined through the linear equation [cf. Eq. (A12) in the Appendix]

$$\hat{\boldsymbol{g}}^{(n)}=\frac{\boldsymbol{\mathcal{C}}\cdot\boldsymbol{g}^{(n-1)}}{|\boldsymbol{\mathcal{C}}\cdot\boldsymbol{g}^{(n-1)}|}. \tag{17}$$

Here the matrix $\boldsymbol{\mathcal{C}}=\boldsymbol{E}^T\cdot\boldsymbol{E}$ emerges from applying first Eq. (13) and then Eq. (14). As we mentioned before, the fixed points of this linear recursion are the eigenvectors of $\boldsymbol{\mathcal{C}}$.

Let us consider the simplest scenario corresponding to a single set of coregulated genes $G_1\subset G$ whose coregulation is triggered by the conditions in $C_1\subset C$. Specifically, we assume that all the genes in G_1 are equally important, such that a noise-free measurement would result in identical condition profiles for these genes. In this ideal case the matrix elements $\mathcal{C}^{gg'}$ would equal some constant if both g and g' belong to G_1 and be zero otherwise. In order to model the effect of noisy data we consider the elements of $\boldsymbol{\mathcal{C}}$ as random variables with mean value

$$\langle\mathcal{C}^{gg'}\rangle=\begin{cases}\mu_{\mathcal{C}}, & g,g'\in G_1\\ 0, & \text{otherwise,}\end{cases} \tag{18}$$

and variance $V_{\mathcal{C}}$ for all $g,g'\in G$. In the absence of noise (i.e., $V_{\mathcal{C}}=0$) the matrix $\boldsymbol{\mathcal{C}}$ possesses only a single (nontrivial) eigenvector $\boldsymbol{g}^{(0)}$, whose nonzero components specify the genes of the TM. However, for $V_{\mathcal{C}}>0$ this is not true anymore.

Assume that we knew the eigenvector of $\boldsymbol{\mathcal{C}}$ for $V_{\mathcal{C}}=0$ and use it as a (binary) seed $\boldsymbol{g}^{(0)}$ for Eq. (17) with a noisy realization of $\boldsymbol{\mathcal{C}}$ (i.e., $V_{\mathcal{C}}>0$). The question is whether the fixed point resulting from $\boldsymbol{g}^{(0)}$ still characterizes the genes of the module. In general the vector $\hat{\boldsymbol{g}}^{(1)}$ obtained by the first iteration does not coincide with $\hat{\boldsymbol{g}}^{(0)}$. Due to the probabilistic description of $\boldsymbol{\mathcal{C}}$ we can only determine the mean and the variance of the components of $\boldsymbol{g}^{(1)}=\boldsymbol{\mathcal{C}}\cdot\boldsymbol{g}^{(0)}$. The mean of $g_g^{(1)}=\Sigma_{g'}\mathcal{C}^{gg'}g_{g'}^{(0)}$ is equal to the number of genes in the module, $N_G^{(m)}$, times $\mu_{\mathcal{C}}$ if $g\in G_1$, and zero otherwise. Similarly the variance of $g_g^{(1)}$ is $N_G^{(m)}V_{\mathcal{C}}$. Here we only used the additivity of the mean and the variance. However, already for $g_g^{(2)}$ in the next iteration we need to deal with products of random variables. To this end, we note that for two independent random variables a and b we have (see part 2 of the Appendix for proof)

$$\langle ab\rangle=\langle a\rangle\langle b\rangle$$

and

$$V(ab)=V(a)V(b)+V(a)\langle b\rangle^2+V(b)\langle a\rangle^2. \tag{19}$$

Using these results we find that the mean values of the components of the vector $\boldsymbol{g}^{(n)}=\boldsymbol{\mathcal{C}}\cdot\boldsymbol{g}^{(n-1)}$ are given by

$$\langle g_g^{(n)}\rangle=\begin{cases}\mu_G^{(n)}\equiv N_G^{(m)}\mu_{\mathcal{C}}\mu_G^{(n-1)}, & g\in G_1\\ 0, & g\notin G_1,\end{cases} \tag{20}$$

where $\mu_G^{(n-1)}$ denotes the mean of the components $g_g^{(n-1)}$ associated with the module ($g\in G_1$). Only for the genes in G_1 there are $N_G^{(m)}$ matrix elements in $\boldsymbol{\mathcal{C}}$ that contribute constructively to $\langle g_g^{(n)}\rangle$. Similarly, the variances of $g_g^{(n)}$ are

$$V(g_g^{(n)})=\begin{cases}V_G^{(n)}\equiv\Delta N_G V_{\mathcal{C}}\tilde{V}_G^{(n-1)}+N_G^{(m)}[V_{\mathcal{C}}V_G^{(n-1)}+V_{\mathcal{C}}(\mu_G^{(n-1)})^2+V_G^{(n-1)}\mu_{\mathcal{C}}^2], & g\in G_1\\ \tilde{V}_G^{(n)}\equiv\Delta N_G V_{\mathcal{C}}\tilde{V}_G^{(n-1)}+N_G^{(m)}V_{\mathcal{C}}[V_G^{(n-1)}+(\mu_G^{(n-1)})^2], & g\notin G_1,\end{cases} \tag{21}$$

where $\Delta N_G\equiv N_G-N_G^{(m)}$ denotes the number of genes that do not belong to the module. Note that $V_G^{(n)}$ has an additional term with respect to $\tilde{V}_G^{(n)}$, due to the contribution of the nonzero mean values in $\boldsymbol{\mathcal{C}}$.

In order to assess whether the iterations improve the separability between distributions of the genes within ($g\in G_1$) and outside ($g\notin G_1$) the module, we introduce the rescaled variances

$$v_G^{(n)}\equiv\frac{V_G^{(n)}}{(\mu_G^{(n)})^2}\quad\text{and}\quad\tilde{v}_G^{(n)}\equiv\frac{\tilde{V}_G^{(n)}}{(\mu_G^{(n)})^2}. \tag{22}$$

Note that $v_G^{(n)}$ and $\tilde{v}_G^{(n)}$ are dimensionless and invariant under the normalization of the gene vectors. $v_G^{(n)}\ll 1$ implies that the distribution of the genes associated with the module is well separated from the distribution of the genes that do not

belong to the module. Using Eqs. (20) and (21) we obtain the following recursive equations:

$$\tilde{v}_G^{(n)} = \frac{\Delta N_G v_{\mathcal{C}}}{(N_G^{(m)})^2} \tilde{v}_G^{(n-1)} + \frac{v_{\mathcal{C}}}{N_G^{(m)}} (v_G^{(n-1)} + 1), \tag{23}$$

$$v_G^{(n)} = \tilde{v}_G^{(n)} + \frac{v_G^{(n-1)}}{N_G^{(m)}}, \tag{24}$$

where $v_{\mathcal{C}} \equiv V_{\mathcal{C}}/\mu_{\mathcal{C}}^2$ is the (fixed) noise-to-signal ratio of the expression matrix.

If $N_G^{(m)} \gg 1$ the second term in Eq. (24) is negligible and we can ignore the small difference between $v_G^{(n)}$ and $\tilde{v}_G^{(n)}$. Then, setting $\tilde{v}_G^{(n)} = v_G^{(n)}$ in Eq. (23) leads to the approximate recursive equation

$$v_G^{(n)} = \frac{N_G v_{\mathcal{C}}}{(N_G^{(m)})^2} v_G^{(n-1)} + \frac{v_{\mathcal{C}}}{N_G^{(m)}}. \tag{25}$$

This equation converges to

$$v_G^{(*)} \equiv \left(\frac{N_G^{(m)}}{v_{\mathcal{C}}} - \frac{N_G}{N_G^{(m)}} \right)^{-1}, \tag{26}$$

provided that

$$v_{\mathcal{C}} < v_{\mathcal{C}}^{crit} \equiv \frac{(N_G^{(m)})^2}{N_G}. \tag{27}$$

For further reference we state this result also for the *signal-to-noise ratio*

$$\rho_G^{(n)} \equiv \frac{\mu_G^{(n)}}{\sqrt{V_G^{(n)}}} = (v_G^{(n)})^{-1/2}. \tag{28}$$

The corresponding fixed-point value equals

$$\rho_G^{(*)} = \{ N_G^{(m)} [\rho_{\mathcal{C}}^2 - (\rho_{\mathcal{C}}^{crit})^2] \}^{1/2}, \tag{29}$$

if

$$\rho_{\mathcal{C}} \equiv \frac{\mu_{\mathcal{C}}}{\sqrt{\sigma_{\mathcal{C}}}} > \rho_{\mathcal{C}}^{crit} \equiv \frac{\sqrt{N_G}}{N_G^{(m)}}, \tag{30}$$

and is zero otherwise.

The interpretation of the critical value $v_{\mathcal{C}}^{crit}$ for the noise in the expression data is straightforward: Only sets of genes that are sufficiently large and whose coregulation is recorded in the expression matrix with relatively low noise (i.e., $v_{\mathcal{C}} < v_{\mathcal{C}}^{crit}$) can be captured by the iterative procedure without threshold in Eq. (17). Actually Eq. (30) is only a necessary condition for the identification of a module, since for a reliable separation of the distributions of the gene scores associated with the module, we need $\rho_G^{(*)} \gg 0$. As we mentioned before, the number of genes associated with cellular functions is expected to be rather limited, $N_G^{(m)} \ll N_G$. Therefore we conclude that Eq. (30) presents a serious limitation for the extraction of biologically relevant modules through the analysis of the eigenvectors of $\boldsymbol{\mathcal{C}}$ (as in SVD).

B. Noise reduction by the threshold function

As discussed in the preceding section the noise in the expression data may obstruct the identification of a TM. A fundamental aspect of the threshold functions in the ISA is to reduce the effect of such noise by excluding the bulk of the genes and conditions that do not contribute information but rather increase the level of background noise.

To illustrate this point, let us repeat the study of noise propagation presented above for the simplified iterative scheme like in Eq. (17), but with the linear map followed by a threshold function:

$$\boldsymbol{g}^{(n)} = f_t(\boldsymbol{\mathcal{C}} \cdot \hat{\boldsymbol{g}}^{(n-1)}), \tag{31}$$

where f_t is defined in Eq. (9) and we use a linear weight function $w(x) = x$. Let us assume that the gene scores are distributed according to normal distributions $\mathcal{N}(x; \mu, \sigma)$, where μ and σ refer to the mean and the standard deviation of the random variable x. As a result of the threshold function only

$$\tilde{N}_G^{(m)} = N_G^{(m)} \int_t^{\infty} \mathcal{N}(\rho; \rho_G^{(n-1)}, 1) d\rho \tag{32}$$

genes from the module contribute constructively to the mean in Eq. (20). Similarly, only $\tilde{N}_G^{(m)}$ genes from the module and

$$\Delta \tilde{N}_G = \Delta N_G \int_t^{\infty} \mathcal{N}(\rho; 0, 1) d\rho \tag{33}$$

genes outside the module contribute to the variance of $g_g^{(n)}$ in Eq. (21). $\tilde{N}_G^{(m)}$ is the expected number of genes in the module, whose score has not been set to zero by the threshold function. Similarly, $\Delta \tilde{N}_G$ is the expected number of genes that do not belong to the module, but have a nonzero score. The crucial point is that, because of the different mean values of the two distributions, the threshold function excludes more genes that do not belong to the module than genes that do belong to the module. For example, if $\rho_G^{(0)} = 3$ for the initial (normal) distribution, then a threshold $t = 2$ would remove almost 98% of the genes outside the module ($\Delta \tilde{N}_G \simeq 0.023 \times \Delta N_G$), but less than 16% of the genes associated with the module ($\tilde{N}_G^{(m)} \simeq 0.841 \times N_G^{(m)}$). We note that the precise shape of the distribution function is in fact not crucial, since our derivation relies only on the additivity of the mean values and variances, and Eq. (19).

It follows that the mean values and variances of the components of the vector $\boldsymbol{g}^{(n)}$ are given by the same expression as in Eqs. (20) and (21), respectively, except that we have to replace $N_G^{(m)}$ by $\tilde{N}_G^{(m)}$ and ΔN_G by $\Delta \tilde{N}_G$. Substituting the effective numbers $\tilde{N}_G^{(m)}$ and $\Delta \tilde{N}_G$ into Eqs. (20) and (21) the argument leading to the expression for the fixed-point signal-to-noise ratio in Eq. (29) is essentially unchanged, and we have

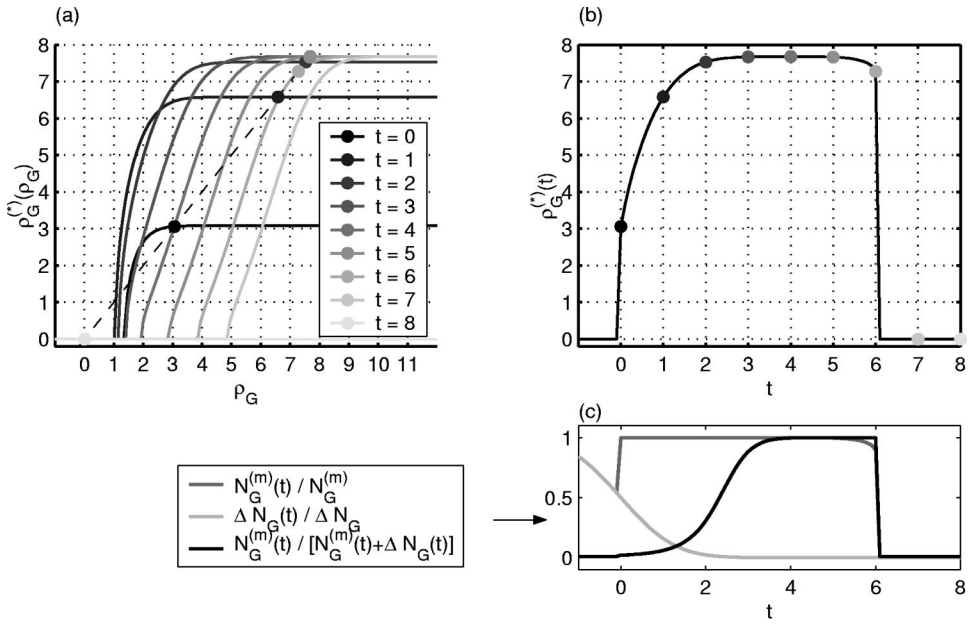

FIG. 2. Finding the fixed point value of the signal-to-noise ratio. (a) The fixed point value of the signal-to-noise ratio $\rho_G^{(*)}(t)$ is found by solving Eq. (34) (cf. Sec. V). We plot its right-hand side RHS $(\rho_G,t)\equiv[\tilde{N}_G^{(m)}\rho_C^2-(\tilde{N}_G^{(m)}+\Delta\tilde{N}_G)/\tilde{N}_G^{(m)}]^{1/2}$ as a function of ρ_G for several values of the threshold t as indicated in the legend (setting $N_G=6000$, $N_G^{(m)}=60$, $\rho_C=1$). RHS (ρ_G,t) depends on ρ_G and t through the effective numbers $\tilde{N}_G^{(m)}(t,\rho_G)$ and $\Delta\tilde{N}_G(t)$ [defined in Eqs. (32) and (33)] that denote the expected number of genes inside and outside the module that passed the threshold. Each curve increases monotonically from zero to its maximal value $\rho_G^{max}(t)$. For $\rho_G \gg t$, the effective number $\tilde{N}_G^{(m)}$ approaches $N_G^{(m)}$. In this limit $\rho_G^{max}(t)$ depends on t only through $\Delta\tilde{N}_g$, which goes to zero for $t\gg 1$. Thus $\rho_G^{max}(t)\to\sqrt{N_G^{(m)}\rho_C^2-1}$ asymptotically. According to Eq. (34) the fixed-point solutions for the signal-to-noise ratio $\rho_G^{(*)}(t)$ are given by $\rho_G=\mathrm{RHS}(\rho_G,t)$ and therefore correspond to the intersections (indicated by the big dots) of these curves with the diagonal (shown as a dashed line). (b) The solutions in (a) are plotted as a function of the threshold t. For a relatively small threshold ($t\lesssim 2$) $\rho_G^{(*)}(\rho_G,t)$ increases rapidly as a function of t, saturates to ρ_G^{max} for $t\gtrsim 2$, and suddenly falls off to zero at a certain threshold $t_{trans}(\approx 6)$. This behavior can be understood from (a): For a low threshold the intersection of curves for RHS(ρ_G,t) with the diagonal appears at small values of ρ_G. For larger t the intersections occur in the saturated regime of RHS(ρ_G,t), such that $\rho_G\simeq\rho_G^{max}(t)$. However, if t is too large the curves do not intersect with the diagonal and there is no solution. (c) $\tilde{N}_G^{(m)}(t)/N_G^{(m)}$ (dark gray) as well as $\Delta\tilde{N}_G(t)/\Delta N_G$ (light gray) and $\varrho(t)\equiv\tilde{N}_G^{(m)}(t)/[\tilde{N}_G^{(m)}(t)+\Delta\tilde{N}_G(t)]$ (black) are shown as functions of t. $\varrho(t)\simeq 1$ for $3\lesssim t<6$, indicating the optimal regime for the threshold.

$$\rho_G^{(*)}=\{\tilde{N}_G^{(m)}[\rho_C^2-(\tilde{\rho}_C^{crit})^2]\}^{1/2}, \tag{34}$$

with

$$\tilde{\rho}_C^{crit}\equiv\frac{\sqrt{\tilde{N}_G^{(m)}+\Delta\tilde{N}_G}}{\tilde{N}_G^{(m)}}. \tag{35}$$

Note that unlike Eq. (29), the right-hand side of Eq. (34) still depends on $\rho_G^{(*)}$ through $\tilde{N}_G^{(m)}$. Therefore Eq. (34) is an integral equation for $\rho_G^{(*)}$ which can be solved numerically. A graphical solution of this equation is provided in Fig. 2 for different thresholds and a specific choice of the parameters N_G, $N_G^{(m)}$, and v_C (see caption for details).

As can be seen in Fig. 3(a), applying a threshold function improves significantly the identification of the module. We show the fixed point value of the signal-to-noise ratio, $\rho_G^{(*)}$, as a function of both the threshold t and the (fixed) signal-to-noise ratio ρ_C of the expression data. In the absence of a threshold function $\rho_G^{(n)}$ converges to zero if ρ_C is below some critical value ρ_C^{crit}. Applying a threshold, $\rho_G^{(n)}$ converges to a finite value, even if $\rho_C<\rho_C^{crit}$ (but $\rho_C>\tilde{\rho}_C^{crit}$), indicating the identification of the module. Moreover, one can see from Fig. 3(a) that there is an optimal regime for the threshold t, where $\rho_G^{(*)}(t,\rho_C)$ is (nearly) maximal. Within this regime $\rho_G^{(*)}(t,\rho_C)$ depends only weakly on t, so the convergence is robust with respect to the exact choice of the threshold. The size of this regime increases with ρ_C.

In order to quantify the relative increase of the fixed point value of the signal-to-noise ratio $\rho_G^{(*)}(t,\rho_C)$ due to the application of the threshold function we define the ratio

$$r(t,\rho_C)\equiv\frac{\rho_G^{(*)}(t,\rho_C)-\rho_G^{(*)}(\rho_C)}{\rho_G^{(*)}(t,\rho_C)}, \tag{36}$$

where $\rho_G^{(*)}(\rho_C)$ refers to the value to which the signal-to-noise ratio converges when no threshold is applied. For $\rho_G^{(*)}(t,\rho_C)=0$ we set $r(t,\rho_C)$ to zero. We show $r(t,\rho_C)$ as a function of t and ρ_C in Fig. 3(b). The figure shows that there exists a large region in the parameter space of t and $\rho_C<\rho_C^{crit}$, where the iterations only converge to a positive value due to the threshold. Moreover, even for $\rho_C>\rho_C^{crit}$, where the iterative schemes converges to a positive value also without a threshold, there exists a large region, where

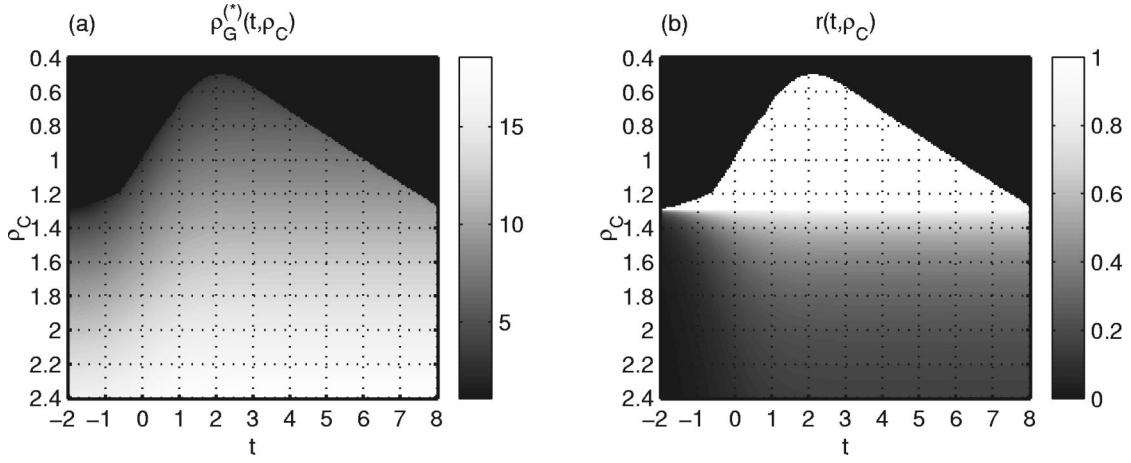

FIG. 3. Properties of the fixed point value of the signal-to-noise ratio. (a) The fixed point value of the signal-to-noise ratio, $\rho_G^{(*)}(t,\rho_C)$, characterizes the separability between the gene score distributions for the genes inside and outside the single module (cf. Sec. V for details). The plot shows $\rho_G^{(*)}(t,\rho_C)$ as a function of both the threshold t and the (fixed) signal-to-noise ratio in the expression matrix ρ_C. For very small thresholds $\rho_G^{(*)}(t,\rho_C)$ vanishes if ρ_C is below some critical value $\rho_C^{crit}\approx 1.3$. However, increasing the threshold the iterations converge to a finite fixed point, $\rho_G^{(*)}(t,\rho_C)>0$, even if $\rho_C<\rho_C^{crit}$ (but $\rho_C>\tilde{\rho}_C^{crit}\gtrsim 0.5$). There is an optimal regime for the threshold t, where $\rho_G^{(*)}(t,\rho_C)$ is (near to) maximal. Within this regime $\rho_G^{(*)}(t,\rho_C)$ depends only weakly on t, so the convergence is robust with respect to the exact choice of the threshold. The size of this regime increases with ρ_C. (b) The ratio $r(t,\rho_C)\equiv(\rho_G^{(*)}(t,\rho_C)-\rho_G^{(*)}[\rho_C)]/\rho_G^{(*)}(t,\rho_C)$ characterizes the improvement in the identification of transcription modules that is achieved by the application of the threshold function. $[\rho_G^{(*)}(\rho_C)$ denotes the fixed-point value of the signal-to-noise ratio in the absence of a threshold, and $r(t,\rho_C)$ is set to zero for $\rho_G^{(*)}(t,\rho_C)=0.]$ We show $r(t,\rho_C)$ as function of t and ρ_C. The regime, where $\rho_C<\rho_C^{crit}$ is subdivided into a white region $[r(t,\rho_C)=1]$, where the iterative scheme only converges to a positive value, $\rho_G^{(*)}(t,\rho_C)>0$, due to the threshold and a black area $[r(t,\rho_C)=0]$, where the iterative schemes does not converge to a positive value implying that the module cannot be identified in this regime. Note that also for $\rho_C>\rho_C^{crit}$, where the iterative scheme converges to a positive value even without a threshold, there exists a large region in the parameter space of t and ρ_C [the light gray area for $r(t,\rho_C)$], where $\rho_G^{(*)}(t,\rho_C)$ is significantly larger than $\rho_G^{(*)}(t)$.

$\rho_G^{(*)}(t,\rho_C)$ is significantly larger than $\rho_G^{(*)}(t)$. Thus we conclude that the threshold function improves significantly (and in certain cases makes at all possible) the convergence of a noisy input set to a gene vector that specifies the TM.

We have also performed numerical simulations of the iterative scheme in Eq. (31). To this end we employed *in silico* expression data that were generated according to Eq. (18) and superimposed with a certain level of noise. The initial gene sets were composed such that only the distribution of the genes scores associated with the module had a nonzero mean value, while the distribution of the remaining genes was centered around zero. The simulation allowed us to trace the evolution of the two distributions under the iterations. The results indicate a good agreement between the numerical and the analytical results. Details of this analysis are presented in Fig. 4. In particular, in Fig. 4(d) we show an example where only the application of a proper threshold leads to a separation between the two distributions.

VI. BEYOND THE SINGLE MODULE

In order to study the ISA in a more realistic scenario, we have performed further numerical simulations based on *in silico* expression data encoding several, possibly overlapping transcription modules. These data were generated according to the following simple model. Each module M_m is governed by a single (virtual) transcription factor whose activity is described by a pair of vectors $\{\boldsymbol{g}_m,\boldsymbol{c}_m\}$. The nonzero components $g_m^{(g)}$ of the gene vector $\boldsymbol{g}_m$ specify the genes that are transcribed if the transcription factor m is active, while the nonzero components $c_m^{(c)}$ of the condition vector $\boldsymbol{c}_m$ specify the conditions that activate this transcription factor. Then for N_M modules the log expression of gene g at condition c is defined as $E^{cg}=\Sigma_{m=1}^{N_M} g_m^{(g)} c_m^{(c)}$. The final expression matrix is obtained by adding noise to these matrix elements.

A. Expression data corresponding to two modules

As initial example we consider *in silico* expression data based on two transcription factors. We defined the components $c_m^{(c)}$ and $g_m^{(g)}$ for $m=1,2$ such that there are two overlapping transcription modules M_1 and M_2 (see Fig. 5 for details). We applied the ISA to a collection of input sets composed of randomly chosen genes. We found that the structure of the resulting fixed points depends strongly on the threshold t_G. Figure 5(b) shows the corresponding output sets for a discrete choice of thresholds. For a very low threshold ($t\simeq -2$) the output sets contain essentially all the genes. Applying a somewhat higher threshold ($t\simeq -1$) yields output sets containing all the genes that are associated with either of the two modules. For a moderate threshold ($t\simeq 0$) there are two types of output sets, comprising either the genes of M_1 or M_2. For a high threshold ($t\simeq 1$) all the output sets contain only those genes that belong to both modules. Finally, for a very high threshold ($t\simeq 2$) the output sets are empty. For intermediate values of the threshold value one observes relatively sharp transitions between these well-defined fixed points [Fig. 5(c)]. At these transitions the cor-

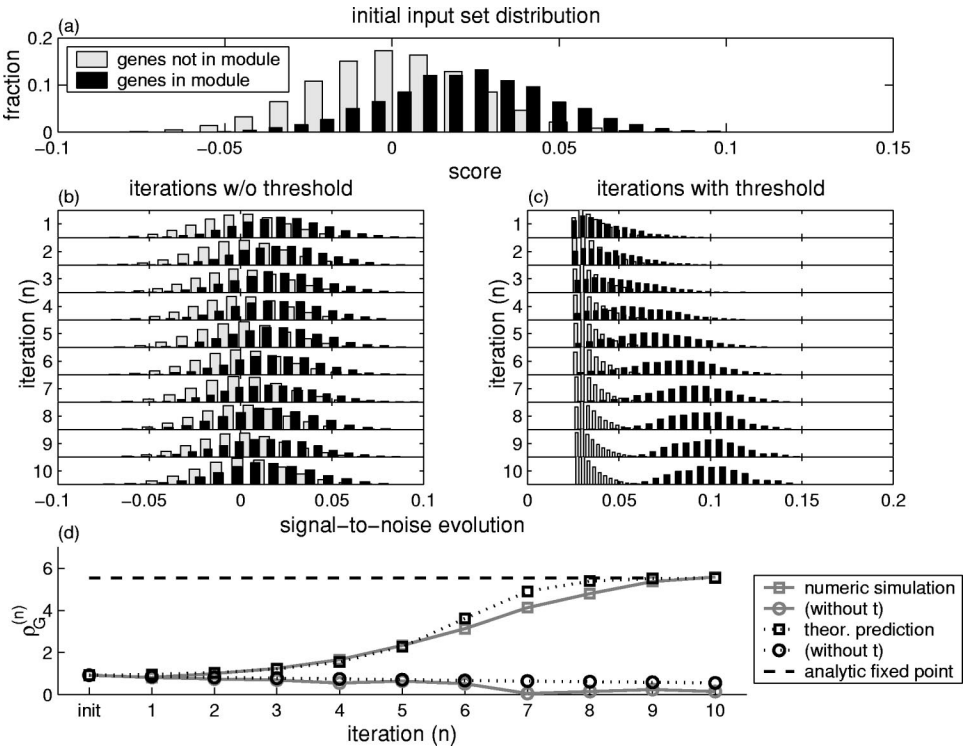

FIG. 4. Evolution of the score distributions under the ISA. (a) The distributions of the gene scores of 100 input sets which serve as seeds for the iterations of our algorithm: The distribution of the genes that are not part of the TM (light gray) has a vanishing mean value. The genes belonging to the module (black) are distributed with a positive mean value. Note that the two initial distributions cannot be distinguished from each other accurately. (b), (c) Evolution of the two distributions under the iterative scheme defined by Eq. (17). (b) Without applying a threshold, the mean of the signal distribution decreases in each iteration and the separability of the two distributions does not improve. (c) When a threshold ($t=1$) is applied the mean of the signal distribution increases in each step until it saturates at a value where the two distributions are well separated. (d) The signal-to-noise ratio $\rho_G^{(n)}$ characterizes the separability between the gene score distributions for the genes within and outside the module (cf. Sec. V for details). We plot $\rho_G^{(n)}$ as a function of the number of iterations n. The evolution of $\rho_G^{(n)}$ under the iteration schemes with (squares) and without (circles) a threshold obtained from the numerical simulation (gray) are in good agreement with the theoretical predictions (black) according to Eq. (25). We used $N_G=1700$, $N_G^{(m)}=40$, and $\rho_C=1$ for this figure.

respondence between the output sets and the modular structure of the data is less precise.

We have also varied the condition threshold t_C. Interestingly, for not too large a threshold ($t_C \lesssim 2$) the resulting gene output sets are almost independent of the choice of t_C. However, the condition output sets depend critically on the value of t_C and exhibit a similar behavior as the gene output sets in terms of structure (not shown). This is not surprising, since the ISA is symmetric with respect to genes and conditions. We conclude that scanning over different values of t_G and t_C reveals the modular structure of the expression data, starting from the "supermodule" $M_1 \cup M_1$, over its overlapping components M_1 and M_2, to the "submodule" $M_1 \cap M_1$.

B. Expression data corresponding to many modules

The above example shows that the ISA can identify overlapping modules. However, for $N_M=2$ there exist only $2^2=4$ possible transcriptional states, so the 100 conditions of the expression data are highly redundant. For real data the situation is reverse: The number of experimental conditions is much smaller than the possible number of transcriptional states. In order to study how the ISA deals with such a scenario, we considered a set of more realistic models based on many transcription modules. We investigated to what extend the ISA, as well as hierarchical clustering and SVD, were able to reconstruct these modules from the respective *in silico* expression data.

In the first numerical experiment we studied how the different algorithms handle noisy data. To this end we generated expression matrices corresponding to 1050 genes and 1000 experimental conditions that belong to 25 modules of different sizes, each associated with a transcription factor. In order to focus on the effect of noise we considered only nonoverlapping modules that do not share any genes or conditions. Onto the binary expression data we superimposed noise from a random distribution. We varied the width σ of this distribution, simulating different levels of noise.

In order to quantify how well the modules were identified by the different methods we proceeded as follows: For SVD we collected the 25 eigenvectors of the gene gene correlation matrix that were associated with the largest eigenvalues. For each of the 25 modules we selected the eigenvector that had the largest overlap with the gene vector characterizing the module, and in Fig. 6 we show the average Pearson coefficient between these two vectors (triangles). For hierarchical clustering we used the MATLAB implementation for average

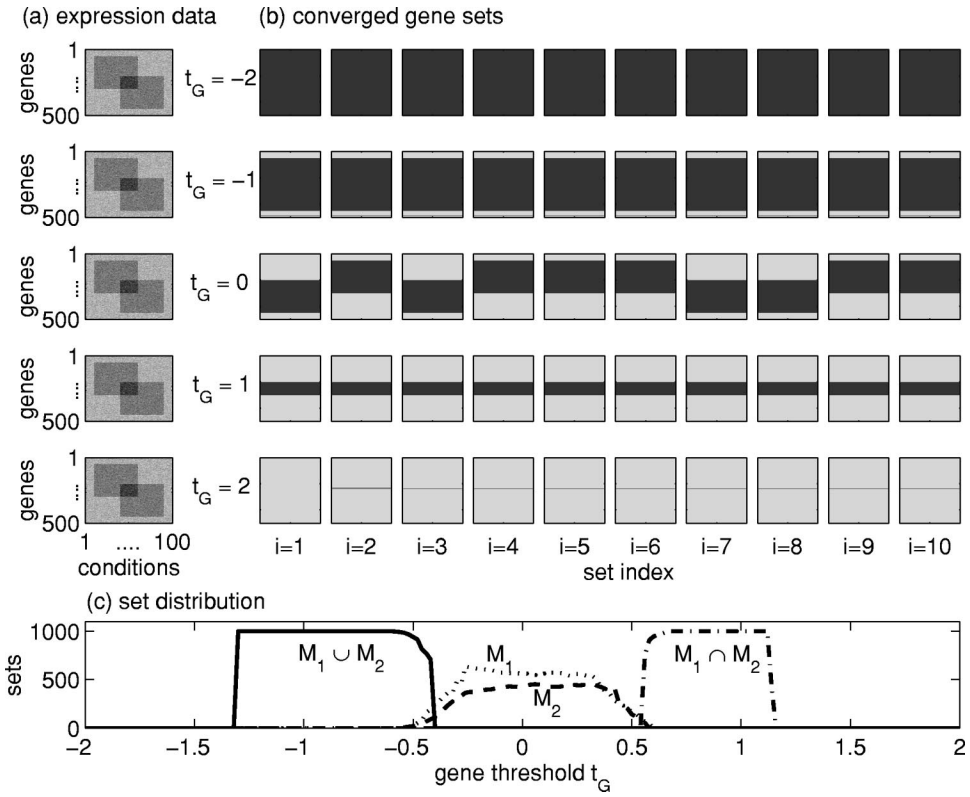

FIG. 5. Identification of overlapping modules. An *in silico* expression matrix describing 500 genes under 100 experimental conditions was generated according to the model introduced in the text. The data correspond to two overlapping transcription modules M_1 and M_2, each containing 250 genes and 50 conditions. (a) The expression matrix is shown for comparison on the left of each row. (b), (c) Using this matrix we applied the ISA to 1000 input sets composed of randomly chosen genes. Iterations were performed using different choices of the threshold t_G. (b) The boxes in each row represent ten of the resulting converged gene sets that were obtained for t_G as indicated on the left. Each box $i=1,\ldots,10$ is composed of 500 lines that specify the genes that appear in the corresponding fixed point. Genes that belong to the converged set are represented by a dark gray line, while the remaining genes are shown in light gray. For $t_G \simeq -2$ the output sets contain all the genes, $t_G \simeq -1$ yields output sets containing the genes that are associated with either of the two modules, for $t_G \simeq 0$ there are two types of output sets, comprising either the genes of M_1 or of M_2, for $t_G \simeq 1$ all the output sets contain only those genes that belong to both modules and for $t_G \simeq 2$ the output sets are essentially empty. (c) The number of sets that converged (within 95% accuracy) to $M_1 \cup M_1$ (solid line), M_1 (dotted line), M_2 (dashed line), or $M_1 \cap M_1$ (dash-dotted line) are plotted as functions of t_G. Scanning over different thresholds reveals the modular structure of the expression data ($M_1 \cup M_1 \rightarrow M_1, M_2 \rightarrow M_1 \cap M_1$).

linkage to compute the complete hierarchical cluster tree. Using this cluster tree we partitioned the expression matrix using different cutoffs such that the resultant partitions contained at least 15 and at most 40 clusters. From all these partitions we selected the one whose clusters had the highest average overlap with the gene content of the modules. This overlap is shown in Fig. 6 (squares). Finally, for the ISA we reconstructed the modules from the fixed points that occurred repeatedly. Namely, in order to avoid artifacts due to distinct, but very similar, fixed points, we ''fused'' these solutions using a procedure that resembles agglomerative clustering, albeit for modules rather than genes (see Ref. [25] for details). The fraction of correctly identified genes per module (circles) as well as the fraction of correctly identified modules (asterisks) is shown in Fig. 6. We conclude that for noisy data the identification capability of the ISA is superior to that of SVD and clustering. In particular, SVD is very sensitive to the addition of noise and fails to identify the modules accurately, even for a small level of noise. Clustering can handle a moderate amount of noise, but not as much as the ISA.

A second numerical experiment was designed to study quantitatively the ability to identify overlapping modules. We specify the regulatory complexity by the the number of transcription factors per gene n_{TF}. Only if each gene (and condition) is associated with exactly one transcription factor ($n_{TF}=1$) the expression matrix can be written in block-diagonal form. For larger values of n_{TF} distinct modules share common genes and conditions and the expression matrix cannot be reorganized into in block-diagonal shape. We applied the SVD, hierarchical clustering, and the ISA to the expression matrices generated for $n_{TF}=1,\ldots,6$ and evaluated the outputs in the same manner as described above (see Ref. [25] for related results). The results are shown in Fig. 7. One can see that the ISA could successfully identify all the transcription modules even in the case of highly overlapping modules. In contrast, for $n_{TF}>1$ the identification capabilities of SVD and clustering rapidly decrease. This is because the clustering algorithm does not allow for multiple assignments of one gene to different modules and therefore usually captures only small, incomplete fractions of the overlapping

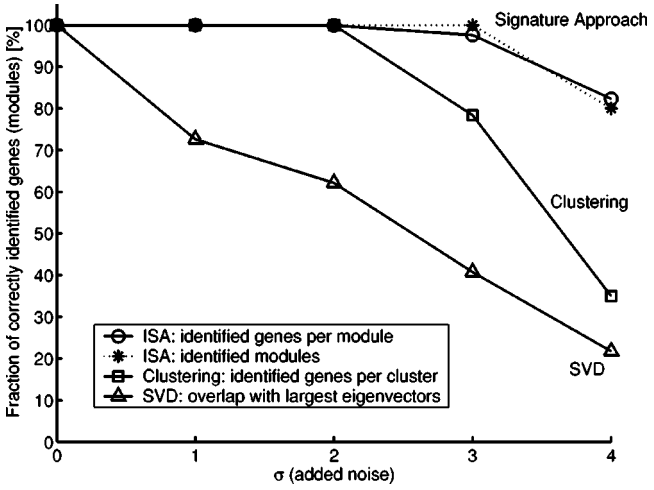

FIG. 6. Module identification from noisy expression data. *In silico* expression matrices for 1050 genes under 1000 conditions, corresponding to 25 nonoverlapping transcription modules of different sizes, were generated according to the model described in the text. Noise from a uniform distribution was superimposed onto these expression data. The width σ of this noise distribution was varied, simulating different levels of noise. We quantified the efficiency of different algorithms to retrieve the modules from the expression data as described in the text. We show the fraction of correctly identified genes for the ISA (circles), hierarchical clustering (squares), and SVD (triangles). For the ISA also the fraction of correctly identified modules is indicated (asterisks). SVD is very sensitive to the addition of noise and fails to identify the modules accurately, even for a small level of noise. Clustering can handle a moderate amount of noise, but not as much as the ISA.

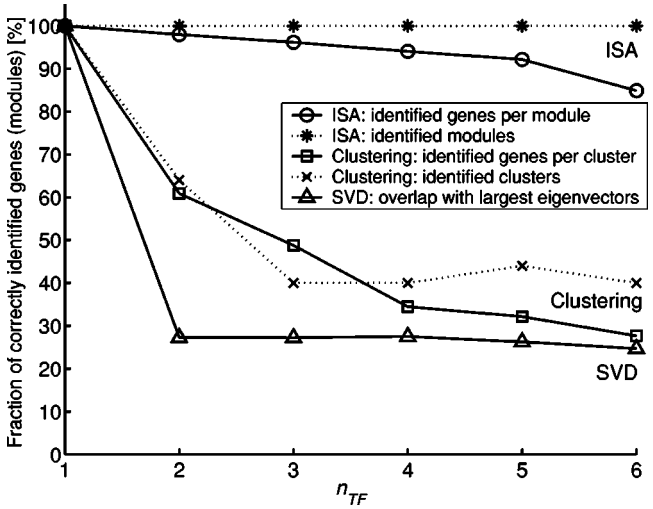

FIG. 7. Module identification in the presence of combinatorial regulation. *In silico* expression matrices corresponding to 25 overlapping modules were generated according to a model that allows for combinatorial regulation (see text for details). The degree of overlap between the modules is specified by the average number of transcription factors involved in the regulation of each gene (n_{TF}). Only for $n_{TF}=1$ each gene is associated with exactly one transcription factor. For larger values of n_{TF} distinct modules share common genes. We applied the SVD, hierarchical clustering (see Ref. [25] for related results), and the ISA to the expression matrices generated for $n_{TF}=1,\ldots,6$ and evaluated the outputs as described in the text. The ISA could successfully identify all the transcription modules even in the case of highly overlapping modules (asterisks). The fraction of correctly identified genes per module only decreases slightly as a function of n_{TF} (circles). In contrast, for $n_{TF}>1$ the identification capabilities of clustering (squares and crosses) and SVD (triangles) rapidly decrease. This is because the clustering algorithm does not allow for multiple assignments of one gene to different modules and therefore usually captures only small, incomplete fractions of the overlapping modules. Similarly, if the expression matrix cannot be reorganized into block-diagonal shape due to the overlap between the modules, the eigenvectors identified by SVD fail to characterize the modules properly.

modules. Similarly, if the expression matrix cannot be reorganized into block-diagonal shape due to the overlap between the modules, the eigenvectors identified by SVD fail to characterize the modules properly.

VII. APPLYING THE ISA TO YEAST EXPRESSION DATA

The analytical and numerical studies presented above indicate that the ISA is well suited for the analysis of expression data. In this section we give a brief presentation of the biological insight that can be obtained from applying our method to real data. We analyzed a diverse set of more than 1000 DNA-chip experiments that were obtained by different groups [6]. The yeast *Saccharomyces cerevisiae* is an ideal model organism to test our algorithm, due to the wealth of expression data and the large amount of additional biological knowledge that exists for this organism.

We have applied the ISA to the yeast expression data using different values for the gene threshold $t_G=1.8,1.9,\ldots,4.0$, while the condition threshold was fixed to $t_C=2.0$. (As we pointed out previously, the gene content of the modules depends only weakly on the exact choice for t_C.) For each value of t_G we employed $\sim 20\,000$ randomly composed initial gene sets of various sizes in the search for fixed points. The modules were reconstructed from the recurrent fixed points using a similar algorithm as for the *in silico* expression data. Indeed, such a processing of the ''raw'' fixed points is needed to avoid many similar modules that biologically correspond to the same coregulated unit.

The number of modules increases with t_G, ranging between five at the lowest level ($t_G=1.8$) to ~ 100 at the highest resolution ($t_G=4$). In contrast, the typical module size declines rapidly as a function of t_G. The stepwise increasing of t_G exposed many chains of closely related modules that persist for finite ranges $t_G \in [t_G^{bottom}, t_G^{top}]$. Increasing t_G, the number of genes assigned to each element of the chain decreases until the size of the module declines sharply at $t_G=t_G^{top}$ and either disappears completely or splits into two or more submodules. Likewise decreasing t_G beyond t_G^{bottom} destabilizes the fixed point, since many unrelated genes are added to the module that pull the module towards a different fixed point. In this case the module may either ''merge'' with another module or flow into a completely different fixed point.

The five stable fixed points identified for $t_G=1.8$ correspond to the central functions of the yeast organism: protein synthesis, cell cycle (G1), mating, amino-acid biosynthesis,

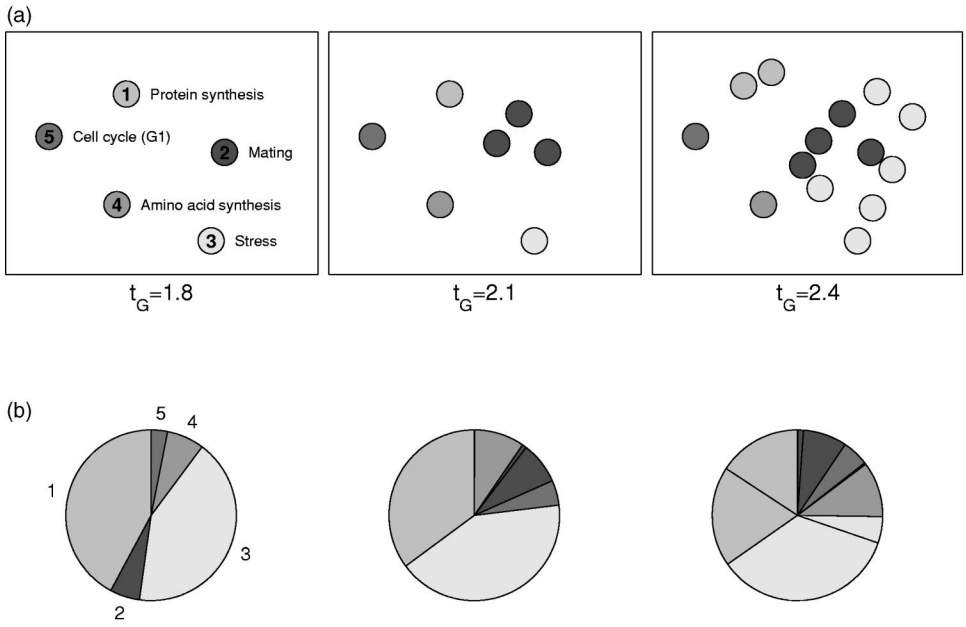

FIG. 8. Modular organization of yeast expression data. The iterative signature algorithm was applied to genome wide yeast expression data gathered by more than 1000 DNA-chip experiments. (a) The figure shows the identified modules at three different gene thresholds $t_G=\{1.8,2.1,2.4\}$. For each threshold the corresponding modules are displayed in a plane, such that their distance reflects their correlation with respect to conditions. Moving to a higher threshold, corresponding modules are kept in the same position in each plane, while the ''new'' modules are placed such that their position reflects best their correlation with the other modules. The leftmost plane corresponds to the lowest threshold ($t_G=1.8$), where only five fixed points exist. The corresponding modules can be associated with central functions of the yeast organism: protein synthesis, cell cycle (G1), mating, amino-acid biosynthesis, and stress response. We use different gray scales to indicate which of the fixed points that emerge at higher thresholds are related to these five central modules (i.e., they would converge to the respective module at the lowest threshold). (b) The pie charts show the number of random input sets that converged to the respective fixed point. The gray scales are as in (a).

and stress response. Each module contains between 100 and 300 genes. Protein synthesis and stress are the most dominant modules and comprise most of the experimental conditions of the data set. In fact, these modules remain fixed points throughout the entire range of thresholds considered here, and therefore can be considered the backbone of the transcriptional network.

A visualization of this network is presented in Fig. 8(a). For each threshold the corresponding modules are displayed in a plane, such that their distance reflects their correlation with respect to conditions. Moving to a higher threshold, nested sets of modules are kept in the same position in each plane, while the ''new'' modules are placed such that their position reflects best their correlation with the other modules. This organization of the chains of nested modules is somewhat similar to the data presentation by hierarchical trees commonly produced by cluster algorithms. However, in our case, chains of modules may extend over a finite range of t_G and distinct chains can contain common genes. Additional information, such as the number of input seeds that converged to the same fixed point [shown as pie charts in Fig. 8(b)], provides further inside into the transcriptional network.

In a previous analysis of the same data [25] we applied the map in Eqs. (10) and (11) to a variety of biologically motivated input sets $\{\boldsymbol{g}_i^{(0)}\}$ assembled according to prior knowledge of the regulatory sequence or function of the genes, and reconstructed the modules from recurrent realizations of the output sets defined by $\boldsymbol{g}^{(1)}$ and $\boldsymbol{c}^{(1)}$. Remarkably, the ISA (which requires no information beyond the expression data whatsoever) revealed essentially all the coregulated units that we found in this analysis, as well as several new transcription modules that had not been identified previously. Moreover, the ISA provides additional insight into the modular organization through the evolution of the modules over different threshold values. Studying the functional annotations of the genes assigned to the modules, we observed a strong coherence for the genes that have been annotated in most of these modules. This suggests that the ISA provides a biologically meaningful decomposition into coregulated units. A comprehensive discussion of the biological implications of this analysis is beyond the scope of this work and will be pursued elsewhere [30].

VIII. CONCLUSIONS

We have presented a method for the analysis of gene expression data. The innovation of our approach is twofold. On the conceptual level we provide a rigorous definition of what we want to extract from the expression data by introducing the notion of a TM. Our definition in Eq. (6) assigns to a TM both a set of coregulated genes and the set of experimental conditions under which this coregulation is the most stringent. The size of a TM depends critically on the associated

set of two thresholds that determine the similarity between the genes and conditions of the module, respectively. The genes and conditions of a TM are mutually consistent implying that the latter can be obtained from the former and vice versa. The notion of a TM is well motivated biologically. Ideally the genes and conditions can be associated with a transcription factor or a (fraction of) a pathway. Importantly, distinct modules may share both common genes and conditions.

On the computational level our definition of a TM provides the basis for a simple but efficient algorithm to obtain the modules encoded in the expression data. Starting from a set of randomly selected genes (or conditions) one refines iteratively the genes and conditions until they are mutually consistent and match the definition of a TM. The important point is that at each step of the iterations we apply a threshold function, thus maintaining only significantly coregulated genes and the associated coregulating conditions. The threshold stabilizes compact sets of coregulated genes and prevents the introduction of noise from unrelated genes and conditions. Using a sufficiently large number of initial random sets it is possible to determine all the fixed points of the iterative scheme for a given pair of thresholds. Scanning through a range of values for these thresholds decomposes the data into modules at different resolutions. Since the computation time for each iteration of our algorithm scales only linearly with the total number of genes it is particularly well suited for the analysis of large-scale expression data.

Considering a simplified scenario of a single transcription module embedded in a noisy background of unrelated genes, we showed analytically that the application of a threshold improves the convergence properties of the iterative scheme. Specifically, we considered the gene vector that undergoes iterations as a stochastic entity and studied the evolution of its distribution under the iterations for a given threshold. This allowed us to quantify how the successful identification of the module depends on the size of the module and the noise in the expression data.

Our analytical insights were confirmed numerically using computer-generated expression data. More complex gene regulation was also simulated *in silico*. Considering a model with two overlapping transcription modules, we showed that applying the ISA using a range of threshold values reveals the structure of the expression data at different resolutions. Depending on the value of the threshold our algorithm can reveal each of the two modules, as well as their union and intersection. Using large computer-generated expression matrices we studied the capability of the ISA to reveal a large number of overlapping transcription modules from noisy expression data. We find that our method is significantly more efficient at this task than standard tools, such as SVD and clustering.

The threshold functions as a resolution parameter in our analysis of real expression data. Using genome-wide expression data gathered in more than 1000 experimental conditions, we decomposed the yeast genome into sets of transcription modules at different resolutions. The modular decomposition reveals a hierarchical structure of the regulatory network. At the lowest resolution we identified five transcription modules that correspond to the central functions of the yeast organism. Increasing the threshold the number of modules increases while their size decreases. The functional coherence of these modules indicates both the reliability of our approach and the strong correlation between cofunction and coregulation at the transcriptional level in yeast. A comprehensive discussion of the biological implications of this analysis will be presented elsewhere [30].

Finally we note that our formalism can be applied to analyze any data set that consists of multicomponent measurements. While we presented our method in the context of gene expression data, it is clear that our approach is well suited to reveal the modular organization encoded in any data matrix. Applications of the ISA could include the analysis of biological data on protein-protein interactions or cell growth assays, as well as other large-scale data, where a meaningful reduction of complexity is needed.

ACKNOWLEDGMENTS

We thank J. Doyle for bringing our attention to the similarity between SVD and the ISA. We thank E. Domany, Y. Kafri, and S. Shnider for discussions and comments on the manuscript. This work was supported by the NIH Grant No. A150562, the Israeli Science Ministry and the Y. Leon Benoziyo Institute for Molecular Medicine. S.B. is supported by a Koshland grant.

APPENDIX

1. Singular value decomposition

This appendix reviews SVD, which is a common tool for the analysis of expression data. We use notations that make the similarities with the ISA the most apparent. SVD is used to reduce the dimensionality of the data by projecting them onto a subspace in such a way that as little information is lost as possible. To this end, consider the following matrix:

$$\boldsymbol{E}_m = \boldsymbol{c}_m \cdot \boldsymbol{g}_m^T, \tag{A1}$$

whose elements $E_m^{cg} = g_m^{(g)} c_m^{(c)}$ are simply the products of the components of a given gene vector $\boldsymbol{g}_m$ and condition vectors $\boldsymbol{c}_m$. For two binary vectors $\boldsymbol{g}_m$ and $\boldsymbol{c}_m$ (whose elements are either 0 or 1) E_m^{cg} is unity if the module m contains the gene g and the condition c (i.e., the relevant vector components are $g_m^{(g)}=1$ and $c_m^{(c)}=1$). For real vectors $\boldsymbol{g}_m \in \mathbb{R}^{N_G}$ and $\boldsymbol{c}_m \in \mathbb{R}^{N_C}$ it is useful to rewrite the matrix in Eq. (A1) as

$$\boldsymbol{E}_m = \mu_m \hat{\boldsymbol{c}}_m \cdot \hat{\boldsymbol{g}}_m^T, \tag{A2}$$

in terms of the normalized vectors $\hat{\boldsymbol{g}}_m = \boldsymbol{g}_m / |\boldsymbol{g}_m|$ and $\hat{\boldsymbol{c}}_m = \boldsymbol{c}_m / |\boldsymbol{c}_m|$. This normalization removes the ambiguity in the choice of $\boldsymbol{g}_m$ and $\boldsymbol{c}_m$ due to the invariance of $\boldsymbol{E}_m$ under the transformation $\boldsymbol{g}_m \to \phi \boldsymbol{g}_m$ and $\boldsymbol{c}_m \to \boldsymbol{c}_m / \phi$, where $\phi \neq 0$ is an arbitrary real number. The prefactor $\mu_m = |\boldsymbol{g}_m||\boldsymbol{c}_m|$ is just the product of the lengths of $\boldsymbol{g}_m$ and $\boldsymbol{c}_m$. Then each module is associated with a triple $(\mu_m, \hat{\boldsymbol{g}}_m, \hat{\boldsymbol{c}}_m)$ of a real number and two normalized vectors. Comparing the magnitude of any

two matrix elements E_m^{cg} and $E_m^{g'c'}$ reveals the relative importance between the gene condition pairs (g,c) and (g',c') for module m.

Multiplying $\boldsymbol{E}_m$ with an arbitrary gene vector $\boldsymbol{g}$ gives

$$\boldsymbol{E}_m \cdot \boldsymbol{g} = \alpha \cdot \hat{\boldsymbol{c}}_m \quad \text{with} \quad \alpha = \mu_m \hat{\boldsymbol{g}}_m^T \cdot \boldsymbol{g}, \tag{A3}$$

while multiplication of $\boldsymbol{E}_m^T = \mu_m \hat{\boldsymbol{g}}_m \cdot \hat{\boldsymbol{c}}_m^T$ with any condition vector $\boldsymbol{c}$ gives

$$\boldsymbol{E}_m^T \cdot \boldsymbol{c} = \beta \hat{\boldsymbol{g}}_m \quad \text{with} \quad \beta = \mu_m \hat{\boldsymbol{c}}_m^T \cdot \boldsymbol{c}. \tag{A4}$$

Thus $\boldsymbol{E}_m$ and $\boldsymbol{E}_m^T$ are projection operators onto the one-dimensional spaces spanned by $\hat{\boldsymbol{g}}_m$ and $\hat{\boldsymbol{c}}_m$, respectively. Consequently, theses matrices have rank 1.

Now the basic idea of SVD is to reduce the complexity of the data by expressing $\boldsymbol{E}$ in terms of a relatively small number $N_M (\ll N_G, N_C)$ of such rank 1 matrices,

$$\boldsymbol{E} = \sum_m^{N_M} \boldsymbol{E}_m + \boldsymbol{R}_{N_M}. \tag{A5}$$

Here $\boldsymbol{R}$ denotes the residual term whose Euclidean norm $|\boldsymbol{R}| = \sqrt{\Sigma_{g,c} (R^{cg})^2}$ has to be minimized in order to optimize the decomposition into modules in the above equation.

It is instructive to consider first the minimization for the case $N_M = 1$. We have

$$|\boldsymbol{R}|^2 = \sum_{g,c} (E^{cg} - E_m^{cg})^2 = \sum_{g,c} (E^{cg} - \mu_m \hat{c}_m^{(c)} \hat{g}_m^{(g)})^2 \tag{A6}$$

$$= \sum_{g,c} (E^{cg})^2 - 2\mu_m E^{cg} \hat{c}_m^{(c)} \hat{g}_m^{(g)} + \mu_m^2 (\hat{c}_m^{(c)} \hat{g}_m^{(g)})^2. \tag{A7}$$

Setting the derivative of $|\boldsymbol{R}|^2$ with respect to the component $\hat{c}_m^{(c)}$,

$$\frac{\partial |\boldsymbol{R}|^2}{\hat{c}_m^{(c)}} = \sum_g -2\mu_m E^{cg} g_m^{(g)} + 2\mu_m^2 (\hat{g}_m^{(g)})^2 \hat{c}_m^{(c)}, \tag{A8}$$

to zero we find that $\mu_m \hat{c}_m^{(c)} = \Sigma_g E^{cg} g_m^{(g)} / \Sigma_g (g_m^{(g)})^2$ or, recalling the normalization of $\hat{\boldsymbol{g}}_m$ and switching to vector notation,

$$\mu_m \hat{\boldsymbol{c}}_m = \boldsymbol{E} \cdot \hat{\boldsymbol{g}}_m. \tag{A9}$$

Similarly, equating $\partial |\boldsymbol{R}|^2 / \hat{g}_m^{(g)}$ to zero it follows that

$$\mu_m \hat{\boldsymbol{g}}_m = \boldsymbol{E}^T \cdot \hat{\boldsymbol{c}}_m. \tag{A10}$$

This remarkable result implies that $\boldsymbol{E}_m$ can be determined simply by solving simultaneously the linear equations in Eqs. (A9) and (A10). The latter is equivalent to a SVD of the matrix $\boldsymbol{E}$,

$$\boldsymbol{G}^T \cdot \boldsymbol{E} \cdot \boldsymbol{C} = \boldsymbol{M}, \tag{A11}$$

where $\boldsymbol{G} = (\hat{\boldsymbol{g}}_1, \hat{\boldsymbol{g}}_2, \ldots, \hat{\boldsymbol{g}}_r)$ and $\boldsymbol{C} = (\hat{\boldsymbol{c}}_1, \hat{\boldsymbol{c}}_2, \ldots, \hat{\boldsymbol{c}}_r)$ are orthogonal matrices. $\boldsymbol{M}$ is a diagonal matrix of the same dimensions as $\boldsymbol{E}$ whose nonzero elements are given by μ_m and ordered such that $\mu_1^2 \geqslant \mu_2^2 \geqslant \cdots \geqslant \mu_r^2$. $r \leqslant \min(N_G, N_C)$ is the rank of the expression matrix $\boldsymbol{E}$. Combining Eqs. (A9) and (A10), one finds

$$\boldsymbol{E}^T \cdot \boldsymbol{E} \cdot \hat{\boldsymbol{g}}_m = \mu^2 \hat{\boldsymbol{g}}_m, \tag{A12}$$

$$\boldsymbol{E} \cdot \boldsymbol{E}^T \cdot \hat{\boldsymbol{c}}_m = \mu^2 \hat{\boldsymbol{c}}_m, \tag{A13}$$

implying that $\boldsymbol{G}$ is composed of the eigenvectors $\hat{\boldsymbol{g}}_m$ of $\boldsymbol{E}^T \cdot \boldsymbol{E}$ and $\boldsymbol{C}$ consist of the eigenvectors $\hat{\boldsymbol{c}}_m$ of $\boldsymbol{E} \cdot \boldsymbol{E}^T$. One way to solve the above equations is to start with some initial gene vector $\hat{\boldsymbol{g}}^{(0)}$, obtain the corresponding condition vector via $\hat{\boldsymbol{c}}^{(1)} = \boldsymbol{E} \cdot \hat{\boldsymbol{g}}^{(0)} / |\boldsymbol{E} \cdot \hat{\boldsymbol{g}}^{(0)}|$ according to Eq. (A9), and use the result to compute $\hat{\boldsymbol{g}}^{(1)} = \boldsymbol{E}^T \cdot \hat{\boldsymbol{c}}^{(1)} / |\boldsymbol{E}^T \cdot \hat{\boldsymbol{c}}^{(1)}|$ using Eq. (A10). Iterating this alternating procedure as in Eqs. (13) and (14) converges to the pair $(\hat{\boldsymbol{g}}_1, \hat{\boldsymbol{c}}_1)$ associated with largest eigenvalue $\mu_1^2 = |\boldsymbol{E} \cdot \hat{\boldsymbol{g}}_1|^2$ provided that the initial vector $\hat{\boldsymbol{g}}^{(0)}$ was not orthogonal to $\hat{\boldsymbol{g}}_1$. Thus the predominant module emerges as the ''fixed point'' of the above coupled equations.

From Eq. (A6) it follows that $|\boldsymbol{R}|^2 = \Sigma_{g,c} (E^{cg})^2 - \mu_m^2$. Hence for $N_M = 1$ the norm of the residual term, $|\boldsymbol{R}|^2$, is minimized exactly by the triple $(\mu_1, \hat{\boldsymbol{g}}_1, \hat{\boldsymbol{c}}_1)$. It is straightforward to extend this approach to the expansion of the expression matrix in terms of several modules as in Eq. (A5). To this end one first computes $\boldsymbol{E}_1 = \mu_1 \hat{\boldsymbol{c}}_1 \cdot \hat{\boldsymbol{g}}_1^T$ as described above and applies the same scheme to the residual term $\boldsymbol{R}_1 = \boldsymbol{E} - \boldsymbol{E}_1$. This yields $\boldsymbol{E}_2 = \mu_2 \hat{\boldsymbol{c}}_2 \cdot \hat{\boldsymbol{g}}_2^T$ associated with the second largest eigenvalue μ_2. Repeating this procedure *sequentially* yields eventually the complete SVD of the matrix $\boldsymbol{E}$. However, for practical purposes it is usually sufficient to compute only a limited numbers of triples $(\mu_m, \hat{\boldsymbol{g}}_m, \hat{\boldsymbol{c}}_m)$ with $m = 1, \ldots, N_M$ until the norm of the residual term $|\boldsymbol{R}_{N_M}|^2 = \Sigma_{g,c} (E^{cg})^2 - \Sigma_{m=1}^{N_M} \mu_m^2$ is below a certain threshold. Thus, approximating the expression matrix in terms of a relatively small number of modules, $N_M \ll r$ reduces the complexity of the data.

There are two interpretations for the expansion in Eq. (A5) that depend on the way the expression data are viewed. If we consider the data as a collection of gene vectors $\boldsymbol{g}_c$ as in Eq. (1), then Eq. (A5) translates into an expansion of these vectors in terms of a collection of gene vectors, i.e.,

$$\boldsymbol{g}_c = \sum_{m=1}^{N_M} \mu_m \hat{c}_m^{(c)} \hat{\boldsymbol{g}}_m + \boldsymbol{g}_c^R \quad (c = 1, \ldots, N_C), \tag{A14}$$

where $\{\hat{\boldsymbol{g}}_m\}$ is the basis (one for *all* $\boldsymbol{g}_c$), and the expansion coefficients are given by $\mu_m \hat{c}_m^{(c)}$ (one for *each* $\boldsymbol{g}_c$). Moreover, for each $\boldsymbol{g}_c$ there is a residual gene vector $\boldsymbol{g}_c^R$ that determines how well $\boldsymbol{g}_c$ is approximated by the sum. Conversely, if we consider the data as a collection of condition vectors $\boldsymbol{c}_g$ as in Eq. (2), then the expansion in Eq. (A5) can be read as

$$\boldsymbol{c}_g = \sum_{m=1}^{N_M} \mu_m \hat{g}_m^{(g)} \hat{\boldsymbol{c}}_m + \boldsymbol{c}_g^R \quad (g=1,\ldots,N_G), \tag{A15}$$

where $\boldsymbol{c}_g^R$ denotes the residual condition vector. In this case the condition vectors of the modules, $\{\hat{\boldsymbol{c}}_m\}$, provide the basis of expansion, while the expansion coefficients for each $\boldsymbol{c}_g$ are given by $\mu_m \hat{g}_m^{(g)}$.

So far we have left the normalization of $\boldsymbol{E}$ unspecified. In fact the choice of normalization follows from the interpretation of the data, if, instead of a minimal residual term in Eq. (A6), one demands maximal variance among the *principal components* (the projections of the data rows or columns onto the eigenvectors associated with the largest eigenvalues). For example, if the expression data are viewed as a collection of gene vectors, one would like to find the vector $\hat{\boldsymbol{g}}_1$ that maximizes the variance of the principal components $c_1^{(c)} = \boldsymbol{g}_c^T \cdot \hat{\boldsymbol{g}}_1$, i.e.,

$$V_1^g = \frac{1}{N_C} \sum_{c=1}^{N_C} (c_1^{(c)} - \langle c_1^{(c)} \rangle_c)^2 = \frac{1}{N_C} \hat{\boldsymbol{g}}_1^T \cdot \boldsymbol{S}_g \cdot \hat{\boldsymbol{g}}_1. \tag{A16}$$

Here the bilinear term has been written in terms of the scatter matrix

$$\boldsymbol{S}_g \equiv \sum_{c=1}^{N_C} (\boldsymbol{g}_c - \langle \boldsymbol{g}_c \rangle_c) \cdot (\boldsymbol{g}_c - \langle \boldsymbol{g}_c \rangle_c)^T. \tag{A17}$$

Maximizing V_1^g under the constraint that $\hat{\boldsymbol{g}}_1^T \cdot \hat{\boldsymbol{g}}_1 = 1$ is equivalent to finding the eigenvector of $\boldsymbol{S}_g$ associated with the largest eigenvalue. For normalized data, $\boldsymbol{S}_g$ coincides with the gene gene correlation matrix

$$\boldsymbol{\mathcal{C}}_g = \boldsymbol{E}_C^T \cdot \boldsymbol{E}_C \quad \text{with} \quad \mathcal{C}_g^{gg'} = \hat{\boldsymbol{c}}_g^T \cdot \hat{\boldsymbol{c}}_{g'}. \tag{A18}$$

Conversely, if the expression data are viewed as a collection of condition vectors, the vector $\hat{\boldsymbol{c}}_1$ that maximizes the variance of the components $g_1^{(g)} = \boldsymbol{c}_g^T \cdot \hat{\boldsymbol{c}}_1$, is the eigenvector associated with the largest eigenvalue of the scatter matrix

$$\boldsymbol{S}_c \equiv \sum_{g=1}^{N_G} (\boldsymbol{c}_g - \langle \boldsymbol{c}_g \rangle_g)(\boldsymbol{c}_g - \langle \boldsymbol{c}_g \rangle_g)^T. \tag{A19}$$

For normalized data, $\boldsymbol{S}_c$ equals the condition condition correlation matrix

$$\boldsymbol{\mathcal{C}}_c = \boldsymbol{E}_G \cdot \boldsymbol{E}_G^T \quad \text{with} \quad \mathcal{C}_c^{cc'} = \hat{\boldsymbol{g}}_c^T \cdot \hat{\boldsymbol{g}}_{c'}. \tag{A20}$$

Note, however, that since $\boldsymbol{E}_G \neq \boldsymbol{E}_C$, the matrices $\boldsymbol{E}_C \cdot \boldsymbol{E}_C^T$ and $\boldsymbol{E}_G^T \cdot \boldsymbol{E}_G$ are different from $\boldsymbol{\mathcal{C}}_c$ and $\boldsymbol{\mathcal{C}}_g$, and do not represent correlation matrices.

2. The variance of a product of random variables

By definition, the mean of the product of two *independent* random variables a and b is the product of their mean values, i.e.,

$$\langle ab \rangle = \langle a \rangle \langle b \rangle. \tag{A21}$$

Since the expression for the variance of the product ab in Eq. (19) may be somewhat less obvious, we give its derivation here. From the definition of the variance

$$V(a) \equiv \langle (a - \langle a \rangle)^2 \rangle = \langle a^2 \rangle - \langle a \rangle^2, \tag{A22}$$

we obtain

$$V(a)V(b) = (\langle a^2 \rangle - \langle a \rangle^2)(\langle b^2 \rangle - \langle b \rangle^2) \tag{A23}$$

$$= \langle a^2 \rangle \langle b^2 \rangle - \langle a \rangle^2 \langle b^2 \rangle - \langle a^2 \rangle \langle b \rangle^2 + \langle a \rangle^2 \langle b \rangle^2. \tag{A24}$$

Then using Eqs. (A21)–(A24) it follows that

$$V(ab) = \langle a^2 b^2 \rangle - \langle ab \rangle^2 \tag{A25}$$

$$= \langle a^2 \rangle \langle b^2 \rangle - \langle a \rangle^2 \langle b \rangle^2 \tag{A26}$$

$$= V(a)V(b) + \langle a \rangle^2 \langle b^2 \rangle + \langle a^2 \rangle \langle b \rangle^2 - 2\langle a \rangle^2 \langle b \rangle^2 \tag{A27}$$

$$= V(a)V(b) + (\langle a^2 \rangle - \langle a \rangle^2)\langle b \rangle^2 + (\langle b^2 \rangle - \langle b \rangle^2)\langle a \rangle^2 \tag{A28}$$

$$= V(a)V(b) + V(a)\langle b \rangle^2 + V(b)\langle a \rangle^2. \tag{A29}$$

3. Accurate treatment of the noise propagation

In order to simplify our presentation of the propagation of the noise under the iterative scheme in Eq. (17) we used the approximate recursive equation in Eq. (25) to derive the fixed point noise-to-signal ratio in Eq. (26). Here we give an accurate treatment that is valid even if $N_G^{(m)} \gg 1$ is not satisfied.

First, note that if the iterative scheme converges, then for $n \to \infty$ we have $v_G^{(n)} = v_G^{(n-1)} = v_G^{(*)}$ and $\tilde{v}_G^{(n)} = \tilde{v}_G^{(n-1)} = \tilde{v}_G^{(*)}$. In this case we can write two fixed-point equations

$$\tilde{v}_G^{(*)} \left(1 - \frac{\Delta N_G v_{\mathcal{C}}}{(N_G^{(m)})^2} \right) = \frac{v_{\mathcal{C}}}{N_G^{(m)}} (v_G^{(*)} + 1), \tag{A30}$$

$$v_G^{(*)} \left(1 - \frac{1}{N_G^{(m)}} \right) = \tilde{v}_G^{(*)}. \tag{A31}$$

Solving Eqs. (A30) and (A31) for $v_G^{(*)}$ we get

$$v_G^{(*)} = \left[\left(1 - \frac{1}{N_G^{(m)}} \right) \left(\frac{N_G^{(m)}}{v_{\mathcal{C}}} - \frac{\Delta N_G}{N_G^{(m)}} \right) - 1 \right]^{-1}$$

$$\simeq \left(\frac{N_G^{(m)}}{v_{\mathcal{C}}} - \frac{N_G}{N_G^{(m)}} \right)^{-1}. \tag{A32}$$

Here, the approximation on the right-hand side neglects the $1/N_G^{(m)}$ term and yields exactly the same result as obtained from the simplified iterative scheme in Eq. (25) that ignores the difference between $v_G^{(n)}$ and $\tilde{v}_G^{(n)}$.

Interestingly, a necessary condition for convergence can be derived also without any approximation directly from Eqs. (23) and (24). To this end note that Eq. (24) implies trivially that $v_G^{(n)} \geqslant \tilde{v}_G^{(n)}$. Then it follows that

$$v_G^{(n)} \leqslant \frac{N_G v_C + N_G^{(m)}}{(N_G^{(m)})^2} v_G^{(n-1)} + \frac{v_C}{N_G^{(m)}}. \tag{A33}$$

Thus if

$$v_C \leqslant v_C^{crit} \equiv \frac{N_G^{(m)}(N_G^{(m)}-1)}{N_G} \tag{A34}$$

the noise-to-signal ratio $v_G^{(n)}$ converges to a finite value.

[1] M. Schena, D. Shalon, R.W. Davis, and P.O. Brown, Science **20**, 467 (1995).
[2] J.L. DeRisi, V.R. Iyer, and P.O. Brown, Science **24**, 680 (1997).
[3] E. Lander, Nat. Genet. **21**, 3 (1999). (See also other papers in this issue.)
[4] A. Schulze and J. Downward, Nat. Cell Biol. **3**, E190 (2001).
[5] A comprehensive database for expression data from various organisms has been established by G. Sherlock *et al.*, Nucleic Acids Res. **29**, 152 (2001); see also http://genome-www.stanford.edu/microarray
[6] A complete list of the references used to compile the yeast expression data studied in this paper can be found at http://www.weizmann.ac.il/~jan/NG/MainFrames.html
[7] M.B. Eisen, P.T. Spellman, P.O. Brown, and D. Botstein, Proc. Natl. Acad. Sci. U.S.A. **95**, 14 863 (1998).
[8] P.T. Spellman *et al.*, Mol. Biol. Cell **9**, 3273 (1998).
[9] U. Alon *et al.*, Proc. Natl. Acad. Sci. U.S.A. **96**, 6745 (1999).
[10] S. Tavazoie, J.D. Hughes, M.J. Campbell, R.J. Cho, G.M. Church, Nat. Genet. **22**, 281 (1999).
[11] C.M. Perou *et al.*, Proc. Natl. Acad. Sci. U.S.A. **96**, 9212 (1999).
[12] M. Bittner *et al.*, Nature (London) **3**, 536 (2000).
[13] U. Scherf *et al.*, Nat. Genet. **24**, 236 (2000).
[14] J.E. Staunton *et al.*, Proc. Natl. Acad. Sci. U.S.A. **98**, 10 787 (2001).
[15] A. Brazma and J. Vilo, FEBS Lett. **480**, 17 (2000).
[16] R.B. Altman and S. Raychaudhuri, Curr. Opin. Struct. Biol. **11**, 340 (2001).
[17] N.S. Holter *et al.*, Proc. Natl. Acad. Sci. U.S.A. **97**, 8409 (2000).
[18] O. Alter, P.O. Brown, and D. Botsein, Proc. Natl. Acad. Sci. U.S.A. **97**, 10 101 (2000).
[19] P. Tamayo *et al.*, Proc. Natl. Acad. Sci. U.S.A. **96**, 2907 (1999).
[20] M. Bittner, P. Meltzer, and J. Trent, Nat. Genet. **22**, 213 (1999).
[21] Y. Cheng and G.M. Church, Proc. Int. Conf. Intell. Syst. Mol. Biol. **8**, 93 (2000).
[22] G. Getz, E. Levine, and E. Domany, Proc. Natl. Acad. Sci. U.S.A. **97**, 12 079 (2000).
[23] T. Hastie *et al.*, Genome Biol. **1(2)**, research 0003.1-0003.21 (2000).
[24] L. Lazzeroni and A. Owen, Statistica Sinica **12**, 61 (2002).
[25] J. Ihmels, G. Friedlander, S. Bergmann, O. Sarig, Y. Ziv, and N. Barkai, Nat. Genet. **31**, 370 (2002).
[26] T.G. Kolda and D.P. O'Leary, ACM Trans. Inf. Syst. **16**, 322 (1998).
[27] D.D. Lee and H.S. Seung, Nature (London) **401**, 788 (1999).
[28] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification*, 2nd ed. (Wiley, New York, 2001).
[29] G.H. Golub and C.F. Van Loan, *Matrix Computation* (Johns Hopkins University Press, Baltimore, 1996).
[30] S. Bergmann, J. Ihmels, and N. Barkai (unpublished).