

Iterative Signature Algorithm for the Analysis of Large-Scale Gene Expression Data

By S. Bergmann, J. Ihmels, N. Barkai

Reasoning

- Both clustering and Singular Value Decomposition(SVD) are useful tools but have draw backs
- Analysis based on expression over all conditions, but only a small number are important
- SVD is sensitive to noise in the data

Goal

- Provide a method for large-scale gene expression analysis without the drawbacks of clustering and SVD
- Does not require any prior knowledge beyond the expression data

Transcription Modules

- Defines the concept of Transcription Module(TM)
- A TM is a bicluster with some additional constraints
- TM may be associated with particular cellular function
- Ideally there is one transcription factor per TM, but since different transcription factors can regulate the same gene distinct TM can share common genes and conditions

Transcription Modules

- Each module M contains a subset of genes and conditions
- Maximize similarity of the gene expression when compared over the conditions
- Similarity is determined by two threshold parameters T_G, T_C
- Mathematical definition of TM:

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

Matrix Normalization

- What are E_G^{cg} and E_c^{cg} ?
- E_G and E_c are normalized matrices, derived from the original expression matrix E
- g, c are a gene and a condition
- More on normalization, E can be written as:

$$E = \begin{pmatrix} \mathbf{g}_1^T \\ \mathbf{g}_2^T \\ \vdots \\ \mathbf{g}_{N_G}^T \end{pmatrix}.$$

$$E = \mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_{N_C}.$$

Matrix Normalization

- Rows are conditions and columns are genes, row vector \mathbf{g} contains the expression profile for that condition, and vice versa
- The normalized matrices are defined by:

$$E_G \equiv \begin{pmatrix} \hat{\mathbf{g}}_1^T \\ \hat{\mathbf{g}}_2^T \\ \vdots \\ \hat{\mathbf{g}}_{N_C}^T \end{pmatrix}$$

$$E_C \equiv (\hat{\mathbf{c}}_1, \hat{\mathbf{c}}_2, \dots, \hat{\mathbf{c}}_{N_G}).$$

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

Matrix Normalization

- With Matrices E_C , E_G compute the projection of a gene set g_m or condition set c_m onto the normalized matrices

$$c_m^{proj} \equiv E_G \cdot g_m = \begin{pmatrix} \hat{g}_1^T g_m \\ \hat{g}_2^T g_m \\ \vdots \\ \hat{g}_{N_C}^T g_m \end{pmatrix}$$

$$g_m^{proj} \equiv E_C^T \cdot c_m = \begin{pmatrix} \hat{c}_1^T c_m \\ \hat{c}_2^T c_m \\ \vdots \\ \hat{c}_{N_G}^T c_m \end{pmatrix} .$$

Redefining TM

- With this information a revised definition of TM can be created, with $f_t(\mathbf{x})$ the threshold function

$$\Xi(t_C, t_G): \begin{cases} \mathbf{c}_m = f_{t_C}(\mathbf{c}_m^{proj}), \\ \mathbf{g}_m = f_{t_G}(\mathbf{g}_m^{proj}), \end{cases}$$

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

The Threshold Function

- The concept of a threshold is very important as we will see later
- The function is made up of two parts a weight $w(x_i)$ and a step function $\Theta(x_i - t)$
- $w(x)$ in this paper is either the $\text{sign}(x)$, or x , but could have other values

More on $f_t(\mathbf{x})$

- $x_i^{\sim} = [x_i - \text{mean}(\mathbf{x})]/\text{stddev}(\mathbf{x})$
- The step function sets to zero all elements that do not exceed the mean by at least $t \cdot \text{stddev}(\mathbf{x})$
- Can capture down regulation by using the absolute value x_i
- Applying the threshold function to C_m^{proj} results in a non zero component in c_m if the corresponding normalized gene profile is sufficiently aligned with g_m

Iterative Signature Algorithm

- Now with all of this information we can define the iterative signature algorithm
- A randomly chosen gene set (or condition set) called g_i^0 is needed to begin, then the following equations can be used.

$$c^{(n+1)} = f_{t_C}(E_G \cdot g^{(n)}),$$

$$g^{(n+1)} = f_{t_G}(E_C^T \cdot c^{(n+1)}).$$

Iterative Signature Algorithm

- Using the initial gene set C_i^1 can be computed, then G_i^1 , and so on
- After some number of iterations the g_i^n values quickly converge to a fixed point $g^{(*)}$, that will satisfy the equation below for some given error

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

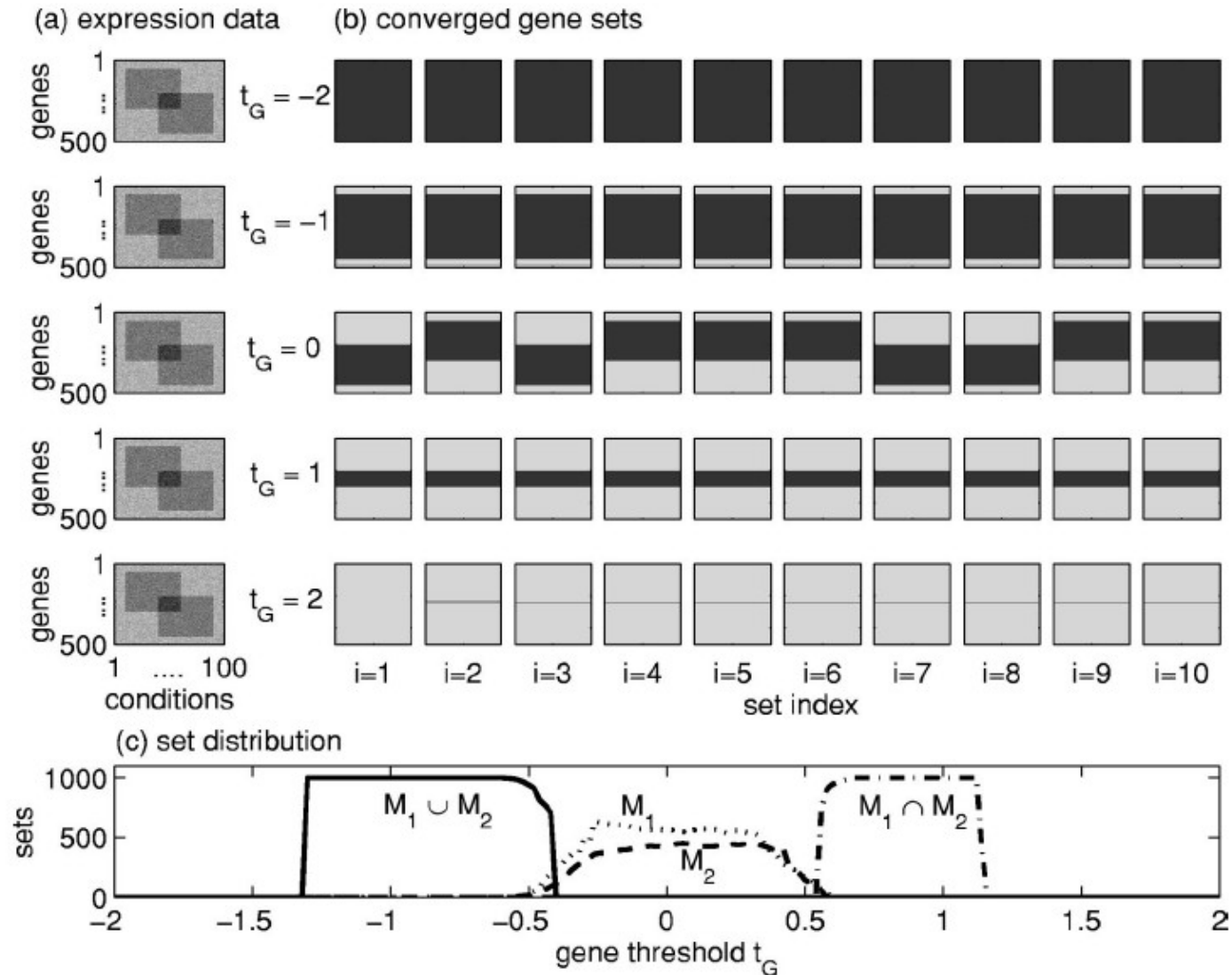
Iterative Signature Algorithm

- So a general application of ISA would
- 1)Generate large sample of “seeds” or g_i^0 s
- 2)Find the fixed points corresponding to each seed using iterations
- 3)Collect distinct fixed points in order to decompose into modules

Importance of Thresholds

- Lower thresholds will produce larger modules whose coregulation is loose, and higher thresholds give smaller more closely coregulated modules
- Choosing the threshold low enough will create a “super module” that contains all of the conditions and genes. Choosing the threshold higher will produce the intersection of modules, and choosing the threshold high enough will give you empty modules

Importance of Thresholds



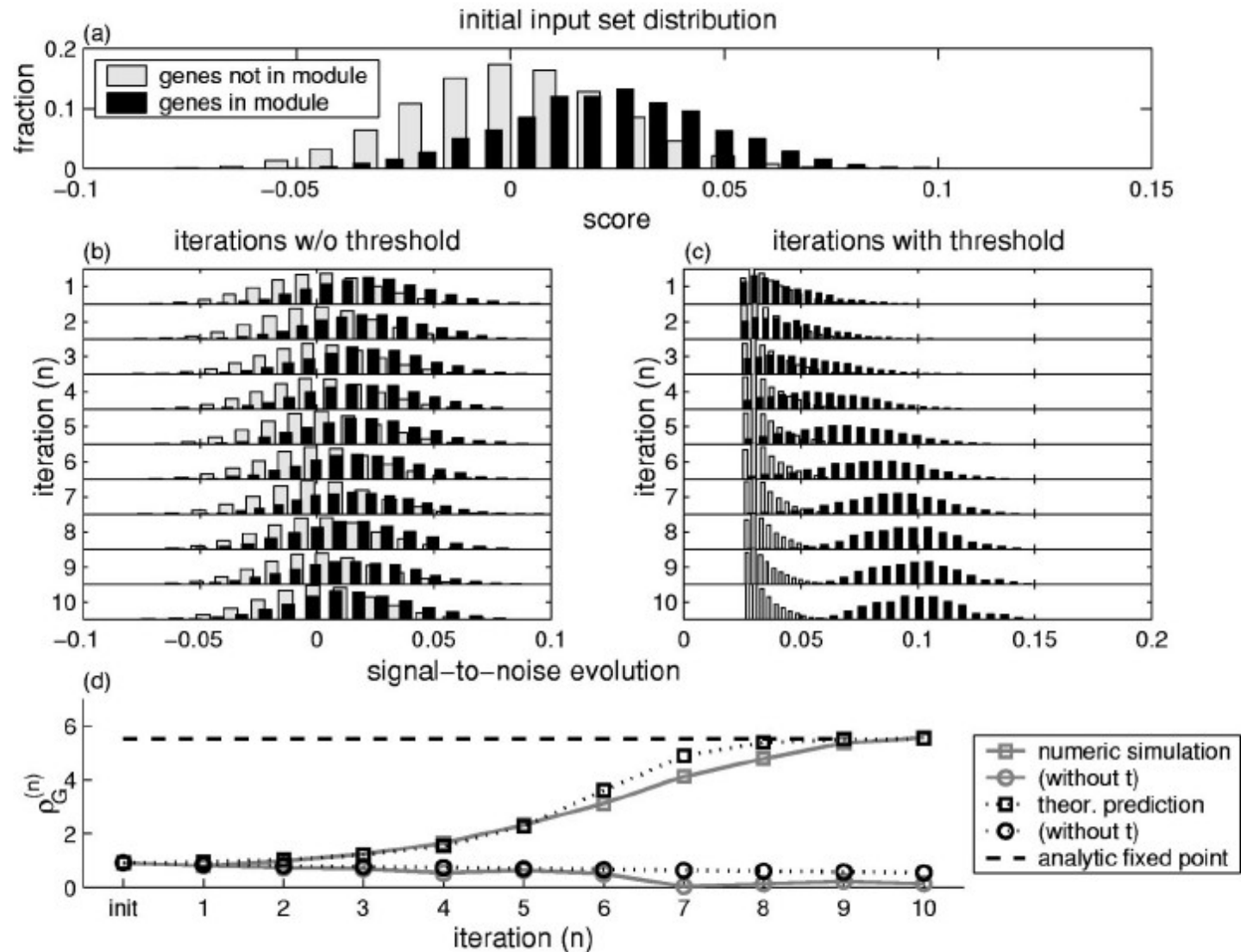
Importance of Thresholds

- Thresholds can greatly reduce the noise in expression data, more on this later.
- The paper compares a simple iterative method that is similar to ISA but without threshold much like SVD
- Also tested how ISA handles noise at with different number of modules

Importance of Thresholds

- Created tests sets and artificially added noise
- Example: $t=2$ removes 98% of genes outside the module, while only removing 16% inside the module
- The result of these tests show that ISA in combination with the threshold handles noise well compared with no threshold

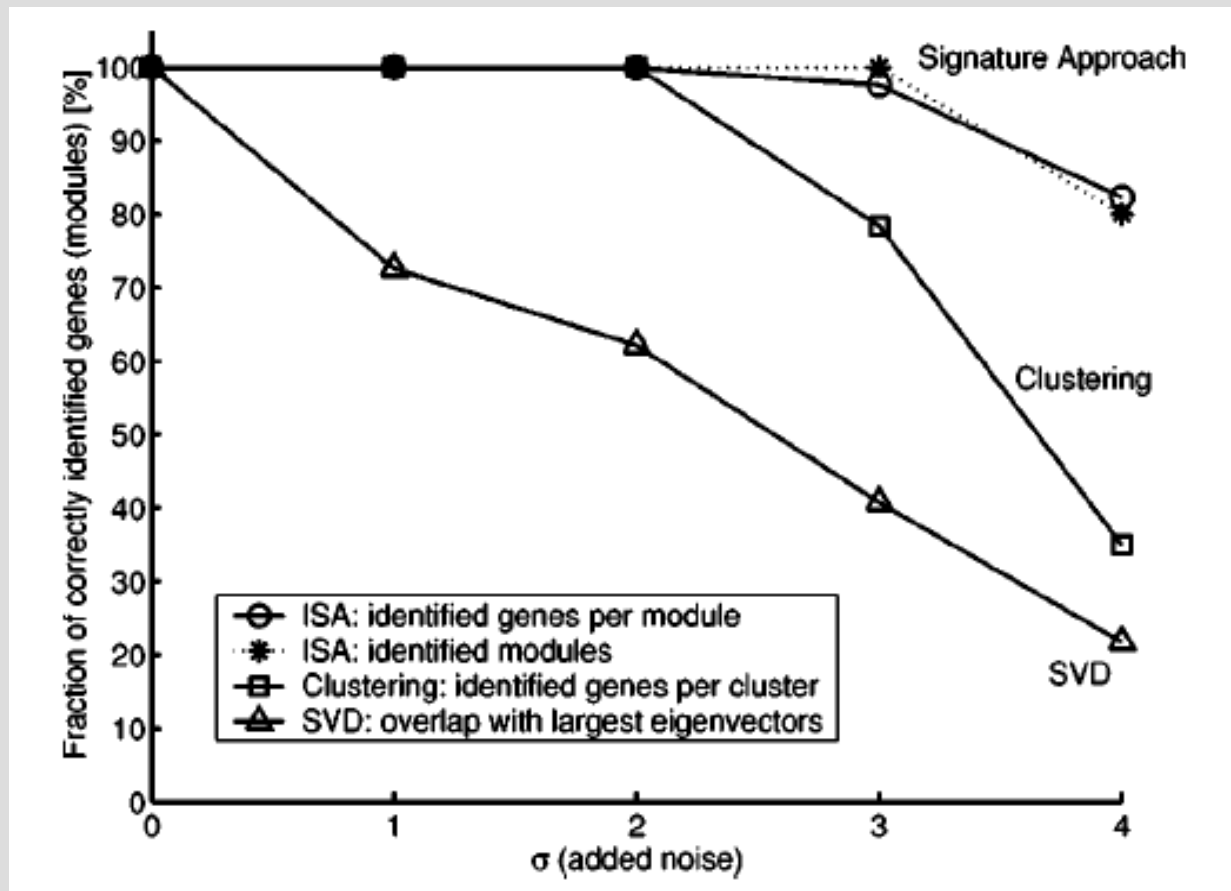
Importance of Thresholds



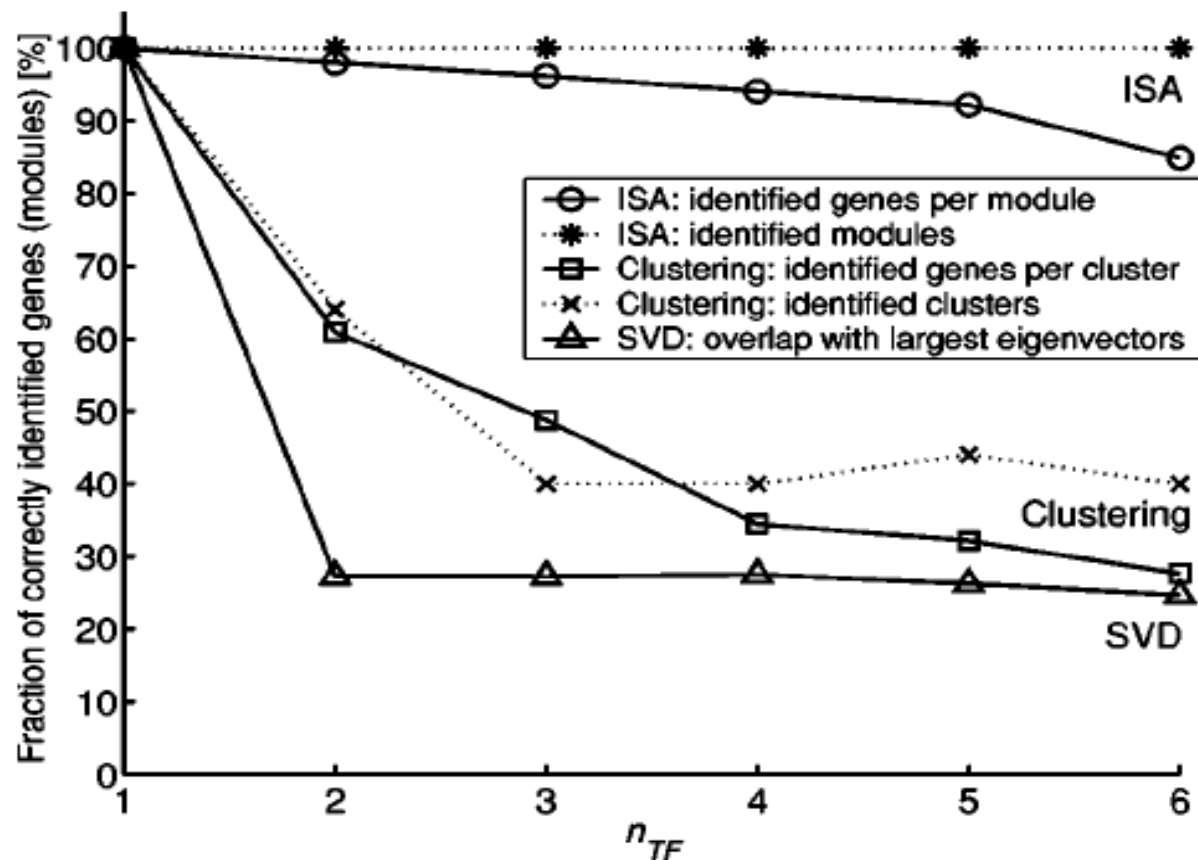
Comparing ISA to other methods

- Compared ISA to SVD and Hierarchical Clustering
- Created experimental data and artificially added noise, data had 25 modules
- Superimposed noise from a random distribution on to the expression set
- Even as noise became large ISA performed considerably better than SVD or Hierarchical Clustering

Comparing ISA to other methods



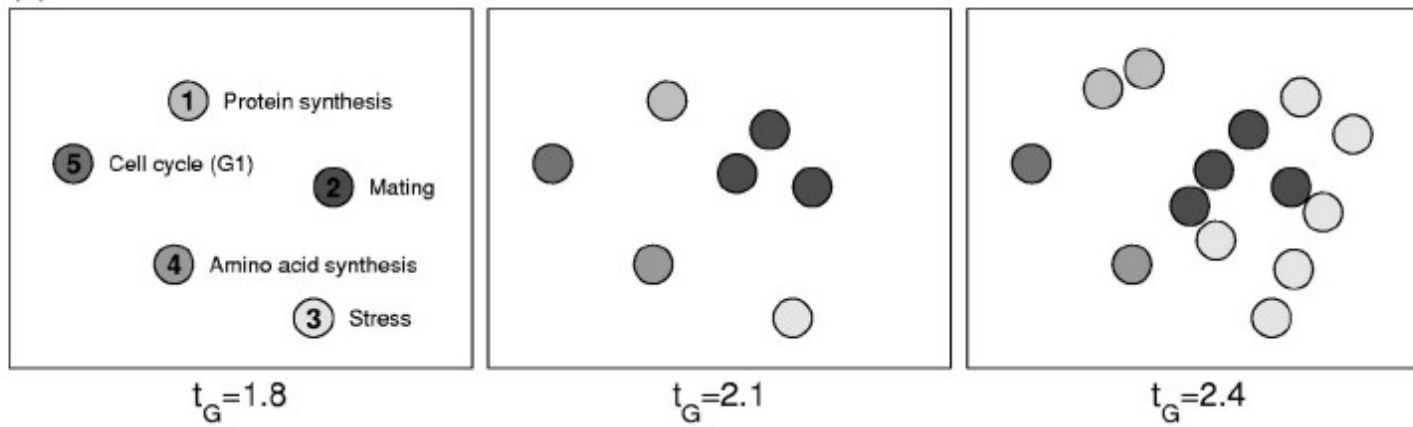
Comparing ISA to other methods



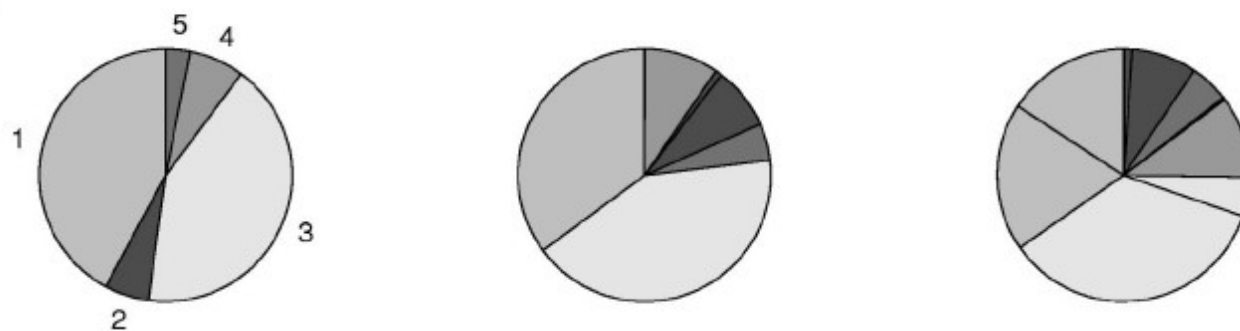
ISA and Yeast Expression Data

- Used a diverse set of 1000 DNA-chip experiments that were obtained by different groups
- Tried 20,000 randomly created initial sets, and tried values of $T_G = 1.8, 1.9 \dots 4.0$, T_C set to 2.0
- With $T_G = 1.8$ found 5 known modules of Yeast, and at higher values found TM that had not been previously identified

(a)



(b)



Conclusion

- ISA clearly outperforms both hierarchical clustering and SVD when dealing with noise in data sets
- The use of thresholds in ISA is a very powerful tool enabling structures at different levels to become clear
- References: all figures in this presentation were taken from the paper.
- Any Questions?