Reverse-Engineering Transcriptional Modules from Gene Expression Data

Tom Michoel,[a,b] Riet De Smet,[c] Anagha Joshi,[a,b] Kathleen Marchal,[c,d] and Yves Van de Peer[a,b]

[a]*Department of Plant Systems Biology, VIB, Gent, Belgium*

[b]*Department of Molecular Genetics, UGent, Gent, Belgium*

[c]*CMPG, Department Microbial and Molecular Systems, KULeuven, Leuven, Belgium*

[d]*ESAT-SCD, KULeuven, Leuven, Belgium*

"Module networks" are a framework to learn gene regulatory networks from expression data using a probabilistic model in which coregulated genes share the same parameters and conditional distributions. We present a method to infer ensembles of such networks and an averaging procedure to extract the statistically most significant modules and their regulators. We show that the inferred probabilistic models extend beyond the dataset used to learn the models.

Key words: **reverse engineering; transcriptional modules; probabilistic graphical models; ensemble methods**

Introduction

Methods for reverse engineering transcription regulatory networks from high-throughput microarray data come in many different flavors.[1–5] An important class of methods are those that not only seek to identify the topological wiring of the network,[6–8] but also attempt to infer a model of the biological system that explains the observed gene expression patterns and generates testable hypotheses. Such models can take the form of probabilistic graphical models,[1,9,10] simplified kinetic equation models,[11] or biophysical models.[5] A common property of all modeling approaches is that the number of parameters is much larger than the number of experimental data points available to define them. Dimensionality reduction is usually achieved by a coarse-graining step, which collapses individual genes into clusters of coexpressed genes or modules, where all genes in a cluster share the same model parameters.[12]

This conceptual simplification has as a drawback that inferred interactions are influenced by the module quality. Moreover, it is hard to translate the concept of a biological module in a strict mathematical definition. When searching for modules, often many local optima exist with partially overlapping modules differing from each other in a few genes or conditions. Therefore, in our approach we exploit the "fuzzy" property of a module to increase the reliability of the predicted interactions. Instead of reporting only one cluster solution (local optimum), we use a stochastic approach to generate many partially redundant cluster solutions (bootstrapping) and generate an ensemble solution by averaging over multiple high-scoring models.

Crucial for the success of the ensemble approach is the availability of an efficient method for sampling a large number of different models covering the whole search space of possible models.[13] Therefore, we use the deterministic

Address for correspondence: Tom Michoel, Department of Plant Systems Biology, VIB, Ghent University, Technologiepark 927, B-9052 Gent, Belgium. Voice: +32 9 33 13 758. tom.michoel@psb.ugent.be

The Challenges of Systems Biology: Ann. N.Y. Acad. Sci. 1158: 36–43 (2009).
doi: 10.1111/j.1749-6632.2008.03943.x

approach of Michoel and colleagues[14] as a basis for our sampling method. In this approach gene and condition clustering are decoupled from learning the regulatory programs compared to the original method of Segal and coworkers,[9] which optimizes both simultaneously. This allows for a higher efficiency while maintaining an equal performance rate.[14] Several extensions of the method of Segal and coworkers[9] infer transcriptional modules from gene expression and genome-wide location analysis.[15–18] It is entirely feasible to develop ensemble strategies for these methods as we did for Segal and coworkers.[9] This constitutes an interesting line of future research. In this paper we give a brief description of the algorithm and highlight some results for an expression compendium for *Escherichia coli*.[8] A complete analysis of this dataset and comparison of our approach to the mutual information based CLR method[8] is given in Michoel and colleagues.[19] A detailed comparison between the ensemble approach and the direct-optimization based method of Segal and coworkers[9] on *Saccharomyces cerevisiae* data is given in Joshi and colleagues.[20]

Results and Discussion

The Algorithm

The algorithm takes as input a gene-expression dataset and a list of candidate regulators. It gives as output a large number of probabilistic models consisting of a set of gene clusters, with—for each gene cluster—a partition of the experiments and a probabilistic regulatory program explaining the observed experiment partitions in terms of the expression of a small number of regulators. The number of gene and experiment clusters is determined automatically and can vary from one solution to the next. Using overrepresentation in the ensemble, the most probable interactions can be identified. The regulatory programs can be validated on new experimental data and generate testable hypotheses about conditional regulation of the inferred gene clusters.

The first step of the algorithm consists of generating an ensemble of gene clusters with experiment partitions. A Gibbs sampling method iteratively updates the assignment of each gene given the current gene and experiment clusters, and the assignment of each experiment in each gene cluster given the current assignment of all other experiments in that gene cluster, iterating until a stationary state is reached. Details about the Gibbs sampler algorithm and a complete analysis of its convergence properties can be found in Joshi and colleagues.[21] Briefly, in a single run, the Gibbs sampler reaches a local optimum in the direction of genes, but covers the whole search space in the direction of experiments. This implies that for a given cluster of co-expressed genes there are multiple equiprobable ways of partitioning the experiments. To also sample from the whole search space in the direction of genes, we perform several independent Gibbs sampler runs with random restarts. In Joshi and colleagues,[21] it is shown that each of the local optima in the direction of genes is (approximately) of the same height, and therefore equally important, and that a relatively small number of local optima is sufficient to cover the whole search space. (Typically two distinct sets of 10 local optima for a dataset of 1,000 genes agree for 95% on the probability for each pair of genes to be clustered together, see Ref. 21 for details.) A graphical cartoon representation of the Gibbs sampling procedure is given in Figure 1.

In the second step of the algorithm, regulatory programs are learned for each experiment partition for each gene cluster. This is achieved by linking the sets in the experiment partition hierarchically in a decision tree. For each split in this tree, a candidate regulator is found whose expression is significantly different on both sides of the split, as measured by an entropy measure. Details can be found in Ref. 14.

A gene cluster with a regulatory program is called a *transcriptional module*, and a partition of all genes into clusters, each with a regulatory program, a *module network*. A sample module is shown in Figure 2. To each module network

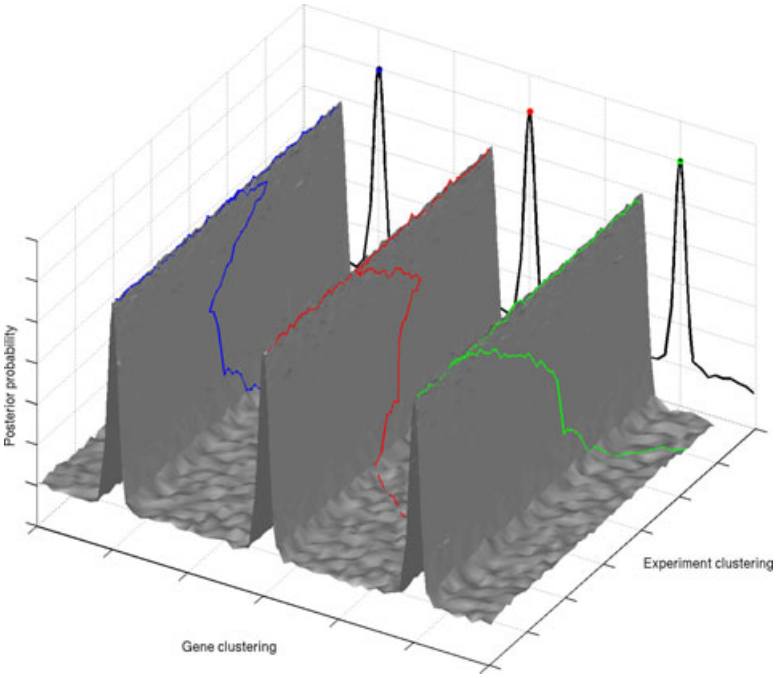

FIGURE 1. Graphical cartoon representation of the Gibbs sampling procedure for two-way clustering of genes and conditions.[21] Each colored curve represents a random restart converging on a distinct local optimum in the direction of genes. In the direction of experiments, the whole search space can be covered in one run.

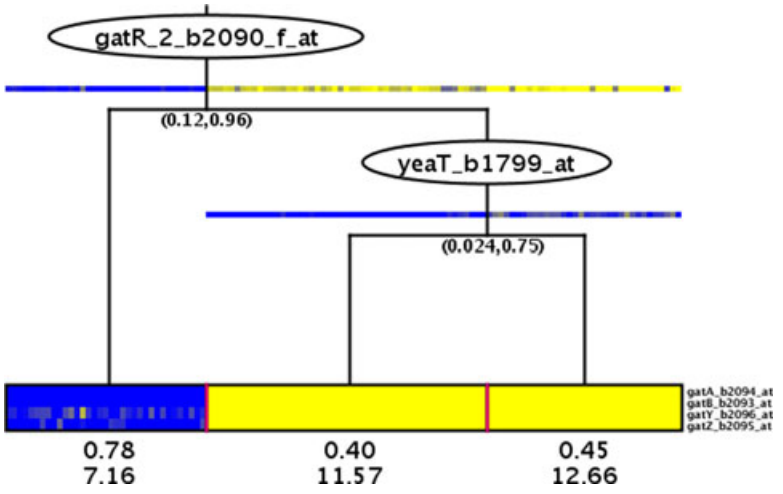

FIGURE 2. Example module with a regulatory program. The numbers under the experiment clusters are the standard deviations and mean expression values of the data in the cluster. The numbers under each tree node are the normalized Bayesian scores gained in the Gibbs sampler by making this data split[21] and a percentage quality score for the assignment of this regulator.[14] The colored bars on each tree node are the expression levels of the regulators with experiments sorted in the same order as in the experiment clusters.

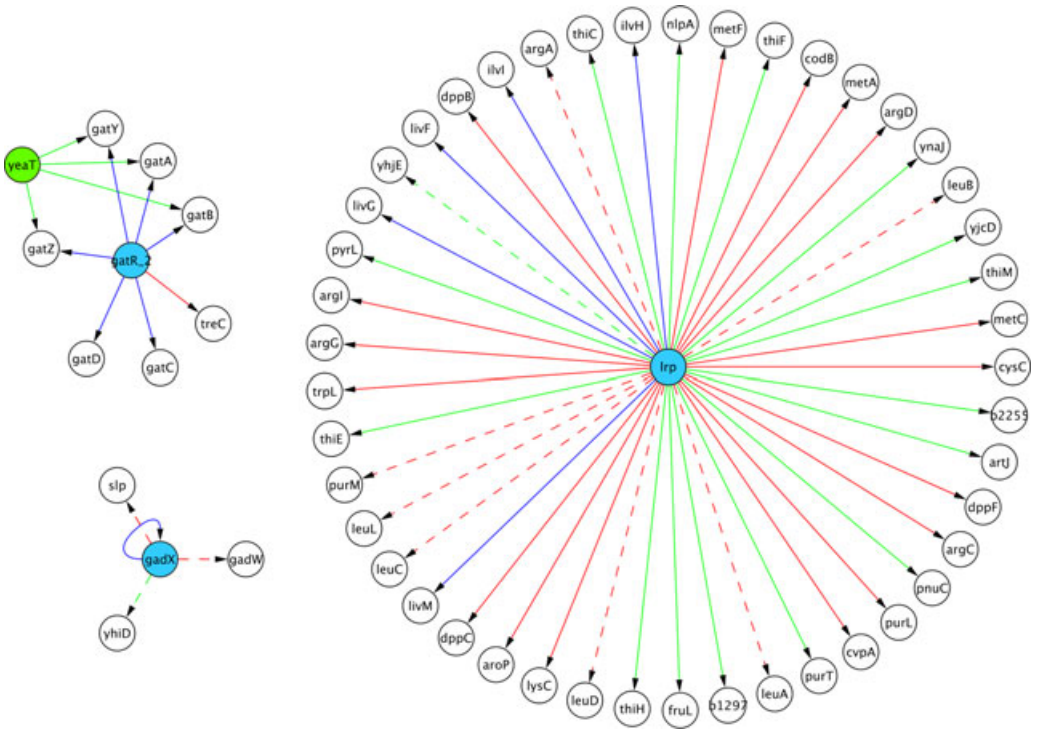

FIGURE 3. Sample interactions discussed in the text. Blue edges are present in RegulonDB, green edges connect to genes not in RegulonDB, and dashed edges are new predictions validated by literature.

corresponds a probabilistic model defined by a probability density function $p(x_1, \ldots, x_N)$ with x_i the continuous valued expression level of gene i (see Methods). The value of $p(x_1, \ldots, x_N)$ measures how well the model explains a particular experiment with expression levels for all genes.

Ensemble Averaging

We have applied the algorithm on a compendium of Affymetrix microarrays for *E. coli*.[8] The compendium contains 445 expression profiles for 4,345 genes under 189 different stress conditions and genetic perturbations. The results are validated by comparison with existing knowledge of transcriptional interactions in RegulonDB[22] and EcoCyc.[23] A complete analysis of the inferred module networks and their biological significance is beyond the scope of this paper and will be given elsewhere.[19] Here we highlight a few examples of inferred interactions that are representative for the working mechanism of the ensemble approach (see Figure 3).

A first example is given by the *gatYZABCD* operon (involved in galactitol metabolism). As expected, the genes in this operon consistently cluster together. GatR_2, a subunit of the GatR transcriptional repressor, is consistently assigned as a candidate regulator. It is known that GatR negatively controls transcription initiation of the *gatYZABCD* operon, which is the only known target of GatR according to RegulonDB. The many local optima, generated by the Gibbs sampling approach resulted in 81 additional genes, which were clustered at least once with one of the *gatYZABCD* genes. Except for one potential new target *treC,* which was found repeatedly in combination with *gatC* and *gatD*, the interactions between the remaining 80 genes and GatR were not sufficiently significant to be retained in the final solution. This illustrates how the ensemble approach guarantees an effective filtering of false positives while retaining the true positives. This example also

illustrates how combinatorial regulation can be detected. For *gatY, gatZ, gatA,* and *gatB*, a second regulator, YeaT (a predicted regulator involved in malate metabolism), with an anticorrelated expression pattern seemed to play a role in a subset of the conditions (see Fig. 2).

Four interactions could be inferred for GadX (a transcriptional activator involved in acid resistance): GadX itself, which makes sense as it is known to be autoregulated,[24] and three novel targets *slp*, *gadW,* and *yhiD*. This finding is supported by literature, as the expression of both *slp*[25] and *yhiD*[25] seemed to be affected in a *gadX* mutant, while *gadX* and *gadW* seem to tightly control each other's expression.[26]

For Lrp (a regulator involved in the high-affinity transport of branched-chain amino acids and a mediator of the leucine response), 44 interactions could be found, of which five could be confirmed by RegulonDB (*ilvI*, *ilvH*, *livG*, *livM,* and *livN*) and 39 were new. Among the predicted interactions with Lrp we found the *leuLABCD* genes. According to literature, the Lrp-dependent regulation of the *leuLABCD* operon is only indirect.[27] However, without additional data, no distinction can be made between direct and indirect effects of a regulator if both give rise to a correlated expression level with the targets. Four of our predicted targets (*purM*, *argA*, *yhjE,* and *aroP*) were tested by ChIP analysis,[8] and three (*purM*, *argA,* and *yhjE*) were confirmed. *YhjE* was also found to be differentially expressed in a microarray analysis of *lrp* mutants.[28] For the remaining genes, no clear indication for their regulation by Lrp could be found. However, it should be noted that Lrp not only acts as a regulator, binding specific DNA sequences, but also functions as a DNA-organizing protein, extending its global role in regulation.[29,30] A large regulon of Lrp, as was detected by our method, thus is in line with this more global role of Lrp.

Model Evaluation

One of the purposes of model-based reverse-engineering methods is to infer a model of the system that extends beyond the dataset used to learn the model. As such these methods can form the basis for developing methods that use new data to refine and extend a partially validated model, rather than inferring a completely new network model each time a dataset is updated. Validation of a model is done by comparing the distribution of $\frac{1}{N}\log p(x_1, \ldots, x_N)$ (see Methods) for new data with the distribution for data used to learn the model. In general, higher values of $\log p$ mean better explanation of the data by the model. The results of this comparison of 75 experiments recently added to the M^{3D} database[8] with the 189 original experiments, for 100 module network models selected at random from the ensemble, are shown in Figure 4. The overlap between the distributions shows that the probabilistic models indeed generalize to unseen data. For comparison we also perform a randomization test by permuting the gene indices of the data matrix. As expected, the evaluation of the models on randomized data is several orders of magnitude smaller than on real data, and in fact around 15% of the randomized experiments have a value $\log\ p = -\infty$, that is, zero likelihood.

Methods

Datasets

We downloaded a compendium of expression profiles for *E. coli*[8] and a list of 328 candidate regulators from http://gardnerlab.bu.edu/netinfer_plos_2007/. We averaged over replicate experiments to obtain a data matrix of 4,345 genes and 189 experiments. Additional data for 4,292 genes and 75 experiments for validation of the probabilistic models were downloaded from http://m3d.bu.edu/norm/.

Probabilistic Model Evaluation

The probabilistic model introduced by Segal and coworkers[9] associates to each gene i a continuous valued random variable X_i measuring

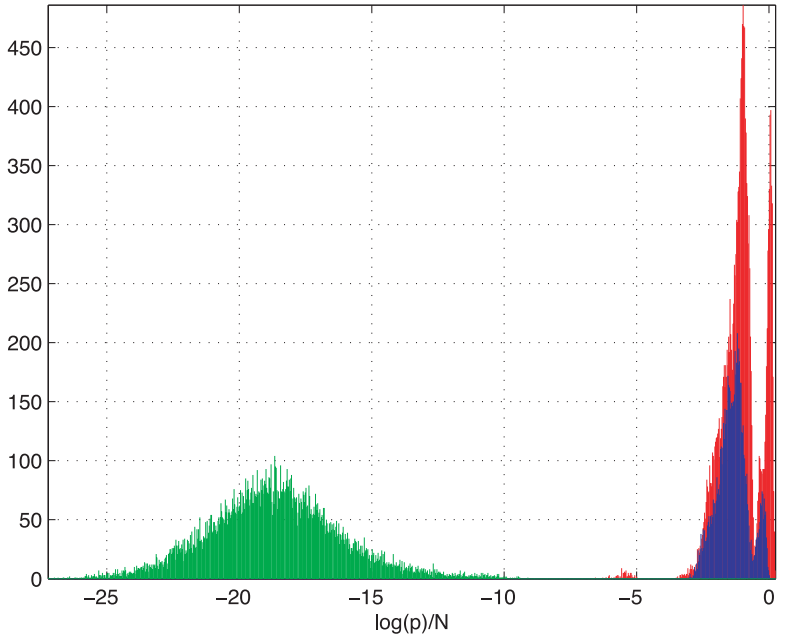

FIGURE 4. Histogram of $\frac{1}{N}\log p(x_1, \ldots, x_N)$ for 100 models in the ensemble, for the original data (red), new data (blue), and randomized data (green, around 15% zero likelihood values [log $p = -\infty$] not shown).

the gene's expression level. The distribution of X_i depends on the expression level of a set of parent genes chosen from a list of candidate regulators. Genes regulated by the same parents form a cluster and share the same model parameters. The joint probability distribution for the expression levels of all genes decomposes as a product of conditional distributions,

$$p(x_1, \ldots, x_N) = \prod_{k=1}^{K} \prod_{i \in A_k} p_k\left(x_i \mid \{x_r : r \in \mathcal{R}_k\}\right), \tag{1}$$

where $\{\mathcal{A}_k, k = 1, \ldots, K\}$ is the set of clusters (*i.e.*, a partition of the gene set $\{1, \ldots, N\}$) and $\mathcal{R}_k$ is the set of regulators for cluster k. The distribution (Eq. (1)) is normalized if the network from parents to children is acyclic. The conditional distribution p_k of the expression level of the genes in cluster k is a normal distribution with parameters determined by the expression levels of the parents $\mathcal{R}_k$:

$$p_k(x \mid \{x_r : r \in \mathcal{R}_k\}) = p(x \mid \mu_\ell, \tau_\ell). \tag{2}$$

The parameters μ_ℓ and τ_ℓ are determined by arranging the parents in a decision tree. The tests on the nodes of the decision tree are of the form '$x_r > z$?' for some threshold value z, where x_r is the expression value of the parent r associated to the node. The leaves ℓ of the decision tree are the sets of an experiment partition for cluster k, and μ_ℓ and τ_ℓ are the mean, respectively precision of the expression levels of the genes in the cluster in this subset of experiments. See Ref. 14 for more details.

To evaluate a model of the form (Eq. (1)), we are only interested in genes for which the model makes actual predictions, namely, genes belonging to clusters with a regulation tree. If the clustering procedure does not find distinct experiment clusters for a certain gene cluster, the model predicts one broad normal distribution for the genes in this cluster. Any expression data for these genes will fit the model and thereby obscure the signal of the genes for which true predictions are made. For the particular data used in the Model Evaluation section,

the number of genes in the new dataset is 4,292, versus 4,345 in the dataset used to learn the models. Six of the missing genes belong to the regulator list for the learned models. Hence in some models we may not be able to compute the conditional probability distributions for all genes. Altogether around 2,700 genes remain for each model.

Software Availability

The Java software package LeMoNe for learning module networks is available from our website http://bioinformatics.psb.ugent.be/software/details/LeMoNe.

Acknowledgments

RDS is a research assistant of the IWT. AJ is supported by an Early-Stage Marie Curie Fellowship. This work is supported by (1) Research Council KUL: GOA AMBioRICS,GOA/08/011, CoE EF/05/007 SymBioSys, (2) FWO: projects G.0318.05, (3) IWT: SBO-BioFrame, and (4) IUAP P6/25 (BioMaGNet).

Conflicts of Interest

The authors declare no conflicts of interest.

References

1. Friedman, N. 2004. Inferring cellular networks using probabilistic graphical models. *Science* **308:** 799–805.
2. Gardner, T.S. & J.J. Faith. 2005. Reverse-engineering transcription control networks. *Phys. Life Rev.* **2:** 65–88.
3. Lemmens, K., T. Dhollander, T. De Bie, *et al*. 2006. Inferring transcriptional modules from ChIP-chip, motif and microarray data. *Genome Biol.* **7:** R37.
4. Bansal, M., V. Belcastro, A. Ambesi-Impiombato & D. di Bernardo. 2007. How to infer gene networks from expression profiles. *Mol. Syst. Biol.* **3:** 78.
5. Bussemaker, H.J., B.C. Foat & L.D. Ward. 2007. Predictive modeling of genome-wide mRNA expression: from modules to molecules. *Annu. Rev. Biophys. Biomol. Struct.* **36:** 329–347.
6. Butte, A.J. & I.S. Kohane. 2000. Mutual information relevance networks: functional genomic clustering using pairwise entropy measurements. *Pac. Symp. Biocomputing* **5:** 415–426.
7. Basso, K., A.A. Margolin, G. Stolovitzky, *et al*. 2005. Reverse engineering of regulatory networks in human β cells. *Nat. Genet.* **37:** 382–390.
8. Faith, J.J., B. Hayete, J.T. Thaden, *et al*. 2007. Large-scale mapping and validation of *Escherichia coli* transcriptional regulation from a compendium of expression profiles. *PLoS Biol.* **5:** e8.
9. Segal, E., M. Shapira, A. Regev, *et al*. 2003. Module networks: identifying regulatory modules and their condition-specific regulators from gene expression data. *Nat. Genet.* **34:** 166–167.
10. Beer, M.A. & S. Tavazoie. 2004. Predicting gene expression from sequence. *Cell* **117:** 185–198.
11. Bonneau, R., D.J. Reiss, P. Shannon, *et al*. 2006. The Inferelator: an algorithm for learning parsimonious regulatory networks from systems-biology data sets *de novo*. *Genome Biol.* **7:** R36.
12. Van den Bulcke, T., K. Lemmens, Y. Van de Peer & K. Marchal. 2006. Inferring transcriptional networks by mining 'omics' data. *Curr. Bioinformatics* **1:** 301–313.
13. Dietterich, T.G. 1997. Machine learning research: four current directions. *AI Mag.* **18:** 97–136.
14. Michoel, T., S. Maere, E. Bonnet, *et al*. 2007. Validating module networks learning algorithms using simulated data. *BMC Bioinformatics* **8:** S5.
15. Bar-Joseph, Z., G.K. Gerber, T.I. Lee, *et al*. 2003. Computational discovery of gene modules and regulatory networks. *Nat. Biotechnol.* **21:** 1337–1342.
16. Xu, X., L. Wang & D. Ding. 2004. Learning module networks from genome-wide location and expression data. *FEBS Lett.* **578:** 297–304.
17. Gao, F., B.C. Foat & H.J. Bussemaker. 2004. Defining transcriptional networks through integrative modeling of mRNA expression and transcription factor binding data. *BMC Bioinformatics* **5:** 31.
18. Wu, W.-S., W.-H. Li & B.-S. Chen. 2006. Computational reconstruction of transcription regulatory modules of the yeast cell cycle. *BMC Bioinformatics* **7:** 421.
19. Michoel, T., R. De Smet, A. Joshi, *et al*. 2008. Topological and evolutionary characterization of *Escherichia coli* transcriptional modules inferred from expression data. Submitted.
20. Joshi, A., R. De Smet, K. Marchal, *et al*. 2009. Module networks revisited: computational assessment and prioritization of model predictions. *Bioinformatics* (to appear).
21. Joshi, A., Y. Van de Peer & T. Michoel. 2008. Analysis of a Gibbs sampler for model based clustering of gene expression data. *Bioinformatics* **24:** 176–183.

22. Salgado, H., S. Gama-Castro, M. Peralta-Gil, *et al*. 2006. RegulonDB (version 5.0): *Escherichia coli* K-12 transcriptional regulatory network, operon organization, and growth conditions. *Nucleic Acids Res.* **34:** D394–D397.
23. Keseler, I.M., J. Collado-Vides, S. Gama-Castro, *et al*. 2005. EcoCyc: a comprehensive database resource for *Escherichia coli*. *Nucleic Acids Res.* **33:** 334–337.
24. Tramonti, A., P. Visca, M. De Canio, *et al*. 2002. Functional characterization and regulation of gadX, a gene encoding an AraC/XylS-like transcriptional activator of the *Escherichia coli* glutamic acid decarboxylase system. *J. Bacteriol.* **184:** 2603–2613.
25. Tucker, D.L., N. Tucker, Z. Ma, *et al*. 2003. Genes of the GadX-GadW regulon in *Escherichia coli*. *J. Bacteriol.* **185:** 3190–3201.
26. Foster, J.W. 2004. *Escherichia coli* acid resistance: tales of an amateur acidophile. *Nat. Rev. Microbiol.* **2:** 898–907.
27. Landgraf, J.R., J.A. Boxer & J.M. Calvo. 1999. *Escherichia coli* Lrp (leucine-responsive regulatory protein) does not directly regulate expression of the leu operon promoter. *J. Bacteriol.* **181:** 6547–6551.
28. Hung, S., P. Baldi & G.W. Hatfield. 2002. Global gene expression profiling in *Escherichia coli* K12. The effects of leucine-responsive regulatory protein. *J. Biol. Chem.* **277:** 40309–40323.
29. Peterson, S.N., F.W. Dahlquist & N.O. Reich. 2007. The role of high affinity non-specific DNA binding by Lrp in transcriptional regulation and DNA organization. *J. Mol. Biol.* **369:** 1307–1317.
30. Pul, U., R. Wurm & R. Wagner. 2007. The role of LRP and H-NS in transcription regulation: involvement of synergism, allostery and macromolecular crowding. *J. Mol. Biol.* **366:** 900–915.