On Phase Transitions in Learning Sparse Networks

Goele Hollanders[1], Geert Jan Bex[1], Marc Gyssens[1],
Ronald L. Westra[2], and Karl Tuyls[2]

[1] Department of Mathematics, Physics, and Computer Science,
Hasselt University and Transnational University of Limburg,
Hasselt, Belgium
goele.hollanders@uhasselt.be
[2] Department of Mathematics and Computer Science,
Maastricht University and Transnational University of Limburg,
Maastricht, the Netherlands

Abstract. In this paper we study the identification of sparse interaction networks as a machine learning problem. Sparsity means that we are provided with a small data set and a high number of unknown components of the system, most of which are zero. Under these circumstances, a model needs to be learned that fits the underlying system, capable of generalization. This corresponds to the student-teacher setting in machine learning. In the first part of this paper we introduce a learning algorithm, based on L_1-minimization, to identify interaction networks from poor data and analyze its dynamics with respect to phase transitions. The efficiency of the algorithm is measured by the generalization error, which represents the probability that the student is a good fit to the teacher. In the second part of this paper we show that from a system with a specific system size value the generalization error of other system sizes can be estimated. A comparison with a set of simulation experiments show a very good fit.

Keywords: machine learning, sparse network reconstruction, feature identification.

1 Introduction and Motivation

In this paper we consider the problem of identifying interaction networks from a given set of observations. An example of such a network is a sparse gene-protein interaction network, for more details see [1,5,2,7,10,11].

In some engineering applications, the number of measurements M available for system identification and model validation is much smaller than the system order N, which represents the number of components. This substantial lack of data can give rise to an identifiability problem, in which case a larger subset of the model class is entirely consistent with the observed data so that no unique model can be proposed. Since conventional techniques for system identification are not well suited to deal with such situations, it thus becomes important to work around this by exploiting as much additional information as possible about the underlying system. In particular, we are interested in the relation between the number of measurements and the number of components, the sparsity of the regulatory network and the influence of noise.

J.N. Kok et al. (Eds.): ECML 2007, LNAI 4701, pp. 591–599, 2007.

In this setting, it is natural to link network identification to feature selection. Only very few components influence the expression level of any given component, so one can restate the problem as selecting exactly those few among the large amount of components under consideration. Hence the results presented here will not only be applicable to network identification, but more generally to feature selection as well.

In Section 2, we will introduce the definitions of several concepts we use. Section 3 summarizes five research questions we will answer on experimental results. A brief discussion and our conclusions will be presented in Section 4.

2 Definitions and Algorithm

In the first paragraph we translate the problem of network identification formally into machine learning terminology [6]. In the next paragraph we introduce and elucidate the learning algorithm. Then we elaborate on the relation to feature selection. Finally, we discuss the issue of noisy data.

Terminology: Data and Teacher. In order to formalize the problem stated in the previous section we now introduce the model which we will consider more rigorously below. We assume that a *training set* of M input/output pairs $\chi_{\text{tr}} = \{(x_m, \dot{x}_m) \mid m : 1, \ldots, M\}$ is given, where $x_m, \dot{x}_m \in \mathbb{R}^N$. The components of the input vectors x_m are independently and identically distributed so that they are linearly independent. Since the data is assumed to be generated by some interaction network, this network will be denoted by $T = (A^T, B^T)$ where $A^T \in \mathbb{R}^{N \times N}$ and $B^T \in \mathbb{R}^N$. In this context, we refer to T as the unknown *teacher*. For each $(x_m, \dot{x}_m) \in \chi_{\text{tr}}$, $\dot{x}_m = A^T \cdot x_m + B^T$, i.e., $\dot{x}_m$ is the output produced by the teacher T on input x_m. In general, the teacher's output $\dot{x}$ on some input x is computed as follows: $\dot{x} = T(x) \equiv A^T \cdot x + B^T$. Moreover, we consider sparse networks, for each row of the matrix A^T, only K_T components are non-zero. Since the latter models the interactions in the network, a non-zero value of $A^T_{i,j}$ indicates that component i of input x influences component j of the output $\dot{x}$. So the sparsity constraint implies that each component of the output is determined by exactly K_T components of the input.

Learning Algorithm. The learning algorithm should return a network $S = (A^S, B^S)$, referred to as the *student*, with $A^S \in \mathbb{R}^{N \times N}$ and $B^S \in \mathbb{R}^N$, that reproduces the training set χ_{tr}: $\dot{x}_m = A^S \cdot x_m + B^S$ for $m : 1, \ldots, M$. More importantly however, the student should also perform well on input that was not used by the algorithm, i.e., the algorithm should be able to generalize beyond the training set χ_{tr}. To test the student's generalization ability, we use a validation set $\chi_{\text{v}} = \{(x_v, \dot{x}_v) \mid v : 1, \ldots, V\}$ such that $\dot{x}_v = A^T \cdot x_v + B^T$ for each $v : 1, \ldots, V$. The *generalization error* ε_{gen} is defined as the ratio of the number of tuples in χ_{v} that is not reproduced by the student to the total number of tuples. More formally, it is the fraction of the patterns in χ_{v} for which $|\dot{x}_v - A^S \cdot x_v - B^S|/|\dot{x}_v| > \varepsilon_{\text{err}}$, where ε_{err} is the maximum deviation from zero that is considered insignificant. The learning task can now be formulated as follows: *the algorithm should produce a student* S *given* χ_{tr} *such that* ε_{gen} *is minimal.*

The algorithm we use is a reformulation of the problem in terms of linear programming: the objective is to minimize $\|A^S\|_1$ subject to the M constraints $\dot{x}_m = A^S \cdot x_m + B^S$.

In the target function, $\|C\|_1$ denotes the 1-norm of the matrix C, i.e., $\|C\|_1 = \sum_{i,j} |C_{i,j}|$. This choice is motivated by the sparsity constraint on the networks to be identified. If the student S reproduces the teacher T, it will be sparse, hence we prefer solutions with as few non-zero components as possible. It is known from the literature [3,4] that the 1-norm is an acceptable approximation for the 0-norm. Since the latter can only be computed by explicit enumeration, it is unsuitable in practice due to the ensuing combinatorial explosion. For more details about this technique, see [8] and [9].

The constraints can be written more explicitly as:

$$\sum_{i=1}^{N} A_{i,j}^{S} x_{m,j} + B_i^S = \dot{x}_{m,i}, \quad j : 1, \ldots, N; \; m : 1, \ldots, M. \tag{1}$$

Hence each row of A and B is a solution to a set of M equations and can be determined independently, an observation to which we will return later on. For $M \leq N$, infinitely many solutions can be found, from which linear programming will select the most sparse. Trivially, for $M = N + 1$ the set of equations will have a unique solution: the teacher T. This implies that one can expect a generalization error $\varepsilon_{\text{gen}} \approx 1$ for very small training sets, i.e., $M \ll N$, while $\varepsilon_{\text{gen}} \approx 0$ for $M \approx N$. We may conclude that ε_{gen} will be a function of the training set size M. By convention, the number of patterns such that $\varepsilon_{\text{gen}} = 1/2$ is denoted by M_{gen}, the *generalization threshold*.

Although the generalization error is a good measure to evaluate the student's quality, it will nevertheless be useful to consider a measure to compare the student's structure to that of the teacher. Since our setting is that of identifying interaction networks, the presence or absence of such an interaction in the inferred model S is important. This can be characterized by the following three quantities: (1) n_{fneg}, the number of *false negatives*, i.e., interactions that are modeled by T, but not by S; (2) n_{fpos}, the number of *false positives*, i.e., interaction modeled by S, but not by T; and (3) n_{corr} the number of *correlation errors*, i.e., those components of S and T that are significantly non-zero, but have opposite sign. These three quantities measure the quality of the identification process. By definition, $0 \leq n_{\text{fneg}} \leq NK_T$, $0 \leq n_{\text{fpos}} \leq N(N - K_T)$ and $0 \leq n_{\text{corr}} \leq NK_T$. Note that these error measures can all be zero, even if the student does not generalize well, i.e., $\varepsilon_{\text{gen}} > 0$. Also notice that $0 \leq n_{\text{fneg}} + n_{\text{fpos}} + n_{\text{corr}} \leq N^2$. Therefore, we aggregate these three measures into the operator $S \odot T = (N^2 - n_{\text{fneg}} - n_{\text{fpos}} - n_{\text{corr}})/N^2$ that measures the quality of the identification.

Relation to Feature Selection. From Eq. (1), it is clear that the problem of identifying the interactions within a network modeled by the matrix A^T can be decomposed into identifying the N rows of that matrix. Since, apart from the sparsity constraint, interactions in the teacher are completely random, these rows can be determined independently. Hence we can reformulate the original problem in terms of N simpler ones: given an input vector $x \in \mathbb{R}^N$, which of the N components of x will effectively contribute to the output $\dot{x} \in \mathbb{R}$? This can be viewed as a feature selection problem, since the sparsity of the teacher implies that only very few components will contribute. As for network identification, we can define the generalization error for feature selection $\varepsilon_{\text{gen}}^{\text{fs}}$. At this point, it is useful to note that the generalization error can be interpreted as the probability that the student will not compute the correct output on a random input. The

probability that N independent feature selection problems will *all* compute the correct answer is thus given by $(1-\varepsilon_{\text{gen}}^{\text{fs}})^N$, which allows us to compute the generalization error for network identification ε_{gen} from that for feature selection $\varepsilon_{\text{gen}}^{\text{fs}}$ as follows:

$$\varepsilon_{\text{gen}} = 1 - (1 - \varepsilon_{\text{gen}}^{\text{fs}})^N \tag{2}$$

Noisy data. Until now, we have considered an ideal situation in the sense that the data χ_{tr} used to identify the network was noise-free. Obviously, the quality of real world data is typically far from ideal and an algorithm can only be used effectively in practice if it is robust to noise. To model this situation, we will consider a training set with noise: $\chi_{\text{tr}} = \{(x_m, \dot{x}_m + \delta_m) \mid m : 1, \ldots, M\}$ where $\delta_m \in \mathbb{R}^N$. The δ_m are identically and independently distributed and randomly drawn from a normal distribution with zero mean and standard deviation σ_{noise}. To quantify the quality of a student derived from a noisy training set, we introduce the *output deviation*, defined as $\delta\dot{x} = \sum_{x \in \chi_{\text{v}}} |T(x) - S(x)|/|T(x)|$.

3 Experiments

In this section, we will consecutively address the following research questions:

1. Is it possible to identify T with a training set that contains less than $N + 1$ input-output pairs? If so, what is the value of the generalization error ε_{gen} as a function of the training set size?
2. Does the generalization error ε_{gen} depend on the teacher's sparsity?
3. What is the evolution of the student when compared with the teacher as a function of the training set size?
4. Is the algorithm robust against noise?
5. How does the generalization error ε_{gen} scale with the system size N?

All experiments have been carried out using the algebra package Maple 9.5 on a Pentium-M class processor of 1.73 GHz and 1 GB of RAM. The standard implementation of linear programming in Maple is used, which is very convenient since it allows to specify the objective function and the constraints symbolically.

To facilitate the discussion, we first introduce some convenient notation. The ratio of the training set size to the system size is denoted by $\alpha = M/N$. In particular, $\alpha_{\text{gen}} = M_{\text{gen}}/N$. The fraction of non-zero components per row of a system is denoted by $\kappa = K/N$. In particular, $\kappa_T = K_T/N$. The amplitude of the noise should be considered relative to the amplitude of the signal, i.e., we define $\sigma = \sigma_{\text{noise}}/\sigma_{\dot{x}}$, where $\sigma_{\dot{x}}$ is the standard deviation of the output vectors' components $\dot{x}_{m,i}$. The components of the teacher A^T, B^T and of the input x_m are drawn from an uniform distribution over $]-1, 1[$.

Generalization error. To determine the generalization error, we randomly generate a set of M input vectors and a random teacher system so that we can compute the output to obtain a training set χ_{tr}. The algorithm produces a student, for which we calculate the generalization error. Since its value depends on the particular selection of input and teacher, we independently repeat this procedure many times to compute the average. Fig. 1 shows the observed generalization error as a function of the training set size α.

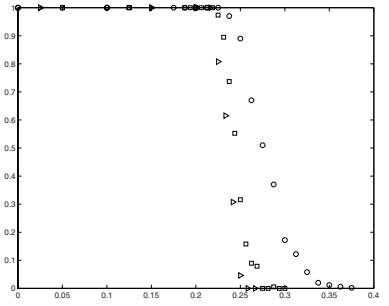

Fig. 1. The generalization error ε_{gen} as a function of the training set size α for $N = 100$ ($\circ$), $N = 160$ ($\square$) and $N = 300$ ($\triangleright$) for $\kappa_T \approx 0.03$

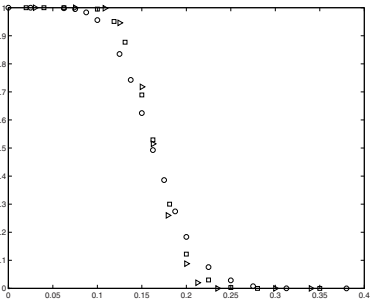

Fig. 2. The generalization error for feature selection $\varepsilon_{\text{gen}}^{\text{fs}}$ as a function of the training set size α for $N = 100$ ($\circ$), $N = 160$ ($\square$) and $N = 300$ ($\triangleright$) for $\kappa_T \approx 0.03$

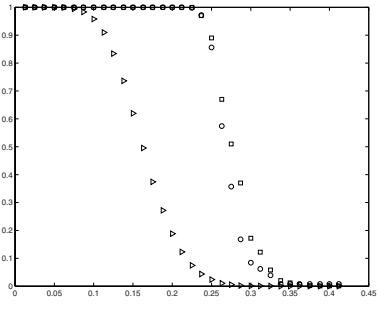

Fig. 3. The observed ($\square$) versus the computed ($\circ$) generalization error ε_{gen} as a function of α for $N = 80$, $\kappa_T \approx 0.03$. The curve representing $\varepsilon_{\text{gen}}^{\text{fs}}$ ($\triangleright$) is given as reference.

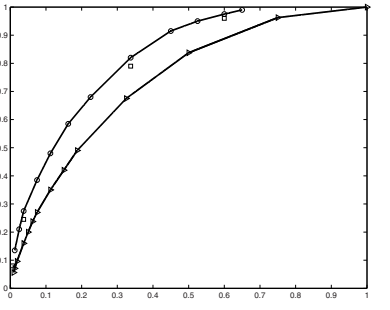

Fig. 4. The generalization threshold α_{gen} as a function of the sparsity κ_T for $N = 80$ ($\circ$) and a few values for $N = 160$ ($\square$). The generalization threshold for feature selection $\alpha_{\text{gen}}^{\text{fs}}$ ($\triangleright$) is given as reference.

This result is surprising in two respects: (1) the generalization error decreases to zero for a training set size $\alpha < 1$ and (2) the transition towards generalization is quite abrupt. It is also clear from Fig. 1 that the transition is increasingly abrupt for increasing system size N. It is instructive to relate the generalization error ε_{gen} for network identification to that for feature selection $\varepsilon_{\text{gen}}^{\text{fs}}$ in Fig. 2. The latter figure illustrates that $\alpha_{\text{gen}}^{\text{fs}}$, the training set size α for $\varepsilon_{\text{gen}}^{\text{fs}} = 1/2$, is independent of the system size N and that the $\varepsilon_{\text{gen}}^{\text{fs}}$ converges to a step-function for $N \to \infty$. Hence we observe a first order phase transition between a regime for $\alpha < \alpha_{\text{gen}}^{\text{fs}}$ where the student simply reproduces the training set, and another regime for $\alpha > \alpha_{\text{gen}}^{\text{fs}}$ where he is able to reproduce the teacher's output perfectly.

The relation between the generalization error for the network identification problem ε_{gen} and that for feature selection $\varepsilon_{\text{gen}}^{\text{fs}}$, given by Eq. (2), is illustrated in Fig. 3. It is clear that ε_{gen} for the network identification problem can reliably be estimated from $\varepsilon_{\text{gen}}^{\text{fs}}$ for

the feature selection problem. Values beyond α_{gen} are less reliable since inaccuracies for $\varepsilon_{\text{gen}}^{\text{fs}}$ are amplified considerably due to the mathematical form of Eq. (2).

Sparsity. The next question concerns the relation between the sparsity of the teacher and the generalization threshold. For a non-sparse teacher, i.e., $\kappa_T \approx 1$, one would need a training set of size $\alpha \approx 1$ since each of the $N(N+1)$ components has to be determined. However, as Fig. 1 illustrated, the fact that the teacher is sparse simplifies the identification process considerably. Fig. 4 shows the generalization threshold α_{gen} for network identification as a function of κ_T. It is clear that training sets of increasing size α are required to facilitate the transition to the generalization regime as κ_T increases, i.e., as the sparsity decreases. As expected, for $\kappa_T \approx 1$, $\alpha_{\text{gen}} \approx 1$. It is clear that the advantage sparsity offers to the efficiency of the learning algorithm virtually vanishes for $\kappa_T \approx 0.5$. However, it is very pronounced for $\kappa_T < 0.2$. As before, these results have been obtained for many independent instances of the training set and teacher.

Learning process. To gain a better understanding of the learning process, i.e., the evolution of the student with respect to the teacher as a function of the training set size we first consider a fixed training set and teacher. Define a sequence of training sets χ_m for $m : 1, \ldots, M$ such that $\chi_m \subset \chi_{m+1}$ and $|\chi_{m+1}| = |\chi_m| + 1$. These sets are used to determine a sequence of students S_m for $m : 1, \ldots, M$. Fig. 5 shows $S_{\alpha N} \odot T$ as a function of the training set size α. For $\alpha N = 1$, the number of false negatives is $N^2\kappa_T$ and the number of false positives is 0. For increasing α, the number of false positives increases approximately linearly with α, while the number of false negatives decreases very slowly. The plot illustrates clearly that the transition to generalization is very sudden: at α_{gen}, $S_{\alpha_{\text{gen}} N} \odot T = 1$. Fig. 6 and 7 confirm that the scenario sketched above is indeed the typical behavior when it is averaged over many independent training sets and teachers. The latter plot illustrates the explanation given above for the behavior of $S_{\alpha N} \odot T$.

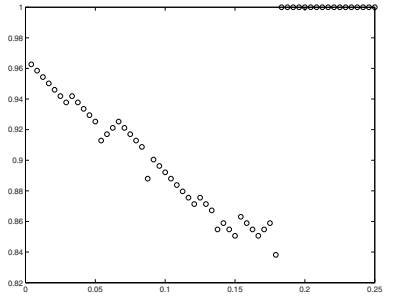

Fig. 5. The learning process characterized by $S_m \odot T$ as a function of the size of the training set m/N for an individual run

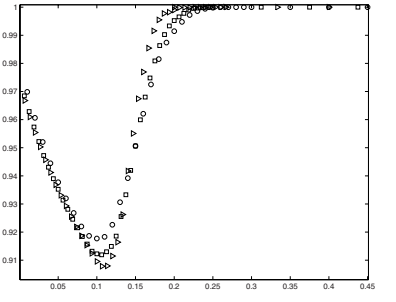

Fig. 6. The learning process characterized by $S_m \odot T$ as a function of the size of the training set m/N, system sizes $N = 100$ (∘), $N = 160$ (□) and $N = 300$ (▷) for $\kappa_T \approx 0.03$

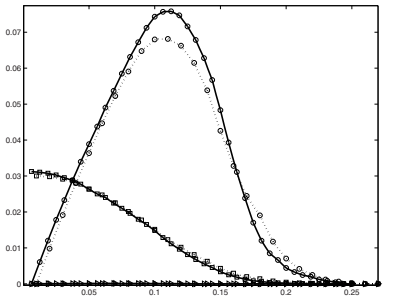

Fig. 7. Measures n_{fneg} (□, $N = 100$ dotted line, $N = 160$ solid line), n_{fpos} (○, $N = 100$ dotted line, $N = 160$ solid line) and n_{corr} (▷, $N = 100$ dotted line, $N = 160$ solid line) as a function of the training set size α for $\kappa_T \approx 0.03$

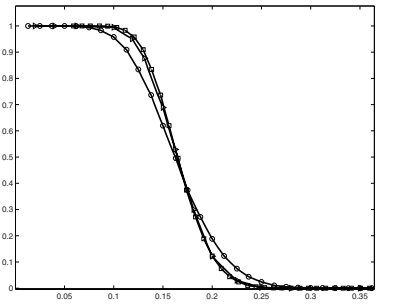

Fig. 8. Fig. 2 using Eq. (3) for $N = 100$ (○), $N = 160$ (□) computed, $N = 160$ observed (▷), $\kappa_T \approx 0.03$

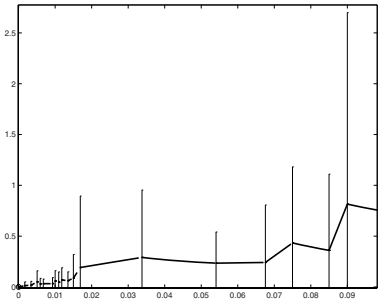

Fig. 9. Deviation of the student and teacher output $\delta\dot{x}$ as a function of the noise level σ on the training set

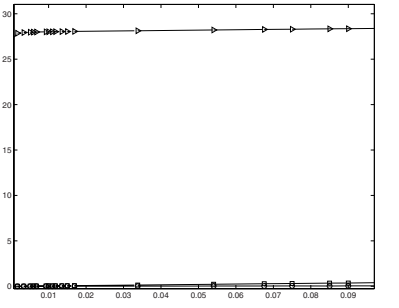

Fig. 10. n_{fneg} (□), n_{fpos} (▷) and n_{corr} (○) as a function of the noise level σ for $N = 100$, $\kappa_T \approx 0.03$

Noisy data. Noise is ubiquitous in real world applications, hence it is mandatory to test the algorithm's robustness. Fig. 9 shows the relative deviation of the respective output of student and teacher $\delta\dot{x}$ as a function of the noise level σ. As can be expected for a linear system, the quality is acceptable for low noise levels only. In particular, $\sigma < 0.01$ still yields a reasonably accurate output. The breakdown for higher noise levels is explained by Fig. 10 which shows a very large increase in the number of false positives n_{fpos} for an increasing noise level σ. Although these components are very small, they nevertheless preclude perfect identification of the network.

Scaling with system size. How the system scales with the system size has already been illustrated in Fig. 1, 4, 6 and 7. However, it is Fig. 2 that provides the most insight. It turns out that the generalization error curves for various system sizes can be computed

by applying the correct scaling on the system size α. Suppose we have a curve for $\varepsilon^{\text{fs}}_{\text{gen}}$ versus α for N_0, then the curve for system size N can be obtained by scaling

$$\alpha(N) = \alpha^{\text{fs}}_{\text{gen}} + \sqrt{N_0/N}\,(\alpha(N_0) - \alpha^{\text{fs}}_{\text{gen}}) \tag{3}$$

The result is shown in Fig. 8 for system sizes $N = 100$ and $N = 160$ with sparsity $\kappa_T = 0.03$. The curve computed for $N = 160$ from that for $N = 100$ is in very good agreement with the one observed for that system size.

4 Discussion and Conclusions

It is quite remarkable that a simple model such as the one considered here exhibits so many interesting features. With respect to the research questions addressed, we may conclude that the algorithm identifies a network with $N(N+1)$ interactions using a training set of considerably smaller size. This turns out to be a consequence of the teacher's sparsity. Moreover, a first order phase transition occurs during the learning process. The system shows a sudden transition to perfect generalization during the learning process. The latter can be explained by considering the geometric interpretation of linear programming. Adding an additional constraint in the form of an input-output pair can lead to abrupt changes of the minimal values that can be attained by the objective function when its domain is further restricted. The relation between the feature selection problem and the network identification task is of note, especially since the generalization behavior of the latter can be derived from the former's. Moreover, the scaling properties of feature selection have been demonstrated: given the generalization curve for a certain size and a fixed sparsity, one can compute the generalization curve for a system of any size with that sparsity. Unfortunately, the algorithm's robustness to noise is fairly limited. This is to be expected given the nature of linear programming as mentioned above. This is definitely an area for future research.

References

1. de Jong, H.: Modeling and simulation of genetic regulatory systems: A literature review. Comp. Biol. 9, 67–103 (2002)
2. Jong, H.d., et al.: Qualitative simulation of genetic regulatory networks using piecewise-linear models. Bulletin of Mathematical Biology 66(2), 301–340 (2004)
3. Fuchs, J.: More on sparse representations in arbitrary bases. In: Proc. 13th IFAC Symp. on System Identification, pp. 1357–1362 (2003)
4. Fuchs, J.: On sparse representations in arbitrary redundant bases. IEEE Trans. Infor. Theory 50(6), 1341–1344 (2004)
5. Glass, L., Kauffman, S.: The logical analysis of continuous non-linear biochemical control networks. J. Theor. Biol. 39(1), 103–129 (1973)
6. Mitchell, T.M.: Machine Learning. McGraw-Hill, New York (1997)
7. Novak, B., Tyson, J.: Modeling the control of dna replication in fission yeast. PNAS 94, 9147–9152 (1997)
8. Peeters, R.L.M., Westra, R.L.: On the identification of sparse gene regulatory networks. In: Proc. of the 16th Intern. Symp. on Math. Theory of Networks and Systems (2004)

9. Westra, R.L.: Piecewise linear dynamic modeling and identification of gene-protein interaction networks. In: Nisis/JCB Workshop Reverse Engineering (2005)
10. Westra, R.L., Hollanders, G., Bex, G., Gyssens, M., Tuyls, K.: The identification of dynamic gene-protein networks. In: Tuyls, K., Westra, R., Saeys, Y., Nowé, A. (eds.) KDECB 2006. LNCS (LNBI), vol. 4366, pp. 157–170. Springer, Heidelberg (2007)
11. Yeung, M.K.S., Tegnér, J., Collins, J.: Reverse engineering gene networks using singular value decomposition and robust regression. PNAS 99(9), 6163–6168 (2002)