BMC Systems Biology

RESEARCH ARTICLE Open Access

Inferring genetic interactions via a nonlinear model and an optimization algorithm

Chung-Ming Chen[1†], Chih Lee[2†], Cheng-Long Chuang[1], Chia-Chang Wang[2], Grace S Shieh[2*]

Abstract

Background: Biochemical pathways are gradually becoming recognized as central to complex human diseases and recently genetic/transcriptional interactions have been shown to be able to predict partial pathways. With the abundant information made available by microarray gene expression data (MGED), nonlinear modeling of these interactions is now feasible. Two of the latest advances in nonlinear modeling used sigmoid models to depict transcriptional interaction of a transcription factor (TF) for a target gene, but do not model cooperative or competitive interactions of several TFs for a target.

Results: An S-shape model and an optimization algorithm (GASA) were developed to infer genetic interactions/ transcriptional regulation of several genes simultaneously using MGED. GASA consists of a genetic algorithm (GA) and a simulated annealing (SA) algorithm, which is enhanced by a steepest gradient descent algorithm to avoid being trapped in local minimum. Using simulated data with various degrees of noise, we studied how GASA with two model selection criteria and two search spaces performed. Furthermore, GASA was shown to outperform network component analysis, the time series network inference algorithm (TSNI), GA with regular GA (GAGA) and GA with regular SA. Two applications are demonstrated. First, GASA is applied to infer a subnetwork of human T-cell apoptosis. Several of the predicted interactions are supported by the literature. Second, GASA was applied to infer the transcriptional factors of 34 cell cycle regulated targets in *S. cerevisiae*, and GASA performed better than one of the latest advances in nonlinear modeling, GAGA and TSNI. Moreover, GASA is able to predict multiple transcription factors for certain targets, and these results coincide with experiments confirmed data in YEASTRACT.

Conclusions: GASA is shown to infer both genetic interactions and transcriptional regulatory interactions well. In particular, GASA seems able to characterize the nonlinear mechanism of transcriptional regulatory interactions (TIs) in yeast, and may be applied to infer TIs in other organisms. The predicted genetic interactions of a subnetwork of human T-cell apoptosis coincide with existing partial pathways, suggesting the potential of GASA on inferring biochemical pathways.

Background

Biologists are gradually recognizing that pathways, rather than individual genes, control tumorigenesis [1]. Moreover, altered pathways have recently been reported to be crucial factors for colorectal and breast cancer [2]. The present approach (GASA) was motivated by the inference of genetic interactions, which have potential for inferring pathways in yeast [3,4]. Because genetic networks derived from yeast are likely to be conserved in humans, the prediction of genetic interactions may shed light on the pathways of complex human diseases, such as cancers and type II diabetes. In addition, GASA can also be applied to infer other types of networks, for example transcriptional regulatory networks. With the abundant sets of microarray gene expression data (MGED) now available, inferring genetic interactions has become feasible, and various approaches have been proposed. Most of the approaches may be classified into three classes: graphical models, discrete variable models and continuous variable models. Due to space constraints, here we concentrate our review on continuous variable models that are directly relevant to GASA; see [3] for further reviews of models from other classes.

* Correspondence: gshieh@stat.sinica.edu.tw
† Contributed equally
[2]Institute of Statistical Science, Academia Sinica, No 128, Sec 2, Academia Road, Taipei 115, Taiwan

To approximate the nonlinear relationship of a target (T) and its activator (A) and repressor (R), [5] proposed ordinary differential equations including perturbations from genes of interest; the perturbations provided information of the underlying network topology. The time series network inference algorithm (TSNI) [6] further characterized the perturbations by a few linear external perturbations, and solved the system of equations by using principle component analysis (PCA) and singular value decomposition. On the other hand, network component analysis (NCA) [7] employs partial knowledge of the underlying network and requires no statistical assumption as PCA does [8] and [9] proposed order-two models on interactions of A, R and T. Some latest advances of nonlinear models are as follows. Climescu-Haulica and Quirk proposed a beta-sigmoid function to model the local transcriptional effect on a target in [10]. And Vu and Vohradsky presented a sigmoid model to depict the interaction between a target and its transcriptional factors (TFs) in [11], where order-n polynomials were used to approximate the model. The algorithm was efficient and it predicted more regulators for 40 yeast cell cycle regulated targets than the generalized linear model in [12]. However, the model in [11] is not able to depict cooperative or competitive TFs for a target gene simultaneously. An alternative nonlinear model is the S-system [13,14], which satisfies several cellular processes. However, to model a network of k factors regulating a given gene, the proposed sigmoid model requires n (n + k + 2) parameters while the S-system requires 2n (n + 1). Despite the many merits of the S-system, the large number of parameters required restricts its applications in the area.

To model biological processes occurring simultaneously among a small set of genes such as in genetic interactions, we propose a system of nonlinear equations; for an earlier version of this system see [15]. This model also extends the aforementioned sigmoid models to depict cooperative/competitive interactions among genes. Note that each *tanh* function in this system can include a term, called *factor*, to model genes that regulate other genes but are not regulated by others in the network, and in this sense it extends the linear dynamic factor model in [3].

The transcriptional rate is known to be S-shaped [16]. Let $g_i(t)$ and Δg_i (t + 1) denote gene i's expression at time t and its expression change at time t+ + 1 respectively, where Δg_i (t + 1) = $g_i(t + 1)$ - g_i (t). This model states that in the network, each gene's expression change is regulated by a weighted sum of certain genes cooperatively at time t only if this weighted (combined) sum surpasses a certain threshold. The proposed nonlinear model depicts the transcriptional rate of gene i as follows.

$$\Delta g_i(t+1)/\Delta t = \alpha_1 \tanh\left[\sum_{j=1}^{n} w_{ji} g_j(t) + \sum_{k=1}^{K} w'_{ki} F_k(t) - \beta_i\right] + \varepsilon_i(t), \quad (1)$$

where $\varepsilon_i(t) \sim N(0, \sigma_i^2)$ and σ_i^2 is the variance of gene i's expression levels, which can be estimated from real MGED. For a given gene i, $|\alpha_i|$ is the range of $\Delta g_i/\Delta t$ (the rate of expression change of g_i), β_i is the location parameter at which the *tanh* function crosses zero, w_{ji} is the regulation of $g_j(t)$ on $\Delta g_i(t + 1)$ before being amplified by the α_i *tanh*($\cdot$) function, K is the number of factors F_k*'s* in the local network and $K \le n$. Note that [11] can be regarded as a special case of Eq. (1), namely when only one gene inside the *tanh* function regulates g_i*'s* transcription rate.

An optimization algorithm, consisting of a genetic algorithm (GA) and a simulated annealing (SA) algorithm, to evolve the optimal genetic network and then estimate the parameters using time course MGED, is developed in the Methods section. This SA is enhanced by a steepest-gradient-descent algorithm to prevent it from being trapped in local minima. A strategy to identify factors in the network is also proposed. The Results section consists of applications of GASA to both simulated data and real time course MGED. First, an S-shape non-linear model (a network) is applied to simulate data with various degrees of noise. Using these generated data, we study how GASA with two model selection criteria and two search spaces performs compared to TSNI, NCA, GA with regular SA (GA-regular SA) and GAGA. Note that GAGA applies a GA algorithm and some model selection criterion to predict networks, but uses another GA to optimize interactions w_{ji} 's in Eq. (1), instead of the enhanced SA in GASA. Second, GASA is applied to real gene expression data sets to infer a partial pathway of human T-cell apoptosis, and the TFs of 34 targets in yeast cell cycle to provide a comparison to TSNI, GAGA and NLDE in [11]. Both predictions are checked against published literature.

Results and Discussion

When implemented with a factor analysis algorithm, AIC and BIC model selection criteria (MSC) outperform the other four MSC on inferring genetic networks using data simulated from a linear dynamic model [3]. However, the suitable MSC and search space (power law or non-power law) for GASA applied to data from a non-linear model remain unknown. In this section, we first simulate data from a sigmoid model with various degrees of noise to see how GASA with two MSC and two types of search space perform. The effectiveness of the proposed factor finding strategy and how smoothing circumvents contamination from noise are also studied. Next, GASA is applied to two real datasets to infer a small network of human T-cell apoptosis and the TFs of 34 target genes involved in yeast cell cycle.

Results on simulated data

Time course data from an 11-gene network including two external factors are simulated. These external factors model TFs or other known proteins, published in the literature, that regulate genes in the network. Two factors F_1 and F_2 were extracted from 51 yeast genes involved in DNA synthesis and repair [17], by applying independent component analysis in MATLAB to these genes' aggregated microarray data of the alpha, cdc15 and cdc28 sets (59 time points in total). The expression profiles of these two factors exhibited sinusoid patterns as shown in Figure 1 of [18]. The coefficients W_{ji}'s and W'_{ki}'s in Eq. (2) were determined by trial and error, and the initial values of the 11 genes were generated randomly then tuned manually such that these genes' expression curves also showed sinusoid patterns. For each gene, the initial value and the values of the two factors were plugged into Eq. (2) to recursively generate the rest time points. Simulated gene networks of two factors and 11 genes were generated by the following model.

$$\begin{aligned}
\Delta g_1(t) &= 0.2\tanh[-0.5g_1(t-1) + g_3(t-1) - 1.8g_7(t-1) + 0.1] + \varepsilon_1(t)\\
\Delta g_2(t) &= 0.3\tanh[-0.8g_2(t-1) - g_5(t-1) + 0.1] + \varepsilon_2(t)\\
\Delta g_3(t) &= 0.3\tanh[-0.6F_2(t-1) - 0.2g_3(t-1)] + \varepsilon_3(t)\\
\Delta g_4(t) &= 0.2\tanh[-2g_1(t-1) - g_4(t-1) - 0.3] + \varepsilon_4(t)\\
\Delta g_5(t) &= 0.3\tanh[-2g_5(t-1) + g_7(t-1) + 0.2] + \varepsilon_5(t)\\
\Delta g_6(t) &= 0.25\tanh[2g_2(t-1) - g_6(t-1) + 0.3] + \varepsilon_6(t)\\
\Delta g_7(t) &= 0.3\tanh[-F_1(t-1) + 0.1g_6(t-1) - 0.2g_7(t-1) - 0.4g_9(t-1) - 0.05] + \varepsilon_7(t)\\
\Delta g_8(t) &= 0.1\tanh[-0.75g_8(t-1) + 0.15g_9(t-1)] + \varepsilon_8(t)\\
\Delta g_9(t) &= 0.1\tanh[1.5g_7(t-1) + 0.2g_9(t-1)] + \varepsilon_9(t)\\
\Delta g_{10}(t) &= 0.1\tanh[-1.6g_3(t-1) + 0.2g_{10}(t-1) + 0.15] + \varepsilon_{10}(t)\\
\Delta g_{11}(t) &= 0.1\tanh[1.3g_3(t-1) - 0.8g_4(t-1) + 0.3g_{11}(t-1) + 0.4] + \varepsilon_{11}(t),
\end{aligned} \tag{2}$$

where $\varepsilon_i \sim N(0, \sigma_i^2)$, $\sigma_i^2 = Var(g_i)/c$, and c = 10 or 4 for i = 1,..., 11, representing data with signal noise ratio equal to 10 or 4 (denoted by SNR10 and SNR4), respectively. $Var(g_i)$ was calculated from the sample variance of real microarray data of g_i. SNR10 and SNR4 data were generated to mimic data with medium and low quality, respectively, while noise free data were generated without the noise terms $\varepsilon_i(t)$'s. Note that networks generated by Eq. (2) are sparse which roughly follows the property of *cis*-regulatory networks [19].

Before applying GASA to the datasets, it is necessary to distinguish factors from other genes. We first applied the proposed factor-finding strategy written in Python to the simulated noise free data, and GASA identified the two factors correctly. For data contaminated with noise, mean filters with the kernel size 1 × 3 and 1 × 5 were applied to smooth SRN10 and SRN4 data, respectively. Then, GASA identified the factors correctly; for details see http://www.stat.sinica.edu.tw/~gshieh/GASA/boxplots.pdf.

The performances of GASA with a model selection criterion (MSC) AIC or BIC and with or without the power law restriction (PL or no PL) in the network search space were studied using the three simulated data sets. We also compared GASA to NCA, TSNI, GAGA and GA-regular SA. Both 100% and 50% true connectivity information were inputted into NCA to see how the performances of NCA vary. Table 1, 2 and 3 summarize the comparisons of these algorithms. The true positive rate (TPR), true negative rate (TNR), and modified false positive rate (mFPR) of the predicted top 1 network in terms of AIC (BIC) score, are reported, where TPR (also known as sensitivity) is the percentage

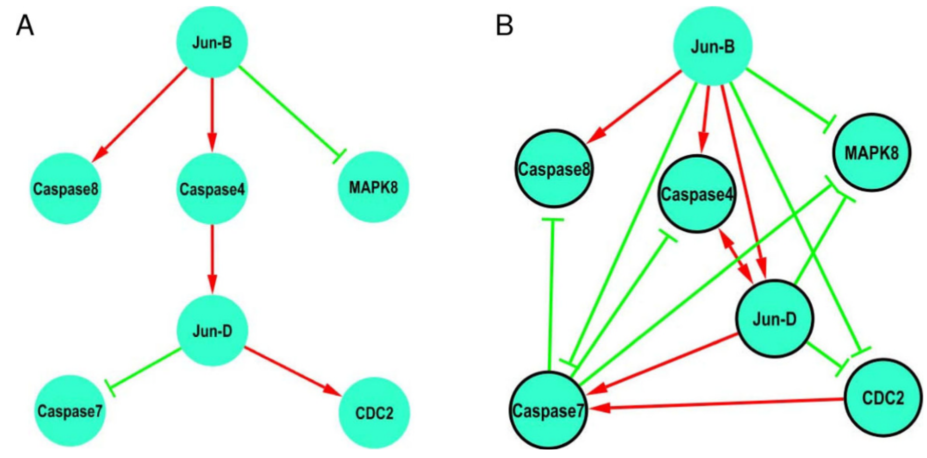

Figure 1 (A) The network inferred by Rangel *et al.* (2004) with red links denoting enhancing, and green links denoting repression. (B) The network predicted by GASA using AIC and without the power law restriction in the search space. Black circles around genes denote repressive self-loops.

Table 1 Performances of GASA, TSNI, NCA, GAGA and GA-regular SA applied to one repeat of data simulated from Eq. (2) with no noise.

		# int[a]	# pc[b]	TPR[c]	TNR[d]	FPR[d]	mFPR[e]
GASA	AIC/no power law			0.81	0.99	0.01	0.05
	BIC/power law			0.77	0.99	0.01	0.05
GA-regular SA	AIC/no power law			0.69	0.97	0.03	0.18
	BIC/power law			0.73	0.97	0.03	0.14
NCA	100% true connectivity			0.62	0.95	0.05	0.27
	50% true connectivity			0.24	0.85	0.15	0.75
TSNI	inputting prior knowledge: 26 true links	3	1	0.50	0.89	0.11	0.50
		3	2	0.50	0.89	0.11	0.50
		3	3	0.50	0.89	0.11	0.50
GA-GA	AIC/no power law			0.46	0.79	0.21	0.68
	BIC/power law			0.35	0.83	0.17	0.69

[a] '# int' denotes the number of interpolations.
[b] '# PC' denotes the number of principal components
[c] TPR is the ratio of the correctly predicted links to the total number of existing links in a simulated network. Note signs of interactions were not accounted toward TPR and other performance measures.
[d] TNR (FPR) is the ratio of correctly predicted non-existing links (false positives) over the total true negatives.
[e] mFPR is the ratio of incorrectly predicted links to the total predicted links.

Table 2 Performances of GASA, TSNI, NCA, GAGA and GA-regular SA applied to data simulated from Eq. (2) with medium level of noise, where the averaged results of five repeats are reported

		# int[a]	# pc[b]	TPR[c]	TNR[d]	FPR[d]	mFPR[e]
GASA	AIC/no power law			0.79	0.99	0.01	0.05
	BIC/power law			0.79	0.99	0.01	0.05
GA-regular SA	AIC/no power law			0.46	0.97	0.03	0.25
	BIC/power law			0.42	0.93	0.07	0.42
NCA	100% true connectivity			0.51	0.92	0.08	0.41
	50% true connectivity			0.29	0.86	0.14	0.68
TSNI	Inputting prior knowledge: 26 true links	3	1	0.50	0.89	0.11	0.50
		3	2	0.50	0.89	0.11	0.50
		3	3	0.50	0.89	0.11	0.50
GA-GA	AIC/no power law			0.35	0.78	0.22	0.74
	BIC/power law			0.31	0.85	0.15	0.69

[a] '# int' denotes the number of interpolations.
[b] '# PC' denotes the number of principal components
[c] TPR is the ratio of the correctly predicted links to the total existing links in a simulated network. Note signs of interactions were not accounted toward TPR and other performance measure.
[d] TNR (FPR) is the ratio of correctly predicted non-existing links (false positives) over the total true negatives.
[e] mFPR is the ratio of incorrectly predicted links to the total predicted links.

of correctly predicted links out of the total number of existing interactions (links) in a simulated network. Similarly, TNR (FPR) is the ratio of correctly predicted non-existing interactions (predicted false positives) over the total true non-existing interactions (negatives) in a simulated network, and mFPR is the ratio of incorrectly predicted interactions to the total predicted ones. There are much more non-existing interactions (117) than existing ones (26) in the true network (Eq. (2)), mFPR can distinguish the performances of the algorithms well, so it is reported in addition to FPR.

For each simulated network, we limited the maximum number of incoming links to four for each gene, hence GASA sought through a space of 4^{11} possibilities. Applying the algorithms to the three simulated datasets, their overall performances in terms of both TPR and mFPR in descending order are GASA, GA-regular SA, NCA (100% true connectivity), TSNI, GAGA and NCA (50% true connectivity). These results were computed without taking the signs of interactions into account. When the signs were also checked, the performance ranking is the same except that GAGA performed better

Table 3 Performances of GASA, TSNI, NCA, GAGA and GA-regular SA applied to data simulated from Eq. (2) with high level of noise, where the averaged results of five repeats are reported

		# int[a]	# pc[b]	TPR[c]	TNR[d]	FPR[d]	mFPR[e]
GASA	AIC/no power law			0.66	0.97	0.03	0.19
	BIC/power law			0.65	0.97	0.03	0.19
GA-regular SA	AIC/no power law			0.5	0.95	0.05	0.30
	BIC/power law			0.47	0.96	0.04	0.31
NCA	100% true connectivity			0.58	0.94	0.06	0.31
	50% true connectivity			0.25	0.85	0.15	0.73
TSNI	inputting prior knowledge: 26 true links	3	1	0.50	0.89	0.11	0.50
		3	2	0.50	0.89	0.11	0.50
		3	3	0.50	0.89	0.11	0.50
GA-GA	AIC/no power law			0.31	0.83	0.17	0.71
	BIC/power law			0.31	0.90	0.10	0.60

[a] '# int' denotes the number of interpolations.

[b] '# PC' denotes the number of principal components

[c] TPR is the percentage of correctly predicted links out of the total number of existing links in a simulated network. Note signs of interactions were not accounted toward TPR and other performance measure.

[d] TNR (FPR) is the ratio of correctly predicted non-existing links (false positives) over the total true negatives.

[e] mFPR is the ratio of incorrectly predicted links to the total predicted links.

than TSNI; see the additional file 1 for details. Specifically, for noise free data (with one repeat), the TPR, TNR and mFPR of GASA with the four combinations of MSCs and search spaces are similar and equal to 77-81%, 99% and 5%, respectively, whereas those of TSNI are 50%, 89% and 50%, respectively. GAGA has about 30% lower TPR and 60% higher mFPR than GASA using the two combinations BIC/PL and AIC/no PL. From five repeated experiments of SNR10 (SNR4 data), the TPR, TNR and mFPR of GASA with AIC/no PL and GASA with BIC/PL are the same (quite close), and they are both equal to 79%, 99% and 5% (66%, 97% and 19%), respectively; while the performances of TSNI for SNR10 and SNR4 are both equal to 50%, 89% and 50%, respectively. GAGA with AIC/no PL performs similarly to GAGA with BIC/PL, and both have about 35%, 78% and 74% of mTPR, TNR and mFPR. See Table 1, 2 and 3 for detailed performances; the predicted networks and the true one are plotted in additional file 2. The implementation of TSNI and GAGA is summarized in additional file 3. Each simulation of GASA took about 32 h and was conducted by PC cluster (limited to five nodes) with Pentium 3.4 GHz and 4.0 GB RAM; GASA was written in Python 2.1.3 and GNU C. Note that this CPU time can be significantly shortened by using modern multi-core architecture as stated at the end of Application 2.

Results with real time course microarray data

Application 1: A human T-cell apoptosis subnetwork

In [20], two experiments were conducted to characterize the response of a human T cell line (Jurkat) to PMA and ionomycin treatments. The mechanism studied is a key for clonal expansion and controlling long term behavior of T cells such as programmed cell death. Identical experimental protocols were used in the two experiments including more than 88 genes, but only 58 genes with good reproducibility were retained. There were 10 time points, with 34 and 10 replicates for each time point in the first and second experiments, respectively; we called this data set 'human T-cell line data'.

In this 7-gene sub-network, JNK, JUNB (alias name AP-1) and caspase-8 are involved in the apoptosis pathway. As reviewed in [21], apoptosis is a cell suicide mechanism, though which metazoans control cell number in tissues and eliminate individual cells that threaten the animal's survival. The physiological role of apoptosis is extremely important. For instance, unscheduled apoptosis of certain brain neurons leads to diseases such as Alzheimer's disease; and in dividing cells, failure to initiate apoptosis in cells that have serious damages in DNA contributes to cancer. Moreover, JNK (alias name MAPK8) and JunD are involved in the JNK signaling pathways, and JunD inhibits fibroblast proliferation and counteracts transformation by *ras*.

Prior biological information indicated that JUNB was not regulated by other genes in the network, thus JunB was specified as a factor. Next, we applied GASA with AIC/BIC and with/without power law restriction in the search space to the human T-cell line data. The score for GASA with AIC/no PL was the lowest (2526.9), lower than GASA with AIC/PL (2957.3). GASA with AIC/no power law predicted a few interactions that are in the existing pathways; see Figure 1 for details. In particular, JunB activates the apoptotic genes caspase-4 and caspase-8, and represses MAPK8. Moreover, GASA predicted two interactions that are consistent with known protein-protein interactions. Specifically, JunD interacts with MAPK8 which is involved in the JNK signaling pathways in mouse [22]; caspase-4 cleaves and activates its own precursor and caspase-1 precursor (Entrez Gene database at NCBI, http://www.ncbi.nlm.nih.gov/sites/entrez). Caspase-8 activates downstream effectors caspases and commits the cell to apoptosis [21]. Finally, eight testable predictions including three self-regulations were also inferred by GASA. GASA took about 1.4 h using 10-node PC cluster with Intel Xeon 2.0 GHz and 6 GB RAM.

Application 2: Inferring the regulators of 34 selected cell cycle regulated targets in yeast

The procedure in [11] (abbreviated as NLDE) is one of the latest advances in nonlinear modeling of transcriptional regulation. Because we could not access the NLDE code, to provide a possible comparison we applied GASA to infer the regulators of 34 yeast cell cycle regulated targets, which were inferred by [11]. Their TFs were collected from YEASTRACT http://www.yeastract.com. The data sets used were cDNA microarray data from three synchronization experiments in [18]; the *Elu* data set was not included because it was synchronized differently. The experiment and control groups were mRNAs extracted from synchronized and non-synchronized yeast cultures, respectively, where the synchronization was conducted by treating yeast cultures with alpha factor arrest and arrests of a temperature-sensitive mutant *cdc15* and mutant *cdc28*. The red (R) and green (G) fluorescence intensities were measured from the mRNA abundance in the experiment group and control group, respectively. A full description of data preprocessing is available in Additional file 4, and the data sets are available at http://genome-www.stanford.edu/cellcycle/ . We aggregated 18, 24 and 17 time points from these three datasets to 59 time points. This aggregation method was applied in [23] and [3], and it led to meaningful gene networks.

For each cell cycle regulated target, NLDE in [11] predicted one by one whether any of these 184 potential regulators was a true regulator. NLDE is good at screening large numbers of regulators but it can not predict a network of two or more genes simultaneously. To infer TFs for a given target gene, we inputted all 184 TFs simultaneously as potential regulators into one equation in Eq. (1) for GASA to infer. The mTPR* (mFPR) of GASA, GAGA (both using AIC/no PL to select the number of TFs) and TSNI are 65% (45%), 29% (77%) and 8% (92%), respectively, where mTPR* is the ratio of the number of predicted positives over the minimum between the number of validated TFs in YEASTRACT and the total predicted links. TSNI was implemented with no perturbation, three interpolations and three principle components, which were obtained by the curve maximizing the area of TNR versus TPR as [6]; the results of one and two principle components are also reported. Note that YEASTRACT consists of both documented regulators (experimentally confirmed to date) and potential regulators (predicted by matching TF promoter and TFBS). The confirmed TFs therein were used (Yeastract_TFs_34_targets.pdf at http://www.stat.sinica.edu.tw/~gshieh/GASA, but they are far from complete, therefore the mFPRs computed here may be higher than they should be. We could not access the executive code of NLDE; however, summarizing from Table 2 of [11], the mTPR* and mFPR of NLDE are about 2% and 98%, respectively. These prediction accuracies are presented to serve as contrasts, not as direct comparisons. See Application2.pdf at http://www.stat.sinica.edu.tw/~gshieh/GASA for details of the results. On average GASA took about 88 min to predict TFs for each target using one node from a mini-cluster (six Intel i7-950 with 3.06 GHz per node sharing 24 GB RAM, and accelerated by 480 GPU per node sharing 2 GB RAM.

Discussion

The prediction results of simulated data show that GASA can infer small networks more accurately than TSNI, NCA, GAGA and GA-regular SA. The results of inferring TFs for the 34 cell-cycle regulated targets using real gene expression data in Application 2 suggest that GASA characterizes the nonlinear transcriptional regulation better than NLDE, GAGA and TSNI. Furthermore, the results of simulation study and Application 2 indicate that SGD-enhanced SA helps infer the transcriptional interactions (w_{ji} 's in Eq. (1)), and results in accurate networks.

Due to intensive computation of systems of nonlinear equations, initial networks inferred by GASA are limited to 10+ genes and factors. Nevertheless, these small networks can be extended to large ones using the following two schemes. First, a large network can be partitioned into a few small networks using biological information, then GASA can be applied to infer each small network. For instance, to infer transcriptional compensation interactions from synthetic sick or lethal (SSL) gene pairs, one can partition a network of 200+ genes in [17] into a few small networks via SSL pairs. Moreover, for inferring transcriptional regulatory interactions, a large network can be divided into small networks centering on several TFs or target genes [4]. Second, a few inferred small networks can be integrated into a large network using a merging scheme. This scheme can be applied iteratively to result in a fairly large network. Below we illustrate how two small gene networks can be merged to a large one using the network in Eq. (2).

Given two small networks, e.g. Γ_1 consisting of the first four equations of Eq. (2) and Γ_2 the last seven equations. Specifically, $\Gamma_1 = \{f_1, f_2, g_1, g_2, g_3, g_4\}$ and $\Gamma_2 = \{f_1, f_2, g_5, g_6, g_7, g_8, g_9, g_{10}, g_{11}, g_{12}, g_{13}\}$. We first calculate the SSE/Var of each gene in the network Γ_1. Next, genes of Γ_2 are added one by one into Γ_1, then SA is applied to estimate parameters of each potential link and calculate its SSE/Var value. The potential link with the highest score of SSE/Var of g_1 and its associated gene is added to Γ_1 (which forms the merged network Γ'_1). Then, calculate the fitness score (AIC or BIC) of Γ'_1 to check network complexity; if Γ'_1 is better than Γ_1, then Γ_1 is updated to Γ'_1. These procedures are repeated for the rest of genes in Γ_1. Similarly, the same procedures can be repeated to integrate the genes of Γ_1 and the associated links to Γ_2. We have conducted an experiment on these small networks Γ_1 and Γ_2 using data simulated from Eq. (2) with no noise and SNR4. The TPRs and TNRs of the merged networks are compatible to those inferred from one whole network, but the FPR of the merged network is a little worse than that inferred from one network; the merging scheme and the results are in Additional file 5.

Conclusions

GASA extended one of the latest advances in nonlinear modeling of transcriptional interactions [11] to: (1) infer cooperative/competitive interactions, and (2) infer genetic networks/transcriptional networks of a few genes simultaneously. In particular, when inferring TFs for 34 yeast cell cycle regulated targets in yeast, GASA has an averaged mTPR about 65% and mFPR 46%, which performs better than NLDE, GAGA and a linear model based algorithm (TSNI). For inferring transcriptional interactions of higher organisms such as 2000+ TFs in human, we can apply biological knowledge, e.g. using differentially expressed genes at one or more time points or/and incorporating experimental conditions, to narrow down the TFs to a few hundreds genes of interest [24], then apply GASA as demonstrated in Application 2. Moreover, several predicted genetic interactions of GASA in the human T-cell subnetwork are consistent with protein-protein interactions in human/mouse JNK signaling/T-cell apoptosis pathways, which suggests that GASA might be applied to predict biochemical pathways. In the simulated networks, GASA identified factors correctly and outperformed NCA, TSNI, GAGA and GA-regular SA. Although the scale of the simultaneous networks, e.g. Eq. (1), inferred by GASA is limited to 10+ genes due to intensive computation, the inferred networks can be expanded either by integration of several small inferred networks into a large one or partitioning a large network into several small ones and then applying GASA as stated in the Discussion section. Recently, other types of genomics data such as ChIP-chip have become available. Incorporating ChIP-chip data to identify a small set of plausible regulators for a target then applying GASA to infer the regulatory/genetic relationship seems promising [25]. This would also allow GASA to infer large networks in addition to improving the prediction accuracy. We leave this avenue for exploration in future research.

Methods

Applying a mean filter to smooth microarray data

To dampen noise in MGED, a mean filter [26] in the discrete signal processing area was applied. A mean filter with kernel size $r \times c$ can be viewed as a window of size $r \times c$ centered at an original datum, and replacing the datum (pixel) with the average of all pixels in the window. See mean_filter.pdf http://www.stat.sinica.edu.tw/~gshieh/GASA/mean_filter.pdf of Supplementary data or Figure 2 of [4] for the effects of mean filters on MGED. In the simulation study of the Results section, mean filters were applied, then the factor-identifying strategy (stated below) found the factors correctly when simulated data have noise.

A strategy to identify factors in a network

Factors are defined as special genes if they regulate (have only outgoing links to) other genes within a closed network of interest. They may have incoming links from genes outside the closed network but not from within. With this definition in mind, if GASA treats the factors as other regulated genes and tries to find their incoming links from genes within the closed network, then this will result in a relatively large *SSE/Var*, which indicates a lack of fit.

Therefore, assuming there is no factor in the closed network, we calculate the smallest *SSE/Var* of each gene as follows. First, we set the maximum number of incoming links of each gene to $l_{\max}$ and enumerate all possible combinations. The smallest *SSE/Var* of a given gene can be obtained by applying the SGD algorithm to enumerate each combination. Then the components that have extremely large values of *SSE/Var* (namely the outliers in the boxplot of all *SSE/Var* values) can be identified as factors; see http://www.stat.sinica.edu.tw/~gshieh/GASA/boxplots.pdf for examples. Moreover, recently there have been several ChIP-chip data sets published. These data can also provide *a priori* information on potential TFs for a given target gene. This data could be used to identify factors before implementing GASA. Integrating ChIP-chip into GASA is a promising future work.

The search space of networks

Finding the optimal genetic network with n genes involves the estimation of $n(n + k + 2)$ parameters in the model of Eq. (1). Ideally, these unknown parameters may be determined using a global optimization technique provided that there is sufficient data. However, it is intractable in practice when n is large since this is NP-complete [27]. The computation time of the task increases exponentially with n.

To circumvent this intractable problem, the problem of finding the optimal fully-connected network in a search space of dimension $n(n+k)$ is reformulated to finding the best network among some optimal partially-connected networks. This is feasible because in general the matrix of connectivity W in Eq. (1) is sparse [19]. There are two sets of unknown parameters: the number of partially-connected networks (structures) and $2n + L(\pi)$ parameters, where $L(\pi)$ denotes the number of links in a given partially-connected network π. Specifically, GASA searches through the space of partially-connected gene networks via a GA and for each fixed structure π, we estimate the associated $2n + L(\pi)$ parameters by an SA algorithm. The optimal gene network is the one that minimizes the cost function. More details of GASA are given in Subsections Parts 1 and 2 of GASA.

Reducing a high dimensional problem to a set of low dimensional problems will substantially save computation time. To reduce the search space of π, we set a limit on the number of incoming links of each gene by $l_{\max}$. Therefore, when determining the combination of incoming links for each gene, the size of search space reduces from 2^{n+k} to $\binom{n+k}{l_{\max}}$. Moreover, when the number of genes is large, many of their cellular networks follow a power law [28], which states that the probability of k interactions (incoming links for each gene) decays according to a negative power of k, namely $P(K = k) \sim k^{-\gamma}$, where $2<\gamma<3$. Therefore, we may further restrict the search space to follow the power law.

GASA Part 1-GA

Given the factors, we implement a GA as follows. First, we encode a network structure by a $n(n + k)$-bit array or a n by $n + k$ bit-matrix, where a 1 at position (i, j) represents a link from g_j to g_i, while a 0 denotes no links. Let the maximum number of links be $l_{\max}$, then the number of bits required to encode a network structure into a chromosome is reduced to $n \times l_{\max}$. In the GA part of GASA, a network structure uses $l_{\max}$ bits to specify the in-degree of each gene, which is the number of 1's in the $l_{\max}$ bits. Therefore, given a network structure or a chromosome, we enumerate all possible sets of incoming links for each gene, and retain the set with the smallest value of $SSE(g_i)$, where

$$SSE(g_i) = \sum_t (g_i(t) - \hat{g}_i(t))^2 \qquad (3)$$

and $\hat{g}_i(t)$ the predicted expression level of gene i at time t with the estimated parameter; the SSE can be obtained after SA (in the subsection GASA Part 2) estimates the parameters. The self-link for each gene is also considered but it is not counted toward the number of incoming links.

Starting with a population of N chromosomes, these chromosomes in a simple GA evolve mimicking natural mechanisms such as selection, crossover and mutation. While selection may be considered as an evolution operator, crossover and mutation serve as genetic operators. In each generation, the evolution process starts with N chromosomes, called parent chromosomes. A fitness score, AIC or BIC score, is first computed for each parent chromosome, where

$$AIC = \sum_i (SSE(g_i) / Var(g_i)) + 2(2n + L(\pi)),$$
$$BIC = \sum_i (SSE(g_i) / Var(g_i)) + \ln(T)(2n + L(\pi))$$

and $Var(g_i)$ is the sample variance of g_i across time.

Next, new chromosomes are generated by the genetic operators. The operator crossover exchanges the same segments of two parent chromosomes to create two new child chromosomes. One-point crossover is implemented in GASA, and crossover is not applied to the end points of each chromosome to avoid inefficient perturbation. These two parent chromosomes are chosen randomly without replacement. On the other hand, the operation mutation perturbs the steady state of each chromosome by randomly toggling very few bits to evolve into new child chromosomes. The probability of being selected for mutation is inversely proportional to the fitness score. Namely, the lower the fitness score of a chromosome, the better chance it will be selected for mutation to reach a higher fitness value.

After the fitness score of each new child chromosome is computed, as in any conventional GA, a new generation of N chromosomes is selected from the pool of parent and child chromosomes. The selection rules vary from one algorithm to another, for instance, fitness-proportional selection, rank-based selection, elitist selection, and others [29]. Our system selects $N/2$ chromosomes with the highest fitness values from the pool of parent chromosomes and from child chromosomes, respectively. As such, GA evolves through generations until the best fitness scores of generations converge.

GASA Part 2: Stochastic Gradient Descent-enhanced SA

From the model in Eq. (1), it is clear that $\hat{g}_i\,(t+1)$ depends on $g_i(t)$'s but not on $\hat{g}_j\,(t)$'s. Thus minimizing $SSE(g_i)$ is independent from minimizing $SSE(g_j)$ when $i \neq j$. For each structure π generated by GA, the parameters will be estimated by an SA gene-wise which is a stochastic search method that incorporates randomness in the search process. SA simulates the annealing process that cools a molten substance into a crystalline solid. Since the quality and stability of a crystalline solid depends on the cooling process, annealing may be regarded as an optimization process searching through the space of the unknown parameters. In an SA, the temperature T decreases from an initial high temperature gradually. At a given temperature T, the probability that the system stays in a particular state s ($p(s, T)$) depends on the free energy $E(s)$, and it follows the Boltzmann distribution [30]

$$p(s,T) = (1/n)\exp(-E(s)/kT),$$

where $n = \sum_{s \in S} \exp(-E(s)/kT)$ and k is the Boltzmann constant.

The main idea of an SA algorithm is to search for the minimum energy at each temperature T, and this is usually achieved by the Metropolis-Hastings (MH) algorithm [31,32]. In the MH algorithm, a move from state s_1 to state s_2 follows the rules that (i) if $E(s_2) < E(s_1)$, then state s_2 is accepted, otherwise (ii) the move is rejected with probability $1\text{-}exp\{[E(s_2) - E(s_1)]/kT\}$.

The rule (ii) distinguishes the SA algorithm from other gradient descent approaches. By allowing the system to move into a state with a higher energy, this SA algorithm inherently can escape from the local minimums. Furthermore, it is shown that the SA achieves the global minimum given a sufficient number of movements. Let π_i denote the incoming links of g_i, $L(\pi_i)$ the number of incoming links of g_i, and $\Theta(\pi_i) = \{\theta_1, \theta_2, \ldots, \theta_{2N+L(\pi_i)}\}$ the parameter vector to be optimized by the SA algorithm. The cost function to be minimized is defined as

$$f(\Theta(\pi_i)) = SSE(g_i)/T,$$

where SSE(g_i) is defined in Eq. (3), and $\Theta(\pi_i)$ denotes a plausible state in the search space. That is, $\Theta(\pi_i)$ is a state vector and $\Theta(\pi_i) = \{\alpha_i, w_{ji}, w'_{ki}, \beta_i\}$, where $j = 1,\ldots, n$ and $k = 1,\ldots,K$. If there are no links from gene j to gene i or from factor k to gene i in π_i, the corresponding w_{ji} or w'_{ki} is fixed at 0 during the optimization process.

To find the optimal set of $\Theta(\pi_i)$, one naive approach is to generate new state vector $\Theta(\pi_i)$ randomly and apply the MH algorithm to determine the movement from one state to another. Although this approach may converge to a global minimum in theory, the computational time may be too intensive to carry out. Alternatively, we propose a stochastic gradient descent (SGD) algorithm to accelerate the convergence process as follows.

```
1. Initialize $\Theta(\pi_i)$
2. Initialize control parameters: $t \leftarrow T_{Max}\ p \leftarrow 1$
3.
Compute-
$\nabla E(\Theta(\pi_i)) = (\partial E(\Theta(\pi_i))/\partial\theta_1, \partial E(\Theta(\pi_i))/\partial\theta_2, \ldots, \partial E(\Theta(\pi_i))/\partial\theta_{2+L(\pi_i)})$

4. Compute $\Theta^{new}(\pi_i)$:
    Generate a uniformly distributed random number in the interval [0, 1], namely $r \sim U(0, 1)$.
    $P_{Gradient} \leftarrow 0.2+0.3t/T_{Max}$
    a. If $r > P_{Gradient}$:
        $\Theta^{new}(\pi_i) \leftarrow \Theta(\pi_i) - \lambda\nabla E(\Theta(\pi_i))$, where $\lambda$ is a damping constant.
    b.
    Else: $\theta_k^{new}(\pi_i) \leftarrow \theta_k - r_k\lambda\partial E(\Theta(\pi_i))/\partial\theta_k$, where $r_k \sim U(0,1)$.

5. Update $\Theta(\pi_i)$:
    If $E(\Theta(\pi_i)) > E(\Theta^{new}(\pi_i))$:
        $\Theta(\pi_i) \leftarrow \Theta^{new}(\pi_i)$
    Else:
        $P_{MH} \leftarrow exp((E(\Theta(\pi_i)) - E(\Theta^{new}(\pi_i)))/Kt)$, where $K$ is a constant.
        Generate a random number $r$ from $U(0,1)$.
        If r $< P_{MH}$:
        $\Theta(\pi_i) \leftarrow \Theta^{new}(\pi_i)$
6. Check for convergence at temperature $t$:
    If the stop condition at temperature $t$ is met:
        $E_t(\Theta(\pi_i)) \leftarrow E(\Theta(\pi_i))$, where $E_t(\Theta(\pi_i))$ is the energy at temperature $t$.
        GOTO step 7
    Else:
        GOTO step 3
7. Check for convergence throughout consecutive $t$'s
    If $t + N_C < T_{Max}$ and $|E_{t+k+1}(\Theta(\pi_i)) - E_{t+k}(\Theta(\pi_i))| < \varepsilon$ for $k = 1, 2,\ldots, N_C$, where $\varepsilon$ and $N_C$ are constants:
        $Ep(\Theta(\pi_i)) \leftarrow E_t(\Theta(\pi_i))$
        $\Theta_p(\pi_i) \leftarrow \Theta(\pi_i)$
        $p \leftarrow p + 1$
    Else:
        $t \leftarrow t - 1$
        GOTO step 3
8. Check the effect of perturbing network parameters $best \leftarrow \arg\min_k \Theta_k(\pi_i)$
```

```
If $p > N_F + 1$ and $E(\Theta_k(\pi_i)) > E(\Theta_{p-1-F}(\pi_i))$ for $k = p-F, p-F+1, ..., p-1$, where $N_F$ is a constant:
    SGD completed, outputting $\Theta_{best}(\pi_i)$
Else:
    $\Theta(\pi_i) \leftarrow \{r_k\theta_k | \theta_k$ in $\Theta_{best}(\pi_i)\}$, where $r_k \sim U(0.75, 1.25)$
    $t \leftarrow T_{Max}$
    GOTO step 3
```

In the SGD algorithm, step 3 is to derive the direction of the steepest gradient descent and step 4a is to update the parameter vector along that direction, which in general leads to a faster convergence. However, the direction of the steepest gradient descent may be trapped in a local minimum. To remedy this drawback, in step 4b we introduce randomness into the converging direction so that the global minimum may be achieved by the second rule of the MH algorithm in step 5. Step 7 checks the absolute difference of energy between pairs of two adjacent temperature points. If the difference is smaller than a constant ε for N_C successive pairs, then the optimization process converges. To avoid converging to a local minimum, the algorithm repeatedly perturbs the best $\Theta(\pi_i)$ obtained, and iterates the optimization process until no improvements on $E(\Theta(\pi_i))$ can be made for N_F times.

A preliminary version of this article was accepted by IEEE, BMEI conference proceedings, 2009.

Additional file 1: Simulation_results_signs_checked.pdf. Performances of GASA, TSNI, NCA, GAGA and GA-regular SA applied to data simulated from Eq. (2) with no, low to high level of noise, where signs of interactions were counted.
Click here for file
[http://www.biomedcentral.com/content/supplementary/1752-0509-4-16-S1.pdf]

Additional file 2: Fig_predicted_network.pdf. Predicted networks and the true one presented in Table 1, 2 and 3 from one experiment.
Click here for file
[http://www.biomedcentral.com/content/supplementary/1752-0509-4-16-S2.pdf]

Additional file 3: Implementation_TSNI_GAGA.pdf. Detailed procedures of the implementations of TSNI and GAGA.
Click here for file
[http://www.biomedcentral.com/content/supplementary/1752-0509-4-16-S3.pdf]

Additional file 4: Data-preprocessing.pdf. A detailed description of data pre-processing of Application 2.
Click here for file
[http://www.biomedcentral.com/content/supplementary/1752-0509-4-16-S4.pdf]

Additional file 5: Merging.pdf. Detailed procedures of the network merging scheme and the preliminary results of an experiment.
Click here for file
[http://www.biomedcentral.com/content/supplementary/1752-0509-4-16-S5.pdf]

Acknowledgements
We wish to thank Dr Yu-Chi Lu for writing the factor identifying code. We are indebted to an AE and two anonymous reviewers' invaluable comments that improved our manuscript greatly. This research was partially supported by National Research Council, ROC grants NSC 95-2221-E-002-029-MY2 (for CMC), and NSC 98-2118-M-001-017 and NSC 99-3112-B-001-015 (for GSS).

Author details
[1]Institute of Biomedical Engineering, National Taiwan University, No 1, Sec 4, Roosevelt Road, Taipei, 106, Taiwan. [2]Institute of Statistical Science, Academia Sinica, No 128, Sec 2, Academia Road, Taipei 115, Taiwan.

Authors' contributions
CMC and CL devised the method. CL, CLC and CC implemented the method. GSS conceived the research. CMC and GSS supervised the methodology and implementation, and wrote the manuscript. All of the authors read and approved the final manuscript.

Received: 1 July 2009 Accepted: 26 February 2010
Published: 26 February 2010

References

1. Vogelstein B, Kinzler KW: Cancer genes and the pathways they control. *Nature Medicine* 2004, **10**:789-799.
2. Wood LD, Williams Parsons D, Jones S, Lin J, Sjöblom T, Leary RJ, Shen D, Boca SM, Barber T, Ptak J, Silliman N, Szabo S, Dezso Z, Ustyankshy V, Nikolskaya T, Nikolsky Y, Karchin R, Wilson PA, Kaminker JS, Zhang Z, Croshaw R, Willis J, Dawson D, Shipitsin M, Willson JKV, Sukumar S, Polyak K, Park BH, Pethiyagoda CL, Krishna Pant PV, Ballinger DG, Sparks AB, Hartigan J, Smith DR, Suh E, Papadopoulos N, Buckhaults P, Markowitz SD, Parmigiani G, Kinzler KW, Velculescu VE, Vogelstein B: The Genomic Landscapes of Human Breast and Colorectal Cancers. *Science* 2007, **318**:1108-1113.
3. Shieh GS, Chen CM, Yu CY, Huang J, Wang WF, Lo YC: Inferring transcriptional compensation interactions in yeast via stepwise structure equation modeling. *BMC Bioinformatics* 2008, **9**:134.
4. Chuang CL, Jen CH, Chen CM, Shieh GS: A pattern recognition approach to infer time-lagged genetic interactions. *Bioinformatics* 2008, **24**:1183-1190.
5. Tegner J, Yeung MKS, Hasty J, Collins JJ: Reverse engineering gene networks - Integrating genetic perturbations with dynamical modeling. *Proceedings of the National Academy of Sciences USA* 2003, **100**:5944-5949.
6. Bansal M, Gatta GD, Bernardo D: Inference of gene regulatory networks and compound mode of action from time course gene expression profiles. *Bioinformatics* 2006, **22**:815-822.
7. Liao JC, Boscolo R, Yang YL, Tran LM, Sabatti C, Roychowdhury VP: Network component analysis: Reconstruction of regulatory signals in biological systems. *Proc Natl Acad Sci USA* 2003, **100**:15522-15527.
8. Wu X, Ye Y, Sybramanian KR: Interactive analysis of gene interactions using graphical Gaussian model. *Proceedings of the ACM SIGKDD Workshop on Data Mining in Bioinformatics: 26 August, 2001; San Francisco* Toivonen: Springer-VerlagMohammed J Zaki, Jason TL Wang, Hannu TT 2001, **3**:63-69.
9. Shieh GS, Jiang YC, Hung YC, Wang TF: A regression approach to reconstruct gene networks. *Proceedings of Taipei Symposium on Statistical Genomics: 15-18 December, Taipei* 2004, 357-370.
10. Climescu-Haulica A, Quirk MD: A stochastic differential equation model for transcriptional regulatory networks. *BMC Bioinformatics* 2007, **8(Suppl 5)**:S4.
11. Vu TT, Vohradsky J: Nonlinear differential equation model for quantification of transcriptional regulation applied to microarray data of *Saccharomyces cerevisiae*. *Nucleic Acids Res* 2007, **35**:279-287.
12. Chen KC, Wang TY, Tseng HH, Huang CY, Kao CY: A stochastic differential equation model for quantifying transcriptional regulatory network in. *Saccharomyces cerevisiae* 2005, **21**:2883-2890.
13. Kikuchi S, Tominaga D, Arita M, Takahashi K, Tomita M: Dynamic modeling of genetic net works using genetic algorithm and S-system. *Bioinformatics* 2003, **19**:643-650.

14. Liu PK, Wang FS: **Inference of biochemical network models in S-system using multiobjective optimization approach.** *Bioinformatics* 2008, **24**:1085-1092.
15. Chen CM, Chang CF, Lee C, Shieh GS: **Evaluating Genetic Networks Reconstruction by Simulated Microarray Data.** *Technical Report 04-02* Institute of Statistical Science, Academia Sinica, Taiwan 2004.
16. Wray GA, Hahn MW, Abouheif E, Balhoff JP, Pizer M, Rockman MV, Romano LA: **The evolution of transcriptional regulation in Eukaryotes.** *Mol Biol Evol* 2003, **20**:1377-1419.
17. Tong AH, Evangelista M, Parsons AB, Xu H, Bader GD, Page N, Robinson M, Raghibizadeh S, Hogue CW, Bussey H, Andrews B, Tyers M, Boone C: **Systematic genetic analysis with ordered arrays of Yeast deletion mutants.** *Science* 2001, **294**:2364-2366.
18. Spellman PT, Sherlock G, Zhang MQ, Iyer VR, Anders K, Eisen MB, Brown PO, Botstein D, Futcher B: **Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization.** *Mol Biol Cell* 1998, **9**:3273-3297.
19. Van Someren EP, Wessels LF, Backer E, Reinders MJ: **Genetic network modeling.** *Pharmacogenomics* 2002, **3**:507-525.
20. Yazgan O, Pfarr CM: **Regulation of two JunD isoforms by Jun N-terminal kinases.** *J Biol Chem* 2002, **277**:29710-29718.
21. Rangel C, Angus J, Ghahramani Z, Lioumi M, Sotheran E, Gaiba A, Wild DL, Falciani F: **Modelling T-cell activation using gene expression profiling and state space models.** *Bioinformatics* 2004, **20**:1361-1372.
22. Ashkenazi A, Dixit VM: **Death receptors: Signaling and Modulation.** *Science* 1998, **281**:1305-1308.
23. Xie J, Bentler PM: **Covariance structure models for gene expression microarray data.** *Structural Equation Modeling* 2003, **10**:566-582.
24. Chuang CL, Wu JH, Cheng CS, Shieh GS: **WebPARE: Web-computing for inferring genetic or transcriptional interactions.** *Bioinformatics* 2010, **26**:582-584.
25. Chuang CL, Hung K, Chen CM, Shieh GS: **Uncovering transcriptional interactions via an adaptive fuzzy logic approach.** *BMC Bioinformatics* 2009, **10**:400.
26. Gonzalez RC, Woods RE: *Digital Image Processing* New Jersey: Prentice Hall, 2 2002.
27. Garey MR, Johnson DS: *Computers and Intractability: A Guide to the Theory of NP-Completeness* New York: W. H. Freemann 1979.
28. Jeong H, Tombor B, Albert R, Oltvai ZN, Barabási AL: **The large-scale organization of metabolic networks.** *Nature* 2000, **407**:651-654.
29. Jacob C: **Evolution and coevolution of developmental programs.** *Computer Physics Communications* 1999, **121-122**:46-50.
30. Aarts E, Korst J: *Simulated Annealing and Boltzmann Machines* Chichester: John Wiley & Sons 1989.
31. Metropolis N, Rosenbluth A, Rosenbluth M, Teller A, Teller E: **Equation of State Calculations by Fast Computing Machines.** *J Chem Phys* 1953, **21**:1087-1092.
32. Hastings WK: **Monte Carlo sampling methods using Markov chains and their applications.** *Biometrika* 1970, **57**:97-109.

doi:10.1186/1752-0509-4-16
Cite this article as: Chen *et al.*: **Inferring genetic interactions via a nonlinear model and an optimization algorithm.** *BMC Systems Biology* 2010 **4**:16.