GNS: A Tool for the Analysis of Gene Regulatory Networks

Georgios Ch. Sirakoulis[1], Demetrios Soudris[1], Ioannis Andreadis[1], and Ioannis Karafyllidis[1]

[1] Democritus University of Thrace, Department of Electrical and Computer Engineering, GR 67100 Xanthi, Greece
{gsirak, dsoudris, iandread, ykar}@ee.duth.gr

Abstract

In order to understand the functioning of organisms at the molecular level, we need to know the genes which are expressed in the organism (when, where and how). The regulation of gene expression is achieved through gene regulatory networks of interactions between DNA, RNA, proteins, and small molecules. As most gene regulatory networks of interest involve many components connected through positive and negative feedback interlocking loops, an intuitive understanding of their dynamics is hard to obtain. As a consequence, computer tools for the study of genetic regulatory networks will be valuable. In this paper, we present a computational tool for gene regulatory networks. Our results demonstrate the utility of the specific computational tool in the quantitative analysis of the gene regulatory networks. Finally, the proposed tool allows the user to change several of its parameters, in order to study various hypotheses concerning the analysis of the genetic network under consideration.

Keywords: Gene Regulatory Networks, Computational Tool, Simulation, Analysis

1. Introduction

Gene networks are increasingly used as models to represent phenomena at the level of gene expression, and research in their construction from experimental data is rife. The gene network model has several applications and advantages over other approaches:

Gene networks provide a large-scale, coarse-grained view of the physiological state of an organism at the mRNA level. The mRNA phenotype can be a very important representation of cell function, offering a much more precise description than the achieved with words. For instance, if the gene of a certain protein kinase is linked to genes involved in synthesis of a flagellum, one could conclude that it has a role on the chemotaxis signal transduction pathway. In this sense, gene networks (and especially their graphical representations) are not only capable of describing a large number of interactions in a concise way, but also they can represent, at a systems level, the dynamic properties underlying these interactions. Cells exhibit complex interacting behaviour that is usually not predictable from the properties of individual system components alone. Gene networks provide such a system view at the level of gene activities [de la Fuente et al. (2002)]. Furthermore, gene networks are useful to rationalise phenomena in terms of how external perturbations propagate through the expression of genes. As a result, gene networks could be used to annotate functions in genomics because these networks describe in unambiguous ways the processes each gene is involved. Taking into account the progress in gene-expression profiling, elucidating gene networks is an appropriate and timely step on the way to uncovering the complete biochemical network of a cell type. Starting from a high-level description of gene regulation in cells provided by the gene network, one could systematically add details of the mechanism of physical interaction and expand the network to include explicitly proteins and metabolites. Computational tools can considerably aid in the process in several specific ways [Brown et al. (2002)].

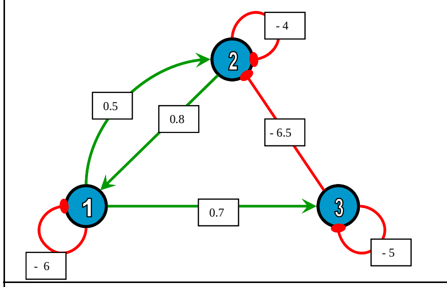
In this paper, we present a simulation tool for modelling gene networks, namely Gene Networks Simulator (*GNS*). The proposed *GNS* based on data entries that is related with the dynamics of genetic network, calculates the changes of concentrations corresponding in the genes of network of proteins as interrelation of time and presents the corresponding graphic representations, respectively. The above characteristics, in combination with easy and direct access in the data entries, render *GNS* as a useful tool for the study and analysis of dynamics of various genetic networks. Furthermore, the presented computational tool was designed and developed as an interactive tool that offers automated modelling with the assistance of a dynamic and user friendly graphical environment. *GNS* user interface has been implemented using Matlab® enabling platform independence and possible cooperation with other known simulation tools aiming at adaptive reverse engineering of gene regulatory networks based also on Matlab®, like the one developed earlier by the authors [Mamakou et al. (2005)]. As a result, *GNS* system targets successfully not only on the analysis of complicated gene networks, but also on their quantitative simulation with the help of the annotated computational tools. In such a case, the proposed computational tool would be able to handle successfully some real data experiments, and to serve as a powerful “virtual lab” dedicated to the modelling of the genetic networks.

2. Gene Networks Analysis

The detailed molecular mechanisms of how the products of one gene affect expression of another gene are often unknown but the effect itself can be easily observed in gene-expression experiments. It is therefore appropriate to use genome-wide gene-expression data to identify gene networks, an important step towards uncovering the complete biochemical networks of cells. Research focused on developing methods for this identification of gene networks from microarray data is now an important part of bioinformatics.

Cellular responses and actions are often a result of coordinated activity of a group of genes. Gene networks might allow genes to be ranked according to their importance in controlling and regulating cellular events. There is a growing indication that most single-gene mutations do not have marked phenotypes (most genes in genomes are not of ‘known function’). Most phenotypes are the result of a collective response of a group of genes. Gene networks help rationalise how these complex traits arise and which groups of genes are responsible for them.

As mentioned above, gene networks are models that display causal relationships between gene activities, usually at the mRNA level, and are commonly represented by directed graphs (Figure 1). The nodes of the graph are genes and the directed edges are causal relationships between genes. A widely adopted norm is to use arrow tips on edges to represent positive interactions, where an increase in activity of the originating gene causes an increase in the target gene, and bars on edges to represent negative interactions, where an increase in activity of the originating gene causes a decrease inactivity of the target gene. Gene networks can also be represented through matrices [Figure 1(b)].



(a)

	G1	G2	G3
G1	-6	0,5	0,7
G2	0,8	-4	0
G3	0	-6,5	-5

(b)

Figure 1. *(a) Graph representation of a random gene network with three genes. (b) Matrix representation of the same gene network*

The matrix in Figure 1(b) was obtained by applying regulatory strengths to simulated data [de la Fuente et al. (2002)]. Each column and row of the matrix represents one gene, and the matrix elements represent causal relationships. These matrices qualitatively, indicate positive interactions, represented by positive number, negative interactions, represented by negative number, and no interaction between genes represented by number 0. Matrices are also well suited for quantitative representations. In this case its elements take real values representing the strength and sign of the interaction. Graph representations can also express quantitative values, which are expressed with real numbers next to the edges to indicate the strength of the interaction.

An interaction between two genes is said to be direct if it does not run through any other genes in the network. For example, in Figure 1(a), gene 1 directly affects gene

2. Gene 1 also affects gene 4, but only in an indirect way because the effect has to run through gene 2. Non-additive interactions are those that require the simultaneous action of two or more genes (i.e. when each of them alone has no effect and only together do they become a cause).

Research in gene networks has been geared towards two major goals: first, to understand the dynamics and design principles of gene regulation; and second, to reverse engineer gene networks from experimental measurements. Activities started with the pioneering work of Kauffman (1969) on random Boolean (gene) networks. More recently, the assumption that the topology of gene networks is random has been called into question, as more convincing arguments indicate that gene networks follow a ‘small-world’ topology with a power law distribution for node connectivities [Barabási and Albert (1999)].

Experimental data of mRNA levels obtained with the use of high-throughput technologies is now abundant. These are snapshots of the molecular state of cell populations at the transcript level and are rich in information about gene networks. This process of establishing cause–effect relationships between genes on the basis of observed expression levels is referred to as ‘reverse engineering’. Several approaches have been proposed for inferring gene networks from experimental data. A popular method used for gene-expression data analysis, sometimes called ‘guilt by association’, assumes that genes with similar expression patterns are functionally related to each other. Other methods are based on more sophisticated statistical analysis, including Bayesian belief networks and pair-wise correlation methods. Several methods exist that rely on the simplification of considering genes to be either expressed at a fixed rate, or not at all [Kauffman (1969)]. These methods also consider time to be a discrete process, and have rules that govern whether genes are on or off at a given time step, based on the values taken by the genes at the previous time step. Boolean approaches suffer from their inability to capture intermediate levels of gene expression, and can easily generate spurious results owing to their discrete nature [Sirakoulis et al. (2006)].

More challenging, but potentially more accurate representations of gene networks use continuous functions, in which expression levels are allowed to take any positive value. These approaches are mathematically implemented by difference or differential equations, either linear or nonlinear. In linear additive models, each interaction is characterized by one parameter that is positive for activation, negative for inhibition, or zero for no interaction. More realistic, but also more difficult approaches use nonlinear kinetics to represent the rates of transcription, such as neural network-like sigmoidal functions [Wahde and Hertz (2000)]. In both cases, nonlinear optimization methods are used to fit the model equations to the observed data. The use of nonlinear kinetics is cumbersome compared to linear or Boolean methods since it requires larger amounts of data. However, it has the advantage of much greater predictive power.

A graph theoretical approach has been proposed to analyse gene-expression data obtained from null mutants [Wagner (2001)]. This method is promising because it uses the most abundant type of data currently available. Unfortunately, it would not be possible to distinguish between different gene networks of the same class, so the most parsimonious network must be adopted [de la Fuente et al. (2004)]. However, evidence from molecular biology suggests that the underlying gene networks will not necessarily be parsimonious. In addition, this approach is only applicable to acyclic graphs. Finally, a method based on systematic perturbation of gene transcription rates and microarray measurements to infer the underlying gene networks was proposed [de la Fuente et al. (2002)]. The method itself is based on developments from metabolic control analysis, particularly co-response analysis and regulatory strengths. Briefly, the aforementioned method is capable of identifying and quantifying direct interactions between genes, requiring several experiments equal to the number of genes considered in the network analysis.

A conclusion drawn from the above discussion is that many more experiments are needed to infer gene networks with high accuracy. Furthermore, it is obvious that gene networks are also a good way to describe function unequivocally, and that they could be used for genome functional annotation; and finally, the ability to create gene networks from experimental data and use them to reason about their dynamics and design principles will increase our understanding of cellular function. It is now clear that the investigation, analysis and simulation of gene regulatory networks requires computational tools specific to the task.

3. The Proposed Simulation Tool

The proposed computational tool, namely *GNS*, can be used for gene network simulation comprising the computational inference of pathway interactions and network logical organization directly from gene expression data as given. *GNS* user interface has been implemented using Matlab® GUI facilities, enabling interactive simulation. This interface is shown in Figure 2.

In more details, *GNS* analyses gene networks, presenting the change of proteins concentrations that are produced at the transcription and translation of the examined genes, as time depended. Furthermore, beyond the graphical presentation of the time dependence of the proteins concentration, *GNS* presents also in its output an analytical table of the concentration values of all proteins at each time step in order to provide a quantitative representation of the gene network under question. To accomplish network analysis, four different inputs are needed by the *GNS* user. More specifically, these inputs are: An one dimensional vector representing the initial concentrations of the proteins corresponding to the genes of the gene network under question; A vector that would include all the initial instant rates of the concentration changes of the proteins corresponding to their initial concentrations; A correlation matrix identical to the one depictured in Figure 1(b) that would indicate the dependence weights among the genes; and, finally, a matrix of thresholds that would comprise the proteins concentration values above which the effect of each gene (in the protein of which corresponds this concentration) on an other gene is expressed. It is evident that the threshold and correlation matrixes foreknowledge results in the precise knowledge of the gene network itself. In order to proceed with reverse engineering, the user of the simulation tool should have already composed and restructured the examined gene network with the help of a reverse engineering tool such as the one presented in [Mamakou et al. (2005)]. The aforementioned computational tool, based on Genetic Algorithms (GAs), is able to predict with observed data the regulatory pathways that are represented as influence matrix.

In previous Sec., the gene networks were represented through cross-correlation matrices. Each element (*i*, *j*) of this matrix annotates the dependence of gene *j* from gene *i*. In the presented tool, these elements constitute, not only simple metric of genes dependence, but also the rate of change of gene j due to the effect of gene *i*. This speed has units of concentration per time. In general, the rate of the concentration change for each protein depends on the algebraic sum of the corresponding column elements found in the correlation matrix. However, the way the elements are chosen for this sum, is defined directly by the thresholds matrix. As mentioned before, in correlation matrix reference, the concentration of protein of gene *j* is altered due to the effect of gene *i*. This actually happens when the concentration of protein of gene *i* exceeds some limit or threshold. Only then the effect of gene *i* is considered in the total rate of change of protein *j* concentration based on the (*i*, *j*) couple. In the model that was developed in the frame of this work, the effect of gene *i* begins at the moment when the corresponding threshold is exceeded and remains active, even if the concentration of protein of gene *i* results in values lower than the aforementioned threshold. This effect terminates only when the concentration of protein of gene *j* fades to zero.

In order to produce the vector that would include all the initial instant rates of the concentration changes of the proteins corresponding to their initial concentrations, the user of the system should proceed with two concentration measurements. These two measurements should be taken in brief time so that the possibility of thresholds extension during these measurements would be negligible. In this way it would be rather easy to compute the average rate of change of each protein concentration during the two measurements. This average rate would be equal to the rate of change at the last, second, measurement. Finally, the following steps should be followed in order to establish high accuracy concentration measurements. After measuring the concentrations of proteins C_i at time moment 0 (first measurement), the differences $T_{ij}–C_i$ are calculated (where T_{ij} are the elements of the thresholds matrix) and as a result the minimal differences for each protein are assigned. Furthermore, for each protein, all the positive elements of the corresponding column of cross-correlations matrix are added in order to find the biggest theoretical positive rate that can be assigned in each protein. Then, the minimal differences are divided by the corresponding biggest rates, and, therefore, result in the minimal theoretical times of threshold achievement for each protein. The smallest one of these moments will be the theoretical global minimum time of threshold achievement of the examined network (and consequently the worst case). Finally, the time that would result between the two measurements, should be in any case smaller than this minimal theoretical time so as to eliminate the existence of some intermediate threshold.

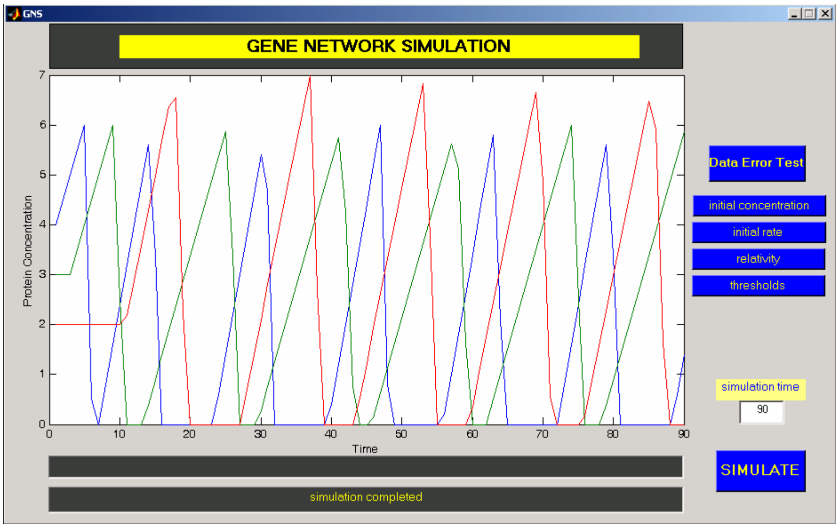


Figure 2. *Graphical representation of the GNS simulation results*

The user of *GNS* is able to insert and change any one or all of the above input parameters until the *GNS* better depicts the examined network. More specifically, by pressing the “Initial Concentration” button the user inserts the aforementioned initial concentrations of the proteins corresponding to the genes of the gene network chromosome. The user can enter his/hers initial instant speeds of the concentration changes of the proteins corresponding to their initial concentrations by clicking on the “Initial rate” button and entering the sequence of speed matrix as he/she desires to the corresponding matrix. With the usage of button “Relative” the user enters his/hers correlation matrix that would indicate the dependence weights between the examined genes and by clicking on “Thresholds” button he/she can insert the aforementioned thresholds matrix of the examined network. More information about the suitability of the form of the entered matrixes can be found in the empty fields just below the axes of the graphical representation. Finally, he/she can set simulation time by entering her/his choice in the field “Simulation time”. The simulation time should be in the same units as found in the rates of concentration changes. After setting all the simulation parameters, the computational tool is activated by clicking the button

“Simulate” on the bottom-right of the interface. Figure 2 shows simulation results for the gene network of Figure 1(a).

The aforementioned *GNS* simulation results were produced with the help of an algorithm for the computation of proteins concentrations. More specifically, the scope of this algorithm is to compute the proteins concentrations changes corresponding to the network genes during the simulation time. Although simulation time is discrete, the computation is continuous. Between each two simulation time steps, a time interval is setting. The algorithm is able at the end of this interval to compute the vector of concentrations as well as the vector of rate variations of the corresponding concentrations. At each time step, the algorithm checks out if some proteins due to their rate change, overshoot their thresholds. The time needed for a protein to overshoot the corresponding threshold results from the following equation:

$$\textit{Time to threshold} = (\textit{threshold} - \textit{present concentration}) / \textit{concentration rate} \qquad \textbf{(1)}$$

From all the proteins that will succeed to overshoot the corresponding threshold the first one to succeed will be responsible for the change of the rate vector. It is evident that due to the aforementioned change, the rest of the proteins concentrations should be recalculated to determine if they still overshoot the corresponding thresholds or probably some new ones, not close enough at the previous time steps, are able to do so.

Figure 3(a) shows simulation results produced by the *GNS* for a gene network of five genes. It is obvious that in the presented example most of the genes are by themselves suppressed. Most genes in gene networks will have a negative effect on their own concentration because the degradation rate of their mRNA is proportional to its concentration. The simulation results produced by the proposed tool after 80 time steps are shown in Figure 3(b).

In order to establish the ability of *GNS* tool to interact instantly with the slightest differences in its input we present another example. In this case, a slightly different gene network is shown in Figure 3(c). As shown in Figure 3(a) the auto calibration of gene numbered 1 in order to suppress itself the corresponding protein produced by it, is missing. As a result the expression of gene 1 is directly related to gene 2. The only difference between gene networks (a) and (c) is induced by the lack of any possible influence of gene 2 to the expression of gene 1. All other nets parameters, meaning their input matrices as given above, remain exactly the same. As shown in Figure 3(d), the examined gene network is rather unstable because of the continuous increment in protein 1 concentration. Of course, the same could result if there the examined network was the same as before, but some other changes were appeared in user inputs, i.e. a slight modification of the thresholds matrix regarding gene 2 auto suppression threshold value.

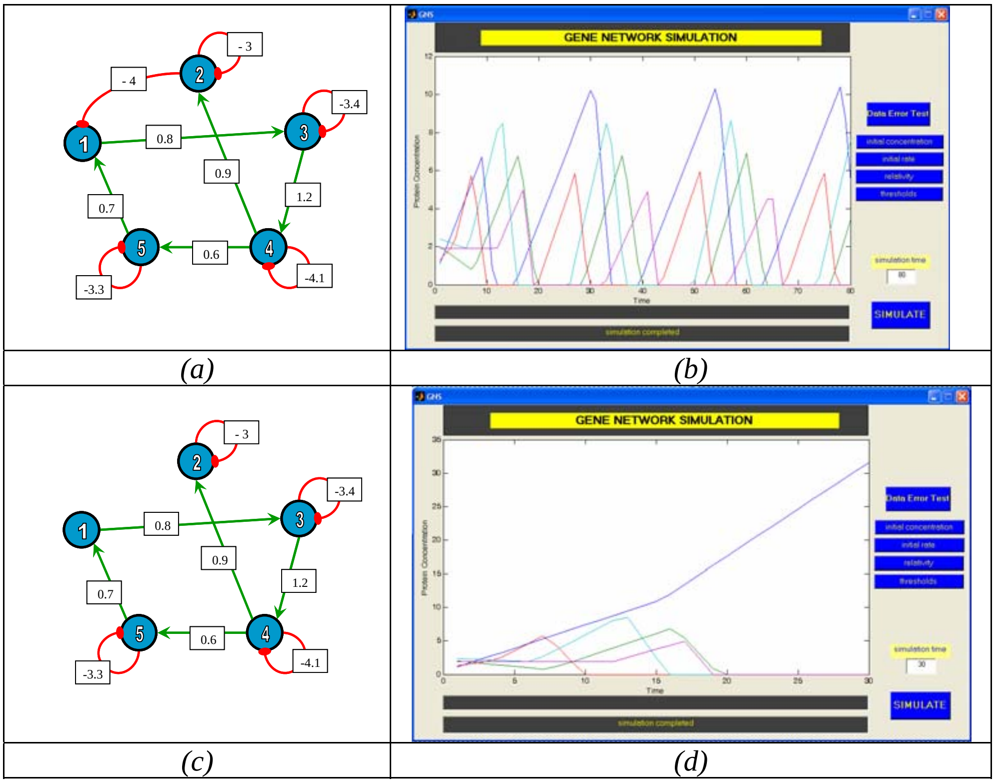


Figure 3. *(a) Random gene networks of five genes. (b) Simulation results of GNS tool after 80 time steps. (c) The same random gene network of five genes without the suppression arc between genes 1 and 2. (d) Simulation results of GNS tool after 30 time steps that show the instability of the examined gene network.*

4. Conclusions

Gene networks are phenomenological models of how changing activity of genes affects the activity of other genes. Knowledge about gene networks might provide valuable information and lead to new ideas for treating complex diseases. Taking into account the progress in gene-expression profiling, elucidating gene networks is an appropriate and timely step on the way to uncovering the complete biochemical network of a cell type. On the other hand, as most gene regulatory networks of interest involve many components connected through interlocking positive and negative feedback loops; an intuitive understanding of their dynamics is hard to obtain. As a consequence, computer tools for the study of genetic regulatory networks will be indispensable. In this paper we presented a computational tool, namely *GNS*, for the quantitative analysis of the gene regulatory networks. The proposed tool has an automated user-friendly interface and enables the user to change several of its parameters, in order to study various hypotheses concerning the simulation of the genetic network under consideration. Furthermore, the efficiency of *GNS* increases because of its high compatibility with previously developed computational tools able to compose and reconstruct gene networks. These gene network reverse engineering tools comprise the computational inference of pathway interactions and network logical organization directly from gene expression data (e.g. RNA and protein concentrations) exhibited by the organism or biological system of interest. As a result, the proposed computational tool would be able to handle successfully some real data experiments, and to serve as a powerful "virtual lab" dedicated to the modelling of the genetic networks.

Acknowledgements

This work was partially supported by the project PYTHAGORAS-II, which is funded by the Ministry of Education in the context of EPEAEK Framework. The authors would thank Konstantinos Paltoglou for his helpful feedback during simulations.

References

Barabási A.L. and Albert R. (1999). Emergence of scaling in random networks. Science vol. 286, 509-512.

Brown C.T. *et al.* (2002). New Computational Approaches for Analysis of cis-Regulatory Networks. Developmental Biology vol. 246, 86-102.

de la Fuente A., Brazhnik P. and Mendes, P. (2002). Linking the genes: inferring quantitative gene networks from microarray data. Trends in Genetics vol. 18, 395-398.

de la Fuente A., Brazhnik P. and Mendes P. (2004). Regulatory Strength Analysis for Inferring Gene Networks. In: Kholodenko, B.N., Westerhoff, H.V. (eds.): Metabolic Engineering in the Post Genomic Era. Horizon Scientific Press, Norwich, UK.

Kauffman S. (1969). Homeostasis and differentiation in random genetic control networks. Nature 224:177.

Mamakou M., Sirakoulis G.Ch., Andreadis I., Karafyllidis I. (2005): Adaptive Reverse Engineering of Gene Regulatory Networks using Genetic Algorithms. In: Proceedings of IEEE International Conference on "Computer as a tool" (EUROCON'2005), IEEE, New Jersey, pp. 401–404.

Sirakoulis G.Ch., Soudris D., Karafyllidis I., Andreadis I. (2006). Modelling Gene Regulatory Networks Using Cellular Automata. In: Proceedings of *5th* European Symposium on Biomedical Engineering (ESBME 2006), Patra, Greece, P.55:1–4.

Wahde M. and Hertz J. (2000). Coarse-grained reverse engineering of genetic regulatory networks. Biosystems vol. 55, 129–136.

Wagner A. (2001). How to reconstruct a large genetic network from n gene perturbations in fewer than n^2 easy steps. Bioinformatics vol. 17, 1183–1197.