3

Toward Personalized Therapy Using Artificial Intelligence Tools to Understand and Control Drug Gene Networks

Alexandru G. Floares
SAIA - Solutions of Artificial Intelligence Applications;
IOCN - Oncological Institute Cluj-Napoca
Romania

1. Introduction

The real implementation of individualized therapy and gene therapy of diseases, which are most often multi-gene disorders, is an important goal of modern personalized medicine, but needs a solid rational foundation. The deluge of complex, high-dimensional biomedical data is continuously increasing; however, our modeling capacity is much smaller and increasing only slowly - particularly in fields using high-throughput techniques such as genomics, transcriptomics, proteomics, and pharmacogenomics. Knowing which genes are expressed, when, where, and to what extent is important for understanding organisms, as well as for controlling genes through adequate drugs and dosage regimens development. The regulation of gene expression is achieved through complex regulatory systems—gene regulatory networks (GRNs) - which are networks of interactions among DNA, RNA, proteins, and small molecules.

Let us remark that not only a key ingredient but a whole dimension is missing from this view. A large variety of external molecular species interfere with gene networks, but we will focus only on drugs, drug discovery being one of the important routes to personalized medicine. A more general concept of drug gene regulatory networks (DGRN), or simply drug gene networks (DGN), first introduced in (Floares, 2007b), is presented together with some mathematically definitions.

Besides the high-throughput experimental approaches, allowing to simultaneously monitor thousands of genes or other molecular species, mathematical modeling is essential for understanding and controlling gene networks by drugs or gene replacements. Various formalisms, such as Bayesian networks, Boolean networks, differential equation models, qualitative differential equations, stochastic equations, and rule-based systems, have been used (see (Jong, 2002); (Gardner & Faith, 2005); (Bansal, 2007) for reviews).

The ordinary differential equations (ODE) approach tries to elucidate a deeper understanding of the exact nature of the regulatory circuits and their regulation mechanisms. In a pharmacogenomic context, it allows the design of controls that are optimal, individualized drug dosage regimens (Floares, 2005); (Floares, 2006). Unfortunately, this is also the most difficult, tedious, expensive, and time-consuming approach. The models are high-dimensional systems of nonlinear-coupled stiff ODEs. The number of parameters is extremely large, and many of them have unknown values. Although in principle one can find the best set of parameter values by sampling the whole parameter space, many degenerate solutions may be expected. These are due to the correlations between parameters, and to the fact that biological systems have built-in regulation mechanisms that make them robust to changes in many of their parameter values. These facts suggest that it is the network structure rather than the precise value of the parameters that confers stability to the system.

There is a need for algorithms to automatically infer such models from high-throughput time-series data, and artificial intelligence is better suited than conventional modeling. We proposed a series of reverse engineering algorithms for drug gene networks (Floares, 2007b), based on artificial intelligence methods - neural networks (NN) for identification and control, and linear genetic programming (GP) (M. Brameier & W. Banzhaf, 2007) for symbolic regression. The algorithms take as inputs high-throughput (e.g., microarray) time-series data and automatically infer an accurate ordinary differential equations model, revealing the networks structure and parameter and giving insights into the molecular mechanisms involved.

RODES algorithms, from reversing ordinary differential equations systems, decouple the systems of differential equations, reducing the problem to that of revere engineering individual algebraic equations. Using GP involve evaluating the fitness of hundreds of models (computer programs) every generation, in the simulated evolution. Our approach drastically reduces the complexity of the problem and the execution time, because for evaluating the fitness function is not necessary to integrate the ODE system. In addition, the possibility of incorporating common domain knowledge in RODES reduces the structure search space and further speeds up the algorithms.

Other studies, proposing GRN reverse-engineering algorithms based on evolutionary computation, require integration of the ODE systems hundreds or thousands of times for each generation ((Sakamoto & Iba, 2001); (Kikuchi et al., 2003); (Noman & Iba, 2005); (Cho et al., 2006)). Similar methods have been proposed in the past, but most of them require a predefined model structure, however, and are limited to parameter estimation. For example, the S-system model refers to a particular type of ODE system in which the component processes are power-law functions (Savageau, 1976) (Voit, 2000). Despite the elegance and computational simplicity of the S-system model, this formalism has its limitations for biochemical networks (e.g., (Beard, 2004)).

Usually, due to various experimental constraints, essential information is missing from data, and even the most powerful artificial intelligence techniques are not creating information, but just extracting it from data. Missing information from data is an important data mining and artificial intelligence problem, by no means restricted to the problem investigated here. To our knowledge, the problem of missing information from data, in the form of variables or features missing, does not have adequate technical solutions in the data mining and computational intelligence literature. In the present context, not all variables or time-series are simultaneously measured, as it is required to reconstruct the drug gene networks, as systems of ordinary differential equations. RODES algorithms can reveal if some information from the input set is either missing or not related to the output. This is possible because the genetic programming version of RODES (GP RODES) requires the temporal series of all variables of the system to infer an accurate mechanistic model. It also means that it does not discover false input–output relations.

One of the unique features of RODES is its ability to deal with the common but challenging situations of information as variables missing from data. Thinking in a systemic way one can conjecture that, due to the interactions in these networks, information must be implicitly present in the data. Therefore we used some ideas and techniques from control theory, mainly feedback linearization. To automate the algorithm the neural networks counterpart of the conventional feedback linearization (Garces, 2003) was used. Applied to drug gene networks the neural networks version of RODES algorithm (NN RODES) enable and automate the reconstruction of the time-series of the transcription factors, microRNAs, or drug related compounds which are usually missing in microarray experiments.

The tricky solution consists of transforming the modeling problem in a tracking control problem. The measured mRNA temporal series become the desired or reference trajectories. The problem is to find the control(s) such that the plant output - the solution of the mRNA ODE - tracks the desired trajectory with an acceptable accuracy level. These controls are the missing variables of the (D)GRNs that are identified in this way. To the best of our knowledge, this are the first realistic reverse-engineering algorithm, based on linear GP and NN FBL, for large gene networks including pharmacogenomic variables and interactions, capable to deal with missing information as variables from data.

2. Methods

2.1 (D)GRN Fundamental ODE Patterns and Building Blocks

The rate at which the concentration of a protein changes inside a cell depends mainly on the following:

1. the rate at which its mRNA is produced and degraded;
2. the rates at which the mRNA molecules are translated;
3. the rate at which the protein itself degrades.

Because these are (bio)chemical reactions the corresponding rates are described by (bio)chemical kinetic equations. There are two main frameworks for modeling and simulations (bio)chemical reactions, a deterministic and a stochastic one.

The deterministic modeling is based on the construction of a set of rate equations to describe the biochemical reactions. These rate equations are non-linear ordinary differential equations with concentrations of chemical species as variables. Deterministic simulation produces concentrations by integrating the ODEs. The stochastic modeling involves the formation of a set of chemical master equations with probabilities as variables (van Kampen, 1992) [17]. Stochastic simulation produces counts of molecules of the chemical species as realizations of random variables drawn from the probability distribution described by the chemical master equations. Which framework is appropriate for a given biological system depends on the investigated biological phenomena, and is influenced by the simplifying assumptions of the analysis (Wolkenhauer et al., 2004). The present report is focused on the deterministic approach.

Usually, these rates have the same mathematical form as the pharmacokinetic (PK) blocks describing the drug movement into, within, and out of the body:

1. Zero order: $dX/dt = k$, where k is a zero-order rate constant and X is the concentration of the drug;
2. First order: $dX/dt = k \cdot X$, where k is a first-order rate constant and X is as above; and
3. Michaelis–Menten: $dX/dt = V_m \cdot X/(K_m + X)$, where V_m is a maximum rate, K_m is the Michaelis constant, and X is as above.

Some of the rates of mRNA production, degradation, and translation are regulated by TFs and microRNAs (miRNA) respectively, in GRN.

In a pharmacogenomic context, new regulatory interactions, or exogenous control factors represented by drugs, are added. We introduced the more general concept of drug gene regulatory networks or drug gene networks in (Floares, 2007b). If the regulation is restricted to transcription factors and microRNAs, the network is a GRN. If the regulation is exerted by transcription factors, microRNAs and by drugs or drugs related compounds, e.g., drug–receptor complexes, the network is a DGRN. Thus, GRN could be considered a subset of DGRN.

The mathematical descriptions of the mechanisms of regulation, by TFs, microRNAs, and drug-related compounds, are the same. They have the form of the most common pharmacodynamic (PD) blocks, describing the relationship between drug doses or concentration and effects:

1) Linear (stimulation [+] or inhibition [-]) model: $E = E_0 \pm S \cdot C_e$

2) Log-linear (stimulation [+] or inhibition [-]) model: $E = E_0 \pm S \cdot \log (C_e)$

3) Ordinary (γ = 1) or sigmoid (γ > 1) $E_{\max}$ (stimulation [+] or inhibition [-]) model: $E = E_0 \pm E_{\max} \cdot C_e^\gamma / \left(C_e^\gamma + EC_{50}^\gamma\right)$

where E is the effect variable; E_0 is the baseline effect; E_{max} is the maximum drug-induced effect, also called capacity; EC_{50} (sometimes IC_{50} [50% inhibitory concentration] is used instead of EC_{50} for inhibitory effect) are the plasma concentration at 50% of maximal effect, also called sensitivity; S is the slope of the line relating the effect to the concentration; C_e is the concentration to which the effect is related, and γ is the sigmoidicity factor (Hill exponent).

GRN ODE systems models have one ODE for each mRNA, microRNA and protein, corresponding to transcription and translation, respectively. The protein can be a transcription factor too. DGRN ODE systems have some additional equations for the drug related compounds, which can act as transcription factors, e.g., a drug-receptor complex in the cellular nucleus. The corresponding ODE describes the translocation of the drug-receptor complex from cytoplasm to nucleus and its degradation. The ODE systems for DGRN and GRN results from the following:

1. summing up the pharmacokinetic blocks and
2. multiplying the rate constants of the regulated processes by pharmacodynamic blocks.

Usually, it is assumed that other processes, such as diffusion and transport, are fast with respect to transcription and translation and may thus be ignored.

In words, the structure of these equations, for any molecular specie, is simple:

Rate of Change =
Production Rate x Production Regulation - Degradation Rate x Degradation Regulation

Thus, the rates of change in a specific mRNA concentration (mRNA), and in the translated product concentration (e.g., a transcription factor, TF, in our case) are

$$\frac{dmRNA}{dt} = k_{sm} \cdot R_s - k_{dm} \cdot R_d \cdot mRNA \tag{1}$$

$$\frac{dTF}{dt} = k_{sTF} \cdot mRNA \cdot R_{sTF} - k_{dTF} \cdot R_{dTF} \cdot TF \tag{2}$$

where k_{sm} is the rate at which mRNA is produced and k_{dm} is the mRNA degradation rate constant; k_{dTF} is the TF degradation rate constant, and k_{sTF} is the average TF translation rate constant. R_s and R_d are generic notations for different regulatory factors of mRNA synthesis and degradation, respectively. Usually, R_s represents TFs regulating mRNA synthesis and R_{sTF} represents microRNAs regulating translation and probably drugs related compounds; R_d and R_{dTF} could represent drug-related compounds, e.g., a drug–receptor complex. A regulatory factor $R_{s,\ d} = 1$ indicates no regulation, and an $R_{s,\ d}$ having the form of one of the pharmacodynamic blocks indicates the action and the mechanism of action of a regulatory factor.

Equation (1) is a simple description for both mRNA and miRNA rates and their regulation. Equation (2) is a simple description of translation rate and its regulation for any protein, including the special case of transcription factors, were the protein also regulates transcription. It is worth mentioning that, while many molecular species could act as transcription or translation regulators, embedding all these regulatory interactions in a single variable or function is just a highly accurate but first approximation, as our results will show.

Equations (1) and (2) together with the above PK and PD blocks form a fundamental ODE patterns or building blocks of the (D)GRN models. This common domain knowledge, together with the information obtained via the data and knowledge mining approach, can be simply used to reduce the structure search space of the algorithm, and to identify the biochemical mechanisms involved, in the resultant model. As we will show bellow, the information concerning the direction, sign and mechanisms of such interactions can be at least partially extracted from data by RODES algorithms, in a data mining and network discovery from data approach. It possible and very useful to integrate the data mining approach using numbers, with a knowledge mining approach, extracting information from processed published literature databases, using dedicated systems biology software (e.g., IPA™ from Ingenuity, or GeneGo™ from GeneGo).

Three cases, of increasing complexity, are possible for both equation (1) and equation (2), and for simplicity, they will be presented only for equation (1), and for mRNA:

1. unregulated mRNA transcription and unregulated mRNA degradation,
2. regulated mRNA transcription and unregulated mRNA degradation, and
3. regulated mRNA transcription and regulated mRNA degradation.

For unregulated transcription and degradation ($R_s = 1$, $R_d = 1$), all variables (mRNAs) are available and one can use GP RODES, the RODES algorithm based on Genetic Programming (see (Floares, 2008) for details) to automatically infer the corresponding ODE.

For regulated transcription and missing information about the TFs or drug related compounds (variable missing), RODES was extended in (Floares, 2008), using neural networks and simulated data. The application of this NN RODES version to real experimental microarray data is a central theme of this contribution. Usually, while the equations' structure is known – it should be a version of equation (1)), and the parameters' values can be found in the literature or in public databases - only the temporal series of the mRNAs are available from microarray experiments, but not those of the TFs.

It is important to emphasize that this is an important and difficult data mining or knowledge discovery in data problem. Remember that even the most sophisticate artificial (computational) intelligence methods are just extracting information from data and not producing information. Information could be missing from data in two major distinct ways:

1. some variables from data have missing values, or
2. some variables are missing from data.

Both missing values and missing variables from data are encountered very frequently in practice. The first one is very easy to indentify just by carefully examining the data. The second one is not so evident, because in the most interesting data mining experiments one does not know exactly the number of relevant variables. Heuristically, missing variables manifests itself by a relatively low accuracy, which is very similar for different algorithms. Without entering into details, a typical situation could be for example a medical data mining problem -one has a dataset of say 150 patients, 10 input variables, and a binary diagnosis output variable, and (almost) no missing values; despite that, the prerequisites for a high accuracy are present, one only obtains say 80% accuracy. More than this, the accuracy is almost the same +/- 3% for all the algorithms tried - neural networks, support vector machines and decision trees - with the best settings for the algorithms. In addition, from the 10 input variables, only 7 proved relevant for the diagnosis problem. In such a situation, it is clear that information is missing from data and this is related to variables missing. To our knowledge, this is the first time a solution based on artificial intelligence is proposed, for the important problem of information missing, in the form of variables missing from data. Our solution allows the reconstruction of missing variables, is accurate, and is by no means limited to the biomedical problems reported here.

2.2 RODES Algorithm: No Missing Information, All variables measured

The goal of the proposed algorithm for reverse engineering is threefold:

1. to automatically identify the structure of accurate ODE systems models of GRN and DGRN,
2. to automatically estimate their parameters, and
3. to identify the biochemical and pharmacological mechanisms involved.

The RODES algorithm starts from complex time-series data. The name of the algorithm is related to its results, not to the biological systems investigated. This is because we successfully applied it to various biological networks: the subthalamopallidal neural network of the basal ganglia (Floares, 2008) and the vascular networks of tumors (work in progress). The result is an ODE system, $dX/dt = f(X)$.

In the time-series data, at any given discrete time point, t, where $t = 1, 2,...,T$, dX/dt is equal to $f(x)$ at the same time point t. Equivalently, for any individual ODE of the system, $dX_i{=}dt$ (at t) = $f_i(X)$ (at t), where $i = 1, 2,..., n$ is the number of variables. Thus, each equations of the ODE system can be reconstructed one by one, via a simple data mining approach, as algebraic relations f_i between the inputs X and output dX_i/dt. The algorithm can be used for experimental or simulated data. For simulated data, the true structure of the (D)GRN models is known, and this allows a faithful evaluation of the predicted models. We therefore used simulated time-series data to illustrate our algorithm; the accuracy for experimental data is similar, as will be shown for the more difficult problem of missing information (variables) from data (see next section). The RODES version with no missing information from data is based on genetic programming, as a machine learning method, and consists of the following steps:

1. Compute the time derivative of each variable, $dXi{=}dt$, at all discrete time points t:
    (a) differentiate each variable with respect to time for simulated data;
    (b) fit first a function to smooth the data, and then differentiate it, for noisy experimental data.
2. Build input-output pairs, (X_i; $dX_i{=}dt$), at the corresponding discrete time points t:
    (a) use all variables supposed to belong to the right hand side of the reconstructed ODE as inputs,

   (b) use the time derivative of one of the variables as output, if the GP implementation accepts many inputs but only one output, or
   (c) use the time derivatives of all the variables as output, if the GP implementation accepts many inputs and many outputs.
3. Build training, validation (optional, to avoid overfitting), and testing sets from the input-output pairs.
4. Initialize a population of randomly generated programs, coding mathematical models relating the inputs X_i to the output(s) dX_i=dt.
5. Perform a tournament contest:
   (a) Randomly select four programs and evaluate their fitness (mean squared error) - how well they map the input data X_i to the output data dX_i=dt.
   (b) Select two programs as winners and the other two as losers.
   (c) Copy the two winner programs and transform them probabilistically by:
      i. exchanging parts of the winner programs with each other to create two new programs (crossover) and/or
      ii. randomly changing each tournament winner to create two new programs (mutation).
   (d) Replace the loser programs with the transformed winner programs. The winners of the tournament remain in the population unchanged.
6. Repeat steps 5(a) - 5(d) until a program is developed that predicts the behavior sufficiently.
7. Extract the ODE model from the resultant program or directly use it.

Steps 1 - 3 reduce the problem of reversing a system of coupled ODEs, $dX/dt = f(X)$, in that of reversing individual, decoupled, algebraic equations, $dX_i/dt = f_i(X)$. Even though the output is in reality a time derivative, dX_i=dt, the algorithm is simply searching for an algebraic equation relating the inputs to the output, at each discrete time point t. The corresponding relation is the predicted function, $\hat{f}_i(X)$, for the right-hand side of each differential equation of the system.

This approach drastically reduces the CPU time of the algorithm, by orders of magnitude, because in step 5(a) the fitness evaluation does not require the integration of the ODE system. More precisely, one can use a fitness function based on (e.g., (Spieth et al., 2006)):

$$E_j = \sum_{i=1}^{n}\sum_{t=1}^{T}\left(\hat{X}_i(t) - X_i(t)\right)^2 \tag{3}$$

where j is the number of programs, n is the number of variables, T is the number of sampling points, $\hat{X}_i(t)$ is the numerically calculated time course of the variable X_i at time t from the ODE system predicted by the program j, and $X_i(t)$ represents the experimentally or simulated time course of X_i at time t. Therefore, for every programs fitness calculation, at each generation, the ODE system must be numerically integrated. We used a fitness function of the form

$$E_j = \frac{1}{T}\sum_{i=1}^{T}\left(\frac{d\hat{X}_i(t)}{dt} - \frac{dX_i(t)}{dt}\right)^2 \tag{4}$$

where *j* and *T* are as above, $d\hat{X}_i(t)/dt$ is the time derivative at time point *t* of the variable X_i predicted by the program *j*, and $dX_i(t)/dt$ represents the time derivative at time *t* of the experimental or simulated variable X_i calculated in step 1 of the algorithm.

While the time needed to integrate a system of ODE seems negligible, during fitness evaluation the integration has to be executed hundreds or thousands of times per generation. These, and the results of our previous studies (Floares, 2005), (Floares, 2006), (Floares, 2007b), suggest that RODES will scale up well, as required by modern high-throughput biomedical techniques.

We used a linear version of a steady-state genetic programming proposed by Banzhaf (see (Brameier & Banzhaf, 2007)) for a detailed introduction, and the literature cited there). In linear genetic programming the individuals are computer programs represented as a sequence of instructions from an imperative programming language or machine language. Nordin introduced the use of machine code in this context (cited in (Brameier & Banzhaf, 2007)). The major preparatory steps for GP consist of determining

1. the set of terminals (see below),
2. the set of functions (see below),
3. the fitness measure (see equation (4)),
4. the parameters for the run (see below),
5. the method for designating a result, and
6. the criterion for terminating a run.

The function set, also called instruction set in linear GP, can be composed of standard arithmetic or programming operations, standard mathematical functions, logical functions, or domain-specific functions.

We used the following Genetic Programming parameter setting:

- *Population size* 500
- *Mutation frequency* 95%
    - Block mutation rate 30%
    - Instruction mutation rate 30%
    - Instruction data mutation rate 40%
- *Crossover frequency* 50%
    - Homologous crossover 95%
- *Program Size* 80-128
- *Demes*
    - Crossover between demes 0%
    - Number of demes 10
    - Migration rate 1%
- *Dynamic Subset Selection*
    - Target subset size 50
    - Selection by age 50%
    - Selection by difficulty 50%
    - Stochastic selection 0%
    - Frequency (in generation equivalents) 1
- *Function set* {+, -, *, /}
- *Terminal set* 64 = *j* + *k*
    - Constants *j*
    - Inputs *k*

Using simple and common domain knowledge, such as the set of mathematical functions that appear in the models, e.g., arithmetic functions but not trigonometric function, is enough for RODES to find the proper structure of the reconstructed equations, also greatly increasing execution speed. The terminals are variables and parameters. In microarray experiments, the number of mRNAs is usually of the order of 10^2 or 10^3 after filtering, but the number of the clusters of genes with similar temporal signatures is small. One needs only to discover this small number of prototype ODE structures.

All the equations have one of these prototype structures, and the equations in the same cluster have the same structure but different parameter values. We still do not know which are the input variables for each mRNA ODE equation. From the fundamental ODE patterns of DGRN, we know that the equations for each mRNA (see equation (1)) contains a synthetic and a degradation term.

The inputs variables for these mRNA equations are

1. the mRNA concentration - in a degradation term proportional with mRNA concentration - for unregulated transcription and degradation,
2. the concentration of a transcription factor (for GRN) and/or of a drug related compound (for DGRN) - in a PD block (R_s in eqn (1))multiplying a PK block (the constant mRNA synthesis) - for regulated transcription and unregulated degradation, and
3. as above but also the concentration of a drug-related compound (for DGRN), contained in a PD block (R_d in eqn (1)) multiplying a PK block (the linear mRNA degradation) - for regulated transcription and regulated degradation.

The RODES version described in this section requires all inputs to be available. This condition is certainly true for the first situation but is usually false for the second and the third. The next section will extend RODES to cope with the second situation.

Because we know the structure of this ODE (see eqn (1)), this is also the route to automate the discovery of the biochemical and pharmacological molecular mechanisms involved. Analyzing the resultant equations, one can easily identify

1. cellular processes such as syntheses and degradations and their mechanisms as PK blocks,
2. the presence of regulation and
   (a) which are the regulated processes - their rate constants are multiplied by PD blocks,
   (b) which are the regulatory factors - transcription factors for GRN, drugs, or both for DGRN - the corresponding PD blocks can be functions of the TF concentrations or drug-related compound concentrations, respectively,
3. the regulation mechanisms - by looking at the corresponding PD blocks and at the rate constants they are multiplying.

There are situations in which the PK/PD mechanisms in the resultant mathematical model need to be clarified. When we have the product of two or more constants, in the symbolic form of the model, the algorithm will find only one numerical value. Using elementary domain knowledge, one can easily and clearly identify the PK/PD mechanisms (see (Floares, 2006) for details).

2.3 NN RODES - Neural Network Feedback Linearization for Missing Variable Identification

We focused only on the genes which expression is affected by the synthetic glucorticoid methylprednisolone treatment, the goal being to reverse engineer this drug gene regulatory network. The genes response to glucorticoid treatment can be classified into three categories:

1) genes stimulated by methylprednisolone,
2) genes inhibited by methylprednisolone,
3) genes with biphasic behavior - stimulation followed by inhibition or inverse.

While these three categories can be discriminated even by simple visual inspection of the temporal series of the microarray data, this approach does not entail objective criteria for selection of probes for further consideration. To screen for the probe sets objectively, the entire dataset was filtered with various filters (Almon, et al., 2007). We selected only the genes belonging to the first two aforementioned categories, stimulated and inhibited. For the categories of stimulated/inhibited genes, we tested two mechanisms - linear stimulation/inhibition and ordinary and sigmoid Emax stimulation/inhibition (see the pharmacodynamic blocks in section 2.1).

From the point of view of our theory of drug gene regulatory networks, we investigated the case of regulated mRNA transcription and unregulated mRNA degradation (see equation (1)), where the missing variable is either the temporal series of the regulatory transcription factor (in GRN and DGRN) or those of the drug–receptor complex (in DGRN).

This requires the neural networks feedback linearization version of RODES (NN RODES) which can cope with missing information. NN RODES was first introduced in (Floares, 2008) and applied to simulated data in order to test it on equations with known structure and parameters. The tricky solution we proposed in (Floares, 2008) consists in transforming the modeling problem in a tracking control problem:

1. The measured mRNA temporal series becomes the desired or reference trajectory.
2. The ODE with known structure (see equation (1)) and missing variable(s) becomes the plant to be controlled.
3. The missing variables become the control inputs; they are PD blocks incorporated in the position of the regulatory factor R_s of the transcription in equation (1).

The problem is to find the control(s) such that the plant output - the solution of the mRNA ODE - tracks the desired trajectory with an acceptable accuracy, while all the states and the control remain bounded in a physiological range.

It is tempting to speculate that this might be similar to the problem faced by the real living systems during evolution. This idea is corroborated by the fact that regulation appears to evolve on a faster time scale than the coding regions of the genes. For example, related animals, such as mice and humans, have similar genes, but the transcription regulation of these genes is quite different.

Also, an approach like this could offer a rational foundation for *gene therapy*, based on understanding and controlling (D)GRN, which are complex networks of interactions, instead of the pedestrian prevailing approach based on a ”one gene - one disease” rule.

Feedback linearization can be considered one of the most important nonlinear control design strategies developed in the last few decades (Garces et al., 2003). This approach algebraically transforms a nonlinear dynamic system into a linear dynamic system, by using a static-state feedback and a nonlinear coordinate transformation, based on differential geometric analysis of the system.

Because our goal is to automate the modeling process, we intended to use a computational intelligence version of feedback linearization. The massive parallelism, natural fault tolerance, and implicit programming of neural networks suggest that they may be good candidates. We successfully applied neural network feedback linearization, based on multilayer perceptrons (MLPs), to complex pharmacogenomic systems to find adequate drug dosage regimens (Floares, 2005); (Floares, 2006).

Owing to the reformulation of the modeling problem as a control problem, a NN FBL approach seems adequate and feasible. We used the NARMA-L2 version of input–output feedback linearization (Narendra, cited in (Garces et al., 2003); see also the literature cited in (Garces, et al., 2003), in which the output becomes a linear function of a new control input.

Fortunately, the prerequisites of the approach, represented by the equations' structure and parameters, are usually known. In the particular situation investigated here the following domain knowledge and features of the experimental design are taking into account:

1. The equations structure - there is a strong theoretical foundations for building kinetic equations like equations (1) and (2).
2. The equations parameter - for the control untreated subjects there is no drug regulation (R) in equation 1, supposed at steady state; at time t = 0 the concentration of mRNA is $mRNA_0$ = 1, because all expression profiles are normalized to that of the control subjects, and thus k_{sm} and k_{dm} are easily estimated.
3. The regulation and its mechanisms
    a. with the proper filters only genes regulated by the drug are supposed to be selected,
    b. filters can discriminate between three category of regulation: stimulation, inhibition, or both,
    c. the number of possible regulation mechanisms is small and their equations are known (see section 2.1).

The control input is the unknown regulator: a pharmacodynamic block (see section 2.1) containing the TF concentration in GRN, a drug-related compound, or both in DGRN. In this approach a neural network model of the "plant" is first identified, even if, as in our situation, the mathematical model of the "plant" is known. The mathematical model of the "plant" is simply the equation 1, where only the mRNA synthesis is regulated, and the degradation is not:

$$\frac{dmRNA}{dt} = k_{sm} \cdot R_s - k_{dm} \cdot mRNA. \tag{5}$$

As we previously stated, the values of k_{sm} and k_{dm} are usually known. For example, for the ornithine decarboxylase gene we have k_{sm} = k_{dm} = 0.30. It is known from the experiments that the drug stimulates this gene, but we do not know exactly the mechanism and the corresponding formula for R_s. One can try a linear stimulation mechanism, followed by an E_{max} one if the first failed, these being the most common mechanisms in the order of increasing complexity. Thus, the Rs for these two simulation experiments are:

$$R_s = E_0 + S \cdot DR(N) \tag{6}$$

and

$$R_s = E_0 + DR(N) / \left(EC_{50} + DR(N)\right) \tag{7}$$

respectively; the so called basal effect of the drug, E_0 = 1, and $DR(N)$ is the drug-receptor complex in the nucleus, the control input which we are trying to find.

A random input control, between zero and the estimated maximal value, is injected into the model at random intervals. The NN model structure is the standard nonlinear autoregressive moving average (NARMA) model, adapted to the feedback linearization of affine systems - the controller input is not contained in the nonlinearity.

We want the system output represented by the mRNA to track a reference trajectory. This reference trajectory could be related to the measured level of genes expression, as it will be shown bellow, or to a therapeutic objective in a gene therapy context, for example. In the last situation, the idea is that we can in a similar way constrain the genes expression, via the drug inputs to follow some desired or normal trajectories, when they are pathologically perturbed.

We need the time-series data of the mRNA of the investigated genes. Usually, the expression levels in microarray experiments are measured at a couple of specific time points, on a small number of subjects for each time point (see (Almon et al., 2005); (Almon et al. 2007) for the particular datasets used in this investigation).

In order to apply the approach previously described, we first calculated the mean expression level for each time point, and then interpolated using a cubic spline interpolant. Because taking the derivative of noisy data can increase the noise, the result of the interpolation is differentiated instead, with respect to time. The number of hidden layers is one for all neural networks. The number of neurons in the hidden layer, of the two MLPs, is between 5 and 9, depending on the complexity of the problem and the results of the simulation experiments. The activation functions are tangent hyperbolic for the hidden layer and linear in the output layer for all NNs. The parameters and their values are the following:

- *Network Architecture*
  - Number of hidden layers 1
  - Size of hidden layer 5-9
  - Sampling time 0.01
  - Number of delayed plant inputs 3
  - Number of delayed plant outputs 2
- *Training Data*
  - Training samples 4320
  - Minimum plant input 0
  - Maximum plant input maximum DR(N)
  - Minimum time interval value 0.1
  - Minimum time interval value 1
  - Minimum plant output 0
  - Maximum plant output maximum mRNA
  - Training epochs 150

We investigate the prediction errors by cross-validation on a test set. We used Bayesian regularization (MacKay, 1992), a training function that updates the weight and bias values according to Levenberg–Marquardt optimization. It minimizes a combination of squared errors and weights and then determines the correct combination to produce a network that generalizes well. We start with different random initial conditions to avoid ending in "bad" local minima. In NN FBL the controller is simply a rearrangement of the neural network plant model. The time-series of the missing variables are identified as the control inputs, and the complete equation is thus reconstructed.

Three important problems can be approached by the proposed methods:

1) finding the unknown transcription factor profile in a drug-gene regulatory network, using the measured mRNA profile as a reference trajectory;
2) finding the unknown drug-receptor complex profile in a drug-gene regulatory network, using again the measured mRNA profile as a reference trajectory; and
3) finding the optimal/individualized drug–receptor complex profile, corresponding to the optimal drug dosage regimen, capable of constraining the mRNA profile to track a desired therapeutic objective, in a pharmacogenomic context.

Because of the mechanistic and mathematical similarities between transcription regulation by TFs and by drug–receptor complexes, the example is illustrative for both situations.

3. Results

Most often, the temporal series of the TF or of the drug-receptor complex is not known for regulated genes. This is also true for the temporal series of the mRNA of the TF in proteomics experiments, when one wants to reconstruct the TF ODE (see eqn (2)). In these situations, RODES clearly indicates that information is missing from the input set (see (Floares, 2008)), and one has to use the extension based on neural network feedback linearization.

For illustration we tested the NN RODES algorithm on reconstructing equation (3) for two genes: the ornithine decarboxylase gene and the *a*-2Macroglobulin gene. The parameters for the two genes are: $k_{sm} = k_{dm} = 0.30$ for ornithine decarboxylase, and $k_{sm} = k_{dm} = 0.038$ for *a*-2Macroglobulin. There are three temporal series in equation 3:

1) one for the *mRNA* - obtained by fitting the mean expression level at each time point of the experiment, using a cubic spline interpolant (see Fig. 1)
2) one for the *dmRNA/dt* - obtained by differentiating with respect to time the result of the interpolation (see Fig. 1)
3) one for the regulator - the drug-receptor complex in the cellular nucleus, *DR*(*N*), which we are trying to reconstruct with the aid of NN FBL.

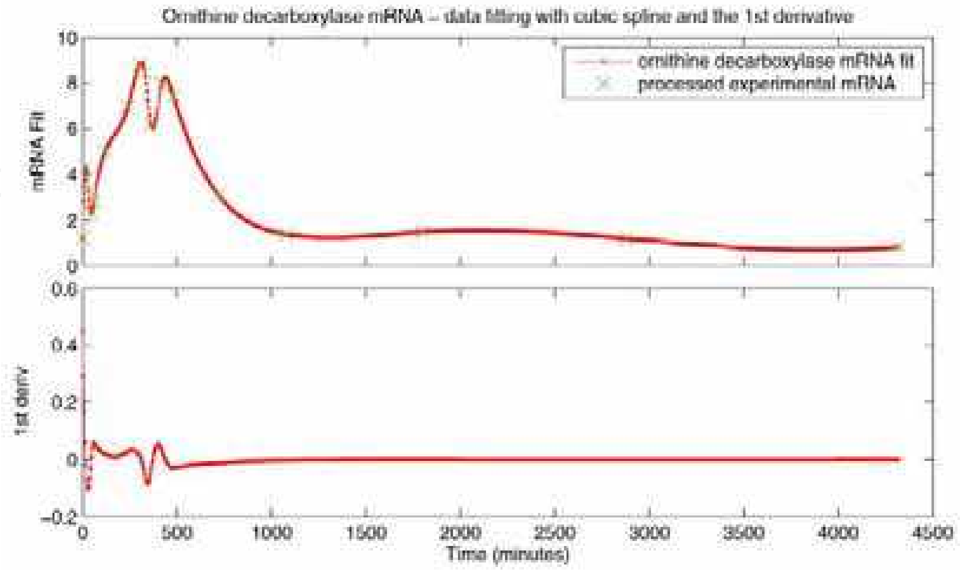

Fig. 1. Ornithine decarboxylase mRNA - processed mRNA experimental data fitted with a cubic spline interpolant and the 1st derivative with respect to time (explanation in text).

The settings of the neural networks feedback linearization experiments are presented at the end of section 3, above.
In our previous studies (see (Floares, 2008)) the performance of the identification step of NN FBL was very good, but also somehow expected because the data were simulated. Here, with real microarray time-series data the performance were very high too, and the order of magnitude of the error is 10^{-3} - 10^{-4} for both genes (see Fig. 2 and Fig. 3).

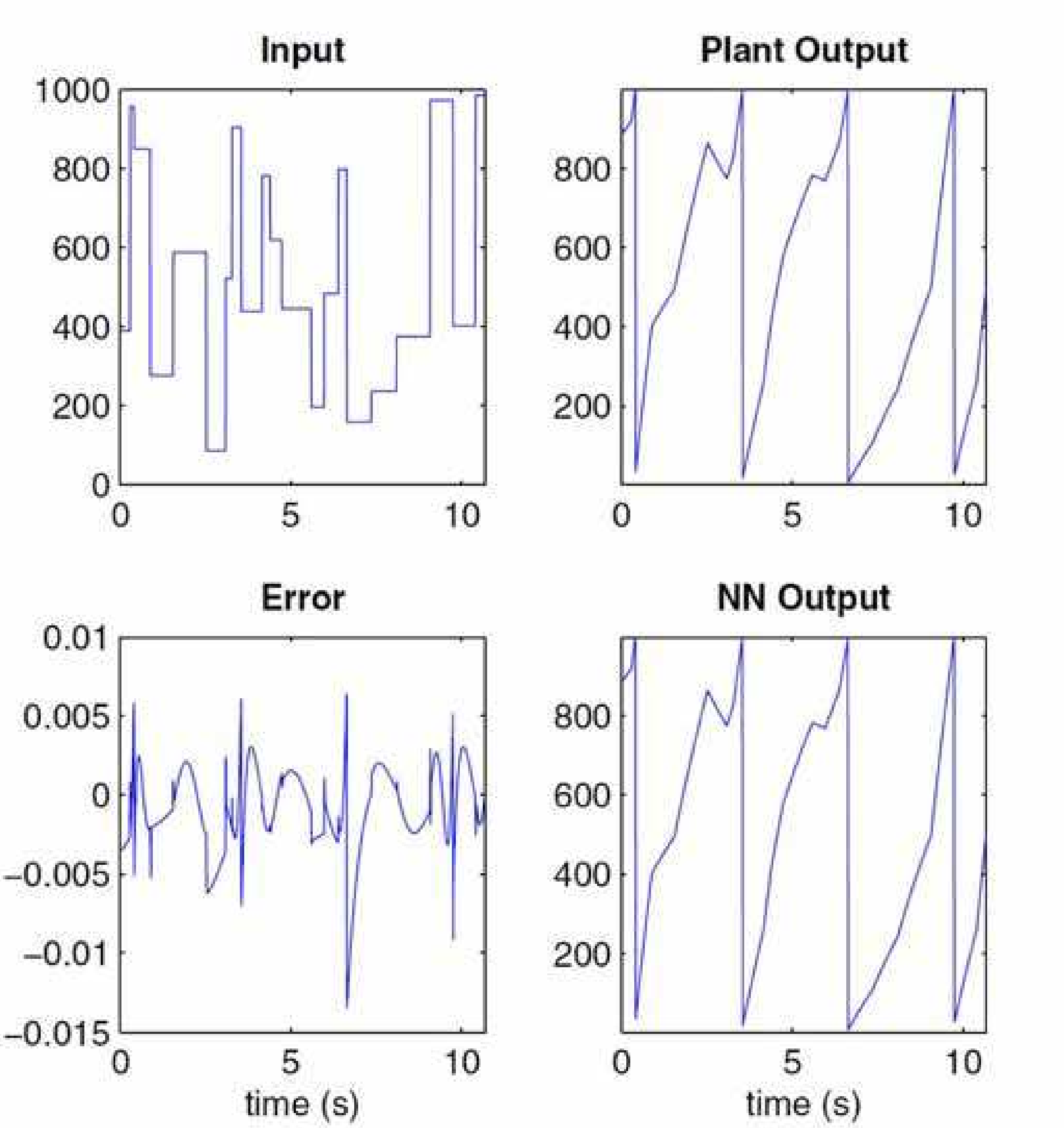

Fig. 2. Neural networks feedback linearization model identification for ornithine decarboxylase mRNA; the NN model have an almost identical output for the same input as the plant (mathematical model), the order of magnitude of the error is 10^{-3} (un-scaled data; explanation in text).

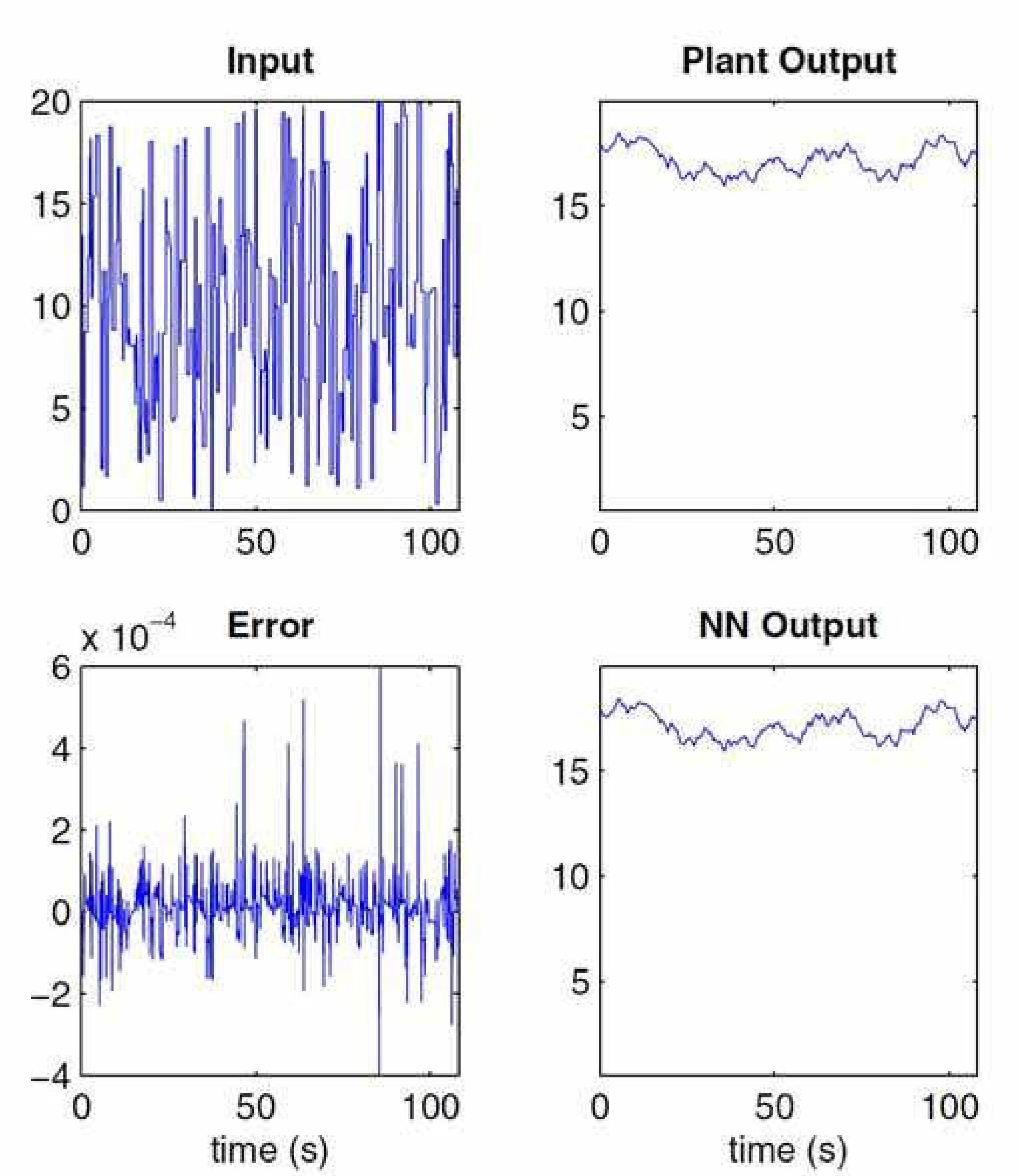

Fig. 3. Neural networks feedback linearization model identification for *a*2-Macroglobulin mRNA; the NN model have an almost identical output for the same input as the plant (mathematical model), the order of magnitude of the error is 10^{-4} (scaled data; explanation in text).

Also, the performance of the identification of the drug-receptor complex concentration profile in the nucleus, *DR*(*N*), is very good for both genes; the results are shown just for ornithine decarboxylase, those for the *a*2-macroglobulin being similar (see Fig. 4).

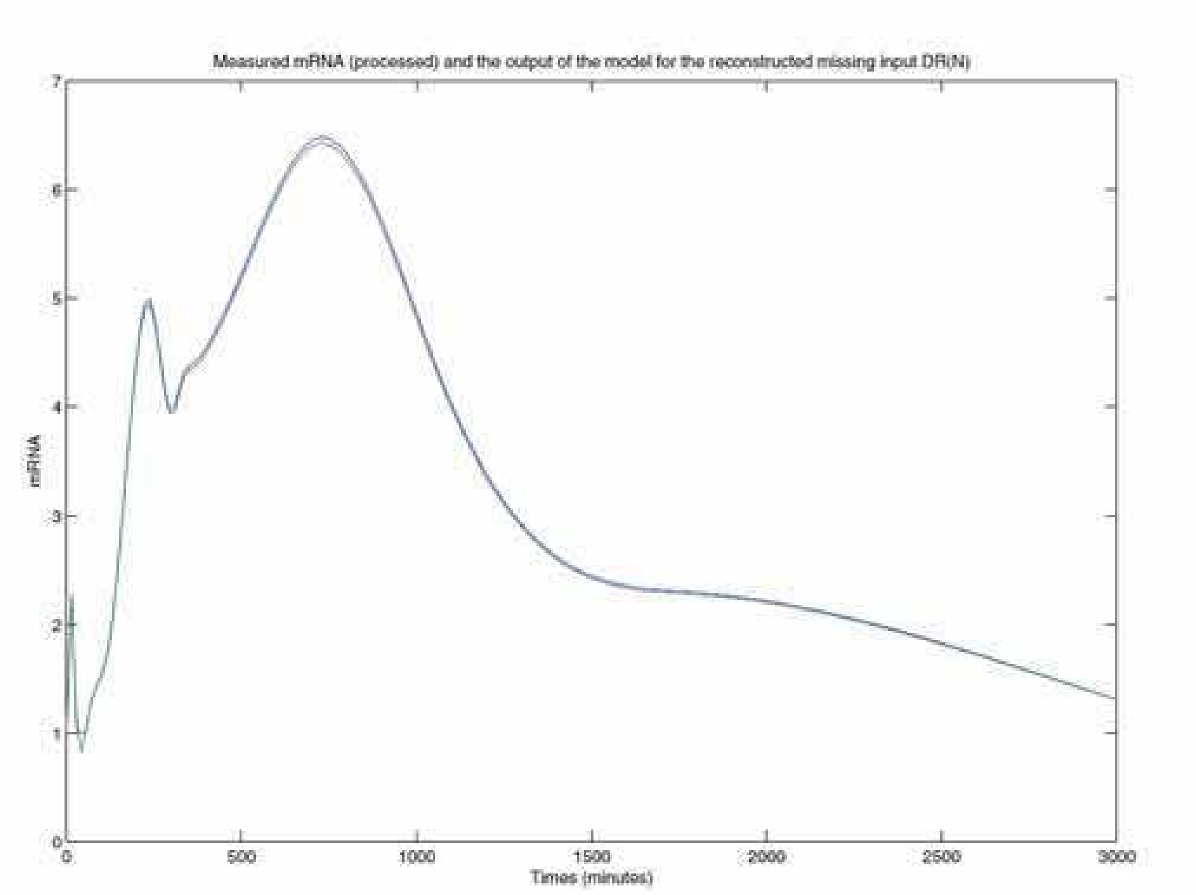

Fig. 4. Neural networks feedback linearization control for ornithine decarboxylase mRNA; the missing temporal-series of drug-receptor complex in the nucleus, DR(N), are reconstructed such as to constraint the mRNA output of the model to follow the measured mRNA (processed). The accuracy of the control is very good (explanation in text).

Thus, NN RODES using neural networks feedback linearization is able to reconstruct with high accuracy a system of ordinary differential equations, modeling (drug) gene regulatory networks, in the very difficult but common situation of having only the time-series of the gene expression levels, while the time-series of the regulators - transcription factors and drug related compounds - are missing.

Because of the very low tracking error, the reference trajectory, which is the measured mRNA profile (after the processing previously described) the mRNA output of the controlled system cannot be distinguished.

4. Conclusion

ODE systems are one of the most sophisticated approaches to modeling gene regulatory networks and drug gene regulatory networks, the superset of GRN we have recently proposed; it is also one of the most difficult. This contribution showed the applications of RODES, our algorithm for reverse-engineering gene networks, based on linear genetic programming and neural networks control by feedback linearization method, to real microarray time-series data (NN RODES), and simulated data (GP RODES).

Common to both neural networks and genetic programming component is the proposed method for decoupling the ordinary differential equations system. This allows reversing the ordinary differential equations of the system one by one. The neural networks component enables RODES to deal with the very difficult but common situation in which only the microarray time-series data are available, but the regulators time-series, transcription factors and drug related compounds, are missing.

Here we focused on the case of regulated mRNA transcription and unregulated mRNA degradation, when either the temporal series of the regulatory transcription factor (in GRN) or those of the drug–receptor complex (in DGRN) are missing. The tricky solution consists of transforming the modeling problem in a tracking control problem. The measured mRNA temporal series becomes the desired, or reference, trajectory. The problem is to find the control(s) such that the plant output - the solution of the mRNA ODE - tracks the desired trajectory with an acceptable level of accuracy. These control inputs are the missing variables that can be identified in this way, thus completing the automatic reconstruction of the ODE equation. To the best of our knowledge, RODES is the only reverse-engineering algorithm based on neural network feedback linearization, that has been applied to reverse engineer gene networks, as a highly accurate system of ordinary differential equations, in the very difficult but common situation of missing information from data, as missing variables - having only the time-series of the gene expression levels, but not the time-series of transcription factors and drug related compounds. In addition, the algorithm is by no means restricted to the biomedical field, automating the ODE modeling of complex time series, even when information is missing from data in the form of variable missing, in any scientific and technical field.

5. References

Almon, R. R.; Dubois, D. C.; Jin, J. Y. & Jusko, W. J. 2005. Pharmacogenomic responses of rat liver to methylprednisolone: An approach to mining a rich microarray time series. *The AAPS Journal*, vol. 7, no. 1, pp. 156–194

Almon, R. R. ; DuBois, D. C. & Jusko, W. J., 2007. A microarray analysis of the temporal response of liver to methylprednisolone: A comparative analysis of two dosing regimens. *Endocrinology*, vol. 148, no. 5, pp. 2209–2225

Bansal, M. ; Belcastro, V.; Ambesi-Impiombato, A. & di Bernardo, D. 2007. How to infer gene networks from expression profiles. *Molecular Systems Biology*, vol. 3, no. 78, pp. 1–10

Beard, D. A. ; Qian, H. & Bassingthwaighte, J. B., 2004 Stoichiometric foundation of large-scale biochemical system analysis, in *Modelling in Molecular Biology*, ser. Springer Natural Computing Series, G. Ciobanu and G. Rozenberg, Eds. Springer, pp. 1–19

Brameier, M. & Banzhaf, W., 2007. *Linear Genetic Programming*, ser. Genetic and Evolutionary Series. Springer

Cho, D.-Y.; Cho, K.-H. & Zhang, B.-T., 2006. Identification of biochemical networks by s-tree based genetic programming. *Bioinformatics*, vol. 22, no. 13, pp. 1631–1640

Floares, A. G., 2005. Genetic programming and neural networks feedback linearization for modeling and controlling complex pharmacogenomic systems, in *Fuzzy Logic and Applications, 6th International Workshop, WILF 2005, Revised Selected Papers*, ser. Lecture Notes in Computer Science, I. Bloch, A. Petrosino, and A. Tettamanzi, Eds., vol. 3849. Crema, Italy: Springer, Sep. 15-17, pp. 178–187

Floares, A. G., 2006. Computational intelligence tools for modeling and controlling pharmacogenomic systems: Genetic programming and neural networks, in *Proceedings of the 2006 IEEE World Cogress on Computational Intelligence*, G. C. Yen, L. Wang, P. Bonissone, and S. M. Lucas, Eds. Vancouver, CA: IEEE Press, pp. 7510–7517

Floares, A. G., 2007a. Reverse engineering algorithm for neural networks applied to the subthalamopallidal network of the basal ganglia, in *Proceedings of the International Joint Conference on Neural Networks*, Orlando, Florida, USA, August

Floares, A. G. , 2007b. Automatic reverse engineering algorithm for drug gene regulating networks, in *Proceedings of The 11th IASTED International Conference on Artificial Intelligence and Soft Computing*, Palma de Mallorca, Spain

Floares, A. G., 2008. Automatic inferring drug gene regulatory networks with missing information using neural networks and genetic programming," in *IEEE World Congress on Computational Intelligence*

Floares, A.G., 2008. A reverse engineering algorithm for neural networks, applied to the subthalamopallidal network of basal ganglia. *Neural Networks*, vol. 21, no 2-3, March/April, pp. 379-386

Garces, F. R.; Becerra, V. M.; Kambhampati, C., Warwick ,K., 2003. *Strategies for Feedback Linearisation: A Dynamic Neural Network Approach*. Springer-Verlag.

Gardner, T. S. & Faith, J. J. , 2005. Reverse-engineering transcription control networks. *Physics of Life Reviews*, no. 2, pp. 65–68

Jong, H. D. , 2002. Modeling and simulation of genetic regulatory systems: A literature review. *Journal of Computational Biology*, vol. 9, no. 1, pp. 67–103

Kikuchi, S.; Tominaga, D.; Arita, M.; Takahashi, K. & Tomita, M. , 2003. Dynamic modeling of genetic networks using genetic algorithm and s-system. *Bioinformatics*, vol. 19, no. 5, pp. 643–650

MacKay, D. J. C., 1992. Bayesian interpolation. *Neural Computation*, vol. 4, no. 3, pp. 415–447, [Online]. Available: citeseer.ist.psu.edu/article/mackay91bayesian.html

Noman N. & Iba, H. , 2005. Reverse engineering genetic networks using evolutionary computation. *Genome Informatics*, vol. 16, no. 2, pp. 205–214

Sakamoto E. & Iba, H., 2001 "Inferring a system of differential equations for a gene regulatory network by using genetic programming," in *Proceedings of the 2001 Congress on Evolutionary Computation CEC2001*. COEX, World Trade Center, 159 Samseong-dong, Gangnam-gu, Seoul, Korea: IEEE Press, 27-30, pp. 720–726. [Online]. Available: citeseer.ist.psu.edu/sakamoto01inferring.html

Savageau, M. A. , 1976. *Biochemical System Analysis: a Study of Function and Design in Molecular Biology*. Reading, MA: Addison-Wesley

Spieth, C.; Worzischek, R. & Streichert, F., 2006. Comparing evolutionary algorithms on the problem of network inference in *GECCO '06: Proceedings of the 8th annual conference on Genetic and evolutionary computation*. New York, NY, USA: ACM Press, pp. 305–306.

van Kampen N. G. , 1992. *Stochastic processes in physics and chemistry*. North-Holland

Voit, E. O. , 2000. *Computational Analysis of Biochemical Systems*. Cambridge University Press

Wolkenhauer, O. ; Ullah, M.; Kolch, W. ; Kwang-Hyun Cho, 2004. Modelling and simulation of intra cellular dynamics: Choosing an appropriate framework. *IEEE Trans. NanoBioSci.*, vol. 3, no. 3, pp. 200–207

Wang, J. Ed., 2008. IEEE Computational Intelligence Society. Hong Kong: IEEE Press, 1-6 Jun.

New Trends in Technologies
Edited by Blandna ramov

ISBN 978-953-7619-62-6
Hard cover, 242 pages
Publisher InTech
Published online 01, January, 2010
Published in print edition January, 2010

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821