Modelling fixed points in genetic regulatory networks using mathematical programming techniques

Leo Liberti[1], Camilo La Rota[2], Fabien Tarissan[1]

[1] *LIX, École Polytechnique, F-91128 Palaiseau, France*
Email:{liberti,tarissan}@lix.polytechnique.fr

[2] *ISC, École Normale Supérieure, Lyon, France*
Email:camilo.larota@ens-lyon.fr

October 2, 2007

Abstract

This paper employs mathematical programming and mixed integer linear programming techniques for solving a problem arising in the study of genetic regulatory networks. More precisely, we solve the inverse problem consisting in the determination of the arc weights in the digraph representing the network in such a way that the number of fixed points is minimized.

1 Introduction

[to do]

The rest of this paper is organized as follows. In Sect. 2 we model the inverse problem as an optimization problem — in particular, in Sect. 2.2 we propose a nonlinear mathematical programming formulation, and an exact linearization in Sect. 2.3. In Sect. 3 we discuss our computational results, and Sect. 4 concludes the paper.

2 Modelling the problem

Given a directed graph $G = (V, A)$, a discrete set of time instants T (which we suppose to be an initial contiguous proper subset of $\mathbb{N}$) and the following functions:

- a function $\alpha : A \to \{+1, -1\}$ called the *arc sign function*;
- a function $\omega : A \to \mathbb{R}_+$ called the *arc weight function*;
- a function $\gamma : V \times T \to \{0, 1\}$ called the *gene activation function*;
- a function $\iota : V \to \{0, 1\}$ called the *initial configuration*;
- a function $\theta : V \to \mathbb{R}_+$ called the *threshold function*,

a *gene regulatory network* (GRN) is a 7-tuple $(G, T, \alpha, \omega, \gamma, \iota, \theta)$ such that:

$$\forall v \in V \quad \gamma(v, 1) = \iota(v) \tag{1}$$

$$\forall v \in V, t \in T \smallsetminus \{1\} \quad \gamma(v, t) = \begin{cases} 1 & \text{if } \sum\limits_{u \in \delta^-(v)} \alpha(u, v)\omega(u, v)\gamma(u, t) \geq \theta(v) \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

where $\delta^-(v) = \{u \in V \mid (u,v) \in A\}$ for all $v \in V$. Eqns. (1)-(2) together are called the *evolution rules* of the GRN. For any particular $t \in T$, $\gamma(\cdot,t) : V \to \{0,1\}$ is called a *configuration.* Since the evolution rules relate a configuration at time t with a configuration at time $t-1$, if $\gamma(\cdot,t) = \gamma(\cdot,t-1)$ then $\gamma(\cdot,t') = \gamma(\cdot,t)$ for all $t' > t$: such configurations are called *fixed points* of the GRN. Furthermore, as long as the evolution rules are purely deterministic (as is modelled above), a fixed point of a GRN is determined by its initial configuration.

In this paper we deal with an inverse problem relating to parameter estimation in GRNs. More precisely, we address the following.

> Parameter Estimation in GRNs (PEGRN). Given a digraph G, a time instant set T, an arc sign function α, and a set C of initial configurations, find an arc weight function ω and a threshold function θ with the property that for all $\iota \in C$ there exists a gene activation function γ such that $(G, T, \alpha, \omega, \gamma, \iota, \theta)$ is a GRN whose number of fixed points is minimum.

In other words, we attempt to estimate the arc weight and threshold functions of a GRN from the knowledge of the digraph topology, the arc sign function in such a way that (a) the GRN evolution rules are consistent with respect to a certain set of initial configurations and (b) the number of fixed points of the GRN is minimized.

The methodolgy we shall follow is that of modelling the PEGRN by means of a mathematical programming formulation [7]:

$$\left.\begin{array}{rrcl} \min_x & f(x) & & \\ \text{subject to} & g(x) & \leq & 0, \end{array}\right\} \tag{3}$$

where $x \in \mathbb{R}^n$ are the *decision variables* and $f : \mathbb{R}^n \to \mathbb{R}$ is the *objective function* to be minimized subject to a set of constraints $g : \mathbb{R}^n \to \mathbb{R}^m$ which may also include variable ranges or integrality constraints on the variables. A problem where f, g are nonlinear is known as a Nonlinear Programming problem (NLP); if some integrality constraints are present on the variable bounds the problem is known as a Mixed-Integer NLP (MINLP). A mathematical programming problem which has some integer variables but whose objective function and constraints are linear forms is called a Mixed-Integer Linear Programming (MILP) problem. There exists general-purpose solution algorithms, both exact and heuristic, for all problem forms in NLP, MINLP, MILP. Currently, MILP solution methods are the most advanced, and the *de facto* standard solver is the ILOG CPLEX [2] solver. MINLPs and nonconvex NLPs can be solved by many different global optimization methods such as BARON [6]. Most frequently, NLPs and MINLPs undergo a reformulation stage before being solved [3, 4].

The primary concern in solving the PEGRN is thus modellistic rather than algorithmic. One of the foremost difficulties is that of employing a static modelling paradigm — such as mathematical programming — in order to describe a problem whose very definition depends on time. Another important difficulty resides in describing the necessary and sufficient conditions for a configuration to be a fixed point in a mathematical form suitable for use in a formulation like (3). We solve the first difficulty by introducing variables indexed by a time instant $t \in T$. The solution of the second difficulty is discussed below.

2.1 GRN fixed point conditions

Let C be a set of initial configurations, and $x : C \to \{0,1\}$ be a function such that $\forall \iota \in C$, $x(\iota) = 1$ iff ι evolves into a fixed point of the GRN. Furthermore, let $y : C \times V \times T \to \{0,1\}$ be a function such that for all $\iota \in C, v \in V, t \in T$, $y(\iota, v, t) = \gamma(v,t)$ when the GRN evolves from the initial configuration ι.

2.1 Lemma
If $x(\iota) = 1$ for some $\iota \in C$, then $\prod_{t \in T \smallsetminus \{1\}} \sum_{v \in V} (y(\iota,v,t) - y(\iota,v,t-1))^2 = 0.$

Proof. If $x(\iota) = 1$ then ι is an initial configuration evolving into a fixed point of the GRN. This means that there is a $t \in T$ such that $\gamma(v,t) = \gamma(v,t-1)$ for all $v \in V$, or alternatively $(y(\iota,v,t) - y(\iota,v,t-1))^2 = 0$. Thus the product over all t of all such squared differences is also zero. $\square$

2.2 Lemma
If $x(\iota) = 0$ for some $\iota \in C$, then $\prod\limits_{t \in T \smallsetminus \{1\}} \sum\limits_{v \in V} (y(\iota,v,t) - y(\iota,v,t-1))^2 \geq 1$.

Proof. If $x(\iota) = 0$ then ι does not evolve into a fixed point of the GRN. This means that for all $t \in T$ there exists $v \in V$ such that $\gamma(v,t) \neq \gamma(v,t-1)$. Since γ maps into $\{0,1\}$, this implies that $(y(\iota,v,t) - y(\iota,v,t-1))^2 = 1$, which in turn means that $\sum_{v \in V}(y(\iota,v,t) - y(\iota,v,t-1))^2 \geq 1$. The result follows. $\square$

2.3 Corollary
The following relationships hold:

$$\forall \iota \in C \quad x(\iota) \prod_{t \in T \smallsetminus \{1\}} \sum_{v \in V} (y(\iota,v,t) - y(\iota,v,t-1))^2 = 0 \tag{4}$$

$$\forall \iota \in C \quad x(\iota) + \prod_{t \in T \smallsetminus \{1\}} \sum_{v \in V} (y(\iota,v,t) - y(i,v,t-1))^2 \geq 1. \tag{5}$$

Cor. 2.3 provides a way to count the fixed points in the GRN ($\sum_{\iota \in C} x(\iota)$) and to relate this number to the gene activation function. We now have all the elements to describe the PEGRN by means of a mathematical programming formulation.

2.2 Mathematical programming formulation

- *Sets*:
  1. set V of genes in the network;
  2. set A of arcs in the network;
  3. set C of possible initial configurations;
  4. set T of time instants.
- *Parameters*:
  1. $\alpha : A \to \{+1, -1\}$ is the sign of the arc weights;
  2. for all $i \in C, v \in V$, $y_{iv}^{\text{boundary}} \in \{0,1\}$ are the initial configurations (activation value of gene v in initial configuration i);
  3. θ^L, θ^U are the bounds on the threshold values;
  4. w^L, w^U are the bounds on the arc weights.
- *Variables*:
  1. for all $i \in C$, $x_i = 1$ if initial configuration i evolves into a fixed point or 0 otherwise;
  2. for all $i \in C, v \in V, t \in T$, $y_{ivt} \in \{0,1\}$ is the activation status of gene v at time t from initial configuration i;
  3. $\theta : V \to \mathbb{R}$ is the threshold function;
  4. $w : A \to \mathbb{R}_+$ is the arc weight function;

5. M: a "large enough" real value.

- *Objective function*:
$$\min \sum_{i\in C} x_i.$$

- *Constraints*:

1. evolution rule:
$$\forall i \in C, t \in T \smallsetminus \{1\}, v \in V \quad \sum_{u\in\delta^-(v)} \alpha_{uv} w_{uv} y_{i,u,t-1} \geq \theta_v y_{ivt} - M(1-y_{ivt}) \tag{6}$$
$$\forall i \in C, t \in T \smallsetminus \{1\}, v \in V \quad \sum_{u\in\delta^-(v)} \alpha_{uv} w_{uv} y_{i,u,t-1} \leq \theta_v (1-y_{ivt}) + M y_{ivt}; \tag{7}$$

2. fixed point conditions:
$$\forall i \in C \quad x_i \prod_{t\in T\smallsetminus\{1\}} \sum_{v\in V} (y_{ivt} - y_{i,v,t-1})^2 = 0 \tag{8}$$
$$\forall i \in C \quad x_i + \prod_{t\in T\smallsetminus\{1\}} \sum_{v\in V} (y_{ivt} - y_{i,v,t-1})^2 \geq 1; \tag{9}$$

3. boundary conditions:
$$\forall i \in C, v \in V \quad y_{iv1} = y_{iv}^{\text{boundary}}.$$

The value to be assigned to M should be an upper bound to $|\sum_{u\in\delta^-(v)} \alpha_{uv} w_{uv} y_{i,u,t-1}|$ over all $v \in V, t \in T \smallsetminus \{1\}$.

2.3 Linearization

In this section we discuss an exact reformulation [4] that transforms the problem of Sect. 2.2 into a MILP. Both the evolution rule and the fixed point condition constraints in the formulation of Sect. 2.2 are nonconvex constraints because of the presence of the multiplication between variables or terms involving variables:

- in Constraints (6)-(7) we have two types of products, both multiplying a binary variable by a continuous variable: $w_{uv} y_{i,u,t-1}$ and $\theta_v y_{iut}$;
- in Constraints (8)-(9) we have several types of products of variables/terms: y_{ivt}^2, $y_{ivt} y_{i,v,t-1}$, the product $\prod_t \tau_{it}$ where $\tau_{it} = \sum_v (y_{ivt} - y_{i,v,t-1})^2$ and the product $x_i \prod_t \tau_{it}$.

We remark that aside from the product $\prod_t \tau_{it}$, all the other products only have two multiplicative terms, one of which only takes on values in $\{0,1\}$. Without loss of generality, we are going to suppose that all continous variables will be bounded both above and below: this does not lose generality because of the fundamentally physical nature of this problem. For such products, an exact linearization is readily available. Consider two variables $q \in \{0,1\}$ and $r \in [r^L, r^U]$. Whenever it appears, the product qr can be replaced by a (continuous) *linearization variable* s, as long as the following (linear) constraints are also added to the problem:

$$s \leq r^U q \tag{10}$$
$$s \geq r^L q \tag{11}$$
$$s \leq r + (|r^U| + |r^L|)(1-q) \tag{12}$$
$$s \geq r - (|r^U| + |r^L|)(1-q). \tag{13}$$

In other words, if $q = 0$, then (10)-(11) force $s = 0$ and (12)-(13) are inactive. If $q = 1$, then (10)-(11) are inactive and (12)-(13) force $s = r$. The particular case where r is also a binary variable can be more appropriately dealt with by adding the following constraints:

$$\begin{aligned} s &\leq q && (14)\\ s &\leq r && (15)\\ s &\geq r + q - 1 && (16)\\ s &\geq 0. && (17) \end{aligned}$$

The only extant case is the squared term y_{ivt}^2. Notice that since $y_{ivt} \in \{0, 1\}$, $y_{ivt} = y_{ivt}^2$, so squared binary terms can simply be replaced by their argument (in this case y_{ivt}).

In order to deal with the term $\prod_t \tau_{it}$, for all $t \in C$ and $t \in T$ we introduce binary variables χ_{it} defined as follows:

$$\chi_{it} = \begin{cases} 1 & \text{if } \tau_{it} > 0 \\ 0 & \text{otherwise.} \end{cases}$$

The relation contraints between χ and τ:

$$\begin{aligned} \forall i \in C, t \in T \quad \sum_{v \in V} (y_{ivt} - y_{i,v,t-1})^2 &\leq \chi_{it} M \\ \forall i \in C, t \in T \quad \sum_{v \in V} (y_{ivt} - y_{i,v,t-1})^2 &\geq (\chi_{it} - 1) M \end{aligned}$$

force χ_{it} to be 1 if the corresponding τ_{it} is positive, and 0 if $\tau_{it} = 0$. We now re-write the fixed point constraints (8)-(9) by means of the χ variables as follows:

$$\begin{aligned} \forall i \in C \quad x_i \prod_{t \in T \smallsetminus \{1\}} \chi_{it} &= 0 && (18)\\ \forall i \in C \quad x_i + \prod_{t \in T \smallsetminus \{1\}} \chi_{it} &\geq 1. && (19) \end{aligned}$$

In order to linearize (18) and (19) we extend the idea of (14)-(17) to work with products of several variables. Consider m binary variables $q_1, \ldots, q_m$. Whenever it appears, the product $\prod_{i \leq m} q_i$ can be replaced by a (continuous) linearization variable s as long as the following linear constraints are added to the formulation:

$$\begin{aligned} \forall i \leq m \quad s &\leq q_i && (20)\\ s &\geq \sum_{i \leq m} q_i - m + 1 && (21)\\ s &\geq 0. && (22) \end{aligned}$$

By employing the reformulation techniques explained in this session, all the nonlinear terms can be replaced by linearization variables and linear constraints without changing the set of optima of the problem. This yields a MILP formulation with a polynomial number of variables and constraints (in terms of the size of the problem instance) that models the PEGRN. We also remark that the linearized formulation only involves binary and continuous variables (i.e. no variable is integer but non-binary).

3 Computational experiments

3.1 The solution algorithm

MILP problems are usually solved by implicit enumeration via a Branch-and-Bound (BB) algorithm. A BB algorithm works by recursively partitioning the domain of selected integer variables. This gives rise to a search tree whose corresponds to sub-problems where a subset of the variables are subject to tighter constraints imposed by the branching (in the case where all integer variables are binary, each node corresponds to a sub-problem where some of the variables have been fixed to either 0 or 1). A node is *fathomed* (i.e. no further branching occurs on the node) either because the global optimum restricted to the node has been found, or because the global optimum restricted to the node cannot be better than the overall best solution found so far (the *incumbent*). In order to test these two conditions at each node, we compute a lower bound and an upper bound to the objective function value of the node's problem restriction. The first condition is verified if these bounds have the same value, and the second if the lower bound for the node exceeds the incumbent (for minimization problems — for maximization problems, if the upper bound for the node is lower than the incumbent). The most important stages for a BB algorithm are the branching policies and the tightness of the lower bound (for minimization, upper bound for maximization). Branching policies are usually defined by a complex set of heuristics [1], and bounding is really a research field on its own. The CPLEX solver [2] uses a continuous relaxation which is tightened by the addition of general-purpose and instance-specific cuts (i.e. inequalities which are redundant in the original problem but valid and hopefully active in the continuous relaxation).

3.2 Results

4 Conclusion

References

[1] T. Achterberg, T. Koch, and A. Martin. Branching rules revisited. *Operations Research Letters*, 33:42–54, 2005.

[2] ILOG. *ILOG CPLEX 8.0 User's Manual*. ILOG S.A., Gentilly, France, 2002.

[3] L. Liberti. Writing global optimization software. In Liberti and Maculan [5], pages 211–262.

[4] L. Liberti. Reformulation techniques in mathematical programming, in preparation. Thèse d'Habilitation à Diriger des Recherches.

[5] L. Liberti and N. Maculan, editors. *Global Optimization: from Theory to Implementation*. Springer, Berlin, 2006.

[6] N.V. Sahinidis. Baron: Branch and reduce optimization navigator, user's manual, version 4.0. *http://archimedes.scs.uiuc.edu/baron/manuse.pdf*, 1999.

[7] H.P. Williams. *Model Building in Mathematical Programming*. Wiley, Chichester, 4th edition, 1999.