# MORE: Mixed Optimization for Reverse Engineering—An Application to Modeling Biological Networks Response via Sparse Systems of Nonlinear Differential Equations

Francesco Sambo, Marco A. Montes de Oca, Barbara Di Camillo, Gianna Toffolo, and Thomas Stützle

**Abstract**—Reverse engineering is the problem of inferring the structure of a network of interactions between biological variables from a set of observations. In this paper, we propose an optimization algorithm, called MORE, for the reverse engineering of biological networks from time series data. The model inferred by MORE is a sparse system of nonlinear differential equations, complex enough to realistically describe the dynamics of a biological system. MORE tackles separately the discrete component of the problem, the determination of the biological network topology, and the continuous component of the problem, the strength of the interactions. This approach allows us both to enforce system sparsity, by globally constraining the number of edges, and to integrate a priori information about the structure of the underlying interaction network. Experimental results on simulated and real-world networks show that the mixed discrete/continuous optimization approach of MORE significantly outperforms standard continuous optimization and that MORE is competitive with the state of the art in terms of accuracy of the inferred networks.

**Index Terms**—Reverse engineering, mixed optimization, biological networks, sparse systems of differential equations.

## 1 Introduction

Genes of the DNA encode the information necessary for the synthesis of proteins and, when a gene is activated or *expressed*, this information is carried by messenger RNA (mRNA) to the machinery responsible for protein creation. Some proteins, called *transcription factors*, have in turn the role of activating or inhibiting the expression of genes, thus closing the loop of a feedback control mechanism of the DNA called *gene regulation*.

Proteins may interact with each other to carry out different biological functions. For example, proteins may interact to form a protein complex; a protein may carry another protein (for example, from cytoplasm to nucleus); a protein may interact with another protein to modify it (for example, a protein kinase adds a phosphate to a target protein).

Modeling biological systems as networks of interacting components has become, in recent years, a leading paradigm in molecular biology [1]. Biological networks are modeled as graphs in which nodes represent the biological variables under analysis, such as gene expression or protein concentration, and edges represent interactions between the variables. Three main types of biological networks can be defined: metabolic, protein-protein interaction, and transcriptional regulatory networks.

In general, the aim of an analysis based on the biological network formalism is to infer the unknown topology of the network from protein and, more frequently, mRNA expression data; such a process is known as *reverse engineering* [2]. An example of an international effort in this direction is the DREAM initiative, i.e., Dialogue for Reverse Engineering Assessments and Methods [3], [4], [5].

A meaningful approach for the identification of new relations between a set of observed biological variables can be to search for the best fit of a system of nonlinear differential equations to the temporal profiles of the variables [6]. The problem can be cast to the one of finding the parameters of the system that minimize the fitting error. The search space corresponds to the continuous space of system parameters and the function to be minimized is a measure of the error between real and predicted temporal profiles.

The purpose of the work described in this paper is to design a reverse engineering algorithm to solve the optimization problem of fitting systems of nonlinear differential equations to biological time series.

The model adopted for the fit should be complex enough to realistically describe the dynamics of a biological system, which are known for being strongly differential and nonlinear when the system operates far from the equilibrium [7], [8], [9].

- *F. Sambo, B. Di Camillo, and G. Toffolo are with the Department of Information Engineering, University of Padova, via Gradenigo 6a, 35131 Padova, Italy.*
  *E-mail: {francesco.sambo, barbara.dicamillo, toffolo}@dei.unipd.it.*
- *M.A. Montes de Oca is with the Department of Mathematical Sciences, University of Delaware, 501 Ewing Hall, Newark, DE 19716.*
  *E-mail: mmontes@math.udel.edu.*
- *T. Stützle is with IRIDIA-CoDE, Université Libre de Bruxelles, 50, Av. F. Roosevelt, CP 194/6, B-1050 Brussels, Belgium.*
  *E-mail: stuetzle@ulb.ac.be.*

*Manuscript received 27 May 2011; revised 14 Mar. 2012; accepted 25 Mar. 2012; published online 16 Apr. 2012.*
*For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-2011-05-0135.*
*Digital Object Identifier no. 10.1109/TCBB.2012.56.*

We decided to provide the algorithm with the following features:

- Limit the effect of noise by numerically integrating the whole system of equations describing the model, rather than amplifying the noise by estimating the derivatives from temporal data [8],
- Enforce sparsity, a common feature of biological networks [10], [11], [12], on a global scale by imposing constraints on the total number of edges in the network rather than on the in- or out-degree of the single nodes,
- Allow the easy addition of a priori information on the network structure, in terms of knowledge on potential connections between the nodes; such information is often available when reverse engineering experiments are designed and analyzed and can greatly reduce the size of the search space [13].

To the best of our knowledge, a reverse engineering algorithm with all the aforementioned features has never been proposed in the literature. Among the algorithms based on dynamical models, some works rely on linear or additive systems [14], [15], [16], [17]: the power of these models is, however, limited and the authors usually make the assumption that the biological system operates near a steady state [15].

In the vast majority of the works that exploit a sufficiently complex model for system dynamics [8], [9], [18], [19], [20], [21], derivatives are estimated from temporal data, either with finite differences, data interpolation, or by approximating differential equations with algebraic equations. Without a large amount of equally spaced time points, however, such an approach is sensitive to noise in the data [8].

Sparsity of the system is usually enforced in the literature by either adding penalty terms to the cost function [18], [21], [22], [23] or constraining the maximum number of incoming edges per node [24], [25]. The former approach poses the problem of weighting the penalty terms with respect to the cost function by adding one or more parameters, which have to be tuned a posteriori and dramatically affect the outcome of the algorithm. The latter approach tends to be unsuitable for biological networks, which generally exhibit both highly connected and loosely connected nodes [26], making it difficult to set a global threshold on node connectivity.

A major exception is the work from Marbach et al. [27], [28], in which a system of log-sigmoid differential equations is fit to time-course data and the system is numerically integrated to compute temporal profiles. The authors define a strategy to implicitly encode the system in an artificial genome. A genetic algorithm lets the genome evolve until a state is reached in which the square error between real and estimated temporal profiles is sufficiently small. The implicit encoding of the system, however, provides limited control on the addition of a priori knowledge on potential edges in the network.

With the aim of acquiring a better understanding of the optimization problem and to inform the design of our algorithm, we first carried out a *fitness-distance correlation* (FDC) analysis [29], [30], i.e., a study of the features of the search landscape around the optimum. This is particularly interesting in the context of reverse engineering, since a search landscape analysis gives an idea of the problem difficulty, of the presence of local optima and on how to improve the inference process. As far as we are aware, such an analysis has never been applied to the problem under study. The analysis is described in detail in one of our previous works [31]. The main results, together with some additional, new results, are reported in this paper.

The results of the fitness landscape analysis provide evidence in favor of a decomposition of the problem into two interconnected subproblems: a discrete search of the optimal network structure in the space of connectivity matrices and a continuous search of the optimal network parameters in $\mathbb{R}^E$, where $E$ is the number of edges in the network.

Thus, in this paper, we propose Mixed Optimization for Reverse Engineering (MORE), a mixed discrete/continuous optimization algorithm composed of two interacting layers: at a higher level, an Iterated Best Improvement component searches in the discrete space of network structures; each structure is then scored, at a lower level, by a derivative-free continuous optimization algorithm, which searches for the optimal values of the continuous parameters of the system induced by the network structure.

Two candidate continuous optimization algorithms are considered from the state of the art: the NEW Unconstrained Optimization Algorithm (NEWUOA [32]) and Covariance Matrix Adaptation-Evolution Strategy (CMA-ES [33]). As shown in this paper, the latter exhibits a better performance in terms of reliability of the results and is thus chosen as the continuous search component of MORE.

A set of feedback strategies between the two components of the algorithm are designed, in order to enhance performance by caching and reusing information and by exploiting locality and problem-specific features during the search.

The presence of a separate discrete component for searching in the space of network structures allows both enforcing network sparsity, by globally constraining the total number of edges in the network, and exploiting a priori information on possible network edges, by skewing the probabilities with which these edges are sampled in the local search procedure.

The stochastic nature of the discrete optimization component is exploited to gather an ensemble of solutions, each obtained by a random restart of the discrete search component. The ensemble is exploited to estimate confidence values for each edge of the inferred network.

Both the fitness landscape analysis and an extensive assessment of the performance of MORE are carried out on a rich set of simulated data, exploiting network topologies generated with the SimBioNeT simulator [34], [35]. The analyses are performed both in the absence of simulated noise, to assess the full power of the algorithm in an ideal case, and in the presence of low, medium, and high levels of noise. MORE is then tested on a real scenario with a data set of protein activity, collected for the Predictive Signaling Network Modeling challenge of the international DREAM4 competition [3], [4], [5]. The latter data set allows us to test

the benefits of globally handling sparsity and a priori information in the search process.

The performance assessment on simulated data shows the effectiveness of the mixed optimization approach as opposed to standard continuous optimization, as soon as even a low level of noise is present in the data. Furthermore, the designed feedback strategies between the two levels of optimization are proven effective in reducing computation time. From a qualitative comparison of the results, the algorithm is shown to exhibit a performance equal or better than the state-of-the-art on problems of the same complexity. Tested on the DREAM4 data, both with and without the addition of a priori information on the network structure, MORE returns a meaningful and sparse biological network.

The remainder of the paper is organized as follows: a description of the model chosen to describe variables dynamics is given in Section 2. The simulated data set is described in Section 3. Section 4 reports the main results of the analysis of the fitness landscape for the given problem and the comparison between NEWUOA and CMA-ES. Section 5 presents the MORE algorithm. Section 6 describes the adopted performance measures and presents the experimental results on simulated data. Section 7 describes the DREAM4 data set and presents the experimental results on the real scenario. Finally, Section 8 draws conclusions and proposes some possible future directions.

## 2 Model

The model we choose for describing the temporal dynamics of a biological network is a system of nonlinear sigmoidal differential equations, formally identical to what is known as Dynamic Recurrent Neural Networks (DRNN [36], [37]) in the machine learning literature. Equations of the system have the following form:

$$\frac{dx_i}{dt} = \frac{k_{1i}}{1 + e^{-\left(\sum_{j=1\ldots n} a_{ij}x_j + \sum_{l=1\ldots m} b_{il}u_l\right)}} - k_{2i}x_i, \quad i = 1 \ldots n, \tag{1}$$

where $n$ is the number of variables in the system, $x_i$ is the activity level of variable $i$, $a_{ij}$ represents the relative effect of $x_j$ on $x_i$ $(1 \leq i, j \leq n)$, $u_l$ is the $l$-th external input to the system $(1 \leq l \leq m)$, which can be time varying, constant or null, $b_{il}$ represents the relative effect of $u_l$ on $x_i$, $k_{1i}$ is the maximal level of activity, and $k_{2i}$ is the degradation rate.

The system can be expressed in matrix form as

$$\dot{\mathbf{X}} = \mathbf{K}_1 \times \sigma(\mathbf{AX} + \mathbf{BU}) - \mathbf{K}_2 \times X, \tag{2}$$

where matrices $\mathbf{A}^{n\times n}$ and $\mathbf{B}^{n\times m}$ contain the coefficients $a_{ij}$ and $b_{il}$, $\mathbf{X}$ and $\mathbf{U}$ are the column vectors of system variables and external inputs, $\mathbf{K}_1$ and $\mathbf{K}_2$ are the column vectors of maximal activity levels and degradations rates, $\sigma$ is the nonlinear sigmoid function and $\times$ represents the element-wise product between two column vectors.

An identical model is used in [38] for the analysis of microarray data from an experiment on *Saccharomyces Cerevisiae* cell cycle, and is adopted in [9] and [19] for a reverse engineering algorithm based on Particle Swarm Optimization [39].

## 3 Simulated Data Set

A reliable assessment of the average performance of an algorithm in reconstructing a biological network needs to be carried out on a large set of known networks. This objective is still difficult to accomplish with real data: very few biological networks are known with sufficient precision and it is often difficult to exclude the effects of unobserved variables [2].

Simulated networks are generated with the simulator recently introduced in [34]: the topology is generated according to the current knowledge of biological network organization, including scale-free distribution of the connectivity[1] and a clustering coefficient independent of the number of nodes in the network [10]. Twenty networks of 5, 8, and 10 nodes were generated.

Both for simplicity and for coherence with the experiments carried out in the majority of the papers sharing a similar model [9], [19], [27], [28], the presence of external inputs is excluded in the simulated data set, thus fixing the matrix $\mathbf{B}$ from (2) to zero, and the parameters $k_{1i}$ and $k_{2i}$ are fixed to 1 for each $i$. The dynamics of the system are thus fully described by matrix $\mathbf{A}$.

Simulated networks are sparse: since each $a_{ij}$ in (1) is representative of an edge from $j$ to $i$ in the network, the majority of $a_{ij}$ are thus equal to zero; the remaining nonzero elements are sampled uniformly at random from the interval $[-a_{\max}, a_{\max}]$, where $a_{\max}$ is set so that system dynamics could reach the steady state within the simulated time.

For each network, time series are generated initializing the level of activation of each variable uniformly at random and simulating the evolution of the system, by numerically integrating the system of equations (1) with the Runge-Kutta-Fehlberg method with adaptive step size control [40]. Temporal profiles are then sampled at logarithmically spaced time points, so that the majority of samples are taken right after the initialization. This practice is common in real experiments, because meaningful information usually concentrates right after the stimulation of a dynamical system [41].

When needed in the analyses, we simulate measurement error by adding Gaussian noise to temporal dynamics, with zero mean and constant Coefficient of Variation (CV): such a model for the noise has been considered an approximation of the error observed in both gene expression microarray data [42] and protein mass spectrometry data [43] and has already been used to generate simulated data for testing several reverse engineering algorithms [18], [20], [44].

Simulated time series and corresponding system matrices are available as Supplementary Material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2012.56.

## 4 Fitness Landscape and Problem Difficulty

The term *fitness* is used in the Evolutionary Computation literature to refer to the quality of a solution with respect to

1. For networks in the size range of the ones we consider in this paper, the scale-free distribution cannot be properly defined. However, generated networks exhibit some of the main properties of scale-free networks: they are sparse, with few highly connected nodes and the others being loosely connected.

an objective function (fitness function) that has to be maximized (or minimized). A fitness landscape is, thus, the set of values on which the set of candidate solutions to an optimization problems are mapped. The features of this landscape can give us information about the difficulty of the optimization problem represented by an objective function.

In our case, the objective function is the following:

$$RSE = \frac{1}{Tn}\sum_{t=1}^{T}\sum_{i=1}^{n}\frac{[\hat{x}_i(t) - x_i(t)]^2}{x_i^2(t)}.$$

This function computes the Relative Squared Error (RSE) between the observed values of the variables at time $t$, $x_i(t)$, and the corresponding value predicted by the model, $\hat{x}_i(t)$. Such an objective function is consistent with the assumption of a constant coefficient of variation of the noise and is commonly adopted in reverse engineering algorithms [18], [20], [44], [45].

### 4.1 Fitness-Distance Correlation Analysis

A measure that gives some information about the difficulty of an optimization problem is called fitness-distance correlation (FDC) [29], [30]. To compute a problem's FDC, a set of sample solutions are randomly generated. For each sample, its fitness value and its distance to a reference point, which can be the known optimum or the best known solution to the problem, are computed. Pearson correlation between fitness and distance is then computed for the whole sample set. If a problem's FDC is sufficiently close to one, an algorithm that searches around the best-so-far-solution is expected to perform well. If the problem's FDC is close to zero or negative the problem becomes much harder, because the best-so-far solution does not give much information about which regions of the search space are promising.

We performed an FDC analysis of the search space that consists of the set of all possible continuous values for each element $a_{ij}$ of the weight-matrix **A** from (1). The fitness of a solution is inversely proportional to the RSE it generates. The euclidean distance of a solution to the optimal matrix is used as distance measure in the FDC computation.

For the FDC analysis, we exploited 20 simulated networks of 10 nodes, from each of which 10 time series of 50 time points were generated as described in Section 3. To study the complexity of the problem in the ideal condition of absence of both model and measurement error, no simulated noise was added at this stage of the analysis.

The search space around the optimal weight-matrix **A** was sampled using three schemes:

- Uninformed sampling: each element of the optimal matrix, both zeros and nonzeros, was perturbed with the addition of a log-uniformly distributed random variable. The resulting samples are equally distributed around the point representing the optimal solution in the solution space.
- Informed sampling: nonzero elements of the optimal matrix were perturbed with the addition of a log-uniformly distributed random variable. The resulting samples are distributed along the axes corresponding to zero elements of the matrix and the structure of the system is unchanged.
- Structural sampling: the optimal connectivity matrix of the system was perturbed at increasing levels of Hamming distance,[2] by replacing an increasing number of elements in the optimal weight matrix. Nonzero elements were turned to zero and zero elements were replaced with a random nonzero value.

Example scatterplots of fitness versus distance from the optimum for the three sampling schemes are represented in Fig. 1.

The three sampling schemes are meant to reveal different aspects of the search landscape: the general shape of the fitness landscape around the optimum (Fig. 1(upper panel)), how this shape changes when additional information on the network structure is present (Fig. 1(middle panel)) and how it changes with an increasing number of differences in the network structure (Fig. 1(lower panel)). It can be seen how the FDC is in general positive. However, as the structure of the network departs from the optimal one, the scatter plot tends to concentrate in a small cluster on the upper right corner of the plot, indicating an increase in problem difficulty. We can thus conclude that, while the overall FDC is positive as shown in Fig. 1, the landscape is in fact multimodal, that is, it features many local optima.

The results of our FDC analysis can be summarized as follows:

- There is a strong positive correlation between euclidean distance and RSE.
- The fitness landscape exhibits extremely deep valleys, in which a general purpose continuous optimization algorithm can be trapped.
- Portions of the search space which correspond to networks structurally closer to the optimum (i.e., at a lower Hamming distance) present a smoother fitness landscape and a higher FDC.

Consider now the case of a general purpose continuous optimization algorithm, which receives as input a connectivity matrix and which has to minimize RSE by finding the optimal continuous values for the nonzero elements of the matrix. The results of the analysis suggest that the closer the connectivity matrix is to the optimal system matrix, in terms of Hamming distance, the higher will be the probability that such an algorithm will reach low RSE values. The RSE returned by the algorithm can thus be used as a measure of the fitness of a candidate connectivity matrix.

These considerations motivate the design of the mixed optimization algorithm presented in this paper, in which a discrete search component explores the space of network structures and a continuous search component is exploited to evaluate the fitness of each structure. The continuous algorithm searches for the optimal values of the nonzero elements of the matrix and returns the minimum RSE to the discrete search component.

The analysis described next is aimed at choosing a good candidate for the continuous optimization component, comparing two algorithms from the state of the art.

2. Hamming distance between two matrices is defined as the number of bits that differ between the two matrices. In case of connectivity matrices representing network structures, it measures the number of edges that differ between the two networks.

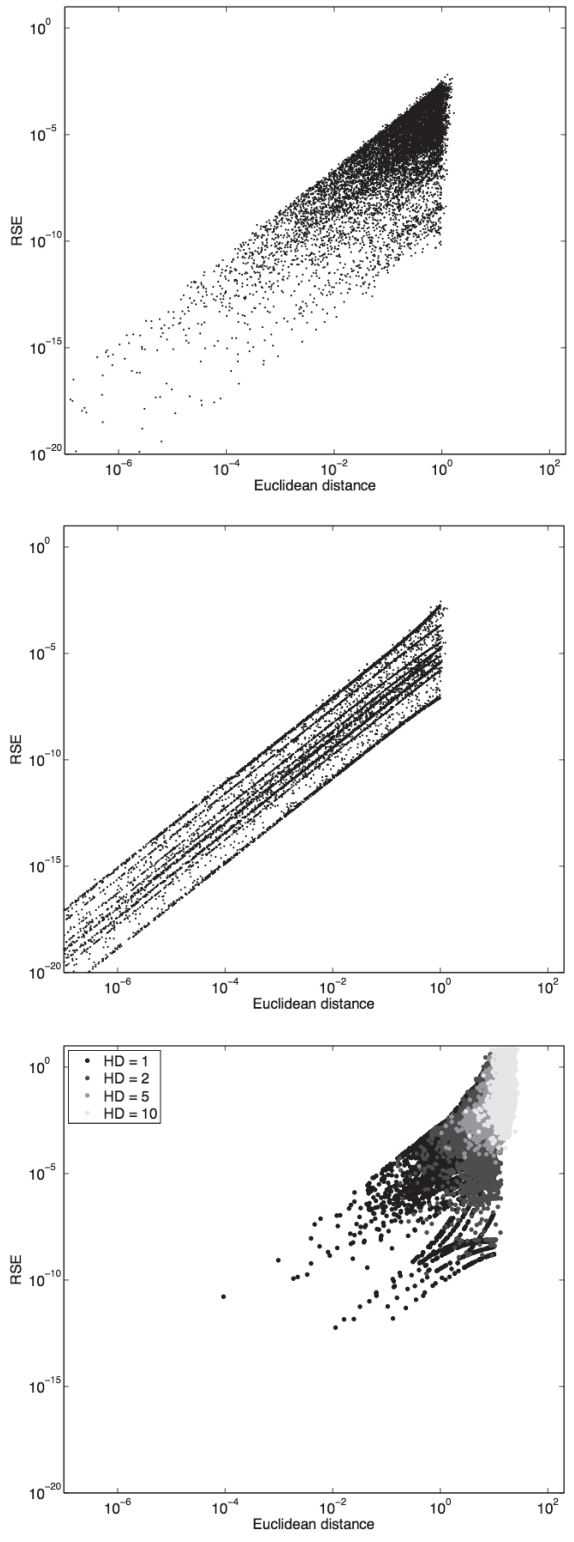



Fig. 1. Example scatterplots of fitness versus distance from the optimum, for the three sampling schemes: uninformed sampling (first panel), informed sampling (second panel), and structural sampling (third panel). In the third panel, HD stands for Hamming distance.

### 4.2 Behavior of Continuous Optimization Algorithms

In this section, we study the behavior of two algorithms, NEWUOA [32] and CMA-ES [33], when dealing with the task described at the end of the previous section: provided with a connectivity matrix, the algorithms have to minimize RSE by finding the optimal continuous values for the nonzero elements of the matrix.

The two algorithms we choose to compare are considered to be state-of-the-art for continuous derivative-free optimization [46]:

1. **NEWUOA.** NEWUOA is a trust-region method [47] for unconstrained continuous optimization. Typically, a trust-region method works as follows: at each iteration, a model of the objective function is defined over a trust region of a certain radius. The trust region is centered on the current best-so-far solution $\vec{P}$. A trial point $\vec{s}$ is then generated such that a new point $\vec{P} + \vec{s}$ reduces sufficiently the model, and it is still within the trust region. An evaluation of the objective function at $\vec{P} + \vec{s}$ is performed, and the value returned is compared to the prediction of the model. If the prediction is sufficiently accurate, the new point is accepted as the new best-so-far solution, and the next iteration is executed. In the next iteration, the trust-region radius can be larger or equal to the one used in the previous iteration. However, if the prediction is not sufficiently accurate, the new point is rejected and the next iteration is executed with a reduced trust-region radius. For approximating an $n$-dimensional objective function, NEWUOA uses a quadratic interpolation of $\mathrm{O}(n)$ points within the trust region (a common value being $2n + 1$ points). This fact makes NEWUOA useful for tackling large-scale optimization problems [32]. In addition to the number of interpolation points, the initial and final trust region radii are parameters of the method.
2. **CMA-ES.** The Covariance Matrix Adaptation-Evolution Strategy algorithm belongs to the class of population-based optimization algorithms called Evolution Strategies. In this kind of algorithms, a population of solutions is sampled from a multivariate normal distribution, with mean $m$ and covariance matrix $C$, for a certain number of generations; at each generation, the best individuals are selected and used to adapt the sampling mechanism, in order to select potentially better solutions. In CMA-ES, the covariance matrix is dynamically adapted with three concurring strategies: a) reproduce the best steps in the search space from the current generation, b) reproduce the set of moves from the previous generations and c) obtain uncorrelated steps with step size control. The effect of the adaptation is to skew the sampling distribution so to obtain the highest variance along the steepest direction of the search space, remaining robust to badly scaled problems and avoiding premature convergence.

We study the behavior of NEWUOA and CMA-ES when provided with the same time series used for the FDC analysis and with the connectivity matrices of the optimal networks, their aim thus being to find the correct values for system parameters by minimizing RSE. Both algorithms are given the same maximum number of function evaluations.

Fig. 2 shows the boxplots of RSE values achieved by 20 runs of both algorithms, with the maximum number of fitness function evaluations set to $4 \times 10^4$. From the figure, it

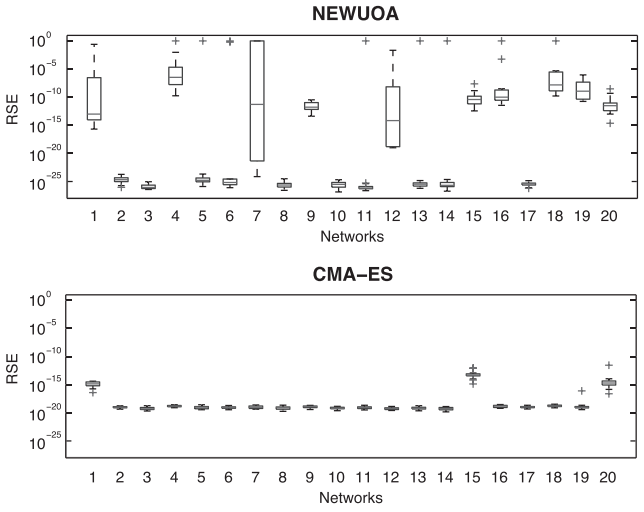



Fig. 2. Boxplots of RSE reached in 20 runs of NEWUOA and CMA-ES on 20 network instances of 10 nodes, with complete information on the topology of the networks; the maximum number of function evaluations for both algorithms is set to $4 \times 10^4$.

is clear that CMA-ES obtains RSE values smaller than $10^{-15}$ in almost every run; NEWUOA, on the other hand, gets better results for some network structures, but for several others it is very poorly performing when compared to CMA-ES.

The RSE value reached by CMA-ES on a candidate structure, even with a limited number of fitness function evaluations, can thus be considered a reliable and stable estimate of the fitness of the structure; the same is not true for NEWUOA.

## 5 Mixed Optimization Algorithm

The analysis of the fitness landscape described in the previous section provides evidence in favor of a decomposition of the problem of biological network inference into two interconnected subproblems: a discrete search of the optimal network structure in the space of connectivity matrices and a continuous search of the optimal network parameters in $\mathbb{R}^E$, where $E$ is the number of edges in the network. Moreover, the study of the performance of two continuous optimization algorithms, NEWUOA and CMA-ES, suggested that

- CMA-ES exhibits a more stable behavior, when provided with the correct topology and a limited number of fitness function evaluations.
- The RSE returned by CMA-ES on a candidate network structure can be used as an estimate for the fitness of that particular structure.

In this section, we present MORE, an implementation of such a mixed discrete/continuous optimization approach: Section 5.1 describes a basic implementation, exploiting an Iterative Best Improvement Local Search strategy with Multiple Restarts for the discrete search step and CMA-ES for the continuous search step; in Section 5.2, we present a set of feedback strategies between the two optimization layers that enhance the performance of the base algorithm; Section 5.3 presents the ensemble strategy we adopted for estimating the confidence on each edge of the resulting network from multiple runs of the discrete optimization step; Section 5.4 shows the pseudocode of the complete algorithm.

### 5.1 Base Algorithm

The Iterative Best Improvement Local Search strategy belongs to the class of Stochastic Local Search (SLS [48]) algorithms, in which a complex or high-dimensional search space is locally explored, starting from a random solution in the search space and iteratively moving in the direction that locally minimizes (or maximizes) an objective function. From the current position, SLS algorithms iteratively generate a set of *neighbors*, i.e., solutions that do not differ much from the current one, and replace the current solution with one of the neighbors, based on some acceptance criterion. The process is iterated until no further improvement of the current solution can be found; in this case, the current solution is called a *local optimum*.

The various SLS algorithms differ in the strategies they adopt to explore the neighborhood, to choose the next solution and to avoid local optima. In the Iterative Best Improvement Local Search strategy, the current solution is iteratively replaced with the best improving solution from its neighborhood and the process is repeated until a local optimum is reached. Restaring multiple times the algorithm from different random solutions guarantees that the algorithm, if provided with a sufficiently large number of restarts, eventually converges to the *global optimum*, i.e., the best solution in the search space [48].

In our case, the discrete search space consists of all Boolean connectivity matrices pairs $(\mathbf{A}_{nz}, \mathbf{B}_{nz})$ ($nz$ stands for *nonzero*) from (2), of size $n \times n$ and $n \times m$, where $n$ is the number of observed variables and $m$ is the number of inputs.

A set of constraints are imposed on feasible matrices pairs:

- each row of either $\mathbf{A}_{nz}$ or $\mathbf{B}_{nz}$ has at least one element equal to one, thus each variable is controlled by at least one other variable or one input (and the number of nonzero elements in $(\mathbf{A}_{nz}, \mathbf{B}_{nz})$ is greater than or equal to $\min(n, m)$),
- the number of nonzero elements in $\mathbf{A}_{nz}$ and $\mathbf{B}_{nz}$ is smaller than the user-defined values $M_A$ and $M_B$, respectively, to force global sparsity of the network,
- elements on the diagonal of $\mathbf{A}_{nz}$ are zero, i.e., self-interaction is not allowed.[3]

Initial matrices are sampled at random: the probability distribution used for sampling can be exploited to consider additional knowledge on network structure. When no information is available, the sampling probability is uniform for all edges and such that the majority of sampled networks are feasible. If additional knowledge on the presence (or absence) of edges is present, probabilities can be increased (or decreased) accordingly.

Initial solutions are then locally perturbed to generate a set of neighboring solutions: as perturbations, we chose all possible flips of one bit in the connectivity matrices pair, excluding the flips that lead to unfeasible solutions. The number of solutions in each neighborhood is thus $O[n(n+m)]$.

Nonzero parameters of each candidate structure, together with parameters $k_{1i}$ and $k_{2i}$ for each variable $i$, are optimized with CMA-ES; the returned RSE between observed and estimated temporal profiles is used as cost

3. This last constraint is in line with the choices made by the DREAM team [5], but can be relaxed without affecting the global behavior of the algorithm.

function for the structure. The neighbor solution with the lowest RSE value is chosen as origin of the new neighborhood, and the process is iterated until no improving solution can be found in the current neighborhood.

The complete algorithm is then restarted a maximum of $R$ times, to increase the possibility of exploring interesting regions of the search space.

### 5.2 Enhanced Algorithm

In the basic version of our mixed algorithm, the two optimization components, Iterated Best Improvement Local Search and CMA-ES, communicate with each other exchanging candidate structures and RSE values. In this section, we describe a set of feedback techniques for increasing the information exchange between the two components and enhancing the overall algorithm behavior.

#### 5.2.1 Discrete to Continuous Optimization Feedback

To exploit locality and cache previously computed results while evaluating the neighborhood of a candidate network structure, we developed the following feedback strategies from the discrete to the continuous optimization layer:

- Each neighbor differs from the current solution by just one bit, i.e., by the presence or absence of an edge in the corresponding network. Since the network is sparse, there can be regions of the network which are not affected by the edge modification, being upstream in the signal flow through the network. Continuous values for the parameters corresponding to edges in these regions do not need to be reoptimized by CMA-ES and are thus kept fixed.
- For the parameters that need to be reoptimized in each neighbor, initial search values for CMA-ES are chosen equal to the ones obtained when evaluating the current solution, under the assumption that optimal parameters values for two networks that differ by just one edge are probably close to each other.

The motivation for these strategies is that CMA-ES, when used to optimize a smaller set of values, tends to converge faster. Furthermore, starting from a good search point has a positive effect on continuous optimization, given the positive fitness-distance correlation reported in Section 4.

#### 5.2.2 Continuous to Discrete Optimization Feedback

To provide additional feedback from CMA-ES, thus reducing the size of the search space for subsequent iterations and restarts of the Best Improvement Local Search, we developed the following feedback strategies from the continuous to the discrete optimization layer:

- When all solutions in the neighborhood have been evaluated, some elements of the best neighbor solution may be close to zero, i.e., with absolute value below a fixed threshold. If that is the case, when the center of the next neighborhood is chosen, these elements are set to zero, thus implementing a longer move toward more promising regions of the search space.
- Elements of matrices $\mathbf{A}_{nz}$ and $\mathbf{B}_{nz}$, when set to zero by the previous strategy, are kept to zero for the subsequent iterations and restarts of the Local Search algorithm. Note that, on the contrary, elements flipped to zero by the local search procedure can still be reflipped to one.

### 5.3 Ensemble Strategy

The result of $R$ restarts of the discrete optimization component consists of an ensemble of $R$ matrix pairs, each one with a locally optimal structure and with nonzero elements estimated by the continuous optimization component. Rather than simply taking the matrix pair with the lowest RSE, we decided to adopt the *signed voting* strategy described in [28] for estimating the confidence on each of the edges of the resulting network from the whole ensemble of solutions. Such an approach, in the presence of noise in the data, was proven to lead to network predictions which are more accurate than any of the individual networks taken alone [28].

From each matrix pair $i$, $i = 1 \dots R$, we retain the network structure and the sign of each edge in the signed connectivity matrices $(\mathbf{A}^s_{nz}, \mathbf{B}^s_{nz})_i$ ($s$ stands for *signed*): elements of the matrices are set to 1 if the corresponding edges are present in the inferred networks with positive sign, $-1$ when present with negative sign and 0 otherwise. The signed voting scheme consists in summing each element of the signed connectivity matrix pairs across the $R$ restarts and taking the absolute value of the result: this measure can be considered an indicator of the confidence on the corresponding edge, ranging from zero to $R$. The computational complexity of the signed voting strategy is $O(n^2 R)$, where $n$ is the number of nodes in the network.

### 5.4 Pseudocode

The pseudocode of the complete algorithm is the following:

```
D-SEARCH(D, U, R)
 1  S^s_ens = ∅
 2  for i = 1 to R
 3        Sample a feasible solution S_curr = (A_nz, B_nz)
 4        rows = Boolean vector, each value set to TRUE
 5        c_lbest = C-SEARCH(D, U, S_curr, rows)
 6        c_nhbest = c_lbest
 7        while improvement
 8           improvement = FALSE
 9           for each element e ∈ S_curr
10              S_nh = FLIP(S_curr, e)
11              if ISFEASIBLE(S_nh)
12                 rows = PROPAGATE(S_curr, e)
13                 c_nh = C-SEARCH(D, U, S_nh, rows)
14                 if c_nh < c_nhbest
15                    c_nhbest = c_nh
16                    S_nhbest = S_nh
17                    improvement = TRUE
18                    ZERO-SET(S_nhbest)
19              if improvement
20                 c_lbest = c_nhbest
21                 S_curr = S_nhbest
22        S^s_i = (A^s_nz, B^s_nz)_i = SIGN(S_curr)
23        S^s_ens = S^s_ens ∪ S^s_i
24  S_conf = SIGNED-VOTE(S^s_ens)
25  return S_conf
```

C-SEARCH($\mathbf{D}, \mathbf{U}, \mathbf{S} = (\mathbf{A}_{nz}, \mathbf{B}_{nz}), \mathbf{rows}$)
// optimize nonzero elements of $\mathbf{S}$ and
// all the elements of $\mathbf{K}_1$ and $\mathbf{K}_2$
1 $size =$ NONZEROS($\mathbf{S}, \mathbf{rows}$) $+ 2n$
2 **if** first call of the procedure
// Evaluating $\mathbf{S}_{curr}$
3 $x_{start} = size$ random values
4 **else**
// Evaluating $\mathbf{S}_{nh}$
5 **if** the current bit flip is $1 \rightarrow 0$
6 $x_{start} = (size - 1)$ optimal values for $\mathbf{S}_{curr}$
7 **else**
8 $x_{start} = size$ optimal values for $\mathbf{S}_{curr}$, plus an additional random value.
9 $cost =$ CMA-ES($size, x_{start}, \mathbf{D}, \mathbf{U}$)
10 return $cost$

The discrete optimization procedure receives as input the data matrix of time course experiments $\mathbf{D}^{n \times T}$, the temporal profiles of inputs $\mathbf{U}^{m \times T}$ and the number of Local Search restarts $R$ and returns a pair of confidence matrices $\mathbf{S}_{conf} = (\mathbf{A}_{conf}^{n \times n}, \mathbf{B}_{conf}^{n \times m})$, where each element of the two matrices reports the confidence on the presence of the corresponding edge in the network, ranging from 0 to $R$. In each of the $R$ iterations from line 2 of D-SEARCH, an initial solution is sampled at random and evaluated (lines 3-6); at each iteration, the process of neighborhood generation (10-12), neighbor evaluation (lines 13-18) and update of the current solution (lines 19-21) is repeated until a local optimum is reached.

The procedure FLIP($\mathbf{S}, e$) flips the bit corresponding to edge $e$ in either $\mathbf{A}_{nz}$ or $\mathbf{B}_{nz}$ to its complement. ISFEASIBLE($\mathbf{S}$) returns $true$ if $\mathbf{S}$ does not violate the constraints reported in Section 5.1. PROPAGATE($\mathbf{S}, e$) propagates the effects of the bit flip $e$ in the matrices $\mathbf{A}_{nz}$ and $\mathbf{B}_{nz}$: a bit flip consists in the addition or the removal of an edge in the network, thus all edges downstream of the modification are affected and PROPAGATE($\mathbf{S}, e$) returns TRUE for each row corresponding to the destination of an affected edge.

The procedure ZERO-SET($\mathbf{S}$) sets to zero the elements of $\mathbf{A}_{nz}$ and $\mathbf{B}_{nz}$ which have been estimated as close to zero by C-SEARCH. These elements are avoided during the neighborhood generation (line 10) in all the subsequent iterations of the algorithm, thus greatly reducing the size of the search space.

The procedure SIGN($\mathbf{S}$), applied to the local optima resulting from each restart of D-SEARCH, returns a pair of signed connectivity matrices $(\mathbf{A}_{nz}^s, \mathbf{B}_{nz}^s)$ and the ensemble of signed connectivity matrices is saved in $\mathbf{S}_{ens}^s$. The procedure SIGNED-VOTE, applied to $\mathbf{S}_{ens}^s$, computes the confidence on each edge by summing the $R$ signed connectivity matrices and taking the absolute value of the results.

The continuous optimization procedure is exploited to evaluate candidate network structures: the procedure searches for the optimal values of nonzero elements of matrices $\mathbf{A}_{nz}$ and $\mathbf{B}_{nz}$ and of the elements of matrices $\mathbf{K}_1$ and $\mathbf{K}_2$. When called for the first time, the procedure starts from a complete random vector[4] $x_{start}$. The procedure scores its progress with the RSE between real-time course data $\mathbf{D}$ and time course data generated with the current values for the set of parameters to be optimized; the optimization is carried out through the CMA-ES algorithm.

Subsequent calls of C-SEARCH need only to optimize nonzero elements of the rows of $\mathbf{A}$ and $\mathbf{B}$ indicated by $\mathbf{rows}$ (C-SEARCH, line 1) and can exploit the parameters values optimized in the previous function call as starting point for the search (lines 5-8).

## 6 Results on Simulated Data

To assess the average performance of MORE in reverse engineering, we test it on a simulated data set generated as described in Section 3. The effectiveness of the mixed optimization strategy is assessed by comparing the results of MORE with the ones of a standard continuous optimization approach, obtained by running CMA-ES on the entire system matrix (excluding the diagonal, which is fixed to zero). Furthermore, the effectiveness of the enhancements described in Section 5.2 is assessed by comparing the results of the base and enhanced version of MORE.

Simulated data are composed of temporal profiles generated from three groups of 20 networks of size 5, 8, and 10 nodes each. Problem sizes are in line with experimental results from the state of the art [9], [19], [45]. From each network, we generate four sets of five time series: each time series is obtained by randomly initializing the system and by sampling its temporal evolution at 10 logarithmically spaced time points, thus each set consists of 50 time points. Simulated noise with constant coefficient of variation is then added to temporal profiles: we consider the four cases of no ($CV = 0\%$), low ($CV = 2\%$), medium ($CV = 5\%$) and high ($CV = 10\%$) noise.

We set the number $R$ of random restarts of the two versions of MORE to 20. As mentioned in Section 3, the dynamics of the simulated systems are described just by the matrix $\mathbf{A}^{n \times n}$: to enforce the sparsity of the network, we set the maximum number $M_A$ of nonzero elements of $\mathbf{A}$ to $2n$.

For a fair comparison, in the standard continuous optimization approach we consider the results of 20 random restarts of CMA-ES. No limit on the maximum number of function evaluations is given to CMA-ES in this case, thus relying only in its internal stopping criteria.[5]

When the result of a reverse engineering algorithm consists of a set of confidence levels, one for each edge, it is common practice [17], [28] to compare different algorithms by means of the Area Under the Precision versus Recall curve (AUC PvsR). Precision (P) and recall (R) are widely used in the reverse engineering community to assess algorithmic performance [2] and are defined as follows:

4. The elements of matrices $\mathbf{A}$ and $\mathbf{B}$ are sampled from normal distributions $\mathcal{N}(0, a_{\max}/3)$ and $\mathcal{N}(0, b_{\max}/3)$, where $a_{\max}$ and $b_{\max}$ are the maximum allowed absolute values for the elements of matrices $\mathbf{A}$ and $\mathbf{B}$, respectively. The elements of matrices $\mathbf{K}_1$ and $\mathbf{K}_2$ are sampled from the log-normal distribution $10^{\mathcal{N}(0, k_{\max}/3)}$, for allowed elements of $\mathbf{K}_1$ and $\mathbf{K}_2$ in the range $[10^{-k_{\max}}; 10^{k_{\max}}]$.

5. Execution halts if the difference between the best values in two consecutive generations is less than $10^{-17}$ or if one element of the diagonal of the covariance matrix increases more than three orders of magnitude.

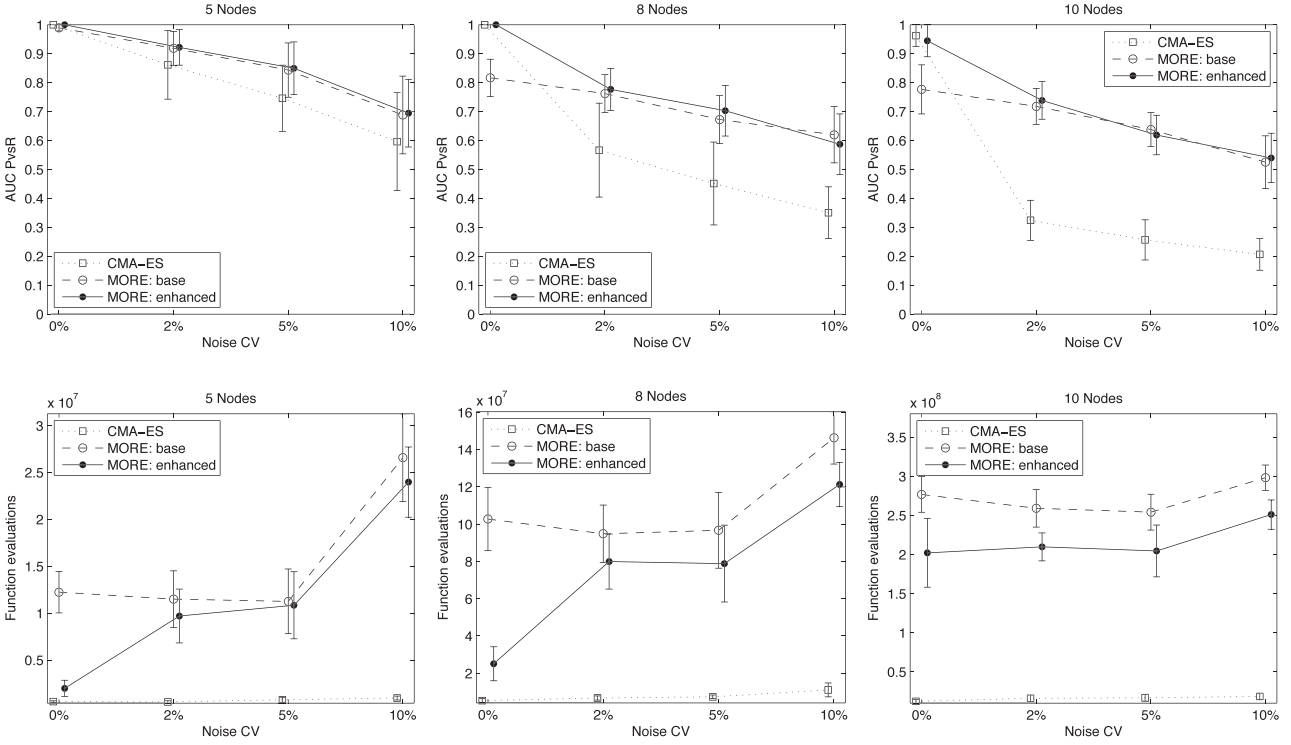



Fig. 3. (Top row) Median of the area under the Precision versus Recall curve obtained by CMA-ES and by the base and enhanced version of MORE on four runs for 20 networks of size 5 (first column), 8 (second column), and 10 (third column), for different coefficient of variations of the simulated noise. Each run consists of 20 random restarts of either the discrete optimization procedure (for the two versions of MORE) or of the CMA-ES algorithm. (Bottom row) Number of objective function evaluations needed to complete the 20 random restarts. Whiskers in all plots extend to $\pm 1$ median absolute deviation.

$$P = \frac{tp}{tp + fp},$$

$$R = \frac{tp}{tp + fn},$$

where *tp* is the number of true positives, i.e., the number of relations correctly identified by the algorithm, *fp* is the number of false positives, i.e., the number of relations identified by the algorithm which are not correct, and *fn* is the number of false negatives, i.e., the number of relations present in the real network but not identified.

The Area Under the P versus R curve is obtained by sorting the edges in decreasing order of confidence, adding them to the network one at a time, computing each time P and R of the partial networks, drawing them on the P versus R plane and computing the area under the resulting curve. For the two versions of MORE, we exploit directly the confidence matrix returned by the algorithm, while for CMA-ES we merge the results of the 20 runs by averaging the values of each weight matrix element across the different runs.

The whole process of 20 subsequent random restarts of the three algorithms is repeated, with different levels of noise, on the four sets of time series generated for each of the 20 simulated networks and for each problem size. Together with the area under the P versus R curve, we measure also the number of objective function evaluations needed by each algorithm to complete the 20 random restarts.

All tests are run on a single Intel Xeon E5410 quad core 2.33 GHz processor. The average computational time for a function evaluation, consisting in the generation of temporal profiles from the estimated model and of the computation of RSE, is independent of the reverse engineering algorithm and is approximately equal to 0.52, 0.79, and 1.00 ms for networks of size 5, 8, and 10, respectively.

Fig. 3(top row) reports the median $\pm$ median absolute deviation[6] of the area under the P versus R curve obtained by the three algorithms for networks of sizes 5, 8, and 10 and for the different levels of noise. Fig. 3(bottom row) reports the median $\pm$ median absolute deviation of the number of function evaluations needed by the three algorithms to complete the 20 restarts.

All comparisons between the algorithms are carried out with paired Wilcoxon signed-rank tests: differences are considered significant for p-values $< 0.05$.

Analyzing first the behavior of CMA-ES, one can observe that the algorithm is considerably faster than both versions of MORE in reaching convergence (all p-values $< 2 \times 10^{-14}$) but that its performance is strongly affected by noise: performance is close to optimal in the absence of noise (median AUC PvsR $> 0.96$ for all network sizes) but, as long as even low levels of noise are present in the data, it quickly deteriorates and becomes significantly lower than the one of both versions of MORE (all p-values $< 10^{-4}$).

Comparing the two versions of MORE, one can observe that the increase in AUC PvsR of the enhanced version over the base version is significant only for lower levels of noise (p-values $< 0.05$ for $CV = 0\%$ on networks of five nodes, for $CV = 0\%, 2\%, 5\%$ on networks of eight nodes and for

6. Median and median absolute deviation, rather than average and standard deviation, are chosen because they are better suited to represent distributions that can be far from Gaussian.

$CV = 0\%$ on networks of 10 nodes), but that the number of objective function evaluations required by the enhanced version is always significantly lower than the ones required by the base version (all p-values $< 2 \times 10^{-11}$).

Experimental results, thus, show that the mixed discrete/continuous optimization approach outperforms standard continuous optimization, as long as noise is present in the data. Furthermore, results prove the effectiveness of the feedback strategies between the two optimization components in terms of reduction of computation time.

Comparing MORE with the state of the art is not straightforward: we identified four algorithms from the literature, namely Marbach et al. [27], Chen et al. [21], and Xu et al. [9], [19], which are all based on stochastic local search methods and infer dynamical systems similar to the one we adopted. In the aforementioned papers, performance is assessed on simulated and real time series of 26 to 90 samples, generated from one or two networks of 4 to 11 nodes. Accuracy, as in our case, is close to optimal for smaller networks and lower levels of noise and decreases as network size and noise increase, in a range similar to the one observed for MORE.

Quantitative comparison would require to assess all algorithms on the same set of data and networks, but source code is not available and the associated publications do not contain enough information to fully reimplement the proposed algorithms. Furthermore, noise models used to simulate test data are different and complete details for replicating the same noise conditions are missing in the text.

However, from a qualitative comparison, we can conclude that MORE is competitive with other state-of-the-art approaches.

## 7 Results on Real Data

To study the behavior of MORE on a real scenario, we tested it on the Predictive Signaling Network Modeling challenge of the DREAM4 competition [3], [4], [5].

The data set consists of a set of time series of protein activity level of HepG2 cell lines, gathered after a number of systematic perturbations of the biological system.

The activity of seven phosphoproteins (AKT, ERK12, Ikb, JNK12, p38, HSP27, MEK12) is measured at three time points (0, 30 minutes, and 3 hours) during 25 different perturbations: each perturbation consists in the combinatorial treatment of the system with zero or one cytokine (TNFa, IL1a, IGF1, TGFa) acting as a stimulus and zero or one inhibitor (MEKi, p38i, PI3Ki, IKKi).

We model protein level dynamics, rescaled in the range $[0, 1]$ by dividing each sample for the saturation limit of the detector (29,000), with the complete system from (1): the seven phosphoproteins are modeled as observed variables $x_1 \dots x_7$, whereas the four cytokines and the two inhibitors PI3Ki and IKKi are modeled as constant inputs $u_1, \dots, u_6$, fixed to either 1 or 0 if present or absent in the particular perturbation experiment. Inhibitors MEKi and p38i, directly targeting two observed variables, are treated differently: when the inhibitor is present, the inhibited variable is set to zero and kept constant for the whole perturbation experiment.

The error function to be minimized is Normalized Squared Error (NSE), defined by DREAM4 organizers as

$$NSE = \frac{1}{Tn} \sum_{t=1}^{T} \sum_{i=1}^{n} \frac{[\hat{x}_i(t) - x_i(t)]^2}{300^2 + [0.08 \cdot x_i(t)]^2}.$$

In addition to protein expression time series, DREAM4 organizers provided a network of canonical signaling pathways relating the observed variables, gathered from the literature (Fig. 4(Top)). It has to be stressed that this is not necessarily the true network underlying the data, which can possibly depend on the specific cell line: the canonical network has in fact to be customized with the addition or removal of edges, to accurately represent the provided data set.

To enforce network sparsity while keeping the prior network feasible, we set the maximum number of nonzero elements for matrices $\mathbf{A}$ and $\mathbf{B}$, $M_A$ and $M_B$, to 7 and 23, respectively. The number $R$ of restarts of the discrete optimization component was set to 20.

We first run the full procedure without any a priori information on network structure, by sampling sparse initial networks for each random restart from the complete graph. The result of the ensemble learning procedure is represented in Fig. 4(bottom left): the gray levels are proportional to the confidence on each edge. Edges with confidence below 40 percent are not considered as in [28], in which the signed voting ensemble strategy is proposed.

As it is clear from the figure, the network is sparse and the majority of the identified edges are present also in the custom network (solid lines). Only one edge that is not present in the canonical network was identified (dashed line).

The canonical network was then exploited as a priori information, by letting the algorithm sample the edges of the initial networks for each random restart only from the edges of the canonical network. Please note that this does not prevent the algorithm from adding edges not present in the canonical network during the Local Search procedure.

The result of the ensemble learning procedure is represented in Fig. 4(bottom right). As it is clear from the figure, the algorithm removed 11 edges from the canonical network and added only one edge, the same added when no a priori information was provided.

Moreover, the edges inferred in the former case (Fig. 4(bottom left)) are an exact subset of the ones inferred in this case, with large agreement on the confidence levels, thus suggesting consistency in the results.

Finally, we compared the NSE score of the two final networks with the score of the single networks in the ensemble, each one resulting from a single restart of the discrete search procedure. The NSE of the network obtained without a priori information (Fig. 4(bottom left)) lies in the lowest quartile of the scores of the networks in the ensemble, while the network obtained exploiting a priori information (Fig. 4(bottom right)) has a NSE score lower than the one of any of the single networks.

## 8 Conclusions

In this paper, we propose MORE, a mixed discrete and continuous optimization algorithm for the problem of

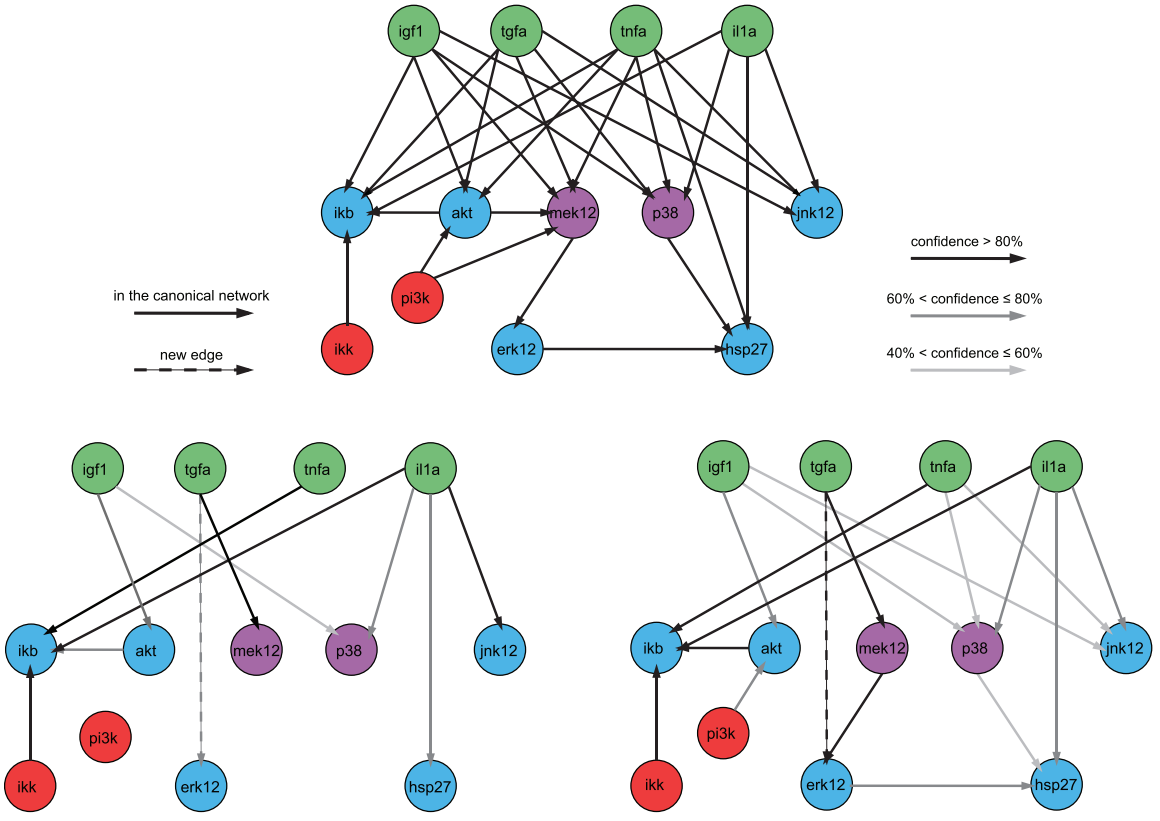



Fig. 4. (Top) Canonical network. (Bottom left) Network inferred without a priori information. (Bottom right) Network inferred with the canonical network as a priori information. The color code for the nodes is: green—stimuli, red—unobserved inhibited proteins, blue—observed proteins, magenta—observed inhibited proteins. Gray levels of the edges are proportional to the confidence on them. Dashed edges are inferred by MORE but are not present in the canonical network.

fitting a sparse system of nonlinear differential equations to biological time series.

MORE is made of two interacting layers, an Iterated Best Improvement Local Search component, which searches in the discrete space of network structures, and a Covariance Matrix Adaptation-Evolution Strategy component for the optimization of continuous system parameters. A set of additional feedback strategies between the two layers were designed for the algorithm, to cache and reuse information gathered during the exploration of the search space and to exploit locality and problem-specific features. The stochasticity of the local search component is exploited to gather an ensemble of network solutions, from which confidence values for the edges are computed with a voting strategy.

The particular design of the algorithm is based on an analysis of the fitness landscape, which suggests the effectiveness of such an approach in solving the reverse engineering problem.

Keeping separate the two tasks of searching in the space of network structures and of optimizing continuous parameters allows us to easily handle the optimization of the system matrix as a whole, rather than decomposing the problem and solving it for each variable. This strategy has three main advantages: first, the numerical integration of the whole system limits the effects of noise and requires fewer time points for accurately estimating the temporal evolution of the system. Second, system sparsity can be handled globally by limiting the total number of edges in the network. Third, a priori information on network structure can be seamlessly exploited in the search process.

Extensive testing on a rich set of simulated time course experiments proved the effectiveness of the mixed discrete and continuous optimization approach: as long as even low levels of noise are present in the data, in fact, MORE significantly outperforms a state-of-the-art continuous optimization algorithm (CMA-ES) in terms of accuracy of the results. Moreover, the designed feedback strategies between the two optimization layers are proven effective in significantly reducing the computation time. As far as a comparison is possible, we showed that MORE is competitive with other state-of-the-art approaches on problems of the same size.

It is worth mentioning, moreover, that the many papers proposing novel reverse engineering algorithms assess performance on a small set of networks, usually not more than 3 or 4. We believe that a richer data set, like the one adopted in this paper, is needed to properly assess a reverse engineering algorithm.

The application of MORE to a real data set of protein activity levels demonstrated its ability in handling a priori information and network sparsity. Without a priori information, the algorithm identified a network of interaction between proteins close to the canonical pathway, retaining edges that could be specific of the particular cell line and excluding potentially unessential edges. The introduction of a priori information results in an additional set of edges in

the inferred network, whose importance is supported by a lower error in estimating the temporal profiles.

We believe that the effective exploitation of a priori information will be a key factor for the application of our algorithm to larger biological systems, by iteratively solving smaller subsystems and exploiting the results as a priori information for the solution of larger systems.

## ACKNOWLEDGMENTS

This work was partially supported by CARIPARO 2008/2010 "Systems biology approaches to infer gene regulation from gene and protein time series data." The funders had no role in the study design, data collection and analysis, decision to publish, or preparation of the manuscript. The authors would like to thank Julio Saez-Rodriguez, Leonidas Alexopoulos, and Peter Sorger for their permission to publish the results on the Predictive Signaling Network Modeling challenge data set of the DREAM4 competition. Thomas Stützle acknowledges support from the Belgian F.R.S.-FNRS, of which he is a research associate. Francesco Sambo would like to thank Professor Silvana Badaloni for her precious advice and for letting this collaboration happen. Source code of the MORE algorithm is available upon request at sambofra@dei.unipd.it.

## REFERENCES

[1] L. Hunter, "Life and Its Molecules: A Brief Introduction," *AI Magazine,* special issue on AI and bioinformatics, vol. 25, no. 1, pp. 9-22, 2004.

[2] N. Soranzo, G. Bianconi, and C. Altafini, "Comparing Association Network Algorithms for Reverse Engineering of Large-Scale Gene Regulatory Networks: Synthetic versus Real Data," *Bioinformatics,* vol. 23, no. 13, pp. 1640-1647, July 2007.

[3] L.G. Alexopoulos, J.Saez Rodriguez, B.D. Cosgrove, D.A. Lauffenburger, and P.K. Sorger, "Networks Inferred from Biochemical Data Reveal Profound Differences in TLR and Inflammatory Signaling between Normal and Transformed Hepatocytes," *Molecular and Cellular Proteomics,* vol. 9, no. 9, pp. 1849-65, 2010.

[4] J. Saez-Rodriguez, L.G. Alexopoulos, J. Epperlein, R. Samaga, D.A. Lauffenburger, S. Klamt, and P.K. Sorger, "Discrete Logic Modelling as a Means to Link Protein Signalling Networks with Functional Analysis of Mammalian Signal Transduction," *Molecular Systems Biology,* vol. 5, article 331, Dec. 2009.

[5] R.J. Prill, D. Marbach, J. Saez-Rodriguez, P.K. Sorger, L.G. Alexopoulos, X. Xue, N.D. Clarke, G. Altan-Bonnet, and G. Stolovitzky, "Towards a Rigorous Assessment of Systems Biology Models: The DREAM3 Challenges," *PLoS ONE,* vol. 5, no. 2, e9202 (epub), 2010.

[6] I.-C. Chou and E.O. Voit, "Recent Developments in Parameter Estimation and Structure Identification of Biochemical and Genomic Systems," *Math. Biosciences,* vol. 219, no. 2, pp. 57-83, 2009.

[7] H.V. Westerhoff, A. Kolodkin, R. Conradie, S.J. Wilkinson, F.J. Bruggeman, K. Krab, J.H. van Schuppen, H. Hardin, B.M. Bakker, M.J. Moné, K.N. Rybakova, M. Eijken, H.J. van Leeuwen, and J.L. Snoep, "Systems Biology toward Life in Silico: Mathematics of the Control of Living Cells," *J. Math. Biology,* vol. 58, nos. 1/2, pp. 7-34, Jan. 2009.

[8] S. Kimura, S. Nakayama, and M. Hatakeyama, "Genetic Network Inference as a Series of Discrimination Tasks," *Bioinformatics,* vol. 25, no. 7, pp. 918-925, 2009.

[9] R. Xu, D. Wunsch II, and R. Frank, "Inference of Genetic Regulatory Networks with Recurrent Neural Network Models Using Particle Swarm Optimization," *IEEE/ACM Trans. Computational Biology and Bioinformatics,* vol. 4, no. 4, pp. 681-692, Oct.-Dec. 2007.

[10] M. Arnone and E. Davidson, "The Hardwiring of Development: Organization and Function of Genomic Regulatory Systems," *Development,* vol. 124, pp. 1851-1864, 1997.

[11] A.-L. Barabasi and R. Albert, "Emergence of Scaling in Random Networks," *Science,* vol. 286, no. 5439, pp. 509-512, Oct. 1999.

[12] E. Ravasz, A.L. Somera, D.A. Mongru, Z.N. Oltvai, and A.L. Barabasi, "Hierarchical Organization of Modularity in Metabolic Networks," *Science,* vol. 297, no. 5586, pp. 1551-1555, Aug. 2002.

[13] P. Le Phillip, A. Bahl, and L.H. Ungar, "Using Prior Knowledge to Improve Genetic Network Reconstruction from Microarray Data," *Silico Biology,* vol. 4, no. 3, pp. 335-353, 2004.

[14] P. D'Haeseleer, X. Wen, S. Fuhrman, and R. Somogyi, "Linear Modeling Of mRNA Expression Levels during CNS Development and Injury," *Proc. Pacific Symp. Biocomputing,* pp. 41-52, 1999.

[15] T.S. Gardner, D.di Bernardo, D. Lorenz, and J.J. Collins, "Inferring Genetic Networks and Identifying Compound Mode of Action via Expression Profiling," *Science,* vol. 301, no. 5629, pp. 102-105, July 2003.

[16] M. Bansal, G.D. Gatta, and D. di Bernardo, "Inference of Gene Regulatory Networks and Compound Mode of Action from Time Course Gene Expression Profiles," *Bioinformatics,* vol. 22, no. 7, pp. 815-822, 2006.

[17] D. Nam, S.H. Yoon, and J.F. Kim, "Ensemble Learning of Genetic Networks from Time-Series Expression Data," *Bioinformatics,* vol. 23, no. 23, pp. 3225-3231, 2007.

[18] S. Kimura, K. Ide, A. Kashihara, M. Kano, M. Hatakeyama, R. Masui, N. Nakagawa, S. Yokoyama, S. Kuramitsu, and A. Konagaya, "Inference of S-System Models of Genetic Networks Using a Cooperative Coevolutionary Algorithm," *Bioinformatics,* vol. 21, no. 7, pp. 1154-1163, 2005.

[19] R. Xu, G.K. Venayagamoorthy, and D.C. Wunsch II, "Modeling of Gene Regulatory Networks with Hybrid Differential Evolution and Particle Swarm Optimization," *Neural Networks,* vol. 20, no. 8, pp. 917-927, 2007.

[20] P.-K. Liu and F.-S. Wang, "Inference of Biochemical Network Models in S-System Using Multiobjective Optimization Approach," *Bioinformatics,* vol. 24, no. 8, pp. 1085-1092, 2008.

[21] C.M. Chen, C. Lee, C.L. Chuang, C.C. Wang, and G. Shieh, "Inferring Genetic Interactions via a Nonlinear Model and an Optimization Algorithm," *BMC Systems Biology,* vol. 4, no. 1, article no. 16, 2010.

[22] R. Tibshirani, "Regression Shrinkage and Selection via the Lasso," *J. Royal Statistical Soc. (Series B),* vol. 58, pp. 267-288, 1996.

[23] H. Zou and T. Hastie, "Regularization and Variable Selection via the Elastic Net," *J. Royal Statistical Soc. Series B,* vol. 67, no. 2, pp. 301-320, 2005.

[24] F. Ferrazzi, P. Sebastiani, M.F. Ramoni, and R. Bellazzi, "Bayesian Approaches to Reverse Engineer Cellular Systems: A Simulation Study on Nonlinear Gaussian Networks," *BMC Bioinformatics,* vol. 8, Suppl. 5, S2 (epub) 2007.

[25] G.-W. Weber, O. Defterli, S.Z. Alparslan Gök, and E. Kropat, "Modeling, Inference and Optimization of Regulatory Networks Based on Time Series Data," *European J. Operational Research,* vol. 211, pp. 1-14, 2011.

[26] R. Albert, "Scale-Free Networks in Cell Biology," *J. Cell Science,* vol. 118, pp. 4947-4957, 2005.

[27] D. Marbach, C. Mattiussi, and D. Floreano, "Replaying the Evolutionary Tape: Biomimetic Reverse Engineering of Gene Networks," *Annals of the New York Academy of Sciences,* vol. 1158, pp. 234-245, 2009.

[28] D. Marbach, C. Mattiussi, and D. Floreano, "Combining Multiple Results of a Reverse Engineering Algorithm: Application to the DREAM Five Gene Network Challenge," *Ann. New York Academy of Sciences,* vol. 1158, pp. 102-113, 2009.

[29] T. Jones and S. Forrest, "Fitness Distance Correlation as a Measure of Problem Difficulty for Genetic Algorithms," *Proc. Sixth Int'l Conf. Genetic Algorithms,* pp. 184-192, 1995.

[30] T. Jones, "Evolutionary Algorithms, Fitness Landscapes and Search," Working Papers 95-05-048, Santa Fe Inst., May 1995.

[31] F. Sambo, M. Montes de Oca, B. Di Camillo, and T. Stützle, "On the Difficulty of Inferring Gene Regulatory Networks: A Study of the Fitness Landscape Generated by Relative Squared Error," *Proc. Ninth Int'l Conf. Artificial Evolution,* P. Collet, N. Monmarché, P. Legrand, M. Schoenauer, and E. Lutton, eds., pp. 74-85, 2010.

[32] M.J.D. Powell, "The NEWUOA Software for Unconstrained Optimization," *Large-Scale Nonlinear Optimization,* ser. Nonconvex Optimization and Its Applications, vol. 83, pp. 255-297, Springer-Verlag, 2006.

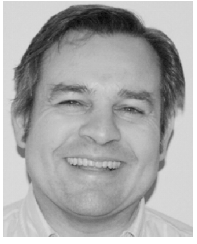
[33] N. Hansen, "The CMA Evolution Strategy: A Comparing Review," *Towards a New Evolutionary Computation, Advances on Estimation of Distribution Algorithms,* pp. 75-102, Springer, 2006.
[34] B. Di Camillo, G. Toffolo, and C. Cobelli, "A Gene Network Simulator to Assess Reverse Engineering Algorithms," *Ann. New York Academy of Sciences,* vol. 1158, no. 1, pp. 125-142, 2009.
[35] B. Di Camillo, M. Falda, G. Toffolo, and C. Cobelli, "SimBioNeT: A Simulator of Biological Network Topology," *IEEE/ACM Trans. Computational Biology and Bioinformatics,* vol. 9, no. 2, pp. 592-600, Mar./Apr. 2012.
[36] J.F. Kolen and S.C. Kremer, *A Field Guide to Dynamical Recurrent Networks.* IEEE Press, 2001.
[37] B.A. Pearlmutter, "Dynamic Recurrent Neural Networks," Technical Report CMU-CS-90-196, Carnegie Mellon Univ., Pittsburgh, PA, 1990.
[38] T.T. Vu and J. Vohradsky, "Nonlinear Differential Equation Model for Quantification of Transcriptional Regulation Applied to Microarray Data of Saccharomyces Cerevisiae," *Nucleic Acids Research,* vol. 35, no. 1, pp. 279-287, Jan. 2007.
[39] J. Kennedy, R. Eberhart, and Y. Shi, *Swarm Intelligence.* Morgan Kaufmann, 2001.
[40] E. Fehlberg, "Low-Order Classical Runge-Kutta Formulas with Step Size Control and Their Application to Some Heat Transfer Problems," Technical Report 315, NASA, 1969.
[41] F. He, R. Balling, and A.-P. Zeng, "Reverse Engineering and Verification of Gene Networks: Principles, Assumptions, and Limitations of Present Methods and Future Perspectives," *J. Biotechnology,* vol. 144, no. 3, pp. 190-203, 2009.
[42] Y. Wang, Y. Ma, and R.J. Carroll, "Variance Estimation in the Analysis of Microarray Data," *J. Royal Statistical Soc.: Series B (Statistical Methodology),* vol. 71, no. 2, pp. 425-445, 2009.
[43] M. Anderle, S. Roy, H. Lin, C. Becker, and K. Joho, "Quantifying Reproducibility for Differential Proteomics: Noise Analysis for Protein Liquid Chromatography-Mass Spectrometry of Human Serum," *Bioinformatics,* vol. 20, no. 18, pp. 3575-3582, 2004.
[44] N. Noman and I. Iba, "Reverse Engineering Genetic Networks Using Evolutionary Computation," *Genome Informatics,* vol. 16, no. 2, pp. 205-214, 2005.
[45] C. Spieth, R. Worzischek, F. Streichert, J. Supper, N. Speer, and A. Zell, "Comparing Evolutionary Algorithms on the Problem of Network Inference," *Proc. Genetic and Evolutionary Computation Conf. (GECCO '06),* M. Cattolico, ed., pp. 305-306, 2006.
[46] A. Auger, N. Hansen, J.M. Perez Zerpa, R. Ros, and M. Schoenauer, "Empirical Comparisons of Several Derivative Free Optimization Algorithms," *Acte du 9ime colloque nat'l en calcul des structures,* May 2009.
[47] A.R. Conn, N.I.M. Gould, and P.L. Toint, *Trust-Region Methods.* ser. MPS-SIAM Series in Optimization. SIAM, 2000.
[48] H.H. Hoos and T. Stützle, *Stochastic Local Search : Foundations & Applications,* The Morgan Kaufmann Series in Artificial Intelligence. Morgan Kaufmann, Sept. 2004.

**Francesco Sambo** received the BS degree in computer science engineering in 2004, the MS degree in computer science engineering in 2006, and the PhD degree in information and communication technologies in 2010 from the University of Padova, Italy. Since 2010, he has been enrolled as a postdoctoral researcher in the Information Engineering Department, University of Padova. His main research interests are optimization and machine learning, applied to the analysis of genetic and metabolic data and to the study of complex systems.

**Marco A. Montes de Oca** received the BS degree in computer systems engineering from Escuela Superior de Cómputo, Instituto Politcnico Nacional, Mexico City, Mexico, in 2001. He received the MS degree in intelligent systems with honors from the Tecnológico de Monterrey, N.L., Mexico, in 2005. He received the PhD degree in engineering sciences from the Université libre de Bruxelles, Brussels, Belgium, in 2011. He is currently a postdoctoral researcher in the Department of Mathematical Sciences, University of Delaware, Newark. His main research interests are swarm intelligence, optimization, and complex systems.

**Barbara Di Camillo** received the doctoral degree in electronics engineering and the PhD degree in biomedical engineering from the University of Padova, in 2000 and 2004, respectively. From March 2004 to February 2006, she was enrolled as a postdoctoral researcher in the Information Engineering Department, University of Padova. Since April 1, 2006, she has been enrolled as a assistant professor of bioengineering at the Information Engineering Department of the University of Padova. Her research activity, carried out in collaboration with Italian and foreign investigators, regards modeling of biological systems, bioinformatics and its application to time series gene and protein expression analysis; in particular, the reconstruction of regulatory networks by reverse engineering and data integration. She has been coinvestigator in several research projects funded by MIUR-MURST and EU and has served as reviewer for several congresses and journals in the bioinformatics and bioengineering fields.

**Gianna Toffolo** received the degree in electronic engineering (with honors) from the University of Padova, Italy, in 1978. She is currently a full professor of biological signal processing at the University of Padova. Since 2009, she has chaired the graduate program in biomedical engineering. Her research activity, carried out in collaboration with Italian and foreign investigators, mainly regards modeling of biological and physiological systems, and includes methodological aspects as well as specific applications in biology, physiology, and medicine, with particular focus on endocrine-metabolic systems. She is an author/coauthor of a book and more than 120 full papers on international peer-reviewed journals.



**Thomas Stützle** received the MS degree (Diplom) in business engineering from the Universität Karlsruhe (TH), Karlsruhe, Germany, in 1994, and the PhD degree and the "Habilitation" in computer science both from the Computer Science Department, Technische Universität Darmstadt, Germany, in 1998 and 2004, respectively. He is currently a research associate of the Belgian F.R.S.-FNRS working in the Institut de Recherches Interdisciplinaires et de Développements en Intelligence Atificielle (IRIDIA), Université libre de Bruxelles, Brussels, Belgium. He is author of the two books: *Stochastic Local Search: Foundations and Applications* (Morgan Kaufmann) and *Ant Colony Optimization* (MIT Press). He has published extensively in the wider area of metaheuristics (more than 150 peer-reviewed articles in journals, conference proceedings, or edited books). His research interests range from stochastic local search (SLS) algorithms, large-scale experimental studies, automated design of algorithms, to SLS algorithms engineering.