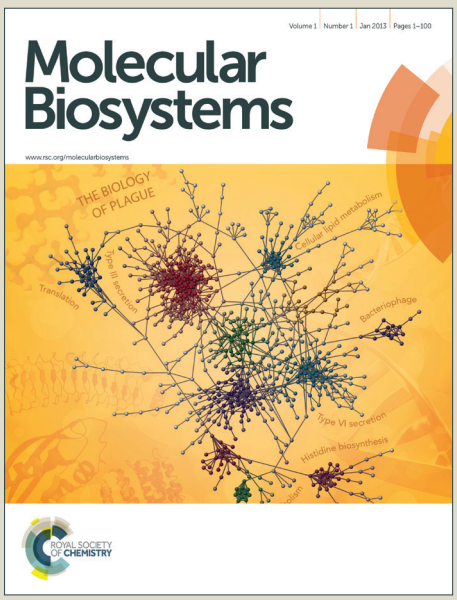
Molecular BioSystems

Accepted Manuscript

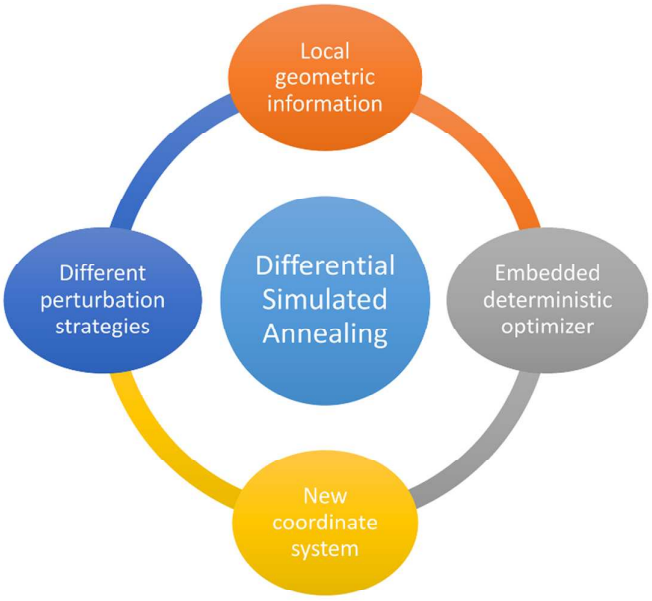


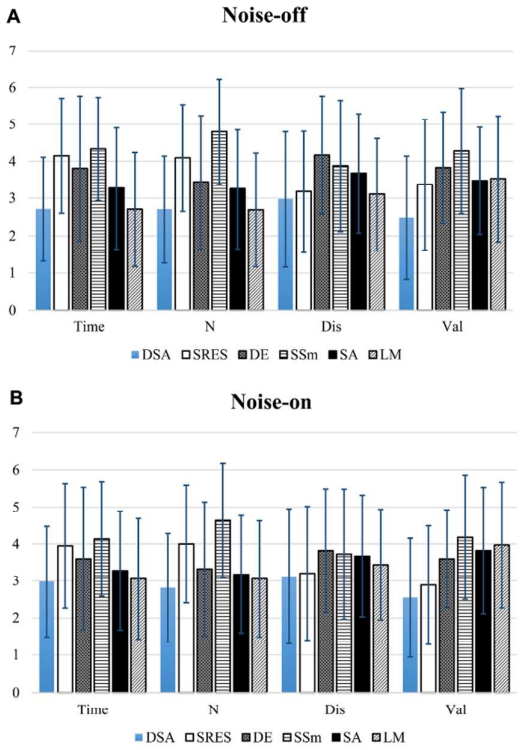
ROYAL SOCIETY OF CHEMISTRY

www.rsc.org/molecularbiosystems

Table of contents entry

DSA outperformed five other algorithms in parameter estimation of 95 biological networks and showed significant advantage in large networks.





Differential Simulated Annealing: A Robust and Efficient Global Optimization Algorithm for Parameter Estimation of Biological Networks†

Ziwei Dai[a] **and Luhua Lai**[*a,b]

Received Xth XXXXXXXXXX 20XX, Accepted Xth XXXXXXXXX 20XX
First published on the web Xth XXXXXXXXXX 200X
DOI: 10.1039/b000000x

Ordinary differential equations (ODEs) are widely used to model dynamic properties of biological networks. Due to the complexity of biological networks and limited quantitative experimental data available, estimating kinetic parameters for these models remains challenging. We present a novel global optimization algorithm, differential simulated annealing (DSA), for estimating kinetic parameters for biological network models robustly and efficiently. DSA was tested on 95 models sizing from a few to several hundreds of parameters from the BioModels database[1] and compared with other five widely used algorithms for parameter estimation, including both deterministic and stochastic optimization algorithms. Our study showed that DSA gave the highest success rate in the whole dataset and performed especially well for large models. Further analysis revealed that DSA outperformed the five algorithms compared in both accuracy and efficiency.

1 Introduction

Mathematical modeling of biological processes such as metabolism, gene regulation and cellular signaling plays an important role in systems biology. There are various ways to delineate the dynamics of a biological system, and the most often used form is ordinary differential equations (ODEs), which can balance the trade-off between efficiency and accuracy for most biological systems. An ODEs description of a biological network can be written in the form below:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}, \mathbf{p}, t) \tag{1}$$

In which $\mathbf{x}$ stands for the current state of the system and $\mathbf{p}$ is a vector of kinetic parameters. Knowledge about values of the parameters is indispensable for obtaining the dynamical behavior of the system and making reliable predictions. Although a fraction of the parameters can be directly measured by experiments, most of them are not known or difficult to be measured. Moreover, the internal sloppy property of parameter space of biochemical models makes it more complicated, since measurement error introduced by experiments can fall on stiff directions, thus bringing more uncertainty to the model predictions[2]. Therefore in most cases parameters are estimated by optimization algorithms minimizing an objective function which quantifies the difference between the model outputs simulated according to values of model parameters and corresponding experimental measurements. This objective function can be defined as the weighted sum of squared errors between measured time-series profiles and output of the parameterized model:

$$F(\mathbf{p}) = \frac{1}{mn}\sum_{i=1}^{m}\sum_{t=1}^{n}[w_{it}(x_{it}(\mathbf{p}) - x_{it}^{\exp})]^2 \tag{2}$$

where $x_{it}^{\exp}$ stands for the value of the ith measurement on time t, $x_{it}(\mathbf{p})$ means the corresponding simulated value and w_{it} means the weight coefficients.

Traditional gradient-based methods usually fail to find well fitting parameters due to the multimodal property of the objective function. Various global optimization algorithms, including genetic algorithm (GA), evolutionary strategy (ES), differential evolution (DE), simulated annealing (SA) and scatter search metaheuristics (SSm) have been applied in parameter estimation of biological models[3,4]. In some applications a local optimizer is embedded into these global optimization algorithms to improve their efficiency and accuracy. SRES[5] was claimed to be the most effective algorithm in a comparative study on an artificial small network published in 2003[6]. However, the task of parameter estimation remains challenging because of the rapidly increasing size of kinetic models and the inner property of parameter estimation problem, which is actually a nonlinear least squares fitting problem. Studies of model manifold of nonlinear least squares problems show that

† Electronic Supplementary Information (ESI) available. See DOI: 10.1039/b000000x/

[a] *Center for Quantitative Biology, Peking University, Beijing 100871, China*
[b] *BNLMS, State Key Laboratory for Structural Chemistry of Unstable and Stable Species, and Peking-Tsinghua Center for Life Sciences at College of Chemistry and Molecular Engineering, Peking University, Beijing 100871, China. Tel: 010-62757486; E-mail: lhlai@pku.edu.cn*

the vector of best fitting parameters often lies in a long and narrow canyon [2,7], making it easy for optimization algorithms to get lost in badly fitting area. Though many algorithms were developed to solve this problem [8–13], most of these algorithms were tested on small networks with a few to less than 30 parameters, thus presenting little evidence about their ability to deal with larger models. On parameter estimation of larger networks, Nim *et al.* presented SPEDRE [14], an algorithm based on derivatives evaluation by spline fitting and belief propagation. This method showed good performance in large low-degree networks with enough experimental data, but it still cannot handle many real-world networks because in many pathways not all species are measurable and high-degree hubs are very common in biological networks [15,16].

Simulated annealing (SA) [17] is a global optimization algorithm first designed for solving combinatorial optimization problems (*e.g.*, the travelling salesman problem). SA is first developed mainly for solving combinatorial optimization problems on discrete sets. It is also useful in optimizing continuous multi-modal functions by applying self-adapting step size control strategies [18]. Improvements of SA in solving continuous problems involves techniques called basin hopping and eigenvector following, in which the eigenvector of the Hessian with the smallest eigenvalue is chosen to form a new configuration along it. By iteration of this process and combination with a deterministic minimizer, the annealing schedule can switch in the set of local minima, thus accelerating convergence [19].

In the present study, we developed a novel parameter estimation algorithm based on SA named differential simulated annealing (DSA). The major innovation of DSA is using all eigenvectors of the Hessian of the objective function to construct move classes adopting two kinds of strategies for perturbation to generate new configurations. This methodology combined with a deterministic minimizer can obtain high robustness and efficiency in parameter estimation of biological networks of various sizes. We also compared DSA with other widely-used optimization algorithms for parameter estimation on some real-world biochemical computational models from the BioModels database [1]. Although our discussion and comparison are under a framework of parameter estimation of ODEs-based models, it is obvious that DSA can also be extended to other nonlinear least squares fitting problems with box constraints.

2 Methods

The workflow of DSA is shown in Figure 1. By evaluating the approximate Hessian of the objective function periodically, DSA constructs a new coordinate system to form the move class by which new configurations are generated from the old ones. In generating new configurations, two alternative strategies of perturbation may be adopted according to the directional derivative on the direction of perturbation. Each step in the workflow will be described in detail in the following parts.

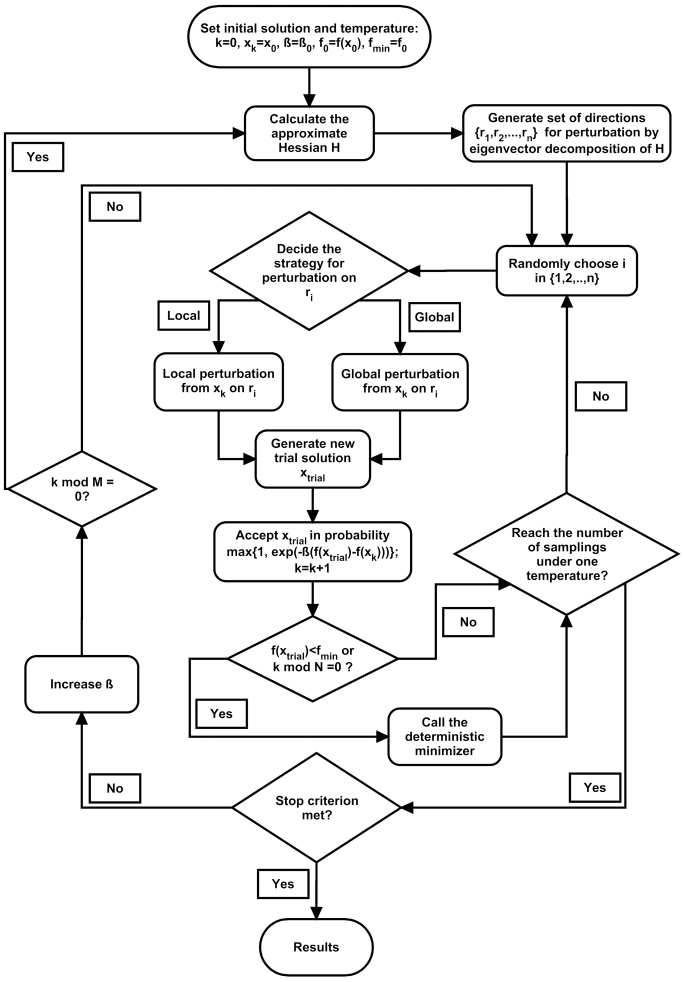


Fig. 1 Workflow of DSA. *N* is the number of objective function value evaluations after which a deterministic minimizer will be called to find a local minimum. *M* is the number of objective function value evaluations after which the coordinate system for perturbation will be reconstructed.

2.1 Calculate the approximate Hessian

DSA uses local geometric information to guide the Metropolis sampler by adjusting both the coordinate system it moves along and the mode of step in each movement. Now we consider the general nonlinear least squares fitting problem because most parameter estimation problems can be written in this form. The constraints on values of parameters are their

lower and upper bounds.

$$\text{argmin}_{\mathbf{p}}\{F(\mathbf{p}) = \sum_{i=1}^{m} r_i^2(\mathbf{p})\} \quad (3)$$

$$\text{s.t.}\ \mathbf{p}_{lb} < \mathbf{p} < \mathbf{p}_{ub} \quad (4)$$

Considering the order-2 Taylor expansion of $F(\mathbf{p})$:

$$F(\mathbf{p}_0+\mathbf{dp}) = F(\mathbf{p}_0)+\mathbf{dp}^T\nabla F(\mathbf{p}_0)+\frac{1}{2}\mathbf{dp}^T\mathbf{H}(\mathbf{p}_0)\mathbf{dp}+O(\|\,\mathbf{dp}\,\|^2) \quad (5)$$

In which $\mathbf{p}_0$ denotes the best fit parameters. Note that $\nabla F(\mathbf{p}_0)$ is a zero vector since $\mathbf{p}_0$ is a local minima, the expansion can be rewritten as

$$F(\mathbf{p}_0+\mathbf{dp}) = F(\mathbf{p}_0)+\frac{1}{2}\mathbf{dp}^T\mathbf{H}(\mathbf{p}_0)\mathbf{dp}+O(\|\,\mathbf{dp}\,\|^2) \quad (6)$$

So we can evaluate the increment of objective function on direction **dp**:

$$F(\mathbf{p}_0+\mathbf{dp})-F(\mathbf{p}_0) \approx \frac{1}{2}\mathbf{dp}^T\mathbf{H}(\mathbf{p}_0)\mathbf{dp} \quad (7)$$

This approximate increment is a quadratic function of **dp**. The shape of contour line of such a function is an ellipsoid centered in $\mathbf{p}_0$ and long axis on the eigenvector of $\mathbf{H}(\mathbf{p}_0)$ corresponding to its smallest eigenvalue. This ellipsoid-like geometry is illustrated in Figure 2A.

Calculating $\mathbf{H}(\mathbf{p}_0)$ is a time-consuming task when the number of parameters to be evaluated, N, is large. The time complexity of this problem is $O(N^2)$. In order to save time spent on the Hessian evaluation, we use the Fisher information matrix (FIM) as an approximation of the Hessian. Consider the sum of squares form of $F(\mathbf{p})$ we have:

$$\frac{\partial^2 F(\mathbf{p})}{\partial p_i \partial p_j} = 2\sum_{k=1}^{m}\left(\frac{\partial r_k(\mathbf{p})}{\partial p_i}\frac{\partial r_k(\mathbf{p})}{\partial p_j}+r_k(\mathbf{p})\frac{\partial^2 r_k(\mathbf{p})}{\partial p_i \partial p_j}\right) \quad (8)$$

In a small residue ($r_k(\mathbf{p}) \approx 0$) condition, the second term in the equation above can be ignored. So the Hessian can be approximately represented by two folds of the product of Jacobian matrix's transposition and itself:

$$\mathbf{H} \approx 2\left(\frac{\partial \mathbf{r}}{\partial \mathbf{p}}\right)^T\left(\frac{\partial \mathbf{r}}{\partial \mathbf{p}}\right) \quad (9)$$

Using this approximation the time cost will be reduced to $O(N)$. The approximate Hessian will be calculated after every M steps of Metropolis sampling.

2.2 Generate directions for perturbation

By eigenvector decomposition of the approximate Hessian, a set of directions representing the axes of the ellipsoid-like contour can be generated. In DSA, these eigenvectors are adopted as new coordinate system instead of the Cartesian coordinate system to form the move class of simulated annealing. Figure 2A shows the eigenvectors of the Hessian of a 2-dimension objective function. The advantage of taking such a coordinate system is that it separates directions with tiny and huge directional derivatives to the largest extent. With different strategies suitable for movements in directions with large or small directional derivatives, we expected that this choice of coordinate system would lead to more significant improvement of performance of simulated annealing than any other coordinate systems with relatively narrowly distributed directional derivatives on the axes. The strategies of perturbation on both kinds of directions will be described in detail in the next subsection.

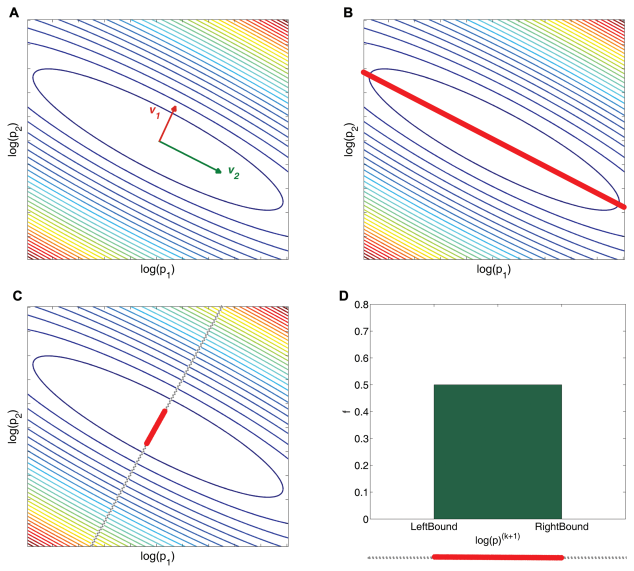


Fig. 2 A two-dimensional illustration of local quadratic approximation and different perturbation strategies. (**A**) Contour of the quadratic approximation of objective function. v_1 and v_2 are two eigenvectors of the Hessian. (**B**) Global perturbation on direction v_2. The bold red line stands for the range where the new trial solution can lie. (**C**) Local perturbation on direction v_1. (**D**) The distribution to sample the new trial solution. It is a uniform distribution on [LeftBound, RightBound] according to the perturbation strategy adopted.

2.3 Perturbation strategies

Two different perturbation strategies are adopted in DSA to accelerate its convergence and improve the efficiency of sampling. Which strategy to choose in a certain step relies on the current direction r_i of perturbation: global strategy for sloppy directions and local strategy for stiff directions. 'Sloppy' or

'stiff' describes eigenvectors with smaller or bigger directional derivative, respectively. When a global strategy is chosen, the new solution will be sampled from uniform distribution on the intersection of the line representing current direction of perturbation and the box restraining the values of parameters, or it will be sampled from a small neighbor $[\mathbf{x}_i - \varepsilon\mathbf{r}_i, \mathbf{x}_i + \varepsilon\mathbf{r}_i]$ of the old solution on the direction. Figure 2B and 2C shows the two perturbation strategies. DSA uses a threshold λ_0 to decide between the two perturbation strategies: if the eigenvalue is larger than λ_0 the relevant direction of perturbation is considered to be stiff and a local perturbation is carried out. Otherwise, if the eigenvalue of current direction is smaller than the threshold, a global perturbation is adopted.

Choice of λ_0 affects the proportion of global perturbations in the optimization process. There will be no global perturbations if the value of λ_0 is negative. We also want the choice between local and global perturbation strategies to be adaptive to the cooling of temperature. In 'hot' systems where it is much easier to accept new trial solutions, we want a bigger global/local ratio to enhance the movement of solution in the parameter space. Because both the temperatures during annealing and the eigenvalues of FIM distribute over a wide range crossing many magnitudes, we can directly set a connection between them. In hotter temperature, namely lower β, we use a higher threshold to enable more global perturbations. The choice of λ_0 is coupled with the cooling of the system temperature β using a reciprocal relationship:

$$\lambda_0 = \frac{c}{\beta} \tag{10}$$

2.4 Set initial conditions

Similar to original SA, DSA needs careful choice of parameters involving initial temperature, number of samplings under one temperature, and the annealing schedule. It has been proved that the annealing schedule should be slower than logarithmic to guarantee global convergence[20], but such an inefficient annealing schedule is not practical at all since time cost of this process will be longer than that of complete enumerating all available solutions. Decision of these parameters is actually another optimization algorithm. Here we introduce some empirical rules to help choose them. By scaling with the dimension of the problem and sampling the whole region of feasible solutions, DSA tries to set appropriate values for some of them. A Latin hypercube sampler[21] is called first, generating a set containing m solutions uniformly distributed in the feasible region. Values of objective function for these diverse solutions are then calculated and ranked. The maximum value among them is used to set the initial temperature according to the equation below:

$$\beta_0 = \frac{-\ln 0.99}{f_{\max}^{\text{sam}}} \tag{11}$$

Initial temperature chosen in this way can ensure that more than 99% of newly generated trial solutions will be accepted under it, given that the real maximum value of objective function in the feasible region, $f_{\max}$, is not higher than $f_{\max}^{\text{sam}}$. Generally speaking, $f_{\max}^{\text{sam}}$ will be closer to $f_{\max}$ if we choose larger sample size m, but it may also result in longer time of computation due to smaller β_0. In logarithmic annealing schedule the performance of DSA will not depend on the choice of initial solution, but we still need to be cautious in decision of it since the annealing schedule adopted in DSA is much faster than logarithmic (exponential as default) thus cannot guarantee global convergence. In order to avoid beginning from a bad solution trapped in the neighborhood of a local minimum, DSA ranks the solutions sampled by Latin hypercube sampling by their corresponding objection function values, then picks the solution with the lowest 'fitness' (*i.e.* the largest objective function value). More local minima will be found by DSA using such 'non-greedy' choice of initial guess. Some default values of parameters for DSA are shown in Table 1.

Table 1 Default values for control parameters of DSA

c	N_{MC}	N_T	s_0	N	M
1.03	$50\sqrt{dim}$	$50\sqrt{dim}$	0.05	$500\sqrt{dim}$	$100dim$

Meaning of the parameters: c is the cooling factor of annealing: $\beta_{n+1} = c\beta_n$. N_{MC} means number of Monte Carlo samplings under one temperature. N_T is number of trial solutions sampled by Latin hypercube sampling before annealing. s_0 means the initial step size of local perturbation. N is the number of objective function value evaluations after which a deterministic minimizer will be called to find a local minimum. M is the number of objection function value evaluations after which the coordinate system for perturbation will be reconstructed.

2.5 Local search

The local solver embedded in DSA will be called in two conditions. First it will be called periodically during the annealing process, after every N steps of Metropolis sampling. The solution returned by the local solver called in this condition will only be accepted if its objective function value is smaller than any solution found before. In order to find as many local minima as possible, the local solver will also be called if the current value of objective function is the smallest value ever found. We chose Levenberg-Marquardt algorithm[22] in our work because of its wide application in solving nonlinear least squares problems, but other choices will also be suitable.

2.6 Stop criterion

DSA will terminate if at least one of the conditions below is satisfied:

- No new trial solution is accepted under one temperature;
- The value of objective function reaches or is better than expected global optimum;
- The maximum number of objective function evaluation is exceeded;
- The maximum time of computation is exceeded.

3 Results

3.1 Case studies

We chose 95 models from the BioModels database[1] to test the performance of DSA. These models scale from very small networks with less than 10 nodes to large ones with up to 76 nodes and 234 parameters. Detailed information about these models are listed in supplementary information†. Five other algorithms are also taken into comparison: simulated annealing (SA)[17,18], differential evolution (DE)[23], stochastic ranking evolutionary strategy (SRES)[5,24], scatter search metaheuristics (SSm)[25] and multi-start Levenberg-Marquardt algorithm (LM)[22]. These algorithms have all shown promise in parameter estimation of biological networks. Among these algorithms SA, DE, SRES, SSm are stochastic algorithms and LM is deterministic. Parameters for these algorithms are shown in Table 2.

It is worth notice that combination with local minimizers can improve efficiency of those global algorithms[24,25]. The four global algorithms compared are all combined with Levenberg-Marquardt algorithm to form hybrid algorithms. The way to combine the global and local optimizer is the same as that in DSA, in which the local algorithm is called either after N iterations in the global algorithm or when a new optimum is found.

First we created artificial experimental measurements by simulating the models using their published values of kinetic parameters attached in the model files. Two conditions were considered: one is the noise-off condition in which simulated concentrations for all the species in the model were directly taken as experimental data, the other is the noise-on condition in which 5% Gaussian noise was added to each simulated data point. The time of simulation was carefully chosen according to the original publications to ensure that simulation during this time can afford enough information about dynamics of the systems. For each species, 50 points uniformly distributed in the time range of simulation were taken as time point of experimental measurements. The artificial time-course concentration profile of one of the models (BioModels ID BIOMD0000000228) in both noise-off and noise-on condition is shown in Fig 3. We used weighted sum of squared errors averaged over all data points as the objective function:

$$F(\mathbf{p}) = \frac{1}{mn}\sum_{i=1}^{m}\sum_{t=1}^{n}\left[\frac{(x_{it}(\mathbf{p}) - x_{it}^{\exp})}{\frac{1}{n}\sum_{t=1}^{n} x_{it}^{\exp}}\right]^2 \quad (12)$$

Kinetic parameters in these models can be grossly classified into three categories: coefficients, orders and qualitative parameters. Coefficients are those parameters which are added or multiplied to variables or other parameters in the equations. These parameters must have non-zero 'true' value in order to be significant in the model. For these parameters the range is a randomly generated region containing its published value covering 4 magnitudes. The purpose of randomly generating lower and upper bounds for these parameters is to avoid the tendency of all 'real' solution to be in the center of the plau-

Table 2 Control parameters for SA, DE, SRES, SSm and LM in the comparative study

Algorithm	Control parameter	Value
Universal	$t_{\max}$	200h
	$f_{\text{off}}^{\text{opt}}$	$< 10^{-4}$
	$f_{\text{on}}^{\text{opt}}$	$< 1.01 f_{\text{on}}^{\text{real}}$
SA	Same as DSA	Same as DSA
DE	x	best
	y	2
	z	bin
	F	0.5
	CR	1
	NP	$10 \times dim$
SRES	λ	$10 \times dim$
	μ	$\lambda/7$
SSm	N_{div}	$10 \times dim$
	N_{ref}	dim
	p	4
LM	N_{SP}	$50 \times dim$

Meaning of the control parameters for the algorithms: Universal (parameters shared by all algorithms): $t_{\max}$ is the maximum computational time, $f_{\text{off}}^{\text{opt}}$ and $f_{\text{on}}^{\text{opt}}$ is the expected optimum value of the objective function under noise-off and noise-on conditions, respectively. $f_{\text{on}}^{\text{real}}$ is the value of objective function yielded by the published value of the parameters under noise-on condition. For definition of the two conditions and the objective function, see the following paragraphs. DE: x means the way to choose a vector to be mutated, y is the number of difference vectors used, z denotes the crossover scheme, F is the amplification factor, CR is the crossover constant and NP is the number of solution vectors in the population. dim is the dimension of the problem, which is the number of kinetic parameters in the case studies. SRES: λ is the number of off-springs created in one generation and μ is the number of individuals joining production of the next generation. SSm: N_{div} is the number of diverse vectors and N_{ref} is the size of refset. LM: N_{SP} is the number of starting points in the multi-start algorithm. The starting points were sampled by Latin hypercube sampling method.

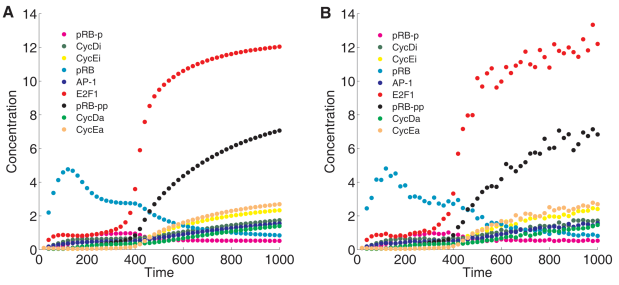


Fig. 3 Artificial time-course concentration profile of one model for performance test of DSA and other algorithms. (**A**) The noise-off condition. (**B**) The noise-on condition in which 5% Gaussian noise was added to the data points generated by simulation.

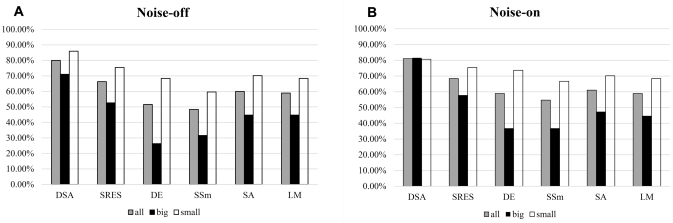


Fig. 4 Success rates of DSA, SRES, SSm, DE, SA and LM. The grey bars stand for success rates among all networks and the black and white bars for success rates in big (not less than 30 parameters) and small (less than 30 parameters) networks, respectively. (**A**) Noise-off condition. (**B**) Noise-on condition.

sible region. Orders are parameters appearing in the exponent (*e.g.*, the Hill coefficients in gene regulatory models). For orders the lower bounds were 0 and the upper bounds were 10. Qualitative parameters are parameters which are not orders and have zero published values. These parameters are often used in order to exclude some flux or interaction from the original models, thus forming alternative models. Range of these parameters was set to [-2, 2] since we do not know the sign of the excluded interactions.

3.2 Result of comparison

First of all we compared the overall success rates of the six algorithms over all models. In the noise-off condition, an algorithm is considered to be successful if the weighted sum of squared error averaged over all data points (*i.e.* the objective function) is less than 10^{-4}, which means that the difference between observance and simulation is less than 1%. In the noise-on case, successful parameter estimation is defined as those who find values for the parameters which yield objective function value less than 101% of that yielded by the 'real' values of parameters. Maximum number of objective function evaluations for the six algorithms were all set to $10^5 \times dim$, where *dim* means the number of parameters to be evaluated.

Success rates of DSA and the other five algorithms in both noise-on and noise-off conditions are shown in Figure 4 (the grey bar). These results show that DSA has higher success rate than all other five algorithms in both noise-off and noise-on conditions, which indicates that DSA is a very robust algorithm for parameter estimation.

3.3 Model size effect

In order to test the effect of model size on the performance of DSA and the other five algorithms, we divided the 95 models into two classes according to their size (number of parameters to be evaluated). The threshold to distinguish small and large network models was set to 30, which gives 57 small models and 38 large models. The success rates of the six algorithms when applied to small models and large models are shown in Figure 4. The success rate of DSA is the highest among the five algorithms, which shows similar performance for small and large models. All the other five algorithms give significantly lower success rate for the large models either with or without noises, indicating that these algorithms cannot effectively deal with large models with up to hundreds of parameters. One possible reason for the performance deterioration in large models is that they do not take into consideration any information about the size and dimensionality of parameter space, which makes the movement of trial solutions in parameter space even more inefficient in high dimensional space. But for DSA, because of the skewed distribution of eigenvectors of the Hessian, its efficiency of movement during global perturbation will increase with dimensionality, thus breaking the curse of dimensionality to some extent.

3.4 Mean rank score for algorithm performance

In addition to overall success rates, more indices are also necessary to judge performance of DSA and the other five algorithm compared. We chose four indices to comprehensively evaluate performance of the six algorithms: computation time (*time*), number of objective function evaluations (*N*), Euclidean distance (*dis*) between the best solution found by the algorithm and the real solution, and optimum value of objective function (*val*). For each model we ranked the indices of all six algorithms in increasing order with algorithms succeeded always prior to those failed, then calculated the mean rank of each algorithm over the 95 models. Since smaller value of the indexes indicates less computational cost or smaller deviation, algorithms with smaller mean rank score will be a better choice considering its performance over the 95 models. The mean rank scores for DSA and the other five algorithms are displayed in Figure 5. DSA exhibits smaller mean rank

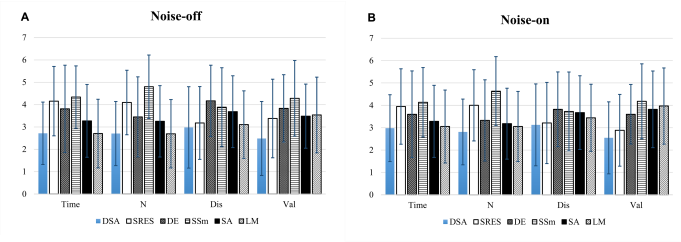


Fig. 5 Mean rank scores for DSA, SRES, DE, SSm, SA and LM. Lengths of the error bars denote the standard deviation. (**A**) Noise-off condition. (**B**) Noise-on condition.

scores than other five algorithms in all four indices, which reveals its advantages over them in both efficiency and accuracy. Complete results for the six algorithms under both noise-off and noise-on conditions are listed in the supplementary information†.

3.5 Failures

DSA failed in finding proper parameters in 19 models under noise-off condition and 18 models under noise-on condition. Parameter estimation of these models is very difficult no matter which algorithm is used: other algorithms compared also failed in treating these models, except for BIOMD0000000105 under noise-on condition, in which SRES, DE and SSm successfully achieved well fitting parameters. This implies that these models may suffer from some intrinsic difficulty in parameter estimation.

In order to find what factors may affect the difficulty in parameter estimation, we first manually checked the dynamical properties of all the 95 models and classified them into two categories according to whether concentrations of species in a model oscillate. There are 26 oscillators in the 95 models and DSA failed in 10 of them under both noise-off condition and noise-on condition. Failures in parameter estimation of oscillating systems constitutes the majority of all failures under both condition (11 out of 19 under noise-off condition and 12 out of 18 under noise-on condition). It seems that oscillation, as appearance of limit cycles in a dynamic system, introduces more strict constraints on values of parameters, hence hinders most optimization algorithms from successful parameter estimation. However, DSA still outperformed the other five algorithms in parameter estimation of these oscillating systems (Figure S5, supplementary information†) with significantly higher success rates under both noise-off and noise-on conditions.

Furthermore, we found that the difficulty of parameter estimation for some models can be elucidated by local geometry near the global minimum, which can be quantitatively measured by using quadratic approximation. Under such approximation the region containing all well fitting parameter vectors around the global minimum is a hyper-ellipsoid with semi-principal axes of length $\sqrt{\frac{f_m}{\lambda_i}}$ located on eigenvectors of the Hessian, where f_m is the largest tolerable objective function value for well fitting parameter vectors and λ_i is the corresponding eigenvalue. Volume of this hyper-ellipsoid can be calculated as below:

$$V_n = \frac{2\pi^{n/2}}{n\Gamma(n/2)} \prod_{i=1}^{n} \sqrt{\frac{f_m}{\lambda_i}} \tag{13}$$

In which $\Gamma(n/2)$ is the Gamma function:

$$\Gamma(s) = \int_0^{+\infty} x^{s-1} e^{-x} dx \tag{14}$$

We calculated the proportion of this well fitting ellipsoid in the box containing all available parameter vectors for the 95 models under noise-off condition: $r_n = V_n / D^n$, where D is the length of edges of the available box. Intuitively, it will be easier for parameter estimation algorithms to find the global minimum if the well-fitting ellipsoid occupies larger fraction of the available box. This is the fact: the values of r_n in models that DSA failed are significantly smaller than in those models that DSA succeeded (p=0.0028). For more information, see Figure S1 in supplementary information†. This reveals that the degree of difficulty in parameter estimation largely depends on the local geometry of objective function. The analysis also provides a new way of optimal experimental design, that is, maximizing the proportion of well fitting ellipsoid while maintaining parameter identifiability, thus facilitating the searching of global minimum.

4 Implementation details

Programs implementing SA, DE, SSm, SRES, LM and DSA were all written in C++. Models downloaded from the BioModels database were translated to C++ codes using SBML translator in Systems Biology Workbench (SBW)[26]. Some data structures and functions from the GNU Scientific Library (GSL)[27], including the ODEs solver gsl_odeiv2_msbdf, the eigenvalues and eigenvectors calculator for real symmetric matrixes, the random number generator, were used in the codes. Jacobians of the ODEs were symbolically calculated by scripts based on SymPy, a Python library for symbolic mathematics[28]. The SRES program was written based on a published C library of this algorithm named libSRES[29]. The local optimizers in all programs using Levenberg-Marquardt algorithm were implemented making use of MPFIT[30], a C library for both constrained and unconstrained nonlinear least squares fitting.

We also developed two packages, libDSA and DSA_Matlab, to help users with different demands solve their own problems with DSA. libDSA is a static library written in C++.

It allows users to tune control parameters of DSA by themselves to maximize efficiency and adapt to different problems. On the other hand, DSA_Matlab is a MATLAB package which is very easy to use but sacrifices some controllability by users. These two packages can be downloaded from: `http://mdl.ipc.pku.edu.cn/mdlweb/register.php?id=15`. For more details about how to use these two packages, see the README documents attached to them.

5 Conclusions

This work presents DSA as a new global optimization algorithm for solving nonlinear least squares problems and demonstrates its higher effectiveness in parameter estimation of biochemical models than SA, DE, SRES, SSm and multi-start LM via a comparative study on 95 ODEs models covering networks of different sizes. By using local geometric information to guide movements of trial solution in parameter space, DSA can find parameters satisfyingly fitting experimental observation for most models. DSA shows high robustness and good efficiency in parameter estimation, especially in treating large models. Furthermore, DSA is easy to programme and parallelize due to its similarity to the original simulated annealing algorithm.

As we are gaining more and more knowledge about the molecular mechanisms of biological processes and the availability of high throughput experimental data, the necessity to build large, multiscale models to quantitatively describe dynamics of living systems is becoming even more significant. There already exists some primary attempts in whole-cell modeling of model organisms[31], but effective ways to parameterize these huge models are still lacking, partly because of the high dimensionality of parameter space. Although some parameters can be obtained from existing researches, databases, direct measurements or manual adjustment, this will be a very time-consuming task and can introduce additional bias. As the present study shows that DSA performs well in large models, we hypothesize that it can be widely used in comprehensive and detailed dynamical modeling of biological systems in the future.

6 Acknowledgement

This work was supported in part by the Ministry of Science and Technology of China (2012AA020308) and the National Natural Science Foundation of China (91313302, 90913021). The authors thank Dr. Ning Yin for help with the Jacobian generating script and all the helpful discussions and Bin Shao for help in testing and improving the two packages for DSA.

References

1 C. Li, M. Donizelli, N. Rodriguez, H. Dharuri, L. Endler, V. Chelliah, L. Li, E. He, A. Henry, M. I. Stefan, J. L. Snoep, M. Hucka, N. Le Novère and C. Laibe, *BMC Syst. Biol.*, 2010, **4**, 92.
2 R. N. Gutenkunst, J. J. Waterfall, F. P. Casey, K. S. Brown, C. R. Myers and J. P. Sethna, *PLoS Comput. Biol.*, 2007, **3**, 1871–1878.
3 I.-C. Chou and E. O. Voit, *Math. Biosci.*, 2009, **219**, 57–83.
4 J. Sun, J. M. Garibaldi and C. Hodgman, *IEEE/ACM Trans. Comput. Biol. Bioinforma.*, 2011, **9**, 185–202.
5 T. Runarsson and X. Yao, *IEEE Trans. Evol. Comput.*, 2000, **4**, 284–294.
6 C. G. Moles, P. Mendes and J. R. Banga, *Genome Res.*, 2003, **13**, 2467–2474.
7 M. K. Transtrum, B. B. Machta and J. P. Sethna, *Phys. Rev. Lett.*, 2010, **104**, 060201.
8 J. S. Almeida and E. O. Voit, *Genome Informatics*, 2003, **14**, 114–123.
9 O. R. Gonzalez, C. Küper, K. Jung, P. C. Naval and E. Mendoza, *Bioinformatics*, 2007, **23**, 480–486.
10 E. Balsa-Canto, M. Peifer, J. R. Banga, J. Timmer and C. Fleck, *BMC Syst. Biol.*, 2008, **2**, 26.
11 G. Jia, G. N. Stephanopoulos and R. Gunawan, *Bioinformatics*, 2011, **27**, 1964–1970.
12 G. Jia, G. Stephanopoulos and R. Gunawan, *BMC Syst. Biol.*, 2012, **6**, 142.
13 A. Abdullah, S. Deris, M. S. Mohamad and S. Anwar, *PLoS One*, 2013, **8**, e61258.
14 T. H. Nim, L. Luo, M.-V. Clement, J. K. White and L. Tucker-Kellogg, *Bioinformatics*, 2013, **29**, 1044–1051.
15 A.-L. Barabási and R. Albert, *Science*, 1999, **286**, 509–512.
16 H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai and A.-L. Barabási, *Nature*, 2000, **407**, 651–654.
17 S. Kirkpatrick, C. Gelatt and M. Vecchi, *Science*, 1983, **220**, 671–680.
18 A. Corana, M. Marchesi, C. Martini and S. Ridella, *ACM Trans. Math. Softw.*, 1987, **13**, 262–280.
19 P. Salamon, P. Sibani and R. Frost, *Facts, Conjectures, and Improvements for Simulated Annealing*, Society for Industrial and Applied Mathematics, New York, 2002.
20 S. Geman and D. Geman, *IEEE Trans. Pattern Anal. Mach. Intell.*, 1984, **6**, 721–41.
21 M. Mckay and R. Beckman, *Technometrics*, 1979, **21**, 239–245.
22 D. W. Marquardt, *J. Soc. Ind. Appl. Math.*, 1963, **11**, 431–441.
23 R. Storn and K. Price, *J. Glob. Optim.*, 1997, **11**, 341–359.
24 M. Rodriguez-Fernandez, P. Mendes and J. R. Banga, *Biosystems.*, 2006, **83**, 248–265.
25 M. Rodriguez-Fernandez, J. A. Egea and J. R. Banga, *BMC Bioinformatics*, 2006, **18**, 1–18.
26 F. T. Bergmann and H. M. Sauro, *Proc. 2006 Winter Simul. Conf.*, 2006, 1637–1645.
27 M. Galassi, J. Davies, B. Gough, G. Jungman and P. Alken, *GNU Scientific Library Reference Manual*, 2011.
28 SymPy Development Team, *SymPy: Python library for symbolic mathematics*, 2013.
29 X. Ji and Y. Xu, *Bioinformatics*, 2006, **22**, 124–126.
30 C. B. Markwardt, *Astron. Data Anal. Softw. Syst. XVII*, 2008, **411**, 251–254.
31 J. R. Karr, J. C. Sanghvi, D. N. Macklin, M. V. Gutschow, J. M. Jacobs, B. Bolival, N. Assad-Garcia, J. I. Glass and M. W. Covert, *Cell*, 2012, **150**, 389–401.