Engineering Applications of Artificial Intelligence 62 (2017) 164–180

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

An enhanced scatter search with combined opposition-based learning for parameter estimation in large-scale kinetic models of biochemical systems

Muhammad Akmal Remli[a], Safaai Deris[b], Mohd Saberi Mohamad[a,*], Sigeru Omatu[c], Juan Manuel Corchado[d]

[a] Artificial Intelligence and Bioinformatics Research Group, Faculty of Computing, Universiti Teknologi Malaysia, 81310 Skudai, Johor, Malaysia
[b] Faculty of Creative Technology & Heritage, Universiti Malaysia Kelantan, Locked Bag 01, 16300 Bachok, Kota Bharu, Kelantan, Malaysia
[c] Department of Electronics, Information and Communication Engineering, Osaka Institute of Technology, Osaka 535-8585, Japan
[d] University of Salamanca, Biomedical Research Institute of Salamanca/BISITE Research Group, Salamanca, Spain

ARTICLE INFO

Keywords:
Bioinformatics
Artificial intelligence
Metabolic engineering
Evolutionary algorithm
Scatter search
Opposition-based learning

ABSTRACT

An enhanced scatter search (eSS) with combined opposition-based learning algorithm is proposed to solve large-scale parameter estimation in kinetic models of biochemical systems. The proposed algorithm is an extension of eSS with three important improvements in terms of: reference set (*RefSet*) formation, *RefSet* combination, and *RefSet* intensification. Due to the difficulty in estimating kinetic parameter values in the presence of noise and large number of parameters (high-dimension), the aforementioned eSS mechanisms have been improved using combination of quasi-opposition and quasi-reflection, which were under the family of opposition-based learning scheme. The proposed algorithm is tested using one set of benchmark function each from large-scale global optimization (LSGO) problem as well as parameter estimation problem. The LSGO problem consists of 11 functions with 1000 dimensions. For parameter estimation, around 116 kinetic parameters in Chinese hamster ovary (CHO) cells and central carbon metabolism of *E. coli* are estimated. The results revealed that the proposed algorithm is superior to eSS and other competitive algorithms in terms of its efficiency in minimizing objective function value and having faster convergence rate. The proposed algorithm also required lower computational resources, especially number of function evaluations performed and computation time. In addition, the estimated kinetic parameter values obtained from the proposed algorithm produced the best fit to a set of experimental data.

1. Introduction

Metabolic engineering is an important technique in analyzing metobolic pathway of microorganism to support the production and improvement of cellular properties (Keasling, 2012; Mendes and Kell, 1998). This technique which is commonly used in bioprocess engineering or/and genetic engineering is conducted through modeling, experimental and computational procedures (Cvijovic et al., 2011). The outcomes of metabolic engineering is sustainable bioproduct, specifically for industrial biotechnology application (Almquist et al., 2014). Many bioproducts are produced through capitalizing living cells as cell factories. Microorganisms are reported to be efficient cell factories that are able to convert sugar into chemical of interest (Liu et al., 2013a, 2013b). This method can be achieved either by using natural or genetically modified microorganisms. Genetically modified cells are proven to improve cells production, substrate utilization, product quality as well as process design (Almquist et al., 2014). Cell factories have multiple uses ranging from producing bacteria and yeast to developing therapeutic protein in mammalian cells. Nonlinear mathematical models are important tools in the development of this application as they represent the dynamic and mechanistic nature of the cellular processes. The models are used for understanding and analyzing, for example, the concentration changes in fermentation processes before they can be used for predicting and improving production in order to meet industrial demands (Smallbone et al., 2013). Mathematical models are important in metabolic engineering because they are used for the development of various bioproducts such as biofuels and other chemicals (Almquist et al., 2014). Among all mathematical models, kinetic model is considered to be the most efficient tool for in silico metabolic engineering (Cvijovic et al., 2011). This model has attracted a lot of attention from the research community and industrial biotechnology players. The model has several advantages over other models, namely it can describe a complex biological behavior and it can be used for rational design in cell factory.

* Corresponding author.
E-mail addresses: saberi@utm.my, mohd.saberi@gmail.com (M.S. Mohamad).

http://dx.doi.org/10.1016/j.engappai.2017.04.004
Received 2 November 2016; Received in revised form 13 February 2017; Accepted 4 April 2017

Building an efficient kinetic model that is beneficial in metabolic engineering is considered as an iterative task (Almquist et al., 2014) which involves these processes:

1) Determine the purpose of the model;
2) Design the model structure;
3) Estimate the parameter; and
4) Validate.

First, determining the model's purpose is a key step towards building a kinetic model. The modeler should identify various organisms and types of biological processes that can be used in metabolic engineering. Second, the general structure of the kinetic model is designed based on mathematical formulation using Ordinary Differential Equation (ODE). This design can be done by formulating enzymatic reaction and regulation using ODEs starting from small to large-scale reactions, specifically from small biological pathways to large microbial genomes. The ODEs contain time-dependence state variables (metabolites) and kinetic parameters that measure the rate of changes in metabolites concentrations. Third, values of kinetic parameters such as Michaelis-Menten constant K_m and rate of reaction V are subject to estimation; they empirically influence the model prediction or model output. The final stage of developing a kinetic model is model validation, which consists of various experiments and statistical analyses before the model can be routinely used in industrial biotechnology. Due to the highly nonlinear nature of biochemical reactions, building a kinetic model is a difficult and time consuming process. One of the most difficult tasks in this process is parameter estimation that is used to determine the best possible parameter values that are able to measure the goodness of the predictive model by reproducing the data that is as close to the experimental or real data. Also known as model calibration, system identification or inverse problem, this task is widely used in various application domains ranging from metabolic engineering (Copeland et al., 2012), signal processing (Perez-ramirez et al., 2016) and also control systems engineering (Alfi and Fateh, 2011a). It is important to use highly accurate nonlinear model together with optimal kinetic parameters value for the aforementioned domains to save both time and resources.

This work focuses on the task of parameter estimation of kinetic model in metabolic engineering field, assuming that the structures and experimental data for the kinetic models are provided. Due to the highly nonlinear nature of biological systems, parameter estimation is considered as a multimodal and non-convex optimization problem with the existence of several local minima. To estimate kinetic parameter values in ODEs, optimization methods are employed by minimizing the distance between prediction models (models with parameter estimates) and experimental data. The methods can be divided into two categories: local and global. Local optimization methods such as hill climbing and Newton methods can give unsatisfactory results because their local nature can cause the solutions to be easily stuck in local minima. In addition, their efficiency solely depends on the value of initial solutions that are commonly obtained by in vitro measurements or random guesses (Moles et al., 2003). Since most objective functions in real world problems have several local minima, initial solutions are crucial for local method in finding the global minima. If the initial solution is located far from global minima, the solution might be stuck in local minima although its convergence rate is high. This major drawback has spurred the development of global optimization methods in order to comprehensively find the global minima. Metaheuristic algorithm is one of the most efficient global optimization methods which can be divided into single-solution and population-based searches. Single-solution searches that include iterated local search (ILS), simulated annealing, and variable neighbourhood search are operated through improving single solution within the search space. On the other hand, population-based searches operate through maintaining and improving a set of candidate solutions. The set of solutions qualities are iteratively improved using a particular search mechanism to obtain a better solution. In global optimization, the search process can be divided into intensification (exploitation) and diversification (exploration) (Blum and Roli, 2003). Intensification in search process depends on information obtained from the problem to generate better solution from previous solutions using small changes. This is a typically local process which is suitable in local search method. One of the advantage of intensification process is it has very high convergence rate. However, it may be easily stuck in local minima. On the other hand, diversification process explores the broad search space more efficiently. Hence, it is capable of finding the global solution that is far from the initial point. However, the diversification process may cause slower convergence rate and sometimes leads to high computational cost. Thus, finding the balance between these two search processes is crucial in global optimization problem (Liu et al., 2013a, 2013b).

In parameter estimation and system identification problems, metaheuristic algorithms have been mainly applied in various areas including biochemical kinetic models, control systems engineering and aquatic ecosystems. Single-solution based search, namely differential simulated annealing (DSA) (Dai and Lai, 2014) is proposed to estimate biological network model and the proposed method seems robust and efficient compared to other metaheuristic methods. In order to investigate which methods perform well in this area, several comparative studies of state-of-the-art metaheuristic algorithms in parameter estimation is listed as below. A study in nonlinear dynamic model of an aquatic ecosystem has been presented by Tashkova et al. (2012). Several methods were tested and compared to obtain the most accurate model of ecosystems. Another comparative study of parameter estimation is obtained using crop growth model (Zúñiga et al., 2014). In this study, the authors compare state-of-the-art algorithms such as Differential Evolution (DE), Particle Swarm Optimization (PSO) and Artificial Bee Colony (ABC). Similar study has also been conducted by estimating reservoir parameter for predicting reservoir performance (Awotunde, 2015). Three global optimization methods are tested including Covariance Matrix Adaptation Evolution Strategy (CMAES), DE and PSO. The outcomes of this study indicated that DE and PSO are the most efficient algorithms to be used in parameter estimation problem. In system identification of control systems, Adaptive Particle Swarm Optimization (APSO) (Darabi et al., 2012) has been proposed to identify parameters of an exciter machine. Two modifications are made in order to avoid local convergence as well as to obtain excellent quality of final result. Another interesting work has been proposed based on a novel modified particle swarm optimization (MPSO) (Alfi and Fateh, 2011a) to identify nonlinear system for hydraulic suspension system applications. In their contributions, novel mutation mechanism is introduced in MPSO to enhance the global search ability and it is also capable to increase the convergence speed. Several other variants of PSO have also been applied in intelligent identification and control system using improved fuzzy particle swarm optimization (IFPSO) method (Alfi and Fateh, 2011b) and adaptive particle swarm optimization (APSO) (Alfi and Modares, 2011). Excellent results were obtained from these PSO variants compared to other state-of-the-art metaheuristic methods.

Another notable algorithm that has been proposed in the parameter estimation and bioprocess engineering field is scatter search (SS). SS is one of the most early evolutionary algorithms developed by Glover (1977) that is derived from surrogate constraints method. The main difference between this algorithm and modern metaheuristic algorithm is that search mechanism technique is applied to population members. Unlike other evolutionary algorithms, SS does not use crossover and mutation operator. Instead, it uses a solution combination method that operates among population members. New solutions are generated using systematic (partial random) combination rather than fully random solution. One of the main benefits of SS is it maintains a low number of population sizes, even for large problems. Since small population size is not preferred in many algorithms (because it may

lead to premature convergence or becomes easily trapped in local minima), SS managed to overcome this weakness by ensuring that near optimal solution can be reached. However, not many studies have been conducted using this algorithm, and only a handful of SS algorithms have been proposed by researchers. Although this algorithm is not very popular compared to other metaheuristics algorithms, it is proven that it is able to produce promising results in the field of metabolic and bioprocess engineering. In recent years, several new variants of SS have been proposed with different implementation techniques and search strategies. In parameter estimation of biological systems, a hybrid of stochastic-deterministic global optimization method was suggested by Rodriguez-Fernandez et al. (2006). They proposed a novel optimization algorithm based on SS that can reduce computational efforts and guarantees robustness. The main contribution of their work is a hybrid of local search that runs intensively based on initial solution obtained from SS. Egea et al. (2007) further extended SS by proposing more advanced methods and strategies to obtain a good balance between intensification (exploitation) and diversification (exploration) in SS search mechanism. These strategies include a new mechanism to generate an initial solution with a different order of magnitudes, a new rule for linear combination, and also integrating SS with kriging-based prediction method for reducing computational efforts (Egea et al., 2009). Later, an enhanced scatter search (eSS) algorithm using control vector parameterization (CVP) has been proposed. This algorithm provided a good balance between robustness and efficiency in solving nonlinear process of biological systems (Egea and Balsa-Canto, 2009). Parameter tuning of the search process is a challenging task especially for high dimension problems. To surmount this challenge, Egea et al. (2010) proposed a new evolutionary method using path relinking in eSS combination phase. A more significant work was proposed by Villaverde et al. (2012) which is a novel method based on a cooperative enhanced scatter search (CeSS) to reduce computational time in solving large-scale parameter estimation problems. Although global methods can be used to obtain global optimum, unfortunately their performances deteriorate when used to solve high dimension problems. The process of estimating hundreds of kinetic parameters in a large-scale biochemical system cannot be handled efficiently by this method. High computational costs are expected, especially in terms of total function evaluations, CPU computing time, and slow convergence rate.

The main objective of this work is to efficiently minimize an objective function, which measures the difference between model output and experimental data. SS algorithm was chosen for solving parameter estimation problem. We chose SS instead of other metaheuristic algorithms because the method has been proven to be effective and efficient in solving bioprocess engineering and optimization problems (Egea et al., 2009). SS is robust, requires less tuning parameter, and has higher probability in finding global minima, making it a suitable method to be used to solve the aforementioned problem (Egea et al., 2010). As a result, this study comes out with an improved method that is a hybrid of enhanced scatter search (eSS) together with combined opposition-based learning (SSCOL). This new method modified three important mechanisms in eSS algorithm, namely, reference set (*RefSet*) formation, *RefSet* combination, and *RefSet* intensification in order to improve its performance. The improved algorithm is able to obtain near optimal solutions efficiently and have higher speed of convergence for large-scale parameter estimation problems. In addition, the estimated kinetic parameter values obtained from this work have produced a better predictive model which provided the best fit to the experimental data.

This paper is organized as follows. In Section 2, the general formulation of parameter estimation in kinetic models of biochemical systems is presented. In Section 3, both the original SS and eSS algorithms are presented. In addition, a fundamental background of opposition-based learning is also discussed. In Section 4, a proposed improvement of eSS algorithm is thoroughly described. In Section 5, an experiment is done using large-scale benchmark datasets to test the optimization performance of the proposed algorithm for parameter estimation problems. Finally, conclusions are provided in Section 6.

2. Problem formulation

Parameter estimation consists of searching for parameter values in mathematical model (formulated using ODE) that gives the best fit to the experimental data. This estimation can be done by minimizing a scalar distance between model prediction and experimental data with respect to experimental errors or noise. This problem can be categorized as multimodal, continuous and single objectives optimization problem (Moles et al., 2003). There are many formulas for parameter estimation problems. One of the most popular formulas is weighted nonlinear least squares (Mendes and Kell, 1998; Villaverde et al., 2015) that is considered in this work, which is defined as:

$$J = \sum_{i=1}^{N_E} \sum_{j=1}^{N_{Oi}} \sum_{k=1}^{N_{Sij}} \frac{(ym_{ijk} - ym_{ijk}(\hat{\theta}))^2}{(\sigma_{ijk})^2} \tag{1}$$

where N_E is the number of experiments used in the measurements, N_{Oi} is the number of observables per experiment, N_{Sij} is the number of sample per observable per experiment. The experimental data is denoted as ym_{ijk} and the model prediction is denoted as $ym_{ijk}(\hat{\theta})$, where $\hat{\theta}$ is a set of kinetic parameter vectors that needs to be estimated. Finally, σ_{ijk} are weights used to balance the contributions of observables according to their magnitudes.

Minimization of objective function J is subject to the following constraints:

$$\dot{x} = f(x, \hat{\theta}, t) \tag{2}$$

$$x(t_0) = x_0 \tag{3}$$

$$y = g(x, \hat{\theta}, t) \tag{4}$$

$$\theta^L \leq \hat{\theta} \leq \theta^U \tag{5}$$

where x is the state variable and f is the function describing systems dynamics in nonlinear biochemical process model. The initial condition (concentrations) of x at time zero $t_0$is denoted as x_0 and g is an observation function. Finally, θ^L and θ^U are lower and upper bounds of kinetic parameter vectors $\hat{\theta}$ that need to be estimated. Due to nonlinear nature of biological problems and existence of noises, estimating the best parameters for this problem is difficult because many local minima (non-convex solutions) exist. They cause most optimization algorithms to be easily trapped in local minima, resulting in slow convergence speed. Due to this reason, the resulting parameter estimation will produce poor experimental data fit, resulting to low model prediction accuracy. In addition, the large number of parameters (high dimension) that need to be estimated caused a large amount of computational cost. The costs in this context refer to CPU processing time and number of function evaluations.

3. Fundamental works

3.1. Scatter search (SS) algorithm

Scatter search (SS) algorithm designed by Glover (1998) is based on a formula that combines decision rules and problem constraints. There are five mechanisms in the original SS algorithm namely: 1) diversification generation method, 2) improvement method, 3) reference set (*RefSet*) update method, 4) subset generation method, and 5) solution combination method. Diversification generation method generates a random trial solution within the boundaries and the number of diverse solution *ndiverse* is quite large ($ndiverse = 10 \times npar$, where $npar$ is the problem dimension). The large *ndiverse* solution size ensures a broad search space for feasible and promising directions. Meanwhile, im-

provement method enhances the trial solutions using sophisticated local search or/and any other procedures. Next, *RefSet* update method generates and maintains *RefSet* members to be used in the entire search processes. The size of *RefSet* members is small, leading to low computational costs but high possibility of obtaining the global optimum. Subset generation method generates pairs of *RefSet* members whereas solution combination method combines every element in *RefSet* to produce a new solution. These scatter search mechanisms are considered straightforward and easy to implement, but in the same time, they allow more sophisticated improvements and modifications to tackle the unique nature of the optimization problem at hand.

3.2. Enhanced scatter search (eSS) algorithm

Enhanced scatter search (eSS) algorithm is a recent variant of SS developed by Egea et al. (2010). This algorithm is designed with advanced mechanisms to solve various optimization problems including large-scale parameter estimation. The main difference between the original SS and eSS is the strategy used in combination method. A conventional linear combination introduced by Glover (1998) is suitable for small dimension problems, while a hyper-rectangle combination proposed in eSS makes use of relative position and direction from every *RefSet* member and systematically generates a new solution inside them. Another main contribution of eSS is the hybridization of local search that is controlled based on merit. Local search cannot be launched when some criteria are not met (i.e. fitness merit and distance filter). In addition, this algorithm uses less tunable control parameters since their *RefSet* size is calculated automatically.

In short, eSS works as follows:

Step 1: Diverse solution vectors generation

The initial set S of m diverse vectors are generated randomly from uniform distribution in the range of $[lb, ub]$ with $m = 10 \times npar$, where $npar$ is decision variables or problem dimension. The generated vectors are evaluated and the half best ones ($b/2$) in terms of quality are selected to form a new *RefSet*, where b is the *Refset* size that is calculated by obtaining the positive value of the quadratic equation roots:

$$b = x^2 - x - \frac{10 \times npar}{1} \tag{6}$$

The remaining *RefSet* members are chosen from S by random permutation to enhance the diversity of the initial solution.

Step 2: Solution combination

The *Refset* members are sorted and each member is combined to produce a pair of every *RefSet* member. Bias β is introduced with respect to relative distance and position of every *RefSet* member. The solution combinations are defined as:

$$c_1 = x^i - d\,(1 + \alpha \times \beta) \tag{7}$$

$$c_2 = x^i - d\,(1 - \alpha \times \beta) \tag{8}$$

where

$$d = \frac{x^j - xi}{2} \tag{9}$$

$$\alpha = \begin{cases} 1 & if i < j \\ -1 & if j < i \end{cases} \tag{10}$$

and

$$\beta = \frac{|j - i| - 1}{b - 2} \tag{11}$$

The new solutions x^{new} are created randomly within the defined combination of c_1 and c_2:

$$x^{new} = c_1 + (c_2 - c_1) \cdot rand \tag{12}$$

where *rand* is a random values generated using uniform distribution within the range [0,1]. The advantage of using this strategy is that it produces high quality solutions for every *RefSet* member. This strategy will ensure that low quality solutions will be improvised, while maintaining high quality solutions. This strategy is capable of avoiding the solution from being stuck in local minima.

Step 3: Population update

In evolutionary algorithm, a new population is generated based on their parents μ and offspring λ. The $(\mu + \lambda)$ strategy generates a new offspring between μ and λ which provides a good balance between intensification and diversification. However, this strategy is likely to converge prematurely in parameter estimation problems. The (μ,λ) strategy generates a new offspring based on the previous offspring and enables it to obtain global optima. However, this method requires high computational costs (high number of function evaluations). On the other hand, eSS uses (1+1) strategy, which means it generates 1 offspring from 1 parent for every member of *RefSet*. This strategy provides a good balance between global search and local search in order to avoid the solution getting stuck in local minima while preserving minimum computational costs.

Step 4: Intensification

Besides *RefSet* combinations, eSS also implements an advanced specific intensification method. If the new solution generated by combination method outperforms their parents (in terms of fitness value), the new non-convex solution is created in the direction defined by their parents and child. The new solution (child) becomes a new parent and the new created solution becomes a new child. This improvement continues until no new child can outperform its parents. Following this method, the final solution will be of very high quality and has a promising direction, possibly close to the global optimum. One thing to note is this strategy will generate an inconsistent number of function evaluations in every run, since evaluations in this step depend on the quality of *RefSet* members that are produced from randomization.

Step 5: Local search

Several local search methods are already implemented in eSS to be used in wide range optimization problems including algorithm based on Sequential Quadratic Programming (SQP) and dynamic hill climbing (DHC). The local search method incorporated in eSS is used to perform intensification phase to accelerate the convergence speed using best found minima from previous searches. The choices of initial solution to be used as candidate solution in local search are controlled using a mechanism that balance between quality and diversity. This systematic selection is performed based on competitive ranking. The solution found from Step 4 is divided into two lists. The first one contains the original solutions that are sorted based on quality. The second one contains the solutions from the first list that have been sorted based on minimal distance. In this case, Euclidean distance is used to calculate the distance of the existing minima found. By using competitive ranking, high quality solutions which are far from the previous search (found minima) are favoured to be used as initial point local search. Detailed explanation on this topic can be found in Egea and Balsa-Canto (2009).

For stopping criteria, eSS takes the combination of number of function evaluations and CPU time. If one of the stopping criteria is met, computation is terminated. Otherwise, steps 2–4 need to be repeated. Generally, the most important steps in eSS are solution combination and intensification. These mechanisms give a good balance between diversification and intensification, and avoid too much evaluation without finding promising directions.

3.3. Opposition-based learning (OBL)

Opposition-based learning (OBL) was introduced by Tizhoosh (2005) in machine learning field. Its concept is quite simple. If the random estimated point or initial guess is far from the optimal solution, then approximation and search for the solution consumes

more computation time and in the worst case, it becomes intractable. Using opposite point of initial guess, the search process will most likely becomes fast, in other words, it is able to accelerate convergence speed. Both initial and opposite guesses must be evaluated simultaneously and the one closest to the solution is selected for generating new solutions. In the past few years, different variants of OBL were proposed, namely quasi opposition-based learning (Rahnamayan et al., 2007), quasi-reflection (Ergezer and Simon, 2014) and center-based sampling (Seif and Ahmadi, 2015; Xu et al., 2014). Recent studies suggested that quasi-opposition and quasi-reflection have shown promising results in solving general optimization problems. For initialization methods in optimization algorithm, using quasi opposition-based learning can provide fitter starting candidate even if there is no prior knowledge about the problem (Rahnamayan et al., 2007). The authors have presented mathematical proof and empirical test was also done over a set of benchmark problems. This study confirms that this method has higher chance to obtain a result that is close to the global solution. Meanwhile, quasi-reflection solution is placed randomly between the solution candidate and the center of the solution space. The empirical studies and mathematical proof found that quasi-reflection has a higher probability of being closer to the solution compared to quasi-opposition (Ergezer and Simon, 2014). Hence, quasi-opposition and quasi-reflection tend to search center part of the parameters and the combination of these methods may improve the performance of the algorithm. The definition of OBL is shown in Eqs. (13)–(16) while Fig. 1 shows the graphical representation of OBL.

Let $X(x_1, x_2,..,x_D)$ be the point of variable x_i in the dimension D that is generated randomly and each variable x_i is bounded in a lower and an upper bounds $[lb_i, ub_i]$. The opposite point of $\hat{X}_i(\hat{x}_i, \hat{x}_2,..,\hat{x}_D)$ is defined by:

$$\hat{x}_i = lb_i + ub_i - x_i \tag{13}$$

Now, considering the opposite points in Eq. (13), quasi-opposite points of $\hat{X}_{qo}$ are selected randomly between the opposite point of $\hat{X}$ and center point C as follows:

$$\hat{x}_{qo} = \begin{cases} rand(c, \hat{x}) \text{ if } x \leq c \\ rand(\hat{x}, c) \text{ if } x > c \end{cases} \tag{14}$$

where *rand* is a random values generated within $\hat{X}$ and center point C:

$$c_i = \frac{lb_i + ub_i}{2}, \quad \forall i \in \{1, 2, ..D\} \tag{15}$$

Considering center point C in Eq. (15), quasi-reflected points $\hat{X}_{qr}$ are drawn randomly in the range between center point C and X as follows:

$$\hat{x}_{qr} = \begin{cases} rand(c, x) \text{ if } x > c \\ rand(x, c) \text{ if } x \leq c \end{cases} \tag{16}$$

Now, let J be the cost function to be minimized, if $J(\hat{x}_{qo})$ or $J(\hat{x}_{qr})$ has better fitness than $J(x)$, the initial point X will be replaced with $\hat{x}_{qo}$ or $\hat{x}_{qr}$. Otherwise, X stays in the current population.

4. Enhanced scatter search algorithm with combined opposition-based learning (the proposed algorithm)

The key differences between the proposed algorithm in this study and eSS are the strategies used to generate initial *RefSet* which contains high quality members, modifications of every *RefSet* member combination, and probabilities to perform the specific search intensification for selected high quality members in *RefSet*. In particular, this study proposed a combination of quasi-opposition *RefSet* formation, quasi-reflection combination, and quasi-reflection intensification in eSS. The quasi-opposition *RefSet* formation may improve the diversification process of the algorithm by incorporating quasi-opposition based learning in initial *Refset* formation. Meanwhile, the combination of quasi-reflection in combination and intensification process can improve intensification for the search process. These mechanisms give a good balance (which is important an aspect of global optimization algorithm) between diversification and intensification (Egea et al., 2010; Liu et al., 2013a, 2013b). These proposed strategies may improve the performance of eSS algorithm in terms of its efficiency to search global minimum (in the case of parameter estimation and global optimization), accelerate convergence speed and also reduce computational cost.

4.1. Building the initial RefSet by diversification generation method

The algorithm is started by creating an initial set of m number of diverse vectors (*ndiverse*) using uniformly distributed random number. The size of m is ten times bigger than the problem dimension, which is the recommended size in eSS. All these vectors are evaluated and sorted according to their fitness. A high quality *ndiverse* set is selected to form half of the *dim_refset*/2 *RefSet* members, where *dim_refset* is *RefSet* size. Then, the remaining members are selected from $m - dim\_refset/2$ by random permutation to complete the *RefSet* formation. Here, half of the *RefSet* members which are selected by the random permutation will produce the quasi-opposite *Refset* as defined in the dotted box in Algorithm 1 (Table 1). If each of the generated quasi-opposite *RefSet* member has a better fitness than the original *RefSet* member, then the quasi-opposite *RefSet* member is selected to join the remaining *RefSet*. Otherwise, the original *RefSet* members stay in the current *RefSet*. It should be noted that this strategy does not produce quasi-opposite of all *RefSet* members, but only half of them. Thus, it eliminates high computational costs in terms of function evaluations, while preserving high quality *RefSet* and remaining *RefSet* (quasi-opposite) members. This strategy may result to high quality initial RefSet members.

4.2. Combination method

After the initial *RefSet* is formed, the quality of every *RefSet* member is sorted again before the combination method is performed. The combination method is based on hyper-rectangles which created $dim\_refset - 1$ hyper-rectangles. Every $dim\_refset^2 - dim\_refset$ iteration will create new solutions inside every hyper-rectangle. In this combination method, the eSS algorithm uses the new bias point of every member in *RefSet* (*V1, V2, V3*) for every pair of solution to create a new solution inside them. In this work, a new center point is defined

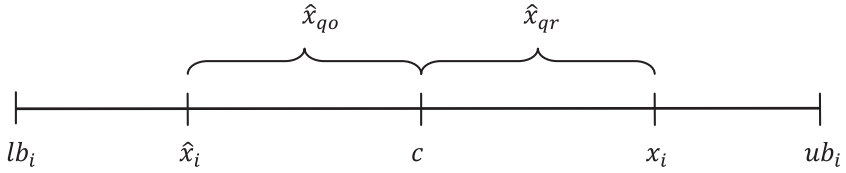

Fig. 1. The boundary of search space is defined in the range $[lb_i, ub_i]$. x_i are the random generated values and their opposites are $\hat{x}_i$. c is the center-based point between lb_i and ub_i. x_{qo} and x_{qr} is the quasi-opposition and quasi-reflected point, respectively.

Table 1
Algorithm 1. Pseudo-code of quasi-opposition *RefSet* formation.

Set parameters: $ndiverse\ m\ (m = 10\ \times\ nvar)$, RefSet size *dim_refset*
Initialize lower bound $[LB]$ and upper bound $[UB]$

Generate set of m $ndiverse$ vectors x^i randomly with $x^i \in [LB, UB]$
Sort x^i $[x^1, x^2, \dots, x^m]$ so that $f(x^i) \le f(x^j)\ \forall j > i$
Generate first $RefSet^i$ from x^i, $x^i \in [x^1, x^2, \dots, x^{dim\_refset/2}]$
Generate remaining $Refset$ from $ndiverse - ndiverse/2$ by random permutation

Compute center point C between LB and UB for every *nvar* **(Eq. 15)**
for $i = \frac{dim\_refset}{2} + 1$ to dim_refset **do**
  Generate opposite *RefSet* $\hat{x}^i$ **(Eq. 13)**
  Generate quasi-opposite *RefSet* $\hat{x}_q^i$ **(Eq. 14)**
  if $f(\hat{x}_q^i) < f(x^i)$ **then**
    $RefSet^i = \hat{x}_q^i$
  else
    $RefSet^i = x^i$
  end if
end for

Note: The improvement using quasi-opposition is depicted in the dotted box. *ndiverse* is number of diverse vectors. *nvar* is problem dimension.

Table 2
Algorithm 2. Pseudo-code of quasi-reflection combination.

Initialize: x_{parent}^1 and x_{parent}^2 from $RefSet^i$, bias points (V_1, V_2, V_3) and a pair of new *RefSet* combination ($New\_Comb\_QR_1$ and $New\_Comb\_QR_2$)

Compute bias β **(Eq. 11)**
$V_1 = x_{parent}^1 - \beta$
$V_2 = x_{parent}^2 - \beta$
$V_3 = 2 \times x_{parent}^1 - x_{parent}^2 - \beta$

$New\_Comb_1 = V_1 + (V_2 - V_1) \cdot rand$
$New\_Comb_2 = V_2 + (V_3 - V_2) \cdot rand$

Compute center point C_1 between V_1 and V_2, and C_2 between V_2 and V_3 **(Eq. 15)**

$New\_Comb\_QR_1 = New\_Comb_1 + (C_1 - New\_Comb_1) \cdot rand$ **(Eq.12)**
$New\_Comb\_QR_2 = New\_Comb_2 + (C_2 - New\_Comb_2) \cdot rand$ **(Eq.12)**

Merge $New\_Comb\_QR_1$ and $New\_Comb\_QR_2$
Update new offspring x_{off}^i

Note: The improvement using quasi-reflection combination is depicted in the dotted box.

between *RefSet* members and quasi-reflection *RefSet*. Then, these combinations are merged to form a new offspring x_{off}^i. These steps are depicted in dotted box of Algorithm 2 (Table 2). Unlike quasi-opposition *RefSet* formation, in this combination method, objective function f is not evaluated on all quasi-reflection members of *RefSet* to maintain small numbers of function evaluations. This strategy also aims to reduce the size of search region defined by the bias point. Thus, this method does not only produce a better *RefSet* member with quasi-reflection and bias point, but also consumes zero function evaluation.

4.3. Intensification method

Another key element of eSS algorithm is the mechanism to perform specific intensification by selecting a pair of *RefSet* members that outperform its parents (initial *RefSet*). The new solution is created based on the direction defined by the child and its parent. The child becomes a parent and the newly generated solution becomes the new child. This mechanism continues to generate new child until no further improvement is achieved. To further enhance this intensification method, the same strategy as used in the previous section is employed, namely quasi-reflection. However, in this process, a control parameter is introduced which is jumping rate J_r, where $J_r \in [0,1]$. In the case of an optimal set, the jumping rate introduced in this process can produce a good result (Rahnamayan et al., 2008). The algorithm for quasi-reflection intensification is depicted in the dotted box in Table 3. In this table, *rand* is the uniformly distributed random number in the range of [0,1].

An issue may rise on why quasi-reflection is not being employed in

Table 3
Algorithm 3. Pseudo-code of quasi-reflection intensification.

Set parameter: Jumping rate J_r
Initialize offspring x_{off}^i and parent x^i,

x_{temp}=x^i
$improvement = 1$
$\Lambda = 1$

while $f(x_{off}^i) < f(x_{temp})$ **do**

if $rand > J_r$
Create a new solution x_{new} in the rectangle defined by $[x_{off}^i - \frac{x_{temp} - x_{off}^i}{\Lambda}, x_{off}^i]$
Compute center point x_{cb} between x_{new} and x_{off}^i **(Eq. 15)**
Create a quasi-reflection $\hat{x}_{new}$ in the rectangle defined by $[x_{new} + (x_{cb} - x_{new}) \cdot rand]$ (**Eq. 12**)
$x_{temp} = x_{off}^i$
$x_{off}^i = \hat{x}_{new}$
improvement=improvement+1
else
Create a new solution x_{new} in the rectangle defined by $[x_{off}^i - \frac{x_{temp} - x_{off}^i}{\Lambda}, x_{off}^i]$
$x_{temp} = x_{off}^i$
$x_{off}^i = x_{new}$
improvement=improvement+1
end if

if improvement=2 **then**
$\Lambda = \Lambda/2$
$improvement = 0$
end if
end while

Note: The quasi-reflection with jumping rate $J_r$improvement is depicted in the dotted box.

RefSet formation or why quasi-opposition is not used in the combination method. To answer this, from the initial experiment conducted, it is found that quasi-opposition based learning is the most suitable option for *RefSet* formation which is also confirmed by previous work (Kazimipour et al., 2013). On the other hand, quasi-reflection is chosen for performing the combination method because according to the literature, it a has higher probability of being closer to the solution compared to quasi-opposition method, specifically in search processes (Ergezer and Simon, 2015). The overall improvement of eSS algorithm is depicted in Table 4 where the modified search processes namely quasi-opposite *Refset* formation, quasi-reflection combination and quasi-reflection intensification are shown in the dotted box. In this proposed algorithm, the objective function evaluation is done in several phases namely: initial *Refset* formation phase, before and after performing quasi-reflection intensification and during the local search phase. It should be noted that this algorithm consumes a large number of fitness evaluation, hence, it is unfair to compare this algorithm with other algorithms based on equal number of iteration (Črepinšek et al., 2016; Mernik et al., 2015).

4.4. Control parameters

The proposed algorithm used specific control parameters settings according to the standard eSS algorithm. The parameters and their descriptions are shown in Table 5. The number of diverse solution (*ndiverse*) is typically large enough to sample the broad search space. The *RefSet* size on the other hand is equivalent to the population size in evolutionary algorithm, which means it must be small enough to reduce computational times that are typically below 20 for most optimization problems. It should be noted that the best quality solution from *ndiverse* is selected to form an initial *RefSet*. *Local.n2* denotes the number of algorithm iteration between two constitutive local searches. The local search phase will not launch if the algorithm's number of iterations (*Local.n2*) is not met. The selection of initial solution of local search is based on the best solution found from previous search (Egea and Balsa-Canto, 2009). Finally, *Balance* denotes the value of quality (intensification) and diversity (diversification) for the local search's initial point selection.

All these parameters have the same influence on the algorithms' performance. If the parameters are large, they will focus on exploring broad search space by spending more time to combine parameters vectors. This will result to long computational time with higher possibility to obtain global minimum. Meanwhile, if the parameters are small, they tend to locate the solutions aggressively and quickly including performing frequent local search and keeping small number of *RefSet* members. This will consume shorter computational time with less probability to obtain global minimum. Large parameters value is suitable for problems with rugged parameter spaces where several local minima exist. Meanwhile, small parameters value is suitable for problems that have a smooth space. Hence, it is crucial to find the balance between these values to be used in various optimization problems (Egea et al., 2010; Villaverde et al., 2012).

5. Experiments

5.1. Dataset and experimental setup

To test and evaluate the proposed algorithm's performance, two different sets of experiment were carried out. In the first one, a set of well known benchmark functions from large-scale global optimization

Table 4
Algorithm 4. Overall pseudo-code for an enhanced scatter search algorithm with combined opposition-based learning (SSCOL).

Generate initial $RefSet$ with high quality and quasi-opposite members **(Algorithm 1)**

repeat

Sort $RefSet$ by quality $[x^1, x^2, \ldots, x^{dim\_refset}]$ so that $f(x^i) \leq f(x^j)$ where $i, j \in [1,2, \ldots, dim\_refset]$ and $i < j$

if $max\left(abs\left(\frac{x^i - x^j}{x^j}\right)\right) \leq \epsilon$ with $i < j$ **then**

Replace x^j by a random solution

end if

for $i = 1$ *to* dim_refset **do**

Combine x^i with the rest of population members to generate a set of *dim_refset* new members, **offspring**^{i} **(Algorithm 2)**

x^i_{off} = best solutions in **offspring**^{i}

if x^i_{off} outperforms x^i **then**

Apply quasi-reflection intensification (**Algorithm 3**)

end if

end for

Update best solution found x_{best} and its objective function value f_{best}

Perform a local search from x_{best} to obtain x^* based on competitive ranking initial selection and balance between quality and diversity

if $f(x^*) < f(x_{best})$

Update x_{best}, f_{best}

end if

until stopping criterion is met

Note: The improvements are depicted in the dotted box.

Table 5
Control parameters in SSCOL.

Parameter	Description	Default value
Number of diverse solutions (*ndiverse*)	Number of initial diverse solutions	10×*nvar*
RefSet size (*dim_refset*)	Number of elements in the Reference Set	"auto generated" using Eq. (6)
Local.n2	Minimum number of iterations of the eSS algorithm between two local searches (local.n2)	10
Balance	Balance between intensification (0) and diversification (1) in the selection of initial points for local searches	0.5

Note: *nvar* is the number of decision variable/problem dimension.

(LSGO) problem is used. In this experiment, the benchmark functions are taken from a competition on LSGO, which is organized in conjunction with IEEE Congress on Evolutionary Computation 2012 (CEC12). The benchmark functions can be downloaded from http://staff.ustc.edu.cn/~ketang/cec2012/lib/lsgo_benchmark.zip. Three categories with different challenges and complexities (unimodal/multimodal/non-separable/full non-separable) of 11 functions were selected for testing:

- *D*/2*m*-group m-non-separable functions (F10-F13):
  a. F10: *D*/2*m*-group Shifted and *m*-rotated Rastrigin's Function
  b. F11: *D*/2*m*-group Shifted and *m*-rotated Ackley's Function
  c. F12: *D*/2*m*-group Shifted *m*-dimensional Schwefel's Function
  d. F13: *D*/2*m*-group Shifted *m*-dimensional Rosenbrock's Function
- *D*/*m*-group *m*-non-separable functions (F14-F18):
  a. F14: *D*/*m*-group Shifted and *m*-rotated Elliptic Function
  b. F15: *D*/*m*-group Shifted and *m*-rotated Rastrigin's Function
  c. F16: *D*/*m*-group Shifted and *m*-rotated Ackley's Function
  d. F17: *D*/*m*-group Shifted *m*-dimensional Schwefel's Function
  e. F18: *D*/*m*-group Shifted *m*-dimensional Rosenbrock's Function
- Fully non-separable functions (F19-F20):
  a. F19: Shifted Schewefel's Function
  b. F20: Shifted Rosenbrock's Function

where *D* is the problem dimension (*D=1000*) and *m* is the grouping structure (*m=50*) (if applicable).

In the second experiment, a set of large-scale kinetic model of biochemical systems were used as summarized in Table 6. The two datasets used in this study are Chinese hamster ovary (CHO) cells (Wurm, 2004) and central carbon metabolism (CCM) in *E. coli* (Chassagnole et al., 2002). CHO cells are used in clinical applications specifically for therapeutic protein production. In these cells, 13 metabolites are observed; in fermenter: glucose, lactate, product protein, leucine and methionine; in cytosol: aspartate, malate, pyruvate, oxaloacetate, ATP and ADP; and in mitochondria: ATP and ADP. The number of kinetic parameters that need to be estimated is 117, which is considered as high dimension. In CCM, 116 kinetic para-

Table 6
Description of the datasets.

Dataset	Level	Number of kinetic parameters	Number of observed metabolites	Noise level	Lower bound	Upper bound	References
Chinese hamster ovary (CHO) cell	Metabolic	117	13	Variable	$0.2 \times p_{nom}$	$5 \times p_{nom}$	(Wurm, 2004)
E. coli	Metabolic: central carbon metabolism (CCM)	116	9	Real	$0.1 \times p_{ex}$	$10 \times p_{ex}$	(Chassagnole et al., 2002)

Note: Variable noise contains added Gaussian noise with 20% standard deviation. Real noise is noise data from experimental error reported in the publication. p_{nom} is the nominal kinetic parameter vector that produces the experimental data and p_{ex} is the nominal kinetic parameter vector from original publication. Lower and upper bounds are the functions of the nominal parameters used in optimization methods.

meters need to be estimated consisting of 9 metabolites: pep, glucose, g6p, pyr, f6p, g1p, 6pg, fdp and gap. The purpose of this model is to reproduce the response to a pulse in extracellular glucose concentration. For the initialization of boundaries values, function of nominal parameter p_{nom} is used. For CHO, p_{nom} is the parameter vector that refers to the reference value, or parameters that produced the data (without noise), which is also known as pseudo-experimental data. For CCM, p_{ex} is the original parameter obtained from the original publication. The mathematical formulation and modeling procedure can be referred from Villaverde et al. (2014) for CHO and Chassagnole et al. (2002) for CCM. The lists of ODE mathematical models of both data can be found in Supplementary file (Model Equations) which is not included in this paper due to large number of equations involved. In addition, the models can also be downloaded from BioPredyn (Villaverde et al., 2015) which is available at http://gingproc.iim.csic.es/biopredynbench/.

For the sake of comparison and to justify the choices of our proposed algorithm, we compared it with the original enhanced scatter search (eSS) algorithm which was obtained from MEIGO toolbox (Egea et al., 2014). In addition, the comparison is also made using two other competitive metaheuristic algorithms which are widely used in parameter estimation problem namely Differential Evolution (DE) (Storn and Price, 1997) and Particle Swarm Optimization (PSO) (Eberhart and Kennedy, 1995). We also compared this work with a published benchmark that used the original eSS with the same datasets. All experiments were conducted using MATLAB 2015a on Dell Precision T1700 workstation with Intel Core i7 3.6 Ghz processor and 16 GB RAM in Windows 10. For each algorithm, the optimization was carried out 20 times using different seeds for the random number generator. It should be noted that these benchmark problems require lengthy CPU processing time, which takes around 1–3 h for a single run (approximately 20–60 h for 20 runs). To surmount this burden, all eight available cores of the Intel Core i7 processors are used to run in parallel setting. To do this, parallel computing toolbox in MATLAB is utilized to run eight parallel independent experiments. Using this setting, it only takes approximately 3–9 h to perform 20 runs for each algorithm.

5.2. Control parameter setting

The search parameter values used in each algorithm are shown in Table 7. It should be noted that to provide a proper comparison of the benchmark results, total number of function evaluations is used as the stopping criteria. The previous benchmark used CPU time as stopping criteria resulting to unfair comparison due to different hardware and platform specifications. On the other hand, using a local search method in this problem is very important as it can accelerate convergence speed and increase intensification phase. This experiment used the recommended gradient-based local search FMINCON which is based on sequential quadratic programming (SQP) algorithm.

Table 7
Parameter setting in SSCOL and eSS.

Parameters	Values
Number of diverse solutions (*ndiverse*)	1170 for CHO and 1160 for CCM
RefSet size (*dim_refset*)	36
Local.n2	10
Balance	0.5
Jumping rate J_r	0.3

Note: Jumping rate J_r is only applicable to SSCOL. The number of function evaluation for CHO dataset is 120,000 and CCM is 90,000.

For *ndiverse, dim_refset, Local.n2* and *Balance* parameters, we set them to default values as shown in Table 7. These values are achieved after conducting extensive initial experiments. We found that the default value is quite robust, and can be used as initial value to perform more rigorous experiments. This means that users do not need to spend much time on tuning these parameters especially for computationally expensive problem. The default values presented here have also been suggested in work by Gábor and Banga (2015) and also in original implementation of the benchmark model in using original eSS (Villaverde et al., 2015), which is considered the best parameters value found so far for this problem.

Meanwhile, for DE, the control parameters are set as follows: scaling factor F =0.7, crossover rate Cr =0.8 and search strategy is *DE/rand/1/bin*. These parameters are suggested for parameter estimation problem in biochemical kinetic models (Zúñiga et al., 2014; Da Ros et al., 2013). Regarding PSO, inertial weight w is 0.7, $c1$ and $c2$ are 1.5 and 2.0, respectively. The PSO parameters are set after we have found in initial experiments that these values gave the best performance. Both population sizes Np for PSO and DE are set to 36, which is the same as *dim_refset* for SSCOL. Since both DE and PSO used number of iteration as the stopping criteria, for a proper comparison, we have adjusted the number of iteration for CHO and CCM models so that SSCOL also consumed the same number of function evaluations.

5.3. Experimental results

5.3.1. Large-scale global optimization (LSGO) benchmark functions

Finding the global minimum of the LSGO function in the first set of problem is difficult and time consuming. To assess the performance of the proposed algorithm, the result is compared to four different methods: DECC-G (Yang et al., 2008a), DECC-G* (a modification of DECC-G), MLCC (Yang et al., 2008b) and eSS (Villaverde et al., 2012). SSCOL was run 25 times using different seeds for the random number generator and maximum of 300,000 function evaluations was allowed, following the evaluation criteria for LSGO competition. Results of the three methods: DECC-G, DECC-G* and MLCC are taken from the competition's webpage (http://staff.ustc.edu.cn/~ketang/cec2012/lsgo_competition.htm) and the result of eSS are taken from Villaverde et al. (2012). It should be noted that DECC-G* method uses grouping structure which is not allowed for benchmark competition and is regarded as an "unfair" method. Table 8 compares the results obtained using DECC-G, DECC-G*, MLCC and eSS with our proposed algorithm. Finally, Table 9 ranks the methods according to their performance score (based on mean value).

Following Tables 8 and 9, no single algorithm outperforms other

Table 8
Optimization results for large-scale global optimization problem (F10-F20).

F10	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$1.03 \cdot 10^{4}$	$2.33 \cdot 10^{3}$	$2.52 \cdot 10^{3}$	$5.57 \cdot 10^{3}$	$4.74 \cdot 10^{3}$
Median	$1.07 \cdot 10^{4}$	$2.49 \cdot 10^{3}$	$3.16 \cdot 10^{3}$	$5.96 \cdot 10^{3}$	$5.97 \cdot 10^{3}$
Worst	$1.17 \cdot 10^{4}$	$2.64 \cdot 10^{3}$	$5.90 \cdot 10^{3}$	$6.26 \cdot 10^{3}$	$6.69 \cdot 10^{3}$
Mean	$1.06 \cdot 10^{4}$	$2.48 \cdot 10^{3}$	$3.43 \cdot 10^{3}$	$5.94 \cdot 10^{3}$	$5.91 \cdot 10^{3}$
Std	$2.95 \cdot 10^{2}$	$7.63 \cdot 10^{1}$	$8.72 \cdot 10^{2}$	$1.88 \cdot 10^{2}$	$4.51 \cdot 10^{2}$
F11	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$2.06 \cdot 10^{1}$	$5.82 \cdot 10^{-8}$	$1.96 \cdot 10^{2}$	$1.93 \cdot 10^{2}$	$1.90 \cdot 10^{2}$
Median	$2.33 \cdot 10^{1}$	$7.52 \cdot 10^{-8}$	$1.98 \cdot 10^{2}$	$1.95 \cdot 10^{2}$	$1.93 \cdot 10^{2}$
Worst	$2.79 \cdot 10^{1}$	$8.79 \cdot 10^{-1}$	$1.98 \cdot 10^{2}$	$1.96 \cdot 10^{2}$	$2.03 \cdot 10^{2}$
Mean	$2.34 \cdot 10^{1}$	$3.52 \cdot 10^{-2}$	$1.98 \cdot 10^{2}$	$1.95 \cdot 10^{2}$	$1.94 \cdot 10^{2}$
Std	$1.78 \cdot 10^{0}$	$1.76 \cdot 10^{-1}$	$6.98 \cdot 10^{-1}$	$6.09 \cdot 10^{-1}$	$2.70 \cdot 10^{0}$
F12	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$7.78 \cdot 10^{4}$	$6.16 \cdot 10^{1}$	$2.42 \cdot 10^{4}$	$1.97 \cdot 10^{4}$	$1.09 \cdot 10^{4}$
Median	$8.87 \cdot 10^{4}$	$7.72 \cdot 10^{1}$	$3.47 \cdot 10^{4}$	$2.93 \cdot 10^{4}$	$2.19 \cdot 10^{4}$
Worst	$1.07 \cdot 10^{5}$	$1.19 \cdot 10^{2}$	$4.25 \cdot 10^{4}$	$4.79 \cdot 10^{4}$	$3.01 \cdot 10^{4}$
Mean	$8.93 \cdot 10^{4}$	$7.87 \cdot 10^{1}$	$3.49 \cdot 10^{4}$	$3.15 \cdot 10^{4}$	$2.09 \cdot 10^{4}$
Std	$6.87 \cdot 10^{3}$	$1.41 \cdot 10^{1}$	$4.92 \cdot 10^{3}$	$8.46 \cdot 10^{3}$	$4.99 \cdot 10^{3}$
F13	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$1.78 \cdot 10^{3}$	$3.78 \cdot 10^{2}$	$1.01 \cdot 10^{3}$	$6.12 \cdot 10^{2}$	$1.17 \cdot 10^{2}$
Median	$3.00 \cdot 10^{3}$	$5.40 \cdot 10^{2}$	$1.91 \cdot 10^{3}$	$1.02 \cdot 10^{3}$	$6.89 \cdot 10^{2}$
Worst	$1.66 \cdot 10^{4}$	$7.55 \cdot 10^{2}$	$3.47 \cdot 10^{3}$	$2.51 \cdot 10^{3}$	$1.58 \cdot 10^{3}$
Mean	$5.12 \cdot 10^{3}$	$5.50 \cdot 10^{2}$	$2.08 \cdot 10^{3}$	$1.17 \cdot 10^{3}$	$7.19 \cdot 10^{2}$
Std	$3.95 \cdot 10^{3}$	$9.78 \cdot 10^{1}$	$7.27 \cdot 10^{2}$	$4.81 \cdot 10^{2}$	$3.75 \cdot 10^{2}$
F14	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$6.96 \cdot 10^{8}$	$2.46 \cdot 10^{7}$	$2.62 \cdot 10^{8}$	$2.37 \cdot 10^{7}$	$1.27 \cdot 10^{6}$
Median	$8.07 \cdot 10^{8}$	$2.90 \cdot 10^{7}$	$3.16 \cdot 10^{8}$	$3.31 \cdot 10^{7}$	$2.46 \cdot 10^{6}$
Worst	$9.06 \cdot 10^{8}$	$3.56 \cdot 10^{7}$	$3.77 \cdot 10^{8}$	$4.68 \cdot 10^{7}$	$3.99 \cdot 10^{6}$
Mean	$8.08 \cdot 10^{8}$	$2.91 \cdot 10^{7}$	$3.16 \cdot 10^{8}$	$3.29 \cdot 10^{7}$	$2.44 \cdot 10^{6}$
Std	$6.07 \cdot 10^{7}$	$2.91 \cdot 10^{6}$	$2.77 \cdot 10^{7}$	$6.15 \cdot 10^{6}$	$7.03 \cdot 10^{5}$
F15	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$1.09 \cdot 10^{4}$	$3.62 \cdot 10^{3}$	$5.30 \cdot 10^{3}$	$6.84 \cdot 10^{3}$	$4.32 \cdot 10^{3}$
Median	$1.18 \cdot 10^{4}$	$3.88 \cdot 10^{3}$	$6.89 \cdot 10^{3}$	$7.71 \cdot 10^{3}$	$5.27 \cdot 10^{3}$
Worst	$1.39 \cdot 10^{4}$	$4.25 \cdot 10^{3}$	$1.04 \cdot 10^{4}$	$8.09 \cdot 10^{3}$	$6.64 \cdot 10^{3}$
Mean	$1.22 \cdot 10^{4}$	$3.88 \cdot 10^{3}$	$7.11 \cdot 10^{3}$	$7.71 \cdot 10^{3}$	$5.25 \cdot 10^{3}$
Std	$8.97 \cdot 10^{2}$	$1.76 \cdot 10^{2}$	$1.34 \cdot 10^{2}$	$2.36 \cdot 10^{2}$	$5.79 \cdot 10^{2}$
F16	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$5.97 \cdot 10^{1}$	$7.04 \cdot 10^{-8}$	$2.08 \cdot 10^{2}$	$3.83 \cdot 10^{2}$	$3.79 \cdot 10^{2}$
Median	$7.51 \cdot 10^{1}$	$1.04 \cdot 10^{-7}$	$3.95 \cdot 10^{2}$	$3.84 \cdot 10^{2}$	$3.81 \cdot 10^{2}$
Worst	$9.24 \cdot 10^{1}$	$2.18 \cdot 10^{0}$	$3.97 \cdot 10^{2}$	$3.85 \cdot 10^{2}$	$3.84 \cdot 10^{2}$
Mean	$7.66 \cdot 10^{1}$	$4.01 \cdot 10^{-1}$	$3.76 \cdot 10^{2}$	$3.84 \cdot 10^{2}$	$3.81 \cdot 10^{2}$
Std	$8.14 \cdot 10^{0}$	$6.59 \cdot 10^{-1}$	$4.71 \cdot 10^{1}$	$5.58 \cdot 10^{-1}$	$1.37 \cdot 10^{0}$
F17	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$2.50 \cdot 10^{5}$	$8.09 \cdot 10^{1}$	$1.38 \cdot 10^{5}$	$4.48 \cdot 10^{4}$	$4.48 \cdot 10^{4}$
Median	$2.89 \cdot 10^{5}$	$1.03 \cdot 10^{2}$	$1.59 \cdot 10^{5}$	$7.18 \cdot 10^{4}$	$6.19 \cdot 10^{4}$
Worst	$3.26 \cdot 10^{5}$	$1.33 \cdot 10^{2}$	$1.86 \cdot 10^{5}$	$9.65 \cdot 10^{4}$	$8.14 \cdot 10^{4}$
Mean	$2.87 \cdot 10^{5}$	$1.03 \cdot 10^{2}$	$1.59 \cdot 10^{5}$	$7.16 \cdot 10^{4}$	$6.21 \cdot 10^{4}$
Std	$1.98 \cdot 10^{4}$	$1.38 \cdot 10^{1}$	$1.43 \cdot 10^{4}$	$1.59 \cdot 10^{4}$	$9.15 \cdot 10^{3}$
F18	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$5.61 \cdot 10^{3}$	$8.37 \cdot 10^{2}$	$2.51 \cdot 10^{3}$	$1.46 \cdot 10^{3}$	$4.02 \cdot 10^{2}$
Median	$2.30 \cdot 10^{4}$	$1.08 \cdot 10^{3}$	$4.17 \cdot 10^{3}$	$2.39 \cdot 10^{3}$	$1.08 \cdot 10^{3}$
Worst	$4.71 \cdot 10^{4}$	$1.53 \cdot 10^{3}$	$1.62 \cdot 10^{4}$	$5.61 \cdot 10^{3}$	$1.78 \cdot 10^{3}$
Mean	$2.46 \cdot 10^{4}$	$1.08 \cdot 10^{3}$	$7.09 \cdot 10^{3}$	$2.73 \cdot 10^{3}$	$1.11 \cdot 10^{3}$
Std	$1.05 \cdot 10^{4}$	$1.61 \cdot 10^{2}$	$4.77 \cdot 10^{3}$	$1.04 \cdot 10^{3}$	$3.93 \cdot 10^{2}$
F19	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$1.02 \cdot 10^{6}$	$9.90 \cdot 10^{5}$	$1.21 \cdot 10^{6}$	$6.78 \cdot 10^{5}$	$3.14 \cdot 10^{1}$
Median	$1.11 \cdot 10^{6}$	$1.15 \cdot 10^{6}$	$1.36 \cdot 10^{6}$	$1.09 \cdot 10^{6}$	$2.95 \cdot 10^{2}$
Worst	$1.20 \cdot 10^{6}$	$1.23 \cdot 10^{6}$	$1.54 \cdot 10^{6}$	$1.58 \cdot 10^{6}$	$8.30 \cdot 10^{2}$
Mean	$1.11 \cdot 10^{6}$	$1.14 \cdot 10^{6}$	$1.36 \cdot 10^{6}$	$1.04 \cdot 10^{6}$	$3.30 \cdot 10^{2}$
Std	$5.15 \cdot 10^{4}$	$5.85 \cdot 10^{4}$	$7.35 \cdot 10^{4}$	$2.74 \cdot 10^{5}$	$2.03 \cdot 10^{2}$
F20	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Best	$3.59 \cdot 10^{3}$	$2.83 \cdot 10^{3}$	$1.70 \cdot 10^{3}$	$9.74 \cdot 10^{2}$	$5.03 \cdot 10^{2}$
Median	$3.98 \cdot 10^{3}$	$3.21 \cdot 10^{3}$	$2.04 \cdot 10^{3}$	$9.85 \cdot 10^{2}$	$8.88 \cdot 10^{2}$
Worst	$5.32 \cdot 10^{3}$	$6.23 \cdot 10^{3}$	$2.34 \cdot 10^{3}$	$1.04 \cdot 10^{3}$	$1.13 \cdot 10^{3}$
Mean	$4.06 \cdot 10^{3}$	$3.33 \cdot 10^{3}$	$2.05 \cdot 10^{3}$	$9.97 \cdot 10^{2}$	$8.42 \cdot 10^{2}$
Std	$3.66 \cdot 10^{2}$	$6.63 \cdot 10^{2}$	$1.80 \cdot 10^{2}$	$2.28 \cdot 10^{1}$	$1.76 \cdot 10^{2}$

Note: The shaded cell represents the best mean value for every problem. The results of DECC-G, DECC-G* and MLCC are taken from http://staff.ustc.edu.cn/~ketang/cec2012/lsgo_competition.htm while result of eSS is taken from original publication (Villaverde et al., 2012).

Table 9
Performance ranking for each of benchmark function (F10–F20).

	F10	F11	F12	F13	F14	F15	F16	F17	F18	F19	F20
DECC-G*	1	1	1	1	2	1	1	1	1	4	4
SSCOL	3	3	2	2	1	2	4	2	2	1	1
eSS	4	4	3	3	3	4	5	3	3	2	2
MLCC	2	5	4	4	4	3	3	4	4	5	3
DECC-G	5	2	5	5	5	5	2	5	5	3	5

Note: The ranking is based on the mean value for every function. The shaded rows represent the best overall result, since DECC-G* is for reference only (it used grouping structure).

Table 10
Ranking of each algorithm using Friedman test across all benchmark functions.

Algorithm	DECC-G	DECC-G*	MLCC	eSS	SSCOL
Rank	4.27	1.64	3.73	3.27	2.09

Note: The lowest rank value is the best method and highest value is the worst method based on Friedman test.

Table 11
Result of the Wilcoxon signed-rank test for all benchmark functions.

Wilcoxon test	R+	R-	*p*-value (Asymp. Sig.2-tailed)
SSCOL vs DECC-G	63.0	3.00	0.008
SSCOL vs DECC-G*	27.0	39.0	0.594
SSCOL vs MLCC	58.00	8.00	0.026
SSCOL vs eSS	66.00	0.00	0.003

Note: R+ represents positive rank (the best rank) of SSCOL over other methods. R^- represents the sum of ranks for the opposite ranks

algorithms for all benchmark functions tested (F10–F20). In this test, DECC-G* is the best overall method; however, since it uses grouping information (previous knowledge), this method is considered unfair, and it is included only for reference purpose. In Table 9, it can be noticed that SSCOL performs well compared to other state-of-the-art methods including the original eSS. To further investigate the conclusion obtained thus far, nonparametric statistical analysis is performed using SPSS software. First, the Friedman test (Friedman, 1937) according to mean rank is used as depicted in Table 10.

Overall, Table 10 shows that SSCOL ranks second best, eSS is third, MLCC is fourth and DECC-G is fifth. To further investigate this matter, paired Wilcoxon signed-rank test (Derrac et al., 2011) is employed. It is a nonparametric statistical test for pairwise comparisons. SSCOL represents individual performance and is used as the performance score for other methods. The null hypothesis, H_0, assumes that there is no statistical difference between the performance of two methods being compared, while the alternative hypothesis, H_1, assumes that there is significant statistical difference between the two methods, with a significance level of 0.05. From results shown in the Table 11, since

Table 13
Result of the Wilcoxon signed ranks test for CHO based on objective function value.

Wilcoxon test	R+	R-	*p*-value (Asymp. Sig.2-tailed)
SSCOL vs DE	210	0	$8.9{\cdot}10^{-5}$
SSCOL vs PSO	210	0	$8.9{\cdot}10^{-5}$
SSCOL vs eSS	131	79	0.332

Note: R^+ represents the sum of ranks (SSCOL outperformed the others) and R^- represents the sum of ranks for the opposite ranks. SSCOL shows significant difference compared to DE and PSO but not significant enough to reject H_0 when compared to ESS with level of significance $\alpha = 0.05$.

DECC-G* is considered as an unfair method, SSCOL performs well compared to other methods with all of its *p*-values are below 0.05 (Asymp. Sig.2-tailed). This test shows that there is significant evidence to reject H_0 and accept H_1. The value of ranks R^+ also indicates that SSCOL performs well compared to DECC-G, MLCC and eSS. Therefore, this test has 95% confidence that SSCOL's performance is statistically different compared to others. Thus, it can be concluded that the proposed algorithm is suitable not only for parameter estimation problem in biological systems, but also for large-scale global optimization problems.

5.3.2. Parameter estimation of Chinese hamster ovary (CHO) cells

Due to the stochastic nature of the problem, result for each algorithm varies with every run. Hence, we report the best, worst and average objective function values of the 20 runs. Table 12 depicts these three values as well as the standard deviation and average CPU time (seconds) consumed. From this table, DE has failed to achieve satisfactory solution where it obtained the largest value (53075) and also consumed largest average computational time (37055 s). Among all algorithms, PSO has the lowest CPU time (3413.6 s) and is considered as a good alternative algorithm for large-scale parameter estimation problem although it has the third lowest best objective function value (223.78). Overall, the table depicts that SSCOL obtained the lowest best value of 34.169 compared to eSS (36.705). SSCOL also recorded the most consistent and stable results, it obtained the lowest worst value (154.99) as well as average value (76.727) and lowest

Table 12
Comparative experimental results for CHO cells over 20 runs.

Algorithm	Best value	Worst value	Average value	Standard deviation	Average CPU time (seconds)
DE	$5.3075 \cdot 10^4$	$1.7121 \cdot 10^5$	$9.2626 \cdot 10^4$	$3.3562 \cdot 10^4$	$3.7055 \cdot 10^4$
PSO	$2.2378 \cdot 10^2$	$6.5946 \cdot 10^3$	$1.4175 \cdot 10^3$	$1.5119 \cdot 10^3$	$3.4136 \cdot 10^3$
eSS (rerun)	$3.6705 \cdot 10^1$	$2.0747 \cdot 10^2$	$9.5537 \cdot 10^1$	$5.2290 \cdot 10^1$	$8.7075 \cdot 10^3$
SSCOL	$3.4169 \cdot 10^1$	$1.5499 \cdot 10^2$	$7.6727 \cdot 10^1$	$4.1455 \cdot 10^1$	$6.6216 \cdot 10^3$

Note: The best objective function (nonlinear least squares) value is shown in shaded cell.

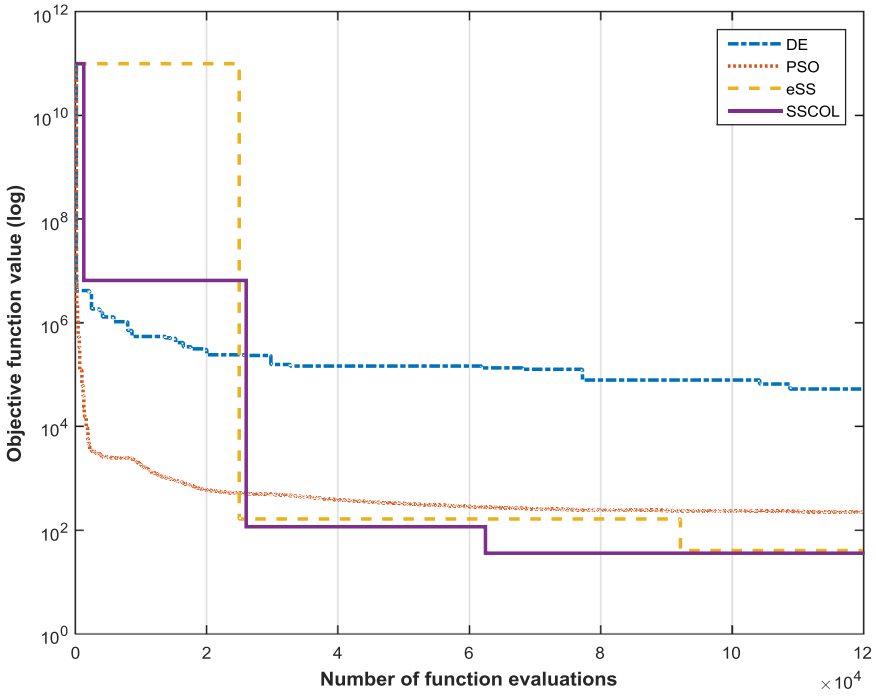

Fig. 2. Convergence curves of best run for CHO cells.

Table 14
Comparison of the best run between this work (SSCOL) and previous benchmark (ESS) for the case of CHO.

Algorithm	SSOL (this work)	ESS (Villaverde et al., 2015)
CPU time (s)	$6.7116 \cdot 10^3$	$\approx 3.600 \cdot 03$
J_f	$3.4169 \cdot 10^3$	$4.5718 \cdot 10^1$
J_{nom}	$3.9068 \cdot 10^1$	$3.9068 \cdot 10^1$
$\sum NRMSE_f$	2.8048	2.8010
$\sum NRMSE_{nom}$	2.8273	2.8273

Note: The best (minimum) number of function evaluations, the best (minimum) CPU time (s), the best (minimum) objective function (nonlinear least squares) values J_f and the best sum of normalized-root-mean-square-error $\Sigma NRMSE_f$ are indicated in shaded cells.
J_{nom}= objective function values obtained from nominal parameters p_{nom}.
$\Sigma NRMSE_{nom}$= Sum of normalized root-mean square-error with nominal parameters.

standard deviation (41.1455). Likewise, SSCOL has better average CPU time (6621.6 s) compared to other algorithms.

In order to be able to compare the results accurately, a pairwise comparison using Wilcoxon signed ranks test was conducted to test the significant difference of SSCOL compared to other algorithms. Table 13 shows the result of the test based on objective function value over 20 runs. SSCOL recorded better results compared to DE and PSO. This is indicated by the lowest p-value obtained (p-value < 0.05) with level of significance $\alpha = 0.05$. The null hypothesis H_0, where there is no statistically significant difference between SSCOL with both DE and PSO can be rejected and alternate hypothesis H_1, which is the opposite of H_0, can be accepted. However, p-value obtained for pairwise SSCOL against eSS is larger than 0.05. In this case, although SSCOL obtained larger sum of ranks R^+ value (131) over eSS, the H_0 can be accepted where there is no significant difference between SSCOL and eSS.

Additional information for this comparison is presented in Fig. 2. This figure shows the convergence curves for the best run of DE, PSO, eSS and SSCOL. The curves show how the value of the best solution (in log-scale) improved over number of function evaluations. It should be noticed that the starting values of all methods are the same due to the same initial kinetic parameter value (that is randomly generated within the specific range) used. Based on the result, it can be noted that SSCOL has better convergence speed. It can be observed that quasi-opposition based learning for the *RefSet* formation produced high quality value which leads to achieving faster minimum value when function evaluations reached approximately 3000 evaluations. Meanwhile, the combination and intensification mechanisms using quasi-reflection ensure that the whole search process for high quality parameters is effective, thus are able to accelerate convergence speed and reduce computational costs. Further details of all convergence curves (20 runs) can be found in Supplementary file (Convergence Results).

Additionally, the results obtained from this study are compared with previous benchmark (Villaverde et al., 2015) as shown in Table 14. It can be seen that this study obtained a better objective function value J_f (34.169) compared to the benchmark (45.718). In

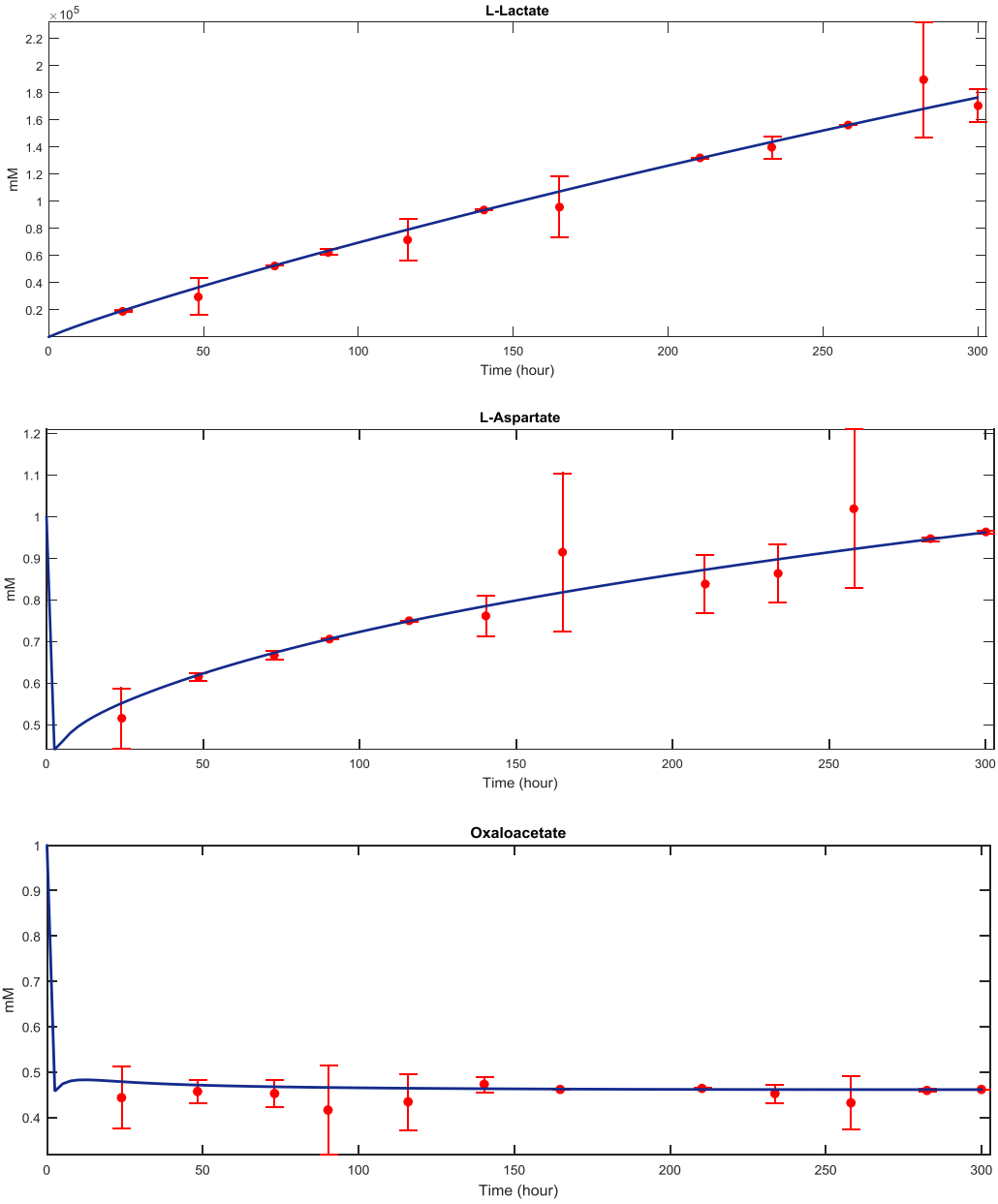

Fig. 3. Data fits for 3 metabolites in CHO cells: L-Lactate, L-Aspartate and Oxaloacetate. y-axis represents metabolites concentration in millimolar (mM), while x-axis denotes time taken for fermentation process in hour. The pseudo-experimental data with noise is denoted by the red bar. The model simulation with parameter estimated from this work is denoted by the solid blue line.

Table 15
Comparative experimental results for CCM over 20 runs.

Algorithm	Best value	Worst value	Average value	Standard deviation	Average CPU time (seconds)
DE	$4.1203 \cdot 10^2$	$6.1407 \cdot 10^2$	$5.1741 \cdot 10^2$	$4.5680 \cdot 10^1$	$5.2554 \cdot 10^4$
PSO	$2.3787 \cdot 10^2$	$4.1090 \cdot 10^3$	$1.1403 \cdot 10^3$	$1.5838 \cdot 10^3$	$2.5494 \cdot 10^4$
eSS (rerun)	$2.2918 \cdot 10^2$	$2.7007 \cdot 10^2$	$2.4843 \cdot 10^2$	$1.0465 \cdot 10^1$	$3.5085 \cdot 10^4$
SSCOL	$2.0992 \cdot 10^2$	$2.5642 \cdot 10^2$	$2.4107 \cdot 10^2$	$1.1891 \cdot 10^1$	$3.4536 \cdot 10^4$

Note: The best objective function (nonlinear least squares) value is shown in shaded cell.

terms of computational cost, specifically for the number of function evaluations, this work produced smaller evaluation (120,000 evaluations) than the previous benchmark (161,930 evaluations). It should be noted that the method employed in this study is able to reduce computational costs and is more efficient in estimating kinetic parameters. However, the benchmark has lower CPU time than this work due to the different hardware specifications and different stopping criteria used. In this regard, number of function evaluations can be used as a better stopping criterion when comparing different optimization algorithms using different platforms and hardware specifications.

Table 16
Result of the Wilcoxon signed ranks test for CCM based on objective function value.

Comparison	R^+	R^-	*p*-value (Asymp. Sig.2-tailed)
SSCOL vs DE	210	0	$8.9{\cdot}10^{-5}$
SSCOL vs PSO	190	20	0.002
SSCOL vs eSS (rerun)	165	45	0.025

Note: R^+ represents the sum of ranks (SSCOL outperformed the others) and R^- represents the sum of ranks for the opposite. SSCOL shows an improvement over DE, PSO and eSS with a level of significance $\alpha = 0.05$.

To assess the quality of model fit to the experimental data, cumulative sum of normalized-root-mean-square-error (∑NRMSE) is used. NRMSE is a standard measure for goodness of model fit which is defined as:

$$RMSE^O = \sqrt{\frac{\sum_{\epsilon}^{n_\epsilon} \sum_{s=1}^{n_s^{\epsilon,o}} (ym_s^{\epsilon,o} - y_s^{\epsilon,o}(p))^2}{n_\epsilon \cdot n_s^{\epsilon,o}}} \tag{17}$$

with the same notation as in Eq. (1). Apart from this, root-mean-square-error (RMSE) is used to measure the goodness of fit for every observed metabolite. Due to a different order of magnitude in metabolites concentrations, it is useful to normalize each $RMSE^O$ by dividing it with the range of observable values:

$$NRMSE^O = \frac{RMSE^O}{max\,(ym^{\epsilon,o}) - min\,(ym^{\epsilon,o})} \tag{18}$$

Cumulative ∑NRMSE is simply a sum of all observable $NRMSE^O$. Based on Table 8, it should be noticed that this work obtained similar ∑NRMSE value (2.8048) with previous benchmark, but lower $\sum NRMSE_{nom}$ value. In this data, the different J_f may lead to different *NRMSE* because its behavior is different due to lack of identifiability. This is caused by the different portion of kinetic parameters value that gives the same model prediction. This data also causes the objective function value obtained via optimization algorithm to be overfitting, which gives a better fit to pseudo-experimental data than the one obtained from nominal parameters that generated the data. Moreover, considering the presence of noise, the optimal objective function value does not only fits the systems dynamics, but also the noise itself which cannot be achieved by nominal parameters.

Finally, Fig. 3 shows the data fit of 3 metabolites which their kinetic parameter values were obtained in this work. For brevity, only three metabolites (L-Lactate, L-Aspartate and Oxaloacetate) are shown out of the total 13 metabolites. These chosen metabolites have very high nonlinear systems. The figure depicts the change of metabolites concentration over 300 h in the fermentation process, starting from initial concentrations at time 0. It can be seen that the concentration of L-Lactate has a higher order of magnitude than L-Aspartate and Oxaloacetate. The figure shows that this proposed work is able to obtain near optimal kinetic parameters that gave the best fit to experimental data with noise.

5.3.3. Parameter estimation of central carbon metabolism (CCM) of *E. coli*

Estimation of kinetic parameters for this model consumes larger computational costs (CPU time) compared to the previous one. This is because the model is more complex with many nonlinear metabolic processes involved. For example, one run of each algorithm takes approximately more than six hours. Table 15 depicts the experimental results of DE, PSO, eSS (rerun) and SSCOL for the CCM of *E. coli* case. The result reports the best, worst and average values obtained as well as their standard deviation and average CPU time over 20 runs. DE which is the most recommended algorithm for parameter estimation problem again failed to achieve the best result and it also consumed the largest computational cost (CPU time=52554 s). However, its average value (517.41) is lower than PSO which indicated that DE has better ability in avoiding local minima compared to PSO. Overall, the result also revealed that the proposed algorithm (SSCOL) managed to obtain better results with the best minimum and average values of the objective function (209.922 and 241.07). Although eSS has a slightly better standard deviation, SSCOL is still the best algorithm because it consumes the lowest average CPU time.

To test the significant difference of SSCOL's performance over other algorithms, same procedure was conducted using the Wilcoxon signed ranks test. A pairwise comparison over 20 runs of SSCOL with DE, PSO and eSS employed based on the objective function value is shown in

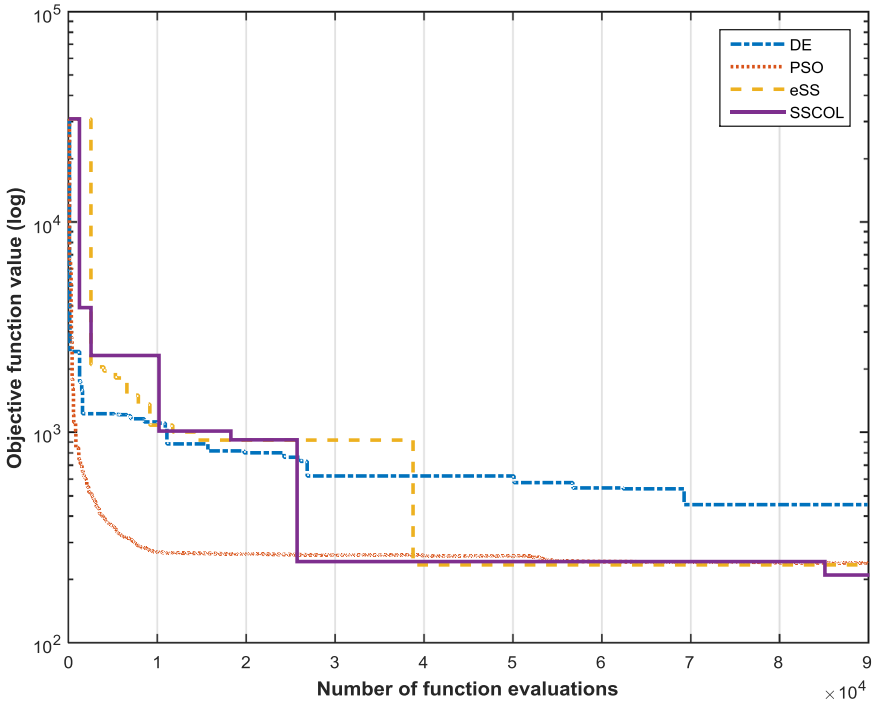

Fig. 4. Convergence curves of best run for CCM.

Table 17
Comparison of the best run between this algorithm (SSCOL) with previous benchmark (ESS) for CCM.

Algorithm	SSCOL (this work)	ESS (Villaverde et al., 2015)
CPU time (s)	$2.7605 \cdot 10^4$	$\approx 1.0800 \cdot 10^4$
J_f	$2.0992 \cdot 10^2$	$2.3390 \cdot 10^2$
J_{ex}	$3.1087 \cdot 10^4$	$3.1087 \cdot 10^4$
$\sum NRMSE_f$	2.3879	2.4921
$\sum NRMSE_{ex}$	3.6065	3.6065

Note: The best (minimum) number of function evaluations, the best (minimum) CPU time (s), the best (minimum) objective function (nonlinear least squares) values J_f and the best sum of normalized-root-mean-square-error $\Sigma NRMSE_f$ are indicated in shaded cells.
J_{ex}= Objective function values obtained from original parameters reported in publication.
$\Sigma NRMSE_{ex}$= Sum of normalized root-mean-square-error with original parameters reported in publication.

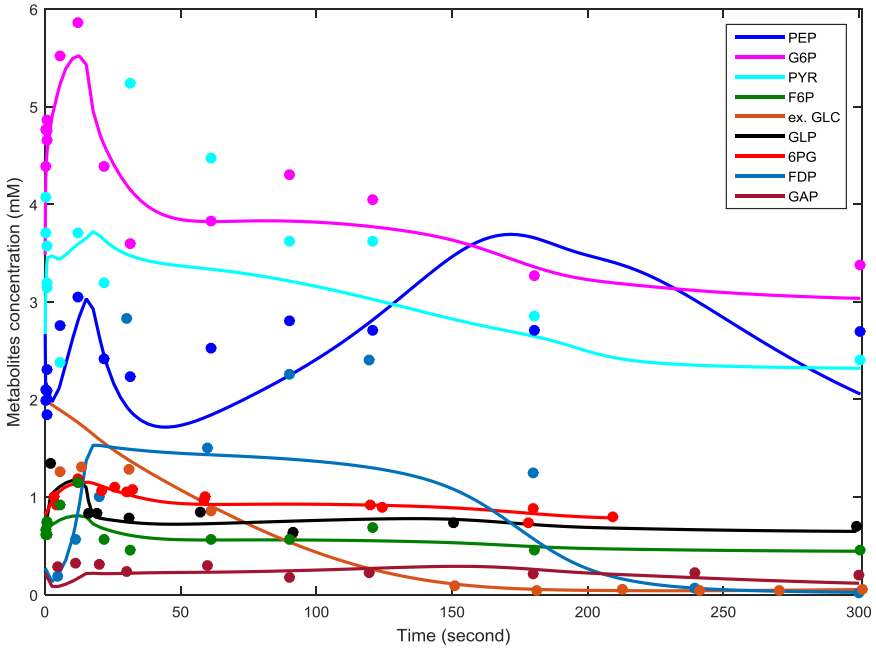

Fig. 5. Data fits for 9 metabolites in CCM. The experimental data with real noise is denoted as circles. The model simulation with parameter estimated from this work is denoted in solid lines. The x-axis represents time taken for fermentation process in unit of seconds, while the y-axis represents metabolites concentration (in millimolar units, mM).

Table 16. The table shows that SSCOL is significantly better than eSS and performed well compared to other algorithms. This is indicated by the lowest p-value obtained (p-value < 0.05) for all pairwise comparisons with level of significance $\alpha = 0.05$. The null hypothesis H_0, where there is no statistically significant difference between SSCOL and other algorithms can be rejected and alternate hypothesis H_1, the opposite of $H0$, can be accepted.

Additional information in terms of objective functions' progression speed over number of function evaluations is shown in Fig. 4. In this figure, the graphs plot convergence curves of the best run among 20 runs of DE, PSO, eSS(rerun) and SSCOL. It can be noted that PSO produced high speed of convergence in the early stage of evaluation, approximately before 25,000 evaluations. However after 85,000 evaluations, SSCOL produced the best objective value followed by eSS, PSO and DE. The graph also shows that DE has very low convergence rate compared to others. It can be concluded that, similar to the CHO case, algorithm used in this study is capable of accelerating speed of convergence using minimum computational cost. Further details of the all convergence curves (20 runs) can be found in Supplementary file (Convergence Results).

Table 17 depicts a comparison made between this work and the benchmark. The table shows that this work produced better results in terms of lower number of function evaluations (since we used only 90,000 evaluations), lower value of objective function J_f and lower value of $\sum NRMSE_f$. Since the benchmark uses different stopping criteria (CPU time), its value is relatively lower than the one obtained in this work. As mentioned earlier, the benchmark also uses different hardware specifications. From the comparison, it can be noticed that this study's $\sum NRMSE_f$ value is the lowest compared to the benchmark and original $\sum NRMSE_{ex}$, which means that the parameters estimated by this study may provide the best possible fit to the real experimental data. To illustrate this, Fig. 5 portrays the data fits for 9 metabolites using parameter estimates from this work. The fits showed perfect match especially for PEP, G6P and PYR metabolites. It can be concluded that this work does not only reduced computational efforts, but also produced the best model in terms of accuracy.

6. Conclusion

This paper considered parameter estimation problem for large-

scale kinetic models of biochemical systems from the context of metabolic engineering and cell factories applications. It is difficult to solve this problem due to highly nonlinear nature of the biological systems, existence of noise in the experimental data and large numbers of kinetic parameters. Due to the high dimension problem of the study, the use of gradient-based (local search) method to address this problem can cause premature convergence or stuck in local minima, while the use of metaheuristic (global search) method can result in excessive computational costs and slow convergence rates.

In order to surmount these difficulties, this study proposed an enhanced scatter search with combined opposition-based learning (SSCOL) to solve large-scale global optimization and parameter estimation problem. SSCOL is an improved version of enhanced scatter search (eSS) that introduced quasi-opposition based learning in the reference set (*Refset*) formation to obtain initial high-quality value that is close to the global minimum. Moreover, a combination of *RefSet* members has been employed using quasi-reflection to reduce the search region. In addition, this study also introduced quasi-reflection with jumping probability in performing specific intensification method. These new improvements increased the efficiency of the global optimization and parameter estimation significantly while maintaining its robustness. Large-scale global benchmark functions and two kinetic models of biochemical systems have been used to extensively investigate the performance of the proposed algorithm. Experimental results showed that the performance of SSCOL algorithm proposed in this study is superior in finding the near optimal value of hundreds of kinetic parameters. The results also revealed that SSCOL was able to achieve the best (minimum) solution in two datasets with minimum computational efforts and faster convergence rate compared to other algorithms. The main contribution of this research is the coming up with a proposed algorithm for solving large-scale optimization problem. The application presented in this study is not only limited to the bioinformatics/metabolic engineering field, but also to real world global optimization problem, as well as other engineering areas and real world applications.

Acknowledgement

We would like to thank Malaysian Ministry of Higher Education and Universiti Teknologi Malaysia for supporting this research by a Fundamental Research Grant Scheme (grant number: R.J130000.7828.4F886) and a GUP Tier 1 Research Grant (grant number: Q.J130000.2528.11H05).

Appendix A. Supporting information

Supplementary data associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.engappai.2017.04.004.

References

Alfi, A., Fateh, M.M., 2011a. Identification of nonlinear systems using modified particle swarm optimisation: a hydraulic suspension system. Veh. Syst. Dyn. 49, 871–887. http://dx.doi.org/10.1080/00423114.2010.497842.

Alfi, A., Fateh, M.M., 2011b. Intelligent identification and control using improved fuzzy particle swarm optimization. Appl. Math. Model. 38, 12312–12317. http://dx.doi.org/10.1016/j.apm.2010.08.008.

Alfi, A., Modares, H., 2011. System identification and control using adaptive particle swarm optimization. Appl. Math. Model. 35, 1210–1221. http://dx.doi.org/10.1016/j.apm.2010.08.008.

Almquist, J., Cvijovic, M., Hatzimanikatis, V., Nielsen, J., Jirstrand, M., 2014. Kinetic models in industrial biotechnology - Improving cell factory performance. Metab. Eng.. http://dx.doi.org/10.1016/j.ymben.2014.03.007.

Awotunde, A.A., 2015. Estimation of well test parameters using global optimization techniques. J. Pet. Sci. Eng. 125, 269–277. http://dx.doi.org/10.1016/j.petrol.2014.11.033.

Blum, C., Roli, A., 2003. Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput. Surv. 35, 189–213. http://dx.doi.org/10.1007/s10479-005-3971-7.

César Trejo Zúñiga, E., López Cruz, I.L., García, A.R., 2014. Parameter estimation for crop growth model using evolutionary and bio-inspired algorithms. Appl. Soft Comput. 23, 474–482. http://dx.doi.org/10.1016/j.asoc.2014.06.023.

Chassagnole, C., Noisommit-Rizzi, N., Schmid, J.W., Mauch, K., Reuss, M., 2002. Dynamic modeling of the central carbon metabolism of Escherichia coli. Biotechnol. Bioeng. 79, 53–73. http://dx.doi.org/10.1002/bit.10288.

Copeland, W.B., Bartley, B.A., Chandran, D., Galdzicki, M., Kim, K.H., Sleight, S.C., Maranas, C.D., Sauro, H.M., 2012. Computational tools for metabolic engineering. Metab. Eng. 14, 270–280. http://dx.doi.org/10.1016/j.ymben.2012.03.001.

Črepinšek, M., Liu, S.H., Mernik, L., Mernik, M., 2016. Is a comparison of results meaningful from the inexact replications of computational experiments? Soft Comput. 20, 223–235. http://dx.doi.org/10.1007/s00500-014-1493-4.

Cvijovic, M., Bordel, S., Nielsen, J., 2011. Mathematical models of cell factories: moving towards the core of industrial biotechnology. Microb. Biotechnol. 4, 572–584. http://dx.doi.org/10.1111/j.1751-7915.2010.00233.x.

Da Ros, S., Colusso, G., Weschenfelder, T.A., de Marsillac Terra, L., de Castilhos, F., Corazza, M.L., Schwaab, M., 2013. A comparison among stochastic optimization algorithms for parameter estimation of biochemical kinetic models. Appl. Soft Comput. 13, 2205–2214. http://dx.doi.org/10.1016/j.asoc.2013.01.019.

Dai, Z., Lai, L., 2014. Differential simulated annealing: a robust and efficient global optimization algorithm for parameter estimation of biological networks. Mol. Biosyst. 10, 1385–1392. http://dx.doi.org/10.1039/c4mb00100a.

Darabi, A., Alfi, A., Kiumarsi, B., Modares, H., 2012. Employing adaptive particle swarm optimization algorithm for parameter estimation of an exciter machine. J. Dyn. Syst. Meas. Control 134, 11013. http://dx.doi.org/10.1115/1.4005371.

Derrac, J., García, S., Molina, D., Herrera, F., 2011. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm Evol. Comput. 1, 3–18. http://dx.doi.org/10.1016/j.swevo.2011.02.002.

Eberhart, R., Kennedy, J., 1995. A new optimizer using particle swarm theory. Proc. Sixth Int. Symp. Micro Mach. Hum. Sci. 39–43. ⟨http://dx.doi.org/10.1109/MHS.1995.494215⟩.

Egea, J.A., Rodríguez-Fernández, M., Banga, J.R., Martí, R., 2007. Scatter search for chemical and bio-process optimization. J. Glob. Optim. 37, 481–503. http://dx.doi.org/10.1007/s10898-006-9075-3.

Egea, J.A., Martí, R., Banga, J.R., 2010. An evolutionary method for complex-process optimization. Comput. Oper. Res. 37, 315–324. http://dx.doi.org/10.1016/j.cor.2009.05.003.

Egea, J.A., Vazquez, E., Banga, J.R., Martí, R., 2009. Improved scatter search for the global optimization of computationally expensive dynamic models. J. Glob. Optim. 43, 175–190. http://dx.doi.org/10.1007/s10898-007-9172-y.

Egea, J.A., Henriques, D., Cokelaer, T., Villaverde, A.F., MacNamara, A., Danciu, D., Banga, J.R., Saez-Rodriguez, J., 2014. MEIGO: an open-source software suite based on metaheuristics for global optimization in systems biology and bioinformatics. BMC Bioinform. 15, 136. http://dx.doi.org/10.1186/1471-2105-15-136.

Egea, J., Balsa-Canto, E., 2009. Dynamic optimization of nonlinear processes with an enhanced scatter search method. Ind. Eng. Chem. Res. 48, 4388–4401.

Ergezer, M., Simon, D., 2015. Probabilistic properties of fitness-based quasi-reflection in evolutionary algorithms. Comput. Oper. Res. 63, 114–124. http://dx.doi.org/10.1016/j.cor.2015.03.013.

Ergezer, M., Simon, D., 2014. Mathematical and experimental analyses of oppositional algorithms. IEEE Trans. Cybern. 44, 2178–2189. http://dx.doi.org/10.1109/TCYB.2014.2303117.

Friedman, M., 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. J. Am. Stat. Assoc. 32, 675–701. http://dx.doi.org/10.1080/01621459.1937.10503522.

Gábor, A., Banga, J.R., 2015. Robust and efficient parameter estimation in dynamic models of biological systems. BMC Syst. Biol. 9, 74. http://dx.doi.org/10.1186/s12918-015-0219-2.

Glover, F., 1998. A template for scatter search and path relinking. Artif. Evol. 1363, 3–51 . http://dx.doi.org/10.1016/0042-207X(68)92558-X.

Glover, F., 1977. Heuristics for integer programming using surrogate constraints. Decis. Sci. 8, 156–166. http://dx.doi.org/10.1111/j.1540-5915.1977.tb01074.x.

Kazimipour, B., Li, X., Qin, A.K., 2013. Initialization methods for large scale global optimization. 2013 IEEE Congr. Evol. Comput. 2750–2757. ⟨http://dx.doi.org/10.1109/CEC.2013.6557902⟩.

Keasling, J.D., 2012. Synthetic biology and the development of tools for metabolic engineering. Metab. Eng. 14, 189–195. http://dx.doi.org/10.1016/j.ymben.2012.01.004.

Liu, D., Hoynes-O'Connor, A., Zhang, F., 2013a. Bridging the gap between systems biology and synthetic biology. Front. Microbiol.. http://dx.doi.org/10.3389/fmicb.2013.00211.

Liu, S.H., Mernik, M., Hrnčič, D., Črepinšek, M., 2013b. A parameter control method of evolutionary algorithms using exploration and exploitation measures with a practical application for fitting Sovova's mass transfer model. Appl. Soft Comput. J. 13, 3792–3805. http://dx.doi.org/10.1016/j.asoc.2013.05.010.

Mendes, P., Kell, D., 1998. Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation. Bioinformatics 14, 869–883. http://dx.doi.org/10.1093/bioinformatics/14.10.869.

Mernik, M., Liu, S.H., Karaboga, D., Črepinšek, M., 2015. On clarifying misconceptions when comparing variants of the Artificial Bee Colony Algorithm by offering a new implementation. Inf. Sci. 291, 115–127. http://dx.doi.org/10.1016/j.ins.2014.08.040.

Moles, C.G., Mendes, P., Banga, J.R., 2003. Parameter estimation in biochemical pathways: a comparison of global optimization methods. Genome Res. 13, 2467–2474. http://dx.doi.org/10.1101/gr.1262503.

Perez-ramirez, C.A., Amezquita-sanchez, J.P., Adeli, H., Valtierra-rodriguez, M.,

Camarena-martinez, D., Romero-troncoso, R.J., 2016. New methodology for modal parameters identification of smart civil structures using ambient vibrations and synchrosqueezed wavelet transform. Eng. Appl. Artif. Intell. 48, 1–12. http://dx.doi.org/10.1016/j.engappai.2015.10.005.

Rahnamayan, S., Tizhoosh, H.R., Salama, M.M., 2008. Opposition-based differential evolution. Stud. Comput. Intell. 143, 155–171. http://dx.doi.org/10.1007/978-3-540-68830-3_6.

Rahnamayan, S., Tizhoosh, H.R., Salama, M.M.A., 2007. Quasi-oppositional differential evolution. 2007 IEEE Congr. Evol. Comput. CEC 2007 2229–2236. ⟨http://dx.doi.org/10.1109/CEC.2007.4424748⟩.

Rodriguez-Fernandez, M., Egea, J.A., Banga, J.R., 2006. Novel metaheuristic for parameter estimation in nonlinear dynamic biological systems. BMC Bioinform. 7, 483. http://dx.doi.org/10.1186/1471-2105-7-483.

Seif, Z., Ahmadi, M.B., 2015. An opposition-based algorithm for function optimization. Eng. Appl. Artif. Intell. 37, 293–306.

Smallbone, K., Messiha, H.L., Carroll, K.M., Winder, C.L., Malys, N., Dunn, W.B., Murabito, E., Swainston, N., Dada, J.O., Khan, F., Pir, P., Simeonidis, E., Spasić, I., Wishart, J., Weichart, D., Hayes, N.W., Jameson, D., Broomhead, D.S., Oliver, S.G., Gaskell, S.J., McCarthy, J.E.G., Paton, N.W., Westerhoff, H.V., Kell, D.B., Mendes, P., 2013. A model of yeast glycolysis based on a consistent kinetic characterisation of all its enzymes. FEBS Lett. 587, 2832–2841. http://dx.doi.org/10.1016/j.febslet.2013.06.043.

Storn, R., Price, K., 1997. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. J. Glob. Optim. 11, 341–359. http://dx.doi.org/10.1023/A:1008202821328.

Tashkova, K., Šilc, J., Atanasova, N., Džeroski, S., 2012. Parameter estimation in a nonlinear dynamic model of an aquatic ecosystem with meta-heuristic optimization. Ecol. Modell. 226, 36–61. http://dx.doi.org/10.1016/j.ecolmodel.2011.11.029.

Tizhoosh, H.R., 2005. Opposition-Based Learning: A New Scheme for Machine Intelligence. Comput. Intell. Model. Control Autom. 2005 Int. Conf. Intell. Agents, Web Technol. Internet Commer. Int. Conf. 1, 695–701. ⟨http://dx.doi.org/10.1109/CIMCA.2005.1631345⟩.

Villaverde, A.F., Bongard, S., Mauch, K., Müller, D., Balsa-Canto, E., Schmid, J., Banga, J.R., 2014. High-confidence predictions in systems biology dynamic models. Adv. Intell. Syst. Comput. 294, 161–171. http://dx.doi.org/10.1007/978-3-319-07581-5_20.

Villaverde, A.F., Egea, J.A., Banga, J.R., 2012. A cooperative strategy for parameter estimation in large scale systems biology models. BMC Syst. Biol. 6, 75. http://dx.doi.org/10.1186/1752-0509-6-75.

Villaverde, A.F., Henriques, D., Smallbone, K., Bongard, S., Schmid, J., Cicin-Sain, D., Crombach, A., Saez-Rodriguez, J., Mauch, K., Balsa-Canto, E., Mendes, P., Jaeger, J., Banga, J.R., 2015. BioPreDyn-bench: a suite of benchmark problems for dynamic modelling in systems biology. BMC Syst. Biol. 9, 1–15. http://dx.doi.org/10.1186/s12918-015-0144-4.

Wurm, F.M., 2004. Production of recombinant protein therapeutics in cultivated mammalian cells. Nat. Biotechnol. 22, 1393–1398. http://dx.doi.org/10.1038/nbt1026.

Xu, Q., Wang, L., Wang, N., Hei, X., Zhao, L., 2014. A review of opposition-based learning from 2005 to 2012. Eng. Appl. Artif. Intell. 29, 1–12. http://dx.doi.org/10.1016/j.engappai.2013.12.004.

Yang, Z., Tang, K., Yao, X., 2008a. Large scale evolutionary optimization using cooperative coevolution. Inf. Sci. 178, 2985–2999. http://dx.doi.org/10.1016/j.ins.2008.02.017.

Yang, Z., Tang, K., Yao, X., 2008b. Multilevel cooperative coevolution for large scale optimization. 2008 IEEE Congr. Evol. Comput. CEC 2008 1663–1670. ⟨http://dx.doi.org/10.1109/CEC.2008.4631014⟩.