A New Competitive Implementation of the Electromagnetism-like Algorithm for Global Optimization

Ana Maria A.C. Rocha, Andreia Silva, and Jorge Gustavo Rocha

Algoritmi Research Centre,
University of Minho, 4710-057 Braga, Portugal
arocha@dps.uminho.pt, andreia56837@gmail.pt, jgr@di.uminho.pt

Abstract. The Electromagnetism-like (EM) algorithm is a population-based stochastic global optimization algorithm that uses an attraction-repulsion mechanism to move sample points towards the optimal. In this paper, an implementation of the EM algorithm in the Matlab environment as a useful function for practitioners and for those who want to experiment a new global optimization solver is proposed. A set of benchmark problems are solved in order to evaluate the performance of the implemented method when compared with other stochastic methods available in the Matlab environment. The results confirm that our implementation is a competitive alternative both in term of numerical results and performance. Finally, a case study based on a parameter estimation problem of a biology system shows that the EM implementation could be applied with promising results in the control optimization area.

Keywords: global optimization; unconstrained minimization; Matlab environment; Electromagnetism-like algorithm; derivative-free algorithm

1 Introduction

Many real life global optimization problems that arise in areas such as physics, chemistry, and molecular biology, involve multimodal and non-differentiable nonlinear functions of many variables that are difficult to handle by gradient-based algorithms. As a result, many researchers have devoted themselves in finding reliable stochastic global optimization methods that do not require any derivative computation.

The Electromagnetism-like (EM) algorithm, developed by Birbil and Fang [3,4], is a population-based stochastic search method for global optimization that mimics the behavior of electrically charged particles. The method uses an attraction-repulsion mechanism to move a population of points towards optimality. The EM algorithm is specifically designed for solving bound constrained problems in the form:

$$\begin{aligned} &\min\ f(x) \\ &\text{s. t.}\ \ x \in \Omega, \end{aligned} \tag{1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a nonlinear continuous function and $\Omega = \{x \in \mathbb{R}^n : lb \leq x \leq ub\}$ is a bounded feasible region. We do not assume that the objective function f is convex and may possess many local minima in the feasible region.

The goal of global optimization is to find the globally best solution of problems, in the presence of multiple local optima. There are several stochastic search methods that were developed to find the global minimum such as the Simulated Annealing, Genetic Algorithm, Particle Swarm Optimization, Ant Colony Optimization and Evolutionary Methods [6,10,11,13]. Some of these methods may combine the search process with local refinements like random search methods or gradient-based methods.

Systems biology has been responsible for a revolution in the ability of understanding biologic events and organisms. It made possible to have an comprehensive, quantitative and temporal analysis of the interaction between the components of a biological system. Through a mathematical model it is possible to summarize the knowledge of a biological system allowing to make experimentally verifiable predictions. In systems biology, the determination of parameters is a central challenge, because the majority of the mathematical models present some characteristics that make the problem difficult to solve, as the large number of parameters to be estimated as the highly non-linearity of the problems [14,16,28,29,30]. Hence, there is a need for global optimization methods capable of solving this kind of problems efficiently.

In the Matlab environment, the Global Optimization Toolbox [19] provides methods that search for global solutions. It includes global search, multistart, pattern search, genetic algorithm, and simulated annealing solvers. These methods use interactive tools for defining and solving optimization problems and monitoring the solution progress.

The aim of this paper is to present a new implementation of the EM algorithm in the Matlab® environment. To accomplish this assessment, we started by porting the algorithm to Matlab programming language and a set of functions were defined in order to develop the EM with a similar syntax to the methods available in the Matlab Global Optimization Toolbox. Tests were performed against a well known set of problems, and some benchmarks were performed to validate this implementation. With a stable implementation in Matlab, we select a well known problem to assess the application of this algorithm to estimate biological parameters.

This paper is organized as follows. Section 2 gives a general overview of the EM algorithm. The implementation details, functionalities and how to use EM in the Matlab environment are described in Section 3. Section 4 reports the results of the numerical experiments carried out with a benchmark set of unconstrained problems, as well as a comparison with other stochastic based methods. The parameter estimation problem, called α-pinene, is briefly described and solved by EM in Section 4.2. It is compared against other stochastic methods. Finally, the paper is concluded in Section 5 and some remarks for future developments are presented.

2 Electromagnetism-like Algorithm

The EM algorithm simulates the electromagnetism theory of physics by considering each point in the population as an electrical charge that is released to the space. The charge of each point is related to the objective function value and determines the magnitude of attraction of the point over the others in the population. The better the objective function value, the higher the magnitude of attraction. The charges are used to find a direction for each point to move in subsequent iterations. The regions that have higher attraction will signal other points to move towards them. In addition, a repulsion mechanism is also introduced to explore new regions for even better solutions [3,4].

The EM algorithm is described in Algorithm 1 and comprises four main procedures: initialization of the algorithm, computation of the total force, movement along the direction of the force and a simple random line search algorithm [22,23].

Algorithm 1 EM algorithm

```
1: Initialization;
2: while stopping criteria are not met do
3:    Compute Force
4:    Move Points
5:    Local Search
6: end while
```

The "Initialization" procedure starts by randomly generating a sample of m points. Each point is uniformly distributed between the lower and upper bounds. Then, for each point of the population the objective function value is calculated. Finally the point which yields the least objective function value, denoted by the best point of the population, x^{best}, is identified as well as its corresponding objective function value f^{best}.

In the "Compute Force" procedure, each particle charge is calculated by

$$q^i = \exp\left(-n\frac{f(x^i) - f(x^{best})}{\sum_{k=1}^{m}(f(x^k) - f(x^{best}))}\right), \quad i = 1, \ldots, m. \tag{2}$$

that determines the power of attraction or repulsion for the point x^i. In this way the points that have better objective function values possess higher charges. After the charge calculation, the total force exerted on each point x^i is calculated by adding the individual component forces, F_j^i, between any pair of points x^i and x^j. According to the electromagnetism theory, the total force, F^i, is inversely proportional to the square of the distance between the points and directly proportional to the product of their charges:

$$F^i = \sum_{j \neq i}^{m} F_j^i = \begin{cases} (x^j - x^i)\frac{q^i q^j}{\|x^j - x^i\|^2} & \text{if } f(x^j) < f(x^i) \text{ (attraction)} \\ (x^i - x^j)\frac{q^i q^j}{\|x^j - x^i\|^2} & \text{if } f(x^j) \geq f(x^i) \text{ (repulsion)} \end{cases}, i = 1, \ldots, m. \tag{3}$$

The "Move Points" procedure uses the total force vector, F^i, to move the point x^i in the direction of the force by a random step length λ. The best point, x^{best}, is not moved. To maintain feasibility, the force exerted on each point is normalized and scaled by the allowed range of movement towards the lower bound or the upper bound of the set Ω, for each coordinate. Thus, for $i = 1, \ldots, m$ and $i \neq best$

$$x^i = \begin{cases} x^i + \lambda \frac{F^i}{\|F^i\|}(ub - x^i) \text{ if } F^i > 0 \\ x^i + \lambda \frac{F^i}{\|F^i\|}(x^i - lb) \text{ otherwise} \end{cases}. \tag{4}$$

The random step length λ is assumed to be uniformly distributed in $[0, 1]$.

Finally, the "Local search" procedure performs a local refinement around a point or around each point of the population [3,4]. In this implementation, this procedure is only applied to the best point of the population, since previous works showed it was sufficient and efficient in the improvement of the accuracy of EM [22,23,24]. This procedure implements a simple random line search algorithm, using the maximum feasible step length

$$\Delta = \delta_{local}(\max[ub - lb]) \tag{5}$$

with $\delta_{local} > 0$, to guarantee that the local search always generates feasible points [1,3,22,23,24]. A random movement of length Δ is carried out and if a better position is obtained within a maximum number of local iterations, x^{best} is updated.

3 MATLAB Implementation

Matlab® is a technical computing environment that integrates numerical and matrix analysis which has a simplistic environment to the user. Nowadays, Matlab is the standard tool that supports introductory and advanced mathematical based courses which is the choice tool used in industrial research.

3.1 Implementation Details

The original version of the Electromagnetism-like algorithm discussed on Section 2, was implemented as a Matlab global optimization solver. The implementation respects the structure of other optimization solvers where each module defines a specific routine of the algorithm.

The user can see the details of the `emalgorithm` using the following command.

```
>> help emalgorithm
```

The basic call of EM algorithm is:

```
>> [x,fval,exitflag,output]=emalgorithm(Fun,lb,ub,options)
```

The input and output arguments are described next.

Output arguments The output variables are `x`, as the minimizer, `fval` is the value of the objective function at the solution `x`, `exitflag` is a flag describing the exit condition and `output` is a structure that shows some values about the iterative process and other information. The possible values of `exitflag` and the corresponding exit conditions are detailed in Table 1.

Table 1. Exitflag conditions.

Flag	Condition
1	Average change in value of the objective function over `options.StallIterLimit` iterations less than `options.TolFun`
5	`options.ObjectiveLimit` limit reached
0	Maximum number of function evaluations or iteration exceeded
-1	Optimization terminated by the output or plot function
-2	No feasible point found
-5	Time limit exceeded

The description of the items about the iterative process concerning the `output` argument are presented in Table 2.

Table 2. Output structure.

Item	Description
problemtype	Type of problem: bound constrained
iterations	Total number of iterations
funccount	Total number of function evaluations
message	Termination message of the solver
totaltime	Time taken by the solver

Input arguments The input variables are `Fun`, which is the objective function, `lb` and `ub` are the lower and upper bounds on the variables, respectively, and `options` (optional) define parameters used to control the algorithm.

To know the list of the fields in the options structure as well as the valid parameters and the default parameters use:

```
>> help emoptimset
```

The available `options` are presented in Table 3 and can be defined with:

```
>> emoptimset(option, value)
```

where `option` should be a valid term and `value` is the assigned value of the option. To check the default option values of the EM algorithm, use:

```
>> emoptimset
```

Table 3. EM algorithm optional parameters.

Option	Type	Default value	Description
PopulationSize	positive scalar	10*numberOfVariables	Defines the population size
TolFun	non-negative scalar	1e-6	Termination tolerance on function value
StallIterLimit	positive integer	500*numberOfVariables	Number of iterations over which average change in objective function value at current point is less than `TolFun`
MaxFunEvals	positive integer	3000* numberOfVariables	Maximum number of function evaluations
MaxIter	positive integer	Inf	Maximum number of allowed iterations
MaxLocalIterations	positive integer	5	Maximum number of iterations allowed in the local search procedure
TimeLimit	positive scalar	Inf	Total time (in seconds) allowed for optimization
ObjectiveLimit	scalar	-Inf	Minimum objective function value desired
Delta	positive scalar	1e-3	Positive scalar that defines the local refinement (δ_{local})
PrecisionTolerance	non-negative scalar	1e-3	
Display		final	Controls the level of display results, valid values, `final`, `iter`, `diagnose`, and `off`
DisplayInterval	positive integer	10	Interval for iterative display
PlotFcns	function_handle		Plot function(s) to be called during iterations (`@emplotbestf`, `@emplotbestx`, `@emplotstopping`, `@emplotmeanf`)
PlotInterval	positive integer	1	Interval at which `PlotFcns` are called

3.2 Example Usage

Suppose we want to run the EM to solve the Branin function [12]:

$$\min \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + 5\frac{x_1}{\pi} - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos(x_1) + 10$$

$$\text{s. t. } -5 \leq x_1 \leq 0,$$

$$10 \leq x_2 \leq 15$$

Firstly, the function to be optimized should be created:

```
function y = branin(x)
% Branin function
y = (x(2)-(5.1/(4*pi^2))*x(1)^2+5*x(1)/pi-6)^2+...
  10*(1-1/(8*pi))*cos(x(1))+10;
end
```

Then, the following commands should be done in order to define the lower and upper bounds of the problem, define some options to control the algorithm, if desired, and finally call the EM algorithm:

```
lb = [-5 0];
ub = [10 15];
options=emoptimset('Display','iter');
[x,f,exitflag,output]=emalgorithm('branin',lb,ub,options)
```

The obtained solution is:

```
Iteration   f-count     f(x)
     10    256        0.397899
     20    469        0.397887
     30    679        0.397887
     40    889        0.397887
            ...
    270   5720        0.397887
    280   5930        0.397887
Maximum number of function evaluations exceeded: increase
options.MaxFunEvals.

x = 3.1415    2.2750
f =  0.3979
exitflag =  0
output =
     iterations: 284
      funccount: 6014
        message: 'Maximum number of function evaluations exceeded'
      totaltime: 1.3600
    problemtype: 'boundconstraints'
```

4 Numerical Results

Computational tests were performed on a PC with a 2.94GHz Pentium IV microprocessor and 4Gb of memory. At first, we compare the performance of EM algorithm with Genetic Algorithm (GA) and Simulated Annealing (SA) when applied to a benchmark set of nine functions. They are all stochastic and derivative-free methods available in the Global optimization Toolbox of Matlab. The GA is a method based on natural selection, selecting, at each step, individuals at random from the current population to be parents and uses them to produce the children for the next generation [10]. The SA mimics the physical process of heating a material and then slowly lowering the temperature to decrease defects in order to minimize the energy optimization problem [11]. GA and EM are population-based methods, while SA is a single point method. Secondly, we compare the EM method with other stochastic methods when solving a parameter estimation problem, the isomerization of α-pinene problem.

4.1 Application to a Test Set of Problems

In order to evaluate the efficiency of the implemented algorithm we considered nine test functions [12], summarized in Table 4, as benchmarks for comparing global search methods. The columns of the table refer to the name of the problem, 'Prob.'; information about the abbreviation usually used, 'Abbr.'; the dimension of the problem, 'n'; the known global minimum available in the literature, 'f^*'; the default box constraints, 'Box constraints'; the number of known local minimizers, 'loc', and the number of known global minimizers, 'glob'.

Table 4. Test functions.

Prob.	Abbr.	f^*	n	Box constraints	loc	glob
Branin	BR	0.3978874	2	$[-5,0]\times[10,15]$	3	3
Golstein-Price	GP	3.0000000	2	$[-2,2]^2$	4	1
Hartman 3	H3	-3.8627821	3	$[0,1]^3$	4	1
Hartman 6	H6	-3.3223680	6	$[0,1]^6$	4	1
Six-hump camel	C6	-1.0316285	2	$[-3,3]\times[-2,2]$	6	2
Shekel 5	S5	-10.1531997	4	$[0,10]^4$	5	1
Shekel 7	S7	-10.4029406	4	$[0,10]^4$	7	1
Shekel 10	S10	-10.5364098	4	$[0,10]^4$	10	1
Shubert	SHU	-186.7309088	2	$[-10,10]^2$	760	18

In order to assess the performance of the implemented EM, it is compared with the GA and SA packages available in the Matlab Global Optimization Toolbox. For more details on the methods, see [19]. The justification for the choice of these solvers is due to the fact that they are implemented in Matlab and are derivative-free methods, such as EM. Besides, other global stochastic methods in Matlab environment, like the implemented in functions `GloabSearch`

and `Multistart`, they are not used for comparison since they use gradient-based methods in the local search phase.

The EM algorithm used the default parameters, set by $PopulationSize = 20$, $\delta_{local} =$ 1e-3 and $MaxLocalIterations = 5$. For a fair comparison, GA and SA also used the default parameter values. The value of f^* was set to the parameter $ObjectiveLimit = f^*$. If the optimal solution is not reached within an absolute tolerance of 1e-5, the algorithm stops for a maximum of 10000 function evaluations, $MaxFunEvals = 10000$. In order to obtain statistically significant results, we run each method 30 times.

Table 5 reports the name of the problem, 'Prob.'; the best function value obtained by each algorithm, 'f_{best}'; the standard deviation between the best function values obtained by the algorithm, '*StdDev*'; the average number of function evaluations, 'N_{avg}'. Generally, EM reaches the global optimal solutions

Table 5. Comparison of EM results with GA and SA.

Prob.	EM			GA			SA		
	f_{best}	*StdDev*	N_{avg}	f_{best}	*StdDev*	Navg	f_{best}	*StdDev*	N_{avg}
BR	0.39789	7.77e-07	1086	0.39789	8.96e-07	825	0.39789	3.03e-06	6077
GP	3.00000	2.89e-06	1187	3.00000	2.09e+01	2048	3.00000	3.02e-06	485
H3	-3.86278	2.99e-06	1662	-3.86278	1.41e-01	1875	-3.86278	1.76e-04	9815
H6	-3.32237	3.66e-02	2682	-3.32237	5.84e-02	5584	-3.32075	1.49e-02	10001
C6	-1.03163	2.22e-06	491	-1.03163	2.28e-06	579	-1.03163	1.93e-06	1794
S5	-10.15320	3.30e+00	6175	-5.05520	5.93e-09	10020	-10.15308	3.50e-01	10001
S7	-10.40294	3.16e-06	3010	-5.08767	7.20e-09	10020	-10.40288	1.54e-01	10001
S10	-10.53641	5.16e-06	2728	-5.12848	5.22e-09	10020	-10.53638	1.36e+00	10001
SHU	-186.73091	1.30e-05	6155	-186.73091	5.83e+01	7599	-186.73091	5.11e-06	492

with less function evaluations than GA and SA. From *StdDev* column we can conclude EM obtained more consistent and effective solutions except for problem S5, where EM converged to attractive locals, for some runs (see Fig. 1). For problems S5, S7 and S10, GA didn't reach the global optimum (always converged to local solutions), and EM has more accurate solutions than SA.

4.2 Parameter Estimation Problem

Most biological processes are nonlinear dynamic systems, which are usually modeled as a nonlinear programming problem subject to dynamic (usually differential-algebraic) constraints that describe the evolution over time of certain quantities of interest. In the context of bioprocess engineering optimization, the parameter estimation problem goal is to find the set of parameters of a mathematical model to obtain the best possible fit to the existing experimental data [17,26,30]. Since these problems can be non-convex, some of the main optimization methods based on the gradient, such as, Levenberg-Marquardt or Gauss-Newton [21], may not work well, because if there is a local minimum very close to the global minimum,

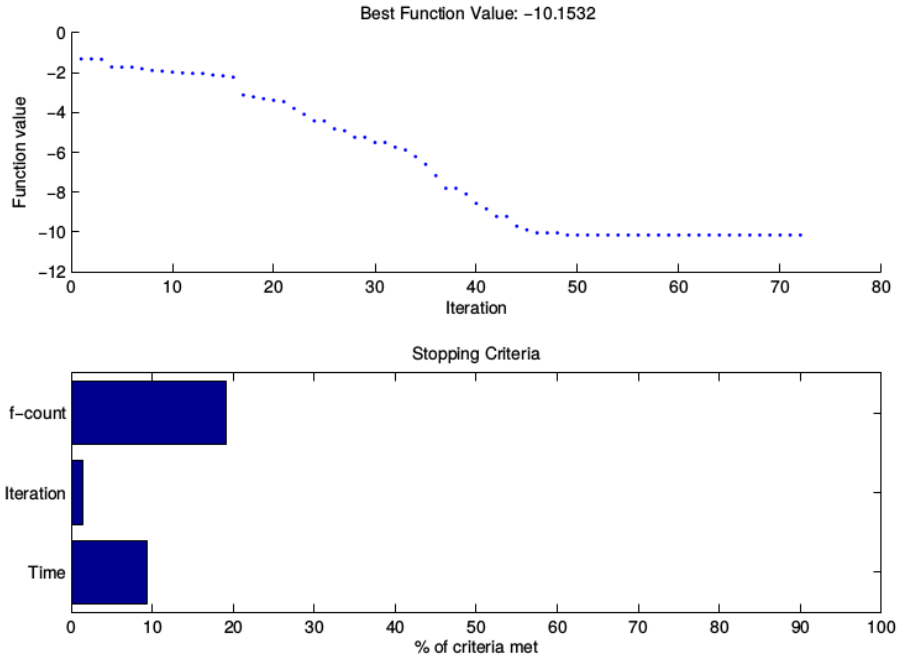

Fig. 1. Example of a run of EM algorithm with Shekel 5 problem.

the method may fail, failing to reach the optimal solution. Thus, there is a need for using global optimization methods to ensure convergence to a global optimal solution. The literature presents some advances in global optimization for problems of parameter estimation in dynamic systems in deterministic methods [8,18] and in stochastic methods [15,20,25].

In this paper, we are interested in solving the isomerization of α-pinene problem, that is a parameter estimation problem that arises from the modeling of the chemical phenomena of an isomerization reaction. The α-pinene estimation problem was firstly discussed by Box et al. [5] in 1973 and by Bates and Watts [2] in 1988 and by Seber and Wild [27] in 1989. Fuguitt and Hawkins [9] in 1947 studied the reaction of α-pinene where (y_1) is thermally isomerized to dipentene (y_2) and allo-ocimene (y_3). Allo-ocimene in turn yields pyronene (y_4) and a dimer (y_5). The converting of allo-ocimene (y_3) to the dimer (y_5) is a reversible reaction while the other conversions are irreversible. The goal of α-pinene estimation problem is to estimate the five rate constants $(p_1, \ldots, p_5)$. Fig. 2 shows the proposed reaction scheme for this homogeneous chemical reaction describing the thermal isomerization of α-pinene.

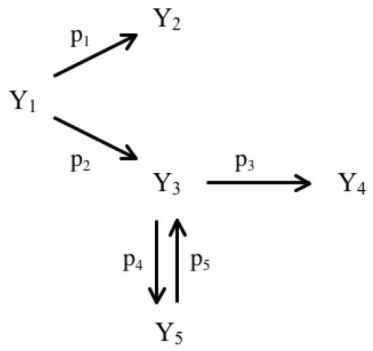

Fig. 2. Mechanism for thermal isomerization of α-pinene.

The model is defined by the following system of five ordinary differential equations:

$$\frac{d[y_1]}{dt} = -(p_1 + p_2)y_1 \tag{6}$$

$$\frac{d[y_2]}{dt} = p_1 y_1 \tag{7}$$

$$\frac{d[y_3]}{dt} = p_2 y_1 - (p_3 + p_4)y_3 + p_5 y_5 \tag{8}$$

$$\frac{d[y_4]}{dt} = p_3 y_3 \tag{9}$$

$$\frac{d[y_5]}{dt} = p_4 y_3 - p_5 y_5 \tag{10}$$

considering the given set of time intervals:

$$t = \{1230.0, 3060.0, 4920.0, 7800.0, 10680.0, 15030.0, 22620.0, 36420.0\} \tag{11}$$

It is assumed that the initial values (i.e. at time $t \approx 0$) of one reactant and four products are $y_1(0) = 100$, $y_2(0) = 0$, $y_3(0) = 0$, $y_4(0) = 0$, $y_5(0) = 0$.

The goal of the α-pinene estimation problem is to estimate (predict) the unknown coefficients, $p_1, ..., p_5$, minimizing the objective function that corresponds to a weighted distance measure between the experimental values, corresponding to the measured variables, and the predicted values for those same variables, formulated as:

$$f(p) = \sum_{j=1}^{5} \sum_{i=1}^{8} \left(y_j(p, t_i) - \widetilde{y}_{ji}\right)^2 \tag{12}$$

The experimental values for each of the five responses $(y_1, \ldots, y_5)$ in the eight interval times are given in the following matrix,

$$\widetilde{y}_{ji} = \begin{bmatrix} 88.35 & 7.3 & 2.3 & 0.4 & 1.75 \\ 76.4 & 15.6 & 4.5 & 0.7 & 2.8 \\ 65.1 & 23.1 & 5.3 & 1.1 & 5.8 \\ 50.4 & 32.9 & 6.0 & 1.5 & 9.3 \\ 37.5 & 42.7 & 6.0 & 1.9 & 12.0 \\ 25.9 & 49.1 & 5.9 & 2.2 & 17.0 \\ 14.0 & 57.4 & 5.1 & 2.6 & 21.0 \\ 4.5 & 63.1 & 3.8 & 2.9 & 25.7 \end{bmatrix}$$

The best known solution is $p_1^* = 5.9256\text{e-}5$, $p_2^* = 2.9632\text{e-}5$, $p_3^* = 2.0450\text{e-}5$, $p_4^* = 2.7473\text{e-}4$, $p_5^* = 4.0073\text{e-}5$ with optimal value $f^*(p) = 19.872$.

Hereafter, the performance of EM algorithm will be tested when solving the hard optimization problem from the chemical and bio-process engineering area - the isomerization of α-pinene problem.

We used EM as described in Section 3 with the default parameters: population size of 20 points, $m = 20$, $\delta_{local} = 1e-3$ and MaxLocalIterations=5. The lower bounds considered for the five parameters arise from physical considerations, $p_i \geq 0$, and we took the upper bounds to be $p_i \leq 5\text{e-}4$, for $i = 1, \ldots, 5$.

Table 6 shows the average results obtained with EM, among 10 independent runs, when running the code with a time limit of 50 seconds.

Table 6. EM solutions after 50 seconds.

	f_{best}	f_{median}	f_{worst}	N_{feavg}	It_{avg}
EM	19.874010	20.325178	21.805897	4070	147

The best solution found by EM gives p_1 =5.9260e-05, p_2 =2.9632e-05, p_3 =2.0470e-05, p_4 = 2.7425e-04, p_5 =3.9906e-05 and was found after 3824 function evaluations and 147 EM iterations. The variable values of the best solution obtained are represented in Fig. 3. Fig. 4 depicts the convergence curve of the objective function value along the iterations and the stopping criterion levels of the optimization process. The relation between experimental data and the predicted values of the α-pinene using the EM best solution is illustrated in Fig. 5.

In the following, we intend to compare the results obtained with the implemented EM algorithm when solving the problem of isomerization of α-pinene, with the ones obtained with GA and SA packages available in the Matlab Global Optimization Toolbox, using the default parameter values. Table 7 reports the best, f_{best}, the average, f_{avg}, the worst, f_{worst}, solution values and the average number of function evaluations, N_{avg}, of the 10 runs. Each run was limited to 50 seconds, as the stopping condition for all the stochastic solvers.

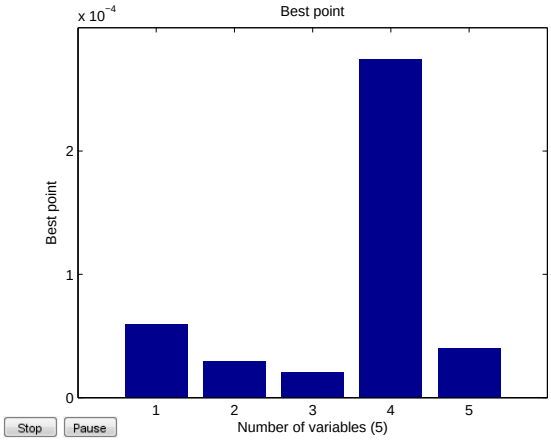

Fig. 3. Best solution found by EM for thermal isomerization of α-pinene.

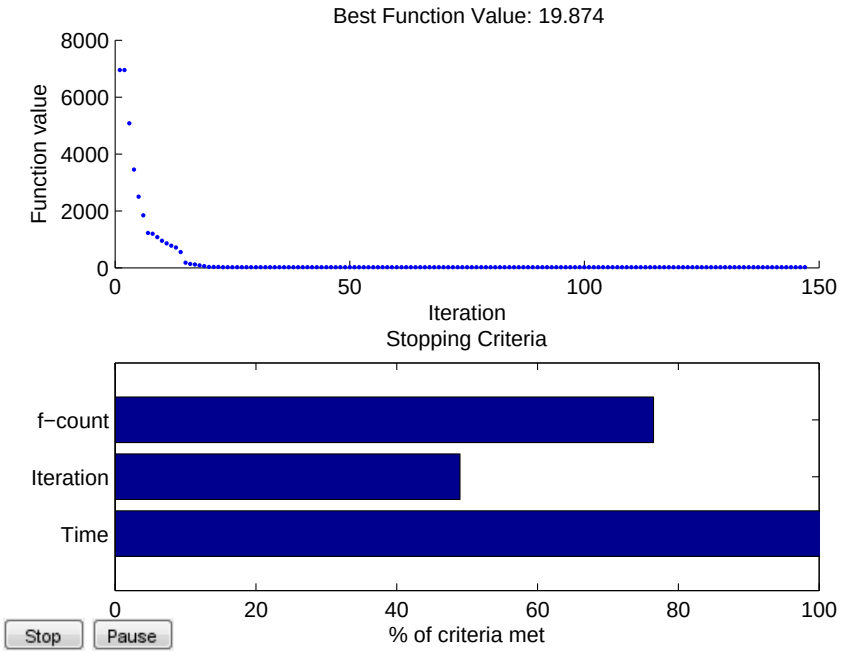

Fig. 4. Convergence curves for the α-pinene case study.

From Table 7 we may conclude that EM algorithm achieved more accurate solutions with less computational effort, when compared with GA and SA algorithms.

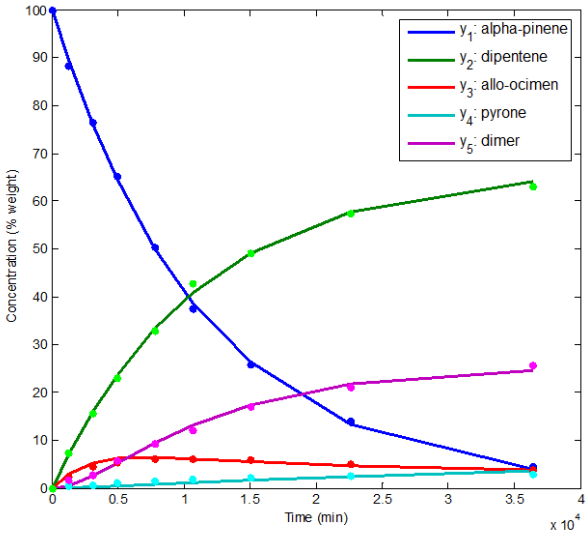

Fig. 5. Experimental data versus model prediction for the α-pinene case study.

Table 7. Comparison results for the isomerization of α-pinene problem.

Solver	f_{best}	f_{avg}	f_{worst}	N_{avg}
EM	19.874010	20.549177	21.805897	4070
GA	20.414430	167.137782	817.643129	4350
SA	24.718882	54.241137	108.635052	3982

In [7], we can find the solution obtained with the SSm method, a Scatter Search procedure with an improvement method, when solving the α-pinene: $f_{best} = 19.872000$, $f_{avg} = 24.747000$, $f_{worst} = 68.613000$ and $N_{avg} = 1144$. In fact, SSm reached the known global minimum but we remark that the improvement method used a gradient-based method implemented in the fmincon command of Matlab. However, the average results obtained with EM are more closer to the global optimal solution, with a standard deviation of 0.73279 of the ten best solutions found. Moreover, the relative error of the best solution found by EM algorithm was 1e-4 .Thus, EM gives competitive results when comparing with the SSm method, and was able to get good solution to the isomerization of α-pinene problem.

5 Conclusions and Future Work

In this paper, we implemented the EM algorithm in the Matlab® environment as a useful function for the scientific community with similar syntax to other op-

timization functions there already available. The implementation was considered successfully when tested under a set of benchmark problems that were solved and compared with other stochastic and derivative-free methods available in the Global Optimization Toolbox of Matlab. The implementation is freely available at `http://www.norg.uminho.pt/arocha/code.htm`.

The application of the EM algorithm for solving the hard isomerization of α-pinene problem showed good and competitive results when comparing with the other solvers in the Global Optimization Toolbox. This example shows how important it is to optimize mathematical models. The optimization plays a key role on systems biology. This parameter estimation problem was taken form the real world, and it could bring many advantages, to reduce costs of experimental measurements, for example.

In the future, we intend to integrate EM in the Matlab Optimization Graphical Interface, creating a user-friendly interface. As other global optimization functions available in the global optimization toolbox, we want to include hybrid functions to improve the quality of the solutions. The final purpose is to include the EM code as a Matlab built-in function in the Global Optimization Toolbox for unconstrained optimization problems.

Acknowledgments This work has been supported by FCT (Fundação para a Ciência e Tecnologia, Portugal) in the scope of the project PEst-UID/CEC/00319/2013.

References

1. Ali, M.M., Golalikhani, M., Zhuang, J.: A computational study on different penalty approaches for solving constrained global optimization problems with the electromagnetism-like method. Optimization. 63(3), 403–419 (2014)
2. Bates, D., Watts, D.: Nonlinear Regression Analysis and Its Applications. John Wiley & Sons, Inc. (2008)
3. Birbil, S. I., Fang, S.-C.: An electromagnetism-like mechanism for global optimization. J. Global Optim. 25, 263–282 (2003)
4. Birbil S. I., Fang S.-C., Sheu R.L.: On the convergence of a population-based global optimization algorithm. J. Global Optim. 30, 301–318 (2004)
5. Box, G.E.P., Hunter, W.G., MacGregor, J.F., Erjavec, J.: Some problems associated with the analysis of multiresponse data. Technometrics. 15, 33–51 (1973)
6. Colorni, A., Dorigo, M., Maniezzo, V.: Distributed Optimization by Ant Colonies. In: European Conference on Artificial Life, pp. 134–142. Elsevier Publishing (1991)
7. Egea, J.A., Rodríguez-Fernández, M., Banga, J.R., Martí, R.: Scatter search for chemical and bio-process optimization. J. of Global Optimization. 37(3), 481–503 (2007)
8. Esposito, W., Floudas, C.A.: Global optimization for the parameter estimation of differential-algebraic systems. Ind. Eng. Chem. Res. 39(5), 1291–1310 (2000)
9. Fuguitt, R.E., Hawkins, J.E.: Rate of the thermal isomerization of α-pinene in the liquid phase. J. Am. Chem. Soc. 69, 19–322 (1947)
10. Goldberg, D.E.: Genetic Algorithms in Search. Optimization & Machine Learning. Addison-Wesley (1989)

11. Ingber, L.: Simulated annealing: practice versus theory. Mathl. Comput. Modelling. 18, 29–57 (1993)
12. Jones, D.R., Perttunen, C.D., Stuckman, B.E.: Lipschitzian optimization without the Lipschitz constant. J. Optimiz. Theory App. 79, 157–181 (1993)
13. Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proceedings of IEEE International Conference on Neural Networks, vol. 4, pp. 1942–1948 (1995)
14. Kohl, P., Noble, D.: Systems biology and the virtual physiological human. Mol Syst Biol. 5, 292 (2009)
15. Larrosa, E.: New heuristics for global optimization of complex bioprocesses. Master Thesis, University of Vigo, Spain (2008)
16. Lieu, C., Elliston K.: Applying a causal framework to system modeling. In: Bringmann, P., Butcher, E., Parry, G., Weiss, B. (eds.) Ernst Schering Research Foundation Workshop. Systems Biology, ser., vol. 61, pp. 139–152. Springer, Berlin Heidelberg (2007)
17. Lillacci, G., Khammash, M.: Parameter estimation and model selection in computational biology. PLoS computational biology 6(3), p. e1000696 (2010)
18. Lin, Y., Stadtherr, M.A.: Deterministic global optimization for parameter estimation of dynamic systems. Ind. Eng. Chem. Res. 45, 8438–8448 (2006)
19. MATLAB and Global Optimization Toolbox Release 2013, The MathWorks, Inc., Natick, Massachusetts, United States. (2013)
20. Moles, C.G., Mendes, P., Banga, J.R.: Parameter estimation in biochemical pathways: a comparison of global optimization methods. Genome research, Vol. 13, No. 11, 2467–2474 (2003)
21. Nocedal, J., Wright, S.: Numerical Optimization. Springer Series in Operations Research. Springer, (1999)
22. Rocha, A.M.A.C., Fernandes, E.M.G.P.: Modified movement force vector in an electromagnetism-like mechanism for global optimization. Optim. Method. Softw. 24, 253–270 (2009)
23. Rocha, A.M.A.C., Fernandes, E.M.G.P.: Performance profile assessment of electromagnetism-like algorithms for global optimization. AIP Conf. Proc. 1060, 15–18 (2008)
24. Rocha, A.M.A.C., Fernandes, E.M.G.P.: Numerical study of augmented Lagrangian algorithms for constrained global optimization. Optimization. 60, 10–11 (2011)
25. Rodríguez-Fernández, M., Egea, J.A., Banga, J.R.: Novel metaheuristic for parameter estimation in nonlinear dynamic biological systems. BMC Bioinformatics. 7, 483 (2006)
26. Schittkowski, K.: Parameter estimation in systems of nonlinear equations. Numerische Mathematik. 68(1), 129–142 (1994)
27. Seber, G., Wild, C.: Nonlinear Regression. John Wiley & Sons, Inc. (1989)
28. Sun, J., Garibaldi, J.M., Hodgman, C.: Parameter estimation using metaheuristics in systems biology: A comprehensive review. IEEE/ACM Trans. Comput. Biol. Bioinformatics 9(1), 185–202 (2012)
29. Villaverde, A., Egea, J., Banga, J.: A cooperative strategy for parameter estimation in large scale systems biology models. BMC Syst. Biol. 6, 75 (2012)
30. Zhan, C., Yeung, L.F.: Parameter estimation in systems biology models using spline approximation. BMC Syst. Biol. 5, 14 (2011)