JOURNAL OF OPTOELECTRONICS AND ADVANCED MATERIALS Vol. 14, No. 11- 12, November – December 2012, p. 976 - 982

# Fast algorithm for blind optimization of optical systems. Statistics and methodology

M. R. N. AVANAKI[a,d], Y. LONG[b], M.-A. PAUN[c*], S. A. HOJJATOLESLAMI[a], A. GH. PODOLEANU[d]
[a]*Medical Image computing group, School of Biosciences, University of Kent, Canterbury, CT2 7PD, United Kingdom*
[b]*Mathematics department, University of Cambridge, Cambridge, C2 7PD, United Kingdom*
[c]*Electronics Lab, EPFL (Swiss Federal Institute of Technology), CH-1015 Lausanne, Switzerland*
[d]*Applied Optics Group, School of Physical Sciences, University of Kent, Canterbury, CT2 7NH, United Kingdom*

Adaptive Optics (AO) improves the efficiency of the optical devices in confocal imaging systems by reducing wavefront aberrations. Aberration is caused by imperfections within the system and reduces the optical signal to noise ratio of the resultant images besides deteriorating the quality of the images. An adaptive optics system comprises a wavefront sensor and a deformable mirror (DM), is a straightforward solution to compensate for this distortion. In confocal microscopy, the wavefront sensor can be supplanted with an optimization algorithm. We have previously implemented the general simulated annealing (SA) algorithm for optimizing confocal microscopes. In this paper the modified version of the simulated annealing algorithm, fast simulated annealing (FSA) is investigated which takes less than one hundredth of the optimization time required by the general version.

(Received October 17, 2012; accepted October 30, 2012)

*Keywords:* Fast simulated annealing (FSA), Blind optimization, Adaptive Optics (AO), Boltzmann distribution, Optical system

## 1. Introduction

Simulated annealing (SA) is a metaheuristic algorithm whose purpose is to locate an appropriate approximation to the global optimum of a given cost function in a defined and discrete search space [1]. The concept of the SA technique comes from the physical process of heating to a temperature that permits many atomic rearrangements, and then slowly cooling a substance allowing it to reach to the thermal equilibrium at each temperature until the material freezes into an ordered crystalline structure or into a structure with the lowest energy [1]. With SA, random samples from the search space are generated to be searched for which direct sampling is difficult. The goal of the SA algorithm is merely to find the best possible solution.

In the optimization problem, each point in the search space corresponds to a state *s* of a physical system, and the steps within the SA algorithm moves from one point to another in its neighbourhood, which is accepted probabilistically. The energy function is *x(s)* which serves as the objective function to be optimized. *x(s)* is equivalent to the internal energy of the system when it is in state *s*. SA brings the system from an arbitrarily chosen initial state to a state with the energy function as close as possible to the optimal value. In the other word SA can find the state at which the maximum or minimum of the optimizing variable occurs [2-3].

Unlike many other iterative improvement algorithms which do not allow movements towards higher energy states, SA allows occasional increases in energy caused by introducing some random perturbation. This increase in energy is moderated by the synthetic temperature and a probability function. The synthetic temperature in the SA algorithm is used in order to determine how often the algorithm switches between local and global searches. In the SA algorithm, the synthetic temperature initially is high enough to permit an aggressive random search, and most uphill moves are allowed. As the temperature cools down, fewer uphill moves are allowed. At very cold temperatures, very few disruptive uphill moves are permitted. In this temperature regime, annealing closely resembles standard downhill-only iterative improvement

SA is favoured due to its efficiency with less computational complexity. Compared to SA, most of the known techniques for obtaining an optimum solution require an exponentially increasing number of steps as the problem becomes larger. SA serves as a more efficient method than exhaustive enumeration in finding an optimal solution in a limited time of optimization [4].

In SA the entire solution space is represented using the approximation of the space by a distribution, which can be Normal, Boltzmann or Cauchy. These distribution functions are the most common used ones. SA algorithm involves 3 functional relationships:

(1) *g(x):* Probability generating function of search space. The function is a distribution used to approximate the search space. This distribution can be Normal, Boltzmann or Cauchy, depending on the variant of the SA technique used.

(2) *h(x):* Probability of accepting the new value of energy function given the previous value. For SA, this takes the form of a function known as the Metropolis probability. Metropolis probability, also known as Boltzmann probability function.

(3) *T(k):* The schedule adapted to anneal the synthetic temperature T with each execution of the entire loop, measured in annealing time steps. The solution will

fluctuate in a random fashion when T is large initially, but as T decreases due to the annealing schedule, the solution should be more likely to improve and move towards the global optimum. An annealing schedule, explains the choice of initial temperature, how many iterations are performed at each temperature, and how much the temperature is decreased at each step as cooling proceeds. Table 1, shows each of the above functional relationships for different versions of the SA algorithm [5].

*Table 1. Variations on the SA algorithm. D in the table is the dimension of the search space. k is incremented each time when the optimization problem reaches to the equilibrium state [5]. ΔX shows the rate of change of X (variables' vector). So X = X0 + ΔX where X0 denotes the current state and X shows the next state of variables' vector.*

| | *General SA* | *FSA* |
|---|---|---|
| *Generating function, g(x)* | $(2\pi T)^{-\frac{D}{z}} e^{\frac{-\Delta x^{z}}{zT}}$ | $\frac{T}{(\Delta x^2 + T^2)^{\frac{D+1}{2}}}$ |
| *Acceptance Probability* | $P = e^{\frac{-\Delta x}{T}}$ | |
| *Annealing Schedule, T(k)* | $T_k = \frac{T}{\ln k}$ | $T_k = \frac{T}{\ln k}$ |

According to Table 1, annealing schedule and the start temperature are the two main parameters that can cause the algorithm to be faster or slower. In the newer versions of the SA, it has been tried to modify these two parameters along with taking into account a more effective generating function.

In this study we explain the application of Cauchy distribution function as the generating function and using the corresponding annealing schedule to reduce the elapsed time for the algorithm to converge to the optimal value. We added a few more modifications on the search space such that each variable would be searched in its dynamic range to make the algorithm even faster.

## 2. Methodology

SA comprises 2 loops, as follows.

### 2.1 The Ordinary Optimization Loop

After entering a random initial state, SA considers a neighbour *s'* of the current state *s*, which is reached by slightly altering *s* and decides whether to move the system to the new state. If the value of *x(s')* is better than that of *x(s)*, then the system will move to state s'. Otherwise the algorithm will probabilistically accept the new solution and enter a separate loop, the Boltzmann Optimization Loop (BOL). The ordinary loop in fact acts as a local search engine.

### 2.2 The Boltzmann Optimization Loop

If the value of *x(s)* is better than that of *x(s')*, then the state *s'* is accepted with the probability P (Eq. 1).

$$P = e^{\frac{-\Delta x}{T}} \tag{1}$$

where $\Delta x = |x(s) - x(s')|$ is the change in the energy function caused by the change in state. This stands as the choice for *h(x)* in all versions of the SA algorithm. After this stage, the algorithm enters the Boltzmann optimization loop, which introduces a large perturbation (usually with the full range of variable) to $s'$ in order to escape the neighbourhood within which this state lies. This prevents the algorithm to trap into the local extremum. The Boltzmann loop plays the role of the global search in the algorithm. The new state s'' is then tested in the same way as in the ordinary loop: If *x(s'')* is a better solution than *x(s)* then *x(s'')* becomes the new optimal solution, and the loop is repeated until the Boltzmann loop has been executed five times. The number five has been empirically obtained. The system once again performs the ordinary optimization loop on the optimal solution obtained from the five iterations (the number five was optimized experimentally, [1]). The ordinary loop is repeated until stopping conditions are satisfied for the current synthetic temperature. The algorithm then applies the annealing schedule to the synthetic temperature before the entire loop is carried out once again. A target temperature, named freezing temperature, is initially set and when the synthetic temperature falls below that, the algorithm terminates.

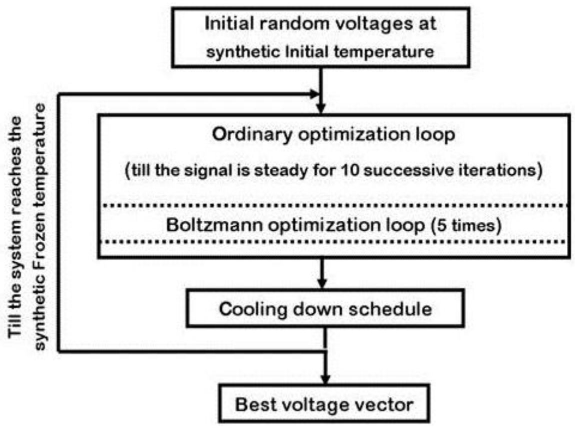
Initial random voltages at synthetic Initial temperature
Ordinary optimization loop
(till the signal is steady for 10 successive iterations)
Boltzmann optimization loop (5 times)
Cooling down schedule
Best voltage vector
Till the system reaches the synthetic Frozen temperature

*Fig. 1. General SA algorithm; the algorithm is used to optimize 52 voltage values sent to a deformable mirror in an adaptive optics system [6].*

As indicated in Fig. 1, there is an initial voltage set that is sent to the corrector. The initial voltage set is usually a vector of zero. However, a random search algorithm is used to generate a start voltage vector for the SA algorithm, so that the algorithm begins with this voltage vector instead of using the zero vector. Using the random search algorithm helps the optimization algorithm to have a shorter convergence time. The algorithm for the random search is based on the same algorithm used in the BOL.

It can be proven in general SA that any point within the search space can be sampled infinitely often in annealing time: if we let $g_k$ be the probability of sampling the search space on the kth iteration then the product of the probabilities of not generating a state during each iteration must be 0 (Eqs. 2 and 3):

$$\prod_{k=1}^{\infty}(1-g_k)=0 \tag{2}$$

or equivalently:

$$\prod_{k=1}^{\infty} g_k=\infty \tag{3}$$

The problem is to find an appropriate annealing schedule *T(k)* such that the above formula is satisfied. Given that for the general SA the schedule is:

$$T(k)=\frac{T}{\ln(k)} \tag{4}$$

Hence,

$$\sum_{k=1}^{\infty} g_k \geq \sum_{k=1}^{\infty}\exp(-\ln k)=\sum_{k=1}^{\infty}\frac{1}{k}=\infty \tag{5}$$

As explained earlier, the SA algorithm takes over an hour to find the global optimum. The algorithm used in this study though is the fast simulated annealing (FSA) algorithm which has a search space generated by the Cauchy distribution (Eq. 6).

$$g(x)=\frac{A}{\pi(B^2+(x-A)^2)} \tag{6}$$

where $A$ and $B$ are constants which determine the shape of the distribution. It also uses a different annealing schedule (Eq. 7) for the synthetic temperature, which allows this synthetic temperature to cool at a faster rate than in general SA, and so reach the target temperature and therefore fulfil the stopping condition with a quicker rate.

$$T_k=\frac{T_0}{k} \tag{7}$$

Hence,

$$\sum_{k=1}^{\infty} g_k \approx \frac{T_0}{\Delta x^{D+1}}\sum_{k=1}^{\infty}\frac{1}{k}=\infty \tag{8}$$

To evaluate the performance and error of the algorithm we applied the algorithm on a 4th order polynomial. The reason that the polynomial function was chosen is as follows. Firstly, SA algorithm is a solving method which operates on the domain of those optimization problems, in which the sets of feasible solutions are discrete or can be reduced to discrete and the polynomial function has this feature. Secondly, SA algorithm is well suited to those problems in which many variables needs to be optimized independently at once; here four variables are to be optimized simultaneously.

The SA algorithm is used to reduce the deviation of the graph plotted of the 4th order polynomial $ax^4+bx^3+cx^2+dx+e$ from the graph for $e^x$, finding the value of the coefficients, a, b, c, d, and e, which accomplish this task in the process. Each perturbation of the system involves altering the five coefficients within the polynomial curve, and the sum of the squared differences between each of the plotted points on the two graphs was taken as the energy function to be minimized. Although the evaluation was done on the 4th order equation to optimize only the five variables, but the algorithm is able to optimize any number of variables in a solution space.
The initial temperature was considered as 1000 as the performance of the algorithm versus time elapsed to converge the algorithm to the optimal value for this temperature was the lowest. The other values of initial temperatures that have been tested were 10, 100, and 10000. In the same way, the synthetic freezing temperature was considered as 0.2, while 0.000001, 0.001, 0.01, 1, and 10 were tested. Stopping criteria was satisfied by the freezing temperature, it can however be set as the elapsed time, the number of iterations, and the amount of error.

## 3. Evaluation

The FSA algorithm following the algorithm given in Figure 2 was implemented in MATLAB 2010a and tested on a Pentium IV computer with 2.2 GHz CPU and 2.2 GB memory.

*Table 2. Results obtained from 6 independent repeats of the algorithm. The optimal deviation is the deviation calculated for the obtained optimal solution at each trial.*

| Trial | a | b | c | d | e | Optimal deviation |
|---|---|---|---|---|---|---|
| 1 | 1.813 | 0.338 | 0.112 | 3.931 | 0.122 | 87.40 |
| 2 | 0.006 | 0.028 | 0.185 | 0.419 | 0.138 | 69.10 |
| 3 | 1.813 | 0.338 | 0.112 | 3.931 | 0.122 | 87.40 |
| 4 | 0.316 | 0.120 | 0.449 | 1.491 | 0.239 | 57.14 |
| 5 | 0.213 | 0.183 | 0.264 | 0.897 | 0.256 | 19.82 |
| 6 | 0.328 | 0.190 | 0.286 | 1.021 | 0.172 | 159.342 |

Table 2 shows the final values of the coefficients obtained from each repeat of the algorithm which allows the best fit of the graph for $ax^4+bx^3+cx^2+dx+e$ to $e^x$, along with the energy function of the state the final graph represents. The graphs shown in Figure 3 illustrate four states of the progression of the algorithm in optimizing the polynomial curve. The solutions of the ordinary and Boltzmann optimization loops produced at these states are shown as well.

The speed of the FSA compared to the general SA is given in Table 3. The elapsed time is reduced more than 100 times.

*Table 3. Comparison between the speeds of the SA algorithms in the general SA and the FSA.*

| | Time taken to optimize system (s) |
|---|---|
| General SA | >3600 |
| FSA | 26 |

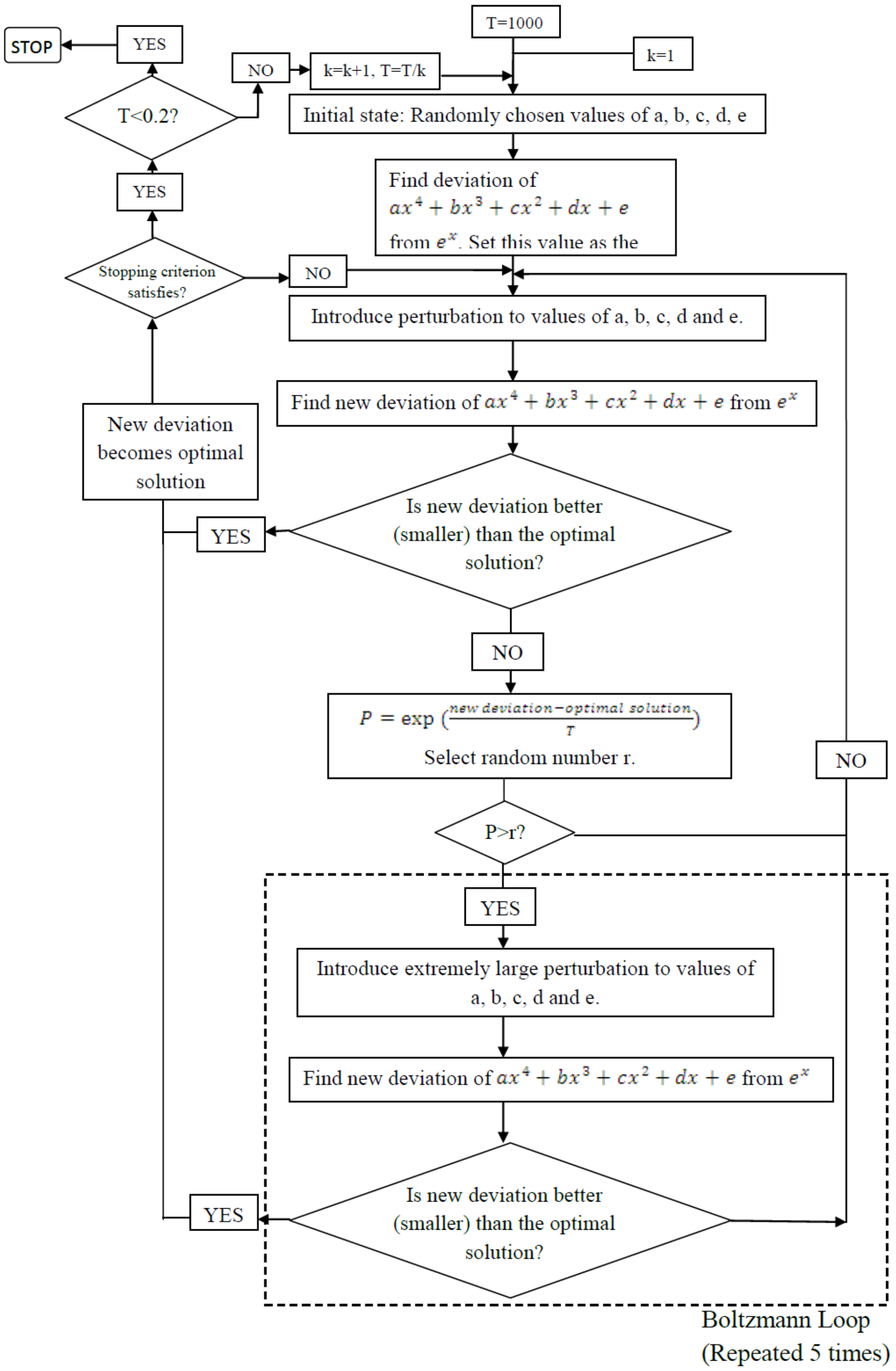
STOP
YES
T=1000
NO
k=k+1, T=T/k
k=1
T<0.2?
Initial state: Randomly chosen values of a, b, c, d, e
YES
Find deviation of $ax^4 + bx^3 + cx^2 + dx + e$ from $e^x$. Set this value as the
Stopping criterion satisfies?
NO
Introduce perturbation to values of a, b, c, d and e.
Find new deviation of $ax^4 + bx^3 + cx^2 + dx + e$ from $e^x$
New deviation becomes optimal solution
YES
Is new deviation better (smaller) than the optimal solution?
NO
$P = \exp\left(\frac{new\ deviation - optimal\ solution}{T}\right)$
Select random number r.
NO
P>r?
YES
Introduce extremely large perturbation to values of a, b, c, d and e.
Find new deviation of $ax^4 + bx^3 + cx^2 + dx + e$ from $e^x$
YES
Is new deviation better (smaller) than the optimal solution?
Boltzmann Loop
(Repeated 5 times)

*Fig. 2. The FSA flowchart; the algorithm is executed repeatedly until the median of the differences between the 10 consecutive solutions and their mean is smaller than one tenth of their range. In this flowchart, the deviation of the polynomial curve from the exponential function is minimized to obtain the optimal solution.*

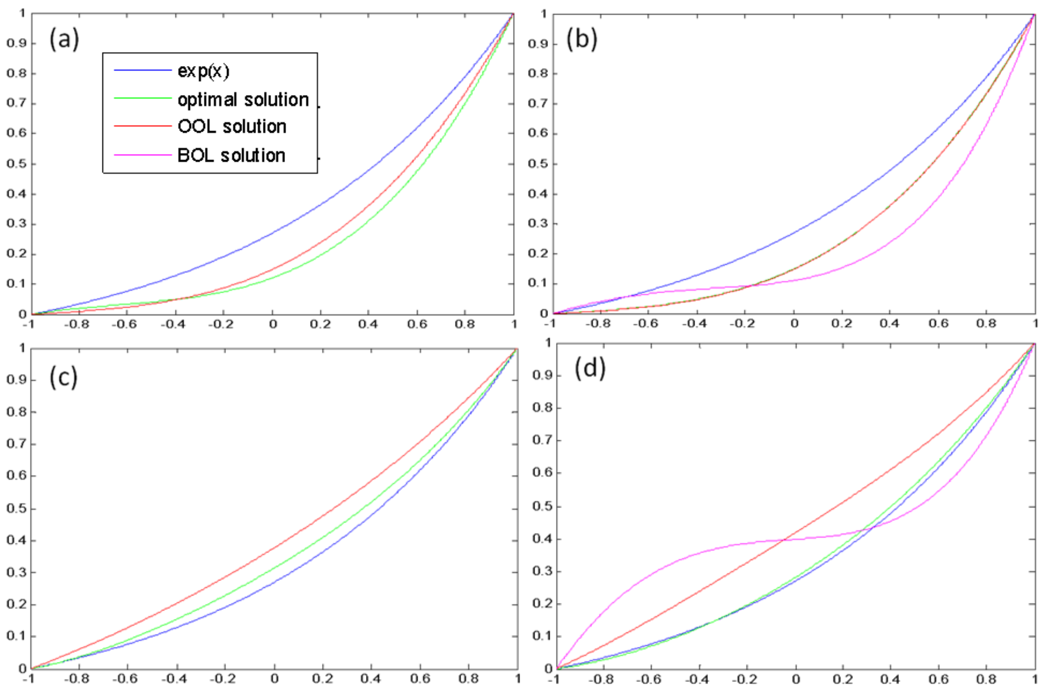

*Fig. 3. Four states of the algorithm in the progress of the optimization problem; (a), (b), (c), and (d) are four states of the polynomial curve towards to be exponential curve. Synthetic temperature is (a) 5000, (b) 500, (c) 5, (d) 0.2. Graph for $e^x$ is plotted in blue. Graph of $ax^4 + bx^3 + cx^2 + dx + e$ with optimal combination of coefficients obtained so-far during execution of program is plotted in green. The solution obtained for each iteration in the Ordinary Optimization Loop (OOL) is in red and for the Boltzmann Optimization Loop (BOL) is in magenta.*

## 4. Discussion

Previously we have experimented three well-established optimization algorithms, namely SA [6-8], genetic algorithm [9] and particle swarm optimization [10-11]. All the three algorithms took sometimes up to several hours to converge to their optimal value. This was due to the fact that the algorithm was not searched in an efficient probability distribution function with its cooling schedule. Moreover the algorithms did not search in an appropriate range of values for the optimizing variables.

The optimization algorithms that we have used are probabilistic hence they may converge to different values each time when executed if the runtime is not sufficient. The graphs shown in Fig. 4, illustrate the optimal polynomial graphs obtained from each of the executions of the algorithm displayed in Table 2, along with the solutions of the ordinary and Boltzmann optimization loops produced at the time of the termination of the program on stopping conditions having been met. The optimal polynomial graphs given in Figure 4, are different. This is due to the fact that the optimal deviation in each trial given in Table 2 is different one from another. In practice, usually the algorithm is executed couple of times and then the best solution is chosen as the solution of the optimization problem. The best optimal values of the coefficients of a, b, c, d, and e from the optimization of the polynomial curve are given in Table 4. The coefficients correspond to the graph in Figure 4(d).

*Table 4. The best optimal values of the coefficients which provided the best fitting of the polynomial curve to the exponential curve on Fig. 4(d)*

| a | b | c | d | e |
|---|---|---|---|---|
| 0.213 | 0.183 | 0.264 | 0.897 | 0.256 |

The energy function of the initial solution calculated for the polynomial was 143295.32. The final value of the energy function being reduced to 19.82 after the optimal solution was attained. The optimal values of the energy function obtained throughout the course of the execution of the algorithm are given in the bar chart shown in Fig. 5.

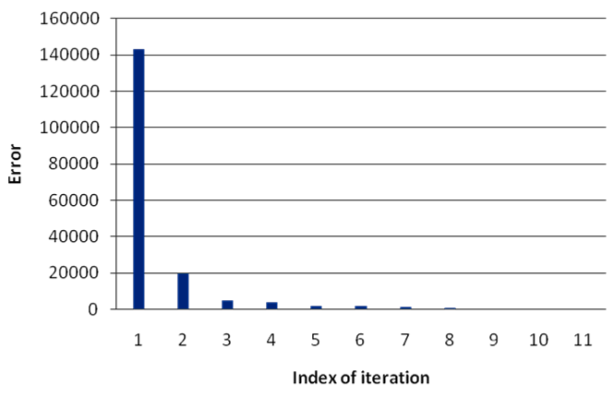

*Fig. 5. Energy minimization in the progress of the SA algorithm.*

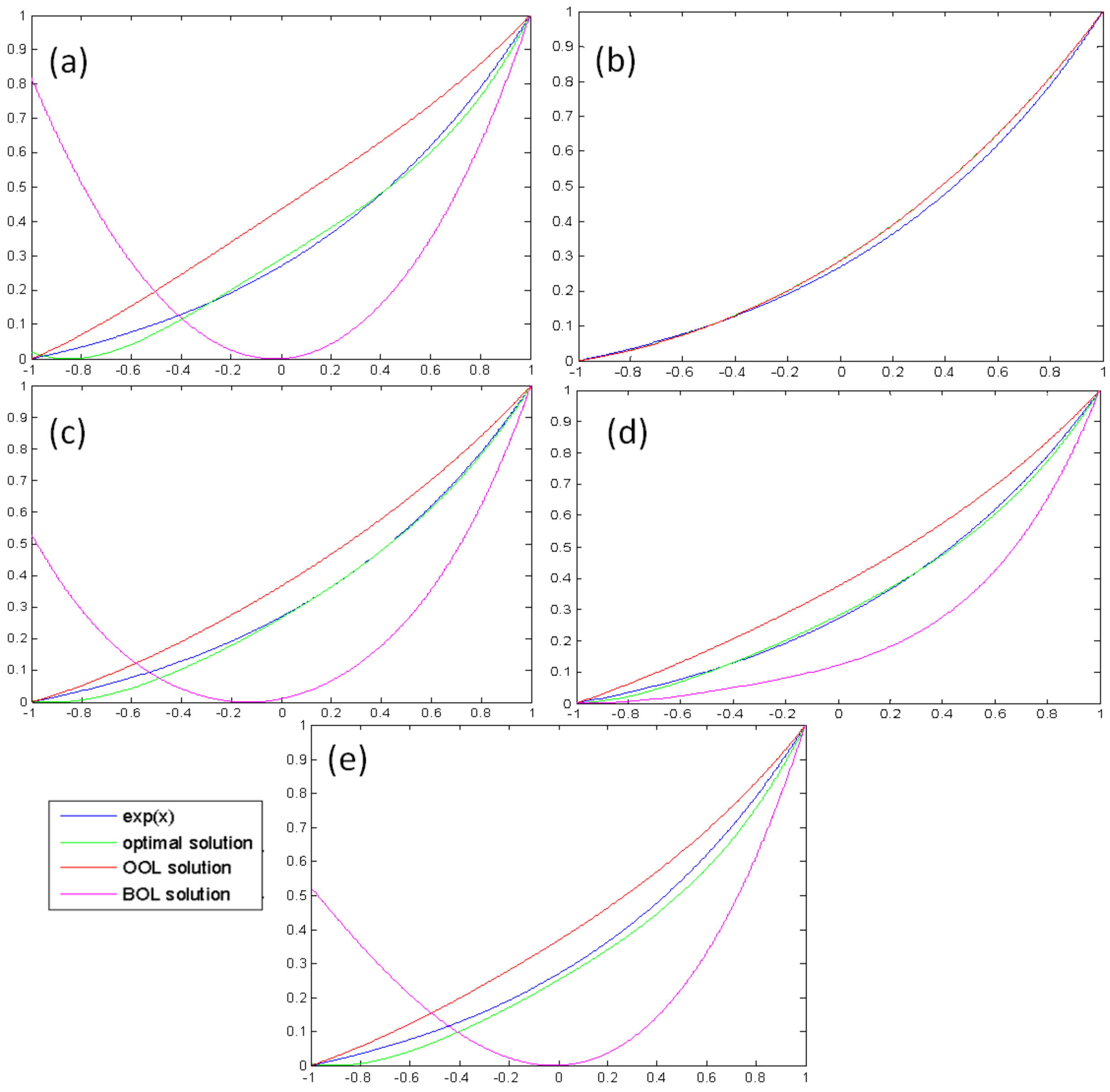

*Fig. 4. Optimization convergence results in the trails given in Table 2. (a) Graph of the polynomial curve from trial 1, (b) Graph of polynomial curve from trial 4, (c) Graph of polynomial curve from trials 2, (d) Graph of polynomial curve from trial 5, (e) Graph of polynomial from trial 6.*

We note that all the values in the progression of the optimizing process have been normalized ($x_{normalized}$) using the formulae given in Eq. 9 such that the different graphs and their deviations can be compared to each other. Ideal solution (exponential curve), optimal solution obtained so far, solution obtained from ordinary loop, solution obtained from Boltzmann loop are the variables that have been normalized. Given $x$ be the variable is to be normalized:

$$x_{normalized} = \frac{x - \min(x)}{\max(x - \min(x))} \quad (9)$$

To be able to find the range of each of the optimizing variables, we used a random search algorithm [9,12]. In fact, as explained earlier, the ranges of the variables that most likely generate the optimal solution can be found. The ordinary loop and Boltzmann loop then search more efficiently in these ranges.

The FSA will be used for optimization of the confocal microscope using a sensor-less method that we have previously implemented [8]. The signal processing block diagram of the sensor-less optimization technique is given in Fig. 6.

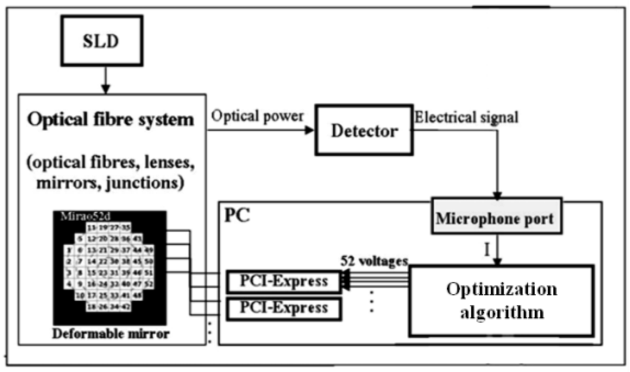

*Fig. 6. Signal processing block diagram of the sensor-less optimization technique. SLD: super luminescent diode, PS: personal computer, I: intensity of the photodetector acquired by the microphone port.*

In this setup, the photodetector signal is transferred into the microphone port. The signal is then processed in Matlab by the optimization algorithm. The output of the algorithm is a set of 52 voltage values which are sent through PCI7200 to the deformable mirror (DM). The shape of the mirror's surface is changed and a new wavefront generated hence the photodetector signal level varied. This process continues until one of the stop conditions is fulfilled. The final shape of the mirror's surface produces a higher photodetector signal level than its initial shape.

## 5. Conclusion

In this study, we have investigated the methodology of the FSA algorithm. We explain the application of Cauchy distribution function as the generating function and using the corresponding annealing schedule to reduce the elapsed time for the algorithm to converge to the optimal value. We discussed the performance of the general and FSA. By optimizing the number of iteration in the ordinary and Boltmann loops, we reached a faster optimization algorithm. Searching around the most likely value found using the random search algorithm, the FSA became even faster. The simulation was carried out on optimizing the coefficients of a 4th order equation to be fitted on the exponential curve. Results showed that the algorithm is able to find the optimal coefficients through the FSA algorithm more efficiently and much faster. We reached one hundredth time shorter convergence time than that of the original SA.

## References

[1] S. Kirkpatric, C. Gelatt, M. Vecchi, Science, **220**, 671 (1983).
[2] R. Rutenbar, Simulated Annealing Algorithms: An Overview. IEEE Circuits Devices Mag., no. 5, pp.19-26, 1989.
[3] Lawler, E. L., Rinnooy-Kan, A., Lenstra, J. K. et al.: The traveling salesman problem: A guided tour of combinatorial optimization. John Wiley & Sons Inc, 1985.
[4] D. Merino, C. Dainty, A. Bradu, A. G. Podoleanu, Optics Express, **14**, 3345 (2006).
[5] M. T. Vakil-Baghmisheh, A. Navarbaf, "A modified very fast simulated annealing algorithm," in Telecommunications, 2008, IST 2008, International Symposium on, 2008, pp. 61-66.
[6] Mohammad R.N. Avanaki, S. A. Hojjatoleslami, M. Paun, S. Tuohy, A. Meadway, G. Dobre, A. GH. Podoleanu, Proceedings of Mathematical Methods and Applied Computing, WSEAS, pp. 669-674, (2009).
[7] Mohammad R.N. Avanaki, S. A. Hojjatoleslami, A. Meadway, G. Dobre and A. Gh. Podoleanu, Iranian Conference on Optics and Photonics (ICOP) 2010, 1, pp. 120-124, 2010.
[8] M. Paun, Mohammad R.N. Avanaki, G. Dobre, A. GH. Podoleanu, S. A. Hojjatoleslami, J. Optoelecvtron. Adv. Mater. **11**, 1681 (2009).
[9] M. R. N. Avanaki, S. A. Hojjatoleslami, H. Sarmadi, R. Ebrahimpour, A. GH. Podoleanu, Proceeding of Iranian Conference on Electrical Engineering (ICEE) 2010, IEEE, **1,** 172 (2010).
[10] Mohammad R.N. Avanaki, S. A. Hojjatoleslami, A. Meadway, G. Dobre, A. Gh. Podoleanu, Conference Proceeding SPIE, **7904**, 790415-1 (2011).
[11] Mohammad R.N. Avanaki, R. Mazraeh Khoshki, S. A. Hojjatoleslami, and Adrian Gh. Podoleanu, Conference Proceeding SPIE, **8351**, 83510B (2012).
[12] A. Hojjatoleslami, M. R. Nasiriavanaki, Applied optics journal, **51**(21), 4927 (2012).

*Corresponding author: maria-alexandra.paun@epfl.ch