Simulated Annealing using Hybrid Monte Carlo

R. Salazar[1], R. Toral[1,2]

(1) Departament de Física, Universitat de les Illes Balears

(2) Instituto Mediterráneo de Estudios Avanzados (IMEDEA, UIB-CSIC)

07071 Palma de Mallorca, Spain.

June 6, 1997

Abstract

We propose a variant of the Simulated Annealing method for optimization in the multivariate analysis of differentiable functions. The method uses global actualizations via the Hybrid Monte Carlo algorithm in their generalized version for the proposal of new configurations. We show how this choice can improve upon the performance of simulated annealing methods (mainly when the number of variables is large) by allowing a more effective searching scheme and a faster annealing schedule.

KEYWORDS: Simulated Annealing. Hybrid Monte Carlo. Multivariate minimization.

1 Introduction

An important class of problems can be formulated as the search of the absolute minimum of a function of a *large* number of variables. These problems include applications in different fields such as Physics, Chemistry, Biology, Economy, Computer Design, Image processing, etc.[1]. Although in some occasions, such as the NP-complete class of problems[2], it is known that no algorithm can surely find the absolute minimum in a polynomial time with the number of variables, some very successful heuristic algorithms have been developed. Amongst those, the Simulated Annealing (SA) method of Kirkpatrick, Gelatt and Vecchi[3], has proven to be very successful in a broad class of situations. The problem can be precisely defined as finding the value of the N–dimensional vector $\mathbf{x} \equiv (x_1, x_2, ..., x_N)$, which is an absolute minimum of the real function $E(\mathbf{x})$. For large N, a direct search method is not effective due to the large configuration space available. Moreover, more sophisticated methods, such as downhill simplex or those using the gradient of $E(\mathbf{x})$[4], are likely to get stuck in local minima and, hence, might not able to reach the absolute minimum.

SA is one of the most effective methods devised to overcome these difficulties. It allows escaping from local minima through tunnelling and also by accepting higher values of $E(\mathbf{x})$ with a carefully chosen probability[3]. The method is based on an analogy with Statistical Physics: the set of variables $(x_1, \ldots, x_N)$ form the phase space of a fictitious physical system. The function $E(\mathbf{x})$ is considered to be the system's energy and the problem is reduced to that of finding the ground state configuration of the system. It is known that if a system is heated to a very high temperature T and then it is *slowly* cooled down to the absolute zero (a process known as annealing), the system will find itself in the ground state. The cooling rate must be slow enough in order to avoid getting trapped in some metastable state. At temperature T, the probability of being on a state with energy $E(\mathbf{x})$ is given by the Gibbs factor:

$$P(\mathbf{x}) \propto \exp(-E(\mathbf{x})/T). \tag{1}$$

From this relation we can see that high energy states can appear with a finite probability at high T. If the temperature is lowered, those high energy states become less probable and, as $T \to 0$, only the states near the minimum of $E(\mathbf{x})$ have a non–vanishing probability to appear. In this way, by appropriately decreasing the temperature we can arrive, when $T \to 0$, to the (absolute) minimum energy state. In practice, the method proceeds as follows: at each annealing step k there is a well defined temperature $T(k)$ and the system is let to evolve long enough such that it thermalizes at temperature $T(k)$. The temperature is then lowered according to a given annealing schedule $T(k)$ and the process is repeated until the temperature reaches $T = 0$.

To completely specify the SA method, one should give a way of generating representative configurations at temperature T, and also the variation of the temperature with annealing step, $T(k)$. For the generation of the configurations, the Monte Carlo method (MC) is widely used[5, 6, 7]. MC introduces an stochastic dynamics in the system by proposing configuration changes $\mathbf{x} \rightarrow \mathbf{x}'$ with probability density function (pdf) $g(\mathbf{x}'|\mathbf{x})$, i.e, if the system variables adopt presently the value $\mathbf{x}$, the probability that the new proposed value lies in the interval $(\mathbf{x}', \mathbf{x}' + d\mathbf{x}')$ is $g(\mathbf{x}'|\mathbf{x})d\mathbf{x}'$. This proposal is accepted with a probability $h(\mathbf{x}'|\mathbf{x})$. Much freedom is allowed in the choice of the proposal and acceptance probabilities. A sufficient condition in order to guarantee that the Gibbs distribution is properly sampled, is the detailed balance condition:

$$g(\mathbf{x}'|\mathbf{x})h(\mathbf{x}'|\mathbf{x})\exp(-E(\mathbf{x})/T) = g(\mathbf{x}|\mathbf{x}')h(\mathbf{x}|\mathbf{x}')\exp(-E(\mathbf{x}')/T). \tag{2}$$

Once the proposal pdf $g(\mathbf{x}'|\mathbf{x})$ has been conveniently specified, the acceptance probability $h(\mathbf{x}'|\mathbf{x})$ is given as a convenient solution of the previous detailed balance equation. Usually (see next section) the proposal probability $g(\mathbf{x}'|\mathbf{x}) = g(\Delta\mathbf{x})$ is a symmetric function of the difference $\Delta\mathbf{x} \equiv \mathbf{x}' - \mathbf{x}$, $g(\Delta\mathbf{x}) = g(-\Delta\mathbf{x})$ and a commonly used solution to the detailed balance equation is the Metropolis choice:

$$h(\mathbf{x}'|\mathbf{x}) = \min\left(1, \exp\left[-\left(E(\mathbf{x}') - E(\mathbf{x})\right)/T\right]\right), \tag{3}$$

although other solutions have been also widely used in the literature.

The various SA methods differ essentially in the choice of the proposal probability $g(\Delta\mathbf{x})$ and the annealing schedule $T(k)$. One can reason that the cooling schedule $T(k)$ might not be independent of the proposal probability $g(\Delta\mathbf{x})$, i.e. $T(k)$ should be chosen consistently with the selected $g(\Delta\mathbf{x})$ in such a way that the configuration space is efficiently sampled. In the next section we briefly review the main choices used in the literature. We mention here that most of them involve only the change of *one* single variable x_i at a time, i.e. they consist generally of small *local* moves. N of these local moves constitute what is called a Monte Carlo Step (MCS). The reason for using only local moves is that the acceptance probability given by (3) is very small if all the variables are randomly changed at once, because the change in energy $E(\mathbf{x}') - E(\mathbf{x})$ is an extensive quantity that scales as the number of variables N. Hence, the acceptance probability near a minimum of $E(\mathbf{x})$ becomes exponentially small. Since $\Delta\mathbf{x}$ is a small quantity, the cooling schedule must be consequently small, because a large cooling rate would not allow the variables to thermalize at the given temperature. It is then conceivable that the use of a global update scheme could improve upon the existing methods by allowing the use of larger cooling rates.

In this paper we investigate the effect of such a global update dynamics. Specifically, we use the

Hybrid Monte Carlo (HMC) algorithm[8] for the generation of the representative configurations at a given temperature. By studying some examples, we show that the use of this global dynamics allows quite generally an exponentially decreasing cooling schedule, which is the best one can probably reach with other methods. Another advantage of the use of the HMC is that the number of evaluations of the energy function $E(\mathbf{x})$ is greatly reduced. Finally, we mention that the use of a generalized HMC [9, 10] allows to treat efficiently minimization problems in which the range of variation is different for each variable.

The rest of the paper is organized as follows: in section II we briefly review some of the existing SA methods; in section III we explain how to implement Hybrid Monte Carlo in an optimization problem; in section IV we use some standard test functions to compare our method with previous ones; and in section V we end with some conclusions and outlooks.

2 Review of Simulated Annealing Methods

Amongst the many choices proposed in the literature, we mention the following:

-*Boltzmann Simulated Annealing* (BSA)[11]: Based on a functional form derived for many physical systems belonging to the class of Gaussian-Markovian systems, at each annealing step k the algorithm chooses a proposal probability given by local moves governed by a Gaussian distribution:

$$g(\Delta\mathbf{x}) \sim \exp\left[-\frac{|\Delta\mathbf{x}|^2}{2T(k)}\right]. \tag{4}$$

The Metropolis choice (3) is then used for the acceptance. This choice for the proposal probability and the use of purely local moves imply that the annealing schedule must be particularly slow: $T(k) = T_0/\ln(1+\lambda k)$, for some value of the cooling rate λ.

-*Fast Simulated Annealing* (FSA)[12]: States are generated with a proposal probability that has a Gaussian–like peak and Lorentzian long–range tails that imply occasional long jumps in configuration space. These eventual long jumps make FSA more efficient than any algorithm based on any bounded variance distribution (in particular, BSA). The proposal probability at annealing step k is a N–dimensional Lorentzian distribution:

$$g(\Delta\mathbf{x}) \sim T(k)(|\Delta\mathbf{x}|^2 + T(k)^2)^{-\frac{N+1}{2}}. \tag{5}$$

One of the most significant consequences of this choice is that it is possible to use a cooling schedule inversely proportional to the annealing step k, $T(k) = T_0/(1 + \lambda k)$, which is exponentially faster than the BSA.

-*Very Fast Simulated Reannealing* (VFSR)[13]: In the basic form of this method, the change $\Delta\mathbf{x}$ is generated using the set of random variables $\mathbf{y} \equiv (y_1, \ldots, y_N)$

$$\Delta x_i = (B_i - A_i) y_i, \tag{6}$$

(A_i and B_i are the minimum and maximum value of the i–th dimension range). The proposal probability is defined as

$$g(\mathbf{y}) = \prod_{i=1}^{N} \frac{1}{2(\mid y_i \mid + T_i(k)) \ln(1 + 1/T_i(k))}. \tag{7}$$

Notice that different temperatures $T_i(k)$ can be in principle used for the updating of different variables x_i. For the acceptance probability, one uses the Metropolis choice (3) with yet another temperature $T_0(k)$. This proposal allows the following annealing schedule: $T_i(k) = T_i(0) \exp(-\lambda_i k^{\frac{1}{N}})$, $i = 0, 1, \ldots, N$, which is not very efficient for large number of variables N. A more detailed description of the VFSR algorithm can be found in [13].

-*Downhill Simplex with Annealing* (DSA)[4]: This method combines the Downhill Simplex (DS) method (which is basically a searcher for local minima) with a Metropolis like procedure for the acceptance. The DS samples the configuration space by proposing moves of the "simplex". A simplex being a geometrical figure with $N + 1$ vertices in the N–dimensional phase space. The moves are usually reflections, expansions, and contractions. The acceptance part is implemented by adding logarithmically distributed random variables proportional to the temperature to the energy before the move and subtracting a similar random variable after the move. The move is accepted if the energy difference is negative. According to reference [4] different annealing schedules $T(k)$ should be used for different problems. In the implementation we have made of this method (see section IV) an exponential decay has been used.

3 Hybrid Simulated Annealing

The alternative method we propose –Hybrid Simulated Annealing (HSA)– uses the Hybrid Monte Carlo (HMC)[8] in their generalized version[9, 10] to generate the representative configurations.

We first review the HMC method.

In its simplest and original form, HMC introduces a set of auxiliary momenta variables $\mathbf{p} \equiv (p_1, \ldots, p_N)$ and the related Hamiltonian function $\mathcal{H}(\mathbf{x}, \mathbf{p})$:

$$\mathcal{H}(\mathbf{x}, \mathbf{p}) = E(x_1, \ldots, x_N) + \frac{1}{2} \sum_{i=1}^{N} p_i^2 = E(\mathbf{x}) + \mathbf{p}^2/2. \tag{8}$$

From the Gibbs factor:

$$P(\mathbf{x}, \mathbf{p}) \propto \exp[-\mathcal{H}(\mathbf{x}, \mathbf{p})/T] = \exp[-E(\mathbf{x})/T] \exp[-\mathbf{p}^2/2T], \tag{9}$$

we deduce that, from the statistical point of view, the momenta $\mathbf{p}$ are nothing but a set of independent, Gaussian distributed, random variables of zero mean and variance equal to the system temperature T. There is no simple closed form for the proposal probability $g(\mathbf{x}'|\mathbf{x})$, and the proposal change $\mathbf{x} \to \mathbf{x}'$ is done in the following way: first, a set of initial values for the momenta $\mathbf{p}$ are generated by using the Gaussian distribution $\exp[-\mathbf{p}^2/2T]$ as suggested by the equation (9); next, Hamilton's equations of motion, $\dot{x}_i = p_i$, $\dot{p}_i = F_i$, where $F_i(\mathbf{x}) = -\partial E(\mathbf{x})/\partial x_i$ is the "force" acting on the variable x_i, are integrated numerically using the *leap–frog* algorithm with a time step δt:

$$\begin{aligned} x_i' &= x_i + \delta t p_i + \frac{\delta t^2}{2} F_i(\mathbf{x}) && (10) \\ p_i' &= p_i + \frac{\delta t}{2}[F_i(\mathbf{x}) + F_i(\mathbf{x}')], \qquad && i = 1, \ldots, N. \end{aligned}$$

The proposal $\mathbf{x}'$ is obtained after n iterations of the previous basic integration step. In other words: by numerical integration of Hamilton's equations during a "time" $n\delta t$. The value $\mathbf{x}'$ must now be accepted with a probability given by:

$$h(\mathbf{x}'|\mathbf{x}) = \min\left(1, \exp\left[-\left(\mathcal{H}(\mathbf{x}', \mathbf{p}') - \mathcal{H}(\mathbf{x}, \mathbf{p})\right)/T\right]\right). \tag{11}$$

Notice that this acceptance probability uses the total Hamiltonian function $\mathcal{H}(\mathbf{x}, \mathbf{p})$ instead of simply the function $E(\mathbf{x})$ as in the methods of last section (compare (11) and (3))[14]. Although Hamilton's equations exactly conserve the energy $\dot{\mathcal{H}} = 0$, the difference $\Delta\mathcal{H} \equiv \mathcal{H}(\mathbf{x}', \mathbf{p}') - \mathcal{H}(\mathbf{x}, \mathbf{p})$ is not equal to zero due to the finite time step discretization errors and one has quite generally $\Delta\mathcal{H} = O(N\delta t^l)$ for some value of l. In this way, although the mapping is a global one, i.e. all the variables are updated at once, it is still possible to have an acceptance probability of order unity by properly choosing the time step δt and one can have large changes in phase space at a small cost in the Hamiltonian. Notice that the Hamiltonian difference $\Delta\mathcal{H}$ being small, does not necessarily imply that ΔE is small and once can in principle accept moves which imply a large change in the energy $E(\mathbf{x})$.

In order to generate configurations at temperature T, one still must satisfy the detailed balance condition, equation (2). One can prove that sufficient requirements for this detailed balance condition to hold are that the mapping given by eqs.(10) satisfies time reversibility and area preserving[16]. These two properties are exactly satisfied by Hamilton's equations and are also kept by the leap–frog integration scheme. Under those conditions, the Gibbs distribution (1) for the original variables $\mathbf{x}$ is properly sampled. It is possible to further generalize the HMC method by using more general mappings satisfying the conditions of time reversibility and area preserving. In reference [10] it was shown that those conditions were satisfied by the mapping induced by n iterations of the following basic step:

$$\begin{aligned} x_i' &= x_i + \delta t \sum_{j=1}^{N} A_{ij} p_j + \frac{\delta t^2}{2} \sum_{j,k=1}^{N} A_{ik} A_{jk} F_j(\mathbf{x}), \\ p_i' &= p_i + \frac{\delta t}{2} \sum_{j=1}^{N} A_{ji} [F_j(\mathbf{x}) + F_j(\mathbf{x}')], \qquad i = 1, \ldots, N, \end{aligned} \tag{12}$$

where A_{ij} is an arbitrary matrix. This mapping can be thought as the leap–frog numerical integration of the following equations of motion:

$$\begin{aligned} \dot{x}_i &= \sum_j A_{ij} p_j, \\ \dot{p}_i &= \sum_j A_{ji} F_j. \end{aligned} \tag{13}$$

An straightforward calculation shows that these equations, although not being Hamiltonian, still conserve energy, $\dot{\mathcal{H}} = 0$, and the main features mentioned above of the standard HMC method are still maintained. Convenient choices for matrix A_{ij} are: diagonal in Fourier space (Fourier acceleration), or a diagonal matrix: $A_{ij} = A_i \delta_{ij}$. This last choice allows an effective integration time step $\delta t_i = \mathcal{A}_i \delta t$ different for each variable (compare with (10)):

$$\begin{aligned} x_i' &= x_i + \delta t_i p_i + \frac{\delta t_i^2}{2} F_i(\mathbf{x}), \\ p_i' &= p_i + \frac{\delta t_i}{2} [F_i(\mathbf{x}) + F_i(\mathbf{x}')], \qquad i = 1, \ldots, N \end{aligned} \tag{14}$$

The possibility of using different time steps for each variable accounts for the fact that the range of variation might differ for each variable. This is the case, for instance, of Corana's function (see next section).

Summing up, the HMC proceeds by generating representative configurations by using a proposal obtained by some of the mappings given above. This proposal must now be accepted with a probability given by (11). In this paper, we have used mainly the basic mapping given by (10) except in one case (Corana's function) in which the mapping (14) has been used instead. The temperature must then be decreased towards zero as in other SA methods. Notice that in the case $T = 0$ the random component of the evolution (the momenta variables) in Eq.(10) is zero and then the proposal coincides with that of gradient methods.

The HMC has been extensively used in problems of Statistical Physics[17]. For our purpose here, we have found that the use of the previous Hamiltonian based global update of the statistical system associated with the energy $E(\mathbf{x})$, allows a much more effective annealing schedule and searching scheme than, for instance, the Boltzmann, Fast annealing and Very Fast Reannealing methods mentioned above. In particular we have been able to use quite generally an exponential annealing schedule: $T(k) = T_0 e^{-\lambda k}$. Moreover, since in HMC the acceptance decision is taken after all the N variables have been updated, the number of energy function evaluations is greatly reduced. This turns out to be important in those problems in which the calculation of the energy function $E(\mathbf{x})$ takes comparatively a large amount of computer time.

4 Results

In order to compare our algorithm with the different ones proposed in the literature, we have used a set of five test functions: a multidimensional paraboloid, a function from De Jong's test[18], Corana's highly multi–modal function[19] and two other functions with many local minima. We now define and describe in some detail these functions.

The first function, $f_1(\mathbf{x})$, is a N–dimensional paraboloid:

$$f_1(\mathbf{x}) = \sum_{i=1}^{N} x_i^2. \tag{15}$$

Here we use the test value $N = 200$ and to compare with the results in [20], we also use the value $N = 3$. Although this is a particularly simple function with a single minimum $f_1 = 0$ located at $x_i = 0$, $i = 1 \dots N$, it ultimately describes the late stages of the behaviour of the SA algorithm when we are near a local or global minimum of any differentiable function.

The second function, $f_2(\mathbf{x})$, is a two dimensional ($N = 2$) function taken from De Jong's test typically used for benchmarking Genetic Algorithms[18]:

$$f_2(\mathbf{x}) = \left[0.002 + \sum_{j=1}^{25} [j + (x_1 - a_j)^6 + (x_2 - b_j)^6]^{-1} \right]^{-1}, \tag{16}$$

where the vectors a, b have the following 25 components:

$$\begin{aligned} a_j &= \{-32, -16, 0, 16, 32, -32, -16, 0, 16, 32, \ldots, -32, -16, 0, 16, 32\}, \\ b_j &= \{-32, -32, -32, -32, -32, -16, -16, -16, -16, -16, \ldots, 32, 32, 32, 32, 32\}, \end{aligned}$$

this function has 25 local minima, and the global minimum is $f_2 = 0.998004$, at $x_1 = x_2 = -32$.

The $f_3(\mathbf{x})$ function is the Corana's function:

$$f_3(\mathbf{x}) = \sum_{i=1}^{N} \begin{cases} 0.15\ (0.05\ \mathrm{sgn}(z_i) + z_i)^2\ d_i & \text{if } |x_i - z_i| < 0.05 \\ d_i x_i^2 & \text{otherwise} \end{cases} \tag{17}$$

$$z_i = 0.2\ \lfloor |5\ x_i| + 0.49999 \rfloor \mathrm{sgn}(x_i) \tag{18}$$

d_i is an N–dimensional vector. In our tests (and following [20] we have used $N = 10$ and $d = (1, 1000, 10, 100, 1, 1000, 10, 100, 1, 1000)$. This function, which has many local minima and is discontinuous and piecewise differentiable, turns out to be one of the most difficult test functions, because the different variables have different scales of variation. The global minimum is $f_3(\mathbf{x}) = 0$, at $x_i = 0$, $i = 1 \ldots N$.

The $f_4(\mathbf{x})$ function is defined by:

$$f_4(\mathbf{x}) = \frac{1}{2N} \sum_{i=1}^{N} \frac{\sin(4\pi K x_i)}{\sin(2\pi x_i)}, \tag{19}$$

with $N = 200$, $K = 2$. This function is periodic and has $(2K - 1)^N$ local minima per period. The absolute minima are at $x_i = (2m + 1)/2$, $m \in \mathbf{Z}$, $i = 1 \ldots N$, and the minimum value is $f_4(\mathbf{x}) = -K$ (see figure 1).

And, finally, the $f_5(\mathbf{x})$ function is defined by:

$$f_5(\mathbf{x}) = \sum_{i=1}^{N} |x_i|^\alpha - \prod_{i=1}^{N} \cos(4\pi x_i), \tag{20}$$

with $N = 10$ and $\alpha = 1.3$. Again, this function has many local minima. The absolute minimum is $f_5 = -1$ at $x_i = 0$, $i = 1 \ldots N$.

We present results of the optimization of these typical test functions performed with the methods described above: Fast Simulated Annealing (FSA), Very Fast Simulated Reannealing (VFSR), Downhill Simplex with annealing (DSA) and the Hybrid Simulated Annealing (HSA). Amongst other quantities, we have focused, as usual in this field, on the number of evaluations of the function and the CPU time needed to achieve a given accuracy ϵ in the minimum value of each function. These minimum values being exactly known for the test functions used. The results are summarized in tables (1) and (2) after averaging over 10 realizations. An accuracy value of $\epsilon = 10^{-3}$ has been used, although similar results hold for other values of ϵ. We have programmed the algorithms for the FSA, DSA and HSA methods, whereas the results for VFSR have been taken directly from [20]. For a given test function, we have used the same initial condition, $\mathbf{x}_{initial}$, for each method. As a general trend, we can see that HSA performs better than the other methods when the number of variables N is large. This does not imply that HSA performs extremely worse for small values of N. An important advantage of HSA in front of other methods is that the number of function evaluations is much smaller (in table (1) the number of function evaluations includes also the calculation of the forces necessary in the HSA method). This might turn out to be very important in those problems in which the function evaluation takes a long computer time. We now report in some detail the results of each test function:

As mentioned before, the f_1 function, a parabolic function with a single minimum, serves to model the behavior close to a minimum of any function, i.e. the situation for low enough temperature. When the number of variables is small, $N = 3$, it turns out that the fastest method (in the sense that it reaches the minimum in less computer time) is DSA although HSA needs less function evaluations. However, when the number of variables is large, $N = 200$, the cost in CPU time and number of function evaluations is very favorable to HSA. In general, the performance of the DSA method worsens when the problem has many minima. This is obvious when looking at the results for the De Jong's f_2, the Corana f_3 and the f_4 functions for which the DSA could not even find the absolute minimum.

The f_2 function is another example in which the HSA can not offer a better alternative than other methods, stressing the fact again that for small number of variables the use of a global actualization turns out to be irrelevant. In this case, VFSR needs less number of function evaluations than any other method. However, for large number of variables N, the cooling schedule required for VFSR is necessarily slow (see the discussion in section 2) making it inefficient for large N.

The functions in which the variables have a wide range of variation (for instance Corana's function f_3) can be better handled using the general version of HSA. Remember that the rescaling in (14) allows an effective integration time step $\delta t_i = \mathcal{A}_i \delta t$ different for each variable. So, one can tune A_i to solve efficiently this kind of problems. In our case, the range of variation of the variables come essentially from the part $V(\mathbf{x}) = d_i x_i^2$ of the Corana's function. Then, from the equations of motion $\dot{x}_i = A_i p_i$ and $\dot{p}_i = A_i F_i$ we have $\ddot{x}_i = A_i^2 F_i$. The force is $F_i = -2\ d_i x_i$ and we have $\ddot{x}_i = -A_i^2 d_i x_i$, so we chose $A_i = \frac{1}{\sqrt{d_i}}$ in order that each variable has the same effective time scale for evolution.

The f_4 and f_5 functions have the feature of possessing a large number of minima (for example, f_4 has $(2K-1)^N$ local minima in a period). The results show again that HSA is a much better alternative when the number of variables is large, both from the point of view of CPU time used or the number of function evaluations. We have chosen the f_4 function to compare in figure (2) the evolution of the minimum value of the function with the actual number of function evaluations, for both the FSA and HSA methods, showing again in a different manner that HSA can find a better minimum with a less number of function evaluations. From the results for these functions we infer that in minimization problems with a large number of variables and a large number of local minima, the HSA has the best performance. Needless to say, we have made our best effort to use the optimal values for the parameters in each method. It is possible, though, that these values could be further improved and the results of tables (1) and (2) slightly modified. We believe, though, that this will not affect the main conclusions of this paper.

5 Conclusions

We have shown by some examples how the use of the global update using Hybrid Monte Carlo algorithm can indeed improve the performance of simulated annealing methods. The global updating implicit in HSA allows an effective searching scheme and fast annealing schedules and becomes highly effective, mainly in those problems with a large number of variables and a large number of metastable minima.

It is clear from the results in the previous section that HSA requires in some cases orders of magnitude less evaluations of the function than other methods and can, therefore, give a solutions in less computer time. This conclusion remains despite the fact that HSA requires some extra work when computing the evolution equations since it needs to compute also the forces F_i acting on the different variables. In those cases in which the evaluation of the function takes a considerable amount of computer time, HSA will have an optimal performance, since the number of function evaluations is greatly reduced as compared to other simulated annealing methods. It is conceivable also that one could then use efficiently some of the acceleration schemes (Fourier, wavelet, etc.) available for Monte Carlo methods in order to improve upon the convergence of the simulated annealing techniques. Further developments include applying HSA to techniques such as the Car–Parrinello method for finding the ground state of quantum many body systems, for which the calculation of the energy function is very time consuming. Work on this direction is under progress.

ACKNOWLEDGEMENTS

We wish to thank J. Rubio and M. Planas for several discussions. We acknowledge financial support from DGICyT, grants PB94-1167 and PB94-1172. R. Salazar is supported by the Agencia Española de Cooperación Internacional in the Mutis program.

References

[1] P.J.M. van Laarhoven and E.H.L. Aarts. *Simulated Annealing: Theory and Applications.* Kluwer Academic Publishers, Netherlands, 1988.

[2] M.R.Garey and D.S.Johnson. *Computers and Intractability: A Guide to the Theory of NP–Completeness.* Freeman, San Francisco, 1979.

[3] S. Kirkpatrick, Jr. C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[4] W.T. Vetterling W.H. Press, S.A. Teukolsky and B.P. Flannery. *Numerial Recipes in FORTRAN: the art of scientific computing.* Cambridge University, New York, 1994.

[5] M.H. Kalos and P.A. Whitlock. *Monte Carlo Methods.* John Wiley and Sons., New York, 1986.

[6] D. Heermann. *Computer Simulation Methods.* Springer Verlag, Berlin, Heidelberg, 1986.

[7] K. Binder, editor. *The Monte Carlo Method in Condensed Matter Physics.* Springer Verlag, Berlin, Heidelberg, 1992.

[8] S. Duane, A.D. Kennedy, B.J. Pendelton, and D. Roweth. Hybrid monte carlo. *Phys. Lett. B*, 195:216–222, 1987.

[9] R. Toral and A.L. Ferreira. Generalized Hybrid Monte–Carlo. In editors R. Gruber, M. Tomasini, editor, *Proceedings of the Conference Physics Computing'94*, page 265. European Physical Society, Geneva, Switzerland, 1994.

[10] A.L. Ferreira and R. Toral. Hybrid monte carlo method for conserved–order–parameter systems. *Phys. Rev. E*, 47:R3848–R3851, 1993.

[11] S. German and D. German. Stochastic relaxation, gibbs distribution and the bayesian restoration in images. *IEEE Trans. Patt. Anan. Mach. Int.*, 6:721–741, 1984.

[12] H. Szu and R. Hartley. Fast simulated annealing. *Phys. Lett. A*, 3–4:157–162, 1987.

[13] L. Ingber and B. Rosen. Genetic algorithms and very fast simulated reannealing: A comparison. *Mathl. Comput. Modelling*, 16:87–100, 1992.

[14] A similar procedure is used in [15] in the context of sampling Tsallis statistics. In their method, the configuration energy difference $\Delta E(\mathbf{x})$ is used instead of the total Hamiltonian difference

$\Delta\mathcal{H}(\mathbf{x})$ as the acceptance criterion. In general, this procedure does not properly sample the Gibbs distribution, Eq.(1), at temperature T, although it coincides with the one used in this paper as $T \to 0$.

[15] I. Andricioaei and J.E. Straub. Generalized simulated annealing algorithms using tsallis statistics: Application to conformational optimization of a tetrapeptide. *Physical Review E*, 53:R3055–R3058, 1996.

[16] Time reversibility implies that the original coordinates are exactly recovered after numerical integration during a time step if the momenta are reversed. Area preserving implies that the Jacobian of the mapping (10) is equal to one.

[17] R. Toral. Computational Field Teory and Pattern Formation. In P.L. Garrido and J. Marro, editors, *3rd Granada Lectures in Computational Physics*, page 1. Springer-Verlag, Heildelberg, 1995.

[18] K. A. De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive system.* PhD thesis, University of Michigan, 1981.

[19] A. Corana, M. Martini, and S. Ridella. Minimizing multimodal functions of continuous variables with the simulated annealing algorithm. *ACM Trans. on Mathematical Software*, 13:272–280, 1987.

[20] S.G. Dykes and B.E. Rosen. Parallel very fast simulated reannealing by temperature block partitioning. In *IEEE International Conference on Systems, Man, and Cybernetics. Humans, Information and Technology*, pages 1914–19. IEEE press, New York, USA, 1994.

Function	*Dimension*	*FSA*	*VFSR*	*DSA*	*HSA*
f_1	3	480	4875	79	18
f_1	200	8420000	–	474000	30
f_2	2	9900	1476	(*)	165000
f_3	10	2100000	319483	(*)	720000
f_4	200	12925000	–	(*)	163000
f_5	10	7230000	–	570000	118000

Table 1: Number of function evaluations averaged over 10 realizations, for each of the simulated annealing methods used for optimization of the different functions to reach the absolute minimum with an accuracy of $\epsilon = 10^{-3}$. In those cases marked (*) it was not possible to reach the absolute minimum. For the HSA, the displayed number is the number of function evaluations including the calculations of the force.

Function	*Dimension*	*FSA*	*DSA*	*HSA*
f_1	3	0.023	0.003	0.021
f_1	200	182.898	163.763	0.039
f_2	2	0.181	(*)	5.834
f_3	10	29.454	(*)	11.730
f_4	200	1662.177	(*)	61.863
f_5	10	119.434	13.929	4.00

Table 2: Similar to table (1) but showing the CPU time (in seconds) needed to reach the absolute minimum with an accuracy $\epsilon = 10^{-3}$ for each of the simulated annealing methods explained in the text. All the programs were run on a Silicon Graphics Origin200 (CPU: R10000 running at 180 MHz, Speed: 15.5 SPECfp95).

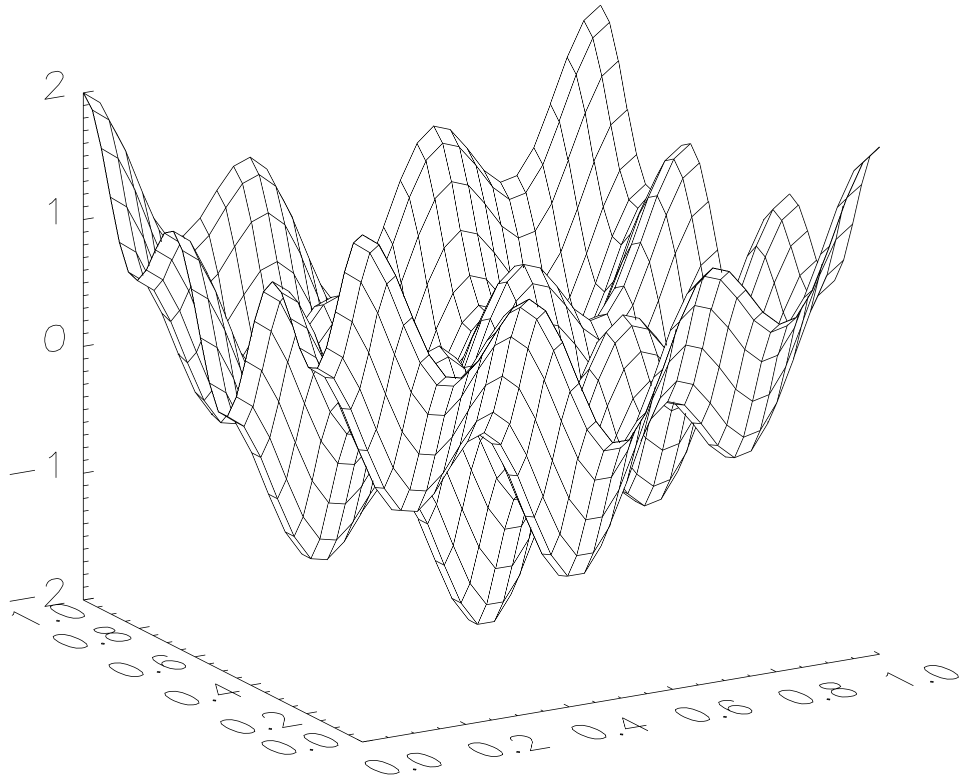

Figure 1: Plot of f_4 function, equation (19) for $N = 2$, in one period. Notice the presence of many relative minima, but only one absolute minimum at $x_1 = x_2 = 0.5$.

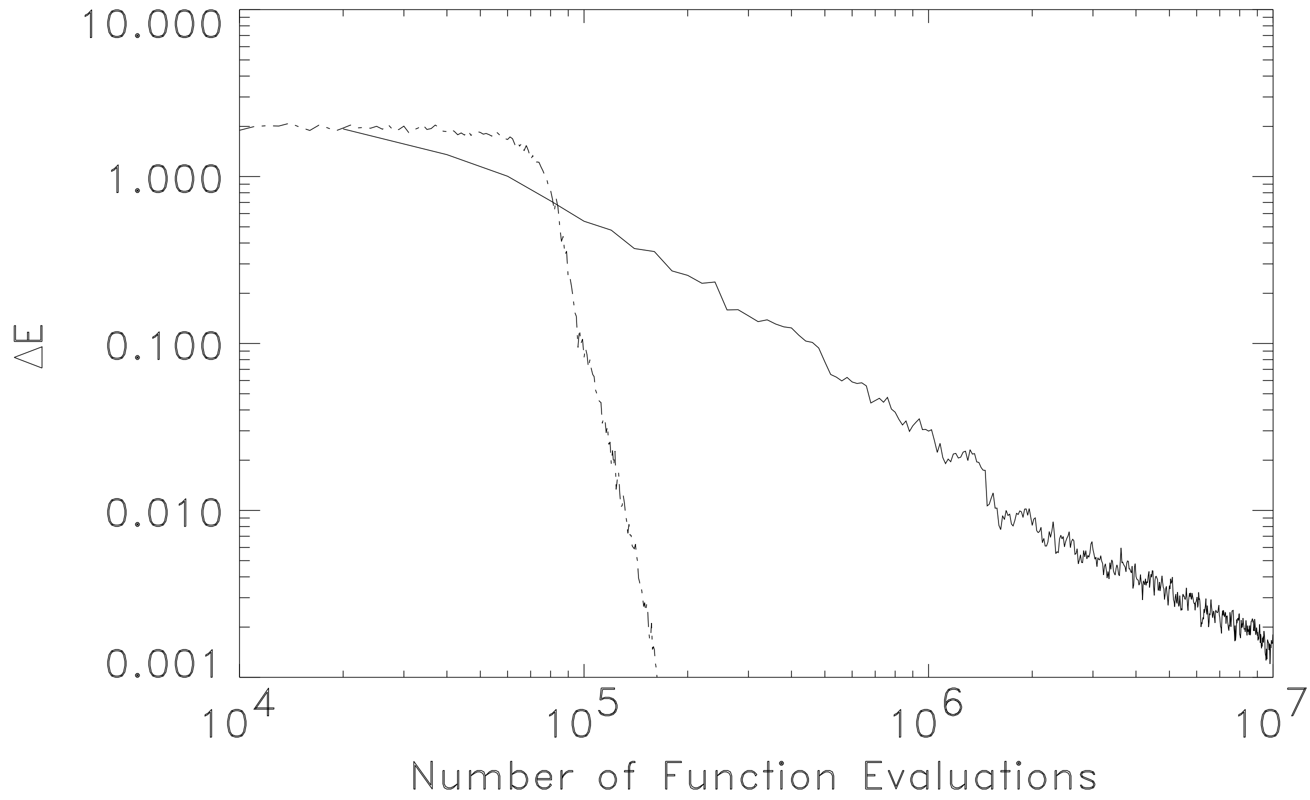

Figure 2: Plot of "energy" difference with respect to the ground state value, versus the number of function evaluations, for the f_4 function (19) with $N = 200$ using HSA (dotted line) and FSA (continuous line), both initialized in $\mathbf{x}_{initial} = 1.0$, the other parameters have the following values, for FSA: $T_0 = 0.8$, $m = 100$, $\lambda = 100$; and for HSA: $T_0 = 1.0$, $m = 10$, $n = 10$, $\delta t = 0.3$ $\lambda = 0.007$, where m is the number of MCS used for thermalization at temperature $T(k)$