Enhanced Simulated Annealing for Globally Minimizing Functions of Many-Continuous Variables

PATRICK SIARRY
Ecole Centrale de Paris
and
GÉRARD BERTHIAU, FRANÇOIS DURBIN, and JACQUES HAUSSY
C.E.A.

A new global optimization algorithm for functions of many continuous variables is presented, derived from the basic Simulated Annealing method. Our main contribution lies in dealing with high-dimensionality minimization problems, which are often difficult to solve by all known minimization methods with or without gradient. In this article we take a special interest in the variables discretization issue. We also develop and implement several complementary stopping criteria. The original Metropolis iterative random search, which takes place in a Euclidean space $\mathbb{R}^n$, is replaced by another similar exploration, performed within a succession of Euclidean spaces $\mathbb{R}^p$, with $p << n$. This Enhanced Simulated Annealing (ESA) algorithm was validated first on classical highly multimodal functions of 2 to 100 variables. We obtained significant reductions in the number of function evaluations compared to six other global optimization algorithms, selected according to previously published computational results for the same set of test functions. In most cases, ESA was able to closely approximate known global optima. The reduced ESA computational cost helped us to refine further the obtained global results, through the use of some local search. We have used this new minimizing procedure to solve complex circuit design problems, for which the objective function evaluation can be exceedingly costly.

Categories and Subject Descriptors: G.1.6 [**Numerical Analysis**]: Optimization—*nonlinear programming*; G.3 [**Mathematics of Computing**]: Probability and Statistics—*probabilistic algorithms*; G.4 [**Mathematics of Computing**]: Mathematical Software—*certification and testing*

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Global optimization, stochastic optimization, test functions

Authors' addresses: P. Siarry, Laboratoire Electronique et Physique Appliquée, Ecole Centrale de Paris, Grande Voie des Vignes, Châtenay-Malabry 92295, France. email: siarry@epap.ecp.fr; G. Berthiau, Département Electronique et Instrumentation Nucléaire, C.E.A., C.E.N. de Saclay, Gif-sur-Yvette 91191, France; F. Durbin and J. Haussy, Service Electronique, C.E.A, Centre U, B.P. 12, Bruyères-le-Châtel, 91680, France.

1. INTRODUCTION

The problem of finding the minimum of a given function of n variables has received much attention for many years. Several algorithms that give good results for unimodal functions have been proposed. Direct methods, such as Hooke and Jeeves' [1961] or Nelder and Mead's [1965] methods (see also Barabino et al. [1980]), involve only function evaluations. Others belonging to the Davidon-Fletcher-Powell family [Fletcher and Powell 1963; Fletcher and Reeves 1964] try to improve the minimization efficiency by using derivative evaluations of the cost function. However, those local search methods do not work satisfactorily when the cost function is multimodal within the domain of interest, because they tend to stop at the first minimum encountered. In many research and development fields, like VLSI design, engineers have to deal with minimization problems of ever-increasing dimensionality, leading to a combinatorial-like explosion of the number of local minima. For handling such global optimization problems, new approaches have recently emerged, some of them inspired by physical phenomena, in particular, Simulated Annealing [Kirkpatrick et al. 1983], genetic algorithms [Goldberg 1989], neural networks [Hopfield and Tank 1985], and tabu search [Glover 1977].

We propose a new global optimization algorithm for high-dimensional continuous problems, derived from the basic Simulated Annealing (SA) method. The founding principles of Simulated Annealing [Kirkpatrick et al. 1983; Metropolis et al. 1953] are now well known [Aarts et al. 1987; Cerny 1985; Siarry et al. 1987], and their efficiency has been demonstrated for the solution of difficult problems, such as the Traveling Salesman Problem. The Metropolis algorithm allows all SA-based methods to estimate first (at high temperature) the coarse-grain behavior of the function under test and then to examine (at low temperature) its finer-grain behavior. Our investigation has been primarily motivated by high-dimensionality problems, such as complex analog circuit optimization. In this case, the objective functions to be minimized can only be evaluated through costly circuit simulation runs [Durbin et al. 1990; 1991]. Moreover, derivative estimation can produce inaccurate and unreliable numerical values in the case of local nonsmoothness and subsequently can lead many gradient-like optimization methods toward erroneous results. Thus basic SA, which is already a nongradient and robust method, appears to be an adequate candidate for further enhancement in order to solve complicated problems when only the objective function is available.

Previous work on minimizing analytical/numerical functions of many variables, defined over a continuum in $\mathbb{R}^n$, was published during the last few years (see, in particular, Catthoor [1988], Corana [1987], and Vanderbilt and Louie [1984]), and several authors proposed various discretization step control schemes. We propose here an effective step control strategy involving a balanced use of large steps (coarse-grain function behavior) and short steps (fine-grain function behavior) for each of the function's n

variables. Moreover, we show how the discretization control scheme is related to the temperature-decreasing strategy.

We are interested in solving higher-dimension optimization problems characterized by R. Bellman's "curse of dimension." If n, the problem dimension, is large enough, prohibitive CPU times are obtained when numerous geometric moves in $\mathbb{R}^n$ are performed, each move involving one n-dimensional objective function evaluation. In order to attenuate to some extent this difficulty, we have replaced the original Metropolis iterative random search in $\mathbb{R}^n$ by another similar exploration, performed this time within a succession of $\mathbb{R}^p$ spaces with $p << n$. Next, we show how successive p-variables subsets are taken from the n-variables original set. Then we detail the selection rules, yielding the greatest possible number of different p-variables subsets, each of them being generated with uniform probability. We also propose several complementary stopping criteria that contribute efficiently toward reducing the number of objective function evaluations.

Finally, in Section 4, we demonstrate the validity of our approach, by minimizing some classical highly multimodal functions. Numerical results with corresponding numbers of objective function evaluations are given. They are compared with those originating from various algorithms operating inside the same hyperrectangular $\mathbb{R}^n$ box. In some cases, the Nelder and Mead simplex algorithm [1965] is used as a local optimization method. We propose to use it as a natural continuation to the Enhanced Simulated Annealing (ESA) algorithm, considered as a global search method. This extra optimizing stage yields more accurate results, while keeping the number of objective function evaluations at a reasonable level.

2. GENERAL SETTING OF THE METHOD

$\mathbf{X}$ is a vector in n-dimensional space $\mathbb{R}^n$, with components $(x1, x2, \ldots, x_n)$ satisfying hyperrectangular box constraints, i.e.,

$$\text{X0MIN}\ (k) \leq x_k \leq \text{X0MAX}\ (k),\ k = 1, \ldots, n.$$

Those inequalities geometrically define the variation domain $\mathbb{D}^n$ for $\mathbf{X}$. We are mainly concerned with minimizing bounded functions that admit a Lipschitz constant K. Thus, the following inequality holds for f at any points $\mathbf{X}$, $\mathbf{Y}$ in $\mathbb{D}^n$:

$$|f(\mathbf{X}) - f(\mathbf{Y})| \leq K||\mathbf{X} - \mathbf{Y}||.$$

The notation $||$ stands for any suitable Euclidean norm. In what follows, the objective function f is designated by FOBJ.

SA principles are now well understood. Moreover, they have received various implementations, which have produced interesting numerical results. Consequently, we give only a quick synthetic description of our algorithm (see Figure 1). Topics that we worked specifically on are now pointed out. Details are given in Section 3.

Starting from an initial point **XINIT**, SA generates a succession of points $\mathbf{X}_1$, $\mathbf{X}_2$,. . . in suitably chosen $\mathbb{R}^p$ subspaces taken from $\mathbb{R}^n$. A new point is generated from the current point **XSTART** by making uniform random moves along each of the p coordinate directions that have been retained by the partitioning algorithm (see Section 3).

The maximum amplitude of those moves is given by the step vector **STPMST**. The n components of **STPMST** are updated at each new temperature stage, following a specific rule (see Section 3). If one or more coordinates of the generated point fall outside the boundaries defined above, we generate one or more symmetrical coordinates with respect to the current point **XSTART**. This is an approach less costly than recomputing all p coordinates of a new point **XTRY**. The question of the initial choice of the step vector **STPMST** will be discussed further.

A potential point **XTRY** is submitted to the Metropolis criterion for acceptance:

```
Let $\Delta f = f(\mathbf{XTRY}) - f(\mathbf{XSTART})$
If $\Delta f \leq 0$, then $\mathbf{XSTART} \leftarrow \mathbf{XTRY}$
else
accept $\mathbf{XTRY}$ with probability $P(\Delta f, T) = \exp(-\Delta f / T)$;
   $T =$ temperature parameter
```

Therefore, SA allows uphill moves under the control of the T temperature parameter. It is clear that, at higher temperature, only the coarse-grain behavior of the objective function f is relevant to the search. As the temperature decreases, the finer-grain behavior of f is examined, finally leading to a local or global minimum of f.

The SA algorithm starts at temperature TMPINI, which is deduced from some user-chosen initial acceptance probability, PROBOK, for uphill moves. In order to estimate the FOBJ initial variation average, DGYINI, for uphill moves, one performs about 50-100 random moves before starting the SA process itself. TMPINI is then computed according to the following formula:

$$\text{PROBOK} = \exp(-\text{DGYINI}/\text{TMPINI})$$

Typically PROBOK is equal to 0.5.

The sequence of points generated by SA at the fixed temperature TMPINI is stopped as soon as the thermodynamical equilibrium, detected by some adequate condition, is reached. Then the temperature and step vector coordinates are suitably adjusted, before beginning a new sequence of moves, and so on. This process is finally stopped when at least one of four independent stopping tests on the number of downhill moves, temperature, step vector, and FOBJ evaluations is satisfied. The three first tests are basically equivalent in the sense that they all aim at terminating the

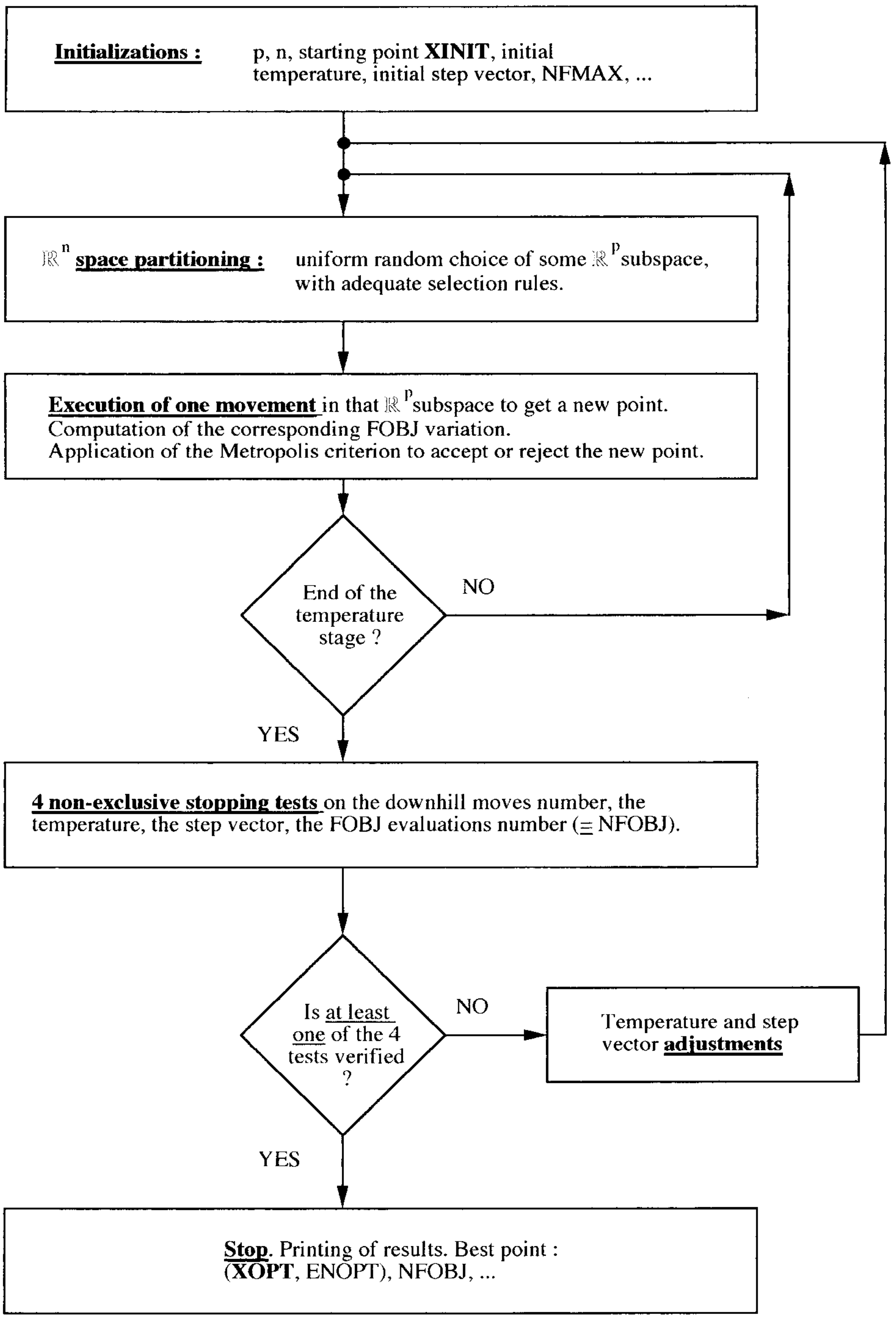
Initializations : p, n, starting point **XINIT**, initial temperature, initial step vector, NFMAX, ...
$\mathbb{R}^n$ **space partitioning :** uniform random choice of some $\mathbb{R}^p$ subspace, with adequate selection rules.
Execution of one movement in that $\mathbb{R}^p$ subspace to get a new point. Computation of the corresponding FOBJ variation. Application of the Metropolis criterion to accept or reject the new point.
End of the temperature stage ?
NO
YES
4 non-exclusive stopping tests on the downhill moves number, the temperature, the step vector, the FOBJ evaluations number (≡ NFOBJ).
Is at least one of the 4 tests verified ?
NO
Temperature and step vector **adjustments**
YES
Stop. Printing of results. Best point : (**XOPT**, ENOPT), NFOBJ, ...

Fig. 1. The ESA algorithm.

annealing process through the estimation of FOBJ local behavior: SA search is stopped if FOBJ is sufficiently flat in the neighborhood under exploration, as detected by at least one of the tests.

A detailed description of all the above-mentioned rules or tests follows in Section 3.

3. DETAILED DESCRIPTION OF THE ESA ALGORITHM

Step 1: Initializations

Choose:

—An objective function f = FOBJ involving the n variables: x_k, $k = 1,\ldots, n$

—Lower and upper bounds of each variable x_k: X0MIN(k), $k = 1,\ldots, n$ and X0MAX (k), $k = 1,\ldots, n$

—A starting point **XINIT** (possibly at the center of the hyperrectangular above-defined box)

—End of temperature stage parameters $N1$ and $N2$

—The type of normalization of each variable x_k: INORM(k), $k = 1,\ldots, n$ (INORM(k) = 1 for a linear normalization of x_k in [–1, 1] and INORM(k) = 2 for a logarithmic normalization of x_k in [–1, 1]; see the exact definition of normalization types at the beginning of Section 4)

—Three stopping tests parameters EPSREL, EPSABS, and NFMAX (typically EPSREL = 1.0D-06, EPSABS = 1.0D-08, and NFMAX = 5000: see the discussion at the end of this section; EPSREL and EPSABS are related to floating-point precision and are problem independent)

Note. The following other input parameters—which are subsequently defined in Steps 2 to 9—have been fixed by default inside the program (they can be easily changed by the user if the necessity arose after the first tests are performed on a new problem):

—Size p (NVPAR) of successive subsets to be taken from the n-variables (NVALL) original set

—Initial acceptance probability, PROBOK, for uphill moves

—Initial seed, ISEED, for the pseudorandom numbers generator

—End of temperature stage parameters $N1$ and $N2$

—Temperature adjustment rule parameters RMXTMP and RMITMP

—Step vector adjustment rule parameters RATMAX, RATMIN, EXTSTP, and SHRSTP

For details about the function and the typical value of these last eight parameters, see Step 5 for $N1$ and $N2$, Step 6 for RMXTMP and RMITMP, and Step 7 for RATMAX, RATMIN, EXTSTP, and SHRSTP.

Compute:

—Initial FOBJ value: EINIT = f(**XINIT**)

—Initial step vector **STPINI**: STPINI(k) = (X0MAX(k) – X0MIN(k)) * ROSTEP(k), k = 1,...,n (typical value of ROSTEP (k): 0.25)

—Initial FOBJ variation average, DGYINI, obtained from (typically) 50 uphill moves performed before beginning SA minimization itself.

—Initial temperature TMPINI computed from PROBOK = exp (–DGYINI/ TMPINI)

Set:

—Initial number of FOBJ evaluations: NFOBJ = 1

—Initial optimal point: **XOPT** = **XINIT**; ENOPT = EINIT

—**XSTART** = **XINIT**; OLDRGY = EINIT

—temperature: TEMP = TMPINI; step vector: **STPMST** = **STPINI**

—Initial number of accepted moves at current temperature stage: MVOKST = 0

—Initial vector of numbers of accepted moves (one number for each variable) at current temperature stage: **MOKST** = **0**

—Initial number of accepted uphill moves at current temperature stage: NMVUST = 0

—Initial number of attempted moves at current temperature stage: NMVST = 0

—Initial vector of numbers of attempted moves (one number for each variable) at current temperature stage: **MTOTST** = **0**

—Initial minimal FOBJ value at current temperature stage: ELOWST = EINIT

—Initial sum of successive FOBJ values at current temperature stage: AVGYST = 0

—Initial sum of accepted uphill FOBJ variations at current temperature stage: SDGYUP = 0

Step 2: $\mathbb{R}^n$ *Space Partitioning*

—Generate one p-variables subset, taken from the n-variables set.

—p-variables subsets, successively generated at the present step, are built according to the following selection rules:

variable $x_k (1 \leq k \leq n)$ is selected uniformly and

```
    randomly among the $n$ available variables;
  variable $x_k$ is effectively included into a
    $p$-variables subset if all other variables have already
    been retained at least the same number of times as $x_k$
```

—These p-variables sets selection rules make sure that one obtains the greatest possible number of different p-variables sets, without overselecting any particular x_k.

Step 3: Execution of One Movement

—x_k is denoted XSTART(k) in the program.

—For each variable XSTART(k) belonging to the p-variables subset selected above, compute XTRY(k), according to the formula

$$\mathrm{XTRY}(k) = \mathrm{XSTART}(k) \pm \mathrm{XRAND} * \mathrm{STPMST}(k)$$

where XRAND is a number randomly and uniformly generated in the range [0, 1] by a pseudorandom number generator. The sign $\pm$ is first randomly selected. If XTRY(k) falls outside the range [X0MIN(k), X0MAX(k)], then the opposite sign is retained.

—Those elementary operations on only p coordinates move the point **XSTART** to the point **XTRY** satisfying all hyperrectangular box constraints.

Step 4: Acceptance or Rejection of that Movement

—Δf and f(**XTRY**) are respectively denoted DELTAE and RNEWGY in the program.

—Compute DELTAE = RNEWGY – OLDRGY

—Add 1 to MTOTST(k), for all k corresponding to moved components of **XSTART**

—Add 1 to NMVST and to NFOBJ

—Set AVGYST = AVGYST + RNEWGY

```
If DELTAE $\leq$ 0, then
  accept XTRY
  update ENOPT: if RNEWGY < ENOPT, then set:
    XOPT = XTRY; ENOPT = RNEWGY
  update ELOWST: if RNEWGY < ELOWST, then set:
    ELOWST = RNEWGY
Else
  accept XTRY with probability exp(−DELTAE/TEMP)
  if XTRY accepted, then add 1 to NMVUST;
```

```
        set SDGYUP = SDGYUP + DELTAE
    Endif
```

—In cases of acceptance: **XSTART** = **XTRY**; OLDRGY = RNEWGY; add 1 to MOKST(k), for all k corresponding to moved components of **XSTART**, and to MVOKST

Step 5: Test for End of Temperature Stage

```
If [MVOKST < $N1*n$.and.(NMVST < $N2*n$)], then
    go to Step 2
Else
    end of temperature stage
Endif
```

Typical values of $N1$ and $N2$:

$N1 = 12$
$N2 = 100$

Step 6: Temperature Adjustment

```
Compute
    AVGYST = AVGYST/NMVST
    RFTMP = Max [Min [ELOWST/AVGYST, RMXTMP], RMITMP]
    TEMP = RFTMP * TEMP
```

Typical values of RMXTMP and RMITMP:

RMXTMP = 0.9
RMITMP = 0.1

The temperature adjustment rule is discussed at the end of this section.

Step 7: Step Vector Adjustment

```
Compute ROK($k$) = MOKST($k$)/MTOTST($k$), $k = 1,\ldots,n$
If ROK($k$) > RATMAX), then
    STPMST($k$) = STPMST($k$) * EXTSTP
Else if (ROK($k$) < RATMIN), then
    STPMST($k$) = STPMST($k$) * SHRSTP
Endif
```

Typical values of RATMAX, RATMIN, EXTSTP, and SHRSTP:

RATMAX = 0.2
RATMIN = 0.05
EXTSTP = 2
SHRSTP = 0.5

The step vector adjustment rule is discussed at the end of this section.

Step 8: Four Nonexclusive Stopping Tests

If at least one of the four following conditions is verified:

(1) No downhill moves during the (typically) four last temperature stages

(2) TEMP < TSTOP

```
(3) $\exists\ k \in [1, n]$: STPMST($k$) < EPSREL * STPINI($k$) + EPSABS
(4) NFOBJ $\geq$ NFMAX * $n$

        then print results: best point: **XOPT**, ENOPT, NFOBJ
        stop
    Else
        begin a new temperature stage
    Endif
```

Reasonable values of EPSREL, EPSABS, and NFMAX:

EPSREL = 10^{-6}
EPSABS = 10^{-8}
NFMAX = 5000

The tests (2) and (3) are discussed at the end of this section.

Step 9: Initialization of a New Temperature Stage

Set:

```
    MVOKST = NMVUST = NMVST = 0
    **MOKST** = **MSOTST** = **0**
    ELOWST = OLDRGY (FOBJ value of the last accepted point)
    AVGYST = 0
    SDGYUP = 0
Go to Step 2
```

3.1 Discussion of Temperature and Step Vector Adjustment Rules

3.1.1 *Temperature Adjustment.* In metallurgy, when annealing a piece of material, the temperature is lowered rather quickly at the beginning of the process, and then more slowly while approaching the transition between a disordered and an ordered state. Similarly, we have checked that one can save substantial CPU time by decreasing the temperature more quickly at the beginning of the SA process. More precisely, we use the ratio between minimal and average FOBJ values at the previous temperature stage to decide whether the temperature must be decreased at a fast or a slow rate. When the SA process starts, the FOBJ minimum value is frequently and substantially improved. So the above-mentioned ratio is small, and the temperature is lowered quickly. Moreover, the temperature adjustment rule is built such that the temperature reduction rate RFTMP is kept between predetermined upper and lower bounds.

3.1.2 *Step Vector Adjustment.* Several strategies have been proposed to adjust the step vector **STPMST** [Catthoor et al. 1988; Corana et al. 1987; Vanderbilt and Louie 1984]. They differ on step adjustment periodicity, which can be more or less tightly related to the temperature decrease rate. Several updating algorithms have been reported, too. To observe and control the optimization efficiency during the SA process, various criteria have been used—in particular, the average amplitude of previously accepted moves and the acceptance rate of previously attempted moves. Our strategy, detailed at Step 7, has been chosen among many candidates on account of both its efficiency and its simplicity. Its parameters have been determined through extensive numerical tests. The step vector is updated jointly with the temperature, according to the acceptance rate of attempted moves at the previous temperature stage. All its components are updated together. This procedure keeps the acceptance rate for moves at a reasonable level during the whole SA process. We stress the fact that a good step vector adjustment scheme must not yield steps that are too large, which makes the SA process similar to an ordinary random search, with too many uncorrelated and unaccepted moves, whereas steps that are too small result in an incomplete exploration of the variation domain, with small and frequent Δf reductions.

3.2 Discussion of the Stopping Tests

3.2.1 *Test on the Temperature.* As indicated in Section 2, the initial temperature TMPINI is related to the FOBJ initial variation average DGYINI and to the uphill moves' initial acceptance rate PROBOK by the relation

$$\text{PROBOK} = \exp\left(-\text{DGYINI}/\text{TMPINI}\right),$$

equivalent to

$$\text{TMPINI} = -\text{DGYINI}/\text{Log}\left(\text{PROBOK}\right).$$

This explicit expression for TMPINI shows clearly how a large DGYINI value is related to a large TMPINI, i.e., how the FOBJ coarse-grain behavior is taken into account. The initial moves are performed with large spatial discretization steps and thus produce a large FOBJ initial variation average DGYINI.

As T is gradually lowered, we must estimate a low-enough temperature TSTOP at which it is reasonable to interrupt the annealing process. TSTOP must reflect the fine-grain local FOBJ behavior. The above considerations suggest a straightforward definition of TSTOP, closely related to a P_stop uphill moves' final acceptance rate and to a $\langle \Delta f \rangle$_stop FOBJ final variation average. At the end of the annealing process, the uphill moves' acceptance rate has been progressively reduced to a small value P_stop, which can be suitably expressed as a small fraction of PROBOK, directly related to the stopping temperature TSTOP << TMPINI and to the corresponding $\langle \Delta f \rangle$_stop << DGYINI.

Let us take two small numbers EPSREL and EPSABS, and let PROBOK, P_stop, DGYINI, $\langle \Delta f \rangle$_stop, and TSTOP be related by

P_stop = EPSREL * PROBOK + EPSABS
(typical values: PROBOK = 0.5, EPSREL = 10^{-6},
and EPSABS = 10^{-8},

$\langle \Delta f \rangle$_stop = EPSREL * DGYINI + EPSABS
P_stop = exp (– $\langle \Delta f \rangle$_stop/TSTOP),
thus, TSTOP = – $\langle \Delta f \rangle$_stop/Log (P_stop)
TSTOP = – (EPSREL * DGYINI
+ EPSABS/Log (EPSREL * PROBOK + EPSABS)

This last expression shows clearly the link between FOBJ finer-grain behavior and TSTOP. A low TSTOP means a small variation average $\langle \Delta f \rangle$_stop, corresponding to small FOBJ local variations, themselves related to small spatial moves. Those small FOBJ variations indicate that the current point might be placed in the neighborhood of a local or eventually of the global minimum.

3.2.2 *Test on the Spatial Discretization Steps.* As temperature T is gradually lowered, we are led to shrink progressively each component of the spatial step vector **STPMST**. We can therefore consider that we have reached another type of local stopping conditions, when at least one component of the step vector, say the kth, starting with the initial value STPINI(k), has been reduced to EPSREL * STPINI(k) + EPSABS. As above EPSREL is a relative factor (e.g., EPSREL = 10^{-6}) and EPSABS is a small number (e.g., EPSABS = 10^{-8}), which prevents EPSREL * STPINI(k) from decreasing to less than $\varepsilon_{\mathrm{MACHINE}}$, in case STPINI($k$) is itself originally small due to the problem at hand. Again it seems natural to consider that we have reached reasonable local stopping conditions when the fine-grain FOBJ local behavior is reflected by the reduced size of spatial moves, compared to larger initial steps STPINI(k), $k = 1,\ldots,n$.

4. TESTS AND RESULTS

4.1 Normalization of Variables

In complex circuit optimization problems that we have to solve, some variables—say, the X_{real} variables—take small or large absolute values within very large dynamic ranges (e.g., $X_{\mathrm{real\_min}} = 10^{-2}$, $X_{\mathrm{real\_max}} = 10^{-3}$ or $X_{\mathrm{real\_min}} = 10^{1}$, $X_{\mathrm{real\_max}} = 10^{14}$). The natural logarithmic variable change, defined by $X_{\mathrm{norm}} = \alpha * \mathrm{Log}\,(X_{\mathrm{real}} + X_0) + \beta$ (α,β = constants), gives a greatly reduced X_{norm} dynamic range and ensures a nearly uniform exploration of the new defined range, for instance, $X_{\mathrm{norm\_min}} = 1$, $X_{\mathrm{norm\_max}} = 10$. X_0 is chosen to ensure a nonnegative argument for the Log function, across the original X_{real} range. α and β are computed from two simple equations relating $X_{\mathrm{real\_min}}$, $X_{\mathrm{real\_max}}$, $X_{\mathrm{norm\_min}}$, and $X_{\mathrm{norm\_max}}$.

Other variables have a much smaller range (e.g., $X_{\mathrm{real\_min}} = 10$, $X_{\mathrm{real\_max}} = 100$ or $X_{\mathrm{real\_min}} = 10^{3}$, $X_{\mathrm{real\_max}} = 10^{4}$). A simple linear variable change $X_{\mathrm{norm}} = \alpha * X_{\mathrm{real}} + \beta$ will bring X_{norm} into, say, the range [–1, +1], which is the optimal numerical range for floating-point operations. We have noticed an improved convergence toward correct results when using normalized variables type SA, instead of unnormalized real variables range exploration with the same SA algorithm.

4.2 Experimental Procedure

To test the ESA algorithm for globally minimizing functions of many continuous variables, we used 14 classical multiminima test functions (with 2 to 100 variables), which are listed in the Appendix, all with known global minima. For each test function, we adopted the hyperrectangular search domain associated with its definition, as given in the usual literature [Dixon and Szegö 1978; Schittkowski and Hock 1981; 1987]. To avoid any hazardous interpretation of optimization results, related to the choice of particular starting points, we performed each test 100 times, starting from 20 different points randomly selected in the search domain. Indeed, since SA is partly a stochastic algorithm, a single run does not provide a statistically significant result. Therefore, for each starting point, the SA tests were systematically performed five times, with five different initial seeds for the pseudorandom numbers generator.

The tests were performed in two separate stages. In the first stage, we looked for optimal values of new SA parameters which we introduced in order to achieve the $\mathbb{R}^n$ space partitioning and to control the variables discretization. Second, we tested and compared our SA algorithm efficiency against other SA implementations and other methods. For functions of two to six variables, we compared our results with those previously published. We did actually accept those published results as valid ones, and we did not program ourselves the corresponding algorithms. For functions of more than six variables, there were no available results. Consequently our SA results were not compared to those produced by other algorithms.

4.3 SA Parameters Adjustment

This section reports results of three computational investigations. We first looked for an optimal ratio of the size p of successive variables subsets to the total number n of variables. Then we studied the SA results sensitivity to step adjustment parameters (RATMAX, RATMIN, EXTSTP, and SHRSTP) variations. Finally we evaluated the influence of initial steps sizes by varying **ROSTEP**.

For the parameter p, we could draw the following conclusions. Most analytical test functions, like many listed in the Appendix, display an obvious symmetry in the arrangement and contribution of each variable to the function value. In that case the optimal choice is simply $p = 1$, which gives the shortest possible selection and move CPU times. But we noticed a special behavior for two of the 14 functions listed in the Appendix: the 50- and 100-variables Zakharov functions, Z_{50} and Z_{100}, which exhibit a very clear dissymmetry between their variables. For instance, for Z_{50}, the optimal choice, $p = 10$, saves about 35% of the CPU time which is necessary with $p = 1$. Similarly, when SA is used, for instance, in the

Table I. Minimization of 14 Test Functions with ESA Algorithm

Test Function	Rate of Successful Minimizations	FOBJ Initial-Value Average	Average of FOBJ Gaps between Best Successful Points Found and the Known Global Optimum	Standard Deviation of Previous FOBJ Gaps	Average of X Distances between Best Successful Points Found and the Known Global Optimum	Average of Objective Function Evaluations Numbers	Average CPU Time in Standard Units of Time
G	100	$0.6\ 10^{+3}$	0.8610^{-2}	0.3110^{-2}	0.3410^{-2}	783	0.86
$H_{3,4}$	100	$-0.5\ 10^{-1}$	0.4910^{-3}	0.9510^{-3}	0.6710^{-2}	698	1.50
$S_{4,5}$	54	$-0.1\ 10^{-1}$	0.4210^{-2}	0.2310^{-4}	0.1210^{-2}	1487	2.70
$S_{4,7}$	54	$-0.3\ 10^{+0}$	0.8410^{-2}	0.4410^{-2}	0.3910^{-2}	1661	3.18
$S_{4,10}$	50	$-0.4\ 10^{+0}$	0.4310^{-1}	0.5410^{-2}	0.9010^{-2}	1363	4.45
$H_{6,4}$	100	$-0.2\ 10^{+0}$	0.5910^{-1}	0.3110^{-1}	0.1510^{+0}	1638	4.37
R_{10}	100	$0.1\ 10^{+6}$	0.4410^{-1}	0.3310^{-1}	0.4110^{-2}	12,403	17.02
Z_{10}	100	$0.6\ 10^{+8}$	0.1510^{-2}	0.7110^{-2}	0.2610^{-2}	15,820	13.55
R_{20}	100	$0.2\ 10^{+6}$	0.1010^{+0}	0.5210^{-2}	0.2010^{-2}	24,623	31.03
Z_{20}	100	$0.4\ 10^{+9}$	0.3410^{-1}	0.7510^{-2}	0.2510^{-1}	69,799	35.30
R_{50}	100	$0.5\ 10^{+6}$	0.8810^{+1}	0.1210^{+0}	0.1010^{+0}	78,224	144.29
Z_{50}	100	0.910^{+13}	0.1710^{-1}	0.7010^{-2}	0.1410^{-1}	195,726	207.48
R_{100}	100	$0.1\ 10^{+7}$	0.1710^{+2}	0.4610^{+1}	0.1210^{+0}	188,227	192.26
Z_{100}	100	0.210^{+16}	0.5710^{+0}	0.4110^{+0}	0.5810^{-1}	789,718	754.60

solution of complex circuit design problems, objective functions generally depend on all kinds of variables of very different magnitudes. In that case the maximal efficiency of the $\mathbb{R}^n$ space exploration within a succession of $\mathbb{R}^p$ spaces was empirically obtained [Barat et al.1993; Berthiau and Siarry 1993; Berthiau et al. 1994; Durbin et al. 1991] when p is about $n/3$. Other systematic experiments are still in progress in order to obtain, if possible, a rule for the p-optimal choice, adapted to FOBJ-specific characteristics.

Various initial step sizes and step adjustment parameters had little effect on the overall efficiency. Best values obtained are given in Section 3.

4.4 Results

The results of SA tests performed on 14 functions are reported in Table I. To evaluate the program efficiency, we retained the following criteria summarizing results from 100 minimizations for each function: rate of successful minimizations (yielding points very close to—in a sense defined further—the FOBJ global optimum), FOBJ initial-value average, average of FOBJ gaps between best successful points found and the known global optimum, standard deviation of these gaps, average of **X** distances between best points found (in case of success) and the known global optimum, average of objective function evaluation numbers, and average CPU time in standard units of time [Dixon and Szegö 1978] (1000 Shekel 4,5 evaluations at the point (4,4,4,4)).

Table II. Listing of Various Methods Used in the Comparison

Method	Reference
Enhanced Simulated Annealing (ESA)	This article
Monte Carlo Simulated Annealing	Vanderbilt and Louie [1984]
Multistart	Rinnooy Kan and Timmer [1984]
Multilevel Single Linkage	Rinnooy Kan and Timmer [1987]
Improved SA (narrow search domain)	Tsoi and Lim [1988]
Improved SA (large search domain)	Tsoi and Lim [1988]
Sniffer Global Optimization	Butler and Slaminka [1992]

When at least one of the four stopping tests is verified, ESA stops and provides the coordinates of a point in $\mathbb{R}^n$, along with the $\mathrm{FOBJ}_{\mathrm{ESA}}$ value at this point. We compared this result with the known analytical minimum $\mathrm{FOBJ}_{\mathrm{ANAL}}$: we considered this result $[\mathrm{FOBJ}_{\mathrm{ESA}}, (x_1, x_2, \ldots, x_n)_{\mathrm{ESA}}{}^{\mathrm{T}}]$ to be successful if the following inequality held:

$$|\mathrm{FOBJ}_{\mathrm{ESA}} - \mathrm{FOBJ}_{\mathrm{ANAL}}| < \mathrm{EPSREL}^*\mathrm{FOBJ}_{\mathrm{ANAL}} + \mathrm{EPSABS}$$

Our SA results were then compared to those produced by six other global optimization algorithms, for the first six functions of 2 to 6 variables listed in the Appendix. The various methods—three other SA implementations and three different methods—used in the comparison, listed in Table II, were selected according to previously published computational results, for the same set of six test functions proposed by Dixon and Szegö [1978].

We retained the following three comparison criteria: the successful executions rate (Table III), the number of objective function evaluations (Table IV), and the running time in standard units of time (Table V).

Note that some results are not available for some methods. In particular, the rate of successful executions is not mentioned when the number of executions per test function is too low or is not specified by the authors. Our results compare favorably with most methods, except with methods "Multilevel Single Linkage" [Rinnooy Kan and Timmer 1987] and "Sniffer Global Optimization" [Butler and Slaminka 1992]. But results from the method "Multilevel Single Linkage" are published with only four executions per test function, and no information is given about the distance of the starting points to the global minimum. Our results are rather bad for the three Shekel functions. But for some authors, the search domain is smaller or is not specified. Besides, methods "Monte Carlo Simulated Annealing" [Vanderbilt and Louie 1984] and "Sniffer Global Optimization" use the first and eventually the second derivatives of the objective function to perform each move. It should be recalled here that for high-dimensionality problems, such as complex analog circuit optimization, derivatives of objective functions are normally unavailable. Our ESA method is specially suited to this kind of problems. A comparison of our results with those produced by an improved version of the Nelder and Mead Simplex algorithm, for functions of more than 10 variables, is in progress and will be published shortly.

Table III. Rate of Successful Minimizations

Method	G	$H_{3,4}$	$S_{4,5}$	$S_{4,7}$	$S_{4,10}$	$H_{6,4}$
Enhanced Simulated Annealing (ESA)	100	100	54	54	50	100
Monte Carlo Simulated Annealing	99	100	54	64	81	62
Improved SA (narrow search domain)	—	99	7	1	3	97
Improved SA (large search domain)	—	100	19	28	18	0
Sniffer Global Optimization	100	99	90	96	95	99

Table IV. Total Number of Objective Function Evaluations

Method	G	$H_{3,4}$	$S_{4,5}$	$S_{4,7}$	$S_{4,10}$	$H_{6,4}$
Enhanced Simulated Annealing (ESA)	783	698	1487	1661	1363	1638
Monte Carlo Simulated Annealing	1186	1224	3910	3421	3078	1914
Multistart	4400	2500	6500	9300	11,000	6000
Multilevel Single Linkage	148	197	404	432	564	487
Improved SA (narrow search domain)	—	6965	6030	2936	3562	21,802
Improved SA (large search domain)	—	13,758	8568	8631	7824	1116
Sniffer Global Optimization	664	534	3695	2655	3070	1760

Table V. Running Time in Standard Units of Time

Method	G	$H_{3,4}$	$S_{4,5}$	$S_{4,7}$	$S_{4,10}$	$H_{6,4}$
Enhanced Simulated Annealing (ESA)	0.86	1.50	2.70	3.18	4.45	4.37
Monte Carlo Simulated Annealing	2.0	4.0	16.0	15.0	15.0	12.0
Multistart	4.5	7.0	13.0	21.0	32.0	22.0
Multilevel Single Linkage	0.15	0.5	1.0	1.0	2.0	2.0
Improved SA (narrow search domain)	—	0.14	10.30	0.58	7.64	20.98
Improved SA (large search domain)	—	0.28	7.34	1.88	14.30	10.88

Table VI. Values of the Parameters of the HARTMANN 3,4 Function

i	a_{ij}			c_i	p_{ij}		
1	3.0	10.0	30.0	1.0	0.3689	0.1170	0.2673
2	0.1	10.0	35.0	1.2	0.4699	0.4387	0.7470
3	3.0	10.0	30.0	3.0	0.1091	0.8732	0.5547
4	0.1	10.0	35.0	3.2	0.03815	0.5743	0.8828

APPENDIX

LIST OF TEST FUNCTIONS

(1) GOLDSTEIN-PRICE—2 variables:

$$G(x) = [1 + (x_1 + x_2 + 1)^2 * (19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]*$$

$$[30 + (2x_1 - 3x_2)^2 * (18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$$

—search domain: $-2 \leq x_j \leq 2,\ j = 1, 2;$

Table VII. Values of the Parameters of the SHEKEL 4,n Functions

i	a^T_i				c_i
1	4.0	4.0	4.0	4.0	0.1
2	1.0	1.0	1.0	1.0	0.2
3	8.0	8.0	8.0	8.0	0.2
4	6.0	6.0	6.0	6.0	0.4
5	3.0	7.0	3.0	7.0	0.4
6	2.0	9.0	2.0	9.0	0.6
7	5.0	5.0	3.0	3.0	0.3
8	8.0	1.0	8.0	1.0	0.7
9	6.0	2.0	6.0	2.0	0.5
10	7.0	3.6	7.0	3.6	0.5

Table VIII. Values of the Parameters of the HARTMANN 6,4 Function

i	a_{ij}						c_i
1	10.0	3.00	17.0	3.50	1.70	8.00	1.0
2	0.05	10.0	17.0	0.10	8.00	14.0	1.2
3	3.00	3.50	1.70	10.0	17.0	8.00	3.0
4	17.0	8.00	0.05	10.0	0.10	14.0	3.2

i	p_{ij}					
1	0.1312	0.1696	0.5569	0.0124	0.8283	0.5886
2	0.2329	0.4135	0.8307	0.3736	0.1004	0.9991
3	0.2348	0.1451	0.3522	0.2883	0.3047	0.6650
4	0.4047	0.8828	0.8732	0.5743	0.1091	0.0381

—4 local minima; lowest minimum: $x^* = (0, -1)$; $f(x^*) = 3$.

(2) HARTMANN 3,4—3 variables (see Table VI):

$$H_{3,4}(x) = -\sum_{i=1}^{4} c_i \, exp \, [-\sum_{j=1}^{3} a_{ij} (x_j - p_{ij})^2]$$

—search domain: $0 \leq x_j \leq 1, j = 1, 3$;

—4 local minima: $\mathbf{p_i} = (p_{i1}, p_{i2}, p_{i3})$ = ith local minimum approximation: $f(\mathbf{p_i}) \cong -c_i$.

(3) SHEKEL 4,5, SHEKEL 4,7, SHEKEL 4,10—4 variables (see Table VII):

$$S_{4,n}(x) = -\sum_{i=1}^{n} [(x - \mathbf{a_i})^T (x - \mathbf{a_i}) + c_i]^{-1}$$

—3 functions $S_{4,n}$ were considered: $S_{4,5}$, $S_{4,7}$, and $S_{4,10}$;

—search domain: $0 \leq x_j \leq 10$, $j = 1, 4$;

—10 local minima; $\mathbf{a_i}^T$ = ith local minimum approximation: $f(\mathbf{a_i}^T) \cong -1/c_i$.

(4) HARTMANN 6,4—6 variables (see Table VIII):

$$H_{6,4}(x) = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right]$$

—search domain: $0 \leq x_j \leq 1, j = 1, 6$;

—4 local minima; $\mathbf{p_i} = (p_{i1}, \ldots, p_{i6})$ = ith local minimum approximation: $f(\mathbf{p_i}) \cong -c_i$.

(5) ROSENBROCK 10, 20, 50, and 100 variables:

$$R_n(x) = \sum_{j=1}^{n-1} [100\,(x_j^2 - x_{j+1})\,2 + (x_j - 1)^2]$$

—4 functions were considered: R_{10}, R_{20}, R_{50}, and R_{100}, for n = 10, 20, 50, and 100;

—search domain: $-5 \leq x_j \leq 10$, $j = 1, n$;

—lowest minimum: $x^* = (1, \ldots, 1)$; $f(x^*) = 0$.

(6) ZAKHAROV 10, 20, 50, and 100 variable

$$Z_n(x) = \left(\sum_{j=1}^{n} x_j^2\right) + \left(\sum_{j=1}^{n} 0.5jx_j\right)^2 + \left(\sum_{j=1}^{n} 0.5jx_j\right)^4$$

—4 functions were considered: Z_{10}, Z_{20}, Z_{50}, and Z_{100}, for n = 10, 20, 50, and 100;

—search domain: $-5 \leq x_j \leq 10$, $j = 1, n$;

—lowest minimum: $x^* = (0, \ldots, 0)$; $f(x^*) = 0$.

REFERENCES

AARTS, E. H. L. AND VAN LAARHOVEN, P. J. M. 1987. *Simulated Annealing: Theory and Applications*. Kluwer Academic Publishers, Hingham, Mass.

BARABINO, G. P., BARABINO, G. S., BIANCO, B., AND MARCHESI, M. 1980. A study on the performance of simplex methods for function minimization. In *Proceedings of the IEEE International Conference on Circuits and Computers*. IEEE Press, Piscataway, N.J., 1150–1153.

BARAT, E., OUSI BENOMAR, K., AND BERTHIAU, G. 1993. Separation of radiations sources by means of statistical analysis of the observed signal. In *Proceedings of the International Atomic Energy Agency Specialist Meeting on Improvements in Nuclear and Radiation*

Instrumentation for NPPs: Impact of Experience and New Technologies (Saclay, France, Oct.).

BERTHIAU, G. AND SIARRY, P. 1993. An improved characterization of the Materka model through simulated annealing. In *Proceedings of the 11th European Conference on Circuit Theory and Design. Part I (ECCD '93)* (Davos, Switzerland, Aug.-Sept.). Elsevier Science Publishers B. V., Amsterdam, The Netherlands, 217–222.

BERTHIAU, G., DURBIN, F., HAUSSY, J., AND SIARRY, P. 1994. Learning of neural networks approximating continuous functions through circuit simulator SPICE-PAC driven by simulated annealing. *Int. J. Electron. 76*, 437–441.

BUTLER, R. A. R. AND SLAMINKA, E. E. 1992. An evaluation of the Sniffer Global Optimization algorithm using standard test functions. *J. Comput. Phys. 99*, *1* (Mar.), 28–32.

CATTHOOR, F., DE MAN, H., AND VANDEWALLE, J. 1988. SAMURAI: A general and efficient simulated-annealing schedule with fully adaptive annealing parameters. *Integr. VLSI J. 6*, *2* (July), 147–178.

CERNY, V. 1985. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *J. Optim. Theory Appl. 45*, *1*, 41–51.

CORANA, A., MARCHESI, M., MARTINI, C., AND RIDELLA, S. 1987. Minimizing multimodal functions of continuous variables with the "simulated annealing" algorithm. *ACM Trans. Math. Softw. 13*, *3* (Sept.), 262–280.

DIXON, L. C. W. AND SZEGÖ, G. P., Eds. 1978. *Towards Global Optimization 2*. North-Holland, Amsterdam, The Netherlands.

DURBIN, F., HAUSSY, J., BERTHIAU, G., SIARRY, P., AND ZUBEREK, W. M. 1990. Integrated circuit performance optimization with simulated annealing algorithm and SPICE-PAC circuit simulator. In *Proceedings of the EURO ASIC '90 Conference* (Paris, France). IEEE Press, Piscataway, N.J., 407–412.

DURBIN, F., HAUSSY, J., BERTHIAU, G., AND SIARRY, P. 1991. Integrated circuit performance optimization and transistor model fitting to experimental data with simulated annealing algorithm. In *Proceedings of the ECCTD '91 Conference* (Copenhagen, Denmark), 833–842.

FLETCHER, R. AND POWELL, M. J. D. 1963. A rapidly convergent descent method for minimization. *Comput. J. 6*, 163–168.

FLETCHER, R. AND REEVES, C. M. 1964. Function minimization by conjugate gradients. *Comput. J. 7*, 149–154.

GLOVER, F. 1977. Heuristics for integer programming using surrogate constraints. *Dec. Sci. 8*, 156–166.

GOLDBERG, D. E. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Publishing Co., Inc., Redwood City, Calif.

HOOKE, R. AND JEEVES, T. A. 1961. "Direct search" solution of numerical and statistical problems. *J. ACM 8*, 212–229.

HOPFIELD, J. J. AND TANK, D. W. 1985. "Neural" computation of decisions in optimization problems. *Biol. Cybern. 52*, 141–142.

KIRKPATRICK, S., GELATT, C. D., AND VECCHI, M. P. 1983. Optimization by simulated annealing. *Science 220*, *4598* (May), 671–680.

METROPOLIS, N., ROSENBLUTH, A., ROSENBLUTH, M., TELLER, A., AND TELLER, E. 1953. Equation of state calculations by fast computing machines. *J. Chem. Phys. 21*, 1087–1092.

NELDER, J. A. AND MEAD, R. 1964. A simplex method for function minimization. *Comput. J. 7*, 308–313.

RINNOOY KAN, A. H. G. AND TIMMER, G. T. 1984. Stochastic methods for global optimization. *Am. J. Math. Manage. Sci. 4*, 7–40.

RINNOOY KAN, A. H. G AND TIMMER, G. T. 1987. Stochastic global optimization methods: Part II: Multi level methods. *Math. Program. 39*, *1* (Sept. 1), 57–78.

SCHITTKOWSKI, K. AND HOCK, W. 1981. *Test Examples for Nonlinear Programming Codes*. Lecture Notes in Economics and Mathematical Systems, vol. 187. Springer-Verlag New York, Inc., New York, N.Y.

SCHITTKOWSKI, K. AND HOCK, W. 1987. *More Test Examples for Nonlinear Programming Codes*. Lecture Notes in Economics and Mathematical Systems, vol. 282. Springer-Verlag New York, Inc., New York, N.Y.

Siarry, P., Bergonzi, L., and Dreyfus, G. 1987. Thermodynamic optimization of block placement. *IEEE Trans. Comput.-Aided Des. 6*, 211–221.

Tsoi, A. C. and Lim, M. 1988. Improved simulated annealing technique. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics* (Beijing-Shenyang, China). IEEE Press, Piscataway, N.J., 594–597.

Vanderbilt, D. and Louie, S. G. 1984. A Monte Carlo simulated annealing approach to optimization over continuous variables. *J. Comput. Phys. 56*, 259–271.

Received November 1995; revised January 1996 and November 1996; accepted January 1997