JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 77, No. 1, APRIL 1993

Terminal Repeller Unconstrained Subenergy Tunneling (TRUST) for Fast Global Optimization[1]

B. C. CETIN,[2] J. BARHEN,[3] AND J. W. BURDICK[4]

Communicated by G. Di Pillo

Abstract. A new method for unconstrained global function optimization, acronymed TRUST, is introduced. This method formulates optimization as the solution of a deterministic dynamical system incorporating terminal repellers and a novel subenergy tunneling function. Benchmark tests comparing this method to other global optimization procedures are presented, and the TRUST algorithm is shown to be substantially faster. The TRUST formulation leads to a simple stopping criterion. In addition, the structure of the equations enables an implementation of the algorithm in analog VLSI hardware, in the vein of artificial neural networks, for further substantial speed enhancement.

Key Words. Global optimization, dynamical systems, terminal repellers, subenergy tunneling function, artificial neural networks.

1. Introduction

Many engineering applications can be formulated as nonlinear function optimization problems in which the function to be optimized possesses many local minima in the parameter region of interest. In most cases, it is desired to find the local minimum at which the function takes its lowest value, i.e., the global minimum. The problem of designing algorithms that can distinguish between the global minimum and the numerous local minima is known as the global optimization problem. This paper presents a new

[1]This work was supported by the Department of Energy, Office of Basic Energy Sciences, Grant No. DE-A105-89-ER14086.

[2]Graduate Research Assistant, Department of Electrical Engineering, California Institute of Technology, Pasadena, California.

[3]Head, Nonlinear Science and Information Processing Group, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, California.

[4]Assistant Professor, Department of Mechanical Engineering, California Institute of Technology, Pasadena, California.

0022-3239/93/0400-0097$07.00/0

global optimization scheme whose acronym is TRUST (terminal repeller unconstrained subenergy tunneling). In this approach, we formulate optimization as the solution to a deterministic dynamical system which incorporates a novel subenergy tunneling functional and terminal repellers. In addition, the TRUST formulation leads to a well-defined stopping criterion.

In standard benchmark tests, TRUST has proven to be significantly faster than previously published techniques. More importantly, this algorithm has been especially designed for implementation in parallel analog VLSI circuitry (i.e., artificial neural network architectures) for substantial speed enhancements. In related work (Ref. 1), the authors have successfully designed, fabricated, and tested analog VLSI circuits which implement most of the basic components of this algorithm. We hope to report in a future article a complete hardware implementation.

The TRUST computational scheme can be guaranteed to find the global minimum for functions of one variable. The method is currently not guaranteed to find the global minima in multiple dimensions. However, in the multidimensional case, the method will always escape from one local minimum to another with a lower functional value. In practice, the global minimum was found in all benchmark simulations, including 10-dimensional test functions. Furthermore, the structure of the optimizing dynamical system is highly parallel, allowing implementation in a form whose computational complexity is only weakly dependent on problem dimensionality.

The global optimization problem to be considered in this paper can be stated as follows. Let $f(\bar{x}):\mathbb{R}^n \to \mathbb{R}$ be a twice continuously differentiable function, where $\bar{x}$ is a vector of n state variables or parameters. Hereafter, $f(\bar{x})$ will be referred to as the objective function. The goal is to find the value $\bar{x}_{GM}$ of the state variables which minimizes $f(\bar{x})$,

$$f^* = f(\bar{x}_{GM}) = \min\{f(\bar{x}) \mid \bar{x} \in \mathscr{D}\}, \tag{1}$$

where $\mathscr{D}$ is the domain of interest over which one seeks the global minimum; $\mathscr{D}$ is assumed to be compact and connected. In the sequel, and without loss of generality, we assume $\mathscr{D}$ to be the hyperparallelepiped

$$\mathscr{D} = \{x_j \mid x_{jL} \leq x_j \leq x_{jU}\,; j = 1, 2, \ldots, n\}, \tag{2}$$

where x_{jL} and x_{jU} are respectively the lower and upper bounds on the jth state variable. The compactness of $\mathscr{D}$ and continuity of $f(\bar{x})$ ensure that $f(\bar{x})$ is bounded away from infinite magnitude in the domain of interest. Further, we assume that every local minimum $\bar{x}_{LM}$ of $f(\bar{x})$ in $\mathscr{D}$ satisfies the conditions

$$\partial f(\bar{x}_{LM})/\partial \bar{x} = 0, \tag{3}$$

$$\bar{y}^T(\partial^2 f(\bar{x}_{LM})/\partial \bar{x}^2)\bar{y} \geq 0, \qquad \forall \bar{y} \in \mathbb{R}^n. \tag{4}$$

We further assume that the global minimum satisfies these local minimum criteria and that the global minimum does not occur on the boundary of $\mathscr{D}$.

Section 2 reviews previous global optimization approaches which are relevant to this work. This review focuses on tunneling methods, since the TRUST algorithm introduces a novel approach to tunneling. Section 3 presents the one-dimensional TRUST optimization algorithm. Section 4 discusses the convergence properties of the one-dimensional algorithm, while Section 5 considers the multi-dimensional TRUST scheme. Section 6 presents the results of benchmark simulations and compares the TRUST performance to other global optimization methods. Section 7 summarizes our conclusions.

2. Methodologies for Global Optimization: Background

Previously developed global optimization algorithms can be roughly categorized into two classes: probabilistic and deterministic. An extensive review of probabilistic computational schemes can be found in Ref. 2. Here, we focus on deterministic tunneling methods, as these are most closely related to the concept presented in this paper.

Tunneling for global optimization was introduced by Levy and Montalvo (Ref. 3). Their tunneling method is composed of a sequence of cycles, where each cycle has two phases: a local minimization phase and a tunneling phase. In the first phase, minimization algorithms such as gradient descent or Newton's method are employed to minimize $f(\bar{x})$. We assume that, starting from an initial point $\bar{x}^{0(0)}$, the minimization converges to the first local minimum $\bar{x}^{1(*)}$, which satisfies conditions (3) and (4).

In the second phase, a tunneling function is defined,

$$T(\bar{x}, \bar{x}^{1(*)}) = \hat{f}(\bar{x})/[(\bar{x}-\bar{x}^{1(*)})^T(\bar{x}-\bar{x}^{1(*)})]^{\alpha}, \tag{5}$$

where

$$\hat{f}(\bar{x}) = f(\bar{x}) - f(\bar{x}^{1(*)}). \tag{6}$$

The tunneling phase searches for the zeros of $T(\bar{x}, \bar{x}^{1(*)})$; that is, $T(\bar{x}, \bar{x}^{1(*)}) = 0$ is solved for any $\bar{x}^{1(0)}$ such that $\bar{x}^{1(0)} \neq \bar{x}^{1(*)}$, but $f(\bar{x}^{1(0)}) = f(\bar{x}^{1(*)})$. The denominator of (5) is a pole of strength α, located at the previously determined local minimum $\bar{x}^{1(*)}$, thus preventing the zero-finding algorithm from rediscovering $\bar{x}^{1(*)}$ as a zero of the tunneling function. The zero $\bar{x}^{1(0)}$ of (5) is used as the starting point of the next cycle, and the process is repeated sequentially, as shown in Fig. 1, until a stopping criterion, such as the failure to find a zero within a prescribed CPU time, is met. The last local minimum to be found is assumed to be the global minimum.

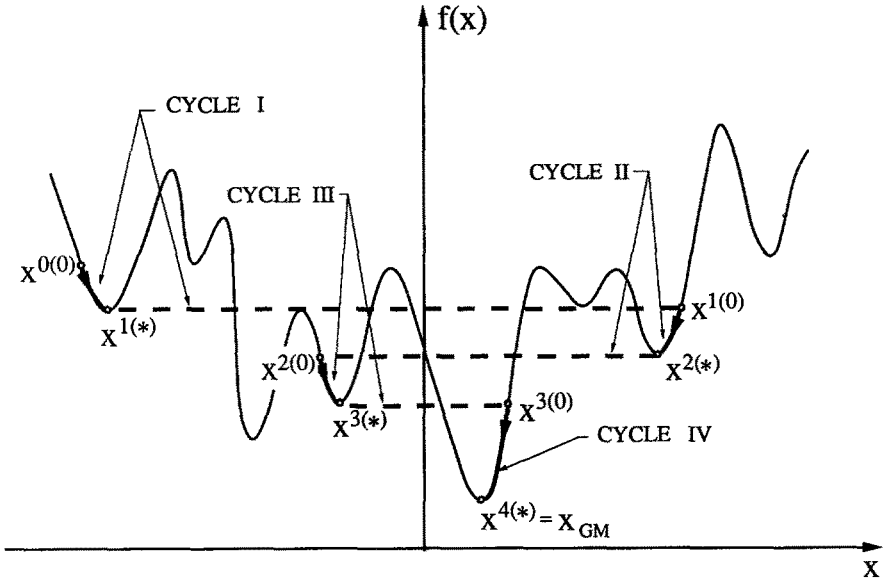

Fig. 1. Schematic diagram of tunneling operation.

If we denote $\bar{x}^{i(*)}$ as the minimum reached during the ith minimization phase, the tunneling algorithm implements a global descent property,

$$f(\bar{x}^{i+1(*)}) \leq f(\bar{x}^{i(*)}).$$

However, this method has a number of disadvantages:

(i) The pole strength α is problem dependent. While searching for a zero, α should be incrementally increased until the pole in the denominator of (5) becomes strong enough to eliminate the last local minimum of higher order. Every increase in α requires the algorithm to be restarted, leading to increased computational effort.

(ii) The tunneling algorithm may find another local minimum $\bar{x}^{2(*)}$, such that $f(\bar{x}^{1(*)}) = f(\bar{x}^{2(*)})$. In this case, an additional pole must be placed at the second local minimum, and the tunneling process must be restarted.

(iii) Division by a pole causes smoothing of $f(\bar{x})$ as $\bar{x} \to \infty$; that is, $f(\bar{x}) \to 0$ as $\bar{x} \to \infty$. This smoothing increases with α, yielding a tunneling function that becomes very flat. In this case, zeros can be difficult to detect correctly.

(iv) The zero-finding algorithm in Ref. 3 is based on a modified Newton iteration which requires finding the roots of a scalar function with multiple variables. This can be a computationally expensive procedure, and

as yet there are no globally convergent zero-finding algorithms. Thus, stopping critieria cannot easily be defined.

The difficulties associated with finding the zeros of (5) have been partly overcome by the dynamical tunneling algorithm of Yao (Ref. 4). His dynamical tunneling procedure has two phases: dynamic optimization and dynamic tunneling. The dynamic optimization phase implements minimization via gradient descent,

$$\dot{\bar{x}} = -\partial f(\bar{x})/\partial \bar{x}. \tag{7}$$

Starting from an initial point $\bar{x}^{0(0)}$, the system (7) reaches its first equilibrium at a local minimum $x^{1(*)}$. However, in the second phase, instead of finding the zeros of the tunneling function (5), Yao defines an energy function,

$$E(\bar{x}, \bar{x}^{1(*)}) = T(\bar{x}, \bar{x}^{1(*)}) + k \int_0^{\hat{f}(\bar{x})} zu(z)\, dz, \tag{8}$$

where $u(z)$ is the Heaviside step function,

$$u(z) = \begin{cases} 1, & z \geq 0, \\ 0, & z < 0. \end{cases} \tag{9}$$

The energy function in (8) is minimized in Yao's tunneling phase, instead of finding the zeros of $T(\bar{x}, \bar{x}^{1(*)})$. The derivative of (8) with respect to its state vector $\bar{x}$ is

$$\begin{aligned} &\partial E(\bar{x}, \bar{x}^{1(*)})/\partial \bar{x} \\ &= \frac{(\partial f/\partial \bar{x})\|\bar{x} - \bar{x}^{1(*)}\|^{2\alpha} - 2\alpha(\bar{x} - \bar{x}^{1(*)})\|(\bar{x} - \bar{x}^{1(*)})\|^{2(\alpha-1)}\hat{f}(\bar{x})}{\|(\bar{x} - \bar{x}^{1(*)})\|^{4\alpha}} \\ &+ k(\partial f(\bar{x})/\partial \bar{x})\hat{f}(\bar{x})u(\hat{f}(\bar{x})). \end{aligned} \tag{10}$$

From (10), it is clear that the second term in (8) enforces the constraint $\hat{f}(\bar{x}) \leq 0$ [i.e., $f(\bar{x}) \leq f(\bar{x}^{1(*)})$] if the magnitude of k is chosen large enough. When gradient descent is applied to $E(\bar{x}, \bar{x}^{1(*)})$ in (8), we obtain the dynamical system

$$\dot{\bar{x}} = -\partial E(\bar{x}, \bar{x}^{1(*)})/\partial \bar{x}. \tag{11}$$

The initial conditions for this system are $\bar{x}^{1(*)} + \bar{\epsilon}$, where $\bar{\epsilon}$ is a small perturbation which displaces the system from the tunneling function pole located at $\bar{x}^{1(*)}$. When (11) converges to its final equilibrium state, it minimizes the tunneling function with respect to the constraint $\hat{f}(\bar{x}) \leq 0$. Thus, the system in (11) will reach an equilibrium point $\bar{x}^{1(0)}$ that lies in another basin of attraction, with functional values lower than $f(\bar{x}^{1(*)})$, if one exists. This new equilibrium point will be the starting point for the

dynamic optimization phase of the next cycle. The procedure is repeated until a new equilibrium in a lower valley cannot be found in a prescribed amount of time. It is then assumed that the last minimum is the global minimum.

This approach also has a number of deficiencies:

(i) The pole strength α must be chosen sufficiently high to enable the pole in the denominator of (5) to cancel the last local minimum of higher order, and thereby prevent restarting of the tunneling phase, as this necessitates backtracking of (11).

(ii) The penalty constant k is problem dependent, and a global minimum cannot be guaranteed for a prescribed k.

(iii) An implementation of global optimization in terms of the solution of two different dynamical systems in two different phases makes the algorithm impractical for implementation in analog VLSI hardware of the neural network type. A method based on a single differentiable equation would be preferable.

In this article, we introduce a deterministic global optimization methodology which is also based upon the concept of dynamic tunneling. However, in contrast to these previous approaches, tunneling is implemented here in a substantially different manner, by employing so-called terminal repellers and a novel subenergy tunneling function. The next section introduces these concepts and assembles them into an optimization algorithm which is the solution of a single vector differential equation. This characteristic simplifies the hardware implementation of our algorithm.

3. Terminal Repeller Unconstrained Subenergy Tunneling Algorithm

3.1. Subenergy Tunneling Function. We define a subenergy tunneling function, or subenergy function for short, as follows:

$$E_{\text{sub}}(\bar{x}, \bar{x}^*) = \log(1/[1+\exp(-(\hat{f}(\bar{x})+a))]), \tag{12}$$

where

$$\hat{f}(\bar{x}) = f(\bar{x}) - f(\bar{x}^*) \tag{13}$$

and a is a constant whose value will be considered below. In the above expression, $\bar{x}^*$ is a fixed value of $\bar{x}$, whose selection will also be discussed in the sequel.

Equation (12) is a nonlinear but monotonic transformation of $f(\bar{x})$ which has several useful properties. First, the derivative of $E_{\text{sub}}(\bar{x}, \bar{x}^*)$ with respect to $\bar{x}$ is

$$\partial E_{\text{sub}}(\bar{x}, \bar{x}^*)/\partial \bar{x} = (\partial f(\bar{x})/\partial \bar{x})(1/[1+\exp(\hat{f}(\bar{x})+a)]). \tag{14}$$

Since

$$1/[1+\exp(\hat{f}(\bar{x})+a)]>0, \qquad \bar{x}\in\mathscr{D},$$

we conclude that

$$\partial E_{\text{sub}}(\bar{x},\bar{x}^*)/\partial\bar{x}=0 \quad\Leftrightarrow\quad \partial f(\bar{x})/\partial\bar{x}=0. \tag{15}$$

From (15), it is clear that $E_{\text{sub}}(\bar{x},\bar{x}^*)$ has the same critical points as $f(\bar{x})$ and the same relative ordering of the local and global minima. In other words, $E_{\text{sub}}(\bar{x},\bar{x}^*)$ is a transformation of $f(\bar{x})$ which preserves all properties relevant for optimization. In addition, this transformation is intended to have the following effect. We wish $E_{\text{sub}}(\bar{x},\bar{x}^*)$ to asymptotically but quickly approach zero for $\hat{f}(\bar{x})\geq 0$. Second, we would like to leave $\hat{f}(\bar{x})$ nearly unmodified for $\hat{f}(\bar{x})<0$. Hereafter, $f(\bar{x}^*)$ will be referred to as the zero subenergy limit, since

$$E_{\text{sub}}(\bar{x},\bar{x}^*)\simeq 0, \qquad \text{for } f(\bar{x})\geq f(\bar{x}^*).$$

The monotonicity of the transformation is not affected by the particular value of the constant a, though the asymptotic properties are affected by its value. Figure 2 plots $E_{\text{sub}}(\bar{x},\bar{x}^*)$ vs $\hat{f}(\bar{x})$ for various values of a. The algorithm can be formulated to work for nearly any reasonable value of this parameter. In subsequent analyses, the necessary and sufficient values of other TRUST algorithm parameters are derived in terms of a. However, for practical applications, a value $a=2$ is chosen, as it leads to the most desirable asymptotic behavior of the subenergy tunneling transformation.

Figure 3 shows an example of a one-dimensional function,

$$f(x)=[\sin(2x)-x-1]^2,$$

to which the transformation in (12) has been applied for the case

$$x^*=-6.80678, \qquad a=2.$$

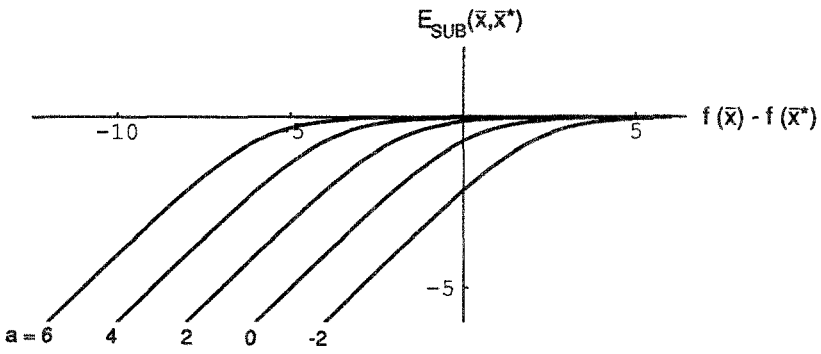

Fig. 2. Behavior of $E_{\text{sub}}(\bar{x},\bar{x}^*)$ vs $\hat{f}(\bar{x})$ for various values of a.

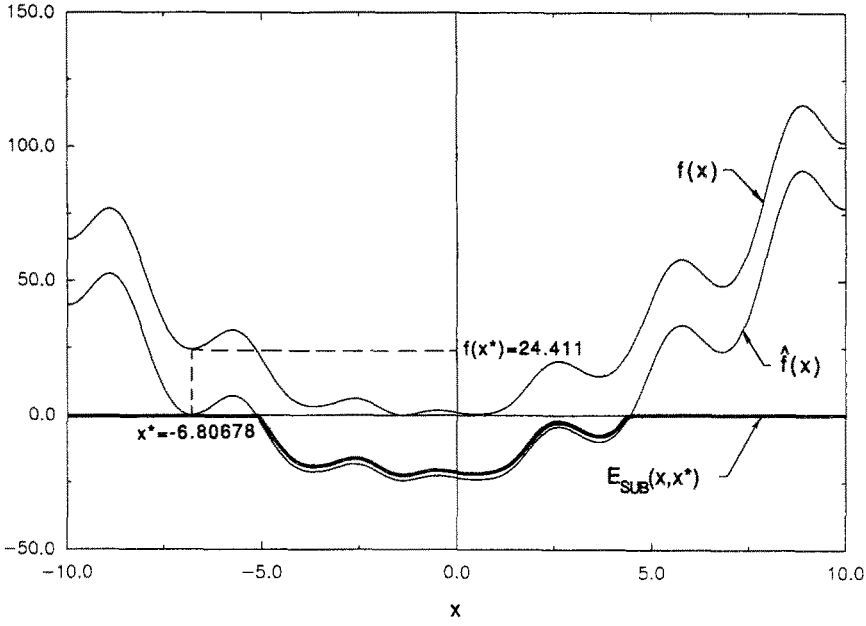

Fig. 3. Example of one-dimensional subenergy tunneling transformation.

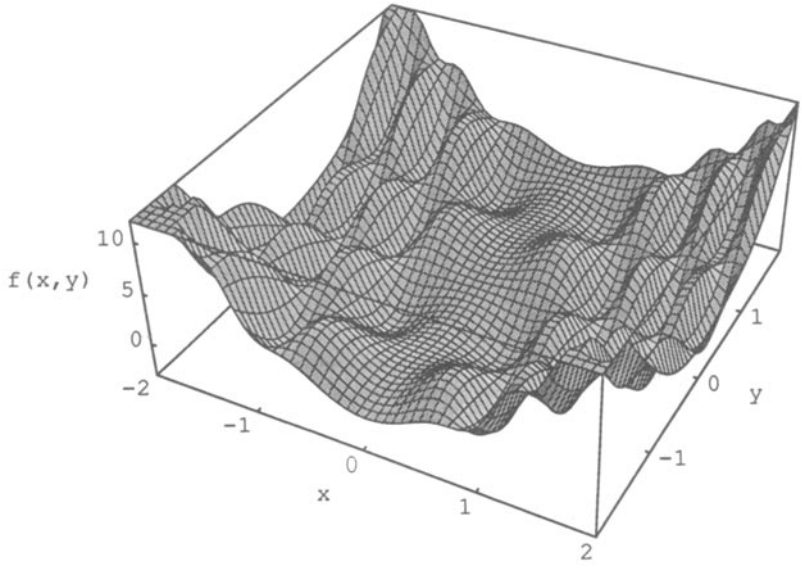

Fig. 4A. Example of a two-dimensional function.

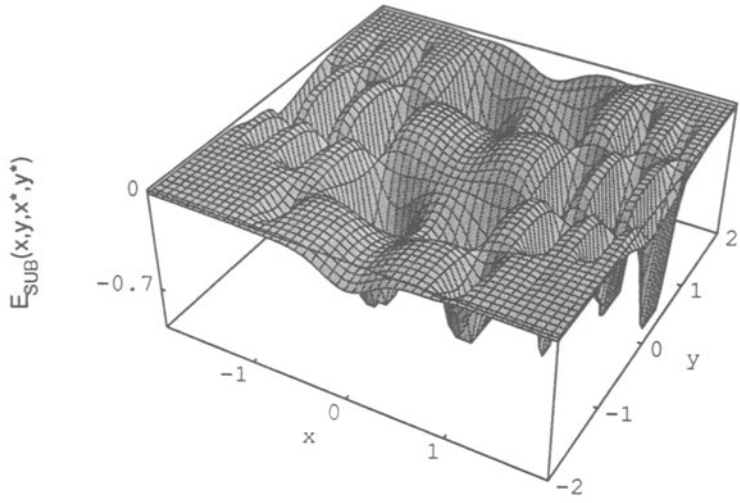

Fig. 4B. Subenergy tunneling transformation applied to the example of Fig. 4A.

Figure 4 shows an example of the transformation applied to the two-dimensional function:

$$f(x, y) = (x-0.1)^2(y-0.2)^2+3\sin(0.2+1.5\pi x^2)\sin(0.3+\pi y)$$

for the case

$$(x^*, y^*) = (0, -3/2), \qquad a = 2.$$

As can be observed, the subenergy function has the following approximate behavior, which is key to this optimization algorithm:

$$E_{\text{sub}}(\bar{x}, \bar{x}^*) \simeq \begin{cases} 0, & \hat{f}(\bar{x}) \geq 0, \text{ i.e., } f(\bar{x}) \geq f(\bar{x}^*), \\ \hat{f}(\bar{x}), & \hat{f}(\bar{x}) < 0, \text{ i.e., } f(\bar{x}) < f(\bar{x}^*), \end{cases} \tag{16}$$

$$\partial E_{\text{sub}}(\bar{x}, \bar{x}^*)/\partial \bar{x} \simeq \begin{cases} 0, & f(\bar{x}) \geq f(\bar{x}^*), \\ \partial f(\bar{x})/\partial \bar{x}, & f(\bar{x}) < f(\bar{x}^*). \end{cases} \tag{17}$$

Next we summarize and review the properties of terminal repellers.

3.2. Terminal Repellers. An equilibrium point $\bar{x}_{\text{eq}}$ of the dynamical system

$$\dot{\bar{x}} = \bar{g}(\bar{x}) \tag{18}$$

is termed an attractor (repeller) if no (at least one) eigenvalue of the matrix $\mathcal{M}$,

$$\mathcal{M} = \partial \bar{g}(\bar{x}_{\text{eq}})/\partial \bar{x}, \tag{19}$$

has a positive real part. Typically, dynamical systems such as (18) obey the Lipschitz condition

$$|\partial \bar{g}(\bar{x}_{eq})/\partial \bar{x}| < \infty, \tag{20}$$

which guarantees the existence of a unique solution for each initial condition $\bar{x}(0)$. Theoretically, the system relaxation time to an attractor and escape time from a repeller is infinite, because the transient solution cannot intersect the corresponding solution to which it tends.

Zak, Barhen, and Toomarian (Refs. 5-9) have used the concept of terminal attractors and repellers in the context of neural network dynamics to obviate the infinite-time solution limitations of regular attractors and repellers. Based on the violation of the Lipschitz condition at equilibrium points, these points induce singular solutions such that each solution approaches the terminal attractor or escapes from the terminal repeller in finite time.

For example, the system

$$\dot{x} = -x^{1/3} \tag{21}$$

has an attracting equilibrium point at $x = 0$ which violates the Lipschitz condition,

$$|d\dot{x}/dx| = |-1/3x^{-2/3}| \to \infty, \qquad \text{as } x \to 0. \tag{22}$$

The attractor is termed terminal, since from any initial condition $x_0 \neq 0$, the dynamical system in (21) reaches the equilibrium point $x = 0$ in a finite time,

$$t_0 = -\int_{x_0}^{x \to 0} x^{-1/3}\, dx = (3/2)x_0^{2/3}. \tag{23}$$

Similarly, the dynamical system:

$$\dot{x} = x^{1/3} \tag{24}$$

has a repelling unstable equilibrium point at $x = 0$ which violates the Lipschitz condition. Any initial condition which is infinitesimally close to the repelling point $x = 0$ will escape the repeller, to reach point x_0 in a finite time,

$$t_0 = \int_{\epsilon \to 0}^{x_0} x^{-1/3}\, dx = (3/2)x_0^{2/3}. \tag{25}$$

The behavior of the terminal attractor and repeller is shown in Fig. 5. Terminal repellers, in conjunction with the subenergy tunneling function introduced above, form the basis of our global optimization algorithm.

3.3. TRUST Algorithm: One-Dimensional Case. We now assemble the above concepts into the TRUST global optimization scheme. For simplicity, the case of one-dimensional optimization is considered first. Section 5 discusses the multi-dimensional case.

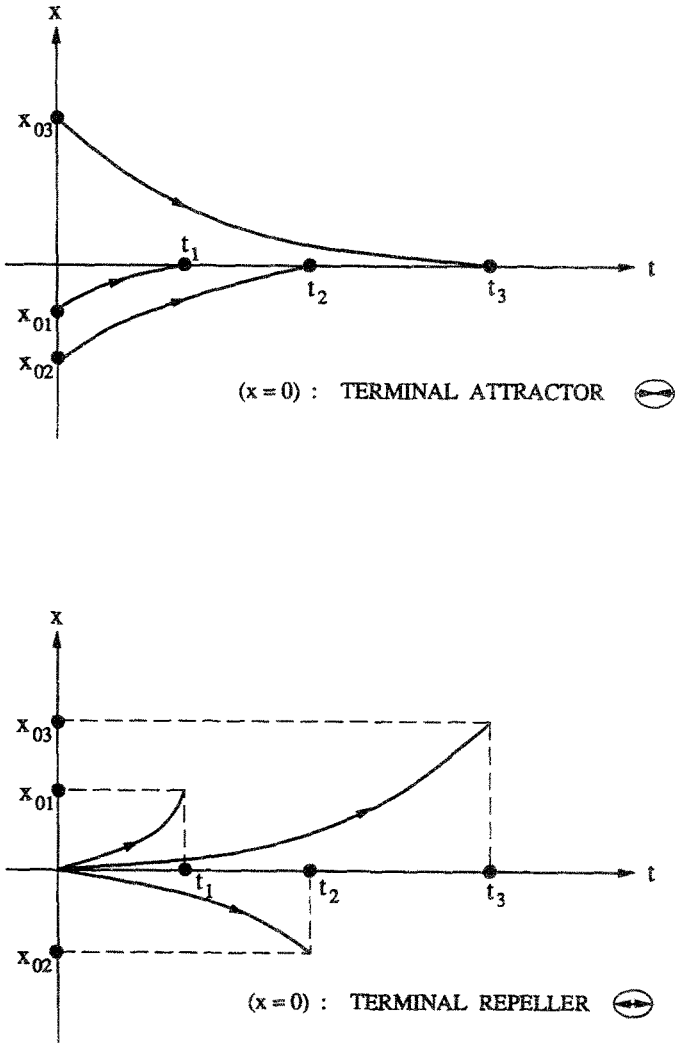

Fig. 5. Behavior of terminal attractor and repeller.

Let $f(x)$ be a scalar function which is to be globally minimized over a given interval. Define a new cost function to be minimized,

$$\begin{aligned} E(x, x^*) &= \log(1/[1+\exp(-(\hat{f}(x)+a))]) \\ &\quad -(3/4)k(x-x^*)^{4/3}u(\hat{f}(x)) \\ &= E_{\text{sub}}(x, x^*) - kE_{\text{rep}}(x, x^*)u(\hat{f}(x)). \end{aligned} \tag{26}$$

The Heaviside step function $u(\cdot)$ was defined in (9), and $\hat{f}(x) = f(x) - f(x^*)$ as in (13). The first term in the right-hand side of Eq. (26) corresponds to the subenergy function; the second term is referred to as the repeller energy term, i.e., a term which when differentiated will yield an expression of the form (24). The parameter $k > 0$ is referred to as the power of the repeller. The selection of its value will be addressed below.

Application of gradient descent to $E(x, x^*)$ in (26) results in the dynamical system

$$\begin{aligned}\dot{x} &= -\partial E(x, x^*)/\partial x \\ &= -(\partial f(x)/\partial x)(1/[1+\exp(\hat{f}(x)+a)]) + k(x-x^*)^{1/3}u(\hat{f}(x)) \\ &\quad + (3/4)k(x-x^*)^{4/3}\delta(\hat{f}(x)). \end{aligned} \tag{27}$$

The third term in the r.h.s. of Eq. (27) is identically zero for any x. Consequently, (27) simplifies to

$$\dot{x} = -(\partial f(x)/\partial x)(1/[1+\exp(\hat{f}(x)+a)]) + k(x-x^*)^{1/3}u(\hat{f}(x)). \tag{28}$$

Equation (28) represents gradient descent on $E(x, x^*)$; therefore, its equilibrium state will be a local minimizer of $E(x, x^*)$.

To qualitatively discuss the behavior of this sytem, we refer to the components of (28) as follows:

$$1/[1+\exp(\hat{f}(x)+a)] = \text{gradient multiplier},$$

$$-(\partial f(x)/\partial x)(1/[1+\exp(\hat{f}(x)+a)]) = \text{subenergy gradient},$$

$$k(x-x^*)^{1/3}u(\hat{f}(x)) = \text{repeller term}.$$

The dynamical system (28) autonomously switches between the following two phases:

Phase I. This phase, which is effectively a tunneling phase, is characterized by $f(x) \geq f(x^*)$. Since the gradient multiplier rapidly tends toward zero for increasing $\hat{f}(x)$, the subenergy gradient magnitude is nearly zero,

$$\partial E_{\text{sub}}(x, x^*)/\partial x \simeq 0.$$

In other words, the subenergy function is nearly flat and approximately zero in magnitude in the vicinity of x. Since the subenergy gradient magnitude is negligible compared to the magnitude of the repeller term, in this phase (28) behaves approximately as

$$\dot{x} \simeq k(x-x^*)^{1/3}.$$

Thus, the dynamical system (28) is repelled from x^* across the surface of the flattened subenergy tunneling function, until $f(x) < f(x^*)$. In effect, this phase tunnels through portions of $f(x)$ where $f(x) \geq f(x^*)$.

Phase II. In this phase, which is a minimization phase, $f(x) < f(x^*)$. The gradient multiplier term has approximately unit magnitude, and the repeller term is identically zero. Thus, (28) behaves approximately as

$$\dot{x} \simeq -\partial f(x)/\partial x. \tag{29}$$

This phase implements minimization via gradient descent.

In summary, (28) behaves approximately as:

$$\dot{x} \simeq \begin{cases} k(x-x^*)^{1/3}, & f(x) \geq f(x^*), \\ -\partial f(x)/\partial x, & f(x) < f(x^*). \end{cases} \tag{30}$$

A more detailed analysis of the TRUST algorithm represented by (28) is considered below.

3.4. Initial Conditions and Overview of the TRUST Algorithm Operation. In the one-dimensional case,

$$\mathcal{D} = [x_L \leq x \leq x_U].$$

To initiate optimization, x^* is chosen to be one of the boundary points of $\mathcal{D}$. In effect, a repeller is placed at x^*, and the dynamical system in (28) is given initial conditions $x^* + \epsilon$, where ϵ is a small perturbation which drives the system into the domain of interest.

Remark 3.1. Consistency in the flow direction is necessary, i.e., ϵ is of constant sign throughout a particular optimization. A system will be termed "positive flow" if it is initiated at x_L and $\epsilon > 0$ is consistently chosen. Likewise, a system is termed "negative flow" if initiated at x_U and $\epsilon < 0$ is consistently chosen.

The selection of x^* defines a zero subenergy limit $f(x^*)$ above which $E_{\text{sub}}(x, x^*)$ is nearly zero in value and approximately flat. If $f(x^* + \epsilon) < f(x^*)$, the system immediately enters a gradient descent phase (phase II above), which equilibrates at $x = x^{1(*)}$. Typically, $x^{1(*)}$ is a local minimum, though it could be an inflection point (or saddle point in higher dimensions). We refer to $x^{1(*)}$ as a lower critical point. Here, we assume that it is a local minimum, though the case of an inflection point is considered in the sequel.

We then set $x^* = x^{1(*)}$ in (28), and perturb x to $x^* + \epsilon$. Since $x^{1(*)}$ is a local minimum, $f(x) \geq f(x^*)$ in a neighborhood of x^*. Consequently, the repelling term is active in this phase (phase I above). Although the gradient of the objective function is uphill, the associated subenergy surface is essentially flat in the vicinity of x^*. If the magnitude of k is chosen sufficiently large (see below), the repeller located at x^* repels the system across the flattened subenergy surface, which in effect pushes the system up the hill of the associated objective function surface. The dynamical system remains

in the repelling phase until it reaches a lower basin of attraction, where $\hat{f}(x) < 0$. In effect, this phase tunnels through all of the state space region with functional values that lie above that of the the last found lower critical point $f(x^{1(*)})$.

As the dynamical system enters the next basin, $\hat{f}(x) < 0$, and the algorithm automatically switches to gradient descent, leading to minimization of $f(x)$. The system will equilibrate at the next lower local minimum $x^{2(*)}$. We set $x^* = x^{2(*)}$ and repeat the process. This is shown graphically in Fig. 6.

If $f(x^* + \epsilon) \geq f(x^*)$ when the optimization procedure is initiated, (28) is initially in a tunneling phase. The tunneling will proceed to a lower basin, at which point it enters a gradient descent phase and follows the behavior discussed above.

A sufficient value of k to ensure tunneling can be determined as follows. After reaching a critical point x^*, the zero-energy limit is reset, effectively placing a repeller at the minimum x^*. The dynamical system is restarted with initial condition $x_0 = x^* + \epsilon$, where $\epsilon > 0$ (assuming positive flow). The repeller need only be strong enough to push the system over the relatively flattened surface. If x^* is an inflection point, then any positive value of k

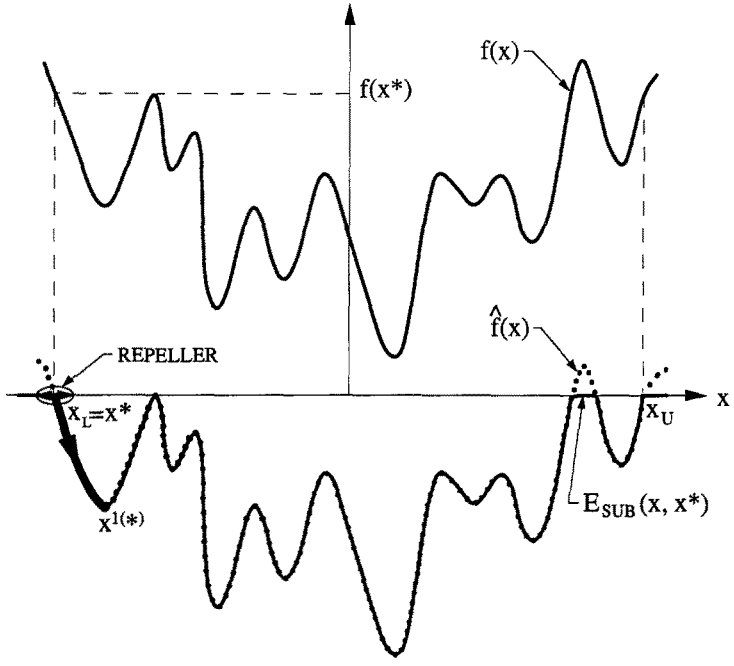

Fig. 6A. Schematic of TRUST operation (Cycle I).

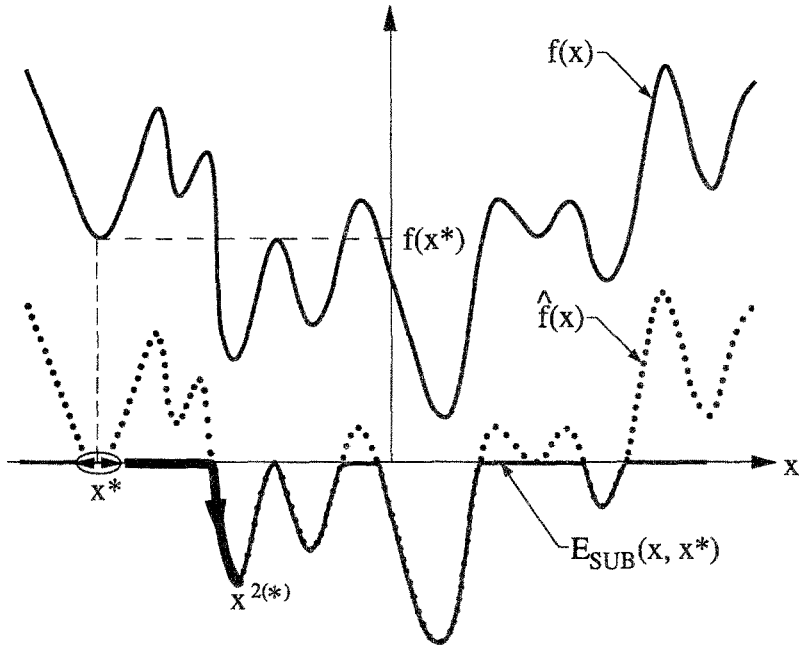

Fig. 6B. Schematic of TRUST operation (Cycle II).

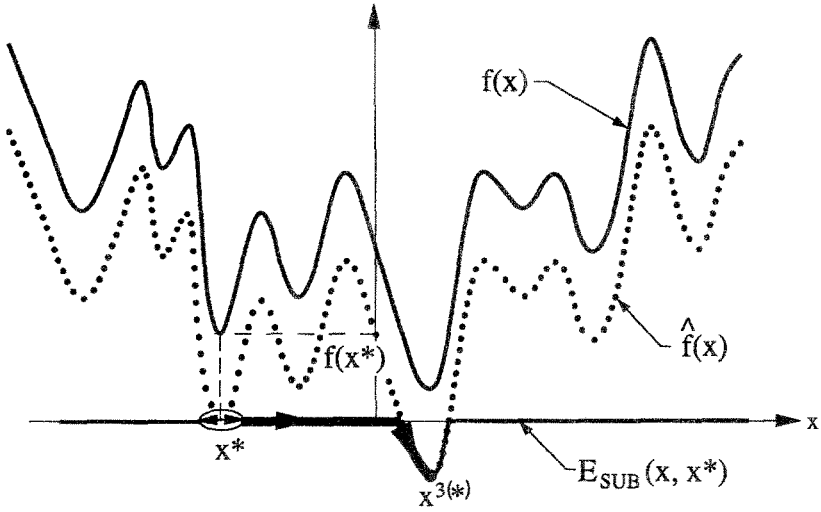

Fig. 6C. Schematic of TRUST operation (Cycle III).

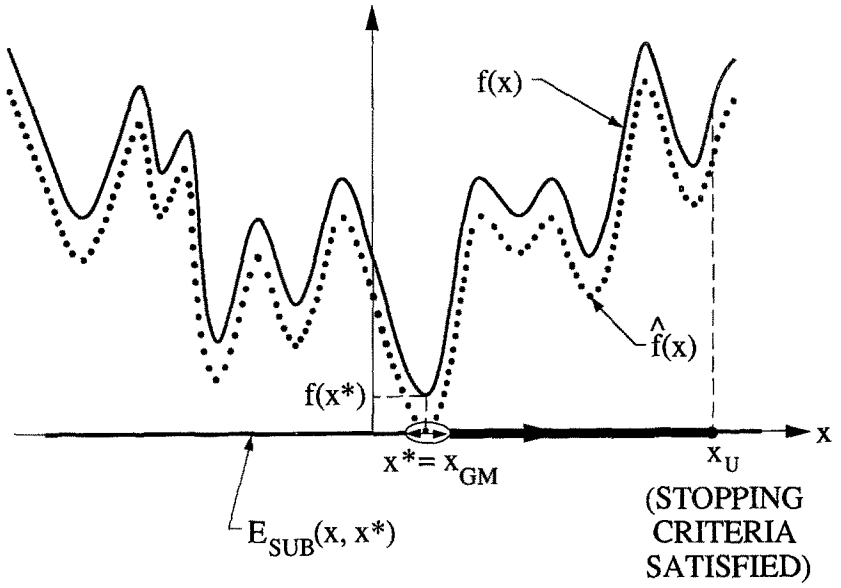

Fig. 6D. Schematic of TRUST operation (Cycle IV).

is sufficient. If x^* is a local minimum, then for $\dot{x}$ to be positive when the positive flow dynamical system is restarted at the perturbed location $x_0 = x^* + \epsilon$, the following condition must be satisfied:

$$k(x_0 - x^*)^{1/3} > (\partial f(x_0)/\partial x)(1/[1+\exp(\hat{f}(x_0)+a)]). \tag{31}$$

A sufficient condition to satisfy (31) is that

$$\begin{aligned} k &> (1/[\epsilon^{1/3}(1+\exp(a))])(\partial f(x_0)/\partial x) \\ &\simeq (\epsilon^{2/3}/[1+\exp(a)])(\partial^2 f(x^*)/\partial x^2). \end{aligned} \tag{32}$$

Note that ϵ is typically a small number, like 0.001 or 0.01; hence, necessary values of k are typically very reasonable. Thus, stiffness considerations in the integration of (28) do not arise from the choice of k. For example, for $\epsilon = 0.01$ and $a = 2$, a value of k such that $k > 0.0056(\partial^2 f(x^*)/\partial x^2)$ is sufficiently large to ensure proper tunneling behavior.

During the remainder of the tunneling phase, we need only ensure that, at any point x',

$$k > [1/(x' - x^*)^{1/3}](\partial f(x')/\partial x)(1/[1+\exp(a)]), \tag{33}$$

since $\hat{f}(x) \geq 0$ during tunneling. Note that the value of k computed using

(32) at the beginning of the tunneling process is almost always sufficiently large for the entire tunneling process. The gradient multiplier term decreases at an exponential rate with respect to increasing $\hat{f}(x)$. Thus, $f(x)$ must increase at a rate faster than exponential to ever require an increase in the value of k over the value computed at x_0 in (32); i.e., generally (32) is sufficient for (33). A similar analysis of the negative flow case shows that (32) and (33) hold in this case as well.

TRUST's implementation of tunneling as a repeller-induced flow over a subenergy surface has a number of advantages over other tunneling methods. First, the tunneling operation is algorithmically and computationally quite simple. Second, if $f(x^{2(*)}) = f(x^{1(*)})$, the associated subenergy surface is still flat, and the system tunnels past this local minimum or inflection point into a basin with a lower local minimum. This feature eliminates the difficulty with multiple poles in the tunneling algorithm of Levy and Montalvo. Third, convergence of the gradient descent phase to an inflection point does not cause a problem, as the dynamical system will escape the inflection point during the next gradient descent phase.

It must be stressed that TRUST was developed to be implemented in continuous analog circuitry, where the integration of (28) is stable. In digital computer implementation, some care must be exercised during the numerical integration of (28) to ensure that a basin of attraction is not jumped over due to the finite-step-length integration of (28). Determination of an appropriate stepsize could follow from Ref. 10. Finally, the TRUST tunneling method will always reach a point in the adjacent basin of attraction with lower functional values. Other tunneling methods which find the zeros of a tunneling function are not guaranteed to find the most adjacent tunneling point, and therefore have complicated and less reliable stopping criteria. The TRUST stopping criterion is outlined below, and a more detailed examination of the convergence behavior of TRUST is given in Section 4.

3.5. Stopping Criteria. The successive minimization and tunneling computational processes continue until a suitable stopping criterion is satisfied. For the one-dimensional case, the stopping criterion is quite simple. As soon as a local minimum x_{LM} in $\mathcal{D}$ has been reached, the optimization cycle is repeated by placing a repeller at x_{LM} and perturbing the system to initiate the next tunneling phase. If x_{LM} were the lowest local minimum (i.e., if $x_{LM} = x_{GM}$), the subenergy transformation would flatten $f(x)$ in the entire domain of interest, since $f(x_{GM})$ sould be the lowest objective function value in $\mathcal{D}$. The perturbed dynamical system, which is now in a repeller tunneling phase, will eventually flow beyond the upper boundary of $\mathcal{D}$. Assuming positive flow, when the state flows out of the domain boundary, $x > x_U$, the last local minimum found is taken as the global minimum.

4. Analysis of One-Dimensional Convergence

We now examine the convergence of the TRUST algorithm in light of the above discussion. In the one-dimensional case, we seek to globally minimize $f(x)$, a twice differentiable function, over the domain $\mathscr{D} = [x_L, x_U]$. To show that TRUST will converge, under the assumptions of Section 1, to a global minimum (if one exists), we analyze its behavior during the different phases of operation. The analysis proceeds as follows. First, the tunneling behavior of TRUST is considered, assuming a local minimum has been found (after an initialization phase). We show that, from a local minimum, the tunneling phase of TRUST reaches a point of the same functional value in an adjacent basin of attraction of a lower critical point, or flows to a boundary of $\mathscr{D}$ if no such point exists. Next, we show the obvious result that the gradient descent behavior of TRUST will converge to a lower critical point. An inductive analysis of these two phases leads to the global minimization behavior and stopping criterion. Finally, we consider the initialization of the TRUST algorithm, showing that, from all possible initial conditions, TRUST will reach the first effective local minimum, if it exists, or flow out of $\mathscr{D}$. The case of inflection points considered throughout the discussion as necessary.

Let us first consider the tunneling behavior of TRUST after finding the jth local minimum $x^{j(*)}$ of $f(x)$. To simplify the discussion, introduce the following notation. Let

$$\mathscr{D}_U(x^{j(*)}) = (x^{j(*)}, x_U] \quad \text{and} \quad \mathscr{D}_L(x^{j(*)}) = [x_L, x^{j(*)})$$

respectively be termed the lower and upper domains of $x^{j(*)}$. Let $S_L(x^{j(*)})$ and $S_U(x^{j(*)})$ respectively denote the sets of lower and upper tunneling points of $x^{j(*)}$,

$$S_L(x^{j(*)}) = \{x \in \mathscr{D}_L \mid f(x) = f(x^{j(*)})\}, \tag{34a}$$

$$S_U(x^{j(*)}) = \{x \in \mathscr{D}_U \mid f(x) = f(x^{j(*)})\}. \tag{34b}$$

That is, $S_L(x^{j(*)})$ and $S_U(x^{j(*)})$ are points with the same functional values as $f(x^{j(*)})$. Note that $S_L(x^{j(*)})$, or $S_U(x^{j(*)})$, or possibly both are empty sets depending on chosen direction of flow. If $S_L(x^{j(*)})$ or $S_U(x^{j(*)})$ are not empty, define the adjacent lower and upper tunneling points as follows:

$$x_{A_L} = \min_{x \in S_L(x^{j(*)})} \|x - x^{j(*)}\|, \tag{35a}$$

$$x_{A_U} = \min_{x \in S_U(x^{j(*)})} \|x - x^{j(*)}\|. \tag{35b}$$

If either $S_L(x^{j(*)})$ or $S_U(x^{j(*)})$ are empty, define the adjacent tunneling

points respectively as

$$x_{A_L} = x_L, \tag{36a}$$

$$x_{A_U} = x_U. \tag{36b}$$

Now define the lower and upper tunneling intervals,

$$\mathscr{D}_{T_L}(x^{j(*)}) = [x_{A_L}, x^{j(*)}), \tag{37a}$$

$$\mathscr{D}_{T_U}(x^{j(*)}) = (x^{j(*)}, x_{A_U}]. \tag{37b}$$

Finally, we construct the tunneling interval $\mathscr{D}_T(x^{j(*)})$ as follows:

$$\begin{aligned}\mathscr{D}_T(x^{j(*)}) &= \mathscr{D}_{T_L}(x^{j(*)}) \bigcup \mathscr{D}_{T_U}(x^{j(*)}) \bigcup \{x^{j(*)}\} \\ &= \{x \in \mathscr{D}(x) \,|\, x_{A_L} \leq x \leq x_{A_U}\}.\end{aligned} \tag{38}$$

That is, the tunneling region $\mathscr{D}_T(x^{j(*)})$ is the connected interval containing $x^{j(*)}$ and whose endpoints are either points with the same functional value (and thus points for initiating a subsequent local optimization phase) or a boundary of $\mathscr{D}$. Note that $f(x)$ may assume local minima, maxima, and inflection points in $\mathscr{D}_T(x^{j(*)})$, though

$$f(x) \geq f(x^{j(*)}), \qquad x \in \mathscr{D}_T(x^{j(*)}).$$

We wish to show that the dynamical system (28) is unstable on $\mathscr{D}_T(x^{j(*)})$ and will flow toward the boundary of this interval (thus performing the tunneling operation, or satisfying the stopping criterion). To do this, we define a Lyapunov energy function

$$\tilde{E}(x(t), x^{j(*)}) = (3/4)k(x(t) - x^{j(*)})^{4/3}. \tag{39}$$

We note that $\tilde{E}(x)$ is positive definite on $\mathscr{D}_{TL}(x^{j(*)})$ and $\mathscr{D}_{T_U}(x^{j(*)})$, and is positive semidefinite in $\mathscr{D}_T(x^{j(*)})$, assuming a zero value only at $x^{j(*)}$. Further, note that $\tilde{E}(x, x^{j(*)})$ is a strictly increasing function of $\|x - x^{j(*)}\|$ on both $\mathscr{D}_{T_L}(x^{j(*)})$, and respectively assumes its maximum values on the lower and upper boundaries of $\mathscr{D}_{T_L}(x^{j(*)})$ and $\mathscr{D}_{T_U}(x^{j(*)})$. The time derivative of $\tilde{E}(x(t), x^{j(*)})$ is

$$\begin{aligned}&(d/dt)\tilde{E}(x, x^{j(*)}) \\ &= k(x - x^{j(*)})^{1/3}\dot{x} \\ &= k^2(x - x^{j(*)})^{2/3} - k(\partial f(x)/\partial x)(1/[1 + \exp(\hat{f}(x) + a)])(x - x^{j(*)})^{1/3};\end{aligned} \tag{40}$$

k is a positive constant; from the discussion in Section 3.4, the value of k for positive flow is chosen so that $\dot{x} > 0$ on $\mathscr{D}_{T_U}(x^{j(*)})$. Similarly, for negative flow, k is chosen so that $\dot{x} < 0$ on $\mathscr{D}_{T_L}(x^{j(*)})$. Note that, for both cases (i.e., positive and negative flow), the same sufficient condition for k in (32) holds. Thus, $(d/dt)\tilde{E}(x, x^{j(*)})$ is positive on $\mathscr{D}_{T_L}(x^{j(*)})$ and $\mathscr{D}_{T_U}(x^{j(*)})$;

$(d/dt)\tilde{E}(x, x^{j(*)})$ assumes a zero value only at $x^{j(*)}$. This implies that $\|x - x^{j(*)}\|$ must also be increasing with time on $\mathscr{D}_{T_L}(x^{j(*)})$ or $\mathscr{D}_{T_U}(x^{j(*)})$. That is, from any initial condition in $\mathscr{D}_{T_L}(x^{j(*)})$, (28) will flow to x_{A_L} (negative flow). Similarly, from any initial condition in $\mathscr{D}_{T_U}(x^{j(*)})$, (28) will flow to x_{A_U} (positive flow).

Hence, we have just shown that (28), when perturbed to $x^{j(*)} + \epsilon$, will flow to a point $x^{j(0)}$ whose functional value is just below $f(x^{j(*)})$, i.e., $f(x^{j(0)}) \lesapprox f(x^{j(*)})$. If no such point exists, the system will flow to the boundary of $\mathscr{D}$. Also note that the analysis shows that any nonzero perturbation size ϵ leads to correct tunneling behavior. Further, because of the properties of the terminal repellers, the tunneling flow must occur in finite time.

We also need to consider the behavior of the tunneling phase if $x^{j(*)}$ is actually an inflection point, and not a local minimum. Assume that the inflection point $x^{j(*)}$ was reached by a minimization phase which originated in $\mathscr{D}_L$ (i.e., from a positive flow system). In this case, $\mathscr{D}_{T_U}(x^{j(*)})$ is a zero length interval. A small perturbation $x^{j(*)} + \epsilon$ will put TRUST in another gradient descent phase. Similarly, if $f(x)$ is infinitely degenerate, and thus flat in $\mathscr{D}_U(x^{j(*)})$, the repeller-induced flow will push the system over the degenerate interval.

Next, consider the behavior of the TRUST dynamical system in a gradient descent phase. Assume that a tunneling phase has been completed, and we are at point $x^{j(0)}$. This point must be within a basin of attraction of a lower local critical point, e.g., such that $|\partial f(x)/\partial x| \neq 0$ and $f(x^{j(0)}) \lesapprox f(x^{j(*)})$ holds. The dynamical system (28) then becomes

$$\dot{x} = -(\partial f/\partial x)(1/[1 + \exp(f(x) - f(x^{j(*)}) + a)]). \tag{41}$$

Again, we can analyze the convergence properties of this system by defining a Lyapunov energy function,

$$\hat{E}(x) = f(x) - f(x^{j+1(*)}),$$

where $x^{j+1(*)}$ is the next adjacent lower critical point of $f(x)$ and $\hat{E}(x)$ is defined on the interval $[x^{j(0)}, x^{j+1(*)}]$.

The time derivative of $\hat{E}(x)$ in the domain is

$$(d/dt)\hat{E}(x) = (\partial f(x)/\partial x)\dot{x} = -(\partial f/\partial x)^2(1/[1 + \exp(\hat{f}(x) + a)]), \tag{42}$$

which is a negative semidefinite function, assuming zero value only at $\partial f(x)/\partial x = 0$. Thus, from $x^{j(0)}$, the dynamical system (28) will converge to a lower critical point $x^{j+1(*)}$, where we reset x^* to $x^{j+1(*)}$ and repeat the same process, and the above analysis procedure holds.

Thus, the above analysis has shown that, starting from a local minimum or inflection point $x^{j(*)}$ and applying the algorithm outlined in Section 3, Eq. (28) will converge to another local minimum or inflection point $x^{j+1(*)}$, or flow out of $\mathscr{D}$ if there are no lower minima. We call $x^{j+1(*)}$ the next

effective local minimum, as there may be many local minima located between $x^{j(*)}$ and $x^{j+1(*)}$, but these lower minima have functional values greater than $f(x^{j(*)})$. Thus, by the inductive analysis of the two above phases, TRUST (assuming a positive flow system) will find a sequence of effective minima,

$$x^{1(*)} < x^{2(*)} < \cdots < x^{l(*)}, \tag{43}$$

such that

$$f(x^{1(*)}) > f(x^{2(*)}) > \cdots > f(x^{l(*)}). \tag{44}$$

From $x^{l(*)}$, (28) will flow to x_U, and we know from the above discussion that no lower local minima can exist in the interval $(x^{l(*)}, x_U]$. Thus, the last local minimum found must be the global minimum.

The above inductive analysis assumed that the TRUST algorithm was initiated at local minimum $x^{l(*)}$. We now turn to the operation of TRUST from its initial conditions, to show that it will converge to the first effective local minimum $x^{1(*)}$, if it exists. From there, the previous inductive analysis holds. Assume a positive flow system (a similar analysis holds for negative flow). Several possible different conditions at x_L have to be considered.

Case 1. x_L is a local minimum. The above analysis holds immediately.

Case 2. x_L is an inflection point. If $f(x)$ is increasing in a positive flow neighborhood of x_L, then an upper tunneling region exists. Initiation of (28) at $x_L + \epsilon$ will initiate a tunneling phase, which as shown above will either flow out of the domain $\mathcal{D}$ if no global minimum [that satisfies the local minima constraints (3) and (4)] exists, or will reach a point where subsequent gradient descent converges to the first effective local minimum. If $f(x)$ is decreasing in a positive flow neighborhood of x_L, then the system enters a gradient descent phase, which will converge to a lower local critical point.

Case 3. x_L is a local maximum. Initiating (28) at $x_L + \epsilon$ puts (28) in a gradient descent phase, which will converge to $x^{1(*)}$.

Case 4. $\partial f(x)/\partial x > 0$ at x_L. An upper tunneling region $D_{T_U}(x_L)$ exists. According to the previous analysis, perturbing x to $x_L + \epsilon$ will cause (28) to reach either an adjacent tunneling point, where subsequent gradient descent will find the first effective local minimum or inflection point $x^{1(*)}$, or will flow to x_U if in fact $f(x_L)$ is the lowest value $f(x)$ assumes in $\mathcal{D}$.

Case 5. $\partial f(x)/\partial x < 0$ at x_L. At $x_L + \epsilon$, (28) immediately enters a gradient descent phase, converging to the first effective local minimum or

inflection point, if one exists; else, gradient descent will flow to x_U if no such point exists in $\mathcal{D}$.

Thus, in the continuous case and under the assumptions in Section 1, TRUST is guaranteed to find the global minimum in a one-dimensional interval. If the function is degenerate (i.e., several global minima), TRUST will determine only the first encountered global minimum. In order to locate the consequent global minima, we iteratively reset x_L to $x_{GM}+\epsilon$ and restart there.

5. TRUST Algorithm: Multi-Dimensional Case

The one-dimensional algorithm of Sections 3 can be extended to handle multi-dimensional global optimization, though convergence to the global minimum is not absolutely guaranteed. Let $f(\bar{x})$ be a function of the $n\times 1$ state vector $\bar{x}$, and define the multi-dimensional functional

$$\begin{aligned} E(\bar{x},\bar{x}^*) &= \log(1/[1+\exp(-(\hat{f}(\bar{x})+a))]) \\ &\quad - k(3/4)\sum_{j=1}^{n}(x_j-x_j^*)^{4/3}u(\hat{f}(\bar{x})) \\ &= E_{\text{sub}}(\bar{x},\bar{x}^*)+kE_{\text{rep}}(\bar{x},\bar{x}^*)u(\hat{f}(\bar{x})). \end{aligned} \tag{45}$$

The multi-dimensional subenergy term is analogous to the one-dimensional subenergy function. The portions of the objective function surface which lie above the zero subenergy limit $f(\bar{x}^*)$ are flattened by the use of the subenergy function (as shown in Fig. 4).

Upon application of gradient descent to $E(\bar{x},\bar{x}^*)$ in (45), we obtain the dynamical system

$$\dot{x}_j = -(\partial f(\bar{x})/\partial x_j)(1/[1+\exp(\hat{f}(\bar{x})+a)])+k(x_j-x_j^*)^{1/3}u(\hat{f}(\bar{x})), \tag{46}$$

where x_j denotes the jth component of $\bar{x}$. Equation (46) has a highly parallel structure consisting of n weakly coupled differential equations. This dynamical system is analogous to the dynamical system described by Eq. (28). The initial conditions, operation, and stopping criterion for Eq. (46) are also highly analogous to those discussed above.

In the multi-dimensional case, $\bar{x}^*$ is initially chosen to be one corner of the hyperparallelepiped $\mathcal{D}$, usually $x_i^*=x_{iL}$, $\forall i$. A repeller is placed at $\bar{x}^*$. It should be noted that the repelling terms in the multi-dimensional case can be interpreted as hyperplane repellers and are active whenever $\hat{f}(\bar{x})\geq 0$. The initial state of the system is set to $\bar{x}^*+\bar{\epsilon}$, where $\bar{\epsilon}$ is a small perturbation which drives the system into $\mathcal{D}$. We assume that $\bar{\epsilon}$ has uniform sign during the optimization, analogous to the consistent positive or negative

flow operation of the one-dimensional algorithm. Depending upon the relative values of $f(\bar{x}^*)$ and $f(\bar{x}^* + \bar{\epsilon})$, the dynamical system will initially be in a tunneling phase or a gradient descent phase. These phases are analogous to the one-dimensional case. An appropriate value for the repeller power k can be determined by analogy to (31)-(33). The multi-dimensional stopping criterion is also similar to the one-dimensional case. When the system state flows out of the domain boundaries, the last local minimum found is taken as the global minimum.

Theoretically, convergence of the method to a global minimum is not formally guaranteed in the multidimensional case due to the constant perturbation direction vector $\bar{\epsilon}$. However, in practice, due to its global descent property, the system dynamics escapes local minima valleys with help of the repeller effect, and flows into lower valleys of the error energy function using the information it gets from the gradient term.

6. Benchmarks and Comparison to Other Methods

This section presents results of benchmarking tests carried out for the TRUST algorithm using several standard one- and multi-dimensional test functions taken from the literature. In Tables 1-4, the performance of TRUST is compared to well-known global optimization procedures. Specifically in Tables 3 and 4, TRUST is compared against the best competing global optimization methods, where the term "best" indicates the best widely reported results the authors could find for the particular benchmark test function. The criteria for comparison is the number of function evaluations. For the TRUST algorithm, the function evaluation count includes every iteration from the initial conditions to the satisfaction of the stopping

Table 1. Comparison of TRUST and other algorithms based on number of function evaluations.

	Method					
Function	SM	TM	DT	IM	FFA	TRUST
1(i)	10822	1496	1469			168
1(ii)	10822	1496	1132			168
1(iii)	10822	1496				32
1(iv)					375	76
2(i)	241215	12160	6000	7424		588
2(ii)	241215	12160	6000	7424		269
2(iii)					408	256

Table 2. Comparison of TRUST and other algorithms based on number of function evaluations.

	Method			
Function	SM	DT	FSA	TRUST
3(i)				38
3(ii)		1414		22
3(iii)				21
3(iv)		7871	9228	21
4(i)	19940			74
4(ii)				58
5(i)	7390			40
5(ii)	4853			94
5(iii)	8235			163
5(iv)	27859			1449

Table 3. Comparison of TRUST and other algorithms based on number of function evaluations.

	Method						
Function	SA	MRS	P	CRS	SCA	MLSL	TRUST
6	5917	1176	179				77
7		160	133	1800	1558	206	60

Table 4. Comparison of TRUST and other algorithms based on number of function evaluations.

	Method					
Function	PIJ	BAT	STR	ZIL	BRE	TRUST
8	462	120	45	33	25	19
9(i)	3817	816	150	125	161	69
9(ii)	3817	816	150	125	161	99

criterion outlined in Section 3.5. We note that, in every benchmark, TRUST converged to the global minimum.

In accordance with Section 3.1, the constant a assumes the value $a = 2$ in the sequel. Furthermore, Eq. (28) was integrated using a simple Euler integration scheme; that is,

$$\dot{\bar{x}} = [\bar{x}(k+1) - \bar{x}(k)]/\Delta t = -\tau[\partial E(\bar{x}, \bar{x}^*)/\partial \bar{x}], \tag{47}$$

where Δt is the stepsize. The time constant τ is taken to be 1 in all cases studied here. For highly nonlinear and stiff objective functions, more robust integration schemes are preferable (Refs. 6 and 7). We note that, for Euler integration, the selection of the integration stepsize must be done carefully to ensure stability. We do not provide an analysis of the stepsize in this paper, since (as we have previously stated) our ultimate goal is implementation of this algorithm in continuous analog VLSI circuitry (Ref. 1), where such considerations do not apply.

A description of each test function, the relevant initial conditions, domain of interest $\mathcal{D}$, TRUST parameters, and integration stepsize are given in the Appendix. In Tables 1 and 2, the following abbreviations are used: SM is the stochastic method of Aluffi-Pentini (Ref. 11); TM is the tunneling method of Ref. 3; DT is the dynamic tunneling method presented in Ref. 4; IM is the interval methods of Walster (Ref. 12); FFA is the filled function approach of Ref. 13; and FSA is the fast simulated annealing method of Ref. 14.

In Table 3, SA is an abbreviation of simulated annealing (Ref. 15); MRS is the multiple random start method (Ref. 16); P is an abbreviation of the P-algorithm of Zilinskas (Ref. 17); CRS is the controlled random search of Price (Ref. 18); SCA is the search clustering approach of Törn (Ref. 19); and MLSL is the multi-level single linkage method of Timmer (Ref. 2).

In Table 4, PIJ, BAT, STR, ZIL, and BRE are respectively abbreviations for the results of Pijavskij, Batishchev, Strongin, Zilinkskas, and Brent (Ref. 17).

7. Discussion and Conclusions

This paper has introduced TRUST, a novel deterministic methodology for unconstrained global function optimization, which combines the concept of terminal repellers with a new subenergy tunneling function. Global optimization is formulated as the solution to a system of deterministic differential equations which incorporate these novel features. The flow of

this dynamical system leads to global optimization. It was shown that, under very general assumptions (see Section 1), the algorithm is provably convergent to the global minimum in the one-dimensional case.

Benchmark comparisons (Section 6) with other global optimization procedures have demonstrated that TRUST is significantly faster, as measured by the number of function evaluations, than the best currently available methods for these standard functions. Furthermore, our algorithm systematically converged to the global minimum in all benchmark simulations, even in the multi-dimensional case.

The number of function evaluations is only one criterion to be used in comparing this algorithm with other algorithms. It is important to emphasize that TRUST has a number of other advantages. First, while the algorithm is not guaranteed to find the global minima in multiple dimensions, it does have a global descent property. It is thus practically useful for multi-dimensional problems. For n-dimensional functions, the algorithm can be computed as the parallel solution of n weakly coupled differential equations. Consequently, the complexity and computational cost of the algorithm is not strongly dependent upon the problem dimensionality. Second, this formulation naturally leads to a simple and computationally efficient stopping criterion. Third, TRUST is robust with respect to the basic algorithm parameters. Necessary conditions on the algorithm parameters were derived in Section 3.4. Finally, as also discussed there, the effective tunneling procedure employed in TRUST has a number of additional advantages over other deterministic tunneling methods.

Most importantly, the structure of our formulation makes it suitable for implementation in parallel analog VLSI circuits of the type used for artificial neural network architectures. Such a hardware implementation will lead to even more dramatic speed enhancements. For many applications, the algorithm may become real-time. In fact, analog VLSI circuits which implement terminal repellers and gradient descent have already been successfully designed, fabricated, and tested (Ref. 1). These circuits will be the subject of a forthcoming paper.

8. Appendix: Test Functions and Parameters Used in Benchmark Studies

The functions used in the benchmark studies of Section 6 are listed below. For the first function, we also summarize in tabular form the relevant parameters used in benchmark study. In this table, $\bar{x}_L$ and $\bar{x}_U$ are respectively lower and upper bounds of the domain of interest $\mathcal{D}$; $\bar{x}_I$ is the initial condition; $\bar{\epsilon}$ is the TRUST perturbation; Δt is the Euler integration stepsize; and k is the repeller power. The benchmarking parameters for the other

functions can be found in Ref. 20. In all simulations, TRUST used the same values for $\mathcal{D}$, $\bar{x}_I$, and Δt as the methods to which its performance is compared.

Function 1. Two-Dimensional 6-Hump Camelback Function:

$$f(x_1, x_2) = [4 - 2.1x_1^2 + (x_1^4/3)]x_1^2 + x_1x_2 + (-4 + 4x_2^2)x_2^2.$$

Number of local minima = 6;
number of global minima = 2;
global minimum found by TRUST:

$$[x_{1GM}, x_{2GM}] = [0.08983, -0.71265], \quad \text{for (i), (iv)},$$

$$[x_{1GM}, x_{2GM}] = [-0.08983, 0.71265], \quad \text{for (ii), (iii)}.$$

Function 2. Two-Dimensional Shubert Function:

$$f(x_1, x_2) = \left\{ \sum_{i=1}^{5} i \cos[(i+1)x_1 + i] \right\} \left\{ \sum_{i=1}^{5} i \cos[(i+1)x_2 + i] \right\}.$$

Number of local minima = 760;
number of global minima = 18;
global minimum found by TRUST:

$$[x_{1GM}, x_{2GM}] = [-7.08351, -7.70831], \quad \text{for (i), (ii)},$$

$$[x_{1GM}, x_{2GM}] = [-0.80032, -1.42513], \quad \text{for (iii)}.$$

Function 3. N-Dimensional Test Function:

$$f(\bar{x}) = (1/2) \sum_{j=1}^{N} (x_j^4 - 16x_j^2 + 5x_j),$$

$$\bar{x} = [x_1, x_2, \ldots, x_j, \ldots, x_N].$$

Number of local minima = 2^n;
number of global minima = 1;
global minimum found by TRUST:

$$[\bar{x}_{GM}] = [-2.90354, -2.90354, \ldots, -2.90354].$$

Table 5. Benchmark parameters for Function 1.

Trial	x_{1L}	x_{1U}	x_{2L}	x_{2U}	x_{1I}	x_{2I}	ϵ_1	ϵ_2	Δt	k
(i)	−3	3	−2	2	−3.0	−2.0	0.01	0.01	0.01	10
(ii)	−3	3	−2	2	3.0	2.0	−0.01	−0.01	0.01	10
(iii)	−3	3	−2	2	−2.0	−1.0	0.01	0.01	0.10	10
(iv)	−3	3	−2	2	−1.6	0.9	0.01	−0.01	0.10	10

Function 4. Two-Dimensional Test Function:

$$f(x_1, x_2) = 0.5x_1^2 + 0.5[1 - \cos(2x_1)] + x_2^2.$$

Number of local minima = several;
number of global minima = 1;
global minimum found by TRUST:

$$[x_{1GM}, x_{2GM}] = [0, 0].$$

Function 5. Two-Dimensional Test Function:

$$f(x_1, x_2) = 10^n x_1^2 + x_2^2 - (x_1^2 + x_2^2)^2 + 10^m (x_1^2 + x_2^2)^4, \qquad n = -m.$$

Number of local minima > 3;
number of global minima = 2;
global minimum found by TRUST:

$$[\bar{x}_{GM}] = [0, 1.38695], \qquad \text{for } n = 1,$$

$$[\bar{x}_{GM}] = [0, 2.60891], \qquad \text{for } n = 2,$$

$$[\bar{x}_{GM}] = [0, 4.70174], \qquad \text{for } n = 3,$$

$$[\bar{x}_{GM}] = [0, 8.39401], \qquad \text{for } n = 4.$$

Function 6. The Two-Dimensional Rastrigin Function:

$$f(x_1, x_2) = x_1^2 + x_2^2 - \cos(18x_1) - \cos(18x_2).$$

Number of local minima = 50;
number of global minima = 1;
global minimum found by TRUST:

$$[\bar{x}_{1GM}, x_{2GM}] = [0, 0].$$

Function 7. Two-Dimensional Branin Function:

$$f(x_1, x_2) = [x_2 - (5.1/4\pi^2)x_1^2 + (5/\pi)x_1 - 6]^2 + 10(1 - 1/8\pi)\cos x_1 + 10.$$

Number of local minima = 3;
number of global minima = 3;
global minimum found by TRUST:

$$[x_{1GM}, x_{2GM}] = [3.14158, 2.27505].$$

Function 8. One-Dimensional Test Function:

$$f(x) = \sin x + \sin(10x/3) + \log x - 0.84x.$$

Number of local minima = 3;
number of global minima = 1;
global minimum found by TRUST:

$$x_{GM} = 5.19978.$$

Function 9. One-Dimensional Test Function:

$$f(x) = -\left\{ \sum_{i=1}^{5} \sin[(i+1)x + i] \right\}.$$

Number of local minima = 20;
number of global minima = 3;
global minimum found by TRUST:

$$x_{GM} = -6.72004, \qquad \text{for (i)},$$

$$x_{GM} = \ 5.84633, \qquad \text{for (ii)}.$$

References

1. CETIN, B. C., KERNS, D. A., BURDICK, J. W., and BARHEN, J., *Analog Circuits for Terminal Attractors, Repellers and Gradient Descent*, Robotics and Mechanical Systems Report No. RMS-92-01, Department of Mechanical Engineering, California Institute of Technology, Pasadena, California, 1991.
2. KAN, A. H. G. R., and TIMMER, G. T., *A Stochastic Approach to Global Optimization*, Numerical Optimization, Edited by P. T. Boggs, R. H. Byrd, and R. B. Schnabel, SIAM, Philadelphia, Pennsylvania, pp. 245–262, 1985.
3. LEVY, A. V., and MONTALVO, A., *The Tunneling Algorithm for the Global Minimization of Functions*, SIAM Journal on Scientific and Statistical Computing, Vol. 6, pp. 15–29, 1985.
4. YAO, Y., *Dynamic Tunneling Algorithm for Global Optimization*, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 19, pp. 1222–1230, 1989.
5. ZAK, M., *Terminal Attractors in Neural Networks*, Neural Networks, Vol. 2, pp. 259–274, 1989.
6. ZAK, M., and BARHEN, J., *Neural Networks with Creative Dynamics*, Mathematical & Computer Modeling, Vol. 14, pp. 290–294, 1990.
7. BARHEN, J., TOOMARIAN, N., and GULATI, S., *Application of Adjoint Operators to Neural Learning*, Applied Mathematical Letters, Vol. 3, pp. 13–18, 1990.
8. BARHEN, J., ZAK, M., and TOOMARIAN, N., *Non-Lipschitzian Neural Dynamics*, Advanced Neural Computers, Edited by R. Eckmiller, North-Holland, Amsterdam, Holland, pp. 102–112, 1990.
9. BARHEN, J., ZAK, M., and TOOMARIAN, N., *Adjoint Operator Algorithms for Faster Learning in Neural Networks*, Advanced Neural Information Processing Systems, Vol. 2, pp. 498–508, 1990.
10. GRUVER, W. A., and SACHS, E., *Algorithmic Methods in Optimal Control*, Pitman Publishing, Melbourne, Australia, 1990.
11. ALUFFI-PENTINI, F., PARISI, V., and ZIRILLI, F., *Global Optimization and Stochastic Differential Equations*, Journal of Optimization Theory and Applications, Vol. 47, pp. 1–15, 1985.

12. WALSTER, G. W., HANSEN, E. R., and SENGUPTA, S., *Test Results for a Global Optimization Problem*, Numerical Optimization, Edited by P. T. Boggs, R. H. Byrd, and R. B. Schnabel, SIAM, Philadelphia, Pennsylvania, pp. 272–287, 1985.
13. GE, R., *A Filled Function Method for Finding a Global Minimizer of a Function of Several Variables*, Mathematical Programming, Vol. 46, pp. 191–204, 1990.
14. SZU, H., and HARTLEY, R., *Fast Simulated Annealing*, Physics Letters A, Vol. 122, pp. 157–162, 1987.
15. KIRKPATRICK, S., GELATT, C. D., and VECCHI, M. P., *Optimization by Simulated Annealing*, Science, Vol. 220, pp. 671–680, 1983.
16. BREMERMANN, H. A., *A Method of Unconstrained Global Optimization*, Mathematical Biosciences, Vol. 9, pp. 1–15, 1970.
17. TÖRN, A., and ZILINSKAS, A., *Global Optimization*, Springer-Verlag, Berlin, Germany, 1989.
18. PRICE, W. L., *A Controlled Random Search Procedure for Global Optimization*, Toward Global Optimization 2, Edited by L. C. W. Dixon and G.-P. Szegö, North-Holland, Amsterdam, Holland, 1978.
19. TÖRN, A. A., *A Search Clustering Approach to Global Optimization*, Toward Global Optimization 2, Edited by L. C. W. Dixon and G. P. Szegö, North-Holland, Amsterdam, Holland, 1978.
20. CETIN, B. C., BARHEN, J., and BURDICK, J. W., *Terminal Repeller Unconstrained Subenergy Tunneling (TRUST) for Global Optimization*, Robotics and Mechanical Systems Report No. RMS-90-03, Department of Mechanical Engineering, California Institute of Technology, Pasadena, California, 1990.