On the Global Convergence of Univariate Dynamic Encoding Algorithm for Searches (uDEAS)

Jong-Wook Kim, Taegyu Kim, Joon-Young Choi, and Sang Woo Kim

Abstract: This paper analyzes global convergence of the univariate dynamic encoding algorithm for searches (uDEAS) and provides an application result to function optimization. uDEAS is a more advanced optimization method than its predecessor in terms of the number of neighborhood points. This improvement should be validated through mathematical analysis for further research and application. Since uDEAS can be categorized into the generating set search method also established recently, the global convergence property of uDEAS is proved in the context of the direct search method. To show the strong performance of uDEAS, the global minima of four 30 dimensional benchmark functions are attempted to be located by uDEAS and the other direct search methods. The proof of global convergence and the successful optimization result guarantee that uDEAS is a reliable and effective global optimization method.

Keywords: Direct search method, function optimization, generating set search, global convergence, univariate dynamic encoding algorithm for searches (uDEAS).

1. INTRODUCTION

Consider the problem of finding a local minimizer of a real valued function $f(x)$. If f is differentiable and the gradient of f at x, $\nabla f(x)$, can be computed or accurately estimated by finite-differences, a wide selection of derivative-based optimization methods is available. However, many engineering problems have unavailable or untrustworthy $\nabla f(x)$. In addition, other forms of noise occur when the objective function involves limited precision in the observed data, stochastic variation, or unpredictable fluctuations from experiments.

For these problems direct search methods have been developed with an appreciable number of successes since the 1950s. The direct search methods examine trial solutions involving a comparison of each trial solution with the best solution obtained up to that time together with a straightforward search strategy for determining what the next trial solution will be, as defined by Hooke and Jeeves [1]. Thus, direct search methods are easy to implement with a large number of variations. The compass search [2] and the Nelder-Mead simplex algorithm [3] are well-known direct search methods.

From the viewpoint of variable representation, direct search methods are generally classified into the methods with binary encoding and real encoding. Binary encoding was originally employed in the binary-coded genetic algorithm (BCGA) [4], while the real-coded genetic algorithm (RCGA) adopted real encoding, i.e., real values are directly used as chromosome vectors [5,6]. Owing to the fact that BCGA operates with long chromosomes made by the concatenation of binary strings of each variable, RCGA is known to be more suitable in terms of solution precision and convenience for multidimensional and high-precision problems in real space [5]. However, binary digits offer the maximum number of schemata per bit of information of any coding [4], which implies that BCGA is intrinsically guided by similarities at certain string positions. Moreover, the weakness mentioned above for BCGA can be overcome by transforming real variables into a binary matrix rather than a binary string and by gradually elongating each row of the matrix to increase space resolution.

As a combination of the direct search method and BCGA, dynamic encoding algorithm for searches

Manuscript received August 18, 2006; revised January 18, 2008; accepted April 7, 2008. Recommended by Editorial Board member Poo Gyeon Park under the direction of Editor Young-Hoon Joo. This study was supported by research funds from Dong-A University.

Jong-Wook Kim and Taegyu Kim are with the Department of Electronics Engineering, Dong-A University, Hadan-dong, Saha-gu, Busan 604-714, Korea (e-mails: kjwook@dau.ac.kr, taeguya@hotmail.com).

Joon-Young Choi is with the Department of Electronic Engineering, Pusan National University, 30 Jangjeon-dong, Geumjeong-gu, Busan 609-735, Korea (e-mail: jyc@pusan.ac.kr).

Sang Woo Kim is with the Electrical and Computer Engineering Division, Pohang University of Science and Technology, Pohang, Gyoengbuk 790-784, Korea (e-mail: swkim@postech.ac.kr).

(DEAS) has been developed and applied to various fields including parameter identification of an induction motor [7], parameter optimization of state vector machine [8], optimal design of a transformer core [9], design of PID control [10], and the like.

The binary representation specific in DEAS has another merit for revisit check. In the case the objective function is expensive, i.e., function evaluation consumes long computation time, evaluation of formerly visited points should be prevented in advance. There are two revisit check routines in DEAS; redundancy check during local search and history check during global search [11]. These routines are remarkably simple by adopting the bit masking technique and concatenation of binary rows into one string, respectively. Most global optimization methods such as tabu search [12] and GA [4] rarely possess these simple and effective revisit check routines.

The initial version of DEAS, however, has the problem of generating exponential number of trial points with increase of dimension. The name exhaustive DEAS (eDEAS) attributes to this search aspect. As an alternative, univariate DEAS (uDEAS) was developed with reference to the univariate method [13] and found more precise solutions within drastically shorter time than eDEAS. uDEAS successfully performed within reasonable run time the identification of 13 parameters in an induction motor [14] and the estimation of 30 emission factors in a billet heat transfer model [15]. For further research and application with uDEAS, global and local convergence analysis should be carried out. To this end, this paper analyzes global convergence of uDEAS with the methodology rigorously established in the direct search methods.

To validate the applicability of uDEAS to real-world high dimensional problems, four benchmark functions with 30 variables are minimized by three recent direct search methods including uDEAS, and the search results are compared via function evaluation numbers as performance measure.

This paper is organized as follows. Section 2 roughly describes the generating set search method and its methodology of convergence analysis for application to uDEAS. Section 3 provides a brief explanation of uDEAS and a proof of global convergence. Section 4 compares the search performance of uDEAS with two competing direct search methods in function optimization. Section 5 concludes the work with remarks and future work.

2. GENERATING SET SEARCH

This section provides a self-contained description about the generating set search (GSS) which covers most direct search methods with very inclusive mathematical definitions and theorems [16]. GSS includes the generalized pattern search method, which covers as special cases the pattern search algorithms of Hooke and Jeeves [1], the compass search [2], multidirectional search [17], and the like. Since uDEAS can also be interpreted by GSS, the theorems and lemmas in this section will be adopted for convergence analysis in uDEAS.

The unconstrained optimization problem (minimization in this case) is defined as

$$\text{minimize } f(x),$$

where the function $f:\mathbb{R}^n \to \mathbb{R}$ is the *objective function* or the *cost function*, and the *variables* $x \in \mathbb{R}^n$ are those directly manipulated by an optimization method.

In this paper, as in the nonlinear programming literature, *global convergence* is used to mean that one (or some, or all) of the limit points of the iterates from arbitrary starting points is a stationary point x^* of f, i.e., $\nabla f(x^*) = 0$. In contrast, *local convergence* is used to mean convergence when the initial point is close enough to a minimizer.

Fig. 1 describes the algorithm of GSS. The core principle of GSS is the rules for updating the iterate x_k and the step-length control parameter Δ_k as follows:

$$x_{k+1} = \begin{cases} x_k + \Delta_k d_k, & k \in \mathcal{S} \\ x_k, & k \in \mathcal{U}, \end{cases} \tag{1}$$

$$\Delta_{k+1} = \begin{cases} \varphi_k \Delta_k, & k \in \mathcal{S} \\ \theta_k \Delta_k, & k \in \mathcal{U}, \end{cases} \tag{2}$$

where $\mathcal{S}$ and $\mathcal{U}$ denote the subsequences of successful and unsuccessful iterations, respectively, and the expansion factor ϕ_k and the contraction factor θ_k in (2) have the properties of $\phi_k \geq 1$ and $0 < \theta_k \leq \theta_{\max} < 1$. The term *successful* means that there exists a direction vector $d_k \in \mathcal{G}_k$, a set of search directions, which satisfies the decrease condition

$$f(x_k + \Delta_k d_k) < f(x_k) - \rho(\Delta_k). \tag{3}$$

In (3) the nonnegative function $\rho \in [0, +\infty)$ is called the *forcing function* and must satisfy one of the following two requirements. Either

$$\rho(t) \equiv 0 \tag{4}$$

or

$$\rho \text{ is continuous, } \rho(t)/t \to 0 \text{ as } t \to 0, \text{ and } \rho(t_1) \leq \rho(t_2) \text{ for } t_1 < t_2. \tag{5}$$

Initialization:

Let $f:\mathbb{R}^n \to \mathbb{R}$ be given.

Let $x_0 \in \mathbb{R}^n$ be an initial guess.

Let $\Delta_{tol} > 0$ be the step-length convergence tolerance.

Let $\Delta_0 > \Delta_{tol}$ be the initial value of the step-length control parameter.

Let $\theta_{\max} < 1$ be an upper bound on the contraction parameter.

Let the forcing function $\rho:[0,+\infty) \to \mathbb{R}$ be a continuous function such that $\rho(t)$ is decreasing as $t \to 0$ and $\rho(t)/t \to 0$ as $t \to 0$. The choice $\rho \equiv 0$ is acceptable.

Let $\beta_{\max} \geq \beta_{\min} > 0$ be upper and lower bounds, respectively, on the lengths of the vectors in any generating set.

Let $\kappa_{\min} > 0$ be a lower bound on the cosine measure of any generating set.

Algorithm: For each iteration $k = 1,2,\cdots$

Step 1: Let $\mathcal{G}_k = \mathcal{G}_k \cup \mathcal{H}_k$. Here $\mathcal{G}_k$ is a generating set for $\mathbb{R}^n$ satisfying $\beta_{\min} \leq \| d \| < \beta_{\max}$ for all $d \in \mathcal{G}_k$ and $\kappa(\mathcal{G}_k) \geq \kappa_{\min}$, and $\mathcal{H}_k$ is a finite (possibly empty) set of additional search directions such that $\beta_{\min} \leq \| d \|$ for all $d \in \mathcal{H}_k$.

Step 2: If there exists $d_k \in \mathcal{G}_k$ such that $f(x_k + \Delta_k d_k) < f(x_k) - \rho(\Delta_k)$, then do the following:

- Set $x_{k+1} = x_k + \Delta_k d_k$ (change the iterate).
- Set $\Delta_{k+1} = \phi_k \Delta_k$, where $\phi_k \geq 1$ (optionally expand the step-length control parameter).

Step 3: Otherwise, $f(x_k + \Delta_k d) < f(x_k) - \rho(\Delta_k)$ for all $d_k \in \mathcal{G}_k$, so do the following:

- Set $x_{k+1} = x_k$ (no change to the iterate).
- Set $\Delta_{k+1} = \theta_k \Delta_k$, where $0 < \theta_k < \theta_{\max} \leq 1$ (contract the step-length control parameter).
- If $\Delta_{k+1} < \Delta_{tol}$, then **terminate**.

Fig. 1. Algorithm of GSS.

Choosing ρ to be identically zero as in (4) imposes a *simple decrease* condition on the acceptance of the step. Otherwise, choosing ρ as in (5) imposes a *sufficient decrease* condition. In uDEAS, the simple decrease condition is used for acceptance of iterates.

Each iteration of the GSS method described in Fig. 1 requires a set $\mathcal{G}_k$, a *generating set* for $\mathbb{R}^n$, which is also interpreted as a positive spanning set. The definition of the generating set is written as follows.

Definition 1: Let $\mathcal{G} = \{d^{(1)},\cdots,d^{(p)}\}$ be a set of $p \geq n+1$ vectors in $\mathbb{R}^n$. Then the set $\mathcal{G}$ *generates* $\mathbb{R}^n$ if for any vector $v \in \mathbb{R}^n$ there exists $\lambda^{(1)},\cdots,\lambda^{(p)} \geq 0$ such that

$$v = \sum_{i=1}^{p} \lambda^{(i)} d^{(i)}.$$

To span all the arbitrary n-dimensional vectors, the generating set must contain minimally $n+1$ vectors and maximally $2n$ vectors. In a two-dimensional space, for example, the minimal and the maximal generating sets are

$$\mathcal{G}_{\min} = \left\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \begin{bmatrix} -1 \\ 1 \end{bmatrix} \right\},$$

$$\mathcal{G}_{\max} = \left\{ \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} -1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \end{bmatrix} \right\}.$$

These generating sets are illustrated in Fig. 2.

The coordinate directions used in uDEAS include a descent direction as shown below with the generalization to arbitrary generating sets [18].

Lemma 1: The set $\mathcal{G}$ *generates* $\mathbb{R}^n$ if and only if for any vector $v \in \mathbb{R}^n$ such that $v \neq 0$, there

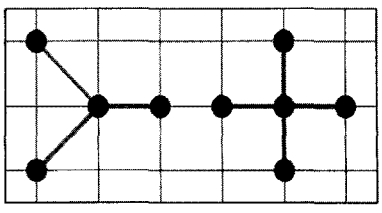


Fig. 2. Illustration of minimal and maximal generating sets for $\mathbb{R}^2$.

exists $d \in \mathcal{G}$ such that

$$v^T d > 0.$$

The significance of Lemma 1 to GSS and uDEAS is that at every iteration k, there must be some $d \in \mathcal{G}_k$ such that

$$-\nabla f(x_k)^T d > 0. \tag{6}$$

This means that $\mathcal{G}_k$ is guaranteed to contain a descent direction whenever $\nabla f(x_k) \neq 0$.

In the case of a general generating set $\mathcal{G}$, the *cosine measure* of $\mathcal{G}$ denoted by $\kappa(\mathcal{G})$ is an important measure. Formally the cosine measure is defined as

$$\kappa(\mathcal{G}) \equiv \min_{v \in \mathbb{R}^n} \max_{d \in \mathcal{G}} \frac{v^T d}{\|v\| \|d\|}, \tag{7}$$

which indicates how far the steepest descent direction $v = -\nabla f(x)$ can be, in the worst case, from the vector in $\mathcal{G}$ making the smallest angle with v.

In terms of descent and cosine measure, (6) means that for any generating set $\mathcal{G}$, there must exist a $d \in \mathcal{G}$ such that

$$\kappa(\mathcal{G}) \| \nabla f(x) \| \| d \| \leq -\nabla f(x)^T d. \tag{8}$$

Then as the cosine measure goes to zero, i.e., the angle between $d \in \mathcal{G}$ and $v = -\nabla f(x)$ approaches $90°$, the quality of the descent direction in $\mathcal{G}_k$ becomes poor and the iterates may converge prematurely to a point that is not stationary.

In order to prove global convergence of the GSS method, two steps are required. The first step is showing that for any subsequence $\mathcal{K}$ of unsuccessful iterations, i.e., $\mathcal{K} \subseteq \mathcal{U}$,

$$\lim_{\substack{k \to +\infty \\ k \in \mathcal{K} \subseteq \mathcal{U}}} \Delta_k = 0 \Rightarrow \lim_{\substack{k \to +\infty \\ k \in \mathcal{K} \subseteq \mathcal{U}}} \| \nabla f(x_k) \| = 0. \tag{9}$$

The second step is only showing there is indeed a subsequence of step-length control parameters going to zero as

$$\lim_{\substack{k \to +\infty \\ k \in \mathcal{K} \subseteq \mathcal{U}}} \Delta_k = 0. \tag{10}$$

The above steps are applied to uDEAS to prove global convergence in the following section.

3. GLOBAL CONVERGENCE OF UDEAS

3.1. uDEAS

uDEAS is a global optimization method that possesses local and global search strategies. As the global search strategy, uDEAS adopts the multistart method where the local search is iterated from random points scattered over the search space. After conducting a finite number of local searches from selected random points, uDEAS attains a global minimum that is the best of the local minima found so far. Since uDEAS uses binary representation, i.e. search space is divided by finite grids; a random initial point is one of the intersection points of the grids. Therefore, initial points should be checked to determine if they were previously evaluated to avoid unnecessary cost evaluation. To this end, the routine named HISTORY CHECK is easily implemented in uDEAS by concatenating all the rows in initial variable matrices into one string which is then transformed to an integer number, storing it in memory, and comparing it with corresponding integers of former matrices.

For example, consider that HISTORY CHECK is carried out for the following binary matrix at the third restart. The resultant binary string concatenated can be decoded into 28 with base 2 like the following:

$$\mathbf{D}^{(3)} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} \Rightarrow [0 \quad 1 \quad 1 \quad 1 \quad 0 \quad 0] \Rightarrow 28.$$

Then the representative number 28 is compared with those of the binary matrices optimized at previous restarts, which have the same row length. Assume that at the first and the second restarts optimal matrices are constructed as the followings:

$$\mathbf{D}^{(1)} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 0 \end{bmatrix} (\Rightarrow 22), \quad \mathbf{D}^{(2)} = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix} (\Rightarrow 28).$$

Since the second matrix has the same integer number 28, which implies its binary matrix is also the same, the matrix $\mathbf{D}^{(3)}$ is decided as a revisited matrix. For computational efficiency, no more search from $\mathbf{D}^{(3)}$ is conducted, and uDEAS restarts from another random point. All these routines constitute HISTORY CHECK.

The global optimization performance of HISTORY CHECK is quite strong for code simplicity. Since this paper deals with the global convergence in performing local search, analysis on global search is deferred.

The local search strategy in uDEAS comprises a bisectional search (BSS) and a unidirectional search (UDS). BSS comes from the property that insertion of 0 (or 1) at the right position of LSB in a binary number leads to a decrease (or increase) of transformed real values from that of the original binary number. The following theorem describes the above property, which is proved in [11].

Theorem 1: Let an n-bit-long binary string, $s_p = a_n a_{n-1} \cdots a_1$, $a_i \in \{0,1\}$, $i = 1, \cdots, n$, be termed a

parent string. If 0 is added to the LSB of s_p, the new string is termed a '*left-hand child string*', s_c^l. On the other hand, if 1 is added, it is termed a '*right-hand child string*', s_c^r. If these strings are decoded into normalized real numbers within 0 and 1 by the following decoding function

$$f_d(b_m b_{m-1}\cdots b_1)=\frac{1}{2^m-1}\sum_{j=1}^{m} b_j 2^{j-1}, \tag{11}$$

where

$$b_i \in \{0,1\}, \quad i=1,\cdots,m,$$

their relations are described as

$$f_d(s_c^l) \le f_d(s_p) \le f_d(s_c^r),$$

where the left- and right-hand side equalities hold only if s_p are an all-zero string and an all-one string, respectively.

BSS is compared to yielding child strings which probe the adjacent search area in a bidirectional manner. The encoding function (11), however, makes the real-valued differences between a parent string and its two child strings vary according to the position of the parent string in the tree [19]. This unbiasedness deteriorates a symmetry condition and complicates the convergence analysis of uDEAS. Thus, the following unbiased binary decoding function proposed in [19] is adopted for analysis

$$f_d(b_m b_{m-1}\cdots b_1)=\frac{1}{2^{m+1}}\left(\sum_{j=1}^{m} b_j 2^j+1\right). \tag{12}$$

Fig. 3 shows the unbiased tree structure established with nodes and branches which represent binary strings and BSS directions, respectively. Although BSS retains a high convergence rate, successive BSS confines a search range within a limited area such that $0\to 01\to 011\to\cdots\to 011\cdots 111$.

UDS is employed to supplement this in-depth search of BSS by hopping between horizontal nodes in a promising direction. This hopping is implemented by increment addition or decrement subtraction to a binary string or a row of a binary matrix. UDS is iterated, maintaining search resolution, until no better point is found, while BSS doubles the resolution at each transition. Owing to the complementary property of UDS, a combination of single BSS and multiple UDS is used in DEAS at every increase of string length.

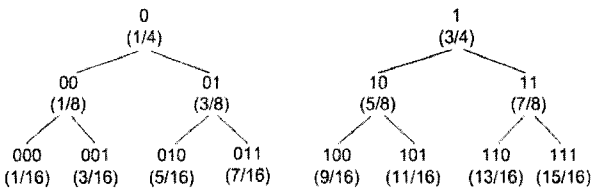


Fig. 3. Unbiased binary tree used in uDEAS.

Fig. 4 shows the working principles of BSS and UDS in uDEAS with the pseudo-code for an initial binary matrix $\mathbf{D}^*_{n\times k}$ where n is variable dimension and k is row length. It should be noted that for multi-dimensional problems uDEAS stacks binary strings into a matrix, while GA concatenates them into a single chromosome string [4]. For the i-th step, the i-th binary row representing the i-th variable is selected from the current best pseudo-matrix $\mathbf{D}^*$ and is modified by BSS and UDS. UDS is guided by the optimal direction of $\mathbf{d}_{opt}(i)$ computed by BSS, until no better solution is found.

After termination of UDS for the i-th row, the $(i+1)$-th row is selected, and the BSS-UDS pair is performed in the same way. It is worth noting that the resultant i-th row whose length is $k+1$ replaces the old best i-th row whose length is k. This implies that the current best variable is described by a pseudo-matrix whose rows that have been treated by the search are lengthened by one. When all the rows have been lengthened equally, the pseudo-matrix becomes a regular matrix $\mathbf{D}^*_{n\times(k+1)}$.

The term *session* in uDEAS means that BSS and UDS have been carried out for all the variables, increasing the row length by one as shown in Fig. 4. The sessions are iterated from initial string length (*initLen*) to final string length (*finLen*) in order to complete the local search from a given initial binary matrix:

Randomly generate an initial binary matrix; $\mathbf{T}_{n\times initLen}$

For $m = initLen : finLen$

$\mathbf{T}_{n\times(m+1)}$=**SESSION** $(\mathbf{T}_{n\times m})$

end for

Fig. 5 illustrates a session for a two-dimensional problem whose minimizer is located at the star point. The number of dashed grids is four in the case of the row length of a starting matrix to be two, while the dotted grids represent that the row length of their child matrices is increased to three. Fig. 5(a) and Figs. 5(b)-(d) show search aspects of BSS and UDS in x_1 direction, respectively. BSS finds out that the right direction is promising and hands it over to UDS. At the third iteration of UDS the cost function of the point depicted with a white circle is no more decreased and thus UDS for x_1 is stopped. Then BSS is carried out along the x_2 direction, followed by UDS as shown in the Figs. 5(e)-(h). As a result, uDEAS computed the cost function 10 times in locating the local minimizer.

$\mathbf{D}^*_{n\times(k+1)} = \mathbf{SESSION}(\mathbf{D}^*_{n\times k})$

Decode the initial matrix: $\mathbf{D}^* \xrightarrow{f_d} \theta^*_{n\times 1}$.

for $i = 1:n$

BSS:

Load the i-th row of current best matrix: $\mathbf{r}_{1\times k} = \mathbf{D}^*(i,1:k)$.

for j=0:1

Load the current best variable vector into a temporary vector: $\theta^{(j)} \leftarrow \theta^*$.

Add a binary bit j to the i-th row vector; $\mathbf{s}^{(j)}_{1\times(k+1)} = [\mathbf{r}\ j]$.

Update the variable vector with the decoded i-th variable; $\theta^{(j)}(i) = f_d(\mathbf{s}^{(j)})$.

Evaluate the cost of a new variable vector; $J^{(j)} = f(\theta^{(j)})$

end

Compare the cost values; $J^* = \min(J^{(0)}, J^{(1)})$

If $\mathbf{J}^* = J^{(0)}$ **then**

$\mathbf{d}_{opt}(i) = 0, \mathbf{s}^* = \mathbf{s}^{(0)}, \theta^* = \theta^{(0)}$.

else

$\mathbf{d}_{opt}(i) = 1, \mathbf{s}^* = \mathbf{s}^{(1)}, \theta^* = \theta^{(1)}$.

end if

UDS:

while a better solution is sought **do**

$\theta' \leftarrow \theta^*$.

if $\mathbf{d}_{opt}(i) = 0$ **then** $\mathbf{s}' = \mathbf{s}^* - 1$.

else $\mathbf{s}' = \mathbf{s}^* + 1$.

end if

Update the variable vector by decoding the i-th variable; $\theta'(i) = f_d(\mathbf{s}')$.

Evaluate the cost of a new variable vector; $J = f(\theta')$

if $\mathrm{J} < \mathrm{J}^*$ **then**

$\mathbf{s}^* \leftarrow \mathbf{s}, J^* \leftarrow J, \theta^* = \theta'$.

else Stop UDS

end if

end while

Save the i-th best row into the current best pseudo-matrix: $\mathbf{D}^*(i,1:k+1) = \mathbf{s}^*$

end for

Fig. 4. Local search algorithm of uDEAS.

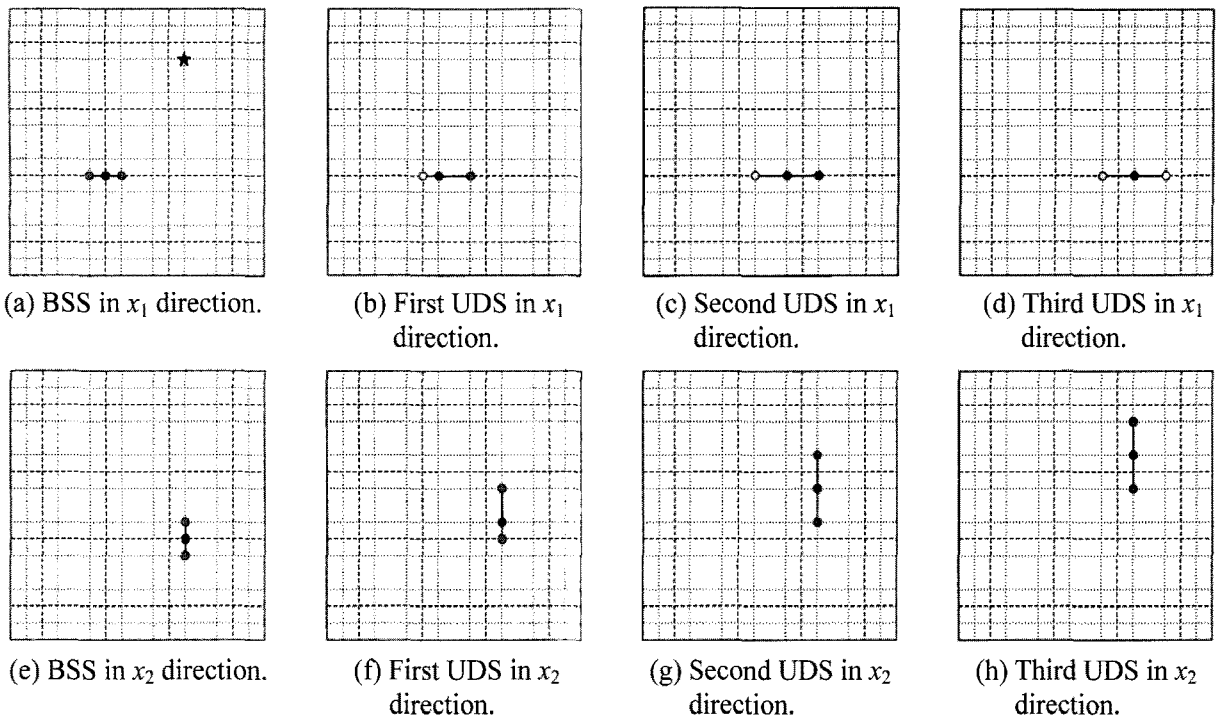
uDEAS is somewhat different from GSS via a rational lattice [20] which stretches $2n$ directions at every update of iterates as shown in Fig. 6. Therefore, GSS will evaluate the cost function 24 times to locate the local minimizer. Comparing Figs. 5 and 6 leads to the fact that GSS inevitably revisits one point at each iteration.

The update rule of uDEAS is described as

$$x_{k+1} = \begin{cases} x_k + \Delta_k d_k, & k \in \mathcal{K}_{BSS} \text{ or } k \in \mathcal{S}_{UDS} \\ x_k, & k \in \mathcal{U}_{UDS}, \end{cases} \tag{13}$$

where $\mathcal{K}_{BSS}$ and $\mathcal{S}_{UDS}$ denote subsequences of iterations at BSS and successful iterations at UDS, respectively, and $\mathcal{U}_{UDS}$ denotes subsequences of unsuccessful iterations at UDS. The direction vector d_k in (13) is one of the $2n$ coordinate directions such that $d_k \in \mathcal{D}_{\oplus} = \{e_1, e_2, \cdots, e_n, -e_1, -e_2, \cdots, -e_n\}$. For instance, all the direction vectors in three dimensions are arranged as

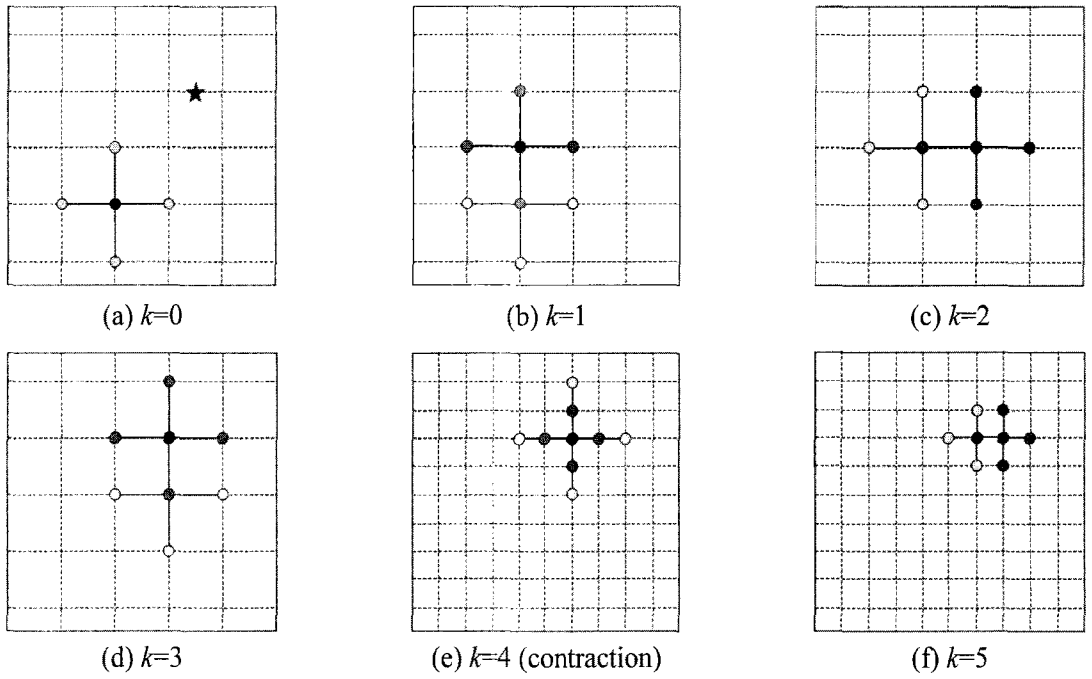
$$d_k \in \left\{ \begin{bmatrix}1\\0\\0\end{bmatrix}, \begin{bmatrix}-1\\0\\0\end{bmatrix}, \begin{bmatrix}0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\-1\\0\end{bmatrix}, \begin{bmatrix}0\\0\\1\end{bmatrix}, \begin{bmatrix}0\\0\\-1\end{bmatrix} \right\}.$$



(a) BSS in x_1 direction. (b) First UDS in x_1 direction. (c) Second UDS in x_1 direction. (d) Third UDS in x_1 direction.

(e) BSS in x_2 direction. (f) First UDS in x_2 direction. (g) Second UDS in x_2 direction. (h) Third UDS in x_2 direction.

Fig. 5. Illustration of uDEAS for a two-dimensional problem on the underlying lattice.



(a) k=0 (b) k=1 (c) k=2

(d) k=3 (e) k=4 (contraction) (f) k=5

Fig. 6. Illustration of GSS for a two-dimensional problem on the underlying lattice. k denotes the iteration number.

The step-length control parameters Δ_k in (13) are directly associated with the row length of the current binary string and the search phase, i.e., BSS or UDS; In the case that BSS is done to an m-bit binary string, the resultant step-length parameters for the current BSS and the subsequent UDS will be

$$\Delta_k = \frac{1}{2}\frac{1}{2^{m+1}}, \quad \Delta_{k+1} = \frac{1}{2^{m+1}},$$

which implies that $\Delta_{k+1} = 2\Delta_k$.

With regard to UDS, there exist two cases; successful and unsuccessful UDS. For the successful UDS with an $(m+1)$-bit binary string, the following relations are obtained

$$\Delta_k = \frac{1}{2^{m+1}}, \quad \Delta_{k+1} = \frac{1}{2^{m+1}}.$$

Thus, $\Delta_{k+1} = \Delta_k$. However, in the case of an

unsuccessful UDS, BSS is subsequently carried out after changing the search direction along the next variable's axis. If the current session is incomplete, i.e., the current variable is not x_n, the string length of the next binary string for BSS changes from $m+1$ to m. Thus the step-length parameters correspond to

$$\Delta_k = \frac{1}{2^{m+1}},\ \Delta_{k+1} = \frac{1}{2}\frac{1}{2^{m+1}},$$

with the relation of $\Delta_{k+1} = \frac{1}{2}\Delta_k$. Finally, if the current session is complete with $(m+1)$-bit rows, the resultant string after BSS becomes $(m+2)$-bit long, which implies that $\Delta_{k+1} = \frac{1}{2}\frac{1}{2^{m+2}}$. Comparing it with $\Delta_k = \frac{1}{2^{m+1}}$ gives the relation of $\Delta_{k+1} = \frac{1}{4}\Delta_k$.

A summary of the above behaviors of the step-length control parameters under various situations establishes the following relations:

$$\Delta_{k+1} = \begin{cases} 2\Delta_k, & k \in \mathcal{K}_{BSS}, \\ \Delta_k, & k \in \mathcal{S}_{UDS} \\ \frac{1}{2}\Delta_k, & k \in \mathcal{U}_{UDS}, \\ \frac{1}{4}\Delta_k, & k \in \overline{\mathcal{U}}_{UDS}. \end{cases} \tag{14}$$

where $\overline{\mathcal{U}}_{UDS}$ denotes the subsequence of unsuccessful UDS for the last variable x_n, which completes each session.

3.2. Global convergence analysis

In order to prove global convergence of uDEAS according to (9), the relation between $\nabla f(x_k)$ and Δ_k should be established. The following theorem shows that a norm of $\nabla f(x_k)$ is bounded above by Δ_k. For simplicity, we assume $\nabla f(x_k)$ is Lipschitz continuous and $\| d \| \le \beta_{\max}$ for all $d \in \mathcal{G}_k$.

Theorem 2: Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable, and suppose $\nabla f(x)$ is Lipschitz continuous with constant M. Then uDEAS produces iterates such that for any $\mathcal{K} \subseteq \mathcal{U}_{UDS} \cup \overline{\mathcal{U}}_{UDS}$, we have

$$\| \nabla f(x_k) \| \le M \Delta_k \beta_{\max}.$$

Proof: Choose $\hat{d}_k \in \mathcal{G}_k \subseteq \mathcal{G}_k$ satisfying (8). Such a $\hat{d}_k$ exists because $\mathcal{G}_k$ generates $\mathbb{R}^n$; so,

$$\kappa(\mathcal{G}_k) \| \nabla f(x_k) \| \| \hat{d}_k \| \le -\nabla f(x_k)^T \hat{d}_k.$$

The cosine measure of the coordinate directions is $\kappa(\mathcal{G}) = \frac{1}{\sqrt{n}}$ [2], and uDEAS searches along only one direction, i.e., $n = 1$. Therefore the above relation becomes

$$\| \nabla f(x_k) \| \| \hat{d}_k \| \le -\nabla f(x_k)^T \hat{d}_k.$$

By the mean value theorem, for some $\alpha_k \in [0,1]$,

$$f(x_k + \Delta_k \hat{d}_k) - f(x_k) = \Delta_k \nabla f(x_k + \alpha_k \Delta_k \hat{d}_k)^T \hat{d}_k.$$

Because k is an unsuccessful iteration for the simple decrease condition,

$$0 \le f(x_k + \Delta_k \hat{d}_k) - f(x_k).$$

Putting the last two relations together, dividing through by Δ_k, and subtracting $\nabla f(x_k)^T \hat{d}_k$ from both sides yields

$$-\nabla f(x_k)^T \hat{d}_k \le (\nabla f(x_k + \alpha_k \Delta_k \hat{d}_k) - \nabla f(x_k))^T \hat{d}_k.$$

From this, the following is attained

$$\| \nabla f(x_k) \| \| \hat{d}_k \| \le (\nabla f(x_k + \alpha_k \Delta_k \hat{d}_k) - \nabla f(x_k))^T \hat{d}_k.$$

Then

$$\begin{aligned} \| \nabla f(x_k) \| &\le \| \nabla f(x_k + \alpha_k \Delta_k \hat{d}_k) - \nabla f(x_k) \| \\ &\le M \Delta_k \| \hat{d}_k \|. \end{aligned}$$

□

The next step is showing (10) for uDEAS. To show that the simple decrease condition (3) forces a subsequence of Δ_k's to 0, the only requirement is that f is bounded below.

Theorem 3: Let f be bounded below. Then uDEAS produces iterations satisfying

$$\liminf_{k \to \infty} \Delta_k = 0.$$

Proof: Suppose not. Then there exists $\Delta_* > 0$ such that $\Delta_k \ge \Delta_*$ for all k. Considering the updating rule for Δ_k, given in (14), the existence of $\Delta_* > 0$ implies that the number of BSS and successful iterations must be infinite.

What is distinctive in uDEAS is that in BSS all the neighborhood points are relatively compared with each other, which implies that in some cases a slightly worse point than the center or incumbent point can be selected. This deterioration may be due to ruggedness rarely present in the landscape of cost function, but it can be easily overcome by the UDS that follows. Therefore, BSS produces both successful and unsuccessful, though it is not often the case, subsequences denoted $\mathcal{S}_{BSS}$ and $\mathcal{U}_{BSS}$, respectively.

As a whole, the relation of cost function at an

updated point can be classified as

$$f(x_{k+1}) < f(x_k),\ k \in \mathcal{K}_{BSS} \cup \mathcal{K}_{UDS},$$
$$f(x_{k+1}) = f(x_k),\ k \in \mathcal{K}_{UDS},$$
$$f(x_{k+1}) > f(x_k),\ k \in \mathcal{K}_{BSS}.$$

The second case is due to the fact that, after an unsuccessful UDS, x_{k+1} is remained at x_k as shown in (13).

It is evident from the first case that if the number of successful iterations is infinite, then $f(x_k) \to -\infty$, which contradicts the assumption that f is bounded below. Hence, the claim.

The objective functions minimized by eDEAS and uDEAS so far are the sum squared values of the errors between measured data and predicted output signals computed by the mathematical model [7] and the absolute deviations between desired values and controlled values [10]. Since the sum squared errors are always nonnegative, the requirement of f to be bounded below is satisfied. Therefore, Theorem 3 is valid, and uDEAS is globally convergent as

$$\lim_{\substack{k \to +\infty \\ k \in \mathcal{K} \subseteq \mathcal{K}_{UDS} \cup \overline{\mathcal{K}}_{UDS}}} \| \nabla f(x_k) \| = 0.$$

4. OPTIMIZATION RESULT

In order to validate the reliability and the performance of uDEAS, function optimization for relatively high-dimensional benchmark functions is carried out in this section. In the latest literature, uDEAS successfully located the global minima of ten test functions whose dimensions vary from 2 to 30 [21]. Specifically in high dimensions, it is reported that uDEAS outperforms eDEAS as expected.

Since uDEAS can be classified as a direct search method, performance comparison with the most recent direct search method named mesh adaptive direct search (MADS) [22] is also significant. Therefore function evaluation numbers required for the three optimization methods, i.e., GSS, MADS, and uDEAS, to locate the known global minima are compared as a performance measure.

Four 30-dimensional benchmark functions are selected in this experiment from the well-known literature of the evolutionary programming [23]:

- Sphere Model

$$f_1(\mathbf{x}) = \sum_{i=1}^{30} x_i^2$$

The global minimum value f_1^* is 0 at the point $\mathbf{x}^* = [0, \cdots, 0]$, and each variable is bounded as $-100 \le x_i \le 100,\ i = 1, \cdots, 30.$

- Schwefel's Problem

$$f_2(\mathbf{x}) = \sum_{i=1}^{30} |x_i| + \prod_{i=1}^{30} |x_i|.$$

The global minimum value f_2^* is 0 at $\mathbf{x}^* = [0, \cdots, 0]$, and $\mathbf{x} \in [-10, 10]^{30}$.

- Ackley's Function

$$f_3(\mathbf{x}) = -20 \exp\left(-0.2 \sqrt{\frac{1}{30} \sum_{i=1}^{30} x_i^2} \right) - \exp\left(\frac{1}{30} \sum_{i=1}^{30} \cos(2\pi x_i) \right) + 20 + e.$$

The global minimum value f_3^* is 0 at the point $\mathbf{x}^* = 0$, and $\mathbf{x} \in [-32, 32]^{30}$.

- Generalized Griewank Function

$$f_4(\mathbf{x}) = \frac{1}{4000} \sum_{i=1}^{30} x_i^2 - \prod_{i=1}^{30} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1.$$

The global minimum value f_4^* is 0 at the point $\mathbf{x}^* = 0$, and the variable bound is $\mathbf{x} \in [-600, 600]^{30}$.

GSS is coded in a standard manner such that in (2) the expansion factor ϕ_k is set at 1 and the contraction factor θ_k is set at 0.5 for the simple decrease condition, i.e., $\rho(t) = 0$ in accepting iterates. In order to maximize search performance, $2n$ (=60) vectors are created as the generating set $\mathcal{G}_k$ at the k-th iteration.

The structural difference of MADS over GSS lies in

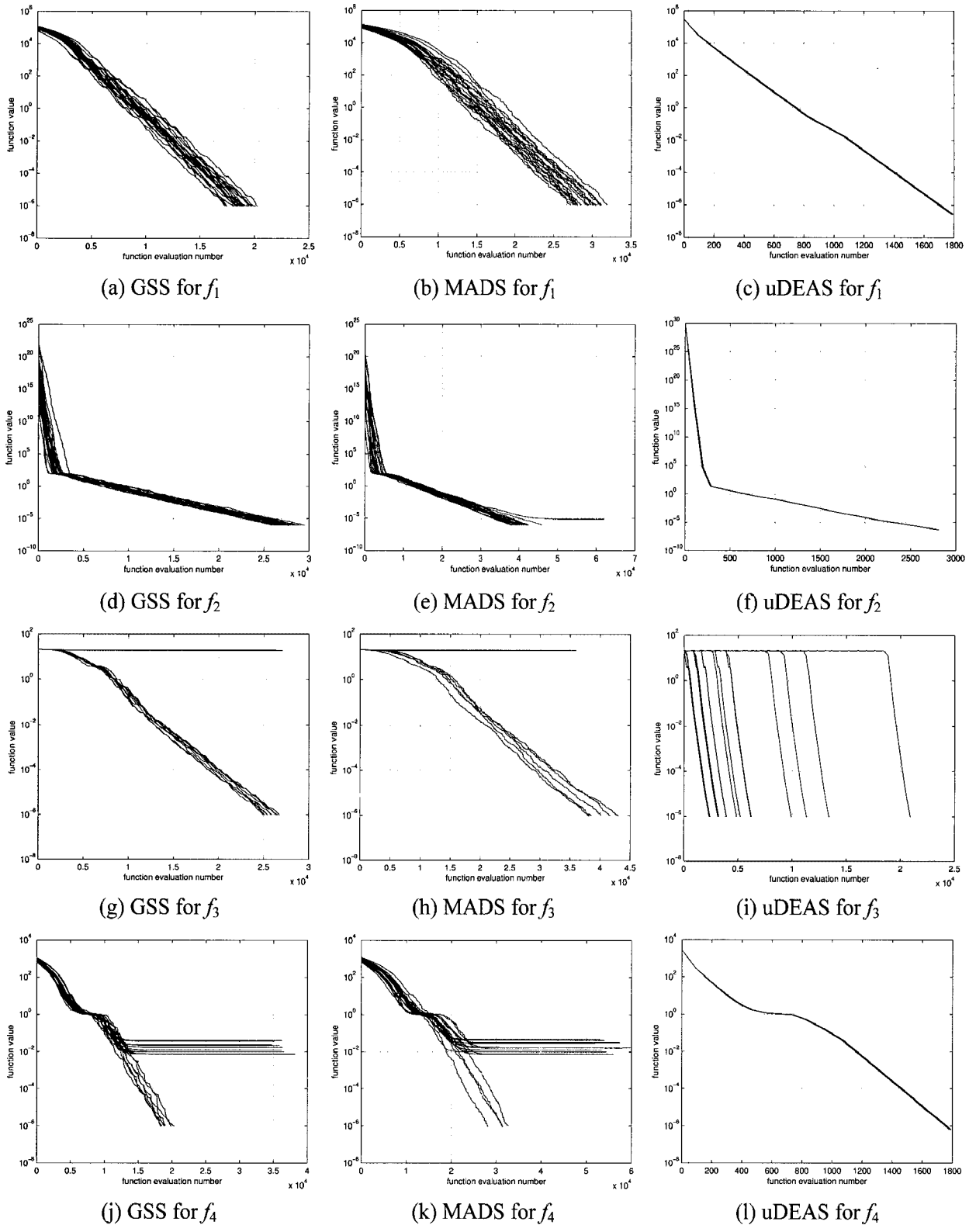
Table 1. Average function evaluation numbers and successful trial numbers attained by GSS, MADS, and uDEAS in locating the global minima of each test function. The values in parentheses are the number of successful local searches during 20 independent restarts. For successful trials, function evaluation numbers are summed and averaged.

Test function	Optimization method		
	GSS	MADS	uDEAS
f_1	18925 (20)	28816 (20)	1787(20)
f_2	27550 (20)	40330 (19)	2806(20)
f_3	25831 (8)	40273 (5)	6641 (20)
f_4	19149 (9)	31581 (6)	1786 (20)

the addition of the poll step that generates a dense set for producing a constrained Clarke stationary point [22]. In MADS, the poll size parameter Δ_k^p is introduced to dictate the magnitude distance from the trial points generated by the poll step to the current incumbent solution. The strategy for updating Δ_k^p must be such that $\Delta_k^m \leq \Delta_k^p$ for all k, satisfying the following condition:

$$\lim_{k \in K} \Delta_k^m = 0 \text{ if and only if } \lim_{k \in K} \Delta_k^p = 0.$$

In this paper, LTMADS [22], a stochastic



(a) GSS for f_1 (b) MADS for f_1 (c) uDEAS for f_1

(d) GSS for f_2 (e) MADS for f_2 (f) uDEAS for f_2

(g) GSS for f_3 (h) MADS for f_3 (i) uDEAS for f_3

(j) GSS for f_4 (k) MADS for f_4 (l) uDEAS for f_4

Fig. 7. Multistart search aspects of GSS, MADS, and uDEAS for the test functions over 20 independent trials.

implementation of the MADS algorithm, is coded and employed for function optimization. By following the guide, $\Delta_0^m = 1, \Delta_0^p = 1$ are chosen as initial values, and the lower triangular matrix B, where each term on the diagonal is either plus or minus $\frac{1}{\sqrt{\Delta_k^m}}$ with the integral lower components randomly selected in the open interval $\left]-\frac{1}{\sqrt{\Delta_k^m}}, \frac{1}{\sqrt{\Delta_k^m}}\right[$, are again permuted by lines and columns. As a completion to a positive basis for this poll step, the maximal positive bases of $2n$ directions are constructed like GSS.

For global optimization, the multistart approach is commonly applied to the three methods by selecting an initial point randomly in a search space, iterating a local search from it to the prescribed limit, and restarting this routine from another random point. After termination of the multiple restarts, the best local minimum found so far is regarded as a global minimum. Termination criterion is whether the solution accuracy, i.e., difference between the current function value and the global minimum value f^*, is below 10^{-6} or whether the number of restarts is over 20.

Table 1 summarizes the function optimization result attained after 20 independent restarts per experiment. Fig. 7 shows the overall global optimization aspects with respect to function evaluation numbers. GSS searches reliably for f_1 and f_2, while the success ratios are below 50 percent for f_3 and f_4. MADS yields a slightly worse result in terms of both the function evaluation numbers and the success ratios. The results of uDEAS are, however, about nine times faster than GSS with excellent reliability.

Although this type of comparison may not be perfectly fair due to the possible lack of optimality in parameter setting for each method, it is justifiable to conclude that uDEAS is sufficiently feasible for the global optimization of an unconstrained multimodal problem with short running time.

5. CONCLUSION

This paper provides a rigorous proof of global convergence for uDEAS and performance comparison with other direct search methods, GSS and MADS, in the area of function optimization. Despite the simple working principle of uDEAS for global optimization, the success in function optimization up to 30 dimensions provided in this paper is quite promising.

Based on the guarantee of global convergence and the fast and reliable result of function optimization, uDEAS will be more widely used in the optimal trajectory generation of robots, real-time identification, real-time control, and the like. As a future work, local convergence and convergence rate will be analyzed for uDEAS and the other DEAS series.

REFERENCES

[1] R. Hooke and T. A. Jeeves, "Direct search solution of numerical and statistical problems," *Journal of the ACM*, vol. 8, no. 2, pp. 212-229, 1961.

[2] V. Torczon, "On the convergence of pattern search algorithms," *SIAM Journal on Optimization*, vol. 7, no. 1, pp. 1-25, 1997.

[3] J. A. Nelder and R. Mead, "A simplex method for function minimization," *Computer Journal*, vol. 7, pp. 308-313, 1965.

[4] D. E. Goldberg, *Genetic Algorithm in Search, Optimization and Machine Learning*, Addison Wesley, 1989.

[5] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer, 1996.

[6] J.-W. Kim and S. W. Kim, "New encoding/ converting methods of binary GA/real-coded GA," *IEICE Trans. on Fundamentals*, vol. E88-A, no. 6, pp. 1554-1564, June 2005.

[7] J.-W. Kim and S. W. Kim, "Parameter identification of induction motors using dynamic encoding algorithm for searches (DEAS)," *IEEE Trans. on Energy Conversion*, vol. 20, no. 1, pp. 16-24, March 2005.

[8] Y. S. Park, Y. Lee, J.-W. Kim, and S. W. Kim, "Parameter optimization for SVM using dynamic encoding algorithm," *Proc. of International Conference on Control, Automation, and Systems*, KINTEX, Korea, pp. 2542-2547, June 2005.

[9] T. Kim and J.-W. Kim, "Optimal design of a transformer core using DEAS," *Trans. KIEE*, vol. 56, no. 6, pp. 1055-1063, June 2007.

[10] J.-W. Kim and S. W. Kim, "PID control design with exhaustive dynamic encoding algorithm for searches (eDEAS)," *International Journal of Control, Automation, and Systems*, vol. 5, no. 6, pp. 691-700, Dec. 2007.

[11] J.-W. Kim and S. W. Kim, "Numerical method for global optimization: Dynamic encoding algorithm for searches," *IEE Proc.-Control Theory Appl.*, vol. 151, no. 5, pp. 661-668, Sept. 2004.

[12] F. Glover, "Tabu search methods in artificial intelligence and operations research," *ORSA Artificial Intelligence*, vol. 1, no. 2, p. 6, 1987.

[13] S. S. Rao, *Engineering Optimization*, John Wiley & Sons Inc., 1996.

[14] J.-W. Kim, N. G. Kim, S.-C. Choi, and S. W. Kim, "On-load parameter identification of an

induction motor using univariate dynamic encoding algorithm for searches," *Proc. of International Conference on Control, Automation and Systems*, Bangkok, Thailand, pp. 852-856, August, 2004.
[15] Y. J. Jang and S. W. Kim, "Estimation of a billet temperature during reheating furnace operation," *International Journal of Control, Automation, and Systems*, vol. 5, no. 1, pp. 43-50, Feb. 2007.
[16] T. G. Kolda, R. M. Lewis, and V. Torczon, "Optimization by direct search: New perspectives on some classical and modern methods," *SIAM Review*, vol. 45, no. 3, pp. 385-482, 2003.
[17] V. Torczon, "On the convergence of the multidirectional search algorithm," *SIAM Journal on Optimization*, vol. 1, no. 1, pp. 123-145, 1991.
[18] C. Davis, "Theory of positive linear dependence," *Amer. J. Math.*, vol. 76, pp. 733-746, 1954.
[19] N. G. Kim, J.-W. Kim, and S. W. Kim, "A study for global optimization using dynamic encoding algorithm for searches," *Proc. of International Conference on Control, Automation and Systems*, Bangkok, Thailand, pp. 857-862, Aug. 2004.
[20] G. Berman, "Lattice approximations to the minimum of functions of several variables," *Journal of the ACM*, vol. 16, pp. 286-294, 1969.
[21] J.-W. Kim and S. W. Kim, "A fast computational optimization method: Univariate dynamic encoding algorithm for searches (uDEAS)," *IEICE Trans. on Fundamentals*, vol. E90-A, no. 8, pp. 1679-1689, Aug. 2007.
[22] C. Audet and J. E. Dennis JR, "Mesh adaptive direct search algorithms for constrained optimization," *SIAM Journal on Optimization*, vol. 17, no. 1, pp. 188-217, 2006.
[23] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. on Evolutionary Computation*, vol. 3, no. 2, pp. 82-102, July, 1999.

Jong Wook Kim received the B.S., M.S., and Ph.D. degrees from the Electrical and Computer Engineering Division at Pohang University of Science and Technology (POSTECH), Pohang, Korea, in 1998, 2000, and 2004, respectively. Currently, he is an Assistant Professor in the Department of Electronics Engineering at Dong-A University, Busan, Korea. His current research interests are numerical optimization methods, robot control, intelligent control, diagnosis of electrical systems, and system identification.

Taegyu Kim received the B.S. degree from the Department of Electronics Engineering at Dong-A University, Busan, Korea, in 2007, where he is currently undertaking a Master's course. His research interests include embedded systems and robot control.

Joon-Young Choi received the B.S., M.S., and Ph.D. degrees in Electronic and Electrical Engineering from Pohang University of Science and Technology (POSTECH), Pohang, Korea in 1994, 1996, and 2002, respectively. From 2002 to 2004, he worked as a Senior Engineer at the Electronics and Telecommunication Research Institute (ETRI), Daejeon, Korea. From 2004 to 2005, he worked as a Visiting Associate in the Departments of Computer Science and Electrical Engineering at the California Institute of Technology (CALTECH), Pasadena, CA. Since 2005, he has been an Assistant Professor in the school of Electrical Engineering at Pusan National University, Busan, Korea. His research interests include nonlinear control, internet congestion control, embedded systems, and automation.

Sang Woo Kim received the B.S., M.S., and Ph.D. degrees from the Department of Control and Instrumentation Engineering, Seoul National University, in 1983, 1985, and 1990, respectively. Currently, he is an Associate Professor in the Department of Electronics and Electrical Engineering at POSTECH, Pohang, Korea. He joined POSTECH in 1992 as an Assistant Professor and was a Visiting Fellow in the Department of Systems Engineering, Australian National University, Canberra, Australia, in 1993. His current research interests are in optimal control, optimization algorithms, intelligent control, wireless communication, and process automation.