Journal of Global Optimization **8:** 393–411, 1996.

An Improved Univariate Global Optimization Algorithm with Improved Linear Lower Bounding Functions*

XIAOJUN WANG[1] and TSU-SHUAN CHANG[2]
[1] *Applied Mathematics Group, Department of Mathematics, University of California, Davis, CA 95616, U.S.A.*
[2] *Department of Electrical and Computer Engineering, University of California, Davis, CA 95616, U.S.A.*

(Received: 7 April 1994; accepted: 26 September 1995)

Abstract. Recently linear lower bounding functions (LLBF's) were proposed and used to find ϵ-global minima. Basically an LLBF over an interval is a linear function which lies below a given function over the interval and matches the function value at one end point. By comparing it with the best function value found, it can be used to eliminate subregions which do not contain ϵ-global minima. To develop a more efficient LLBF algorithm, two important issues need to be addressed: how to construct a better LLBF and how to use it efficiently. In this paper, an improved LLBF for factorable functions over n-dimensional boxes is derived, in the sense that the new LLBF is always better than those in [3] for continuously differentiable functions. Exploration of the properties of the LLBF enables us to develop a new LLBF-based univariate global optimization algorithm, which is again better than those in [3]. Numerical results on some standard test functions indicate the high potential of our algorithm.

Key words: univariate global optimization, linear lower bounding functions, LLBF methods.

1. Introduction

In recent years, a variety of methods for solving global optimization problems have been proposed [5, 6, 11]. One class of deterministic approaches, which is called covering methods, emerged from a natural strategy to find a global minimum for sure.

Recently the concept of linear lower bounding functions (LLBF's) was introduced to develop global optimization algorithms [1, 2, 3].** For univariate problems an LLBF $l(x)$ of a given object function $f(x)$ on an interval $[a, b]$ is a linear underestimating function of $f(x)$ whose function value at one end point matches that of the given function. Not only can LLBF's for a large class of functions such as factorable functions be constructed, but also they can be easily used to eliminate

* This work was supported in part by VLSI Technology Inc. and Tyecin Systems Inc. through the University of California MICRO program with grant number 92-024.

** An LLBF was called a linear lower bound in [1, 2, 3] Since an LLBF is actually a function, its new name is more appropriate.

subregions not containing ϵ-global minima. These advantages make the LLBF method a promising approach to global optimization problems.

As mentioned in [3], to guarantee locating a global minimizer or an ϵ-global one, it is impossible to go through every individual point in practice. The question then is reduced to how to efficiently cover the feasible region to make such an evaluation computationally feasible. In this paper, we will develop an improved global optimization algorithm for univariate problems using LLBF's.

In Section 2 a better LLBF is derived. In section 3 the basic algorithm is developed. In section 4, the improved algorithm is presented by using the basic algorithm for the global phase, and integrating it with a newly developed local minimization algorithm in the local phase. The local minimization algorithm locates a local minimum with its associated interval, so that the local minimum found is also a global minimum in the interval found. Section 5 presents the numerical results. Section 6 is a short discussion.

2. An Improved Linear Lower Bounding Function

Although this paper deals with univariate global optimization and only the univariate LLBF is needed, we do provide the procedure for constructing multivariate LLBF's, since there is no conceptual difference.

Let $f(x) : R^n \to R$ be a continuous function, and D an n-dimensional box in R^n, i.e.,

$$D = \{x \in R^n \mid a(i) \leq x(i) \leq b(i),\ i = 1, \cdots, n\}. \tag{2.1}$$

Let x_0 be a vertex of D. As defined in [1, 2], a linear function

$$l(x) = m^T x + r \tag{2.2}$$

is a linear lower bounding function (LLBF) of $f(x)$ over the box D with the matching point x_0 if

$$f(x) \geq l(x), \quad \forall x \in D, \tag{2.3}$$

$$f(x_0) = l(x_0), \tag{2.4}$$

$$\|m\| \leq G, \tag{2.5}$$

where G is a finite number. A linear upper bounding function (LUBF) is similarly defined by replacing $\geq$ in (2.3) by $\leq$.

We will use linear bounding function (LBF) to refer to either LLBF or LUBF or to both. If a bounding function is piecewise linear (convex), it is called a piecewise linear (convex) bounding function. For univariate problems, the box is reduced to an interval $[a, b]$. An LBF is called a right(left) LBF if the right(left) end point is matched.

An LBF over other compact regions such as a simplex can be similarly defined. In fact, the LBF's over simplices have been developed in [2] for those functions which can be recursively obtained by using the difference of two convex functions and the composition of monotonic convex functions. In [3], LBF's over boxes were developed for factorable functions, since many functions are factorable [7]. In this section we will develop an improved LBF for factorable functions over boxes, in the sense that it is as good as that in [3] at least for continuously differentiable functions.

A *factorable* function is a function of n variables which is generated by first composing (adding or multiplying) functions of a single variable, transforming those functions, composing those, transforming, ... a finite number of times [8]. The general factorable function $f_N(x)$ can be expressed as

$$f_N(x) \equiv \sum_{p=1}^{N-1} T_{N,p}(f_p(x)) + \sum_{p=1}^{N-1} \sum_{q=1}^{p} U_{N,p,q}(f_q(x)) \cdot V_{N,q,p}(f_p(x)), \tag{2.6}$$

where

$$f_j(x) \equiv x_j, \quad j = 1, \cdots, n, \tag{2.7a}$$

$$f_j(x) \equiv \sum_{p=1}^{j-1} T_{j,p}(f_p(x)) + \sum_{p=1}^{j-1} U_{j,p,q}(f_q(x)) \cdot V_{j,q,p}(f_p(x)), \tag{2.7b}$$

$$j = n+1, \cdots, N-1, \tag{2.7c}$$

and the T's, U's and V's are scalar functions of one variable.

To obtain LBF's for factorable functions, we need to construct LBF's for the minimum or maximum, or sum of several functions, the composite functions, and the product functions as in [3]. While the LBF for the minimum or the maximum of several functions has been developed in [3], we will present a new method to obtain LBF's for composite and product functions. For completeness, the result for min or max functions from [3] are presented in Section 2.1. In Sections 2.2 and 2.3, improved LBF's for composite and product functions have been developed.

2.1. LBF'S FOR MIN AND MAX FUNCTIONS

As mentioned, the results in [3] are presented in this subsection for completeness. Let $l_{f_i}(x)$ denote the LLBF of $f_i(x)$ over the box D with (the matching vertex) x_0, i.e.,

$$l_{f_i}(x) \leq f_i(x), \quad \forall x \in D, \tag{2.8}$$

$$l_{f_i}(x_0) = f_i(x_0). \tag{2.9}$$

Define

$$l_l(x) = \min\{\, l_{f_i}(x),\ i = 1, \cdots, p \,\}. \tag{2.10}$$

Denote the n vertices of D adjacent to x_0 by $x_1, x_2, \cdots, x_n$. Let $l_f(x)$ be the hyperplane passing through the $(n+1)$ points below,

$$(x_j, l_l(x_j)), \quad j = 0, 1, \cdots, n. \tag{2.11}$$

We then have Lemma 2.1.

LEMMA 2.1. *The linear function $l_f(x)$ is an LLBF of $f(x) = \min\{f_i(x), i = 1, \ldots, p\}$.*

Sketch of the proof. First, from (2.9), (2.10), and (2.11), we have

$$l_f(x_0) = l_l(x_0) = \min\{l_{f_i}(x_0)\} = \min\{f_i(x_0)\} = f(x_0). \tag{2.12}$$

Second, from (2.10) and (2.11)

$$l_f(x_j) \le l_{f_i}(x_j), \quad j = 0, 1, \cdots, n; \; i = 1, \cdots, p. \tag{2.13}$$

It is not difficult to see $l_f(x) \le l_{f_i}(x)$ over the box, since a box is a convex set, and a linear function is convex and is uniquely determined by $n+1$ points.

LEMMA 2.2. *Let $I = \{ i \mid f_i(x_0) = f(x_0) \}$. The function $l_g(x)$ is defined by*

$$l_g(x) = \sum_{i \in I} \lambda_i l_{f_i}(x), \tag{2.14}$$

where $\sum_{i \in I} \lambda_i = 1, \lambda_i \ge 0$, is an LLBF for the function

$$g(x) = \max\{f_i(x), i = 1, \cdots, m\}. \tag{2.15}$$

The LUBF's $L_f(x)$ and $L_g(x)$ can be obtained from the LUBF's L_{f_i} of $f_i(x)$ similarly.

2.2. LBF'S FOR COMPOSITE FUNCTIONS

Given the function $T(\cdot) : R \to R$ and $t(\cdot) : R^n \to R$, our goal is to find LBF's for the composite function $T(t(x))$ over a given box D with the matching vertex x_0. Let $l_t(x)$ and $L_t(x)$ be the LLBF and LUBF of $t(x)$, respectively. Assume that $t(x)$ is also bounded by some given numbers a_t and b_t, i.e.,

$$a_t \le t(x) \le b_t, \quad \forall x \in D. \tag{2.16}$$

Also assume that two-piece LBF's for $T(\cdot)$ are given; i.e., we have

$$l_T(t) = \begin{cases} l_1(t), & \text{if } t \in [a_t, t(x_0)], \\ l_2(t), & \text{if } t \in [t(x_0), b_t], \end{cases} \tag{2.17}$$

$$L_T(t) = \begin{cases} L_1(t), & \text{if } t \in [a_t, t(x_0)], \\ L_2(t), & \text{if } t \in [t(x_0), b_t], \end{cases} \tag{2.18}$$

$$l_T(t(x_0)) = l_1(t(x_0)) = l_2(t(x_0)) = T(t(x_0)), \tag{2.19}$$

$$L_T(t(x_0)) = L_1(t(x_0)) = L_2(t(x_0)) = T(t(x_0)). \tag{2.20}$$

LEMMA 2.3. *Let*

$$l(x) = \min\{l_1(lL_1(x)),\ l_2(lL_2(x))\}, \tag{2.21}$$

$$L(x) = \max\{L_1(Ll_1(x)),\ L_2(Ll_2(x))\}, \tag{2.22}$$

where

$$lL_i(x) \triangleq \begin{cases} l_t(x), & \text{if } l_i \text{ is nondecreasing,} \\ L_t(x), & \text{if } l_i \text{ is nonincreasing,} \end{cases} \tag{2.23}$$

$$Ll_i(x) \triangleq \begin{cases} L_t(x), & \text{if } L_i \text{ is nondecreasing,} \\ l_t(x), & \text{if } L_i \text{ is nonincreasing,} \end{cases} \tag{2.24}$$

$$i = 1, 2.$$

Then $l(x)$ and $L(x)$ are the piecewise LLBF and LUBF of $T(t(x))$ over the box D with the matching corner x_0.

Proof. For the LLBF $l(x)$ it is enough to consider one case; say, for example, that l_1 is nondecreasing and l_2 is nonincreasing. Then

$$T(t(x)) \geq l_T(t(x)) = \begin{cases} l_1(t(x)) \geq l_1(l_t(x)), & \text{if } t(x) \in [a_t,\ t(x_0)] \\ l_2(t(x)) \geq l_2(L_t(x)), & \text{if } t(x) \in [t(x_0),\ b_t], \end{cases} \tag{2.25}$$

i.e.,

$$T(t(x)) \geq \min\{l_1(l_t(x)),\ l_2(L_t(x))\}. \tag{2.26}$$

Considering other cases similarly we obtain $l(x)$ given by (2.22) and (2.24).

The proof for $L(x)$ is similar.

Note that one-piece LBF's can be constructed from the above piecewise LBF's. Note also that the LBF's thus obtained are at least as good as those obtained from [3] for continuously differentiable functions. This can be proved by exhausting all the possible cases.

2.3. LBF'S FOR PRODUCT FUNCTIONS

To find linear bounds for $U(u(x)) \cdot V(v(x))$, we can first obtain the LBF's and estimations of min and max for $U(u(x))$ and $V(v(x))$ by using the technique discussed. Denote the LBF's for $U(u(x))$ and $V(v(x))$ by $l_u(x)$, $L_u(x)$, $l_v(x)$, $L_v(x)$ and the estimations of min and max for $U(u(x))$ and $V(v(x))$ by k_u, K_u, k_v and K_v, respectively,

$$l_u(x) \leq U \leq L_u(x), \tag{2.27}$$

$$l_v(x) \le V \le L_v(x), \quad x \in D. \tag{2.28}$$

$$k_u \le U \le K_u, \tag{2.29}$$

$$k_v \le V \le K_v, \quad x \in D. \tag{2.30}$$

Thus,

$$0 \le (U - k_u)(V - k_v) \le (L_u(x) - k_u)(L_v(x) - k_v), \tag{2.31}$$

$$0 \le (K_u - U)(K_v - V) \le (K_u - l_u(x))(K_v - l_v(x)), \tag{2.32}$$

$$0 \le (U - k_u)(K_v - V) \le (L_u(x) - k_u)(K_v - l_v(x)), \tag{2.33}$$

$$0 \le (K_u - U)(V - k_v) \le (K_u - l_u(x))(L_v(x) - k_v), \tag{2.34}$$

and we have

$$\begin{aligned} &k_vU + k_uV - k_uk_v \le UV \le \\ &L_u(x)L_v(x) + k_u(V - L_v(x)) + k_v(U - L_u(x)) \triangleq H_1(x), \end{aligned} \tag{2.35}$$

$$\begin{aligned} &K_vU + K_uV - K_uK_v \le UV \le \\ &l_u(x)l_v(x) + K_u(V - l_v(x)) + K_v(U - l_u(x)) \triangleq H_2(x), \end{aligned} \tag{2.36}$$

$$\begin{aligned} &h_1(x) \triangleq L_u(x)l_v(x) + k_u(V - l_v(x)) + K_v(U - L_u(x)) \le UV \le \\ &k_uV + K_vU - k_uK_v, \end{aligned} \tag{2.37}$$

$$\begin{aligned} &h_2(x) \triangleq l_u(x)L_v(x) + K_u(V - L_v(x)) + k_v(U - l_u(x)) \le UV \le \\ &K_uV + k_vU - K_uk_v. \end{aligned} \tag{2.38}$$

To match the corner x_0, we only consider $h_1(x)$, $h_2(x)$, $H_1(x)$ and $H_2(x)$. Also, denote

$$h_3(x) \triangleq \min\{l_u(x)l_v(x), l_u(x)L_v(x), L_u(x)l_v(x), L_u(x)L_v(x)\}, \tag{2.39}$$

$$H_3(x) \triangleq \max\{l_u(x)l_v(x), l_u(x)L_v(x), L_u(x)l_v(x), L_u(x)L_v(x)\}. \tag{2.40}$$

We have from (2.27), (2.28) and interval analysis

$$h_3(x) \le UV \le H_3(x). \tag{2.41}$$

Let $l(x)$ be the LLBF of $\max\{h_1(x), h_2(x), h_3(x)\}$ and $L(x)$ the LUBF of $\min\{H_1(x), H_2(x), H_3(x)\}$. The functions $l(x)$ and $L(x)$ are then the LLBF and LUBF for $U(u(x)) \cdot V(v(x))$ with the matching point x_0, respectively. The

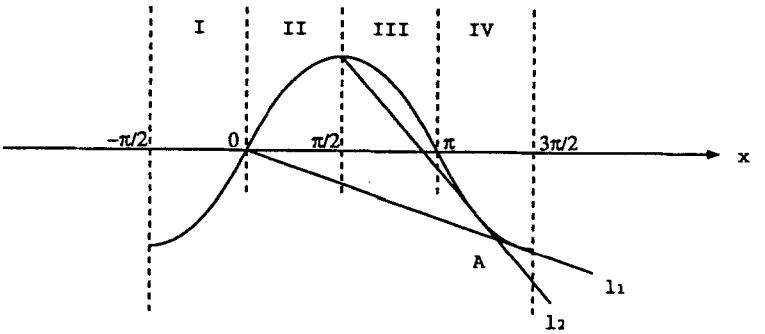

Figure 1. An LLBF for sine function: case 1.

LLBF (or LUBF) for a product of two linear functions, such as $l_u \cdot l_v$, only involves quadratic terms over a box with two variables, and they can be found rather easily by matching a two-dimensional point which comes from components of x_0. Note that the LLBF(LUBF) of any one of the h_i's (H_i's) is an LLBF(LUBF) of UV. Using all of them will give us better LBF's. Also note that the LBF's thus obtained are at least as good as those derived in [3] for any function by direct comparison.

2.4. Construction of LBF's for Factorable Functions

An LBF library can be built to construct LBF's for factorable functions. In the library, the function form can be chosen from some commonly used univariate generic functions, the minimum or maximum of several functions, and function form in summation, composition, or multiplication. For a given function form f with a specified interval, the library provides its LLBF(l_f), LUBF(L_f), and lower (upper) bound α_f (β_f) of the function value over the considered interval. α_f and β_f can be obtained either by the LBF's generated or by interval arithmetic methods [9]. For a univariate function, α_f (β_f) is not less (greater) than the minimum (maximum) of the LLBF (LUBF) over the given interval.

A final remark on the construction of LBF's is in order here. The LBF construction for generic functions is done by brute-force methods to exhaust all the possible cases as in [3]. Let us use the sine function as an example, since the construction of its left LLBF over $[a, b]$ is typical. Let us divide naturally its basic period into four subintervals as in Figure 1.

If the two endpoints a and b are in different periods, its left LLBF only depends on the end point a. For example, if $a \in II$, then the perfect left LLBF is the tangent line passing through $(a, \sin a)$ along the rightward direction. However, if we have to find such a tangent line every time, it is computationally a burden. Instead, we use an imperfect left LLBF which is easy to compute.

Let l_1 and l_2 denote the tangent lines in Figure 1; their intersection point is A, which can be calculated in advance and stored. Then the straight line connecting $(a, \sin a)$ and A is a left LLBF.

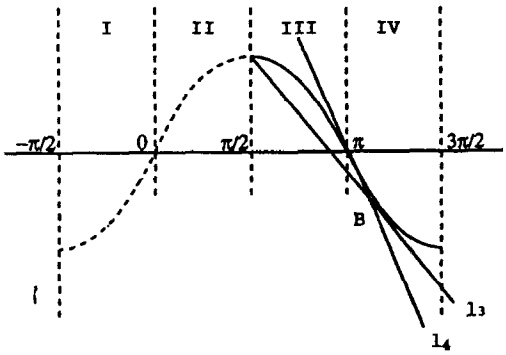

Figure 2. An LLBF for sine function: case 2.

If the interval lies within a single period, the situation is more complicated. Nevertheless, the fundamental idea of constructing a left LLBF is about the same. Let m denote the slope of the left LLBF and consider $a \in III$; the scenario is shown in Figure 2.

Let B be intersection point of the two tangent lines l_3 and l_4; then

$$m = \begin{cases} \dfrac{\sin a - \sin b}{a - b}, & \text{if } b \in III \text{ or } \{b \in IV \text{ and } \dfrac{\sin a - \sin b}{a - b} > \cos b\}; \\ \text{the slope of the line connecting } (a, \sin a) \text{ and } B, & \text{otherwise.} \end{cases} \tag{2.42}$$

All the other situations can be done in a similar way as the two representative cases mentioned.

3. A Univariate Global Optimization Algorithm

The basic way to remove a subregion not containing an ϵ-global solution by an LLBF can be explained as follows. Let $\bar{f}$ be the best (the lowest) function value we have so far, and $[x_0, x_1] \subset [a, b]$ is a subregion. Suppose a left LLBF $l_l(x) = m_l(x - x_0) + f(x_0)$ is available and α represents a lower bound of f in $[x_0, x_1]$, i.e., $\alpha \triangleq est \min_{[x_0, x_1]} f$. Obviously, if $m_l \geq 0$ or $\alpha \geq \bar{f} - \epsilon$, then $f(x) \geq \bar{f} - \epsilon$ for all $x \in [x_0, x_1]$. In this case, the whole region $[x_0, x_1]$ does not contain ϵ-global minima and can be thrown away. Otherwise let x_0' be the solution of $l_l(x) = \bar{f} - \epsilon$. We then have $x_0' \in [x_0, x_1]$ due to the construction of α as mentioned in Section 2.4. Since $f(x) \geq \bar{f} - \epsilon$ for $x \in [x_0, x_0']$, we can now discard $[x_0, x_0']$ and store $[x_0', x_1]$ for further consideration.

To efficiently use the left LLBF over $[x_0, x_1]$, it should be stored with the remaining interval $[x_0', x_1]$, since it is reusable. For example, assume that the left LLBF $l_l(x)$ was generated on the region $[x_0, x_1]$ with $\alpha = est \min_{[x_0, x_1]} f$, and the subregion $[x_0, x_0']$ was previously discarded for the best function value $\bar{f}_{old}$ found at the time when using $l_l(x)$. Suppose a better $\bar{f}_{new}$ ($< \bar{f}_{old}$) was obtained after the

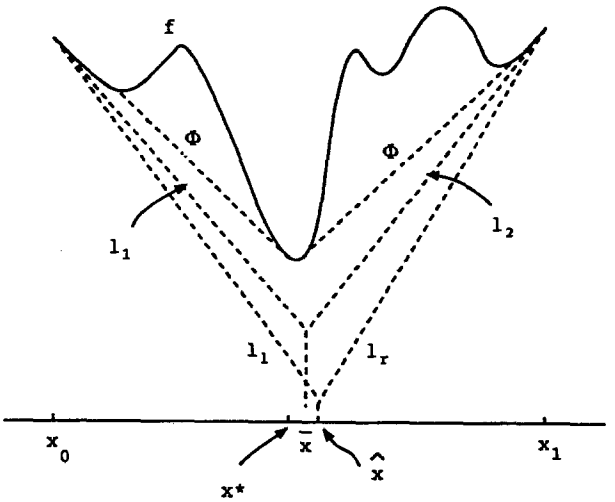

Figure 3. Geometry of a nonconvex minimization problem.

region $[x_0', x_1]$ was stored. When the region is finally selected for processing, we can at least use the same $l_l(x)$ to remove a subregion $[x_0', \min\{x_0'', x_1\}]$, where x_0'' is the solution of $l_l(x) = \bar{f}_{new}$.

Note that the lower $\bar{f}$ is, the bigger the subregion $[x_0, x_0']$ that might be discarded. Thus in the algorithm development, we should try to reduce $\bar{f}$ as fast as possible. Intuitively, LBF's may improve our chance of getting a better point, which is a point with lower function value than the current $\bar{f}$. To see this, let us examine the geometry of a univariate nonconvex minimization problem with a unique global minimum point x^* as shown in Figure 3. Let ϕ denote the convex envelope of the given function f. Then

$$\arg \min_{[x_0,x_1]} \phi = \arg \min_{[x_0,x_1]} f \triangleq x^*, \tag{3.1}$$

according to the result of convex analysis [10].

Consider the intersection point $\bar{x}$ of the two tangent lines of ϕ, l_1 and l_2, from both end points. Apparently, $\bar{x}$ is closer to x^* than any end points. If f has a reasonable large basin for x^* and $\bar{x}$ is close enough to x^*, then very likely $\bar{x}$ will fall into that basin. Of course, for a general nonconvex minimization problem, l_1 and l_2 are not available, since computing ϕ is at least as difficult as solving the original problem. However, the left and right LLBF's (l_l and l_r) can be considered as the approximations of l_1 and l_2. Even when they are not good approximations, their intersection point may still not be too far away from $\bar{x}$, if they are generated in the same way and have fairly consistent approximation errors. In other words, we may have a good chance to locate a better point when using both left and right LLBF's.

For a problem with multiple global minimum points, we might have different cases. For the case depicted in Figure 4, $\hat{x}$ could be regarded as a good point to

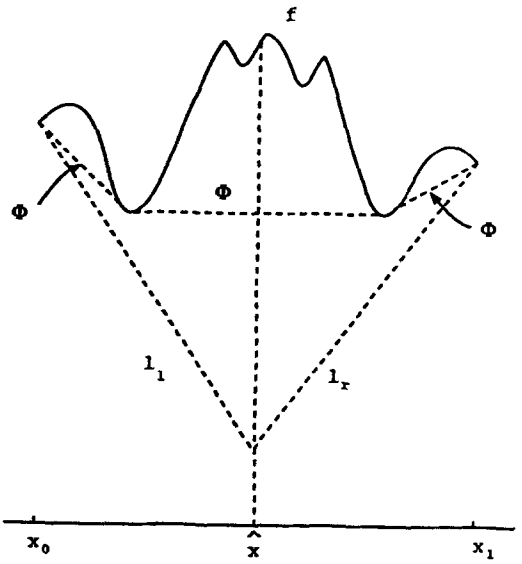

Figure 4. A good point to divide a region.

divide the region into two subregions. For more complicated cases, further region division might also lead the problem in a small subregion to the first case. In short, even though we do not know in advance which case we might encounter, it seems still worthwhile to try such a strategy.

To implement the idea, let $[x_0', x_1']$ denote the remaining subregion that comes from $[x_0, x_1]$ after the left and right LLBF operations have been performed (suppose $[x_0', x_1']$ is not empty). Then we divide $[x_0', x_1']$ into two subregions $[x_0', \hat{x}]$ and $[\hat{x}, x_1']$, saving $l_l(x)$ for $[x_0', \hat{x}]$ and $l_r(x)$ for $[\hat{x}, x_1']$. Note that we may have a possible improvement of α by $l_l(\hat{x})$ $(= l_r(\hat{x}))$. Note also that even though there is a chance that $\hat{x}$ is a better point, we will not evaluate $f(\hat{x})$ right away. This is becuase the LLBF gives us the function value at $\hat{x}$ when the procedure continues. By this way, we can save the number of function evaluations.

Based on the foregoing discussion, we have our basic algorithm, which stops when all the remaining intervals are shorter than δ, a specified accuracy.

Algorithm 1: Basic algorithm

Let $\mathcal{P}$ and $\mathcal{Q}$ be collections of subregions of $[a, b]$. Initially, $\mathcal{P}$ is a finite partition of $[a, b]$ and $\mathcal{Q}$ is empty. Denote $\alpha_i = est \min_{D_i} f$, where $D_i \in \mathcal{P}$.

Step 1. Get $D \in \mathcal{P}$ with $\alpha = est \min_{D} f = \min\{\alpha_1, \alpha_2, \cdots\}$ and the corresponding LLBF of f.

Step 2. Use the corresponding saved LLBF to shrink D, still denoting the remaining region by D. If D is empty, go to Step 1; otherwise go to Step 3.

Step 3. If the size of $D \leq \delta$, store D into $\mathcal{Q}$, save the corresponding LLBF, go to Step 1; otherwise go to Step 4.

Step 4. Get the other side LLBF, update α and $\bar{f}$, use the LLBF and $\bar{f}$ to shrink D. If $D = \emptyset$, go to Step 1; otherwise go to Step 5.

Step 5. If the size of $D \leq \delta$, store D into $\mathcal{Q}$, save the corresponding LLBF, go to Step 1; otherwise go to Step 6.

Step 6. Compute the intersection point $\hat{x}$ of left and right LLBF's. Divide D into two subregions, $D_1 = [x_0, \hat{x}]$ and $D_2 = [\hat{x}, x_1]$. Update α if $l_l(\hat{x}) < \alpha$. If $0 < \hat{x} - x_0 \leq \delta$ or/and $0 < x_1 - \hat{x} \leq \delta$, store D_1 or/and D_2 into $\mathcal{Q}$, save the left or/and right LLBF, else store D_1 or/and D_2 into $\mathcal{P}$, save the corresponding LLBF.

Step 7. If $\min\{\alpha_1, \alpha_2, \cdots\} \geq \bar{f} - \epsilon$ or $\mathcal{P} = \emptyset$, STOP; otherwise go to Step 1.

Numerical tests indicate that the performance of the basic algorithm is consistent with our intuition. It can eliminate big regions quickly. However, it is not efficient for a small region. For a small region, the slope of the LLBF is close to the derivative of the function; and thus the above technique is close to a gradient method. It is well known that the rate of convergence of a gradient method is only linear. In summary, the basic algorithm serves very well for the purpose of the global phase to eliminate large regions and to find a function value as low and as fast as possible. Nevertheless, efficient algorithms need to be developed for the local phase to search for ϵ-global minima from a small region which possibly contain a global solution.

4. An Improved Algorithm

4.1. Algorithm Development

As mentioned, it seems that an efficient algorithm should consist of a global phase and a local phase. As discussed, the basic algorithm can be used in the global phase. Thus, after the global phase is done, we have a collection of small subregions stored with their LLBF's for either one or both sides.

Note that the final $\bar{f}$ at the end of the global phase is typically lower than those $\bar{f}$'s at the time such subregions were stored. By using the lower value $\bar{f}$ and the stored LLBF's, we expect that the stored subregions can be further shrunk or simply removed. Then, at the end of the global phase, we should use the final $\bar{f}$ from the basic algorithm and the stored LLBF's to further shrink or eliminate such small subregions.

In the local phase, it seems favorable to use local minimization algorithms to find an ϵ-global minimizer, since these local methods can typically speed up the computation considerably. Let us consider the LLBF-based globally convergent superlinear rate local algorithm in [4]. Figure 5 illustrates graphically how the

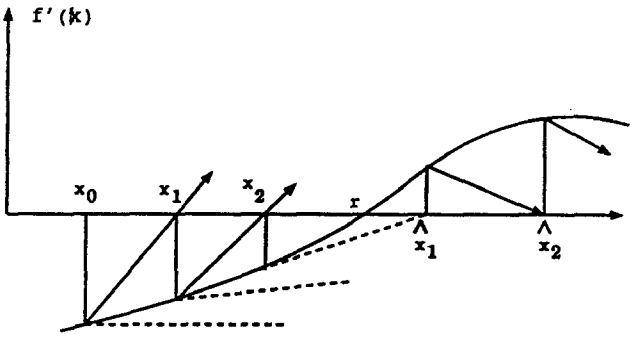

Figure 5. A local minimization algorithm and its expansion.

local minimization procedure works and why it can be blended into a global optimization algorithm in a natural manner.

Assume that we start from a point x_0 such that $f'(x_0) < 0$. We use left LUBF's to approach a local minimizer $x = r$, and left LLBF's to bracket the local minimizer r to be the global minimizer in $[x_0, \hat{x}_1]$. We can further expand rightward to a larger interval which contains the local minimizer as its global minimizer.

Note that even though we can find a local minimizer in a small subregion, and verify it is an ϵ-global minimizer in its neighborhood, the local minimizer may be not an ϵ-global solution. Thus the computation effort of searching for a local minimizer is wasted even with fast convergence algorithms. When $\bar{f}$ is below the local minimum, the LLBF of f may have a good chance to remove a larger part of or even the whole subregion. In the case that $\bar{f}$ is above the local minimum, the function value provided by the LLBF of f at one end point may even improve $\bar{f}$, since it gives us more direct information on the function value over the current subregion (while the LLBF of f' cannot). Because we do not know in advance whether or not a subregion contains a global solution, we will use the LBF's of f and f' alternately. From our numerical testing experiences, this provides an efficient strategy for dealing with small subregions.

Let $[x_0, x_1]$ be a small subregion, i.e., $0 < x_1 - x_0 \leq \delta$, $\delta > 0$ is a small number. The major steps of the local phase by using one-side LBF's, say, left LBF's, are as follows.

Step 1: Get left LLBF of f on $[x_0, x_1]$; use the technique described in Section 3 to shrink the region and still denote the remaining region by $[x_0, x_1]$. If $[x_0, x_1]$ is empty or its length $< \dfrac{\epsilon}{|m|}$, where m is the slope of the LLBF of f over $[x_0, x_1]$, STOP; otherwise go to Step 2.

Step 2: Get left LBF's of f' on $[x_0, x_1]$; use the technique explained later to shrink the region; and still denote the remaining region by $[x_0, x_1]$. If $[x_0, x_1]$ is empty or its length $< \dfrac{\epsilon}{|m|}$, STOP; otherwise go to Step 1.

To present how Step 2 is carried out, note that the usage of the LBF's of f' is different from that of a LLBF of f. From the LBF's of f', we can decide a subregion of $[x_0, x_1]$ on which f is either nondecreasing or nonincreasing. This implies that the subregion cannot contain a local minimizer and thus can be removed. We shall demonstrate how this can be done for all possible cases below.

To be specific, denote the LBF's of f' by $l^d(x)$ and $L^d(x)$ and the estimations of min and max of f' by d_l and d_u for a given region $[x_0, x_1]$, i.e.,

$$l^d(x) \le f'(x) \le L^d(x), \tag{4.1}$$

$$d_l \le f'(x) \le d_u, \ x \in [x_0, x_1], \tag{4.2}$$

$$l^d(x) = f'(x_0) + m^d(x - x_0), \tag{4.3}$$

$$L^d(x) = f'(x_0) + M^d(x - x_0). \tag{4.4}$$

Case 1. The whole region can be deleted.

(1) If $d_l \ge 0$ or $d_u \le 0$, then $f(x)$ is monotonic on $[x_0, x_1]$. Since the function values at the end points are taken care of by either $\bar{f}$ or the neighbor subregions, the whole region $[x_0, x_1]$ can be discarded.

(2) For the case $d_l < 0 < d_u$, let $l_1(x)$ be the stored LLBF over $[\bar{x}_0, x_1]$ (the region before stored, so $\bar{x}_0 \le x_0$) with slope m_1 ($m_1 < 0$) and $f_0 = f(\bar{x}_0) = l_1(\bar{x}_0)$. Note that the straight line $l_2(x)$ with slope $m_2 = \min\{m_1, d_l\}$ and passing through $(x_0, f(x_0))$ is also an LLBF of f over $[x_0, x_1]$. If

$$\Delta y \triangleq |m_1|(x_0 - \bar{x}_0) + |m_2|(x_1 - x_0) \le f_0 - \bar{f} + \epsilon, \tag{4.5}$$

then

$$\begin{aligned} f(x) &\ge f_0 + m_1(x - \bar{x}_0) \\ &\ge f_0 + m_1(x_1 - x_0) + m_1(x_0 - \bar{x}_0) \\ &\ge f_0 + m_2(x_1 - x_0) + m_1(x_0 - \bar{x}_0) \\ &= f_0 - \Delta y \\ &\ge \bar{f} - \epsilon, \qquad \forall x \in [x_0, x_1], \end{aligned} \tag{4.6}$$

thus $[x_0, x_1]$ can be removed. At the time of removing the whole region, $f(x_0)$ should be evaluated and checked since $(x_0, f(x_0))$ might be an ϵ-global solution; save $(x_0, f(x_0))$ if $f(x_0) \le \bar{f} + \epsilon$.

Case 2. Part of the region can be deleted.

If the situation is not as in case 1, then $d_l < 0 < d_u$ and $\Delta y > f_0 - \bar{f} + \epsilon$. Part of the region can be removed, depending on different cases:

(1) $m^d \ge 0$

In this case, we must have $f'(x_0) < 0$. Otherwise $m^d \ge 0$ and $f'(x_0) \ge 0$ implies $d_l = \text{est}\min_{[x_0, x_1]} f' \ge 0$. Let x_M be the point satisfying $L^d(x) = 0$. Obviously $f'(x) \le 0$ for $x \in [x_0, x_M]$; $[x_0, x_M]$ can be deleted: $x_0 \leftarrow x_M$. If $m^d > 0$, let x_m be the point satisfying $l^d(x) = 0$. If $x_m < x_1$, $f'(x_1) \ge 0$ for $x \in [x_m, x_1]$, $[x_m, x_1]$ can be deleted: $x_1 \leftarrow x_m$.

(2) $m^d < 0$

(i) $f'(x_0) \geq 0$

Since $f'(x) \geq 0$ for $x \in [x_0, x_m]$; $[x_0, x_m]$ can be discarded, i.e., $x_0 \leftarrow x_m$. If $M^d < 0$ and $x_M < x_1$, $f'(x) \leq 0$ for $x \in [x_M, x_1]$, $[x_M, x_1]$ can be discarded, i.e., $x_1 \leftarrow x_M$.

(ii) $f'(x_0) < 0$

We must have $M^d > 0$; otherwise $M^d \leq 0$, $f'(x_0) < 0$ implies $d_u = \text{est}\max_{[x_0,x_1]} < 0$. In this case, $f'(x) \leq 0$ for $x \in [x_0, x_M]$; $[x_0, x_M]$ can be discarded: $x_0 \leftarrow x_M$.

There is one particular case that needs to be taken care of before we proceed the major steps of the local phase. If $f'(x_0) = 0$ (or close to zero), $d_l < 0 < d_u$, and $f(x_0) = \bar{f}$ (or close to each other), then both steps will fail to shrink the region. For this rare case we apply the technique used in the global phase (even when $\Delta x = x_1 - x_0 \leq \delta$). In other words, we will get both left and right LLBF's of f over the region. By this way, either the whole region is removed or it is divided into two subregions.

Algorithm 2: An improved univariate global optimization algorithm

Let $\mathcal{P}$ and $\mathcal{Q}$ be collections of subregions of $[a, b]$. Initially, $\mathcal{P}$ is a finite partition of $[a, b]$ and $\mathcal{Q}$ is empty. Denote $\alpha_i = est\min_{D_i} f$, where $D_i \in \mathcal{P}$ (in Phase I) or $D_i \in \mathcal{Q}$ (in Phase II). In Phase II, "df-f-flag" is a flag; "df-f-flag = 0" indicates the process of major step 1 and "df-f-flag = 1" the process of major step 2.

Phase I: The global phase

Step 1. Perform the basic algorithm for a given δ.

Step 2. Apply the above procedure to all the elements of $\mathcal{Q}$ once, without region decomposition.

Phase II: The local phase

While $\mathcal{Q}$ is not empty, do:

Step 1. Get $D = [x_0, x_1] \in \mathcal{Q}$ with $\alpha = \min\{\alpha_1, \alpha_2, \cdots\}$ and the corresponding f_0, calculate Δy in (4.5).

Step 2. If $\Delta y \leq f_0 - \bar{f} + \epsilon$, check $f(x_0)$: if $f(x_0) \leq \bar{f} + \epsilon$, save $(x_0, f(x_0))$ into X^*, and go to Step 1; otherwise go to Step 3.

Step 3. If df-f-flag = 1, go to Step 4; otherwise go to Step 7.

Step 4. Generate left LBF's for $f'(x)$. If $d_l \geq 0$ or $d_u \leq 0$, discard the whole region and go to Step 1; otherwise go to Step 5.

Step 5. If $|f'(x_0)| \leq \epsilon$, $d_l < 0 < d_u$ and $f_0 - \bar{f} \leq \epsilon$, apply the technique of the global phase once, go to Step 8; otherwise go to Step 6.

Step 6. Shrink D by the LBF's of f'. Compute m_2, update Δy, and go to Step 8.

Step 7. Generate a left LLBF for $f(x)$. Update α, f_0, $\bar{f}$ and m_1; if the slope of the LLBF is nonnegative or $\alpha \geq \bar{f} - \epsilon$, go to Step 1; otherwise go to Step 8.

Step 8. df-f-flag $\leftarrow$ 1$-$ df-f-flag; go to Step 2.

Phase III: Check the global solution
Check X^* and get the global solution.

4.2. CONVERGENCE

THEOREM 4.1. (*Convergence in finite steps*).

Assume that $f(x)$ is continuously differentiable, the slopes of linear bounds of f and f' are bounded by a constant G over any subregions of S, and $\epsilon > 0$ is the accuracy tolerance. Then Algorithm 2 can find an ϵ-global minimum within finite steps.

Proof. We prove the theorem by showing that both Phase I and Phase II will terminate within finite steps.

1. Phase I (the global phase) terminates within finite steps.

   The stop criterion of Phase I is that every subregion in $\mathcal{P}$ is either removed or shrunk to one or more subregions with its (their) length(s) not longer than a small positive number δ (and they will be stored in $\mathcal{Q}$).

   Let $D = [x_0, x_1]$ be an arbitrary subregion in $\mathcal{P}$. There are three cases after each iteration: (1) D is totally discarded; (2) D is shrunk to a subregion not longer than δ (and is stored in $\mathcal{Q}$); (3) D is divided into two subregions. It is enough to consider the third case.

   We show that after a decomposition of D, the length of each subregion, Δ_i, $i = 1, 2$, will satisfy $\Delta_i < (x_1 - x_0) - c$, where c is a positive constant.

   Denote left and right LLBF's of $f(x)$ by

   $$l_l(x) = m_l(x - x_0) + f(x_0), \tag{4.7}$$

   $$l_r(x) = m_r(x - x_1) + f(x_1). \tag{4.8}$$

   By the definition of an LLBF, m_l and m_r satisfy the condition

   $$\max\{|m_l|, |m_r|\} \leq G, \tag{4.9}$$

   where G is a constant.

   For case (3) we have $m_l < 0$ and $m_r > 0$. For simplicity, we assume that D is now divided into $[x_0, \hat{x}]$ and $[\hat{x}, x_1]$, where $\hat{x}$ is the intersection point of l_l and l_r, without first using the LLBF's to shrink the region. This is without loss of generality, since the shrunk subregions are always smaller than original ones.

Let $c = \frac{\epsilon}{G}$. If one of subregions, say, $[x_0, \hat{x}]$, is shrunk so little (compared with $[x_0, x_1]$) that $\Delta_1 = \hat{x} - x_0 \geq (x_1 - x_0) - c$, then $\Delta_2 = x_1 - \hat{x} \leq c$, since $\Delta_1 + \Delta_2 = x_1 - x_0$.

Since $m_l < 0$ and $m_r > 0$, it is implied that $f(x) \geq l_r(\hat{x})$ $(= l_l(\hat{x}))$ for all $x \in [x_0, x_1]$ and

$$\Delta l_r \triangleq l_r(x_1) - l_r(\hat{x}) = f(x_1) - l_r(\hat{x})$$

$$= m_r(x_1 - \hat{x}) \leq Gc = \epsilon, \tag{4.10}$$

we have

$$f(x) \geq l_r(\hat{x}) \geq f(x_1) - \epsilon \geq \bar{f} - \epsilon, \quad \forall x \in [x_0, x_1], \tag{4.11}$$

which means that the whole region can be discarded.

Therefore, after a decomposition, D is either totally removed or divided into two subregions, whose lengths satisfy $\Delta_i < x_1 - x_0 - c$. Since $[x_0, x_1]$ is finite ($S = [a, b]$ is finite) and c is a constant, D eventually will either be removed after decomposition or become some elements of $\mathcal{Q}$ (i.e., subregions with length $\leq \delta$) within finite steps. Since this is true for any element of $\mathcal{P}$, Phase I will terminate in finite steps.

2. Phase II terminates within finite steps.

After Phase I, $\mathcal{Q}$ contains finite elements. Phase II terminates when $\mathcal{Q}$ is empty, i.e., all subregions in $\mathcal{Q}$ are removed. The condition for a subregion $D = [x_0, x_1] \in \mathcal{Q}$ to be removed is that (4.5) is satisfied. Since (4.5) is satisfied if $\Delta x = x_1 - x_0 \leq \frac{\epsilon}{G}$, it is enough to show that D will be shrunk to one or more subregions with length $\leq \frac{\epsilon}{G}$ in finite steps. In Phase II, a typical iteration includes two major steps as described in Section 4.1. We show that the length of D will be reduced at least by a constant c after each iteration.

Without loss of generality, suppose the LBF's of f' are generated first. Rewrite (4.2), (4.3), and (4.4)

$$d_l \leq f'(x) \leq d_u, \quad x \in [x_0, x_1], \tag{4.2}$$

$$l^d(x) = m^d(x - x_0) + f'(x_0), \quad \text{the LLBF of } f' \text{ over } [x_0, x_1], \tag{4.3}$$

$$L^d(x) = M^d(x - x_0) + f'(x_0), \quad \text{the LUBF of } f' \text{ over } [x_0, x_1], \tag{4.4}$$

where constants d_l and d_u are estimations of min and max of f' on $[x_0, x_1]$.

Suppose $d_l < 0 < d_u$ (otherwise the whole region can be discarded). We can, just considering the cut on the left side, at least reduce the region $[x_0, x_1]$ by $x_M - x_0 = -\frac{f'(x_0)}{M^d} = \left|\frac{f'(x_0)}{M^d}\right|$ or $x_m - x_0 = -\frac{f'(x_0)}{m^d} = \left|\frac{f'(x_0)}{m^d}\right|$.

If $|f'(x_0)| > \epsilon$, $\min\left\{\left|\frac{f'(x_0)}{M^d}\right|, \left|\frac{f'(x_0)}{m^d}\right|\right\} > \frac{\epsilon}{G} = c$; i.e., the major step on f' already provides us with the required reduction.

If $|f'(x_0)| \leq \epsilon$, we consider the alternative step on f. Denote

$$l_l(x) = m_l(x - x'_0) + f(x'_0), \quad \text{the LLBF of } f \text{ over } [x'_0, x'_1], \tag{4.12}$$

where $[x'_0, x'_1]$ is the region after a cut made by using LBF of f', i.e., $[x'_0, x'_1] \subset [x_0, x_1]$. Suppose $m_l < 0$; otherwise the whole region can be removed.

If $f(x'_0) - \bar{f} > \epsilon$, then we have a cut

$$x''_0 - x'_0 = -\frac{f(x'_0) - \bar{f}}{m_l} = \frac{f(x'_0) - \bar{f}}{|m_l|} > \frac{\epsilon}{G} = c, \tag{4.13}$$

i.e., the major step on f provides us with the required reduction.

If $0 \leq f(x'_0) - \bar{f} \leq \epsilon$ and $|f'(x_0)| \leq \epsilon$, it is possible that both major steps will not provide us with the required reduction. In this case, we apply the technique used in the global phase once, as mentioned in Step 5 of Phase II. After that, D is either totally discarded or divided into two subregions whose lengths are less than $(x_1 - x_0) - c$, where $c = \frac{\epsilon}{G}$.

Therefore, in any cases, the region will be cut by at least a constant c after each iteration. This implies that any $D \in \mathcal{Q}$ will be shrunk to one or more subregions with length $\leq c = \frac{\epsilon}{G}$ and thus is removed from $\mathcal{Q}$ in finite steps. Since $\mathcal{Q}$ contains finite elements, Phase II will terminate within finite steps.

5. Numerical Results

We have tested our global optimization algorithm on the following one-dimensional multi-extremal functions from [11].

$$f_1(x) = \sin(x) + \sin(\frac{10x}{3}) + \ln(x) - 0.84x, \quad 2.7 \leq x \leq 7.5 \tag{5.1}$$

$$f_2(x) = \sin(x) + \sin(\frac{2x}{3}), \quad 3.1 \leq x \leq 20.4 \tag{5.2}$$

$$f_3(x) = -\sum_{i=1}^{5} \sin((i+1)x + i), \quad -10 \leq x \leq 10 \tag{5.3}$$

$$f_4(x) = (x + \sin(x))e^{-x^2}, \quad -10 \leq x \leq 10 \tag{5.4}$$

$$f_5(x) = -\sum_{i=1}^{10} \frac{1}{(k_i(x - a_i))^2 + c_i}, \quad 0 \leq x \leq 10 \tag{5.5}$$

The k_i, a_i, and c_i are parameters chosen and varied to create different problems. $f_6(x)$ has the same function form as $f_5(x)$. The parameters used are the same as those in [11]. f_7 corresponds to minimizations of 100 Shekel functions (f_5) with random coefficients $0 \leq a_i \leq 10$, $1 \leq k_i \leq 3$ and $0.1 \leq c_i \leq 0.3$.

Table I. Comparison for 1-D test problems

function	S-LLB	N-LLB	O-LLB	Zil1	Zil2	Strong	Pijav	Brent	Batish
f1	12	12	19	33	29	45	462	25	120
f2	18	18	24	37	38	442	448	45	158
f3	73	73	94	125	165	150	3817	161	816
f4	13	19	35	35	34	98	376	229	83
f5	36	37	40	42	41	102	280	294	484
f6	32	36	71	45	44	69	624	492	325
f7	29	40	81	32	44	94	360	376	422

The column headings correspond to the following algorithms:
(1) S-LLB: Algorithm 2 with special LBF's.
(2) N-LLB: Algorithm 2 in this paper.
(3) O-LLB: The algorithm in [3] with piecewise LLBF's.
(4) Zil1: The P^* algorithm of Zilinskas reported in [11].
(5) Zil2: The algorithm of Zilinskas reported in [11].
(6) Strong: The algorithm of Strongin reported in [11].
(7) Pijav: The algorithm of Pijavskij and Shubert reported in [11].
(8) Brent: The algorithm of Brent reported in [11].
(9) Batish: The algorithm of Batishchev reported in [11].

The performance of our algorithm is compared with the algorithms in [3] and other one-dimensional algorithms reported in [11]. To be consistent, the same stopping criterion $\epsilon = 10^{-6}$ is used. In Table I, our test results are summarized in column N-LLB (using linear bounds constructed by the general procedure) and S-LLB (using special linear bounds based on the additional information of the given problem). In column O-LLB the best results of the algorithm in [3] are presented. The other columns correspond to those in [11]. For LBF methods, each entry is the total number of function, derivative, and LBF evaluations. For other methods, it refers to the total number of function evaluations.

From the results, we can see that our algorithm has high potential of success. Note that in O-LLB piecewise LLBF's are used, which, of course, cost higher computation burden. Also, for f_4 and f_5 the LLBF's are obtained by treating each iterm as a generic term, instead of using the general product and composite forms. To see the effect due to using additional information based upon the given problem, in column S-LLB, we use the special one-piece LLBF's for f_4 though f_7. The result indicates that the specialized LLBF's may have quite an impact, depending upon the problems.

6. Discussion

In this paper, we present a method to find improved LBF's for factorable functions over n-dimensional boxes. We developed a univariate global optimization algorithm by applying LBF's to a given function and its derivative. The algorithm is proven to converge in a finite number of iterations in order to find ϵ-global minima.

Numerical testing indicates some improvement over another LBF-based global algorithm and the high potential of our algorithm.

Global optimization for n-dimensional cases is obviously more complicated. To extend our method to n-dimensional cases, not all the techinques which are efficient for one-dimensional cases can be preserved. By exploring the properties of LLBF, we have recently developed a three-phase algorithm for multivariate problems along the same direction. The results are reported in [12].

References

1. M. Bromberg and T. S. Chang, One Dimensional Global Optimization Using Linear Lower Bounds, *Recent Advances in Global Optimization*, eds. C. A. Floudas and P. M. Pardalos, Princeton University Press, 1992, pp 200–220.
2. M. Bromberg and T. S. Chang, Global Optimization Using Linear and Convex Lower Bounds, Technical Report No. UCD-ECE-SCR-93/5, October 1993.
3. T. S. Chang and C. L. Tseng, A New Linear Lower Bound and One Dimensional Global Optimization, Technical Report No. UCD-ECE-SCR-94/2, January 1994.
4. T. S. Chang and X. Wang, Univariate Optimization Using Linear Bounding Functions: A Framework of developing algorithms with properties such as global convergence, superlinear/quadratic rate, function and Hessian evaluation free, working paper.
5. C. A. Floudas and P. M. Pardalos, *Recent Advances in Global Optimization*, Princeton University Press, 1992.
6. R. Horst and H. Tuy, *Global Optimization: Deterministic Approaches*, Springer-Verlag, 1990.
7. G. P. McCormick, Converting General Nonlinear Programming Problems to Separable Nonlinear Programming Problems, Technical paper serial T-267, Institute for Management Science and Engineering, The George Washington University, Washington DC, June 1972.
8. G. P. McCormick, Computability of Global Solutions to Factorable Nonlinear Programs: Part I – Convex Underestimating Problems, *Math. Prog.* **10** (no. 2), April 1976, pp 147–175.
9. H. Ratschek and J. Rokne, *New Computer Methods for Global Optimization*, Ellis Horwood Limited, 1988.
10. R. T. Rockafellar, *Convex Analysis*, Princeton University Press, 1970.
11. A. Torn and A. Zilinskas, *Global Optimization*, Springer-Verlag, 1989.
12. X. Wang and T. S. Chang, A Multivariate Global Optimization Algorithm for Factorable Functions Using Linear Bounding Functions, submitted.