Journal of Global Optimization **8:** 323–348, 1996.

Global Optimization of Hölder Functions

ERIC GOURDIN
Ecole Polytechnique de Montréal, Department of Mathematics and Industrial Engineering, P.O. Box 6079, Station "Centre-ville", Montréal (Québec) H3C 3A7, Canada

BRIGITTE JAUMARD
Ecole Polytechnique de Montréal, GERAD and Department of Mathematics and Industrial Engineering, P.O. Box 6079, Station "Centre-ville", Montréal (Québec) H3C 3A7, Canada

and

RACHID ELLAIA
Ecole Mohammadia d'Ingénieurs, Département E.G.T., Avenue Ibn Sina, BP 765 - Rabat - Agdal, Maroc

(Received: 15 July 1994; accepted: 9 May 1995)

Abstract. We propose a branch-and-bound framework for the global optimization of unconstrained Hölder functions. The general framework is used to derive two algorithms. The first one is a generalization of Piyavskii's algorithm for univariate Lipschitz functions. The second algorithm, using a piecewise constant upper-bounding function, is designed for multivariate Hölder functions. A proof of convergence is provided for both algorithms. Computational experience is reported on several test functions from the literature.

Key words: Global optimization, Hölder functions, Lipschitz optimization.

1. Introduction

Consider the problem of globally maximizing a real-valued function f defined on a hyperrectangle $R = [a, b] = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_n, b_n] \subset \mathbb{R}^n$. The function f satisfies the Hölder (or generalized Lipschitz) property, i.e., there exists $L \in \mathbb{R}$ and $\alpha \geqslant 1$ such that:

$$\forall x \in R, \forall y \in R \quad |f(x) - f(y)| \leqslant L\|x - y\|^{\frac{1}{\alpha}}$$

where $\| \cdot \|$ denotes the Euclidean norm. Any constant L for which the above property is satisfied, is called a Lipschitz constant for the function f. Note that most of the results of this paper remain valid if other norms are considered.

Such a problem arises in many applications. One example is the simple plant location problem under a uniform delivered price policy (see, e.g., Hanjoul *et al.* [9]).

The aim of the paper is to study the extension of Piyavskii's method [19] [20] for Lipschitz optimization to Hölder optimization, both in the univariate and the multivariate case. This is done using the general branch-and-bound framework

which was already proposed (see [7], [5], see also [10]) for the multivariate Lipschitz optimization. It is extended and adapted to design two algorithms HOL^1 and HOL^n, respectively, for univariate and multivariate Hölder optimization.

The algorithm for univariate Lipschitz optimization is based on the iterative construction of an upper-bounding function corresponding to the lower envelope of linear functions. When extended to the optimization of Hölder functions, the linear functions change into parabolic functions. The upper-bounding function is then a piecewise concave function and the determination of its successive maxima can be achieved through line search techniques. For integer values of α, the determination of each maximum can be obtained by solving an equation of degree α. Note that for $\alpha = 1$, the problem reduces to the usual Lipschitz problem, which is therefore a special case of the Hölder problem.

The piecewise concave upper-bounding function can be naturally extended to the case of multivariate Hölder functions but becomes very difficult to handle. Indeed, the computation of a local maximum requires the determination of the intersection of several hypersurfaces of equation: $y = f(x^j) + L\|x - x^j\|^{1/\alpha}$. A relaxation of the upper-bounding function is hence proposed. The algorithm for multivariate Hölder optimization uses a piecewise constant upper-bounding function and a partitioning scheme with hyperrectangles.

The paper is organized as follows. The general branch-and-bound framework for the optimization of Hölder functions is stated in the next section. The algorithms for univariate and multivariate Hölder optimization, both derived from the general framework are described in Sections 3 and 4 respectively. A precise description of each algorithm is provided together with a convergence proof. Finally, computational experience is reported in Section 5.

2. A Branch-and-Bound Framework for Optimizing Hölder Functions

In this section we describe a branch-and-bound framework for the global maximization of Hölder functions. The underlying idea is to replace the objective function by an analytical upper-bounding function which is easier to maximize. This idea has already been extensively used when the objective function satisfies a Lipschitz property. Indeed, the Lipschitz property provides a natural way to build upper-bounding functions (see, e.g., Piyavskii [19] [20]). Some notations are given below.

2.1. NOTATION

Considering the general problem of maximizing a continuous function over a compact set, the principle of a branch-and-bound algorithm is to partition iteratively the problem into smaller entities called subproblems. In the branch-and-bound framework we propose in this paper, a subproblem $\mathcal{P}$ is characterized by $R_{\mathcal{P}} \subset R$, a hyperrectangle of $\mathbb{R}^n$ (when $n = 1$, it reduces to an interval), $B_{\mathcal{P}}$, a constant

upper-bound of f over R (for all $x \in R, f(x) \leqslant B_{\mathcal{P}}$) and possibly some other informations (see the detailed descriptions in Sections 3 and 4). We hence use the notation: $\mathcal{P} = (R_{\mathcal{P}}, B_{\mathcal{P}}, \ldots)$. The subproblems are stored in a list denoted by $\mathcal{L}$.

At step k, we denote by f_{opt}^k, x_{opt}^k and F_{opt}^k respectively the best function evaluation, the incumbent point and the largest upper-bound for all subproblems in $\mathcal{L}$. Among all the subproblems in $\mathcal{L}$, all those with an upper-bound lower than $f_{opt}^k + \varepsilon$ can be discarded as they cannot contain an ε-optimal solution, i.e., a value larger than $f_{opt}^k + \varepsilon$ (*fathoming test*). Finally, when the gap between F_{opt}^k and f_{opt}^k becomes lower than the chosen tolerance ε, f_{opt}^k is an ε-optimal value and the branch-and-bound algorithm can be stopped (*optimality test*).

2.2. BRANCH-AND-BOUND FRAMEWORK

The general branch-and-bound framework for the global maximization of Hölder functions can now be formally stated:

1. Step 0 (Initialization).
$k \leftarrow 0$:
$R^0 \leftarrow R$;
Choose a discrete set $D^0 \subset R^0$; *Initial lower-bounding rule*
$f_{opt}^0 \leftarrow \max_{x \in D^0} f(x)$;
$x_{opt}^0 \leftarrow \arg\max_{x \in D^0} f(x)$;
Let F^0 be an upper-bounding function of f on R^0; *Initial upper-bounding rule*
Compute $B^0 = \max_{x \in R^0} F^0(x)$;
$F_{opt}^0 \leftarrow B^0$;
If $F_{opt}^0 - f_{opt}^0 \leqslant \varepsilon$ stop: f_{opt}^0 is an ε-optimal value;
$\mathcal{L} \leftarrow \{\mathcal{P}^0 = (R^0, B^0, \ldots)\}$;

2. Current step ($k = 1, 2, \ldots$).
While $\mathcal{L}$ is non-empty **do**
$k \leftarrow k + 1$;
Extract from $\mathcal{L}$ the subproblem $\overline{\mathcal{P}} = (\overline{R}, \overline{B}, \ldots)$ with the largest upper-bound $\overline{B}$; *Selection rule*

2.1 Branching step.
Partition $\overline{R}$ into p hyperrectangles $\overline{R}^1, \overline{R}^2, \ldots, \overline{R}^p$; *Branching rule*

2.2 Evaluation of the subproblems.
For $j = 1$ to p **do**
Let $\overline{R}^j = [\bar{a}^j, \bar{b}^j]$;

2.2.1 Lower bound

Choose $\bar{x}^j \in \overline{R}^j$; *Lower-bounding rule*

If $f(\bar{x}^j) > f_{opt}^{k-1}$ **then**
$f_{opt}^k \leftarrow f(\bar{x}^j)$;
$x_{opt}^k \leftarrow \bar{x}^j$;
Else
$f_{opt}^k \leftarrow f_{opt}^{k-1}$;
$x_{opt}^k \leftarrow x_{opt}^{k-1}$;
Endif;

2.2.2 Upper bound
Build an upper bounding function $\overline{F}^j$ on $\overline{R}^j$;
Compute $\overline{B}^j = \max_{x \in \overline{R}^j} \overline{F}^j(x)$; *Upper-bounding rule*

2.2.3 Update of $\mathcal{L}$
Add $\overline{\mathcal{P}}^j = (\overline{R}^j, \overline{B}^j, \ldots)$ to $\mathcal{L}$;

EndFor; (End of Step 2.2)

2.3 Fathoming test
Delete from $\mathcal{L}$ all subproblems $\mathcal{P} = (R_\mathcal{P}, B_\mathcal{P}, \ldots)$ with $B_\mathcal{P} \leqslant f_{opt}^k + \varepsilon$;

2.4 Optimality test
Let F_{opt}^k be the maximum of all B^j;
If $F_{opt}^k - f_{opt}^k \leqslant \varepsilon$ **then**
stop: f_{opt}^k is an ε-optimal value;
Endif;

Endwhile

Note that the fathoming test is not necessary for the convergence of the branch-and-bound scheme. Its aim is only to improve the practical efficiency of the algorithm and to reduce the memory requirements. Indeed, if it is not performed, the subproblems which would have been fathomed, remain in $\mathcal{L}$ and will never be considered in a subsequent branching step. Moreover, a double-ended priority queue or *maxminheap* can be used to implement the list $\mathcal{L}$, in order to improve further the efficiency of the algorithm (see Atkinson *et al.* [1] for a definition and properties of double-ended priority queues). Indeed, in a maxminheap, both the highest and the lowest among k values are accessed in constant time. Its structure can be updated in $O(\log_2 k)$ time when adding or removing one element. Hence, when a maxminheap is used, the selection rule and the fathoming test are performed in constant time. The update which is necessary when a subproblem is added to $\mathcal{L}$,

requires $O(\log_2 k)$ operations, instead of $O(k)$ operations if a simple list is used (see [12]).

This optimization scheme is applied in the next two sections in order to design practical algorithms for the univariate case (Section 3) and the multivariate case (Section 4).

3. Univariate Hölder Optimization

In this section, we specialize the branch-and-bound framework of the previous section to the problem of maximizing a univariate function f over an interval $R = [a, b] \subset \mathbb{R}$, assuming that f satisfies the Hölder condition:

$$\forall x \in R, \forall y \in R \quad |f(x) - f(y)| \leqslant L|x - y|^{\frac{1}{\alpha}},$$

with a known Lipschitz constant L and $\alpha \geqslant 1$. A resulting algorithm called HOL^1, is proposed below.

The upper-bounding function used in univariate Lipschitz optimization is extended to the case of univariate Hölder optimization. It leads to a piecewise concave upper-bounding function which is described next. The definition of the subproblems used in HOL^1 follows. The branch-and-bound rules and tests for the algorithm are then stated. A formal description of HOL^1 is obtained by replacing the specific rules and tests in the general branch-and-bound framework of Section 2. The proof of convergence of the algorithm is given at the end of this section.

3.1. A PIECEWISE CONCAVE UPPER-BOUNDING FUNCTION

For any given evaluation point $x^j \in R = [a, b]$ define the function f^j by:

$$\forall x \in R \quad f^j(x) = f(x^j) + L|x - x^j|^{1/\alpha}.$$

It follows from the Hölder property that:

$$\forall x \in R \quad f(x) \leqslant f^j(x),$$

i.e., f^j is an upper-bounding function for f over R.

More generally, given a sequence of k points $(x^j)_{j=1,2,\ldots,k}$ in R, the function F^k defined as the lower envelope of the upper-bounding functions

$$F^k(x) = \min_{j=1,2,\ldots,k} f(x^j) + L|x - x^j|^{1/\alpha}$$

is also an upper-bounding function of f over R.

Assume the sequence $(x^j)_{j=1,2,\ldots,k}$ is ordered: $x^j \leqslant x^{j+1}$ for all $j = 1, 2, \ldots k - 1$. Let a^j and b^j be any two successive points in this sequence. We define the upper-bounding function F_2^k spanning the interval $[a^j, b^j]$ by:

$$\forall x \in [a^j, b^j], F_2^k(x) = \min \left\{ \overbrace{f(a^j) + L(x - a^j)^{1/\alpha}}^{f_a(x)}, \overbrace{f(b^j) + L(b^j - x)^{1/\alpha}}^{f_b(x)} \right\}.$$

Note that this upper-bounding function F_2^k is not the tightest, in the sense that

$$\forall x \in [a^j, b^j], \quad F_2^k(x) \geqslant F^k(x) = \min_{j=1,2,\ldots,k} f(x^j) + L|x - x^j|^{1/\alpha}$$

and the equality does not necessary hold for all x. This comes from the fact that for some Hölder functions, one of the functions $f(x^j) + L|x - x^j|^{1/\alpha}$, evaluated at a point x^j outside the interval $[a^j, b^j]$, might cut the concave tooth defined by $f_a(x)$ and $f_b(x)$ *. However, such a situation being difficult to detect *a priori*, the relaxed upper-bounding function F_2^k is used in the algorithm, as it is much easier to handle.

Also note that both functions f_a and f_b are concave. This implies that the function F^k is also concave over $[a^j, b^j] \subseteq R$, and more generally, piecewise concave over R. Moreover, f_a and f_b are respectively increasing and decreasing over $[a^j, b^j]$. The Hölder condition implies that $f_a(a^j) \leqslant f_b(b^j)$ and $f_a(b^j) \geqslant f_b(b^j)$. Hence, by continuity of the function f on $[a^j, b^j]$, we deduce that there is a unique point $x^{p_j} \in [a^j, b^j]$ such that $f_a(x^{p_j}) = f_b(x^{p_j})$. Denote by B^j this common value. The point (x^{p_j}, B^j) is the intersection of the graphs of the two functions f_a and f_b (see Figure 1). It follows from the monotony properties of both functions that B^j is also the unique maximum of $F^k(x)$ over $[a^j, b^j]$. The concave part of the upper-bounding function spanning $[a^j, b^j]$ can be viewed as a concave tooth of a saw-tooth cover. From now on, we will refer to x^{p_j} as the "peak point" of the concave upper-bounding function spanning $[a^j, b^j]$.

3.2. Rules and tests for algorithm HOL^1

The concave upper-bounding function spanning $R^j = [a^j, b^j]$ is characterized by the interval R^j, its unique maximum B^j and the peak point x^{p_j}. The corresponding subproblem is hence defined by: $\mathcal{P}^j = (R^j, B^j, x^{p_j})$. The branch-and-bound rules defining the algorithm are the following:

Selection rule. The subproblem $\overline{\mathcal{P}} = (\overline{R}, \overline{B}, \bar{x}^p) \in \mathcal{L}$ with the largest upper bound $\overline{B}$ is chosen (ties are broken arbitrarily). Let $\overline{R} = [\bar{a}, \bar{b}]$ be the corresponding interval.

Branching rule. The interval $\overline{R} = [\bar{a}, \bar{b}]$ is split into the two sub-intervals ($p = 2$):

$$\begin{cases} \overline{R}^1 = [\bar{a}, \bar{x}^p] \\ \overline{R}^2 = [\bar{x}^p, \bar{b}]. \end{cases}$$

* Consider, for example, the behaviour of algorithm HOL^1, after two iterations, maximizing the function f defined over $R = [0, 1]$ by

$$\begin{cases} \forall x \in [0, 1/2] & f(x) = 2\sqrt{|x - 1/4|}, \\ \forall x \in [1/2, 1] & f(x) = 1. \end{cases}$$

(f satisfies the Hölder condition with $L = 2$ and $\alpha = 2$).

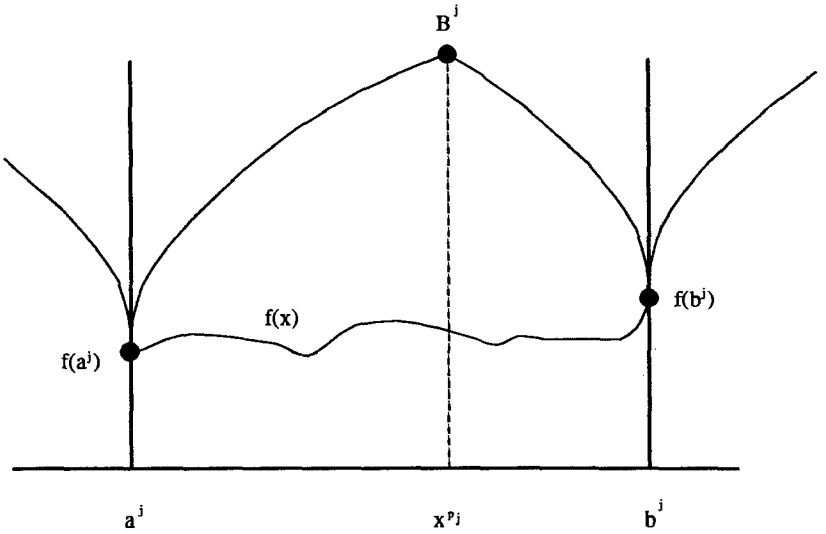

Fig. 1. Piecewise concave upper-bounding function.

Initial lower bounding rule. The function f is evaluated at the extreme points of the starting interval $R = [a, b] : D^0 = \{a, b\}$.

Current lower bounding rule The function f is evaluated at $\bar{x}^{p_j}$, the peak point of the upper-bounding function spanning $\overline{R}^j$.

Initial and current upper bounding rule. The constant upper-bound over $\overline{R}^j$ is defined as the maximum of the current upper-bounding function F_2^k spanning $\overline{R}^j$:

$$\begin{aligned}\overline{B}^j &= \max_{x \in \overline{R}^j} F^k(x) \\ &= \max_{x \in \overline{R}^j} \min\{f(a^j) + L(x - a^j)^{1/\alpha}, f(b^j) + L(b^j - x)^{1/\alpha}\}.\end{aligned}$$

Using the optimization scheme proposed in the previous section with these rules, the algorithm HOL^1 is fully described. The only difficulty comes from the determination of $\max_{x \in \overline{R}^j} F_2^k(x)$, i.e., the maximum value of the current upper-bounding function F_2^k, and the corresponding point $\bar{x}^{p_j}$, i.e., the peak point. As the two values $\overline{B}^j$ and $\bar{x}^{p_j}$ are obtained from the same system of equations, the lower-bounding rule and the upper-bounding rule of the branch-and-bound algorithm are performed simultaneously. Moreover, as the upper-bounding function spanning $[a^j, b^j]$ is unimodal (indeed, it is concave), but not differentiable, its maximum can be found by any line search algorithm, such as the Fibonacci search or the Golden section search (see [14]). In the cases where $\alpha = 2, 3$ or 4, an analytical expression

of the maximum can be derived to improve the efficiency of the algorithm (see the appendix).

Another interesting approach, which could be further investigated, would be to approximate the upper-bounding function by two wisely chosen tangents to f_a and f_b. As the upper-bounding function is concave over each interval $[a^j, b^j]$, it would hence be a relaxed but still valid upper-bounding function for which the determination of the maximum would be much easier to perform. Indeed, an analytical expression can be obtained in such a case.

3.3. CONVERGENCE OF HOL^1

We first derive a condition for the algorithm to be finite, i.e., to stop after a finite number of step when ε is equal to 0.

PROPOSITION 1. *The algorithm HOL^1 is finite if, at a certain step, $\bar{x}^p = \bar{a}$ or $\bar{b}$.*

Proof. Assume $\bar{x}^p = \bar{a}$ at step k of algorithm HOL^1. By definition of the peak point $\bar{x}^p$, this immediately implies that $\overline{B} = f(\bar{a})$. By definition, $F^k_{opt} = \overline{B}$ and clearly $f^k_{opt} = f(\bar{a})$. The stopping test is hence satisfied with $\varepsilon = 0$. A similar reasoning can be made assuming $\bar{x}^p = \bar{b}$. ∎

We now prove the asymptotic convergence of HOL^1, i.e., that the bounds it provides can be made arbitrarily close to f^*, by performing a sufficient number of steps. In practice, this means the algorithm stops after a finite number of steps for $\varepsilon > 0$.

THEOREM 1. *Either the algorithm HOL^1 is finite, or it asymptotically converges:*

$$\lim_{k\to+\infty} F^k_{opt} = \lim_{k\to+\infty} f^k_{opt} = f^*(= \max_{x\in R} f(x)).$$

Proof. It follows from the bounding rule that $(F^k_{opt})_{k\in\mathbb{N}}$ is a non-increasing sequence bounded from below by f^*. By construction, the sequence $(f^k_{opt})_{k\in\mathbb{N}}$ is non-decreasing and bounded from above by f^*. These two sequences are thus convergent and so is the non-increasing sequence $(F^k_{opt} - f^k_{opt})_{k\in\mathbb{N}}$. Denote respectively by F_{opt}, f_{opt} and δ, the limits of these three sequences.

We now prove the result (i.e., that $F_{opt} = f_{opt}$ or in other words that $\delta = 0$) by contradiction. Assume that $\delta > 0$ and let

$$\varepsilon = \frac{\delta^\alpha}{2\alpha L^{\alpha-1}(b-a)^{1-1/\alpha}} > 0. \tag{1}$$

As F^k_{opt} converges to F_{opt}, there is an $N \in \mathbb{N}$ such that: $\forall k \geqslant N \quad F^k_{opt} - F_{opt} \leqslant \varepsilon$. Denote by $\mathcal{P} = (R, B, x^p)$ the subproblem considered at a step $k \geqslant N (B = F^k_{opt})$. The branching on $\mathcal{P}$ leads to two subproblems

$$\begin{cases} \mathcal{P}^1 = (R^1, B^1, x^{p_1}) & \text{with} \quad R^1 = [a, x^p] \\ \mathcal{P}^2 = (R^2, B^2, x^{p_2}) & \text{with} \quad R^2 = [x^p, b]. \end{cases}$$

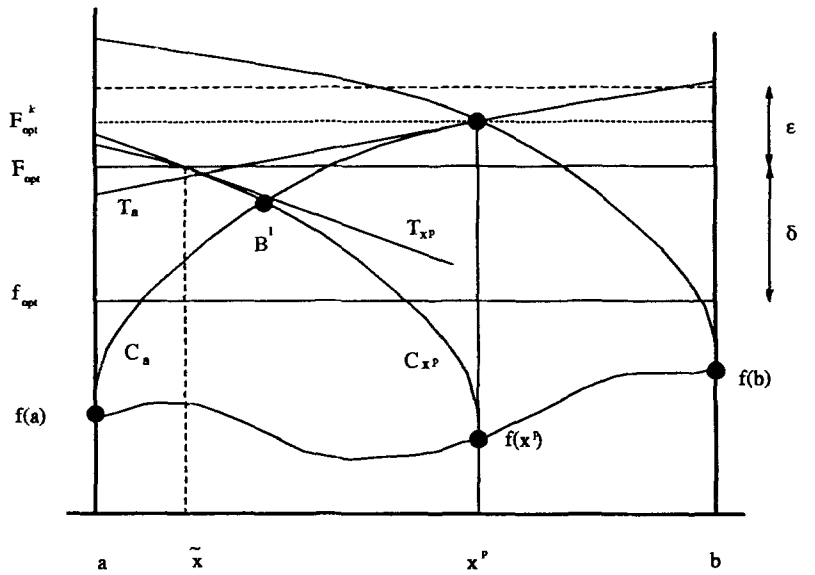

Fig. 2. Illustration of the proof of convergence.

We first consider subproblem $\mathcal{P}^1$. The upper-bounding function spanning R^1 is the lower envelope of two concave functions f_a and f_{x^p}. Denote by C_a and C_{x^p} the corresponding curves:

$$\begin{cases} C_a: \ y = f_a(x) \quad (= f(a) + L(x-a)^{1/\alpha}) \\ C_{x^p}: \ y = f_{x^p}(x) \quad (= f(x^p) + L(x^p - x)^{1/\alpha}). \end{cases}$$

Let $\tilde{x}$ be the intersection point of $y = F_{opt}$ with the curve C_{x^p}. It is the solution of the equation:

$$F_{opt} = f(x^p) + L(x^p - x)^{1/\alpha}.$$

It follows that

$$\tilde{x} = x^p - \left(\frac{F_{opt} - f(x^p)}{L}\right)^{\alpha}.$$

For the convenience of the following computations, we express $\tilde{x}$ as a combination of a and x^p:

$$\tilde{x} = \tilde{\lambda} a + (1 - \tilde{\lambda}) x^p.$$

Using this notation we derive the following useful relations:

$$\tilde{\lambda} = \frac{(F_{opt} - f(x^p))^{\alpha}}{L^{\alpha}(x^p - a)} \tag{2}$$

$$\tilde{\lambda}^{1/\alpha} = \frac{F_{opt} - f(x^p)}{L(x^p - a)^{1/\alpha}}. \tag{3}$$

Consider the tangent at the point x^p to the curve C_a:

$$T_a : y = f(a) + L(x^p - a)^{1/\alpha} + \frac{L}{\alpha}(x - x^p)(x^p - a)^{1/\alpha - 1} \tag{4}$$

and the tangent at the point $\tilde{x}$ to the curve C_{x^p}:

$$T_{x^p} : y = f(x^p) + L(x^p - \tilde{x})^{1/\alpha} + \frac{L}{\alpha}(\tilde{x} - x)(x^p - \tilde{x})^{1/\alpha - 1} \tag{5}$$

$$= f(x^p) + L\tilde{\lambda}^{1/\alpha}(x^p - a)^{1/\alpha} + \frac{L}{\alpha}(\tilde{x} - x)\tilde{\lambda}^{1/\alpha - 1}(x^p - a)^{1/\alpha - 1}. \tag{6}$$

As the functions f_a and f_{x^p} are concave, the tangents T_a and T_{x^p} are respectively above C_a and C_{x^p}. Let y_T denote the intersection point of T_a and T_{x^p}. Adding $\tilde{\lambda}^{1/\alpha} \times$ (4) and $\tilde{\lambda} \times$ (6), yields

$$y_T(\tilde{\lambda} + \tilde{\lambda}^{1/\alpha}) = \tilde{\lambda} f(a) + \tilde{\lambda} f(x^p) + L\tilde{\lambda}^{1/\alpha}(x^p - a)^{1/\alpha}\left(1 + \tilde{\lambda}\left(\frac{\alpha - 1}{\alpha}\right)\right). \tag{7}$$

Using (3) in (7), and after some rearrangements, we obtain:

$$y_T = \frac{\left(f(a) + \left(1 - \frac{\tilde{\lambda}}{\alpha}\right) L(x^p - a)^{1/\alpha}\right)(F_{opt} - f(x^p)) + \tilde{\lambda} F_{opt} L(x^p - a)^{1/\alpha}}{F_{opt} - f(x^p) + \tilde{\lambda} L(x^p - a)^{1/\alpha}}. \tag{8}$$

The upper-bounding rule defines B^1 as the intersection of C_a and C_{x^p}. It follows from the concavity of f_a and f_{x^p} that

$$B^1 \leqslant y_T. \tag{9}$$

From the definition of ε (identity (1)) and from the fact that $F_{opt} - f(x^p) > \delta$ it follows that

$$\frac{(F_{opt} - f(x^p))^\alpha}{\alpha L^{\alpha - 1}(x^p - a)^{1 - 1/\alpha}} > \varepsilon.$$

This implies that

$$\frac{\tilde{\lambda}}{\alpha} L(x^p - a)^{1/\alpha} > \varepsilon. \tag{10}$$

As

$$f(a) + \left(1 - \frac{\tilde{\lambda}}{\alpha}\right) L(x^p - a)^{1/\alpha} = B - \frac{\tilde{\lambda}}{\alpha} L(x^p - a)^{1/\alpha}, \tag{11}$$

using (10), we get

$$f(a) + \left(1 - \frac{\tilde{\lambda}}{\alpha}\right) L(x^p - a)^{1/\alpha} < B - \varepsilon \leqslant F_{opt}. \tag{12}$$

Combining this last inequality with (8) and (9), we finally obtain:

$$B^1 < F_{opt}.$$

Using a similar reasoning on the second subproblem, we also obtain $B^2 < F_{opt}$. As the iteration number k is such that $k \geqslant N$, $\mathcal{L}$ contains at most $N+1$ subproblems as some of them may have been discarded. Hence, after at most $N+1$ iterations after iteration N (after $2N+1$ iterations of algorithm HOL^1), all subproblems $\mathcal{P} = (B, R, x^p)$ in $\mathcal{L}$ are such that $B < F_{opt}$. It follows that for all $m \geqslant 2N+1$, $F_{opt} - f_{opt} \leqslant F^m_{opt} - f_{opt} < \delta$. We therefore have a contradiction with $F_{opt} - f_{opt} = \delta$ as the initial assumption $\delta > 0$ is hence proven to be wrong. As for all $k \in \mathbb{N}$, $F^k_{opt} \geqslant f^k_{opt}$, we have necessarily $\delta \geqslant 0$. It follows that $\delta = 0$. ∎

4. Multivariate Hölder Optimization

We are now considering the problem of maximizing a multivariate Hölder function over a hyperrectangle R of $\mathbb{R}^n$:

$$\begin{array}{ll} \max & f(x) \\ \text{subject to:} & x \in R \end{array}$$

where

$$R = [a, b] = [a_1, b_1] \times [a_2, b_2] \times \cdots \times [a_n, b_n] \subset \mathbb{R}^n,$$

and f satisfies the Hölder property:

$$\forall (x, y) \in R \times R \qquad |f(x) - f(y)| \leqslant L\|x - y\|^{1/\alpha},$$

with $\alpha \geqslant 1$.

The generalization of the algorithm designed for univariate Hölder optimization (see previous section) to the multivariate case appears to be very difficult. Indeed, the crucial step consisting of finding the maximum of the upper-bounding function $F^k(x) = \min_{j=1,2,\ldots,k}\{f(x^j) + L\|x - x^j\|^{1/\alpha}\}$ is again critical as it corresponds to the determination of the intersection of hypersurfaces which could be described as "parabolic cones". In multivariate Lipschitz optimization, it can be shown (see Mladineo [16]) that the intersection of two such hypersurfaces (which are cones of parallel axes) is included in a hyperplane. This allows the enumeration of all local maxima of the upper-bounding function, each of them requiring the solution of a system of n variables and $n+1$ equations (n linear and one quadratic) (see Mladineo [16] and Jaumard, Herrmann and Ribault [11] for an efficient implementation). In

the case of Hölder functions, the intersection of two hypersurfaces is no longer included in a hyperplane. Hence, there is no straightforward generalization of Piyavskii's algorithm to the case of Hölder functions. Other approaches have to be considered. The main feature of HOL^n, the algorithm proposed in this section, consists of a much more tractable upper-bounding function.

4.1. Relaxed upper-bounding function

Considering a set $X \subset \mathbb{R}^n$ and a point $x^j \in X$ where the Hölder function f is evaluated, the tightest upper-bounding for f is $f^j(x) = f(x^j) + L\|x - x^j\|^{1/\alpha}$. The tightest constant upper-bound for f is then the maximum of $f^j(x)$ over X. If X is a hyperrectangle $R = [a, b]$ and $x^j = \frac{a+b}{2} = c$ is the center of R, this constant upper-bound is:

$$B = \max_{x \in R} f^j(x) = f(c) + L\left(\frac{\|b-a\|}{2}\right)^{1/\alpha}.$$

Indeed, the maximum of $f^j(x)$ over R is reached at the most distant point from $x = c$, which is any extreme point e of the hyperrectangle. As c is the center of R, $\|e - c\|$ is half the length of the diagonal: $\|e - c\| = \frac{\|b-a\|}{2}$.

Assuming k points $(x^j)_{j=1,2,\ldots,k}$ are given in R, the tightest upper-bounding function for the Hölder function f is $F^k(x) = \min_{j=1,2,\ldots,k} f^j(x)$. Consider a partition of R in k hyperrectangles $(R^j)_{j=1,2,\ldots,k}$ and denote again by c^j the center of each hyperrectangle R^j. The tightest piecewise constant upper-bounding function based on this partition and denoted by F_2^k, is defined as:

$$\forall j = 1, 2, \ldots, k \quad \forall x \in R^j \quad F_2^k(x) = \overbrace{f(c^j) + L\left(\frac{\|b^j - a^j\|}{2}\right)^{1/\alpha}}^{B^j}.$$

The upper-bounding function is hence relaxed to constant pieces over hyperrectangles. We now provide the rules and tests for the branch-and-bound algorithm.

4.2. Rules and tests of algorithm HOL^n

The branch-and-bound rules defining the algorithm are the following:

Selection rule. The subproblem $\overline{\mathcal{P}} = (\overline{R}, \overline{B}) \in \mathcal{L}$ with the largest upper bound $\overline{B}$ is chosen (ties are broken arbitrarily). Let $\overline{R} = [\bar{a}, \bar{b}] = [\bar{a}_1, \bar{b}_1] \times [\bar{a}_2, \bar{b}_2] \times \cdots \times [\bar{a}_n, \bar{b}_n]$ be the corresponding hyperrectangle.

Branching rule. Let $\bar{l}_t = \max_{i=1,2,\ldots,n}(\bar{b}_i - \bar{a}_i)$ ($\bar{l}_t$ is the length of the longest edge of $\overline{R}$; ties are broken arbitrarily). Then $\overline{R}$ is partitioned into p hyperrectangles $\overline{R}^1; \overline{R}^2, \ldots, \overline{R}^p$ of the same dimension by cutting the t^{th} edge of $\overline{R}$ into p equal parts.

Initial and current lower bounding rule. The function f is evaluated at the center of R^0 and $\overline{R}^j$:

$$\begin{cases} D^0 = \{c^0\} = \left\{\frac{a+b}{2}\right\} \\ \bar{x}^j = \bar{c}^j = \frac{\bar{a}^j+\bar{b}^j}{2} \end{cases}$$

Initial and current upper bounding rule. The constant upper-bound is defined over $\overline{R}^j$ by:

$$\begin{cases} B^0 = f(c^0) + L\left(\frac{\|b-a\|}{2}\right)^{1/\alpha} \\ \overline{B}^j = \min\left\{\overline{B}, f(\bar{c}^j) + L\left(\frac{\|\bar{b}^j-\bar{a}^j\|}{2}\right)^{1/\alpha}\right\}. \end{cases}$$

Note that for some Hölder functions, we might have

$$f(\bar{c}^j) + L\left(\frac{\|\bar{b}^j - \bar{a}^j\|}{2}\right)^{1/\alpha} > \overline{B}.$$

As the current subproblem $\overline{\mathcal{P}}^j$ results from the branching step performed on subproblem $\overline{\mathcal{P}} = (\overline{R}, \overline{B}), \overline{B}$ is also a valid upper-bound for $\overline{\mathcal{P}}^j$. Note also that the definition of the upper-bounding function given above guarantees that the sequence of successive upper-bounds $\overline{B}$ is non-increasing.

4.3. Convergence of HOL^n

Assuming some conditions are satisfied, the proof of convergence proposed by Horst and Tuy [10] for a general branch-and-bound algorithm could then be applied to both algorithms HOL^1 and HOL^n. Indeed, it would suffice to prove that, according to the terminology of Horst and Tuy, the bounding rule is consistent and that the selection rule is bound improving. In order for the paper to be self-contained, we propose a specialized convergence proof.

Some conditions could be derived for the algorithm to be finite, i.e., to stop after a finite number of iterations when $\varepsilon = 0$. Such a situation occurs when, at a given step, the objective function coincides with the upper-bounding function, at least in a neighborhood of the global maximum $f^* = \max_{x \in R} f(x)$. Otherwise, the algorithm is asymptotically convergent.

Denote by $h = B - f(c)$ the height of any subproblem $\mathcal{P} = (R, B)$, where c is the center of R. As a preliminary result, we prove that the height decreases when a subproblem is considered by a branching process.

PROPOSITION 2. *Let $\mathcal{P}^j = (R^j, B^j)$ be one of the new subproblems obtained when branching on subproblem $\mathcal{P} = (R, B)$. Denoting by Δ the quantity $\left(1 - \frac{p^2-1}{np^2}\right)^{1/2\alpha}$, we have*

$$h^j \leqslant \Delta h \tag{13}$$

and $\Delta < 1$, *for any* $p > 1$.

Proof. If $R = [a, b]$, denote by $D = \|b-a\| = \left(\sum_{i=1}^{n} \ell_i^2\right)^{1/2}$, where $\ell_i = b_i - a_i$, the length of the diagonal of the hyperrectangle R. Suppose $\ell_m = \max_{i=1,2,\ldots,n}\{\ell_i\}$. The set R is divided into p hyperrectangles by partitioning its m-th edge into p equal parts. Hence, the diagonal of all resulting hyperrectangles are of equal length. Let R^j denote one of the resulting hyperrectangle and D^j the length of its diagonal. We have

$$(D^j) = \sum_{i=1}^{n} (\ell_i^j)^2 = \sum_{\substack{i=1 \\ i \neq m}}^{n} \ell_i^2 + \left(\frac{\ell_m}{p}\right)^2 = \sum_{i=1}^{n} \ell_i^2 - \ell_m^2 + \left(\frac{\ell_m}{p}\right)^2.$$

It follows that

$$\frac{(D^j)^2}{D^2} = 1 - \left(\frac{p^2 - 1}{p^2}\right) \frac{\ell_m^2}{D^2}.$$

Finally, as for all $i \neq m$, $\ell_i \leqslant \ell_m$ implies $D^2 = \sum_{i=1}^{n} \ell_i^2 \leqslant n\ell_m^2$, we have

$$\frac{(D^j)^2}{D^2} \leqslant 1 - \left(\frac{p^2 - 1}{np^2}\right).$$

Combining this last inequality with the definition of a subproblem height

$$h = B - f(c) = L\left(\frac{D}{2}\right)^{1/\alpha},$$

the inequality (13) holds. The result $\Delta < 1$ for $p > 1$ is immediate. ∎

As $p = 1$ would imply that the initial hyperrectangle is never partitioned, we assume that a value p greater than 1 is chosen. For the following convergence proof, we also denote f^k the function evaluation $f(\bar{c})$, where $\bar{c}$ is the center of the subproblem $\overline{P}$ on which the branching occurs at step k.

We now prove that if the algorithm is not finite, there is a subsequence of subproblems for which the gap between the upper and lower-bounds goes to zero when the number of steps goes to infinity.

PROPOSITION 3. *Either the algorithm is finite or there exists an infinite subsequence of unfathomed branching subproblems such that*

$$\lim_{q \to +\infty} (F_{opt}^{k_q} - f^{k_q}) = 0.$$

Proof. Assume that the algorithm does not stop after a finite number of steps. Then there necessarily exists an infinite sequence of nested subproblems on which branching occurs at steps $k_0, k_1, \ldots, k_q, \ldots$. By nested, we mean that each subproblem in the sequence results from the branching process performed on the

previous subproblem in the sequence. By successive applications of Proposition 1, we have

$$F_{opt}^{k_q} - f^{k_q} \leqslant \Delta(F_{opt}^{k_{q-1}} - f^{k_{q-1}}) \leqslant \cdots \leqslant \Delta^q(F_{opt}^{k_0} - f^{k_0}).$$

Note also that for any q, we have $F_{opt}^{k_q} - f^{k_q} \geqslant 0$. Finally, as Δ is smaller than 1 and $F_{opt}^{k_0} - f^{k_0} = B^0 - f(c^0) = \frac{L}{2}\|b - a\|^{1/\alpha}$ is a finite quantity, the result follows. ∎

We now prove the asymptotic convergence of algorithm HOL^n.

PROPOSITION 4. *Either the algorithm is finite or it asymptotically converges, that is*

$$\lim_{k\to+\infty} F_{opt}^k = \lim_{k\to+\infty} f_{opt}^k = f^*(= \max_{x\in R} f(x)).$$

Proof. It follows from the bounding rules proposed in the algorithm that

$$\forall k \quad f^k \leqslant f_{opt}^k \leqslant f^* \leqslant F_{opt}^k. \tag{14}$$

This implies

$$\forall k \quad 0 \leqslant F_{opt}^k - f_{opt}^k \leqslant F_{opt}^k - f^k. \tag{15}$$

It also follows from the bounding rule that $(F_{opt}^k)_{k\in\mathbb{N}}$ is a non-increasing sequence bounded from below by f^*. By construction, the sequence $(f_{opt}^k)_{k\in\mathbb{N}}$ is non-decreasing and bounded from above by f^*. These two sequences are thus convergent and so is the non-increasing sequence $(F_{opt}^k - f_{opt}^k)_{k\in\mathbb{N}}$. From Proposition 3, we known that

$$\lim_{q\to+\infty} (F_{opt}^{k_q} - f^{k_q}) = 0.$$

From (15), it then follows that

$$\lim_{q\to+\infty} (F_{opt}^{k_q} - f_{opt}^{k_q}) = 0.$$

Finally, as the sequence is known to be convergent and as there exists a subsequence converging to zero, by uniqueness of a limit of a convergent sequence, we have

$$\lim_{k\to+\infty} (F_{opt}^k - f_{opt}^k) = 0.$$

The result then immediately follows from (14). ∎

TABLE I. Univariate test functions for Hölder optimization

Function number	Hölder function $f(x)$	Interval $[a, b]$	Lipschitz constant L_1	Source
1.1	$-x^6 + 15x^4 - 27x^2 - 250$	[-4, 4]	2520	[13]
1.2	$(-x^2 + 5x - 6)/(x^2 + 1)$	[-5, 5]	6.5	[4]
1.3	$\begin{cases} -(x-2)^2 & \text{if } x \leq 3 \\ -2\ln(x-2) - 1 & \text{otherwise} \end{cases}$	[0, 6]	4	[22]
1.4	$\begin{cases} \sqrt{2x - x^2} & \text{if } x \leq 2 \\ \sqrt{-x^2 + 8x - 12} & \text{otherwise} \end{cases}$	[0, 6]	4	new
1.5	$(-3x + 1.4)\sin 18x$	[0, 1]	36	[2]
1.6	$-2(x-3)^2 - e^{\frac{x^2}{2}}$	[-3, 3]	85	[18]
1.7	$\sum_{k=1}^{5} k \sin[(k+1)x + k]$	[-10, 10]	67	[2]
1.8	$\sum_{k=1}^{5} k \cos[(k+1)x + k]$	[-10, 10]	67	[13]
1.9	$\max\left\{\sqrt{\frac{x}{2}}\sin(2x-1), \frac{1}{2}\exp(\frac{\cos(2x)}{2}\log x)\right\}$	[0, 10]	7.5	new
1.10	$\max\left\{\exp(\frac{\cos(2x)}{2}\log x), \exp(\frac{\sin(2x)}{2}\log x), 3 + \frac{2\sin(3x)}{x}\right\}$	[0, 10]	15.75	new

5. Computational Experiences

In this section, we report the results of computational experiences performed on twenty Hölder test functions. Half of them are univariate functions, whereas the other ones are functions of two variables. Most of these functions are test functions drawn from the Lipschitz optimization literature. Some additional test functions were defined to enrich the computational experiments. The expressions of the functions, the initial hyperrectangles [a,b], the Lipschitz constants and the references are given in Table I for the univariate functions and in Table III for the 2-variable functions. The original lipschitz constants (for the case $\alpha = 1$ and hence denoted L_1) that were not available in the literature were computed using a fine grid search on the norm of the gradient. For each other value of α, a lipschitz constant $L = L_1\|b - a\|^{(1-1/\alpha)}$ was computed in order to satisfy the Hölder condition. Indeed, as the lipschitz condition holds with Lipschitz constant L_1, we have:

$$\forall x, y \in R \qquad |f(x) - f(y)| \leqslant L_1\|x - y\|,$$

TABLE II. Optimal values and vectors for the univariate test functions

Function number	Optimal value f_{opt}	Precision ε	Optimal Point(s) x_{opt}
1.1	-7.0	2×10^{-5}	3.0
1.2	0.0355339	7×10^{-8}	2.414213
1.3	0.0	10^{-8}	2.0
1.4	2.0	10^{-8}	4.0
1.5	1.48907253	10^{-8}	0.966085
1.6	-7.5159241	3×10^{-7}	1.590717
1.7	12.03124944	7×10^{-8}	-6.774576,-0.49139,5.791785
1.8	14.5080079	2×10^{-7}	-7.083506,-0.8003,5.48286
1.9	1.94638464	10^{-8}	7.585057
1.10	9.43068151	10^{-8}	9.436578

TABLE III. 2-variable test functions for Hölder optimization

Function number	Hölder function $f(x, y)$	Interval $[a, b]$	Lipschitz constant L_1	Source
2.1	$\sin(x)\sin(xy)$	$[0,4] \times [0,4]$	4.299	new
2.2	$\sin(2x+y)/(\sin(y)+2)$	$[-5,5] \times [-5,5]$	2.237	new
2.3	$-\sin(x+y) - (x-y)^2 + 1.5x - 2.5y - 1$	$[-1.5,4] \times [-3,3]$	17.034	[15]
2.4	$\sin((x-1)(x-2)(y+1))$	$[-1,1] \times [-2,0]$	7.5	new
2.5	$-(x-2)^2 - (y-1)^2 - \frac{0.04}{-\frac{x^2}{4}-y^2+1} - \frac{(x-2y+1)^2}{0.2}$	$[1,2] \times [1,2]$	47.426	[21]
2.6	$-\left(y - \frac{5x^2}{4\pi^2} + \frac{5x}{\pi} - 6\right)^2 - 10(1 - \frac{1}{8\pi})\cos x - 10$	$[-5,10] \times [0,15]$	112.44	[3]
2.7	$-100(y-x^2)^2 - (x-1)^2$	$[-3,3] \times [-1.5,4.5]$	12781.7	[21]
2.8	$-0.1\left[12 + x^2 + \frac{(1+y^2)}{x^2} + \frac{x^2y^2+100}{x^4y^4}\right]$	$[1,3] \times [1,3]$	56.852	[21]
2.9	$-\frac{1}{2}\sum_{i=1}^{2} x_i^2 + \prod_{i=1}^{2} \cos(10\ln((i+1)x_i)) - 1$	$[0.01,1] \times [0.01,1]$	988.82	[17]
2.10	$-[1+(x+y+1)^2(19-14x+3x^2-14y+6xy+3y^2)]$ $\times [30+(2x-3y)^2(18-32x+12x^2+48y-36xy+27y^2)]$	$[-2,2] \times [-2,2]$	2 225 892	[6]

and clearly

$$L_1\|x-y\| = L_1\|x-y\|^{(1-1/\alpha)}\|x-y\|^{1/\alpha} \leqslant \overbrace{L_1\|b-a\|^{(1-1/\alpha)}}^{L}\|x-y\|^{1/\alpha}.$$

The optimal values and vectors are given respectively in Tables II and IV together with the best precision for which the optimal values were obtained.

Both the univariate and multivariate Hölder algorithms have been implemented in C and run on a SUN Sparc station (135.5 mips, 65.2 Mflops, 128 Meg). A double-ended priority queue or maxminheap ([12]) is used to store the subproblems. In the multivariate algorithm HOL^n, the hyperrectangles are partitioned in three ($p = 3$).

TABLE IV. Optimal values and vectors for the 2-variable test functions

Function number	Optimal value f_{opt}	Precision ε	Optimal Vector x_1	x_2
2.1	1.00000	5×10^{-5}	1.57079713	0.99999435
2.2	1.0000	2×10^{-4}	1.57078584	-1.57078584
2.3	1.91322295	10^{-8}	-0.54719386	-1.54720091
2.4	1.000	2×10^{-3}	-0.66087486	-0.64456637
2.5	-0.16904267	10^{-8}	1.79541003	1.37786415
2.6	-0.39	2×10^{-2}	3.14158580	2.25003810
2.7	-0.00000570	3×10^{-6}	-1.00045725	1.00068587
2.8	-1.74	9×10^{-3}	1.74333181	2.02987349
2.9	0.00017018	10^{-4}	0.01152703	0.01440453
2.10	-3	5×10^{4}	0.00000000	-1.00045725

It has been shown that for the Lipschitz version ($\alpha = 1$) of the multivariate case (see [5]), an odd value seems to be a better choice for p: indeed, in this case, some function evaluations can be reused at each step. Computational experience have shown that $p = 3$ is a good compromise for both Lipschitz and Hölder optimization (see Gourdin, Hansen and Jaumard [5]).

A normalization is used in order to counterbalance the variations in the Lipschitz constant, the initial rectangle and the range of f from one test function to another. This normalization involves a "worst-case" theoretical algorithm called the *passive algorithm*. The passive algorithm covers the initial hyperrectangle R with hypercubes having sides of length d. The function is evaluated at the center of each hypercube. The difference between the upper and the lower-bounds in each hypercube is given by the height $h = L\left(\frac{\sqrt{n}d}{2}\right)^{1/\alpha}$ of the corresponding subproblem. The length d is computed in order to guarantee an ε-optimal value:

$$h \leqslant \varepsilon \Longleftrightarrow d \leqslant \frac{2}{\sqrt{n}} \left(\frac{\varepsilon}{L}\right)^{\alpha}.$$

The minimal number of function evaluation needed by the passive algorithm in order to satisfy the above condition is then:

$$N_{pass} = \prod_{i=1}^{n} \left(\left\lfloor \frac{b_i - a_i}{d} \right\rfloor + 1 \right) = \prod_{i=1}^{n} \left(\left\lfloor \frac{(b_i - a_i)\sqrt{n}L^{\alpha}}{2\varepsilon^{\alpha}} \right\rfloor + 1 \right),$$

which is approximated by

$$N_{pass} \simeq \prod_{i=1}^{n} \left(\frac{(b_i - a_i)\sqrt{n}L^{\alpha}}{2\varepsilon^{\alpha}} \right) = \left(\frac{\sqrt{n}L^{\alpha}}{2\varepsilon^{\alpha}} \right)^{n} \prod_{i=1}^{n} (b_i - a_i).$$

TABLE V. Computational results for univariate functions with $N_{pass} = 10^4$

Function	$1/\alpha$	ε	N_{best}	N_{hol}/N_{best}	N_{pass}/N_{best}	N_{pass}/N_{hol}	t_{cpu}
1.1	1.00	1.01e+00	394	1.39	25.38	18.21	0.04
	0.75	1.20e+01	1021	1.42	9.79	6.92	0.97
	0.50	1.43e+02	2915	1.55	3.43	2.21	0.42
1.2	1.00	3.25e-03	462	1.35	21.65	16.05	0.04
	0.75	3.86e-02	1173	1.38	8.53	6.20	1.03
	0.50	4.60e-01	2657	1.48	3.76	2.54	0.34
1.3	1.00	1.20e-03	182	1.49	54.95	36.90	0.01
	0.75	1.43e-02	449	1.42	22.27	15.70	0.41
	0.50	1.70e-01	1089	1.44	9.18	6.39	0.13
1.4	1.00	1.20e-03	357	1.35	28.01	20.70	0.03
	0.75	1.43e-02	879	1.32	11.38	8.65	0.71
	0.50	1.70e-01	2096	1.33	4.77	3.59	0.25
1.5	1.00	1.80e-03	96	1.43	104.17	72.99	0
	0.75	2.14e-02	247	1.28	40.49	31.55	0.15
	0.50	2.55e-01	766	1.42	13.05	9.20	0.08
1.6	1.00	2.55e-02	289	1.47	34.60	23.53	0.03
	0.75	3.03e-01	715	1.43	13.99	9.81	0.61
	0.50	3.61e+00	1741	1.44	5.74	4.00	0.22
1.7	1.00	6.70e-02	147	1.31	68.03	51.81	0.01
	0.75	7.97e-01	496	1.44	20.16	14.03	0.67
	0.50	9.48e+00	2401	1.48	4.16	2.81	0.4
1.8	1.00	6.70e-02	129	1.40	77.52	55.25	0.01
	0.75	7.97e-01	402	1.35	24.88	18.38	0.51
	0.50	9.48e+00	1916	1.40	5.22	3.72	0.29
1.9	1.00	3.75e-03	129	1.37	77.52	56.50	0.02
	0.75	4.46e-02	370	1.39	27.03	19.38	0.41
	0.50	5.30e-01	1593	1.39	6.28	4.50	0.25
1.10	1.00	7.87e-03	60	1.35	166.67	123.46	0.01
	0.75	9.36e-02	171	1.46	58.48	40.16	0.2
	0.50	1.11e+00	686	1.36	14.58	10.72	0.11

The normalization consists in choosing for each test function, the same number of function evaluations required by the passive algorithm. A tolerance is then computed for each test function:

$$\varepsilon = L\left(\frac{\sqrt{n}}{2}\right)^{1/\alpha}\left(\frac{\prod_{i=1}^{n}(b_i - a_i)}{N_{pass}}\right)^{\frac{1}{\alpha n}}.$$

The performances of both algorithms are measured in terms of N_{hol}, the number of function evaluations (including those performed during the line search) and t_{cpu}, the total processing time (in seconds). Three values of $1/\alpha$ (1, 0.75 and 0.5) are used for each test problem and all computations performed with $N_{pass} = 10^4$. The results are given in Table V for the univariate functions, and in Table VI for the functions of two variables. Detailed results with various values of α are provided in Tables VII and VIII, for the first univariate function and the first function of 2 variables respectively. The number of function evaluations (N_{hol})

TABLE VI. Computational results for the 2-variable functions with $N_{pass} = 10^4$

Function	$1/\alpha$	ε	N_{hol}	N_{pass}/N_{hol}	t_{cpu}
2.1	1.00	1.22e-01	1013	9.87	0.11
	0.75	4.57e-01	2527	3.96	0.27
	0.50	1.72e+00	6101	1.64	0.66
2.2	1.00	1.58e-01	1089	9.18	0.08
	0.75	5.95e-01	2939	3.40	0.27
	0.50	2.24e+00	6775	1.48	0.65
2.3	1.00	6.92e-01	1123	8.90	0.1
	0.75	2.60e+00	2333	4.29	0.2
	0.50	9.79e+00	5215	1.92	0.47
2.4	1.00	1.06e-01	1821	5.49	0.16
	0.75	3.99e-01	3123	3.20	0.26
	0.50	1.50e+00	5541	1.80	0.48
2.5	1.00	3.35e-01	2673	3.74	0.22
	0.75	1.26e+00	4591	2.18	0.38
	0.50	4.74e+00	7355	1.36	0.63
2.6	1.00	1.19e+01	3519	2.84	0.31
	0.75	4.48e+01	6133	1.63	0.57
	0.50	1.69e+02	9735	1.03	0.92
2.7	1.00	5.42e+02	7967	1.26	0.69
	0.75	2.04e+03	11249	0.89	1
	0.50	7.67e+03	14301	0.70	1.31
2.8	1.00	8.04e-01	12643	0.79	1.17
	0.75	3.02e+00	18253	0.55	1.71
	0.50	1.14e+01	19133	0.52	1.8
2.9	1.00	6.92e+00	15687	0.64	1.73
	0.75	2.60e+01	19683	0.51	2.23
	0.50	9.79e+01	19683	0.51	2.18
2.10	1.00	6.30e+04	13945	0.72	1.26
	0.75	2.37e+05	16139	0.62	1.44
	0.50	8.90e+05	17771	0.56	1.64

of both algorithms are compared with those of the passive algorithm (N_{pass}). The univariate Hölder algorithm HOL^1 is also compared with a *best possible algorithm.*

A best possible algorithm is a theoretical algorithm which requires the minimum number of function evaluations to provide an ε-optimal value. It uses a best upper-bounding function, which for a given ε, is an upper-bounding function of maximum height lower than $f^* + \varepsilon$ (where f^* is the a priori unknown maximum value of $f(x)$ over R). In the case of univariate functions, the construction for Lipschitz functions (see Hansen, Jaumard and Lu [8]) can easily be generalized. Assuming f^* is known, the best upper-bound function is a saw-tooth cover bouncing between $f(x)$ and $f^* + \varepsilon$ (see Figure 3). Note that the number of function evaluations required by two such upper-bounding functions may differ by 1 due to side effects. A best upper-bounding function is build and the number of function evaluations required is denoted by N_{best}. This value is used as a reference for N_{hol}. Note that

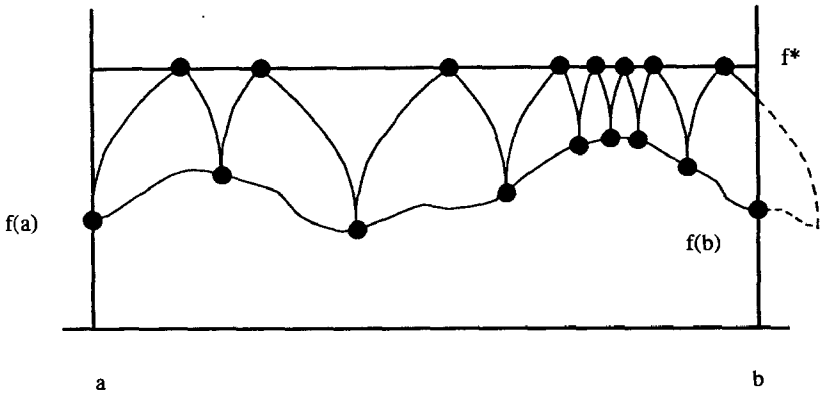

Fig. 3. Illustration of a best upper-bounding function.

it is very difficult to generalize the construction of a best upper-bounding function to the case of multivariate functions. No comparison with N_{best} is thus provided for algorithm HOL^n.

We observe in Table V that the performances of algorithm HOL^1 are close to those of the best possible algorithm. Indeed, N_{hol} is on the average only 1.4 larger than N_{best}, which is the same as the ratio between Piyavskii's algorithm and a best possible algorithm in the case of Lipschitz optimization. We observe however that the number of function evaluations N_{best} and N_{hol} increase with α. This last feature appears even more clearly in Table VII. Algorithm HOL^1 is also much more efficient than the passive algorithm, which requires many function evaluations, especially for small values of α. The computing time required by HOL^1 (which appears in the last column) are smaller for $\alpha = 1$ and $\alpha = 2$. Indeed, for these values, the analytical expressions providing the peak point (see appendix) are used, and the computing times are thus considerably reduced. For $\alpha = 4/3$, the peak point is obtained at each step, by solving an equation with the Fibonacci line search method and hence requires more computing time.

For 2-variable functions, the difference between HOL^n and the passive algorithm is not as significant as for the univariate case. Indeed, we observe in Table VI that for the last three test functions, algorithm HOL^n requires even more function evaluations than the passive algorithm. This feature was already observed when testing multivariate Lipschitz algorithms (see [5], [7]). It seems that the last three test functions are particularly badly shaped for Lipschitz and Hölder optimization (one has many local maxima, another is very flat except for a narrow region where the slope increases exponentially). This is also confirmed by the fact that these functions require much more computing time than the others.

In Table VII, it appears more explicitly than in Table V that N_{best}, the number of function evaluation required to build the best upper-bounding function, increases with α. This behaviour can be explained by the fact that when the intervals becomes very narrow, each single "tooth" of the upper-bounding function is much higher for

TABLE VII. Performance of HOL^1 on Function 1.1 for several α

$1/\alpha$	ε	N_{best}	N_{hol}/N_{best}	N_{pass}/N_{best}	N_{pass}/N_{hol}	t_{cpu}
1.00	1.01e+00	394	1.39	25.38	18.21	0.04
0.95	1.65e+00	475	1.35	21.05	15.55	0.44
0.90	2.71e+00	572	1.35	17.48	12.94	0.55
0.85	4.45e+00	692	1.37	14.45	10.54	0.65
0.80	7.31e+00	839	1.36	11.92	8.76	0.81
0.75	1.20e+01	1021	1.42	9.79	6.92	1.06
0.70	1.97e+01	1247	1.37	8.02	5.84	1.21
0.65	3.23e+01	1531	1.39	6.53	4.68	1.49
0.60	5.30e+01	1891	1.43	5.29	3.70	1.84
0.55	8.69e+01	2346	1.40	4.26	3.05	2.23
0.50	1.43e+02	2915	1.49	3.43	2.30	2.93

TABLE VIII. Performances of HOL^n on Function 2.1 for several α

$1/\alpha$	ε	N_{hol}	N_{pass}/N_{hol}	t_{cpu}
1.00	1.22e-01	1013	9.87	0.11
0.95	1.58e-01	1247	8.02	0.13
0.90	2.07e-01	1475	6.78	0.15
0.85	2.69e-01	1765	5.67	0.19
0.80	3.51e-01	2121	4.71	0.22
0.75	4.57e-01	2527	3.96	0.26
0.70	5.96e-01	3045	3.28	0.33
0.65	7.77e-01	3753	2.66	0.46
0.60	1.01e+00	4369	2.29	0.5
0.55	1.32e+00	5313	1.88	0.64
0.50	1.72e+00	6101	1.64	0.71

large values of α. The table also seems to confirm that HOL^1 takes only 1.4 more function evaluations than the best possible algorithm. As expected, the computing time required by HOL^1 is proportional to N_{hol}, when no analytical expression are available for the peak point, i.e., for all values of α except 1 and 2. For these two values, the computing times are much smaller.

The observation that the number of function evaluations increases with α does not seem to extend to the case of the 2-variable functions. Indeed, it appears in Table VIII that N_{hol} remains approximately constant, and so the computing times.

6. Acknowledgements

Research of the second author was supported by NSERC (Natural Sciences and Engineering Research Council of Canada) grant #GP0036426, FCAR (Fonds

pour la formation de Chercheurs et l'Aide à la Recherche) grants 92EQ1048 and 95ER1048, and a FRSQ (Fonds pour la Recherche en Santé du Québec) fellowship.

7. Appendix

Determination of the peak point when $\alpha = 2, \alpha = 3$ *and* $\alpha = 4$.
We assume that the function f satisfies a Hölder condition with constants L and α. Let C_a^α and C_b^α be the two curves defined by

$$C_a^\alpha \begin{cases} y = f(a) + L\sqrt[\alpha]{x-a} \\ x \geqslant a, \end{cases}$$

and

$$C_b^\alpha \begin{cases} y = f(b) + L\sqrt[\alpha]{b-x} \\ x \leqslant b. \end{cases}$$

It has been shown in Section 3.1 that there is a unique intersection point between the two curves. We now propose some additional results in order to obtain an explicit expression for the intersection point coordinates for some cases where $\alpha \in \mathbb{N}$.

PROPOSITION 5. *The intersection of the two curves* C_a^α *and* C_b^α *satisfies the following system of equations:*

$$\begin{cases} \displaystyle\sum_{i=0}^{\alpha/2} \binom{\alpha}{2i} \left(y - \frac{f(a)+f(b)}{2}\right)^{\alpha-2i} \left(\frac{f(b)-f(a)}{2}\right)^{2i} = \frac{L^\alpha(b-1)}{2} \\ \displaystyle\sum_{i=0}^{\alpha/2} \binom{\alpha}{2i+1} \left(y - \frac{f(a)+f(b)}{2}\right)^{\alpha-2i-1} \left(\frac{f(b)-f(a)}{2}\right)^{2i+1} = \frac{L^\alpha(2x-a-b)}{2} \end{cases}$$

Proof. The two curves C_a^α and C_b^α can be equivalently defined as follows:

$$C_a^\alpha \begin{cases} (y-f(a))^\alpha = L^\alpha(x-a) \\ y \geqslant f(a) \end{cases}, \quad C_b^\alpha \begin{cases} (y-f(b))^\alpha = L^\alpha(b-x) \\ y \geqslant f(b) \end{cases},$$

Hence, the intersection point satisfies:

$$C_a^\alpha \cap C_b^\alpha \begin{cases} (y-f(a))^\alpha = L^\alpha(x-a), \\ (y-f(b))^\alpha = L^\alpha(b-x), \\ y \geqslant \max\{f(a), f(b)\}. \end{cases} \tag{16}$$

Defining the new variable

$$Y + \left(y - \frac{f(a)+f(b)}{2}\right), \tag{17}$$

and denoting

$$\gamma = \left(\frac{f(b) - f(a)}{2}\right), \tag{18}$$

the intersection between the curves C_a^α and C_b^α can be restated in a system involving the variables x and Y:

$$C_a^\alpha \cap C_b^\alpha \begin{cases} (Y+\gamma)^\alpha = L^\alpha(x-a), \\ (Y-\gamma)^\alpha = L^\alpha(b-x), \\ Y \geqslant |\gamma|. \end{cases} \tag{19}$$

Adding the two first equations of (19), we obtain:

$$(Y+\gamma)^\alpha + (Y-\gamma)^\alpha = L^\alpha(b-a).$$

Expanding the left-hand-side using the Binomial formula, this equation can be successively rewritten as follows:

$$\begin{aligned} \sum_{k=0}^{\alpha} \binom{\alpha}{k} Y^{\alpha-k}\gamma^k + \sum_{k=0}^{\alpha} \binom{\alpha}{k} Y^{\alpha-k}(-1)^k\gamma^k &= L^\alpha(b-a), \\ \sum_{k=0}^{\alpha} \binom{\alpha}{k} Y^{\alpha-k}(1+(-1)^k)\gamma^k &= L^\alpha(b-a), \\ 2\sum_{i=0}^{\alpha/2} \binom{\alpha}{2i} Y^{\alpha-2i}\gamma^{2i} &= L^\alpha(b-a), \end{aligned}$$

as $(1+(-1)^k)$ equals 2 when $k = 2i$ (is even), and 0 otherwise. Replacing Y and γ by their respective expression (17) and (18), the first equation of Proposition 5 is obtained.

Similarly, substracting the two first equations of (19) yields:

$$(Y+\gamma)^\alpha - (Y-\gamma)^\alpha = L^\alpha(2x-a-b),$$

which can be successively restated in

$$\begin{aligned} \sum_{k=0}^{\alpha} \binom{\alpha}{k} Y^{\alpha-k}(1-(-1)^k)\gamma^k &= L^\alpha(2x-a-b), \\ 2\sum_{i=0}^{\alpha/2} \binom{\alpha}{2i+1} Y^{\alpha-2i-1}\gamma^{2i+1} &= L^\alpha(2x-a-b), \end{aligned}$$

as $(1-(-1)^k)$ equals 2 when $k = 2i+1$ (is odd), and 0 otherwise. The second equation of Proposition 5 is then also obtained. ∎

Using the previous result and notations, the intersection point between the two curves C_a^α and C_b^α can be obtained according to the following lemma:

LEMMA 1. *Let $\tilde{Y}$ be the unique solution of the α-th degree equation:*

$$\sum_{i=0}^{\alpha/2} \binom{\alpha}{2i} Y^{\alpha-2i}\gamma^{2i} = \frac{L^\alpha(b-a)}{2}, \tag{20}$$

such that

$$\tilde{Y} \geqslant |\gamma|.$$

The coordinates $(\tilde{x}, \tilde{y})$ of the unique intersection point between the curves C_a^α and C_b^α are then given by:

$$C_a^\alpha \cap C_b^\alpha \begin{cases} \tilde{x} = \dfrac{a+b}{2} + (1/L^\alpha) \displaystyle\sum_{i=0}^{\alpha/2} \binom{\alpha}{2i+1} \tilde{Y}^{\alpha-2i-1}\gamma^{2i+1}, \\ \tilde{y} = \dfrac{f(a)+f(b)}{2} + \tilde{Y}. \end{cases}$$

The peak point coordinates are now successively provided for the cases $\alpha = 2, 3$ and 4.

PROPOSITION 6. *Assuming the function f satisfies a Hölder condition with $\alpha = 2$, the intersection point $(\tilde{x}, \tilde{y})$ between the curves C_a^2 and C_b^2 is given by:*

$$C_a^2 \cap C_b^2 \begin{cases} \tilde{x} = \dfrac{a+b}{2} + \left(\dfrac{f(b)-f(a)}{2L^2}\right)\sqrt{2L^2(b-a) - (f(a)-f(b))^2} \\ \tilde{y} = \dfrac{f(a)+f(b)}{2} + \dfrac{1}{2}\sqrt{2L^2(b-a) - (f(a)-f(b))^2}. \end{cases}$$

PROPOSITION 7. *Assuming the function f satisfies a Hölder condition with $\alpha = 3$, the intersection point $(\tilde{x}, \tilde{y})$ between the curves C_a^3 and C_b^3 is given by:*

$$\begin{cases} \tilde{x} = \dfrac{a+b}{2} + \left(\dfrac{f(b)-f(a)}{8L^3}\right)(3A^{2/3} + 3B^{2/3} - 5(f(a)-f(b))^2) \\ \tilde{y} = \dfrac{f(a)+f(b)}{2} + \dfrac{\sqrt[3]{A} + \sqrt[3]{B}}{2} \end{cases}$$

where

$$\begin{cases} A = 2L^3(b-a) + \sqrt{(f(a)-f(b))^6 + 4L^6(b-a)^2} \\ B = 2L^3(b-a) - \sqrt{(f(a)-f(b))^6 + 4L^6(b-a)^2}. \end{cases}$$

PROPOSITION 8. *Assuming the function f satisfies a Hölder condition with $\alpha = 4$, the intersection point $(\tilde{x}, \tilde{y})$ between the curves C_a^4 and C_b^4 is given by:*

$$\begin{cases} \tilde{x} = \dfrac{a+b}{2} + \left(\dfrac{4\gamma}{L^4}\right)\left(\sqrt{\dfrac{L^4(b-a)}{2} + 8\gamma^4} - 2\gamma^2\right)\sqrt{\sqrt{\dfrac{L^4(b-a)}{2} + 8\gamma^4} - 3\gamma^2}, \\ \tilde{y} = \dfrac{f(a)+f(b)}{2} + \sqrt{\sqrt{\dfrac{L^4(b-a)}{2} + 8\gamma^4} - 3\gamma^2}, \end{cases}$$

where

$$\gamma = \frac{f(b) - f(a)}{2}.$$

References

1. Atkinson, M.D., Sack, J.-R., Santoro, N., and Strotholte, T. (1986), Min-max Heaps and Generalized Priority Queues, *Communications of the ACM* **29**, 996–1000.
2. Basso, P. (1982), Iterative Method for the Localization of the Global Maximum, *SIAM Journal on Numerical Analysis* **19**, 781–792.
3. Branin, F.H. Jr. (1972), Widely Convergent Method for Multiple Solution of Simultaneous Nonlinear Equations, *IBM Journal of Research and Development* **16**, 504–522.
4. Fichtenholz, G.M. (1964), *Differential- und Integralrechnung I*, Berlin.
5. Gourdin, E., Hansen, P., and Jaumard, B. (1996) 'Multivariate Lipschitz Optimization: a Survey', submitted for publication.
6. Goldstein, A.A. and Price, J.-F. (1971), On Descent from Local Minima, *Mathematics of Computation* **25**, 569–574.
7. Hansen, P. and Jaumard, B. (1994), Lipschitz Optimization, in *Handbook of Global Optimization* (R. Horst and P. Pardalos, eds.), Kluwer, The Netherlands, pp. 407–494.
8. Hansen, P., Jaumard, B., and Lu, S.-H. (1991), On the Number of Iterations of Piyavskii's Global Optimization Algorithm, *Mathematics of Operations Research* **16**, 334–350.
9. Hanjoul, P., Hansen, P., Peeters, D., and Thisse, J.-F. (1990), Uncapacitated Plant Location under Alternative Space Price Policies, *Management Science* **36**, 41–47.
10. Horst, R. and Tuy, H. (1992), *Global Optimization – Deterministic Approaches*, 2nd Edition, Berlin: Springer.
11. Jaumard, B., Ribault, H., and Herrmann, T. (1995), An On-line Cone Intersection Algorithm for Global Optimization of Multivariate Lipschitz Functions, to appear in *Mathematical Programming*.
12. Jaumard, G. and Gourdin, E. (1992), Techniques Informatiques pour la Recherche Opérationnelle: Partie I, *Cahiers du Gerard*, G-92-43, Montréal, Canada.
13. Levy, A.V., Montalvo, A., Gomez, S., and Calderon, A. (1981), Topics in Global Optimization, in *Lecture Notes on Mathematics* **909**, Springer, Berlin.
14. Luenberger, D.G. (1984), *Linear and Nonlinear Programming*, 2nd Edition, Addison-Wesley, Reading, Massachusetts.
15. McCormick, G.P. (1976), Computability of Global Solutions to Factorable Non-convex Programs: Part 1 – Convex Understanding Problems, *Mathematical Programming* **10**, 147–175.
16. Mladineo, R.H. (1986), An Algorithm for Finding the Global Maximum of a Multimodal, Multivariate Function, *Mathematical Programming* **34**, 188–200.
17. Mladineo, R.H. (1992), Stochastic Minimization of Lipschitz Functions, in *Recent Advances in Global Optimization*, C. Floudas and P. Pardalos, eds., Princeton University Press, pp. 369–383.
18. Phillips, D.T., Ravindran, A., and Solberg, J.J. (1976), *Operations Research Principles and Practice*, Wiley, New York.
19. Piyavskii, S.A. (1967), An Algorithm for Finding the Absolute Minimum of a Function, *Theory of Optimal Solutions*, **2**, Kiev, IK AN USSR (in Russian), pp. 13–24.
20. Piyavskii, S.A. (1972), An Algorithm for Finding the Absolute Extremum of a Function, *USSR Computational Mathematics and Mathematical Physics* **12**, 57–67 (*Zh. vychisl Mat. mat. Fiz.* 12(4) (1972) 888–896).
21. Schittkowski, K. (1987), *More Test Examples for Nonlinear Programming Codes*, Lecture Notes in Economics and Mathematical Systems **282**, Springer Verlag, Berlin, Heidelberg, New York.
22. Timonov, L.N. (1977), An Algorithm for Search of a Global Extremum, *Engineering Cybernetics* **15**, 38–44.