Optimal algorithms for global optimization in case of unknown Lipschitz constant

Matthias Horn

Mathematisches Institut, Universität Jena, D - 07737 Jena

Abstract

We consider the global optimization problem for d-variate Lipschitz functions which, in a certain sense, do not increase too slowly in a neighborhood of the global minimizer(s). On these functions, we apply optimization algorithms which use only function values. We propose two adaptive deterministic methods. The first one applies in a situation when the Lipschitz constant L is known. The second one applies if L is unknown. We show that for an optimal method, adaptiveness is necessary and that randomization (Monte-Carlo) yields no further advantage. Both algorithms presented have the optimal rate of convergence.

Key words: Global optimization, Lipschitz functions, complexity, optimal rate of convergence

MSC: primary 90C60, 90C56; secondary 68Q25, 26B35

1 Introduction

For a continuous function

$$f : [0,1]^d \to \mathbb{R}$$

we are looking for a point

$$x_* \in [0,1]^d \tag{1}$$

such that $f(x_*)$ is close to $\inf f$. Continuity is already sufficient to guarantee that the second algorithm we propose converges. However, for cost estimation and optimality results we make two further assumptions on f:

Email address: `mhorn@mathe.uni.jena.de` (Matthias Horn).

Preprint submitted to Elsevier Science *27 January 2005*

(1) The function f is Lipschitz with constant $L > 0$:

$$\forall x, y \in [0,1]^d \qquad |f(x) - f(y)| \leq L\|x - y\|_\infty. \tag{2}$$

(2) For the level sets

$$A(f,\delta) := \{x \in [0,1]^d : \, f(x) \leq \inf f + \delta\} \tag{3}$$

and constants $\varrho, D > 0$ we have

$$\forall \delta \leq \varrho \qquad \lambda^d(A(f,\varrho)) \leq D\,\delta^{d/2}. \tag{4}$$

Here, λ^d denotes Lebesgue measure on $[0,1]^d$. We say

$$f \in F^d_{L,D,\varrho}.$$

For the constants L, D, ϱ we assume

$$\varrho < \min\{L, \tfrac{1}{4}L^2 D^{2/d}\}. \tag{5}$$

This way, we have $F^d_{L,D,\varrho} \neq \emptyset$, and for every point $x \in [0,1]^d$, the class $F^d_{L,D,\varrho}$ contains several functions f with $f(x) = \min f$.

A function $f \in F^d_{L,D,\varrho}$ may have many global minimizers.

Assumption (4) guarantees a minimum increase in a neighborhood of the global minimizer or, if there are several global minimizers, in a neighborhood of each of them. The upper bound $D\,\delta^{d/2}$ for the level sets $A(f,\delta)$ allows that for

$$f \in C^2[0,1]^d$$

with a finite number of minimizers x^* and a positive Hessian $(\nabla^2 f)(x^*)$ for each of them, we can always find parameters L, D, ϱ such that $f \in F^d_{L,D,\varrho}$. This is a consequence of Taylor expansion.

In order to find an x_* with $f(x_*)$ close to $\inf f$ we want to consider numerical methods which use only function values which are chosen at sequentially or adaptively chosen knots $x^1, \ldots, x^n$. We are interested in the interdependence of the error $f(x_*) - \inf f$ and the effort we spent to reach this error level. We define $\mathrm{cost}(A, F^d_{L,D,\varrho})$ to be the number of oracle calls a method A needs for $f \in F^d_{L,D,\varrho}$ in the worst case and $\Delta(A, F^d_{L,D,\varrho})$ to be the worst case error of A.

The error numbers

$$e_n(F^d_{L,D,\varrho}) := \inf\{\Delta(A, F^d_{L,D,\varrho}) : \, A : \, \mathrm{cost}(A, F^d_{L,D,\varrho}) \leq n\} \tag{6}$$

give information about the intrinsic difficulty of the optimization problem. Any method yielding an error of at most $e_n(F^d_{L,D,\varrho})$ for all $f \in F^d_{L,D,\varrho}$ uses at least n function calls for at least one function $f \in F^d_{L,D,\varrho}$. The error numbers are benchmark for our algorithms.

In Section 2, we introduce two deterministic adaptive optimization algorithms for the class $F^d_{L,D,\varrho}$. The first one is applicable if the Lipschitz constant L or an upper bound $L' \geq L$ is known. The second one needs no information about L. Both algorithms work without knowledge of the other parameters D, ϱ or any other parameter one might know or guess. We prove upper bounds of error and cost for both methods.

In Section 3, we prove $e_n(F^d_{L,D,\varrho}) \asymp L^2 D^{2/d}\, n^{-2/d}$ and see that both algorithms of Section 2 have the optimal rate of convergence. Furthermore, wee see that compared to optimal non-adaptive methods, adaptiveness yields a quadratic speed-up while randomization (Monte-Carlo methods) cannot improve the rate of convergence any further.

For illustration, we apply the second algorithm to a test function. See Section 4.

1.1 Some known complexity results

A well examined optimization problem is convex programming. Here, we have a unique global minimizer and local and global search coincide. For this situation, methods are known whose cost behave polynomial in dimension d and error level ε. The ellipsoid method yields an approximation to the error level $\varepsilon > 0$ using $\mathcal{O}(d^2 \ln(1/\varepsilon))$ calls of f and ∇f. For details and further complexity results for convex functions we refer to the mini-course of Nemirovski (1995).

The problem class $F^d_{L,D,\varrho}$ allows many global minimizers. In this property, it is closely related to Lipschitz optimization. For classes

$$F_L := \{f : [0,1]^d \to \mathbb{R},\ |f(x) - f(y)| \leq L\,\|x - y\| \text{ for all } x \in [0,1]^d\},$$

we have that a non-adaptive method using equidistant points delivers the optimal result. The class F_L is a special case of a convex and symmetric (i.e. $f \in F \Rightarrow -f \in F$) class. In this situation, adaptiveness can, if at all, yield only a minor improvement compared to best non-adaptive algorithms. See Novak (1988), prop.1.3.2 for details. For F_L, we have $e_n(F_L) \asymp Ln^{-1/d}$. In contrast, we will see for the class $F^d_{L,D,\varrho}$ that adaption is essential for optimality and that it leads to a quadratic speed-up: $e_n(F^d_{L,D,\varrho}) \asymp L^2 D^{2/d} n^{-2/d}$. In both cases, however, the error even of optimal methods depends exponentially in d, which means that for F_L and $F^d_{L,D,\varrho}$, the optimization problem is not tractable.

One may say that in many situations the worst case definition of error or cost is too pessimistic and that the average case may give a more realistic picture. A prominent model for $d = 1$ is to assume the Wiener measure on $C[0, 1]$. Even for this case we have only little knowledge of the complexity. Ritter (1990) (Theorem 4) shows that the mean error of best non-adaptive methods using n function calls behaves like $1/\sqrt{n}$. Calvin (2004) uses sophisticated methods to show that for best adaptive methods, the error cannot decrease exponentially.

Remark 1 *We compare our results with those of two recent papers.* Perevozchikov *(1990) defines a class similar to $F^d_{L,D,\varrho}$. For*

$$\widetilde{F}^d_{L,r,\varrho} := \{f : [0,1]^d \to \mathbb{R}, |f(x) - f(y)| \le L\|x - y\|,\ \lambda^d(A(\delta)) \le \delta^r \text{ for } \delta \le \varrho\}$$

he develops an algorithm which yields the upper bound

$$e_n(\widetilde{F}^d_{L,r,\varrho}) \le \begin{cases} \mathcal{O}(n^{-1/d(1-r)}), & r < 1, \\ \mathcal{O}(e^{-n}), & r = 1. \end{cases}$$

Lower bounds for $e_n(\widetilde{F}^d_{L,r,\varrho})$ are missing. Furthermore, the method of Perevozchikov assumes a fixed, i.e. known Lipschitz constant.

Like our second algorithm, the method described in Jones et al. *(1993) applies for Lipschitz optimization when the Lipschitz constant is not known. They also have in common some constructional elements. In contrast to most other algorithms, they mix global and local search and do not apply a two step scheme, first to search globally and then to search locally. The algorithm of Jones et al. yields good results on some popular test functions. However, error bounds are missing.*

2 Optimization algorithms

We propose two adaptive algorithms. The first one is useful for a function class $F^d_{L,D,\varrho}$ with known Lipschitz parameter L. The second one is suitable for the situation that the Lipschitz parameter is unknown. For both algorithms, the knowledge of the other parameters D, ϱ is unimportant for their definition and their success, nevertheless they are important for the cost estimation.

First, we want to make precise what kind of methods we want to consider and how cost and error are defined. We are interested in such algorithms which use function values at adaptively chosen knots $x^1, \ldots, x^n$. The first knot x^1 is independent of the objective f and fixed for a particular method. The knots x^j with $j \geq 2$ may depend on the previously chosen knots and obtained function values. Also, the number of function calls may depend on the observed data. Every such method A can be expressed by

$$A(f) = \phi \circ N(f).$$

The information operator

$$N : F^d_{L,D,\varrho} \to \bigcup_{n=1}^{\infty} \mathbb{R}^n$$

gives the function values at the adaptively chosen knots. It obtains these knots by applying functions $\psi_j : \mathbb{R}^{j-1}$ such that

$$x^j = \psi_j(f(x^1), \ldots, f(x^{j-1})).$$

It stops after n function calls according to a stopping rule $s : \bigcup_{j=1}^{\infty} \mathbb{R}^j \to \{0,1\}$ iff

$$\forall j < n \quad s(f(x^1), \ldots, f(x^j)) = 1, \qquad s(f(x^1), \ldots, f(x^n)) = 0.$$

The mapping

$$\phi : \bigcup_{i=1}^{\infty} \mathbb{R}^n \to [0,1]^d$$

constructs the point x_* using the information vector $N(f)$.

For our methods we assume the real number model as described in detail in Novak (1995). In particular, we assume that we can represent real numbers exactly and that we can calculate with them exactly.

The error of a method A applied to a function $f \in F^d_{L,D,\varrho}$ and returning $A(f) = x_*$ is

$$\Delta(A, f) := |\inf f - f(x_*)|.$$

The (worst case) error of the method A is

$$\Delta(A, F^d_{L,D,\varrho}) := \sup\{\Delta(A, f) : \; f \in F^d_{L,D,\varrho}\}.$$

The cost(A, f) of a method A applied to a function $f \in F^d_{L,D,\varrho}$ is the number of function calls the method A uses for f. The (worst case) cost of A is

$$\mathrm{cost}(A, F^d_{L,D,\varrho}) := \sup\{\mathrm{cost}(A, f) : \; f \in F^d_{L,D,\varrho}\}.$$

We come to some particular preliminaries for the two algorithms we propose. Let e_i be the i-th unit vector in $\mathbb{R}^d$ having $(e_i)_i = 1$ and $(e_i)_j = 0$ for $j \neq i$.

The algorithms use

$$Y(j) := \left\{ \sum_{l=1}^{d} a_l \cdot e_l, a_l \in \{-3^{-j+1}, 0, 3^{-j+1}\} \right\} \setminus \{0\}, \quad j \in \mathbb{N}.$$

Each of the sets $Y(j)$ consists of $3^d - 1$ points. Let

$$\mathcal{M} := (\tfrac{1}{2}, \ldots, \tfrac{1}{2})^T$$

denote the midpoint of the unit cube in $\mathbb{R}^d$.

For the case of known Lipschitz parameter L, we propose the following optimization algorithm $S(L,k)$ performing k steps as described in figure 1. After k steps, $S(L,k)$ returns x_*. It is similar to the one of Perevozchikov (1990).

Lemma 2 *Let $f : [0,1]^d \to \mathbb{R}$ be Lipschitz with constant $L > 0$. Then, after step j and for each global minimizer x^*, there exist a pair $(x_j, f(x_j)) \in N_{L,j}$ such that $\|x_j - x^*\|_\infty \leq 2^{-1}3^{-j+1}$.*

Proof by induction.
$j = 1$: We have $(\mathcal{M}, f(\mathcal{M})) \in N_{L,1}$. For all $x \in [0,1]^d$ we have $\|x - \mathcal{M}\|_\infty \leq 2^{-1}$.
$j \to j+1$: Let $(x_j, f(x_j)) \in N_{L,j}$ such that $\|x_j - x^*\|_\infty \leq 2^{-1}3^{-j+1}$. In step $j+1$ of the algorithm, we check whether $f(x_j) \leq f_* + L\,2^{-1}3^{-j+1}$ which is true:

$$|f(x_j) - f_*| \leq |f(x_j) - f(x^*)| \leq L\,2^{-1}3^{-j+1}. \tag{7}$$

So we choose the pairs

$$(x_j + y, f(x_j + y)), \quad y \in Y(j+1),$$

to be in $N_{L,j+1}$. For (at least) one of these y we have

$$\|y - x^*\|_\infty \leq 2^{-1}3^{-j}.$$

Choose $x_{j+1} := x_j + y$ for such a y. □

The sets $N_{L,j}$ are subsets of the equidistant meshes

$$\text{mesh}(j) := \left\{ \sum_{i=1}^{d} \alpha_i\, e_i,\ \alpha_i \in \{2^{-1}3^{-j+1} + l \cdot 3^{-j+1}, l = 0, \ldots, 3^{j-1} - 1\} \right\}.$$

Furthermore, $S(L,k)$ guarantees the same level of approximation as mesh(k) in the following sense: For every global minimizer x^* we have

$$\min_{x \in \text{mesh}(k)} \|x - x^*\|_\infty \leq 2^{-1}3^{-k+1}, \qquad \min_{x \in N_{L,k}} \|x - x^*\|_\infty \leq 2^{-1}3^{-k+1}.$$

```
step 1 applying Lipschitz constant $L$:
oracle call: get $f(\mathcal{M})$.
set $N_{L,1} := \{(\mathcal{M}, f(\mathcal{M}))\}$
set $x_* := \mathcal{M}$; $f_* := f(\mathcal{M})$.

step $j$, $j \geq 2$, applying Lipschitz constant $L$:
set $N_{L,j} := \emptyset$;
for $(x, f(x)) \in N_{L,j-1}$ do
    if ($f(x) \leq f_* + L\, 2^{-1}3^{-j+2}$ ) then
        set $N_{L,j} := N_{L,j} \cup \{(x, f(x))\}$;
        for $y \in Y(j)$ do
            oracle call: get $f(x+y)$;
            set $N_{L,j} := N_{L,j} \cup \{(x+y, f(x+y))\}$,
            if ($f(x+y) < f_*$) then
                set $x_* := x+y$; $f_* := f(x+y)$;
            end if;
        next $y$;
    end if;
next $x$;
```

Fig. 1. The steps performed by $S(L,k)$

We are now ready to prove bounds for error and cost of $S(L,k)$. We use the following constants:

$$\varepsilon_k := L2^{-1}3^{-k+1}, \tag{8}$$

$$j(L,\varrho) := \lceil \log_3(L/(2\varrho)) \rceil + 3, \tag{9}$$

$$j(L,\varrho,d) := \lceil \log_3(L/(2\varrho^2 D^{2/d})) \rceil + 6, \tag{10}$$

$$c(d) := \frac{3^{d/2}}{3^{d/2}-1}. \tag{11}$$

For $d \geq 1$, we have $c(d) \in (1, 2.37]$. We use an idea of Perevozchikov (1990) to prove the following

Theorem 3 *(1) Error estimation: For $k \in \mathbb{N}$, we have*

$$\Delta(S(L,k), F^d_{L,D,\varrho}) \leq \varepsilon_{L,k}.$$

(2) Cost estimation: We have
(a) for $k \in \mathbb{N}$

$$\mathrm{cost}(S(L,k), F^d_{L,D,\varrho}) \leq 3^{d(k-1)},$$

(b) for $k \geq j(L,\varrho)$

$$\mathrm{cost}(S(L,k), F^d_{L,D,\varrho})$$

$$\leq \left(\tfrac{27}{2}\, L/\varrho\right)^d + c(d)\,(3^d-1)\, D\, L^{d/2} 2^{-d/2} \left(3^{(k-1)d/2} - \left(\tfrac{3}{2} L/\varrho\right)^{d/2}\right).$$

(c) for $k \geq \max\{j(L,\varrho), j(L,\varrho,d)\}$

$$\mathrm{cost}(S(L,k), F^d_{L,D,\varrho}) \leq D L^{d/2} 2^{-d/2} 3^{(k+1)d/2+1} = D L^d 2^{-d} 3^{d+1} \varepsilon_{L,k}^{-d/2}.$$

Proof. 1. Let $f \in F^d_{L,D,\varrho}$. From Lemma 2 we know that there exists a pair $(x_k, f(x_k)) \in N_{L,k}$ such that $\|x_k - x^*\|_\infty \leq 2^{-1}3^{-k+1}$. Then

$$|f(x_*) - f(x^*)| \leq |f(x_k) - f(x^*)| \leq L\, 2^{-1}3^{-k+1}.$$

2. For *(a),* we have that $S(L,k)$ chooses only points in mesh(k), which consists of $3^{d(k-1)}$ points.
(b). Let

$$N^*_{L,j-1} \tag{12}$$

be the set of pairs $(x, f(x)) \in N_{L,j-1}$ which in step j pass the test $f(x) \leq f_* + L/2 \cdot 3^{-j+2}$. Then in step j, the number of new function evaluations is bounded by

$$|N_{L,j} \setminus N_{L,j-1}| \leq (3^d - 1) N^*_{j-1}.$$

We can use this estimation for steps j if $N^*_{L,j-1} \subset A(f,\varrho)$. As in *1.*, one can show

$$\min\{f(y) : y \in N^*_{L,j-1}\} \leq \min f + L\, 2^{-1}3^{-j+2}.$$

Furthermore,

$$\forall x \in N^*_{L,j-1} \quad f(x) \leq f_* + L\, 2^{-1}3^{-j+2} \leq \min\{f(y) :\, y \in N^*_{L,j-1}\} + L\, 2^{-1}3^{-j+2}.$$

So we get

$$\forall x \in N^*_{L,j-1} \qquad x \in A(L\, 3^{-j+2}).$$

For $x, y \in N^*_{L,j-1}$ and $x \neq y$ we have

$$B(x, 2^{-1}3^{-j+2}) \subset A(L\, 2^{-1}3^{-j+3}), \quad B(x, 2^{-1}3^{-j+2}) \cap B(y, 2^{-1}3^{-j+2}) = \emptyset.$$

For $L\,2^{-1}3^{-j+3} \le \varrho$, i.e.

$$j-1 \ge \lceil \log_3(L/(2\varrho)) \rceil + 2 = j(L,\varrho) - 1,$$

we can estimate

$$\begin{aligned} |\,N^*_{L,j-1}\,| &\le \frac{\lambda^d(A(L\,2^{-1}3^{-j+3}))}{\lambda^d(B(x,\frac{1}{2}\,3^{-j+2}))} \le \frac{D\,(L\,2^{-1}3^{-j+3})^{d/2}}{(3^{-j+2})^d} \\ &= D L^{d/2} 2^{-d/2} 3^{(j-1)d/2}\,. \end{aligned} \tag{13}$$

So the number of new points in level j can be estimated by

$$(3^d-1)\,D L^{d/2} 2^{-d/2} 3^{(j-1)d/2}. \tag{14}$$

It follows immediately

$$\begin{aligned} &\operatorname{cost}(S(L,k), F^d_{L,D,\varrho}) \\ &\le |\operatorname{mesh}(j(L,\varrho)-1)| + \sum_{j=j(L,\varrho)}^{k} (3^d-1)\,D\,L^{d/2}2^{-d/2}3^{(j-1)d/2} \\ &\le \left(\tfrac{27}{2}\,L/\varrho\right)^d + c(d)\,(3^d-1)\,D\,L^{d/2}2^{-d/2}(3^{(k-1)d/2} - 3^{(k(L,\varrho)-2)d/2}) \\ &\le \left(\tfrac{27}{2}\,L/\varrho\right)^d + c(d)\,(3^d-1)\,D\,L^{d/2}2^{-d/2}\left(3^{(k-1)d/2} - \left(\tfrac{3}{2}L/\varrho\right)^{d/2}\right). \end{aligned}$$

So we proved *(b).*
Under the assumptions of *(c)* we have for $d=1$

$$\tfrac{27}{2}L/\varrho \le c(d) D L^{1/2} 2^{-1/2} 3^{(k-1)/2}$$

and for $d \ge 2$

$$\left(\tfrac{27}{2}\,L\varrho\right)^d \le \tfrac{3}{2}\cdot 3^d D L^{d/2} 2^{-d/2} 3^{(k-1)d/2}.$$

In both cases we conclude from *(b)*

$$\operatorname{cost}(S(L,k), F^d_{L,D,\varrho}) \le D\,L^{d/2}2^{-d/2}3^{(k+1)d/2+1} = D L^d 2^{-d} 3^{d+1} \varepsilon_{L,k}^{-d/2}.$$

□

We now turn to the case of unknown Lipschitz parameter L. For this situation, we propose the algorithm $Z(k)$ as described in figure 2. It uses the steps j of $S(L,k')$ for several Lipschitz constants $L(1) < L(2) < \ldots$. An additional

```
for $l$ from 1 to $k$ do    # diagonal $l$
    for $i$ from 1 to $lastconst(l)$    # constant $L(i)$
        for $j$ from 1 to $laststep(l,i)$ do    # step $j$
            if $(i \neq 1,\ j = 1)$ then
                **step 1'** applying Lipschitz constant $L(i)$;
            else
                **step $j$** applying Lipschitz constant $L(i)$;
            end if;
        next $j$;
    next $i$;
next $l$;
return $x_*$;
```

Fig. 2. The algorithm $Z(k)$

step 1' is also used:

step 1' using Lipschitz constant L:
oracle call: get $f(\mathcal{M})$;
set $N_{L,1} := \{(\mathcal{M}, f(\mathcal{M}))\}$.

The constants $L(i)$ and two controlling functions

$$\mathit{lastconst}: \mathbb{N} \to \mathbb{N}, \quad l \mapsto \mathit{lastconst}(l)$$

$$\mathit{laststep}: \mathbb{N} \times \mathbb{N} \to \mathbb{N}, \quad (l,i) \mapsto \mathit{laststep}(l,i)$$

determine the behavior of the algorithm. In a first definition of the algorithm, we only require the following properties:

- $L(i+1) > L(i)$ for all $i \in \mathbb{N}$,
- *lastconst* increasing,
- $\mathit{laststep}(l,i)$ increasing in l, and decreasing in i.

$Z(k)$ is a diagonal scheme. The parameter k in $Z(k)$ is the number of per-

formed diagonals. In diagonal l, the algorithm examines the function assuming the Lipschitz constants $L(1),\ldots,L(\mathit{lastconst}(l))$, i.e.: For constant $L(i)$, the algorithms performs step *laststep*$(l-1,i)+1$ to step *laststep*(l,i).
While the algorithm has only one instance of f_* and x_* it uses separate sets $N_{L(i),j}$ for each constant $L(i)$.
Note that the objective f will be evaluated at certain points for several constants $L(i)$, i.e. several times. The midpoint $\mathcal{M}$ for example *lastconst(k)* times. We discuss later in remark 7 why this is reasonable.

Let $m,k\in\mathbb{N}$ such that *lastconst*$(k)\geq m$. (Among others,) the method $Z(k)$ performs step 1' (step 1 if $m=1$) to step *laststep(k,m)* applying Lipschitz constant $L(m)$. This way, $Z(h)$ determines the sets

$$N_{L(m),1},\ldots,N_{L(m),\mathit{laststep}(k,m)}.$$

We have the following analogue to Lemma 2:

Lemma 4 *Let $f\in F^d_{L,D,\varrho}$ with $L\leq L(m)$ for some $m\in\mathbb{N}$ and $k\in\mathbb{N}$ such that lastconst$(k)\geq m$. Let x^* be a global minimizer. For $1\leq j\leq$ laststep(k,m), there exists a pair $(x_j,f(x_j))\in N_{L(m),j}$ such that $\|x_j-x^*\|_\infty\leq 2^{-1}3^{-j+1}$.*

Proof. The proof is similar to that of Lemma 2. The only difference to the situation there is that the least value found so far f_* is shared and altered by function evaluations applying different constants $L(i)$. However, f_* only once enters the proof: in (7). Here, only $f(x^*)\leq f_*\leq f(x_j)$ is needed, which is true for the new situation, too. □

We want to examine $Z(k)$ for the choice

$$L(i):=3^{i-1},\quad i\in\mathbb{N},\tag{15}$$

$$\mathit{lastconst}(l):=\left\lceil\frac{l}{h}\right\rceil,\qquad \mathit{laststep}(l,i):=\begin{cases}l-h(i-1), & \text{if } i\leq b(l),\\ 0, & \text{else,}\end{cases}\tag{16}$$

with parameter $h\in\{3,4,\ldots\}$. We will discuss the choice of h in remark 6. Let

$$Z(h,k)$$

be the so defined algorithm. We use the constants

$$\varepsilon_{L,h,k}:=L^{h+1}2^{-1}3^{-k+1},\tag{17}$$

$$c(d,h):=\frac{1}{1-3^{-hd/2}},\qquad c'(d,h):=\frac{1}{1-3^{(1-h/2)d}},\tag{18}$$

$$C(L,d,h):=c(d)\left[c(d,h)3^{-d/2}+c'(d,h)L^{-hd/2}\right],\tag{19}$$

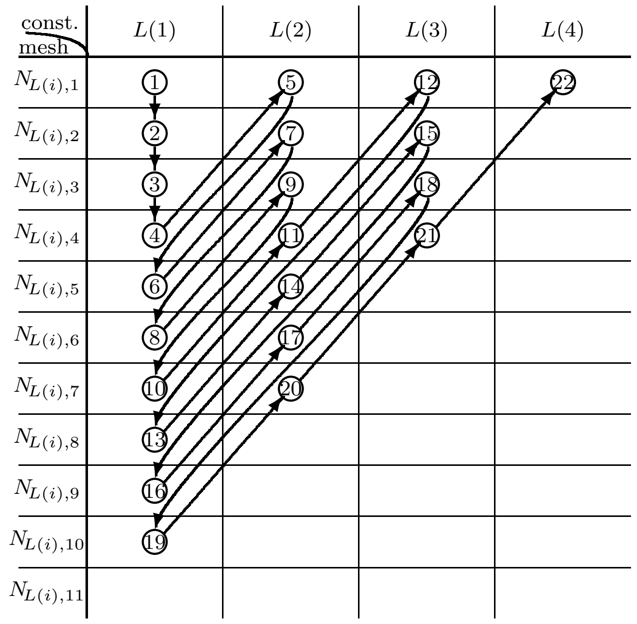

Fig. 3. Scheme of $Z(h,k)$ for $h=3$ and $k=10$

with $c(d)$ defined as in (11). For $h \geq 3$, $d \geq 1$ and $L \geq 1$ we have

$$c(d,h) \in (1, 1.25], \quad c'(d,h) \in (1, 2.37], \quad C(L,d,h) \in (0, 7.37].$$

Theorem 5 *Let $f \in F_{L(m),D,\varrho}$ for some $m \in \mathbb{N}$, and $k \geq h(m-1)+1$.*

(1) Error estimation:

$$\Delta(Z(h,k), F^d_{L,D,\varrho}) \leq \varepsilon_{L,h,k},$$

(2) Cost estimation:

$$\begin{aligned}
&\operatorname{cost}(Z(h,k), F^d_{L,D,\varrho}) \\
&\leq \left[\log_3 L(m) + c(2d) L(m)^{-1} 3^{\lceil k/h \rceil}\right] \left(\tfrac{9}{2}\, L(m)/\varrho\right)^d + \\
&\quad C(L(m),d,h)\, L(m)^{d/2} D\, 2^{-d/2} 3^{(k+1)d/2}, \\
&= \left[\log_3 L(m) + c(2d) 2^{-1/h} 3^{1-1/h} L(m)^{1/h} \varepsilon^{-1/h}_{L(m),h,k}\right] \left(\tfrac{9}{2}\, L(m)/\varrho\right)^d + \\
&\quad C(L(m),d,h)\, 2^{-d} 3^d\, D\, L(m)^{(h+2)d/2} \varepsilon^{-d/2}_{L(m),h,k}.
\end{aligned}$$

Proof 1. For the constant $L(m)$, the algorithm performs step 1 (or 1') up to step $k-h(m-1)$. From Lemma 4 we know that for every global minimizer x^* there exists $x_{k-h(m-1)} \in N_{L(m),k-h(m-1)}$ such that $\|x_{k-h(m-1)} - x^*\|_\infty \leq$

$2^{-1}3^{-k+h(m-1)+1}$. It follows immediately

$$|f(x_*) - f(x^*)| \leq |f(x_{k-h(m-1)}) - f(x^*)| \leq L(m)\, 2^{-1}3^{-k+h(m-1)+1} = \varepsilon_{L(m),h,k}.$$

2. The cost estimation is similar to the one in the proof of Theorem 3. Recall $N^*_{L,j-1}$ to be defined as in (12). Let $1 \leq i \leq \mathit{lastconst}(k)$. For the new points in step j applying Lipschitz constant $L(i)$, we have

$$|N_{L(i),j} \setminus N_{L(i),j-1}| \leq (3^d - 1)|N^*_{L(i),j-1}|.$$

We can use this estimation for steps j with $N^*_{L(i),j-1} \subset A(f,\varrho)$. As in *1.*, one can show

$$\min\{f(y) : y \in N_{L(i),j-1}\} \leq \min f + L(m)\, 2^{-1}3^{-j+2}.$$

Furthermore,

$$\forall\, x \in N^*_{L(i),j-1} \qquad f(x) < \min\{f(y) : y \in N_{L(i),j-1}\} + L(i)\, 2^{-1}3^{-j+2},$$

so we get

$$\forall\, x \in N^*_{L(i),j-1} \qquad x \in A(f,(L(i)+L(m))2^{-1}3^{-j+2}).$$

For $x, y \in N^*_{L(i),j-1}$ with $x \neq y$ we have

$$B(x, 2^{-1}3^{-j+2}) \subset A((2L(m)+L(i))\, 2^{-1}3^{-j+2}),$$

$$B(x, 2^{-1}3^{-j+2}) \cap B(y, 2^{-1}3^{-j+2}) = \emptyset.$$

For $(2L(m)+L(i))2^{-1}3^{-j+2} \leq \varrho$, i.e.

$$j - 1 \geq \lceil \log_3((2L(m)+L(i))/(2\varrho)) \rceil + 1 =: j(m,i,\varrho) - 1, \tag{20}$$

we get

$$\begin{aligned} |N^*_{L(i),j-1}| &\leq \frac{\lambda^d(A((2L(m)+L(i))2^{-1}3^{-j+2}))}{\lambda^d(B(2^{-1}3^{-j+2}(x)))} \\ &\leq D(L(m)+2^{-1}L(i))^{d/2}\, 3^{(j-2)d/2}. \end{aligned} \tag{21}$$

It follows immediately

$$|N_{L(i),j} \setminus N_{L(i),j-1}| \leq (3^d-1)D(L(m)+2^{-1}L(i))^{d/2}\, 3^{(j-2)d/2}. \tag{22}$$

For the steps 1 (or 1') to $\mathit{laststep}(k,i) = k - h(i-1)$ and $k - h(i-1) \geq j(m,i,\varrho)$ we get the cost estimation

$$
\begin{aligned}
&|N_{L(i),1}| + \sum_{j=2}^{k-h(i-1)} |N_{L(i),j} \setminus N_{L(i),j-1}| \\
&\le |\,\mathrm{mesh}(j(m,i,\varrho)-1)\,| + \sum_{j=j(m,i,\varrho)}^{k-h(i-1)} (3^d-1)D(L(m)+2^{-1}L(i))^{d/2}\, 3^{(j-2)d/2} \\
&\le \left(\frac{9(2L(m)+L(i))}{2\varrho}\right)^d + \\
&\quad (3^d-1)D(L(m)+2^{-1}L(i))^{d/2} 3^{(j(m,i,\varrho)-2)d/2}\, \frac{3^{(k-h(i-1)-j(m,i,\varrho)+1)d/2}-1}{3^{d/2}-1} \\
&= \left(\frac{9(2L(m)+L(i))}{2\varrho}\right)^d + \\
&\quad (3^d-1)D(L(m)+2^{-1}L(i))^{d/2} c(d) \left[3^{(k-h(i-1)-2)d/2} - 3^{(j(m,i,\varrho)-3)d/2}\right].
\end{aligned}
$$

In order to get an estimation for $\mathrm{cost}(Z(h,k), F_{L(m),D,\varrho}$, we sum up these numbers for constants $L(1),\dots,L(\lceil k/h\rceil)$:

$$
\begin{aligned}
&\mathrm{cost}(Z(h,k),F_{L(m),D,\varrho}) \le \sum_{i=1}^{\lceil k/h\rceil}\left[|N_{L(i),1}| + \sum_{j=2}^{k-h(i-1)} |N_{L(i),j}\setminus N_{L(i),j-1}|\right] \\
&\le \sum_{i=1}^{m-1}\left(\frac{9(L(m))}{2\varrho}\right)^d + \sum_{i=m}^{\lceil k/h\rceil}\left(\frac{9(3L(i))}{2\varrho}\right)^d + \\
&\quad \sum_{i=1}^{m-1}(3^d-1)D(2^{-1}L(m))^{d/2}c(d)\left[3^{(k-h(i-1)-2)d/2} - 3^{(j(m,i,\varrho)-3)d/2}\right] + \\
&\quad \sum_{i=m}^{\lceil k/h\rceil}(3^d-1)D(2^{-1}\cdot 3L(i))^{d/2}c(d)\left[3^{(k-h(i-1)-2)d/2} - 3^{(j(m,i,\varrho)-3)d/2}\right] \\
&\le (m-1)\cdot\left(\frac{9}{2}\frac{L(m)}{\varrho}\right)^d + \left(\frac{27}{2}\frac{L(m)}{\varrho}\right)^d \sum_{i=0}^{\lceil k/h\rceil-m} 3^{di} + \\
&\quad (3^d-1)D(2^{-1}L(m))^{d/2}c(d)3^{(k-2)d/2}\sum_{i=0}^{m-2}3^{-dhi/2} + \\
&\quad (3^d-1)D(2^{-1}\cdot 3L(m))^{d/2}c(d)3^{(k-h(m-1)-2)d/2}\sum_{i=0}^{\lceil k/h\rceil-m}3^{(1-h/2)di} \qquad (23) \\
&\le \left[m-1+c(2d)\,3^{\lceil k/h\rceil-m+1}\right]\left(\tfrac{9}{2}\,L(m)/\varrho\right)^d + \\
&\quad (3^d-1)D(2^{-1}L(m))^{d/2}c(d)3^{(k-2)d/2}c(d,h) + \\
&\quad (3^d-1)D(2^{-1}L(m))^{d/2}c(d)3^{(k-h(m-1)-1)d/2}c'(d,h) \\
&\le \left[m-1+c(2d)\,3^{\lceil k/h\rceil-m+1}\right]\left(\tfrac{9}{2}\,L(m)/\varrho\right)^d + \\
&\quad (3^d-1)D(2^{-1}L(m))^{d/2}c(d)3^{(k-1)d/2}\left[c(d,h)3^{-d/2} + c'(d,h)3^{-(m-1)hd/2}\right]
\end{aligned}
$$

$$\le \left[\log_3 L(m) + c(2d)L(m)^{-1}3^{\lceil k/h\rceil}\right]\left(\tfrac{9}{2}\,L(m)/\varrho\right)^d +$$
$$C(L(m),d,h)\,L(m)^{d/2}D\,2^{-d/2}3^{(k+1)d/2}.$$

With

$$\varepsilon_{L(m),h,k}^{-d/2} = L(m)^{-(h+1)d/2}2^{d/2}3^{(k-1)d/2}, \quad \varepsilon_{L(m),h,k}^{-1/h} = L(m)^{-1-1/h}2^{1/h}3^{(k-1)/h},$$

we get

$$\begin{aligned}&\mathrm{cost}(Z(h,k),F^d_{L,D,\varrho})\\ &\le \left[\log_3 L(m) + c(2d)2^{-1/h}3^{1-1/h}L(m)^{1/h}\varepsilon_{L(m),h,k}^{-1/h}\right]\left(\frac{9}{2}\frac{L(m)}{\varrho}\right)^d +\\ &\quad C(L(m),d,h)\,2^{-d}3^d\,D\,L(m)^{(h+2)d/2}\varepsilon_{L(m),h,k}^{-d/2}.\end{aligned}$$

□

Remark 6 *We can explain now the restriction on the parameter h: In order that the sum in (23) is bounded for all $k \in \mathbb{N}$, we need that $h \ge 3$.*
The parameter h allows to decide whether to focus on local or global search in the following sense: The cost used by performing steps applying constant $L(i)$ are approx.

$$(3^d-1)D(L(m)+2^{-1}L(i))^{d/2}c(d)\left[3^{(k-h(i-1)-2)d/2} - 3^{(j(m,i,\varrho)-3)d/2}\right].$$

For constant $L(i+1)$ we spend $3^{-hd/2}$ times as much as we spend for $L(i)$. Choosing a high value for h leads to focus on a precise approximation of found (local) minima. This is done in steps for small constants $L(i)$. On the other hand, a low value of h, 3 or 4 say, will focus more on global search, performed by steps for big constants $L(i)$.

Remark 7 *As mentioned before, the algorithm Z does not store all function evaluations, but only those in the sets $N_{L(i),j}$. Consequently, the algorithms evaluates f at certain points several times. The alternative would be to store all data and, before we make an oracle call, check whether this function value is already known. This way, we would save some oracle calls, i.e. we would reduce cost.*
Still, we do not stick to this idea. The reason is in the arithmetic cost. One can show for $Z(h,k)$ that the arithmetic cost (and also the storing cost) develop linearly to the information cost. So, the chosen cost definition $\mathrm{cost}(Z(h,k),F^d_{L,D,\varrho})$ is an appropriate measure for all cost we face when we want to implement the algorithm.
On the other hand, data storing would lead to an only negligible cost reduction while the arithmetic cost no longer behaves linear to the information cost. To

check whether a certain function value is already known leads to an additional logarithmic factor for the arithmetic cost.

3 Optimality results

We find lower bounds for adaptive deterministic, non-adaptive deterministic and adaptive randomized methods which show that

- the algorithms $S(L,\cdot)$ and $Z(h,\cdot)$ have the optimal convergence rate,
- adaptiveness is essential for optimality,
- up to constants, randomization (Monte-Carlo methods) gives no further advantage.

We will need the following (technical)

Lemma 8 *Let $g : [0,\varrho] \to [0,\infty)$ be piecewise linear with $n \in \mathbb{N}$ nodes $0 = \delta_1 < \delta_2 < \ldots < \delta_n = \varrho$. Let $g(\delta_i) \leq D^{1/d}\delta_i^{1/2}$. For $f : [0,1]^d \to \mathbb{R}$ and $0 \leq \delta \leq \varrho$ let $\lambda^d(A(f,\delta)) \leq g^d(\delta)$. Then*

$$\forall\, 0 < \delta \leq \varrho \qquad \lambda^d(A(f,\delta)) \leq D\delta^{d/2}.$$

Proof. From $g(\delta_i) \leq D^{1/d}\delta_i^{1/2}$ for $i = 1,\ldots,n$ and the convexity of the root function we conclude

$$\forall\, 0 \leq \delta \leq \varrho \qquad g(\delta) \leq D^{1/d}\delta^{1/2}.$$

Using $\lambda^d(A(f,\delta)) \leq g^d(\delta)$ and the strict monotony of $x \mapsto x^d$ we get

$$\forall\, 0 < \delta \leq \varrho \qquad \lambda^d(A(f,\delta)) \leq D\delta^{d/2}.$$

□

Recall the error numbers to be defined as

$$e_n(F^d_{L,D,\varrho}) := \inf\{\Delta(A, F^d_{L,D,\varrho}) : A : \operatorname{cost}(A, F^d_{L,D,\varrho}) \leq n\}.$$

We start with a lower bound for $e_n(F^d_{L,D,\varrho})$. The basic idea of the proof, to construct a set of fooling functions, goes back to Bakvalov (1959). We will use this principle for non-adaptive methods, too.

Theorem 9 *Let $m \in \mathbb{N}$ with $m \geq L^2D^{2/d}(LD^{2/d} - \varrho)^{-1}$ and $n = m^d - 2$. Then*

$$e_n(F^d_{L,D,\varrho}) \geq \frac{L^2D^{2/d}}{4(n+2)^{2/d}}.$$

Proof. Let

$$I := \{\boldsymbol{i} : \boldsymbol{i} = (i_1, \dots, i_d),\ i_k \in \{1, \dots, m\},\ k = 1, \dots, d\}, \tag{24}$$

$$l := \frac{D^{2/d} L}{2m}, \tag{25}$$

$$y^{\boldsymbol{i}} := \frac{l}{m} \cdot (i_1 - 1/2, \dots, i_d - 1/2)^T .$$

For $\boldsymbol{i} \in I$ define

$$f_{\boldsymbol{i}}(x) := \begin{cases} L\|x - y^{\boldsymbol{i}}\|_\infty, & \|x - y^{\boldsymbol{i}}\|_\infty \le l/(2m), \\ \dfrac{Ll}{2m}, & x \in [0, l]^d \setminus \overline{B(y^{\boldsymbol{i}}, 1/(2m))}, \\ \dfrac{Ll}{2m} + L \max\limits_{1 \le j \le d}(x_j - l), & \|x\|_\infty > l. \end{cases} \tag{26}$$

We show $f_{\boldsymbol{i}} \in F^d_{L,D,\varrho}$. Obviously, f is Lipschitz with constant L. If

$$l \le D^{2/d} - \frac{\varrho}{L} \quad \left(\Leftrightarrow \quad m \ge \frac{L^2 D^{2/d}}{L D^{2/d} - \varrho}\right), \tag{27}$$

then $\lambda^d(A(f_{\boldsymbol{i}}, 0)) = 0$, $\lambda^d(A(f_{\boldsymbol{i}}, L/(2m))) = D(L/(2m))^{d/2}$, and $\lambda(\varrho) \le D\varrho^{d/2}$. Condition (27) is fulfilled due to (5) and

$$D^{2/d} - \frac{\varrho}{L} > 0 \quad \Leftrightarrow \quad \varrho < D^{2/d} L^2.$$

Using Lemma 8, we get $f_{\boldsymbol{i}} \in F^d_{L,D,\varrho}$.
Now, let $A_n = \phi \circ N$ be an algorithms which uses at most n function calls. Then there exist (at least) two different $\boldsymbol{i}, \boldsymbol{j} \in I$ such that

$$N(f_{\boldsymbol{i}}) = N(f_{\boldsymbol{j}}).$$

No matter where the algorithm chooses $x_* = \phi \circ N(f_{\boldsymbol{i}}) = \phi \circ N(f_{\boldsymbol{j}})$, we will have $f_{\boldsymbol{i}}(x_*) \ge Ll/(2m)$ or $f_{\boldsymbol{j}}(x_*) \ge Ll/(2m)$, but $\min f_{\boldsymbol{i}} = \min f_{\boldsymbol{j}} = 0$. Consequently,

$$\Delta(A_n, F^d_{L,D,\varrho}) \ge \frac{l\,L}{2m} = \frac{L^2 D^{2/d}}{4(n+2)^{2/d}}.$$

□

Corollary 10 *The algorithms* $(S(L,k))_{k\in\mathbb{N}}$ *and* $(Z(h,k))_{k\in\mathbb{N}}$ *have the optimal speed of convergence.*

Proof. Let $m_0 := \lceil L^2 D^{2/d}(LD^{2/d} - \varrho)^{-1}\rceil$ and $n(m) := m^d - 2$. Let $n \geq \max\{n(m_0), 3, [(9/8)^{d/2} - 1]^{-1}\}$. Choose m s.th. $n(m-1)+1 \leq n \leq n(m)$. Then

$$e_n(F^d_{L,D,\varrho}) \geq e_{n(m)}(F^d_{L,D,\varrho}) \geq \frac{D^{2/d}L^2}{4m^2} \geq \frac{D^{2/d}L^2}{9\,n^{2/d}}.$$

For $k \in \mathbb{N}$, let

$$n_k := \lfloor DL^{d/2} 2^{-d/2} 3^{(k+1)d/2+1}\rfloor.$$

From Theorem 3 we know that $S(L,k)$ uses at most n_k oracle calls and delivers an error level

$$\Delta(S(L,k), F^d_{L,D,\varrho}) \leq \varepsilon_{L,k} \leq D^{2/d}L^2 2^{-2} 3^{2+2/d} n_k^{-2/d},$$

that means that the algorithms $S(L,k)$, $k \in \mathbb{N}$, have the optimal convergence rate $n^{-2/d}$.

For $Z(h,k)$, we have $\Delta(Z(h,k), F^d_{L,D,\varrho}) \leq \varepsilon_{L(m),h,k}$ and

$$\mathrm{cost}(Z(h,k), F^d_{L,D,\varrho}) \leq \alpha(L(m),d,h)\,\varepsilon^{-1/h}_{L(),h,k} + \beta(L,D,h,d)\,\varepsilon^{-2/d}_{L,h,k},$$

with constants α, β which can be determined with Theorem 5. For $\varepsilon > 0$ small enough (i.e. for large k) we have

$$\mathrm{cost}(Z(h,k), F^d_{L,D,\varrho}) \leq (\beta(L,D,h,d)+1)\,\varepsilon^{-2/d}_{L,h,k}.$$

Proceeding as for $S(L,k)$, $k \in \mathbb{N}$, we see that $Z(h,k)$, $k \in \mathbb{N}$, have the optimal speed of convergence. $\square$

We turn to non-adaptive methods, i.e. algorithms $A = \phi \circ N$ with the following property:

$$\exists\, n \in \mathbb{N} \quad \exists\, x_1, \ldots, x_n \in [0,1]^d \quad \forall f \in F^d_{L,D,\varrho} \qquad N(f) = (f(x_1), \ldots, f(x_n)).$$

Lemma 11 *Let* $m \geq \max\{(\frac{1}{4}D^{2/d}L)^{-1}, (2D^{1/d}\varrho^{1/2} - 4\varrho/L)^{-1}, L/(4\varrho)\}$. *For* $n = m^d - 1$, *let* A_n *be a non-adaptive method with* $\mathrm{cost}(A_n, F^d_{L,D,\varrho}) = n$. *Then*

$$\Delta(A_n, F^d_{L,D,\varrho}) \geq \frac{L}{4(n+1)^{1/d}}.$$

Proof. Let I as in (24). For $\boldsymbol{i} \in I$ define

$$\begin{aligned} x^{\boldsymbol{i}} &:= \tfrac{1}{m}(i_1 - \tfrac{1}{2}, \ldots, i_d - \tfrac{1}{2})^T, \\ x^{\boldsymbol{i},1} &:= \tfrac{1}{m}(i_1 - \tfrac{3}{4}, i_2 - \tfrac{1}{2}, \ldots, i_d - \tfrac{1}{2})^T, \\ x^{\boldsymbol{i},2} &:= \tfrac{1}{2}(i_1 - \tfrac{1}{4}, i_2 - \tfrac{1}{2}, \ldots, i_d - \tfrac{1}{2})^T, \end{aligned} \tag{28}$$

$$f_{\boldsymbol{i},1}(x) := \begin{cases} L\,\|x - x^{\boldsymbol{i},1}\|_\infty, & x \in \overline{B(x^{\boldsymbol{i},1}, 1/(4m))}, \\ \dfrac{L}{4m}, & x \in \overline{B(x^{\boldsymbol{i}}, 1/(2m))} \setminus \overline{B(x^{\boldsymbol{i},1}, 1/(4m))}, \\ L\,\|x - x^{\boldsymbol{i}}\|_\infty - \dfrac{L}{4m}, & x \in [0,1]^d \setminus \overline{B(x^{\boldsymbol{i}}, 1/(2m))}, \end{cases}$$

$$f_{\boldsymbol{i},2}(x) := \begin{cases} L\,\|x - x^{\boldsymbol{i},2}\|_\infty, & x \in \overline{B(x^{\boldsymbol{i},2}, 1/(4m))}, \\ \dfrac{L}{4m}, & x \in \overline{B(x^{\boldsymbol{i}}, 1/(2m))} \setminus \overline{B(x^{\boldsymbol{i},2}, 1/(4m))}, \\ L\,\|x - x^{\boldsymbol{i}}\|_\infty - \dfrac{L}{4m}, & x \in [0,1]^d \setminus \overline{B(x^{\boldsymbol{i}}, 1/(2m))}. \end{cases}$$

One can show

$$\forall \boldsymbol{i} \in I \qquad f_{\boldsymbol{i},1}, f_{\boldsymbol{i},2} \in F^d_{L,D,\varrho}.$$

For the method A_n using n oracle calls, we have that in (at least) one cube

$$Q_{\boldsymbol{i}} := \{x \in \mathcal{D} :\ (\boldsymbol{i} - (1/m, \dots, 1/m) < x < \boldsymbol{i}/m\}, \qquad \boldsymbol{i} \in I,$$

there is no evaluation. For such an index $\boldsymbol{i}$, we have

$$N(f_{\boldsymbol{i},1}) = N(f_{\boldsymbol{i},2}).$$

Consequently,

$$\Delta(A_n, F^d_{L,D,\varrho}) \geq \frac{L}{4m} = \frac{L}{4(n+1)^{1/d}}.$$

□

Finally, we turn to Monte-Carlo methods (MCM). For a definition of Monte-Carlo methods, we refer to Novak (1988).
Let Q be a MCM which uses at most n oracle calls and $(\Omega, \mathcal{C}, P)$ be the probability space Q refers to. Let $f \in F^d_{L,D,\varrho}$. Then the error of Q with respect to f is defined as

$$\Delta(Q, f) := \int\limits_{\Omega} \Delta(Q(\omega)(f), f) P(d\omega),$$

the error of Q as

$$\Delta(Q, F^d_{L,D,\varrho}) := \sup_{f \in F^d_{L,D,\varrho}} \Delta(Q, f).$$

Lemma 12 *Let m be as in Theorem 9, even, and $n = 2m^d/2$. Let Q be a Monte-Carlo method using at most n oracle calls. Then*

$$\Delta(Q, F^d_{L,D,\varrho}) \geq \frac{n-1}{n} \frac{L^2 D^{2/d}}{8(2n)^{2/d}}.$$

Proof. Let l as in (25) and I and $f_{\boldsymbol{i}}$ as in (24) and (26). Define

$$g(x) := \begin{cases} \dfrac{Ll}{2m}, & \|x\|_\infty \le l, \\ \dfrac{Ll}{2m} + L \max\limits_{1 \le j \le d} (x_j - l), & \|x\|_\infty > l. \end{cases}$$

Let $F_I := \{f_{\boldsymbol{i}}, \boldsymbol{i} \in I\}$. We know from Theorem 9 that $F_I \subset F^d_{L,D,\varrho}$. Now, let $A = \phi \circ N$ be an (adaptive) deterministic algorithm using at most n oracle calls. For at least n different $\boldsymbol{i} \in I$ we have $N(f_{\boldsymbol{i}}) = N(g)$. Consequently,

$$\sum_{f \in F_I} (\inf f - f(A(f))) \ge \frac{(n-1)Ll}{2m} = \frac{(n-1)\, L^2 D^{2/d}}{4m^2}.$$

The proof is complete with Novak (1988), Prop. 2.1.9 using the uniform distribution on F_I. □

4 A numerical experiment

To illustrate the behavior of the algorithm $Z(h,k)$, we apply it to the test function $f_{BR}(x) : [-5, 10] \times [0, 15] \to \mathbb{R}$,

$$f_{BR}(x) := \left(x_2 - \frac{5.1}{4\pi^2} x_1^2 + \frac{5}{\pi} - 6\right)^2 + 10 \left(1 - \frac{1}{8\pi}\right) \cos x_1 + 10,$$

which we found in a collection of popular test functions in Törn, Žilinskas (1989). This function has 3 global minimizers $(-3.142, 12.275)$, $(3.142, 2.275)$, $(9.429, 2.425)$ and a global minimum of approximately 0.398. We choose $h = 4$ and three different values for k each representing a different stadium of the approximation. In each stadium, the algorithm approximates one more global minimizer.

We mention once more that the focus of this paper is on theoretical results. They tell us that in the worst case setting, the algorithm Z cannot be improved dramatically. However, we want to change the algorithm in some heuristically promising ways such that on the one hand the cost estimation of the original algorithm still holds and on the other hand we have a function wise speed-up. We will discuss these results in our forthcoming PhD thesis.

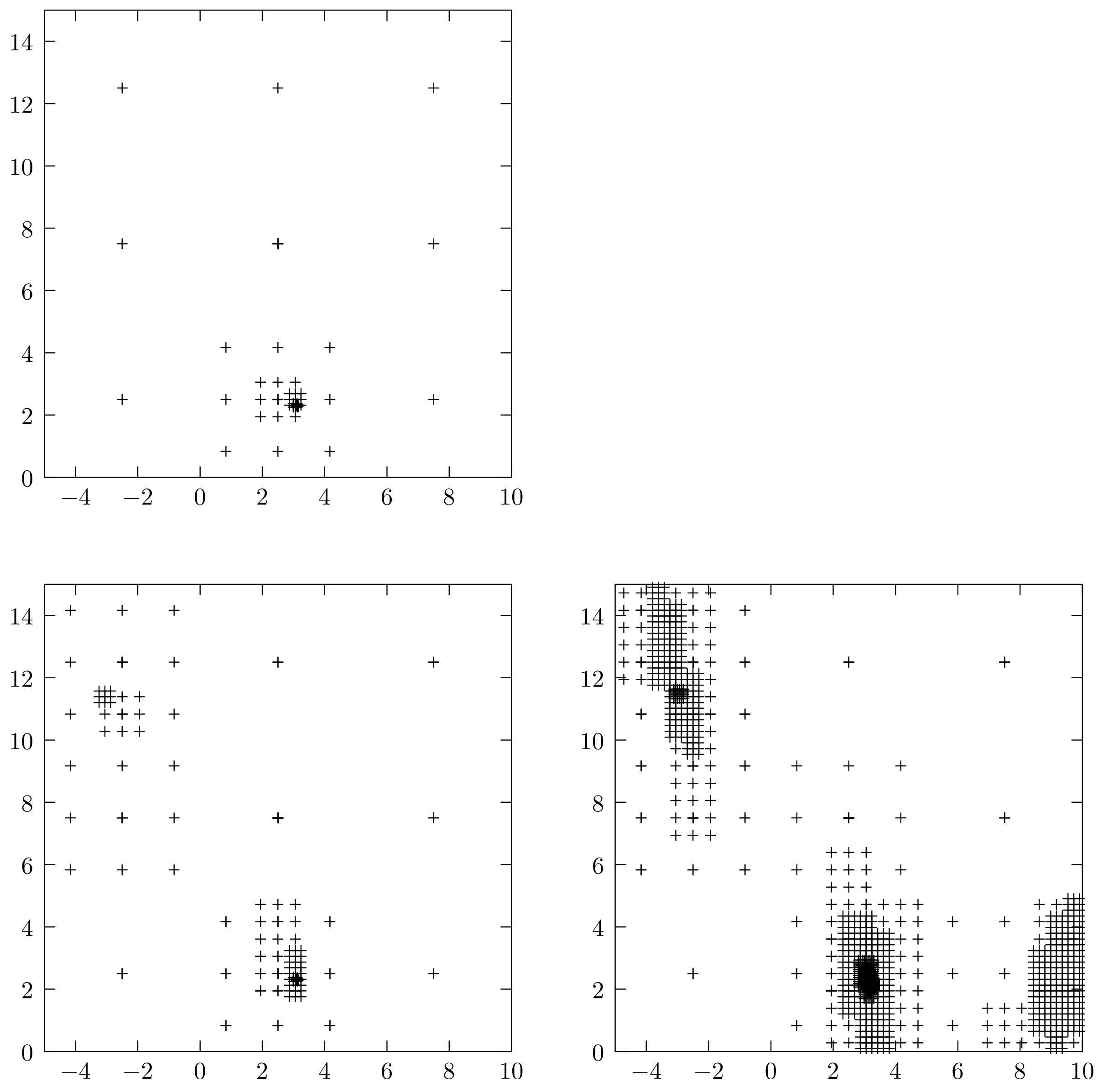

Fig. 4. The knots chosen by $Z(4, 10)$, $Z(4, 25)$, $Z(4, 29)$ for f_{BR}

k	cost	x_*	f_*
10	75	(3.14015, 2.28433)	0.3979647
25	4455	(3.14031, 2.28395)	0.3979585
29	11864	(3.14243, 2.27366)	0.3978912

Fig. 5. Results for $Z(4, k)$ and f_{BR}

References

Bakvalov, N. S. (1959), On approximate computation of multiple integrals (in Russian), in *Vestnik Moscow Univ. Ser. I Mat. Mekh.* **4**, pp. 3-18.

Calvin, J. M. (2004), Lower bounds on complexity of optimization of continuous functions, in *Journal of Complexity,* **20**, pp. 773-795.

Jones, D. R., Perttunen, C. D., and Stuckman, B. E. (1993), Lipschitzian optimization without the Lipschitz constant, in *Journal of Optimization Theory and Application* **79**, No. 1, pp. 157-181.

Nemirovski, A. (1995), Polynomial time methods in convex programming, in *AMS-SIAM Summer Seminar in Applied Mathematics (1995: Park City, Utah),* pp. 543-589. American Mathmatical Society, Providence.

Novak, E. (1988), *Deterministic and Stochastic Error Bounds in Numerical Analysis,* Springer Lecture Notes in Mathematics **1349**, Berlin.

Novak, E. (1995), The real number model in numerical analysis, in *Journal of Complexity* **11**, pp. 57-73.

Perevozchikov, A. G. (1990), The complexity of the computation of the global extremum in a class of multi-extremum problems, in *U.S.S.R. Comp. Maths. Math. Phys.* **30**, No. 2, pp. 28-33.

Ritter, K. (1990), Approximation on the Wiener space, in *Journal of Complexity,* **6**, pp. 337-364.

Törn, A., and Žilinskas, A. (1989), *Global Optimization,* Springer Lecture Notes in Computer Science **350**, Berlin.