Global Optimization with Space-Filling Curves

B. GOERTZEL
Intelligenesis Corp.
50 Broadway, New York, NY 10004, U.S.A.

(Received October 1997; accepted November 1997)

Abstract—It is shown that, contrary to a claim of Törn and Zilinskas, it is possible to efficiently optimize functions on n dimensions by projecting them into a single dimension using a space-filling curve. © 1999 Elsevier Science Ltd. All rights reserved.

1. INTRODUCTION

Suppose one is confronted with the problem of maximizing a function $f : [0, 1]^d \rightarrow R$. Given a space-filling curve $g : R \rightarrow [0, 1]^d$, one may "project" one's problem into one dimension by asking for the maximum of $f(g(x))$. The function $f \circ g$ maps $[0, 1]$ into $R$, and may thus be maximized by a suitable one-dimensional optimization routine. If $x$ is a maximizer of $f \circ g$, $g(x)$ is a maximizer of $f$.

As reported in [1,2], this procedure has been implemented in two, three, and four dimensions with some success. But Törn and Zilinskas [3], in their survey of global optimization methods, express doubts as to the general effectiveness of space-filling curve techniques. Their objection is that even if $f$ is convex, $f \circ g$ will be horribly ill-behaved—it will have local optima in every subinterval of $[0, 1]$.

The aim of this note is to show that the objection of Törn and Zilinskas is not necessarily relevant. From the fact that $f \circ g$ has many more local optima than $f$, it does not follow that $f \circ g$ is more difficult to optimize than $f$. Of course, for some optimization methods—e.g., those involving local search—$f \circ g$ will be vastly more troublesome than $f$. But for methods which are robust with respect to multiple local extrema and fuzzy data, this should not be the case.

Nemirovsky and Yudin [4] have described an optimal method for the "black box" optimization of a Lipschitz continuous function. Here I show that, for appropriate $g$, another optimal method is to apply the method described by Nemirovsky and Yudin to $f \circ g$ in $[0, 1]$, and "project" the answer back into $[0, 1]^d$.

2. SUBDIVIDING SPACE-FILLING CURVES

In general, a space-filling curve may be defined as a continuous function $g$ which maps some subset of $R$ into $R^d$, the range of which contains some $d$-dimensional sphere. Here, in particular, we will be concerned with space-filling curves that map $[0, 1]$ onto $[0, 1]^d$. 

0893-9659/99/$ - see front matter. © 1999 Elsevier Science Ltd. All rights reserved. Typeset by A\textsc{M}S-\textsc{T}EX
Space-filling curves come in all shapes and sizes. But the ones that will interest us here have a special property: they are what I call subdividing space-filling curves.

**Definition 1.** A continuous map \( g : [0, 1] \to [0, 1]^d \) is subdividing if for \( k = 1, 2, 3, \ldots, 0 < i < k, \)

\[
\left( \left[ \frac{i}{2^dk}, \frac{i+1}{2^dk} \right] \right)
\]

is a \( d \)-dimensional hypercube,

\( \quad (1) \)

\[
g \left( \left[ \frac{i}{2^dk}, \frac{i+1}{2^dk} \right] \right) = \bigcup_{j=0}^{2^{d-1}} g \left( \left[ \frac{i}{2^d(k+1)}, \frac{i + j + 1}{2^d(k+1)} \right] \right),
\]

\( \quad (2) \)

the center of \( g \left( \left[ \frac{i}{2^dk}, \frac{i+1}{2^dk} \right] \right) \) is \( g \left( \frac{i}{2^dk} + \frac{1}{2^dk+1} \right). \)

\( \quad (3) \)

A good example is the Hilbert curve—not only is it subdividing, but Fisher [5] has given an extremely rapid computer algorithm for generating it and its inverse.

**Lemma 1.** Let \( g \) be a subdividing space-filling curve with range \([0, 1]^d\). Then, for any positive integer \( k \), and any integer \( 0 < i < 2^k \), we have

\[
\sup_{x,y \in \left[ \frac{i}{2^dk}, \frac{i+1}{2^dk} \right]} \|g(x) - (y)\| \leq \frac{\sqrt{2}}{2^k},
\]

\( \quad (1) \)

\[
\sup_{x,y \in \left[ \frac{i}{2^dk}, \frac{i+1}{2^dk} \right]} \left\| g(x) - g \left( \frac{i}{2^dk} + \frac{1}{2^dk+1} \right) \right\| \leq \frac{1}{2^k}.
\]

\( \quad (2) \)

**Proof.** Part (1) is clear from the fact that \( g \) is subdividing. All the numbers in \( \left( \frac{i}{2^dk}, \frac{i+1}{2^dk} \right) \) are mapped into a hypercube of side \( 1/2^k \), the diameter of which is \( \sqrt{2}/2^k \). As for part (2), it suffices to recall that \( g(i/2^dk + 1/2^dk+1) \) is the center of this hypercube, from which all points in the hypercube are at a distance of less than \( 1/2^k \).

**3. Optimal Optimization of Lipschitz Functions**

One method of approximating the maximum of \( f : [0, 1]^d \to \mathbb{R} \) is to subdivide \([0, 1]^n\) into equally sized hypercubes, evaluate \( f \) at the center of each hypercube, and take the maximum of these values as one’s approximation. If \( f \) is Lipschitz with constant \( L \), then from \( N = 2^dk \) evaluations of \( f \), this method yields an error bounded by \( \sqrt{2} L N^{-1/d} \), where \( L \) is the Lipschitz constant of \( f \).

Using the sophisticated machinery of information-based complexity theory, Nemirovsky and Yudin [4] have shown that if the only fact known about \( f \) is that it is Lipschitz, then this simple method is “optimal”. For a precise definition of optimality as it is used here, see [4,6]. Roughly, the idea is as follows. Where \( f : [0, 1]^d \to \mathbb{R} \) and \( A \) is some (deterministic) optimization algorithm, let \( A_n(f) \) denote the approximation to the maximum of \( f \) which \( A \) supplies when allowed \( n \) evaluations of \( f \). Let \( \epsilon_{A,n}(f) = \|A_n(f) - y_f\| \), where \( y_f \) is the true maximum of \( f \). Then, where \( C \) is some class of functions from \([0, 1]^d\) to \( \mathbb{R} \), \( A \) is said to be optimal over \( C \) if for every other algorithm \( B \), there is some constant \( k_B \) so that

\[
\lim_{N \to \infty} \sup_{f \in C} \epsilon_{B,N}(f) \geq k_B \lim_{N \to \infty} \sup_{f \in C} \epsilon_{A,N}(f).
\]

Here the class \( C \) is the class of Lipschitz functions, and it is intuitively tempting to conjecture that \( k_B \) can be set equal to 1 for all algorithms \( B \).
However, Theorem 1 shows that if \( g : [0, 1] \rightarrow [0, 1]^d \) is subdividing, and one approximates the maximum of \( f \circ g \) by subdividing \([0, 1]\) into \( N = 2^{dk} \) equally sized subintervals, evaluating \( f \) at the center of each one, and taking the maximum of these values, then one's error will obey

\[
\sup_{x,y \in [i/2^{dk},(i+1)/2^{dk}]} \left\| f \circ g(x) - f \circ g \left( \frac{i}{2^{dk}} + \frac{1}{2^{dk}+1} \right) \right\| \\
\leq \sqrt{2} 2^{-kL} \sup_{x,y \in [i/2^{dk},(i+1)/2^{dk}]} \left\| g(x) - g \left( \frac{i}{2^{dk}+1} \right) \right\| \\
\leq \sqrt{2} L N^{-1/d}.
\]

This establishes the following theorem.

**Theorem 1.** If the only fact known about \( f : [0, 1]^d \rightarrow \mathbb{R} \) is that it is Lipschitz, then an optimal strategy for maximizing \( f \) is to maximize \( f \circ g \) for some subdividing space-filling curve \( g : [0, 1] \rightarrow [0, 1]^d \).

Note that this optimality property refers to the number of evaluations of \( f \) required.

In practice, it takes some extra work to evaluate \( f \circ g \) instead of just \( f \). But this extra work does not represent "optimization complexity", only "overhead" that is independent of \( f \), and it becomes negligible as the difficulty of evaluating \( f \) increases.

**REFERENCES**