Algorithm 825: A Deep-Cut Bisection Envelope Algorithm For Fixed Points

SPENCER SHELLMAN and K. SIKORSKI
University of Utah

We present the BEDFix (Bisection Envelope Deep-cut Fixed point) algorithm for the problem of approximating a fixed point of a function of two variables. The function must be Lipschitz continuous with constant 1 with respect to the infinity norm; such functions are commonly found in economics and game theory. The computed approximation satisfies a residual criterion given a specified error tolerance. The BEDFix algorithm improves the BEFix algorithm presented in Shellman and Sikorski [2002] by utilizing “deep cuts,” that is, eliminating additional segments of the feasible domain which cannot contain a fixed point. The upper bound on the number of required function evaluations is the same for BEDFix and BEFix, but our numerical tests indicate that BEDFix significantly improves the average-case performance. In addition, we show how BEDFix may be used to solve the absolute criterion fixed point problem with significantly better performance than the simple iteration method, when the Lipschitz constant is less than but close to 1. BEDFix is highly efficient when used to compute residual solutions for bivariate functions, having a bound on function evaluations that is twice the logarithm of the reciprocal of the tolerance. In the tests described in this article, the number of evaluations performed by the method averaged 31 percent of this worst-case bound. BEDFix works for nonsmooth continuous functions, unlike methods that require gradient information; also, it handles functions with minimum Lipschitz constants equal to 1, whereas the complexity of simple iteration approaches infinity as the minimum Lipschitz constant approaches 1. When BEDFix is used to compute absolute criterion solutions, the worst-case complexity depends on the logarithm of the reciprocal of 1-q, where q is the Lipschitz constant, as well as on the logarithm of the reciprocal of the tolerance.

Categories and Subject Descriptors: G.1.3 [**Numerical Analysis**]: Numerical Linear Algebra

General Terms: Algorithms, Performance

Additional Key Words and Phrases: Fixed points, economics, game theory, nonlinear partial differential equations

1. INTRODUCTION

The development of constructive algorithms for approximating fixed points started in the 1920’s with Banach’s simple iteration algorithm [Sikorski 2000]. Several algorithms have been developed since then, including homotopy

Authors’ address: School of Computing, University of Utah, Salt Lake City, UT 84112; email: shellman@losalamos.com; sikorski@cs.utah.edu.

continuation, simplicial and Newton type methods [Allgower and Georg 1990; Eaves 1972; Eaves and Saigal 1972; Scarf 1967]. It has been shown [Hirsch et al. 1989] that for Lipschitz functions with constant $q > 1$ with respect to the infinity norm, the latter algorithms exhibit exponential complexity in the worst case (when computing ϵ-residual solutions $\tilde{x} : \|\tilde{x} - f(\tilde{x})\|_\infty \le \epsilon$) and that the lower bound on the complexity is also exponential.

In our research, we have developed algorithms for approximating a fixed point x^* of a Lipschitz function that is contractive ($q < 1$) or nonexpanding ($q = 1$) with respect to the second norm. For $q < 1$ and large dimension d, the Banach simple iteration algorithm $x_{i+1} = f(x_i)$ is optimal [Nemirovsky 1991]. It requires $n = \lceil \log(1/\epsilon)/\log(1/q) \rceil$ function evaluations to compute $\tilde{x}$ such that $\|\tilde{x} - x^*\|_2 \le \epsilon \|x^*\|_2$. In the univariate case a class of bisection-envelope algorithms is optimal with respect to various error criteria [Sikorski 2000]. For moderate dimension d and $q = 1$, the interior ellipsoid algorithm is optimal [Huang et al. 1999; A. Nemirovsky (Private communication)]. This algorithm requires $c \cdot d \cdot \log(1/\epsilon)$ function evaluations to compute an ϵ-residual approximation $\tilde{x} : \|\tilde{x} - f(\tilde{x})\|_2 \le \epsilon$. We stress that the worst-case complexity of computing an ϵ-absolute approximation $\tilde{x} : \|\tilde{x} - x^*\|_2 \le \epsilon$ for $q = 1$ is infinite [Sikorski 2000]. This means that there exists no algorithm based on function evaluations that solves this problem for all functions in this class. For $q < 1$ the interior ellipsoid algorithm [Huang et al. 1999] computes $\tilde{x} : \|\tilde{x} - x^*\| \le \epsilon$ within $c \cdot d(\log(1/\epsilon) + \log(1/(1-q)))$ function evaluations.

The above results hold for the case of the second norm. The case of the infinity norm and $q \le 1$ requires further research. In Shellman and Sikorski [2002], we presented a two-dimensional bisection-envelope algorithm that exhibits worst-case complexity $2\lceil \log_2(1/\epsilon) \rceil + 1$ for the residual error criterion, $q = 1$ and the infinity norm. In this article, we present an improved algorithm that has the same worst case complexity, but improves the average case complexity by eliminating larger segments of the feasible domain at each step. We believe that the worst-case complexity of BEDFix is optimal, and conjecture that the optimal d-dimensional lower bound is $\mathcal{O}(c(d)\log(1/\epsilon))$, where $c(d)$ is a polynomial in d with small degree. A subsequent conclusion is that the complexity with the absolute error criterion, $q < 1$, and small d is at most $\mathcal{O}(c(d)(\log(1/\epsilon) + \log(1/(1-q))))$.

The BEDFix algorithm is an alternative to the simple iteration method $x_{i+1} = f(x_i)$ when $q < 1$. We compare the performance of BEDFix, using the absolute criterion, with the complexity bounds of simple iteration for a number of test problems. We find that our method is considerably more efficient than simple iteration in the case $q < 1$, and still computes residual solutions in the case $q = 1$ where simple iteration is not guaranteed to converge.

Our method is applicable to difficult two-dimensional fixed point problems, for which the corresponding functions are mildly contractive (q close to 1) or nonexpanding ($q = 1$). Such problems arise in many disciplines, including economics, game theory, meromorphic functions [Howland and Vaillancourt 1985], and dynamical systems [Vrahatis 1995]. In the study of dynamical systems with two degrees of freedom, such fixed point problems model conservative or dissipative systems depending on whether the mapping is area-preserving or area-contracting, respectively [Birkhoff 1917; Bountis and Helleman 1981; Greene 1979; Helleman 1977; Hénon 1969]. In the theory of dynamical systems, the fixed points are called "periodic orbits" of such mappings [Verhulst 1990].

2. PROBLEM FORMULATION

Given $a, b \in \Re$ with $a < b$, we define the domain $D_{a,b} \equiv [a, b]^2$ and consider the class of Lipschitz continuous functions

$$\mathcal{F}_{a,b} \equiv \{f : D_{a,b} \to D_{a,b} : \|f(x) - f(y)\| \leq \|x - y\| \; \forall\, x, y \in D_{a,b}\}, \tag{1}$$

where $\|\cdot\| = \|\cdot\|_\infty$, henceforth. We define $D \equiv D_{0,1}$ and $\mathcal{F} \equiv \mathcal{F}_{0,1}$. Any $f \in \mathcal{F}_{a,b}$ maps $D_{a,b}$ into $D_{a,b}$, so by the Brouwer fixed point theorem there exists $x^* \in D_{a,b}$ such that $f(x^*) = x^*$.

In this article, we present an algorithm that, for every $f \in \mathcal{F}$ and for a given positive $\epsilon < 0.5$, computes a solution $\tilde{x} = \tilde{x}(f) \in D$ satisfying the residual criterion,

$$\|f(\tilde{x}) - \tilde{x}\| \leq \epsilon. \tag{2}$$

Under certain conditions the computed solution also satisfies the absolute criterion,

$$\|\tilde{x} - x^*\| \leq \epsilon. \tag{3}$$

(If $\epsilon \geq 0.5$, then $\tilde{x} = (0.5, 0.5)$ clearly satisfies (2) and (3)). The algorithm requires n evaluations of f, where

$$1 \leq n \leq 2\lceil \log_2(1/\epsilon) \rceil + 1. \tag{4}$$

3. PRELIMINARY RESULTS

In this section, we summarize some fixed point properties of the functions in $\mathcal{F}_{a,b}$. The proofs of these theorems are found in Shellman and Sikorski [2002].

3.1 Envelope Theorem

For a given $f \in \mathcal{F}_{a,b}$, we define the fixed point sets

$$F_1(f) \equiv \{x \in \mathrm{Dom}(f) : f_1(x) = x_1\},$$

$$F_2(f) \equiv \{x \in \mathrm{Dom}(f) : f_2(x) = x_2\},$$

$$F(f) \equiv F_1(f) \cap F_2(f).$$

We observe that $F(f)$ is the set of all fixed points of f, and that $F(f) \neq \emptyset$.

In Figure 1, given $f \in \mathcal{F}$, we illustrate envelopes of the graph of f_1 after it is evaluated at some point $x' \in D$. The variables in the graph are $z = (z_1, z_2, z_3) \in [0, 1]^3$. The plane L consists of the points $\{z \in [0, 1]^3 : z_1 = z_3\}$. We assume that $f_1(x') > x'_1$, so that the point $z' = (x'_1, x'_2, f_1(x'))$ lies above L. We define the set $P \equiv \{z \in [0, 1]^3 : z'_3 - z_3 \geq \|(z'_1, z'_2) - (z_1, z_2)\|\}$, which resembles a pyramid descending from z'. By Lipschitz continuity of f, $\mathrm{Int}(P)$ cannot contain a point z such that $z_3 = f_1(z_1, z_2)$. It follows that, if $z \in \mathrm{Int}(P) \cap L$, then

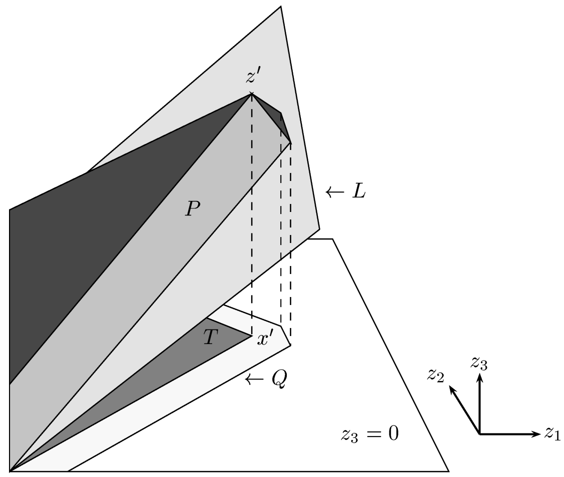


Fig. 1. Graphical representation of Theorem 3.1.

$z_1 \neq f_1(z_1, z_2)$. We let Q be the projection onto the (x_1, x_2)-plane of the intersection of P and L. Then, for all $x \in \mathrm{Int}(Q)$, $f_1(x) \neq x_1$. Clearly, $x' \in Q$. Let T be the triangular set $\{x : |x'_2 - x_2| \leq x'_1 - x_1\}$. Then, $T \subset Q$. This fixed point property is formalized in Theorem 3.1, the "envelope theorem".

THEOREM 3.1. *For any $f \in \mathcal{F}_{a,b}$ and any permutation (i, j) of $(1, 2)$, let $x \in D_{a,b}$ be such that $f_i(x) \neq x_i$. Then*

(i) *If $f_i(x) > x_i$, then, for every $y \in D_{a,b}$ such that $\|y - x\| < (f_i(x) - x_i)/2$, the set*

$$A_i(y) \equiv \{z \in D_{a,b} : |y_j - z_j| \leq y_i - z_i\}$$

does not intersect $F_i(f)$.

(ii) *If $f_i(x) < x_i$, then, for every $y \in D_{a,b}$ such that $\|y - x\| < (x_i - f_i(x))/2$, the set*

$$B_i(y) \equiv \{z \in D_{a,b} : |z_j - y_j| \leq z_i - y_i\}$$

does not intersect $F_i(f)$.

The BEDFix algorithm differs from the BEFix algorithm in that while BEFix subdivides the feasible domain along the boundary of T at each step, BEDFix subdivides it along the boundary of a larger domain U, $T \subset U \subseteq Q$.

Figure 2 illustrates the "envelope" sets $A_1(x)$, $A_2(x)$, $B_1(x)$, and $B_2(x)$ defined in Theorem 3.1. Henceforth, we assume the grid directions shown in this figure, that is, for $x \in D_{a,b}$, x_1 increases to the right and x_2 increases upward.

The following corollary shows that if $x \in D_{a,b}$ is in $F_1(f)$ or $F_2(f)$ but not both, then we can choose an envelope set of x that is guaranteed to contain a fixed point of f.

COROLLARY 3.2. *Let (i, j) be a permutation of $(1, 2)$. Take $x \in F_i(f)$ such that $f_j(x) < x_j$ (respectively, $f_j(x) > x_j$), and suppose there exists $y \in F(f)$ such that $y \notin A_j(x)$ (respectively, $y \notin B_j(x)$). Let E be the closed edge in*

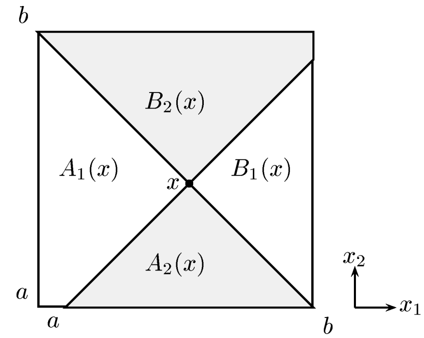


Fig. 2. Envelopes $A_1(x)$, $A_2(x)$, $B_1(x)$, and $B_2(x)$.

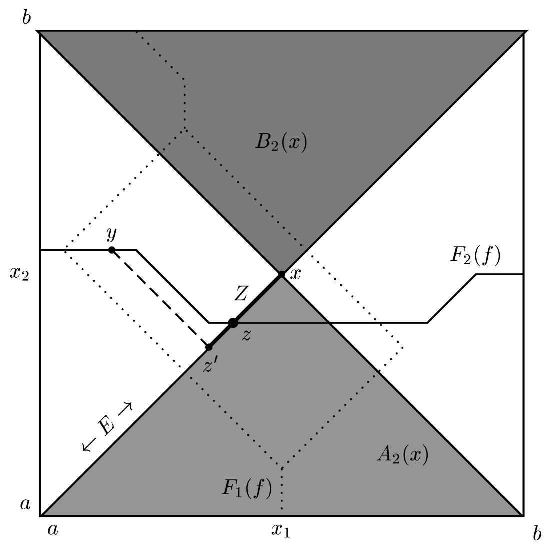


Fig. 3. Corollary 3.2. $f_2(x) < x_2$, $y \in F(f)$, and $F(f)$ intersects Z at z.

$\partial A_j(x)$ (respectively, $\partial B_j(x)$) such that for every $e \in E$, the open-line segment with endpoints at e and y does not intersect $A_j(x)$ (respectively, $B_j(x)$). Define $Z \equiv Z(x, y) \equiv \{z \in E : |z_j - y_j| \le |z_i - y_i|\}$. Then, $F(f) \cap Z \neq \emptyset$. Since $F(f) \neq \emptyset$, we conclude that $F(f) \cap A_j(x) \neq \emptyset$ (respectively, $F(f) \cap B_j(x) \neq \emptyset$) (see Figure 3).

The following corollary shows that if $x \in D_{a,b}$ is in $F_1(f)$ or $F_2(f)$ but not both, $y \in D_{a,b}$ is a fixed point of f, and both x and y are contained in a closed rectangle D' whose sides have slope 1 or -1, then we can choose a certain envelope set of x whose intersection with D' is guaranteed to contain a fixed point of f.

COROLLARY 3.3. *Let (i, j) be a permutation of $(1, 2)$. Let $D' \subset D_{a,b}$ be a closed rectangle whose sides have slope 1 or -1, and suppose there exists $y \in D' \cap F(f)$. Let $x \in D' \cap F_i(f)$ be such that $f_j(x) < x_j$ (respectively, $f_j(x) > x_j$). Then, $D' \cap A_j(x) \cap F(f) \neq \emptyset$ (respectively, $D' \cap B_j(x) \cap F(f) \neq \emptyset$).*

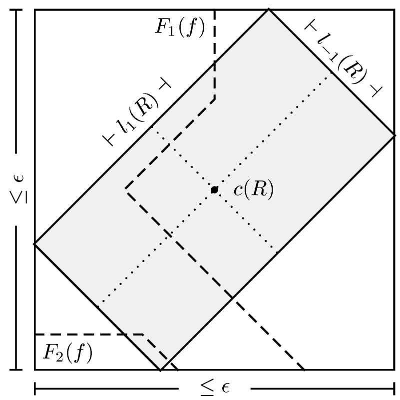


Fig. 4. R satisfies (5) so $c(R)$ satisfies (2).

3.2 Extension Theorem

Given $a', b' \in \Re$ with $a' < a$ and $b' > b$, and a function $f \in \mathcal{F}_{a,b}$, we define an extended function $\bar{f} : D_{a',b'} \to D_{a,b}$ as $\bar{f}(x) = f(P(x))$, where the projection P is given by

$$P(x) \equiv (\max(a, \min(b, x_1)), \max(a, \min(b, x_2))).$$

The following are true:

—The function $\bar{f}$ is Lipschitz continuous with constant 1 on $D_{a',b'}$, $\bar{f}$ has at least one fixed point, and $D_{a,b}$ contains all fixed points of $\bar{f}$.

—Let $\tilde{y} \in D_{a',b'}$ be such that $\|\bar{f}(\tilde{y}) - \tilde{y}\| \leq \epsilon$. Then, $\|f(\tilde{x}) - \tilde{x}\| \leq \epsilon$, where $\tilde{x} = P(\tilde{y})$.

—Let $\tilde{y} \in D_{a',b'}$ be such that $\|\tilde{y} - x^*\| \leq \epsilon$ for some fixed point x^* of $\bar{f}$. Then $\|\tilde{x} - x^*\| \leq \epsilon$, where $\tilde{x} = P(\tilde{y})$.

3.3 Termination Theorem

Given a closed rectangle R whose sides have slope 1 or -1, we define $l_1(R)$ (respectively, $l_{-1}(R)$) as the length of a side of R with slope 1 (respectively, -1). In addition, we define $l_{\min}(R) \equiv \min(l_1(R), l_{-1}(R))$ and $l_{\max}(R) \equiv \max(l_1(R), l_{-1}(R))$. We define $c(R) \equiv (c_1(R), c_2(R))$ as the center of gravity of R. Obviously, a line with slope $s \in \{1, -1\}$ through $c(R)$ divides R into two halves R_1, R_2 such that $l_{-s}(R_1) = l_{-s}(R_2) = l_{-s}(R)/2$.

The following theorem establishes conditions under which $c(R)$ satisfies the absolute or residual criterion.

THEOREM 3.4. *Let $R \subset D_{a,b}$ be a rectangle as above, such that*

$$l_1(R) + l_{-1}(R) \leq \sqrt{2}\epsilon \tag{5}$$

and both $F_1(f)$ and $F_2(f)$ intersect R. Then $\tilde{y} = c(R)$ satisfies (2)*. If, in addition, R contains a fixed point of f, then $\tilde{y}$ satisfies* (3) (*see Figure 4*).

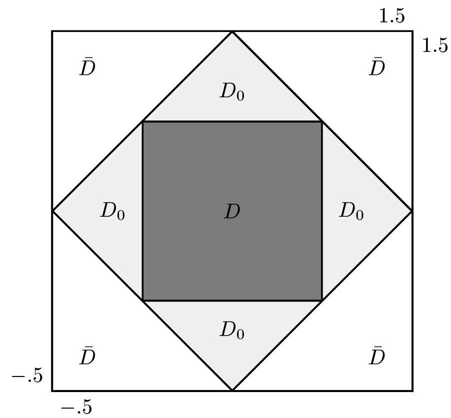


Fig. 5. D, D_0, and $\bar{D}$.

4. THE BEDFIX ALGORITHM

In this section, we provide necessary definitions, describe the algorithm, and list its pseudocode.

4.1 Definitions

We define the domains $\bar{D} \equiv D_{-0.5,1.5}$ and $D_0 \equiv \{x \in \Re^2 : \|x - (0.5, 0.5)\|_1 \leq 1\}$. Observe that D_0 is the smallest 1-norm ball containing D and the largest contained in $\bar{D}$ (see Figure 5). Henceforth, the sets A_1,A_2,B_1,B_2 are defined relative to $\bar{D}$.

Given a function $f \in \mathcal{F}$ we define its extension $\bar{f} : \bar{D} \to D$ as in Theorem 3.2. By this theorem, $\bar{f}$ is Lipschitz continuous with constant 1 on $\bar{D}$ and has at least one fixed point, and D contains all of its fixed points. Hence, if $\tilde{y}$ is a residual solution for $\bar{f}$, then $\tilde{x} = P(\tilde{y})$ is a residual solution for f, where P projects from $\bar{D}$ to D, that is,

$$P(x) \equiv (\max(0, \min(1, x_1)), \max(0, \min(1, x_2))).$$

4.2 Description

The algorithm computes $\tilde{y} \in D_0$ satisfying the residual criterion

$$\|\bar{f}(\tilde{y}) - \tilde{y}\| \leq \epsilon \tag{6}$$

and takes $\tilde{x} = P(\tilde{y})$ as a solution to (2).

The logical output variable *abs* will be true if $\tilde{y}$ (and by extension $\tilde{x}$) satisfies the absolute criterion

$$\|\tilde{y} - x^*\| \leq \epsilon \tag{7}$$

for some fixed point x^* of $\bar{f}$, according to the conditions of Theorem 3.4). There may be cases where $\tilde{y}$ satisfies (7) but *abs* is returned false. Section 5.3 explains how to guarantee that $\tilde{y}$ always satisfies (7) for contractive functions ($q < 1$).

The parameter *deep*, if true, uses “deep cuts” to achieve greater domain reduction in accordance with Theorem 3.1. If false, the algorithm behaves identically to the BEFix algorithm. This parameter is present in the algorithm to illustrate the difference between the BEFix and BEDFix algorithms.

The function $\mathrm{dist}(z, S)$ returns the minimal infinity-norm distance from z to an element of the closed set $S \in \Re^2$. This function does not need to be explicitly computed by the algorithm, but is used to describe the reduction of the feasible domain by a deep cut.

Step k of the algorithm ($k \geq 1$) evaluates $\bar{f}$ at x^k, the center of the rectangle D_{k-1}. It then constructs a rectangle $D_k \subset D_{k-1}$ by subdividing D_{k-1} along lines with slope 1 or -1, such that D_k has no more than half the area of D_{k-1}. Each D_k is a closed rectangle whose sides have slope 1 or -1 (this is clearly true of D_0). The algorithm terminates at step k if a residual criterion is satisfied at x^k, or if $l_1(D_k) + l_{-1}(D_k) \leq \sqrt{2}\epsilon$ so that the center of D_k satisfies the residual criterion by Theorem 3.4. At each step D_k is guaranteed to contain a fixed point of $\bar{f}$, so if D_k satisfies (5) then the center of D_k also satisfies the absolute criterion.

4.3 Pseudocode

```
k := 0;
deep := true;
repeat
   k := k + 1;
   D_k := D_{k-1};
   x^k := c(D_k);
   v_1 := \bar{f}_1(x^k) - x_1^k;
   v_2 := \bar{f}_2(x^k) - x_2^k;
   if v_1 = 0 and v_2 = 0 then
      terminate with \tilde{x} = P(x^k), abs = true;
   end(if)
   if max(|v_1|, |v_2|) \leq \epsilon then
      terminate with \tilde{x} = P(x^k), abs = false;
   end(if)
   ! Condition 1
   if l_{min}(D_{k-1}) > (\sqrt{2}/2)\epsilon then
      m_1 := min(|v_1|, |v_2|)/2;
      if v_1 > 0 and v_2 > 0 then
         if deep = true then
            D_k := {z \in D_k : dist(z, A_1(x^k) \cup A_2(x^k)) \geq m_1};
         else
            D_k := D_k \cap (B_1(x^k) \cup B_2(x^k));
         end(if)
      else if v_1 < 0 and v_2 > 0 then
         if deep = true then
            D_k := {z \in D_k : dist(z, B_1(x^k) \cup A_2(x^k)) \geq m_1};
         else
            D_k := D_k \cap (A_1(x^k) \cup B_2(x^k));
         end(if)
      else if v_1 < 0 and v_2 < 0 then
         if deep = true then
            D_k := {z \in D_k : dist(z, B_1(x^k) \cup B_2(x^k)) \geq m_1};
         else
            D_k := D_k \cap (A_1(x^k) \cup A_2(x^k));
         end(if)
      else if v_1 > 0 and v_2 < 0 then
         if deep = true then
            D_k := {z \in D_k : dist(z, A_1(x^k) \cup B_2(x^k)) \geq m_1};
```

```
            else
                $D_k := D_k \cap (B_1(x^k) \cup A_2(x^k))$;
            end(if)
        else if $v_1 = 0$ and $v_2 < 0$ then
            $D_k := D_k \cap A_2(x^k)$;
        else if $v_1 = 0$ and $v_2 > 0$ then
            $D_k := D_k \cap B_2(x^k)$;
        else if $v_1 < 0$ and $v_2 = 0$ then
            $D_k := D_k \cap A_1(x^k)$;
        else if $v_1 > 0$ and $v_2 = 0$ then
            $D_k := D_k \cap B_1(x^k)$;
    end(if)
    ! Condition 2
    if $l_1(D_{k-1}) \le \sqrt{2}\epsilon$ and $l_{-1}(D_{k-1}) > (\sqrt{2}/2)\epsilon$ then
        $m_2 := \max(|v_1|, |v_2|)/2 - (\sqrt{2}/4) l_1(D_{k-1})$;
        if $v_1 < -\epsilon$ or $v_2 > \epsilon$ then
          if $deep = true$ then
            $D_k := \{z \in D_k : \text{dist}(z, B_1(x^k) \cup A_2(x^k)) \ge m_2\}$;
          else
            $D_k := D_k \cap (A_1(x^k) \cup B_2(x^k))$;
          end(if)
        else
            if $deep = true$ then
              $D_k := \{z \in D_k : \text{dist}(z, A_1(x^k) \cup B_2(x^k)) \ge m_2\}$;
            else
                $D_k := D_k \cap (B_1(x^k) \cup A_2(x^k))$;
            end(if)
        end(if)
    ! Condition 3
    if $l_{-1}(D_{k-1}) \le \sqrt{2}\epsilon$ and $l_1(D_{k-1}) > (\sqrt{2}/2)\epsilon$ then
        $m_3 := \max(|v_1|, |v_2|)/2 - (\sqrt{2}/4) l_{-1}(D_{k-1})$;
        if $v_1 > \epsilon$ or $v_2 > \epsilon$ then
          if $deep = true$ then
            $D_k := \{z \in D_k : \text{dist}(z, A_1(x^k) \cup A_2(x^k)) \ge m_3\}$;
          else
            $D_k := D_k \cap (B_1(x^k) \cup B_2(x^k))$;
          end(if)
        else
            if $deep = true$ then
              $D_k := \{z \in D_k : \text{dist}(z, B_1(x^k) \cup B_2(x^k)) \ge m_3\}$;
            else
                $D_k := D_k \cap (A_1(x^k) \cup A_2(x^k))$;
            end(if)
        end(if)
    end(if)
until $l_1(D_k) + l_{-1}(D_k) \le \sqrt{2}\epsilon$;
terminate with $\tilde{x} = P(c(D_k))$, $abs$ =true;
```

5. ALGORITHM ANALYSIS

An analysis similar to that in Shellman and Sikorski [2002] shows that BEDFix computes $\tilde{y}$ satisfying (6) in at most $2\lceil \log_2(1/\epsilon) \rceil + 1$ loop iterations. Each iteration evaluates $\bar{f}$ (equivalently f) once, so we obtain the desired complexity

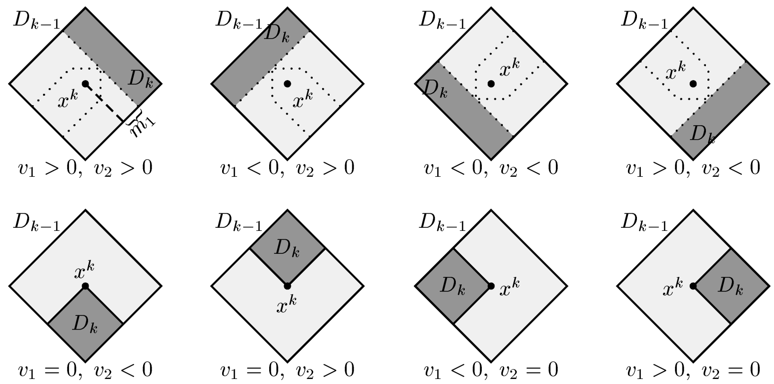


Fig. 6. Condition 1: Choice of D_k at step k. Dotted lines represent deep cuts.

(4). The BEDFix algorithm differs from BEFix only in the three conditions for domain reduction, which we analyze below for BEDFix.

Condition 1. $l_{\min}(D_{k-1}) > (\sqrt{2}/2)\epsilon$.

Here, $m_1 \equiv \min(|v_1|, |v_2|)/2$ represents the size of the deep cut; Theorem 3.1 allows a cut at any point y such that $\|y - x^k\| < \min(|v_1|, |v_2|)/2$.

See Figure 6 for an illustration of the following rules.

—If $v_1 > 0$ and $v_2 > 0$, then $D_k := \{z \in D_k : \text{dist}(z, A_1(x^k) \cup A_2(x^k)) \geq m_1\}$ if $deep = true$, $D_k := D_k \cap (B_1(x^k) \cup B_2(x^k))$, otherwise.
—If $v_1 < 0$ and $v_2 > 0$, then $D_k := \{z \in D_k : \text{dist}(z, B_1(x^k) \cup A_2(x^k)) \geq m_1\}$ if $deep = true$, $D_k := D_k \cap (A_1(x^k) \cup B_2(x^k))$, otherwise.
—If $v_1 < 0$ and $v_2 < 0$, then $D_k := \{z \in D_k : \text{dist}(z, B_1(x^k) \cup B_2(x^k)) \geq m_1\}$ if $deep = true$, $D_k := D_k \cap (A_1(x^k) \cup A_2(x^k))$, otherwise.
—If $v_1 > 0$ and $v_2 < 0$, then $D_k := \{z \in D_k : \text{dist}(z, A_1(x^k) \cup B_2(x^k)) \geq m_1\}$ if $deep = true$, $D_k := D_k \cap (B_1(x^k) \cup A_2(x^k))$, otherwise.
—If $v_1 = 0$ and $v_2 < 0$, then, $D_k := D_k \cap A_2(x^k)$.
—If $v_1 = 0$ and $v_2 > 0$, then, $D_k := D_k \cap B_2(x^k)$.
—If $v_1 < 0$ and $v_2 = 0$, then, $D_k := D_k \cap A_1(x^k)$.
—If $v_1 > 0$ and $v_2 = 0$, then, $D_k := D_k \cap B_1(x^k)$.

Condition 2. $l_1(D_{k-1}) \leq \sqrt{2}\epsilon$ and $l_{-1}(D_{k-1}) > (\sqrt{2}/2)\epsilon$.

We define the points

$$w^1 \equiv x^k - \left(\frac{\sqrt{2}}{4} l_1(D_{k-1}), \frac{\sqrt{2}}{4} l_1(D_{k-1})\right), \quad w^2 \equiv x^k + \left(\frac{\sqrt{2}}{4} l_1(D_{k-1}), \frac{\sqrt{2}}{4} l_1(D_{k-1})\right)$$

(see Figure 7); w^1 and w^2 are the midpoints of the edges of D_{k-1} with slope -1.

Here $m_2 \equiv \max(|v_1|, |v_2|)/2 - (\sqrt{2}/4) l_1(D_{k-1})$ represents the size of the deep cut allowed by Theorem 3.1. This depends on the facts that the deep cut region is limited by $\max(|v_1|, |v_2|)/2$ and must contain w^1 and w^2, and $\|w^i - x^k\| = (\sqrt{2}/4) l_1(D_{k-1})$ for $i = 1, 2$ (see Figure 7).

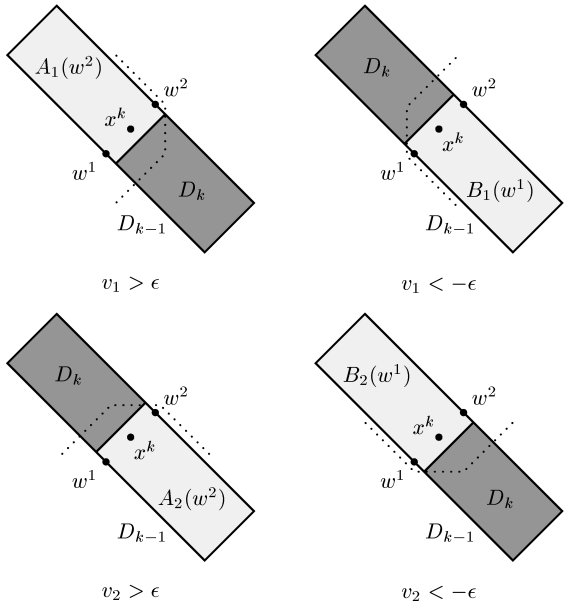


Fig. 7. Condition 2: Choice of D_k at step k. Dotted lines represent deep cuts.

If $v_1 > \epsilon$, then since $\|w^2 - x^k\| \le \epsilon/2 < v_1/2$, by Theorem 3.1 $A_1(w^2) \cap F_1(\bar{f}) = \emptyset$. Similarly, if $v_1 < -\epsilon$, then $B_1(w^1) \cap F_1(\bar{f}) = \emptyset$; if $v_2 > \epsilon$, then $A_2(w^2) \cap F_1(\bar{f}) = \emptyset$; if $v_2 < -\epsilon$, then $B_2(w^1) \cap F_1(\bar{f}) = \emptyset$. Hence, we obtain the following rules.

—If $v_1 < -\epsilon$ or $v_2 > \epsilon$, then $D_k := \{z \in D_k : \text{dist}(z, B_1(x^k) \cup A_2(x^k)) \ge m_2\}$ if *deep* = *true*, $D_k := D_k \cap (A_1(x^k) \cup B_2(x^k))$, otherwise.

—If $v_1 > \epsilon$ or $v_2 < -\epsilon$, then $D_k := \{z \in D_k : \text{dist}(z, A_1(x^k) \cup B_2(x^k)) \ge m_2\}$ if *deep* = *true*, $D_k := D_k \cap (A_2(x^k) \cup B_1(x^k))$, otherwise.

Condition 3. $l_{-1}(D_{k-1}) \le \sqrt{2}\epsilon$ and $l_1(D_{k-1}) > (\sqrt{2}/2)\epsilon$.

Here $m_3 \equiv \max(|v_1|, |v_2|)/2 - (\sqrt{2}/4) l_{-1}(D_{k-1})$ represents the size of the deep cut allowed by Theorem 3.1; the reasoning is analogous to that of m_2 in Condition 2.

We define the points

$$w^1 \equiv x^k + \left(-\frac{\sqrt{2}}{4} l_{-1}(D_{k-1}), \frac{\sqrt{2}}{4} l_{-1}(D_{k-1}) \right),$$
$$w^2 \equiv x^k + \left(\frac{\sqrt{2}}{4} l_{-1}(D_{k-1}), -\frac{\sqrt{2}}{4} l_{-1}(D_{k-1}) \right)$$

(see Figure 8); w^1 and w^2 are the midpoints of the edges of D_{k-1} with slope 1. If $v_1 > \epsilon$, then since $\|w^2 - x^k\| \le \epsilon/2 < v_1/2$, by Theorem 3.1 $A_1(w^2) \cap F_1(\bar{f}) = \emptyset$. Similarly, if $v_1 < -\epsilon$, then $B_1(w^1) \cap F_1(\bar{f}) = \emptyset$; if $v_2 > \epsilon$, then $A_2(w^1) \cap F_1(\bar{f}) = \emptyset$; if $v_2 < -\epsilon$, then $B_2(w^2) \cap F_1(\bar{f}) = \emptyset$. Hence, we obtain the following rules.

—If $v_1 > \epsilon$ or $v_2 > \epsilon$, then $D_k := \{z \in D_k : \text{dist}(z, A_1(x^k) \cup A_2(x^k)) \ge m_3\}$ if *deep* = *true*, $D_k := D_k \cap (B_1(x^k) \cup B_2(x^k))$, otherwise.

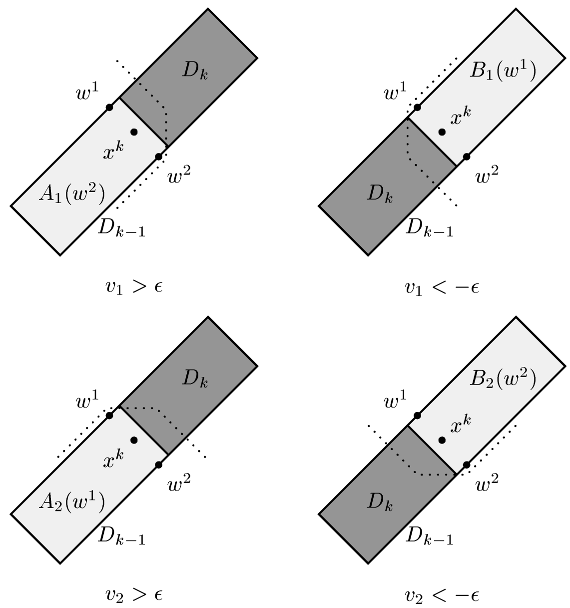


Fig. 8. Condition 3: Choice of D_k at step k. Dotted lines represent deep cuts.

—If $v_1 > \epsilon$ or $v_2 < -\epsilon$, then $D_k := \{z \in D_k : \text{dist}(z, B_1(x^k) \cup B_2(x^k)) \geq m_3\}$ if *deep* = *true*, $D_k := D_k \cap (A_1(x^k) \cup A_2(x^k))$, otherwise.

5.1 Termination

At the end of step k the algorithm terminates with $\tilde{x} = P(c(D_k))$ if D_k satisfies

$$l_1(D_k) + l_{-1}(D_k) \leq \sqrt{2}\epsilon, \tag{8}$$

or proceeds to the next step otherwise. By Theorem 3.4, if D_k satisfies (8), then $\tilde{y} = c(D_k)$ satisfies (6) and (7).

5.2 Complexity

At each step, the feasible domain resulting from the BEDFix algorithm is a subset of the feasible domain resulting from the BEFIX algorithm [Shellman and Sikorski 2002], so the same arguments may be used to establish the worst-case complexity of BEDFix. That is, D_k will satisfy $l_{\max}(D_k) \leq (\sqrt{2}/2)\epsilon$, and thus (8), while

$$k \leq 2\lceil \log_2(1/\epsilon) \rceil + 1.$$

5.3 Absolute Criterion with $q < 1$

We now show how to apply the BEDFix algorithm to the problem of computing $\tilde{x}$ satisfying (3) when f is Lipschitz continuous with constant $q < 1$, that is, for every $x, y \in D$, $\|f(x) - f(y)\| \leq q\|x - y\|$ with $0 < q < 1$. Since f is contractive, it has exactly one fixed point x^*. Furthermore,

$$\|\tilde{x} - x^*\| = \|\tilde{x} - f(\tilde{x}) + f(\tilde{x}) - x^*\| \le \|\tilde{x} - f(\tilde{x})\| + q\|\tilde{x} - x^*\|$$

so that

$$\|\tilde{x} - x^*\| \le \frac{1}{1-q}\|\tilde{x} - f(\tilde{x})\|.$$

Thus, if we use the BEDFix algorithm to compute $\tilde{x}$ satisfying the tolerance $(1-q)\epsilon$, the resulting $\tilde{x}$ satisfies (3). The upper bound on the number of required function evaluations is

$$2\left\lceil \log_2 \frac{1}{(1-q)\epsilon} \right\rceil + 1 = 2\left\lceil \log_2 \frac{1}{\epsilon} + \log_2 \frac{1}{1-q} \right\rceil + 1,$$

which has the same order as the complexity of the interior ellipsoid algorithm for the 2-norm case [Huang et al. 1999]. This complexity is much more desirable than that of the simple iteration method, which has complexity at most $\lceil \log(1/\epsilon)/\log(1/q) \rceil$.

6. NUMERICAL TESTS

As in Shellman and Sikorski [2002], we tested our algorithm on a number of pyramid functions, defined as follows. For $b \in D$ and $h \in [0,1]$, we define the pyramid basis function $P_b^h : D \to [0,1]$ as

$$P_b^h(x) \equiv \min(1, \max(h - q\|x - b\|_\infty, 0)), \tag{9}$$

where q is a Lipschitz constant. Figure 9 shows plots of P_b^h with $q = 1$ and for several values of b and h.

Our test functions are based on the following pyramid basis functions:

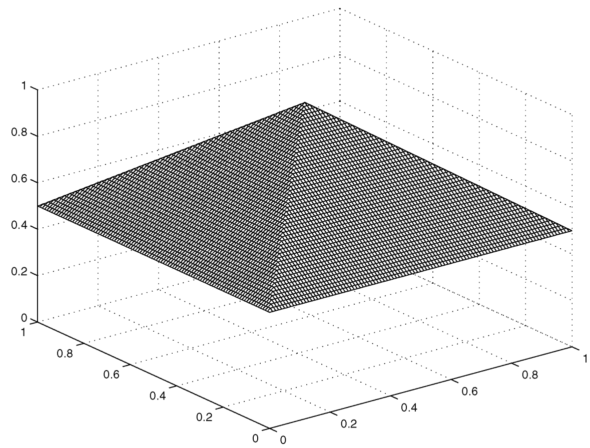
- $b_1 = (0.5, 0.5),\ h_1 = 0.8$
- $b_2 = (0.6, 0.4),\ h_2 = 1.2$
- $b_3 = (0.4, 0.6),\ h_3 = 0.9$
- $b_4 = (0.6, 0.98),\ h_4 = 0.99$
- $b_5 = (0.98, 0.3),\ h_5 = 0.99$
- $b_6 = (0.27, 0.64),\ h_6 = 1.01$
- $b_7 = (0.64, 0.27),\ h_7 = 0.99$
- $b_8 = (0, 0),\ h_8 = 0.1$

Given the distinct integers $i_1, \ldots, i_k,\ 1 \le k \le 8,\ 1 \le i_j \le 8\ \forall j$, we define the pyramid function

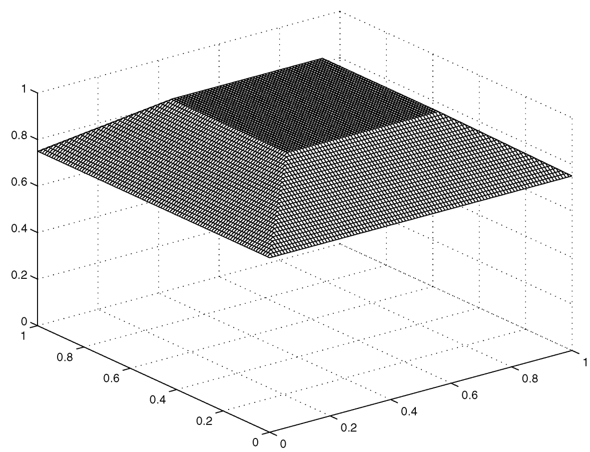
$$P_{i_1,\ldots,i_k}(x) \equiv \max\left(P_{b_{i_1}}^{h_{i_1}}(x), \ldots, P_{b_{i_k}}^{h_{i_k}}(x)\right). \tag{10}$$

Figure 10 shows the plot of $P_{5,6,7}(x)$ with $q = 1$.

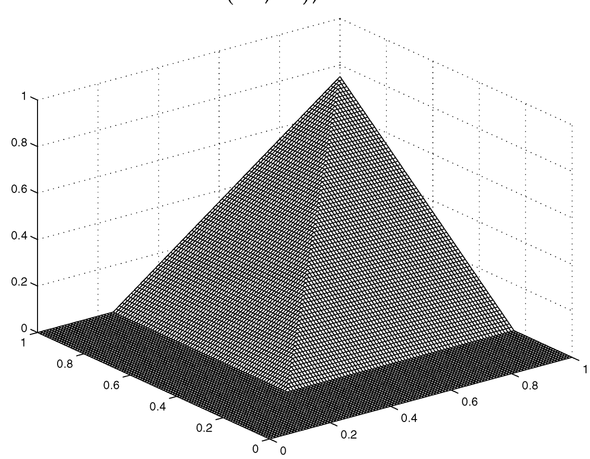
To compare BEDFix (deep cuts) with BEFix (no deep cuts) we tested the algorithm on the functions $f(x) = (P_{S_1}(x), P_{S_2}(x))$ for all pairs of nonempty subsets S_1 and S_2 of $\{1, \ldots, 8\}$, with $q = 1$, and with $\epsilon = 0.0001$. The total number of tests was 65,025, or $(2^8 - 1)^2$ (in Shellman and Sikorski [2002], we successfully



$b = (0.5, 0.5),\ h = 1$



$b = (0.5, 0.5),\ h = 1.25$



$b = (1, 1),\ h = .75$

Fig. 9. The pyramid basis function P_b^h.

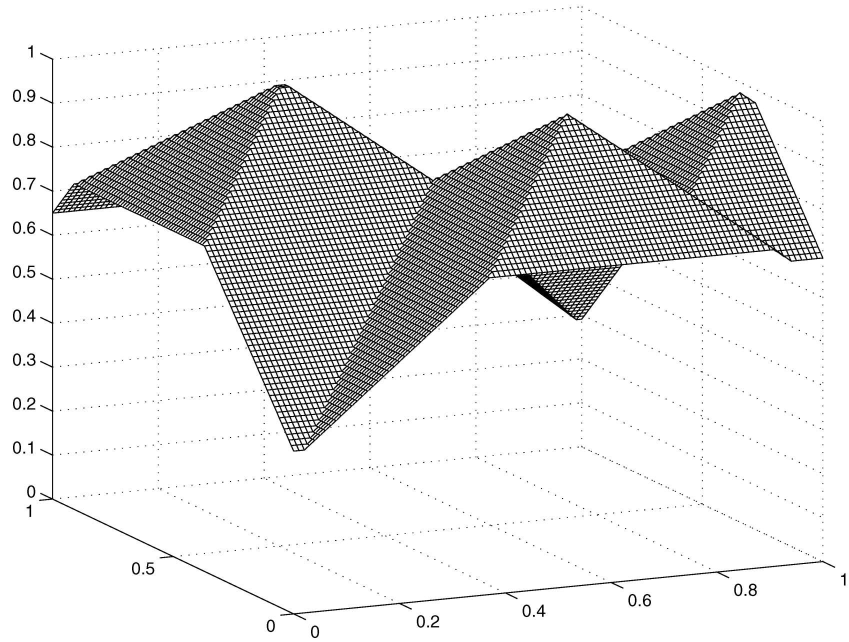


Fig. 10. The pyramid function $P_{5,6,7}$.

	BEDFix	BEFix
Average number of function evaluations per test	9.1	20.4
Average ratio of a test's function evaluations to the upper bound, $2\lceil \log_2(1/\epsilon)\rceil + 1 = 29$	0.314	0.704
Total number of tests satisfying the absolute error criterion	22,413	8
Average ratio of a test's function evaluations to $2\lceil \log_2(1/\epsilon)\rceil + 1 = 29$ for tests satisfying the absolute error criterion	0.42	0.97
Minimum number of function evaluations achieved by a test	3	11
Maximum number of function evaluations performed by a test	23	28

Fig. 11. Test results, BEDFix vs. BEFix.

executed $(2^{13} - 1)^2$ tests). All tests computed verifiable ϵ-residual solutions. The test results, summarized in Figure 11, indicate a significant improvement in average-case complexity using BEDFix. We recall that the "total number of tests satisfying the absolute error criterion" figure is only a minimum, and may not reflect all computed solutions that satisfy the absolute criterion.

To compare BEDFix with simple iteration, we tested the algorithm on the functions $f(x) = (P_{S_1}(x), P_{S_2}(x))$ for all pairs of nonempty subsets S_1 and S_2 of $\{1, \ldots, 8\}$, with varying $q < 1$ and with $\epsilon = 0.0001$. To ensure that BEDFix returned ϵ-absolute solutions, we executed it with tolerance $\epsilon(1 - q)$

	$q = 0.9$	$q = 0.99$	$q = 0.999$	Total cpu time for all tests
SI	84.58	904.28	9,088.39	41 m 29 s
BEDFix	16.35	16.52	16.65	16 s
Ratio SI/BEDFix	5.17	54.74	545.85	155.6

Fig. 12. Average number of function evaluations per test, BEDFix vs. SI.

(see Section 5.3). We executed the SI (simple iteration) method by setting $x^0 = (0, 5.0.5)$ and calling $x^{i+1} = f(x^k)$ $\lceil \log(1/\epsilon)/\log(1/q) \rceil$ times (at which the ϵ-relative criterion, as well as the ϵ-absolute criterion, are satisfied) or until $\| f(\cdot) - \cdot \| \leq \epsilon(1-q)$ (at which the ϵ-absolute criterion is satisfied according to Section 5.3). Figure 12 summarizes the results.[1]

7. FUTURE WORK

We plan to develop a version of our algorithm that works for any dimension $d \geq 2$. We believe that this extended algorithm will have optimal worst-case complexity in the class of algorithms that use information consisting of function evaluations. We conjecture that the complexity of the algorithm and the lower bound are $\mathcal{O}(c(d)\log(1/\epsilon))$, where $c(d)$ is a polynomial in d. The average case complexity should also be explored.

We believe that the absolute criterion problem, that is, the computation of $\tilde{x} \in D_d$ satisfying

$$\|\tilde{x} - x^*\| \leq \epsilon \tag{11}$$

where $x^* \in D_d$ is a fixed point of f, has infinite worst-case complexity when information consists of function evaluations. We plan to investigate restricted function classes that may make the complexity finite in the absolute criterion case.

ACKNOWLEDGMENTS

The authors would like to thank the referee of this article for making suggestions to improve the quality, and for executing the tests.

REFERENCES

ALLGOWER, G. AND GEORG, K. 1990. *Numerical Continuation Methods*. Springer-Verlag, New York.

BIRKHOFF, G. 1917. Dynamic systems with two degrees of freedom. *Trans. AMS 18*, 199.

BOUNTIS, T. AND HELLEMAN, R. 1981. On the stability of periodic orbits of two-dimensional mappings. *J. Math. Phys. 22*, 1867.

EAVES, B. 1972. Homotopies for computation of fixed points. *Math. Prog. 3*, 1, 1–22.

EAVES, B. AND SAIGAL, R. 1972. Homotopies for computation of fixed points on unbounded regions. *Math. Prog. 3*, 2, pp. 225–237.

GREENE, J. 1979. A method for determining a stochastic transition. *J. Math. Phys. 20*, 1183–1201.

HELLEMAN, R. 1977. *Statistical Mechanics and Statistical Methods in Theory and Application: On the Iterative Solution of a Stochastic Mapping*. U. Landman, Ed. Plenum, New York, 343.

HÉNON, M. 1969. Numerical study of quadratic area-preserving mappings. *Quart. Appl. Math. 27*, 291–312.

[1]The test times were measured on an 800 Mhz Pentium II system running Red Hat Linux 7.2.

HIRSCH, M., PAPADIMITRIOU, C., AND VAVASIS, S. 1989. Exponential lower bounds for finding Brouwer fixed points. *J. Complex. 5*, 379–416.

HOWLAND, J. AND VAILLANCOURT, R. 1985. Attractive cycles in the iteration of meromorphic functions. *Num. Math. 46*, 323–337.

HUANG, Z., KHACHIYAN, L., AND SIKORSKI, K. 1999. Approximating fixed points of weakly contracting mappings. *J. Complex. 15*, 200–213.

NEMIROVSKY, A. 1991. On the complexity of Krylov's information when solving linear operator equations. *J. Complex. 7*, 121–130.

SCARF, M. 1967. The approximation of fixed points of a continuous mapping. *SIAM J. Appl. Math.* 1328–1343.

SHELLMAN, S. AND SIKORSKI, K. 2002. A two-dimensional bisection envelope algorithm for fixed points. *J. Complex. 18*, 2 (June), 641–659.

SIKORSKI, K. 2000. *Optimal Solution of Nonlinear Equations*. Oxford Press, New York.

VERHULST, F. 1990. *Nonlinear Differential Equations and Dynamical Systems*. Springer-Verlag, Berlin/Heidelberg.

VRAHATIS, M. 1995. An efficient method for locating and computing periodic orbits of nonlinear mappings. *J. Comput. Phys. 119*, 105–119.

Received May 2001; revised December 2001 and May 2002; accepted October 2002