The Simplex Gradient and Noisy Optimization Problems*

D. M. Bortz
C. T. Kelley
North Carolina State University, Department of Mathematics
Center for Research in Scientific Computation
Box 8205, Raleigh, N. C. 27695-8205

Many classes of methods for noisy optimization problems are based on function information computed on sequences of simplices. The Nelder-Mead, multidirectional search, and implicit filtering methods are three such methods. The performance of these methods can be explained in terms of the difference approximation of the gradient implicit in the function evaluations. Insight can be gained into choice of termination criteria, detection of failure, and design of new methods.

1. Introduction

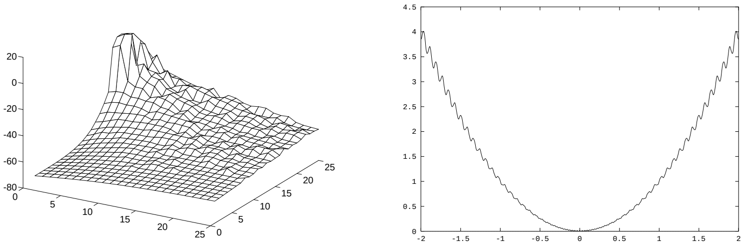
Noisy, nonsmooth, and discontinuous, optimization problems arise in many fields of science and engineering. A few of these are semiconductor modeling and manufacturing [23], [20], [24], [19], design and calibration of instruments, [13], design of wireless systems [10], and automotive engineering, [6], [5].

In this paper we consider objective functions that are perturbations of simple, smooth functions. The surface in on the left in Figure 1, taken from [24], and the graph on the right illustrate this type of problem.

The perturbations may be results of discontinuities or nonsmooth effects in the underlying models, randomness in the function evaluation, or experimental or measurement errors. Conventional gradient-based methods will be trapped in local minima even if the noise is smooth.

*This research was partially supported by National Science Foundation grant #DMS-9700569.

Figure 1: Optimization Landscapes



Many classes of methods for noisy optimization problems are based on function information computed on sequences of simplices. The Nelder-Mead, [18], multidirectional search, [8], [21], and implicit filtering, [12], methods are three examples. The performance of such methods can be explained in terms of the difference approximation of the gradient that is implicit in the function evaluations they perform.

In this paper we show how use of that gradient information can unify, extend, and simplify the analysis of these methods in the context of this important class of problems.

We begin by recalling the simplex gradient from [14], the first order estimates it satisfies, and its application to the Nelder-Mead method. In § 2 we show how this idea can be directly applied to the multi-directional search and implicit filtering algorithms in a way that allows for aggressive attempts to improve the performance and/or exploit parallelism.

The algorithms we discuss in this paper all examine a simplex of points in R^N at each iteration and the change the simplex in response. We consider problems where the objective f that is sampled is a perturbation of a smooth function f_s by a small function ϕ

$$f(x) = f_s(x) + \phi(x). \tag{1}$$

The small oscillations could cause f to have several local minima that would trap any conventional gradient-based algorithms. The perturbation ϕ can be random, [19], and therefore need not even be a function. We take $\phi \in L^\infty$ only to make the analysis simpler. The ideas in this section were originally used in [14] to analyze the Nelder-Mead, [18], algorithm and we will restate those results at the end of this section.

Definition 1 *A* **simplex** *S in R^N is the convex hull of $N+1$ points. $\{x_j\}_{j=1}^{N+1}$. x_j is the jth* **vertex** *of S. We let V (or $V(S)$) denote the $N \times N$ matrix of* **simplex directions**

$$V(S) = (x_2 - x_1, x_3 - x_1, \ldots, x_{N+1} - x_1) = (v_1, \ldots, v_N).$$

We say S is **nonsingular** *if V is nonsingular. The* **simplex diameter** *$diam(S)$ is*

$$diam(S) = \max_{1 \le i,j \le N+1} \|x_i - x_j\|.$$

We will refer to the l^2 condition number $\kappa(V)$ of V as the **simplex condition**.

We let $\delta(f : S)$ denote the vector of objective function differences

$$\delta(f : S) = (f(x_2) - f(x_1), f(x_3) - f(x_1), \ldots, f(x_{N+1}) - f(x_1))^T.$$

We will not use the simplex diameter directly in our estimates or algorithms. Rather we will use two **oriented lengths**

$$\sigma_+(V) = \max_{2 \le j \le N+1} \|x_1 - x_j\| \text{ and } \sigma_-(V) = \min_{2 \le j \le N+1} \|x_1 - x_j\|.$$

Clearly,

$$\sigma_+(S) \le \mathrm{diam}(S) \le 2\sigma_+(S).$$

Definition 2 *Let S be a nonsingular simplex with vertices*

$$\{x_j\}_{j=1}^N.$$

The **simplex gradient** *$D(f : S)$ is*

$$D(f : S) = V^{-T}\delta(f : S).$$

Note that the matrix of simplex directions and the vector of objective function differences depend on which of the vertices is labeled x_1. Each of the algorithms we consider in this section uses a vertex ordering and hence, at least implicitly, maintains a simplex gradient.

This definition of simplex gradient is motivated by the first order estimate, [14]:

Lemma 1 *Let S be a simplex. Let ∇f be Lipschitz continuous in a neighborhood of S with Lipschitz constant $2K$. Then*

$$\|\nabla f(x_1) - D(f:S)\| \leq K\kappa(V)\sigma_+(S). \tag{2}$$

Search algorithms are not intended, of course, for smooth problems. Minimization of objective functions of the form in (1) are one of the applications of these methods. Lemma 2 is a first order estimate that takes perturbations into account.

We will need to measure the perturbations on each simplex. To that end we define for a set T

$$\|\phi\|_T = \operatorname{esssup}_{x\in T}\|\phi(x)\|.$$

The analog of Lemma 1 for objective functions that satisfy (1) is, [14],

Lemma 2 *Let S be a nonsingular simplex. Let f satisfy* (1) *and let ∇f_s be continuously differentiable in a neighborhood of S. Then, there is $K > 0$ such that*

$$\|\nabla f_s(x_1) - D(f:S)\| \leq K\kappa(V)\left(\sigma_+(S) + \frac{\|\phi\|_S}{\sigma_+(S)}\right) \tag{3}$$

In [14] these ideas were applied to the Nelder-Mead algorithm with a view toward detecting stagnation in the iteration. The Nelder-Mead algorithm uses a simplex S of approximations to an optimal point. In this algorithm the vertices $\{x_j\}_{j=1}^{N+1}$ are sorted according to the objective function values

$$f(x_1) \leq f(x_2) \leq \ldots \leq f(x_{N+1}). \tag{4}$$

x_1 is called the best vertex and x_{N+1} the worst. The specific nature of the sort and tie-breaking rules have no effect on the performance of the algorithm.

The algorithm attempts to replace the worst vertex x_{N+1} with a new point of the form

$$x(\delta) = (1+\delta)\overline{x} - \delta x_{N+1}$$

where $\overline{x}$ is the centroid of the convex hull of $\{x_i\}_{i=1}^N$

$$\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i.$$

The value of δ is selected from a sequence

$$-1 < \delta_{ic} < 0 < \delta_{oc} < \delta_r < \delta_e$$

by rules that we formally describe in Algorithm `nelder`. Our formulation of the algorithm allows for termination if either $f(x_{N+1}) - f(x_1)$ is sufficiently small or a user-specified number of function evaluations has been expended.

Formally, the algorithm is

Algorithm 1 `nelder`$(S, f, \tau, kmax)$

1. *Evaluate f at the vertices of S and sort the vertices of S so that* (4) *holds.*
2. *Set $fcount = N + 1$.*
3. *While $f(x_{N+1}) - f(x_1) > \tau$*
   (a) *Compute $\overline{x}$ and $f_r = f(x(\delta_r))$. $fcount = fcount + 1$.*
   (b) **Reflect:** *If $fcount = kmax$ then exit. If $f(x_1) \le f_r < f(x_N)$, replace x_{N+1} with $x(\delta_r)$ and go to to step 3g.*

(c) **Expand** *If* $fcount = kmax$ *then exit. If* $f_r < f(x_1)$ *then compute* $f_e = f(x(\delta_e))$. $fcount = fcount + 1$. *If* $f_e < f_r$ *replace* x_{N+1} *with* $x(\delta_e)$, *otherwise replace* x_{N+1} *with* $x(\delta_r)$. *Go to to step 3g.*

(d) **Outside Contraction:** *If* $fcount = kmax$ *then exit. If* $f(x_N) \leq f_r < f(x_{N+1})$ *compute* $f_c = f(x(\delta_{oc}))$. $fcount = fcount + 1$.
If $f_c \leq f_r$ *replace* x_{N+1} *with* $x(\delta_{oc})$ *and go to step 3g, otherwise go to step 3f.*

(e) **Inside Contraction:** *If* $fcount = kmax$ *then exit. If* $f_r \geq f(x_{N+1})$ *compute* $f_c = f(x(\delta_{ic}))$. $fcount = fcount + 1$ *If* $f_c < f(x_{N+1})$ *replace* x_{N+1} *with* $x(\delta_{ic})$ *and go to step 3g, otherwise go to step 3f.*

(f) **Shrink** *If* $fcount \geq kmax - N$, *exit. For* $2 \leq i \leq N + 1$: *set* $x_i = x_1 - (x_i - x_1)/2$; *compute* $f(x_i)$.

(g) **Sort:** *Sort the vertices of* S *so that* (4) *holds.*

A typical sequence, [15], of candidate values for δ is

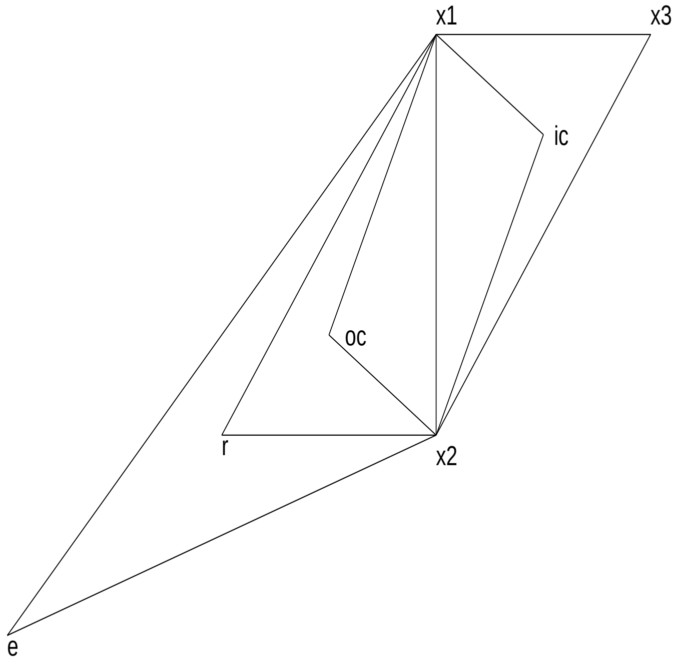
$$\{\delta_r, \delta_e, \delta_{oc}, \delta_{ic}\} = \{1, 2, 1/2, -1/2\}$$

Figure 2 is an illustration of the options in two dimensions. The vertices labeled $x1, x2$, and $x3$ are those of the original ordered simplex.

Figure 2 illustrates both the benefits and disadvantages of the Nelder-Mead algorithm. Unlike the other algorithms we consider in this paper, the simplex shape is free to adapt to the optimization landscape. However, the price for that adaptability is that the simplex can become highly ill-conditioned. The results from [14], which we now state, must assume that the conditioning of the simplices remains under control in order to guarantee convergence.

Another difference is that there is no distinguished vertex in the simplex whose function value is reduced. Unless a shrink

Figure 2: Nelder-Mead Simplex and New Points



step occurs, the Nelder-Mead iteration reduces the average

$$\underline{f} = \frac{1}{N+1} \sum_{j=1}^{N+1} f(x_j)$$

because the worst vertex is replaced by one with a lower function value. We will assume that shrink steps, which are rare, do not occur.

The two theorems below are from [14]. Both theorems require an analog of the sufficient decrease condition, [7], from gradient-based optimization and an assumption on the conditioning of the simplices. Failure of these conditions, which can happen in the Nelder-Mead algorithm, [17], is an indicator of stagnation.

Theorem 1 *Assume that the Nelder-Mead simplices are such that $V^k = V(S^k)$ is nonsingular and that*

$$\underline{f}^{k+1} - \underline{f}^k < -\alpha \|D(f : S^k)\|^2. \tag{5}$$

holds for some $\alpha > 0$ and all but finitely many k. Let the assumptions of Lemma 1 hold, with the Lipschitz constants K^k uniformly bounded. Then if the product $\sigma_+(S^k)\kappa(V^k) \to 0$, then any accumulation point of the simplices is a critical point of f.

Theorem 2 makes an assumption similar to one made in [12] that the noise decays to zero as the minimum is approached.

Theorem 2 *Assume that the Nelder-Mead simplices are such that V^k is nonsingular and let the assumptions of Lemma 2 hold with the Lipschitz constants K_s^k uniformly bounded. Then if* (5) *holds for all but finitely many k and that*

$$\lim_{k\to\infty} \kappa(V^k) \left(\sigma_+(S^k) + \frac{\|\phi\|_{S^k}}{\sigma_+(S^k)} \right) = 0, \tag{6}$$

then any accumulation point of the simplices is a critical point of f_s.

2. Convergence Results

2.1. Implicit Filtering

Implicit filtering is a difference-gradient implementation of the gradient projection algorithm [2] in which the difference increment is reduced in size as the iteration progresses. In this way the simplex gradient is used directly. It was originally proposed in [23], [20], [24], for various problems in semiconductor modeling and analyzed in [12].

In this paper we focus on the unconstrained form (see also [25]). This is sufficient to show how the simplex gradient can be used and to show how superlinear convergence might be possible. Implicit filtering is a point-based algorithm and, unlike the simplex-based algorithms, does distinguish the best point on a simplex. Rather the current iterate x_c is the point from which a simplex is build to compute a difference gradient. The new iterate x_+ is computed using a line search (which may fail, even for smooth problems, because the forward difference gradient may not be a descent direction). For a given $x \in R^N$ and $h > 0$ we let the simplex $S(x,h)$ be the right simplex from x with edges having length h. Hence the vertices are x and $x + hv_i$ for $1 \leq i \leq N$ with $V = I$. So $\kappa(V) = 1$.

The forward difference gradient is, of course,

$$\nabla_h f(x) = D(f : S(x,h)).$$

While a centered difference can be better in practice, [12], [16], [19], a forward difference will illustrate the idea and we use that in this paper. We use a simple Armijo [1] line search and demand that the sufficient decrease condition

$$f(x - \lambda \nabla_h f(x)) - f(x) < -\alpha\lambda \|\nabla_h f(x)\|^2 \tag{7}$$

hold (compare to (5)) for some $\alpha > 0$. Our forward difference steepest descent algorithm `fdsteep` terminates when

$$\|\nabla_h f(x)\| \leq \tau h \tag{8}$$

for some $\tau > 0$, when more than $kmax$ iterations have been taken, or when the line search fails by taking more than $amax$ backtracks. Even the failures of `fdsteep` can be used to advantage by triggering a reduction in h. The line search parameters α, β and the parameter τ in the termination criterion (8) do not affect the convergence analysis that we present here, but can affect performance.

Algorithm 2 `fdsteep`$(x, f, kmax, \tau, h, amax)$

1. *For* $k = 1, \ldots, kmax$

   (a) *Compute* f *and* $\nabla_h f$; *terminate if* (8) *holds.*

(b) *Find the least integer* $0 \le m \le amax$ *such that* (7) *holds for* $\lambda = \beta^m$. *If no such* m *exists, terminate.*

(c) $x = x - \lambda \nabla_h f(x)$.

Algorithm `fdsteep` will terminate after finitely many iterations because of the limits on the number of iterations and the number of backtracks. If the set $\{x \mid f(x) \le f(x_0)\}$ is bounded then the iterations will remain in that set. Implicit filtering calls `fdsteep` repeatedly, reducing h after each termination of `fdsteep`. Aside from the data needed by `fdsteep`, a sequence of difference increments (called *scales* in [23], [20], [24], [19], [12], [6], and [5]), $\{h_k\}_{k=0}^{\infty}$ is needed for the form of the algorithm given here.

Algorithm 3 `imfilter1`$(x, f, kmax, \tau, \{h_j\}, amax)$

1. *For* $k = 0, \ldots$
   Call `fdsteep`$(x, f, kmax, \tau, h_k, amax)$

Since $h_k = \sigma_+(S^k)$ and $\kappa(V^k) = 1$ the first order estimate, (3) implies a convergence result that is different from the one in [12].

Theorem 3 *Let* $h_k \to 0$ *and let* f *satisfy* (1). *Let* $\{x_k\}$ *be the implicit filtering sequence and let* $S^k = S(x, h_k)$. *Assume that* (7) *holds (*i. e. *there is no line search failure) for all but finitely many* k. *Then if*

$$\lim_{k \to \infty} (h_k + h_k^{-1} \|\phi\|_{S^k}) = 0 \tag{9}$$

then any limit point of the sequence $\{x_k\}$ *is a critical point of* f_s.

Proof. If (7) holds for all but finitely many k then, as is standard,

$$\nabla_{h_k} f(x_k) = D(f : S^k) \to 0.$$

Hence, using (9) and Lemma 2

$$\nabla f_s(x_k) \to 0,$$

as asserted. $\square$

Because implicit filtering directly maintains an approximate gradient and uses that to compute a descent direction, it is natural to try a quasi-Newton Hessian. Successful experiments with SR1 [3], [9], update have been reported in [11], [12], and [19].

2.2. Multidirectional Search

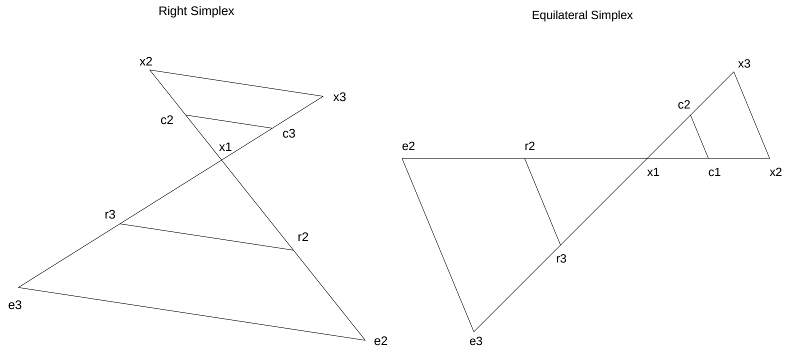
A natural way to address the possible ill-conditioning in the Nelder-Mead algorithm is to require that the condition numbers of the simplices be bounded. The most direct way to do that is to insist that the simplices have the same shape. The multidirectional search method, [8], [21], does this by making each new simplex congruent to the previous one. In the special case of equilateral simplices, V^k is a constant multiple of V^0 and the simplex condition number is constant. If the simplices are not equilateral, then $\kappa(V)$ may vary depending on which vertex is called x_1, but we will have, for some $\mu_- \in (0,1)$ and $\kappa_+ > 0$,

$$\kappa(V) \leq \kappa_+ \text{ and } x^T V V^T x \geq \mu_- \sigma_+(V)^2 \|x\|^2 \text{ for all } x. \tag{10}$$

The algorithm is best understood by consideration of Figure 3, which illustrates the two-dimensional case for two types of simplices. Beginning with the ordered simplex S^c with vertices x_1, x_2, x_3 one first attempts a **rotation** step, leading to a simplex S^r with vertices x_1, r_1, r_2.

If the best function value of the vertices of S^r is better than the best $f(x_1)$ in S^0, S^r is (provisionally) accepted and and **expansion** is attempted. The expansion step is similar to that in the Nelder-Mead algorithm. The expansion simplex S^e has vertices x_1, e_1, e_2 and is accepted over S^r if the best function value of the vertices of S^e is better than the best in S^r. If the best function value of the vertices of S^r is not better than the best in S^c, then the simplex is **contracted** and the new simplex has vertices x_1, c_1, c_2. After the new simplex is identified, the vertices are reordered to create the new ordered simplex S^+.

Figure 3: MDS Simplices and New Points



Similarly to the Nelder-Mead algorithm, there are expansion and contraction parameters δ_e and δ_c. Typical values for these are 2 and 1/2.

Algorithm 4 `mds`$(S, f, \tau, kmax)$

1. *Evaluate f at the vertices of S and sort the vertices of S so that* (4) *holds.*

2. *Set $fcount = N + 1$.*

3. *While $f(x_{N+1}) - f(x_1) > \tau$*

   (a) **Reflect:** *If $fcount = kmax$ then exit.*
   For $j = 1, \ldots, N$: $r_j = x_1 - (x_j - x_1)$, Compute $f(r_j)$
   If $f(x_1) < \min_j\{f(r_j)\}$ then goto step 3b else goto step 3c.

   (b) **Expand:**

      i. *For $j = 1, \ldots, N$: $e_j = x_1 - \delta_e(x_j - x_1)$, Compute $f(e_j)$*

ii. *If* $\min_j\{f(r_j)\} < \min_j\{f(e_j)\}$ *then*
for $j = 1, \ldots N$*:* $x_j = e_j$
else
for $j = 1, \ldots N$*:* $x_j = r_j$

iii. *Goto step 3d*

(c) **Contract:** *For* $j = 1, \ldots, N$*:* $x_j = x_1 + \delta_c(x_j - x_1)$*,* *Compute* $f(x_j)$

(d) **Sort:** *Sort the vertices of* S *so that* (4) *holds.*

IF the function values at the vertices of S^c are known, then the cost of computing S^+ is $2N$ additional evaluations. Just as with Nelder-Mead, the expansion step is optional, but has been observed to improve performance.

Assume that the simplices are either equilateral or right simplices (having one vertex from which all N edges are at right angles). In those cases, as pointed out in [21], the possible vertices created by expansion and reflection steps form a regular lattice of points. If the MDS simplices remain bounded, only finitely many reflections and expansions are possible before every point on that lattice has been visited and a contraction to a new maximal simplex size must take place. This exhaustion of a lattice takes place under more general conditions, [21], but is most clear for the equilateral case.

The point of Lemma 3 is that infinitely many contractions and convergence of the simplex diameters to zero imply convergence of the simplex gradient to zero.

Lemma 3 *Let* S *be an ordered simplex such that* (10) *holds. Let* f *satisfy* (1)*, let* ∇f_s *be Lipschitz continuously continuously differentiable in a ball of* B *radius* $2\sigma_+(S)$ *about* x_1*. Assume that*

$$f(x_1) < \min_j\{f(r_j)\}. \tag{11}$$

Then, if K *is the constant from Lemma 2,*

$$\|\nabla f_s(x_1)\| \leq 8\mu_-^{-1} K \kappa_+ \left(\sigma_+(S) + \frac{\|\phi\|_B}{\sigma_+(S)}\right) \tag{12}$$

Proof. Let R, the (unordered!) reflected simplex, have vertices x_1 and $\{r_j\}_{j=1}^N$. (11) implies that each component of $\delta(f : S)$ and $\delta(f : R)$ is positive. Now since

$$V = V(S) = -V(R),$$

we must have

$$\begin{aligned} 0 &< \delta(f : S)^T \delta(f : R) \\ &= (V^T V^{-T} \delta(f : S))^T (V(R)^T V(R)^{-T} \delta(f : R)) \\ &= -D(f : S)^T V V^T D(f : R). \end{aligned} \tag{13}$$

We apply Lemma 2 to both $D(f : S)$ and $D(f : R)$ to obtain

$$D(f : S) = \nabla f_s(x_1) + E_1 \text{ and } D(f : R) = \nabla f_s(x_1) + E_2$$

where, since $\kappa(V) = \kappa(V(R)) \leq \kappa_+$,

$$\|E_k\| \leq K \kappa_+ \left(\sigma_+(S) + \frac{\|\phi\|_B}{\sigma_+(S)} \right).$$

Since $\|V\| \leq 2\sigma_+(S)$ we have by (13)

$$\begin{aligned} \nabla f_s(x_1)^T V V^T \nabla f_s(x_1) \leq \quad & 4\sigma_+(S)^2 \|\nabla f_s(x_1)\| (\|E_1\| + \|E_2\|) \\ & +4\sigma_+(S)^2 \|E_1\| \|E_2\|. \end{aligned} \tag{14}$$

The assumptions of the lemma give a lower estimate of the left side of (14),

$$w^T V V^T w \geq \mu_- \sigma_+(V)^2 \|w\|^2.$$

Hence,

$$\|\nabla^2 f(x_1)\| \leq B \|\nabla^2 f(x_1)\| + C$$

where, using (14),

$$B = 8\mu_1^{-1} K_s \kappa_+ \left(\sigma_+(S) + \frac{\|\phi\|_B}{\sigma_+(S)} \right)$$

and

$$C = 4\mu_-^{-1}(K_s\kappa_+)^2 \left(\sigma_+(S) + \frac{\|\phi\|_B}{\sigma_+(S)}\right)^2 = \frac{\mu_-}{16}B^2.$$

So $B^2 - 4C = B^2(1 - \mu_-/4)$ and the quadratic formula then implies that

$$\|\nabla^2 f(x_1)\| \leq \frac{B + \sqrt{B^2 - 4C}}{2} = B\frac{1 + \sqrt{1 - \mu_-/4}}{2} \leq B$$

as asserted. □

The similarity of Lemma 3 to Lemma 2 and of Theorem 4, the convergence result for multidirectional search, to Theorem 2 is no accident. The Nelder-Mead iteration, which is more aggressive that the multidirectional search iteration, requires far stronger assumptions (well conditioning and sufficient decrease) for convergence, but the ideas are the same. Lemma 3 and Theorem 4 extends the results in [21] to the noisy case. The observation in [8] that one can apply any heuristic or machine-dependent idea to improve performance, say by exploring far away points on spare processors (the "speculative function evaluations" of [4]), without affecting the analysis is still valid here.

Theorem 4 *Let f satisfy* (1) *and assume that the set*

$$\{x \mid f(x) \leq f(x_1^0)\}$$

is bounded. Assume that the simplex shape is such that

$$\lim_{k\to\infty} \sigma_+(S^k) \to 0. \tag{15}$$

Let B^k be a ball of radius $2\sigma_+(S^k)$ about x_1^k. Then if

$$\lim_{k\to\infty} \frac{\|\phi\|_{B^k}}{\sigma_+(S^k)} = 0$$

then every limit point of the vertices is a critical point of f_s.

Recall that if the simplices are equilateral or right simplices, then (15) holds.

The more general class of pattern search algorithms studied in [22] can also be analyzed in this way and we plan to do that in future work.

References

[1] L. Armijo, *Minimization of functions having Lipschitz-continuous first partial derivatives*, Pacific J. Math., 16 (1966), pp. 1–3.

[2] D. B. Bertsekas, *On the Goldstein-Levitin-Polyak gradient projection method*, IEEE Trans. Autom. Control, 21 (1976), pp. 174–184.

[3] C. G. Broyden, *Quasi-Newton methods and their application to function minimization*, Math. Comp., 21 (1967), pp. 368–381.

[4] R. H. Byrd, R. B. Schnabel, and G. A. Schultz, *Parallel quasi-Newton methods for unconstrained optimization*, Math. Prog., 42 (1988), pp. 273–306.

[5] J. W. David, C. Y. Cheng, T. D. Choi, C. T. Kelley, and J. Gablonsky, *Optimal design of high speed mechanical systems*, Tech. Rep. CRSC-TR97-18, North Carolina State University, Center for Research in Scientific Computation, July 1997. Mathematical Modeling and Scientific Computing, to appear.

[6] J. W. David, C. T. Kelley, and C. Y. Cheng, *Use of an implicit filtering algorithm for mechanical system parameter identification.* SAE Paper 960358, 1996 SAE International Congress and Exposition Conference Proceedings, Modeling of CI and SI Engines, pp. 189–194.

[7] J. E. Dennis and R. B. Schnabel, *Numerical Methods for Nonlinear Equations and Unconstrained Optimization*, no. 16 in Classics in Applied Mathematics, SIAM, Philadelphia, 1996.

[8] J. E. Dennis and V. Torczon, *Direct search methods on parallel machines*, SIAM J. Optim., 1 (1991), pp. 448 – 474.

[9] A. V. Fiacco and G. P. McCormick, *Nonlinear Programming*, John Wiley and Sons, New York, 1968.

[10] S. J. Fortune, D. M. Gay, B. W. Kernighan, O. Landron, R. A. Valenzuela, and M. H. Wright, *WISE design of indoor wireless systems*, IEEE Computational Science and Engineering, Spring (1995), pp. 58–68.

[11] P. Gilmore, *An Algorithm for Optimizing Functions with Multiple Minima*, PhD thesis, North Carolina State University, Raleigh, North Carolina, 1993.

[12] P. Gilmore and C. T. Kelley, *An implicit filtering algorithm for optimization of functions with many local minima*, SIAM J. Optim., 5 (1995), pp. 269–285.

[13] P. Gilmore, C. T. Kelley, C. T. Miller, and G. A. Williams, *Implicit filtering and optimal design problems: Proceedings of the workshop on optimal design and control, Blacksburg VA, April 8–9, 1994*, in Optimal Design and Control, J. Borggaard, J. Burkhardt, M. Gunzburger, and J. Peterson, eds., vol. 19 of Progress in Systems and Control Theory, Birkhäuser, Boston, 1995, pp. 159–176.

[14] C. T. Kelley, *Detection and remediation of stagnation in the Nelder-Mead algorithm using a sufficient decrease condition*, Tech. Rep. CRSC-TR97-2, North Carolina State University, Center for Research in Scientific Computation, January 1997. Submitted for Publication.

[15] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, *Convergence properties of the Nelder-Mead simplex algorithm in low dimensions*, Tech. Rep. 96-4-07, AT&T Bell Laboratories, April 1996.

[16] D. Q. Mayne and E. Polak, *Nondifferential optimization via adaptive smoothing*, J. Optim. Theory Appl., 43 (1984), pp. 601–613.

[17] K. I. M. McKinnon, *Convergence of the Nelder-Mead simplex method to a non-stationary point*, tech. rep., Department of Mathematics and Computer Science, University of Edinburgh, Edinburgh, 1996.

[18] J. A. Nelder and R. Mead, *A simplex method for function minimization*, Comput. J., 7 (1965), pp. 308–313.

[19] D. Stoneking, G. Bilbro, R. Trew, P. Gilmore, and C. T. Kelley, *Yield optimization using a GaAs process simulator coupled to a physical device model*, IEEE Transactions on Microwave Theory and Techniques, 40 (1992), pp. 1353–1363.

[20] D. E. Stoneking, G. L. Bilbro, R. J. Trew, P. Gilmore, and C. T. Kelley, *Yield optimization using a GaAs process simulator coupled to a physical device model*, in Proceedings IEEE/Cornell Conference on Advanced Concepts in High Speed Devices and Circuits, IEEE, 1991, pp. 374–383.

[21] V. Torczon, *On the convergence of the multidimensional direct search*, SIAM J. Optim., 1 (1991), pp. 123–145.

[22] ——, *On the convergence of pattern search algorithms*, SIAM J. Optim., 7 (1997), pp. 1–25.

[23] T. A. Winslow, R. J. Trew, P. Gilmore, and C. T. Kelley, *Doping profiles for optimum class B performance*

of GaAs mesfet amplifiers, in Proceedings IEEE/Cornell Conference on Advanced Concepts in High Speed Devices and Circuits, IEEE, 1991, pp. 188–197.

[24] ——, *Simulated performance optimization of GaAs MESFET amplifiers*, in Proceedings IEEE/Cornell Conference on Advanced Concepts in High Speed Devices and Circuits, IEEE, 1991, pp. 393–402.

[25] S. K. Zavriev, *On the global optimization properties of finite-difference local descent algorithms*, J. Global Optimization, 3 (1993), pp. 67–78.