DETECTION AND REMEDIATION OF STAGNATION IN THE NELDER-MEAD ALGORITHM USING A SUFFICIENT DECREASE CONDITION *

C. T. KELLEY†

Abstract. The Nelder-Mead algorithm can stagnate and converge to a non-optimal point, even for very simple problems. In this note we propose a test for sufficient decrease which, if passed for the entire iteration, will guarantee convergence of the Nelder-Mead iteration to a stationary point if the objective function is smooth. Failure of this condition is an indicator of potential stagnation. As a remedy we propose a new step, which we call an oriented restart, which reinitializes the simplex to a smaller one with orthogonal edges which contains an approximate steepest descent step from the current best point. We also give results that apply when objective function is a low-amplitude perturbation of a smooth function. We illustrate our results with some numerical examples.

Key words. Nelder-Mead algorithm, sufficient decrease, stagnation

AMS subject classifications. 65K05, 65K10, 90C30

1. Introduction. In this paper we consider the Nelder-Mead, [14], direct search algorithm for the unconstrained minimization of a possibly nonconvex or even discontinuous function f. The problem is

$$\min_{x \in R^N} f(x). \tag{1.1}$$

As in [9], we also consider objective functions that are small perturbations of smooth and easy to minimize functions.

The Nelder-Mead algorithm maintains a simplex of approximations to an optimal point. We assume throughout that the vertices $\{x_j\}_{j=1}^{N+1}$ are sorted according to the objective function values

$$f(x_1) \leq f(x_2) \leq \ldots \leq f(x_{N+1}). \tag{1.2}$$

We will refer to x_1 as the best vertex and x_{N+1} as the worst. The algorithm attempts to change the worst vertex x_{N+1} to 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 $\{x_i\}_{i=1}^N$

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

A typical sequence, [12], of candidate values for δ is

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

corresponding to the reflection, expansion, outside contraction, and inside contraction steps of the Nelder-Mead iteration. In general we require that

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

* Version of April 20, 1997. This research was supported by National Science Foundation grant #DMS-9321938.

† North Carolina State University, Center for Research in Scientific Computation and Department of Mathematics, Box 8205, Raleigh, N. C. 27695-8205, USA, Tim_Kelley@ncsu.edu).

Algorithm `nelder` is a formal description of the iteration, with a termination condition on small differences in the function values at the vertices. The input is the initial simplex, the objective, and a tolerance for termination based on the difference between the best and worst values.

ALGORITHM 1.1. `nelder`(S, f, τ)

1. *Sort the vertices of* S *so that* (1.2) *holds.*
2. *While* $f(x_{N+1}) - f(x_1) > \tau$
   (a) *Compute* $\overline{x}$ *and* $f_r = f(x(\delta_r))$.
   (b) **Reflect:** *If* $f(x_1) \leq f_r < f(x_N)$, *replace* x_{N+1} *with* $x(\delta_r)$ *and go to to step 2g.*
   (c) **Expand** *If* $f_r < f(x_1)$ *then compute* $f_e = f(x(\delta_e))$. *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 2g.*
   (d) **Outside Contraction:** *If* $f(x_N) \leq f_r < f(x_{N+1})$ *compute* $f_c = f(x(\delta_{oc}))$. *If* $f_c \leq f_r$ *replace* x_{N+1} *with* $x(\delta_{oc})$ *and go to step 2g, otherwise go to step 2f.*
   (e) **Inside Contraction:** *If* $f_r \geq f(x_{N+1})$ *compute* $f_c = f(x(\delta_{ic}))$. *If* $f_c < f(x_{N+1})$ *replace* x_{N+1} *with* $x(\delta_{ic})$ *and go to step 2g, otherwise go to step 2f.*
   (f) **Shrink** *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* (1.2) *holds.*

The sort step is not precisely defined by Algorithm `nelder`. One could simply use a sort from one's computing environment, or specify a sort algorithm with a tie-breaking rule. Our results only require that each simplex satisfy (1.2), so we can follow [14] and accept any sort. One specific tie-breaking rule was proposed in [12].

As one can see from the algorithm, if $f(x(\delta)) \geq f(x_{n+1})$ for all four possibilities ($\delta = 1, 2, \pm 1/2$), then the simplex is reduced in size, keeping only the vertex x_1 with the lowest objective function value. This last scenario, the shrink step, is rare and the analysis we present will assume that shrinks do not occur.

We regard the simplex, and not just the best point, as the state of the iteration. If a shrink step is not taken, then the worst vertex is replaced with the new, better, vertex, the vertices are resorted, and the average objective function value

$$\underline{f} = \frac{1}{N+1}\sum_{j=1}^{N+1} f(x_j) \tag{1.3}$$

has been decreased.

Unlike the pattern search algorithms, [4], [7], [10], [18], [19], that maintain the shape of the simplex, or the hybrid algorithm from [20], the Nelder-Mead algorithm can stagnate and converge to a non-optimal point, [13], [8], [23], [12], [24], even for very simple, smooth, and convex objective functions. Our results and analysis are simplex oriented and assume, even if f is discontinuous or nonsmooth, an underlying smooth structure of the objective that, while not present in the general case, is present in many applications.

In § 2 we set notation and state a few simple lemmas. In particular we define a *simplex gradient* which we use to monitor the performance of the Nelder-Mead iteration and for our modification of the shrink step. We use those ideas in § 3, where we describe our condition for sufficient decrease (3.1) and show in § 3.1 that if the objective function f is sufficiently smooth, the Nelder-Mead iterates satisfy this sufficient decrease condition, and the simplex diameters converge to zero in a certain way, then any limit point of the simplex vertices is stationary. Our decrease condition is simpler than the one in [20], reflecting our interest in high-frequency low amplitude perturbations of smooth functions and in exploiting the one function evaluation/iteration cost of the Nelder-Mead algorithm as aggressively as safely possible.

In § 3.2 we propose an alternative to the shrink step that is to be used when (3.1) fails to hold. This new step, which we call an *oriented restart*, reinitializes the simplex to a smaller one with orthogonal edges which contains an approximate steepest descent step from the current best point.

Throughout the paper we present results that apply when objective function is a low-amplitude perturbation of a smooth function. Finally, in § 4 we show how a modified Nelder-Mead algorithm that incorporates our ideas perform on the examples in [13].

2. Notation. In this paper $\|\cdot\|$ will denote the l^2 norm or the induced matrix norm. We consider algorithms that maintain a simplex S of potential optima with vertices $\{x_j\}_{j=1}^{N+1}$ that satisfy (1.2).

We let V (or $V(S)$) denote the $N \times N$ matrix of *simplex directions*

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

and $\operatorname{diam}(S)$ the *simplex diameter*

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

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,

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

DEFINITION 2.1. *Let S be a simplex with vertices $\{x_j\}_{j=1}^{N}$ ordered so that* (1.2) *holds and $V(S)$ nonsingular. The* simplex gradient *$D(f:S)$ is*

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

This definition of simplex gradient is motivated by the first order estimate:

LEMMA 2.2. *Let S be a simplex with vertices ordered so that* (1.2) *holds. Let ∇f be Lipschitz continuous in a neighborhood of S with Lipschitz constant $2K$. Then*

$$\|\nabla f(x_1) - D(f:S)\| \le K\kappa(V)\sigma_+(S). \tag{2.1}$$

Proof. Our smoothness assumptions on f and Taylor's theorem imply that for all $1 \le j \le N$,

$$\begin{aligned}\left\| f(x_1) - f(x_j) + \frac{\partial f(x_1)}{\partial v_j} v_j \right\| &= \|f(x_1) - f(x_j) + v_j^T \nabla f(x_1)\| \\ &\le K\|v_j\|^2 \le K\sigma_+(S)^2.\end{aligned}$$

Hence

$$\|\delta(f:S) - V^T \nabla f(x_1)\| \le K\sigma_+(S)^2$$

and hence

$$\|\nabla f(x_1) - D(f:S)\| \le K\|V^{-T}\|\sigma_+(S)^2.$$

The conclusion follows from the the fact that $\sigma_+(S) \le \|V\|$. ☐

Objective functions of the form

$$f(x) = g(x) + \phi(x), \tag{2.2}$$

where g is to be thought of as a smooth and easy to optimize function and $\phi \in L^\infty$ a low-amplitude perturbation arise naturally in applications [2], [3], [5], [9], [16], [17], [21], [22]. Algorithms that use difference approximations to the gradient of f have been proposed for bound constrained, [16], [9], and unconstrained, [25], problems as a way to avoid entrapment in local minima caused by the perturbation. The implicit filtering algorithm algorithm described in [9] and [16] also attempts to obtain superlinear convergence in the terminal phase of the iteration if $\phi(x) \to 0$ as x tends to the optimal point. Like the pattern search algorithms, these difference methods require $O(N)$ function evaluations/iteration and therefore may be much less efficient in the initial phase of the iteration that a simplex algorithm like Nelder-Mead that requires only $O(1)$ evaluation/iteration.

One purpose of this paper is to apply simplicial algorithms that use fewer than $O(N)$ function evaluations/iteration to problems with such objective functions. One would hope that the varying sizes of the simplices during the iteration would help avoid local minima. We will need to measure the perturbations on each simplex. To that end we define for a simplex S

$$\|\phi\|_S = \text{esssup}_{x\in S}\|\phi(x)\|.$$

The analog of Lemma 2.2 for objective functions that satisfy (2.2) is

LEMMA 2.3. *Let S be a simplex with vertices ordered so that* (1.2) *holds. Let f satisfy* (2.2) *and let ∇g be continuously differentiable in a neighborhood of S with Lipschitz constant $2K_g$. Then, there is $K > 0$ such that*

$$\|\nabla g(x_1) - D(f:S)\| \le K\kappa(V)\left(\sigma_+(S) + \frac{\|\phi\|_S}{\sigma_+(S)}\right) \tag{2.3}$$

Proof. Lemma 2.2 (applied to g) implies that

$$\|\nabla g(x_1) - D(g:S)\| \le K_g\kappa(V)\sigma_+(S).$$

Now, since $\|\delta(\phi, S)\| \le 2\sqrt{N}\|\phi\|_S$, and $\sigma_+(V) \le \|V\|$,

$$\begin{aligned}\|D(f:S) - D(g:s)\| &\le \|V^{-T}\|\|\|\delta(f:S) - \delta(g:S)\| = \|V^{-T}\|\|\|\delta(\phi:S)\| \\ &\le 2N^{1/2}\|V^{-T}\|\|\|\phi\|_S \le 2N^{1/2}\kappa(V)\frac{\|\phi\|_S}{\sigma_+(S)}.\end{aligned}$$

This completes the proof with $K = K_g + 2N^{1/2}$. ☐

The constants K in (2.1) and (2.3) depend on S and we will express that dependence as $K = K(S)$ when needed.

We will denote the vertices of the simplex S^k at the kth iteration by $\{x_j^k\}_{j=1}^N$. We will simplify notation by suppressing explicit mention of S^k in what follows by denoting

$$V^k = V(S^k), \delta^k = \delta(f : S^k), K^k = K(S^k), \text{ and } D^k(f) = D(f : S^k).$$

If V^0 is nonsingular then V^k is nonsingular for all $k > 0$, [12]. Hence if V^0 is nonsingular, $D^k(f)$ is defined for all k.

We assume that our sequence of simplices satisfies

ASSUMPTION 2.1.

- *V^0 is nonsingular.*
- *The vertices satisfy* (1.2).
- *For each k, $\underline{f}^{k+1} < \underline{f}^k$.*

Assumption 2.1 is satisfied by the Nelder-Mead sequence if no shrink steps are taken and the initial simplex directions are linearly independent. Assumption 2.1 need not be satisfied by the pattern search methods considered in [19], where only conditions on the best value are enforced. One way to illustrate the difference between pattern search methods and methods like Nelder-Mead is to note that the pattern search methods require that the simplices be geometrically similar, so the simplex condition is bounded, and require improvement in the best point. Nelder-Mead demands that the average function value improve, but no control is possible on which value is improved, and the simplex condition number can become unbounded.

3. Sufficient Decrease and the Oriented Restart.

Motivated by conventional line-search decrease criteria for optimization and nonlinear equations [1], [6], [11], [15] we will ask that the $k+1$st iteration satisfy

$$\underline{f}^{k+1} - \underline{f}^k < -\alpha \|D^k f\|^2. \tag{3.1}$$

Here $\alpha > 0$ is a small parameter. A typical choice in line-search methods, which we use in our numerical results, is $\alpha = 10^{-4}$. We propose to use failure of (3.1) as a test for impending stagnation at a non-minimizer. Clearly, if $\{\|D^k f\|\}$ does not converge to zero, the Nelder-Mead sequence is also not converging to a minimizer.

3.1. Convergence Results.

An immediate consequence of Lemma 2.2, valid for smooth f, is

THEOREM 3.1. *Let a sequence of simplices satisfy Assumption 2.1 and let the assumptions of Lemma 2.2 hold, with the Lipschitz constants K^k uniformly bounded. Then if* (3.1) *holds for all but finitely many k and the product $\sigma_+(S^k)\kappa(V^k) \to 0$, then any accumulation point of the simplices is a critical point of f.*

Proof. Assumption 2.1 and (3.1) imply that $\lim_{k\to\infty} D^k f = 0$. Hence (2.1) implies that

$$\lim_{k\to\infty} \|\nabla f(x_1^k)\| \le \lim_{k\to\infty} \left(K^k \kappa(V^k)\sigma_+(S^k) + \|D^k f\|\right) = 0.$$

Hence, if x^* is any accumulation point of the sequence $\{x_1^k\}$ then $\nabla f(x^*) = 0$. This completes the proof since $\sigma_+(V^k) \to 0$. □

Note that the conclusion of Theorem 3.1 also holds, [15], if the sufficient decrease condition (3.1) is replaced by

$$\underline{f}^{k+1} - \underline{f}^k < -\Phi(D^k f), \tag{3.2}$$

where Φ is a monotonically increasing function on $[0, \infty)$ with $\Phi(0) = 0$.

The result for the noisy functions that satisfy (2.2) with g smooth reflects the fact that the resolution is limited by the size of ϕ. In fact, if $\sigma_+(S^k)$ is much smaller than $\|\phi\|_{S^k}$, no information on g g can be obtained by evaluating f at the vertices of S^k and once $\sigma_+(S^k)$ is smaller than $\|\phi\|_{S^k}^{1/2}$ no conclusions on ∇g can be drawn.

THEOREM 3.2. *Let a sequence of simplices satisfy Assumption 2.1 and let the assumptions of Lemma 2.3 hold with the Lipschitz constants K_g^k uniformly bounded. Then if* (3.1) *holds for all but finitely many k and that*

$$\lim_{k \to \infty} \kappa(V^k) \left(\sigma_+(S^k) + \frac{\|\phi\|_{S^k}}{\sigma_+(S^k)} \right) = 0, \tag{3.3}$$

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

Proof. Our assumptions, as in the proof of Theorem 3.1 imply that $D^k f \to 0$. Lemma 2.3 implies that

$$\|D^k g\| \leq \|D^k f\| + K^k \kappa(V^k) \left(\sigma_+(S^k) + \frac{\|\phi\|_{S^k}}{\sigma_+(S^k)} \right), \tag{3.4}$$

and the sequence $\{K^k\}$ is bounded. Hence, by (3.3), $D^k g \to 0$ as $k \to \infty$. $\square$

3.2. Oriented Restarts. One can monitor a simplex-based iteration to see if (3.1) holds, however, unlike the case of a gradient-based line-search method, simply reducing the size of the simplex (for example, a shrink step in Nelder-Mead) will not remedy the problem. We propose performing an *oriented restart* when (3.1) fails but $\underline{f}^{k+1} - \underline{f}^k < 0$. This means to replace the current simplex with vertices $\{x_j\}_{j=1}^{N+1}$, ordered so that (1.2) holds, with a new smaller simplex having vertices (before ordering!) $\{y_j\}_{j=1}^{N+1}$ with $y_1 = x_1$ and

$$y_j = y_1 + \beta_{j-1} e_{j-1}, \text{ for } 2 \leq j \leq N+1 \tag{3.5}$$

where e_l is the kth coordinate vector,

$$\beta_l = \frac{1}{2} \begin{cases} \sigma_-(S^k)\mathrm{sign}((D^k f)_l) & (D^k f)_l \neq 0 \\ \sigma_-(S^k) & (D^k f)_l = 0 \end{cases}$$

and $(D^k f)_l$ is the lth component of $D^k f$. If $D^k f = 0$ we assume that the Nelder-Mead iteration would have been terminated at iteration k because of no difference between best and worst values.

So, before ordering, the new simplex has the same first point as the old. The diameter of the new simplex has not been increased since the diameter of the new simplex is at most $\sigma_+(S^k)$. Moreover all edge lengths have been reduced. So after reordering $\sigma_+(S^{k+1}) \leq \sigma_-(S^k)$. As for κ, after the oriented shrink, but before reordering, $\kappa(V) = 1$. After reordering, of course, the best point may no longer be x_1. If the best point is unchanged $V^{k+1} = I$. If the best point has been changed, then, up to row permutation and multiplication by the scalar $\pm\sigma_+(S^k)/2$, V^{k+1} is given by the lower triangular matrix

$$V^{k+1} = (V^{k+1})^{-1} = \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 0 & -1 & 0 & \dots & 0 \\ \vdots & 0 & -1 & \ddots & \vdots \\ 0 & & \ddots & \ddots & 0 \\ 0 & \dots & \dots & 0 & -1 \end{pmatrix}.$$

Hence either $V^{k+1} = I$ and $\kappa(V) = 1$ or the l^1 condition number is $\kappa_1(V^{k+1}) = \|V^{k+1}\|_1^2 = 4$. The l^2 condition number can be estimated by

$$\kappa(V^{k+1}) = \|V^{k+1}\|^2 \leq (1 + \sqrt{N})^2.$$

In any case, the new simplex is well conditioned.

The new orientation of the simplex is intended to compensate for the kind of stagnation that was exhibited in [13], in which the best vertex, not a minimizer, remained unchanged throughout the entire iteration, the simplices converged to that vertex. The expectation is that once the simplex is small enough, the simplex gradient point in a direction that approximated the true gradient and that the restarted simplex would have a new best vertex. The reduction is the simplex size is, like the reduction in steplength in a line-search method, should easier to satisfy (3.1). While there is nothing special about the factor of $1/2$ in (3.5), there is no reason to expect that more elaborate line search schemes based on polynomial models, such as those presented in [6], would be effective in the context in which Nelder-Mead and related algorithms are used.

4. Numerical Testing. We show how the detection of stagnation and modification of the Nelder-Mead algorithm proposed in § 3 perform in the examples from [13]. Here $N = 2$ and

$$f(x, y) = \begin{cases} \theta\phi|x|^\tau + y + y^2 & x \leq 0 \\ \\ \theta x^\tau + y + y^2 & x > 0. \end{cases}$$

The examples in [13] consider the parameter sets

$$(\tau, \theta, \phi) = \begin{cases} (3, 6, 400) \\ (2, 6, 60) \\ (1, 15, 10) \end{cases}$$

The initial simplex was

$$x_1 = (1, 1)^T, x_2 = (\lambda_+, \lambda_-)^T, x_3 = (0, 0)^T, \text{ where } \lambda_\pm = (1 \pm \sqrt{33})/8.$$

With this data, the Nelder-Mead iteration will stagnate at the origin, which is not a critical point for f.

We terminated the iteration when the difference between the best and worst function values was $< 10^{-8}$ or, in the modified algorithm, after three restarts, using this as a test for stagnation.

We illustrate the behavior of the unmodified Nelder-Mead algorithm in Figures 4.1, 4.3, and 4.5. In all the figures we plot, as functions of the iteration index, the difference between the best and worst function values, σ_+, the maximum oriented length, the norm of the simplex gradient, and the l^2 condition number of the matrix of simplex directions. In all three problems stagnation is evident from the behavior of the simplex gradients. Note also how the simplex condition number is growing rapidly.

Two of the examples in [13] are smooth. For the less smooth of these two, $(\tau, \theta, \phi) = (2, 6, 60)$, the modified form of Nelder-Mead took a single oriented restart at the 17th iteration. For the smoothest example, $(\tau, \theta, \phi) = (3, 6, 400)$ the modified form of Nelder-Mead took a single oriented restart at the 19th iteration. As one can see from Figures 4.4 and 4.6 the restart had an immediate effect on the simplex gradient norm and overcame the stagnation.

For the nonsmooth example, $(\tau, \theta, \phi) = (1, 15, 10)$, in Figure 4.1, the modified algorithm terminated with failure after restarting on the 30th, 31st, and 32nd iterations. Since the objective is not smooth at the stagnation point, this is the best we can expect and is far better that the behavior of the unmodified algorithm, which stagnates with no warning of the failure.

All computations reported here were done using MATLAB 4.2c on a SUN Ultra Enterprise 1 running Sun Solaris v2.1.

FIG. 4.1. *Unmodified Nelder-Mead,* $(\tau, \theta, \phi) = (1, 15, 10)$

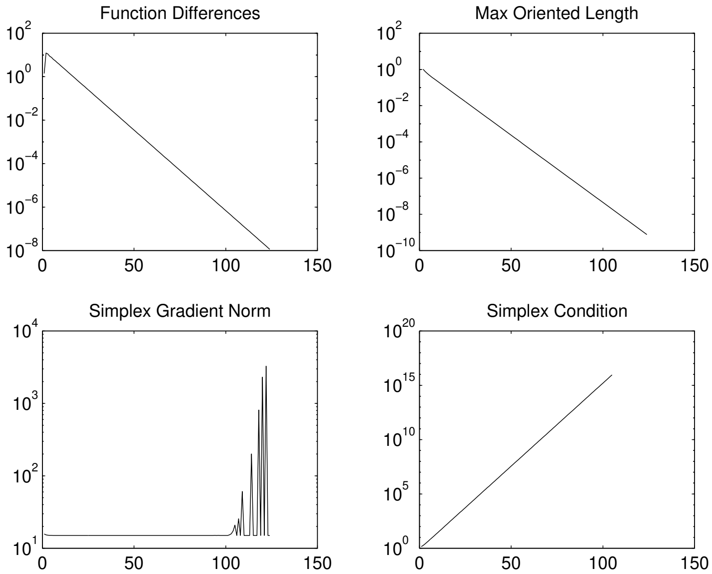


FIG. 4.2. *Modified Nelder-Mead,* $(\tau, \theta, \phi) = (1, 15, 10)$

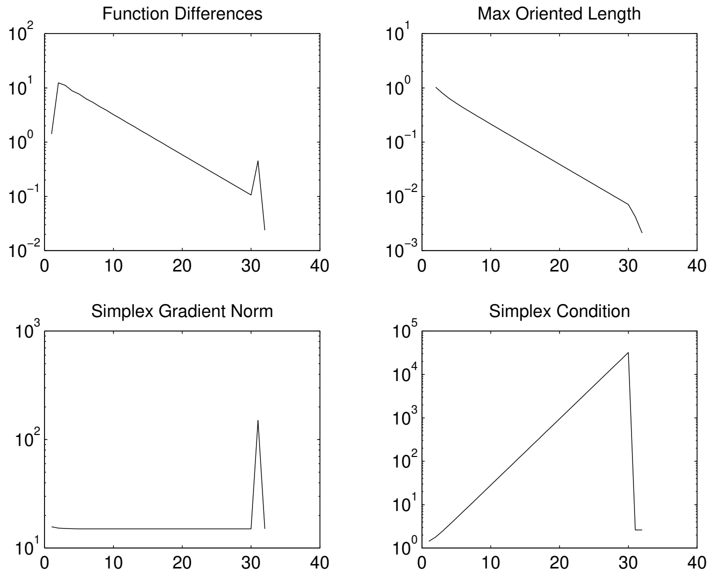


FIG. 4.3. *Unmodified Nelder-Mead,* $(\tau, \theta, \phi) = (2, 6, 60)$

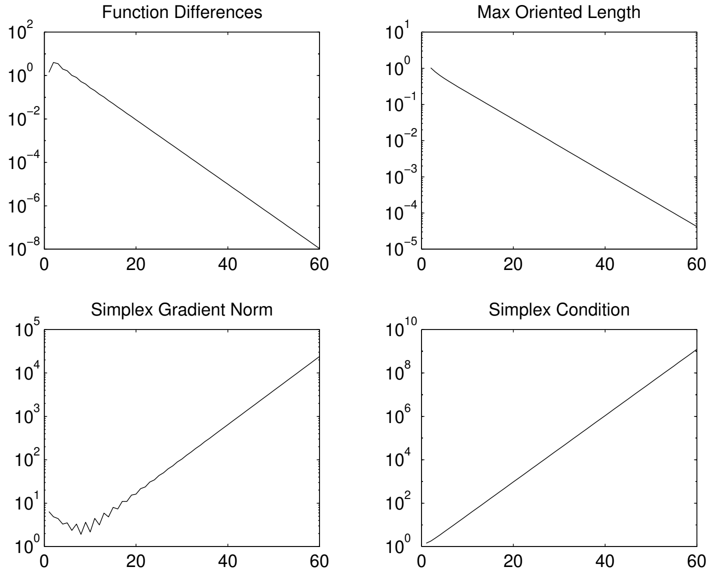


FIG. 4.4. *Modified Nelder-Mead,* $(\tau, \theta, \phi) = (2, 6, 60)$

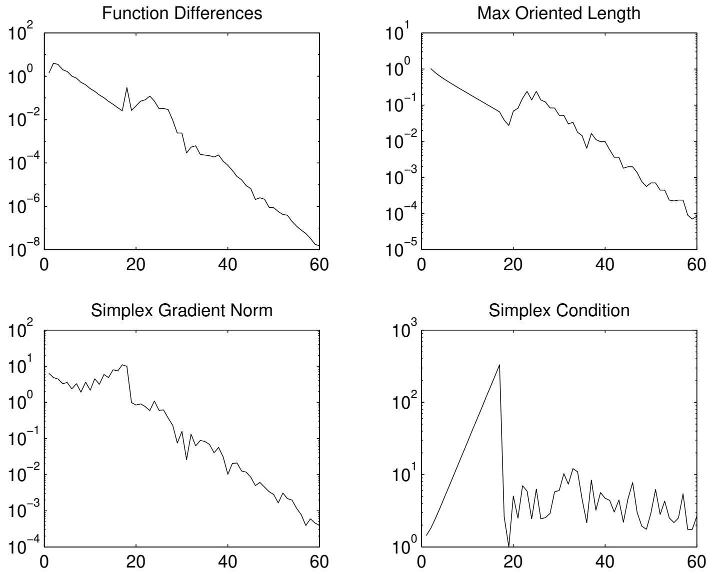


FIG. 4.5. *Unmodified Nelder-Mead,* $(\tau, \theta, \phi) = (3, 6, 400)$

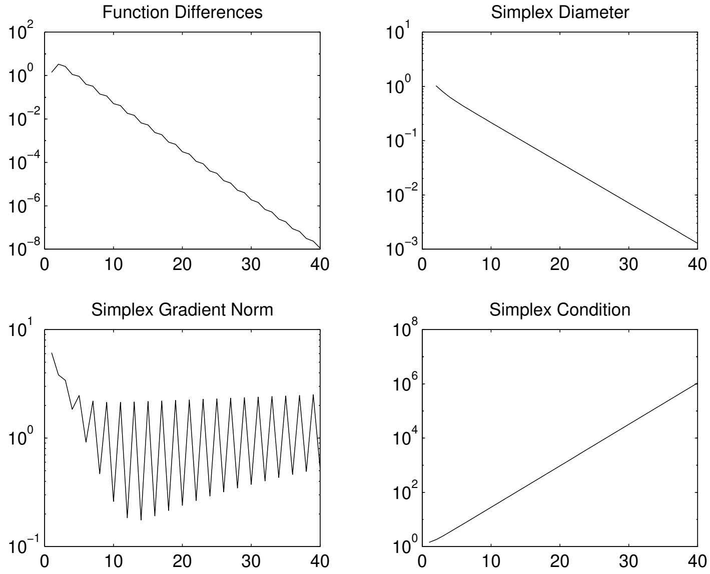
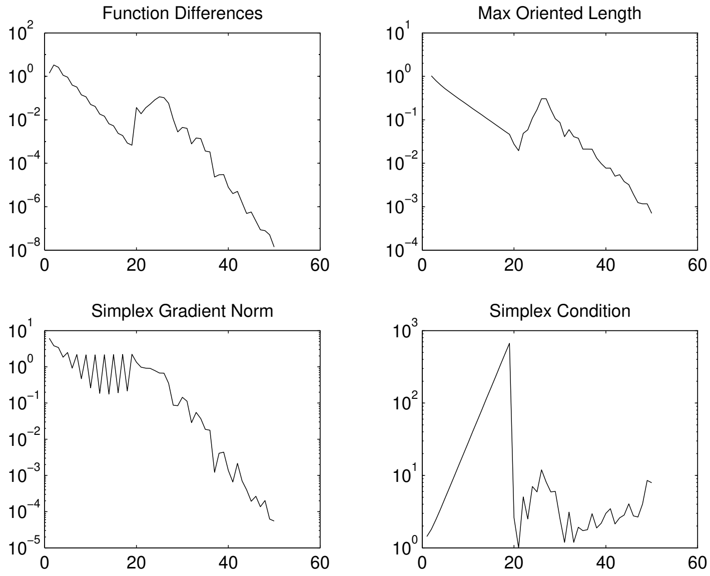


FIG. 4.6. *Modified Nelder-Mead,* $(\tau, \theta, \phi) = (3, 6, 400)$



Acknowledgments. The author is grateful to Carl Meyer and Margaret Wright for their assistance with several aspects of this work.

REFERENCES

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

[2] K. R. BAILEY AND B. G. FITZPATRICK, *Estimation of groundwater flow parameters using least squares*, Tech. Rep. CRSC-TR96-13, North Carolina State University, Center for Research in Scientific Computation, April 1996.

[3] K. R. BAILEY, B. G. FITZPATRICK, AND M. A. JEFFRIES, *Least squares estimation of hydraulic conductivity from field data*, Tech. Rep. CRSC-TR95-8, North Carolina State University, Center for Research in Scientific Computation, February 1995.

[4] A. G. BUCKLEY AND H. MA, *A derivative-free algorihm for parallel and sequential optimization*, tech. rep., Computer Science Department, University of Victoria, October 1994.

[5] 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.

[6] 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.

[7] J. E. DENNIS AND V. TORCZON, *Direct search methods on parallel machines*, SIAM J. Optim., 1 (1991), pp. 448 – 474.

[8] J. E. DENNIS AND D. J. WOODS, *Optimization on microcomputers: the Nelder-Mead simplex algorithm*, in New Computing Environments: Microcomputers in Large-Scale Computing, A. Wouk, ed., Philadelphia, 1987, SIAM, pp. 116–122.

[9] 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.

[10] R. HOOKE AND T. A. JEEVES, *'Direct search' solution of numerical and statistical problems*, Journal of the Association for Computing Machinery, 8 (1961), pp. 212–229.

[11] C. T. KELLEY, *Iterative Methods for Linear and Nonlinear Equations*, no. 16 in Frontiers in Applied Mathematics, SIAM, Philadelphia, 1995.

[12] 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.

[13] 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.

[14] J. A. NELDER AND R. MEAD, *A simplex method for function minimization*, Comput. J., 7 (1965), pp. 308–313.

[15] J. M. ORTEGA AND W. C. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, 1970.

[16] 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.

[17] 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.

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

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

[20] P. TSENG, *Fortified-descent simplicial search method: a general approach*, tech. rep., Department of Mathematics, University of Washington, Seattle, Washington, 1995.

[21] 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.

[22] ———, *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.

[23] D. J. WOODS, *An Interactive Approach for Solving Multi-Objective Optimization Problems*, PhD thesis, Rice University, 1985.

[24] M. H. WRIGHT, *Direct search methods: Once scorned, now respectable*, Tech. Rep. 96-4-02, AT&T Bell Laboratories, January 1996.

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