Gilding the Lily:
a Variant of the Nelder-Mead Algorithm based on Golden-Section Search *

Larry Nazareth† and Paul Tseng‡

January 5, 2000; revised July 30, 2001

Abstract

We propose a variant of the Nelder-Mead algorithm derived from a reinterpretation of univariate golden-section direct search. In the univariate case, convergence of the variant can be analyzed analogously to golden-section search. In the multivariate case, we modify the variant by replacing strict descent with fortified descent and maintaining the interior angles of the simplex bounded away from zero. Convergence of the modified variant can be analyzed by applying results for a fortified-descent simplicial search method. Some numerical experience with the variant is reported.

Key Words. Unconstrained minimization, Nelder-Mead algorithm, golden-section direct search.

---

* The second author is supported by National Science Foundation Grant CCR-9731273.
† Department of Pure and Applied Mathematics, Washington State University, Pullman, WA 99164-2930, U.S.A. Email: *nazareth@amath.washington.edu*.
‡ Department of Mathematics, University of Washington, Seattle, WA 98195, U.S.A. Email: *tseng@math.washington.edu*.
† ‡ This work was completed when its authors were Members of the Mathematical Sciences Research Institute (MSRI), Foundations of Computational Mathematics Program (Fall, 1998), whose support is gratefully acknowledged.

1 Introduction

Consider the problem of finding a (local) minimum of a real-valued function $f(x)$ of n real variables. A well-known class of methods for solving this problem is that of direct search. Such methods iteratively update an initial guess of a solution using a few sampled function values along linearly independent directions (see [1, 3, 5, 7, 13, 17] and references therein). In the univariate case of $n = 1$, a popular direct search method is the golden-section search. This algorithm contains no heuristic (arbitrary) parameters, is easily shown to be convergent when f is strictly unimodal, and has a close relationship with Fibonacci search, which is known to be optimal in a minimax sense [7]. In the multivariate case of $n > 1$, a popular direct search method is the Nelder-Mead (NM) algorithm [11], which extends the simplex method of Spendley et al. [14] by allowing non-isometric reflection steps.

In contrast to the golden-section search, the NM algorithm is an essentially heuristic approach that contains arbitrary parameters and possesses little in the way of convergence theory. In particular, McKinnon [9] constructed a family of strictly convex, coercive functions f of $n = 2$ variables with different degrees of smoothness, on which the simplices generated by the NM algorithm, with particular choices for the initial simplex, contract to a non-stationary point. If f is strictly convex and coercive (not necessarily differentiable), Lagarias et al. [6] showed that the simplices generated by the NM algorithm converge to the unique minimum point of f in the case of $n = 1$ variable and that the diameter of the simplices converge to zero in the case of $n = 2$ variables. Nice surveys of the NM algorithm and its variants are given in the article [17] and the books [1, 4].

A simple reformulation of the golden-section search shows it to be quite closely related to the NM algorithm in one dimension. This suggests a conceptually appealing variant of the NM algorithm that is more amenable to theoretical analysis and possibly also more effective in practice. In this paper, we consider such a variant. This variant inherits the convergence properties of the golden-section search for unimodal f in the univariate case and, by employing certain safeguards discussed in [16], similar convergence properties can be shown for pseudoconvex f in the multivariate case. Some numerical experience with this variant is also reported. The numerical evidence shows that the variant is competitive but the experiments are limited in scope and no firm conclusions can be drawn.

Throughout, we denote by $\Re^n$ the space of n-dimensional real column-vectors and, for any $x \in \Re^n$, by x^T the transpose of x and by $\|x\|$ the 2-norm of x (i.e., $\|x\| = \sqrt{x^T x}$). For any set S of $n+1$ vectors $x_1, ..., x_{n+1}$ in $\Re^n$, we denote $\mathrm{diam}(S) = \max_{i,j} \|x_i - x_j\|$ (the "diameter" of S). Let $\mathrm{von}(S)$ be any nonnegative-valued continuous function of $(x_1^T, ..., x_{n+1}^T)$ such that $\mathrm{von}(S) = 0$ if and only if the simplex with vertex set S has one of its interior angles equal to zero or, equivalently, the edges emanating from each vertex of this simplex are linearly dependent. An example is

$$\mathrm{von}(S) = |\det [\, x_2 - x_1 \quad \cdots \quad x_{n+1} - x_1 \,]\,|/\mathrm{diam}(S)^n,$$

i.e., $\mathrm{von}(S)/n!$ is the volume of the n-dimensional simplex with vertices $(x_i - x_1)/\mathrm{diam}(S)$, $i = 1, ..., n+1$, and a diameter of 1.

2 Golden-Section Search

We describe below the golden-section direct search for finding a local minimum of a function f of 1 variable. Define the golden ratio

$$\rho = (\sqrt{5} + 1)/2 \approx 1.618$$

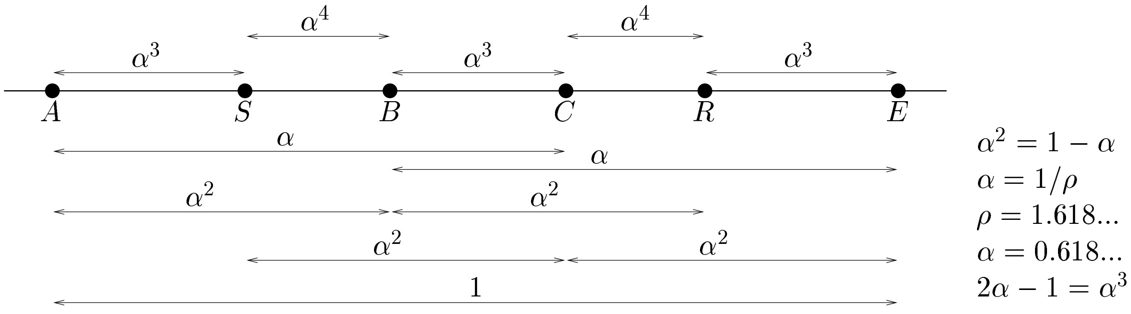

Figure 1: Golden Section Search

and its inverse $\alpha \equiv 1/\rho$ (so $\rho^2 = \rho + 1$ and $\alpha^2 = 1 - \alpha$). Choose three points A, B and E such that (i) $BE/AE = \alpha$ and (ii) $f_A \geq f_B \leq f_E$, where f_X means the value of f at X. (Henceforth points with properties (i) and (ii) are called a *golden-section triple*.) Property (ii) ensures that a local minimum of f lies in the interval between A and E. A point C is then placed in the interval such that $AC/AE = \alpha$. If $f_C \leq f_B$, the above procedure is repeated with the points B, C and E. Thus, a point R would be placed as shown in Figure 1 and so on. The use of the golden ratio ensures that $CE/BE = \alpha$. If instead $f_C < f_B$, the above procedure is repeated with the points C, B, A. Thus, a point S would be placed as shown in Figure 1 and so on. Luenberger [7] is a good reference for this procedure.

Figure 1 illustrates the points that can potentially be generated by two iterations of the golden-section direct search procedure. All points lie on the lattice comprising the integral powers of α. Note in particular that $AB = BR$ and $AB/BE = \alpha$.

When applied to a strictly unimodal function, the golden-section procedure converges to the unique minimum at a linear rate.

3 A Generic Nelder-Mead Algorithm

We describe below a generic version of the Nelder-Mead algorithm based on the descriptions of Wright [17] and Kelly [4, Section 8.1.1]. At each iteration $k = 0, 1, ...$, the method updates an n-dimensional simplex, with vertex set $S^{(k)}$, by either replacing the worst vertex x_{n+1} (highest f-value) with a new point $x[\mu]$ on the ray from x_{n+1} through the (weighted) centroid $\bar{x}$ of the remaining vertices (see Steps 3 and 4a, 4b below) or else shrinking the simplex towards its best vertex (see Step 5 below). Following [4, Eq. (8.2)], we define

$$x[\mu] = \bar{x} + \mu(\bar{x} - x_{n+1}), \qquad S[\mu] = (S^{(k)} \backslash \{x_{n+1}\}) \cup \{x[\mu]\}, \tag{1}$$

where μ is selected from $\mu_{ic}, \mu_{oc}, \mu_r, \mu_e$ satisfying

$$-1 < \mu_{ic} < 0 < \mu_{oc} < \mu_r < \mu_e. \tag{2}$$

Algorithm 1 *[Generic Nelder-Mead algorithm] Choose any set* $S^{(0)}$ *of* $n+1$ *vectors in* $\Re^n$ *satisfying* $\mathrm{von}(S^{(0)}) > 0$*. Choose* $\theta \in (0,1]$*. Choose* $\mu_{ic}, \mu_{oc}, \mu_r, \mu_e$ *satisfying* (2) *and* $0 < \mu_s < 1$*. For* $k = 0, 1, ...,$ *we generate* $S^{(k+1)}$, $x^{(k)}$ *from* $S^{(k)}$ *as follows:*

1. **Sort***: Express* $S^{(k)} = \{x_i\}_{i=1}^{n+1}$ *in ascending order of function value, i.e.,* $f_1 = f(x_1) \leq ... \leq f_{n+1} = f(x_{n+1})$*. Go to Step 2.*

2. **Reflect***: Choose any set of scalars* $\{\omega_i\}_{i=1}^{n}$ *exceeding* θ/n *and summing to 1, and let*

$$\bar{x} = \sum_{i=1}^{n} \omega_i x_i,$$

Let $x^{(k)} = \bar{x}$*. Compute* $x_r = x[\mu_r]$ *and* $f_r = f(x_r)$*. If*

$$f_r < f_n, \tag{3}$$

then go to Step 3; else go to Step 4a if $f_r < f_{n+1}$ *and go to Step 4b if* $f_r \geq f_{n+1}$*.*

3. **Expand***: If* $f_r < f_1$ *and, upon computing* $x_e = x[\mu_e]$ *and* $f_e = f(x_e)$*, if*

$$f_e \leq f_r, \tag{4}$$

then let $S^{(k+1)} = S[\mu_e]$*; else let* $S^{(k+1)} = S[\mu_r]$*. Exit.*

4. **Contract***:*

   a. *Compute* $x_{oc} = x[\mu_{oc}]$ *and* $f_{oc} = f(x_{oc})$*. If*

$$f_{oc} \leq f_r, \tag{5}$$

   then let $S^{(k+1)} = S[\mu_{oc}]$ *and exit; else go to Step 5.*

   b. *Compute* $x_{ic} = x[\mu_{ic}]$ *and* $f_{ic} = f(x[\mu_{ic}])$*. If*

$$f_{ic} < f_{n+1}, \tag{6}$$

   then let $S^{(k+1)} = S[\mu_{ic}]$ *and exit; else go to Step 5.*

5. **Shrink***: Let* $S^{(k+1)} = x_1 + \mu_s(S^{(k)} - x_1)$ *and exit.*

Note 1: A typical choice for the weights ω_i, $i = 1, ..., n$, is

$$\omega_i = 1/n,$$

corresponding to $\theta = 1$. If $f_1 \neq f_{n+1}$, another possible choice is

$$\omega_i = (1-\theta)\frac{f_{n+1} - f_i}{\sum_{j=1}^{n}(f_{n+1} - f_j)} + \frac{\theta}{n}. \quad (7)$$

Note 2: It can be seen that the NM algorithm described in [4, Algorithm 8.1.1] is logically equivalent to Algorithm 1 with μ_s set to $1/2$ and $\omega_i = 1/n$. Similarly, the generic NM algorithm described in [17] is logically equivalent to Algorithm 1 with μ_{oc} set to $-\mu_{ic}\mu_r$ and $\omega_i = 1/n$. In particular, the parameter values $\rho, \chi, \gamma, \sigma$ in [17] are related to the values (2) by

$$\mu_{ic} = -\gamma, \quad \mu_{oc} = \rho\gamma, \quad \mu_r = \rho, \quad \mu_e = \chi\rho, \quad \mu_s = \sigma.$$

The *original* NM algorithm described in [11] (also see [1, Section 1.8.2]) corresponds to the "intersection" of the algorithms in [4] and [17], namely, it corresponds to Algorithm 1 with $\mu_s = 1/2$, $\mu_{oc} = -\mu_{ic}\mu_r$ and $\omega_i = 1/n$ for $i = 1, ..., n$. Algorithm 1 may be viewed as a "union" of the algorithms in [4] and [17], combining the features of both. A typical choice for the values (2) and μ_s is

$$\mu_{ic} = -\frac{1}{2}, \quad \mu_{oc} = \frac{1}{2}, \quad \mu_r = 1, \quad \mu_e = 2, \quad \mu_s = \frac{1}{2}. \quad (8)$$

According to [11], this choice yielded the best performance compared to other choices.

Figure 2 illustrates the points that can potentially be generated by a single iteration of the NM algorithm for the case $n = 2$. This figure is similar to one given by Kelly [4, Figure 8.1]; also see [1, page 164] and [17].

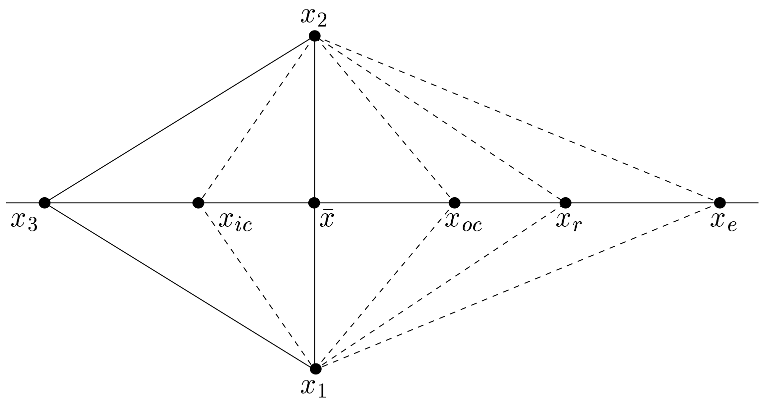

Figure 2: Generic NM Algorithm

Note 3: Algorithm 1 admits further variants such as (i) replacing (3) by $f_r < \bar{f}$, where $\bar{f} = \sum_{i=1}^{n} \omega_i f_i$, or (ii) replacing (4) by $f_e < f_1$ or (iii) replacing (5) by $f_{oc} < f_{n+1}$. The subsequent convergence theory hold for these variants, but their practical performance may vary. In particular, using isometric reflection x_r (corresponding to $\mu_r = 1$) seems crucial for good performance, as it better maintains the simplex from becoming deformed after a reflection.

4 Gilding the Lily

Let us superimpose Figure 1 on Figure 2 and make the identification:

$$x_3, x_{ic}, \bar{x}, x_{oc}, x_r, x_e \quad \text{with} \quad A, S, B, C, R, E,$$

respectively. This suggests the following choice of parameters (2) for Algorithm 1:

$$\mu_{ic} = -\alpha^2, \quad \mu_{oc} = \alpha, \quad \mu_r = 1, \quad \mu_e = \rho = 1/\alpha, \quad \mu_s = \alpha^2. \tag{9}$$

This special case of the generic NM algorithm, which we henceforth refer to as the *NM-GS algorithm*, may be viewed as a golden-section variant of the NM algorithm in the sense that, when applied to a unimodal function f of 1 variable, it generates points lying on the same lattice as golden-section search, and the points "regularly" form golden-section triples converging to some minimum point of f.

Specifically, let A, B and E be a golden-section triple as described in Section 2 (so $f_A \geq f_B \leq f_E$). Apply the NM-GS algorithm with A and B as the vertices of the initial simplex. At the first iteration, A is isometrically reflected about the point B to yield R. If $f_B > f_R$, then a reflection step would be taken and the new vertices B and R, together with the point F that is at distance $\rho \cdot BR$ to the right of R, would form a golden-section triple. The point F, not shown in Figure 1, has function value no less than that of R because it is to the right of E, $f_R \leq f_E$, and f is unimodal. If $f_R \leq f_B$, then a contraction step or a shrink step would be taken and either C or S would replace A as a vertex. If B and C are the new vertices, then it can be seen that they together with either A or E, depending on whether $f_B \leq f_C$ or $f_B > f_C$, form a golden-section triple. If B and S are the new vertices, the analysis is more involved due to a number of cases that need to be considered. A detailed analysis is given in the proof of Theorem 1. In general, the unimodality of f ensures that all generated points lie in the original interval between A and E. Unlike golden-section search, however, the generated intervals/simplices may occasionally increase in length due to expansion steps taken.

In general, the resemblance of golden-section search to the NM algorithm in one dimension provided the motivation for the values (9). By exploiting this connection, we will show in Section 6 that the NM-GS algorithm has the following features:

- It achieves convergence on unimodal univariate functions (see Theorem 1). This is in marked contrast to the approach taken by Lagarias et al. [6] where considerable effort is expended to show convergence of the *original* NM algorithm (with $\mu_r = 1, \mu_e = 2$) on strictly convex univariate functions. With the NM-GS variant, one obtains convergence in the univariate case (using a relatively simple proof) on the broader class of unimodal functions.

- It achieves convergence on pseudoconvex, continuously differentiable multivariate functions by employing safeguards as in Tseng [16] (see the next section and Theorem 2).

5 Safeguards for Convergence and Termination Criteria

McKinnon's example of nonconvergence for the NM algorithm suggests that the NM-GS algorithm is unlikely to have desirable convergence properties for $n > 1$ unless safeguards are employed. We consider two safeguards discussed in [16], namely, replacing strict descent by a stronger criterion of fortified descent and maintaining the interior angles of the simplex S bounded away from zero (i.e., $\mathrm{von}(S)$ bounded away from zero). This leads to the following modified algorithm.

Algorithm 2 *Choose any continuous functions* $\sigma : [0,\infty) \to [0,\infty)$ *and* $\beta : [0,\infty) \to [0,\infty)$ *satisfying* $\lim_{t\to 0} \sigma(t)/t = \lim_{t\to 0} \beta(t)/t = 0$ *and* $\inf_{t\geq a} \sigma(t) > 0$ *for all* $a > 0$*. Then execute as in Algorithm 1 but with the following two modifications, where* $\nu \in (0, \mathrm{von}(S^{(0)})]$ *is arbitrarily chosen.*
1. *Replace the criteria* (3), (4), (5), (6) *by, respectively,*

$$
\begin{aligned}
f_r &\leq f_n - \max\left\{\sigma(\mathrm{diam}(S)), \theta\left(f_{n+1} - \bar{f}\right) - \beta(\mathrm{diam}(S))\right\}, && && (10)\\
f_e &\leq f_r, && \mathrm{von}(S[\mu_e]) \geq \nu, && (11)\\
f_{oc} &\leq f_r - \sigma(\mathrm{diam}(S)), && \mathrm{von}(S[\mu_{oc}]) \geq \nu. && (12)\\
f_{ic} &\leq f_{n+1} - \sigma(\mathrm{diam}(S)), && \mathrm{von}(S[\mu_{ic}]) \geq \nu, && (13)
\end{aligned}
$$

where $\bar{f} = \sum_{i=1}^{n} \omega_i f_i$ *and* $S = S^{(k)}$.
2. *At the beginning of Step 2, if* $\mathrm{von}(S[\mu_r]) < \nu$*, then instead let* $\tilde{z} = 2x_1 - x_{n+1}$ *and check if*

$$
f(\tilde{z}) \leq f_1 - \min\left\{\sigma(\mathrm{diam}(S)), \theta\left(f_{n+1} - f_1\right) - \beta(\mathrm{diam}(S))\right\}. \tag{14}
$$

If yes, then let $S^{(k+1)} = 2x_1 - S$ *and exit; else go to Step 5.*

From a numerial standpoint, the criteria of fortified descent employed in the first modification is only slightly stronger than strict descent since we can choose α to be small everywhere (e.g., $\sigma(t) = 10^{-5}\min\{t^2, 1\}$), β to have fast growth away from zero (e.g., $\beta(t) = 10^5 t^2$) and θ to be near 0. From a theoretical standpoint, the difference between the two is significant since convergence cannot be assured when fortified descent is relaxed to strict descent. The second modification is motivated by the multidimensional search method of Torczon [2, 15], where they consider reflecting all vertices about the best vertex x_1. In contrast to the latter method, we check the f-value at only one of the reflected vertices. This modification is needed to ensure that von(S) stays above its threshold of ν (since reflecting all vertices about one vertex maintains von-value constant while reflecting one vertex about a centroid of the remaining vertices may decrease the von-value). The criterion (11) can alternatively be replaced by (cf. (10))

$$f(x_e) \le f_1 - \min\left\{\sigma(\mathrm{diam}(S)), \theta\left(f_{n+1} - \bar{f}\right) - \beta(\mathrm{diam}(S))\right\}, \quad \mathrm{von}(S_2) \ge \nu$$

(possibly with different α, β and θ than in Step 2) and our convergence result (i.e., Theorem 2) would still hold. However, in our tests, this alternative was not as effective as (11). Related discussions can be found in [16].

As described, Algorithms 1 and 2 each generates an infinite sequence of vectors in $\Re^n$. For practical implementations, a suitable termination criterion is needed. One such criterion, discussed in [16], is to terminate the method with the set of vectors $S = \{x_1, ..., x_{n+1}\}$ in $\Re^n$ whenever both diam(S) and the 2-norm of

$$\begin{bmatrix} (x_2 - x_1)^T \\ \vdots \\ (x_{n+1} - x_1)^T \end{bmatrix}^{-1} \begin{bmatrix} f_2 - f_1 \\ \vdots \\ f_{n+1} - f_1 \end{bmatrix} \tag{15}$$

are below a prespecified tolerance ϵ, where $f_i = f(x_i)$. [It can be seen that if von(S) is bounded away from zero and diam(S) tends to zero, then the vector (15) approaches $\nabla f(x_1)$, assuming f is continuously differentiable. Thus, this criterion yields $\nabla f(x_1) \approx 0$ on termination.] If we wish to avoid the matrix inversion, we can replace the vector (15) by

$$\begin{bmatrix} (f_2 - f_1)/\|x_2 - x_1\| \\ \vdots \\ (f_{n+1} - f_1)/\|x_{n+1} - x_1\| \end{bmatrix}. \tag{16}$$

As long as von(S) is bounded away from zero, this changes the 2-norm of the vector by only a constant factor. The above criterion worked well in our tests (see Sec. 7).

6 Convergence

In this section we analyze the convergence properties of NM-GS algorithm for the case of $n = 1$ and of Algorithm 2 for the general case. The analysis of the NM-GS algorithm exploits its connection to golden-section search.

Theorem 1 *Assume $n = 1$ and f is unimodal on $\Re$. Let $\{(S^{(k)}, x^{(k)})\}_{k=0,1,\ldots}$ be generated by the NM-GS algorithm, i.e., Algorithm 1 with* (9)*. Then, either $\{\mathrm{diam}(S^{(k)})\} \to \infty$ and f has no minimum point or $\{\mathrm{diam}(S^{(k)})\} \to 0$ linearly and $\{x^{(k)}\}$ converges to a minimum point of f.*

Proof. If an expansion step is taken at every iteration, then necessarily $\{\mathrm{diam}(S^{(k)})\} \to \infty$ and $\{f(x^{(k)})\}$ is strictly decreasing. Since f is unimodal, this implies f has no minimum point. Suppose instead the algorithm does not take an expansion step at some iteration k. At the beginning of this iteration, we have a simplex $S^{(k)} = \{x_1, x_2\}$ with $f_1 \le f_2$. Without loss of generality, we assume x_2 is to the left of x_1 on the real line. We consider the two cases: (i) $f_r < f_1$ or (ii) $f_r \ge f_1$. [We can identify $x_2, x_{ic}, x_1 = \bar{x}, x_{oc}, x_r, x_e$ with, respectively, A, S, B, C, R, E in Figure 1.]

In case (i), we go to Step 3. Since we do not take an expansion step, we must have $f_e > f_r$ so the new simplex is $S^{(k+1)} = \{x_1, x_r\}$. Moreover, $f_1 \ge f_r \le f_e$ with $|x_1 - x_r|/|x_r - x_e| = 1/\alpha$. The unimodality of f implies points to the right of x_r have f-values greater than or equal to f_r, so the next iteration $k+1$ must do a contraction, giving the next simplex either $S^{(k+2)} = \{x_r, x_e\}$ with $d(S^{(k+2)}) = \mathrm{diam}(S^{(k)})\alpha$ or $S^{(k+2)} = \{x_{oc}, x_r\}$ with $d(S^{(k+2)}) = \mathrm{diam}(S^{(k)})\alpha^2$. Moreover, the following iteration $k+2$ cannot take an expansion step.

In case (ii), we go to Step 4a or 4b. We have two subcases: (iia) $f_r < f_2$ and $f_{oc} \le f_r$ or (iib) $f_r \ge f_2$ or $f_{oc} > f_r$. In subcase (iia), the new simplex is $S^{(k+1)} = \{x_1, x_{oc}\}$ with $\mathrm{diam}(S^{(k+1)}) = \mathrm{diam}(S^{(k)})\alpha$. Moreover, depending on whether f_1 or f_{oc} is less, either $f_2 \ge f_1 \le f_{oc}$ with $|x_1 - x_{oc}|/|x_2 - x_1| = \alpha$ or $f_1 \ge f_{oc} \le f_r$ with $|x_1 - x_{oc}|/|x_{oc} - x_r| = 1/\alpha$. In subcase (iib), the new simplex is always $S^{(k+1)} = \{x_{ic}, x_1\}$ with $\mathrm{diam}(S^{(k+1)}) = \mathrm{diam}(S^{(k)})\alpha^2$, regardless of whether we go to Step 5. If $f_{ic} \le f_1$, then we have $f_2 \ge f_{ic} \le f_1$ with $|x_{ic} - x_1|/|x_2 - x_{ic}| = \alpha$. Otherwise, since we are in subcase (iib) and $f_2 \ge f_1$, either $f_{ic} \ge f_1 \le f_{oc}$ with $|x_{ic} - x_1|/|x_1 - x_{oc}| = \alpha$ or $f_{ic} \ge f_1 \le f_r$ with $|x_{ic} - x_1|/|x_1 - x_r| = \alpha^2$. Thus, $S^{(k+1)} = \{a, b\}$ together with some e satisfying $f(a) \ge f(b) \le f(e)$ and $|a-b|/|b-e| \in \{1/\alpha, \alpha, \alpha^2\}$. In the subcase of $|a-b|/|b-e| \in \{1/\alpha, \alpha\}$, the unimodality of f implies points outside the interval between a and e have f-value greater than or equal to that of a or e, depending on which side of the interval the point lies, so the next iteration $k+1$ cannot take an expansion step. In the case of $|a-b|/|b-e| = \alpha^2$, we must be in subcase (iib) and $a = x_{ic}, b = x_1, e = x_r$. Then, if an expansion step is taken at iteration $k+1$, the next simplex must be $S^{(k+2)} = \{x_1, x_{oc}\}$ with $f_1 > f_{oc}$. This together with $f_1 \le f_r$ implies that the following iteration $k+2$ cannot take an expansion step. Moreover, since the simplex diameter can increase by a factor of at most $1/\alpha$ at each iteration, we have $\mathrm{diam}(S^{(k+2)}) \le \mathrm{diam}(S^{(k+1)})/\alpha \le \mathrm{diam}(S^{(k)})\alpha$.

To summarize, if we do not take an expansion step at an iteration k, then either (a) we do not take an expansion step at iteration $k+1$ and $\mathrm{diam}(S^{(k+1)}) \le \mathrm{diam}(S^{(k)})\alpha$ or (b) we do not take an expansion step at iteration $k+2$ and $\mathrm{diam}(S^{(k+2)}) \le \mathrm{diam}(S^{(k)})\alpha$. In addition, $S^{(k)} = \{a^{(k)}, b^{(k)}\}$ together with some $e^{(k)}$ satisfy $f(a^{(k)}) \geq f(b^{(k)}) \leq f(e^{(k)})$ and $|a^{(k)} - b^{(k)}|/|b^{(k)} - e^{(k)}| \in \{1/\alpha, \alpha, \alpha^2\}$ and $[a^{(k+1)}, e^{(k+1)}] \subset [a^{(k)}, e^{(k)}]$, etc. ∎

The above proof shows that, in the univariate and unimodal case, if the NM-GS algorithm does not take an expansion step at some iteration, then at subsequent iterations each generated simplex $S = \{a, b\}$ forms a "generalized" golden-section triple with some $e \in \Re$ in the sense that b is between a and e and satisfies $f(a) \geq f(b) \leq f(e)$ and $|a - b|/|b - e| \in \{1/\alpha, \alpha, \alpha^2\}$.[1] So, for example, in case (i), either $S^{(k+2)} = \{x_r, x_e\}$ forms such a triple with x_1 or $S^{(k+2)} = \{x_{oc}, x_r\}$ forms such a triple with x_e. Similarly, in subcase (iia), $S^{(k+1)} = \{x_1, x_{oc}\}$ forms such a triple with either x_2 or x_e, depending on whether $f_1 \leq f_{oc}$.

The analysis of Algorithm 2 exploits the fact that this algorithm may be viewed as a special case of the method in [16] so that the convergence result therein may be applied. Recall that f is said to be *quasiconvex* on $\Re^n$ [8] if

$$f(x + \gamma(y - x)) \leq \max\{f(x), f(y)\} \quad \forall \gamma \in [0, 1],\ \forall x, y \in \Re^n.$$

Notice that quasiconvexity generalizes unimodality to the multivariable case and it is weaker than convexity.

Theorem 2 *Assume that f is continuously differentiable and quasiconvex on $\Re^n$ and is uniformly continuous on $\{x \in \Re^n : f(x) \leq \min_{x \in S^{(0)}} f(x)\}$. Also assume $\inf_{x \in \Re^n} f(x) > -\infty$. Let $\{(S^{(k)}, x^{(k)})\}_{k=0,1,\ldots}$ be generated by Algorithm 2. Then $\{\mathrm{diam}(S^{(k)})\} \to 0$ and every cluster point of $\{x^{(k)}\}$ is a stationary point of f.*

Proof. Consider a special case of the fortified-descent simplicial search method in [16] whereby at each iteration we either choose (i) $m = n$, $S_0 = S_{\text{good}}$ in Step 1, $\Sigma_{\text{bad}} = S_{\text{bad}}$, $\Sigma_{\text{r}} = S_{\text{r}}$ in Step 2 or we choose (ii) $m = 1$ in Step 1, $\Sigma_{\text{bad}} \subset \arg\max_{x \in S} f(x)$ in Step 2. [The preceding notations are from [16].] As is discussed in Remark 1 of [16, Sec. 2], since f is quasiconvex, it is always possible to choose $m = n$ in Step 1. Then it is straightforward, though a bit tedious, to check that Algorithm 2 is a special case of this method (with an iteration of Algorithm 2 corresponding to a visit to Step 1 in this method). Then, applying the convergence result for this method, namely [16, Lemma 3.1(b), Cor. 3.3(b)], and our theorem follows. ∎

As a consequence of Theorem 2, we have that if, in addition to the assumptions therein, it is assumed that f has a unique stationary point on the level set $\{x \in \Re^n : f(x) \leq \min_{x \in S^{(0)}} f(x)\}$, then $\{x^{(k)}\}$ generated by Algorithm 2 converges to this stationary point

[1] The need to generalize the golden-section triple introduced in Section 4 arises because only one vertex of the simplex is changed at each iteration of the NM-GS algorithm.

(which in fact would be the global minimum point of f). We note that the assumptions on f made in Theorem 2 allow for the possibility of nonunique minimum points or even unbounded set of minimum points. Also, by employing additional safeguards in Algorithm 2 as described in [16], it is possible to remove the assumption of f being quasiconvex from Theorem 2.

7 Some Numerical Experience

To gain insight into the numerical behavior of the preceding algorithms and the effects of the algorithm parameters, we implemented and tested Matlab versions of Algorithms 1 and 2, with (9). We will refer to these two methods as NM-GS algorithm 1 and 2. We describe our experience below.

We experimented with both choices of $\omega_i = 1/n$ and ω_i given by (7) and we settled on the former which yields much fewer function values than the latter. This seems to be because the latter leads to greater deformation of the simplex (i.e., small von-value), so the search directions are not as well aligned with directions of sufficient descent. As was noted earlier, maintaining the simplex from being too deformed seems to be a key to good performance. The parameter choices for NM-GS algorithm 2 are the same as those used in [16]: $\theta = .01$, $\nu = 10^{-5}$, $\sigma(t) = 10^{-5}\min\{.5t^2, t\}$, $\beta(t) = 10^6 t^2$. Also, whenever $S[\mu_r]$ is too deformed (i.e., $\mathrm{von}(S[\mu_r]) < \nu$), we replace $\omega_i = 1/n$ with

$$\omega_i = (1-\theta)/|I| \ \forall i \in I, \qquad \omega_i = \theta/(n-|I|) \ \forall i \in \{1,...,n\}\backslash I,$$

where $I = \{i \in \{1,...,n\} : \min_{j\in\{1,...,n\}}(x_{n+1}-x_i)^T(x_j - x_i) < 0\}$. This backup choice of ω_i yields an $S[\mu_r]$ that is less deformed, as it puts more weight on vertices making an obtuse angle with the worst vertex and another vertex.

Table 1 tabulates the performance of NM-GS algorithms 1 and 2 on eight test functions with $n = 4,3,2,3,3,3,4,4$, respectively. The second and fourth test functions are from [12] and [18], and the other test functions are least square problems described in [10] (i.e., numbers 1, 11-14, 16). These functions were chosen because they are widely used and are easy to code. For each test function, the initial simplex was constructed by taking the starting vector used in the above references and adding to this vector the ith unit coordinate vectors in $\Re^n$ for $i = 1,...,n$. Termination occurs when both $\mathrm{diam}(S)$ and the ∞-norm of the vector (16), where $S = \{x_1,...,x_{n+1}\}$ denotes the current simplex, are below 10^{-3}. This ensures $\nabla f(x_1) \approx 0$ upon termination and, as can be seen from Table 1, the corresponding f-value is within good accuracy of the global minimum. For comparison, we also implemented in Matlab the original NM algorithm (as interpreted from the paper [11] with the recommended parameter setting of (8)) using the same initial simplex and termination criterion, and report its performance on the test functions. As can be seen from Table 1, NM-GS algorithms 1 and 2 have about equal performance except on the first

	NM algorithm		NM-GS algorithm 1		NM-GS algorithm 2	
Function	**#f-eval.**[1]	**f-value**[2]	**#f-eval.**[1]	**f-value**[2]	**#f-eval.**[1]	**f-value**[2]
Powell1	329	$1.5 \cdot 10^{-12}$	183	$4.1 \cdot 10^{-9}$	169	$9.5 \cdot 10^{-9}$
Powell2	95	-3.0000	90	-3.0000	90	-3.0000
Rosenbrock	159	$1.9 \cdot 10^{-7}$	182	$1.6 \cdot 10^{-8}$	182	$1.6 \cdot 10^{-8}$
Zangwill	86	$3.1 \cdot 10^{-7}$	95	$2.1 \cdot 10^{-7}$	95	$2.1 \cdot 10^{-7}$
Gulf	382	$7.5 \cdot 10^{-12}$	440	$1.1 \cdot 10^{-12}$	675	$9.5 \cdot 10^{-11}$
Box	193	$2.8 \cdot 10^{-10}$	255	$1.6 \cdot 10^{-10}$	254	$1.6 \cdot 10^{-10}$
Wood	807	$2.6 \cdot 10^{-7}$	601	$6.7 \cdot 10^{-8}$	605	$4.5 \cdot 10^{-8}$
Brown-Dennis	298	85822	322	85822	322	85822

Table 1: Performance of the NM algorithm and NM-GS algorithms 1 and 2 on eight test functions.
[1] This is the number of times that f was evaluated upon termination.
[2] This is the value of f at the best vertex upon termination.

and the fifth functions, due to NM-GS algorithm 2 preventing the simplex S from being too deformed (i.e., $\mathrm{von}(S) \geq \nu$). While we can always set ν very small so that NM-GS algorithm 2 has similar performance as NM-GS algorithm 1, the above result suggests that judiciously setting ν to a higher value, possibly chosen dynamically, can be beneficial. Also, we note that the second modification (14) rarely was invoked in NM-GS algorithm 2, for otherwise the convergence can be slow. NM-GS algorithms 1 and 2 perform better than the NM algorithm on two of the test functions (Powell1 and Wood), worse on two other functions (Gulf and Box), and about equal on the remaining functions. Thus, while NM-GS algorithms 1 and 2 have nicer theoretical convergence properties than the NM algorithm, their practical performance seem to be comparable to the latter.

To test sensitivity of the performances to the starting vector, we made $2n$ additional runs of each algorithm on the same function, with the starting vector perturbed by adding either 1 or -1 to one of the n components. The number of f-evaluations is then averaged over the $2n+1$ runs for the NM algorithm, and NM-GS algorithms 1 and 2. On the Rosenbrock function ($n = 2$), the averages are, respectively, 139.6, 144.6, 145.6. On the Zangwill function ($n = 3$), the averages are, respectively, 91.7, 88.1, 88.1. These averaged performances seem to be comparable.

As with the NM algorithm and related simplicial search methods, NM-GS algorithm 2 can suffer from poor performance even for moderately large n. In particular, the algorithm also exhibited slow convergence on a quadratic example cited by Wright [17, Sec. 7] in which $n = 32$ and $f(x) = \|x\|^2$, and the initial simplex was constructed by taking $(1, 2, ..., 32)^T$ and adding to this vector the ith unit coordinate vector in $\Re^{32}$, for $i = 1, ..., 32$.

References

[1] D. P. Bertsekas, *Nonlinear Programming*, 2nd edition, Athena Scientific, Belmont, 1999.

[2] J. E. Dennis, Jr. and V. Torczon, *Direct search methods on parallel machines*, SIAM Journal on Optimization, 1 (1991), 448–474.

[3] R. Fletcher, *Practical Methods of Optimization: Volume 1*, John Wiley and Sons, New York, NY, 1980.

[4] C. T. Kelly, *Iterative Methods for Optimization*, SIAM, Philadelphia, PA, 1999.

[5] A. G. Kuznetsov, *Nonlinear optimization toolbox*, Report OUEL 1936/92, Department of Engineering Science, University of Oxford, Oxford, United Kingdom, 1992.

[6] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright, *Convergence properties of the Nelder-Mead simplex algorithm in low dimensions*, SIAM Journal on Optimization, 9 (1998), 112–147.

[7] D. G. Luenberger, *Linear and Nonlinear Programming*, 2nd edition, Addison-Wesley, Reading, MA, 1984.

[8] O. L. Mangasarian, *Nonlinear Programming*, McGraw-Hill, New York, NY, 1969.

[9] K. I. M. McKinnon, *Convergence of the Nelder-Mead simplex method to a non-stationary point*, SIAM Journal on Optimization, 9 (1998), 148–158.

[10] J. J. Moré, B. S. Garbow, and K. E. Hillstrom, *Testing unconstrained optimization software*, ACM Transactions on Mathematical Software, 7 (1981), 17–41.

[11] J. A. Nelder and R. Mead, *A simplex method for function minimization*, Computer Journal, 7 (1965), 308–313.

[12] M. J. D. Powell, *An efficient method for finding the minimum of a function of several variables without calculating derivatives*, Computer Journal, 7 (1964), 155–162.

[13] M. J. D. Powell, *Direct search algorithms for optimization calculations*, Acta Numerica, 7 (1998), 287–336.

[14] W. Spendley, G. R. Hext, and F. R. Himsworth, *Sequential application of simplex designs in optimisation and evolutionary operation*, Technometrica, 4 (1962), 441–461.

[15] V. Torczon, *Multi-Directional Search: A Direct Search Algorithm for Parallel Machines*, Ph.D. thesis, Rice University, Houston, TX, 1989; available as Technical Report 90-7, Department of Mathematical Sciences, Rice University, Houston, TX.

[16] P. Tseng, *Fortified-descent simplicial search method: a general approach*, SIAM Journal on Optimization, 10 (2000), 269-288.

[17] M. H. Wright, *Direct search methods: once scorned, now respectable*, in Numerical Analysis 1995 (Proceedings of the 1995 Dundee Biennial Conference in Numerical Analysis), D. F. Griffiths and G. A. Watson, eds., Addison-Wesley Longman, Harlow, United Kingdom, 1996, 191–208.

[18] W. I. Zangwill, *Minimizing a function without calculating derivatives*, Computer Journal, 10 (1967), 293–296.