SIAM J. OPTIM.
Vol. 9, No. 1, pp. 112–147

CONVERGENCE PROPERTIES OF THE NELDER–MEAD SIMPLEX METHOD IN LOW DIMENSIONS*

JEFFREY C. LAGARIAS†, JAMES A. REEDS‡, MARGARET H. WRIGHT§, AND PAUL E. WRIGHT¶

Abstract. The Nelder–Mead simplex algorithm, first published in 1965, is an enormously popular direct search method for multidimensional unconstrained minimization. Despite its widespread use, essentially no theoretical results have been proved explicitly for the Nelder–Mead algorithm. This paper presents convergence properties of the Nelder–Mead algorithm applied to strictly convex functions in dimensions 1 and 2. We prove convergence to a minimizer for dimension 1, and various limited convergence results for dimension 2. A counterexample of McKinnon gives a family of strictly convex functions in two dimensions and a set of initial conditions for which the Nelder–Mead algorithm converges to a nonminimizer. It is not yet known whether the Nelder–Mead method can be proved to converge to a minimizer for a more specialized class of convex functions in two dimensions.

Key words. direct search methods, Nelder–Mead simplex methods, nonderivative optimization

AMS subject classifications. 49D30, 65K05

PII. S1052623496303470

1. Introduction.

Since its publication in 1965, the Nelder–Mead "simplex" algorithm [6] has become one of the most widely used methods for nonlinear unconstrained optimization. The Nelder–Mead algorithm should not be confused with the (probably) more famous simplex algorithm of Dantzig for linear programming; both algorithms employ a sequence of simplices but are otherwise completely different and unrelated—in particular, the Nelder–Mead method is intended for unconstrained optimization.

The Nelder–Mead algorithm is especially popular in the fields of chemistry, chemical engineering, and medicine. The recent book [16], which contains a bibliography with thousands of references, is devoted entirely to the Nelder–Mead method and variations. Two measures of the ubiquity of the Nelder–Mead method are that it appears in the best-selling handbook *Numerical Recipes* [7], where it is called the "amoeba algorithm," and in MATLAB [4].

The Nelder–Mead method attempts to minimize a scalar-valued nonlinear function of n real variables using only function values, without any derivative information (explicit or implicit). The Nelder–Mead method thus falls in the general class of *direct search methods*; for a discussion of these methods, see, for example, [13, 18]. A large subclass of direct search methods, including the Nelder–Mead method, maintain at each step a nondegenerate *simplex*, a geometric figure in n dimensions of nonzero volume that is the convex hull of $n+1$ vertices.

Each iteration of a simplex-based direct search method begins with a simplex, specified by its $n+1$ vertices and the associated function values. One or more test points are computed, along with their function values, and the iteration terminates

*Received by the editors May 13, 1996; accepted for publication (in revised form) November 24, 1997; published electronically December 2, 1998.
http://www.siam.org/journals/siopt/9-1/30347.html
†AT&T Labs–Research, Florham Park, NJ 07932 (jcl@research.att.com).
‡AT&T Labs–Research, Florham Park, NJ 07932 (reeds@research.att.com).
§Bell Laboratories, Murray Hill, NJ 07974 (mhw@research.bell-labs.com).
¶AT&T Labs–Research, Florham Park, NJ 07932 (pew@research.att.com).

with bounded level sets, section 4 analyzes the Nelder–Mead method in one dimension, and section 5 presents limited convergence results for the standard Nelder–Mead algorithm in two dimensions. Finally, section 6 discusses open problems.

2. The Nelder–Mead algorithm. The Nelder–Mead algorithm [6] was proposed as a method for minimizing a real-valued function $f(\mathbf{x})$ for $\mathbf{x} \in \mathcal{R}^n$. Four scalar parameters must be specified to define a complete Nelder–Mead method: coefficients of *reflection* (ρ), *expansion* (χ), *contraction* (γ), and *shrinkage* (σ). According to the original Nelder–Mead paper, these parameters should satisfy

$$\rho > 0, \quad \chi > 1, \quad \chi > \rho, \quad 0 < \gamma < 1, \quad \text{and} \quad 0 < \sigma < 1. \tag{2.1}$$

(The relation $\chi > \rho$, while not stated explicitly in the original paper, is implicit in the algorithm description and terminology.) The nearly universal choices used in the *standard* Nelder–Mead algorithm are

$$\rho = 1, \quad \chi = 2, \quad \gamma = \tfrac{1}{2}, \quad \text{and} \quad \sigma = \tfrac{1}{2}. \tag{2.2}$$

We assume the general conditions (2.1) for the one-dimensional case but restrict ourselves to the standard case (2.2) in the two-dimensional analysis.

2.1. Statement of the algorithm. At the beginning of the kth iteration, $k \geq 0$, a nondegenerate simplex Δ_k is given, along with its $n+1$ vertices, each of which is a point in $\mathcal{R}^n$. It is always assumed that iteration k begins by ordering and labeling these vertices as $\mathbf{x}_1^{(k)}, \ldots, \mathbf{x}_{n+1}^{(k)}$, such that

$$f_1^{(k)} \leq f_2^{(k)} \leq \cdots \leq f_{n+1}^{(k)}, \tag{2.3}$$

where $f_i^{(k)}$ denotes $f(\mathbf{x}_i^{(k)})$. The kth iteration generates a set of $n+1$ vertices that define a different simplex for the next iteration, so that $\Delta_{k+1} \neq \Delta_k$. Because we seek to minimize f, we refer to $\mathbf{x}_1^{(k)}$ as the *best* point or vertex, to $\mathbf{x}_{n+1}^{(k)}$ as the *worst* point, and to $\mathbf{x}_n^{(k)}$ as the *next-worst* point. Similarly, we refer to $f_{n+1}^{(k)}$ as the worst function value, and so on.

The 1965 paper [6] contains several ambiguities about strictness of inequalities and tie-breaking that have led to differences in interpretation of the Nelder–Mead algorithm. What we shall call "the" Nelder–Mead algorithm (Algorithm NM) includes well-defined tie-breaking rules, given below, and accepts the better of the reflected and expanded points in step 3 (see the discussion in section 3.1 about property 4 of the Nelder–Mead method).

A single generic iteration is specified, omitting the superscript k to avoid clutter. The result of each iteration is either (1) a single new vertex—the *accepted point*—which replaces $\mathbf{x}_{n+1}$ in the set of vertices for the next iteration, or (2) if a shrink is performed, a set of n new points that, together with $\mathbf{x}_1$, form the simplex at the next iteration.

One iteration of Algorithm NM (the Nelder–Mead algorithm).

1. **Order.** Order the $n+1$ vertices to satisfy $f(\mathbf{x}_1) \leq f(\mathbf{x}_2) \leq \cdots \leq f(\mathbf{x}_{n+1})$, using the tie-breaking rules given below.

2. **Reflect.** Compute the *reflection point* $\mathbf{x}_r$ from

$$\mathbf{x}_r = \bar{\mathbf{x}} + \rho(\bar{\mathbf{x}} - \mathbf{x}_{n+1}) = (1+\rho)\bar{\mathbf{x}} - \rho\mathbf{x}_{n+1}, \tag{2.4}$$

where $\bar{\mathbf{x}} = \sum_{i=1}^{n} \mathbf{x}_i/n$ is the centroid of the n best points (all vertices except for $\mathbf{x}_{n+1}$). Evaluate $f_r = f(\mathbf{x}_r)$.
If $f_1 \le f_r < f_n$, accept the reflected point $\mathbf{x}_r$ and terminate the iteration.

3. **Expand.** If $f_r < f_1$, calculate the *expansion point* $\mathbf{x}_e$,

$$\mathbf{x}_e = \bar{\mathbf{x}} + \chi(\mathbf{x}_r - \bar{\mathbf{x}}) = \bar{\mathbf{x}} + \rho\chi(\bar{\mathbf{x}} - \mathbf{x}_{n+1}) = (1 + \rho\chi)\bar{\mathbf{x}} - \rho\chi\mathbf{x}_{n+1}, \tag{2.5}$$

and evaluate $f_e = f(\mathbf{x}_e)$. If $f_e < f_r$, accept $\mathbf{x}_e$ and terminate the iteration; otherwise (if $f_e \ge f_r$), accept $\mathbf{x}_r$ and terminate the iteration.

4. **Contract.** If $f_r \ge f_n$,
perform a *contraction* between $\bar{\mathbf{x}}$ and the better of $\mathbf{x}_{n+1}$ and $\mathbf{x}_r$.
a. Outside. If $f_n \le f_r < f_{n+1}$ (i.e., $\mathbf{x}_r$ is strictly better than $\mathbf{x}_{n+1}$), perform an *outside contraction*: calculate

$$\mathbf{x}_c = \bar{\mathbf{x}} + \gamma(\mathbf{x}_r - \bar{\mathbf{x}}) = \bar{\mathbf{x}} + \gamma\rho(\bar{\mathbf{x}} - \mathbf{x}_{n+1}) = (1 + \rho\gamma)\bar{\mathbf{x}} - \rho\gamma\mathbf{x}_{n+1}, \tag{2.6}$$

and evaluate $f_c = f(\mathbf{x}_c)$. If $f_c \le f_r$, accept $\mathbf{x}_c$ and terminate the iteration; otherwise, go to step 5 (perform a shrink).
b. Inside. If $f_r \ge f_{n+1}$, perform an *inside contraction*: calculate

$$\mathbf{x}_{cc} = \bar{\mathbf{x}} - \gamma(\bar{\mathbf{x}} - \mathbf{x}_{n+1}) = (1 - \gamma)\bar{\mathbf{x}} + \gamma\mathbf{x}_{n+1}, \tag{2.7}$$

and evaluate $f_{cc} = f(\mathbf{x}_{cc})$. If $f_{cc} < f_{n+1}$, accept $\mathbf{x}_{cc}$ and terminate the iteration; otherwise, go to step 5 (perform a shrink).

5. **Perform a shrink step.** Evaluate f at the n points $\mathbf{v}_i = \mathbf{x}_1 + \sigma(\mathbf{x}_i - \mathbf{x}_1)$, $i = 2, \ldots, n+1$. The (unordered) vertices of the simplex at the next iteration consist of $\mathbf{x}_1, \mathbf{v}_2, \ldots, \mathbf{v}_{n+1}$.

Figures 1 and 2 show the effects of reflection, expansion, contraction, and shrinkage for a simplex in two dimensions (a triangle), using the standard coefficients $\rho = 1$, $\chi = 2$, $\gamma = \frac{1}{2}$, and $\sigma = \frac{1}{2}$. Observe that, except in a shrink, the one new vertex always lies on the (extended) line joining $\bar{x}$ and x_{n+1}. Furthermore, it is visually evident that the simplex shape undergoes a noticeable change during an expansion or contraction with the standard coefficients.

The Nelder–Mead paper [6] did not describe how to order points in the case of equal function values. We adopt the following tie-breaking rules, which assign to the new vertex the highest possible index consistent with the relation $f(\mathbf{x}_1^{(k+1)}) \le f(\mathbf{x}_2^{(k+1)}) \le \cdots \le f(\mathbf{x}_{n+1}^{(k+1)})$.

Nonshrink ordering rule. When a nonshrink step occurs, the worst vertex $\mathbf{x}_{n+1}^{(k)}$ is discarded. The accepted point created during iteration k, denoted by $\mathbf{v}^{(k)}$, becomes a new vertex and takes position $j+1$ in the vertices of Δ_{k+1}, where

$$j = \max_{0 \le \ell \le n} \{\ \ell \ \mid\ f(\mathbf{v}^{(k)}) < f(\mathbf{x}_{\ell+1}^{(k)})\ \};$$

all other vertices retain their relative ordering from iteration k.

Shrink ordering rule. If a shrink step occurs, the only vertex carried over from Δ_k to Δ_{k+1} is $\mathbf{x}_1^{(k)}$. Only one tie-breaking rule is specified, for the case in which $\mathbf{x}_1^{(k)}$ and one or more of the new points are tied as the best point: if

$$\min\{f(\mathbf{v}_2^{(k)}), \ldots, f(\mathbf{v}_{n+1}^{(k)})\} = f(\mathbf{x}_1^{(k)}),$$

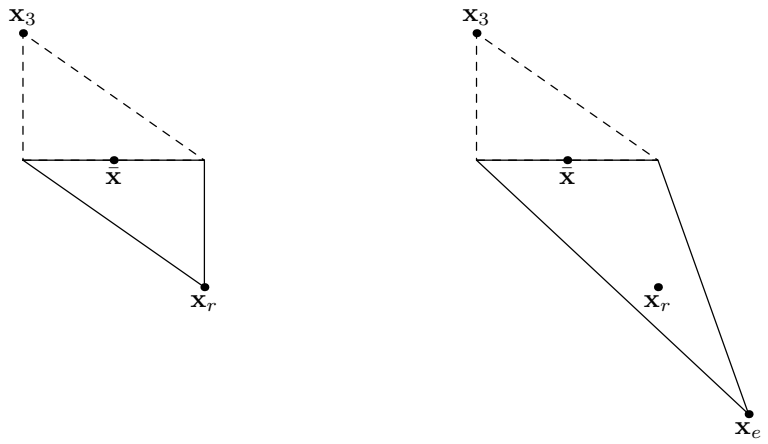

FIG. 1. *Nelder–Mead simplices after a reflection and an expansion step. The original simplex is shown with a dashed line.*

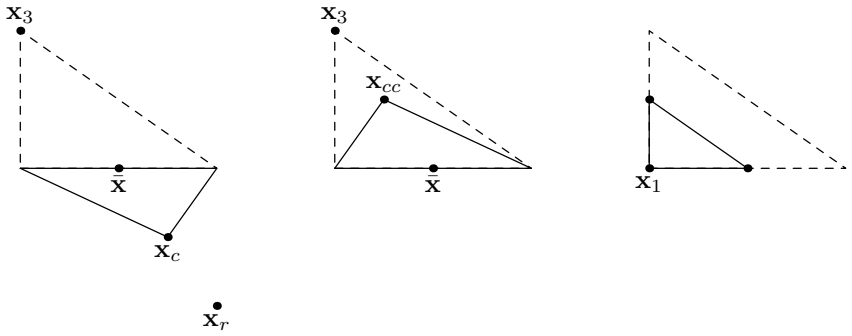

FIG. 2. *Nelder–Mead simplices after an outside contraction, an inside contraction, and a shrink. The original simplex is shown with a dashed line.*

then $\mathbf{x}_1^{(k+1)} = \mathbf{x}_1^{(k)}$. Beyond this, whatever rule is used to define the original ordering may be applied after a shrink.

We define the *change index* k^* of iteration k as the smallest index of a vertex that differs between iterations k and $k+1$:

$$k^* = \min\{\, i \mid \mathbf{x}_i^{(k)} \neq \mathbf{x}_i^{(k+1)} \,\}. \tag{2.8}$$

(Tie-breaking rules are needed to define a unique value of k^*.) When Algorithm NM terminates in step 2, $1 < k^* \leq n$; with termination in step 3, $k^* = 1$; with termination in step 4, $1 \leq k^* \leq n+1$; and with termination in step 5, $k^* = 1$ or 2. A statement that "$\mathbf{x}_j$ changes" means that j is the change index at the relevant iteration.

The rules and definitions given so far imply that, for a nonshrink iteration,

It is interesting that, apart from this further requirement, the conditions (2.1) given in the original Nelder–Mead paper suffice to ensure convergence in one dimension. The behavior of the algorithm in dimension 1 can nonetheless be very complicated; for example, there can be an infinite number of expansions even when convergence is M-step linear (Theorem 4.2).

In two dimensions, the behavior of even the standard Nelder–Mead method (with $\rho = 1$, $\chi = 2$, and $\gamma = \frac{1}{2}$) is more difficult to analyze for two reasons:

1. The space of simplex shapes is not compact, where the *shape* of a simplex is its similarity class; see the discussion at the end of section 2. It appears that the Nelder–Mead moves are dense in this space, i.e., any simplex can be transformed by some sequence of Nelder–Mead moves to be arbitrarily close to any other simplex shape; this property reflects the intent expressed by Nelder and Mead [6] that the simplex shape should "adapt itself to the local landscape." This contrasts strongly with the nature of many pattern search methods [13], in which the simplex shapes remain constant.

2. The presence of the expansion step means that $\text{vol}(\Delta)$ is not a Lyapunov function[3] for the iteration.

The two-dimensional results proved in section 5 seem very weak but conceivably represent the limits of what can be proved for arbitrary strictly convex functions. In particular, Theorem 5.2 leaves open the possibility that the ever-smaller simplices endlessly "circle" the contour line $f(x) = f^*$. Since no examples of this behavior are known, it may be possible to prove the stronger result that the simplices always converge to a *single* point $\mathbf{x}_*$.

An obvious question concerns *how* the Nelder–Mead method can fail to converge to a minimizer in the two-dimensional case. Further analysis suggests that, for suitable strictly convex functions (C^1 seems to suffice), failure can occur only if the simplices elongate indefinitely and their shape goes to "infinity" in the space of simplex shapes (as in the McKinnon counterexample).

An interesting open problem concerns whether there exists *any* function $f(\mathbf{x})$ in $\mathcal{R}^2$ for which the Nelder–Mead algorithm always converges to a minimizer. The natural candidate is $f(x, y) = x^2 + y^2$, which by affine-invariance is equivalent to all strictly convex quadratic functions in two dimensions. A complete analysis of Nelder–Mead for $x^2 + y^2$ remains an open problem.

Our general conclusion about the Nelder–Mead algorithm is that the main mystery to be solved is not whether it ultimately converges to a minimizer—for general (nonconvex) functions, it does not—but rather why it tends to work so well in practice by producing a rapid initial decrease in function values.

Acknowledgments. The authors greatly appreciate the referee's detailed and constructive comments, which helped us to improve the content and presentation of the paper. We also thank Virginia Torczon for making us aware of the connections described in section 4.4. Margaret Wright is extremely grateful to Steve Fortune for many interesting discussions and his consistently enlightening geometric insights.

REFERENCES

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

[3] See the discussion of Lyapunov functions in, for example, [11, pp. 23–27] in the context of stability of nonlinear fixed points.

[2] C. T. Kelley, *Detection and Remediation of Stagnation in the Nelder-Mead Algorithm Using a Sufficient Decrease Condition*, Technical report, Department of Mathematics, North Carolina State University, Raleigh, NC, 1997.
[3] J. C. Lagarias, B. Poonen, and M. H. Wright, *Convergence of the restricted Nelder-Mead algorithm in two dimensions*, in preparation, 1998.
[4] Math Works, Matlab, The Math Works, Natick, MA, 1994.
[5] K. I. M. McKinnon, *Convergence of the Nelder-Mead simplex method to a nonstationary point*, SIAM J. Optim., 9 (1998), 148–158.
[6] J. A. Nelder and R. Mead, *A simplex method for function minimization*, Computer Journal 7 (1965), 308–313.
[7] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vettering, *Numerical Recipes in C*, Cambridge University Press, Cambridge, UK, 1988.
[8] A. Rykov, *Simplex direct search algorithms*, Automation and Robot Control, 41 (1980), 784–793.
[9] A. Rykov, *Simplex methods of direct search*, Engineering Cybernetics, 18 (1980), 12–18.
[10] A. Rykov, *Simplex algorithms for unconstrained optimization*, Problems of Control and Information Theory, 12 (1983), 195–208.
[11] A. M. Stuart and A. R. Humphries, *Dynamical Systems and Numerical Analysis*, Cambridge University Press, New York, 1996.
[12] V. Torczon, *Multi-directional Search: A Direct Search Algorithm for Parallel Machines*, Ph.D. thesis, Rice University, Houston, TX, 1989.
[13] V. Torczon, *On the convergence of pattern search algorithms*, SIAM J. Optim., 7 (1997), 1–25.
[14] V. Torczon, *Private communication*, 1997.
[15] P. Tseng, *Fortified-Descent Simplicial Search Method: A General Approach*, Technical report, Department of Mathematics, University of Washington, Seattle, WA, 1995; SIAM J. Optim., submitted.
[16] F. H. Walters, L. R. Parker, S. L. Morgan, and S. N. Deming, *Sequential Simplex Optimization*, CRC Press, Boca Raton, FL, 1991.
[17] D. J. Woods, *An Interactive Approach for Solving Multi-objective Optimization Problems*, Ph.D. thesis, Rice University, Houston, TX, 1985.
[18] 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, UK, 1996, 191–208.