[18] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vettering (1988). *Numerical Recipes in C*. Cambridge University Press, Cambridge and New York.

[19] A. Rykov (1980). Simplex direct search algorithms. *Automation and Robot Control*, 41:784–793.

[20] A. Rykov (1980). Simplex methods of direct search. *Engineering Cybernetics*, 18:12–18.

[21] A. Rykov (1983). Simplex algorithms for unconstrained optimization. *Problems of Control and Information Theory*, 12:195–208.

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

[23] W. Swann (1972). Direct search methods. In *Numerical Methods for Unconstrained Optimization* (W. Murray, ed.), 13–28. Academic Press, London and New York.

[24] V. Torczon (1989). *Multi-directional Search: A Direct Search Algorithm for Parallel Machines*. PhD thesis, Rice University, Houston, Texas, USA.

[25] V. Torczon (1991). On the convergence of the multidirectional search algorithm. *SIAM Journal on Optimization*, 1:123–145.

[26] V. Torczon (1996). On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, to appear.

[27] P. Tseng (1995). Fortified-descent simplicial search method: a general approach. Technical report, Department of Mathematics, University of Washington, Seattle, Washington, USA.

[28] F. H. Walters, L. R. Parker, S. L. Morgan, and S. N. Deming (1991). *Sequential Simplex Optimization*. CRC Press, Boca Raton, USA.

[29] D. J. Woods (1985). *An Interactive Approach for Solving Multi-objective Optimization Problems*. PhD thesis, Rice University, Houston, Texas, USA.

[30] M. H. Wright (1996). The Nelder-Mead method: numerical experimentation and algorithmic improvements. Technical report, AT&T Bell Laboratories, Murray Hill, New Jersey, USA, to appear.

[31] W.-C. Yu (1979). The convergence property of the simplex evolutionary techniques. *Scientia Sinica, Special Issue of Mathematics*, 1:68–77.

[32] W.-C. Yu (1979) Positive basis and a class of direct search techniques. *Scientia Sinica, Special Issue of Mathematics*, 1:53–67.

search methods deserve further serious attention from the optimization community, and that (at least some) direct search methods are once again respectable.

Acknowledgement

Many thanks to Nick Higham for providing reference [1].

References

[1] M. C. Berenbaum (1990). Direct search methods in the optimisation of cancer chemotherapy regimens. *British Journal on Cancer*, 61:101–109.

[2] G. E. P. Box (1957). Evolutionary operation: a method for increasing industrial productivity. *Applied Statistics*, 6:81–101.

[3] G. F. Brissey, R. B. Spencer, and C. L. Wilkins (1979). High-speed algorithm for simplex optimization calculations. *Analytical Chemistry*, 51:2295–2297.

[4] A. G. Buckley and H. Ma (1994). A derivative-free algorithm for parallel and sequential optimization. Technical report, Computer Science Department, University of Victoria, Victoria, Canada.

[5] A. R. Conn and Ph. L. Toint (1995). An algorithm using quadratic interpolation for unconstrained derivative-free optimization. In *Nonlinear Optimization and Applications* (G. DiPillo and F. Giannessi, eds.), Plenum Press.

[6] W. C. Davidon (1991). Variable metric method for minimization. *SIAM Journal on Optimization*, 1:1–17. Originally published in 1959 as a research report from Argonne National Laboratory, Argonne, Illinois, USA.

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

[8] R. Hooke and T. A. Jeeves (1961). 'Direct search' solution of numerical and statistical problems. *Journal of the Association for Computing Machinery*, 8:212–229.

[9] J. C. Lagarias, J. A. Reeds, M. H. Wright, and P. E. Wright (1995). Convergence properties of the Nelder-Mead algorithm in low dimensions. Technical report, AT&T Bell Laboratories, Murray Hill, New Jersey, USA.

[10] The Math Works (1994). *Matlab Optimization Toolbox*, Natick, Massachusetts, USA.

[11] K. I. M. McKinnon (1995). Two strictly convex examples where the Nelder-Mead simplex method converges to a non-stationary point. Technical report, Department of Mathematics and Computer Science, University of Edinburgh, Edinburgh.

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

[13] J. Nocedal (1992). Theory of algorithms for unconstrained optimization. In *Acta Numerica 1992* (A. Iserles, ed.), 199–242. Cambridge University Press, Cambridge and New York.

[14] J. M. Ortega and W. C. Rheinboldt (1970). *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, London and New York.

[15] J. M. Parkinson and D. Hutchinson (1972). An investigation into the efficiency of variants on the simplex method. In *Numerical Methods for Non-linear Optimization* (F. A. Lootsma, ed.), 115–135. Academic Press, London and New York.

[16] M. J. D. Powell (1994). A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in Optimization and Numerical Analysis, Proceedings of the Sixth Workshop on Optimization and Numerical Analysis*, volume 275, 51–67. Kluwer, Dordrecht, Netherlands.

[17] M. J. D. Powell (1994). A direct search method that models the objective function by quadratic interpolation. Presented at the Fifth Stockholm Optimization Conference, Stockholm, Sweden.

the prevalence of this path to failure: in many of the instances when the Nelder-Mead method bogs down, the simplex is indeed drastically deformed, with a small volume and very ill-conditioned matrix of edges. In most of these situations, however, the function being minimized has highly elongated contours and a very ill-conditioned Hessian, so that the simplex becomes misshapen by responding to the local behavior of f.

In this context, it is interesting to recall that one of the original intentions of Nelder and Mead was to improve efficiency by encouraging the simplex to modify its shape in accordance with the contours of f. Hence precisely the same property of the Nelder-Mead method—its adaptation to the behavior of f—may account for both its successes and failures. Note that a near-loss of linear independence cannot happen to direct search methods whose vertices lie on a scaled lattice (such as the pattern search methods analyzed by Torczon [26]), nor to methods that force the angles of the simplex to stay uniformly bounded away from zero (such as the method of Tseng [27] when $m = n$).

Another potential cause of failure is suggested by Torczon [24], who presents numerical evidence that the Nelder-Mead method fails when its 'search direction' (the vector $\bar{x} - x_{n+1}$; see Section 3) becomes nearly orthogonal to the negative gradient. This observation may be related to the conventional wisdom that performance of the Nelder-Mead method suffers as the problem dimension increases. For example, it is reported in [30] that, when minimizing $\sum_{i=1}^{n} x_i^2$ with $n = 32$ for one initial configuration, the condition number of the simplex edge matrix never exceeds 150, and all sides of the simplex range between 0.5 and 0.9 in length; nonetheless, the best function value hovers at approximately 0.25 for several thousand iterations. The simplex in this case, although well-behaved, has in effect become misoriented with respect to the contours of f. For a more detailed analysis, see [30].

It is not yet known whether these observed behavior patterns of the Nelder-Mead method can be rigorously analyzed and fully explained. At the very least, understanding why the method fails may allow us to design computational tests that warn when trouble begins. Ideally, our analysis would also indicate what could be done to help the method recover. Tests thus far suggest that a suitable adaptive restart strategy can lead to substantial improvements in efficiency [30].

8. Closing thoughts

Since their first appearance in the 1950s, direct search methods have experienced two almost-disparate histories. They have remained astoundingly popular with practitioners for more than 30 years, and have retained this loyalty by repeatedly solving important practical problems that other methods could not handle. In contrast, during this same period direct search methods were being relegated from a mainstream position in the optimization community during the 1960s to less-than-honored status by the end of the 1980s. As we have seen, there are very good reasons for this disfavor—non-existent theory as well as erratic and unreliable performance.

The role of direct search methods is now undergoing yet another change. As discussed in Section 5, mathematical underpinnings are being provided by the evolving body of convergence theory for pattern search methods and other direct search methods. Even the Nelder-Mead method has a growing theory of its own (see Section 6), and we seem at the time of this writing to be within a reasonable distance of understanding its key mathematical properties, including the causes of failure.

From the viewpoint of practice, optimization researchers are increasingly interested in the kinds of applications described at the beginning of this paper, where every function evaluation is precious and costly. Direct search methods constitute one of the few viable alternatives for addressing such problems, especially those in which 'improvement' is sought; hence direct search methods are needed that combine theoretical soundness with both reliability and efficiency. It is still unclear which direct search methods will ultimately emerge as the most effective. Exactly as in large-scale optimization, the 'best' method is likely to be problem-dependent.

Direct search methods offer opportunities today not only for enlightening new theory, but also for successful solution of important real-world problems. This author therefore believes that direct

Some light has recently been shed on question (iv) by McKinnon [11], who gives a strictly convex continuously differentiable function in two dimensions for which the Nelder-Mead simplices converge to a non-minimizing point. In the McKinnon example, the best vertex of the initial simplex is a non-minimizing point of discontinuity in the second derivatives. By construction, every Nelder-Mead iteration is a contraction in which the best point stays fixed, so that all vertices converge to the original best vertex. Thus McKinnon provides a counterexample to the hypothesis that the Nelder-Mead simplices always converge to the minimizer of a continuously differentiable strictly convex function in two dimensions.

A few positive results are also emerging. Lagarias *et al.* [9] have established convergence of the Nelder-Mead method to a minimizer for strictly convex one-dimensional functions with bounded level sets. They have also established a variety of convergence results for strictly convex functions with bounded level sets in dimension two:

1. The function values at all vertices converge to the same value.
2. The simplices have volumes converging to zero, i.e., they collapse to a single point or straight line segment.
3. The simplices in the Nelder-Mead algorithm with no expand step have diameter converging to zero.

The mathematics used to obtain these results is quite different from the standard proof techniques in nonlinear optimization. In [9], the Nelder-Mead algorithm is interpreted as a discrete dynamical system in which the iterations are 'driven' by the function values, and where permitted sequences of moves are restricted because of the descent requirement and the properties of strictly convex functions.

The question of whether the Nelder-Mead simplices converge to a minimizer for a strictly convex *quadratic* function remains open at this time. No counterexample like McKinnon's is known for the quadratic case, but neither is there a proof of convergence to a minimizer, even in dimension 2.

Much analysis needs to be done to complete our theoretical understanding of this popular algorithm. A particularly important question involves the effects of dimensionality, since the Nelder-Mead folklore consistently states that the method may fail in higher dimensions. Clues about the theoretical difficulties are suggested by the conditions imposed in convergence theories for better-understood direct search methods, which include some combination of 'nice' simplices, a scaled lattice structure, or a sufficient descent condition. It would be useful to know whether (and, if so, how) the absence of these properties in the Nelder-Mead method allows pathologies that lead to failure.

7. The Nelder-Mead method: more about practice

The lack of theory about the Nelder-Mead method has obviously not impeded the thousands of practitioners who have happily used it since 1965. The simplicity and (wrongly) perceived robustness of the Nelder-Mead method, as well as the convenience of not having to provide derivatives, are its great strong points. Furthermore, it is undeniable that the Nelder-Mead method can sometimes be more efficient—even much more efficient—than alternative methods. For example, extensive numerical experiments by Wright [30] show that the Nelder-Mead method can converge to an acceptably accurate solution with substantially fewer function evaluations than multidirectional search or a steepest descent method based on finite-difference gradients.

Unfortunately, the results in [30] also show that the Nelder-Mead method can be horrifically inefficient and unreliable. Given this huge range in performance, it seems desirable to analyze *why* the Nelder-Mead method's behavior varies so drastically. Various theories have been offered, but no explanation yet has been entirely convincing.

A longstanding piece of Nelder-Mead folklore is that the method fails when the simplex collapses into a subspace, or becomes extremely elongated and distorted in shape. The results in [30] confirm

with respect to a subset of its m best vertices, where $m = 1$ corresponds to the multidirectional search algorithm [24], and $m = n$ corresponds to the Nelder-Mead method. (In addition, when $m = n$ the simplex angles are forced to stay uniformly bounded away from zero.) Assuming that f is continuously differentiable, bounded below, and has bounded level sets, Tseng proves that either the fortified-descent method terminates with a stationary point of f, or else at least one cluster point of the best vertices is a stationary point. A potentially significant feature of Tseng's methods is that the number of evaluations of f per iteration decreases as m increases. Hence for large m these methods may be more efficient than multidirectional search or other pattern search methods (which necessarily require a multiple of n function evaluations at every iteration).

The methods proposed by Yu, Rykov, and Tseng have in common that they allow a generalized operation of reflection, through varying subsets of the simplex vertices, and that convergence of the methods can be proved under suitable assumptions. For all of these methods, however, published numerical results are either very limited or non-existent.

A different theme in recent work on non-derivative methods is the development of methods with improved performance on smooth functions. As already observed, the logic of the direct search algorithms described thus far (except for Rykov's) depends only on the ordering of the function values, not on their numerical values. If one is prepared to assume some smoothness properties about f, the idea naturally arises of using the already-calculated function values to build a model of f, and then to compute the next trial point based on the model. Powell [16, 17], Buckley and Ma [4], and Conn and Toint [5] have considered direct search methods based on model-building. Along with theoretical analysis, these authors report good numerical performance when their methods are applied to suitably smooth test problems.

6. The Nelder-Mead method: new theory

Given the substantial progress just described on alternative direct search methods, it is logical to wonder why we should not simply abandon the Nelder-Mead method. Beyond sentiment, there are two good reasons for not doing so: the need for a clear theoretical understanding of this nearly ubiquitous and deceptively simple method, and the potential for developing an improved Nelder-Mead-like algorithm that retains the good features of the original method while avoiding its difficulties. The theoretical questions will be discussed in this section; practical implications are considered in Section 7.

From a theoretical perspective, an unsatisfactory gaping hole persists that should be filled before the Nelder-Mead method is left behind. As noted previously, none of the convergence theory published so far applies to the *original* Nelder-Mead method. Even in the most closely related work—of Woods [29] and Tseng [27]—the analysis differs in crucial ways. Woods treats an algorithm with a stronger descent condition; and in the version of Tseng's method analogous to Nelder-Mead (with $m = n$), fortified descent is imposed at each iteration along with a requirement that the angles of the simplices stay bounded away from zero. It seems important to determine rigorously, if possible, whether the Nelder-Mead method does (or does not) converge on any class of function, in any dimension.

Woods [29] has already shown that the Nelder-Mead simplices can converge to a non-minimizing point for a nonconvex function in two dimensions. However, many other questions remain:

(i) Do the function values at all vertices necessarily converge to the same value?

(ii) Does the volume of the simplices converge to zero?

(iii) Do all vertices of the simplices converge to the same point?

(iv) Do all vertices converge to a stationary point of f?

It would be of interest to know whether there are classes of functions in any dimension for which these questions can be answered 'yes'.

Figure 3 depicts the reflected, expanded, and contracted simplices in the multidirectional search algorithm, with $\chi = 2$ and $\gamma = \frac{1}{2}$. Note that, in contrast to the Nelder-Mead simplices shown in Figures 1 and 2, the new simplex always retains the same shape (angles) as the original simplex, although it may grow or decrease in size. The contraction of the multidirectional search method is analogous to the shrink in the Nelder-Mead method in that a new simplex is created by moving all non-best vertices toward the best vertex.

The number of function evaluations required by a multidirectional search iteration is an integer multiple of $2n$. If one of the initial reflection, expansion, or contraction steps produces a strict improvement over f_1, the iteration terminates after $2n$ function evaluations. If not, the sequence of reflection, expansion and/or contraction is repeated with a contracted simplex until an improved best function value is found, where each cycle costs a further $2n$ function evaluations. Consequently, a multidirectional search method can require substantially more function evaluations than the Nelder-Mead method when the latter is performing well; recall that a typical (non-shrink) Nelder-Mead iteration requires either one or two function evaluations. If, however, the multidirectional search method is implemented appropriately on a parallel computer with a sufficiently large number of processors, the n function evaluations needed for each reflection, expansion, or contraction step can be performed in parallel. See [24, 7] for further discussion about implementation issues arising in parallel computing.

A great advantage of the multidirectional search method is its strong convergence properties, which have been refined and extended by Torczon [26] to include all 'pattern search' methods—broadly speaking, direct search methods that generate only points on a scaled lattice. It is shown in [26] that, when a pattern search method is applied to a continuously differentiable function with bounded level sets,

$$\liminf_{k \to \infty} \|\nabla f(x_1^{(k)})\| = 0,$$

where $x_1^{(k)}$ denotes the best vertex at iteration k. (A common framework for several methods and complete details of other convergence results are also given in [26].)

The key ingredients in Torczon's convergence proofs are uniform linear independence of the simplex edges at every iteration, the previously mentioned scaled lattice structure of all generated points, and the step control strategy. These features together ensure that pathological simplices and steps cannot arise in pattern search methods. As noted before, a point of particular interest is that such favorable convergence properties apply even though the methods require only simple descent in the best function value. Torczon's general theory applies to coordinate search, evolutionary design [2], the Hooke-Jeeves method [8], and multidirectional search [24], but *not* to the Nelder-Mead method.

Theoretical analysis of several flavors also exists for some of the more recently proposed direct search methods. In particular, Yu [31] analyzes a version of the Spendley, Hext and Himsworth method [22] with a *sufficient descent* (rather than simple descent) condition, where the descent criterion involves the squared simplex diameter. When this method is applied to a continuously differentiable function f with bounded level sets, it is shown in [31] that at least one cluster point of the iterates is a stationary point of f.

The direct search methods of Rykov [19, 20, 21] are characterized by two properties: (i) the subset of vertices to be reflected at each iteration varies adaptively, and is chosen based on one of six criterion functions, most of which depend on numerical function values; and (ii) a sufficient descent condition similar to that of Yu [31] must be satisfied at each iteration. Certain convergence results have been proved for Rykov's methods when applied to a convex function with Lipschitz-continuous gradient and bounded level sets. The use in these methods of the numerical function values at the vertices can be viewed as a major philosophical difference from Nelder-Mead and other direct search methods, and suggests a point of similarity with the model-building methods to be mentioned at the end of this section.

In 1995, Tseng [27] suggested a class of simplex-based direct search methods involving 'fortified descent'—again, a stronger condition than simple descent. Tseng allows reflection of each simplex

Mead method, but these now involve the n edges of the simplex emanating from the best vertex, so that the entire simplex is reflected, expanded, and contracted. A multidirectional search iteration succeeds when it finds a point of strict improvement over the *best* vertex, in contrast to the much weaker condition in a Nelder-Mead iteration of finding a strict improvement compared to the worst point. As in Nelder-Mead, however, the acceptance criterion in multidirectional search is only simple (rather than sufficient) decrease.

Expansion and contraction coefficients, here denoted by χ and γ, are needed in the multidirectional search method, with standard values $\chi = 2$ and $\gamma = \frac{1}{2}$. (In effect, the reflection coefficient from the Nelder-Mead method is implicitly taken as 1.)

Iteration k of the multidirectional search algorithm.

1. **Order.**

   The best vertex is labeled as x_1, so that $f(x_1) = \arg\ \min_i\{\, f(x_i)\,\}$.

2. **Reflect.**

   Define the n reflected vertices, $x_r^{(i)} = 2x_1 - x_i$, for $i = 2, \ldots, n+1$. Evaluate $f_r^{(i)} = f(x_r^{(i)})$. If $\min_i\{\, f_r^{(i)}\,\} < f_1$, go to step 3; otherwise, go to step 4.

3. **Expand.**

   Compute the expanded vertices, $x_e^{(i)} = x_1 + \chi(x_1 - x_i)$, for $i = 2, \ldots, n+1$, and evaluate $f_e^{(i)} = f(x_e^{(i)})$. If $\min_i\{\, f_e^{(i)}\,\} < \min_i\{\, f_r^{(i)}\,\}$, then accept the expanded simplex, i.e., x_i is replaced by $x_e^{(i)}$ for $i = 2, \ldots, n+1$; otherwise, accept the reflected simplex, i.e., x_i is replaced by $x_r^{(i)}$. In either case, terminate the iteration.

4. **Contract.**

   Calculate the contracted vertices, $x_c^{(i)} = x_1 + \gamma(x_i - x_1)$, for $i = 2, \ldots, n+1$, and evaluate $f_c^{(i)} = f(x_c^{(i)})$. For $i = 2, \ldots, n+1$, replace x_i by $x_c^{(i)}$. Terminate the iteration if $\min_i\{\, f_c^{(i)}\,\} < f_1$; otherwise, return to step 2.

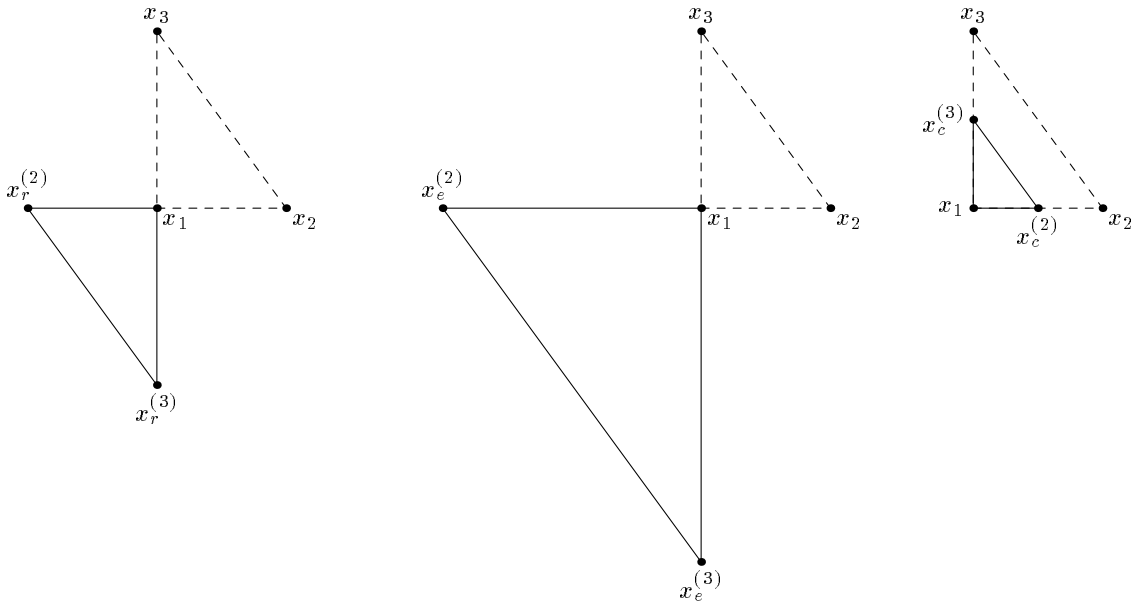

Figure 3: Multidirectional search reflection, expansion, and contraction. The original simplex is indicated with a dashed line.

The first theoretical results concerning the Nelder-Mead method appeared in the 1985 Ph. D. thesis of Woods [29]. Woods provides an interesting negative result by depicting a two-dimensional nonconvex function for which every Nelder-Mead iteration is a shrink and all vertices converge to a non-minimizing point.

Woods then considers strictly convex functions with bounded level sets, and defines a *modified* Nelder-Mead method with a stricter descent requirement for accepting the reflected point in step 2 of the algorithm (see Section 3). Under these assumptions Woods proves that: (i) every convergent subsequence of the simplices generated by the modified algorithm converges to a degenerate simplex (a single point); (ii) the values of f at all limit points are equal; and (iii) the set of limit points is connected. Although a breakthrough in the sense of producing some theory about the Nelder-Mead method, these results leave many questions unanswered. The proofs do not apply to the original Nelder-Mead method; convergence of the simplices to a single point is not guaranteed, but only convergence of a subsequence; and no properties of the limit points were verified.

In addition to concerns about the lack of theory, mainstream optimization researchers were not impressed by the Nelder-Mead method's practical performance, which can be appallingly poor. Even on well-behaved functions, the Nelder-Mead method can 'bog down' (i.e., take endless iterations while making negligible progress); see, for example, [30]. When this happens, the method often satisfies the criterion (3.2) for a small-enough simplex (or the analogous criterion involving close function values) and terminates, despite being nowhere close to a minimizer. The proliferation of papers proposing variations on Nelder-Mead—see [3], for example—strongly indicates that the original method can be unsatisfactory in practice. Torczon [24] describes substantial numerical evidence that the Nelder-Mead method is not robust.

A large amount of folklore, mostly undocumented and imprecise, has accumulated about the Nelder-Mead method. For example, several authors state that the method is good only for small problems, but there is no clear definition of 'small', nor any indication as to why the method might deteriorate with dimensionality. Frequent warnings occur that the Nelder-Mead simplex may collapse into a nearly degenerate figure, and that restarting with a fresh simplex may be helpful. However, no guidelines are provided for making these decisions in an implemented algorithm.

Given the combination of no theory and sometimes dreadful practice, why do so many people use the Nelder-Mead method? The answer is probably that, despite its bad features, the Nelder-Mead method often successfully locates a greatly improved solution with many fewer function evaluations than any of its competitors. Also, users may simply be unaware of the potential difficulties, which are not widely documented. In *Numerical Recipes* [18], for example, the Nelder-Mead method is described rather mildly as 'not very efficient in terms of the number of function evaluations that it requires'.

We shall return in Section 6 to current research on the Nelder-Mead method.

5. A renaissance for direct search methods

Papers proposing simplex-based direct search methods with improved convergence properties were published in 1979 by Yu [31, 32] in China, and in 1980 by Rykov [19, 20] in the Soviet Union. However, until the last few years these papers were not widely known in the English-language optimization research community.

For that community, a renaissance of interest in direct search methods began in 1989 with Torczon's Ph. D. thesis [24]. Her thesis and subsequent papers, some co-authored with Dennis, propose a new form of direct search method—the multidirectional search method [24, 7, 25]. A primary motivation for the multidirectional search method was a desire for efficiency in a parallel computing environment.

The multidirectional search method is simplex-based, and consequently draws on ideas in [22, 12]. Each iteration is associated with a current simplex whose best vertex (with lowest function value) is so labeled. Operations called reflection, expansion and contraction are defined as in the Nelder-

2. A Nelder-Mead iteration requires one function evaluation when the iteration terminates in step 2, two function evaluations when termination occurs in step 3 or step 4, and $n + 2$ function evaluations if a shrink step occurs. The Nelder-Mead method is thus particularly parsimonious in function evaluations per iteration compared to other direct search methods; see Section 5.
3. The next Nelder-Mead simplex is determined by the coordinates of the simplex vertices and the order information (3.1) about f at the vertices—not the numerical function values.
4. In the expand step, the method in the original Nelder-Mead paper accepts x_e if $f(x_e) < f_1$, and accepts x_r otherwise. Standard practice today (followed in the algorithm given above) accepts the better of x_r and x_e if both give a strict improvement over x_1.

To specify a complete Nelder-Mead algorithm, we need to define the initial simplex and a set of termination criteria. In the absence of information about the particular function being minimized, it is customary to specify a starting point in $\mathcal{R}^n$ that is taken as one of the initial simplex vertices. The other n vertices are then generated in one of two ways: perturbing the starting point by a specified step along the n coordinate directions, or creating a regular simplex with specified edge length and orientation. For some problems, of course, it may be possible for the user to specify $n + 1$ suitable starting vertices.

For any non-derivative method, the issue of termination is problematical as well as highly sensitive to problem scaling. Since gradient information is unavailable, it is provably impossible to verify closeness to optimality simply by sampling f at a finite number of points. Most implementations of direct search methods terminate based on two criteria intended to reflect the progress of the algorithm: either the function values at the vertices are close, or the simplex has become very small. In practice, both criteria have been interpreted in various ways. For example, following Woods [29], Torczon [24] suggests termination when the current vertices $x_1, \ldots, x_{n+1}$ satisfy

$$\max_{2 \le i \le n+1} \|x_i - x_1\| \le \epsilon \max(1, \|x_1\|), \tag{3.2}$$

where ϵ is a tolerance. Either form of termination—close function values or a small simplex—can be misleading for badly scaled functions. For discussions of termination criteria, see [15, 29, 24, 18].

4. The state of the Nelder-Mead method

Since its publication in 1965, the Nelder-Mead algorithm has been used in an extraordinarily wide variety of contexts, especially in chemistry, chemical engineering, and medicine. There are literally thousands of published papers about applications of the Nelder-Mead method, as well as numerous proposed variants intended to overcome its defects. The 1991 book [28] contains a fascinating chronological bibliography showing a steady growth in citations involving the Nelder-Mead method. Nelder-Mead codes appear in the best-selling book *Numerical Recipes* [18] and in the pervasive Matlab™ optimization toolbox [10].

Despite its wide use, until quite recently the Nelder-Mead method and its ilk have been deprecated, scorned, or ignored by almost all of the mainstream optimization community. Direct search methods are characterized in many textbooks as 'ad hoc' or 'heuristic', with little to recommend them, and are sometimes not mentioned at all. This negative attitude arises in part as a reaction to users who might choose a Nelder-Mead method simply for its ease of use, even when minimizing a smooth and inexpensive function well suited to a more reliable gradient-based method. In addition, however, there are strong theoretical and practical reasons for taking a dubious view of the Nelder-Mead method.

In retrospect, it is remarkable that for twenty years after publication of the Nelder-Mead method, no analysis of its theoretical properties was published. Theory is not mentioned in *any* of the publications about the Nelder-Mead method from the 1960s and 1970s surveyed while preparing the present paper.

5. **Perform a shrink step.**
   Define n new vertices from

$$v_i = x_1 + \sigma(x_i - x_1), \qquad i = 2, \ldots, n+1,$$

   and evaluate f at these points. The vertices of the simplex at the next iteration consist of x_1, v_2, ..., 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.

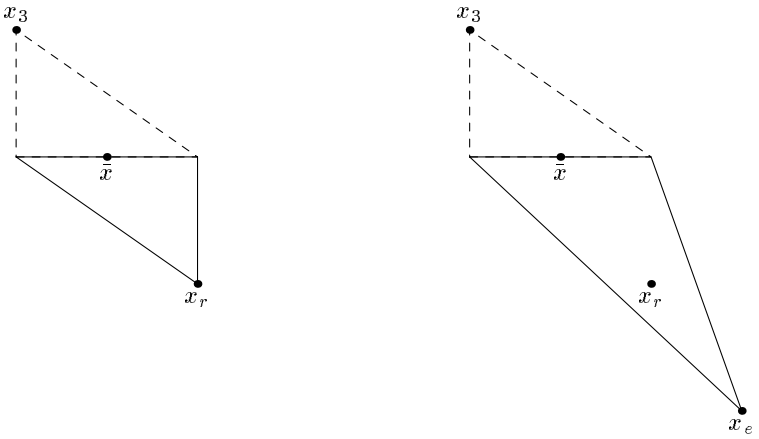

Figure 1: Nelder-Mead simplices after a reflection and an expansion step. The original simplex is shown with a dashed line.

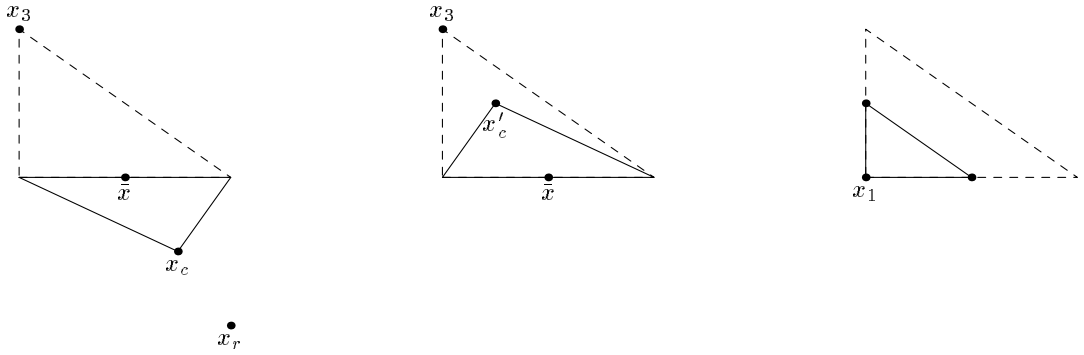

Figure 2: Nelder-Mead simplices after an outside contraction, an inside contraction, and a shrink. The original simplex is shown with a dashed line.

The Nelder-Mead method has several interesting properties:

1. A successful non-shrink iteration produces a new vertex whose function value is strictly less than f_{n+1}. This *simple decrease* requirement is much weaker than the sufficient decrease conditions usually imposed in optimization convergence theory; see, for example, [14].

and labeling the current set of vertices as $x_1^{(k)}, \ldots, x_{n+1}^{(k)}$ such that

$$f_1^{(k)} \le f_2^{(k)} \le \cdots \le f_{n+1}^{(k)}, \tag{3.1}$$

where $f_i^{(k)}$ denotes $f(x_i^{(k)})$. Because we seek to minimize f, we refer to $x_1^{(k)}$ as the *best* point, and to $x_{n+1}^{(k)}$ as the *worst* point. Consistent tie-breaking rules such as those given in [9] are required for the algorithm to be well-defined. After calculating one or more trial points and evaluating f at these points, the kth iteration generates a set of $n+1$ vertices that define a different simplex for the next iteration.

When the iteration index k is unimportant or clear from context, the superscript will be omitted. There are four possible operations: *reflection*, *expansion*, *contraction*, and *shrinkage*, each associated with a scalar parameter. The coefficients of reflection, expansion, contraction, and shrinkage are denoted respectively by ρ, χ, γ, and σ. According to the original Nelder-Mead paper [12], these coefficients should satisfy $\rho > 0$, $\chi > 1$, $0 < \gamma < 1$, and $0 < \sigma < 1$. The standard, nearly universal, choices for these values are

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

A generic Nelder-Mead iteration has two possible outcomes: (1) a single new vertex—the *accepted point*—which replaces x_{n+1} (the worst point) 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 x_1, form the simplex at the next iteration. A kind of 'search direction' is defined by x_{n+1} and $\bar{x}$, the centroid of all vertices except x_{n+1}.

Iteration k of the Nelder-Mead algorithm.

1. **Order.**
   Order the $n+1$ vertices to satisfy $f(x_1) \le f(x_2) \le \cdots \le f(x_{n+1})$, using a consistent tie-breaking rule; see, for example, [9].
2. **Reflect.**
   Compute the *reflection point* x_r from
   $$x_r = \bar{x} + \rho(\bar{x} - x_{n+1}),$$
   where $\bar{x}$ is the centroid of the n best vertices (all except x_{n+1}), i.e., $\bar{x} = \sum_{i=1}^{n} x_i/n$. Evaluate $f_r = f(x_r)$. If $f_1 \le f_r < f_n$, accept the reflected point x_r and terminate the iteration.
3. **Expand.**
   If $f_r < f_1$, calculate the *expansion point* x_e from
   $$x_e = \bar{x} + \chi(x_r - \bar{x}),$$
   and evaluate $f_e = f(x_e)$. If $f_e < f_r$, accept x_e and terminate the iteration; otherwise (if $f_e \ge f_r$), accept x_r and terminate the iteration.
4. **Contract.**
   If $f_r \ge f_n$, perform a *contraction* between $\bar{x}$ and the better of x_{n+1} and x_r.
   a. Outside. If $f_n \le f_r < f_{n+1}$ (i.e., x_r is strictly better than x_{n+1}), perform an *outside contraction*: calculate
   $$x_c = \bar{x} + \gamma(x_r - \bar{x})$$
   and evaluate $f_c = f(x_c)$. If $f_c \le f_r$, accept x_c and terminate the iteration; otherwise, go to step 5 (perform a shrink).
   b. Inside. If $f_r \ge f_{n+1}$ (i.e., x_{n+1} is better than x_r), perform an *inside contraction*: calculate
   $$x_c' = \bar{x} - \gamma(\bar{x} - x_{n+1})$$
   and evaluate $f_c' = f(x_c')$. If $f_c' < f_{n+1}$, accept x_c' and terminate the iteration; otherwise, go to step 5 (perform a shrink).

4. The values of f are 'noisy'. For example, the calculated value of f may depend on discretization, sampling on a grid, inaccurate data, or an adaptively solved subcalculation (such as numerical quadrature).

In many of the practical problems where such functions occur, a highly accurate solution is neither possible nor desired: it may be impossible because of uncertainties and errors in the underlying model or data, or may be undesirable because of the unacceptably high cost required to attain it. In fact, a frequent aim in these applications is 'improvement' rather than 'optimization'. Thus the crucial issue is finding a better answer quickly; asymptotic convergence properties are irrelevant. Examples of such real-world problems abound in medicine, chemistry, and chemical engineering; see, for example, [1] for a fascinating problem in cancer chemotherapy, and the large list of references in [28].

To minimize or improve a function with the properties listed above, the best choice is a *direct search method*. Unfortunately, this term is not precisely defined. Two necessary qualifications are:

- A direct search uses only function values;
- A direct search method does not 'in its heart' develop an approximate gradient.

The second criterion is of course ill-defined: its intent is primarily to exclude methods such as finite-difference quasi-Newton methods that construct a vector subsequently treated as if it were the gradient, but one could argue that any comparison of function values constitutes development of an approximate gradient. Despite this ambiguity, there is general agreement about the methods that do and do not qualify as direct search methods; see, for example, the comments in [24, 4, 27].

2. A brief history of direct search methods before 1965

Direct search methods were first suggested in the 1950s and continued to be proposed at a steady rate during the 1960s. These methods were typically presented and justified in terms of low-dimensional geometric intuition rather than mathematical theory; new algorithms were motivated by a desire to overcome observed or perceived inefficiencies of earlier methods. A 1972 survey article by Swann [23] summarizes the state of the art in direct search methods at that time.

From today's perspective, many of the earliest direct search techniques exhibit a large degree of commonality. In fact, Torczon's convergence theory [26], to be discussed later, applies to three of the oldest direct search methods: a coordinate search method with fixed step sizes implemented by Fermi and Metropolis to fit experimental data on the Los Alamos Maniac computer (mentioned in [6]); evolutionary operation as proposed by Box in 1957 [2]; and the 1961 Hooke and Jeeves pattern search method [8] based on automata theory.

The important class of *simplex-based* direct search methods was introduced in 1962 by Spendley, Hext and Himsworth [22]. A simplex-based method constructs an evolving pattern of $n+1$ points in $\mathcal{R}^n$ that are viewed as the vertices of a simplex. (A simplex in two dimensions is a triangle; a simplex in three dimensions is a tetrahedron.) In [22], a new simplex is formed at each iteration by reflecting away from the vertex with the largest value of f, or by contracting toward the vertex with the smallest value of f. With this approach, the angles of every simplex remain the same throughout, even though the simplex may grow or decrease in size.

The most famous simplex-based direct search method was proposed by Nelder and Mead in their 1965 paper [12]. The Nelder-Mead method is based on the idea in [22] of creating a sequence of changing simplices, but deliberately modified so that the simplex 'adapts itself to the local landscape' [12]. We next describe the Nelder-Mead method in detail.

3. The Nelder-Mead direct search method

At each iteration of the Nelder-Mead algorithm, we have a current simplex, defined by its $n+1$ vertices, each a point in $\mathcal{R}^n$, along with the corresponding values of f. Iteration k begins by ordering

Direct Search Methods: Once Scorned, Now Respectable*

Margaret H. Wright

AT&T Bell Laboratories
Murray Hill, New Jersey 07974

Abstract

The need to optimize a function whose derivatives are unknown or non-existent arises in many contexts, particularly in real-world applications. Various direct search methods, most notably the Nelder-Mead 'simplex' method, were proposed in the early 1960s for such problems, and have been enormously popular with practitioners ever since. Nonetheless, for more than twenty years these methods were typically dismissed or ignored in the mainstream optimization literature, primarily because of the lack of rigorous convergence results. Since 1989, however, direct search methods have been rejuvenated and made respectable. This paper summarizes the history of direct search methods, with special emphasis on the Nelder-Mead method, and describes recent work in this area.

This paper is based on a plenary talk given at the Biennial Dundee Conference on Numerical Analysis, Dundee, Scotland, 1995.

1. Introduction

Unconstrained optimization—the problem of minimizing a nonlinear function $f(x)$ for $x \in \mathcal{R}^n$—is important in a wide variety of applications. When the function to be minimized has suitable smoothness properties, unconstrained problems can often be solved routinely using techniques based on Newton's method. The underlying approach is to develop a model function—usually quadratic—derived from local gradient and Hessian information about f, and then to calculate a 'good' step based on the model. The fundamental theory of Newton-based methods has been understood for more than twenty years, although a few open issues remain. (See [13] for a survey of theory in unconstrained optimization.) The most active areas of research about smooth unconstrained optimization today involve the large-scale case, particularly the associated linear algebraic issues.

This paper focuses on a completely different set of unconstrained problems, for which Newton-based methods are inappropriate or inapplicable. We consider minimization of a nonlinear function f with one or more of the following properties:

1. Calculation of f is *very* expensive or time-consuming. For example, each value of f may be obtained by solving a costly numerical subproblem or performing a sequence of laboratory experiments.
2. Exact first partial derivatives of f cannot be calculated. Either the gradient does not exist—for example, when f is unpredictably discontinuous—or f is defined from a complex or convoluted computational structure that obviates application of automatic differentiation techniques.
3. Numerical approximation of the gradient of f is impractically expensive or slow.

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